

Theory and Results on Restarting Schemes for Accelerated First Order Methods

by

Viktor Pavlovic

A thesis
presented to the University of Waterloo
in fulfilment of the
thesis requirement for the degree of
Master of Mathematics
in
Combinatorics and Optimization

Waterloo, Ontario, Canada, 2024

© Viktor Pavlovic 2024

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Composite convex optimization problems are abundant in industry, and first order methods to solve them are growing in popularity as the size of variables reaches billions. Since the objective function could be possibly non-smooth, proximal gradient methods are one of the main tools for these problems. These methods benefit from acceleration, which uses the memory of past iterates to add momentum to the algorithms. Such methods have a $O(1/k^2)$ convergence rate in terms of function value where k is the iteration number.

Restarting algorithms has been seen to help speed up algorithms. O’Donoghue and Candès introduced adaptive restart strategies for accelerated first order methods which rely on easy to compute conditions, and indicate a large performance boost in terms of convergence. The restart works by resetting the momentum gained from acceleration. Their strategies in general are a heuristic, and there is no proof of convergence.

In this thesis we show that restarting with the O’Donoghue and Candès condition improves the standard convergence rate in special cases. We consider the case of one-dimensional functions where we prove that the gradient based restart strategy from O’Donoghue and Candès improves the $O(1/k^2)$ bound. We also study the restarting scheme applied to the method of alternating projections (MAP) for two closed, convex, and nonempty sets. It is shown in Chapter 6 that MAP falls into the convex composite paradigm and therefore acceleration can be applied. We study the case of MAP applied to two hyperplanes in arbitrary dimension. Furthermore we make observations as to why the restarts help, what makes a good restart condition, as well as what is needed to make progress in the general case.

Acknowledgements

I would like to thank my supervisors Steve Vavasis, and Walaa Moursi. Steve and Walaa have been eternally kind and supportive to me throughout this thesis. I have learned a great deal from both of them. They helped me become a much better mathematician and inspired me, as well as gave me the belief to pursue further education. Steve, thank you for your patience, and willingness to help me get through hurdles. Walaa, thank you for your kindness, and enthusiasm. Thanks to both of you for fostering such a great work environment. I look forward to doing my Ph.D with both of you.

I want to thank my readers, Professor Henry Wolkowicz, and Professor Levent Tuncel. Their valuable feedback and criticism helped make this the best version of the thesis that we could have. There is a lot to learn from both of you, and I hope I have many opportunities to do so in the future.

I want to thank all of my friends, the ones across Canada in Saskatchewan, British Columbia, Quebec, and here in Ontario. I am also grateful for the wonderful friends I have made here in Waterloo, especially the ones in the C&O department.

To my girlfriend Celia: You always supported and believed in me even when I did not. You are truly amazing, and I am lucky to have had you by my side throughout this degree.

Finally, I want to thank my family. My dad Predrag, and my mom Ljubica who moved here from Serbia so that I could have opportunities such as this. I want to thank my brother Sebastian for being supportive and offering me brotherly advice (even if I thought I knew better). I want to thank my family in Serbia, my grandparents, aunt, uncle, and cousin.

Dedication

This thesis is dedicated to my family, and the ones I love.

Table of Contents

Author's Declaration	ii
Abstract	iii
Acknowledgements	iv
Dedication	v
List of Figures	viii
List of Algorithms	ix
1 Introduction	1
2 Background Information	3
2.1 Framework and Problem	3
2.2 Definitions and Results	4
2.3 Gradient Methods	7
2.4 Nonsmooth Convex Analysis	9
2.5 Proximal Operator	11
2.6 Nonexpansive Operators	12
2.7 Proximal Gradient Algorithms	14

3	Restarts	18
3.1	Fixed Restarts	19
3.2	Adaptive Restarts	21
3.3	Numerical Examples	23
3.3.1	The Good	23
3.3.2	A Negative Example	25
4	Literature Review	27
4.1	Monotonicity and Restarts	27
4.2	Restarts Under Quadratic Growth	29
4.3	Other Restart Schemes	31
5	Analysis in One Dimension	33
5.1	Intuition	33
5.2	Analysis	34
5.3	Application to Nearly Separable Functions	46
6	Applications to Alternating Projections	48
6.1	Method of Alternating Projections	48
6.2	Case of Two Hyperplanes	49
7	Conclusion and Future Work	58
	References	60

List of Figures

3.1	A plot showcasing the power of the adaptive restart strategy. The algorithm using the gradient-based strategy achieves a lower objective function value in much fewer iterations than APG, or PG.	24
3.2	A <code>Julia</code> plot comparing PG, APG, and the gradient based restarts on the Hinder and Lubin example function. This showcases that the restarts do not always improve performance. In the numerical experiment we set $\alpha = \delta = 10^{-4}$	26
5.1	A plot of the iterates of AGD for a convex function. Note that y_k crosses the minimizer and thus (5.1) would be satisfied and a restart would occur.	34
5.2	Running APG with restarts in parallel on each coordinate for a function which is nearly non-separable shows that restarting may still be beneficial.	47

List of Algorithms

1	Accelerated Proximal Gradient (APG)	16
2	Strongly Convex AGD	19
3	Giselsson-Boyd Restarted APG	28
4	Alamo et. al. Gradient Restart	31
5	APG Gradient Restart in One Dimension	35
6	MAP	48

Chapter 1

Introduction

First order methods have seen a renaissance with the advent of machine learning and large scale data science applications. One of the most well-known first order algorithms is the accelerated gradient descent (AGD) method developed in the 1980s by Nesterov. In the 2000s, motivated by the linear inverse problem, Beck and Teboulle introduced the Fast Iterative Shrinkage-Thresholding Algorithm (FISTA), a proximal-gradient counterpart of AGD. The proximal-gradient algorithm is capable of handling non-smooth convex minimization problems, in particular it is suited for the class of composite models where the objective function can be broken up into a smooth and nonsmooth component.

Due to the popularity of these accelerated methods in application, heuristics have emerged to offer additional performance boosts. One such example is that of adaptive restarting, introduced by O’Donoghue and Candès [33]. Their heuristic resets the algorithm adaptively depending on whether or not some conditions are satisfied. While their methods have shown great empirical performance, there is no actual proof of convergence of the algorithm under their adaptive strategy. Several different restart strategies with convergence guarantees have been created, but they all require stronger assumptions on the objective function.

In this thesis we will study further one of the methods introduced by O’Donoghue and Candès. In particular, we will show that it offers improvements in the one dimensional case. Furthermore, we will study the behaviour of the algorithm with regards to the convex feasibility problem.

Thesis outline

The thesis is outlined as follows:

- **Chapter 2:** The thesis starts with the necessary background information. This covers convex analysis, results pertaining to first order algorithms, as well as a foray into nonexpansive mappings.
- **Chapter 3:** Here restart strategies are introduced. We differentiate between fixed, and adaptive strategies. The heuristics of O’Donoghue and Candès are introduced.
- **Chapter 4:** In the fourth chapter a literature review of restart strategies for accelerated methods is done. There is a focus on results obtained when further conditions are put on the objective function.
- **Chapter 5:**¹ Our results for the one dimensional case are presented in this chapter.
- **Chapter 6:** The convex feasibility problem for the case of two sets is cast as a composite minimization problem. We explore the special case of where the two sets are two hyperplanes in arbitrary dimension.

¹Results in this section were presented at the Optimization for Machine Learning workshop at NeurIPS 2023.

Chapter 2

Background Information

2.1 Framework and Problem

Throughout this thesis, the main problem of study will be the following *composite model*. Given $f, g : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ we wish to solve

$$\min_{x \in \mathbb{R}^n} f(x) + g(x). \quad (\text{P})$$

We will assume that f and g are convex, lower semi-continuous, and proper functions. In addition f is L -smooth, that is ∇f is Lipschitz continuous (see Definition 2.9). The setting in the above model is that of an unconstrained optimization problem. Although many real world applications are modeled by constrained problems, this can still be accommodated in the composite model.

Definition 2.1 (Indicator Function). Let $C \subseteq \mathbb{R}^n$, then the **indicator function** of C denoted ι_C is defined by

$$\iota_C(x) = \begin{cases} 0, & x \in C \\ +\infty, & x \notin C \end{cases}. \quad (2.1)$$

We can now use the indicator function to transform a constrained problem into an unconstrained one.

Example 2.1 (Constrained Optimization). Consider the following constrained optimization problem where $C \subseteq \mathbb{R}^n$ is closed, convex, and nonempty,

$$\min_{x \in C} f(x). \quad (2.2)$$

By setting $g(x) = \iota_C(x)$ then the constrained problem is equivalent to (P) as any minimizer must be in the set C . The new unconstrained problem is now

$$\min_{x \in \mathbb{R}^n} f(x) + \iota_C(x). \quad (2.3)$$

2.2 Definitions and Results

In this section we will introduce definitions, notation, and theoretical results which will be used throughout the thesis.

A vector in \mathbb{R}^n is denoted by a lowercase letter, e.g $x \in \mathbb{R}^n$. The subscript $x_{(j)}$ denotes the j th coordinate of the vector x . For $x, y \in \mathbb{R}^n$ the notation

$$\langle x, y \rangle = \sum_{i=1}^n x_{(i)} y_{(i)}$$

is the standard inner product. Then the induced norm is denoted as

$$\|x\| = \|x\|_2 = \sqrt{\langle x, x \rangle}.$$

Given $c \in \mathbb{R}^n$ and $r > 0$ we denote the closed ball centered at c with radius r by

$$B(c, r) = \{x \in \mathbb{R}^n : \|x - c\| \leq r\}. \quad (2.4)$$

We will use X^* to denote the set of solutions to (P).

Now consider $f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$, then we introduce the assumptions on the functions in the composite model.

Definition 2.2 (Proper). Let $\text{dom} f = \{x \in \mathbb{R}^n : f(x) < +\infty\}$. If $\text{dom} f \neq \emptyset$ and furthermore $\forall x \in \mathbb{R}^n$ if $f(x) > -\infty$ then we say that f is **proper**.

Definition 2.3 (Lower semi-continuous). The function f is **lower semi-continuous** (lsc.) at $x \in \mathbb{R}^n$ if for every sequence $(x_n)_{n \in \mathbb{N}}$ converging to x we have

$$f(x) \leq \liminf_{n \rightarrow \infty} f(x_n). \quad (2.5)$$

Definition 2.4 (Relative interior). Let $C \subseteq \mathbb{R}^n$. Recall that the interior of C is $\text{int } C = \{x \in C : \exists \varepsilon > 0 \text{ such that } B(x, \varepsilon) \subseteq C\}$. Further recall that the affine hull of C denoted $\text{aff}(C)$ is the smallest affine subspace containing C . Then the **relative interior** of C denoted $\text{ri } C$ is,

$$\text{ri } C = \{x \in C : \exists \varepsilon > 0 \text{ such that } \text{aff}(C) \cap B(x, \varepsilon) \subseteq C\}. \quad (2.6)$$

We recall what it means for a function to be convex.

Definition 2.5 (Convex set). A set $C \subseteq \mathbb{R}^n$ is a **convex set** if for all $x, y \in C$ and $\lambda \in [0, 1]$ the point $z = \lambda x + (1 - \lambda)y \in C$.

Definition 2.6 (Epigraph). The **epigraph** of a function $f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{\infty\}$ is the set of points that lie above the graph. More formally it is defined as

$$\text{epi } f := \{(x, \alpha) \in \mathbb{R}^n \times \mathbb{R} : f(x) \leq \alpha\}. \quad (2.7)$$

Definition 2.7 (Convex function). We say that f is **convex** if the epigraph of f is a convex set. Equivalently, f is convex if for all x, y in the domain of f , and $\lambda \in [0, 1]$ the following inequality holds:

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y). \quad (2.8)$$

If the above inequality is strict, then we say f is **strictly convex**.

Convex functions are very nice to work with when minimizing.

Definition 2.8 (Local and global minimizers). We say that x^* is a **local minimizer** for f if there exists an $\varepsilon > 0$ such that for all $x \in B(x^*, \varepsilon)$ we have $f(x^*) \leq f(x)$.

On the other hand x^* is a **global minimizer** if for all $x \in \mathbb{R}^n$ we have $f(x^*) \leq f(x)$.

The following proposition is what makes convex functions favorable when minimizing.

Proposition 2.1. *Let $f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{\infty\}$ be a convex function. Then every local minimizer of f is a global minimizer.*

The power of this is most algorithms which are used in optimization provide local solutions, but when dealing with convex problems, every such solution will be in fact global. The last assumption we had was that f was L -smooth.

Definition 2.9. A function f is **L -smooth** if it is differentiable, and for some $L \geq 0$ its gradient satisfies the following Lipschitz condition,

$$\forall x, y \in \mathbb{R}^n \quad \|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|. \quad (2.9)$$

We state an important result for functions which are L -smooth and convex.

Theorem 2.2. [14, Theorem 5.8] Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ be convex, differentiable on \mathbb{R}^n , and let $L > 0$. Then the following are equivalent:

(i) f is L -smooth.

(ii) $f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2} \|x - y\|^2 \forall x, y \in \mathbb{R}^n$.

(iii) $f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{1}{2L} \|\nabla f(x) - \nabla f(y)\|^2 \forall x, y \in \mathbb{R}^n$.

(iv) $\langle \nabla f(x) - \nabla f(y), x - y \rangle \geq \frac{1}{L} \|\nabla f(x) - \nabla f(y)\| \forall x, y \in \mathbb{R}^n$.

Note that Theorem 2.2 (ii) provides a convex quadratic upper bound for the function f at a point y given x . An additional definition we include is that of strong convexity. Functions satisfying strong convexity provide the benefit of having a unique minimizer.

Definition 2.10. A function f is μ -strongly convex (or just strongly convex) with $\mu > 0$ if for all $x, y \in \mathbb{R}^n$ and $\lambda \in (0, 1)$ we have,

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y) - \mu \frac{\lambda(1 - \lambda)}{2} \|x - y\|^2. \quad (2.10)$$

Note that strongly convex functions are strictly convex due to the additional term including the norm.

If in addition we have that f is differentiable, then we have the following list of equivalences

Proposition 2.3. [39] Let $f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ be differentiable and $\mu > 0$. The following are equivalent:

(i) f is μ -strongly convex.

(ii) $f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\mu}{2} \|x - y\|^2$ for all $x, y \in \mathbb{R}^n$.

(iii) $\langle \nabla f(x) - \nabla f(y), x - y \rangle \geq \mu \|x - y\|^2$ for all $x, y \in \mathbb{R}^n$.

Similar to how L -smoothness gives an upper bound, item (ii) in the above proposition gives a convex quadratic lower bound for the function f .

2.3 Gradient Methods

In this section we will consider the simpler problem,

$$\min_{x \in \mathbb{R}^n} f(x) \tag{2.11}$$

where f is convex, and L -smooth. We also assume that f has a minimizer, therefore our problem has a solution. The vector $\nabla f(x)$ is thought of as the direction of steepest increase. In the case of a minimization problem, given a point x we would move in the direction $-\nabla f(x)$ to decrease the function value. Given that f is convex, every local minimum is a global minimum and so it is reasonable to move in the direction given by the negative gradient. It is also equally important to consider how far to move in the given direction. The following proposition asserts that taking a step of length $1/L$ will decrease the value of the objective function.

Proposition 2.4. *Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ be L -smooth and let $x \in \mathbb{R}^n$. Then*

$$f\left(x - \frac{1}{L}\nabla f(x)\right) \leq f(x). \tag{2.12}$$

Proof. Set $y = x - \frac{1}{L}\nabla f(x)$. Then by Theorem 2.2 we have

$$f\left(x - \frac{1}{L}\nabla f(x)\right) \leq f(x) - \frac{1}{L}\|\nabla f(x)\|^2 + \frac{1}{2L}\|\nabla f(x)\|^2. \tag{2.13}$$

Rearranging,

$$0 \leq \frac{1}{2L}\|\nabla f(x)\|^2 \leq f(x) - f(y). \tag{2.14}$$

Therefore we see that taking a step in the direction of the negative gradient of size $1/L$ will decrease the objective function value provided that $\nabla f(x)$ is not equal to zero. \square

This gives rise to the classical gradient descent (GD) algorithm. Given $x_0 \in \mathbb{R}^n$, for $k \geq 0$ update via

$$x_{k+1} = x_k - \frac{1}{L}\nabla f(x_k). \tag{2.15}$$

Let $f^* = \inf_{x \in \mathbb{R}^n} f(x)$ be the minimum objective value. Note that $f^* \leq f(x_k)$ for all $k \geq 0$. A classical result on GD is that the rate of convergence of the algorithm is $O(1/k)$ [14, see Chapter 10]. More precisely, this can be expressed as, for all $k \geq 1$

$$f(x_k) - f^* \leq \frac{L\|x_0 - x^*\|^2}{2k}, \tag{2.16}$$

where x^* is a minimizer of f [14, Thm. 10.21]. While GD does have the benefit of guaranteeing progress, it does have the drawback of having a slow rate of convergence. To address this, Nesterov introduced accelerated gradient descent (AGD) in 1983 [30].

AGD differs from classical gradient descent by adding a new sequence of vectors which use the memory of previous iterates to move forward. Consider a point $x_0 \in \mathbb{R}^n$ and define $y_0 = x_0$. Then define a sequence of scalars $(t_k)_{k \in \mathbb{N}}$. The sequence must satisfy the following conditions for all $k \geq 1$ [15]:

$$t_k \geq \frac{k+2}{2} \geq 1 = t_0 \tag{2.17}$$

$$t_k^2 \geq t_{k+1}^2 - t_{k+1}. \tag{2.18}$$

Two popular choices of $(t_k)_{k \in \mathbb{N}}$ are given below,

$$t_k = \frac{k+2}{2}, \quad t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2}, \quad t_0 = 1. \tag{2.19}$$

It can be seen in the original work by Beck and Teboulle that the second sequence satisfies the above requirements [15]. A step in AGD is given by a gradient step as in GD but now it is from the sequence of y'_k s

$$x_{k+1} = y_k - \frac{1}{L} \nabla f(y_k). \tag{2.20}$$

The update on the y_k is

$$y_{k+1} = x_{k+1} + \frac{t_k - 1}{t_{k+1}} (x_{k+1} - x_k). \tag{2.21}$$

It can be seen that the $(y_k)_{k \in \mathbb{N}}$ sequence is an affine combination of the previous iterates of $(x_k)_{k \in \mathbb{N}}$. The term $\frac{t_k - 1}{t_{k+1}} \in (0, 1)$ is commonly called the momentum parameter. The momentum term represents the emphasis on the previous iterates. As k increases, the momentum parameter increases towards 1 and more emphasis is put on the previous iterations. Due to this, the $(y_k)_{k \in \mathbb{N}}$ sequence can take steps which are too big and result in an increase in function value. Even with this non-monotonicity in function value, AGD outperforms GD. The convergence of the function value is $O(1/k^2)$ and is given by

$$f(x_k) - f^* \leq \frac{L \|x_0 - x^*\|^2}{2(k+1)^2} \tag{2.22}$$

where x^* is the closest solution to the initial point x_0 .

2.4 Nonsmooth Convex Analysis

This section of the thesis presents definitions and important results from convex analysis, that are especially important when the objective function f is not necessarily differentiable at all points $x \in \mathbb{R}^n$. To appease this we need to introduce a broader object, the subdifferential operator.

Definition 2.11 (Subdifferential and subgradient). Let $f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ be convex. The **subdifferential** of f at a point $x \in \mathbb{R}^n$ is the set $\partial f(x)$ defined by,

$$\partial f(x) = \{u \in \mathbb{R}^n : \forall y \in \mathbb{R}^n \quad f(y) \geq f(x) + \langle u, y - x \rangle\}. \quad (2.23)$$

An element $u \in \partial f(x)$ is called a **subgradient** of f at x . Furthermore the inequality defining the subgradient is called the subgradient inequality. We refer to the domain of the subdifferential as $\text{dom } \partial f = \{x \in \mathbb{R}^n : \partial f(x) \neq \emptyset\} \subseteq \text{dom } f$.

Example 2.2. One of the most classical examples is the subdifferential of $f = |\cdot|$ on the real line. The absolute value function is not differentiable at 0 so $f'(0)$ does not exist. The subdifferential is computable for all x and is given by,

$$\partial|x| = \begin{cases} \{-1\} & , x < 0 \\ [-1, 1] & , x = 0. \\ \{1\} & , x > 0 \end{cases} \quad (2.24)$$

A result which states when the subdifferential is nonempty is stated below.

Theorem 2.5. [14, Theorem 3.14] Let $f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ be proper and convex. Assume that $x \in \text{ri}(\text{dom } f)$. Then $\partial f(x)$ is convex, nonempty.

The above theorem states effectively that $\text{ri}(\text{dom } f) \subseteq \text{dom } \partial f$. An immediate corollary is then

Corollary 2.6. Let $f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ be proper, convex, and continuous. Then for all $x \in \mathbb{R}^n$ we have that $\partial f(x) \neq \emptyset$.

In view of Example 2.2, we notice that when $x \neq 0$, then $f(x) = |x|$ is differentiable and the subdifferential agrees with $f'(x)$. The following theorem makes this connection explicit.

Theorem 2.7. [34, Theorem 25.1] Let $f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ be convex and let $x \in \mathbb{R}^n$. If f is differentiable at x then $\partial f(x) = \{\nabla f(x)\}$. Conversely, if $\partial f(x)$ is a singleton, then f is differentiable at x .

The subdifferential plays an important role in optimization. Recall in calculus when wanting to minimize a function f one would find the zeros of $f'(x)$. A similar idea holds in general.

Theorem 2.8 (Fermat's Rule). Let $f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ be proper and let $x \in \mathbb{R}^n$. Then

$$0 \in \partial f(x) \iff x \text{ is a global minimizer of } f. \quad (2.25)$$

Proof. The proof is almost immediate. Let $x \in \mathbb{R}^n$ then,

$$0 \in \partial f(x) \iff \forall y \in \mathbb{R}^n \quad f(y) \geq f(x) + \langle 0, y - x \rangle \quad (2.26)$$

$$\iff \forall y \in \mathbb{R}^n \quad f(y) \geq f(x) \quad (2.27)$$

$$\iff x \text{ is a global minimizer of } f. \quad (2.28)$$

□

Consider now the composite model (P).

$$\min_{x \in \mathbb{R}^n} f(x) + g(x) \quad (2.29)$$

where $g : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ is proper, lsc, and convex function and f is convex, and L -smooth. Since g is possibly nonsmooth, the gradient may not exist.

Let $X^* = \operatorname{argmin}_{x \in \mathbb{R}^n} \{f(x) + g(x)\}$ be the set of minimizers of (P). By Theorem 2.8 we must have that for $x \in X^*$,

$$0 \in \partial(f + g)(x). \quad (2.30)$$

It holds that $\partial f + \partial g \subseteq \partial(f + g)$, but there are exceptions to the converse containment. Fortunately, under constraint qualifications it is possible for this sum to hold with equality.

Theorem 2.9. [11, Cor. 16.48 (ii)] Let f and g be proper and convex on \mathbb{R}^n . Suppose that $\operatorname{int}(\operatorname{dom} f) \cap \operatorname{dom} g \neq \emptyset$. Then for all $x \in \mathbb{R}^n$ $\partial(f + g)(x) = \partial f(x) + \partial g(x)$.

Since f is L -smooth on \mathbb{R}^n by assumption when we consider (P) it holds that f is full domain. Therefore we can apply the sum rule to obtain,

$$\partial(f + g)(x) = \partial f(x) + \partial g(x) \quad (2.31a)$$

$$= \nabla f(x) + \partial g(x). \quad (2.31b)$$

Here the second equality holds because f is differentiable. The fact that this equality holds will be critical in the following section where the proximal mapping and proximal gradient based methods are introduced.

2.5 Proximal Operator

In this section the proximal operator is introduced. This operator will be fundamental in dealing with nonsmooth convex problems.

Definition 2.12 (Proximal operator). Let $f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ be convex, lsc, and proper. Then the **proximal operator** of f at $y \in \mathbb{R}^n$ is defined as,

$$\text{prox}_f(y) = \operatorname{argmin}_{x \in \mathbb{R}^n} \left(f(x) + \frac{1}{2} \|x - y\|^2 \right). \quad (2.32)$$

For brevity, we will refer to the proximal operator of f as the prox of f .

The proximal operator was introduced in 1962 by Moreau [26]. The operator has been extensively studied and has seen many applications in problems such as signal recovery [18], and algorithms like ADMM[16], and FISTA[15].

The prox operator generalizes the projection operator. Let $C \subseteq \mathbb{R}^n$ be a nonempty set, then the projection onto C is defined as,

$$P_C(y) = \operatorname{argmin}_{x \in C} \{ \|x - y\| \}. \quad (2.33)$$

Example 2.3. Let $C \subseteq \mathbb{R}^n$ be closed, convex, and nonempty. Then $f = \iota_C$ is lsc, convex, and proper. The prox of the indicator is nothing but the projection onto the set C .

$$\text{prox}_{\iota_C}(y) = \operatorname{argmin}_{x \in \mathbb{R}^n} \{ \iota_C(x) + \frac{1}{2} \|x - y\|^2 \}. \quad (2.34)$$

$$= \operatorname{argmin}_{x \in C} \{ \frac{1}{2} \|x - y\|^2 \} \quad (2.35)$$

$$= \operatorname{argmin}_{x \in C} \{ \|x - y\| \} \quad (2.36)$$

$$= P_C(y). \quad (2.37)$$

The following two theorems provide fundamental properties of the prox operator. The first theorem will guarantee existence and uniqueness under the assumptions in (P). Meanwhile the second theorem will relate the operator to optimization problems.

Theorem 2.10. [14, Theorem 6.8] Let $f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ be proper, closed, and convex and let $x \in \mathbb{R}^n$. Then $\text{prox}_f(x)$ is a singleton.

Since the assumptions on g in (P) are the assumptions of the above theorem we need not worry about whether the prox of g is defined. However, when working with algorithms we will require that g is 'proximable' which loosely means that we can compute the operator easily.

Theorem 2.11. [14, Theorem 6.39] Let $f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ be proper, lsc, and convex. Then for any $x, y \in \mathbb{R}^n$ the following are equivalent:

(i) $y = \text{prox}_f(x)$.

(ii) $x - y \in \partial f(y)$.

(iii) $\langle x - y, z - y \rangle \leq f(z) - f(y) \quad \forall z \in \mathbb{R}^n$.

Items (i) and (ii) tell us that $x = \text{prox}_f(x) \iff 0 \in \partial f(x)$. That is, x is a fixed point of the proximal operator if and only if it is a minimizer of the function f . In view of (ii) we can define the prox as

$$\text{prox}_f = (\text{Id} + \partial f)^{-1}. \tag{2.38}$$

2.6 Nonexpansive Operators

In this section, we introduce a broader class of operators of which the prox operator belongs to. These are the classes of nonexpansive, firmly nonexpansive, and averaged operators. Averaged operators (and as we will see, firmly nonexpansive) have the useful properties that iteratively applying the operator will result in converging to a fixed point, provided that one exists.

In this section let X be a finite dimensional Hilbert Space.

Definition 2.13 (Nonexpansive, firmly nonexpansive, averaged). Let $T : X \rightarrow X$. Then T is **nonexpansive** if for all $x, y \in X$

$$\|Tx - Ty\| \leq \|x - y\|. \tag{2.39}$$

T is **firmly nonexpansive** (or just f.n.e) if for all $x, y \in X$

$$\|Tx - Ty\|^2 \leq \langle x - y, Tx - Ty \rangle. \quad (2.40)$$

T is α -**averaged** if for $\alpha \in (0, 1)$ it holds that,

$$\|Tx - Ty\|^2 \leq \|x - y\|^2 - \frac{1 - \alpha}{\alpha} \|(\text{Id} - T)x - (\text{Id} - T)y\|^2. \quad (2.41)$$

Clearly firmly nonexpansive operators are nonexpansive, as well as averaged. In fact T is f.n.e if and only if T is $1/2$ averaged. An example of a firmly nonexpansive operator is the proximal operator defined in the previous section.

Averaged operators have the nice property that compositions of averaged operators are averaged. For example, if T_1, T_2 are firm, then $T = T_2T_1$ is $2/3$ averaged. The following definition will be useful for when we use averaged operators in algorithms. Below we include several equivalent formulations of firmly nonexpansive operators.

Theorem 2.12. [11, Prop. 4.4] *Let $T : X \rightarrow X$. The following are equivalent:*

(i) T is firmly nonexpansive.

(ii) $\text{Id} - T$ is firmly nonexpansive.

(iii) $\forall x, y \in X, \quad \langle x - y, Tx - Ty \rangle \geq \|x - y\|^2$.

(iv) $2T - \text{Id}$ is nonexpansive.

Let $C \subseteq X$ be closed, convex, and nonempty. Then an example of a firmly nonexpansive operator is the projection P_C (see [11, Prop. 4.16]).

Definition 2.14 (Fejér Monotone). Let $C \subseteq X$ be nonempty. We say that a sequence $(x_k)_{k \in \mathbb{N}}$ is **Fejér monotone** with respect to C if for all $c \in C$

$$\|x_{k+1} - c\| \leq \|x_k - c\|. \quad (2.42)$$

Note that a sequence being Fejér monotone w.r.t a set C does not imply that the sequence converges to a point in C .

Let $\text{Fix } T = \{x \in X : Tx = x\}$ be the set of fixed points of the operator T . The following result is remarkably useful from an algorithmic point of view.

Proposition 2.13. [13, Cor. 22.20] *Let $T : X \rightarrow X$ be α -averaged, and suppose $\text{Fix } T$ is nonempty. Let $x_0 \in X$, update via $\forall k \geq 0$*

$$x_{k+1} = Tx_k = T^k x_0.$$

Then $(x_k)_{k \in \mathbb{N}}$ is Fejér monotone with respect to $\text{Fix } T$ and it converges to a point in $\text{Fix } T$.

An immediate application of the above proposition is the proximal point algorithm. Let $g : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ be convex, lsc, and proper. Then $\text{prox}_g(x)$ is a singleton and defined for all $x \in \mathbb{R}^n$. Furthermore, since the prox operator of such function is firmly nonexpansive (see [11, Prop. 12.28]), then it is averaged and, we know a fixed point of the prox is a minimizer of g . The proximal point algorithm is then described as follows: given $x_0 \in \mathbb{R}^n$ update via

$$x_{k+1} = \text{prox}_g(x_k) \quad \forall k \geq 0. \tag{2.43}$$

By the above proposition, $(x_k)_{k \in \mathbb{N}}$ converges to a fixed point of prox_g and thus a solution to the problem $\min_{x \in \mathbb{R}^n} g(x)$.

2.7 Proximal Gradient Algorithms

In the previous sections we have seen that algorithms like AGD can be used when the objective function is differentiable, and proximal point algorithm can be used when the objective is not smooth. Proximal gradient (PG) puts the two ideas together to solve the composite problem. The algorithm splits the composite problem into two steps; first a gradient step with respect to f is taken, then a prox step with respect to g .

Another way to view the proximal gradient algorithm is to think of it as minimizing a quadratic approximation. Since f in (P) is L -smooth, by Theorem 2.2 there exists a quadratic upper bound of f . Let $y \in \mathbb{R}^n$ be fixed. Define

$$Q_L(x; y) := f(y) + \langle \nabla f(y), x - y \rangle + \frac{L}{2} \|x - y\|^2 + g(x) \tag{2.44}$$

as the quadratic approximation of $f + g$ at y . The quadratic approximation of f is strongly convex, hence Q is strongly convex and admits a unique minimizer. The quadratic approximation provides us a unique minimizer which for now we denote \bar{x}_y , we can explicitly compute it as follows,

$$\bar{x}_y = \operatorname{argmin}_{x \in \mathbb{R}^n} (Q_L(x, y)) \quad (2.45)$$

$$= \operatorname{argmin}_{x \in \mathbb{R}^n} \left(f(y) + \langle x - y, \nabla f(y) \rangle + \frac{L}{2} \|x - y\|^2 + g(x) \right) \quad (2.46)$$

$$= \operatorname{argmin}_{x \in \mathbb{R}^n} \left(\langle \nabla f(y), x \rangle + \frac{L}{2} \left\| \left(x - y + \frac{1}{L} \nabla f(y) \right) - \frac{1}{L} \nabla f(y) \right\|^2 + g(x) \right) \quad (2.47)$$

$$= \operatorname{argmin}_{x \in \mathbb{R}^n} \left(\langle \nabla f(y), x \rangle + \frac{L}{2} \left\| x - y + \frac{1}{L} \nabla f(y) \right\|^2 - \frac{L}{2} \left(\frac{2}{L} \langle \nabla f(y), x \rangle \right) + g(x) \right) \quad (2.48)$$

$$= \operatorname{argmin}_{x \in \mathbb{R}^n} \left(\frac{L}{2} \left\| x - \left(y - \frac{1}{L} \nabla f(y) \right) \right\|^2 + g(x) \right) \quad (2.49)$$

$$= \operatorname{prox}_{\frac{1}{L}g} \left(y - \frac{1}{L} \nabla f(y) \right). \quad (2.50)$$

Given $x_0 \in \mathbb{R}^n$, the proximal gradient algorithm generates a sequence $(x_k)_{k \in \mathbb{N}}$ via

$$x_{k+1} = \operatorname{prox}_{\frac{1}{L}g} \left(x_k - \frac{1}{L} \nabla f(x_k) \right). \quad (2.51)$$

This provides an intuitive view of the PG algorithm, but it can also be viewed from an operator standpoint. In the previous section, it was seen that iterations of averaged operators converge to a fixed point, and that compositions of averaged operators are averaged. To that end, set $T_1 = \operatorname{Id} - \frac{1}{L} \nabla f$ and $T_2 = \operatorname{prox}_{\frac{1}{L}g}$. It was stated that T_2 is firmly nonexpansive, hence $1/2$ averaged. In fact, T_1 is also firmly nonexpansive. The following theorem due to Baillon and Haddad relates nonexpansiveness and firm nonexpansiveness of the gradient.

Theorem 2.14 (Baillon - Haddad [7]). *Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ be convex, and differentiable. Then ∇f is nonexpansive if and only if ∇f is firmly nonexpansive*

It is easy to see by L -smoothness of f that $\frac{1}{L} \nabla f$ is nonexpansive, and hence by the above theorem it is firmly nonexpansive, and thus equivalently $\operatorname{Id} - \frac{1}{L} \nabla f$ is firmly nonexpansive. Therefore,

$$T = T_2 \circ T_1 = \operatorname{prox}_{\frac{1}{L}g} \left(\operatorname{Id} - \frac{1}{L} \nabla f \right)$$

is averaged. By Proposition 2.13 PG converges to a fixed point of T provided one exists. It remains only to verify that the fixed point that PG converges to is in fact a minimizer of (\mathbf{P}) . To that end, let $x \in \mathbb{R}^n$, then

$$\begin{aligned}
x \in \text{Fix } T &\iff x = Tx \\
&\iff x = \text{prox}_{\frac{1}{L}g} \left(\text{Id} - \frac{1}{L} \nabla f \right) (x) \\
&\iff x - \frac{1}{L} \nabla f(x) \in (\text{Id} + \partial(\frac{1}{L}g))(x) \\
&\iff -\frac{1}{L} \nabla f(x) \in \partial(\frac{1}{L}g)(x) \\
&\iff 0 \in \frac{1}{L} \nabla f(x) + \partial(\frac{1}{L}g)(x) \\
&\iff 0 \in \nabla f(x) + \partial g(x) \\
&\iff 0 \in \partial f(x) + \partial g(x) \quad (\text{f is differentiable hence } \partial f(x) = \{\nabla f(x)\}) \\
&\iff 0 \in \partial(f + g)(x) \quad (\text{sum rule holds because } \text{dom } f = \mathbb{R}^n) \\
&\iff x \text{ is a global minimizer of } f + g.
\end{aligned}$$

Seeing that the fixed point provided by repeated applications of T , the algorithm given by the update rule Eq. (2.51) will provide a solution to (\mathbf{P}) .

An acceleration, in the same way that AGD comes from GD, can be applied to PG. This accelerated proximal gradient (APG) algorithm will be the main focus of this thesis. The acceleration was introduced by Beck and Teboulle[15], and an acceleration to proximal gradient was also studied by Nesterov [28] around the same time. Both methods provide a rate of convergence in function value on the order of $\frac{1}{k^2}$.

Algorithm 1 Accelerated Proximal Gradient (APG)

Require: $x_0 \in \mathbb{R}^n$, ∇f is Lipschitz continuous with constant $L > 0$, a parameter sequence $(t_k)_{k \in \mathbb{N}}$ satisfying (2.17)

$y_0 \leftarrow x_0$

$t_0 \leftarrow 1$

for $k = 0, 1, 2 \dots$ **do**

$x_{k+1} \leftarrow \text{prox}_{\frac{1}{L}g} (y_k - \frac{1}{L} \nabla f(y_k))$

$\beta \leftarrow \frac{t_k - 1}{t_{k+1}}$

$y_{k+1} \leftarrow x_{k+1} + \beta(x_{k+1} - x_k)$

end for

The following theorem states the convergence rate of the algorithm.

Theorem 2.15. [15, Theorem 4.4] *Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ be convex, differentiable, and let $g : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ be convex, lsc, and proper. Further assume that f is L -smooth with constant $L > 0$. Let $x^* = P_{X^*}(x_0)$ where X^* is the set of solutions to the composite problem, and denote $F = f + g$. Given $x_0 \in \mathbb{R}^n$, and sequences $(x_k)_{k \in \mathbb{N}}, (y_k)_{k \in \mathbb{N}}$ generated by Algorithm 1, we have for $k \geq 1$*

$$F(x_k) - F(x^*) \leq \frac{2L\|x_0 - x^*\|^2}{(k+1)^2}. \quad (2.52)$$

Setting $g \equiv 0$ recovers AGD as the prox operator reduces to Id. This means that both GD and AGD can be viewed as special cases of their respective proximal based methods, and as such, we can recover the convergence results by looking at the composite case only.

One important difference is that in PG we have convergence of iterates to a fixed point, but in APG there is no such guarantee. In PG, the iterates are fixed point iterations so Proposition 2.13 guaranteed convergence, but in APG, the operator T is applied to the $(y_k)_{k \in \mathbb{N}}$ sequence which is an affine combinations of the x_k 's.

Another important remark is that for both APG, and PG to be efficient, we require that g has a prox that can be exactly computed. If the prox of g is computed inexactly then as long as there is sufficient control on the error, both PG, and APG will recover the same convergence rate as the exact version [35].

Chapter 3

Restarts

The basis of this thesis is on restarting accelerated gradient type methods. This chapter will serve as an overview of the topic of restarting algorithms, as well as the two different strategies of restarting.

Restarting is the idea of running an algorithm for a certain amount of steps, then reinitializing it with the last iterate outputted. This idea is not unique to the methods mentioned earlier in this thesis. Restarts have been studied for derivative free methods [24], stochastic first order methods [23], [37], and algorithms like Primal-Dual Hybrid Gradient (PDHG) [4].

There are two types of strategies when it comes to restarting, namely *fixed restart* and *adaptive restart* strategies. In fixed restart strategies, a number of iterations before the algorithm restarts is specified beforehand. On the other hand, algorithms using adaptive restart strategies run until some condition is met; once the condition is met, the restart is triggered. Both approaches have advantages and disadvantages. When performing fixed restarts, the main question is whether or not it is possible to obtain a specific iteration number at which restarting will help. In adaptive methods, the restart condition must be easy to check, and actually occur.

We will go over both fixed, and adaptive restart strategies as concerned with accelerated first order methods.

3.1 Fixed Restarts

In this section consider the optimization problem

$$\min_{x \in \mathbb{R}^n} f(x). \tag{3.1}$$

Here $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is L -smooth, and μ -strongly convex. This is a candidate problem for AGD. Note that for this class of functions one can slightly modify AGD [31]:

Algorithm 2 Strongly Convex AGD

Require: $x_0 \in \mathbb{R}^n$, ∇f is Lipschitz continuous with constant $L > 0$, and μ strongly convex

```

 $y_0 \leftarrow x_0$ 
 $\beta^* \leftarrow \frac{1 - \sqrt{\mu/L}}{1 + \sqrt{\mu/L}}$ 
for  $k = 0, 1, 2 \dots$  do
     $x_{k+1} \leftarrow y_k - \frac{1}{L} \nabla f(y_k)$ 
     $y_{k+1} \leftarrow x_{k+1} + \beta^*(x_{k+1} - x_k)$ 
end for

```

Using Algorithm 2 will provide the following convergence bound on the function values,

$$f(x_k) - f^* \leq L \left(1 - \sqrt{\frac{\mu}{L}}\right)^k \|x_0 - x^*\|^2. \tag{3.2}$$

The above convergence rate achieves ε accuracy in

$$O\left(\sqrt{\frac{L}{\mu}} \log \frac{1}{\varepsilon}\right) \tag{3.3}$$

iterations.

For L -smooth and μ -strongly convex functions it is possible to compute an optimal restart frequency such that the upper bound on the function values à la (3.3) is recovered while using Algorithm 1 (with $g \equiv 0$) and restarting. In this setting the optimal restart frequency refers to amount of iterations between restarts to recover the convergence guarantee of AGD for an L -smooth and μ -strongly convex function. The optimal iteration number has been studied before, and the rederivation provided below will follow the one in [33].

Let j represent the number of times the algorithm has restarted, and let k be iteration number in the current restart segment. Let x^* be the unique minimizer of f . Then we have

$$f(x_{(j,k)}) - f^* \leq \frac{2L\|x_{(j,0)} - x^*\|^2}{k^2} \quad (3.4)$$

$$\leq \frac{4L}{\mu k^2} (f(x_{(j,0)}) - f^*). \quad (3.5)$$

The above inequality is obtained from using the guarantee on the function values from Algorithm 1 and the strong convexity of the objective function. Strong convexity can be applied jk times

$$f(x_{(j,0)}) - f^* \leq \left(\frac{4L}{\mu k^2}\right)^j (f(x_{(0,0)}) - f^*). \quad (3.6)$$

Minimizing over j , and k subject to the constraint that $jk = C$ where C is the total number of iterations yields

$$k^* = 2e\sqrt{\frac{L}{\mu}}, \quad (3.7)$$

where k^* is the length of the optimal restart interval. We explicitly show this computation. First note that we have $k > 0$ as we want to have a restart interval. Using the constraint $jk = C$ define $h(k) = (4L/(\mu k^2))^{\frac{C}{k}}$. To obtain the optimal k we differentiate

$$h'(k) = \frac{C \left(\frac{L}{\mu}\right)^{C/k} \cdot 4^{C/k} (2 \log k - \log(L/\mu) - \log 4 - 2)}{k^2 k^{\frac{2C}{k}}}. \quad (3.8)$$

We want to find $k \in \mathbb{N}$ such that the right hand side of (3.8) equals 0. To that end we only require that

$$2 \log k - \log(L/\mu) - \log 4 - 2 = 0. \quad (3.9)$$

Indeed we have,

$$2 \log k - \log(L/\mu) - \log 4 - 2 = 0 \tag{3.10}$$

$$\iff \log\left(\frac{k^2}{(L/\mu)4}\right) = 2 \tag{3.11}$$

$$\iff \frac{k^2}{4(L/\mu)} = e^2 \tag{3.12}$$

$$\iff k^2 = 4e^2 \left(\frac{L}{\mu}\right) \tag{3.13}$$

$$\iff k = 2e\sqrt{\frac{L}{\mu}}, \tag{3.14}$$

as desired. Therefore having knowledge of the strong convexity constant, as well as the smoothness constant allows us to recover ε accuracy in $\mathcal{O}(\sqrt{L/\mu} \log(\frac{1}{\varepsilon}))$ iterations.

There are two drawbacks to the fixed restart approach mentioned above. First the parameter μ is hard to exactly calculate. Second, since the number of iterations between restarts is fixed for the whole algorithm, the restarts may occur at times where not restarting would be more beneficial [33].

3.2 Adaptive Restarts

Adaptive strategies aim to address the drawbacks of fixed restarts. They are intended to be based on easy to verify conditions, and these conditions should only hold when it is beneficial to restart. O’Donoghue and Candès introduced two such restart conditions [33] for AGD. They are the *function value* and *gradient* schemes. Both of these strategies are heuristic and the authors showed no proof other than the case of a strongly convex quadratic objective function. This was done by rewriting AGD as a dynamical system and relating the restarts to fixed restart intervals in which they are able to recover the optimal linear convergence rate.

Let $(x_k)_{k \in \mathbb{N}}$, and $(y_k)_{k \in \mathbb{N}}$ be the sequences generated by Algorithm 1. The schemes, by design are inherently simple. In the function value strategy, the restart occurs whenever

$$f(x_{k+1}) > f(x_k). \tag{3.15}$$

The scheme is intuitive as any step which increases function value goes against the goal of minimizing the objective. The gradient scheme requires checking if

$$\langle \nabla f(y_k), x_{k+1} - x_k \rangle > 0. \quad (3.16)$$

The intuition behind this scheme is that $x_{k+1} - x_k$ is the direction the iterates given by the algorithm are moving in, and $-\nabla f(y_k)$ is the direction which points in the greatest decrease from y_k . Therefore, if the above condition is satisfied then the angle between the negative gradient and the direction of the outputs is obtuse. This is a possible discrepancy of the best direction the iterates should take, therefore making it a sensible restart condition.

Note that both of the above restart conditions are easy to check. Furthermore, the function scheme requires an extra evaluation of f at x_k . This could be expensive depending on the objective. On the other hand, all terms in the gradient condition are computed as part of the algorithm, and only the computation of the inner product is required. O’Donoghue and Candès state that the gradient scheme is more numerically stable due to possible cancellation errors when evaluating the function value condition. For those reasons, this thesis will mainly focus on the gradient-based restart scheme.

While both of these methods are heuristics, they perform remarkably well, and in terms of performance, both strategies perform similarly. Although the work of O’Donoghue and Candès focused on the smooth case, they did consider a nonsmooth example.

Suppose that f, g are now as in (P). Since g is possibly nonsmooth it may not be possible to evaluate the gradient of the objective. Instead the composite gradient mapping [29] is used. This is defined as,

$$G(x) = L \left(x - \text{prox}_{g/L} \left(x - \frac{1}{L} \nabla f(x) \right) \right) \quad (3.17)$$

As a remark, this hints to the similarity between gradient descent and proximal gradient method, because if we have an iterate x_k then the next step,

$$x_{k+1} = x_k - \frac{1}{L} G(x_k) = \text{prox}_{g/L} \left(x_k - \frac{1}{L} \nabla f(x_k) \right).$$

The gradient condition for nonsmooth functions (recall Eq. (3.16)) then becomes,

$$\langle G(y_k), x_{k+1} - x_k \rangle > 0. \quad (3.18)$$

Notice that by Algorithm 1 and Eq. (3.17) this in turn can be simply written as,

$$\langle y_k - x_{k+1}, x_{k+1} - x_k \rangle > 0. \quad (3.19)$$

3.3 Numerical Examples

In this short section we will provide two examples. First we will take a look at a classical problem and show the effectiveness of the restarting strategies introduced earlier. Second, we will look at an example due to Hinder and Lubin [22] which illustrates that restarting may not always be beneficial.

3.3.1 The Good

The classic problem that we shall look at is ℓ_1 regularized least squares. Given a matrix $A \in \mathbb{R}^{m \times n}$, and $b \in \mathbb{R}^m$ the objective is to solve

$$\min_{x \in \mathbb{R}^n} \frac{1}{2} \|Ax - b\|^2 + \lambda \|x\|_1. \quad (3.20)$$

Here,

$$\|x\|_1 = \sum_{i=1}^n |x_i|$$

and $\lambda > 0$ is the weight put on the desired sparsity of the solution. To fit the composite model, set $f(x) = \frac{1}{2} \|Ax - b\|^2$ and $g(x) = \lambda \|x\|_1$. Both functions are continuous, and convex. In addition, f is L -smooth with gradient $\nabla f(x) = A^T(Ax - b)$ and L is the maximum eigenvalue of $A^T A$. Note that it can be expensive to compute L this way, a sufficient upper bound is to instead compute the Frobenious norm of A . The ℓ_1 norm is not differentiable, hence we use proximal methods. The proximal operator of $\lambda \|\cdot\|_1$ is the well known soft-thresholding operator (see [14, Example 6.8]),

$$\text{prox}_{\lambda \|\cdot\|_1}(x) = (\max\{|x_i| - \lambda, 0\} \text{sgn}(x_i))_{i=1}^n, \quad (3.21)$$

where the operation on the right hand side is understood to be applied elementwise to a vector $x \in \mathbb{R}^n$.

The use of ℓ_1 regularization is widespread, seeing application in image denoising, signal recovery[17], medicine[20], and deep learning [6]. It falls under the more general linear inverse problem where the goal is to find $x \in \mathbb{R}^n$ such that

$$Ax = b + w,$$

where w is some noise vector.

For the numerical example, we generate random data, sampling the entries of $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$ from a standard normal distribution. We also set $m = 500, n = 400$. Below we compare, proximal gradient, APG, and the gradient based restart strategy.

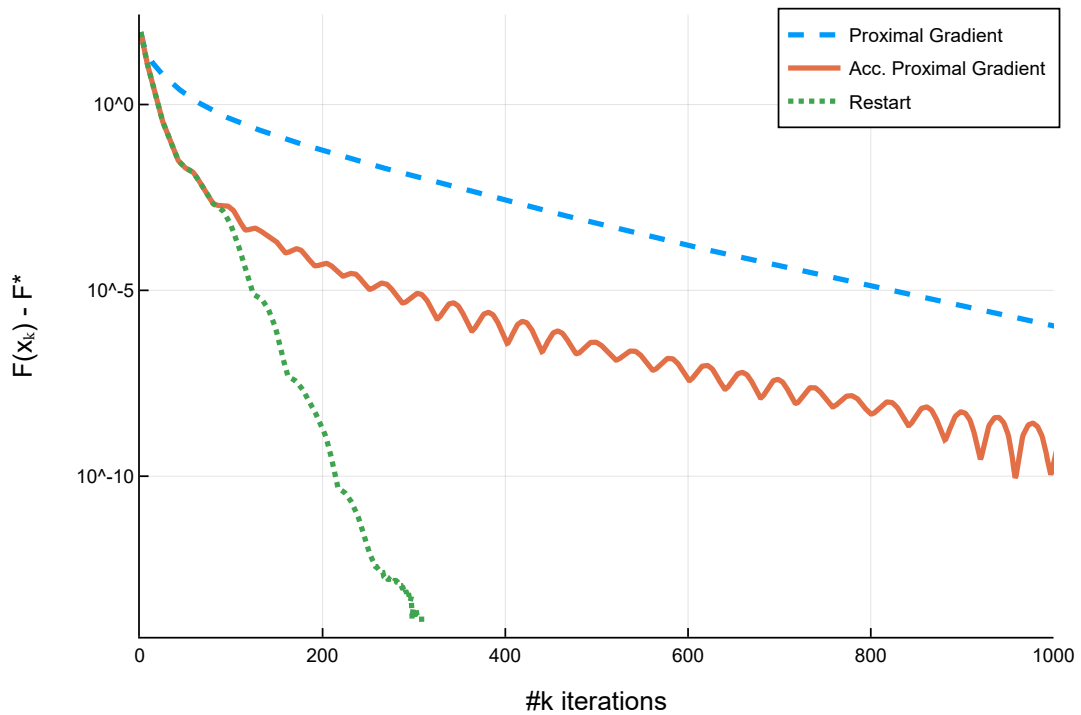


Figure 3.1: A plot showcasing the power of the adaptive restart strategy. The algorithm using the gradient-based strategy achieves a lower objective function value in much fewer iterations than APG, or PG.

We can observe in Fig. 3.1 that using the restart strategy achieves a higher accuracy solution in significantly fewer iterations.

3.3.2 A Negative Example

Hinder and Lubin[22] constructed a function for which the adaptive restart strategies of O’Donoghue and Candès[33] perform poorly while the iterates of APG are not close to the minimizer. Consider the function

$$h_\delta(z) = \begin{cases} z^2/2, & z \geq -\delta; \\ -\delta z - \delta^2/2, & z < -\delta, \end{cases} \quad (3.22)$$

where $\delta > 0$ and $\delta \in \mathbb{R}$. For a real number $\alpha > 0$ define $f : \mathbb{R}^n \rightarrow \mathbb{R}$ as

$$f(x) = \sum_{i=1}^n i h_\delta(x_i) + \frac{\alpha}{2} \|x\|^2. \quad (3.23)$$

The function is $(n + \alpha)$ smooth (can be seen by computing the Hessian) and α strongly convex with a unique minimizer at $x = 0$. In Fig. 3.2 we observe that the gradient based restarts do not speed up the convergence of the algorithm. The iterates of APG decrease in function value much faster, and it is only when the restarted algorithm gets close to the minimizer that the restarts start to make improvements.

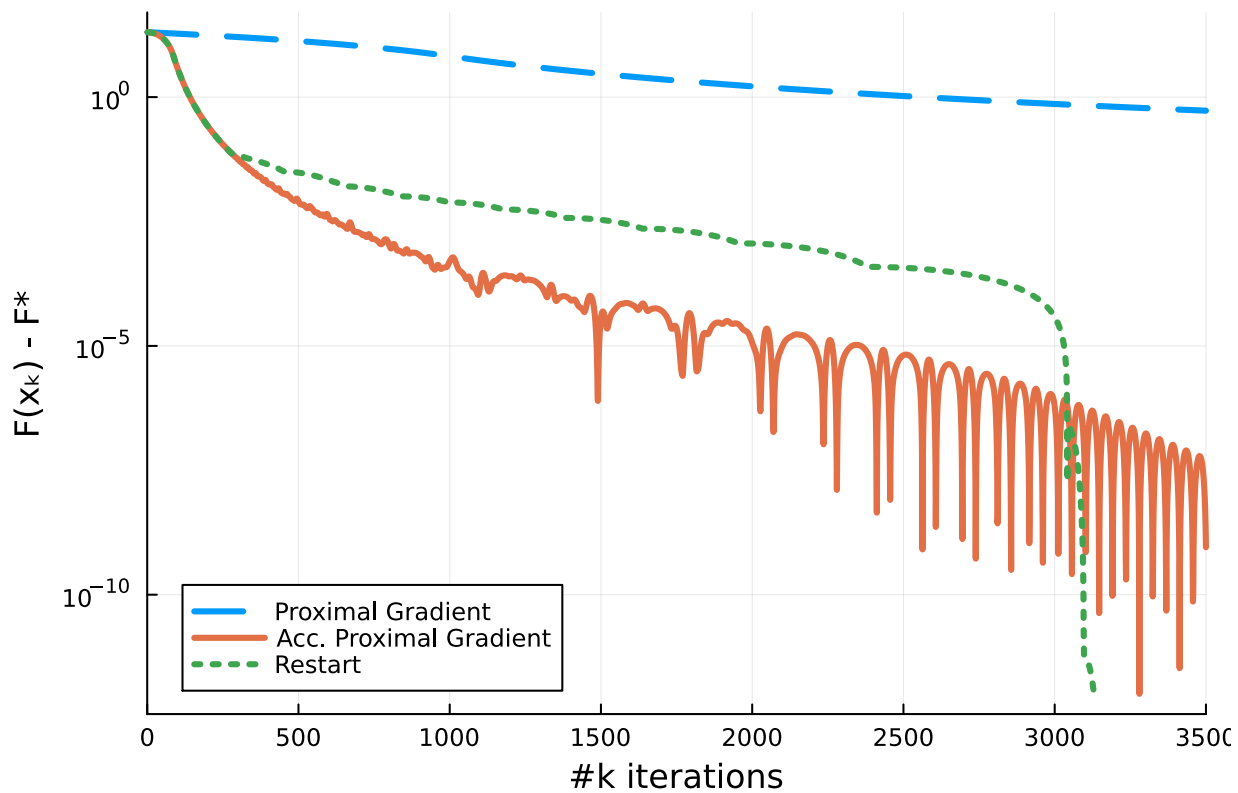


Figure 3.2: A Julia plot comparing PG, APG, and the gradient based restarts on the Hinder and Lubin example function. This showcases that the restarts do not always improve performance. In the numerical experiment we set $\alpha = \delta = 10^{-4}$.

Chapter 4

Literature Review

In this chapter we will go over several related works which concern restarting for accelerated first order methods. These works range from trying to establish the heuristic restart methods described previously; or they impose further properties on the objective function which allows the authors to introduce restart conditions that have convergence guarantees.

Note that we do not make comparisons to other first order methods, such as quasi-Newton. This is because quasi-Newton methods, even though they do not explicitly use second order information, require that the objective is twice continuously differentiable [32, Chapter 6], or some non-singular assumptions on the generalized Jacobian which we do not use. Furthermore we do not look at the Barzilai-Borwein (BB) method, another first order algorithm [8]. Due to the possibly aggressive step-size choice in BB the algorithm may not even converge for general convex functions. Recently we have learned of an Adaptive Barzilai-Borwein (AdaBB) [38] which performs well and has convergence results, which may open new avenues of research.

4.1 Monotonicity and Restarts

Giselsson and Boyd introduced a modified version of the restarted accelerated proximal gradient algorithm which has a provable convergence rate [21]. Their work considers the composite model (P), with a nonsmooth objective. Alongside their convergence result, they also introduce a new restart condition which can perform similarly to the function value, and gradient based restarts.

Algorithm 3 Giselsson-Boyd Restarted APG

Require: $x_0 \in \mathbb{R}^n$, ∇f is Lipschitz continuous with constant $L > 0$ a parameter sequence $(t_k)_{k \in \mathbb{N}}$ satisfying (2.17)

$y_0 \leftarrow x_0$
 $t_0 \leftarrow 1$

for $k = 0, 1, 2 \dots$ **do**

$x_{k+1} \leftarrow \text{prox}_{\frac{1}{L}g} \left(y_k - \frac{1}{L} \nabla f(y_k) \right)$
 $\beta \leftarrow \frac{t_k - 1}{t_{k+1}}$
 $y_{k+1} \leftarrow x_{k+1} + \beta(x_{k+1} - x_k)$

if Restart condition is satisfied **then**

$y_{k+1} \leftarrow x_k$
 $x_{k+1} \leftarrow \text{prox}_{\frac{1}{L}g} \left(y_{k+1} - \frac{1}{L} \nabla f(y_{k+1}) \right)$

end if

end for

Algorithm 3 has the following result.

Theorem 4.1. [21, Giselsson-Boyd] *Suppose that $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is convex, proper, and L -smooth, and that $g : \mathbb{R}^n \rightarrow \mathbb{R}$ is proper, lsc, convex, and denote $F = f + g$. Further suppose that a restart condition is satisfied at iterations $k = k_1, \dots, k_p$, then Algorithm 3 with a parameter sequence satisfying Eq. (2.17) converges with the rate*

$$F(x_{k+1}) - F^* \leq \frac{2L}{(k+2)^2} \left(\|x_0 - x^*\|^2 + \sum_{k_i \leq k} (\|x_{k_i} - x^*\|^2 - \|x^* - z_{k_i}\|^2) \right), \quad (4.1)$$

where $z_k = x_{k-1} + t_{k-1}(x_k - x_{k-1})$.

One observation is that if the restart condition in Algorithm 3 is never satisfied, then the $\mathcal{O}(1/k^2)$ convergence rate of APG is recovered. The key difference between Algorithm 3 and the restarted version of APG is that in the Giselsson and Boyd algorithm the parameter sequence is not reset; this is what allows them to obtain the convergence rate in the above theorem.

In Theorem 4.1 if the sum is negative then the convergence rate is improved over APG. Giselsson and Boyd observed through numerical experiments that $\|x_k - x^*\| < \|z_k - x^*\|$ when non-monotonicity is observed in the function values. Although there is no proof that this is always the case, it does align with the function value based restarts from O’Donoghue

and Candés. To circumvent the possible issues with the function value restarts Giselsson and Boyd introduced a non monotonicity implying restart test. Their test at iteration k requires y_{k-1}, x_k, x_{k+1} , and is

$$\langle y_{k-1} - x_k, x_{k+1} - \frac{1}{2}(x_k + y_{k-1}) \rangle > 0. \quad (4.2)$$

Overall this restart test performs slightly worse than both the function value test, or the gradient test. This is not too surprising as the test is a one way implication for detecting non monotonicity so it will not occur as often as the function value test.

4.2 Restarts Under Quadratic Growth

The restart methods of O’Donoghue and Candès do not put many restrictions on the objective function in (P). However, if further assumptions are made, then more can be said about the advantages of restarts in APG. Fercoq and Qu investigate restarts under a quadratic growth assumption [19].

Let X^* be the set of solutions of the minimization problem and let $F^* = \inf_{x \in \mathbb{R}^n} (f + g)(x)$. We say that F satisfies a *local quadratic growth condition* if there exists some $\mu > 0$ such that for any $x \in \text{dom } F$

$$F(x) - F^* \geq \frac{\mu}{2} \text{dist}^2(x, X^*) \quad (4.3)$$

for all x in the level set of F at x_0 (the level set of F at x_0 is the set of $x \in \mathbb{R}^n$ such that $f(x) \leq f(x_0)$). The quadratic growth condition can be viewed as a generalization of strong convexity. Earlier we saw that for a strongly convex objective it is possible to obtain an optimal restart length for fixed restart strategies, it has been shown that under the quadratic growth assumption it is also possible to recover such an optimal restart interval [27].

Fercoq and Qu show that under quadratic growth, restarting at any fixed frequency of iterations provides a linearly convergent algorithm, hence adaptive restart strategies obtain linear convergence. Since the quadratic growth constant μ is usually not known, it has to be estimated. The restart strategy developed by Fercoq and Qu is based on these estimates. They take an estimate of μ and obtain a restart period based upon it, at the end of the restart period if the initial estimate is too large, then the estimate is halved and a new restart period is computed, otherwise the estimate, and the restart period stay the

same.

Aujol et al. also consider APG under quadratic growth [5]. Their algorithm is a parameter free version of APG, that means there is no need to know L , the smoothness constant, nor the quadratic growth constant μ . The strategy they introduced was to estimate L using backtracking, then after some iterations, they use the function values of the objective to estimate the value L/μ (commonly referred to as the condition number of a function [33],[5]). Using the estimate of the condition number they determine whether the number of iterations in the current restart period should be increased or unchanged. If the algorithm is restarted, a new L is estimated. The immediate benefits of the algorithm is that it is parameter free and thus can be applied to problems where L and μ cannot be computed. A drawback is that in the restarting scheme, the estimate of the condition number depends on function values of the objective. In cases where the function is expensive to compute, this can hinder performance, as opposed to the gradient based restarts.

Alamo et al. also consider restarting for composite problems under quadratic growth [2]. Their algorithm differs from Fercoq and Qu, and Aujol et al. because they make no estimates of the quadratic growth parameter, nor do they need any explicit knowledge of it. Their restart is based on the generalized gradient mapping mentioned earlier in (3.17), i.e.,

$$G(x) = L(x - \text{prox}_{\frac{g}{L}} \left(x - \frac{1}{L} \nabla f(x) \right)). \quad (4.4)$$

They also rely on the following property of Algorithm 1.

Proposition 4.2. [3, Appendix D Property 8(ii)] *Suppose that $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is L -smooth and convex, suppose that $g : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ is proper, lsc, and convex and denote by G the composite gradient (3.17). Furthermore suppose that the minimization problem $\min f + g$ has a solution. Then let $(x_k)_{k \in \mathbb{N}}$ and $(y_k)_{k \in \mathbb{N}}$ be sequences generated by Algorithm 1. It holds that*

$$\frac{1}{L^2} \|G(y_k)\|^2 \leq \frac{4L^2 \|x_0 - x^*\|}{(k+2)^2}, \quad (4.5)$$

where x^* is a the projection of x_0 onto the solution set.

We provide the full algorithm below. The restart condition is based on a reduction in the norm of $G(y_k)$. Furthermore the parameter sequence $(t_k)_{k \in \mathbb{N}}$ is fixed to be the classical sequence from Beck and Teboulle [15].

Algorithm 4 Alamo et. al. Gradient Restart

Require: $z_0 \in \mathbb{R}^n, \varepsilon > 0$ $j \leftarrow 0$ $t_0 \leftarrow 1$ $x_0, y_0 \leftarrow \text{prox}_{\frac{1}{L}g} \left(z_0 - \frac{1}{L} \nabla f(z_0) \right)$ $\rho_j \leftarrow \frac{1}{L} \|G(z_0)\|$ **for** $k = 0, 1, 2 \dots$ **do** $x_{k+1} \leftarrow \text{prox}_{\frac{1}{L}g} \left(y_k - \frac{1}{L} \nabla f(y_k) \right)$ $t_{k+1} \leftarrow \frac{1 + \sqrt{1 + 4t_k^2}}{2}$ $\beta \leftarrow \frac{t_k - 1}{t_{k+1}}$ $y_{k+1} \leftarrow x_{k+1} + \beta(x_{k+1} - x_k)$ **if** $\frac{1}{L} \|G(y_{k+1})\| \leq \frac{\rho_j}{e}$ **then** $j \leftarrow j + 1$ $\rho_j \leftarrow \frac{1}{L} \|G(y_k)\|$ $z_j \leftarrow y_{k+1}$ $t_{k+1} \leftarrow 1$ $x_{k+1}, y_{k+1} \leftarrow \text{prox}_{\frac{1}{L}g} \left(z_j - \frac{1}{L} \nabla f(z_j) \right)$ **end if****if** $\rho_j < \varepsilon$ **then**

Terminate

end if**end for**

The quadratic growth assumption is needed to ensure that the number of iterations needed to reduce the norm of the gradient is bounded by a constant depending on μ .

4.3 Other Restart Schemes

Outside of the quadratic growth assumption there are many restart schemes in the wild. One notable method is that by Su, Boyd, and Candès [36]. Their work investigated AGD through a differential equations point of view, and their restart scheme was not the primary result. The aforementioned restart scheme was called “speed restarting”. Given an iterate x_k if the following condition is satisfied,

$$\|x_k - x_{k-1}\| \leq \|x_{k-1} - x_{k-2}\| \tag{4.6}$$

then a restart occurs. The motivation for this scheme, is when the algorithm is viewed as an ODE, with trajectory X , the velocity \dot{X} is kept high. In practice they found that their restart scheme did not perform as well as the gradient scheme of O’Donoghue and Candès, although the optimization of their scheme was not the goal of the paper.

Hinder and Lubin also provided a restart scheme for the case of strongly convex functions. Their restarts are based on the use of a potential function that is always computable. This potential function differs depending on the algorithm, it is based on the theoretical convergence rate. For example in the case of AGD the potential function is $\varphi(t) = (1 + t)^2$. Let i denote a restart period and let t be the number of iterations in the current restart period, then their restart condition is

$$\frac{\|x_t^i - y_{i-1}\|}{(1 + t)^2} \leq \beta \frac{\|y_{i-1} - y_{i-2}\|}{(1 + \tau_{i-1})^2}, \quad (4.7)$$

where $\beta \in (0, 1)$ is specified by the user, and τ_{i-1} is the number of iterations in the previous restart interval. Through numerical experiments, they found that the above restart scheme is comparable to the heuristic method of O’Donoghue and Candès. Hinder and Lubin do obtain a convergence result for their scheme but it relies on strong convexity in the objective which cannot be relaxed to quadratic growth.

Chapter 5

Analysis in One Dimension

It is important to stress that at the time of completing this thesis, the function value, and the gradient restart scheme of O'Donoghue and Candès are still heuristics. Empirically, both schemes tend to perform better than APG, but there are no convergence guarantees. In this chapter, we will provide an analysis for the $n = 1$ case on the real line to show that the gradient based restarts improve upon the classical convergence rate of APG. The analysis can be extended to separable functions, and furthermore shows potential in application to nearly separable objectives.

5.1 Intuition

First, we will consider the problem of minimizing $f : \mathbb{R} \rightarrow \mathbb{R}$ that is L -smooth and convex. The gradient based restart condition becomes,

$$f'(y_k)(x_{k+1} - x_k) > 0. \tag{5.1}$$

For the restart condition to be satisfied both $f'(y_k)$ and $(x_{k+1} - x_k)$ must have the same sign. There are two observations that can be made. First the sign of $f'(y_k)$ depends on what side of the minimizer y_k is on. If x^* is a minimizer, then if $y_k < x^*$ we have $f'(y_k) < 0$ and vice versa. Second, and this statement will be proven later for the more general nonsmooth case, if say $x < x^*$ then the gradient step from x ,

$$\bar{x} = x - \frac{1}{L}f'(x)$$

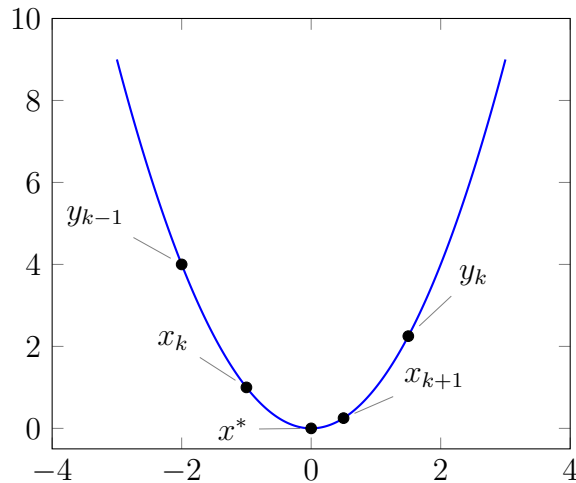


Figure 5.1: A plot of the iterates of AGD for a convex function. Note that y_k crosses the minimizer and thus (5.1) would be satisfied and a restart would occur.

will not cross the minimizer, that is $\bar{x} < x^*$. Looking at the updates for the iterates in Algorithm 1 we have that x_{k+1} is a gradient step from y_k , hence if for all k $y_k < x^*$ then $x_{k+1} < x^*$. In total, we get the ordering on the iterates as

$$x_0 = y_0 \leq x_1 \leq y_1 \leq \dots \leq x_k \leq y_k \leq x_{k+1},$$

if $x_0 < x^*$. Using the above ordering, and numerical experiments we made the observation that (5.1) would be true at iteration $k \geq 2$ if y_k crosses the minimizer.

Additionally, if we just monitor $f'(y_k)$, then if the value is negative, we should move right on the real line to approach the minimizer, and if it is positive, then we should move left. It is intuitive that the restarts should be beneficial when we encounter a sign change in $f'(y_k)$ as that means the iterates of the algorithm have passed over the minimizer. This also explains why the restarts would not be beneficial in gradient descent. Note, also that there is no obvious way to define what restarting would do for gradient descent since there is no history kept in terms of momentum.

5.2 Analysis

In this section, we will show the analysis when applying the gradient restart scheme (5.1) in one dimension. We have made a slight modification to the restart algorithm from

O’Donoghue and Candès. If the restart condition is satisfied at iteration $k \geq 1$ they set x_{k+1}, y_{k+1} to be x_k , whereas our algorithm keeps x_{k+1} and sets y_{k+1} to x_{k+1} . This was done for two reasons: first, it was observed that it offered slightly better performance, and secondly, it made the proof of our main result simpler. Another comment is that while any parameter sequence (t_k) satisfying Eq. (2.17) can be used for APG, in our algorithm we set

$$t_k = \frac{k+2}{2}.$$

The full one algorithm for our analysis is printed below. We restate the problem that is our focus,

$$\min_{x \in \mathbb{R}} F(x) := f(x) + g(x). \quad (5.2)$$

Algorithm 5 APG Gradient Restart in One Dimension

Require: $x_0 \in \mathbb{R}$

$j \leftarrow 0$

$t_0 \leftarrow 1$

$y_0 \leftarrow x_0$

for $k = 0, 1, 2 \dots$ **do**

$x_{k+1} \leftarrow \text{prox}_{\frac{1}{L}g} \left(y_k - \frac{1}{L} \nabla f(y_k) \right)$

$t_{k+1} \leftarrow \frac{(j+1)+2}{2}$

$\beta \leftarrow \frac{t_k-1}{t_{k+1}}$

$y_{k+1} \leftarrow x_{k+1} + \beta(x_{k+1} - x_k)$

if $(y_k - x_{k+1})(x_{k+1} - x_k) > 0$ **then**

$j \leftarrow 0$

$y_k \leftarrow x_{k+1}$

$t_{k+1} \leftarrow 1$

end if

$j \leftarrow j + 1$

end for

In the previous section it was mentioned that taking a gradient step of the appropriate size will not result in crossing the minimizer. Below we introduce a lemma that formalizes this for the proximal gradient step.

Lemma 5.1. *Let $F = f + g$, where $f : \mathbb{R} \rightarrow \mathbb{R}$ is convex and L -smooth, and $g : \mathbb{R} \rightarrow \mathbb{R} \cup \{+\infty\}$ is convex, lsc, and proper. Let $x \in \mathbb{R}$ and $x^* \in X^*$ a minimizer. Let T denote*

$\text{prox}_{g/L}(\text{Id} - \frac{1}{L}\nabla f)$. If $x < x^*$ we have,

$$x < Tx \leq x^*. \quad (5.3)$$

Similarly if $x > x^*$ then,

$$x^* \leq Tx < x. \quad (5.4)$$

Proof. By assumption we have that f is L -smooth, hence by Theorem 2.2(ii) we have that for all $x \in \mathbb{R}$,

$$F(x) \leq f(y) + f'(y)(x - y) + \frac{L}{2}(x - y)^2 + g(x) \quad (5.5)$$

for all $x \in \mathbb{R}$. Set,

$$Q(x; y) := f(y) + f'(y)(x - y) + \frac{L}{2}(x - y)^2 + g(x). \quad (5.6)$$

For a fixed $y \in \mathbb{R}$, $Q(x; y)$ is a strongly convex function in x , therefore it admits a unique minimizer. Let $q^* = \text{argmin} \{Q(x; y) : x \in \mathbb{R}\}$. It was verified that $q^* = \text{prox}_{g/L}(x - \frac{1}{L}f'(x)) = Tx$ (recall Eq. (2.45)). Therefore minimizing the quadratic upper bound is equivalent to taking a proximal gradient step.

Without loss of generality suppose that $x < x^*$. The goal is to show that $x < q^* = Tx \leq x^*$. Since T is nonexpansive, and x^* is a solution of the minimization problem, $x^* = Tx$. It follows that

$$|x^* - Tx| = |Tx^* - Tx| \quad (5.7)$$

$$< |x^* - x|, \quad (5.8)$$

therefore as $x < x^*$, we conclude $x < Tx$. Next, we wish to show that $Tx \leq x^*$. Consider the quadratic upper bound at x and let $z \in \mathbb{R}$, then the subdifferential is given by $\partial Q(z; x) = f'(x) + L(z - x) + \partial g(z)$. By L -smoothness of f we have $f'(z) \leq f'(x) + L(z - x)$ for any $z > x$. Therefore we obtain that,

$$\sup \partial(f + g) \leq \sup \partial Q(z; x), \quad (5.9)$$

when $z > x$. As $Q(z; x)$ is strongly convex, q^* is a unique minimizer, so by Fermat's rule we have $0 \in \partial Q(q^*; x)$.

Now consider $z \in (x, q^*)$. Let $u = \sup \partial Q(z; x)$. By the monotonicity of the subgradient,

$$(u - 0)(z - q^*) \geq 0. \quad (5.10)$$

We know that $z - q^* < 0$ so it must be that $u \leq 0$. In fact, $u < 0$ as we know that $z \neq q^*$ so $0 \notin \partial Q(z; x)$. Hence we can conclude that for all such z we have $\partial(f + g)(z) \leq u < 0$. Therefore, for all $z \in (x, q^*)$, $0 \notin \partial(f + g)(z)$ and so it must be that $q^* \leq x^*$. We now have that, $x < q^* = Tx \leq x^*$, as desired. Showing the case where $x > x^*$ is similar. \square

Next we will prove another lemma that characterizes restarting with crossing the minimizer.

Lemma 5.2. *Let $x_0 \in \mathbb{R}$ and let $(x_k)_{k \in \mathbb{N}}$ and $(y_k)_{k \in \mathbb{N}}$ be the sequences of iterates obtained by Algorithm 1. Let $k \in \mathbb{N}$ be such that for all $\bar{k} \leq k$ the restart condition,*

$$(y_k - x_{k+1})(x_{k+1} - x_k) > 0$$

has not been satisfied. Let x^ be a solution of (P). Then the following hold.*

1. *If $x_k < x^*$ then the restart condition is satisfied at iteration k if and only if $x_k < x^* < x_{k+1}$.*
2. *If $x_k > x^*$ then the restart condition is satisfied at iteration k if and only if $x_k > x^* > x_{k+1}$.*

Proof. (1) Suppose that $x_k \leq x^*$.

“ \Leftarrow ” Suppose we have

$$x_k < x^* < x_{k+1}.$$

By Lemma 5.1,

$$y_{k-1} < x_k < x^* < x_{k+1} < y_k.$$

This gives us that $y_k - x_{k+1} > 0$ and $x_{k+1} - x_k > 0$ hence

$$(y_k - x_{k+1})(x_{k+1} - x_k) > 0.$$

“ \Rightarrow ” Now suppose that the restart condition is satisfied at iteration k so,

$$(y_k - x_{k+1})(x_{k+1} - x_k) > 0.$$

For $i \leq k - 1$ we have $(y_i - x_{i+1})(x_{i+1} - x_i) \leq 0$. Since $x_k < x^*$ then by Lemma 5.1 we have that $y_{k-1} < x_k$. So, we must have $y_{k-1} - x_k < 0$ and since the restart condition is not satisfied at $k - 1$ it must be that $x_k - x_{k-1} > 0$. Now looking at the definition of y_k we have,

$$y_k = x_k + \frac{t_{k-1} - 1}{t_k}(x_k - x_{k-1}).$$

From the definition we obtain $y_k > x_k$. Suppose for an eventual contradiction that $y_k < x^*$. By the previous lemma, we have $x_{k+1} > y_k$ and hence $y_k - x_{k+1} < 0$ but we also have $x_{k+1} - x_k > 0$ so

$$(y_k - x_{k+1})(x_{k+1} - x_k) < 0.$$

Hence, we arrive at a contradiction, since we assumed that at iteration k the restart condition was satisfied.

(2) The second statement is proved in a similar way to (1). □

Lemma 5.2 gives an ordering of the iterates x_k between two restart periods. Suppose we restart at iterations r_1 and r_2 with $r_2 > r_1$. If $y_{r_1} < x^*$ then for all $k \in \{r_1, \dots, r_2 - 1\}$ we have $f'(y_k) < 0$. Since we do not restart at any of these iterations we must have that $x_{k+1} - x_k > 0$ for all such k . Similarly, if $y_{r_1} > x^*$ we deduce that $x_{k+1} - x_k < 0$. The next lemma will provide an inequality which will be fundamental in the convergence analysis of the restarted algorithm.

Lemma 5.3. *Let $x_0 \in \mathbb{R}$ and let $(x_k)_{k \in \mathbb{N}}$ and $(y_k)_{k \in \mathbb{N}}$ be the sequences obtained by Algorithm 5. Let $k \in \mathbb{N}$ be such that for all $\bar{k} \leq k - 1$ the restart condition has not been satisfied and that k is the first iteration where the restart condition is satisfied, i.e.,*

$$(y_k - x_{k+1})(x_{k+1} - x_k) > 0. \tag{5.11}$$

Define $z_{k+1} := (1 - t_k)x_k + t_kx_{k+1}$, and let x^* be a minimizer of P . Then,

$$|z_{k+1} - x^*| \geq t_k|x_{k+1} - x^*|.$$

Proof. Suppose that $x_0 < x^*$. By Lemma 5.2 we have $x_k < x^* < x_{k+1}$. It follows from this, the definition of z_{k+1} and Eq. (2.17) that

$$\begin{aligned} |z_{k+1} - x^*| &= |(1 - t_k)x_k + t_kx_{k+1} - x^*| \\ &= |(1 - t_k)(x_k - x^*) + t_k(x_{k+1} - x^*)| \\ &= (t_k - 1)|x_k - x^*| + t_k|x_{k+1} - x^*| \\ &\geq t_k|x_{k+1} - x^*|. \end{aligned} \tag{5.12}$$

The case where $x_0 > x^*$ is shown similarly. □

Before we get to the main results, we recall a useful fact that will aid us in proving our key theorems.

Fact 5.4. [13, Theorem 30.4] Let $x_0 \in \mathbb{R}$ and let $(x_k)_{k \in \mathbb{N}}$ and $(t_k)_{k \in \mathbb{N}}$ be given by Algorithm 1. For $k \in \{0, 1, 2, \dots\}$ we set

$$z_{k+1} = (1 - t_k)x_k + t_k x_{k+1}, \quad (5.13a)$$

$$\delta_k = F(x_k) - F(x^*). \quad (5.13b)$$

Then we have the following monotonicity property,

$$t_k^2 \delta_k + \frac{L}{2} \|z_{k+1} - x^*\|^2 \leq t_{k-1}^2 \delta_k + \frac{L}{2} \|z_k - x^*\|^2. \quad (5.14)$$

Fact 5.5. ([14, Theorem 10.16]) Let $x \in \mathbb{R}$ and let $y \in \mathbb{R}$. Then

$$f(x) - f(T(y)) \geq \frac{L}{2} \|x - T(y)\|^2 - \frac{L}{2} \|x - y\|^2. \quad (5.15)$$

Remark. The above facts hold in \mathbb{R}^n .

We are now ready to present our two main results. The first theorem shows that restarting once will improve the classical rate of convergence.

Theorem 5.6 (a single restart.). *Let $x_0 \in \mathbb{R}$ and let $(x_k)_{k \in \mathbb{N}}, (y_k)_{k \in \mathbb{N}}, (t_k)_{k \in \mathbb{N}}$ be given by Algorithm 5. Suppose that iteration r is the first iteration where we have*

$$(y_k - x_{k+1})(x_{k+1} - x_k) > 0, \quad (5.16)$$

and that iteration \bar{r} is the second iteration where we have $(y_{\bar{r}} - x_{\bar{r}+1})(x_{\bar{r}+1} - x_{\bar{r}}) > 0$. Then the following hold.

(i) $(\forall k \leq r + 1)$ we have $F(x_k) - F^* \leq \frac{2L(x_0 - x^*)^2}{(k+1)^2}$.

(ii) $F(x_{r+2}) \leq F(x_{r+1})$.

(iii) $(\forall k \in \{r + 3, \dots, \bar{r} + 1\})$ we have

$$F(x_k) - F^* \leq \left(\frac{4}{(k-r)(r+2)} \right)^2 \frac{L(x_0 - x^*)^2}{2} \leq \frac{2L(x_0 - x^*)^2}{(k+1)^2}. \quad (5.17)$$

Proof. Observe that $r \geq 2$. In the following, we let $(\bar{x}_k)_{k \in \mathbb{N}}, (\bar{y}_k)_{k \in \mathbb{N}},$ and $(\bar{t}_k)_{k \in \mathbb{N}}$ be given by the classical APG update from Algorithm 1 with the starting point $\bar{x}_0 := x_0$. Clearly, $\bar{t}_0 = t_0 = 1$. Moreover, for all $k \leq r$ we have,

$$\bar{x}_k = x_k, \bar{t}_k = t_k. \quad (5.18)$$

Furthermore, since we keep the point after the restart iteration, we have $\bar{x}_{r+1} = x_{r+1}$. Now, since at iteration $r + 1$ we reset the parameter sequence we have that now $t_{r+1} = 1$, which means that for $k \geq r + 1$ we have the relation that $t_k = \bar{t}_{k-(r+1)}$. For $k \in \{0, 1, 2, \dots\}$ we set

$$\bar{z}_{k+1} := (1 - t_k)\bar{x}_k + \bar{t}_k\bar{x}_{k+1}, \quad (5.19a)$$

$$\bar{\delta}_k := F(\bar{x}_k) - F(x^*), \quad (5.19b)$$

and we set

$$z_{k+1} := (1 - t_k)x_k + t_kx_{k+1}, \quad (5.20a)$$

$$\delta_k := F(x_k) - F(x^*). \quad (5.20b)$$

Due to the restart,

$$z_{r+2} = (1 - t_{r+1})x_{r+1} + t_{r+1}x_{r+2} = x_{r+2}. \quad (5.21)$$

Let $k \geq r + 2$. Using Fact 5.4 we have

$$\delta_k = F(x_k) - F(x^*) \leq \frac{1}{t_{k-1}^2} \left(\frac{L}{2}(z_k - x^*)^2 + t_{k-1}^2\delta_k \right) \quad (5.22a)$$

$$\leq \frac{1}{t_{k-1}^2} \left(\frac{L}{2}(z_{k-1} - x^*)^2 + t_{k-2}^2\delta_{k-1} \right) \quad (5.22b)$$

$$\leq \quad (5.22c)$$

$$\vdots \quad (5.22d)$$

$$\leq \frac{1}{t_{k-1}^2} \left(\frac{L}{2}(z_{r+2} - x^*)^2 + t_{r+1}^2\delta_{r+2} \right) \quad (5.22e)$$

$$= \frac{1}{t_{k-1}^2} \left(\frac{L}{2}(x_{r+2} - x^*)^2 + \delta_{r+2} \right), \quad (5.22f)$$

where we now used in the equality that $t_{r+1} = 1$, and that $z_{r+2} = x_{r+2}$. Recall from Algorithm 1 that we update x_k by taking a proximal gradient step from y_{k-1} .

$$x_{r+2} = \text{prox}_{g/L} \left(y_{r+1} - \frac{1}{L}f'(y_{r+1}) \right)$$

but since we restarted we have that $y_{r+1} = x_{r+1}$, so

$$x_{r+2} = \text{prox}_{g/L} \left(x_{r+1} - \frac{1}{L}f'(x_{r+1}) \right).$$

This observation allows us to make further progress. On the one hand, recalling $y_{r+1} = x_{r+1}$, it follows from T being nonexpansive and applied to x_{r+1}

that

$$(x_{r+2} - x^*)^2 \leq (x_{r+1} - x^*)^2. \quad (5.23)$$

On the other hand, because x_{r+2} is a gradient step from x_{r+1} with a proper step-length we have that $F(\bar{x}_{r+1}) = F(x_{r+1}) \geq F(x_{r+2}) > F(x^*)$ which, in view of Eq. (5.20b), implies that $\delta_{r+2} \leq \delta_{r+1} = \bar{\delta}_{r+1}$. Combining this and Eq. (5.22f) yields

$$\delta_k \leq \frac{1}{\bar{t}_{k-r-2}^2} \left(\frac{L}{2} (x_{r+1} - x^*)^2 + \bar{\delta}_{r+1} \right) \quad (5.24)$$

where we used that $x_{r+1} = \bar{x}_{r+1}$, and that $t_{k-1} = \bar{t}_{k-(r+1)-1} = \bar{t}_{k-r-2}$. Proposition 5.3 yields

$$\delta_k \leq \frac{1}{\bar{t}_{k-r-2}^2} \left(\frac{L}{2} \frac{1}{\bar{t}_r^2} (\bar{z}_{r+1} - x^*)^2 + \bar{\delta}_{r+1} \right) \quad (5.25a)$$

$$= \frac{1}{\bar{t}_{k-r-2}^2 \bar{t}_r^2} \left(\frac{L}{2} (\bar{z}_{r+1} - x^*)^2 + \bar{t}_r^2 \bar{\delta}_{r+1} \right) \quad (5.25b)$$

$$\leq \quad (5.25c)$$

$$\vdots \quad (5.25d)$$

$$\leq \frac{1}{\bar{t}_{k-r-2}^2 \bar{t}_r^2} \left(\frac{L}{2} (\bar{z}_1 - x^*)^2 + \bar{\delta}_1 \right) \quad (5.25e)$$

$$= \frac{1}{\bar{t}_{k-r-2}^2 \bar{t}_r^2} \left(\frac{L}{2} (\bar{x}_1 - x^*)^2 + \bar{\delta}_1 \right) \quad (5.25f)$$

$$\leq \frac{L}{2(\bar{t}_{k-r-2} \bar{t}_r)^2} (x_0 - x^*)^2 \quad (5.25g)$$

$$\leq \frac{8L}{((k-r)(r+2))^2} (x_0 - x^*)^2, \quad (5.25h)$$

where Eq. (5.25g) follows from Fact 5.5 applied with x replaced by x^* and y replaced by x_0 and Eq. (5.25g) follows from Eq. (2.17). To finish the proof we claim that for $k \geq r+3$ we have

$$\frac{4}{((k-r)(r+2))^2} \leq \frac{1}{(k+1)^2}. \quad (5.26)$$

Indeed, observe that

$$Eq. (5.26) \Leftrightarrow 2(k+1) \leq (k-r)(r+2) \quad (5.27a)$$

$$\Leftrightarrow r(k-(r+2)) \geq 2, \quad (5.27b)$$

which is always true for $k \geq r+3$ by recalling that $r \geq 2$. By definition, $x_{r+2} = T(x_{r+1})$ and hence $|x_{r+2} - x^*| \leq |x_{r+1} - x^*|$ and $F(x_{r+2}) \leq F(x_{r+1})$. \square

Next, we present the result for multiple restarts guaranteeing that the gradient based restarts in one dimension will be helpful.

Theorem 5.7 (multiple restarts.). *Let $x_0 \in \mathbb{R}$ and let $(x_k)_{k \in \mathbb{N}}, (y_k)_{k \in \mathbb{N}}, (t_k)_{k \in \mathbb{N}}$ be given by Algorithm 5. Let $p \in \mathbb{N}$. Suppose that r_p is the p^{th} -iteration such that*

$$(y_k - x_{k+1})(x_{k+1} - x_k) > 0. \quad (5.28)$$

Then $(\forall p \geq 2) (\forall k \in \{r_p + 2, \dots, r_{p+1} + 1\})$ we have

$$F(x_k) - F(x^*) \leq \left(\frac{2^{p+1}}{(k - r_p)(r_p - r_{p-1} + 2) \dots (r_2 - r_1 + 2)(r_1 + 2)} \right)^2 \frac{L}{2} (x_0 - x^*)^2 \quad (5.29a)$$

$$\leq \frac{2L(x_0 - x^*)^2}{(k + 1)^2}. \quad (5.29b)$$

Proof. Let $i \in \{1, 2, \dots, p\}$, we introduce the following notation to identify the multiple restarts. We denote by $(x_j^{(r_i)})_{j \in \mathbb{N}}$ the sequence of iterates given by starting AGD with $x_0^{(r_i)} = x_{r_i+1}$ and $t_0^{(r_i)} = t_{r_i+1} = 1$. We can now identify that between two consecutive restarts say r_i and r_{i+1} we have that $(x_0^{(r_i)}, x_1^{(r_i)}, x_2^{(r_i)}, \dots, x_{r_{i+1}-r_i}^{(r_i)})$ coincide with $(x_{r_i+1}, x_{r_i+2}, x_{r_i+3}, \dots, x_{r_{i+1}})$. For iterations $k > r_p$ we have that, $(x_0^{(r_p)}, x_1^{(r_p)}, \dots, x_{k-r_p-1}^{(r_p)})$ coincide with $(x_{r_p+1}, x_{r_p+2}, \dots, x_k)$. Upholding the notation of the proof of Theorem 5.6. we similarly denote the corresponding values of $z^{(r_i)}$, $\delta^{(r_i)}$ and $t^{(r_i)}$. We proceed similarly to the proof of Theorem 5.6. Let $k \geq r_p + 2$. Then

$$F(x_k) - F(x^*) = F(x_{k-r_p-1}^{(r_p)}) - F(x^*) \quad (5.30a)$$

$$\leq \frac{1}{(t_{k-r_p-2}^{(r_p)})^2} \left(\frac{L}{2} (z_{k-r_p-1}^{(r_p)} - x^*)^2 + (t_{k-r_p-2}^{(r_p)})^2 \delta_{k-r_p-1}^{(r_p)} \right) \quad (5.30b)$$

$$\leq \quad (5.30c)$$

\vdots

$$\leq \frac{1}{(t_{k-r_p-2}^{(r_p)})^2} \left(\frac{L}{2} (z_1^{(r_p)} - x^*)^2 + (t_0^{(r_p)})^2 \delta_1^{(r_p)} \right) \quad (5.30d)$$

$$= \frac{1}{(t_{k-r_p-2}^{(r_p)})^2} \left(\frac{L}{2} (x_1^{(r_p)} - x^*)^2 + \delta_1^{(r_p)} \right). \quad (5.30e)$$

We now use that $x_1^{(r_p)}$ is a gradient step from $x_0^{(r_p)}$ to conclude

$$F(x_k) - F(x^*) \leq \frac{1}{(t_{k-r_p-2}^{(r_p)})^2} \left(\frac{L}{2} (x_0^{(r_p)} - x^*)^2 + \delta_0^{(r_p)} \right) \quad (5.31a)$$

$$= \frac{1}{(t_{k-r_p-2}^{(r_p)})^2} \left(\frac{L}{2} (x_{r_p-r_{p-1}+1}^{(r_p-1)} - x^*)^2 + \delta_{r_p-r_{p-1}+1}^{(r_p-1)} \right) \quad (5.31b)$$

where we get equality because when the r_p restart gets triggered we keep $x_{r_p-r_{p-1}+1}^{(r_p-1)}$ unchanged and set $x_0^{(r_p)} = x_{r_p-r_{p-1}+1}^{(r_p-1)}$. We can now use Lemma 5.3 to obtain

$$|x_{r_p-r_{p-1}+1}^{(r_p-1)} - x^*| \leq \frac{1}{t_{r_p-r_{p-1}}^{(r_p-1)}} |z_{r_p-r_{p-1}+1}^{(r_p-1)} - x^*|. \quad (5.32)$$

Combining Eq. (5.31) and Eq. (5.32) and factoring out the $\frac{1}{t_{r_p-r_{p-1}}^{(r_p-1)}}$ term we obtain

$$F(x_k) - F(x^*) \leq \frac{1}{(t_{k-r_p-2}^{(r_p)})^2 (t_{r_p-r_{p-1}}^{(r_p-1)})^2} \left(\frac{L}{2} (z_{r_p-r_{p-1}+1}^{(r_p-1)} - x^*)^2 + (t_{r_p-r_{p-1}}^{(r_p-1)})^2 \delta_{r_p-r_{p-1}+1}^{(r_p-1)} \right). \quad (5.33)$$

We now repeat the same procedure. Since the terms in the parentheses on the right-hand side of (5.33) have the monotonicity property from Fact 5.4, we use it repeatedly until we obtain an expression of the form of the right-hand side of in Eq. (5.14) that features $z_1^{(r_i)}$. We then apply the fact that $z_1^{(r_i)} = x_1^{(r_i)}$, which is a gradient step from $x_0^{(r_i)}$, and finally use Lemma 5.3. Observe that every time we apply Lemma 5.3 an additional factor of $1/(t_{r_{i+1}-r_i}^{(r_i)})^2$ for some $i \in \{1, \dots, p\}$ appears in the denominator of the right-hand side of the inequality (5.33). This is done p times until the term in the parentheses reduces to $(L/2)(x_0 - x^*)^2$. We therefore obtain

$$F(x_k) - F(x^*) \leq \frac{1}{(t_{k-r_p-2}^{(r_p)})^2 \left(\prod_{i=1}^p (t_{r_i-r_{i-1}}^{(r_i)})^2 \right)} \left(\frac{L}{2} (x_0 - x^*)^2 \right). \quad (5.34)$$

Recall that for each $i \in \{1, \dots, p\}$ ($\forall k \in \{0, \dots, r_i\}$) we have $t_k^{(r_i)} = (k+2)/2$, hence

$$\frac{1}{t_{k-r_p-2}^{(r_p)}} = \frac{2}{k-r_p} \quad (5.35)$$

and

$$\frac{1}{t_{r_i-r_{i-1}}^{(r_i)}} = \frac{2}{r_i - r_{i-1} + 2}. \quad (5.36)$$

Now let $k \in \mathbb{N}$. Observe that if $k \leq r_2 + 1$ we obtain the $\mathcal{O}(1/k^2)$ function value convergence rate from Theorem 5.6. We now show that $(\forall p \geq 2) (\forall k \geq r_p + 2)$ we have

$$\frac{2^{p+1}}{(k - r_p)(r_p - r_{p-1} + 2) \dots (r_2 - r_1 + 2)(r_1 + 2)} \leq \dots \leq \frac{4}{(k - r_1)(r_1 + 2)} \leq \frac{2}{(k + 1)}. \quad (5.37)$$

We proceed by induction on p . We first verify the base case at $p = 2$. Applying Eq. (5.35) and Eq. (5.36) to Eq. (5.34) we get

$$F(x_k) - F(x^*) \leq \left(\frac{8}{(k - r_2)(r_2 - r_1 + 2)(r_1 + 2)} \right)^2 \frac{L}{2} (x_0 - x^*)^2. \quad (5.38)$$

For $k = r_2 + 2$ we have

$$\frac{8}{(k - r_2)(r_2 - r_1 + 2)(r_1 + 2)} = \frac{8}{(r_2 + 2 - r_2)(r_2 - r_1 + 2)(r_1 + 2)} \quad (5.39a)$$

$$= \frac{8}{(2)((r_2 + 2) - r_1)(r_1 + 2)} \quad (5.39b)$$

$$= \frac{4}{(k - r_1)(r_1 + 2)} \leq \frac{2}{k + 1}, \quad (5.39c)$$

where the inequality follows from the fact that $r_1 \geq 2$ and $k = r_2 + 2 \geq r_1 + 3$. Now let $k \geq r_2 + 3$. Then

$$\begin{aligned} & \frac{8}{(k - r_2)(r_2 - r_1 + 2)(r_1 + 2)} - \frac{4}{(k - r_1)(r_1 + 2)} \\ &= \frac{4}{(r_1 + 2)} \left(\frac{2}{(k - r_2)(r_2 - r_1 + 2)} - \frac{1}{k - r_1} \right) \end{aligned} \quad (5.40a)$$

$$= \frac{4}{(r_1 + 2)} \left(\frac{2(k - r_1)}{(k - r_2)(r_2 - r_1 + 2)(k - r_1)} - \frac{(k - r_2)(r_2 - r_1 + 2)}{(k - r_2)(r_2 - r_1 + 2)(k - r_1)} \right) \quad (5.40b)$$

$$= \frac{4}{(r_1 + 2)} \left(\frac{2(k - r_1) - (k - r_2)(r_2 - r_1 + 2)}{(k - r_2)(r_2 - r_1 + 2)(k - r_1)} \right). \quad (5.40c)$$

Write $k = r_2 + a$ where $a \geq 3$. It is sufficient to show that the numerator of the right-hand side of Eq. (5.40c) is nonpositive. To this end, we have

$$2(k - r_1) - (k - r_2)(r_2 - r_1 + 2) = 2(r_2 + a - r_1) - (r_2 + a - r_2)(r_2 - r_1 + 2) \quad (5.41a)$$

$$= 2a + (r_2 - r_1) - 2a - a(r_2 - r_1) \quad (5.41b)$$

$$= (2 - a)(r_2 - r_1) \leq 0. \quad (5.41c)$$

Combining Eq. (5.39), Eq. (5.40), and Eq. (5.41) we conclude that for $k \geq r_2 + 2$ we have

$$\frac{8}{(k - r_2)(r_2 - r_1 + 2)(r_1 + 2)} \leq \frac{4}{(k - r_1)(r_1 + 2)} \leq \frac{2}{k + 1}.$$

This proves the base case. Now suppose that we have restarted p times, $p \geq 2$, and for $k \geq r_p + 2$ the following inequalities hold

$$\frac{2^{p+1}}{(k - r_p)(r_p - r_{p-1} + 2) \dots (r_2 - r_1 + 2)(r_1 + 2)} \leq \dots \leq \frac{4}{(k - r_1)(r_1 + 2)} \leq \frac{2}{k + 1}. \quad (5.42)$$

Consider the iterations $k \geq r_{p+1} + 2$ where r_{p+1} is the iteration of the $p + 1$ restart. If we apply Eq. (5.35) and Eq. (5.36) to Eq. (5.34) we obtain,

$$F(x_k) - F(x^*) \leq \left(\frac{2^{p+2}}{(k - r_{p+1}) \prod_{i=1}^{p+1} (r_i - r_{i-1} + 2)} \right)^2 \frac{L}{2} (x_0 - x^*)^2.$$

We are concerned with iterations $k \geq r_{p+1} + 2$ since those would be generated after the $p + 1$ restart. At $k = r_{p+1} + 2$ we have

$$\frac{2^{p+2}}{(k - r_{p+1}) \prod_{i=1}^{p+1} (r_i - r_{i-1} + 2)} = \frac{2^{p+2}}{(r_{p+1} + 2 - r_{p+1}) \prod_{i=1}^{p+1} (r_i - r_{i-1} + 2)} \quad (5.43a)$$

$$= \frac{2^{p+2}}{2(r_{p+1} - r_p + 2) \prod_{i=1}^p (r_i - r_{i-1} + 2)} \quad (5.43b)$$

$$= \frac{2^{p+1}}{(k - r_p) \prod_{i=1}^p (r_i - r_{i-1} + 2)}. \quad (5.43c)$$

Observe that this is exactly what we obtain when we have p restarts. Therefore, by the inductive hypothesis, we know that for $k = r_{p+1} + 2$ we satisfy the upper bound at $x_{r_{p+1}+2}$. For $k \geq r_{p+1} + 3$ we examine

$$\frac{2^{p+2}}{(k - r_{p+1}) \prod_{i=1}^{p+1} (r_i - r_{i-1} + 2)} - \frac{2^{p+1}}{(k - r_p) \prod_{i=1}^p (r_i - r_{i-1} + 2)}. \quad (5.44)$$

Proceeding similarly to the arguments in the base case (see Eq. (5.40) and Eq. (5.41)), we conclude that it is sufficient to examine the sign of $2(k - r_p) - (k - r_{p+1})(r_{p+1} - r_p + 2)$.

Writing $k = r_{p+1} + a$ where $a \geq 3$, we now examine

$$\begin{aligned} & 2(k - r_p) - (k - r_{p+1})(r_{p+1} - r_p + 2) \\ &= 2(r_{p+1} + a - r_p) - (r_{p+1} + a - r_{p+1})(r_{p+1} - r_p + 2) \end{aligned} \quad (5.45a)$$

$$= 2(r_{p+1} + a - r_p) - (a)(r_{p+1} - r_p + 2) \quad (5.45b)$$

$$= 2a + 2(r_{p+1} - r_p) - 2a - a(r_{p+1} - r_p) \quad (5.45c)$$

$$= (2 - a)(r_{p+1} - r_p) \leq 0. \quad (5.45d)$$

Hence we conclude for $k \geq r_{p+1} + 2$ we have

$$\frac{2^{p+2}}{(k - r_{p+1}) \prod_{i=1}^{p+1} (r_i - r_{i-1} + 2)} \leq \frac{2^{p+1}}{(k - r_p) \prod_{i=1}^p (r_i - r_{i-1} + 2)} \leq \frac{2}{k + 1},$$

where the second inequality follows from the inductive hypothesis. Altogether we have shown that $(\forall p \geq 2) (\forall k \in \{r_p + 2, \dots, r_{p+1} + 1\})$ we have

$$F(x_k) - F(x^*) \leq \left(\frac{2^{p+1}}{(k - r_p)(r_p - r_{p-1} + 2) \dots (r_2 - r_1 + 2)(r_1 + 2)} \right)^2 \frac{L}{2} (x_0 - x^*)^2 \quad (5.46a)$$

$$\leq \frac{2L}{(k + 1)^2} (x_0 - x^*)^2. \quad (5.46b)$$

The proof is complete. □

5.3 Application to Nearly Separable Functions

In this section, we will go over an application of the results above. One immediate application is if the objective function is separable. Recall the Hinder-Lubin example (3.22), the restarts performed poorly. In such a case, where the objective is separable then running in parallel n versions of Algorithm 5 results in good performance. Naturally, if we have knowledge that the objective is separable then the problem becomes much easier, and usually it is not the case. For this section, we will consider a version of the Hinder-Lubin example which is modified so that a non-separable term is introduced. Given a vector $a \in \mathbb{R}^n$ we define $F : \mathbb{R}^n \rightarrow \mathbb{R}$ as

$$F(x) = \sum_{i=1}^n ih_\delta(x_i) + \frac{\alpha}{2} \|x\|^2 + \gamma \left(\langle a, x \rangle + \sqrt{\langle a, x \rangle^2 + 1} \right). \quad (5.47)$$

Here h_δ is defined as in Eq. (3.22), with $\delta, \gamma, \alpha, \lambda \in \mathbb{R}$. Setting $f = \sum_{i=1}^n ih_\delta(x_i) + \frac{\alpha}{2}\|x\|^2 + \gamma \left(\langle a, x \rangle + \sqrt{\langle a, x \rangle^2 + 1} \right)$ we recover that this is $n + \alpha + \gamma\|a\|^2$ smooth, setting $g \equiv 0$ puts us in the composite model. For the numerical experiments, we set $m = 110, n = 100$, and $\alpha = \delta = \gamma = 10^{-4}$. The term γ is the weight on the non-separable part of the objective and it will vary. In the experiment, even though $F(x)$ is not separable, we will run Algorithm 5 along each coordinate showing the benefits as the weight on the separability term varies.

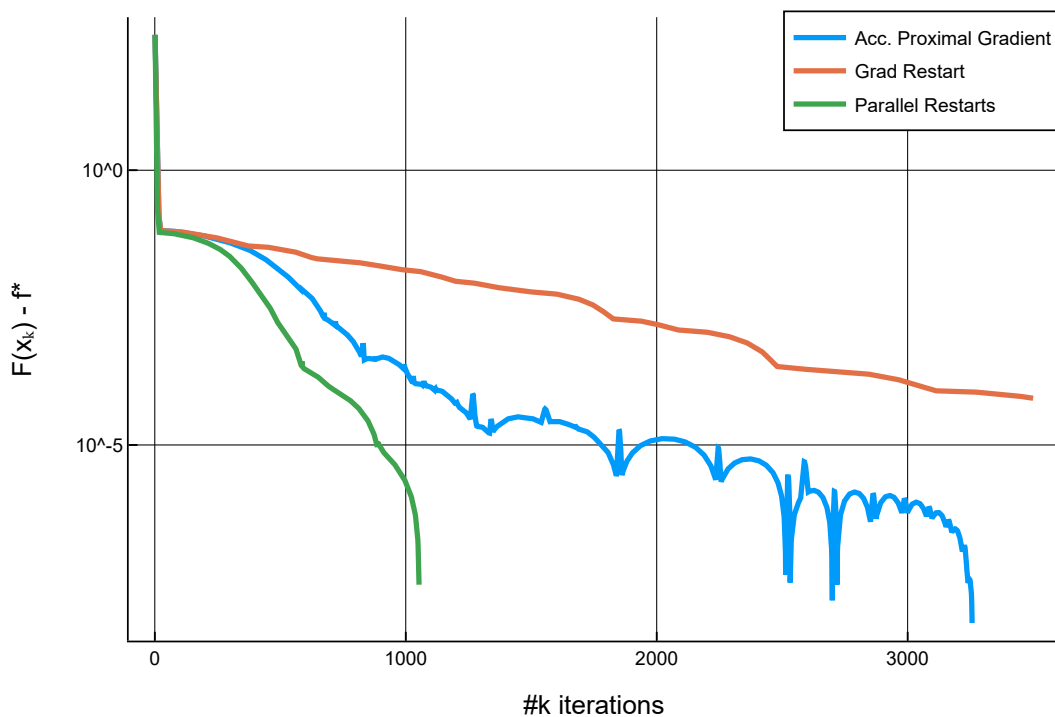


Figure 5.2: Running APG with restarts in parallel on each coordinate for a function which is nearly non-separable shows that restarting may still be beneficial.

In Fig. 5.2 we observe that the gradient-based restart scheme performs poorly. The function is non separable but as γ is small the contribution of the non-separable part is small. Running APG with restarts along each coordinate shows improved performance by having the function value decrease in fewer iterations.

Chapter 6

Applications to Alternating Projections

A well-known problem, that sees application in areas of mathematics, physics, engineering, and many other sciences is the convex feasibility problem [10]. Given C_1, \dots, C_n closed convex subsets of \mathbb{R}^n with $\bigcap_{i=1}^n C_i \neq \emptyset$. The problem is to find $x \in \mathbb{R}^n$ such that $x \in \bigcap_{i=1}^n C_i$. Alternating projections for solving this problem was studied by von Neumann for the case where the C_i are affine spaces [1].

6.1 Method of Alternating Projections

In this section we specialize to the case of two convex sets. For the rest of this section C and D are closed, convex, subsets of \mathbb{R}^n with $C \cap D \neq \emptyset$. Let P_C and P_D denote the projections onto C and D respectively. A classical algorithm to obtain $x \in C \cap D$ is the method of alternating projections (MAP).

Algorithm 6 MAP

Require: $x_0 \in \mathbb{R}^n$

for $k = 0, 1, 2 \dots$ **do**

$x_{k+1} \leftarrow P_D P_C x_k$

end for

In the case of two sets, the convex feasibility problem can be cast into the composite

model. Let $d_C : \mathbb{R}^n \rightarrow \mathbb{R}$ be defined by,

$$d_C(x) = \min_{c \in C} \|x - c\|, \quad (6.1)$$

be the distance function. This is related to the projection as, $d_C(x) = \|x - P_C x\|$. Now, consider $f(x) = \frac{1}{2}d_C^2(x)$, this function is differentiable with gradient,

$$\nabla f(x) = x - P_C(x). \quad (6.2)$$

Recall that P_C is firmly nonexpansive, and furthermore that means $\text{Id} - P_C$ is firm, therefore $\nabla f(x)$ is 1-smooth. Setting $g(x) = i_D(x)$ we can now see that the convex feasibility problem can be cast as the following convex optimization problem:

$$\min_{x \in \mathbb{R}^n} \frac{1}{2}d_C^2(x) + i_D(x). \quad (6.3)$$

As $\frac{1}{2}d_C^2(x) \geq 0$ and equality holds only if $x \in C$, and similarly $i_D(x) \geq 0$ and 0 only if $x \in D$, it is easy to see that a minimizer to the above problem will be a point $x \in C \cap D$. Since this problem is nonsmooth we use the proximal gradient method. Recalling that $\text{prox}_{i_D}(x) = P_D x$, we get

$$\text{prox}_g(x - \nabla f(x)) = P_D(x - (x - P_C x)) \quad (6.4)$$

$$= P_D P_C x. \quad (6.5)$$

Hence, the for the convex feasibility problem the proximal gradient method and MAP are equivalent. The advantage of using the composite formulation is that now acceleration can be applied through APG. With the use of the acceleration it becomes possible to apply the restarts of O'Donoghue and Candès.

6.2 Case of Two Hyperplanes

Let $c, d \in \mathbb{R}^n \setminus \{0\}$ be linearly independent, then throughout this section we set $C = \{x \in \mathbb{R}^n : \langle x, c \rangle = \beta_1\}$ and $D = \{x \in \mathbb{R}^n : \langle x, d \rangle = \beta_2\}$ to be two hyperplanes in \mathbb{R}^n . The projection onto the hyperplanes is given by,

$$P_C(x) = x - \frac{\langle x, c \rangle - \beta_1}{\|c\|^2} c, \quad P_D(x) = x - \frac{\langle x, d \rangle - \beta_2}{\|d\|^2} d. \quad (6.6)$$

Note that a closed form solution of convex feasibility problem for two hyperlanes exists [12] but studying this case provides insights into the behaviour of the restarting algorithm. We have the following theorem for the iterates of Algorithm 1 applied to $f = \frac{1}{2}d_C^2$ and $g = i_D$.

Theorem 6.1. *Let $x_0 \in \mathbb{R}^n$ and let $(x_k)_{k \in \mathbb{N}}$, $(y_k)_{k \in \mathbb{N}}$ be produced by APG for the convex feasibility problem. Then for $k \geq 1$ the iterates x_k are on the line given by $x_1 + \text{span}(P_D c - \frac{\beta_2}{\|d\|^2} d)$.*

Proof. Take $x_0 \in \mathbb{R}^n$, and set $y_0 = x_0$, then,

$$x_1 = P_D P_C x_0. \quad (6.7)$$

We will proceed by strong induction. For the base cases for $k = 2$ and $k = 3$ we have,

$$x_2 = P_D P_C y_1. \quad (6.8)$$

By Algorithm 1 we have,

$$y_1 = x_1 + \frac{t_0 - 1}{t_1} (x_1 - x_0), \quad (6.9)$$

since $t_0 = 1$ we have $y_1 = x_1$. We then proceed with applying the projections,

$$x_2 = P_D P_C x_1 = P_C x_1 - \frac{\langle P_C x_1, d \rangle - \beta_2}{\|d\|^2} d \quad (6.10)$$

$$= \left(x_1 - \frac{\langle x_1, c \rangle - \beta_1}{\|c\|^2} c \right) - \frac{\langle P_C x_1, d \rangle - \beta_2}{\|d\|^2} d \quad (6.11)$$

$$= x_1 - \frac{\langle x_1, c \rangle - \beta_1}{\|c\|^2} c - \frac{\langle x_1 - \frac{\langle x_1, c \rangle - \beta_1}{\|c\|^2} c, d \rangle - \beta_2}{\|d\|^2} d \quad (6.12)$$

$$= x_1 - \frac{\langle x_1, c \rangle - \beta_1}{\|c\|^2} c - \frac{\langle x_1, d \rangle - \frac{\langle x_1, c \rangle - \beta_1}{\|c\|^2} \langle c, d \rangle - \beta_2}{\|d\|^2} d \quad (6.13)$$

$$= x_1 - \frac{\langle x_1, c \rangle - \beta_1}{\|c\|^2} c - \frac{\beta_2 - \frac{\langle x_1, c \rangle - \beta_1}{\|c\|^2} \langle c, d \rangle - \beta_2}{\|d\|^2} d \quad (6.14)$$

$$= x_1 - \frac{\langle x_1, c \rangle - \beta_1}{\|c\|^2} \left(c - \frac{\langle c, d \rangle}{\|d\|^2} d \right) \quad (6.15)$$

$$= x_1 - \frac{\langle x_1, c \rangle - \beta_1}{\|c\|^2} \left(c - \frac{\langle c, d \rangle - \beta_2}{\|d\|^2} d - \frac{\beta_2}{\|d\|^2} d \right) \quad (6.16)$$

$$= x_1 - \frac{\langle x_1, c \rangle - \beta_1}{\|c\|^2} \left(P_D c - \frac{\beta_2}{\|d\|^2} d \right). \quad (6.17)$$

We can see that $x_2 \in x_1 + \text{span}\{P_D c - \frac{\beta_2}{\|d\|^2} d\}$. We will denote $-\frac{\langle x_1, c \rangle - \beta_1}{\|c\|^2}$ by α_2 . Before showing the other base case we rewrite the update for in general for x_{k+1} ,

$$x_{k+1} = P_D P_C y_k \quad (6.18)$$

$$= P_C y_k - \frac{\langle P_C y_k, d \rangle - \beta_2}{\|d\|^2} d \quad (6.19)$$

$$= \left(y_k - \frac{\langle y_k, c \rangle - \beta_1}{\|c\|^2} c \right) - \frac{\langle P_C y_k, d \rangle - \beta_2}{\|d\|^2} d \quad (6.20)$$

$$= y_k - \frac{\langle y_k, c \rangle - \beta_1}{\|c\|^2} c - \frac{\langle P_C y_k, d \rangle - \beta_2}{\|d\|^2} d \quad (6.21)$$

$$= y_k - \frac{\langle y_k, c \rangle - \beta_1}{\|c\|^2} c - \frac{\langle y_k - \frac{\langle y_k, c \rangle - \beta_1}{\|c\|^2} c, d \rangle - \beta_2}{\|d\|^2} d \quad (6.22)$$

$$= y_k - \frac{\langle y_k, c \rangle - \beta_1}{\|c\|^2} \left(c - \frac{\langle c, d \rangle}{\|d\|^2} d \right) \quad (6.23)$$

$$= y_k - \frac{\langle y_k, c \rangle - \beta_1}{\|c\|^2} \left(P_D c - \frac{\beta_2}{\|d\|^2} d \right). \quad (6.24)$$

Using the above formulationn the proof for the $k = 3$ case works similarly; for x_3 the above update is written as,

$$x_3 = y_2 - \frac{\langle y_2, c \rangle - \beta_1}{\|c\|^2} \left(P_D c - \frac{\beta_2}{\|d\|^2} d \right). \quad (6.25)$$

Now we can expand y_2 ,

$$y_2 = x_2 + \frac{t_1 - 1}{t_2} (x_2 - x_1) \quad (6.26)$$

$$= \left(x_1 + \alpha_2 \left(P_D c - \frac{\beta_2}{\|d\|^2} d \right) \right) + \alpha_2 \frac{t_1 - 1}{t_2} \left(P_D c - \frac{\beta_2}{\|d\|^2} d \right). \quad (6.27)$$

This gives us,

$$y_2 = x_1 + \left(\alpha_2 + \alpha_2 \frac{t_1 - 1}{t_2} \right) \left(P_D c - \frac{\beta_2}{\|d\|^2} d \right). \quad (6.28)$$

Plugging this into the equation for x_3 we now have,

$$x_3 = x_1 + \left(\alpha_2 - \frac{\langle y_2, c \rangle - \beta_1}{\|c\|^2} + \alpha_2 \frac{t_1 - 1}{t_2} \right) \left(P_D c - \frac{\beta_2}{\|d\|^2} d \right), \quad (6.29)$$

and we observe we have x_3 in the form $x_1 + \alpha_3 \left(P_D c - \frac{\beta_2}{\|d\|^2} d \right)$.

We can now proceed with the proof using strong induction on k . Suppose that for $j \in \{1, 2, \dots, k\}$ we can write $x_j = x_1 + \alpha_j \left(P_D c - \frac{\beta_2}{\|d\|^2} d \right)$. We wish to show that we can do this for x_{k+1} . We know from above that,

$$x_{k+1} = y_k - \frac{\langle y_k, c \rangle - \beta_1}{\|c\|^2} \left(P_D c - \frac{\beta_2}{\|d\|^2} d \right). \quad (6.30)$$

We proceed by applying the definition of y_k ,

$$y_k = x_k + \frac{t_{k-1} - 1}{t_k} (x_k - x_{k-1}),$$

and applying the induction hypothesis

$$y_k = \left(x_1 + \alpha_k \left(P_D c - \frac{\beta_2}{\|d\|^2} d \right) \right) + \frac{t_{k-1} - 1}{t_k} \left(x_1 + \alpha_k \left(P_D c - \frac{\beta_2}{\|d\|^2} d \right) - x_1 - \alpha_{k-1} \left(P_D c - \frac{\beta_2}{\|d\|^2} d \right) \right) \quad (6.31)$$

Putting like terms together we get,

$$y_k = x_1 + \left(\alpha_k + \frac{t_{k-1} - 1}{t_k} (\alpha_k - \alpha_{k-1}) \right) \left(P_D c - \frac{\beta_2}{\|d\|^2} d \right). \quad (6.32)$$

Applying this to the equation for x_{k+1} above we complete the proof as we have,

$$x_{k+1} = x_1 + \left(\alpha_k - \frac{\langle y_k, c \rangle - \beta_1}{\|c\|^2} + \left(\frac{t_{k-1} - 1}{t_k} (\alpha_k - \alpha_{k-1}) \right) \right) \left(P_D c - \frac{\beta_2}{\|d\|^2} d \right). \quad (6.33)$$

□

Corollary 6.2. *An optimal solution of the convex feasibility problem for the case of two hyperplanes is given by,*

$$x^* = x_1 + \alpha^* \left(P_D c - \frac{\beta_2}{\|d\|^2} d \right), \quad (6.34)$$

where $x_1 = P_D P_C x_0$, and $\alpha^* = \frac{\beta_1 - \langle x_1, c \rangle}{\langle P_D c - \frac{\beta_2}{\|d\|^2} d, c \rangle}$

Proof. Set x^* and α^* as in the statement above. To show that x^* is optimal we must show that we do in fact have $x^* \in C \cap D$. To that end we compute,

$$\langle x^*, d \rangle = \langle x_1 + \alpha^* \left(P_D c - \frac{\beta_2}{\|d\|^2} d \right), d \rangle \quad (6.35)$$

$$= \beta_2 + \alpha^* \left(\langle P_D c, d \rangle - \frac{\beta_2}{\|d\|^2} \langle d, d \rangle \right) \quad (6.36)$$

$$= \beta_2 + \alpha^* (\beta_2 - \beta_2) \quad (6.37)$$

$$= \beta_2. \quad (6.38)$$

We have shown $x^* \in D$, now we proceed to show $x^* \in C$.

$$\langle x^*, c \rangle = \langle x_1 + \alpha^*(P_D c - \frac{\beta_2}{\|d\|^2}d), c \rangle \quad (6.39)$$

$$= \langle x_1, c \rangle + \frac{\beta_1 - \langle x_1, c \rangle}{\langle P_D c - \frac{\beta_2}{\|d\|^2}d, c \rangle} \langle (P_D c - \frac{\beta_2}{\|d\|^2}d), c \rangle \quad (6.40)$$

$$= \langle x_1, c \rangle + \beta_1 - \langle x_1, c \rangle \quad (6.41)$$

$$= \beta_1. \quad (6.42)$$

Hence we have that $x^* \in C$ and so $x^* \in C \cap D$ as desired. \square

Corollary 6.3. *Let C, D be two hyperplanes, let $x_0 \in \mathbb{R}^n$. Set $\alpha^* = \frac{\beta_1 - \langle x_1, c \rangle}{\langle P_D c - \frac{\beta_2}{\|d\|^2}d, c \rangle}$, then if $(x_k)_{k \in \mathbb{N}}$ are given by Algorithm 1 we have that,*

$$x_k \rightarrow x^* = x_1 + \alpha^*(P_D c - \frac{\beta_2}{\|d\|^2}d). \quad (6.43)$$

Proof. By the convergence rate of APG we know that $F(x_k) \rightarrow F^* = 0$. Since by Theorem 6.1 we have that $x_k = x_1 + \alpha_k(P_D c - \frac{\beta_2}{\|d\|^2}d)$ also converges to some point, in particular this means that $\lim_{k \rightarrow \infty} \alpha_k$ exists as $P_D c - \frac{\beta_2}{\|d\|^2}d \neq 0$.

From Theorem 6.1 we know that

$$\alpha_{k+1} = \left(\alpha_k - \frac{\langle y_k, c \rangle - \beta_1}{\|c\|^2} + \left(\frac{t_{k-1} - 1}{t_k} (\alpha_k - \alpha_{k-1}) \right) \right) \quad (6.44)$$

$$\alpha_{k+1} = \alpha_k - \frac{\langle x_1 + (\alpha_k + \frac{t_{k-1} - 1}{t_{k+1}}(\alpha_k - \alpha_{k-1}))(P_D c - \frac{\beta_2}{\|d\|^2}d), c \rangle - \beta_1}{\|c\|^2} + \left(\frac{t_{k-1} - 1}{t_k} (\alpha_k - \alpha_{k-1}) \right). \quad (6.45)$$

Taking the limit as $k \rightarrow \infty$ we get that,

$$0 = - \frac{\langle x_1, c \rangle + \alpha^* \langle P_D c - \frac{\beta_2}{\|d\|^2}d, c \rangle - \beta_1}{\|c\|^2} \quad (6.46)$$

$$0 = -\langle x_1, c \rangle - \alpha^* \langle P_D c - \frac{\beta_2}{\|d\|^2}d, c \rangle + \beta_1 \quad (6.47)$$

$$\langle x_1, c \rangle - \beta_1 = -\alpha^* \langle P_D c - \frac{\beta_2}{\|d\|^2}d, c \rangle \quad (6.48)$$

$$\frac{\beta_1 - \langle x_1, c \rangle}{\langle P_D c - \frac{\beta_2}{\|d\|^2}d, c \rangle} = \alpha^*. \quad (6.49)$$

By the previous corollary we know, that for this value of α^* , the point $x_1 + \alpha^*(P_D c - \frac{\beta_2}{\|d\|^2}d)$ is in both C and D and therefore, we get that x_k converges to an optimal solution. \square

In the analysis of the one dimensional case it was shown that the gradient based restarts correspond to the iterates given by APG crossing the minimizer. In higher dimensions there is no sense of crossing the minimizer as there is no ordering. For the case of two hyperplanes, we notice that all iterates x_k , and y_k for $k \geq 1$ will be on the hyperplane D , and furthermore we notice that the other hyperplane C splits the space into two halfspaces. We wish to show a correspondence between the gradient based restarts and the iterates of APG crossing from one halfspace into the other.

Proposition 6.4. *Let $C^- = \{x : \langle x, c \rangle < \beta_1\}$ and let $C^+ = \{x : \langle x, c \rangle > \beta_1\}$. Suppose that $(x_k)_{k \in \mathbb{N}}$ and $(y_k)_{k \in \mathbb{N}}$ are given by Algorithm 1. Then all x_k and y_k are on the hyperplane given by D . Then,*

(i) *If $y_k \in C^-$ then $x_{k+1} \in C^-$*

(ii) *If $y_k \in C^+$ then $x_{k+1} \in C^+$.*

Proof. We prove (i) as (ii) is similar. Suppose that $y_k \in C^-$. Let, $x_{k+1} = P_D P_C y_k = y_k + \frac{\langle y_k, c \rangle}{\|c\|^2} P_D c$. Then,

$$\langle c, x_{k+1} \rangle = \langle c, P_C y_k - \frac{\langle P_C y_k, d \rangle - \beta_2}{\|d\|^2} d \rangle \quad (6.50)$$

$$= \beta_1 - \langle c, \frac{\langle P_C y_k, d \rangle - \beta_2}{\|d\|^2} d \rangle \quad (6.51)$$

$$= \beta_1 - \langle c, \frac{\langle y_k - \frac{\langle c, y_k \rangle - \beta_1}{\|c\|^2} c, d \rangle - \beta_2}{\|d\|^2} d \rangle \quad (6.52)$$

$$= \beta_1 + \frac{(\langle c, y_k \rangle - \beta_1) \langle c, d \rangle^2}{\|c\|^2 \|d\|^2} \quad (6.53)$$

$$< \beta_1. \quad (6.54)$$

It was shown that $x_{k+1} \in C^-$ as desired. \square

With the above proposition in hand, we can now show that the gradient restart occurs if y_k crosses from one halfspace to the other.

Theorem 6.5. *Let C, D be hyperplanes as described earlier. Let $C^- = \{x : \langle x, c \rangle < \beta_1\}$ and let $C^+ = \{x : \langle x, c \rangle > \beta_1\}$ be halfspaces. Further, let $(x_k)_{k \in \mathbb{N}}$ and $(y_k)_{k \in \mathbb{N}}$ are given by Algorithm 1. Suppose that $x_k \in C^-$ and $y_k \in C^+$. Then the gradient restart condition,*

$$\langle y_k - x_{k+1}, x_{k+1} - x_k \rangle > 0,$$

is satisfied. Likewise, if $x_k \in C^+$ and $y_k \in C^-$.

Proof. Suppose that $x_k \in C^-$ and $y_k \in C^+$. By the previous proposition we know that $x_{k+1} \in C^+$. We directly check the gradient condition,

$$\langle y_k - x_{k+1}, x_{k+1} - x_k \rangle = \langle y_k - P_C y_k + \frac{\langle P_C y_k, d \rangle - \beta_2}{\|d\|^2} d, x_{k+1} - x_k \rangle \quad (6.55)$$

$$= \langle y_k - (y_k - \frac{\langle y_k, c \rangle - \beta_1}{\|c\|^2} c) + \frac{\langle P_C y_k, d \rangle - \beta_2}{\|d\|^2} d, x_{k+1} - x_k \rangle \quad (6.56)$$

$$= \frac{\langle y_k, c \rangle - \beta_1}{\|c\|^2} \langle c, x_{k+1} - x_k \rangle. \quad (6.57)$$

Now $\langle c, x_{k+1} \rangle > \beta_1$ and $\langle c, x_k \rangle < \beta_1$ so $\langle c, x_{k+1} - x_k \rangle > 0$, and since $\langle c, y_k \rangle > \beta_1$ we conclude the restart condition is satisfied.

On the other hand, if $x_k \in C^+$ and $y_k \in C^-$ then we have, $\langle c, x_{k+1} \rangle < 0$ and $\langle c, x_k \rangle > 0$ so $\langle c, x_{k+1} - x_k \rangle < 0$, now $\langle c, y_k \rangle < 0$ so in total we have the product is positive and thus the restart condition is satisfied. \square

We wish to show an equivalence between restarting, and the y_k 's crossing between halfspaces.

Theorem 6.6. *Let C, D be as in the problem setting. Let $(x_k)_{k \in \mathbb{N}}$ and $(y_k)_{k \in \mathbb{N}}$ be given by Algorithm 1. Suppose that $x_k \in C^+$, and the gradient restart condition is satisfied,*

$$\langle y_k - x_{k+1}, x_{k+1} - x_k \rangle > 0.$$

Then we have that $y_k, x_{k+1} \in C^-$.

Proof. Suppose for the sake of contradiction that $y_k \in C^+$, hence so is x_{k+1} . In Theorem 6.5 we saw that the restart condition is equivalent to

$$\frac{\langle y_k, c \rangle - \beta_1}{\|c\|^2} \langle c, x_{k+1} - x_k \rangle > 0.$$

We assumed that $\langle c, y_k \rangle > \beta_1$ so for the restart to hold it must be that $\langle c, x_{k+1} - x_k \rangle > 0$ as well. Applying Theorem 6.1 we have

$$\langle c, x_{k+1} - x_k \rangle = \langle c, (\alpha_{k+1} - \alpha_k)(P_D c - \frac{\beta_2}{\|d\|^2} d) \rangle \quad (6.58)$$

$$= (\alpha_{k+1} - \alpha_k) \langle c, c - \frac{\langle c, d \rangle - \beta_2}{\|d\|^2} d - \frac{\beta_2}{\|d\|^2} d \rangle \quad (6.59)$$

$$= (\alpha_{k+1} - \alpha_k) \langle c, c - \frac{\langle c, d \rangle}{\|d\|^2} d \rangle \quad (6.60)$$

$$= (\alpha_{k+1} - \alpha_k) \left(\|c\|^2 - \frac{\langle c, d \rangle^2}{\|d\|^2} \right) \quad (6.61)$$

$$= (\alpha_{k+1} - \alpha_k) (\|c\|^2 - \|c\|^2 \cos^2 \varphi). \quad (6.62)$$

As the problem is feasible we have that $\cos^2 \varphi \in [0, 1)$, hence $\|c\|^2 - \|c\|^2 \cos^2 \varphi > 0$. Since x_{k+1} is a proximal gradient step from y_k , and both are in C^+ it follows that x_{k+1} is further along the line $x_1 + \alpha(P_D c - \frac{\beta}{\|d\|^2} d)$ than x_k . Furthermore since x_k, x_{k+1} are in C^+ it means that the α_k are decreasing towards α^* and hence $\alpha_{k+1} - \alpha_k < 0$. This means that in the above we have $\langle c, x_{k+1} - x_k \rangle < 0$ and a restart does not occur. This is a contradiction, as we assumed that the restart condition is satisfied. Therefore we must have that $y_k, x_{k+1} \in C^-$. If we assumed that initially $x_k \in C^-$ and the gradient condition is satisfied we would similarly have that $y_k, x_{k+1} \in C^+$. \square

The above two theorems imply that the gradient restart condition is satisfied if and only if the iterates cross the hyperplane defined by $\langle c, x \rangle = \beta_1$. To analyze the gradient restart we proceed similar to the one dimensional case from earlier.

Recall, we defined the vector $z_k = (1 - t_{k-1})x_{k-1} + t_{k-1}x_k$. By previous analysis we know that the optimal point x^* is $x_1 + \alpha^*(P_D c - \frac{\beta_2}{\|d\|^2} d)$. We have the following lemma which will give a bound on distance between z_{k+1} and x^* .

Lemma 6.7. *Suppose that $(x_k)_{k \in \mathbb{N}}$ are given by Algorithm 1 applied to the case of the convex feasibility problem for two hyperplanes. Suppose that at iteration $k = r$ the restart condition*

$$\langle y_r - x_{r+1}, x_{r+1} - x_r \rangle > 0$$

is satisfied. Define, $z_k = (1 - t_{k-1})x_{k-1} + t_{k-1}x_k$ for all $k \geq 1$ and recall that $x^ = x_1 + \alpha^*(P_D c - \frac{\beta_2}{\|d\|^2} d)$. Then,*

$$\|z_{r+1} - x^*\| \geq t_r \|x_{r+1} - x^*\|. \quad (6.63)$$

Proof. By Theorem 6.1 we can write,

$$x_r = x_1 + \alpha_r(P_D c - \frac{\beta}{\|d\|^2}d), \quad (6.64)$$

and,

$$x_{r+1} = x_1 + \alpha_{r+1}(P_D c - \frac{\beta}{\|d\|^2}d). \quad (6.65)$$

We now proceed as follows,

$$\|z_{r+1} - x^*\| = \|(1 - t_r)x_r + t_r x_{r+1} - x^*\| \quad (6.66)$$

$$= \|(1 - t_r)(x_r - x^*) + t_r(x_{r+1} - x^*)\| \quad (6.67)$$

$$= \|(1 - t_r)(\alpha_r - \alpha^*)(P_D c - \frac{\beta_2}{\|d\|^2}d) + t_r(\alpha_{r+1} - \alpha^*)(P_D c - \frac{\beta_2}{\|d\|^2}d)\| \quad (6.68)$$

$$= |(1 - t_r)(\alpha_r - \alpha^*) + t_r(\alpha_{r+1} - \alpha^*)| \|P_D c - \frac{\beta_2}{\|d\|^2}d\|. \quad (6.69)$$

Now we know since the restart condition was met we have say $\alpha_r < \alpha^*$ and $\alpha_{r+1} > \alpha^*$. In this case the absolute value term becomes,

$$|(1 - t_r)(\alpha_r - \alpha^*) + t_r(\alpha_{r+1} - \alpha^*)| = (t_r - 1)|\alpha_r - \alpha^*| + t_r|\alpha_{r+1} - \alpha^*| \quad (6.70)$$

$$\geq t_r|\alpha_{r+1} - \alpha^*|. \quad (6.71)$$

Applying this to inequality we know obtain,

$$\|z_{r+1} - x^*\| \geq t_r|\alpha_{r+1} - \alpha^*| \|P_D c - \frac{\beta_2}{\|d\|^2}d\| \quad (6.72)$$

$$= t_r\|x_{r+1} - x^*\| \quad (6.73)$$

as desired. If instead we have $\alpha_{r+1} < \alpha^* < \alpha_r$ then the proof is similar. \square

Chapter 7

Conclusion and Future Work

First order algorithms have grown in popularity as the size of modern problems increases, such as in machine learning and data science applications. Heuristics which accelerate first order methods further, such as the adaptive restarting techniques of O’Donoghue and Candès are very useful. In this thesis we have studied the gradient restart condition for accelerated proximal gradient (APG) applied to composite convex optimization models. By looking at special cases, such as the one dimensional case, and projection onto the intersection of two hyperplanes we were able to prove that the restarts do improve upon the convergence rate. In APG, the function values of the iterates are not monotonic, this is due to the added momentum terms which cause the iterates to overshoot the minimizer. In the cases for the restarts studied in the thesis we showed that the gradient condition corresponds exactly to the iterates overshooting the minimizer

The application of the adaptive restarts remains a heuristic for more general cases. A further area of development motivated by the cases we studied is the investigating the relationship between Fejér monotonicity of the iterates and the restart condition. If at any iteration k if $\|x_{k+1} - x^*\| \leq \|x_k - x^*\|$ does not hold, then the monotonicity condition is violated. In numerical experiment, using monotonicity as a restart condition performed well, and in the case of the hyperplanes corresponded to the gradient based restarts. While in practice it is not a feasible condition to test, it could provide further insight into why heuristic methods work. Another area of further study comes from alternating projections. One question is how does the geometry of the sets affect the restarts? For the two hyperplanes we had nice formulas for the projections, therefore it is also interesting to ask whether similar results could be shown if just one of the sets is a hyperplane, or if both projections have a closed form. Recently, restarting using more than just the information

at the last iterate has been looked at for primal-dual hybrid gradient (PDHG) [25]. Using information from previous iterates to create a new adaptive restart condition is a possible direction for further research.

References

- [1] On Rings of Operators. Reduction Theory on JSTOR, April 1949. [Online; accessed 16. May 2024].
- [2] Teodoro Alamo, Pablo Krupa, and Daniel Limon. Gradient Based Restart FISTA. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 11–13. IEEE.
- [3] Teodoro Alamo, Daniel Limon, and Pablo Krupa. Restart FISTA with Global Linear Convergence. In *2019 18th European Control Conference (ECC)*, pages 25–28. IEEE.
- [4] David Applegate, Oliver Hinder, Haihao Lu, and Miles Lubin. Faster first-order primal-dual methods for linear programming using restarts and sharpness. *Math. Program.*, 201(1):133–184, September 2023.
- [5] Jean-François Aujol, Luca Calatroni, Charles Dossal, Hippolyte Labarrière, and Aude Rondepierre. Parameter-free FISTA by adaptive restart and backtracking. Arxiv preprint 2307.14323, 2023.
- [6] Akshay Badola, Vineet Padmanabhan Nair, and Rajendra Prasad Lal. An Analysis of Regularization Methods in Deep Neural Networks. In *2020 IEEE 17th India Council International Conference (INDICON)*, pages 10–13. IEEE.
- [7] J. Baillon and G. Haddad. Quelques propriétés des opérateurs angle-bornés et cycliquement monotones. *Israel J. Math.*, 1977.
- [8] JONATHAN BARZILAI and JONATHAN M. BORWEIN. Two-Point Step Size Gradient Methods. *IMA Journal of Numerical Analysis*, 8(1):141–148, 01 1988.
- [9] H. H. Bauschke and J. M. Borwein. On the convergence of von Neumann’s alternating projection algorithm for two sets. *Set-Valued Anal.*, 1(2):185–212, June 1993.

- [10] Heinz H. Bauschke and Jonathan M. Borwein. On projection algorithms for solving convex feasibility problems. *SIAM Review*, 38(3):367–426, 1996.
- [11] Heinz H. Bauschke and Patrick L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer International Publishing, Cham, Switzerland, 2017.
- [12] Heinz H. Bauschke, Dayou Mao, and Walaa M. Moursi. How to project onto the intersection of a closed affine subspace and a hyperplane, 2022.
- [13] Heinz H. Bauschke and Walaa M. Moursi. *An Introduction to Convexity, Optimization, and Algorithms*. 2023.
- [14] Amir Beck. *First-Order Methods in Optimization*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2017.
- [15] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.
- [16] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. 2011.
- [17] Emmanuel J. Candès, Justin K. Romberg, and Terence Tao. Stable signal recovery from incomplete and inaccurate measurements. *Commun. Pure Appl. Math.*, 59(8):1207–1223, August 2006.
- [18] Patrick L. Combettes and Valérie R. Wajs. Signal Recovery by Proximal Forward-Backward Splitting. *Multiscale Model. Simul.*, July 2006.
- [19] Olivier Fercoq and Zheng Qu. Adaptive restart of accelerated gradient methods under local quadratic growth condition. *IMA Journal of Numerical Analysis*, 39(4):2069–2095, 03 2019.
- [20] Subham Ghosh and Yoram Rudy. Application of L1-Norm Regularization to Epicardial Potential Solution of the Inverse Electrocardiography Problem. *Ann. Biomed. Eng.*, 37(5):902, May 2009.
- [21] Pontus Giselsson and Stephen Boyd. Monotonicity and restart in fast gradient methods. In *53rd IEEE Conference on Decision and Control*, pages 5058–5063. IEEE, 2014.
- [22] Oliver Hinder and Miles Lubin. A generic adaptive restart scheme with applications to saddle point algorithms, 2020.

- [23] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017.
- [24] Ilya Loshchilov, Marc Schoenauer, and Michèle Sebag. Alternative Restart Strategies for CMA-ES. In *Parallel Problem Solving from Nature - PPSN XII*, pages 296–305. Springer, Berlin, Germany, 2012.
- [25] Haihao Lu and Jinwen Yang. Restarted halpern pdhg for linear programming, 2024.
- [26] J. Moreau. Fonctions convexes duales et points proximaux dans un espace hilbertien. *Comptes rendus hebdomadaires des seances de l’Academie des sciences*, 1962.
- [27] I. Necoara, Yu. Nesterov, and F. Glineur. Linear convergence of first order methods for non-strongly convex optimization. *Math. Program.*, 175(1):69–107, May 2019.
- [28] Yu. Nesterov. Gradient methods for minimizing composite objective function. *LIDAM Discussion Papers CORE*, September 2007.
- [29] Yu. Nesterov. Gradient methods for minimizing composite functions. *Math. Program.*, 140(1):125–161, August 2013.
- [30] Yurii Nesterov. A method for solving the convex programming problem with convergence rate $O(1/k^2)$. *Proceedings of the USSR Academy of Sciences*, 269:543–547, 1983.
- [31] Yurii Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer Publishing Company, Incorporated, 1 edition, 2014.
- [32] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, NY, USA, 2e edition, 2006.
- [33] Brendan O’Donoghue and Emmanuel Candès. Adaptive Restart for Accelerated Gradient Schemes. *Found. Comput. Math.*, 15(3):715–732, June 2015.
- [34] R. Tyrrell Rockafellar. *Convex analysis*. Princeton Mathematical Series. Princeton University Press, Princeton, N. J., 1970.
- [35] Mark Schmidt, Nicolas Le Roux, and Francis Bach. Convergence rates of inexact proximal-gradient methods for convex optimization. In *NIPS’11: Proceedings of the 24th International Conference on Neural Information Processing Systems*, pages 1458–1466. Curran Associates Inc., Red Hook, NY, USA, December 2011.

- [36] Weijie Su, Stephen Boyd, and Emmanuel Candes. A differential equation for modeling nesterov’s accelerated gradient method: Theory and insights. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.
- [37] Bao Wang, Tan Nguyen, Tao Sun, Andrea L. Bertozzi, Richard G. Baraniuk, and Stanley J. Osher. Scheduled Restart Momentum for Accelerated Stochastic Gradient Descent. *SIAM J. Imag. Sci.*, May 2022.
- [38] Danqing Zhou, Shiqian Ma, and Junfeng Yang. Adabb: Adaptive barzilai-borwein method for convex optimization, 2024.
- [39] Xingyu Zhou. On the Fenchel Duality between Strong Convexity and Lipschitz Continuous Gradient. *arXiv*, March 2018.