Robust Regression via IRLS

A thesis submitted in fulfillment of the requirements

for the degree of Master of Technology

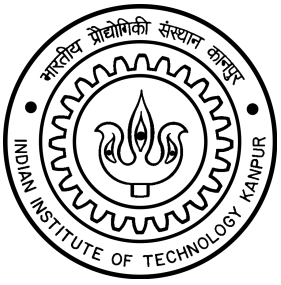
by

Govind Gopakumar

16111009

under the guidance of

Purushottam Kar and Prateek Jain



to the

Department of Computer Science and Engineering

Indian Institute of Technology Kanpur

May 2018

Page intentionally left blank

Statement of Thesis Preparation

I, **Govind Gopakumar**, declare that this thesis titled, **"Robust Regression via IRLS"**, hereby submitted in partial fulfillment of the requirements for the degree of **Master of Technology** and the work contained herein are my own. I further confirm that:

1. The "Thesis Guide" was referred to for preparing the thesis.
2. Specifications regarding thesis format have been closely followed.
3. The contents of the thesis have been organized based on the guidelines.
4. The thesis has been prepared without resorting to plagiarism.
5. All sources used have been cited appropriately.
6. The thesis has not been submitted elsewhere for a degree.

Name: Govind Gopakumar

Roll No.: 16111009

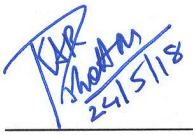
Department of Computer Science and Engineering

May 2018

Page intentionally left blank

Certificate

It is certified that the work contained in the thesis titled **"Robust Regression via IRLS"** has been carried out under my supervision by **Govind Gopakumar** and that this work has not been submitted elsewhere for a degree.



Purushottam Kar

Assistant Professor

Department of Computer Science and Engineering

Indian Institute of Technology Kanpur

Kanpur, 208016.

May 2018

Page intentionally left blank

Abstract

The primitive of regression, especially linear regression, is a useful tool with a wide variety of applications in machine learning, statistics, economics and finance, allowing us to model a real-valued output $y \in \mathbb{R}$ (called the *response*) as a linear combination of real-valued inputs $\mathbf{x} \in \mathbb{R}^d$ (called the *covariates* or *features*). Given a set of n covariates $X = [\mathbf{x}_1, \ldots, \mathbf{x}_n]^\top \in \mathbb{R}^{n \times d}$ and responses $\mathbf{y} = [y_1, \ldots, y_n] \in \mathbb{R}^n$, our aim is to discover a model $\mathbf{w} \in \mathbb{R}^d$ such that $\mathbf{y} \approx X\mathbf{w}$.

Traditional approaches to solving this problem include techniques such as OLS (Ordinary Least Squares), its regularized cousins such as ridge regression and LASSO, which often assume that the response is generated for each of the n data points as $y_i = \langle \mathbf{w}^*, \mathbf{x}_i \rangle + \epsilon_i$ where $\mathbf{w}^*$ denotes an unknown *gold* model and ϵ_i denotes noise. Assuming different noise distributions can give rise to different estimators for $\mathbf{w}^*$. In particular, the OLS technique assumes that noise is independent and identically distributed normally.

Such assumptions need not hold for data that arise in real applications, and there is a need for algorithms that can tolerate structured noise distributions. In this thesis, we focus on noise models where a possibly malicious adversary injects noise, but in a sparse manner. This falls within the general ambit of *robust* regression analysis. In particular, we investigate the properties of the popular Iteratively Reweighted Least Squares (IRLS) algorithm in the robust regression setting. IRLS is a popular choice of solver for the class of generalized linear models, sparse recovery problems, and is a candidate for the robust regression problem as well.

Our results reveal critical flaws in the vanilla IRLS algorithm and demonstrate a class of counterexamples where IRLS cannot guarantee global recovery of the true underlying model. We complement these negative results with local convergence results where we show that IRLS when modified slightly, does indeed converge to the true model if initialized carefully. We believe that this modification can be extended to ensure global recovery without the need for careful initialization.

Page intentionally left blank

Acknowldgements

I will always be grateful to my guides - Professor Purushottam Kar, and Dr. Prateek Jain for taking on the job of advising me for my thesis. Working with the two of them has been an incredible experience. Every Skype call left me feeling uneducated and exposed me to new parts of linear algebra/probability/statistics and mathematics in general. Their enthusiasm has been unwavering, and their support while I struggled through understanding our discussions was immense. I was continuously surprised by how often one of the two would bring up approaches and insights that would never have occurred to me, a mere masters student exploring research for the first time. I'm grateful to have been a small part of such an amazing research program.

The work I did as part of my thesis would not have seen the light of day if not for Prof. Kar's enthusiasm and knowledge. His knowledge of all of optimization and machine learning always astounds me, whenever I feel like I bring something new to our meetings, he would invariably end up having heard of it. I will forever be in his debt for having taken me on, dealt with my extreme laziness and worked through my ability to procrastinate. Large parts of this thesis would have not existed if not for his attention to detail and desire to excel. He gave me enough freedom to study whatever I found interesting, and has been an amazing thesis advisor for the past year and a half.

Having good teachers is a privilege, and I have had several over the past two years. Prof. Raghunath Tewari was among the first professors I spoke to within the department, and he has tried his best to instill in me an appreciation for mathematical rigor. Prof. Rakesh Bansal's classes were eye-opening and his examinations a constant reminder as to why probability theory is a graduate topic. Prof. Piyush Rai introduced me to the world of machine learning, and if not for his lucid explanations and brilliant teaching, I would not have found myself working in the areas I did.

Every lab has its own atmosphere, and mine has been incredibly fostering and stimulating. Placements brought us together - Subhadip, Ankita, Gowtham, Satyandra, Vishak, Ravi, Utkarsh, Aditi, Akanksha, Sneha, Divya, all of you have been wonderful companions for the best part of a year. My other tea companions, Atul, Rohit, Utsab, Shailesh, Manish - thank you for the excellent breaks every day that allowed me to keep myself motivated throughout the thesis submission

period. And lest I forget, my friends from across the department - Arindam, Soumik, Nitish, Gundeep, Sandipan, Susmit for all the jokes, memes and videos that we have shared with each other.

Good company is necessary for getting by, and I have been fortune to have kept the company of some truly smart individuals. Nishit, Amur, Abhibhav, Sayash - I hope we shall continue our intellectual jousting.

Anshul and Samarth, thank you for providing me with the right amount of distraction, and for letting me step out of the academic bubble every once in a while. Thank you, Anurag, for your periodic reminders about how inept I am at mathematics, how a Ph.D. life is far more rewarding than selling out, and for your nagging to get around to writing my thesis.

My parents, brother, and grandmothers have all been exceptionally supportive. I will remain always in their debt.

Every person needs a talisman, and I'm no exception. Ritika, thank you for all your help, motivation, admonishments, and reminders. You never let me give up, and never let me get overconfident. It would have been quite impossible to get by without your encouragement and belief in me.

Contents

Acknowledgements **iii**

1 Introduction **3**

1.1 Our contributions 5

1.2 Structure of this document 5

2 Background **7**

2.1 Linear Regression 7

2.2 The IRLS algorithm 10

2.3 Robustness in Statistical Estimation 12

2.4 Robust Regression 12

2.5 Adversarial analysis 13

3 Related Works **15**

3.1 Results for Robust Regression 15

3.2 Iterative approaches 17

3.3 IRLS 17

4 Robust Regression and IRLS **19**

4.1 Introduction 19

4.2 Overview of the IRLS algorithm 19

4.3 Justifying IRLS: The Optimization Perspective 21

4.4 Relation with Other Approaches 23

4.5 Real world performance 23

4.6 Practical concerns 26

5 Convergence Guarantees for IRLS **27**

5.1 Convergence Analysis for Unidimensional Covariates 27

5.2 A Few Preliminaries 29

5.3 Convergence Analysis for Multidimensional Covariates 30

6 Failure Analysis for IRLS **35**

6.1 The Flaw in the IRLS Methodology 35

6.2 Failure cases for IRLS 36

7 Conclusion and Future Direction **39**

7.1 Regularized IRLS 39

7.2 Truncated IRLS 39

7.3 IRLS for Sparse Recovery 40

7.4 Gradient IRLS 40

List of Figures

4.1 Performance of IRLS against oblivious and partly adaptive adversaries. 25
4.2 A comparison of IRLS with `TORRENT` and OLS against a partly adaptive adversary. 25

6.1 IRLS fails to converge to gold model with unidimensional covariates against a partly adaptive adversary using adversarial models to introduce corruptions. 37
6.2 IRLS fails to converge to gold model with multidimensional covariates against a partly adaptive adversary using adversarial models to introduce corruptions. 38

Page intentionally left blank

Chapter 1

Introduction

Linear regression is a classical problem that pervades several areas including statistics, linear algebra, machine learning, data mining and others. Given a set of n covariates, each of which is d dimensional (arranged as a matrix) $X = [\mathbf{x}_1, \ldots, \mathbf{x}_n]^\top \in \mathbb{R}^{n \times d}$ and real responses (arranged as a vector) $\mathbf{y} = [y_1, \ldots, y_n] \in \mathbb{R}^n$, our aim is to discover a set of weights $\mathbf{w} \in \mathbb{R}^d$ such that $\mathbf{y} \approx X\mathbf{w}$. Various criteria may be adopted to designate a notion of approximation, such as least squares error, the absolute error, and so on.

Linear regression and its variants find applications in a wide variety of settings. Many a machine learning problem can be reduced to (a sequence of) linear regression problems, for example, learning with bandit information [1], multilabel learning [34] and matrix completion [18]. This motivates a desire for a deep understanding of algorithms for linear regression under various problem setting. Indeed, this problem has captivated mathematicians long before contemporary applications emerged, as is evident from classical works, such as those by Legendre [21].

Most approaches to regression assume that individual responses are generated using a simple generative model, such as $y_i = \langle \mathbf{w}^*, \mathbf{x}_i \rangle + \epsilon_i$ where $\mathbf{w}^*$ denotes an unknown *gold* model and ϵ_i denotes noise. A plethora of regression models, including the entire family of generalized linear models, homeo/hetero-scedastic noise models, and many others, can be constructed by choosing various generative models. In particular, the OLS (Ordinary Least Squares) technique assumes that the noise is independently and identically distributed normally.

It turns out that the assumption of independent stochastic noise is especially popular, as it often lends itself to simple estimators and analyses. However, real life applications often present structured noise which may violate these independence assumptions, rendering these estimators and analyses inapplicable. A particularly intriguing problem is that of *robust regression* wherein the noise may have been injected in a non-stochastic and indeed adversarial manner. This is especially true when executing regression algorithms in situations where malicious agents abound,

for example click fraud in recommendation systems, or else impersonation in biometric systems.

Although later chapters will present details of various adversary models, we can formalize the basic robust regression problem as follows: we assume there is a gold model $\mathbf{w}^* \in \mathbb{R}^d$ such that for every data point $i = 1, \ldots, n$, given the covariate $\mathbf{x}_i \in \mathbb{R}^d$, the response y_i is generated as follows $y_i = \langle \mathbf{w}^*, \mathbf{x}_i \rangle + b_i$ where b_i is the corruption introduced by the adversary. More succinctly, we have in vector notation $\mathbf{y} = X\mathbf{w}^* + \mathbf{b}$ where $\mathbf{b} = [b_1, \ldots, b_n]$. The corruption vector $\mathbf{b}$ is chosen by the adversary in a manner consistent with the adversary model, which we shall discuss shortly. However, we will always demand that the adversary introduce *sparse* corruptions $\|\mathbf{b}\|_0 \leq n_0 = \alpha \cdot n$ where $\alpha \in [0, 1)$ i.e. only n_0 of the points, which constitute and α fraction of the total data set, are corrupted and for the rest $y_i = \langle \mathbf{w}^*, \mathbf{x}_i \rangle$. Our task is to receive the generated data $(X, \mathbf{y})$ and return an estimate of the gold model $\mathbf{w}^*$.

The above model can be augmented in several ways, for instance, by allowing a combination of sparse corruptions for some data points and simple i.i.d. noise for the rest of the data points e.g. [5]. However, it should be noted that in order to ensure consistent recovery in the presence of an all-powerful adversary that can decide on the corruption vector $\mathbf{b}$ after observing X and $\mathbf{w}^*$, we must demand $\alpha < 0.5$ since even with $\alpha = 0.5$, such an adversary can render this problem ill-posed by first choosing a new model $\tilde{\mathbf{w}} \in \mathbb{R}^d$ and then setting $b_i = \langle \tilde{\mathbf{w}} - \mathbf{w}^*, \mathbf{x}_i \rangle$ for the corrupted points (of which there are now $0.5n$). In this situation, it is impossible to distinguish whether $\mathbf{w}^*$ is the gold model or $\tilde{\mathbf{w}}$. The largest value of α that an algorithm can tolerate while still offering consistent estimates of $\mathbf{w}^*$ is called its *breakdown point*. We will strive for a breakdown point $\alpha = \mathcal{O}(1)$ which would allow a constant fraction of data points to be corrupted and yet ensure consistent recovery of the gold model $\mathbf{w}^*$. The above argument shows that this is in essence, optimal.

It should be noted that a different line of work, one that seeks to perform linear regression in the presence of *heavy-tailed* noise such as Levy-distributed noise, is also often referred to as robust regression. This clash of nomenclature is unfortunate. However, we hasten to clarify the substantial distinctions between regression in the presence of heavy-tailed noise and that in the presence of a malicious adversary (our setting).

1. In the former, noise is still stochastic (albeit heavy tailed) whereas in our setting, a sentient adversary may be inducing noise after careful consideration.

2. In the former, the induced noise is purely additive and cannot perform operations such as sign flips for the response whereas in our setting, the adversary may very well perform actions such as flipping the sign of the response (for example, by setting $b_i = -\text{sign} y_i \cdot |2y_i|$).

3. Lastly, analyses for heavy-tailed settings often continue to assume that noise is generated i.i.d. whereas we consider structured noise which violates such independence assumptions.

Among the several methods used to solve linear regression problems, an especially prominent one is the technique of iteratively reweighted least squares, or IRLS. This algorithm is used to solve a wide variety of extensions to the linear regression problem, such as robust regression (with heavy tailed losses) and generalized linear modeling problems. In this thesis we question and explore the applicability of the IRLS method to the robust regression setting.

1.1 Our contributions

In this thesis, we analyze the popular Iteratively Reweighted Least Squares algorithm which is widely used to solve generalized linear modeling problems and regression problems in the presence of heavy tailed noise. This algorithm has witnessed a lot of attention and several variants exist, each customized to specific problem settings.

1. We reveal critical flaws in the vanilla IRLS algorithm and demonstrate a class of counter examples where IRLS cannot guarantee recovery of the gold model $\mathbf{w}^*$ even in the limit of infinite data $n \to \infty$. We show this to be the case even in the weak *oblivious adversary* model where the adversary has to commit to the corruption vector $\mathbf{b}$ before witnessing X and $\mathbf{w}^*$.

2. We complement this negative result with local convergence results where we show that IRLS, when modified slightly does indeed converge to the gold model $\mathbf{w}^*$ if initialized carefully. Our results require non-standard analytic techniques that are, to the best of our knowledge, novel in the robust learning literature. We believe that these modifications allow IRLS to offer global recovery without the need for careful initialization.

3. We present an empirical comparison of the performance of the IRLS algorithm as compared to other approaches on the robust regression problem in the presence of a malicious adversary.

1.2 Structure of this document

The thesis is broadly divided into two parts. Chapters 2 and 3 form the first part, and largely act as a self-contained overview of the background required for understanding the contributions of this thesis. In chapter 2, we review basic ideas used in solving the linear regression problem, as well as its variants. We outline two basic extensions to the ordinary least squares method, the lasso formulation as well as ridge regression. In addition, we review the basics behind adversarial analysis, robustness in statistics, and the IRLS algorithm. Chapter 3 covers recent results in these areas. We give an outline of recent advances in computationally efficient robust statistics, especially robust regression. When applicable, we state the main theorems and claims that form the state of the art for this problem.

Chapters 4, 5, and 6 form the core of the contributions of this thesis. In chapter 4, we outline the IRLS algorithm as applied to the robust regression problem. We show how IRLS can be viewed as part of a family of algorithms, and also show why there are some crucial drawbacks to the naive IRLS model. Chapter 5 provides a general overview of local convergence results that were obtained for the IRLS algorithm. We show two results, one in the 1-dimensional case, primarily to gain intuition, and a more powerful result in the general d-dimensional case. These results naturally differ, the first being a lot simpler and involving precise case-by-case analysis to obtain the best constants, whereas the second being applicable more generally.

Chapter 6 will provide a way to construct counter-examples that defeat the IRLS algorithm. For both fully adaptive, as well as oblivious adversaries, we show that naive initializations of the IRLS cannot guarantee convergence to the gold model $\mathbf{w}^*$. We include experimental verification of these examples, as well as provide a method to construct these examples and observe how the IRLS algorithm fails.

Chapter 2

Background

In this chapter, we will cover some basic results and concepts from regression analysis that forms the basis for this thesis. We shall take a look at the regression problem in detail, specifically the least squares estimator. We shall also outline the IRLS algorithm and give a brief overview of what results exist currently. Additionally, we will introduce various adversary models and describe the challenges in tackling each one.

2.1 Linear Regression

The linear regression problem is one of discovering the "best" linear/affine function that fits a given set of n data points, i.e. the covariates $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \in \mathbb{R}^{n \times d}$, and the corresponding set of responses $y = \{y_1, \ldots, y_n\} \in \mathbb{R}^n$. Regression analysis, especially linear regression, finds applications in several areas of learning and inference, such as learning with bandit information [1], multilabel learning [34] and matrix completion [18]. In all these areas, the overall problem can be reduced to (a sequence of) linear regression problems.

For sake of simplicity, we will not consider affine models in our discussion. Affine models can be easily implemented using linear models by appending a dummy feature/coordinate to all the covariates. It is common in regression analysis to assume that there exists a gold model $\mathbf{w}^*$ such that all responses were generated as

$$y_i = \langle \mathbf{w}^*, \mathbf{x}_i \rangle + \epsilon_i, \tag{2.1}$$

where ϵ_i denotes noise in the observed responses. It is common to assume that the noise variables are generated independently from some fixed distribution, say Gaussian for the OLS model. Given this, it is common to obtain a point estimate for $\mathbf{w}^*$, say $\hat{\mathbf{w}}$, using the maximum likelihood or maximum aposteriori estimators (by incorporating a prior distribution on the model $\mathbb{P}^{\text{prior}}[\mathbf{w}]$),

both which result in optimization problems that we describe below. Prediction is performed on a test covariate $\mathbf{x}^t$ as $\hat{y}^t = \langle \hat{\mathbf{w}}, \mathbf{x}^t \rangle$.

The other alternative, the so-called *Bayesian* formulation, is to utilize the prior distribution to obtain a posterior distribution $\mathbb{P}^{\text{post}}[\mathbf{w}]$ for $\mathbf{w}^*$ via the Bayes rule as

$$\mathbb{P}^{\text{post}}[\mathbf{w}] = \mathbb{P}\left[\mathbf{w} \,|\, X, \mathbf{y}\right] \propto \mathbb{P}\left[\mathbf{y} \,|\, X, \mathbf{w}\right] \cdot \mathbb{P}^{\text{prior}}[\mathbf{w}]$$

Given a test covariate $\mathbf{x}^t$, we first obtain the predictive posterior distribution on the response for the test covariate by conditioning.

$$\mathbb{P}^{\text{pred}}[y \,|\, \mathbf{x}^t] = \int \mathbb{P}\left[y \,|\, \mathbf{x}^t, \mathbf{w}\right] \cdot d\mathbb{P}^{\text{post}}[\mathbf{w}]$$

Prediction is then performed by either choosing the mode $\hat{y}^t = \arg\max_{y \in \mathbb{R}} \mathbb{P}^{\text{pred}}[y \,|\, \mathbf{x}^t]$ or the expectation $\hat{y}^t = \mathbb{E}_{y \sim \mathbb{P}^{\text{pred}}}[y \,|\, \mathbf{x}^t]$ of the predictive posterior distribution. For simple cases, such as when likelihood and prior distances are conjugates, the posterior and predictive posterior distributions are obtainable in closed form. In all other cases, sampling techniques are resorted to, in order to perform approximate inference.

In this thesis we will concentrate only on point estimates. However, it should be noted that point estimates for the models obtained using maximum aposteriori techniques can be shown to correspond to the modes of the corresponding posterior distributions.

2.1.1 Ordinary Least Squares

The ordinary least squares (OLS) estimator discovers a point estimate of the model by solving an optimization problem involving the squared loss function.

$$\hat{\mathbf{w}}_{\text{OLS}} = \arg\min_{\mathbf{w} \in \mathbb{R}^d} \sum_{i=1}^{n} (y_i - \langle \mathbf{w}, \mathbf{x}_i \rangle)^2,$$

However, the OLS estimate can also be easily seen to be the maximum likelihood estimate corresponding to the Gaussian likelihood function $\mathbb{P}\left[y_i \,|\, \mathbf{x}_i, \mathbf{w}^*\right] = \mathcal{N}(\langle \mathbf{w}^*, \mathbf{x}_i \rangle, \sigma^2) \propto \exp\left(-\frac{(y_i - \langle \mathbf{w}, \mathbf{x}_i \rangle)^2}{2\sigma^2}\right)$. The above objective function is convex (strongly convex if the covariate matrix is well conditioned) and differentiable, as well as the problem is unconstrained. Thus, all optima must be stationary points i.e. they must satisfy

$$\frac{\partial}{\partial \mathbf{w}} \sum_{i=1}^{n} (y_i - \langle \mathbf{w}, \mathbf{x}_i \rangle)^2 = \mathbf{0},$$

which gives us

$$\hat{\mathbf{w}}_{\text{OLS}} = (X^\top X)^{-1} X^\top \mathbf{y}.$$

The inverse is replaced by the Moore-Penrose pseudoinverse for ill-conditioned covariate matrices.

Given that the OLS uses the squared loss function and the Gaussian likelihood model to measure the "goodness" of the model parameter we are estimating, it is natural to ask if other loss functions can be used as well, for instance if other generative models seem more suited to the data at hand. Indeed, answering this question leads us to the different regression techniques. Below we give some of the other more popular variants for regression problems.

2.1.2 Ridge regression

While conceptually and algorithmically simple, the OLS relies entirely on the least squares objective on the measured data to obtain our estimator which can fail in scenarios where either the covariate matrix is ill conditioned, or when we have far too less data and are at a risk of fitting too closely to the data. Ridge regression is a popular workaround in these settings. At its core, ride regression aims to solve a modified objective,

$$\hat{\mathbf{w}}_{\mathrm{RR}} = \arg\min_{\mathbf{w}\in\mathbb{R}^d} \sum_{i=1}^{n} (y_i - \langle \mathbf{w}, \mathbf{x}_i \rangle)^2 + \lambda \left\| \mathbf{w} \right\|_2^2$$

The ridge regression formulation also corresponds to what is known as the Tikhonov Regularization technique as applied to the linear regression problem. A unique solution to the above formulation exists, and the above objective function is strongly convex, irrespective of whether the covariate matrix is well-conditioned or not. Following the first-order stationarity principle, we get

$$\hat{\mathbf{w}}_{\mathrm{RR}} = \left(X^\top X + \lambda I_d \right)^{-1} X^\top \mathbf{y}$$

It is notable that there exists a tradeoff when choosing the value of λ for our objective. A larger value of λ tends to favor solutions that lie close to the origin (i.e. $\left\| \mathbf{w} \right\|_2$ is small), whereas a smaller value tends to fit better to the data that we have, but at the risk of overfitting.

As with OLS, we may also arrive at the ridge regression formulation in a different way, by interpreting as the maximum aposteriori estimate with Gaussian likelihood as in the OLS example, and a Gaussian prior on the models $\mathbb{P}^{\mathrm{prior}}[\mathbf{w}] = \mathcal{N}(\mathbf{0}, \frac{1}{\lambda} I_d) \propto \exp\left(-\frac{\lambda \cdot \|\mathbf{w}\|_2^2}{2} \right)$.

2.1.3 LASSO

The LASSO, or Least Absolute Shrinkage and Selection Operator introduced in [29] is a seminal technique used very widely for sparse recovery and other problems. The LASSO is similar to the ridge regression estimator. However, whereas ridge regression works with a L_2 norm regularization

(or alternatively, a Gaussian prior), the LASSO imposes an L_1 norm regularization.

$$\hat{\mathbf{w}}_{\text{LASSO}} = \arg\min_{\mathbf{w}\in\mathbb{R}^d} \sum_{i=1}^{n}(y_i - \langle \mathbf{w}, \mathbf{x}_i\rangle)^2 + \lambda \|\mathbf{w}\|_1$$

The above also corresponds to the maximum aposteriori solution with Gaussian likelihoods and Laplacian priors i.e. $\mathbb{P}^{\text{prior}}[\mathbf{w}] \propto \exp\left(-\frac{\lambda\cdot\|\mathbf{w}\|_1}{2}\right)$. The LASSO does not admit any closed form solutions and the estimate is generally obtained in an iterative fashion, using techniques like subgradient descent, proximal gradient descent, least-angle regression, and the expectation maximization (EM) algorithm by exploiting the fact that Laplacian variables possess a moment generating function identical to those of a scale mixture of normally distributed variables.

In contrast to the ridge regression formulation, a larger value for the regularization parameter λ in LASSO does not promote setting all coordinates of the parameter $\mathbf{w}$ to zero. Instead, because of how the epigraphs of the L_1 norm function look, it promotes setting more and more coordinates of the parameter w to zero. Formally, we can say that a larger value of λ promotes greater sparsity, or decreasing the L_0 "norm" of the parameter w. This is because the L_1 norm acts as a *reasonable* proxy for the L_0 "norm". We use the word norm in quotes since the sparsity of a vector is not a true norm since it violates positive homogeneity ($\|c\cdot\mathbf{x}\|_0 \neq |c|\cdot\|\mathbf{x}\|_0$). Nevertheless this ability to promote sparsity in a controlled manner allows the LASSO technique to be used in settings like compressive sensing, where we wish to recover sparse solutions.

2.1.4 L_p norm Regression

Just as we saw different formulations arise by switching different priors, various formulations can arise by using different likelihood models as well. A prominent one is L_1-norm regression which is widely used in heavy-tailed noise settings. The objective used in this formulation is given below.

$$\hat{\mathbf{w}}_{L_1} = \arg\min_{\mathbf{w}\in\mathbb{R}^d} \sum_{i=1}^{n} |y_i - \langle \mathbf{w}, \mathbf{x}_i\rangle| + \lambda\cdot r(\mathbf{w}),$$

where $r(\cdot)$ is the regularizer that can be set to the L_2, L_1 regularizers depending on the application at hand. The above formulation corresponds to a Laplacian likelihood model i.e. $\mathbb{P}\left[y_i \mid \mathbf{x}_i, \mathbf{w}^*\right] \propto \exp\left(-\frac{|y_i - \langle \mathbf{w}, \mathbf{x}_i\rangle|}{2\sigma}\right)$. L_1 regularization has also been used in dealing with robust regression settings in the presence of an adversary (for example [32]).

2.2 The IRLS algorithm

The Iteratively Reweighted Least Squares algorithm takes inspiration from the ordinary least squares method, but augments it in a way that has allowed this method to lend itself well to different

settings, with incredible versatility. Indeed, the IRLS algorithm finds application in learning generalized linear models, robust regression, sparse recovery and many others. The algorithm, as the name suggests, works by solving multiple modified least squares objectives, in sequence. Say we observe the covariate matrix $X \in \mathbb{R}^{n \times d}$ and the response vector $\mathbf{y} \in \mathbb{R}^n$.

At any time step t, IRLS proceeds by choosing a set of weights, one for each data point, say $s_i^t \geq 0, i = 1, \ldots, n$ and then solving the following objective

$$\mathbf{w}_{t+1} = \arg\min_{\mathbf{w} \in \mathbb{R}^d} \sum_{i=1}^{n} s_i^t (\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i)^2 \tag{2.2}$$

Given a new estimate $\mathbf{w}_{t+1}$ for the model, IRLS now sets new weights $s_i^{t+1}, i = 1, \ldots, n$ (this step is done in an application-specific manner). It then obtains the next model iterate $\mathbf{w}_{t+2}$ by solving a weighted least-squares problem, but this time with the new weights $s_i^{t+1}, i = 1, \ldots, n$.

Notice that the objectives being solved in the IRLS algorithm are essentially the least squares objective, but with different data points given different weights. The idea is that as the algorithm progresses, it identifies certain data points as more valuable for a reliable estimation of the model parameter. This is a nice property of the IRLS algorithm – if we do indeed identify the points correctly at a given time step, then we can be assured of progress towards the correct model parameter in the very next iteration.

We can solve the weighted least squares problems in the IRLS algorithm by using the first-order stationarity principle, just as we did for the OLS algorithm

$$\begin{aligned}
\mathbf{w}_{t+1} &= \arg\min_{\mathbf{w}} \sum_{i=1}^{n} s_i^t (\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i)^2 \\
&= \arg\min_{\mathbf{w}} \sum_{i=1}^{n} \left(\langle \mathbf{w}, \sqrt{s_i^t} \mathbf{x}_i \rangle - \sqrt{s_i^t} y_i \right)^2
\end{aligned}$$

This is identical to the OLS estimator, except with the additional scaling ($\sqrt{s_i}$) for each covariate and response. Incorporating this, we can obtain the IRLS solution in the following manner :

$$\mathbf{w}_{t+1} = \left(X^\top S^t X \right)^{-1} X^\top S^t \mathbf{y}$$

where $S^t = \mathrm{diag}(s_1^t, \ldots, s_n^t) \in \mathbb{R}^{n \times n}$ is a diagonal matrix, with the entries as $S_{ii}^t = s_i^t$. We will discuss the IRLS algorithm in much greater detail in a subsequent chapter, this brief introduction being given a bit prematurely so that we may discuss relevant works.

2.3 Robustness in Statistical Estimation

The robustness of an estimator generally speaks about certain statistical properties that the estimator may enjoy with respect to resilience against deviations from modeling assumptions. For instance, consider the simple problem of mean estimation. We are given a set of samples from a probability distribution chosen from a certain parameterized family of distributions, say Gaussian. We know the family (it is Gaussian) but not the parameters (mean, variance). A popular estimator for the mean of a distribution is simply the empirical mean.

However, this estimator can be "fooled" if the samples are tampered with. Estimators that are resilient to various forms of such "tampering" form the object of study in the field of robust statistics. Classical works such as those of [15, 31] helped lay the foundations of this area. A more thorough treatment to this topic can be found in [16].

For instance, in Hüber's ϵ-contamination model, we assume that the samples given to us are drawn from a distribution of the form $(1-\epsilon)P_\theta + \epsilon Q$, where P is the distribution whose mean we actually wish to estimate, with parameter θ, and Q is some corruption distribution, often heavy tailed. Here, each sample has a $1-\epsilon$ probability of being drawn from the "clean" distribution and an ϵ probability of being drawn from the corruption distribution Q.

The empirical mean is not resistant to such corruptions and can return inconsistent estimates i.e. we may not converge to the true mean even in the limit of infinite sample. In fact, the empirical mean is sensitive to even a single sample being corrupted (since a single corrupted sample can allow an adversary to modify the estimate to pretty much anything the adversary desires), much less an ϵ-fraction of points being corrupted. An improved strategy is that of using the median, which is a robust alternative. The median is resistant to corruptions to a far larger fraction of points and cannot be made to arbitrarily deviate by corruptions to a small number of data points.

2.4 Robust Regression

As we have mentioned before, with respect to the generative model for linear regression specified by 2.1, it is usually assumed that the additive noise is benign. For instance, it is popularly assumed that the noise is generated stochastically, in an i.i.d. fashion and is unbiased i.e. $\mathbb{E}[\epsilon_i] = 0$. The OLS, in particular, assumes that the noise is i.i.d. and normal.

Instances where the above assumptions are violated, for instance due to the noise not being stochastic, or independent, result in models such as OLS giving poor results. In this thesis, we will investigate the more challenging *robust regression* setting where our data need not conform to the

above model. To simplify our discussion, we will consider a simple generative model [5, 4, 33]

$$y_i = \langle \mathbf{w}^*, \mathbf{x}_i \rangle + b_i \tag{2.3}$$

More compactly, we can write $\mathbf{y} = X\mathbf{w}^* + \mathbf{b}$ where $\mathbf{b} = [b_1, \ldots, b_n]$. The corruption vector $\mathbf{b}$ is chosen by the adversary. We discuss various adversary models below. However, we will always demand that the adversary introduce *sparse* corruptions $\|\mathbf{b}\|_0 \leq n_0 = \alpha \cdot n$ where $\alpha \in [0, 1)$ i.e. only an α fraction of the points are corrupted and for the rest $y_i = \langle \mathbf{w}^*, \mathbf{x}_i \rangle$. As we discussed in Chapter 1, the above model may be augmented to include both i.i.d. noise and sparse corruptions. We also remind the reader that although the term "robust regression" is also often used to denote approaches towards linear regression with heavy tailed noise, our models, and analyses are distinct.

For any set $S \subseteq [n]$, given any vector $\mathbf{v} \in \mathbb{R}^d$ let $\mathbf{v}_S$ denote the vector with coordinates $i \in S$ identical to those in $\mathbf{v}$ and coordinates $j \notin S$ set to zero. Similarly, given any matrix $X \in \mathbb{R}^{n \times d}$, let X_S denote the matrix with rows $i \in S$ identical to those in X and other rows $j \notin S$ set to $\mathbf{0}^\top \in \mathbb{R}^d$. Our goal can be then stated as solving the following problem

$$\min_{\substack{\mathbf{w} \in \mathbb{R}^d, \\ S \subset [n], |S| = n - n_0}} \|\mathbf{y}_S - X_S \mathbf{w}\|_2^2, \tag{2.4}$$

where n_0 is the number of corrupted points. Thus, our goal is to identify, both the correct model as well as the location of the uncorrupted points. In the next chapter, we shall survey some state of the art algorithms that solve this problem. In the chapters following that, we shall show how the IRLS algorithm performs on the robust regression problem, as well as give a brief overview of what kind of results we can expect.

2.5 Adversarial analysis

Adversarial analysis is a vital tool to ascertain the robustness properties of an algorithm and has recently gained prominence in machine learning applications as well, given the interest in identifying attacks on machine learning algorithms and models. For instance [28] show that contemporary models for object recognition can be fooled into giving a wrong prediction with as little as one pixel corruption in the input image. Several other such works exist [3, 14].

Adversary models can be diverse, depending on how much information is available to the adversary. We will look at three different kinds of adversaries in our analysis of the IRLS algorithm

Oblivious : In this model, the adversary will have to commit to an n_0-sparse corruption vector $\mathbf{b}$ before the covariates X or the gold model $\mathbf{w}^*$ are chosen. However, note that the adversary still has knowledge of our algorithm and more importantly, need not conform to any

distributions while selecting the (locations of the) corruptions.

Fully adaptive : In this model, the adversary can select $\mathbf{b}$ with complete knowlege of X and $\mathbf{w}^*$ in any manner possible. The only restriction is that the corruption vector must be n_0-sparse. Note that such an adversary can implement sign flips of responses etc.

Partly adaptive : In this model, the adversary has to choose the locations of the corruptions (i.e. supp($\mathbf{b}$)) before $X, \mathbf{w}^*$ are revealed but the actual corruptions can be decided with full knowledge of $X, \mathbf{w}^*$. This adversary can also implement sign flips etc.

Chapter 3

Related Works

The literature on robust statistics is too vast to be surveyed here. We refer the reader to monographs such as [17] for more comprehensive surveys. We present below an outline of the state of the art in robust regression, as well as past results for the IRLS algorithm for different settings.

3.1 Results for Robust Regression

Robust Regression is a very well studied problem, with varying notions of robustness, adversary models, and algorithmic techniques. Various factors decide the suitability of an algorithm, some prominent ones being

Speed and Scalability : does the method use a speedy algorithmic technique like hard-thresholding or gradient descent, or a slower one like LASSO, or still slower ones like cone-programming that struggle to scale?

Breakdown Point : what fraction of data can the method tolerate being corrupted while still provably guaranteeing recovery of the gold model?

Assumptions and Applicability : what assumptions does the method require, for instance assuming oblivious/adaptive adversary, or assuming a certain generative model for the data covariates?

We refer the reader to Table 3.1 (adapted from [4]) which summarizes the state of the art for the robust regression problem. For instance, [32] show that exact recovery of the gold model is possible even as the fraction of corrupted point approaches unity $\alpha \to 1$ as well as when the gold model $\mathbf{w}^*$ is sparse by solving the following optimization problem.

$$\min_{\mathbf{w} \in \mathbb{R}^d} \|\mathbf{w}\|_1 + \lambda \cdot \|\mathbf{y} - X^\top \mathbf{w}\|_1,$$

Work	Tolerance on α	Adversary model	Idea
Wright & Ma, [32]	$\alpha \to 1$	Oblivious	Relaxation via L_1
Chen & Dalalyan, [11]	$\alpha \geq \Omega(1)$	Adaptive	SOCP
Chen et. al, [9]	$\alpha \geq \Omega(\frac{1}{\sqrt{d}})$	Adaptive	Robust Thresholding
Nguyen & Tran, [22]	$\alpha \to 1$	Oblivious	Relaxation via L_1
Nguyen & Tran, [23]	$\alpha \to 1$	Oblivious	Relaxation via L_1
McWillians et. al., [32]	$\alpha \geq \Omega(\frac{1}{\sqrt{d}})$	Oblivious	Weighted subsampling
Bhatia et. al., [5]	$\alpha \geq \Omega(1)$	Adaptive	Hard Thresholding
Bhatia et. al., [4]	$\alpha \geq \Omega(1)$	Oblivious	Hard Thresholding

Table 3.1: Summary of different methods for Robust Regression

However, the result requires an oblivious adversary model, i.e. the adversary is compelled to commit to the corruption vector without any knowledge of the data covariates X or the gold model $\mathbf{w}^*$. Moreover, stringent assumptions are put on how data covariates are generated. Moreover, the optimization problem, although convex and similar to the LASSO technique we saw earlier, is still time consuming to solve for very large datasets due to the objective being non-differentiable, as well as due to the need to tune the regularization parameter λ very precisely.

The work of [9] also considers recovery of sparse gold models, but they allow distributed corruptions in the data covariate matrix X. This is more powerful than the corruption model we have seen where corruption is allowed only in the responses since we can always model corruption in responses as corruption in covariates but not the other way round. Their technique uses the notion of a "trimmed" inner product to optimize an objective similar to the LASSO. Although their corruption model allows for corruption in data covariates, their breakdown point is not very large and actually diminishes with the dimensionality of the covariates. Moreover, their method is not consistent and can only recover the gold model upto some constant error i.e. their method cannot provably approach the gold model even in the limit of infinite data.

Of note are the works of [22, 23, 32], all of which can tolerate a significant amount of corruption. The work of [4] was the first to propose a technique that guarantees consistent recovery of the gold model in the presence of (oblivious) adversarial corruptions on some $n_0 = \Omega(n)$ points and Gaussian noise on the rest $n - n_0$ points. All these algorithms run in polynomial time, as compared to some classical techniques such as least median of squares which is an intractable problem in general.

The recent work of [6] present a different setting where they can also tolerate an arbitrary high fraction of corruptions. Their output setting however, is the list decodable setting where the algorithm outputs a list of models instead of a single estimate, an accurate estimate of the gold model being guaranteed to be in the list.

Algorithm 1: `TORRENT`

Data: Covariates $X \in \mathbb{R}^{n \times d}$, responses $\mathbf{y} \in \mathbb{R}^n$, number of corruptions n_0
Result: Estimate of parameter $\mathbf{w}^*$

$S_0 = [n]$
for $t = 1, 2, \ldots T$ **do**
  $\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}} \sum_{i \in S^{t-1}}^{n} \left(\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i\right)^2$
  $r_i^{t+1} \leftarrow |y_i - \langle \mathbf{w}_{t+1}, \mathbf{x}_i \rangle|$
  $S_{t+1} \leftarrow \mathsf{HT}(\mathbf{r}^{t+1}, n - n_0)$ *//Select $n - n_0$ points with smallest residuals*
end
Output: $\mathbf{w}_T$

3.2 Iterative approaches

In this section, we give a brief overview of a recently proposed algorithm `TORRENT` [5] for the robust regression problem. Although the algorithm cannot provably resist a large fraction of data points being corrupted (the formal guarantees can only show tolerance to about 1% corruption rate), the method is very fast in practice, can tolerate a fully adaptive adversary, as well as makes very gentle assumptions regarding covariates and the gold model. Recall the robust regression problem

$$\min_{\substack{\mathbf{w} \in \mathbb{R}^d, \\ S \subset [n], |S| = n-k}} \|\mathbf{y}_S - X_S \mathbf{w}\|_2^2$$

As we can see, this is a problem with two unknowns $\mathbf{w}$ and S with optimal values as $\mathbf{w}^*$ and $S_* = [n] - \text{supp}(\mathbf{b})$ respectively. However, note that if someone were to hand us $\mathbf{w}^*$, we could easily recover the corruption vector as $\mathbf{b} = \mathbf{y} - X^\top \mathbf{w}^*$ and hence obtain S_*. Conversely, if someone were to hand us S_*, we can recover a very good estimate of $\mathbf{w}^*$ by solving a least squares problem using only points in S^*. This immediately motivates an alternating minimization-based algorithm.

The work of [5] provides an iterative scheme (see Algorithm 1) that generalizes the above intuition. At each time step, it tries to maintain an *active* set of points which it believes are uncorrupted i.e for which it thinks $b_i = 0$. These points are then used to obtain a model estimate which is then used to update the active set by choosing the data points which have the least residuals with respect to the model estimate. The authors show that this process provably converges to $\mathbf{w}^*$, if certain assumptions are satisfied (see [5, Theorem 3]). This work was later improved by the work of [4], where the authors could show convergence to the gold model, i.e. a consistent estimate, even when the non-corrupted points contain additive Gaussian noise in their responses.

3.3 IRLS

The IRLS algorithm has been extensively applied to various problem settings for several decades now. In fact, even the `TORRENT` algorithm can be seen as an IRLS variant since it sets weights

to 0 or 1 depending on whether the point is in the active set or not. In the next section we will formally show how TORRENT arises naturally as a technique closely related to IRLS. The IRLS formulation is itself very general, and depending on the form of weights chosen as well as the sort of loss function we choose to optimize on, we can come up with different algorithms.

Unsurprisingly then, IRLS has been widely used for several problems including robust regression in the ϵ contamination model in a line of work that extends back decades [27, 10, 13, 24], sparse recovery [12, 8, 19, 20]. We refer the reader to [7] for a nice exposition. The recent work of [26] showed that the IRLS algorithm in certain settings could be viewed as a variant of an optimization problem that the Physarum slime mold seems to solve in exploring food sources. They show convergence results for a damped version of the IRLS algorithm in the sparse recovery setting, and note that establishing global convergence properties still remains a challenge.

As noted in that work, the book [25] also provides a local convergence result, again noting that establishing global convergence for IRLS is a challenging task. The work of [2] shows that IRLS may be applied to optimize loss functions popular in robust regression tasks, including robust M-estimation problems. They present experimental results for applications to rotation averaging, triangulation and point cloud alignment. However, the theoretical analysis in the work only shows that the IRLS procedure offers monotonic progress. This cannot be translated into a convergence or a model recovery bound.

Chapter 4

Robust Regression and IRLS

4.1 Introduction

This chapter is the first of a three chapter discussion on the IRLS algorithm and its application to the robust regression problem. In this chapter we will present an overview of the IRLS algorithm as applied to the robust regression setting, as well as a variety of experiments run with the IRLS algorithm to demonstrate its practical usage.

We will show how the IRLS can be derived from alternating optimization applied to a regularized form of the naive least squares problem, as well as an alternative derivation of IRLS as an instance of the majorization minimization principle. We will also show how IRLS relates to some other algorithms recently proposed for the robust regression problem. Finally, we will show results of extensive experiments on IRLS for a variety of problem specifications.

4.2 Overview of the IRLS algorithm

In (2.2), we saw the basic formulation of the IRLS algorithm and commented that different versions of the algorithm arise from the way we set the values of the weights s_i^t for each data point. For the robust regression problem, the weights are set as follows.

$$\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}\in\mathbb{R}^d} \sum_{i=1}^{n} s_i^t (\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i)^2,$$
$$s_i^t = \frac{1}{|\langle \mathbf{w}_t, \mathbf{x}_i \rangle - y_i|}$$

Algorithm 2: IRLS for Robust Regression

Data: Covariates $X \in \mathbb{R}^{n \times d}$, responses $\mathbf{y} \in \mathbb{R}^n$
Result: Estimate of parameter $\mathbf{w}^*$
$\mathbf{w}_0 \leftarrow$ init
for $t = 1, 2, \dots T$ **do**
  Set weights $s_i^t = \frac{1}{|\langle \mathbf{w}_t, \mathbf{x}_i \rangle - y_i|}$ for $i = 1, \dots, n$
  Update model $\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}} \sum_{i=1}^n s_i^t \left(\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i\right)^2$
end
Output: $\mathbf{w}_T$

Algorithm 2 gives the pseudo-code for the IRLS procedure. Notice that although in Chapter 2 (2.4), we state the goal of robust regression as solving the following problem

$$\min_{\substack{\mathbf{w} \in \mathbb{R}^d, \\ S \subset [n], |S| = n-k}} \|\mathbf{y}_S - X_S \mathbf{w}\|_2^2$$

IRLS does not attempt to directly optimize the above objective. Instead of identifying a subset of points as corrupt, IRLS instead tries to give these points small weight. Note that the way IRLS assigns weights as inverse of the residuals does seem to effect this goal – data points with large residuals get downweighted whereas those will very small residuals get large weights.

IRLS can be initialized in a variety of ways. One way to initialize the algorithm is to directly set the weights for every point, instead of setting a starting point via $\mathbf{w}_0$. A popular strategy is to set $s_i^0 = 1$ for all the points. This reduces to solving the usual OLS objective for the first iteration. Another way to initialize IRLS is to set $\mathbf{w}_0$ to be a random vector. As we have seen earlier, the IRLS updates the model parameter $\mathbf{w}_t$ can be obtained in closed form at each time step

$$\mathbf{w}_{t+1} = \left(X^\top S^t X\right)^{-1} X^\top S^t \mathbf{y},$$

where $S^t = \mathrm{diag}(s_1^t, \dots, s_n^t) \in \mathbb{R}^{n \times n}$ is a diagonal matrix, with the entries as $S_{ii}^t = s_i^t$. We may choose to stop the IRLS algorithm after a sufficient number of steps, specified by the parameter T, or else check if the algorithm makes significant progress. This can be done by checking $\|\mathbf{w}_t - \mathbf{w}_{t+1}\|_2$ and comparing with an appropriate threshold.

The intuition behind the IRLS algorithm is to utilize good estimates of the model into getting even better estimates. The weight assigned to each point is inversely proportional to the residual incurred by the current model estimate on that point. Consequently, if the current model estimate is close to the gold model $\mathbf{w}^*$, then the residuals on all the uncorrupted points would be low, and those on the corrupted points would be high. Consequently, the next estimate of the model can be expected to neglect corrupted points even more and approach the gold model even more closely.

4.3 Justifying IRLS: The Optimization Perspective

In this section we will provide two independent justifications for the way IRLS performs its updates – one via Alternating Minimization and the other via Majorization Minimization. Recall that the IRLS algorithm performs the following updates at each iteration

$$\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}\in\mathbb{R}^d} \sum_{i=1}^{n} s_i^t (\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i)^2,$$
$$s_i^t = \frac{1}{|\langle \mathbf{w}_t, \mathbf{x}_i\rangle - y_i|}$$

4.3.1 Alternating Minimization

Recall the original robust regression problem

$$\min_{\substack{\mathbf{w}\in\mathbb{R}^d,\\ S\subset[n],|S|=n-k}} \|\mathbf{y}_S - X_S\mathbf{w}\|_2^2,$$

and consider the following regularized relaxation

$$\min_{\mathbf{w}\in\mathbb{R}^d,\mathbf{s}\geq 0} \sum_{i=1}^{n} s_i \left(\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i\right)^2 + \sum_{i=1}^{n} \frac{1}{s_i},$$

where the constraint $\mathbf{s} \geq 0$ is shorthand for $s_i \geq 0$ for all $i = 1, \ldots, n$. Instead of selecting a discrete subset of data points, we are assigning positive weights to each data point. The regularization $\frac{1}{s_i}$ ensures that we do not blindly set $s_i = 0$ to all the points in an effort to minimize the first term in the objective.

This is a non-convex optimization problem in two variables, the parameter vector $\mathbf{w}$ as well as the per-datapoint weights, $\mathbf{s}$. Alternating minimization [18], which involves alternately fixing one of the variables and updating the other, is a popular optimization technique for such objectives. Even though this optimization problem is non-convex, for a fixed model parameter $\mathbf{w}_0$, the optima for $\mathbf{s}$ can be found again by applying first-order stationarity condition. Notice that the terms are actually seperable, so it suffices to optimize on each individual term,

$$\min_{s_i} \left(s_i \left(\langle \mathbf{w}_0, \mathbf{x}_i\rangle - y_i\right)^2 + \frac{1}{s_i} \right)$$

which can be found in closed form, as,

$$s_i = \frac{1}{|\langle \mathbf{w}_0, \mathbf{x}_i\rangle - y_i|}$$

which is precisely the IRLS update for the weights. On the other hand, given a fixed set of weights

$\mathbf{s}^0$, it is easy to see that the optimal value of the model parameter can be found out by solving

$$\min_{\mathbf{w}\in\mathbb{R}^d} \sum_{i=1}^n s_i^0 \left(\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i\right)^2 + \sum_{i=1}^n \frac{1}{s_i} = \min_{\mathbf{w}\in\mathbb{R}^d} \sum_{i=1}^n s_i^0 \left(\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i\right)^2,$$

which again is exactly what the IRLS does in order to perform model updates. Thus, we see that IRLS exactly corresponds to performing alternating minimization on a regularized relaxation to the original robust regression problem.

4.3.2 Majorization Minimization

Consider the following L_1 norm optimization problem

$$\min_{\mathbf{w}\in\mathbb{R}^d} \sum_{i=1}^n \left|\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i\right|$$

We recall this formulation from Chapter 2 and commented there that this formulation is often used in settings with heavy-tailed noise, as well as those with corruptions. It turns out that IRLS can be alternatively derived as majorization–minimization principle applied to this problem.

The majorization-minimization (MM) principle is a powerful technique that has seen much success in optimization literature. Put simply, if trying to minimize an objective function $f : \mathbb{R}^d \to \mathbb{R}$, given a current iterate $\mathbf{w}_t$, the MM principle proceeds in two steps

Majorization : identify a tight *majorizer* of f at $\mathbf{w}_t$, i.e. a function $g_t : \mathbb{R}^d \to \mathbb{R}$ such that

1. g_t dominates f i.e. for all $\mathbf{w} \in \mathbb{R}^d$, we have $g_t(\mathbf{w}) \geq f(\mathbf{w})$
2. the domination is tight at $\mathbf{w}_t$ i.e. we have $g_t(\mathbf{w}_t) = f(\mathbf{w}_t)$

Minimization : set the next iterate as $\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}\in\mathbb{R}^d}\ g_t(\mathbf{w})$

MM turns out to assure monotonic progress no matter what function f we choose. This can be seen easily as we have $f(\mathbf{w}_{t+1}) \leq g_t(\mathbf{w}_{t+1}) \leq g_t(\mathbf{w}_t) = f(\mathbf{w}_t)$. It turns out that if we apply the MM principle to the function $f(\mathbf{w}) = \sum_{i=1}^n \left|\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i\right|$ with the majorizer defined at each step as follows (it is easy to verify that this is a proper majorizer)

$$g_t(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^n \frac{\left(\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i\right)^2}{\left|\langle \mathbf{w}_t, \mathbf{x}_i\rangle - y_i\right|} + \frac{1}{2}\sum_{i=1}^n \left|\langle \mathbf{w}_t, \mathbf{x}_i\rangle - y_i\right|,$$

then it can be seen that we recover back the IRLS algorithm as an instance of MM. A nice corollary we get as a result is that IRLS always makes monotonic progress (although not necessarily strictly) with respect to the L_1 norm regression objective $f(\mathbf{w}) = \sum_{i=1}^n \left|\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i\right|$.

4.4 Relation with Other Approaches

While motivating IRLS as an instance of alternating minimization, we introduced the following objective as a relaxation of the robust regression problem

$$\min_{\mathbf{w}\in\mathbb{R}^d, \mathbf{s}\geq 0} \sum_{i=1}^{n} s_i \left(\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i\right)^2 + \sum_{i=1}^{n} \frac{1}{s_i},$$

where the regularization $\frac{1}{s_i}$ ensures that we do not blindly set $s_i = 0$ to all the points in an effort to minimize the first term in the objective. However, we can prevent such trivial solutions by means of introducing constraints as well. One possible way to do so is the following

$$\begin{aligned} \min_{\mathbf{w}\in\mathbb{R}^d, \mathbf{s}\in\mathbb{R}^n} & \sum_{i=1}^{n} s_i \left(\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i\right)^2 \\ & \sum_{i=1}^{n} s_i = n - n_0 \\ & s_i \in [0, 1], \end{aligned}$$

where $n_0 = \alpha \cdot n$ is the number of corrupted points. The above optimization problem is still a non-convex one since its objective is not jointly convex in $\mathbf{w}$ and $\mathbf{s}$. However, it turns out that if we attempt to perform alternating minimization with this optimization probelm, then we actually recover the recently proposed `TORRENT` algorithm in [5].

Indeed, it is not hard to see that for a given model iterate $\mathbf{w}_t$, the optimal value of the parameter $\mathbf{s}$ is found by setting $s_i^t = 1$ for the $n-k$ points with the least residuals ($|\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i|$) and setting $s_i^t = 0$ for the k points with highest residuals. Thus, the relaxation $s_i \in [0, 1]$ is actually never exploited, the alternating minimization procedure always sets $s_i \in \{0, 1\}$. Note that this implicitly selects a subset of $n - k$ points $S_t = \{i : s_i^t = 1\}$ at each time step.

Thus, it is interesting that alternating minimization when applied to a regularized relaxation gives IRLS whereas when applied to a constrained relaxation gives `TORRENT`.

4.5 Real world performance

The IRLS algorithm performed well in general in our experiments, living quite well upto its popularity for various learning problems. The algorithm usually converged in a few iterations and is a promising candidate for robust regression. We demonstrate the performance of IRLS in this section on a few toy settings.

Experimental setup

Our experiments consist of a variety of synthetically generated datasets with the following model

$$\mathbf{y} = X\mathbf{w}^* + \mathbf{b},$$

where $\mathbf{y} \in \mathbb{R}^n$, $X \in \mathbb{R}^{n \times d}$, $\mathbf{w}_* \in \mathbb{R}^d$ and $\mathbf{b} \in \mathbb{R}^n$ with $\|\mathbf{b}\|_0 \leq \alpha n$. In our experiments, unless otherwise mentioned, we drew the covariates, i.e. rows of the matrix X from a centered, isotropic standard Gaussian distribution, $\mathbf{x}_i \sim \mathcal{N}(0, \mathbb{I}_d)$. The gold model $\mathbf{w}^*$ was also drawn from a centered isotropic Gaussian distribution. We used a standard Intel Core i5-3210M CPU running at 2.50 GHz for all our experiments. All code was written in Python2, using the package `numpy`.

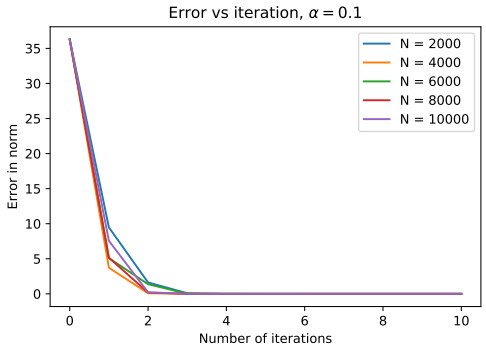
Oblivious Adversary

Figures 4.1a, 4.1b and 4.1c show the convergence offered by IRLS on datasets with varying levels of corruptions $\alpha = 0.1, 0.25, 0.45$, against an oblivious adversary. The locations of the corruptions were chosen randomly and the corruption values were sampled from a normal distribution. The graphs show the model recovery error $\|\mathbf{w}_t - \mathbf{w}^*\|_2$ incurred by successive iterates of IRLS. For each level of corruption, we repeated the experiment with increasing dataset sizes as well.

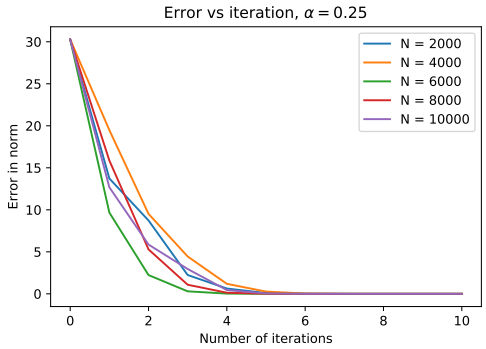
Partly Adaptive Adversary

Figure 4.1d shows the performance of IRLS on a partly adaptive adversary, as the fraction of corrupted data points α is varied. The value of the corruptions (their locations having been chosen randomly, hence obliviously) were based on an adversarially chosen model $\tilde{\mathbf{w}}$ i.e. the adversary set $b_i = \langle \tilde{\mathbf{w}} - \mathbf{w}^*, \mathbf{x}_i \rangle$ so that the observed response becomes $y_i = \langle \tilde{\mathbf{w}}, \mathbf{x}_i \rangle$ instead of $y_i = \langle \mathbf{w}^*, \mathbf{x}_i \rangle$. The experiments were repeated with an increasing number k of such adversarial models. For example for $k = 3$, there were 3 adversarial models $\tilde{\mathbf{w}}_j, j = 1, 2, 3$ and one third of corrupted points received their corruption as $b_i = \langle \tilde{\mathbf{w}}_1 - \mathbf{w}^*, \mathbf{x}_i \rangle$, another one third received their corruption as $b_i = \langle \tilde{\mathbf{w}}_2 - \mathbf{w}^*, \mathbf{x}_i \rangle$ and so on.

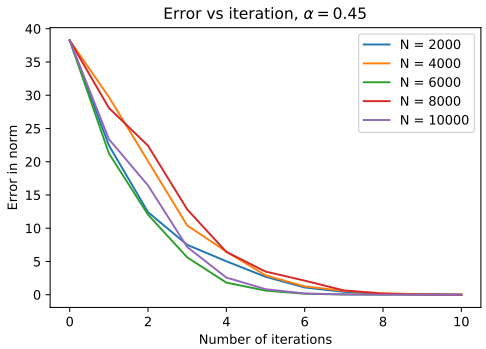
We observe that there does not seem to be much of a correlation, and with the increase in the number of adversarial models being used to introduce corruption. In fact, we observe that the IRLS actually does better than with a larger number of adversarial models and $k = 1$ turns out to be the most challenging case, possibly because with a larger number of adversarial models, only the gold model $\mathbf{w}^*$ dominates in terms of sheer numbers whereas with $k = 1$, the sole adversarial model can compete with the gold model. Note that with $k = 1$ and $\alpha = 0.5$, IRLS does poorly. However, we have already discussed that in this case, it becomes impossible to distinguish whether $\mathbf{w}^*$ is the gold model or is $\tilde{\mathbf{w}}$ the gold model and hence it is not surprising that IRLS fails.



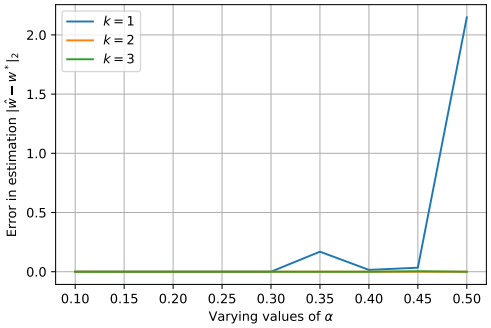
(a) Estimation error vs iteration with an oblivious adversary, for varying number of data points at $\alpha = 0.1$. d = 10



(b) Estimation error vs iteration with an oblivious adversary, for varying number of data points at $\alpha = 0.25$. d = 10

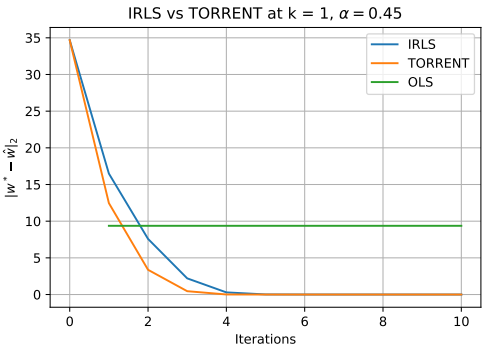


(c) Estimation error vs iteration with an oblivious adversary, for varying number of data points at $\alpha = 0.45$. d = 10

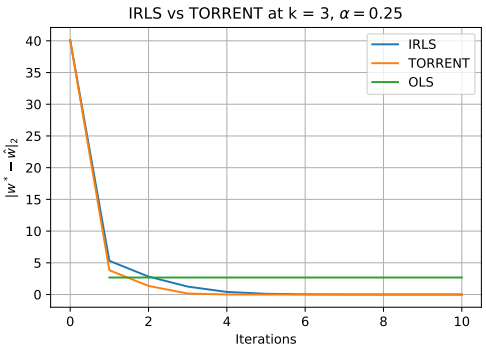


(d) Estimation error after 100 iterations with an adaptive adversary, with different number k of adversarial models. N = 1000, d = 10.

Figure 4.1: The first three graphs plot the recovery error $\|\mathbf{w}_t - \mathbf{w}^*\|_2$ offered by IRLS as a function of time (iteration) for settings with varying corruption levels α and dataset sizes n when the adversary is oblivious. The final graph shows the performance when the adversary is partly adaptive is choosing the corruptions based on adversarially chosen models $\tilde{\mathbf{w}}$ i.e. $b_i = \langle \tilde{\mathbf{w}} - \mathbf{w}^*, \mathbf{x}_i \rangle$ so that we observe $y_i = \langle \tilde{\mathbf{w}}, \mathbf{x}_i \rangle$. The graph shows how IRLS performs when there are varying number of such adversarially chosen models. For example for $k = 3$, there are 3 adversarial models $\tilde{\mathbf{w}}_j, j = 1, 2, 3$ and one third of corrupted points receive their corruption from each one of these models.



(a) Error in estimation $\|\hat{\mathbf{w}} - \mathbf{w}^*\|_2$ vs number of iterations, `TORRENT` vs IRLS. Specifications of input: $N = 10000, d = 10, \alpha = 0.45, k = 1$.



(b) Error in estimation $\|\hat{\mathbf{w}} - \mathbf{w}^*\|_2$ vs number of iterations, `TORRENT` vs IRLS. Specifications of input: $N = 10000, d = 10, \alpha = 0.25, k = 3$.

Figure 4.2: A comparison of IRLS with `TORRENT` and OLS against a partly adaptive adversary.

Figure 4.2 compares IRLS with `TORRENT`, a method proposed for robust regression in [5], in the partly adaptive adversary setting (we used the fully corrective version of the algorithm `TORRENT-FC`). Both `TORRENT` as well as IRLS perform comparably, with IRLS having a slight edge in case there are multiple adversarial models being used to introduce corruptions, and `TORRENT` being better when there is a single adversarial model being used to introduce corruptions.

4.6 Practical concerns

The IRLS algorithm does not impose any magnitude restriction on the weights. On careful inspection, we can see how this is actually not very feasible in practice. If any of the points conforms perfectly with the model at a particular iteration, then the weight on that point s_i tends to infinity. This can cause numerical instability given the finite precision arithmetic environments within which these algorithms are implemented. In fact, we shall see in the following chapters that this can actually cause IRLS to fail to converge at all. We will also see that modifying the IRLS algorithm slightly by imposing upper bounds on the magnitudes of the weights s_i is actually a very desirable step in practice and in theory. In fact, all experimental results we showed in this chapter, actually use a "truncated" version of IRLS where the weights are capped. However, the cap we use is really large, of the order of 10^{12}.

Chapter 5

Convergence Guarantees for IRLS

In this chapter, we will show convergence results for a *truncated* version of the IRLS algorithm, one which, as mentioned before, upper bounds the weights assigned to any data point. We will begin with an analysis of IRLS in the toy case of unidimensional covariates to gain some insight, and then generalize this proof to arbitrary dimensions.

Our proofs for both settings will ensure a linear rate of convergence and will require novel concepts and proof techniques. However, our proofs will only ensure a local convergence bound for the IRLS algorithm. We will justify this by demonstrating empirically in the next chapter, that if not initialized properly, IRLS fails to demonstrate a linear rate of convergence.

5.1 Convergence Analysis for Unidimensional Covariates

Recall that the generative model in the robust regression setting generates the responses as

$$y_i = w^* \cdot x_i + b_i,$$

where the corruption is introduced, i.e. $b_i \neq 0$, for at most $n_0 = \alpha \cdot n$ data points. Note that we have used a scalar notation w^*, x_i for the gold model and covariates instead of the usual vector notation since we are in a unidimensional setting at the moment. Let us call the set of upted points as $G = \{i : b_i = 0\}$ the set of corrupted points as $B = \{i : b_i \neq 0\}$. For sake of simplicity, we will abuse notation and let B, G respectively, denote the sizes of these sets as well. Let us also revisit the IRLS update step, but with truncation put in

$$w_{t+1} = \arg\min_{w} \sum_{i=1}^{n} s_i^t (\langle w, x_i \rangle - y_i)^2$$

$$s_i^t = \min \left\{ \frac{1}{|\langle w_t, x_i \rangle - y_i|}, \frac{1}{\epsilon} \right\}$$

Note that we have specified the truncation as $\frac{1}{\epsilon}$. Let us denote $\Delta_t = w_t - w^*$ be the model discrepancy in the t-th iterate of the algorithm. For the following analysis, we will also further partition the point sets G and B according to whether their weights were truncated or not. Specifically, we will denote $G_T = \left\{i \in G : s_i^t = \frac{1}{\epsilon}\right\}, G_N = G \backslash G_T$ as well as $B_T = \left\{i \in B : s_i^t = \frac{1}{\epsilon}\right\}, B_N = B \backslash B_T$.

We will additionally assume that all covariates satisfy $|x_i| \leq 1$. We note that this is without much loss of generality but will simplify our arguments and help us gain intuition before we move on the multi-dimensional analysis.

As before, applying the first order optimality principle gives us $w_{t+1} = \frac{A}{B}$ where

$$\begin{aligned}
B &= \sum_{i=1}^{n} \frac{x_i^2}{s_i^t} = \sum_{i \in G_N} \frac{|x_i|}{|\Delta_t|} + \sum_{i \in B_N} \frac{x_i^2}{|x_i \cdot \Delta_t - b_i|} + \sum_{i \in G_T \cup B_T} \frac{x_i^2}{\epsilon} \\
&= \frac{1}{|\Delta_t|} \left(\sum_{i \in G_N} |x_i| + \sum_{i \in B_N} \frac{x_i^2 |\Delta_t|}{|x_i \cdot \Delta_t - b_i|} + \sum_{i \in G_T \cup B_T} \frac{x_i^2 |\Delta_t|}{\epsilon} \right) \\
A &= \sum_{i=1}^{n} \frac{x_i y_i}{s_i^t} = w^* \cdot B + \sum_{i \in B} \frac{x_i b_i}{s_i^t} \\
&= w^* \cdot B + \sum_{i \in B_N} \frac{x_i b_i}{|x_i \cdot \Delta_t - b_i|} + \sum_{i \in B_T} \frac{x_i b_i}{\epsilon}
\end{aligned}$$

where we substituted $y_i = w^* \cdot x_i + b_i$ for expanding both A and B, and did some elementary manipulations. The above gives us

$$|\Delta_{t+1}| \leq |\Delta_t| \cdot \frac{\sum_{i \in B_N} \frac{|x_i b_i|}{|x_i \cdot \Delta_t - b_i|} + \sum_{i \in B_T} \frac{|x_i b_i|}{\epsilon}}{\sum_{i \in G_N} |x_i| + \sum_{i \in B_N} \frac{x_i^2 |\Delta_t|}{|x_i \cdot \Delta_t - b_i|} + \sum_{i \in G_T \cup B_T} \frac{x_i^2 |\Delta_t|}{\epsilon}} =: |\Delta_t| \cdot \frac{P}{Q}.$$

Now, for any point $i \in B_T$, we have $|y_i - w^* \cdot x_i| \leq \epsilon$ which gives us $\epsilon \geq |b_i - x_i \cdot \Delta_t| \geq |b_i| - |x_i \cdot \Delta_t|$. This allows us to bound $|b_i| \leq \epsilon + |x_i \cdot \Delta_t|$ i.e. $\frac{|x_i b_i|}{\epsilon} \leq |x_i| + \frac{x_i^2 |\Delta_t|}{\epsilon}$. Moreover, for $i \in B$, in particular any $i \in B_N$, we have $|b_i| \leq |b_i - x_i \Delta_t| + |x_i \Delta_t|$ which gives us $|x_i b_i| \leq |x_i| |b_i - x_i \Delta_t| + x_i^2 |\Delta_t|$. These put together allow us to bound

$$P \leq \sum_{i \in B} |x_i| + \sum_{i \in B_N} \frac{x_i^2 |\Delta_t|}{|x_i \cdot \Delta_t - b_i|} + \sum_{i \in B_T} \frac{x_i^2 |\Delta_t|}{\epsilon},$$

whereas since all covariates satisfy $|x_i| \leq 1$, we have $|x_i| \geq x_i^2$ for any $i \in [n]$, and in particular for any $i \in G_N$. Moreover, we also assume $|\Delta| \geq \epsilon$ since we would anyway be unable to ensure a much closer convergence with truncated weights. Both these together give us

$$Q \geq \sum_{i \in G} x_i^2 + \sum_{i \in B_N} \frac{x_i^2 |\Delta_t|}{|x_i \cdot \Delta_t - b_i|} + \sum_{i \in B_T} \frac{x_i^2 |\Delta_t|}{\epsilon}$$

At this point we notice that if $a \leq b$ and $c \leq d$ then $\frac{a+c}{b+c} \leq \frac{a+d}{b+d}$. Assuming $\sum_{i \in B} |x_i| \leq \sum_{i \in G} x_i^2$ (which is something that holds for a wide variety of sub-Gaussian distributions whenever $|B| / |G|$

is small enough i.e. α is a sufficient small constant), lets us establish, upon observing that for all $i \in B_N$ so we have $|x_i \cdot \Delta_t - b_i| \geq \epsilon$,

$$|\Delta_{t+1}| \leq |\Delta_t| \cdot \frac{\sum_{i \in B} |x_i| + \sum_{i \in B} \frac{x_i^2 |\Delta_t|}{\epsilon}}{\sum_{i \in G} x_i^2 + \sum_{i \in B} \frac{x_i^2 |\Delta_t|}{\epsilon}} \leq |\Delta_t| \cdot \frac{\sum_{i \in B} |x_i| + \sum_{i \in B} \frac{x_i^2 |\Delta_t|}{\epsilon}}{\sum_{i=1}^{n} x_i^2}$$

Given the above, we can now state our convergence result for IRLS.

Theorem 5.1.1 (IRLS convergence - unidimensional)**.** *Let the (unidimensional) covariates presented to the IRLS algorithm satisfy $|x_i| \leq 1$ for all $i \in [n]$. Also, for any set $S \subset [n]$, denote $a_S := \sum_{i \in S} |x_i|$ and $s_S := \sum_{i \in S} x_i^2$. Then if the covariates additionally satisfy $a_B \leq s_G$ as well as if IRLS is initialized at a model w_0 so that $|\Delta_0| = |w_0 - w^*| < \frac{s_{[n]} - a_B}{s_B} \cdot \epsilon$, then IRLS offers $|w_T - w^*| \leq \epsilon$ after atmost $T = \mathcal{O}\left(\log \frac{1}{\epsilon}\right)$ many iterations.*

Note that the above is firmly a local convergence result. The next chapter will justify this by showing that unless properly initialized, IRLS may fail to demonstrate a linear rate of convergence. We note that if the covariates are drawn from a sub-Gaussian distribution with support restricted to the interval $[-1, 1]$, then we may expect $s_S, a_S = \Theta(|S|)$ for $S = B, G, [n]$ since the adversary is partially oblivious and hence is unable to choose the support of the corrupted points.

Thus, $a_B \leq s_G$ would be satisfied whenever $\frac{B}{G}$ is small enough i.e. the corruption rate α is a sufficient small constant. Also note that the initialization expected is of the form $|w_0 - w^*| < \frac{G}{B}\epsilon$ whereas a model satisfying $|w_T - w^*| \leq \epsilon$ is guaranteed. This implies that the procedure is able to reduce the distance to the gold model by a factor of $\frac{B}{G}$.

5.2 A Few Preliminaries

In the previous section we saw a local convergence guarantee for IRLS in the unidimensional case. The analysis, although toy and not extendable in its form, gave us much insight into how IRLS can be analyzed. More specifically, it showed that if analyzed locally, IRLS is capable of bringing the model closer to the gold model by a constant factor.

Ideally, such a result can be bootstrapped, with IRLS being run in *stages*, with each stage bringing the model closer to the gold by another constant factor. We will briefly discuss such schemes in Chapter 7. However, for now, our goal is to establish a local convergence bound for IRLS in the more realistic multidimensional setting. For this we require a more rigorous background in the theory of random matrices. We present the relevant concepts and results below.

We denote the unit sphere in d dimensions using S^{d-1}. For any $\gamma \in (0, 1]$, we let $\mathcal{S}_\gamma = \{S \subset [n] : |S| = \gamma \cdot n\}$ denote the set of all subsets of size $\gamma \cdot n$. For any matrix $X \in \mathbb{R}^{n \times d}$, and set $S \subset [n]$, we let X_S denote the $n \times d$ matrix with rows $i \in S$ identical to those in X and rows

$j \notin S$ filled with zero vectors. Also, for any vector $\mathbf{v} \in \mathbb{R}^n$, we use the notation $\mathbf{v}_S$ to denote the n-dimensional vector with coordinates $i \in S$ identical to those in $\mathbf{v}$ but coordinates $j \notin S$ filled with zeros. We use $\lambda_{\min}(X)$ and $\lambda_{\max}(X)$ to denote, respectively, the smallest and largest eigenvalues of a square symmetric matrix X. We now state two properties, namely, Subset Strong Convexity and Subset Strong Smoothness that were introduced in [5].

Definition 5.2.1 (Subset Strong Convexity and Smoothness [5])**.** *A matrix $X \in \mathbb{R}^{n\times d}$ is said to satisfy the Subset Strong Convexity (SSC) (resp. Subset Strong Smoothness (SSS)) property at level γ with strong convexity constant λ_γ (resp. strong smoothness constant Λ_γ) if it satisfies,*

$$\lambda_\gamma \leq \min_{S\in S_\gamma} \lambda_{\min}\left(X_S^\top X_S\right) \leq \max_{S\in S_\gamma} \lambda_{\max}\left(X_S^\top X_S\right) \leq \Lambda_\gamma$$

However, the above property was designed for the `TORRENT` algorithm which, as we discussed in Chapter 3, worked with different subsets of data points. For analyzing IRLS, which instead sets weights to all data points, we introduce a more generalized property in this thesis.

Definition 5.2.2 (Weighted Strong Convexity and Smoothness)**.** *A matrix $X \in \mathbb{R}^{n\times d}$ is said to satisfy the Weighted Strong Convexity (WSC) (resp. Weighted Strong Smoothness (WSS)) property with respect to a diagonal matrix S with non-negative entries with strong convexity constant $\lambda(S)$ (resp. strong smoothness constant $\Lambda(S)$), if it satisfies,*

$$\lambda(S) \ \leq \lambda_{\min}\left(X^\top S X\right) \leq \lambda_{\max}\left(X^\top S X\right) \leq \Lambda(S)$$

5.3 Convergence Analysis for Multidimensional Covariates

We will denote $X = [\mathbf{x}_1, \ldots, \mathbf{x}_n]^\top \in \mathbb{R}^{n\times d}$ as the covariate matrix, $\mathbf{w}^*$ as the gold model, $\mathbf{b} \in \mathbb{R}^n$ as the $n_0 = \alpha \cdot n$-sparse vector of corruptions and $\mathbf{y} = X\mathbf{w}^* + \mathbf{b} \in \mathbb{R}^n$ as the (corrupted) response vector. Let $S = \mathrm{diag}(s_1, s_2, \ldots, s_n)$ be the weights give to data points $i = 1, \ldots, n$. Recall that the truncated IRLS algorithm considers the current iterate $\mathbf{w}_t$, obtains the corresponding residual $r_i = y_i - \langle \mathbf{w}_t, \mathbf{x}_i \rangle$, sets the weights as $s_i = \min\left\{\frac{1}{|r_i|}, M\right\}$, and obtains the next iterate as

$$\mathbf{w}_{t+1} = (X^\top S^t X)^{-1} X S^t \mathbf{y} = \mathbf{w}^* + (X^\top S^t X)^{-1} X^\top S^t \mathbf{b}$$

Let $B = \mathrm{supp}(\mathbf{b})$ be the coordinates of the corrupted data points and $G = [n] - B$ be the set of uncorrupted points. For sake of simplicity, we will abuse notation and let B, G respectively, denote the sizes of these sets as well. Then note that we have $X^\top S^t \mathbf{b} = X_B^\top S_B^t \mathbf{b}_B$. We will assume that the covariate matrix satisfies the WSC/WSS properties as mentioned above with respect to the weight matrix S^t i.e. $\lambda(S^t) \leq \lambda_{\min}(X^\top S^t X) \leq \lambda_{\max}(X^\top S^t X) \leq \Lambda(S^t)$ for all time steps t.

Our goal would be to obtain an upper bound on the quantity $\left\|X^\top S^t \mathbf{b}\right\|_2$ since we have

$$\|\mathbf{w}_{t+1} - \mathbf{w}^*\|_2 \leq \frac{\left\|X^\top S^t \mathbf{b}\right\|_2}{\lambda(S^t)}$$

Note that IRLS (ignoring truncation for a moment) sets $s_i = \frac{1}{|r_i|}$ and so when $\mathbf{w} = \mathbf{w}^*$ then $\mathbf{r}_B = \mathbf{b}$ and hence $S_B^t \mathbf{b}$ is a vector whose entries are ± 1. This means that at $\mathbf{w} = \mathbf{w}^*$ we have $\left\|X^\top S^t \mathbf{b}\right\|_2 \approx \mathcal{O}(B)$. We will next show how to ensure that IRLS approaches a similar value.

Let us $\mathbf{r}^+$ denote the residuals offered by $\mathbf{w}_{t+1}$ and let S^+ denote the diagonal matrix of weights we assign as a result, using the above procedure. Then we have

$$\mathbf{r}^+ = \mathbf{b} - X(X^\top S^t X)^{-1} X^\top S^t \mathbf{b}$$

Multiplying both sides with $X_B^\top S_B^+$ gives us

$$\begin{aligned} X_B^\top S_B^+ \mathbf{r}^+ &= X_B^\top S_B^+ \mathbf{r}_B^+ \\ &= X_B^\top S_B^+ \mathbf{b} - X_B^\top S_B^+ X (X^\top S^t X)^{-1} X^\top S^t \mathbf{b} \\ &= X \top S^+ \mathbf{b} - X_B^\top S_B^+ X_B^\top (X^\top S^t X)^{-1} X^\top S^t \mathbf{b}, \end{aligned}$$

where the second step holds since $\text{supp}(\mathbf{b}) = B$. Now, had we not performed truncation of the weights, we would have had $S^+ \mathbf{r}^+ =: \boldsymbol{\eta} = \text{sign}(\mathbf{r}^+)$ i.e. it would have been a sign vector. Thus, we would have had $\left\|X_B^\top S_B^+ \mathbf{r}_B^+\right\|_2^2 = \left\|X_B^\top \boldsymbol{\eta}_B\right\|_2^2 \leq \|\boldsymbol{\eta}_B\|_2^2 \cdot \Lambda(I_B) \leq B \cdot \Lambda(I_B)$ where I is the identity matrix. However, notice that even after truncation, we are still assured that $s_i r_i \leq 1$ which ensures that we still have $\left\|X_B^\top S_B^+ \mathbf{r}_B^+\right\|_2^2 \leq B \cdot \Lambda(I_B)$. This gives us

$$\begin{aligned} \left\|X^\top S^+ \mathbf{b}\right\|_2^2 &\leq B \cdot \Lambda(I_B) - \left\|X_B^\top S_B^+ X_B (X^\top S^t X)^{-1} X^\top S^t \mathbf{b}\right\|_2^2 + 2 \cdot \mathbf{b}^\top S^+ X X_B^\top S_B^+ X_B (X^\top S^t X)^{-1} X^\top S^t \mathbf{b} \\ &\leq B \cdot \Lambda(I_B) + 2 \cdot \mathbf{b}^\top S^+ X X_B^\top S_B^+ X_B (X^\top S^t X)^{-1} X^\top S^t \mathbf{b} \\ &\leq B \cdot \Lambda(I_B) + \frac{2 \cdot \Lambda(S_B^+)}{\lambda(S^t)} \left\|X^\top S^+ \mathbf{b}\right\|_2 \left\|X^\top S^t \mathbf{b}\right\|_2 \end{aligned}$$

Solving the quadratic equation yeilds

$$\left\|X^\top S^+ \mathbf{b}\right\|_2 \leq \sqrt{B \cdot \Lambda(I_B)} + \frac{2 \cdot \Lambda(S_B^+)}{\lambda(S^t)} \left\|X^\top S^t \mathbf{b}\right\|_2$$

The above result shows that whenever we have $\frac{2 \cdot \Lambda(S_B^+)}{\lambda(S^t)} < 1$, we make substantial progress in reducing the quantity $\left\|X^\top S^t \mathbf{b}\right\|_2$ towards $\sqrt{B \cdot \Lambda(I_B)}$. Now the subset strong condition from [5] for subGaussian covariates ensures with high probability that $\Lambda(I_B) \leq \sqrt{B + \cdot o(n)}$ which in turn

ensures that $\sqrt{B \cdot \Lambda(I_B)} \leq \sqrt{B^2 + B \cdot o(n)} = \mathcal{O}(B)$ (recall that we earlier agreed that this is its lowest value possible). Also note that due to truncation, we have $\Lambda(S_B^+) \leq \Lambda(M \cdot I_B) \leq \mathcal{O}(MB)$ by a similar argument as before. In the following we will assume that the data covariates satisfy $\lambda(S) \geq \Omega(\text{trace}(S)) - o(n)$. Now, we can have either of two cases

1. Case 1: we have $s_i = M$ for at least $G/2$ of the uncorrupted points. In this case we have $\text{trace}(S) \geq GM/2$. Thus, we get, ignoring constants,

$$\frac{2 \cdot \Lambda(M \cdot I_B)}{\lambda(S)} \leq \frac{2 \cdot (MB + o(n))}{GM/2 - o(n)} \leq \frac{C_1 B}{G},$$

for some constant C_1 where the last step holds for large enough n. In this case, whenever $B/G < \frac{1}{2C_1}$, we have the above quantity less than $1/2$.

2. Case 2: we have $s_i = \frac{1}{|r_i|}$ for at least $G/2$ of the uncorrupted points. Call this set of points G_{m}. These are good points but with large residuals. In this case, we have by a repeated application of Jensen's inequality,

$$\begin{aligned}\text{trace}(S) &\geq \sum_{i \in G} s_i = \sum_{i \in G} \min\left\{\frac{1}{|r_i|}, M\right\} \geq \sum_{i \in G_{\text{m}}} \frac{1}{|r_i|} \\ &\geq \frac{G_{\text{m}}}{\frac{1}{G_{\text{m}}} \sum_{i \in G_{\text{m}}} |r_i|} \geq \frac{G_{\text{m}} \sqrt{G_{\text{m}}}}{\|\mathbf{r}_{G_{\text{m}}}\|_2} \geq \frac{G\sqrt{G}}{2\sqrt{2}\,\|\mathbf{r}_G\|_2}\end{aligned}$$

$\mathbf{b}_G = \mathbf{0}$, means $\mathbf{r}_G = X(\mathbf{w}^* - \mathbf{w})$ i.e. $\|\mathbf{r}_G\|_2 \leq \|\mathbf{w} - \mathbf{w}^*\|_2 \cdot \sqrt{\Lambda(I_G)} \leq \|\mathbf{w} - \mathbf{w}^*\|_2 \cdot \sqrt{2G}$ i.e.

$$\frac{2 \cdot \Lambda(M \cdot I_B)}{\lambda(S)} \leq \frac{2 \cdot (MB + o(n))}{\frac{G}{4 \cdot \|\mathbf{w} - \mathbf{w}^*\|_2} - o(n)} \leq \frac{C_2 MB \|\mathbf{w} - \mathbf{w}^*\|_2}{G}$$

Assuming an initialization $\left\|\mathbf{w}^0 - \mathbf{w}^*\right\|_2 \leq \frac{G}{2C_2 BM}$, the above quantity is less that $1/2$.

The above arguments show that we can ensure $\left\|X^\top S^t \mathbf{b}\right\|_2 \leq C_3 B$, for some constant $C_3 > 0$ within $\mathcal{O}(\log B)$ steps of the algorithm. Once this has happened, we come to the final stages of the algorithm. From now on we assume that the current iterate $\mathbf{w}$ yields residuals $\mathbf{r}$ and weights S^t, such that $\left\|X^\top S^t \mathbf{b}\right\|_2 \leq C_3 B$. Now one of two things can happen

1. Case 1: we have $s_i = M$ for at least $G/2$ of the uncorrupted points. In this case we have $\lambda(S) \geq \Omega(GM/2) - o(n)$ and get,

$$\left\|\mathbf{w}^+ - \mathbf{w}^*\right\|_2 \leq \frac{\left\|X^\top S^t \mathbf{b}\right\|_2}{\lambda(S)} \leq \frac{C_4 B}{GM}$$

This is the final guarantee we have to offer. With truncated weights, IRLS can offer no significantly better convergence. However, note that whereas we assumed an initialization

$\left\|\mathbf{w}^0 - \mathbf{w}^*\right\|_2 \leq \frac{G}{2C_2BM}$ in the discussion above, we are thereafter assuring that we will converge to $\|\mathbf{w}^+ - \mathbf{w}^*\|_2 \leq \frac{C_4 B}{GM}$. Thus, we improved the distance to the gold model by a factor of $\mathcal{O}\left(\frac{G}{B}\right)^2$.

2. Case 2: we have $s_i = \frac{1}{|r_i|}$ for at least $G/2$ of the uncorrupted points. In this case we have, as before,

$$\lambda(S) \geq \Omega\left(\frac{G}{\|\mathbf{w} - \mathbf{w}^*\|_2}\right) - o(n) \geq \Omega\left(\frac{G}{\|\mathbf{w} - \mathbf{w}^*\|_2}\right),$$

for large enough n. The above assures us that

$$\left\|\mathbf{w}^+ - \mathbf{w}^*\right\|_2 \leq \frac{C_5 B}{G} \|\mathbf{w} - \mathbf{w}^*\|_2,$$

for some constant $C_5 > 0$. This means that if $B/G < \frac{1}{2C_5}$ then we have $\|\mathbf{w}^+ - \mathbf{w}^*\|_2 \leq \frac{1}{2}\|\mathbf{w} - \mathbf{w}^*\|_2$, i.e. we now have a linear rate of convergence to the optimum.

The above shows that truncated IRLS keeps decreasing $\|\mathbf{w} - \mathbf{w}^*\|_2$ at a linear rate until it can ensure $\|\mathbf{w} - \mathbf{w}^*\|_2 \leq \frac{C_4 B}{GM}$. We can now state our convergence result for IRLS.

Theorem 5.3.1 (IRLS convergence - multidimensional)**.** *Let the covariates presented to the IRLS algorithm satisfy the subset strong smoothness property for the subset of corrupted points B with $\lambda_{\max}(X_S^\top X_S) \leq \mathcal{O}(|S|)$ for $S = G, B \subset [n]$ and the weighted strong convexity property as $\lambda_{\min}(X^\top S X) \geq \Omega(trace(S)) - o(n)$. Then if IRLS is intialized at a point $\mathbf{w}^0$ satisfying $\left\|\mathbf{w}^0 - \mathbf{w}^*\right\|_2 \leq \frac{G}{2C_2BM}$ for some small enough constant C_2 and if $B/G \leq C_6$ for some small enough constant C_6, then the truncated IRLS algorithm offers an iterate $\mathbf{w}^T$ satisfying $\left\|\mathbf{w}^T - \mathbf{w}^*\right\|_2 \leq \frac{C_4 B}{GM}$ for some constant C_4 after atmost $T = \mathcal{O}\left(\log \frac{1}{\epsilon}\right)$ many iterations.*

Note that the above is also a local convergence result. The next chapter will justify this by showing that unless properly initialized, IRLS may fail to demonstrate a linear rate of convergence in higher dimensional covariates as well. However, note that whereas IRLS assumes an initialization $\left\|\mathbf{w}^0 - \mathbf{w}^*\right\|_2 \leq \mathcal{O}\left(\frac{G}{BM}\right)$, it ensures $\|\mathbf{w}^+ - \mathbf{w}^*\|_2 \leq \mathcal{O}\left(\frac{B}{GM}\right)$. Thus, the algorithm improves the distance to the gold model by a factor of $\mathcal{O}\left(\frac{G}{B}\right)^2$.

The work of [5] establishes the SSS property required by the above result for covariates drawn from a variety of subGaussian distributions. A useful result in establishing the WSC property for random matrices is the following Matrix Chernoff bound by [30]. The text itself is a good source of such tail inequalities for random matrices.

Theorem 5.3.2 (Matrix Chernoff [30])**.** *Let $X_1, \ldots X_K$ be a set of independent, random Hermitian matrices of common dimension $d \times d$. Assume that, $0 \leq \lambda_{min}(X_k) \leq \lambda_{max}(X_k) \leq L$ for each index k and define $Y = \sum_k X_k$. Also define the minimum and maximum eigenvalues of the expectation*

$\mathbb{E}Y$ *as* $\mu_{\min} = \lambda_{\min} \sum_k \mathbb{E}X_k$ *and* $\mu_{\max} = \lambda_{\max} \sum_k \mathbb{E}X_k$ *respectively. Then for any* $\theta > 0$*, we have*

$$\mathbb{P}\left[\lambda_{\min}(Y) \leq (1-\epsilon)\mu_{min}\right] \leq d\left[\frac{e^{-\epsilon}}{(1-\epsilon)^{1-\epsilon}}\right]^{\mu_{min}/L}$$
$$\mathbb{P}\left[\lambda_{max}(Y) \geq (1+\epsilon)\mu_{max}\right] \leq d\left[\frac{e^{\epsilon}}{(1+\epsilon)^{1+\epsilon}}\right]^{\mu_{max}/L}$$

Lemma 5.3.3. *For covariates drawn from an isotropic distribution i.e.* $\mathbb{E}\left[\mathbf{x}\mathbf{w}^\top\right] = I_d$*, we have* $\lambda(S) \geq$ *trace*$(S) - o(n)$ *with high probability.*

Proof. For a fixed S, the result follows from a straightfoward application of the above inequality by taking $X_i = s_i \cdot \mathbf{x}_i\mathbf{x}_i^\top$ and noting that 1) we always invoke this result in settings where $\text{trace}(S) \geq \Omega(BM)$ as in Case 1 we have $\text{trace}(S) \geq GM/2$ and in Case 2 we have $\text{trace}(S) \geq G/\left\|\mathbf{w} - \mathbf{w}^*\right\|_2$ and $\left\|\mathbf{w} - \mathbf{w}^*\right\|_2 \leq B/GM$, 2) we may use $L = M$ since that is the upper bound on the weight any point can receive in truncated IRLS, and 3) due to the isotropic nature of our covariates, we have $\mu_{\min} = \mu_{\max} = \text{trace}(S)$, we get, for any value of $\epsilon \geq \frac{1}{8}$,

$$\mathbb{P}\left[\lambda_{\min}(Y) \leq (1-\epsilon)\cdot\text{trace}(S)\right] \leq d\left[\frac{e^{-\epsilon}}{(1-\epsilon)^{1-\epsilon}}\right]^{B} \leq d\exp(-\Omega(n)) \qquad \square$$

We may even establish a uniform convergence bound over all such matrices by applying an ϵ-net argument. Notice that if we have two diagonal weight matrices such that $S_1 = \text{diag}(\mathbf{s}^1)$ and $S_2 = \text{diag}(\mathbf{s}^2)$ and $\left\|\mathbf{s}_1 - \mathbf{s}_2\right\|_2 \leq \epsilon$, then we have for any unit vector $\mathbf{v} \in \mathbb{R}^d$,

$$\begin{aligned}\left|\mathbf{v}^\top X S_1 X^\top \mathbf{v} - \mathbf{v}^\top X S_2 X^\top \mathbf{v}\right| &= \left|\sum_{i=1}^n (s_i^1 - s_i^2)\left\langle \mathbf{v}, \mathbf{x}_i\right\rangle^2\right| \leq \left\|\mathbf{s}^1 - \mathbf{s}^2\right\|_2 \sqrt{\sum_{i=1}^n \left\langle \mathbf{v}, \mathbf{x}_i\right\rangle^4} \\ &\leq \left\|\mathbf{s}^1 - \mathbf{s}^2\right\|_2 \left(\sum_{i=1}^n \left\langle \mathbf{v}, \mathbf{x}_i\right\rangle^2\right) \leq \epsilon \left\|X\mathbf{v}\right\|_2^2 \leq \epsilon \left\|X\right\|_2^2,\end{aligned}$$

which implies that $\left|\lambda_{\min}(XS_1X^\top) - \lambda_{\min}(XS_2X^\top)\right| \leq \epsilon\left\|X\right\|_2^2$. Taking a union bound over an ϵ-net with $\epsilon \leq \alpha/4$ over all normalized weight vectors (i.e. $\mathbf{s} \in \mathbb{R}^n : \left\|\mathbf{s}\right\|_1 \leq 1$) then finishes the argument since the volume of the unit L_1 ball is $\mathcal{O}\left(\frac{2^n}{n!}\right)$ to get

$$\mathbb{P}\left[\exists t : \text{trace}(S^t) \geq BM, \lambda_{\min}(XS^tX^\top) \leq (1-2\epsilon)\cdot\text{trace}(S^t)\right] \leq \frac{C_7 2^n d}{n!}\left[\frac{e^{-\epsilon}}{(1-\epsilon)^{1-\epsilon}}\right]^{B} \leq d\exp(-\Omega(n))$$

Chapter 6

Failure Analysis for IRLS

In this section, we argue using experimental results, why we believe global convergence results for the IRLS algorithm may not be feasible. We present a series of counterexamples that cause the IRLS algorithm to fail if not initialized properly. While these results are negative, they also let us understand the key drawbacks of IRLS, and what measures can be taken to work around them.

6.1 The Flaw in the IRLS Methodology

Recall that the IRLS algorithm performs the following steps in each iteration

$$\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}} \sum_{i=1}^{n} s_i^t (\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i)^2$$

$$s_i^t = (|\langle \mathbf{w}_t, \mathbf{x}_i \rangle - y_i|)^{-1}$$

The algorithm itself does not impose any restriction on the weights s_i^t. As we mentioned before, in our experiments, we capped the weights at an exaggerated limit $\approx 10^{12}$ to avoid divide-by-zero errors. However, this still leaves it possible (and as we also observed in practice) that some weights become extremely large, and drown out the weights on the rest of the points when it comes to solving the reweighted objective at each iteration. Although this is a happy scenario if these large weights are being placed on uncorrupted points, the same becomes a problem if large weights are placed on corrupted points, which can happen, for instance if IRLS is not initialized properly.

In particular, if IRLS at time t has a model $\mathbf{w}_t$ such that for a particular point, $|\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i| = \epsilon \approx 0$, then the resultant weight on that particular point will blow up for the next time step. Given this, if an adaptive adversary chooses an adversarial model $\tilde{\mathbf{w}}$ and sets $b_i = \langle \tilde{\mathbf{w}} - \mathbf{w}^*, \mathbf{x}_i \rangle$ for all the corrupted data points, and if IRLS ever approaches $\tilde{\mathbf{w}}$ during its iterations, due to improper initialization or otherwise, then weights on all the corrupted points will blow up whereas those on

Table 6.1: Notation used in figures depicting failure cases for IRLS

Symbol	Interpretation
$\tilde{\mathbf{w}}$	Adversarial models that generate responses for the corrupted points
$\mathbf{w}^*$	True parameter that IRLS seeks to recover
$\mathbf{w}_{\text{IRLS}}$	Parameter that IRLS converges to (at convergence criteria)
α	Fraction of points that are allowed to be corrupted
k	Number of adversarial models

the clean points will stay bounded. Thus, IRLS can actually start converging to $\tilde{\mathbf{w}}$, as a vicious cycle will set in, with the weights on the corrupted points increasing with each iteration.

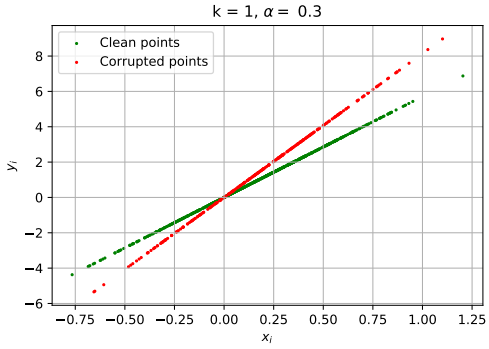
One thing to note is that if we artificially restrict the smallest possible residue any point can obtain during an execution of IRLS (which is equivalent to capping the largest weight any point can receive), then this modified form of IRLS actually converges in practice, but often after several iterations and in the previous chapter we saw that such a technique actually offers a convergence guarantee, albeit a local convergence guarantee, and that too to an error level that depends on the level of the capping. In our experiments, we analyze the performance of IRLS at the end of the first 20-25 iterations. This is generally accepted as a reasonable running time for any algorithm that exhibits linear rate of convergence, to converge to its final solution. In our experiments, we observe that upon running for a long enough time, IRLS almost always converged, but that there was no clear procedure to choose how many iterations to run the IRLS procedure.

6.2 Failure cases for IRLS

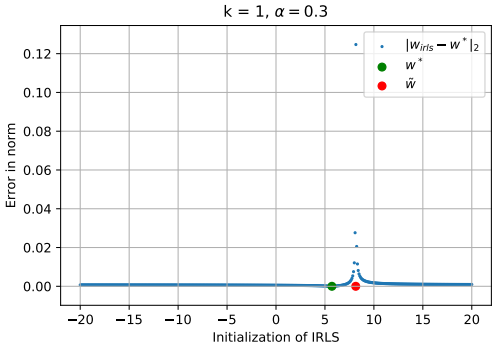
For sake of visual inspection, we will present our counter examples where the IRLS algorithm fails to converge for 1 and 2 dimensional covariates. We will work with a partly oblivious adversary, one that does not get to control which points get corrupted but that can decide on the corruption values on the chosen points. In all cases, the adversary will initially (before data covariates are generated) identify one or more *adversarial models* $\tilde{\mathbf{w}}_j, j = 1, 2, \ldots, k$ where k will denote the number of such adversarial models.

When deciding the corruption value for a certain data point i that was designated as corrupted, our adversary will simply pick up one of these adversarial models at random, say $\tilde{\mathbf{w}}_{j_i}$ and set the corruption value as $b_i = \langle \tilde{\mathbf{w}}_{j_i} - \mathbf{w}^*, \mathbf{x}_i \rangle$ so that the response observed on this data point is simply $y_i = \langle \tilde{\mathbf{w}}_{j_i}, \mathbf{x}_i \rangle$. Note that such corruptions can be used to flip the sign of the response (by setting $\tilde{\mathbf{w}} = -2\mathbf{w}^*$, or in general scale the response by a factor of c (by setting $\tilde{\mathbf{w}} = (c+1) \cdot \mathbf{w}^*$). See Table 6.1 for notation used in the experiments.

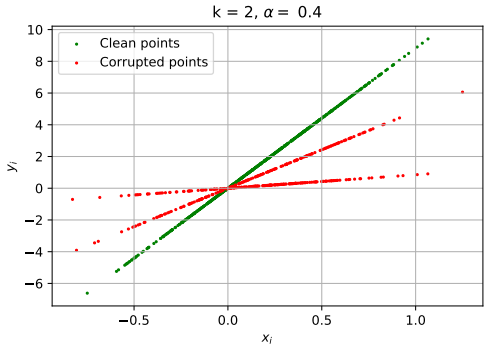
The first set of results in Figure 6.1 consider unidimensional covariates (all experiments used 10000 data points). The figures on the left depict the covariate and response pairs in a 2D plot. The red points indicate corrupted points and the green points indicate clean points. Since our



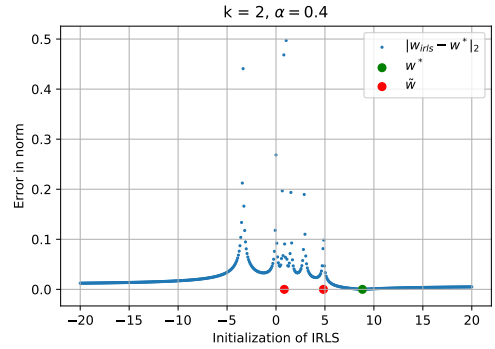
(a) Input points, at corruption fraction of $\alpha = 0.3$ and number of adversarial models $k = 1$.



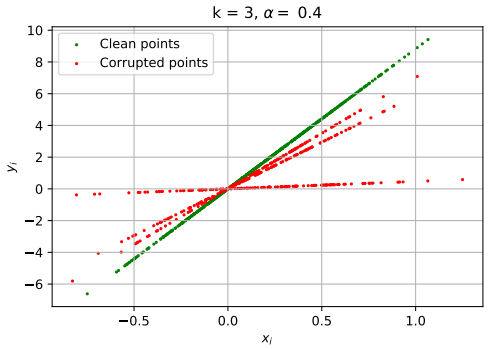
(b) Model recovery error $|\mathbf{w}_{irls} - \mathbf{w}^*|$ versus initialization of IRLS ($\mathbf{w}_0$).



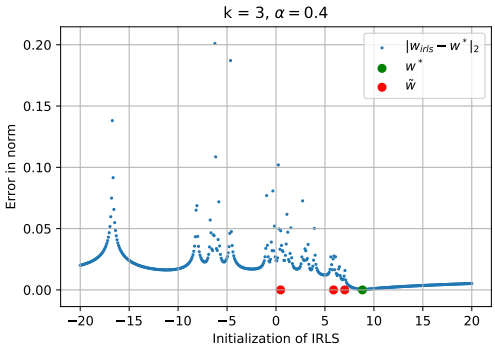
(c) Input points, at corruption fraction of $\alpha = 0.4$ and number of adversarial models $k = 2$.



(d) Model recovery error $|\mathbf{w}_{irls} - \mathbf{w}^*|$ versus initialization of IRLS ($\mathbf{w}_0$).



(e) Input points, at corruption fraction of $\alpha = 0.4$ and number of adversarial models $k = 3$.

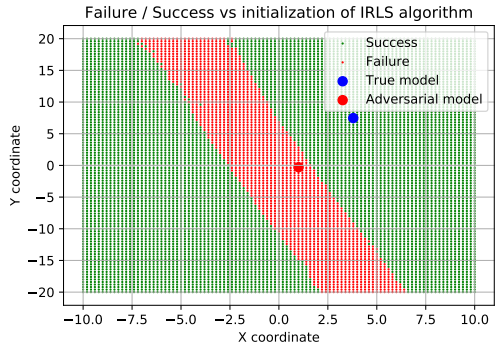


(f) Model recovery error $|\mathbf{w}_{irls} - \mathbf{w}^*|$ versus initialization of IRLS ($\mathbf{w}_0$).

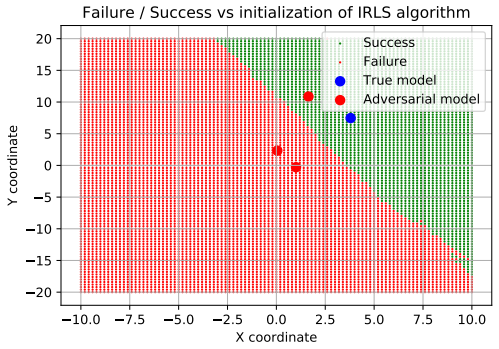
Figure 6.1: IRLS fails to converge to gold model with unidimensional covariates against a partly adaptive adversary using adversarial models $\tilde{\mathbf{w}}_j$ to introduce corruptions as $b_i = \langle \tilde{\mathbf{w}}_j - \mathbf{w}^*, \mathbf{x}_i \rangle$ ($n = 10000$). Figures in the three rows correspond, respectively, to adversaries that used $1, 2$ and 3 adversarial models to introduce corruptions. IRLS was run for 20 iterations in all experiments.

adversaries were allowed to utilize more than one adversarial model to introduce corruptions, we can notice more than one linear model fitting the red points in some of the graphs.

We see how the presence of the adversary causes the IRLS algorithm to incur a large model recovery error even after 20 iterations, if the algorithm is not initialized properly. In the figures on the right-hand side, the x-axis represents the location where IRLS was initialized and the y-axis indicates the model recovery error incurred by IRLS after 20 iterations from the given initialization.



(a) Initialization of IRLS vs success or failure. $n = 1000, \alpha = 0.3$, single adversarial model.



(b) Initialization of IRLS vs success or failure. $n = 1000, \alpha = 0.45$, 3 adversarial models.

Figure 6.2: IRLS fails to converge to gold model with 2D covariates against a partly adaptive adversary using adversarial models $\tilde{\mathbf{w}}_j$ to introduce corruptions as $b_i = \langle \tilde{\mathbf{w}}_j - \mathbf{w}^*, \mathbf{x}_i \rangle$. IRLS was run for 25 iterations in all experiments. An initialization point is colored green if IRLS ensures $\|\mathbf{w}_{\text{IRLS}} - \mathbf{w}^*\|_2 \leq 10^{-5}$ within 25 iterations and is colored red otherwise.

The presence of multiple adversarial models further degrades the performance of IRLS. In fact, as Figures 6.1d and 6.1f indicate, IRLS ends up converging to multiple local optima depending on where exactly it was initialized. We observe that in general, IRLS incurs a large error if initialized close to one of the adversarial models, we notice in all graphs that it performs poorly if initialized at some other specific locations as well. For instance, in Figure 6.1f, we can see that although there are only 3 adversarial models, there appear to be as many as 8 points where, if IRLS is initialized, gives poor convergence. We found that initialization at these *spuriously* bad points is still redeemable by running IRLS for several more iterations. However, initialization at or close to one of the adversarial models continues to cause IRLS to perform poorly.

Figure 6.2 shows instances where IRLS fails to converge to the gold model against a partially adaptive adversary on 2D covariates. The figures show a point plot with each point indicating an initialization point for IRLS and that point colored red if IRLS failed to assure $\|\mathbf{w}_{\text{IRLS}} - \mathbf{w}^*\|_2 \leq 10^{-5}$ within 25 iterations, and green if IRLS did manage to assure a reasonably close estimate of the gold model within those many iterations.

We noted in our experiments that it was easier to lure IRLS into poor performance if the adversarial models were numerous, as well as if the adversarial models had smaller L_2 norms compared to the gold model, possibly since IRLS first attempted to converge to models with smaller norms which nevertheless fit some of the points. We also note that if we ran IRLS for several more iterations then more and more initializations start converging to the gold model. However, these solutions converged at a rather slow rate.

Chapter 7

Conclusion and Future Direction

In this thesis, we undertook a case study of the popular IRLS algorithm for the robust regression problem. We saw how the algorithm can be variously seen as performing alternating minimization, or majorization minimization. We saw that the algorithm relates to other robust regression algorithms proposed recently such as `TORRENT`. We performed an empirical evaluation of the algorithm and although, we found the algorithm to succeed in general and offer admirable convergence rates, we also identified failure cases, which we demonstrated explicitly in experiments. To avoid these failure cases, we offered a local convergence guarantee for a truncated version of the algorithm which puts an upper limit on the magnitude of the weights given to any data point. Although these investigations give us insight into the IRLS algorithm, there are several avenues to be explored.

7.1 Regularized IRLS

It is notable that we have considered only unregularized versions of IRLS in this work, relying instead on the data offering (weighted) strong convexity which itself acts as a regularizer. However, in ill-conditioned settings, an explicit regularizer would help. It would be interesting to explore the convergence properties of L1/L2 regularized IRLS formulations.

7.2 Truncated IRLS

In experiments, we observed that in the partly adaptive adversary settings, unless the IRLS procedure was initialized extremely close to one of the adversarially chosen models $\tilde{\mathbf{w}}_j$, the algorithm did converge to the gold model $\mathbf{w}^*$, although it did take a very large number of iterations to do so at times. This motivated us to analyze the IRLS algorithm for a local convergence guarantee which we were able to show for a truncated version of the algorithm where the weights are capped.

However, this truncation comes at a price – it seems to put a limit on how closely IRLS can

approach the gold model, as our theoretical guarantees also suggest. Thus, if we set a cap that is too small, then we also incur a commensurately large model estimation error. Given this, it is tempting to investigate a scheme which iteratively increases the cap, starting with a modest cap. We believe such an algorithm should be able to exploit the convergence properties of the capped IRLS algorithm, as well as have fine control on the extra error incurred due to truncation, and may indeed offer global convergence results.

7.3 IRLS for Sparse Recovery

The IRLS algorithm is actually very popular for the sparse recovery problem [12] where the goal is to perform linear regression to recover a model parameter that is known to be sparse (or its best sparse approximation). Given our progress in analyzing IRLS for the robust regression problem which can also be seen as a sparse recovery problem, but with sparsity on the data side rather than the model side (recall that the corruption vector $\mathbf{b}$ is sparse), it is tempting to explore if similar techniques can be used to analyze IRLS for sparse recovery as well, especially for L_p regularized problems where $p < 1$. Another nice goal is to attempt high dimensional robust regression with IRLS which several previous works have attempted [9, 5, 32, 22, 23] but using different techniques.

7.4 Gradient IRLS

In its current form, IRLS is a computationally expensive algorithm, having to invert a $d \times d$ matrix at each iteration, in order to perform the weighted least squares computation. Although conjugate gradient methods can reduce this cost somewhat, this may still be expensive, especially since the weights keep changing at every iteration. Does there exist a gradient descent version of IRLS which performs much cheaper operations at each time step, preferably with linear time complexity? Preliminary experiments seem to indicate that a naive implementation that just does a gradient step at each iteration instead of a weighted least squares step is flawed. We desire a greater understanding of these cheaper implementations.

Bibliography

[1] Yasin Abbasi-Yadkori, David Pal, and Csaba Szepesvari. Improved Algorithms for Linear Stochastic Bandits. In *Proceedings of the 25th Annual Conference on Neural Information Processing Systems (NIPS)*, 2011.

[2] Khurrum Aftab and Richard Hartley. Convergence of Iteratively Re-weighted Least Squares to Robust M-Estimators. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 480–487. IEEE, 2015.

[3] Marco Barreno, Blaine Nelson, Russell Sears, Anthony D Joseph, and J Doug Tygar. Can Machine Learning Be Secure? In *Proceedings of the ACM Symposium on Information, Computer and Communications Security (ASIACCS)*, pages 16–25. ACM, 2006.

[4] Kush Bhatia, Prateek Jain, Parameswaran Kamalaruban, and Purushottam Kar. Consistent Robust Regression. In *Proceedings of the 31st Annual Conference on Neural Information Processing Systems (NIPS)*, pages 2107–2116, 2017.

[5] Kush Bhatia, Prateek Jain, and Purushottam Kar. Robust Regression via Hard Thresholding. In *Proceedings of the 29th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 721–729, 2015.

[6] Moses Charikar, Jacob Steinhardt, and Gregory Valiant. Learning from untrusted data. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pages 47–60, New York, NY, USA, 2017. ACM.

[7] Rick Chartrand and Valentina Staneva. Restricted isometry properties and nonconvex compressive sensing. *Inverse Problems*, 24(3):035020, 2008.

[8] Rick Chartrand and Wotao Yin. Iteratively reweighted algorithms for compressive sensing. In *Proceedings of the IEEE international conference on Acoustics, speech and signal processing (ICASSP)*, pages 3869–3872. IEEE, 2008.

[9] Yudong Chen, Constantine Caramanis, and Shie Mannor. Robust Sparse Regression under Adversarial Corruption. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, pages 774–782, 2013.

[10] William S Cleveland. Robust Locally Weighted Regression and Smoothing Scatterplots. *Journal of the American Statistical Association*, 74(368):829–836, 1979.

[11] Arnak Dalalyan and Yin Chen. Fused sparsity and robust estimation for linear models with unknown variance. In *Advances in Neural Information Processing Systems*, pages 1259–1267, 2012.

[12] Ingrid Daubechies, Ronald DeVore, Massimo Fornasier, and C Sinan Güntürk. Iteratively Reweighted Least Squares Minimization for Sparse Recovery. *Communications on Pure and Applied Mathematics*, 63(1):1–38, 2010.

[13] Peter J Green. Iteratively Reweighted Least Squares for Maximum Likelihood. Estimation, and some Robust and Resistant Alternatives. *Journal of the Royal Statistical Society. Series B (Methodological)*, 46(2):149–192, 1984.

[14] Ling Huang, Anthony D Joseph, Blaine Nelson, Benjamin IP Rubinstein, and JD Tygar. Adversarial machine learning. In *Proceedings of the 4th ACM Workshop on Artificial Intelligence & Security (AISec)*, pages 43–58. ACM, 2011.

[15] Peter J Huber. Robust Estimation of a Location Parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964.

[16] Peter J Huber. Robust Statistics. In *International Encyclopedia of Statistical Science*, pages 1248–1251. Springer, 2011.

[17] Prateek Jain, Purushottam Kar, et al. Non-Convex Optimization for Machine Learning. *Foundations and Trends® in Machine Learning*, 10(3-4):142–336, 2017.

[18] Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. Low-rank Matrix Completion using Alternating Minimization. In *Proceedings of the 45th annual ACM Symposium on Theory of Computing (STOC)*, 2013.

[19] Ming-Jun Lai and Jingyue Wang. An Unconstrained l_q Minimization with $q \leq 1$ for Sparse Solution of Underdetermined Linear Systems. *SIAM Journal on Optimization*, 21(1):82–101, 2011.

[20] Ming-Jun Lai, Yangyang Xu, and Wotao Yin. Improved Iteratively Reweighted Least Squares for Unconstrained Smoothed ℓ_q Minimization. *SIAM Journal on Numerical Analysis*, 51(2):927–957, 2013.

[21] Adrien Marie Legendre. *Nouvelles méthodes pour la détermination des orbites des comètes*. F. Didot, 1805.

[22] Nam H Nguyen and Trac D Tran. Exact recoverability from dense corrupted observations via l_1-minimization. *IEEE transactions on information theory*, 59(4):2017–2035, 2013.

[23] Nam H Nguyen and Trac D Tran. Robust lasso with missing and grossly corrupted observations. *IEEE Transactions on Information Theory*, 59(4):2036–2058, 2013.

[24] Dianne P O'Leary. Robust Regression Computation Using Iteratively Reweighted Least Squares. *SIAM Journal on Matrix Analysis and Applications*, 11(3):466–480, 1990.

[25] Michael Robert Osborne. *Finite Algorithms in Optimization and Data Analysis*. John Wiley & Sons, Inc., 1985.

[26] Damian Straszak and Nisheeth K Vishnoi. IRLS and Slime Mold: Equivalence and Convergence. In *Innovations in Theoretical Computer Science (ITCS)*, 2017.

[27] James O Street, Raymond J Carroll, and David Ruppert. A Note on Computing Robust Regression Estimates Via Iteratively Reweighted Least Squares. *The American Statistician*, 42(2):152–154, 1988.

[28] Jiawei Su, Danilo Vasconcellos Vargas, and Sakurai Kouichi. One pixel attack for fooling deep neural networks. arXiv:1710.08864 [cs.LG], 2017.

[29] Robert Tibshirani. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

[30] Joel A Tropp et al. An introduction to matrix concentration inequalities. *Foundations and Trends® in Machine Learning*, 8(1-2):1–230, 2015.

[31] John W Tukey. *Contributions to Probability and Statistics: essays in honor of Harold Hotelling*, chapter A survey of sampling from contaminated distributions, pages 448–485. Stanford University Press, 1960.

[32] John Wright and Yi Ma. Dense Error Correction Via l_1-Minimization. *IEEE Transactions on Information Theory*, 56(7):3540–3560, 2010.

[33] Huan Xu, Constantine Caramanis, and Shie Mannor. Robust Regression and Lasso. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems (NIPS)*, pages 1801–1808, 2009.

[34] Hsiang-Fu Yu, Prateek Jain, Purushottam Kar, and Inderjit Singh Dhillon. Large-scale Multi-label Learning with Missing Labels. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, 2014.