

Provable and Efficient Algorithms for Federated, Batch and Reinforcement Learning

by

Avishek Ghosh

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering – Electrical Engineering and Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Kannan Ramchandran, Co-chair

Professor Aditya Guntuboyina, Co-chair

Professor Thomas Courtade

Professor Anil Aswani

Spring 2021

Provable and Efficient Algorithms for Federated, Batch and Reinforcement Learning

Copyright 2021
by
Avishek Ghosh

Abstract

Provable and Efficient Algorithms for Federated, Batch and Reinforcement Learning

by

Avishek Ghosh

Doctor of Philosophy in Engineering – Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Kannan Ramchandran, Co-chair

Professor Aditya Guntuboyina, Co-chair

We propose and analyze iterative algorithms that are computationally efficient, statistically sound and adaptive (in some settings). We consider three different frameworks in which data is presented to the learner. First, we consider the Federated (Distributed) Learning (FL) setup, where data is only available at the edge, and a center machine learns various models via iteratively interacting with the edge nodes. Second, we study the canonical setting of supervised batch learning, where all the data and label pairs are available to the learner at the beginning. Third, we examine the framework of online learning, where data is presented in a streaming fashion. In particular, we focus on specific settings like Bandit and Reinforcement Learning (RL).

In the Federated Learning (FL) framework, we address the canonical problems of device heterogeneity, communication bottleneck and adversarial robustness for large scale high dimensional problems. We propose efficient and provable first and second order algorithms, and use ideas like quantization of information and apply several robust aggregation schemes to address the above-mentioned problems, while retaining the optimal statistical rates simultaneously. For the (supervised) batch learning framework, we use an efficient and statistically sound algorithm, namely Alternating Minimization (AM) and address the problem of max-affine regression; a non convex problem that generalizes the classical phase retrieval and closely resembles convex regression. We give convergence guarantees of AM, with near optimal statistical rate. Finally, in the online learning setup, we address the problem of adaptation (model selection) for contextual bandits (linear and beyond) and later extend these techniques to Reinforcement Learning (RL). Our algorithms here are efficient, provable and more importantly adaptive to the problem complexity.

Dedicated to Maa and Baba

Contents

Contents	ii
1 Introduction	1
1.1 Algorithms for Federated (Distributed) Learning (FL)	1
1.2 Learning in Supervised Batch Setup	7
1.3 Efficient and Adaptive Algorithms for Bandit and Reinforcement Learning (RL) . .	10
I Learning in Federated Framework	16
2 Clustered Federated Learning	17
2.1 Introduction	17
2.2 Related work	18
2.3 Problem formulation	19
2.4 Algorithm	20
2.5 Theoretical guarantees	22
2.6 Experiments	26
2.7 Conclusion and Open Problems	29
2.8 Proof of Theorem 1	30
2.9 Proof of Theorem 2	35
3 Communication-Efficient and Byzantine-Robust Distributed First Order Learning	41
3.1 Introduction	41
3.2 Problem Formulation	45
3.3 Compression At Worker Machines	46
3.4 Robust Compressed Gradient Descent	47
3.5 Distributed Learning with Restricted Adversaries	48
3.6 Distributed Optimization with Arbitrary Adversaries	51
3.7 Byzantine Robust Distributed Learning with Error Feedback	52
3.8 Experiments	57
3.9 Conclusion and Open Problems	60
3.10 Analysis of Algorithm 4	61

3.11	Proof of Theorem 5	71
4	Efficient and Robust FL with Newton Algorithm	80
4.1	Introduction	80
4.2	Problem Formulation	83
4.3	COMRADE Can Communicate Less	83
4.4	COMRADE Can Resist Byzantine Workers	86
4.5	COMRADE Can Communicate Even Less and Resist Byzantine Workers	87
4.6	Experimental Results	88
4.7	Conclusion	89
4.8	Analysis of Section 4.3	90
4.9	Analysis of Section 4.4	97
4.10	Auxiliary Lemmas	100
4.11	Analysis of Section 4.5	102
5	Escaping Saddle Points in FL: Distributed Cubic Regularized Newton Method	108
5.1	Introduction	108
5.2	Related Work	111
5.3	Problem Formulation	112
5.4	Distributed Cubic Regularized Newton	113
5.5	Byzantine Resilience	116
5.6	Experimental Results	118
5.7	Proofs	121
II	Efficient and Tractable Algorithms for Non-Convex Batch Learning	136
6	Max Affine Regression with Gaussian Design	137
6.1	Introduction	137
6.2	Background and problem formulation	143
6.3	Main results	148
6.4	Discussion	157
6.5	Proofs	160
6.6	Proof of Theorem 12	163
6.7	Proof of Theorem 13	179
6.8	Proof of Theorem 14	185
6.9	Technical Lemmas and Background	188
7	Max-affine Regression Beyond Gaussian Design—Small Ball covariates	207
7.1	Introduction	207
7.2	Main results	210
7.3	Discussion and Open Problems	216

7.4	Proof of Theorem 18	217
7.5	Proof of Corollary 3	227
7.6	Technical Lemmas and Background	227

III Learning and Adaptation in Bandit and RL Framework 234

8	Problem-Complexity Adaptive Model Selection for Stochastic Linear Bandits	235
8.1	Introduction	235
8.2	Related Work	238
8.3	Norm as a measure of Complexity	239
8.4	Dimension as a Measure of Complexity - Continuum Armed Setting	242
8.5	Dimension as a Measure of Complexity - Finite Armed Setting	245
8.6	Simulations	246
8.7	Conclusion	248
8.8	Proofs	249
9	Model Selection for Contextual Bandits Beyond Linear Structure	264
9.1	Problem formulation	264
9.2	Algorithm—Adaptive Contextual Bandits (ACB)	266
9.3	Explore-then-Commit algorithm for model selection	268
9.4	Model Selection with infinite function classes	271
9.5	Proofs	274
10	Model Selection for Reinforcement Learning with Function Approximation	283
10.1	Formulation and Linear Model	283
10.2	Norm as Complexity measure	285
10.3	Dimension as complexity measure	287
10.4	Conclusion	290
10.5	Proofs	290

Bibliography 300

Acknowledgments

I would like to acknowledge the people who have played a crucial part toward this thesis. This thesis would not have been possible without their help and support over the years. This list is nowhere comprehensive, and there are many whose presence I appreciated greatly but did not get a chance to thank here.

I would like to express my heartfelt gratitude to my PhD co-advisors, Prof. Kannan Ramchandran and Prof. Aditya Guntuboyina. Kannan has been a great support and inspiration for me throughout my doctoral journey. First, I was amazed by his dedication as a teacher. I had the good opportunity to work as a teaching assistant with him in an undergraduate probability and a graduate information theory course, and the amount of work he puts towards improving the courses is simply mind blowing. Furthermore, he was always available for stimulating discussion sessions on research, brainstorming about new directions to pursue. He is always open to explore new areas, and motivated me to do the same. Kannan gave me the freedom to explore areas I like, even when it fell outside his research interest. Many of the problems I have in this thesis actually started from our brainstorming sessions. One thing I learnt from Kannan is that, it is of fundamental interest to ask the correct set of questions, even though the answers to them are partially known. He also helped me to improve my presentation skills and taught me how to keep the audience engaged. His enthusiasm for research and devotion to his work are forever going to be an inspiration for me in my career and personal life.

I would like to extend my gratitude to Prof. Aditya Guntuboyina. Most of my research activities can be attributed to a course Aditya taught, Theoretical Statistics (Stat. 210 B), in Spring, 2017. I have started working with Aditya immediately afterwards. Having insightful discussions with Aditya on research problems helped me a lot. While Kannan is mostly about big pictures, Aditya is interested in the technical side of things, and I have the good fortune to work with both of them. I got to experience the *best of both worlds*. Whenever I had a doubt, Aditya was always available for meeting, and even when my doubts are trivial or even sometimes stupid, he always encouraged me.

I would also like to acknowledge the contributions of the amazing faculties who inspired me over the years. I benefited a lot from taking fundamental courses on probability, optimization and statistics from Prof. Venkat Anantharam, Prof. Martin Wainwright and Prof. Aditya Guntuboyina. I would also like to take this opportunity to thank the members of my quals as well as thesis committee—Prof. Thomas Courtade, Prof. Anil Aswani, Prof. Kannan Ramchandran and Prof. Aditya Guntuboyina for giving me valuable feedback and support. I would also like to acknowledge my masters advisors, Prof. Anurag Kumar and Prof. Aditya Gopalan of Indian Institute of Science, Bangalore who believed in me and gave me the opportunity to work in his group, and fortified my interest in academic research by being an amazing advisor and person alike.

Over my graduate studies, I had the privilege to work with extraordinarily talented people on research projects, and this thesis would not have been possible without their integral contributions. In my initial years, I started collaborating with Dong Yin and Ashwin Pananjady. I learnt a lot from them, and had enlightening discussions. In the later years, I had the good fortune of working with Prof. Arya Mazumdar, Raj Kumar Maity and Abishek Sankararaman. I really enjoyed our virtual brainstorming sessions, and their deep knowledge on areas like Distributed Learning and Bandits really helped me a lot. Furthermore, I have had the good fortune to work with students and postdocs

in and outside Berkeley—Swanand Kadhe, Sinho Chewi, Jichan Chung, Sayak Ray Chowdhury, Forest Yang, Abhay Parekh, Vipul Gupta and Justin Hong to name a few. I would also like to thank my mentor Dean Foster at Amazon SCOT, with whom I interned in the summer of 2020.

PhD is a long journey, and one definitely needs friends to make this journey smooth. I would like to thank my awesome friends at BLISS Lab—Vipul Gupta, Soham Phade, Dong Yin, Nived Rajaraman, Abishek Sankararaman, Banghua Zhu, Orhan Ocal, Vidya Muthukumar, Ashwin Pananjady, Payam Delgosh, Raaz Dwivedi, Koulik Khamaru, Sang Min Han, Jichan Chung, Kuan-Yun Lee, Amirali Aghazadeh, Wenglong Mou, Efe Aras, Banghua Zhu and Yaoqing Yang. They were great sources of support, knowledge, and friendship. Engaging coffee breaks and delightful discussions with them always lightened my mood. I will miss going to impromptu hiking trips and dinners with my friends at BLISS. I would also like to thank my friends outside BLISS—Sourav Ghosh, Hari Prasanna Das, Santanu Maity, Priyanka Roy and Milind Hedge, with whom I had an awesome time. Furthermore, I would also like to take this opportunity to have a special shout out to the friendly and extremely helpful staff of EECS at Berkeley. In particular, Shirley Salanio has made my (and everybody's) life at EECS better; whenever I needed help, I knew I could get timely support Shirley that would solve any bureaucratic issue I had, and also hear her nice words, which would brighten any day.

Last but not the least, I would like to thank my parents Dipika Ghosh and Asit Ghosh, to whom I dedicate this thesis. Their love and support always stayed with me, and worked as a driving force in my PhD journey. They taught me the importance of education and always motivated me to pursue a career in academics. I owe all my achievements to them and their unconditional love and support.

Chapter 1

Introduction

In recent years, we are witnessing an unprecedented growth in the amount of high dimensional data being sensed, collected and processed to drive modern applications across many domains. This growth has challenged classical algorithms to scale up to big-data regimes. Furthermore, in time sensitive applications such as control and decision-making, collaborative learning, and medical imaging, efficient implementations of such *scaled-up* algorithms are necessary. Moreover, in applications like personalized recommendation and advertisement placement, adapting to the structure of the problem is of fundamental interest. This *adaptation* can reduce both the problem complexity and compute time. In this thesis, we focus on three fundamental aspects of machine learning: *statistical guarantees*, *efficient implementations* and *adaptation to problem complexity*. We consider three different frameworks in which data is presented to the learner: (a) Federated (Distributed) Learning (FL), (b) supervised batch learning and (c) online (streaming) setup with Bandit/Reinforcement Learning (RL).

1.1 Algorithms for Federated (Distributed) Learning (FL)

In many real-world applications, the size of training datasets has grown significantly over the years to the point that it is becoming crucial to implement learning algorithms in a distributed fashion. A commonly used distributed learning framework is data parallelism, in which large-scale datasets are distributed over multiple *worker machines* for parallel processing in order to speed up computation. In many applications, data are stored in end users' own devices such as mobile phones and personal computers, and in these applications, fully utilizing the on-device machine intelligence is an important direction for next-generation distributed learning. Federated Learning (FL) [1–3] is a recently proposed distributed computing paradigm that is designed towards this goal, and has received significant attention. Many statistical and computational challenges arise in Federated Learning, due to the highly decentralized system architecture.

In a standard Federated (or distributed) learning framework, a set of worker machines store the data, perform local computations, and communicate local updates (ex. gradients, Hessians) to the center machine (e.g., a parameter server). The center machine processes the results from workers

to update the model parameters. Such distributed frameworks need to address the following three fundamental challenges:

1. *Data Heterogeneity*: First, the heterogeneity of data across machines can often dampen the learning rate. This is a major concern in Federated Learning, since the worker machines are end users' personal devices and the data across users are different (in distribution) from one another.
2. *Communication Cost*: Second, the gains due to parallelization are often bottlenecked in practice by heavy communication overheads between workers and the central machine. This is especially the case for large clusters of worker machines or for modern deep learning applications using models with millions of parameters (for example, NLP models, such as BERT [4], may have well over 100 million parameters). Moreover, in Federated Learning, communication from a user device to the central server is directly tied to the user's upload bandwidth costs. Therefore, it is of paramount importance to reduce communication overhead in distributed learning algorithms.
3. *Adversarial Robustness*: Third, messages from workers are susceptible to errors due to hardware faults or software bugs, stalled computations, data crashes, and unpredictable communication channels. In scenarios such as Federated Learning, users may as well be malicious and act adversarially. The inherent unpredictable (and potentially adversarial) nature of compute units is typically modeled as *Byzantine failures*. Even if a single worker is Byzantine, it can be fatal to most learning algorithms ([5]).

In Chapters 2-5, we address these fundamental issues and propose algorithms that are provable and computation friendly. In Chapter 2, we propose an iterative clustering algorithm that partitions the worker machines and learns the optimal model for each clusters simultaneously, yielding an optimal statistical rate in certain settings. In Chapter 3 we propose first order optimization algorithms to deal with communication efficiency and Byzantine attacks, and show the optimality of our proposed scheme in certain parameter regime. Furthermore, in Chapters 4 and 5, we extend this to second order and beyond second order algorithms.

1.1.1 Data Heterogeneity: Clustered FL

Exploiting data heterogeneity is particularly crucial in applications such as recommendation systems and personalized advertisement placement, and it benefits both the users' and the enterprises. For example, mobile phone users who read news articles may be interested in different categories of news like politics, sports or fashion; advertisement platforms might need to send different categories of ads to different groups of customers. These indicate that leveraging the heterogeneity among the users is of potential interest—on the one hand, each machine itself may not have enough data and thus we need to better utilize the similarity among the users; on the other hand, if we treat the data from all the users as i.i.d. samples, we may not be able to provide personalized predictions. This problem has recently received much attention [6–8].

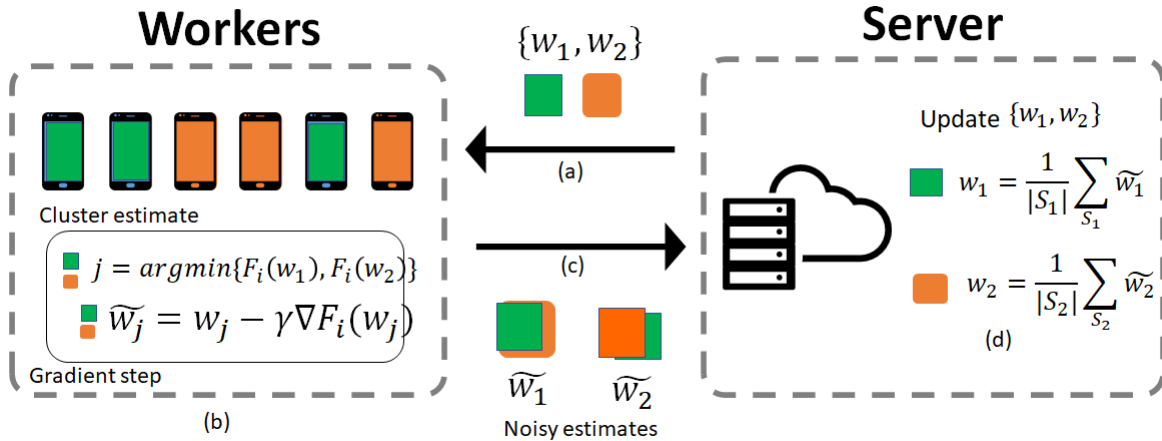


Figure 1.1: An overview of IFCA with 2 clusters, orange and green: (a) The server broadcast models. (b) Worker machines identify their cluster memberships and run local updates. (c) The worker machines send back the local models to server. (d) Average the models within the same estimated cluster S_j .

In Chapter 2, we study one of the formulations of FL with non-i.i.d. data, i.e., the *clustered Federated Learning* [7, 9]. We assume that the users are partitioned into different clusters; for example, the clusters may represent groups of users interested in politics, sports, etc, and our goal is to train models for every cluster of users. We note that cluster structure is very common in applications such as recommender systems [10, 11]. The main challenge of our problem is that the *cluster identities of the users are unknown*, and we have to simultaneously solve two problems: identifying the cluster membership of each user and optimizing each of the cluster models in a distributed setting.

We assume that there are k different data distributions, D_1, \dots, D_k , and that the m machines are partitioned into k disjoint clusters, S_1, \dots, S_k . We assume no knowledge of the cluster identity of each machine, i.e., the partition S_1, \dots, S_k is not revealed to the learning algorithm. We assume that every worker machine $i \in S_j$ contains n i.i.d. data points $z^{i,1}, \dots, z^{i,n}$ drawn from D_j , where each data point $z^{i,j}$ consists of a pair of feature and response denoted by $z^{i,j} = (x^{i,j}; y^{i,j})$.

Let $f(w; z) : W \rightarrow \mathbb{R}$ be the loss function associated with data point z , where $W \subseteq \mathbb{R}^d$ is the parameter space. We choose $W = \mathbb{R}^d$. Our goal is to minimize the population loss function

$$F^j(w) := \mathbb{E}_z \sim D_j [f(w; z)]$$

for all $j \in [k]$. In particular, we try to find solutions $w_j, g_{j=1}^k$ that are close to $w_j = \operatorname{argmin}_{w \in W} F^j(w)$, $j \in [k]$.

In order to achieve this goal, we propose a framework and analyze a distributed method, named the *Iterative Federated Clustering Algorithm (IFCA)* for clustered FL. The basic idea of our algorithm is a strategy that alternates between estimating the cluster identities and minimizing the loss functions, and thus can be seen as an Alternating Minimization algorithm in a distributed

setting. The details of the algorithm is given in Figure 1.1 for two clusters, green and orange. With the broadcasted models, the workers perform an objective based clustering to estimate their cluster identity, and correspondingly update the model by taking gradient steps. Furthermore, when the cluster structure is ambiguous, we propose to leverage the weight sharing technique in multi-task learning [12] and combine it with IFCA. More specifically, we learn the shared representation layers using data from all the users, and use IFCA to train separate final layers for each individual cluster.

We further establish convergence rates of our algorithm, for strongly convex losses under the assumption of good initialization. We prove exponential convergence speed, and for both settings, we can obtain *near optimal* statistical error rates in certain regimes. With quadratic loss, we show that the statistical error rate of our problem is $\mathcal{O}(\frac{d}{mn} + \text{minor-term})$, which is near optimal (see [13]). For strongly convex loss, the error rate obtained is $\mathcal{O}(\frac{d}{mn} + \frac{1}{n})$, which is sub-optimal by a factor of $\frac{1}{n}$. Furthermore, extensive experiments on MNIST, CIFAR-10 and Federated EMNIST data-sets validate our theoretical findings.

1.1.2 Communication Efficiency and Byzantine Resilience

In Chapters 3-5, we address the challenges of communication efficiency and Byzantine resilience (against adversarial machines) simultaneously. As mentioned earlier, Chapter 3 introduces first order gradient based methods, and Chapters 4 and 5, extends those to second order and beyond second order methods respectively.

Gradient Based (First Order) Methods

Both the challenges of communication efficiency and Byzantine-robustness, have recently attracted significant research attention, albeit mostly separately. In particular, several recent works have proposed various quantization or sparsification techniques to reduce the communication overhead ([14–16]). The goal of these quantization schemes is to compute an unbiased estimate of the gradient with bounded second moment in order to achieve good convergence guarantees. The problem of developing Byzantine-robust distributed algorithms has been considered in [17–19].

Once again, in Chapter 3, we consider m worker machines, each storing n data points. The data points are generated from some unknown distribution D . The objective is to learn a parametric model that minimizes a non-convex population loss function $F : W \rightarrow \mathbb{R}$, where F is defined as an expectation over D , and $W \subseteq \mathbb{R}^d$ denotes the parameter space. For gradient compression at workers, we consider the notion of a β -approximate compressor from [20] defined below:

Definition 1 (β -Approximate Compressor). *An operator $Q(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}^d$ is defined as β -approximate compressor on a set $S \subseteq \mathbb{R}^d$ if, $\forall x \in S$,*

$$\|Q(x) - x\|^2 \leq (1 - \beta)\|x\|^2;$$

where $\beta \in (0; 1]$ is the compression factor.

Examples of such compressors include sign-based compressors like QSGD ([21]), ℓ_1 -QSGD ([20]) and top- k sparsification ([15]). Furthermore, for Byzantine robustness, we assume that $0 < \alpha < 1$ fraction of the worker machines are Byzantine, and make no assumptions on the behavior of the Byzantine workers. Byzantine workers can send any arbitrary values to the center machine. In addition, they may completely know the learning algorithm and are allowed to collude with each other. In contrast to blind multiplicative adversaries assumed in literature (see [22]), we consider unrestricted adversaries.

We propose a communication-efficient and robust distributed gradient descent (GD) algorithm. The algorithm takes as input the gradients compressed using a β -approximate compressor along with their norms, and performs a simple thresholding operation on based on gradient norms to discard $\beta > \alpha$ fraction of workers with the largest norm values. We show that our algorithm achieves the following statistical error rate¹ for the regime $\beta > 4 + 4\sqrt{8d^2 + 4d^3}$:

$$\mathcal{O}\left(d^2 \frac{2}{n} + \frac{1}{n} + \frac{1}{mn}\right) \quad (1.1)$$

We first note that when $\beta = 1$ (uncompressed), the error rate is $\mathcal{O}\left(d^2\left[\frac{2}{n} + \frac{1}{mn}\right]\right)$, which matches the optimal rate of [23]. Furthermore, for a fixed d and the compression factor β satisfying $\beta > 4 + 4\sqrt{8d^2 + 4d^3}$, the rate $\mathcal{O}\left(\frac{2}{n} + \frac{1}{mn}\right)$, is optimal and order-wise identical to the case of no compression [23]. In other words, in this parameter regime, we get compression for free.

Furthermore, we strengthen our learning algorithm by using error feedback to correct the direction of the local gradient (Algorithm 3). We show (both theoretically and via experiments) that using error-feedback with a β -approximate compressor indeed speeds up the convergence rate and attains better (statistical) error rate.

Distributed Approximate Newton Method

An alternative way to reduce communication cost is to use second order optimization algorithms; which are known to converge much faster than their first order counterparts. At the expense of increasing computation cost at the worker machines (which is not an issue in applications like Federated Learning; see [3]), the second order algorithms reduce the number of iterations between the worker and center machines. Indeed, a handful of algorithms has been developed using this philosophy, such as DANE [24], DISCO [25], GIANT [26], DINGO [27], Newton-MR [28], INEXACT DANE and AIDE [29].

One major drawback of the above-mentioned algorithms is that they are prone to adversarial or Byzantine attacks. On the other hand, the Byzantine-robust distributed algorithms (see [18, 19, 23, 30–33]) are all variants of first order gradient based optimization algorithm.

In Chapter 4, we propose COMRADE, a distributed approximate Newton-type algorithm that communicates less and is resilient to Byzantine workers. Specifically, with m worker machines and one center machine, we minimize a regularized convex loss $f : \mathbb{R}^d \rightarrow \mathbb{R}$, which is additive over the available data points. Furthermore, similar to previous chapters, we assume that α fraction of the

¹Note that $\mathcal{O}(\cdot)$ hides multiplicative constants, while $\mathcal{O}(\cdot)$ further hides logarithmic factors.

worker machines are Byzantine, where $\epsilon \in [0; 1/2)$. To the best of our knowledge, this is the first work that addresses the problem of Byzantine resilience in second order optimization.

In our proposed algorithm, the worker machines communicate *only once* per iteration with the center machine. This is in sharp contrast with the state-of-the-art distributed second order algorithms (like GIANT [26], DINGO [27], Determinantal Averaging [34]), which sequentially estimates functions of local gradients and Hessians and communicate them with the center machine. In this way, they end up communicating twice per iteration with the center machine. We show that this sequential estimation is redundant. Instead, in COMRADE, the worker machines only send a d dimensional vector, the product of the inverse of local Hessian and the local gradient. Hence, in this way, we save $O(d)$ bits of communication per iteration. Furthermore, in Section 4.5, we argue that, in order to cut down further communication, the worker machines can even compress the local Hessian inverse and gradient product, using a ϵ -approximate compressor.

For Byzantine resilience, COMRADE employs a simple thresholding policy on the norms of the local Hessian inverse and local gradient product to discard ϵ fraction of workers having largest norm values. Since the norm of the Hessian-inverse and gradient product determines the *amount* of movement for Newton-type algorithms, this norm corresponds to a natural metric for identifying and filtering out Byzantine workers.

We prove the linear-quadratic rate of convergence of our proposed algorithm for strongly convex loss functions. In particular, suppose each worker machines contain S data points; and let $\Delta_t = W_t - W$, where W_t is the t -th iterate of COMRADE, and W is the optimal model we want to estimate. In 4 (Theorem 2), we show that

$$\|\Delta_{t+1}\| \leq \max\{f_t^{(1)}\|\Delta_t\|; g_t^{(2)}\|\Delta_t\|^2\} + \left(\frac{g_t^{(3)}}{t} + \frac{1}{S}\right)$$

where $f_t^{(i)}, g_t^{(i)}$ are quantities dependent on several problem parameters. Notice that the above implies a quadratic rate of convergence when $\|\Delta_t\| \leq \frac{g_t^{(1)}}{g_t^{(2)}}$. Subsequently, when $\|\Delta_t\|$ becomes sufficiently small, the above condition is violated and the convergence slows down to a linear rate. The error-floor, which is $O(1/S)$ comes from the Byzantine resilience subroutine in conjunction with the simultaneous estimation of Hessian and gradient.

Cubic Regularized Newton Method

In Chapter 5, we address the problem of saddle point avoidance in non-convex optimization in a Federated Learning (FL) framework. In order to fit complex machine learning models, one often requires to find local minima of a non-convex loss $f(\cdot)$, instead of critical points only, which may include several saddle points. Training deep neural networks and other high-capacity learning architectures [35, 36] are some of the examples where finding local minima is crucial. [36, 37] shows that the stationary points of these problems are in fact saddle points and far away from any local minimum, and hence designing efficient algorithm that escapes saddle points is of interest. Moreover, in [38, 39], it is argued that saddle points can lead to highly sub-optimal solutions in many problems of interest. This issue is amplified in high dimension as shown in [40], and becomes

the main bottleneck in training deep neural nets. Furthermore, a line of recent work [39, 41, 42], shows that for many non-convex problems, it is sufficient to find a local minimum.

The issue of saddle point avoidance becomes non-trivial in the presence of Byzantine workers. Since we do not assume anything on the behavior of the Byzantine workers, it is certainly conceivable that by appropriately modifying their messages to the center, they can create *fake local minima* that are close to the saddle point of the loss function $f(\cdot)$, and these are far away from the true local minima of $f(\cdot)$. This is popularly known as the *saddle-point attack* (see [31]), and it can arbitrarily destroy the performance of any non-robust learning algorithm.

In Chapter 5, we propose a communication efficient distributed optimization algorithm that escapes the saddle points as well as resists the Byzantine attacks simultaneously. In particular, we consider a variation of the famous cubic-regularized Newton algorithm of Nesterov and Polyak [43]. Being a second order algorithm, it converges very fast compared to its first order counterparts (see [43]). Also, in [43–45], it is shown that cubic-regularized Newton can efficiently escape the saddle points of a non-convex function, by pushing the Hessian towards a positive semi-definite matrix.

A point w is said to satisfy the ρ -second order stationary condition of the loss function $f(\cdot)$ if,

$$\| \nabla f(w) \| \leq \rho \quad \min(\lambda(\nabla^2 f(w))) \geq \rho^-;$$

$\nabla f(w)$ denotes the gradient of the function and $\min(\lambda(\nabla^2 f(w)))$ denotes the minimum eigenvalue of the Hessian of the function. Hence, under the assumption (which is standard in the literature, see [31, 46]) that all saddle points are strict (i.e., $\min(\lambda(\nabla^2 f(w_s))) < 0$ for any saddle point w_s), all second order stationary points (with $\rho = 0$) are local minima, and hence converging to a stationary point is equivalent to converging to a local minima.

We consider a distributed variant of the cubic regularized Newton algorithm. In this scheme, the center machine asks the workers to solve an auxiliary function and return the result. The center machine aggregates the solution of the worker machines and takes a descent step. Furthermore, we use a simple norm-based thresholding approach to robustify the distributed cubic-regularized Newton method. We prove that the algorithm convergence at a rate of $\frac{1}{T^{2/3}}$, which is faster than the first order methods (which converge at $\frac{1}{T}$ rate, see [31]). Hence, the number of iterations (and hence the communication cost) required to achieve a target accuracy is much fewer than the first order methods. Experimental results on LIBSVM ([47]) datasets validate our theoretical findings.

1.2 Learning in Supervised Batch Setup

In the batch setup, the learner has data and labels to begin with (standard supervised learning framework). Here, we work with a computationally efficient and statistically sound Alternating Minimization (AM) algorithm, typically used to solve non-convex problems. In particular, we apply AM to a non-convex problem, namely max-affine regression and obtain a near-optimal statistical rate. Max-affine regression refers to a model where the unknown regression function is modeled as a maximum of k unknown affine functions for a fixed $k \geq 1$. This generalizes linear regression and (real) phase retrieval, and is closely related to convex regression. Working within a non-asymptotic framework, we study this problem in the high-dimensional setting assuming that k is a fixed constant,

and focus on the estimation of the unknown coefficients of the affine functions underlying the model. In Chapters 6 and 7, we analyze max-affine regression under 2 designs respectively: Gaussian and Small-ball.

Formally, max-affine regression implies the following regression model:

$$Y = \max_{1 \leq j \leq k} hX; j i + b_j + \epsilon \tag{1.2}$$

where Y is a univariate response, X is a d -dimensional vector of covariates and ϵ models zero-mean noise that is independent of X . We assume that $k \geq 1$ is a known integer and study the problem of estimating the unknown parameters $\beta_1, \dots, \beta_k \in \mathbb{R}^d$ and $b_1, \dots, b_k \in \mathbb{R}$ from independent observations $(x_1; y_1), \dots, (x_n; y_n)$ drawn according to the model (1.2).

Immediately, one may observe that when $k = 1$, equation (1.2) corresponds to the classical linear regression model. Also, when $k = 2$, the intercepts $b_2 = b_1 = 0$, and $\beta_2 = -\beta_1 = \beta$, we have

$$Y = j hX; i j + \beta \tag{1.3}$$

The problem of recovering β from observations drawn according to the above model is known as (real) phase retrieval—variants of which arise in a diverse array of science and engineering applications [48–51]. Furthermore, since $x \mapsto \max_{1 \leq j \leq k} (hX; j i + b_j)$ is always a convex function, the model (1.2) serves as a parametric approximation to the non-parametric convex regression model

$$Y = f(X) + \epsilon \tag{1.4}$$

where $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is an unknown convex function. It is well known that convex regression suffers from the curse of dimensionality (see, e.g., [52–54]). Hence, one would need to work with more structured sub-classes of convex functions. Since convex functions can be approximated to arbitrary accuracy by maxima of affine functions, it is reasonable to enforce a regularization on the problem by considering only those convex functions that can be written as a maximum of a fixed number of affine functions.

With the augmented notation $\beta_j := (\beta_j; b_j) \in \mathbb{R}^{d+1}$ for $j = 1, \dots, k$ and $(x_i; y_i)$ for $i = 1, \dots, n$, where $x_i := (x_i; 1) \in \mathbb{R}^{d+1}$, we minimize the least squares objective

$$(\beta_1^{(ls)}, \dots, \beta_k^{(ls)}) \in \arg \min_{\beta_1, \dots, \beta_k \in \mathbb{R}^{d+1}} \sum_{i=1}^n y_i - \max_{1 \leq j \leq k} h x_i; \beta_j i^2 \tag{1.5}$$

In order to do this, we use alternating minimization (AM) algorithm of [55], which is an iterative algorithm for estimating the parameters β_1, \dots, β_k . In the t -th iteration of the algorithm, the current estimates $\beta_1^{(t)}, \dots, \beta_k^{(t)}$ are used to partition the observation indices $1, \dots, n$ into k sets $S_1^{(t)}, \dots, S_k^{(t)}$ such that $j \in \arg \max_{u \in [k]} h x_i; \beta_u^{(t)} i$ for every $i \in S_j^{(t)}$. For each $1 \leq j \leq k$, the next estimate $\beta_j^{(t+1)}$ is then obtained by performing a least squares fit (or equivalently, linear regression) to the data $(x_i; y_i); i \in S_j^{(t)}$. Figure 1.2 pictorially represents the working of the AM algorithm.

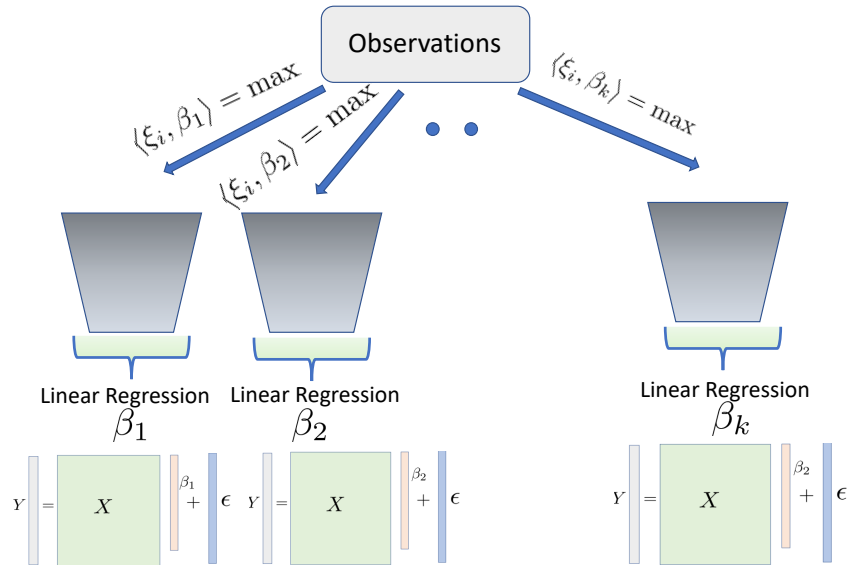


Figure 1.2: Alternating Minimization in action: in the first phase, it disambiguates the max index and partitions the observations in k buckets. Within each bucket, AM solves a linear regression problem to obtain the next iterate.

We show (in Chapter 6) that for each $\delta > 0$, the parameter estimates $\beta_1^{(t)}; \dots; \beta_k^{(t)}$ returned by the AM algorithm at iteration t satisfy, with high probability, the inequality

$$\sum_{j=1}^k \|\beta_j^{(t)} - \beta_j^{(0)}\|_2^2 \leq C(\beta_1, \dots, \beta_k) \frac{2kd \log(kd) \log \frac{n}{kd}}{n} \quad (1.6)$$

for every $t \geq \log_{4/3} \frac{\sum_{j=1}^k \|\beta_j^{(0)}\|_2^2}{\delta^2}$, provided that the sample size n is sufficiently large and that the initial estimates satisfy the condition

$$\min_{c > 0} \max_{1 \leq j \leq k} c \|\beta_j^{(0)}\|_2^2 \leq \frac{1}{k} C(\beta_1, \dots, \beta_k). \quad (1.7)$$

Here $C(\beta_1, \dots, \beta_k)$ and $c(\beta_1, \dots, \beta_k)$ are constants depending only on true parameters β_1, \dots, β_k .

In Chapter 7, we relax the assumption of Gaussian covariates and show that a similar phenomenon occurs under significantly weaker statistical assumptions. In particular, we allow the distribution of the covariates to come from the larger class of sub-Gaussian distributions that satisfy a *small-ball* condition. In addition, we also consider the scenario of *universal* parameter estimation, meaning that our guarantees hold uniformly over all β_1, \dots, β_k . This allows the parameters to be chosen with knowledge of the realized covariates, a robust setting that is commonly studied in signal processing applications like phase retrieval [56]. Our covariate assumption relies on the following definition.

Definition 2. (Small-ball) A distribution P_X satisfies a $(\epsilon; c_s)$ -small-ball property if, for $X \sim P_X$ and each $\epsilon > 0$, we have

$$\sup_{u \in \mathbb{S}^{d-1}; w \in \mathbb{R}} \Pr \left(|hX; u| + w \right)^2 \leq c_s \epsilon^2 \quad (1.8)$$

The small-ball properties of various classes of distributions have been studied extensively in the probability literature [57, 58], and many natural distributions possess this property provided they are not too “peaky”. We now present our assumption on the covariate distribution;

Assumption 1. The covariates are drawn from a distribution, P_X , which is isotropic, ϵ -sub-Gaussian, and satisfies a $(\epsilon; c_s)$ small-ball condition.

We now provide a few examples, where covariates satisfy the above assumption:

(a) $\text{Unif}[\frac{-\epsilon}{3}; \frac{\epsilon}{3}]$, (b) the standard Gaussian and (c) any random variable with density $f(x) \propto e^{-k|x|^c}$ for a positive constant c . In Chapter 7, we discuss these examples in detail and obtain the parameters $(\epsilon; c_s)$ and ϵ .

Applying the same AM algorithm, we obtain guarantees similar to that of Chapter 6. As a special instance, for phase retrieval problem, we obtain some new results. To the best of our knowledge, all previous results on the AM algorithm for phase retrieval [59, 60] only held under the assumptions of Gaussian covariates and noiseless observations, and/or required resampling of the measurements [61]. Ours is thus the first work to handle non-Gaussian covariates in the presence of noise, while also analyzing the algorithm without resampling.

1.3 Efficient and Adaptive Algorithms for Bandit and Reinforcement Learning (RL)

In the online learning framework, data is presented to the user in a streaming fashion, and the job of the learner is to balance between exploration and exploitation judiciously to enforce statistical learning. In this part of the thesis (Chapters 8-9), we provide statistically tractable and efficient algorithms for online learning. In particular, we are interested in the question of model selection or adaptation in online learning. Model selection has been studied in supervised offline setup to a great extent (via popular methods like k -fold cross validation, which has theoretical guarantees as well as practical effectiveness). However, in online learning, the question of model selection has received little interest. Our main contribution in this part of the thesis is to propose an efficient and provable algorithm that adapts to the problem complexity automatically. We study the model selection problem in 3 settings: stochastic linear bandits, contextual bandits (beyond linear structure) and finally Reinforcement learning with linear structure.

In Chapter 8, we consider the standard setup of stochastic linear bandits, where, at each round, having observed a context vector, the learner chooses an action and observes a noisy linearly parameterized reward. We propose a provable, efficient algorithm that captures the complexity of the problem and automatically adapts to it. Via regret analysis, we show optimality of our scheme,

showing that the cost of adaptation is a small additive constant. In Chapter 9, we extend these to the problem of generic contextual bandit beyond linear structure.

Model Selection

Model Selection for Bandits(or Reinforcement) Learning, refers to choosing the appropriate hypothesis class, to model the mapping from arms (actions) to expected rewards. Model selection plays an important role in applications such as personalized recommendations, (see the motivating example in Chapter 8). Moreover, in Reinforcement Learning with huge state and action spaces, it is empirically observed that exploring only a small subset of the state and action spaces are often sufficient to obtain high rewards. Hence, in these applications one needs an adaptive algorithm that identifies the implicit structure of the problem and explores or exploits judiciously.

Formally, a family of nested hypothesis classes $H_f, f \in F$ needs to be specified, where each class posits a plausible model for mapping arms (or actions) to expected rewards. The true model is assumed to be contained in the family F which is totally ordered, where if $f_1 \preceq f_2$, then $H_{f_1} \subseteq H_{f_2}$. Model selection guarantees then refers to algorithms whose regret scales in the complexity of the *smallest hypothesis class containing the true model*, even though the algorithm was not aware apriori.

1.3.1 Adaptation in Stochastic Linear Bandits

We consider the problem of *model selection* for two popular stochastic linear bandit settings, and propose algorithms that adapts to the *unknown* problem complexity. In the first setting, we consider the K armed mixture bandits, where the mean reward of arm $i \in [K]^2$, is

$$\mu_i + \langle h_{i;t}, \theta_i \rangle$$

where $h_{i;t} \in \mathbb{R}^d$ is the known context vector of arm i at time t , and $\theta_i \in \mathbb{R}, \|\theta_i\| \leq L$ are unknown and needs to be estimated. This setting also contains the standard Multi-Armed Bandit (MAB) problem [62, 63] when $\theta_i = 0$.

Popular linear bandit algorithms, like LinUCB, OFUL (see [64–66]) handle the case with no bias ($\theta_i = 0$), while OSOM [67], the recent improvement can handle arm-bias. Implicitly, all the above algorithms assume an upper bound on the norm of $\theta_i \leq L$, which is supplied as an input. Crucially however, the regret guarantees scale linearly in the upper bound L . In contrast, we choose $k \leq K$ as the problem complexity and consider a sequence of nested hypothesis classes, each positing a different upper bound on $\|\theta_i\| \leq k$. We propose Adaptive Linear Bandit (ALB), a novel phase based algorithm, that, without any upper bound on the norm $\|\theta_i\| \leq k$, *adapts to the true complexity* of the problem instance, and achieves a regret scaling linearly in the true norm $\|\theta_i\| \leq k$. The details of this algorithm is given in Figure 1.3. It consistently estimates upper-bounds on parameter norm over a confidence ellipsoid given by [67], and we show that the size of this confidence set shrinks with time and as a result, the norm estimates converge to the true norm estimates.

²By $[r]$, we denote the set of positive integers $\{1, 2, \dots, r\}$.

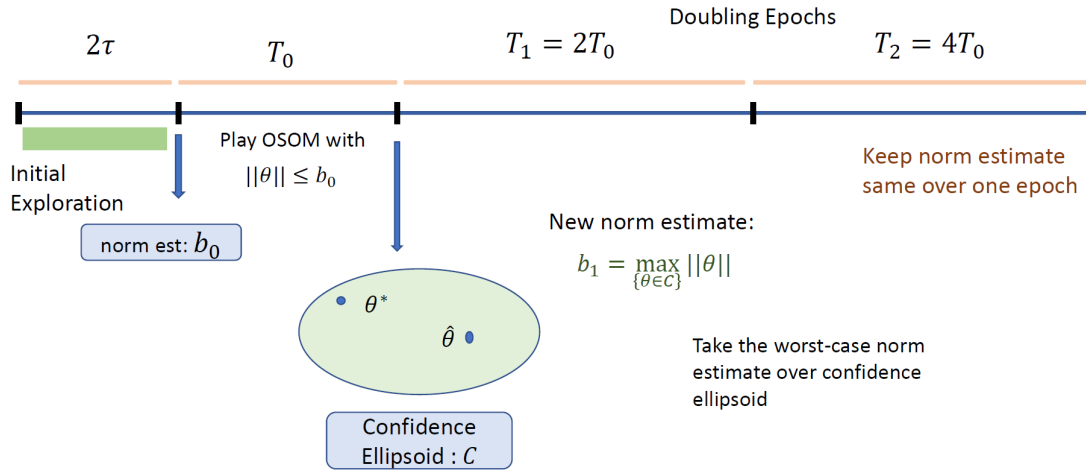


Figure 1.3: Model selection with norm refinement. At the end of each epoch, an over-estimate of norm is computed, and with shrinking confidence set, these norm estimates converge to the true norm of θ^* .

As a corollary, our algorithm’s performance matches the minimax regret of simple MAB when $\beta = 0$, even though the algorithm did not apriori know that $\beta = 0$. Intuitively, if β is very small, the contribution of the contextual term will be low, and on the other hand, a large non zero β , the contextual part dominates. Hence, k can be chosen as a natural complexity parameter. In Chapter 8, we also provide motivations and examples behind the choice of k as a problem complexity parameter, as it balances between the standard MAB and the linear bandit problems.

We show that ALB achieves the regret

$$R(T) = \Theta(k \sqrt{T});$$

where k is apriori unknown, and T is the duration of the interaction between the player and the environment, known as the horizon. As a corollary, when $\beta = 0$, ALB recovers the minimax regret for the simple bandit algorithm without such knowledge of β . ALB is the first algorithm that uses parameter norm as model section criteria for linear bandits. Prior state of art algorithms [67] achieve a regret of $\Theta(L \sqrt{T})$, where L is the upper bound on k , fed as an input to the problem.

In the second setting, we consider is the standard linear stochastic bandit [66] with possibly an infinite number of arms, where the mean reward of any arm $x \in \mathbb{R}^d$ (arms are vectors in this case) given by $\langle \mu, x \rangle$, where $\mu \in \mathbb{R}^d$ is unknown. For this setting, we consider model selection from among a total of d different hypothesis classes, with each class positing a different cardinality for the support of μ . The sparsity of μ , denoted by d , is unknown to the algorithm, and we define d as the problem complexity. We exhibit a novel algorithm, where the regret scales linearly in the unknown cardinality of the support of μ . The regret scaling of our algorithm matches that of an oracle that has knowledge of the optimal support cardinality [68],[69], thereby achieving model selection guarantees. Our algorithm is the first known algorithm to obtain regret scaling matching

that of an oracle that has knowledge of the true support. This is in contrast to standard linear bandit algorithms such as [66], where the regret scales linearly in d . We also extend this methodology to the case when the number of arms is finite (see [70]) and obtain similar regret rates matching the oracle. We further verify through synthetic and real-data experiments that the performance gains are fundamental and not artifacts of mathematical bounds. In particular, we show 1.5–3x drop in cumulative regret over non-adaptive algorithms.

1.3.2 Adaption for Contextual Bandits Beyond Linear Structure

Here, we focus on the main contribution of the paper—a provable model selection guarantee for the (generic) stochastic contextual bandit problem. The interaction between the learner and the environment can be modelled as following:

Let A be the set of K actions, and let X be the set of d dimensional contexts. At time t , Nature picks $(x_t; r_t)$ in an i.i.d fashion, where $x_t \in X$ and a context dependent $r_t \in [0; 1]^K$. Upon observing the context, the learner takes action $a_t \in A$, and obtains the reward of $r_t(a_t)$. We restrict to the class of realizable rewards defined as:

Assumption 2. (Realizability:) *There exists a predictor $f \in F$, such that $E[r_t(a)|x] = f(x; a)$, for all x and a .*

Note that the realizability assumption is standard in the literature ([70, 71]).

In the contextual bandit literature ([71, 72]) it is generally assumed that the true regression function f is unknown, but we know the function class F where it belongs. Hence, we pay some price (denoted by the regret) for this. To set up notation, for any $f \in F$, we define a policy induced by the function, $\pi_f(x) = \operatorname{argmin}_{a \in A} f(x; a)$. So, we need to compete with the policy induced by the true regressor π_f . We define the regret over T rounds as the following:

$$R(T) = \sum_{t=1}^T [r_t(\pi_f(x_t)) - r_t(a_t)];$$

However, in contrast to the standard setting, in the model selection framework does not assume the knowledge of F . Instead, we are given a nested class of M function classes, $F_1 \subset F_2 \subset \dots \subset F_M$. The smallest function class where the true regressor lies is denoted by F_d , where $d \in [M]$. Hence, the regret of the contextual bandit algorithm should depend on F_d (and not the largest class F_M). However, we do not know d , and our goal is to propose adaptive algorithms such that the regret depends on the *actual* problem complexity F_d .

In order to achieve this, we propose an algorithm, namely Adaptive Contextual Bandits (ACB). Our algorithm selects the correct model via successive refinements over multiple (doubling) epochs. We use the FALCON algorithm of [71], and add a model selection phase at the beginning of each epoch. In other words, over multiple epochs, we successively refine our estimates of the *proper* model class where the true regressor function f lies. We show that ACB selects the correct model class with high probability and attains a regret of

$$R(T) \leq \sqrt{KT \log(jF_d jT)} + \text{minor-term};$$

where K is the number of actions, and $|j|$ denotes the cardinality. Although the above expression makes sense only with finite function classes, a simple extension (see Chapter 9) to infinite function classes is possible. Note that, this regret scaling is optimal (see [71]). The cost of model selection therefore is a minor additive term in the regret expression.

1.3.3 Model Selection for Reinforcement Learning

We study the problem of model selection in Reinforcement Learning with function approximation. We restrict our attention to the framework of model based episodic *linear* MDP. In particular, similar to Chapter 8, we study both the *norm* and *sparsity/dimension* of the problem parameter as problem complexity and obtain model selection guarantees, meaning that the performance (regret) of our algorithm depends on the complexity of the problem instance, as compared to some predefined upper-bound on the complexity parameters.

In this chapter, we consider an heterogeneous, episodic Markov decision process denoted by a tuple $M(S; A; H; \{P_h\}_{h=1}^H; \{r_h\}_{h=1}^H)$, where S is the state space, A is the action space, H is the length of each episode, P_h is the transition probability at step h such that $P_h(j|s; a)$ is the distribution over states provided action a is taken for state s at step h , and $r_h : S \times A \rightarrow [0; 1]$ is the deterministic reward function at step h . A policy π is a collection of H functions $f_h : S \rightarrow A$, where each of them maps a state s to an action a .

The RL agent interacts with the environment for K episodes to learn the unknown transition kernels $\{P_h\}$ and hence to improve its policy. For each $k = 1, \dots, K$, the environment picks a starting state s_1^k , and the agent chooses a policy π^k that will be followed through the whole k -th episode. The objective is to design a learning algorithm that constructs a sequence $\{\pi^k\}$ that minimizes the total regret

$$R(K) = \sum_{k=1}^K \sum_{h=1}^H |V_1(s_1^k) - V_1^*(s_1^k)|$$

where $V_1(\cdot)$ is the value function at the beginning of each episode, and $V_1^*(\cdot)$ is the optimal value function. In this chapter, we consider a special class of MDPs called *linear kernel MDPs* (a.k.a., linear mixture MDPs) [73]. Roughly speaking, it means that the transition kernel $\{P_h\}$ can be represented as a linear function of a given feature map $\phi : S \times A \rightarrow \mathbb{R}^d$. Formally, we have the following definition.

Definition 3 (Linear kernel MDP). $M(S; A; H; \{P_h\}_{h=1}^H; \{r_h\}_{h=1}^H)$ is called an heterogeneous, episodic B -bounded linear kernel MDP if there exists a known feature mapping $\phi : S \times A \rightarrow \mathbb{R}^d$ and an unknown vector $\mu_h \in \mathbb{R}^d$ with $\|\mu_h\|_2 \leq B$ such that

(i) For any state-action-state triplet $(s; a; s^0) \in S \times A \times S$ and step $h \in [H]$, $P_h(s^0|s; a) = \sum_{s^0} P_h(s^0|s; a) \mu_h^T \phi(s^0)$.

(ii) For any bounded function $V : S \rightarrow [0; 1]$ and any tuple $(s; a) \in S \times A$, we have $\|V\|_V(s; a) \leq 1$, where $\|V\|_V(s; a) := \sum_{s^0 \in S} P_h(s^0|s; a) V(s^0)$.

In the learning problem, the vectors $f_h g_{h=1}^H$ are unknown to the learner. The episodic MDP is parameterized by $\theta = f_h g_{h=1}^H$ and we denote the MDP by M . In what follows, we define two natural complexity measure of M : (a) $k_h k$ for all $h \in [H]$, and (b) $k_h k_0$ for all $h \in [H]$.

Similar to Chapter 8, we propose algorithms that use successive refinement strategy to estimate the norm and sparsity of $f_h g_{h=1}^H$, and adapts to it. As a result, we obtain regret guarantees that depend on the problem complexity parameter alone. We characterize the cost of model selection, and in both cases we obtain a regret of

$$R(K) = O(\sqrt{K}) + \text{minor-term}$$

with high probability. We observe that $R(K)$ retains the \sqrt{K} regret, which is optimal (see [74]). However, the cost of model selection is the minor term added to $R(K)$. Note that this term is K independent, and hence can be treated as constant. A notable related work, which considers similar model selection problem is [75]. However, the paper obtains a sub-optimal rate of $O(K^{2/3})$. Our main contribution is keeping the optimal rate of $O(\sqrt{K})$, while pushing the cost of model selection in an additive constant independent of K .

Part I

Learning in Federated Framework

In this part of the thesis, we study some problems in Federated Learning (FL). In particular we address (a) data heterogeneity, (b) communication bottleneck and (c) Byzantine robustness.

- **Chapter 2:** We study the FL framework where users are partitioned into clusters, and our proposed algorithm learn the cluster identities and the optimal model for each cluster simultaneously.
- **Chapter 3:** We address the communication bottleneck and robustness issues via proposing first order gradient based algorithms.
- **Chapter 4:** We propose a second order distributed Newton type algorithm to address the communication bottleneck and furthermore, make it Byzantine resilient.
- **Chapter 5:** We target the saddle point avoidance problem in non-convex optimization via proposing an efficient and robust cubic-regularized Newton method.

Chapter 2

Clustered Federated Learning

We address the problem of Federated Learning (FL) where users are distributed and partitioned into clusters. This setup captures settings where different groups of users have their own objectives (learning tasks) but by aggregating their data with others in the same cluster (same learning task), they can leverage the strength in numbers in order to perform more efficient Federated Learning. We propose a new framework dubbed the Iterative Federated Clustering Algorithm (IFCA), which alternately estimates the cluster identities of the users and optimizes model parameters for the user clusters via gradient descent. We analyze the convergence rate of this algorithm first in a linear model with squared loss and then for generic strongly convex and smooth loss functions. We show that in both settings, with good initialization, IFCA converges at an exponential rate, and discuss the optimality of the statistical error rate. When the clustering structure is ambiguous, we propose to train the models by combining IFCA with the weight sharing technique in multi-task learning. In the experiments, we show that our algorithm can succeed even if we relax the requirements on initialization with random initialization and multiple restarts. We also present experimental results showing that our algorithm is efficient in non-convex problems such as neural networks. We demonstrate the benefits of IFCA over the baselines on several clustered FL benchmarks.

2.1 Introduction

In this chapter, we study one of the formulations of FL with non-i.i.d. data, i.e., the *clustered Federated Learning* [7, 9]. We assume that the users are partitioned into different clusters; for example, the clusters may represent groups of users interested in politics, sports, etc, and our goal is to train models for every cluster of users. We note that cluster structure is very common in applications such as recommender systems [10, 11]. The main challenge of our problem is that the *cluster identities of the users are unknown*, and we have to simultaneously solve two problems: identifying the cluster membership of each user and optimizing each of the cluster models in a distributed setting. In order to achieve this goal, we propose a framework and analyze a distributed method, named the *Iterative Federated Clustering Algorithm (IFCA)* for clustered FL. The basic idea of our algorithm is a strategy that alternates between estimating the cluster identities and

minimizing the loss functions, and thus can be seen as an Alternating Minimization algorithm in a distributed setting. One of the major advantages of our algorithm is that it does not require a centralized clustering algorithm, and thus significantly reduces the computational cost at the center machine. When the cluster structure is ambiguous, we propose to leverage the weight sharing technique in multi-task learning [12] and combine it with IFCA. More specifically, we learn the shared representation layers using data from all the users, and use IFCA to train separate final layers for each individual cluster.

We also present experimental evidence of its performance in practical settings: We show that our algorithm can succeed even if we relax the initialization requirements with random initialization and multiple restarts; and we also present results showing that our algorithm is efficient on neural networks. We demonstrate the effectiveness of IFCA on two clustered FL benchmarks created based on the MNIST and CIFAR-10 datasets, respectively, as well as the Federated EMNIST dataset [76] which is a more realistic benchmark for FL and has ambiguous cluster structure.

Here, we emphasize that clustered Federated Learning is not the only approach to modeling the non-i.i.d. nature of the problem, and different algorithms may be more suitable for different application scenarios; see Section 2.2 for more discussions. That said, our approach to modeling and the resulting IFCA framework is certainly an important and relatively unexplored direction in Federated Learning. We would also like to note that our theoretical analysis makes contributions to statistical estimation problems with latent variables in distributed settings. In fact, both mixture of regressions [77] and mixture of classifiers [78] can be considered as special cases of our problem in the centralized setting. We discuss more about these algorithms in Section 2.2.

Notation: We use $[r]$ to denote the set of integers $\{1, 2, \dots, r\}$. We use $\|x\|_k$ to denote the k -norm of vectors. We use $x \asymp y$ if there exists a sufficiently large constant $c > 0$ such that $x \leq cy$, and define $x \lesssim y$ similarly. We use $\text{poly}(m)$ to denote a polynomial in m with arbitrarily large constant degree.

2.2 Related work

During the development of our clustered FL framework, we became aware of a concurrent and independent work by Mansour et al. [9], in which the authors propose clustered FL as one of the formulations for personalization in Federated Learning. The algorithms proposed here and by Mansour et al. are similar. However, we make an important contribution by establishing the *convergence rate* of the *population loss function* under good initialization, which simultaneously guarantees both convergence of the training loss and generalization to test data; whereas in [9], the authors provided only *generalization* guarantees. We discuss other related work in the following.

Federated Learning and non-i.i.d. data: Learning with a distributed computing framework has been studied extensively in various settings [79–81]. As mentioned in Section 2.1, Federated Learning [1–3, 82] is one of the modern distributed learning frameworks that aims to better utilize the data and computing power on edge devices. A central problem in FL is that the data on the users’ personal devices are usually non-i.i.d. Several formulations and solutions have been proposed to tackle this problem. A line of research focuses on learning a single global model from

non-i.i.d. data [83–88]. Other lines of research focus more on learning personalized models [6, 7, 89]. In particular, the MOCHA algorithm [6] considers a multi-task learning setting and forms a deterministic optimization problem with the correlation matrix of the users being a regularization term. Our work differs from MOCHA since we consider a statistical setting with cluster structure. Another approach is to formulate Federated Learning with non-i.i.d. data as a meta learning problem [8, 89, 90]. In this setup, the objective is to first obtain a single global model, and then each device fine-tunes the model using its local data. The underlying assumption of this formulation is that the data distributions among different users are similar, and the global model can serve as a good initialization. The formulation of clustered FL has been considered in two recent works [7, 32]. Both of the two works use *centralized* clustering algorithm such as K -means, in which the center machine has to identify the cluster identities of all the users, leading to high computational cost at the center. As a result, these algorithms may not be suitable for large models such as deep neural networks or applications with a large number of users. In contrast, our algorithm uses a decentralized approach to identify the cluster identities and thus is more suitable for large-scale applications.

Latent variable problems: As mentioned in Section 2.1, our formulation can be considered as a statistical estimation problem with latent variables in a distributed setting, and the latent variables are the cluster identities. Latent variable problem is a classical topic in statistics and non-convex optimization; examples include Gaussian mixture models (GMM) [91, 92], mixture of linear regressions [77, 93, 94], and phase retrieval [95, 96]. Expectation Maximization (EM) and Alternating Minimization (AM) are two popular approaches to solving these problems. Despite the wide applications, their convergence analyses in the finite sample setting are known to be hard, due to the non-convexity nature of their optimization landscape. In recent years, some progress has been made towards understanding the convergence of EM and AM in the centralized setting [61, 97–100]. For example, if started from a suitable point, they have fast convergence rate, and occasionally they enjoy super-linear speed of convergence [91, 101]. Here, we provide new insights to these algorithms in the FL setting.

2.3 Problem formulation

We begin with a standard statistical learning setting of empirical risk minimization (ERM). Our goal is to learn parametric models by minimizing some loss functions defined by the data. We consider a distributed learning setting where we have one center machine and m worker machines (i.e., each worker machine corresponds to a user in the Federated Learning framework). The center machine and worker machines can communicate with each other using some predefined communication protocol. We assume that there are k different data distributions, D_1, \dots, D_k , and that the m machines are partitioned into k disjoint clusters, S_1, \dots, S_k . We assume no knowledge of the cluster identity of each machine, i.e., the partition S_1, \dots, S_k is not revealed to the learning algorithm. We assume that every worker machine $i \in S_j$ contains n i.i.d. data points $z^{i,1}, \dots, z^{i,n}$ drawn from D_j , where each data point $z^{i,j}$ consists of a pair of feature and response denoted by $z^{i,j} = (x^{i,j}; y^{i,j})$.

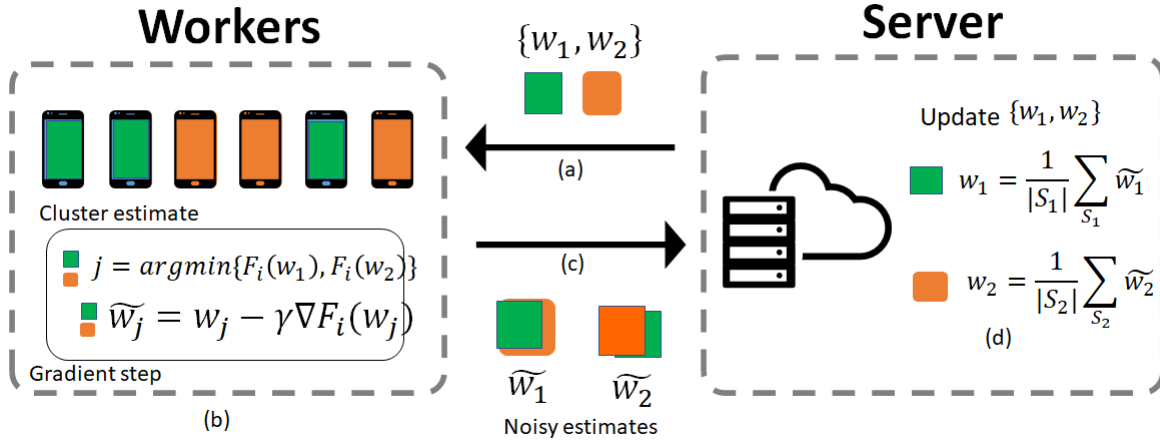


Figure 2.1: An overview of IFCA (model averaging). (a) The server broadcast models. (b) Worker machines identify their cluster memberships and run local updates. (c) The worker machines send back the local models to server. (d) Average the models within the same estimated cluster S_j .

Let $f(w; z) : W \rightarrow \mathbb{R}$ be the loss function associated with data point z , where $W \subseteq \mathbb{R}^d$ is the parameter space. In this paper, we choose $W = \mathbb{R}^d$. Our goal is to minimize the population loss function $F^j(w) := \mathbb{E}_{z \sim D_j}[f(w; z)]$ for all $j \in [k]$. For the purpose of theoretical analysis in Section 2.5, we focus on the strongly convex losses, in which case we can prove guarantees for estimating the unique solution that minimizes each population loss function. In particular, we try to find solutions $\tilde{w}_j, \tilde{g}_{j=1}^k$ that are close to $w_j = \operatorname{argmin}_{w \in W} F^j(w), j \in [k]$. In our problem, since we only have access to finite data, we take advantage of the empirical loss functions. In particular, let $Z = \{z^{i,1}, \dots, z^{i,n}\}$ be a subset of the data points on the i -th machine. We define the empirical loss associated with Z as $F_i(w; Z) = \frac{1}{|Z|} \sum_{z \in Z} f(w; z)$. When it is clear from the context, we may also use the shorthand notation $F_i(w)$ to denote an empirical loss associated with some (or all) data on the i -th worker.

2.4 Algorithm

In this section, we provide details of our algorithm. We name this scheme *Iterative Federated Clustering Algorithm* (IFCA). The main idea is to alternatively minimize the loss functions while estimating the cluster identities. We discuss two variations of IFCA, namely gradient averaging and model averaging. The algorithm is formally presented in Algorithm 1 and illustrated in Figure 2.1.

The algorithm starts with k initial model parameters $w_j^{(0)}, j \in [k]$. In the t -th iteration of IFCA, the center machine selects a random subset of worker machines, $M_t \subseteq [m]$, and broadcasts the current model parameters $\{w_j^{(t)}\}_{j=1}^k$ to the worker machines in M_t . Here, we call M_t the set of *participating devices*. Recall that each worker machine is equipped with local empirical loss function $F_i(\cdot)$. Using the received parameter estimates and F_i , the i -th worker machine ($i \in M_t$) estimates

Algorithm 1: Iterative Federated Clustering Algorithm (IFCA)

```

1: Input: number of clusters  $k$ , step size  $\eta$ ,  $j \in [k]$ , initialization  $w_j^{(0)}$ ,  $j \in [k]$ 
   number of parallel iterations  $T$ , number of local gradient steps  $\ell$  (for model averaging).
2: for  $t = 0; 1; \dots; T - 1$  do
3:   center machine: broadcast  $w_j^{(t)}$ ,  $j \in [k]$ 
4:    $M_t$  random subset of worker machines (participating devices)
5:   for worker machine  $i \in M_t$  in parallel do
6:     cluster identity estimate  $\hat{p} = \operatorname{argmin}_{j \in [k]} F_i(w_j^{(t)})$ 
7:     define one-hot encoding vector  $s_i = \sum_{j=1}^k s_{i,j} g_{j=1}^k$  with  $s_{i,j} = \mathbf{1}\{j = \hat{p}\}$ 
8:     option I (gradient averaging):
9:       compute (stochastic) gradient:  $g_i = \frac{1}{\ell} \sum_{p=1}^{\ell} \nabla F_i(w_{\hat{p}}^{(t)})$ , send back  $s_i, g_i$ 
       to the center machine
10:    option II (model averaging):
11:       $\mathcal{W}_i = \text{LocalUpdate}(w_{\hat{p}}^{(t)}; \eta; \ell)$ , send back  $s_i, \mathcal{W}_i$ 
       to the center machine
12:    end for
13:    center machine:
14:    option I (gradient averaging):  $w_j^{(t+1)} = w_j^{(t)} - \eta \frac{1}{|M_t|} \sum_{i \in M_t} s_{i,j} g_i$ ,  $\forall j \in [k]$ 
15:    option II (model averaging):  $w_j^{(t+1)} = \frac{1}{|M_t|} \sum_{i \in M_t} s_{i,j} \mathcal{W}_i$ ,  $\forall j \in [k]$ 
16:  end for
17: return  $w_j^{(T)}$ ,  $j \in [k]$ 
   LocalUpdate( $w^{(0)}$ ;  $\eta$ ;  $\ell$ ) at the  $i$ -th worker machine
18: for  $q = 0; \dots; \ell - 1$  do
19:   (stochastic) gradient descent  $\mathcal{W}^{(q+1)} = \mathcal{W}^{(q)} - \eta \nabla F_i(\mathcal{W}^{(q)})$ 
20: end for
21: return  $\mathcal{W}^{(\ell)}$ 

```

its cluster identity via finding the model parameter with lowest loss, i.e., $\hat{p} = \operatorname{argmin}_{j \in [k]} F_i(w_j^{(t)})$ (ties can be broken arbitrarily). If we choose the option of gradient averaging, the worker machine then computes a (stochastic) gradient of the local empirical loss F_i at $w_{\hat{p}}^{(t)}$, and sends its cluster identity estimate and gradient back to the center machine. After receiving the gradients and cluster identity estimates from all the participating worker machines, the center machine then collects all the gradient updates from worker machines whose cluster identity estimates are the same and conducts gradient descent update on the model parameter of the corresponding cluster. If we choose the option of model averaging (similar to the Federated Averaging algorithm [3]), each participating device needs to run ℓ steps of local (stochastic) gradient descent updates, get the updated model, and send the new model and its cluster identity estimate to the center machine. The center machine then averages the new models from the worker machines whose cluster identity estimates are the same.

2.4.1 Practical implementation of IFCA

We clarify a few issues regarding the practical implementation of IFCA. In some real-world problems, the cluster structure may be ambiguous, which means that although the distributions of data from different clusters are different, there exists some common properties of the data from all the users that the model should leverage. For these problems, we propose to use the weight sharing technique in multi-task learning [12] and combine it with IFCA. More specifically, when we train neural network models, we can share the weights for the first a few layers among all the clusters so that we can learn a good representation using all the available data, and then run IFCA algorithm only on the last (or last few) layers to address the different distributions among different clusters. Using the notation in Algorithm 1, we run IFCA on a subset of the coordinates of $w_j^{(t)}$, and run vanilla gradient averaging or Federated Averaging on the remaining coordinates. Another benefit of this implementation is that we can reduce the communication cost: Instead of sending k models to all the worker machines, the center machine only needs to send k different versions of a subset of all the weights, and one single copy of the shared layers.

Another technique to reduce communication cost is that when the center machine observes that the cluster identities of all the worker machines are stable, i.e., the estimates of their cluster identities do not change for several parallel iterations, then the center machine can stop sending k models to each worker machine, and instead, it can simply send the model corresponding to each worker machine's cluster identity estimate.

2.5 Theoretical guarantees

In this section, we present convergence guarantees of IFCA. In order to streamline our theoretical analysis, we make several simplifications: we consider the IFCA with gradient averaging, and assume that all the worker machines participate in every rounds of IFCA, i.e., $\mathcal{M}_t = [m]$ for all t . In addition, we also use the *re-sampling* technique for the purpose of theoretical analysis. In particular, suppose that we run a total of T parallel iterations. We partition the n data points on each machine into $2T$ disjoint subsets, each with $n^j = \frac{n}{2T}$ data points. For the i -th machine, we denote the subsets as $\mathcal{Z}_i^{(0)}, \dots, \mathcal{Z}_i^{(T-1)}$ and $Z_i^{(0)}, \dots, Z_i^{(T-1)}$. In the t -th iteration, we use $\mathcal{Z}_i^{(t)}$ to estimate the cluster identity, and use $Z_i^{(t)}$ to conduct gradient descent. As we can see, we use fresh data samples for each iteration of the algorithm. Furthermore, in each iteration, we use different set of data points for obtaining the cluster estimate and computing the gradient. This is done in order to remove the inter-dependence between the cluster estimation and the gradient computation, and ensure that in each iteration, we use fresh i.i.d. data that are independent of the current model parameter. We would like to emphasize that re-sampling is a standard tool used in statistics [61, 98, 101–103], and that it is for theoretical tractability only and is not required in practice as we show in Section 2.6.

Under these conditions, the update rule for the parameter vector of the j -th cluster can be written as

$$S_j^{(t)} = \{i \in [m] : j = \operatorname{argmin}_{j' \in [k]} F_i(w_{j'}^{(t)}; \mathcal{Z}_i^{(t)})\}; \quad w_j^{(t+1)} = w_j^{(t)} - \frac{\gamma}{m} \sum_{i \in S_j^{(t)}} \nabla F_i(w_j^{(t)}; Z_i^{(t)});$$

where $S_j^{(t)}$ denotes the set of worker machines whose cluster identity estimate is j in the t -th iteration. In the following, we discuss the convergence guarantee of IFCA under two models: in Section 2.5.1, we analyze the algorithm under a linear model with Gaussian features and squared loss, and in Section 2.5.2, we analyze the algorithm under a more general setting of strongly convex loss functions.

2.5.1 Linear models with squared loss

In this section, we analyze our algorithm in a concrete linear model. This model can be seen as a warm-up example for more general problems with strongly convex loss functions that we discuss in Section 2.5.2, as well as a distributed formulation of the widely studied mixture of linear regression problem [98, 103]. We assume that the data on the worker machines in the j -th cluster are generated in the following way: for $i \in S_j$, the feature-response pair of the i -th worker machine machine satisfies

$$y^{i;} = hX^{i;} ; w_j i + \epsilon^{i;} ;$$

where $X^{i;} \sim N(0; I_d)$ and the additive noise $\epsilon^{i;} \sim N(0; \sigma^2)$ is independent of $X^{i;}$. Furthermore, we use the squared loss function $f(w; x; y) = (y - hX; wi)^2$. As we can see, this model is the mixture of linear regression model in the distributed setting. We observe that under the above setting, the parameters $\{w_j\}_{j=1}^k$ are the minimizers of the population loss function $F^J(\cdot)$.

We proceed to analyze our algorithm. We define $p_j := |S_j|/m$ as the fraction of worker machines belonging to the j -th cluster, and let $\rho := \min\{p_1; p_2; \dots; p_k\}$. We also define the minimum separation as $\delta := \min_{j \neq j'} \|w_j - w_{j'}\|$, and $\gamma := \frac{\delta^2}{\sigma^2}$ as the signal-to-noise ratio. Before we establish our convergence result, we state a few assumptions. Here, recall that n^ℓ denotes the number of data that each worker uses in each step.

Assumption 3. *The initialization of parameters $w_j^{(0)}$ satisfy $\|w_j^{(0)} - w_j\| \leq \frac{\delta}{4}$, $\forall j \in [k]$.*

Assumption 4. *Without loss of generality, we assume that $\max_{j \in [k]} \|w_j\| \leq 1$, and that $\frac{\delta}{\sigma} \geq \frac{1}{c_1}$. We also assume that $n^\ell \geq \frac{c_2}{\rho} \log m$, $d \leq \frac{c_3}{\rho} \log m$, $\rho \geq \frac{c_4 \log m}{m}$, $\rho n^\ell \geq d$, and $\frac{\delta}{\sigma} \geq \frac{c_5}{\rho} \frac{d}{m n^\ell} + \exp(-c_6 (\frac{\delta}{\sigma})^2 n^\ell)$ for some universal constant c .*

In Assumption 3, we assume that the initialization is close enough to w_j . We note that this is a standard assumption in the convergence analysis of mixture models [99, 104], due to the non-convex optimization landscape of mixture model problems. In Assumption 4, we put mild assumptions on n^ℓ , m , ρ , and d . The condition that $\rho n^\ell \geq d$ simply assumes that the total number of data that we use in each iteration for each cluster is at least as large as the dimension of the parameter space. The condition that $\frac{\delta}{\sigma} \geq \frac{c_5}{\rho} \frac{d}{m n^\ell} + \exp(-c_6 (\frac{\delta}{\sigma})^2 n^\ell)$ ensures that the iterates stay close to w_j .

We first provide a single step analysis of our algorithm. We assume that at a certain iteration, we obtain parameter vectors w_j that are close to the ground truth parameters w_j , and show that w_j converges to w_j at an exponential rate with an error floor.

Theorem 1. Consider the linear model and assume that Assumptions 3 and 4 hold. Suppose that in a certain iteration of the IFCA algorithm we obtain parameter vectors w_j with $\|w_j\| \leq \frac{1}{4}$. Let w_j^+ be iterate after this iteration. Then there exist universal constants $c_1; c_2; c_3; c_4 > 0$ such that when we choose step size $\eta = c_1/p$, with probability at least $1 - \text{poly}(m)^{-1}$, we have for all $j \in [k]$,

$$\|kw_j^+ - w_j\| \leq \frac{1}{2}\|kw_j - w_j\| + c_2 \frac{d}{mn^\ell} + c_3 \exp(-c_4 \left(\frac{1}{4}\right)^2 n^\ell) :$$

We prove Theorem 1 in Appendix 2.8. Here, we briefly summarize the proof idea. Using the initialization condition, we show that the set $\{S_j^k\}_{j=1}^k$ has a significant overlap with $\{S_j^k\}_{j=1}^k$. In the overlapped set, we then argue that the gradient step provides a contraction and error floor due to the basic properties of linear regression. We then bound the gradient norm of the miss-classified machines and add them to the error floor. We complete the proof by combining the contributions of properly classified and miss-classified worker machines. We can then iteratively apply Theorem 1 and obtain accuracy of the final solution w_j in the following corollary.

Corollary 1. Consider the linear model and assume that Assumptions 3 and 4 hold. By choosing step size $\eta = c_1/p$, with probability at least $1 - \frac{\log(\frac{1}{4})}{\text{poly}(m)}$, after $T = \log_{\frac{1}{4}}$ parallel iterations, we have for all $j \in [k]$, $\|kw_j - w_j\| \leq \epsilon$, where $\epsilon = c_5 \frac{d}{pn^\ell} + c_6 \exp(-c_4 \left(\frac{1}{4}\right)^2 n^\ell)$.

Let us examine the final accuracy. Since the number of data points on each worker machine $n = 2n^\ell T = 2n^\ell \log(\frac{1}{4})$, we know that for the smallest cluster, there are a total of $2pmn^\ell \log(\frac{1}{4})$ data points. According to the minimax estimation rate of linear regression [13], we know that even if we know the ground truth cluster identities, we cannot obtain an error rate better than $O(\frac{d}{pmn^\ell \log(\frac{1}{4})})$. Comparing this rate with our statistical accuracy ϵ , we can see that the first term $\frac{d}{pn^\ell}$ in ϵ is equivalent to the minimax rate up to a logarithmic factor and a dependency on p , and the second term in ϵ decays exponentially fast in n^ℓ , and therefore, our final statistical error rate is near optimal.

2.5.2 Strongly convex loss functions

In this section, we study a more general scenario where the population loss functions of the k clusters are strongly convex and smooth. In contrast to the previous section, our analysis do not rely on any specific statistical model, and thus can be applied to more general machine learning problems. We start with reviewing the standard definitions of strongly convex and smooth functions $F : \mathbb{R}^d \rightarrow \mathbb{R}$.

Definition 4. F is μ -strongly convex if $\forall w, w', F(w') \geq F(w) + \mu(w' - w)^T(w' - w) + \frac{\mu}{2}\|w' - w\|^2$.

Definition 5. F is L -smooth if $\forall w, w', \|F(w') - F(w)\| \leq L\|w' - w\|$.

In this section, we assume that the population loss functions $F^j(w)$ are strongly convex and smooth.

Assumption 5. The population loss function $F^j(w)$ is μ -strongly convex and L -smooth, $\forall j \in [k]$.

We note that we do not make any convexity or smoothness assumptions on the individual loss function $f(w; z)$. Instead, we make the following distributional assumptions on $f(w; z)$ and $\nabla f(w; z)$.

Assumption 6. For every w and every $j \in [k]$, the variance of $f(w; z)$ is upper bounded by σ^2 , when z is sampled according to D_j , i.e., $\mathbb{E}_z \in D_j [(f(w; z) - F^j(w))^2] \leq \sigma^2$

Assumption 7. For every w and every $j \in [k]$, the variance of $\nabla f(w; z)$ is upper bounded by ν^2 , when z is sampled according to D_j , i.e., $\mathbb{E}_z \in D_j [k \|\nabla f(w; z) - \nabla F^j(w)\|_2^2] \leq \nu^2$

Bounded variance of gradient is very common in analyzing SGD [105]. In this paper we use loss function value to determine cluster identity, so we also need to have a probabilistic assumption on $f(w; z)$. We note that bounded variance is a relatively weak assumption on the tail behavior of probability distributions. In addition to the assumptions above, we still use some definitions from Section 2.5.1, i.e., $\rho_j = \min_{j \in J^0} k w_j - w_{j^0} k$, and $\rho = \min_{j \in [k]} \rho_j$ with $\rho_j = \min_{j \in J^0} k w_j - w_{j^0} k$. We make the following assumptions on the initialization, n^0 , ρ , and σ .

Assumption 8. Without loss of generality, we assume that $\max_{j \in [k]} k w_j - w_{j^0} k \leq \frac{1}{4}$. We also assume that $k w_j^{(0)} - w_{j^0} k \leq \frac{1}{4} - \frac{\sigma}{L}$, $\forall j \in [k]$, $n^0 \geq \frac{k^2}{2^4}$, $\rho \geq \frac{\log(mn^0)}{m}$, and that

$$\Theta(\max\{n^0\}^{-1-5}; m^{-1-6} (n^0)^{-1-3} g):$$

Here, for simplicity, the Θ notation omits any logarithmic factors and quantities that do not depend on m and n^0 . As we can see, again we need to assume good initialization, due to the nature of the mixture model, and the assumptions that we impose on n^0 , ρ , and σ are relatively mild; in particular, the assumption on ρ ensures that the iterates stay close to an $\frac{1}{2}$ ball around w_j .

Theorem 2. Suppose Assumptions 5-8 hold. Choose step size $\eta = 1/L$. Then, with probability at least $1 - \frac{\sigma}{\rho}$, after $T = \frac{8L}{\rho} \log \frac{1}{\frac{\sigma}{\rho}}$ parallel iterations, we have for all $j \in [k]$, $k w_j - w_{j^0} k \leq \frac{\sigma}{\rho}$, where

$$\frac{\sigma}{\rho} \leq \frac{vkL \log(mn^0)}{\rho^{5-2} m} + \frac{2L^2 k \log(mn^0)}{\rho^{2-4} n^0} + \Theta\left(\frac{1}{n^0 \rho m}\right):$$

We prove Theorem 2 in the Appendix 2.9. Similar to Section 2.5.1, to prove this result, we first prove a per-iteration contraction

$$k w_j^+ - w_{j^0} k \leq \left(1 - \frac{\rho}{8L}\right) k w_j - w_{j^0} k + \Theta\left(\frac{1}{mn^0} + \frac{1}{n^0} + \frac{1}{n^0 \rho m}\right); \forall j \in [k];$$

and then derive the convergence rate. To better interpret the result, we focus on the dependency on m and n and treat other quantities as constants. Then, since $n = 2n^0 T$, we know that n and n^0 are of the same scale up to a logarithmic factor. Therefore, the final statistical error rate that we obtain is $\frac{\sigma}{\rho} = \Theta\left(\frac{1}{mn} + \frac{1}{n}\right)$. As discussed in Section 2.5.1, $\frac{1}{mn}$ is the optimal rate even if we

know the cluster identities; thus our statistical rate is near optimal in the regime where $n \ll m$. In comparison with the statistical rate in linear models $\mathcal{O}(\frac{1}{mn} + \exp(-n))$, we note that the major difference is in the second term. The additional terms of the linear model and the strongly convex case are $\exp(-n)$ and $\frac{1}{n}$, respectively. We note that this is due to different statistical assumptions: in for the linear model, we assume Gaussian noise whereas here we only assume bounded variance.

2.6 Experiments

In this section, we present our experimental results, which not only validate the theoretical claims in Section 2.5, but also demonstrate that our algorithm can be efficiently applied beyond the regime we discussed in the theory. We emphasize that we *do not* re-sample fresh data points at each iteration. Furthermore, the requirement on the initialization can be relaxed. More specifically, for linear models, we observe that random initialization with a few restarts is sufficient to ensure convergence of Algorithm 1. In our experiments, we also show that our algorithm works efficiently for problems with non-convex loss functions such as neural networks.

2.6.1 Synthetic data

We begin with evaluation of Algorithm 1 with gradient averaging (option I) on linear models with squared loss, as described in Section 2.5.1. For all $j \in [k]$, we first generate $w_j \sim \text{Ber}(0.5)$ coordinate-wise, and then rescale their ℓ_2 norm to R . This ensures that the separation between the w_j 's is proportional to R in expectation, and thus, in this experiment, we use R to represent the *separation* between the ground truth parameter vectors. Moreover, we simulate the scenario where all the worker machines participate in all iterations, and all the clusters contain same number of worker machines. For each trial of the experiment, we first generate the parameter vectors w_j 's, fix them, and then randomly initialize $w_j^{(0)}$ according to an independent coordinate-wise Bernoulli distribution. We then run Algorithm 1 for 300 iterations, with a constant step size. For $k = 2$ and $k = 4$, we choose the step size in $\{0.01; 0.1; 1\}$, $\{0.5; 1.0; 2.0\}$, respectively. In order to determine whether we successfully learn the model or not, we sweep over the aforementioned step sizes and define the following distance metric: $\text{dist} = \frac{1}{k} \sum_{j=1}^k \|w_j - \hat{w}_j\|_2$, where \hat{w}_j 's are the parameter estimates obtained from Algorithm 1. A trial is dubbed *successful* if for a fixed set of w_j , among 10 random initialization of $w_j^{(0)}$, at least in one scenario, we obtain $\text{dist} \leq 0.6$.

In Fig. 2.2 (a-b), we plot the empirical success probability over 40 trials, with respect to the separation parameter R . We set the problem parameters as (a) $(m; n; d) = (100; 100; 1000)$ with $k = 2$, and (b) $(m; n; d) = (400; 100; 1000)$ with $k = 4$. As we can see, when R becomes larger, i.e., the separation between parameters increases, and the problem becomes easier to solve, yielding in a higher success probability. This validates our theoretical result that higher signal-to-noise ratio produces smaller error floor. In Fig. 2.2 (c-d), we characterize the dependence on m and n , with fixing R and d with $(R; d) = (0.1; 1000)$ for (c) and $(R; d) = (0.5; 1000)$ for (d). We observe that when we increase m and/or n , the success probability improves. This validates our theoretical finding that more data and/or more worker machines help improve the performance of the algorithm.

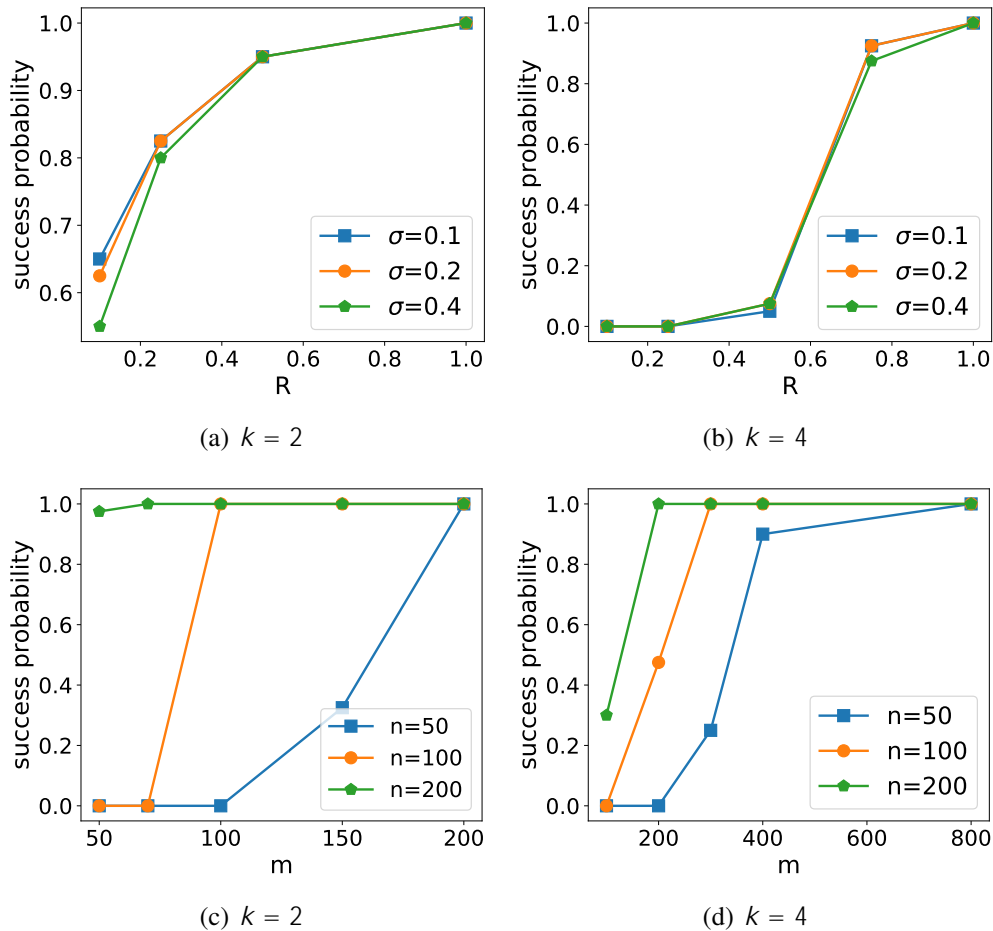


Figure 2.2: Success probability with respect to: (a), (b) the separation scale R and the scale of additive noise σ ; (c), (d) the number of worker machines m and the sample size on each machine n . In (a) and (b), we see that the success probability gets better with increasing R , i.e., more separation between ground truth parameter vectors, and in (c) and (d), we note that the success probability improves with an increase of mn , i.e., more data on each machine and/or more machines.

2.6.2 Rotated MNIST and CIFAR

We also create clustered FL datasets based on the MNIST [106] and CIFAR-10 [107] datasets. In order to simulate an environment where the data on different worker machines are generated from different distributions, we augment the datasets using rotation, and create the Rotated MNIST [108] and Rotated CIFAR datasets. For **Rotated MNIST**, recall that the MNIST dataset has 60000 training images and 10000 test images with 10 classes. We first augment the dataset by applying $0; 90; 180; 270$ degrees of rotation to the images, resulting in $k = 4$ clusters. For given m and n satisfying $mn = 60000k$, we randomly partition the images into m worker machines so that each machine holds n images *with the same rotation*. We also split the test data into $m_{test} = 10000k/n$ worker machines in the same way. The **Rotated CIFAR** dataset is also created in a similar way as

Table 2.1: Test accuracies(%) std on Rotated MNIST ($k = 4$) and Rotated CIFAR ($k = 2$)

m, n	Rotated MNIST						Rotated CIFAR	
	4800, 50		2400, 100		1200, 200		200, 500	
IFCA (ours)	94.20	0.03	95.05	0.02	95.25	0.40	81.51	1.37
global model	86.74	0.04	88.65	0.08	89.73	0.13	77.87	0.39
local model	63.32	0.02	73.66	0.04	80.05	0.02	33.97	1.19

Rotated MNIST, with the main difference being that we create $k = 2$ clusters with 0 and 180 degrees of rotation. We note that creating different tasks by manipulating standard datasets such as MNIST and CIFAR-10 has been widely adopted in the continual learning research community [108–110]. For clustered FL, creating datasets using rotation helps us simulate a federated learning setup with clear cluster structure.

For our MNIST experiments, we use the fully connected neural network with ReLU activations, with a single hidden layer of size 200; and for our CIFAR experiments, we use a convolution neural network model which consists of 2 convolutional layers followed by 2 fully connected layers, and the images are preprocessed by standard data augmentation such as flipping and random cropping.

We compare our IFCA algorithm with two baseline algorithms, i.e., the *global model*, and *local model* schemes. For **IFCA**, we use model averaging (option II in Algorithm 1). For MNIST experiments, we use full worker machines participation ($M_t = [m]$ for all t). For LocalUpdate step in Algorithm 1, we choose $\eta = 10$ and step size $\alpha = 0.1$. For CIFAR experiments, we choose $jM_t = 0.1m$, and apply step size decay 0.99, and we also set $\eta = 5$ and batch size 50 for LocalUpdate process, following prior works [1]. In the **global model** scheme, the algorithm tries to learn single global model that can make predictions from all the distributions. The algorithm does not consider cluster identities, so model averaging step in Algorithm 1 becomes $w^{(t+1)} = \frac{1}{|M_t|} \sum_{i \in M_t} w_i^{(t)}$, i.e. averaged over parameters from all the participating machines. In the **local model** scheme, the model in each node performs gradient descent only on local data available, and model averaging is not performed.

For IFCA and the global model scheme, we perform inference in the following way. For every test worker machine, we run inference on all learned models (k models for IFCA and one model for global model scheme), and calculate the accuracy from the model that produces the smallest loss value. For testing the local model baselines, the models are tested by measuring the accuracy on the test data with the same distribution (i.e. those have the same rotation). We report the accuracy averaged over all the models in worker machines. For all algorithms, we run experiment with 5 different random seeds and report the average and standard deviation.

Our experimental results are shown in Table 2.1. We can observe that our algorithm performs better than the two baselines. As we run the IFCA algorithm, we observe that we can gradually find the underlying cluster identities of the worker machines, and after the correct cluster is found, each model is trained and tested using data with the same distribution, resulting in better accuracy. The global model baseline performs worse than ours since it tries to fit all the data from different distributions, and cannot provide personalized predictions. The local model baseline algorithm overfits to the local data easily, leading to worse performance than ours.

2.6.3 Federated EMNIST

We provide additional experimental results on the Federated EMNIST (FEMNIST) [76], which is a realistic FL dataset where the data points on every worker machine are the handwritten digits or letters from a specific writer. Although the data distribution among all the users are similar, there might be ambiguous cluster structure since the writing styles of different people may be clustered. We use the weight sharing technique mentioned in Section 2.4.1. We use a neural network with two convolutional layers, with a max pooling layer after each convolutional layer, followed by two fully connected layers. We share the weights of all the layers, except the last layer which is trained by IFCA. We treat the number of clusters k as a hyper parameter and run the experiments with different values of k . We compare IFCA with the global model and local model approaches, as well as the one-shot centralized clustering algorithm in [32]. The test accuracies are shown in Table 2.2, with mean and standard deviation computed over 5 independent runs. As we can see, IFCA shows clear advantage over the global model and local model approaches. The results of IFCA and the one-shot algorithm are similar. However, as we emphasized in Section 2.2, IFCA does not run a centralized clustering procedure, and thus reduces the computational cost at the center machine. As a final note, we observe that IFCA is robust to the choice of the number of clusters k . The results of the algorithm with $k = 2$ and $k = 3$ are similar, and we notice that when $k > 3$, IFCA automatically identifies 3 clusters, and the remaining clusters are empty. This indicates the applicability of IFCA in real-world problems where the cluster structure is ambiguous and the number of clusters is unknown.

Table 2.2: Test accuracies (%) std on FEMNIST

IFCA ($k = 2$)		IFCA ($k = 3$)		one-shot ($k = 2$)		one-shot ($k = 3$)		global		local	
87.99	0.35	87.89	0.52	87.41	0.39	87.38	0.37	84.45	0.51	75.85	0.72

2.7 Conclusion and Open Problems

In this paper, we address the clustered FL problem. We propose an iterative algorithm and obtain convergence guarantees for strongly convex and smooth functions. In experiments, we achieve this via random initialization with multiple restarts, and we show that our algorithm works efficiently beyond the convex regime. An immediate future work is to extend the analysis to weakly convex and non-convex functions. Also, the convergence guarantees are local, i.e., a good initialization is required. Obtaining a provable initialization for the clustered FL is an interesting future direction.

Our formulation is one of the problem setups for personalized Federated Learning. We expect that, overall our framework will better protect the users' privacy in a Federated Learning system while still provide personalized predictions. The reason is that our algorithm does not require the

users to send any of their own personal data to the central server, and the users can still learn a personalized model using their on-device computing power. One potential risk is that our algorithm still requires the users to send the estimates of their cluster identities to the central server. Thus there might still be privacy concerns in this step. We suggest that before applying our algorithm, or generally any FL algorithms, in a real-world system, we should first request the users' consent.

Appendix

In our proofs, we use $C; c_1; c_2; \dots$ to denote positive universal constants, the value of which may differ across instances. For a matrix A , we write $\|A\|_{op}$ and $\|A\|_F$ as the operator norm and Frobenius norm, respectively. For a set S , we use \bar{S} to denote the complement of the set.

2.8 Proof of Theorem 1

Since we only analyze a single iteration, for simplicity we drop the superscript that indicates the iteration counter. Suppose that at a particular iteration, we have model parameters $w_j, j \in [k]$, for the k clusters. We denote the *estimation* of the set of worker machines that belongs to the j -th cluster by S_j , and recall that the true clusters are denoted by $S_j, j \in [k]$.

Probability of erroneous cluster identity estimation We begin with the analysis of the probability of incorrect cluster identity estimation. Suppose that a worker machine i belongs to S_j . We define the event $E_i^{j:j^0}$ as the event when the i -th machine is classified to the j^0 -th cluster, i.e., $i \in S_{j^0}$. Thus the event that worker i is correctly classified is $E_i^{j:j}$, and we use the shorthand notation $E_i := E_i^{j:j}$. We now provide the following lemma that bounds the probability of $E_i^{j:j^0}$ for $j^0 \neq j$.

Lemma 1. *Suppose that worker machine $i \in S_j$. Let $\epsilon := \frac{2}{m}$. Then there exist universal constants c_1 and c_2 such that for any $j^0 \neq j$,*

$$\mathbb{P}(E_i^{j:j^0}) \leq c_1 \exp\left(-c_2 n^\theta \left(\frac{\epsilon}{1+\epsilon}\right)^2\right);$$

and by union bound

$$\mathbb{P}(\bar{E}_i) \leq c_1 k \exp\left(-c_2 n^\theta \left(\frac{\epsilon}{1+\epsilon}\right)^2\right);$$

We prove Lemma 1 in Appendix 2.8.1.

Now we proceed to analyze the gradient descent step. Without loss of generality, we only analyze the first cluster. The update rule of w_1 in this iteration can be written as

$$w_1^+ = w_1 - \frac{\eta}{m} \sum_{i \in S_1} \nabla F_i(w_1; Z_i);$$

where Z_i is the set of the n^j data points that we use to compute gradient in this iteration on a particular worker machine.

We use the shorthand notation $F_i(w) := F_i(w; Z_i)$, and note that $F_i(w)$ can be written in the matrix form as

$$F_i(w) = \frac{1}{n^j} \sum_{z \in Z_i} \langle Y_i, X_i w \rangle^2;$$

where we have the feature matrix $X_i \in \mathbb{R}^{n^j \times d}$ and response vector $Y_i = X_i w_1 + \epsilon_i$. According to our model, all the entries of X_i are i.i.d. sampled according to $N(0; 1)$, and $\epsilon_i \sim N(0; \sigma^2 I)$.

We first notice that

$$\| \sum_{i \in S_1} F_i(w) - \sum_{i \in S_2} F_i(w) \| \leq \frac{1}{m} \sum_{i \in S_1} \| \nabla F_i(w) \| + \frac{1}{m} \sum_{i \in S_2} \| \nabla F_i(w) \| \leq kT_1\|w - w_1\| + kT_2\|w - w_1\|;$$

We control the two terms separately. Let us first focus on $kT_1\|w - w_1\|$.

Bound $kT_1\|w - w_1\|$ To simplify notation, we concatenate all the feature matrices and response vectors of all the worker machines in $S_1 \setminus S_1$ and get the new feature matrix $X \in \mathbb{R}^{N \times d}$, $Y \in \mathbb{R}^N$ with $Y = Xw_1 + \epsilon$, where $N := n^j |S_1 \setminus S_1|$. It is then easy to verify that

$$\begin{aligned} T_1 &= \left(\frac{2}{mn^j} X^T X \right) (w_1 - w_1) + \frac{2}{mn^j} X^T \epsilon \\ &= \left(\frac{2}{mn^j} \mathbb{E}[X^T X] \right) (w_1 - w_1) + \frac{2}{mn^j} (\mathbb{E}[X^T X] - X^T X) (w_1 - w_1) + \frac{2}{mn^j} X^T \epsilon \\ &= \left(1 - \frac{2}{mn^j} N \right) (w_1 - w_1) + \frac{2}{mn^j} (\mathbb{E}[X^T X] - X^T X) (w_1 - w_1) + \frac{2}{mn^j} X^T \epsilon; \end{aligned}$$

Therefore

$$\| \sum_{i \in S_1} F_i(w) - \sum_{i \in S_2} F_i(w) \| \leq \left(1 - \frac{2}{mn^j} N \right) \|w - w_1\| + \frac{2}{mn^j} \|X^T X - \mathbb{E}[X^T X]\|_{op} \|w - w_1\| + \frac{2}{mn^j} \|X^T \epsilon\|; \quad (2.1)$$

Thus in order to bound $kT_1\|w - w_1\|$, we need to analyze two terms, $\|X^T X - \mathbb{E}[X^T X]\|_{op}$ and $\|X^T \epsilon\|$. To bound $\|X^T X - \mathbb{E}[X^T X]\|_{op}$, we first provide an analysis of N showing that it is large enough. Using Lemma 1 in conjunction with Assumption 4, we see that the probability of correctly classifying any worker machine i , given by $\mathbb{P}(E_i)$, satisfies $\mathbb{P}(E_i) \geq \frac{1}{2}$. Hence, we obtain

$$\mathbb{E}[|S_1 \setminus S_1|] = \mathbb{E}\left[\sum_{i \in S_1} \mathbb{1}_{E_i}\right] \geq \frac{1}{2} p_1 m;$$

where we use the fact that $\sum_{i \in S_1} \mathbb{1}_{E_i} = |S_1 \setminus S_1|$. Since $|S_1 \setminus S_1|$ is a sum of Bernoulli random variables with success probability at least $\frac{1}{2}$, we obtain

$$\mathbb{P}\left[|S_1 \setminus S_1| \leq \frac{1}{4} p_1 m\right] \leq \mathbb{P}\left[|S_1 \setminus S_1| \leq \mathbb{E}[|S_1 \setminus S_1|] - \frac{1}{4} p_1 m\right] \leq 2 \exp(-cpm);$$

where $\rho = \min\{p_1, p_2, \dots, p_k\}$, and the second step follows from Hoeffding's inequality. Hence, we obtain $|S_1 \setminus S_1^j| \leq \frac{1}{4} p_1 m$ with high probability, which yields

$$P(N \geq \frac{1}{4} p_1 m n^d) \geq 1 - 2 \exp(-c p m). \quad (2.2)$$

By combining this fact with our assumption that $p m n^d \gg d$, we know that $N \gg d$. Then, according to the concentration of the covariance of Gaussian random vectors [13], we know that with probability at least $1 - 2 \exp(-\frac{1}{2} d)$,

$$\|kX^>X - E[X^>X]\|_{k_{op}} \leq \frac{\rho}{6} \frac{1}{dN}. \quad N: \quad (2.3)$$

We now proceed to bound $\|kX^>k$. In particular, we use the following lemma.

Lemma 2. Consider a random matrix $X \in \mathbb{R}^{N \times d}$ with i.i.d. entries sampled according to $N(0; 1)$, and $\mathbf{z} \in \mathbb{R}^N$ be a random vector sampled according to $N(0; \sigma^2 I)$, independently of X . Then we have with probability at least $1 - 2 \exp(-c_1 \min\{d; N\}g)$,

$$\|kXk_{op} \leq c \max\{\sqrt{d}; \frac{\rho}{N}g\};$$

and with probability at least $1 - c_2 \exp(-c_3 \min\{d; N\}g)$,

$$\|kX^>k \leq c_4 \frac{\rho}{dN};$$

We prove Lemma 2 in Appendix 2.8.2. Now we can combine (2.1), (2.3), (2.2), and Lemma 2 and obtain with probability at least $1 - c_1 \exp(-c_2 p m) - c_3 \exp(-c_4 d)$,

$$\|kT_1k \leq (1 + c_5 \rho)kw_1 - w_1k + c_6 \frac{1}{mn^d}. \quad (2.4)$$

Since we assume that $\rho \gg \frac{\log m}{m}$ and $d \gg \log m$, the success probability can be simplified as $1 - \text{poly}(m)$.

Bound $\|kT_2k$ We first condition on S_1 . We have the following:

$$\frac{1}{n^d} F_i(w_1) = \frac{2}{n^d} X_i^>(Y_i - X_i w_1):$$

For $i \in S_1 \setminus S_j$, with $j \neq 1$, we have $Y_i = X_i w_j + \epsilon_i$, and so we obtain

$$\frac{1}{n^d} F_i(w_1) = 2X_i^>X_i(w_j - w_1) + 2X_i^>\epsilon_i;$$

which yields

$$\frac{1}{n^d} \|kF_i(w_1)k\| \leq \|kX_i k_{op}^2 + kX_i^>\epsilon_i k; \quad (2.5)$$

where we use the fact that $\|kw_j - w_1 k\| \leq \|kw_j\| + \|kw_1\| + \|kw_1 - w_1 k\|$. Then, we combine (2.5) and Lemma 2 and get with probability at least $1 - c_1 \exp(-c_2 \min\{fd; n^\rho g\})$,

$$\|k r F_i(w_1) k\| \leq \frac{1}{n^\rho} (c_3 \max\{fd; n^\rho g\} + c_4 \frac{\rho}{dn^\rho}) + c_5 \max\{f; \frac{d}{n^\rho} g\}; \quad (2.6)$$

where we use our assumption that $\rho \geq 1$. By union bound, we know that with probability at least $1 - c_1 m \exp(-c_2 \min\{fd; n^\rho g\})$, (2.6) holds for all $j \in \overline{S_1}$. In addition, since we assume that $n^\rho \geq \log m$, $d \geq \log m$, this probability can be lower bounded by $1 - \frac{1}{\text{poly}(m)}$. This implies that conditioned on S_1 , with probability at least $1 - \frac{1}{\text{poly}(m)}$,

$$\|k T_2 k\| \leq c_5 \frac{1}{m} \sum_{j \in S_1 \setminus \overline{S_1}} \max\{f; \frac{d}{n^\rho} g\}; \quad (2.7)$$

Since we choose $\rho = \frac{c}{p}$, we have $\frac{1}{m} \sum_{j \in S_1 \setminus \overline{S_1}} \max\{f; \frac{d}{n^\rho} g\} \leq \frac{1}{m}$, where we use our assumption that $p m n^\rho \geq d$. This shows that with probability at least $1 - \frac{1}{\text{poly}(m)}$,

$$\|k T_2 k\| \leq c_5 \sum_{j \in S_1 \setminus \overline{S_1}} \frac{1}{m}; \quad (2.8)$$

We then analyze $\sum_{j \in S_1 \setminus \overline{S_1}} \frac{1}{m}$. By Lemma 1, we have

$$\mathbb{E}[\sum_{j \in S_1 \setminus \overline{S_1}} \frac{1}{m}] \leq c_6 m \exp(-c_7 (\frac{1}{c+1})^2 n^\rho); \quad (2.9)$$

According to Assumption 4, we know that $n^\rho \geq c(\frac{1}{c+1})^2 \log m$, for some constant c that is large enough. Therefore, $m \exp(-\frac{1}{c} (\frac{1}{c+1})^2 n^\rho) \leq \frac{1}{m}$, and thus, as long as c is large enough such that $\frac{1}{c} < c_7$ where c_7 is defined in (2.9), we have

$$\mathbb{E}[\sum_{j \in S_1 \setminus \overline{S_1}} \frac{1}{m}] \leq c_6 \exp(-c_8 (\frac{1}{c+1})^2 n^\rho); \quad (2.10)$$

and then by Markov's inequality, we have

$$\mathbb{P}[\sum_{j \in S_1 \setminus \overline{S_1}} \frac{1}{m} \geq c_6 \exp(-\frac{c_8}{2} (\frac{1}{c+1})^2 n^\rho)] \leq \exp(-\frac{c_8}{2} (\frac{1}{c+1})^2 n^\rho) \leq \frac{1}{\text{poly}(m)}; \quad (2.11)$$

Combining (2.8) with (2.11), we know that with probability at least $1 - \frac{1}{\text{poly}(m)}$,

$$\|k T_2 k\| \leq c_1 \exp(-c_2 (\frac{1}{c+1})^2 n^\rho);$$

Using this fact and (2.4), we obtain that with probability at least $1 - \frac{1}{\text{poly}(m)}$,

$$\|kw_1^* - w_1 k\| \leq (1 - c_1 \rho) \|kw_1 - w_1 k\| + c_2 \frac{1}{m n^\rho} + c_3 \exp(-c_4 (\frac{1}{c+1})^2 n^\rho);$$

Then we can complete the proof for the first cluster by choosing $\rho = \frac{1}{2c_1 p}$. To complete the proof for all the k clusters, we can use union bound, and the success probability is $1 - \frac{1}{\text{poly}(m)}$. However, since $k \leq m$ by definition, we still have success probability $1 - \frac{1}{\text{poly}(m)}$.

2.8.1 Proof of Lemma 1

Without loss of generality, we analyze $E_i^{1:j}$ for some $j \notin 1$. By definition, we have

$$E_i^{1:j} = \mathbb{P} \left[F_i(w_j; \mathcal{Z}_i) - F_i(w_1; \mathcal{Z}_i) \geq g \right]$$

where \mathcal{Z}_i is the set of n^ℓ data points that we use to estimate the cluster identity in this iteration. We write the data points in \mathcal{Z}_i in matrix form with feature matrix $X_i \in \mathbb{R}^{n^\ell \times d}$ and response vector $Y_i = X_i w_1 + \epsilon_i$. According to our model, all the entries of X_i are i.i.d. sampled according to $N(0, 1)$, and $\epsilon_i \sim N(0, \sigma^2 I)$. Then, we have

$$\mathbb{P} \left[F_i^{1:j} g \right] = \mathbb{P} \left[\sum_{i=1}^j k X_i (w_1 - w_j) + \epsilon_i k^2 \geq \sum_{i=1}^j k X_i (w_1 - w_j) + \epsilon_i k^2 \right]$$

Consider the random vector $X_i (w_1 - w_j) + \epsilon_i$, and in particular consider the ℓ -th coordinate of it. Since X_i and ϵ_i are independent and we resample $(X_i; Y_i)$ at each iteration, the ℓ -th coordinate of $X_i (w_1 - w_j) + \epsilon_i$ is a Gaussian random variable with mean 0 and variance $k w_j - w_1 k^2 + \sigma^2$. Since X_i and ϵ_i contain independent rows, the distribution of $\sum_{i=1}^j k X_i (w_1 - w_j) + \epsilon_i k^2$ is given by $(k w_j - w_1 k^2 + \sigma^2) u_j$, where u_j is a standard Chi-squared random variable n^ℓ degrees of freedom. We now calculate the an upper bound on the following probability:

$$\begin{aligned} & \mathbb{P} \left[\sum_{i=1}^j k X_i (w_1 - w_j) + \epsilon_i k^2 \geq \sum_{i=1}^j k X_i (w_1 - w_j) + \epsilon_i k^2 \right] \\ & \stackrel{(i)}{\leq} \mathbb{P} \left[\sum_{i=1}^j k X_i (w_1 - w_j) + \epsilon_i k^2 \geq t \right] + \mathbb{P} \left[\sum_{i=1}^j k X_i (w_1 - w_1) + \epsilon_i k^2 > t \right] \\ & \leq \mathbb{P} \left[(k w_j - w_1 k^2 + \sigma^2) u_j \geq t \right] + \mathbb{P} \left[(k w_1 - w_1 k^2 + \sigma^2) u_j > t \right]; \end{aligned} \quad (2.12)$$

where (i) holds for all $t \geq 0$. For the first term, we use the concentration property of Chi-squared random variables. Using the fact that $k w_j - w_1 k \leq k w_j - w_1 k \leq k w_j - w_j k \leq \frac{3}{4}$, we have

$$\mathbb{P} \left[(k w_j - w_1 k^2 + \sigma^2) u_j \geq t \right] \leq \mathbb{P} \left[\left(\frac{9}{16} \sigma^2 + \sigma^2 \right) u_j \geq t \right]; \quad (2.13)$$

Similarly, using the initialization condition, $k w_1 - w_1 k \leq \frac{1}{4}$, the second term of equation (2.12) can be simplified as

$$\mathbb{P} \left[(k w_1 - w_1 k^2 + \sigma^2) u_j > t \right] \leq \mathbb{P} \left[\left(\frac{1}{16} \sigma^2 + \sigma^2 \right) u_j > t \right]; \quad (2.14)$$

Based on the above observation, we now choose $t = n^\ell \left(\frac{5}{16} \sigma^2 + \sigma^2 \right)$. Recall that $\epsilon := \frac{\sigma^2}{2}$. Then the inequality (2.13) can be rewritten as

$$\mathbb{P} \left[(k w_j - w_1 k^2 + \sigma^2) u_j \geq t \right] \leq \mathbb{P} \left[\frac{u_j}{n^\ell} \geq 1 + \frac{4}{9 + 16} \right];$$

According to the concentration results for standard Chi-squared distribution [13], we know that there exists universal constants c_1 and c_2 such that

$$\mathbb{P} \left[(k w_j - w_1 k^2 + \sigma^2) u_j \geq t \right] \leq c_1 \exp \left[-c_2 n^\ell \left(\frac{4}{9 + 16} \right)^2 \right]; \quad (2.15)$$

Similarly, the inequality (2.14) can be rewritten as

$$\mathbb{P} \left(\sum_{j=1}^k (kw_j - w_j k^2 + \sum_{l=1}^k u_l) > t \right) \leq \mathbb{P} \left(\sum_{j=1}^k \frac{u_j}{n^{\theta}} > \frac{4}{t+16} \right);$$

and again according to the concentration of Chi-squared distribution, there exists universal constants c_3 and c_4 such that

$$\mathbb{P} \left(\sum_{j=1}^k (kw_j - w_j k^2 + \sum_{l=1}^k u_l) > t \right) \leq c_3 \exp \left(-c_4 n^{\theta} \left(\frac{t}{t+16} \right)^2 \right); \quad (2.16)$$

The proof can be completed by combining (2.12), (2.15) and (2.16).

2.8.2 Proof of Lemma 2

According to Theorem 5.39 of [111], we have with probability at least $1 - 2 \exp(-c_1 \max\{fd; Ng\})$,

$$kX_{k_{op}} \leq c \max\left\{ \frac{\rho_{-}}{d}; \frac{\rho_{-}}{Ng} \right\};$$

where c and c_1 are universal constants. As for $kX > k$, we first condition on X . According to the Hanson-Wright inequality [112], we obtain for every $t \geq 0$

$$\mathbb{P} \left(kX > k + kX_{k_F} > t \right) \leq 2 \exp \left(-c \frac{t^2}{2kX_{k_{op}}^2} \right); \quad (2.17)$$

Using Chi-squared concentration [13], we obtain with probability at least $1 - 2 \exp(-cdN)$,

$$kX_{k_F} \leq c \frac{\rho_{-}}{dN};$$

Furthermore, using the fact that $kX_{k_{op}} = kX_{k_{op}}$ and substituting $t = c \frac{\rho_{-}}{dN}$ in (2.17), we obtain with probability at least $1 - c_2 \exp(-c_3 \min\{fd; Ng\})$,

$$kX > k + c_4 \frac{\rho_{-}}{dN};$$

2.9 Proof of Theorem 2

The proof of this theorem is similar to that of the linear model. We begin with a single-step analysis.

2.9.1 Analysis for a single step

Suppose that at a certain step, we have model parameters $w_j, j \in [k]$ for the k clusters. Assume that $kw_j - w_j k \leq \frac{1}{4} \frac{\rho_{-}}{L}$, for all $j \in [k]$.

Probability of erroneous cluster identity estimation: We first calculate the probability of erroneous estimation of worker machines' cluster identity. We define the events E_i^{j,j^0} in the same way as in Appendix 2.8, and have the following lemma.

Lemma 3. *Suppose that worker machine $i \in S_j$. Then there exists a universal constants c_1 such that for any $j^0 \neq j$,*

$$P(E_i^{j,j^0}) \leq c_1 \frac{2}{2^4 \rho^{\theta}};$$

and by union bound

$$P(\bar{E}_i) \leq c_1 \frac{k^2}{2^4 \rho^{\theta}}.$$

We prove Lemma 3 in Appendix 2.9.3. Now we proceed to analyze the gradient descent iteration. Without loss of generality, we focus on w_1 . We have

$$kw_1^+ - w_1 k = kw_1 - w_1 \frac{\sum_{i \in S_1} r F_i(w_1) k}{m};$$

where $F_i(w) := F_i(w; Z_i)$ with Z_i being the set of data points on the i -th worker machine that we use to compute the gradient, and S_1 is the set of indices returned by Algorithm 1 corresponding to the first cluster. Since

$$S_1 = (S_1 \setminus S_1) \cup (S_1 \setminus \bar{S}_1)$$

and the sets are disjoint, we have

$$kw_1^+ - w_1 k = kw_1 - w_1 \frac{\sum_{i \in S_1 \setminus S_1} r F_i(w_1) k}{m} - \frac{\sum_{i \in S_1 \setminus \bar{S}_1} r F_i(w_1) k}{m};$$

Using triangle inequality, we obtain

$$kw_1^+ - w_1 k \leq kT_1 k + kT_2 k;$$

and we control both the terms separately. Let us first focus on $kT_1 k$.

Bound $kT_1 k$ We first split T_1 in the following way:

$$T_1 = \underbrace{w_1 - w_1}_{T_{11}} \frac{\sum_{i \in S_1 \setminus S_1} b r F^1(w_1) k}{m} + \underbrace{w_1}_{T_{12}} \frac{\sum_{i \in S_1 \setminus \bar{S}_1} r F_i(w_1) k}{m}; \quad (2.18)$$

where $b := \frac{|S_1 \setminus S_1|}{m}$. Let us condition on S_1 . According to standard analysis technique for gradient descent on strongly convex functions, we know that when $b \geq \frac{1}{L}$,

$$kT_{11} k = kw_1 - w_1 \frac{b r F^1(w_1) k}{m} \leq \left(1 - \frac{b}{L}\right) kw_1 - w_1 k; \quad (2.19)$$

Further, we have $E[kT_{12}k^2] = \frac{v^2}{n^j S_1 \setminus S_{1j}}$, which implies $E[kT_{12}k] \leq \frac{v}{n^j S_1 \setminus S_{1j}}$, and thus by Markov's inequality, for any $\epsilon_0 > 0$, with probability at least $1 - \epsilon_0$,

$$kT_{12}k \leq \frac{v}{n^j S_1 \setminus S_{1j}}. \quad (2.20)$$

We then analyze $jS_1 \setminus S_{1j}$. Similar to the proof of Theorem 1, we can show that $jS_1 \setminus S_{1j}$ is large enough. From Lemma 3 and using our assumption, we see that the probability of correctly classifying any worker machine i , given by $P(E_i)$, satisfies $P(E_i) \geq \frac{1}{2}$. Recall $\rho = \min\{\rho_1, \rho_2, \dots, \rho_k\}$, and we obtain $|jS_1 \setminus S_{1j}| \geq \frac{1}{4}\rho_1 m$ with probability at least $1 - 2\exp(-\rho pm)$. Let us condition on $|jS_1 \setminus S_{1j}| \geq \frac{1}{4}\rho_1 m$ and choose $\epsilon = 1/L$. Then $\epsilon = 1/L$ is satisfied, and on the other hand $\epsilon \leq \frac{\rho}{4L}$. Plug this fact in (2.19), we obtain

$$kT_{11}k \leq (1 + \frac{\rho}{8L})k w_1 - w_1 k. \quad (2.21)$$

We then combine (2.20) and (2.21) and have with probability at least $1 - \epsilon_0 - 2\exp(-\rho pm)$,

$$kT_1k \leq (1 + \frac{\rho}{8L})k w_1 - w_1 k + \frac{2v}{n^j S_1 \setminus S_{1j}}. \quad (2.22)$$

Bound kT_2k Let us define $T_{2j} := \sum_{i \in S_1 \setminus S_j} r F_i(w_1)$, $j = 2, \dots, m$. We have $T_2 = \frac{1}{m} \sum_{j=2}^m T_{2j}$. We condition on S_1 and first analyze T_{2j} . We have

$$T_{2j} = |jS_1 \setminus S_j| r F^j(w_1) + \sum_{i \in 2S_1 \setminus S_j} r F_i(w_1) - r F^j(w_1). \quad (2.23)$$

Due to the smoothness of $F^j(w)$, we know that

$$k r F^j(w_1) k \leq L k w_1 - w_j k \leq 3L, \quad (2.24)$$

where we use the fact that $k w_1 - w_j k \leq k w_j k + k w_1 k + k w_1 - w_1 k \leq 1 + 1 + \frac{1}{4} \frac{\rho_1}{L} \leq 3$. In addition, we have

$$E \sum_{i \in 2S_1 \setminus S_j}^2 (r F_i(w_1) - r F^j(w_1))^2 \leq |jS_1 \setminus S_j| \frac{v^2}{n^j};$$

which implies

$$E \sum_{i \in 2S_1 \setminus S_j}^2 (r F_i(w_1) - r F^j(w_1))^2 \leq |jS_1 \setminus S_j| \frac{v^2}{n^j};$$

and then according to Markov's inequality, for any $\epsilon_1 \geq (0, 1)$, with probability at least $1 - \epsilon_1$,

$$\prod_{i \in S_1 \setminus S_j} \|F_i(w_1) - F^j(w_1)\| \leq \frac{Q}{j|S_1 \setminus S_j|} \frac{V}{\rho_1 n^{\theta}}. \quad (2.25)$$

Then, by combining (2.24) and (2.25), we know that with probability at least $1 - \epsilon_1$,

$$\|kT_{2j}k \leq 3Lj|S_1 \setminus S_j| + \frac{Q}{j|S_1 \setminus S_j|} \frac{V}{\rho_1 n^{\theta}}. \quad (2.26)$$

By union bound, we know that with probability at least $1 - k\epsilon_1$, (2.26) applies to all $j \in \{1, \dots, k\}$. Then, we have with probability at least $1 - k\epsilon_1$,

$$\|kT_2k \leq \frac{3L}{m} j|S_1 \setminus \bar{S}_1^j| + \frac{V \rho_k}{m \rho_1 n^{\theta}} \frac{Q}{j|S_1 \setminus \bar{S}_1^j|}. \quad (2.27)$$

According to Lemma 3, we know that

$$\mathbb{E}[j|S_1 \setminus \bar{S}_1^j|] \leq c_1 \frac{2m}{2^2 4n^{\theta}}.$$

Then by Markov's inequality, we know that with probability at least $1 - \epsilon_2$,

$$j|S_1 \setminus \bar{S}_1^j| \leq c_1 \frac{2m}{2^2 4n^{\theta}}. \quad (2.28)$$

Now we combine (2.27) with (2.28) and obtain with probability at least $1 - k\epsilon_1 - \epsilon_2$,

$$\|kT_2k \leq c_1 \frac{2}{2^2 4n^{\theta}} + c_2 \frac{V \rho_k}{\rho_1 2^2 L} \frac{Q}{2^2 mn^{\theta}}. \quad (2.29)$$

Combining (2.22) and (2.29), we know that with probability at least $1 - \epsilon_0 - k\epsilon_1 - \epsilon_2 - 2 \exp(-cpm)$,

$$\|kw_1^* - w_1k \leq \left(1 + \frac{\rho}{8L}\right) kw_1 - w_1k + \frac{2V}{\rho_0 L} \frac{Q}{\rho mn^{\theta}} + c_1 \frac{2}{2^2 4n^{\theta}} + c_2 \frac{V \rho_k}{\rho_1 2^2 L} \frac{Q}{2^2 mn^{\theta}}. \quad (2.30)$$

In the following, we let $\epsilon_3 := \epsilon_0 + k\epsilon_1 + \epsilon_2 + 2 \exp(-cpm)$, and

$$\epsilon_0 = \frac{2V}{\rho_0 L} \frac{Q}{\rho mn^{\theta}} + c_1 \frac{2}{2^2 4n^{\theta}} + c_2 \frac{V \rho_k}{\rho_1 2^2 L} \frac{Q}{2^2 mn^{\theta}}.$$

Let us simplify this expression. We first choose $\epsilon_2 \geq (0, 1)$ as the failure probability of the entire algorithm. Then, we choose

$$\epsilon_0 = \frac{\rho}{CkL \log(mn^{\theta})}; \quad \epsilon_1 = \frac{\rho}{Ck^2L \log(mn^{\theta})}; \quad \epsilon_2 = \frac{\rho}{CkL \log(mn^{\theta})};$$

for some constant $C > 0$ that is large enough. In addition, since we assume that $\rho \ll \frac{\log(mn^\theta)}{m}$, we have $\exp(-cpm) = 1 - \text{poly}(mn^\theta) \cdot \frac{\rho}{kL \log(mn^\theta)}$. Consider all these facts, we obtain

$$\beta^3 = \frac{4\rho}{CkL \log(mn^\theta)}; \quad (2.31)$$

$$\beta_0 \leq \frac{vk \log(mn^\theta)}{\rho^{3-2} mn^\theta} + \frac{2Lk \log(mn^\theta)}{\rho^{3-4} n^\theta} + \frac{v k^3 \rho L \log^{3-2}(mn^\theta)}{\rho^{3-2} m^{5-2} n^{3-2} mn^\theta}; \quad (2.32)$$

In addition, by union bound, we know that with probability at least $1 - k\beta_0$, for all $j \in [k]$,

$$\|kw_j^{(t)} - w_j\| \leq (1 - \frac{\rho}{8L})^t \|kw_j^{(0)} - w_j\| + \beta_0; \quad (2.33)$$

2.9.2 Convergence of the algorithm

We now analyze the convergence of the entire algorithm. First, we can verify that as long as

$$\beta_0 \leq \frac{\rho}{32} \left(\frac{\rho}{L}\right)^{3-2}; \quad (2.34)$$

we can guarantee that $\|kw_j^{(t)} - w_j\| \leq \frac{1}{4} \frac{\rho}{L}$. We can also verify that as long as there is

$$\Theta(\max\{\rho^{1-5}; m^{1-6} (n^\theta)^{1-3}\} g); \quad (2.35)$$

using the definition of β_0 in (2.32), we know that (2.34) holds. Here, in the Θ notation, we omit the logarithmic factors and quantities that does not depend on m and n^θ . In this case, we can iteratively apply (2.33) for T iterations and obtain that with probability at least $1 - kT\beta_0$,

$$\|kw_j^{(T)} - w_j\| \leq (1 - \frac{\rho}{8L})^T \|kw_j^{(0)} - w_j\| + \frac{8L}{\rho} \beta_0;$$

Then, we know that when we choose

$$T = \frac{8L}{\rho} \log \frac{\rho}{32\beta_0 L}; \quad (2.36)$$

we have

$$(1 - \frac{\rho}{8L})^T \|kw_j^{(0)} - w_j\| \leq \exp(-\frac{\rho}{8L} T) \frac{1}{4} \frac{\rho}{L} \leq \frac{8}{\rho} \frac{\rho}{L} \beta_0;$$

which implies $\|kw_j^{(T)} - w_j\| \leq \frac{16L}{\rho} \beta_0$. Finally, we check the failure probability. The failure probability is

$$kT\beta_0 \leq \frac{8kL}{\rho} \log \frac{\rho}{32\beta_0 L} \leq \frac{4\rho}{CkL \log(mn^\theta)} = \frac{32}{C} \frac{\log(\frac{\rho}{32\beta_0 L})}{\log(mn^\theta)} \leq \frac{\log(\frac{1}{\beta_0})}{\log((mn^\theta)^{C-32})};$$

On the other hand, according to (2.32), we know that

$$\frac{1}{\epsilon_0} \in \Theta(\max\{f^D, mn^{\theta}; n^{\theta}g\});$$

then, as long as C is large enough, we can guarantee that $(mn^{\theta})^{C=32} > \frac{1}{\epsilon_0}$, which implies that the failure probability is upper bounded by ϵ_0 . Our final error floor can be obtained by redefining

$$\epsilon := \frac{16L}{p} \epsilon_0;$$

2.9.3 Proof of Lemma 3

Without loss of generality, we bound the probability of $E_i^{1:j}$ for some $j \notin 1$. We know that

$$E_i^{1:j} = \bigcap_{j=1}^i F_i(w_j; \mathcal{Z}_j) \cap \bigcirc_{j=1}^i F_i(w_j; \mathcal{Z}_j);$$

where \mathcal{Z}_j is the set of n^{θ} data points that we use to estimate the cluster identity in this iteration. In the following, we use the shorthand notation $F_i(w) := F_i(w; \mathcal{Z}_i)$. We have

$$P(E_i^{1:j}) \leq P(F_i(w_1) > t) + P(F_i(w_j) > t)$$

for all $t \geq 0$. We choose $t = \frac{F^1(w_1) + F^1(w_j)}{2}$. With this choice, we obtain

$$P(F_i(w_1) > t) = P\left(F_i(w_1) > \frac{F^1(w_1) + F^1(w_j)}{2}\right) \tag{2.37}$$

$$= P\left(F_i(w_1) - F^1(w_1) > \frac{F^1(w_j) - F^1(w_1)}{2}\right); \tag{2.38}$$

Similarly, for the second term, we have

$$P(F_i(w_j) > t) = P\left(F_i(w_j) - F^1(w_j) > \frac{F^1(w_j) - F^1(w_1)}{2}\right); \tag{2.39}$$

Based on our assumption, we know that $\|w_j - w_1\| \leq \frac{1}{4} \frac{1}{L} \leq \frac{3}{4}$. According to the strong convexity of $F^1(\cdot)$,

$$F^1(w_j) - F^1(w_1) \geq \frac{k}{2} \|w_j - w_1\|^2 \geq F^1(w_1) + \frac{9}{32} \epsilon^2;$$

and according to the smoothness of $F^1(\cdot)$,

$$F^1(w_1) - F^1(w_1) + \frac{L}{2} \|w_1 - w_1\|^2 \leq F^1(w_1) + \frac{L}{2 \cdot 16L} \epsilon^2 = F^1(w_1) + \frac{\epsilon^2}{32};$$

Therefore, $F^1(w_j) - F^1(w_1) \geq \frac{9}{32} \epsilon^2$. Then, according to Chebyshev's inequality, we obtain that $P(F_i(w_1) > t) \leq \frac{64}{2 \cdot 4n^{\theta}}$ and that $P(F_i(w_j) > t) \leq \frac{64}{2 \cdot 4n^{\theta}}$, which complete the proof.

Chapter 3

Communication-Efficient and Byzantine-Robust Distributed First Order Learning

In this chapter, we develop a communication-efficient distributed learning algorithm that is robust against Byzantine worker machines. We propose and analyze a distributed gradient-descent algorithm that performs a simple thresholding based on gradient norms to mitigate Byzantine failures. Furthermore, for communication efficiency, we consider a generic class of ϵ -approximate compressors from Karimireddi et al. [20] that encompasses sign-based compressors and top- k sparsification. Our algorithm uses compressed gradients and gradient norms for aggregation and Byzantine removal respectively. We establish the statistical error rate for non-convex smooth loss functions. We show that, in certain range of the compression factor ϵ , the (order-wise) rate of convergence is not affected by the compression operation. Moreover, we analyze the compressed gradient descent algorithm with error feedback (proposed in [20]) in a distributed setting and in the presence of Byzantine worker machines. We show that exploiting error feedback improves the statistical error rate. Finally, we experimentally validate our results and show good performance in convergence for convex (least-square regression) and non-convex (neural network training) problems.

3.1 Introduction

We propose provable and efficient algorithms that address the issues of communication efficiency and Byzantine robustness simultaneously. Note that, both these challenges, have recently attracted significant research attention, albeit mostly separately. In particular, several recent works have proposed various quantization or sparsification techniques to reduce the communication overhead ([14–16, 21, 113–116]). The goal of these quantization schemes is to compute an unbiased estimate of the gradient with bounded second moment in order to achieve good convergence guarantees. The problem of developing Byzantine-robust distributed algorithms has been considered in [17–19, 23, 30–33].

A notable exception to considering communication overhead separately from Byzantine robustness is the recent work of [22]. In this work, a sign-based compression algorithm signSGD of [117] is shown to be Byzantine fault-tolerant. The main idea of signSGD is to communicate the coordinate-wise signs of the gradient vector to reduce communication and employ a majority vote during the aggregation to mitigate the effect of Byzantine units. However, signSGD suffers from two major drawbacks. First, sign-based algorithms do not converge in general ([20]). In particular, [20, Section 3] presents several convex counter examples where signSGD fails to converge even though [22, Theorem 2] shows convergence guarantee for non-convex objective under certain assumptions. Second, signSGD can handle only a limited class of adversaries, namely *blind multiplicative adversaries* ([22]). Such an adversary manipulates the gradients of the worker machines by multiplying it (element-wise) with a vector that can scale and randomize the sign of each coordinate of the gradient. However, the vector must be chosen before observing the gradient (hence ‘blind’).

In this work, we develop communication-efficient and robust learning algorithms that overcome both these drawbacks¹. Specifically, we consider the following distributed learning setup. There are m worker machines, each storing n data points. The data points are generated from some unknown distribution D . The objective is to learn a parametric model that minimizes a population loss function $F : W \rightarrow \mathbb{R}$, where F is defined as an expectation over D , and $W \subseteq \mathbb{R}^d$ denotes the parameter space. We choose the loss function F to be non-convex. With the rapid rise of the neural networks, the study of *local minima* in non-convex optimization framework has become imperative [35, 36]. For gradient compression at workers, we consider the notion of a ϵ -approximate compressor from [20] that encompasses sign-based compressors like QSGD ([21]), ℓ_1 -QSGD ([20]) and top- k sparsification ([15]). We assume that $0 < \epsilon < 1/2$ fraction of the worker machines are Byzantine. In contrast to blind multiplicative adversaries, we consider unrestricted adversaries.

Our key idea is to use a simple threshold (on local gradient norms) based Byzantine resilience scheme in contrast with computationally complex robust aggregation methods such as coordinate wise median or trimmed mean of [23]. Our main result is to show that, for a wide range of compression factor ϵ , the statistical error rate of our proposed threshold-based scheme is (order-wise) identical to the case of no compression considered in [23]. In fact, our algorithm achieves order-wise optimal error-rate in parameters $(\epsilon; n; m)$. Furthermore, to alleviate convergence issues associated with sign-based compressors, we employ the technique of error-feedback from [20]. In this setup, the worker machines store the difference between the actual and compressed gradient and add it back to the next step so that the *correct* direction of the gradient is not forgotten. We show that using error feedback with our threshold based Byzantine resilience scheme not only achieves better statistical error rate but also improves the rate of convergence. We outline our specific contributions in the following.

Our Contributions: We propose a communication-efficient and robust distributed gradient descent (GD) algorithm. The algorithm takes as input the gradients compressed using a ϵ -approximate compressor along with the norms² (of either compressed or uncompressed gradients), and performs a simple thresholding operation on based on gradient norms to discard $\epsilon > \epsilon$ fraction of workers with

¹We compare our algorithm with signSGD in Section 3.8.

²We can handle any convex norm.

the largest norm values. We establish the statistical error rate of the algorithm for arbitrary smooth population loss functions as a function of the number of worker machines m , the number of data points on each machine n , dimension d , and the compression factor α . In particular, we show that our algorithm achieves the following statistical error rate³ for the regime $\alpha > 4 + 4\sqrt{8d^2 + 4}^{-3}$:

$$\mathcal{O}\left(d^2 \frac{2}{n} + \frac{1}{n} + \frac{1}{mn}\right) \quad (3.1)$$

We first note that when $\alpha = 1$ (uncompressed), the error rate is $\mathcal{O}(d^2[\frac{2}{n} + \frac{1}{mn}])$, which matches [23]. Notice that we use a simple threshold (on local gradient norms) based Byzantine resilience scheme in contrast with the coordinate wise median or trimmed mean of [23]. We note that for a fixed d and the compression factor α satisfying $\alpha > 1 + 2\sqrt{d^2}$, the statistical error rate become $\mathcal{O}(\frac{2}{n} + \frac{1}{mn})$, which is order-wise identical to the case of no compression [23]. In other words, in this parameter regime, the compression term does not contribute (order-wise) to the statistical error. Moreover, it is shown in [23] that, for strongly-convex loss functions and a fixed d , no algorithm can achieve an error lower than $\sim (\frac{2}{n} + \frac{1}{mn})$, implying that our algorithm is order-wise optimal in terms of the statistical error rate in the parameters $(\alpha; n; m)$.

Furthermore, we strengthen our distributed learning algorithm by using error feedback to correct the direction of the local gradient (Algorithm 3). We show (both theoretically and via experiments) that using error-feedback with a α -approximate compressor indeed speeds up the convergence rate and attains better (statistical) error rate. Under the assumption that the gradient norm of the local loss function is upper-bounded by G , we obtain the following (statistical) error rate:

$$\mathcal{O}\left(d^2 \frac{2}{n} + \frac{(1 - \alpha)^2}{d^2} + \frac{1}{mn}\right)$$

provided a similar $(\alpha; \epsilon)$ trade-off⁴. We note that in the no-compression setting ($\alpha = 1$), we recover the $\mathcal{O}(\frac{2}{n} + \frac{1}{mn})$ rate. Furthermore, in Section 3.7.2, we argue that, when $\alpha = 1$, the error rate of Algorithm 3 is strictly better than that of Algorithm 2. In experiments (Section 3.8), we also see a reflection of this fact.

We experimentally evaluate our algorithm for convex and non-convex losses. For the convex case, we choose the linear regression problem, and for the non-convex case, we train a ReLU activated feed-forward fully connected neural net. We compare our algorithm with the non-Byzantine case and signSGD with majority vote, and observe that our algorithm converges faster using the standard MNIST dataset.

A major technical challenge is to handle compression and the Byzantine behavior of the worker machines simultaneously. We build up on the techniques of [23] to control the Byzantine machines. In particular, using certain distributional assumption on the partial derivative of the loss function and exploiting uniform bounds via careful covering arguments, we show that the local gradient on a non-Byzantine worker machine is close to the gradient of the population loss function.

³Throughout the paper $\mathcal{O}(\cdot)$ hides multiplicative constants, while $\mathcal{O}(\cdot)$ further hides logarithmic factors.

⁴See Theorem 5 for details.

Note that in some settings, our results may not have an optimal dependence on dimension d . This is due to the norm-based Byzantine removal schemes. Obtaining optimal dependence on d is an interesting future direction.

Organization: We describe the problem formulation in Sec. 3.2, and give a brief overview of ρ -compressors in Sec. 3.3. Then, we present our proposed algorithm in Sec. 2.4. We analyze the algorithm, first, for a *restricted* (as described next) adversarial model in Section 3.5, and in the subsequent section, remove this restriction. In Section 3.5, we restrict our attention to an adversarial model in which Byzantine workers can provide arbitrary values as an input to the compression algorithm, but they correctly implement the same compression scheme as mandated. In Section 3.6, we remove this restriction on the Byzantine machines. As a consequence, we observe (in Theorem 4) that the modified algorithm works under a stricter assumption, and performs slightly worse than the one in restricted adversary setting. In Section 3.7, we strengthen our algorithm by including error-feedback at worker machines, and provide statistical guarantees for non-convex smooth loss functions. We show that error-feedback indeed improves the performance of our optimization algorithm in the presence of arbitrary adversaries.

3.1.1 Related Work

Gradient Compression: The foundation of gradient quantization was laid in [118] and [119]. In the work of [21, 114, 115] each co-ordinate of the gradient vector is represented with a small number of bits. Using this, an unbiased estimate of the gradient is computed. In these works, the communication cost is $\binom{p}{d}$ bits. In [113], a quantization scheme was proposed for distributed mean estimation. The tradeoff between communication and accuracy is studied in [120]. Variance reduction in communication efficient stochastic distributed learning has been studied in [121]. Sparsification techniques are also used instead of quantization to reduce communication cost. Gradient sparsification has been studied in [14–16] with provable guarantees. The main idea is to communicate top components of the d -dimensional local gradient to get good estimate of the true global gradient.

Byzantine Robust Optimization: In the distributed learning context, a generic framework of one shot median based robust learning has been proposed in [19]. In [30] the issue of byzantine failure is tackled by grouping the servers in batches and computing the median of batched servers. Later in [23, 31], co-ordinate wise median, trimmed mean and iterative filtering based algorithm have been proposed and optimal statistical error rate is obtained. Also, [122, 123] considers adversaries may steer convergence to bad local minimizers. In this work, we do not assume such adversaries.

Gradient compression and Byzantine robust optimization have simultaneously been addressed in a recent paper [22]. Here, the authors use signSGD as compressor and majority voting as robust aggregator. As explained in [20], signSGD can run into convergence issues. Also, [22] can handle a restricted class of adversaries that are *multiplicative* (i.e., multiply each coordinate of gradient by arbitrary scalar) and *blind* (i.e., determine how to corrupt the gradient before observing the true

gradient). Here, for compression, we use a generic approximate compressor. Also, we can handle arbitrary Byzantine worker machines.

Very recently, [20] uses error-feedback to remove some of the issues of sign based compression schemes. In this work, we extend the framework to a distributed setting and prove theoretical guarantees in the presence of Byzantine worker machines.

Notation: We assume $C; C_1; C_2; \dots; c; c_1; \dots$ as positive universal constants, the value of which may differ from instance to instance. $[r]$ denotes the set of natural numbers $\{1; 2; \dots; r\}$. Also, $\| \cdot \|$ denotes the ℓ_2 norm of a vector and the operator norm of a matrix unless otherwise specified.

3.2 Problem Formulation

In this section, we formally set up the problem. We consider a standard statistical problem of risk minimization. In a distributed setting, suppose we have one central and m worker nodes and the worker nodes communicate to the central node. Each worker node contains n data points. We assume that the mn data points are sampled independently from some unknown distribution D . Also, let $f(w; z)$ be the non-convex loss function of a parameter vector $w \in \mathcal{W} \subseteq \mathbb{R}^d$ corresponding to data point z , where \mathcal{W} is the parameter space. Hence, the population loss function is $F(w) = \mathbb{E}_z [f(w; z)]$. Our goal is to obtain the following:

$$w = \operatorname{argmin}_{w \in \mathcal{W}} F(w);$$

where we assume \mathcal{W} to be a convex and compact subset of \mathbb{R}^d with diameter D . In other words, we have $\|w_1 - w_2\| \leq D$ for all $w_1; w_2 \in \mathcal{W}$. Each worker node is associated with a local loss defined as $F_i(w) = \frac{1}{n} \sum_{j=1}^n f(w; z^{ij})$, where z^{ij} denotes the j -th data point in the i -th machine. This is precisely the empirical risk function of the i -th worker node.

We assume a setup where worker i compresses the local gradient and sends to the central machine. The central machine aggregates the compressed gradients, takes a gradient step to update the model and broadcasts the updated model to be used in the subsequent iteration. Furthermore, we assume that a fraction of the total workers nodes are Byzantine, for some $\alpha < 1/2$. Byzantine workers can send any arbitrary values to the central machine. In addition, Byzantine workers may completely know the learning algorithm and are allowed to collude with each other.

Next, we define a few (standard) quantities that will be required in our analysis.

Definition 6. (Sub-exponential random variable) A zero mean random variable Y is called v -sub-exponential if $\mathbb{E}[e^{jY}] \leq e^{\frac{1}{2} v^2 |j|}$, for all $j \in \mathbb{Z}$.

Definition 7. (Smoothness) A function $h(\cdot)$ is L_F -smooth if $h(w) \leq h(w^0) + L_F \|w - w^0\| + \frac{L_F}{2} \|w - w^0\|^2$.

Definition 8. (Lipschitz) A function $h(\cdot)$ is L -Lipschitz if $\|h(w) - h(w^0)\| \leq L \|w - w^0\|$.

3.3 Compression At Worker Machines

In this section, we consider a generic class of compressors from [15] and [20] as described in the following.

Definition 9 (ϵ -Approximate Compressor). An operator $Q(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}^d$ is defined as ϵ -approximate compressor on a set $S \subseteq \mathbb{R}^d$ if, $\forall x \in S$,

$$\|Q(x) - x\|^2 \leq (1 - \epsilon) \|x\|^2;$$

where $\epsilon \in (0; 1]$ is the compression factor.

Furthermore, a randomized operator $Q(\cdot)$ is ϵ -approximate compressor on a set $S \subseteq \mathbb{R}^d$ if,

$$\mathbb{E} \|Q(x) - x\|^2 \leq (1 - \epsilon) \|x\|^2$$

holds for all $x \in S$, where the expectation is taken with respect to the randomness of $Q(\cdot)$. For the clarity of exposition, we consider the deterministic form of the compressor (as in Definition 9). However, the results can be easily extended for randomized $Q(\cdot)$.

Notice that $\epsilon = 1$ implies $Q(x) = x$ (no compression). We list a few examples of ϵ -approximate compressors (including a few from [20]) here:

1. top_k operator, which selects k coordinates with largest absolute value; for $1 \leq k \leq d$, $(Q(x))_i = (x)_{(i)}$ if $i \leq k$, and 0 otherwise, where $(\cdot)_{(i)}$ is a permutation of $[d]$ with $(jx)_{(i)} \leq (jx)_{(i+1)}$ for $i \in [d - 1]$. This is a $k=d$ -approximate compressor.
2. k -PCA that uses top k eigenvectors to approximate a matrix X ([114]).
3. Quantized SGD (QSGD) [21], where $Q(x_i) = \frac{\|x_i\|}{\|x\|} \text{sgn}(x_i) \cdot \tilde{x}_i(x)$, where $\text{sgn}(x_i)$ is the coordinate-wise sign vector, and $\tilde{x}_i(x)$ is defined as following: let $0 \leq l \leq s$, be an integer such that $\sum_{j=1}^l |x_j| = l$ and $\sum_{j=1}^{l+1} |x_j| > l$. Then, $\tilde{x}_i = l$ with probability $1 - \frac{|x_{l+1}|}{\|x\|}$ and $(l + 1)$ otherwise. [21] shows that it is a $1 - \frac{\|x\|}{\|x\|} = 0$ -approximate compressor.
4. Quantized SGD with ℓ_1 norm [20], $Q(x) = \frac{\|x\|_1}{d} \text{sgn}(x)$, which is $\frac{\|x\|_1^2}{d\|x\|^2}$ -approximate compressor. Here, we call this compression scheme as ℓ_1 -QSGD.

Apart from these examples, several randomized compressors are also discussed in [15]. Also, the *signSGD* compressor, $Q(x) = \text{sgn}(x)$, where $\text{sgn}(x)$ is the (coordinate-wise) sign operator, was proposed in [22, 117]. Here the local machines send a d -dimensional vector containing coordinate-wise sign of the gradients.

Algorithm 2: Robust Compressed Gradient Descent

1: **Input:** Step size η , Compressor $Q(\cdot)$, $q > 1$, $\epsilon < 1$. Also define,

$$C(x) = \begin{cases} fQ(x); kxk_q \leq \epsilon & \text{Option I} \\ fQ(x); kQ(x)k_q \leq \epsilon & \text{Option II} \end{cases}$$

2: **Initialize:** Initial iterate $w_0 \in W$

3: **for** $t = 0; 1; \dots; T - 1$ **do**

4: Central machine: broadcasts w_t
for $i \in [m]$ **do in parallel**

5: i -th worker machine:

- Non-Byzantine:
 - Computes $r F_i(w_t)$; sends $C(r F_i(w_t))$ to the central machine,
- Byzantine:
 - Generates \tilde{r} (arbitrary), and sends $C(\tilde{r})$ to the central machine: Option I,
 - Sends \tilde{r} to the central machine: Option II,

end for

6: Central Machine:

- Sort the worker machines in a non decreasing order according to
 - Local gradient norm: Option I,
 - Compressed local gradient norm: Option II,
- Return the indices of the first $\lfloor \epsilon m \rfloor$, fraction of elements as U_t ,
- Update model parameter: $w_{t+1} = w_t - \eta \frac{1}{|U_t|} \sum_{i \in U_t} Q(r F_i(w_t))$.

7: **end for**

3.4 Robust Compressed Gradient Descent

In this section, we describe a communication-efficient and robust distributed gradient descent algorithm for ϵ -approximate compressors. The optimization algorithm we use is formally given in Algorithm 2. Note that the algorithm uses a compression scheme $Q(\cdot)$ to reduce communication cost and a norm based thresholding to remove Byzantine worker nodes. As seen in Algorithm 2,

robust compressed gradient descent operates under two different setting, namely *Option I* and *Option II*.

Option I and II are analyzed in Sections 3.5 and 3.6 respectively. For Option I, we use a q -approximate compressor along with the norm information. In particular, we use: $\mathcal{C}(x) = \lfloor kx \rfloor_{k_q}; Q(x)g$ where $q \geq 1$. $\mathcal{C}(x)$ is comprised of a scalar (norm of x) and a compressed vector $Q(x)$. For compressors such as QSGD ([21]) and q -QSGD ([20]), the quantity $Q(\cdot)$ has the norm information and hence sending the norm separately is not required.

As seen in Option I of Algorithm 2, worker node i compresses the local gradient $r F_i(\cdot)$ sends $\mathcal{C}(r F_i(\cdot))$ to the central machine. Adversary nodes can send arbitrary $\mathcal{C}(\cdot)$ to the central machine. The central machine aggregates the gradients, takes a gradient step and broadcasts the updated model for next iteration.

For Option I, we restrict to the setting where the Byzantine worker machines can send arbitrary values to the input of the compression algorithm, but they adhere to the compression algorithm. In particular, Byzantine workers can provide arbitrary values, \cdot to the input of the compression algorithm, $Q(\cdot)$ but they correctly implement the same compression algorithm, i.e., computes $Q(\cdot)$.

We now explain how Algorithm 2 tackles the Byzantine worker machines. The central machine receives the compressed gradients comprising a scalar ($\|x\|_q; q \geq 1$) and a quantized vector ($Q(x)$) and outputs a set of indices U with $|U| = (1 - \epsilon)m$. Here we employ a simple thresholding scheme on the (local) gradient norm. Note that, if the Byzantine worker machines try to diverge the learning algorithm by increasing the norm of the local gradients; Algorithm 2 can identify them as outliers. Furthermore, when the Byzantine machines behave like inliers, they can not diverge the learning algorithm since $\epsilon < 1/2$. In the subsequent sections, we show theoretical justification of this argument.

With Option II, we remove this restriction on Byzantine machines at the cost of slightly weakening the convergence guarantees. This is explained in Section 3.6. With Option II, the i -th local machine sends $\mathcal{C} = \lfloor Q(r F_i(w_t)); kQ(r F_i(w_t))k_q \rfloor g$ to the central machine, where $q \geq 1$. Effectively, the i -th local machine just sends $Q(r F_i(w_t))$ since its norm can be computed at the central machine. Byzantine workers just send arbitrary (\cdot) vector instead of compressed local gradient. Note that the Byzantine workers here do not adhere to any compression rule.

The Byzantine resilience scheme with Option II is similar to Option I except the fact that the central machine sorts the worker machines according to the norm of the compressed gradients rather than the norm of the gradients.

3.5 Distributed Learning with Restricted Adversaries

In this section, we analyze the performance of Algorithm 4 with *Option I*. We restrict to an adversarial model in which Byzantine workers can provide arbitrary values to the input of the compression algorithm, but they adhere to the compression rule. Though this adversarial model is restricted, we argue that it is well-suited for applications wherein compression happens outside of worker machines. For example, Apache MXNet, a deep learning framework designed to be distributed on cloud infrastructures, uses NVIDIA Collective Communication Library (NCCL) that

employs gradient compression (see [124]). Also, in a Federated Learning setup the compression can be part of the communication protocol. Furthermore, this can happen when worker machines are divided into groups, and each group is associated with a *compression unit*. As an example, cores in a multi-core processor ([125]) acting as a group of worker machines with the compression carried out by a separate processor, or servers co-located on a rack ([126]) acting as a group with the compression carried out by the top-of-the-rack switch.

3.5.1 Main Results

We analyze Algorithm 4 (with Option I) and obtain the rate of the convergence under non-convex loss functions. We start with the following assumption.

Assumption 9. For all z , the partial derivative of the loss function $f(\cdot; z)$ with respect to the k -th coordinate (denoted as $\partial_k f(\cdot; z)$) is L_k Lipschitz with respect to the first argument for each $k \in [d]$, and let $\mathbb{P} = \prod_{k=1}^d L_k^2$. The population loss function $F(\cdot)$ is L_F smooth.

We also make the following assumption on the tail behavior of the partial derivative of the loss function.

Assumption 10. (Sub-exponential gradients) For all $k \in [d]$ and z , the quantity $\partial_k f(w; z)$ is ν sub-exponential for all $w \in \mathcal{W}$.

The assumption implies that the moments of the partial derivatives are bounded. We like to emphasize that the sub-exponential assumption on gradients is fairly common ([18, 23]). For instance, [23, Proposition 2] gives a concrete example of coordinate-wise sub-exponential gradients in the context of a regression problem. Furthermore, in [31], the gradients are assumed to be sub-gaussian, which is stronger than Assumption 10.

To simplify notation and for the clarity of exposition, we define the following three quantities which will be used throughout the chapter.

$$\rho_1 = \nu^{\frac{d}{2}} \max \left\{ \frac{d}{n} \log(1 + 2nD\hat{L}d); \frac{d}{n} \log(1 + 2nD\hat{L}d) + \frac{1}{n} \right\}; \quad (3.2)$$

$$\rho_2 = \nu^{\frac{d}{2}} \max \left\{ \frac{d}{(1 - \epsilon)mn} \log(1 + 2(1 - \epsilon)mnD\hat{L}d); \frac{d}{(1 - \epsilon)mn} \log(1 + 2(1 - \epsilon)mnD\hat{L}d)^\circ \right\}; \quad (3.3)$$

$$= 2 \left(1 + \frac{1}{\epsilon_0} \right) \frac{1}{1 - \epsilon} \rho_2^2 + \frac{\rho_1}{1 - \epsilon} + \frac{1}{1 - \epsilon} \rho_2^2; \quad (3.4)$$

where ϵ_0 is a positive constant. For intuition, one can think of $\rho_1 = \mathcal{O}(\nu^{\frac{d}{2}})$ and $\rho_2 = \mathcal{O}(\nu^{\frac{d}{2}})$ as small problem dependent quantities. Assuming $\nu = c$ for a universal constant $c > 1$, we have

$$= \mathcal{O} \left(c^{\frac{d}{2}} \frac{1}{n} + \frac{1}{n} + \frac{1}{mn} \right); \quad (3.5)$$

Assumption 11. (Size of parameter space W) Suppose that $\|F(w)\| \leq M$ for all $w \in W$. We assume that W contains the ℓ_2 ball $\{w : \|w - w_0\| \leq c[(2 + \frac{c_0}{2})M + \rho] \frac{F(w_0) - F(w^*)}{g}$, where c_0 is a constant, ρ is the compression factor, w_0 is the initial parameter vector and ρ is defined in equation (3.4).

We use the above assumption to ensure that the iterates of Algorithm 4 stays in W . We emphasize that this is a standard assumption on the size of W to control the iterates for non-convex loss function. Note that, similar assumptions have been used in prior works [23, Assumption 5], [31]. We point out that Assumption 11 is used for simplicity and is not a hard requirement. We show (in the proof of Theorem 3) that the iterates of Algorithm 4 stay in a bounded set around the initial iterate w_0 . Also, note that the dependence of M in the final statistical rate (implicit, via diameter D) is logarithmic (weak dependence), as will be seen in Theorem 3.

We provide the following rate of convergence to a critical point of the (non-convex) population loss function $F(\cdot)$.

Theorem 3. Suppose Assumptions 9, 10 and 11 hold, and $\rho < 1/2$. For sufficiently small constant c , we choose the step size $\eta = \frac{c}{L_F}$. Then, running Algorithm 4 for $T = C_3 \frac{L_F(F(w_0) - F(w^*))}{\rho}$ iterations yields

$$\min_{t=0, \dots, T} \|F(w_t)\|^2 \leq C ;$$

with probability greater than or equal to $1 - \frac{c_1(1-\rho)^{md}}{(1+nLD)^d} - \frac{c_2 d}{(1+(1-\rho)^{mnLD})^d}$, provided the compression factor satisfies $\rho > \frac{1}{9} + \frac{4}{9} \frac{1}{\rho^3}$, where $\rho_0 = 1 - \frac{(1-\rho)^2}{1+\rho}$ and ρ_0 is a (sufficiently small) positive constant.

A few remarks are in order. In the following remarks, we fix the dimension d , and discuss the dependence of ρ on $(\rho; n; m)$.

Remark 1. We observe, from the definition of ρ that the price for compression is $\mathcal{O}(\frac{1}{\rho})$.

Remark 2. Substituting $\rho = 1$ (no compression) in ρ , we get $\rho = \mathcal{O}(\frac{2}{n} + \frac{1}{mn})$, which matches the (statistical) rate of [23]. A simple *norm based thresholding* operation is computationally simple and efficient in the high dimensional settings compared to the coordinate wise median and trimmed mean to achieve robustness and obtain the the same statistical error and iteration complexity as [23]

Remark 3. When the compression factor ρ is large enough, satisfying $\rho > 1 - \frac{2}{n}$, we obtain $\rho = \mathcal{O}(\frac{2}{n} + \frac{1}{mn})$. In this regime, the iteration complexity and the final statistical error of Algorithm 4 is order-wise identical to the setting with no compression [23]. We emphasize here that a reasonable high ρ is often observed in practical applications like training of neural nets [20, Figure 2].

Remark 4. (Optimality) For a distributed mean estimation problem, Observation 1 in [23] implies that any algorithm will yield an (statistical) error of $\mathcal{O}(\frac{2}{n} + \frac{d}{mn})$. Hence, in the regime where $\rho > 1 - \frac{2}{n}$, our error-rate is optimal.

Remark 5. For the convergence of Algorithm 4, we require $\epsilon > \epsilon_0 + 4 \frac{9 \epsilon^2 + 4 \epsilon^3}{\epsilon}$, implying that our analysis will not work if ϵ is very close to 0. Note that a very small ϵ does not give good accuracy in practical applications [20, Figure 2]. Also, note that, from the definition of ϵ_0 , we can choose ϵ_0 sufficiently small at the expense of increasing the multiplicative constant in ϵ by a factor of $1/\epsilon_0$. Since the error-rate considers asymptotic in m and n , increasing a constant factor is insignificant. A sufficiently small ϵ_0 implies $\epsilon_0 = O(\epsilon)$, and hence we require $\epsilon > 4 \epsilon + 2 \epsilon$ (ignoring the higher order dependence).

Remark 6. The requirement $\epsilon > 4 \epsilon + 2 \epsilon$ can be seen as a trade-off between the amount of compression and the fraction of adversaries in the system. As ϵ increases, the amount of (tolerable) compression decreases and vice versa.

Remark 7. (Rate of Convergence) Algorithm 4 with T iterations yields

$$\min_{t=0::T} k r F(w_t) k^2 \leq \frac{C_1 L_F (F(w_0) - F(w^*))}{T + 1} + C_2$$

with high probability. We see that Algorithm 4 converges at a rate of $O(1/T)$, and finally plateaus at an error floor of C_2 . Note that the rate of convergence is same as [23]. Hence, even with compression, the (order-wise) convergence rate is unaffected.

3.6 Distributed Optimization with Arbitrary Adversaries

In this section we remove the assumption of restricted adversary (as in Section 3.5) and make the learning algorithm robust to the adversarial effects of both the computation and compression unit. In particular, here we consider Algorithm 4 with Option II. Hence, the Byzantine machines do not need to adhere to the mandated compression algorithm. However, in this setting, the statistical error-rate of our proposed algorithm is slightly weaker than that of Theorem 3. Furthermore, the $(\epsilon; \epsilon)$ trade-off is stricter compared to Theorem 3.

3.6.1 Main Results

We continue to assume that the population loss function $F(\cdot)$ is smooth and non-convex and analyze Algorithm 4 with Option II. We have the following result. For the clarity of exposition, we define the following quantity which will be used in the results of this section:

$$e = 2 \left(1 + \frac{1}{\epsilon_0}\right) \frac{(1 + \epsilon)^2 \frac{1}{\epsilon} + \epsilon}{1} + \frac{\epsilon^2}{\epsilon} + \left(\frac{1}{\epsilon}\right)^2 \frac{\epsilon}{2} ;$$

Comparing e with ϵ , we observe that $e > \epsilon$. Also, note that,

$$e = \Theta \left(\epsilon^2 \frac{2}{n} + \frac{1}{n} + \frac{1}{mn} \right) ; \tag{3.6}$$

which suggests that e and ϵ are order-wise similar. We have the following assumption, which parallels Assumption 11, with ϵ replaced by e .

Assumption 12. (Size of parameter space W) Suppose that $\|F(w)\| \leq M$ for all $w \in W$. We assume that W contains the ℓ_2 ball $\{w : \|w - w_0\| \leq c_0(2 + \frac{c_0}{2})M + \frac{c_0}{e} \frac{F(w_0) - F(w^*)}{e}\}$, where c_0 is a constant, β is the compression factor and e is defined in equation (3.6).

Theorem 4. Suppose Assumptions 9, 10 and 12 hold, and $\beta < 1/2$. For sufficiently small constant c , we choose the step size $\eta = \frac{c}{L_F}$. Then, running Algorithm 4 for $T = C_3 \frac{L_F(F(w_0) - F(w^*))}{e}$ iterations yields

$$\min_{t=0, \dots, T} \|F(w_t)\|^2 \leq C e;$$

with probability greater than or equal to $1 - \frac{c_1(1 + \beta)^{md}}{(1 + n\beta D)^d} - \frac{c_2 d}{(1 + (1 + \beta)^{mn\beta D})^d}$, provided the compression factor satisfies $\beta > \beta_0 + 4\beta^2 + 4\beta^3$, where $\beta_0 = 1 - \frac{(1 + \beta)^2}{(1 + \beta)^2(1 + \beta_0)}$ and β_0 is a (sufficiently small) positive constant.

Remark 8. The above result and their consequences resemble that of Theorem 3. Since $e > \beta$, the statistical error-rate in Theorem 4 is strictly worse than that of Theorem 3 (although order-wise they are same).

Remark 9. Note that the definition of β_0 is different than in Theorem 3. For a sufficiently small β_0 , we see $\beta_0 = O(4\beta)$, which implies we require $\beta > 4\beta + 4\beta^2$ for the convergence of Theorem 4. Note that this is a slightly strict requirement compared to Theorem 3. In particular, for a given β , Algorithm 4 with Option II can tolerate less number of Byzantine machines compared to Option I.

Remark 10. The result in Theorem 4 is applicable for arbitrary adversaries, whereas Theorem 3 relies on the adversary being restrictive. Hence, we can view the limitation of Theorem 4 (such as worse statistical error-rate and stricter $(\beta; \beta)$ trade-off) as a price of accommodating arbitrary adversaries.

3.7 Byzantine Robust Distributed Learning with Error Feedback

We now investigate the role of error feedback [20] in distributed learning with Byzantine worker machines. We stick to the formulation of Section 3.1.

In order to address the issues of convergence for sign based algorithms (like *signSGD*), [20] proposes a class of optimization algorithms that uses *error feedback*. In this setting, the worker machine locally stores the error between the actual local gradient and its compressed counterpart. Using this as feedback, the worker machine adds this error term to the compressed gradient in the subsequent iteration. Intuitively, this accounts for correcting the the direction of the local gradient. The error-feedback has its roots in some of the classical communication system like “delta-sigma” modulator and adaptive modulator ([127]).

We analyze the distributed error feedback algorithm in the presence of Byzantine machines. The algorithm is presented in Algorithm 3. We observe that here the central machine sorts the worker machines according to the norm of the compressed local gradients, and ignore the largest β fraction.

Algorithm 3: Distributed Compressed Gradient Descent with Error Feedback

- 1: **Input:** Step size η , Compressor $Q(\cdot)$, parameter α ($\alpha > 0$).
 - 2: **Initialize:** Initial iterate w_0 , $e_i(0) = 0 \ \forall i \in [m]$
 - 3: **for** $t = 0; 1; \dots; T - 1$ **do**
 - 4: Central machine: sends w_t to all worker
 - 5: **for** $i \in [m]$ **do in parallel**
 - 6: i -th non-Byzantine worker machine:
 - computes $p_i(w_t) = \eta F_i(w_t) + e_i(t)$
 - sends $Q(p_i(w_t))$ to the central machine
 - computes $e_i(t + 1) = p_i(w_t) - Q(p_i(w_t))$
 - 7: Byzantine worker machine:
 - sends γ to the central machine.
 - 8: At Central machine:
 - sorts the worker machines in non-decreasing order according to $\|Q(p_i(w_t))\|_k$.
 - returns the indices of the first $(1 - \alpha)$ fraction of elements as U_t .
 - $w_{t+1} = w_t - \eta \frac{1}{|U_t|} \sum_{i \in U_t} Q(p_i(w_t))$
 - 9: **end for**
-

Note that, similar to Section 3.6, we handle arbitrary adversaries. In the subsequent section, we show (both theoretically and experimentally) that the statistical error rate of Algorithm 3 is smaller than Algorithm 4.

3.7.1 Main Results

In this section we analyze Algorithm 3 and obtain the rate of the convergence under non-convex smooth loss functions. Throughout the section, we select η as the step size and assume that Algorithm 3 is run for T iterations. We start with the following assumption.

Assumption 13. For all non-Byzantine worker machine i , the local loss functions $F_i(\cdot)$ satisfy $\| \nabla F_i(x) \|^2 \leq L^2$, where $x \in \mathbb{R}^d$, and w_0, \dots, w_T are the iterates of Algorithm 3.

Note that since $F_i(\cdot)$ can be written as loss over data points of machine i , we observe that the bounded gradient condition is equivalent to the bounded second moment condition for SGD, and have featured in several previous works, see, e.g., [128], [129]. Here, we are using all the data points and (hence no randomness over the choice of data points) perform gradient descent instead

of SGD. Also, note that Assumption 13 is weaker than the bounded second moment condition since we do not require $\|F_i(x)\|^2$ to be bounded for all x ; just when $x \in \mathcal{W}_j g_j^T$.

We also require the following assumption on the size of the parameter space \mathcal{W} , which parallels Assumption 11 and 12.

Assumption 14. (Size of parameter space \mathcal{W}) Suppose that $\|F(w)\| \leq M$ for all $w \in \mathcal{W}$. We assume that \mathcal{W} contains the ℓ_2 ball $\{w : \|w - w_0\| \leq r\}$, where

$$r = \frac{2}{c} + M + \frac{6(1 + \frac{\rho}{1-\rho})}{(1-\rho)} \left(\frac{1}{c} + M + \frac{r}{3(1-\rho)} \right) + \frac{r}{12(1-\rho)};$$

and $(\frac{1}{c}, \frac{2}{c})$ are defined in equations (3.2) and (3.3) respectively.

Similar to Assumption 11 and 12, we use the above assumption to ensure that the iterates of Algorithm 3 stays in \mathcal{W} , and we emphasize that this is a standard assumption to control the iterates for non-convex loss function (see [23, 31]).

To simplify notation and for the clarity of exposition, we define the following quantities which will be used in the main results of this section.

$$\tau_1 = \frac{9(1 + \frac{\rho}{1-\rho})^2}{2c(1-\rho)^2} \left(\frac{1}{c} + M + \left(\frac{r}{3(1-\rho)} \right)^2 \right) + \frac{50}{c} \frac{2}{c}; \quad (3.7)$$

$$\tau_2 = \frac{L^2}{2} \frac{3(1-\rho)^2}{c} + \frac{2L}{c} \frac{2}{c} + \frac{1}{2} + L \frac{9(1 + \frac{\rho}{1-\rho})^2}{c(1-\rho)^2} \left(\frac{1}{c} + M + \left(\frac{r}{3(1-\rho)} \right)^2 \right) + \frac{3(1-\rho)^2}{c}; \quad (3.8)$$

$$\tau_3 = \left(\frac{L^2}{100} + 25L^2 \right) \frac{3(1-\rho)^2}{c}; \quad (3.9)$$

where c is a universal constant.

We show the following rate of convergence to a critical point of the population loss function $F(\cdot)$.

Theorem 5. Suppose Assumptions 9, 10, 13 and 14 hold, and $\frac{1}{c} < 1/2$. Then, running Algorithm 4 for T iterations with step size $\frac{1}{c}$ yields

$$\min_{t=0, \dots, T} \|F(w_t)\|^2 \leq \frac{F(w_0) - F^*}{c(T+1)} + \tau_1 + \tau_2 + \tau_3;$$

with probability greater than or equal to $1 - \frac{c_1(1-\rho)^{md}}{(1+nLD)^d} - \frac{c_2 d}{(1+(1-\rho)^{mnLD})^d}$, provided the compression factor satisfies $\frac{(1 + \frac{\rho}{1-\rho})^2}{(1-\rho)^2} \left[\tau_2 + \tau_3 + \left(\frac{r}{3(1-\rho)} \right)^2 \right] < 0.107$. Here τ_1, τ_2 and τ_3 are defined in equations (3.7), (3.8) and (3.9) respectively.

Remark 11. (Choice of Step Size ρ) Substituting $\rho = \frac{1}{T+1}$, we obtain

$$\min_{t=0, \dots, T} \mathbb{E} \|F(w_t) - \nabla F(w_0)\|^2 \leq \frac{F(w_0) - F^*}{c} \left(\frac{1}{T+1} + \frac{1}{T+1} + \frac{1}{T+1} \right);$$

with high probability. Hence, we observe that the quantity associated with $\frac{1}{T+1}$ goes down at a considerably faster rate ($O(1/T)$) than the other terms and hence can be ignored, when T is large.

Remark 12. Note that when no Byzantine worker machines are present, i.e., $\beta = 0$, we obtain

$$\frac{1}{c} = \frac{50}{c} \frac{1}{2}, \quad \frac{2}{c} = \frac{L^2}{2} \frac{3(1 - \beta)}{c} + \frac{2L}{c} \frac{1}{2}, \quad \frac{3}{c} = \left(\frac{L^2}{100} + 25L^2 \right) \frac{3(1 - \beta)}{c}.$$

Additionally, if $\beta = 1$ (this is quite common in applications like training of neural nets, as mentioned earlier), we obtain $\frac{2}{c} = C(L^2 \frac{1}{2} + L \frac{1}{2})$, and $\frac{3}{c} = C_1 L^2$. Substituting $\frac{2}{c} = O(\frac{d}{mn})$ and for a fixed d , the upper bound in the above theorem is order-wise identical to that of standard SGD in a population loss minimization problem under similar setting.

Remark 13. (No compression setting) In the setting, where $\beta = 1$ (no compression), we obtain

$$\frac{1}{c} = O \left(d^2 \frac{1}{n} + \frac{1}{mn} \right);$$

and

$$\frac{2}{c} = O \left(d^2 L \frac{1}{n} + \frac{1}{mn} \right);$$

and $\frac{3}{c} = 0$. The statistical rate (obtained by making T sufficiently large) of the problem is $\frac{1}{c}$, and this rate matches exactly to that of [23]. Hence, we could recover the optimal rate without compression. Furthermore, this rate is optimal in $(d; m; n)$ as shown in [23].

Remark 14. In the next section, we show that when $\beta \neq 0$ and $\beta = 1$, the statistical error rate of Algorithm 3 is order-wise identical to the no-compression setting. Hence, we get the compression for free. Furthermore, we argue that error feedback improves the statistical (error) rate.

3.7.2 Comparison with Algorithm 4

We now compare the statistical rate of Algorithm 3 with Algorithm 4, where no feedback is used. Recall from Sections 3.5 and 3.6 that the statistical error rate of is given by,

$$\text{no feedback} = O \left(d^2 \frac{1}{n} + \frac{1}{n} + \frac{1}{mn} \right);$$

Let us compare it with the statistical rate of our algorithm, given by

$$\frac{1}{c} = O \left(\frac{d^2}{n} + \frac{2(1 - \beta)}{c} + \frac{d^2}{mn} \right);$$

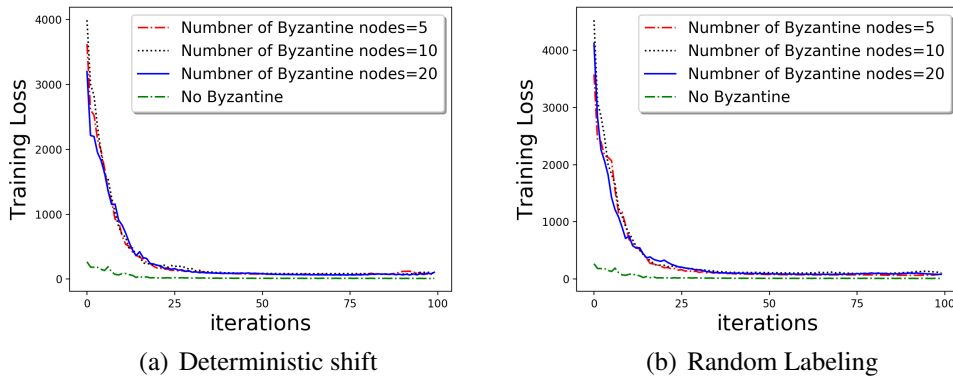


Figure 3.1: Training (cross entropy) loss for MNIST image. Different types of attack (a) labels with deterministic shift (9 label) (b) random labels. Plots show thresholding scheme with different type of byzantine attacks achieve similar convergence as ‘no byzantine’ setup.

Observe that in the setting with error feedback, we have an additional problem parameter ϵ^2 . Hence, for the purpose of comparison, we first argue what the scaling of ϵ^2 should be. Since, $\|k r F_i(w_t)\|^2 = 2k r F_i(w_t) \cdot r F(w_t)k^2 + 2k r F(w_t)k^2$, using Lemma 6 of Appendix 3.10, we have

$$\|k r F_i(w_t)\|^2 \leq 2 \frac{d^2}{n} + 2k r F(w_t)k^2;$$

with high probability. Since from Assumption 13, we obtain $\min_{t=0, \dots, T} \|k r F_i(w_t)\|^2 \geq \epsilon^2$, we obtain

$$\epsilon^2 = \frac{2d^2}{n} + 2 \min_{t=0, \dots, T} \|k r F(w_t)\|^2 ;$$

As seen by [20], $\epsilon^2 = (1)$ is a reasonable and practical parameter regime, and in this setting, with the above ϵ^2 , we observe that

$$1 < \text{no feedback} ;$$

provided $\epsilon^2 \leq c$, for a constant c . Note that this condition is equivalent to the trade-off between the amount of compression and the fraction of Byzantine worker machines, featured in Theorem 5, which was required to show convergence of Algorithm 3. Hence, in the above mentioned parameter regime, the error rate with error feedback is strictly better than no-feedback setting.

In numerical experiments, we observe that the convergence of Algorithm 3 with error feedback is faster than Algorithm 4, which is intuitive since error feedback helps in correcting the direction of the local gradient. We now have a theoretical justification for this fact.

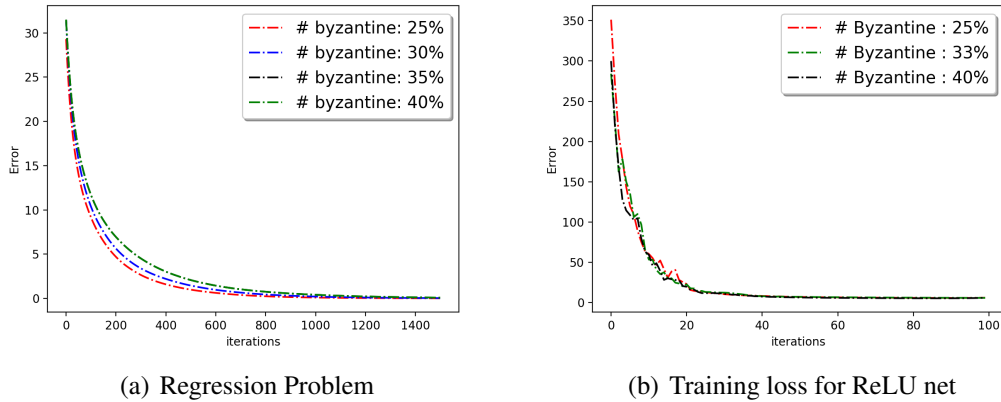


Figure 3.2: Convergence for (a) regression problem (b) training (cross entropy) loss for MNIST image. Plots show convergence beyond the theoretical bound on the number of Byzantine machine.

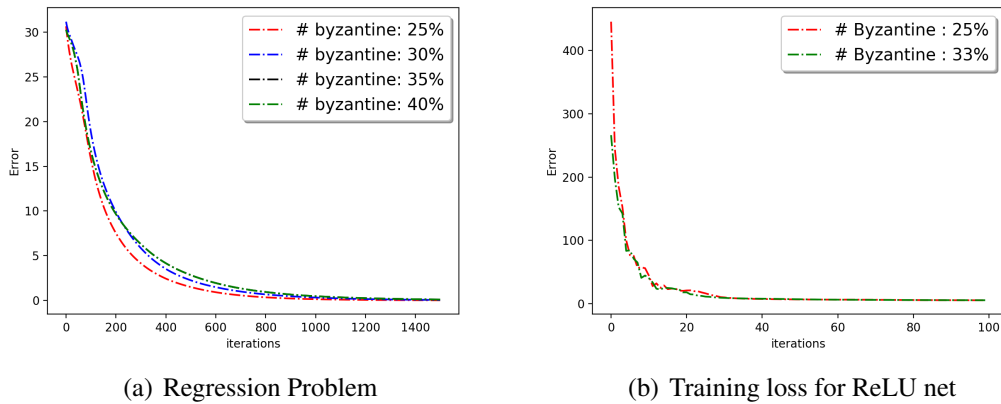


Figure 3.3: Convergence for (a) regression problem (b) training (cross entropy) loss for MNIST image. Plots show convergence with a natural Byzantine attack of γ times the local gradient with high number of Byzantine machines for $\gamma = 0.9$.

3.8 Experiments

In this section we validate the correctness of our proposed algorithms for linear regression problem and training ReLU network. In all the experiments, we choose the following compression scheme: given any $x \in \mathbb{R}^d$, we report $C(x) = \frac{\|x\|_2}{d} \text{sgn}(x)g$ where $\text{sgn}(x)$ serves as the quantized vector and $\frac{\|x\|_2}{d}$ is the scaling factor. All the reported results are averaged over 20 different runs.

First we consider a least square regression problem $w = \arg\min_w \|Aw - b\|_2$. For the regression problem we generate matrix $A \in \mathbb{R}^{N \times d}$, vector $w \in \mathbb{R}^d$ by sampling each item independently from standard normal distribution and set $b = Aw$. Here we choose $N = 4000$ and consider $d = 1000$. We partition the data set equally into $m = 200$ servers. We randomly choose m ($= 10; 20$) workers to be Byzantine and apply norm based thresholding operation with parameter

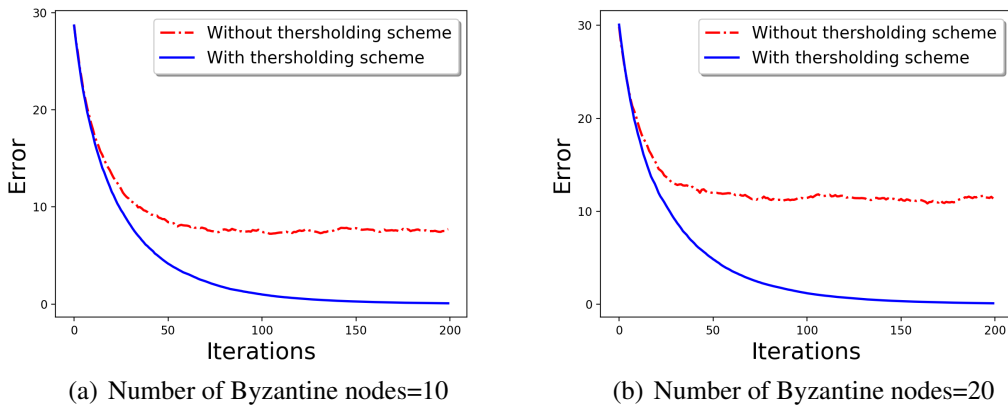


Figure 3.4: Comparison of Robust Compressed Gradient Descent with and without thresholding scheme in a regression problem. The plots show better convergence with thresholding.

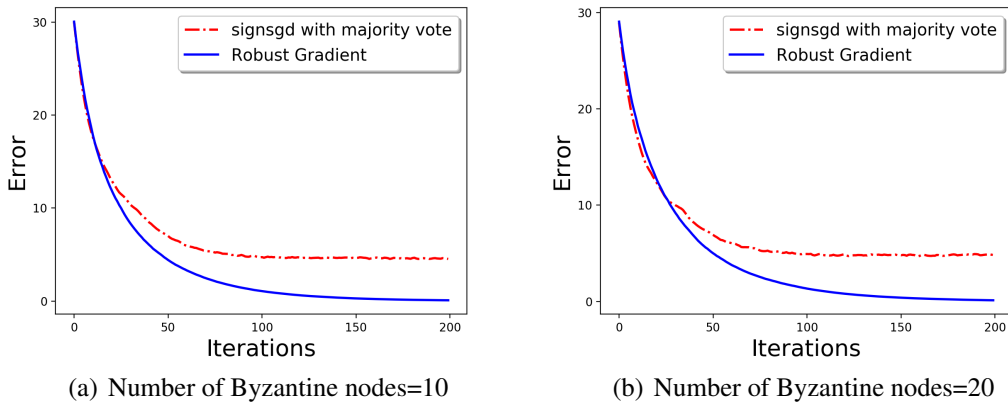


Figure 3.5: Comparison of Robust Compressed Gradient Descent with majority vote based signSGD [22] in regression Problem. The plots show better convergence with thresholding in comparison to the majority vote based robustness of [22]

$m (= 12; 22)$ respectively. We simulate the Byzantine workers by adding i.i.d $N(0; 10/d)$ entries to the gradient. In our experiments the gradient is the most pertinent information of the the worker server. So we choose to add noise to the gradient to make it a Byzantine worker. However, later on, we consider several kinds of attack models. We choose kW_t $w k$ as the error metric for this problem.

Effectiveness of thresholding: We compare Algorithm 4 with compressed gradient descent (with vanilla aggregation). Our method is equipped with Byzantine tolerance steps and the vanilla compressed gradient just computes the average of the compressed gradient sent by the workers. From Figure 3.4 it is evident that the the application of norm based thresholding scheme provides better convergence result compared to the compressed gradient method without it.

Comparison with signSGD with majority vote: In [22], a communication efficient byzantine tolerant algorithm is proposed where communication efficiency is achieved by communicating sign of the gradient and robustness is attained by taking co-ordinate wise majority vote. The robustness in our algorithm comes from thresholding operation on the scaling factor. We show a comparison of both method in Figure 3.5 in the regression setup depicted above. Our method shows a better trend in convergence.

Error-feedback with thresholding scheme: We demonstrate the effectiveness of Byzantine resilience with error-feedback scheme as described in Algorithm 3. We compare our scheme with Algorithm 4 (which does not use error feedback) in Figure 3.6.

Feed-forward Neural Net with ReLU activation: Next, we show the effectiveness of our method in training a fully connected feed forward neural net. We implement the neural net in pytorch and use the digit recognition dataset MNIST ([106]). We partition 60,000 training data into 200 different worker nodes. The neural net is equipped with 1000 node hidden layer with ReLU activation function and we choose *cross-entropy-loss* as the loss function. We simulate the Byzantine workers by adding i.i.d $N(0; 10/d)$ entries to the gradient. In Figure 3.7 we compare our robust compressed gradient descent scheme with the trimmed mean scheme of [23] and majority vote based *signSGD* scheme of [22]. Compared to the majority vote based scheme, our scheme converges faster. Further, our method shows as good as performance of trimmed mean despite the fact the robust scheme of [23] is an uncompressed scheme and uses a more complicated aggregation rules.

Different Types of Attacks: In the previous paragraph we compared our scheme with existing scheme with additive Gaussian noise as a form of byzantine attack. We also show convergence results with the following type of attacks, which are quite common ([23]) in neural net training with digit recognition dataset [106]. (a) *Random label*: the byzantine worker machines randomly replaces the labels of the data, and (b) *Deterministic Shift*: byzantine workers in a deterministic manner replace the labels y with $9 - y$ (0 becomes 9, 9 becomes 0). In Figure 3.1 we show the convergence results with different numbers of byzantine worker nodes.

Large Number of Byzantine Workers: In Figures 3.2 and 3.3, we show the convergence results that holds beyond the theoretical limit (as shown in Theorem 3 and 4) of the number of Byzantine servers in the regression problem and neural net training. In Figure 3.2, for the regression problem, the Byzantine attack is additive Gaussian noise as described before and our algorithm is robust up to 40% ($\epsilon = .4$) of the workers being Byzantine. While training of the feed-forward neural network, we apply a deterministic shift as the Byzantine attack, and the algorithm converges even for 40% ($\epsilon = .4$) Byzantine workers.

Note that our robust algorithm is in essence a stochastic gradient descent algorithm. Thus, a ‘natural’ Byzantine attack would be when a Byzantine worker sends $-g$ where $0 \leq g \leq 1$ and g is the local gradient making the algorithm ‘ascent’ type. We choose $\epsilon = 0.9$ and show convergence

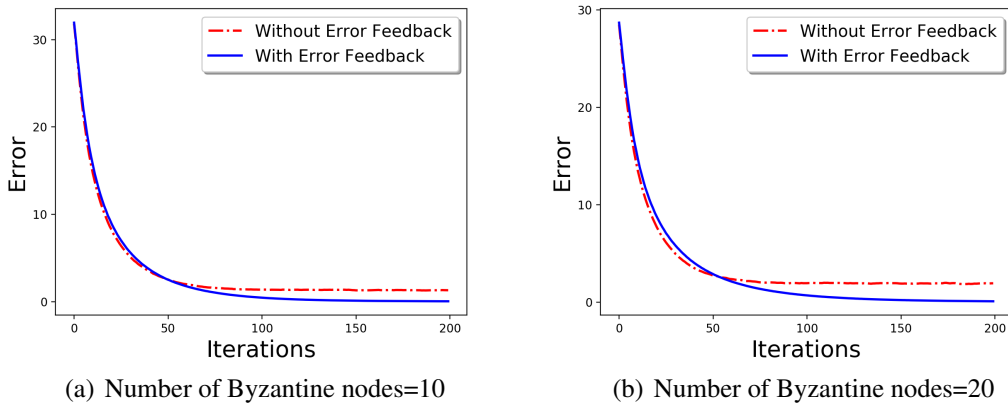


Figure 3.6: Comparison of norm based thresholding with and without error feedback. The plots show that error feedback based scheme offers better convergence.

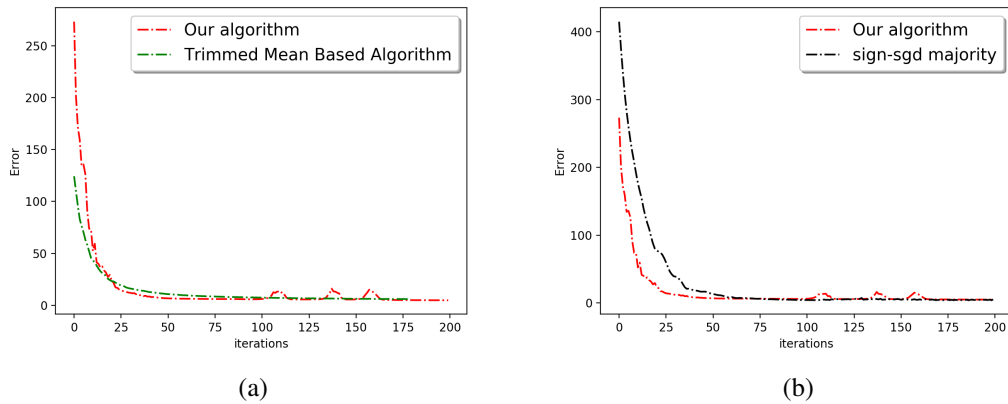


Figure 3.7: Training (cross entropy) loss for MNIST image. Comparison with (a) Uncompressed Trimmed mean [23] (b) majority based signSGD of [22]. In plot (a) show that Robust Gradient descent matches the convergence of the uncompressed trimmed mean [23]. Plot (b) show a faster convergence compared to the algorithm of [22].

for the regression problem for up to 40% byzantine workers, and for the neural network training for up to 33% Byzantine workers in Figure 3.3.

3.9 Conclusion and Open Problems

We address the problem of robust distributed optimization where the worker machines send the compressed gradient (as opposed to the full gradient) to the central machine. We propose a first order optimization algorithm and provide theoretical guarantees and experimental validation under different setup. In some settings, we assume a restricted adversary (that adheres to the compression

algorithm). An immediate future work would be to remove such assumption and obtain a learning algorithm with arbitrary adversaries uniformly for all ϵ -approximate compressors. It might also be interesting to study a second order distributed optimization algorithm with compressed gradients and Hessians.

Appendix

3.10 Analysis of Algorithm 4

In this section, we provide analysis of the Lemmas required for the proof of Theorem 3 and Theorem 4.

Notation: Let \mathcal{M} and \mathcal{B} denote the set of non-Byzantine and Byzantine worker machines. Furthermore, U_t and T_t denote untrimmed and trimmed worker machines. So evidently,

$$|\mathcal{M}| + |\mathcal{B}| = |U_t| + |T_t| = m.$$

3.10.1 Proof of Theorem 3

Let $g(w_t) = \frac{1}{|U_t|} \sum_{i \in U_t} Q(r F_i(w_t))$ and $\tilde{g}(w_t) = g(w_t) - r F(w_t)$. We have the following Lemma to control of $\|k\|^2$.

Lemma 4. For any $\epsilon > 0$, we have,

$$\|k\|^2 \leq (1 + \epsilon) \frac{\rho_1 + 2\epsilon^2}{1} \|kr F(w_t)\|^2 + \tilde{g}(w_t)$$

with probability greater than or equal to $1 - \frac{c_1(1 + \epsilon)^{md}}{(1 + n\epsilon D)^d} - \frac{c_2 d}{(1 + (1 + \epsilon)mn\epsilon D)^d}$, where

$$\tilde{g}(w_t) = 2(1 + \frac{1}{\epsilon}) \frac{\rho_1 + \epsilon}{1} \frac{1}{\epsilon} + \frac{1}{\epsilon} \frac{1}{\epsilon} \frac{\#}{2} :$$

with ρ_1 and ρ_2 as defined in equation (3.2) and (3.3) respectively.

The proof of the lemma is deferred to Section 3.10.3. We prove the theorem using the above lemma.

We first show that with Assumption 11 and with the choice of step size ϵ , we always stay in W without projection. Recall that $g(w_t) = \frac{1}{|U_t|} \sum_{i \in U_t} Q(r F_i(w_t))$ and $\tilde{g}(w_t) = g(w_t) - r F(w_t)$. We have

$$\begin{aligned} \|kw_{t+1} - w\|^2 &\leq \|kw_t - w\|^2 + (kr F(w_t)\|^2 + \|kg(w_t) - r F(w_t)\|^2) \\ &\leq \|kw_t - w\|^2 + \frac{C}{L_F} (\|kr F(w_t)\|^2 + \|k\|^2) \end{aligned}$$

We use Lemma 4 with $\epsilon = \epsilon_0$ for a sufficiently small positive constant ϵ_0 . Define $\epsilon_0 = \frac{1}{1 + \frac{1}{\epsilon_0}}$. A little algebra shows that provided $\epsilon_0 > \frac{1}{9} + \frac{1}{4} \epsilon_0^3$, we obtain

$$k \leq k^2 (1 - c_0)kr F(w_t)k^2 +$$

with probability greater than or equal to $1 - \frac{c_1(1 - \epsilon_0)md}{(1 + n\hat{L}D)^d} - \frac{c_2d}{(1 + (1 - \epsilon_0)mn\hat{L}D)^d}$, where c_0 is a positive constant and ϵ_0 is defined in equation (3.4). Substituting, we obtain

$$\begin{aligned} kW_{t+1} - w &\leq kW_t - w + \frac{c_1}{L_F} (1 + \frac{\rho}{1 - c_0})kr F(w_t)k + \rho \\ kW_t - w &\leq kW_t - w + \frac{c_1}{L_F} (2 - \frac{c_0}{2})kr F(w_t)k + \rho \end{aligned}$$

where we use the fact that $\frac{\rho}{1 - c_0} \leq 1 - c_0 = 2$. Now, running $T = C \frac{L_F(F(w_0) - F(w))}{\epsilon_0}$ iterations, we see that Assumption 11 ensures that the iterations of Algorithm 4 is always in \mathcal{W} . Hence, let us now analyze the algorithm without the projection step.

Using the smoothness of $F(\cdot)$, we have

$$F(w_{t+1}) \leq F(w_t) + hr F(w_t); w_{t+1} - w_t + \frac{L_F}{2}kw_{t+1} - w_tk^2;$$

Using the iteration of Algorithm 4, we obtain

$$\begin{aligned} F(w_{t+1}) - F(w_t) &\leq hr F(w_t); r F(w_t) + \left(i + \frac{2L_F}{2}kr F(w_t) + k^2 \right. \\ &\quad \left. F(w_t) - kr F(w_t)k^2 - hr F(w_t); \left(i + \frac{2L_F}{2}kr F(w_t)k^2 \right. \right. \\ &\quad \left. \left. + \frac{2L_F}{2}k^2 + 2L_Fhr F(w_t); i \right. \right. \\ &\quad \left. \left. F(w_t) - \left(\frac{2L_F}{2} \right)kr F(w_t)k^2 + \left(i + 2L_F \right) \frac{1}{2}kr F(w_t)k^2 + \frac{1}{2}k^2 \right. \right. \\ &\quad \left. \left. + \frac{2L_F}{2}k^2 \right); \end{aligned}$$

where $\epsilon_0 > 0$ and the last inequality follows from Young's inequality. Substituting $\epsilon_0 = 1$, we obtain

$$\left(\epsilon_0 = 2 - 2L_F \right)kr F(w_t)k^2 \leq F(w_t) - F(w_{t+1}) + \left(\epsilon_0 = 2 + 2L_F \right)k^2;$$

We now use Lemma 4 to obtain

$$\begin{aligned} \left(\frac{\rho}{2} - 2L_F \right)kr F(w_t)k^2 &\leq F(w_t) - F(w_{t+1}) \\ &\quad + \left(\epsilon_0 = 2 + 2L_F \right) \left(1 + \frac{\rho}{1} \right) \frac{\rho}{1} + 2 \epsilon_0^2 kr F(w_t)k^2 + \epsilon_0 \left(\epsilon_0 \right); \end{aligned}$$

with high probability. Upon further simplification, we have

$$\frac{1}{2} \leq \frac{1}{2} (1 + \frac{\rho}{1+2}) \frac{\rho}{1+2} \frac{1}{1} + 2 \frac{1}{1} (1 + \frac{\rho}{1+2}) \frac{\rho}{1+2} \frac{1}{1} + 2 L_F \frac{1}{1} + 2 L_F k r F(w_t) k^2$$

$$F(w_t) - F(w_{t+1}) + (2 + \frac{\rho}{1+2}) L_F \frac{1}{1} :$$

We now substitute $\rho = \frac{c}{L_F}$, for a small enough constant c , so that we can ignore the contributions of the terms with quadratic dependence on ρ . We substitute $\frac{1}{1} = \frac{1}{1 + \frac{1}{1}}$ for a sufficiently small positive constant $\frac{1}{1}$. Provided $\frac{1}{1} > \frac{1}{1} + 4 \frac{1}{1} + 4 \frac{1}{1}$, where $\frac{1}{1} = 1 - \frac{(\frac{1}{1})^2}{1 + \frac{1}{1}}$, we have

$$\frac{1}{2} \leq \frac{1}{2} (1 + \frac{\rho}{1+2}) \frac{\rho}{1+2} \frac{1}{1} + 2 \frac{1}{1} (1 + \frac{\rho}{1+2}) \frac{\rho}{1+2} \frac{1}{1} + 2 L_F \frac{1}{1} + 2 L_F = \frac{c_1}{L_F}$$

where c_1 is a constant. With this choice, we obtain

$$\frac{1}{T+1} \sum_{t=0}^T k r F(w_t) k^2 \leq C_1 \frac{L_F (F(w_0) - F(w^*))}{T+1} + C_2$$

where the first term is obtained from a telescopic sum and $\frac{1}{1}$ is defined in equation (3.4). Finally, we obtain

$$\min_{t=0, \dots, T} k r F(w_t) k^2 \leq C_1 \frac{L_F (F(w_0) - F(w^*))}{T+1} + C_2$$

with probability greater than or equal to $1 - \frac{c_1 (1 + \frac{1}{1})^{md}}{(1 + n \frac{1}{1} D)^d} - \frac{c_2 d}{(1 + (1 + \frac{1}{1})^{mn \frac{1}{1} D})^d}$, proving Theorem 3.

3.10.2 Proof of Theorem 4

The proof of convergence for Theorem 4 follows the same steps as Theorem 3. Recall that the quantity of interest is

$$e = g(w_t) - r F(w_t)$$

for which we prove bound in the following lemma.

Lemma 5. For any $\frac{1}{1} > 0$, we have,

$$k e k^2 \leq ((1 + \frac{\rho}{1+2}) \frac{\rho}{1+2} \frac{1}{1} + 2 \frac{1}{1}) \frac{1}{1} j j r F(w_t) j j^2 + e(\frac{1}{1})$$

with probability greater than or equal to $1 - \frac{c_1 (1 + \frac{1}{1})^{md}}{(1 + n \frac{1}{1} D)^d} - \frac{c_2 d}{(1 + (1 + \frac{1}{1})^{mn \frac{1}{1} D})^d}$, where

$$e(\frac{1}{1}) = 2(1 + \frac{1}{1}) \frac{1}{1} \frac{(\frac{1}{1}) \frac{\rho}{1+2} \frac{1}{1} + 2 \frac{1}{1}}{1} + (\frac{1}{1})^2 \frac{1}{1} :$$

with $\frac{1}{1}$ and $\frac{1}{2}$ as defined in equation (3.2) and (3.3) respectively.

Taking the above lemma for granted, we proceed to prove Theorem 4. The proof of Lemma 5 is deferred to Section 3.10.6.

The proof parallels the proof of 3, except the fact that we use Lemma 5 to upper bound $k^e k^2$. Correspondingly, a little algebra shows that we require $\epsilon_0 > \epsilon_0 + 4 \epsilon_0^2 + 4 \epsilon_0^3$, where $\epsilon_0 = 1 - \frac{(1-\epsilon_0)^2}{(1+\epsilon_0)^2(1+\epsilon_0)}$, where ϵ_0 is a sufficiently small positive constant. With the above requirement, the proof follows the same steps as Theorem 3 and hence we omit the details here.

3.10.3 Proof of Lemma 4:

We require the following result to prove Lemma 4. In the following result, we show that for non-Byzantine worker machine i , the local gradient $\nabla F_i(w_t)$ is concentrated around the global gradient $\nabla F(w_t)$.

Lemma 6. *We have*

$$\max_{i \in \mathcal{M}} \|\nabla F_i(w_t) - \nabla F(w_t)\| \leq \epsilon_1$$

with probability exceeding $1 - \frac{2(1-\epsilon_0)^{md}}{(1+n\epsilon_0 D)^d}$, where ϵ_1 is defined in equation (3.2).

Furthermore, we have the following Lemma which implies that the average of local gradients $\nabla F_i(w_t)$ over non-Byzantine worker machines is close to its expectation $\nabla F(w_t)$.

Lemma 7. *We have*

$$\left\| \frac{1}{j\mathcal{M}_j} \sum_{i \in \mathcal{M}_j} \nabla F_i(w_t) - \nabla F(w_t) \right\| \leq \epsilon_2$$

with probability exceeding $1 - \frac{2(1-\epsilon_0)^{md}}{(1+n\epsilon_0 D)^d} - \frac{2d}{(1+(1-\epsilon_0)mn\epsilon_0 D)^d}$, where ϵ_2 is defined in equation (3.3).

Recall the definition of T_1 . Using triangle inequality, we obtain

$$\|k - k\| \leq \underbrace{\left\| \frac{1}{j\mathcal{U}_t} \sum_{i \in \mathcal{U}_t} Q(\nabla F_i(w_t)) - \frac{1}{j\mathcal{U}_t} \sum_{i \in \mathcal{U}_t} \nabla F_i(w_t) \right\|}_{T_1} + \underbrace{\left\| \frac{1}{j\mathcal{U}_t} \sum_{i \in \mathcal{U}_t} \nabla F_i(w_t) - \nabla F(w_t) \right\|}_{T_2}$$

We first control T_1 . Using the compression scheme (Definition 9), we obtain

$$\begin{aligned} T_1 &= \left\| \frac{1}{j\mathcal{U}_t} \sum_{i \in \mathcal{U}_t} Q(\nabla F_i(w_t)) - \frac{1}{j\mathcal{U}_t} \sum_{i \in \mathcal{U}_t} \nabla F_i(w_t) \right\| \leq \frac{\rho_1}{j\mathcal{U}_t} \sum_{i \in \mathcal{U}_t} \|\nabla F_i(w_t)\| \\ &\leq \frac{\rho_1}{j\mathcal{U}_t} \sum_{i \in \mathcal{M}} \|\nabla F_i(w_t)\| + \frac{\rho_1}{j\mathcal{U}_t} \sum_{i \in \mathcal{B} \setminus \mathcal{U}_t} \|\nabla F_i(w_t)\| \\ &\leq \frac{\rho_1}{j\mathcal{U}_t} \sum_{i \in \mathcal{M}} \|\nabla F_i(w_t)\| + \frac{\rho_1}{j\mathcal{U}_t} \sum_{i \in \mathcal{B} \setminus \mathcal{U}_t} \|\nabla F_i(w_t)\| \end{aligned}$$

Since $\mathcal{M} \setminus T_t \in \mathcal{I}$, we ensure that $\mathcal{M} \setminus T_t \in \mathcal{I}$. We have,

$$T_1 = \frac{\rho_1}{jU_t} \max_{i \in \mathcal{I} \setminus T_t} \|r F_i(w_t) - r F(w_t)\| + \frac{\rho_1}{jU_t} \max_{i \in \mathcal{I} \setminus T_t} \|r F_i(w_t) - r F(w_t)\| + \frac{\rho_1}{jU_t} \max_{i \in \mathcal{I} \setminus T_t} \|r F_i(w_t) - r F(w_t)\|$$

$$+ \frac{\rho_1}{jU_t} \max_{i \in \mathcal{I} \setminus T_t} \|r F_i(w_t) - r F(w_t)\| + \frac{\rho_1}{jU_t} \max_{i \in \mathcal{I} \setminus T_t} \|r F_i(w_t) - r F(w_t)\|$$

We now upper-bound T_3 . We have

$$T_3 = \frac{\rho_1}{jU_t} \max_{i \in \mathcal{I} \setminus T_t} \|r F_i(w_t) - r F(w_t)\| + \frac{\rho_1}{jU_t} \max_{i \in \mathcal{I} \setminus T_t} \|r F_i(w_t) - r F(w_t)\|$$

$$+ \frac{\rho_1}{jU_t} \max_{i \in \mathcal{I} \setminus T_t} \|r F_i(w_t) - r F(w_t)\| + \frac{\rho_1}{jU_t} \max_{i \in \mathcal{I} \setminus T_t} \|r F_i(w_t) - r F(w_t)\|$$

with probability exceeding $1 - \frac{2(1-\rho_1)^{md}}{(1+n\hat{L}D)^d}$, where we use Lemma 6. Similarly, for T_4 , we have

$$T_4 = \frac{\rho_1}{jU_t} \max_{i \in \mathcal{I} \setminus T_t} \|r F_i(w_t) - r F(w_t)\| + \frac{\rho_1}{jU_t} \max_{i \in \mathcal{I} \setminus T_t} \|r F_i(w_t) - r F(w_t)\|$$

We now control the terms in T_2 . We obtain the following:

$$T_2 = \frac{1}{jU_t} \max_{i \in \mathcal{I} \setminus T_t} \|r F_i(w_t) - r F(w_t)\| + \frac{1}{jU_t} \max_{i \in \mathcal{I} \setminus T_t} \|r F_i(w_t) - r F(w_t)\|$$

$$+ \frac{1}{jU_t} \max_{i \in \mathcal{I} \setminus T_t} \|r F_i(w_t) - r F(w_t)\| + \frac{1}{jU_t} \max_{i \in \mathcal{I} \setminus T_t} \|r F_i(w_t) - r F(w_t)\|$$

Using Lemma 7, we have

$$\frac{1}{jU_t} k \sum_{i \in 2M} (r F_i(w_t) - r F(w_t)) k \leq \frac{1}{1} \cdot 2;$$

with probability exceeding $1 - \frac{2(1-\epsilon)^{md}}{(1+n\hat{L}D)^d} - \frac{2d}{(1+(1-\epsilon)^{mn\hat{L}D})^d}$. Also, we obtain

$$\frac{1}{jU_t} k \sum_{i \in 2M \setminus T_t} (r F_i(w_t) - r F(w_t)) k \leq \frac{1}{1} \max_{i \in 2M} k r F_i(w_t) - r F(w_t) k \leq \frac{1}{1} \cdot 1;$$

with probability at least $1 - \frac{2(1-\epsilon)^{md}}{(1+n\hat{L}D)^d}$, where the last inequality is derived from Lemma 6. Finally, for the Byzantine term, we have

$$\begin{aligned} \frac{1}{jU_t} k \sum_{i \in 2B \setminus T_t} (r F_i(w_t) - r F(w_t)) k &\leq \frac{1}{1} \max_{i \in 2B \setminus T_t} k r F_i(w_t) k + \frac{1}{1} k r F(w_t) k \\ &\leq \frac{1}{1} \max_{i \in 2M} k r F_i(w_t) k + \frac{1}{1} k r F(w_t) k \\ &\leq \frac{1}{1} \max_{i \in 2M} k r F_i(w_t) - r F(w_t) k + \frac{2}{1} k r F(w_t) k \\ &\leq \frac{1}{1} \cdot 1 + \frac{2}{1} k r F(w_t) k; \end{aligned}$$

with high probability, where the last inequality follows from Lemma 6.

Combining all the terms of T_1 and T_2 , we obtain,

$$k \leq k \left(\frac{\rho}{1} + 2 k r F(w_t) k + \frac{\rho}{1} + \frac{1}{1} \cdot 1 + \frac{1}{1} \cdot 2 \right);$$

Now, using Young's inequality, for any $\epsilon > 0$, we obtain

$$k \leq k^2 \left(1 + \epsilon \right) \left(\frac{\rho}{1} + 2 \right)^2 k r F(w_t) k^2 + \epsilon^{-1} \left(\frac{\rho}{1} + \frac{1}{1} \right)^2;$$

where

$$\epsilon^{-1} \left(\frac{\rho}{1} + \frac{1}{1} \right)^2 = 2 \left(1 + \frac{1}{1} \right) \left(\frac{\rho}{1} + \frac{1}{1} \right)^2 \leq \frac{2}{1} + \frac{1}{1} \cdot 2^{\#} \leq \frac{2}{1};$$

3.10.4 Proof of Lemma 6:

For a fixed $i \in 2M$, we first analyze the quantity $k r F_i(w_t) - r F(w_t) k$. Notice that i is non-Byzantine. Recall that machine i has n independent data points. We use the sub-exponential concentration to control this term. Let us rewrite the concentration inequality.

Univariate sub-exponential concentration: Suppose Y is univariate random variable with $EY = \mu$ and y_1, \dots, y_n are i.i.d draws of Y . Also, Y is v sub-exponential. From sub-exponential concentration (Hoeffding's inequality), we obtain

$$\mathbb{P} \left[\left| \frac{1}{n} \sum_{i=1}^n y_i - \mu \right| > t \right] \leq 2 \exp \left\{ -n \min \left(\frac{t}{v}; \frac{t^2}{v^2} \right) g \right\}$$

We directly use this to the k -th partial derivative of F_i . Let $\partial_k f(w_t; z^{ij})$ be the partial derivative of the loss function with respect to k -th coordinate on i -th machine with j -th data point. From Assumption 10, we obtain

$$\mathbb{P} \left[\left| \frac{1}{n} \sum_{j=1}^n \partial_k f(w_t; z^{ij}) - \partial_k F(w_t) \right| > t \right] \leq 2 \exp \left\{ -n \min \left(\frac{t}{v}; \frac{t^2}{v^2} \right) g \right\}$$

Since $r F_i(w_t) = \frac{1}{n} \sum_{j=1}^n r f(w_t; z^{ij})$, denoting $r F_i^{(k)}(w_t)$ as the k -th coordinate of $r F_i(w_t)$, we have

$$\left| r F_i^{(k)}(w_t) - \partial_k F(w_t) \right| \leq t$$

with probability at least $1 - 2 \exp \left\{ -n \min \left(\frac{t}{v}; \frac{t^2}{v^2} \right) g \right\}$.

This result holds for a particular w_t . To extend this for all $w \in \mathcal{W}$, we exploit the covering net argument and the Lipschitz continuity of the partial derivative of the loss function (Assumption 9). Let $\{w_1, \dots, w_N\}$ be a $\frac{D}{2}$ covering of \mathcal{W} . Since \mathcal{W} has diameter D , from Vershynin, we obtain $N \leq (1 + \frac{D}{\frac{D}{2}})^d$. Hence with probability at least

$$1 - 2Nd \exp \left\{ -n \min \left(\frac{t}{v}; \frac{t^2}{v^2} \right) g \right\}$$

we have

$$\left| r F_i^{(k)}(w) - \partial_k F(w) \right| \leq t$$

for all $w \in \{w_1, \dots, w_N\}$ and $k \in [d]$. This implies

$$\| r F_i(w_t) - r F(w_t) \|_k \leq t + \frac{D}{2}$$

with probability greater than or equal to $1 - 2Nd \exp \left\{ -n \min \left(\frac{t}{v}; \frac{t^2}{v^2} \right) g \right\}$.

We now reason about $w \in \mathcal{W}$ via Lipschitzness (Assumption 9). From the definition of $\frac{D}{2}$ cover, for any $w \in \mathcal{W}$, there exists w' , an element of the cover such that $\|w - w'\|_k \leq \frac{D}{2}$. Hence, we obtain

$$\left| r F_i^{(k)}(w) - \partial_k F(w) \right| \leq t + 2L_k \frac{D}{2}$$

for all $w \in W$ and consequently

$$\| \mathbb{E} F_i(w_t) - \mathbb{E} F(w_t) \| \leq \sqrt{\frac{\rho}{d}} \sqrt{t + 2 \hat{L}}$$

with probability at least $1 - 2Nd \exp(-n \min(\frac{t}{v}, \frac{t^2}{v^2})g)$, where $\hat{L} = \sqrt{\frac{1}{d} \sum_{k=1}^d L_k^2}$.

Choosing $\rho = \frac{1}{2n\hat{L}}$ and

$$t = v \max \left\{ \frac{d}{n} \log(1 + 2n\hat{L}d); \sqrt{\frac{d}{n} \log(1 + 2n\hat{L}d)g} \right\};$$

we obtain

$$\| \mathbb{E} F_i(w_t) - \mathbb{E} F(w_t) \| \leq \sqrt{\frac{\rho}{d}} \max \left\{ \frac{d}{n} \log(1 + 2n\hat{L}d); \sqrt{\frac{d}{n} \log(1 + 2n\hat{L}d)g} \right\} + \frac{1}{n} = \epsilon; \tag{3.10}$$

with probability greater than $1 - \frac{d}{(1+n\hat{L}D)^d}$. Taking union bound on all non-Byzantine machines yields the theorem.

3.10.5 Proof of Lemma 7

We need to upper bound the following quantity:

$$\frac{1}{jM} \sum_{i \in M} \| \mathbb{E} F_i(w_t) - \mathbb{E} F(w_t) \|$$

We now use similar argument (sub-exponential concentration) like Lemma 6. The only difference is that in this case, we also consider *averaging* over worker nodes. We obtain the following:

$$\frac{1}{jM} \sum_{i \in M} \| \mathbb{E} F_i(w_t) - \mathbb{E} F(w_t) \| \leq \epsilon_2$$

where

$$\epsilon_2 = \sqrt{\frac{\rho}{d}} \max \left\{ \frac{d}{(1 - \epsilon)mn} \log(1 + 2(1 - \epsilon)mn\hat{L}d); \sqrt{\frac{d}{(1 - \epsilon)mn} \log(1 + 2(1 - \epsilon)mn\hat{L}d)g} \right\};$$

with probability $1 - \frac{2d}{(1+(1 - \epsilon)mn\hat{L}D)^d}$.

3.10.6 Proof of Lemma 5

Here we prove an upper bound on the norm of

$$e = g(w_t) - rF(w_t)$$

where $g(w_t) = \frac{1}{jU_t} \sum_{i \in U_t} Q(rF_i(w_t))$.

We have

$$\begin{aligned} \|e\|^2 &= \left\| \frac{1}{jU_t} \sum_{i \in U_t} Q(rF_i(w_t)) - rF(w_t) \right\|^2 \\ &= \frac{1}{jU_t} \sum_{i \in U_t} \| [Q(rF_i(w_t)) - rF(w_t)] \|^2 + \sum_{i \in (M \setminus U_t)} \| [Q(rF_i(w_t)) - rF(w_t)] \|^2 \\ &\quad + \sum_{i \in (B \setminus U_t)} \| [Q(rF_i(w_t)) - rF(w_t)] \|^2 \\ &= \frac{1}{jU_t} \sum_{i \in U_t} \| [Q(rF_i(w_t)) - rF(w_t)] \|^2 + \sum_{i \in (M \setminus U_t)} \| [Q(rF_i(w_t)) - rF(w_t)] \|^2 \\ &\quad + \sum_{i \in (B \setminus U_t)} \| [Q(rF_i(w_t)) - rF(w_t)] \|^2 \\ &= \frac{1}{jU_t} \sum_{i \in U_t} \| [Q(rF_i(w_t)) - rF(w_t)] \|^2 + \sum_{i \in (M \setminus U_t)} \| [Q(rF_i(w_t)) - rF(w_t)] \|^2 \\ &\quad + \sum_{i \in (B \setminus U_t)} \| [Q(rF_i(w_t)) - rF(w_t)] \|^2 \end{aligned}$$

Now we bound each term separately. For the first term, we have

$$\begin{aligned} \frac{1}{jU_t} \sum_{i \in U_t} \| [Q(rF_i(w_t)) - rF(w_t)] \|^2 &= \frac{1}{jU_t} \sum_{i \in U_t} \| Q(rF_i(w_t)) - rF_i(w_t) + rF_i(w_t) - rF(w_t) \|^2 \\ &= \frac{1}{jU_t} \sum_{i \in U_t} \| Q(rF_i(w_t)) - rF_i(w_t) \|^2 + \frac{1}{jU_t} \sum_{i \in U_t} \| rF_i(w_t) - rF(w_t) \|^2 \\ &\leq \frac{1}{jU_t} \sum_{i \in U_t} \| Q(rF_i(w_t)) - rF_i(w_t) \|^2 + \frac{1}{j} \sum_{i \in U_t} \| rF_i(w_t) - rF(w_t) \|^2 \\ &\leq \frac{1}{jU_t} \sum_{i \in U_t} \rho \| rF_i(w_t) \|^2 + \frac{1}{j} \sum_{i \in U_t} \| rF_i(w_t) - rF(w_t) \|^2 + \frac{1}{j} \sum_{i \in U_t} \| rF_i(w_t) \|^2 \\ &\leq \frac{\rho}{j} \sum_{i \in U_t} \| rF_i(w_t) \|^2 + \frac{\rho}{j} \sum_{i \in U_t} \| rF_i(w_t) - rF(w_t) \|^2 + \frac{1}{j} \sum_{i \in U_t} \| rF_i(w_t) \|^2 \end{aligned}$$

where we use the definition of a ρ -approximate compressor, Lemma 6 and Lemma 7. Similarly, we can bound T_2 as

$$\begin{aligned} T_2 & \leq \sum_{i \in 2(M \setminus T_t)} \langle \mathbb{Q}(r F_i(w_t)) - r F(w_t), \mathbb{J} \rangle \\ & \leq m \max_{i \in 2M} \langle \mathbb{Q}(r F_i(w_t)) - r F(w_t), \mathbb{J} \rangle \\ & \leq m \max_{i \in 2M} \left(\frac{\rho}{1} \langle \mathbb{J} r F_i(w_t), \mathbb{J} \rangle + \langle \mathbb{J} r F_i(w_t) - r F(w_t), \mathbb{J} \rangle \right) \\ & \leq m \max_{i \in 2M} \left(\frac{\rho}{1} \langle \mathbb{J} r F(w_t), \mathbb{J} \rangle + (1 + \frac{\rho}{1}) \langle \mathbb{J} r F_i(w_t) - r F(w_t), \mathbb{J} \rangle \right) \end{aligned}$$

where we use the definition of ρ -approximate compressor. Hence invoking Lemma 6, we obtain

$$\frac{1}{\mathbb{J} U_t} T_2 \leq \frac{\rho}{1} \langle \mathbb{J} r F(w_t), \mathbb{J} \rangle + \frac{(1 + \frac{\rho}{1})}{1} \epsilon$$

Also, owing to the trimming with $\epsilon > \epsilon$, we have at least one good machine in the set T_t for all t . Now each term in the set $B \setminus U_t$, we have

$$\begin{aligned} T_3 & = \sum_{i \in 2(B \setminus U_t)} \langle \mathbb{Q}(r F_i(w_t)) - r F(w_t), \mathbb{J} \rangle \\ & \leq m (\max_{i \in 2M} \langle \mathbb{Q}(r F_i(w_t)) - r F(w_t), \mathbb{J} \rangle) \\ & \leq m (\max_{i \in 2M} \left(\frac{\rho}{1} \langle \mathbb{J} r F_i(w_t), \mathbb{J} \rangle + \langle \mathbb{J} r F_i(w_t) - r F(w_t), \mathbb{J} \rangle \right)) \\ & \leq m \left((1 + \frac{\rho}{1}) \epsilon + (2 + \frac{\rho}{1}) \langle \mathbb{J} r F(w_t), \mathbb{J} \rangle \right) \\ \frac{1}{\mathbb{J} U_t} T_3 & \leq \frac{(2 + \frac{\rho}{1})}{1} \langle \mathbb{J} r F(w_t), \mathbb{J} \rangle + \frac{(1 + \frac{\rho}{1})}{1} \epsilon \end{aligned}$$

where we use Lemma 6. Putting $T_1; T_2; T_3$ we get

$$\begin{aligned} \langle \mathbb{J} e, \mathbb{J} \rangle & \leq \frac{\rho}{1} (1 + \frac{\rho}{1}) \epsilon + \frac{\rho}{1} \epsilon + \frac{(2 + \frac{\rho}{1})}{1} \langle \mathbb{J} r F(w_t), \mathbb{J} \rangle \\ & + \frac{\rho}{1} (1 + \frac{\rho}{1}) \epsilon + \frac{(1 + \frac{\rho}{1})}{1} \epsilon + \frac{(1 + \frac{\rho}{1})}{1} \epsilon + \frac{1}{1} \epsilon^2 \\ & = \frac{(1 + \frac{\rho}{1}) \rho}{1} \epsilon + 2 \langle \mathbb{J} r F(w_t), \mathbb{J} \rangle + \frac{(1 + \frac{\rho}{1}) \rho}{1} \epsilon + \frac{1}{1} \epsilon + \frac{1}{1} \epsilon^2 \\ \langle \mathbb{J} e, \mathbb{J} \rangle & \leq (1 + \frac{\rho}{1}) \frac{(1 + \frac{\rho}{1}) \rho}{1} \epsilon + 2 \langle \mathbb{J} r F(w_t), \mathbb{J} \rangle + \epsilon \end{aligned}$$

where $\epsilon(\rho) = 2(1 + \frac{\rho}{1}) \frac{(1 + \frac{\rho}{1}) \rho}{1} \epsilon + \frac{1}{1} \epsilon^2 + (1 + \frac{\rho}{1})^2 \frac{1}{2} \epsilon$. Hence, the lemma follows.

3.11 Proof of Theorem 5

We first define an auxiliary sequence defined as:

$$w_t = w_t \prod_{i \in \mathcal{M}} \frac{1}{j_{\mathcal{M}j}} e_i(t)$$

Hence, we obtain

$$w_{t+1} = w_{t+1} \prod_{i \in \mathcal{M}} \frac{1}{j_{\mathcal{M}j}} e_i(t+1):$$

For notational simplicity, let us drop the subscript t from U_t and T_t and denote them as U and T .

Since (we will ensure that the iterates remain in the parameter space and hence we can ignore the projection step),

$$w_{t+1} = w_t \prod_{i \in U} \frac{1}{j_{Uj}} p_i(w_t);$$

we get

$$\begin{aligned} w_{t+1} &= w_t \prod_{i \in U} \frac{1}{j_{Uj}} C(p_i(w_t)) \prod_{i \in \mathcal{M}} \frac{1}{j_{\mathcal{M}j}} e_i(t+1) \\ &= w_t \prod_{i \in U} \frac{1}{j_{Uj}} \left(\prod_{i \in \mathcal{M}} \frac{1}{j_{\mathcal{M}j}} C(p_i(w_t)) + \prod_{i \in \mathcal{M} \setminus U} C(p_i(w_t)) \right) \prod_{i \in \mathcal{M} \setminus T} C(p_i(w_t)) \prod_{i \in \mathcal{M}} \frac{1}{j_{\mathcal{M}j}} e_i(t+1) \\ &= w_t \prod_{i \in U} \frac{1}{j_{Uj}} \prod_{i \in \mathcal{M}} \frac{1}{j_{\mathcal{M}j}} C(p_i(w_t)) \prod_{i \in \mathcal{M}} \frac{1}{j_{\mathcal{M}j}} e_i(t+1) \\ &\quad \prod_{i \in \mathcal{M} \setminus U} \frac{1}{j_{Uj}} C(p_i(w_t)) + \prod_{i \in \mathcal{M} \setminus T} \frac{1}{j_{Uj}} C(p_i(w_t)) \end{aligned}$$

Since $C(p_i(w_t)) + e_i(t+1) = p_i(w_t)$ for all $i \in \mathcal{M}$, we obtain

$$\begin{aligned} \prod_{i \in U} \frac{1}{j_{Uj}} \prod_{i \in \mathcal{M}} \frac{1}{j_{\mathcal{M}j}} C(p_i(w_t)) + \prod_{i \in \mathcal{M}} \frac{1}{j_{\mathcal{M}j}} e_i(t+1) &= \prod_{i \in \mathcal{M}} \frac{1}{j_{\mathcal{M}j}} p_i(w_t) \\ &\quad + \prod_{i \in \mathcal{M}} \frac{1}{j_{\mathcal{M}j}} C(p_i(w_t)) \end{aligned}$$

Let us denote $T_1 = \prod_{i \in \mathcal{M} \setminus U} \frac{1}{j_{Uj}} C(p_i(w_t))$, $T_2 = \prod_{i \in \mathcal{M} \setminus T} \frac{1}{j_{Uj}} C(p_i(w_t))$

and $T_3 = \frac{1}{1 - \frac{1}{jMj}} \sum_{i \in \mathcal{M}} C(p_i(w_t))$. With this, we obtain

$$\begin{aligned} w_{t+1} &= w_t + \frac{1}{jMj} \sum_{i \in \mathcal{M}} p_i(w_t) - T_1 + T_2 - T_3 \\ &= w_t + \frac{1}{jMj} \sum_{i \in \mathcal{M}} e_i(t) - \frac{1}{jMj} \sum_{i \in \mathcal{M}} p_i(w_t) + \bar{F} \\ &= w_t + \frac{1}{jMj} \sum_{i \in \mathcal{M}} r F_i(w_t) + \bar{F} \end{aligned}$$

where $\bar{F} = T_1 - T_2 + T_3$. Observe that the auxiliary sequence looks similar to a distributed gradient step with a presence of \bar{F} . For the convergence analysis, we will use this relation along with an upper bound on $k\bar{F}k$.

Using this auxiliary sequence, we first ensure that the iterates of our algorithm remains close to one another. To that end, we have

$$\begin{aligned} w_{t+1} - w_t &= w_{t+1} - w_t + \frac{1}{jMj} \sum_{i \in \mathcal{M}} e_i(t+1) - \frac{1}{jMj} \sum_{i \in \mathcal{M}} e_i(t) \\ &= \frac{1}{jMj} \sum_{i \in \mathcal{M}} r F_i(w_t) + \bar{F} + \frac{1}{jMj} \sum_{i \in \mathcal{M}} e_i(t+1) - \frac{1}{jMj} \sum_{i \in \mathcal{M}} e_i(t) \end{aligned}$$

Hence, we obtain

$$\begin{aligned} \|w_{t+1} - w_t\| &\leq k \frac{1}{jMj} \sum_{i \in \mathcal{M}} r F_i(w_t) \| + k\bar{F} \| + k \frac{1}{jMj} \sum_{i \in \mathcal{M}} e_i(t+1) \| + k \frac{1}{jMj} \sum_{i \in \mathcal{M}} e_i(t) \| \\ &\leq k \frac{1}{jMj} \sum_{i \in \mathcal{M}} \|r F_i(w_t) - r F(w_t)\| + \|kr F(w_t)\| + k\bar{F} \| + k \frac{1}{jMj} \sum_{i \in \mathcal{M}} \|e_i(t+1)\| \\ &\quad + k \frac{1}{jMj} \sum_{i \in \mathcal{M}} \|e_i(t)\| \\ &\leq 2 + \|kr F(w_t)\| + k\bar{F} \| + k \frac{1}{jMj} \sum_{i \in \mathcal{M}} \|e_i(t+1)\| + k \frac{1}{jMj} \sum_{i \in \mathcal{M}} \|e_i(t)\| \end{aligned}$$

Now, using Lemma 8 and Lemma 9 in conjunction with Assumption 11 ensures the iterates of Algorithm 3 stays in the parameter space \mathcal{W} .

We assume that the global loss function $F(\cdot)$ is L_F smooth. We get

$$F(w_{t+1}) \leq F(w_t) + \langle \nabla F(w_t), w_{t+1} - w_t \rangle + \frac{L_F}{2} \|w_{t+1} - w_t\|^2$$

Now, we use the above recursive equation

$$w_{t+1} = w_t + \frac{1}{jMj} \sum_{i \in \mathcal{M}} r F_i(w_t) + \bar{F}$$

Substituting, we obtain

$$\begin{aligned}
 F(w_{t+1}) - F(w_t) &= \underbrace{h r F(w_t); r F(w_t) i}_{\text{Term I}} + \underbrace{h r F(w_t); r F(w_t) i}_{\text{Term II}} \\
 &+ \frac{L_F}{2} k \frac{1}{jMj} \sum_{i=2M}^{\times} r F_i(w_t) + \mathcal{F} k^2 \\
 &+ \underbrace{h r F(w_t); r F(w_t) i}_{\text{Term III}} \\
 &+ L_F^2 k \frac{1}{jMj} \sum_{i=2M}^{\times} r F_i(w_t) k^2 + L_F k \mathcal{F} k^2
 \end{aligned} \tag{3.11}$$

In the subsequent calculation, we use the following definition of smoothness:

$$k r F(y_1) - r F(y_2) k \leq L_F k y_1 - y_2 k$$

for all y_1 and $y_2 \in \mathbb{R}^d$.

Rewriting the right hand side (R.H.S) of equation (3.11), we obtain

$$\begin{aligned}
 R:H:S &= \underbrace{F(w_t) - h r F(w_t); r F(w_t) i}_{\text{Term I}} + \underbrace{h r F(w_t); r F(w_t) i}_{\text{Term II}} \\
 &+ \underbrace{h r F(w_t); \mathcal{F} i + h r F(w_t) - r F(w_t); \mathcal{F} i}_{\text{Term III}} \\
 &+ \underbrace{2 L_F^2 k \frac{1}{jMj} \sum_{i=2M}^{\times} r F_i(w_t) - r F(w_t) k^2 + 2 L_F^2 k r F(w_t) k^2 + L_F k \mathcal{F} k^2}_{\text{Term IV}}
 \end{aligned}$$

We now control the 4 terms separately. We start with Term-I.

Control of Term-I: We obtain

$$\begin{aligned}
 \text{Term-I} &= F(w_t) - h r F(w_t); r F(w_t) i - h r F(w_t) - r F(w_t); r F(w_t) i \\
 &= F(w_t) - k r F(w_t) k^2 + 25 k r F(w_t) - r F(w_t) k^2 + \frac{1}{100} k r F(w_t) k^2;
 \end{aligned}$$

where we use Young's inequality ($ha; bi \leq \frac{1}{2} a^2 + \frac{1}{2} b^2$ with $a = 50b$) in the last inequality. Using the smoothness of $F(\cdot)$, we obtain

$$\text{Term-I} = F(w_t) - k r F(w_t) k^2 + \frac{1}{100} k r F(w_t) k^2 + 25 L_F^2 k \frac{1}{jMj} \sum_{i=2M}^{\times} e_i(t) k^2; \tag{3.12}$$

Control of Term-II: Similarly, for Term-II, we have

$$\begin{aligned} \text{Term-II} = & \frac{1}{jMj} \sum_{i \in \mathcal{M}} \left(\frac{1}{2} \left(\frac{1}{2} + \frac{1}{200} k r F(w_t) k^2 \right) \right. \\ & \left. + \frac{1}{100} k r F(w_t) k^2 + \frac{L_F^2}{100} k \frac{1}{jMj} \sum_{i \in \mathcal{M}} e_i(t) k^2 \right) \end{aligned} \quad (3.13)$$

Control of Term-III: We obtain

$$\begin{aligned} \text{Term-III} = & \frac{1}{2} k r F(w_t) k^2 + \frac{1}{2} k \bar{\Gamma} k^2 + \frac{L_F^2}{2} k \frac{1}{jMj} \sum_{i \in \mathcal{M}} e_i(t) k^2 + \frac{1}{2} k \bar{\Gamma} k^2 \end{aligned} \quad (3.14)$$

Control of Term-IV:

$$\begin{aligned} \text{Term-IV} = & 2L_F^2 k \frac{1}{jMj} \sum_{i \in \mathcal{M}} \left(\frac{1}{2} \left(\frac{1}{2} + \frac{1}{200} k r F(w_t) k^2 \right) \right. \\ & \left. + \frac{1}{100} k r F(w_t) k^2 + L_F k \bar{\Gamma} k^2 \right) \end{aligned} \quad (3.15)$$

Combining all 4 terms, we obtain

$$\begin{aligned} & F(w_{t+1}) - F(w_t) - \frac{1}{2} \left(\frac{1}{2} + \frac{1}{200} k r F(w_t) k^2 \right) \\ & + \frac{1}{25} L_F^2 + \frac{L_F^2}{100} + \frac{L_F^2}{2} k \frac{1}{jMj} \sum_{i \in \mathcal{M}} e_i(t) k^2 \\ & + 50 \left(\frac{1}{2} + \frac{1}{200} k r F(w_t) k^2 \right) + 2L_F^2 \left(\frac{1}{2} + \frac{1}{200} k r F(w_t) k^2 \right) + \frac{1}{2} + \frac{1}{2} + L_F k \bar{\Gamma} k^2 \end{aligned} \quad (3.16)$$

We now control the error sequence and $k \bar{\Gamma} k^2$. These will be separate lemmas, but here we write it as a whole.

Control of error sequence:

Lemma 8. For all $i \in \mathcal{M}$, we have

$$k e_i(t) k^2 \leq \frac{3(1 - \epsilon)}{2} \epsilon^2$$

for all $t \geq 0$.

Proof. For machine $i \in \mathcal{M}$, we have

$$k e_i(t+1) k^2 = k C(p_i(w_t)) - p_i(w_t) k^2 = (1 - \epsilon) k p_i(w_t) k^2 = (1 - \epsilon) k \left(r F_i(w_t) + e_i(t) \right) k^2$$

Using technique similar to the proof of [20, Lemma 3] and using $\|r F_i(w_t)\|_2^2$, we obtain

$$\|e_i(t+1)\|_2^2 \leq \frac{2(1-\alpha)(1+\alpha)}{1-\alpha^2} \sum_{j \in \mathcal{M}} \|e_j(t)\|_2^2$$

where $\alpha > 0$. Substituting $\sum_{j \in \mathcal{M}} \|e_j(t)\|_2^2 = 2$ implies

$$\|e_i(t+1)\|_2^2 \leq \frac{3(1-\alpha)}{1-\alpha^2} \quad (3.17)$$

for all $i \in \mathcal{M}$. This also implies

$$\max_{i \in \mathcal{M}} \|e_i(t+1)\|_2^2 \leq \frac{3(1-\alpha)}{1-\alpha^2}.$$

□

Control of $\|k\bar{F}\|_2^2$:

Lemma 9. *We obtain*

$$\|k\bar{F}\|_2^2 \leq \frac{9(1+\frac{\rho}{1-\alpha})^2}{(1-\alpha)^2} \sum_{j \in \mathcal{M}} \|e_j(t)\|_2^2 + \left(\frac{\rho}{1-\alpha}\right)^2 \sum_{j \in \mathcal{M}} \|r F_j(w_t)\|_2^2 + \frac{3(1-\alpha)}{1-\alpha^2} \sum_{j \in \mathcal{M}} \|e_j(t)\|_2^2$$

with probability exceeding $1 - \frac{2(1-\alpha)md}{(1+nLD)^d}$.

Proof. We have

$$\|k\bar{F}\|_2^2 = \|kT_1\|_2^2 + \|T_2\|_2^2 + \|T_3\|_2^2 \leq \|kT_1\|_2^2 + \|kT_2\|_2^2 + \|kT_3\|_2^2$$

We control these 3 terms separately. We obtain

$$\|kT_1\|_2^2 = \sum_{j \in \mathcal{U}} \frac{1}{j} \times \sum_{i \in \mathcal{B} \cup \mathcal{U}} C(p_i(w_t)) \|k\|_2^2 \leq \frac{1}{(1-\alpha)m} \sum_{i \in \mathcal{B} \cup \mathcal{U}} \|kC(p_i(w_t))\|_2^2$$

Since the worker machines are sorted according to $\|kC(p_i(w_t))\|_2^2$ (the central machine only gets to

see $C(p_i(w_t))$, and so the most natural metric to sort is $kC(p_i(w_t))k$, we obtain

$$\begin{aligned}
 kT_1k & \frac{m}{(1 + \frac{\rho}{1})^m} \max_{i \in \mathcal{M}} kC(p_i(w_t))k \\
 & \frac{m}{(1 + \frac{\rho}{1})^m} \max_{i \in \mathcal{M}} kp_i(w_t)k \\
 & \frac{m}{(1 + \frac{\rho}{1})^m} \max_{i \in \mathcal{M}} k r F_i(w_t) + e_i(t)k \\
 & \frac{m}{(1 + \frac{\rho}{1})^m} \max_{i \in \mathcal{M}} kr F_i(w_t) - r F(w_t)k \\
 & + (1 + \frac{\rho}{1}) \frac{1}{(1 + \frac{\rho}{1})} kr F(w_t)k + (1 + \frac{\rho}{1}) \frac{1}{(1 + \frac{\rho}{1})} \max_{i \in \mathcal{M}} ke_i(t)k \\
 & \frac{1}{(1 + \frac{\rho}{1})} + (1 + \frac{\rho}{1}) \frac{1}{(1 + \frac{\rho}{1})} kr F(w_t)k \\
 & + (1 + \frac{\rho}{1}) \frac{3(1 + \frac{\rho}{1})}{(1 + \frac{\rho}{1})} :
 \end{aligned}$$

Hence,

$$kT_1k^2 \leq 3 \frac{(1 + \frac{\rho}{1})^2}{(1 + \frac{\rho}{1})^2} \frac{1}{(1 + \frac{\rho}{1})} + kr F(w_t)k^2 + \frac{3(1 + \frac{\rho}{1})}{(1 + \frac{\rho}{1})} :$$

Similarly, we obtain,

$$kT_2k^2 \leq 3 \frac{(1 + \frac{\rho}{1})^2}{(1 + \frac{\rho}{1})^2} \frac{1}{(1 + \frac{\rho}{1})} + kr F(w_t)k^2 + \frac{3(1 + \frac{\rho}{1})}{(1 + \frac{\rho}{1})} :$$

For T_3 , we have

$$\begin{aligned}
 kT_3k & = \frac{1}{(1 + \frac{\rho}{1})} k \frac{1}{j \in \mathcal{M}} \times_{i \in \mathcal{M}} C(p_i(w_t))k \\
 & \frac{1}{(1 + \frac{\rho}{1})} k \frac{1}{j \in \mathcal{M}} \times_{i \in \mathcal{M}} (1 + \frac{\rho}{1}) kp_i(w_t)k \\
 & \frac{1}{(1 + \frac{\rho}{1})} \max_{i \in \mathcal{M}} kp_i(w_t)k
 \end{aligned}$$

Using the previous calculation, we obtain

$$\begin{aligned}
 kT_3k & \leq (1 + \frac{\rho}{1}) \frac{1}{(1 + \frac{\rho}{1})} + (1 + \frac{\rho}{1}) \frac{1}{(1 + \frac{\rho}{1})} kr F(w_t)k \\
 & + (1 + \frac{\rho}{1}) \frac{3(1 + \frac{\rho}{1})}{(1 + \frac{\rho}{1})} ;
 \end{aligned}$$

and as a result,

$$kT_3k^2 \leq 3 \frac{(1 + \frac{\rho}{1})^2}{(1 + \frac{\rho}{1})^2} \frac{1}{(1 + \frac{\rho}{1})} + kr F(w_t)k^2 + \frac{3(1 + \frac{\rho}{1})}{(1 + \frac{\rho}{1})} :$$

Combining the above 3 terms, we obtain

$$k\mathcal{F}k^2 \leq \frac{3kT_1k^2 + 3kT_2k^2 + 3kT_3k^2}{2(1 + \frac{\rho}{1})^2} + \frac{1}{2} + \left(\frac{\rho}{1}\right)^2 \frac{1}{2} + kr F(w_t)k^2 + \frac{3(1 + \frac{\rho}{1})^2}{2} ;$$

□

Back to the convergence of $F(\cdot)$: We use the above bound on $k\mathcal{F}k^2$ and Lemma 8 to conclude the proof of the main convergence result. Recall equation (3.16):

$$\begin{aligned} F(w_{t+1}) - F(w_t) &\leq \frac{1}{2} \frac{1}{50} 2L_F^2 k\mathcal{F}k^2 + \frac{1}{25} L_F^2 + \frac{L_F^2}{100} + \frac{L_F^2}{2} k \frac{1}{jMj_{i2M}} \sum_{i=1}^M e_i(t)k^2 \\ &\quad + 50 \frac{1}{2} + 2L_F^2 \frac{1}{2} + \frac{1}{2} + \frac{1}{2} + L_F k\mathcal{F}k^2 \end{aligned} \tag{3.18}$$

First, let us compute the term associated with the error sequence. Note that (from Cauchy-Schwartz inequality)

$$k \frac{1}{jMj_{i2M}} \sum_{i=1}^M e_i(t)k^2 \leq \frac{1}{jMj_{i2M}} \sum_{i=1}^M ke_i(t)k^2;$$

and from equation (3.17), we obtain

$$k \frac{1}{jMj_{i2M}} \sum_{i=1}^M e_i(t)k^2 \leq \frac{3(1 + \frac{\rho}{1})^2}{2} ;$$

and so the error term is upper bounded by

$$\frac{2L_F^2}{2} + \frac{3L_F^2}{100} + 25 \frac{3L_F^2}{2} \frac{3(1 + \frac{\rho}{1})^2}{2} ;$$

We now substitute the expression for $k\mathcal{F}k^2$. We obtain

$$\frac{1}{2} + \frac{1}{2} + L_F k\mathcal{F}k^2 = \frac{1}{2} k\mathcal{F}k^2 + \frac{1}{2} + L_F k\mathcal{F}k^2 ;$$

The first term in the above equation is

$$\begin{aligned} \frac{1}{2} k\mathcal{F}k^2 &\leq \frac{9(1 + \frac{\rho}{1})^2}{2(1 + \frac{\rho}{1})^2} + \frac{1}{2} + \left(\frac{\rho}{1}\right)^2 \frac{1}{2} + kr F(w_t)k^2 + \frac{3(1 + \frac{\rho}{1})^2}{2} \\ &\quad + \frac{9(1 + \frac{\rho}{1})^2}{2(1 + \frac{\rho}{1})^2} + \frac{1}{2} + \left(\frac{\rho}{1}\right)^2 kr F(w_t)k^2 \\ &\quad + \frac{9(1 + \frac{\rho}{1})^2}{2(1 + \frac{\rho}{1})^2} + \frac{1}{2} + \left(\frac{\rho}{1}\right)^2 \frac{1}{2} + \frac{3(1 + \frac{\rho}{1})^2}{2} ; \end{aligned}$$

and the second term is

$$\begin{aligned} \frac{1}{2} + L_F k^2 & \frac{1}{2} + L_F \frac{9(1 + \frac{\rho}{1})^2}{(1 - \frac{\rho}{1})^2} \epsilon^2 + \epsilon^2 + (\epsilon)^2 \\ & \frac{1}{2} + kr F(w_t)k^2 + \frac{3(1 - \frac{\rho}{1})}{2} \epsilon^2 \\ & \frac{1}{2} + L_F \frac{9(1 + \frac{\rho}{1})^2}{(1 - \frac{\rho}{1})^2} \epsilon^2 + \epsilon^2 + (\epsilon)^2 + kr F(w_t)k^2 \\ & + \frac{1}{2} + L_F \frac{9(1 + \frac{\rho}{1})^2}{(1 - \frac{\rho}{1})^2} \epsilon^2 + \epsilon^2 + (\epsilon)^2 + \frac{1}{2} + \frac{3(1 - \frac{\rho}{1})}{2} \epsilon^2 \end{aligned}$$

Collecting all the above terms, the coefficient of $kr F(w_t)k^2$ is given by

$$\begin{aligned} \frac{1}{2} - \frac{1}{50} - 2L_F \frac{9(1 + \frac{\rho}{1})^2}{2(1 - \frac{\rho}{1})^2} \epsilon^2 + \epsilon^2 + (\epsilon)^2 \\ (\frac{1}{2} + L_F) \frac{9(1 + \frac{\rho}{1})^2}{(1 - \frac{\rho}{1})^2} \epsilon^2 + \epsilon^2 + (\epsilon)^2 : \end{aligned}$$

Provided we select a sufficiently small ϵ , a little algebra shows that if

$$\frac{9(1 + \frac{\rho}{1})^2}{2(1 - \frac{\rho}{1})^2} \epsilon^2 + \epsilon^2 + (\epsilon)^2 < \frac{1}{2} - \frac{1}{50} ;$$

the coefficient of $kr F(w_t)k^2$ becomes c , where $c > 0$ is a universal constant. Considering the other terms and rewriting equation (3.16), we obtain

$$\begin{aligned} F(w_{t+1}) & F(w_t) + c kr F(w_t)k^2 + \frac{2L_F^2}{2} + \frac{3L_F^2}{100} + 25 \frac{3L_F^2}{2} \frac{3(1 - \frac{\rho}{1})}{2} + 50 \frac{\epsilon^2}{2} \\ & + 2L_F \epsilon^2 + \frac{9(1 + \frac{\rho}{1})^2}{2(1 - \frac{\rho}{1})^2} \epsilon^2 + \epsilon^2 + (\epsilon)^2 + \frac{1}{2} + \frac{3(1 - \frac{\rho}{1})}{2} \epsilon^2 \\ & + \frac{1}{2} + L_F \frac{9(1 + \frac{\rho}{1})^2}{(1 - \frac{\rho}{1})^2} \epsilon^2 + \epsilon^2 + (\epsilon)^2 + \frac{1}{2} + \frac{3(1 - \frac{\rho}{1})}{2} \epsilon^2 : \end{aligned}$$

Continuing, we get

$$\begin{aligned} & \frac{1}{T+1} \sum_{t=0}^T kr F(w_t) k^2 \\ & \frac{1}{c(T+1)} \sum_{t=0}^T (F(w_t) - F(w_{t+1})) \\ & + \frac{L_F^2}{2} + \frac{2L_F^2}{100} + 25L_F^2 \frac{3(1-\rho)^2}{c} + \frac{50}{c} \\ & + \frac{2L_F}{c} \frac{1}{2} + \frac{9(1+\rho)^2}{2c(1-\rho)^2} \frac{1}{2} + \frac{1}{2} + \left(\frac{\rho}{1-\rho}\right)^2 \frac{1}{2} + \frac{3(1-\rho)^2}{1+\rho} \frac{1}{2} \\ & + \frac{1}{2} + L_F \frac{9(1+\rho)^2}{c(1-\rho)^2} \frac{1}{2} + \frac{1}{2} + \left(\frac{\rho}{1-\rho}\right)^2 \frac{1}{2} + \frac{3(1-\rho)^2}{1+\rho} \frac{1}{2} \end{aligned}$$

Using the telescoping sum, we obtain

$$\begin{aligned} & \min_{t=0, \dots, T} kr F(w_t) k^2 \leq \frac{F(w_0) - F^*}{c(T+1)} \\ & + \frac{9(1+\rho)^2}{2c(1-\rho)^2} \frac{1}{2} + \frac{1}{2} + \left(\frac{\rho}{1-\rho}\right)^2 \frac{1}{2} + \frac{3(1-\rho)^2}{1+\rho} \frac{1}{2} + \frac{50}{c} \\ & + \left[\frac{L_F^2}{2} \frac{3(1-\rho)^2}{c} + \frac{2L_F}{c} \frac{1}{2} \right. \\ & + \left. \frac{1}{2} + L_F \frac{9(1+\rho)^2}{c(1-\rho)^2} \frac{1}{2} + \frac{1}{2} + \left(\frac{\rho}{1-\rho}\right)^2 \frac{1}{2} + \frac{3(1-\rho)^2}{1+\rho} \frac{1}{2} \right] \\ & + \frac{1}{2} \left(\frac{L_F^2}{100} + 25L_F^2 \right) \frac{3(1-\rho)^2}{c} \end{aligned}$$

Simplifying the above expression, we write

$$\min_{t=0, \dots, T} kr F(w_t) k^2 \leq \frac{F(w_0) - F^*}{c(T+1)} + \frac{1}{2} + \frac{1}{2} + \frac{1}{2} + \frac{1}{2} + \frac{1}{2}$$

where the definition of $\frac{1}{2}$, $\frac{1}{2}$ and $\frac{1}{2}$ are immediate from the above expression.

Chapter 4

Efficient and Robust FL with Newton Algorithm

We develop a distributed second order optimization algorithm that is communication-efficient as well as robust against Byzantine failures of the worker machines. We propose COMRADE (COMunication-efficient and Robust Approximate Distributed nEwton), an iterative second order algorithm, where the worker machines communicate *only once* per iteration with the center machine. This is in sharp contrast with the state-of-the-art distributed second order algorithms like GIANT [26] and DINGO[27], where the worker machines send (functions of) local gradient and Hessian sequentially; thus ending up communicating twice with the center machine per iteration. Moreover, we show that the worker machines can further compress the local information before sending it to the center. In addition, we employ a simple norm based thresholding rule to filter-out the Byzantine worker machines. We establish the linear-quadratic rate of convergence of COMRADE and establish that the communication savings and Byzantine resilience result in only a small statistical error rate for arbitrary convex loss functions. To the best of our knowledge, this is the first work that addresses the issue of Byzantine resilience in second order distributed optimization. Furthermore, we validate our theoretical results with extensive experiments on synthetic and benchmark LIBSVM [47] data-sets and demonstrate convergence guarantees.

4.1 Introduction

In this chapter, we propose COMRADE, a distributed approximate Newton-type algorithm that communicates less and is resilient to Byzantine workers. Specifically, we consider a distributed setup with m worker machines and one center machine. The goal is to minimize a regularized convex loss $f: \mathbb{R}^d \rightarrow \mathbb{R}$, which is additive over the available data points. Furthermore, we assume that α fraction of the worker machines are Byzantine, where $\alpha \in [0; 1/2)$. We assume that Byzantine workers can send any arbitrary values to the center machine. In addition, they may completely know the learning algorithm and are allowed to collude with each other. To the best of our knowledge, this is the first work that addresses the problem of Byzantine resilience in second order optimization.

In our proposed algorithm, the worker machines communicate *only once* per iteration with the center machine. This is in sharp contrast with the state-of-the-art distributed second order algorithms (like GIANT [26], DINGO [27], Determinantal Averaging [34]), which sequentially estimates functions of local gradients and Hessians and communicate them with the center machine. In this way, they end up communicating twice per iteration with the center machine. We show that this sequential estimation is redundant. Instead, in COMRADE, the worker machines only send a d dimensional vector, the product of the inverse of local Hessian and the local gradient. Via sketching arguments, we show that the empirical mean of the product of local Hessian inverse and local gradient is close to the global Hessian inverse and gradient product, and thus just sending the above-mentioned product is sufficient to ensure convergence. Hence, in this way, we save $O(d)$ bits of communication per iteration. Furthermore, in Section 4.5, we argue that, in order to cut down further communication, the worker machines can even compress the local Hessian inverse and gradient product. Specifically, we use a (generic) ϵ -approximate compressor ([20]) for this, that encompasses sign-based compressors like QSGD [21] and top_k sparsification [15].

For Byzantine resilience, COMRADE employs a simple thresholding policy on the norms of the local Hessian inverse and local gradient product. Note that norm-based thresholding is computationally much simpler in comparison to existing co-ordinate wise median or trimmed mean ([23]) algorithms. Since the norm of the Hessian-inverse and gradient product determines the *amount* of movement for Newton-type algorithms, this norm corresponds to a natural metric for identifying and filtering out Byzantine workers.

Our Contributions: We propose a communication efficient Newton-type algorithm that is robust to Byzantine worker machines. Our proposed algorithm, COMRADE takes as input the local Hessian inverse and gradient product (or a compressed version of it) from the worker machines, and performs a simple thresholding operation on the norm of the said vector to discard ϵ fraction of workers having largest norm values. We prove the linear-quadratic rate of convergence of our proposed algorithm for strongly convex loss functions. In particular, suppose there are m worker machines, each containing S data points; and let $\Delta_t = w_t - w$, where w_t is the t -th iterate of COMRADE, and w is the optimal model we want to estimate. In Theorem 2, we show that

$$\|\Delta_{t+1}\| \leq \max\{f_t^{(1)} \|\Delta_t\|; g_t^{(2)} \|\Delta_t\|^2\} + \left(\frac{g_t^{(3)}}{t} + \epsilon\right) \frac{1}{S};$$

where $f_t^{(i)} g_{i=1}^3$ are quantities dependent on several problem parameters. Notice that the above implies a quadratic rate of convergence when $\|\Delta_t\| \leq \frac{g_t^{(1)}}{g_t^{(2)}} = \frac{g_t^{(2)}}{g_t^{(1)}}$. Subsequently, when $\|\Delta_t\|$ becomes sufficiently small, the above condition is violated and the convergence slows down to a linear rate. The error-floor, which is $O(\frac{1}{S})$ comes from the Byzantine resilience subroutine in conjunction with the simultaneous estimation of Hessian and gradient. Furthermore, in Section 4.5, we consider worker machines compressing the local Hessian inverse and gradient product via a ϵ -approximate compressor [20], and show that the (order-wise) rate of convergence remain unchanged, and the compression factor, ϵ affects the constants only.

We experimentally validate our proposed algorithm, COMRADE, with several benchmark data-sets. We consider several types of Byzantine attacks and observe that COMRADE is robust against Byzantine worker machines, yielding better classification accuracy compared to the existing state-of-the-art second order algorithms.

A major technical challenge of this work is to approximate local gradient and Hessian simultaneously in the presence of Byzantine workers. We use sketching, similar to [26], along with the norm based Byzantine resilience technique. Using *incoherence* (defined shortly) of the local Hessian along with concentration results originating from uniform sampling, we obtain the simultaneous gradient and Hessian approximation. Furthermore, ensuring at least one non-Byzantine machine gets trimmed at every iteration of COMRADE, we control the influence of Byzantine workers.

Related Work: Second order optimization has received a lot of attention in the recent years in the distributed setting owing to its attractive convergence speed. The fundamentals of second order optimization is laid out in [24], and an extension with better convergence rates is presented in [29]. Recently, in GIANT [26] algorithm, each worker machine computes an approximate Newton direction in each iteration and the center machine averages them to obtain a *globally improved* approximate Newton direction. Furthermore, DINGO [27] generalizes second order optimization beyond convex functions by extending the Newton-MR [28] algorithm in a distributed setting. Very recently, [34] proposes Determinantal averaging to correct the inversion bias of the second order optimization. A slightly different line of work ([130], [131], [132]) uses Hessian sketching to solve a large-scale distributed learning problems.

Furthermore, as mentioned in Chapter 3, Byzantine robust optimization has also received significant interest over the years, in papers like [19, 30]. Later in [23, 31], co-ordinate wise median, trimmed mean and iterative filtering based algorithm have been proposed and optimal statistical error rate is obtained. Also, [122, 123] consider adversaries may steer convergence to bad local minimizers for non-convex optimization problems.

Organization: In Section 4.3, we first analyze COMRADE with *one round* of communication per iteration. We assume $\epsilon = 0$, and focus on the communication efficiency aspect only. Subsequently, in Section 4.4, we make $\epsilon \neq 0$, thereby addressing communication efficiency and Byzantine resilience simultaneously. Further, in Section 4.5 we augment a compression scheme along with the setting of Section 4.4. Finally, in Section 4.6, we validate our theoretical findings with experiments. Proofs of all theoretical results can be found in the supplementary material.

Notation: For a positive integer r , $[r]$ denotes the set $\{1; 2; \dots; r\}$. For a vector v , we use $\|v\|_2$ to denote the ℓ_2 norm unless otherwise specified. For a matrix X , we denote $\|X\|_2$ denotes the operator norm, $\lambda_{\max}(X)$ and $\lambda_{\min}(X)$ denote the maximum and minimum singular value. Throughout the chapter, we use $C; C_1; c; c_1$ to denote positive universal constants, whose value changes with instances.

4.2 Problem Formulation

We begin with the standard statistical learning framework for empirical risk minimization, where the objective is to minimize the following loss function:

$$f(w) = \frac{1}{n} \sum_{j=1}^n \ell_j(w^T \mathbf{x}_j) + \frac{\lambda}{2} \|w\|^2; \quad (4.1)$$

where, the loss functions $\ell_j : \mathbb{R} \rightarrow \mathbb{R}$, $j \in [n]$ are *convex, twice differentiable and smooth*. Moreover, $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n \in \mathbb{R}^d$ denote the input feature vectors and $y_1, y_2, \dots, y_n \in \mathbb{R}$ denote the corresponding responses. Furthermore, we assume that the function f is *strongly convex*, implying the existence of a unique minimizer of (4.1). We denote this minimizer by w . Note that the response y_j is captured by the corresponding loss function ℓ_j . Some examples of ℓ_j are

$$\text{logistic loss: } \ell_j(z_j) = \log(1 + \exp(-z_j y_j)); \quad \text{squared loss: } \ell_j(z_j) = \frac{1}{2} (z_j - y_j)^2$$

We consider the framework of distributed optimization with m worker machines, where the feature vectors and the loss functions $(\mathbf{x}_1, \ell_1), \dots, (\mathbf{x}_n, \ell_n)$ are partitioned homogeneously among them. Furthermore, we assume that a fraction of the worker machines are Byzantine for some $\epsilon < \frac{1}{2}$. The Byzantine machines, by nature, may send any arbitrary values to the center machine. Moreover, they can even collude with each other and plan malicious attacks with complete information of the learning algorithm.

4.3 COMRADE Can Communicate Less

We first present the Newton-type learning algorithm, namely COMRADE without any Byzantine workers, i.e., $\epsilon = 0$. It is formally given in Algorithm 4 (with $\epsilon = 0$). In each iteration of our algorithm, every worker machine computes the local Hessian and local gradient and sends the local second order update (which is the product of the inverse of the local Hessian and local gradient) to the center machine. The center machine aggregates the updates from the worker machines by averaging them and updates the model parameter w . Later the center machine broadcast the parameter w to all the worker machines.

In any iteration t , a standard Newton algorithm requires the computation of exact Hessian (\mathbf{H}_t) and gradient (\mathbf{g}_t) of the loss function which can be written as

$$\mathbf{g}_t = \frac{1}{n} \sum_{i=1}^n \ell'_j(w_t^T \mathbf{x}_i) \mathbf{x}_i + w_t; \quad \mathbf{H}_t = \frac{1}{n} \sum_{i=1}^n \ell''_j(w_t^T \mathbf{x}_i) \mathbf{x}_i \mathbf{x}_i^T + \mathbf{I}; \quad (4.2)$$

In a distributed set up, the exact Hessian (\mathbf{H}_t) and gradient (\mathbf{g}_t) can be computed in parallel in the following manner. In each iteration, the center machine ‘broadcasts’ the model parameter w_t to the worker machines and each worker machine computes its own local gradient and Hessian. Then the

center machine can compute the exact gradient and exact Hessian by averaging the the local gradient vectors and local Hessian matrices. But for each worker machine the per iteration communication complexity is $O(d)$ for the gradient computation and $O(d^2)$ for the Hessian computation. Using Algorithm 4, we reduce the communication cost to only $O(d)$ per iteration, which is the same as the first order methods.

Each worker machine possess S samples drawn uniformly from $f(\mathbf{x}_1; \xi_1); (\mathbf{x}_2; \xi_2); \dots; (\mathbf{x}_n; \xi_n)g$. By S_i , we denote the indices of the samples held by worker machine i . At any iteration t , the worker machine computes the local Hessian $\mathbf{H}_{i;t}$ and local gradient $\mathbf{g}_{i;t}$ as

$$\mathbf{g}_{i;t} = \frac{1}{S} \sum_{j \in S_i} \nabla_j (W_t^T \mathbf{x}_j) \mathbf{x}_j + W_t; \quad \mathbf{H}_{i;t} = \frac{1}{S} \sum_{j \in S_i} \nabla_j^2 (W_t^T \mathbf{x}_j) \mathbf{x}_j \mathbf{x}_j^T + \mathbf{I}; \quad (4.3)$$

It is evident from the uniform sampling that $E[\mathbf{g}_{i;t}] = \mathbf{g}_t$ and $E[\mathbf{H}_{i;t}] = \mathbf{H}_t$. The update direction from the worker machine is defined as $\hat{\mathbf{p}}_{i;t} = (\mathbf{H}_{i;t})^{-1} \mathbf{g}_{i;t}$. Each worker machine requires $O(Sd^2)$ operations to compute the Hessian matrix $\mathbf{H}_{i;t}$ and $O(d^3)$ operations to invert the matrix. In practice, the computational cost can be reduced by employing conjugate gradient method. The center machine computes the parameter update direction $\hat{\mathbf{p}}_t = \frac{1}{m} \sum_{i=1}^m \hat{\mathbf{p}}_{i;t}$.

We show that given large enough sample in each worker machine (S is large) and with incoherent data points (the information is spread out and not concentrated to a small number of sample data points), the local Hessian $\mathbf{H}_{i;t}$ is close to the global Hessian \mathbf{H}_t in spectral norm, and the local gradient $\mathbf{g}_{i;t}$ is close to the global gradient \mathbf{g}_t . Subsequently, we prove that the empirical average of the local updates acts as a good proxy for the global Newton update and achieves good convergence guarantee.

4.3.1 Theoretical Guarantee

We define the matrix $\mathbf{A}_t^> = [\mathbf{a}_1^>; \dots; \mathbf{a}_n^>] \in \mathbb{R}^{d \times n}$ where $\mathbf{a}_j = \frac{1}{\sqrt{S}} \nabla_j (W^T \mathbf{x}_j) \mathbf{x}_j$. So the exact Hessian in equation (4.2) is $\mathbf{H}_t = \frac{1}{n} \mathbf{A}_t^> \mathbf{A}_t^> + \mathbf{I}$. Also we define $\mathbf{B}_t = [\mathbf{b}_1; \dots; \mathbf{b}_n] \in \mathbb{R}^{d \times n}$ where $\mathbf{b}_i = \nabla_i (W^T \mathbf{x}_i) \mathbf{x}_i$. So the exact gradient in equation (4.2) is $\mathbf{g}_t = \frac{1}{n} \mathbf{B}_t \mathbf{1} + W_t$

Definition 10 (Coherence of a Matrix). *Let $\mathbf{A} \in \mathbb{R}^{n \times d}$ be any matrix with $\mathbf{U} \in \mathbb{R}^{n \times d}$ being its orthonormal basis (the left singular vectors). The row coherence of the matrix \mathbf{A} is defined as $\mu(\mathbf{A}) = \frac{n}{d} \max_i \|\mathbf{u}_i\|_2^2 \in [1; \frac{n}{d}]$, where \mathbf{u}_i is the i th row of \mathbf{U} .*

Remark 15. If the coherence of \mathbf{A}_t is small, it can be shown that the Hessian matrix can be approximated well via selecting a subset of rows. Note that this is a fairly common to use coherence condition as an approximation tool (see [133–135])

In the following, we assume that the Hessian matrix is L -Lipschitz (see definition below), which is a standard assumption for the analysis of the second order method for general smooth loss function (as seen in [26],[34]).

Assumption 15. *The Hessian matrix of the loss function f is L -Lipschitz continuous i.e.,*

$$\| \nabla^2 f(W) - \nabla^2 f(W^0) \|_2 \leq L \|W - W^0\|_F$$

Algorithm 4: COMMunication-efficient and Robust Approximate Distributed nEwton (COMRADE)

- 1: **Input:** Step size η , parameter $\epsilon \in (0, 1)$
 - 2: **Initialize:** Initial iterate $w_0 \in \mathbb{R}^d$
 - 3: **for** $t = 0; 1; \dots; T - 1$ **do**
 - 4: **Central machine:** broadcasts w_t
 for $i \in [m]$ **do in parallel**
 - 5: **i -th worker machine:**
 - Non-Byzantine: Computes local gradient $g_{i;t}$ and local Hessian $\mathbf{H}_{i;t}$; sends $\hat{p}_{i;t} = (\mathbf{H}_{i;t})^{-1} g_{i;t}$ to the central machine,
 - Byzantine: Generates $\tilde{p}_{i;t}$ (arbitrary), and sends it to the center machine
 - 6: **end for**
 - 6: **Center Machine:**
 - Sort the worker machines in a non decreasing order according to norm of updates $\|\hat{p}_{i;t}\|$
 - Return the indices of the first $(1 - \epsilon)$ fraction of machines as U_t ,
 - Approximate Newton Update direction : $\hat{p}_t = \frac{1}{|U_t|} \sum_{i \in U_t} \hat{p}_{i;t}$
 - Update model parameter: $w_{t+1} = w_t - \eta \hat{p}_t$.
 - 7: **end for**
-

In the following theorem, we provide the convergence rate of COMRADE (with $\epsilon = 0$) in the terms of $\Delta_t = w_t - w^*$. Also, we define $\kappa_t = \frac{\max(\mathbf{H}_t)}{\min(\mathbf{H}_t)}$ as the condition number of \mathbf{H}_t , and hence $\kappa_t \geq 1$.

Theorem 6. Let $\beta \geq 1; \frac{n}{q}$ be the coherence of \mathbf{A}_t . Suppose $\epsilon = 1$ and $s \geq \frac{3-d}{2} \log \frac{md}{\epsilon}$ for some $\epsilon \in (0, 1)$. Under Assumption 15, with probability exceeding $1 - \epsilon$, we obtain

$$\|\Delta_{t+1}\| \leq \max\left\{ \frac{\epsilon}{1 - \frac{\epsilon}{2}} \|\Delta_t\|, \frac{L}{\min(\mathbf{H}_t)} \|\Delta_t\|^2 + \frac{2}{\min(\mathbf{H}_t)} \right\};$$

where $\epsilon = \left(\frac{\epsilon}{m} + \frac{2}{1}\right)$, $\beta = \frac{\max(\mathbf{A}^T \mathbf{A})}{\max(\mathbf{A}^T \mathbf{A}) + n} \leq 1$, and

$$= \frac{1}{1 - \frac{\epsilon}{\min(\mathbf{H}_t)}} \left(1 + \frac{\beta}{2 \ln(\frac{m}{\epsilon})}\right) \frac{1}{s} \max_i \|b_i\| \tag{4.4}$$

Remark 16. It is well known that a distributed Newton method has linear-quadratic convergence rate. In Theorem 6 the quadratic term comes from the standard analysis of Newton method. The

linear term (which is small) arises owing to Hessian approximation. It gets smaller with better Hessian approximation (smaller ϵ), and thus the above rate becomes quadratic one. The small error floor arises due to the gradient approximation in the worker machines, which is essential for the one round of communication per iteration. The error floor is $\propto \frac{1}{s}$ where s is the number of samples in each worker machine. So for a sufficiently large s , the error floor becomes negligible.

Remark 17. The sample size in each worker machine is dependent on the coherence of the matrix \mathbf{A}_t and the dimension d of the problem. Theoretically, the analysis is feasible for the case of $s \geq d$ (since we work with $\mathbf{H}_{i;t}^{-1}$). However, when $s < d$, one can replace the inverse by a pseudo-inverse (modulo some changes in convergence rate).

4.4 COMRADE Can Resist Byzantine Workers

In this section, we analyze COMRADE with Byzantine workers. We assume that ϵ ($< 1/2$) fraction of worker machines are Byzantine. We define the set of Byzantine worker machines by B and the set of the good (non-Byzantine) machines by \mathcal{M} . COMRADE employs a ‘norm based thresholding’ scheme on the local Hessian inverse and gradient product to tackle the Byzantine workers.

In the t -th iteration, the center machine outputs a set U_t with $|U_t| = (1 - \epsilon)m$, consisting of the indices of the worker machines with smallest norm. Hence, we ‘trim’ the worker machines that may try to diverge the learning algorithm. We denote the set of trimmed machines as T_t . Moreover, we take $\epsilon > \frac{1}{2}$ to ensure at least one good machine falls in T_t . This condition helps us to control the Byzantine worker machines. Finally, the update is given by $\hat{\mathbf{p}}_t = \frac{1}{|T_t|} \sum_{i \in T_t} \hat{\mathbf{p}}_{i;t}$. We define:

$$\frac{2}{\text{byz}} = [3(\frac{1}{1-\epsilon})^2 + 4 \epsilon (\frac{1}{1-\epsilon})^2]^{-2}; \tag{4.5}$$

$$\frac{2}{\text{byz}} = 2(\frac{1}{1-\epsilon})^2(\frac{1}{1-\epsilon})^2 + \epsilon^2(\frac{1}{1-\epsilon})^2(\frac{1}{(1-\epsilon)m} + \frac{1}{1-\epsilon})^2 + 4 \epsilon (\frac{1}{1-\epsilon})^2[2 + (\frac{1}{1-\epsilon})^2]; \tag{4.6}$$

is defined in (4.4), $\kappa = \frac{\max(\mathbf{A}^T \mathbf{A})}{\max(\mathbf{A}^T \mathbf{A}) + n}$ and ϵ is the condition number of the exact Hessian \mathbf{H}_t .

Theorem 7. Let $\epsilon \geq 1; \frac{n}{d}$ be the coherence of \mathbf{A}_t . Suppose $\epsilon = 1$ and $s \geq \frac{3}{2} \log \frac{md}{\epsilon}$ for some $\epsilon \geq 0; 1$. For $0 < \epsilon < 1/2$, under Assumption 15, with probability exceeding $1 - \epsilon$, Algorithm 4 yields

$$\|\Delta_{t+1}\| \leq \frac{s}{\epsilon} \max \left\{ \epsilon \left(\frac{2}{\text{byz}} \right) \|\Delta_t\|; \frac{L}{\min(\mathbf{H}_t)} \|\Delta_t\|^2 g + \frac{2}{\min(\mathbf{H}_t)} \frac{\text{byz}}{\epsilon} \right\};$$

where byz and $\frac{2}{\text{byz}}$ are defined in equations (4.5) and (4.6) respectively.

The remarks of Section 4.3 is also applicable here. On top of that, we have the following remarks:

Remark 18. Compared to the convergence rate of Theorem 6, the rate here remains order-wise same even with Byzantine robustness. The coefficient of the quadratic term remains unchanged but the linear rate and the error floor suffers a little bit (by a small constant factor).

Remark 19. Note that for Theorem 7 to hold, we require $\frac{1}{1-\epsilon} \leq \frac{\rho}{\epsilon}$ for all t . In cases where ϵ is large, this can impose a stricter condition on ρ . However, we conjecture that this dependence can be improved via applying a more intricate (and perhaps computation heavy) Byzantine resilience algorithm. In this work, we kept the Byzantine resilience scheme simple at the expense of this condition on ρ .

4.5 COMRADE Can Communicate Even Less and Resist Byzantine Workers

In Section 4.3 we analyze COMRADE with an additional feature. We let the worker machines further reduce the communication cost by applying a generic class of ϵ -approximate compressor [20] on the parameter update of Algorithm 4. We first define the class of ϵ -approximate compressor:

Definition 11. An operator $Q : \mathbb{R}^d \rightarrow \mathbb{R}^d$ is defined as ϵ -approximate compressor on a set $S \subseteq \mathbb{R}^d$ if, $\forall x \in S, \|Q(x) - x\| \leq \epsilon \|x\|^2$, where $\epsilon \in [0; 1]$ is the compression factor.

The above definition can be extended for any randomized operator Q satisfying

$$\mathbb{E}(\|Q(x) - x\|^2) \leq \epsilon \|x\|^2;$$

for all $x \in S$. The expectation is taken over the randomization of the operator. Notice that $\epsilon = 0$ implies that $Q(x) = x$ (no compression). Examples of ϵ -approximate compressor include QSGD [21], ϵ -QSGD [20], top_k sparsification and rand_k [15].

Worker machine i computes the product of local Hessian inverse and local gradient and then apply ϵ -approximate compressor to obtain $Q(\mathbf{H}_{i,t}^{-1} \mathbf{g}_{i,t})$; and finally sends this compressed vector to the center. The Byzantine resilience subroutine remains the same—except, instead of sorting with respect to $\|\mathbf{H}_{i,t}^{-1} \mathbf{g}_{i,t}\|$, the center machine now sorts according to $\|Q(\mathbf{H}_{i,t}^{-1} \mathbf{g}_{i,t})\|$. The center machine aggregates the compressed updates by averaging $Q(\hat{\mathbf{p}}) = \frac{1}{\sum_{i \in \mathcal{U}_t} w_{i,t}} \sum_{i \in \mathcal{U}_t} Q(\hat{\mathbf{p}}_{i,t})$, and take the next step as $W_{t+1} = W_t + Q(\hat{\mathbf{p}})$.

Recall the definition of ϵ from (4.4). We also use the following notation: $\frac{\epsilon}{\mathcal{M}} = \left(\frac{\epsilon}{(1-\epsilon)^m} + \frac{\epsilon^2}{1-\epsilon} \right)$; $\frac{1}{1-\epsilon} = \frac{1}{1-\epsilon}$ and $\frac{\epsilon}{\max(\mathbf{A}^T \mathbf{A}) + n}$. Furthermore, we define the following:

$$\frac{\epsilon^2}{\text{comp:byz}} = [3 \left(\frac{1}{1-\epsilon} \right)^2 + 4 \epsilon \left(\frac{1}{1-\epsilon} \right)^2] (1 + (1-\epsilon)) \quad (4.7)$$

$$\begin{aligned} \frac{\epsilon^2}{\text{comp:byz}} &= 2 \left(\frac{1}{1-\epsilon} \right)^2 \left(\frac{\epsilon}{1-\epsilon} + \epsilon (1-\epsilon) \right) \left((1 + \frac{\epsilon}{1-\epsilon}) \right) + \left(\frac{1}{1-\epsilon} \right)^2 \left(\frac{\epsilon}{\mathcal{M}} + \epsilon (1-\epsilon) \right) \left((1 + \frac{\epsilon}{1-\epsilon}) \right) \\ &\quad + 4 \epsilon \left(\frac{1}{1-\epsilon} \right)^2 \left(2 + \left(\frac{\epsilon}{1-\epsilon} + \epsilon (1-\epsilon) \right) \left((1 + \frac{\epsilon}{1-\epsilon}) \right) \right) \end{aligned} \quad (4.8)$$

Theorem 8. Let $\epsilon \in [0; \frac{n}{d}]$ be the coherence of \mathbf{A}_t . Let $\epsilon = 1$ and $s \geq \frac{3}{2} \log \frac{md}{\epsilon}$ for some $\epsilon \in (0; 1)$. For $0 < \epsilon < 1/2$, under Assumption 15 and with Q being the ϵ -approximate

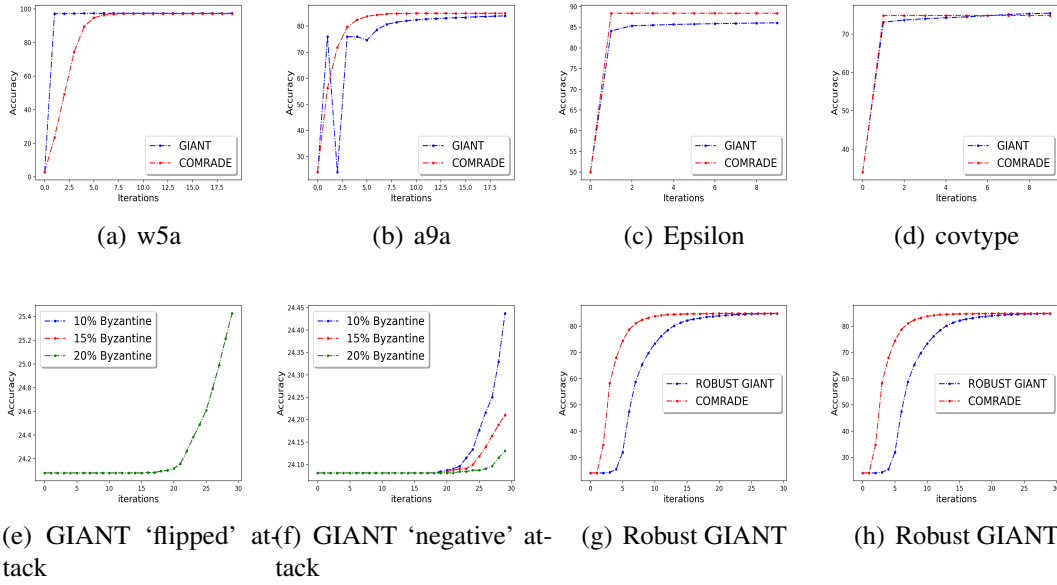


Figure 4.1: (First row) Comparison of training accuracy between COMRADE(Algorithm 4) and GIANT [26] with (a) w5a (b) a9a (c) Epsilon (d) Covtype dataset. (Second row) Training accuracy of (e) GIANT for ‘flipped label’ and (f) ‘negative update’ attack; and comparison of Robust GIANT and COMRADE with a9a dataset for (g) ‘flipped label’ and (h) ‘negative update’ attack.

compressor, with probability exceeding $1 - \epsilon$, we obtain

$$k\Delta_{t+1}k \leq \max\left\{ \frac{S}{t} \left(\frac{2}{1 - \frac{\text{comp;byz}}{2}} \right) k\Delta_tk; \frac{L}{\min(\mathbf{H}_t)} k\Delta_tk^2 g + \frac{\text{comp;byz}}{\min(\mathbf{H}_t)} \right\}$$

where comp;byz and comp;byz are given in equations (4.7) and (4.8) respectively.

Remark 20. With no compression ($\alpha = 1$) we get back the convergence guarantee of Theorem 7.

Remark 21. Note that even with compression, we retain the linear-quadratic rate of convergence of COMRADE. The constants are affected by a α -dependent term.

4.6 Experimental Results

In this section we validate our algorithm, COMRADE in Byzantine and non-Byzantine setup on synthetically generated and benchmark LIBSVM [47] data-set. The experiments focus on the standard logistic regression problem. The logistic regression objective is defined as

$$\frac{1}{n} \sum_{i=1}^n \log(1 + \exp(-y_i \mathbf{x}_i^T \mathbf{w})) + \frac{\lambda}{2n} \|\mathbf{w}\|^2;$$

where $w \in \mathbb{R}^d$ is the parameter, $\{x_i, g_{i=1}^n\} \in \mathbb{R}^d$ are the feature data and $\{y_i, g_{i=1}^n\} \in \{0, 1\}^n$ are the corresponding labels. We use ‘mpi4py’ package for distributed framework (swarm2) at the University of Massachusetts Amherst [136] using mpi4py Python package. We choose ‘a9a’ ($d = 123; n = 32K$), ‘w5a’ ($d = 300; n = 10k$), ‘Epsilon’ ($d = 2000; n = 0.4M$) and ‘covtype.binary’ ($d = 54; n = 0.5M$) classification datasets and partition the data in 20 different worker machines. In the experiments, we choose two types of Byzantine attacks : (1). ‘flipped label’-attack where (for binary classification) the Byzantine worker machines flip the labels of the data, thus making the model learn with wrong labels, and (2). ‘negative update attack’ where the Byzantine worker machines compute the local update (\hat{p}_i) and communicate $c \hat{p}_i$ with $c \in (0, 1)$ making the updates to be opposite of actual direction. We choose $c = -\frac{2}{m}$. We choose the regularization parameter $\lambda = 1$ and fixed step size. We ran the algorithms sufficient number of steps to ensure convergence.

In Figure 4.1(first row) we compare COMRADE in non-Byzantine setup ($\lambda = \lambda = 0$) with the state-of the art algorithm GIANT [26]. It is evident from the plot that despite the fact that COMRADE requires less communication, the algorithm is able to achieve similar accuracy. Also, we show the ineffectiveness of GIANT in the presence of Byzantine attacks. In Figure 4.2((e),(f)) we show the accuracy for flipped label and negative update attacks. These plots are an indicator of the requirement of robustness in the learning algorithm. So we devise ‘Robust GIANT’, which is GIANT algorithm with added ‘norm based thresholding’ for robustness. In particular, we trim the worker machines based on the local gradient norm in the first round of communication of GIANT. Subsequently, in the second round of communication, the non-trimmed worker machines send the updates (product of local Hessian inverse and the local gradient) to the center machine. We compare COMRADE with ‘Robust GIANT’ in Figure 4.1((g),(h)) with 10% Byzantine worker machines for ‘a9a’ dataset. It is evident plot that COMRADE performs better than the ‘Robust GIANT’.

Next we show the accuracy of COMRADE with different numbers of Byzantine worker machines. Here we choose $c = 0.9$. We show the accuracy for ‘negative update’ attack in Figure 4.2(first row) and ‘flipped label’ attack in Figure 4.2 (second row). Furthermore, we show that COMRADE works even when ϵ -approximate compressor is applied to the updates. In Figure 4.2(Third row) we plot the training accuracies. For compression we apply the scheme known as QSGD [21]. Further experiments can be found in the supplementary material.

4.7 Conclusion

In this paper, we address the issue of communication efficiency and Byzantine robustness via second order optimization and norm based thresholding respectively for strongly convex loss. Extending our setting to handle weakly convex and non-convex loss is of immediate interest. We would also like to exploit local averaging with second order optimization. Moreover, an important aspect, privacy, is not addressed in this work. We keep this as our future research direction.

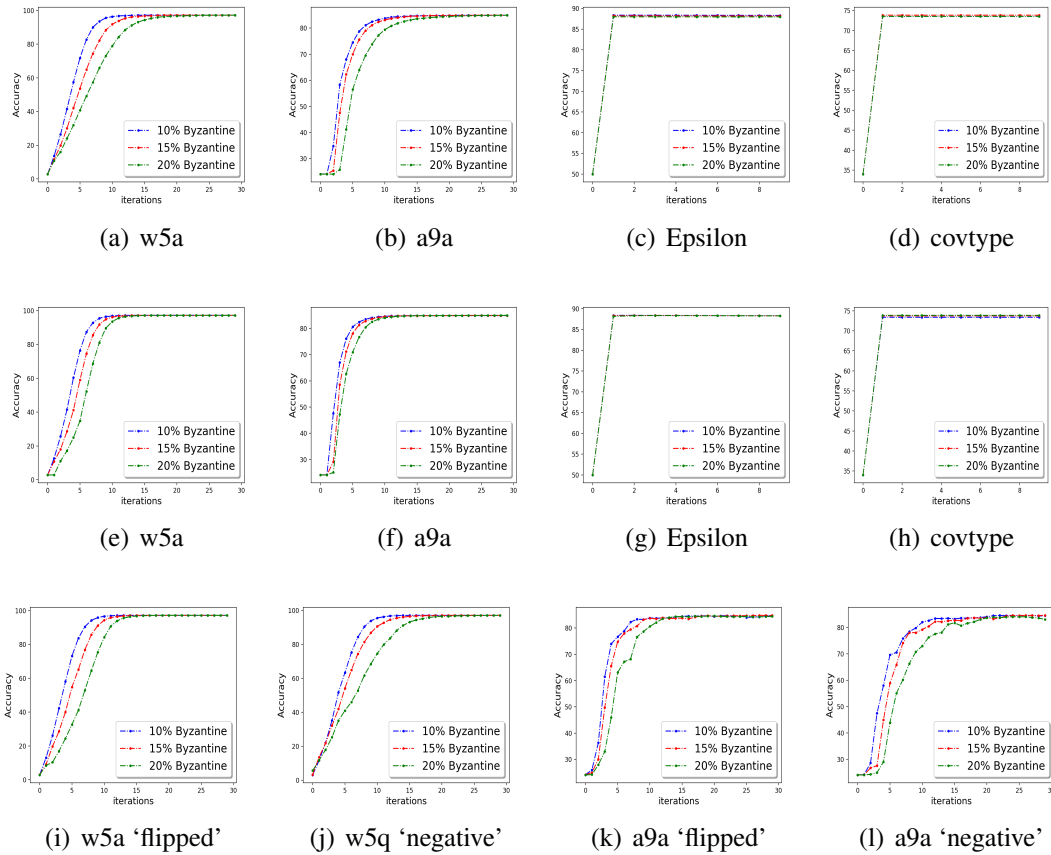


Figure 4.2: (First row) Accuracy of COMRADE with 10%;15%;20% Byzantine workers with ‘negative update’ attack for (a). w5a (b). a9a (c). covtype (d). Epsilon. (Second row) COMRADE accuracy with 10%;15%;20% Byzantine workers with ‘flipped label’ attack for (e) w5a (f) a9a (g) covtype (h) Epsilon. (Third row) Accuracy of COMRADE with -approximate compressor (Section 4.5) with 10%;15%;20% Byzantine workers; (i) ‘flipped label’ attack for w5a (j) ‘negative update’ attack for w5a. (k) ‘flipped label’ attack for a9a . (l) ‘negative update’ attack for a9a dataset.

Appendix

4.8 Analysis of Section 4.3

Matrix Sketching

Here we briefly discuss the matrix sketching that is broadly used in the context of *randomized linear algebra*. For any matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ the sketched matrix $\mathbf{Z} \in \mathbb{R}^{s \times d}$ is defined as $\mathbf{S}^T \mathbf{A}$ where $\mathbf{S} \in \mathbb{R}^{n \times s}$ is the sketching matrix (typically $s < n$). Based on the scope and basis of the application, the sketched matrix is constructed by taking linear combination of the rows of matrix which is known as *random projection* or by sampling and scaling a subset of the rows of the matrix which is known as *random sampling*. The sketching is done to get a smaller representation of the original

matrix to reduce computational cost.

Here we consider a uniform row sampling scheme. The matrix \mathbf{Z} is formed by sampling and scaling rows of the matrix \mathbf{A} . Each row of the matrix \mathbf{A} is sampled with probability $\rho = \frac{1}{n}$ and scaled by multiplying with $\frac{1}{\rho^{1/sp}}$.

$$\mathbb{P} \mathbf{z}_i = \frac{\mathbf{a}_j}{\rho^{1/sp}} = \rho;$$

where \mathbf{z}_i is the i -th row matrix \mathbf{Z} and \mathbf{a}_j is the j th row of the matrix \mathbf{A} . Consequently the sketching matrix \mathbf{S} has one non-zero entry in each column.

We define the matrix $\mathbf{A}_t^> = [\mathbf{a}_1^> \dots \mathbf{a}_n^>] \in \mathbb{R}^{d \times n}$ where $\mathbf{a}_j = \frac{1}{\rho^{1/sp}} \mathbf{a}_j$. So the exact Hessian in equation (4.2) is $\mathbf{H}_t = \frac{1}{n} \mathbf{A}_t^> \mathbf{A}_t + \mathbf{I}$. Assume that S_i is the set of features that are held by the i th worker machine. So the local Hessian is

$$\mathbf{H}_{i,t} = \frac{1}{S} \sum_{j \in S_i} \frac{1}{\rho^{1/sp}} \mathbf{a}_j \mathbf{a}_j^> + \mathbf{I} = \frac{1}{S} \mathbf{A}_{i,t}^> \mathbf{A}_{i,t} + \mathbf{I};$$

where $\mathbf{A}_{i,t} \in \mathbb{R}^{s \times d}$ and the row of the matrix $\mathbf{A}_{i,t}$ is indexed by S_i . Also we define $\mathbf{B}_t = [\mathbf{b}_1 \dots \mathbf{b}_n] \in \mathbb{R}^{d \times n}$ where $\mathbf{b}_i = \frac{1}{\rho^{1/sp}} \mathbf{a}_i$. So the exact gradient in equation (4.2) is $\mathbf{g}_t = \frac{1}{n} \mathbf{B}_t \mathbf{1} + \mathbf{w}_t$ and the local gradient is

$$\mathbf{g}_{i,t} = \frac{1}{S} \sum_{i \in S_i} \frac{1}{\rho^{1/sp}} \mathbf{a}_i \mathbf{w}_t = \frac{1}{S} \mathbf{B}_{i,t} \mathbf{1} + \mathbf{w}_t;$$

where $\mathbf{B}_{i,t}$ is the matrix with column indexed by S_i . If $\mathbf{S}_i \in \mathbb{R}^{s \times d}$ are the sketching matrices then the local Hessian and gradient can be expressed as

$$\mathbf{H}_{i,t} = \mathbf{A}_t^> \mathbf{S}_i \mathbf{S}_i^> \mathbf{A}_t^> + \mathbf{I} \quad \mathbf{g}_{i,t} = \frac{1}{n} \mathbf{B}_t \mathbf{S}_i^> \mathbf{1} + \mathbf{w}_t; \tag{4.9}$$

With the help of sketching idea later we show that the local hessian and gradient are close to the exact hessian and gradient.

The Quadratic function For the purpose of analysis we define an auxiliary quadratic function

$$(\mathbf{p}) = \frac{1}{2} \mathbf{p}^> \mathbf{H}_t \mathbf{p} \quad \mathbf{g}_t^> \mathbf{p} = \frac{1}{2} \mathbf{p}^> (\mathbf{A}_t^> \mathbf{A}_t + \mathbf{I}) \mathbf{p} \quad \mathbf{g}_t^> \mathbf{p}; \tag{4.10}$$

The optimal solution to the above function is

$$\mathbf{p} = \arg \min (\mathbf{p}) = \mathbf{H}_t^{-1} \mathbf{g}_t = (\mathbf{A}_t^> \mathbf{A}_t + \mathbf{I})^{-1} \mathbf{g}_t;$$

which is also the optimal direction of the global Newton update. In this work we consider the local and global (approximate) Newton direction to be

$$\hat{\mathbf{p}}_{i,t} = (\mathbf{A}_t^> \mathbf{S}_i \mathbf{S}_i^> \mathbf{A}_t + \mathbf{I})^{-1} \mathbf{g}_{i,t}; \quad \hat{\mathbf{p}}_t = \frac{1}{m} \sum_{i=1}^m \hat{\mathbf{p}}_{i,t};$$

respectively. And it can be easily verified that each local update $\hat{\mathbf{p}}_{i;t}$ is optimal solution to the following quadratic function

$$\hat{\mathbf{p}}_{i;t}(\rho) = \frac{1}{2} \mathbf{p}^T (\mathbf{A}^T \mathbf{S}_i \mathbf{S}_i^T \mathbf{A} + \mathbf{I}) \mathbf{p} - \mathbf{g}_i^T \mathbf{p}. \tag{4.11}$$

In our convergence analysis we show that value of the quadratic function in (4.10) with value $\hat{\mathbf{p}}_t$ is close to the optimal value.

Singular Value Decomposition (SVD) For any matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ with rank r , the singular value decomposition is defined as $\mathbf{A} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T$ where $\mathbf{U}; \mathbf{V}$ are $n \times r$ and $d \times r$ column orthogonal matrices respectively and $\mathbf{\Sigma}$ is a $r \times r$ diagonal matrix with diagonal entries $\sigma_1, \dots, \sigma_r$. If \mathbf{A} is a symmetric positive semi-definite matrix then $\mathbf{U} = \mathbf{V}$.

4.8.1 Analysis

Lemma 10 (McDiarmid’s Inequality). *Let $X = (X_1, \dots, X_m)$ be m independent random variables taking values from some set A , and assume that $f : A^m \rightarrow \mathbb{R}$ satisfies the following condition (bounded differences):*

$$\sup_{x_1, \dots, x_m, \hat{x}_i} |f(x_1, \dots, x_i, \hat{x}_i, \dots, x_m) - f(x_1, \dots, x_i, x_i, \dots, x_m)| \leq c_i$$

for all $i \in \{1, \dots, m\}$. Then for any $\epsilon > 0$ we have

$$P[|f(X_1, \dots, X_m) - E[f(X_1, \dots, X_m)]| \geq \epsilon] \leq \exp\left(-\frac{2\epsilon^2}{\sum_{i=1}^m c_i^2}\right)$$

The property described in the following Lemma 11 is a very useful result for uniform row sampling sketching matrix.

Lemma 11 (Lemma 8 [26]). *Let $\epsilon \in (0, 1)$ be a fixed parameter and $r = \text{rank}(\mathbf{A}_t)$ and $\mathbf{U} \in \mathbb{R}^{n \times r}$ be the orthonormal bases of the matrix \mathbf{A}_t . Let $\mathbf{S}_i \in \mathbb{R}^{n \times r}$ be sketching matrices and $\mathbf{S} = \frac{1}{m} [\mathbf{S}_1; \dots; \mathbf{S}_m] \in \mathbb{R}^{n \times ms}$. With probability $1 - \epsilon$ the following holds*

$$\mathbf{U}^T \mathbf{S}_i \mathbf{S}_i^T \mathbf{U} \preceq \mathbf{I}_{2 \times 2} \quad \forall i \in [m] \quad \text{and} \quad \mathbf{U}^T \mathbf{S} \mathbf{S}^T \mathbf{U} \preceq \mathbf{I}_{2 \times 2} + \frac{\epsilon}{m}$$

Lemma 12. *Let $\mathbf{S} \in \mathbb{R}^{n \times s}$ be any uniform sampling sketching matrix, then for any matrix $\mathbf{B} = [\mathbf{b}_1; \dots; \mathbf{b}_n] \in \mathbb{R}^{d \times n}$ with probability $1 - \epsilon$ for any $\epsilon > 0$ we have,*

$$\frac{1}{n} \mathbf{B} \mathbf{S} \mathbf{S}^T \mathbf{1} \preceq \frac{1}{n} \mathbf{B} \mathbf{1} \mathbf{k} \left(1 + \frac{r}{2 \ln(\frac{1}{\epsilon})}\right) \preceq \frac{1}{s} \max_i \|\mathbf{b}_i\| \mathbf{k}$$

where $\mathbf{1}$ is all ones vector.

Proof. The vector $\mathbf{B}\mathbf{1}$ is the sum of column of the matrix \mathbf{B} and $\mathbf{BSS}^>\mathbf{1}$ is the sum of uniformly sampled and scaled column of the matrix \mathbf{B} where the scaling factor is $\frac{1}{sp}$ with $p = \frac{1}{n}$. If $(i_1; \dots; i_s)$ is the set of sampled indices then $\mathbf{BSS}^>\mathbf{1} = \sum_{k \in \{i_1, \dots, i_s\}} \frac{1}{sp} \mathbf{b}_k$.

Define the function $f(i_1; \dots; i_s) = k \frac{1}{n} \mathbf{BSS}^>\mathbf{1} - \frac{1}{n} \mathbf{B}\mathbf{1}k$. Now consider a sampled set $(i_1; \dots; i_{j_0}; \dots; i_s)$ with only one item (column) replaced then the bounded difference is

$$\begin{aligned} &= |f(i_1; \dots; i_j; \dots; i_s) - f(i_1; \dots; i_{j_0}; \dots; i_s)| \\ &= \left| j \frac{1}{n} k \frac{1}{sp} \mathbf{b}_{i_j} - \frac{1}{sp} \mathbf{b}_{i_j} k j - \left(j \frac{1}{n} k \frac{1}{sp} \mathbf{b}_{i_{j_0}} - \frac{1}{sp} \mathbf{b}_{i_{j_0}} k j \right) \right| \\ &= \frac{2}{s} \max_i k \mathbf{b}_i k \end{aligned}$$

Now we have the expectation

$$\begin{aligned} &E \left[k \frac{1}{n} \mathbf{BSS}^>\mathbf{1} - \frac{1}{n} \mathbf{B}\mathbf{1}k \right]^2 = \frac{n}{sn^2} \sum_{i=1}^r k \mathbf{b}_i k^2 = \frac{1}{s} \max_i k \mathbf{b}_i k^2 \\ &E \left[k \frac{1}{n} \mathbf{BSS}^>\mathbf{1} - \frac{1}{n} \mathbf{B}\mathbf{1}k \right] \leq \frac{1}{s} \max_i k \mathbf{b}_i k \end{aligned}$$

Using McDiarmid inequality (Lemma 10) we have

$$P \left[\left| k \frac{1}{n} \mathbf{BSS}^>\mathbf{1} - \frac{1}{n} \mathbf{B}\mathbf{1}k - \frac{1}{s} \max_i k \mathbf{b}_i k \right| \geq t \right] \leq \exp \left(-\frac{2t^2}{s} \right)$$

Equating the probability with ϵ we have

$$t = \frac{s}{2} \ln \left(\frac{1}{\epsilon} \right) = \max_i k \mathbf{b}_i k \frac{s}{2} \ln \left(\frac{1}{\epsilon} \right)$$

Finally we have with probability $1 - \epsilon$

$$\left| k \frac{1}{n} \mathbf{BSS}^>\mathbf{1} - \frac{1}{n} \mathbf{B}\mathbf{1}k \right| \leq \left(1 + \frac{s}{2 \ln \left(\frac{1}{\epsilon} \right)} \right) \frac{1}{s} \max_i k \mathbf{b}_i k$$

□

Remark 22. For m sketching matrix $\{S_i\}_{i=1}^m$, the bound in the Lemma 12 is

$$\left| k \frac{1}{n} \mathbf{BS}_i S_i^>\mathbf{1} - \frac{1}{n} \mathbf{B}\mathbf{1}k \right| \leq \left(1 + \frac{s}{2 \ln \left(\frac{1}{\epsilon} \right)} \right) \frac{1}{s} \max_i k \mathbf{b}_i k$$

with probability $1 - \epsilon$ for any $\epsilon > 0$ for all $i \in \{1, 2, \dots, m\}$. In the case that each worker machine holds data based on the uniform sketching matrix the local gradient is close to the exact gradient. Thus the local second order update acts as a good approximate to the exact Newton update.

Now we consider the update rule of GIANT [26] where the update is done in two rounds in each iteration. In the first round each worker machine computes and send the local gradient and the center machine computes the exact gradient \mathbf{g}_t in iteration t . Next the center machine broadcasts the exact gradient and each worker machine computes the local Hessian and send $\mathbf{p}_{i;t} = (\mathbf{H}_{i;t})^{-1} \mathbf{g}_t$ to the center machine and the center machine computes the approximate Newton direction $\hat{\mathbf{p}}_t = \frac{1}{m} \sum_{i=1}^m \mathbf{p}_{i;t}$. Now based on this we restate the following lemma (Lemma 6 [26]).

Lemma 13. Let $\{\mathbf{S}_i\}_{i=1}^m \subset \mathbb{R}^{n \times s}$ be sketching matrices based on Lemma 11. Let \mathcal{L}_t be defined in (4.10) and $\hat{\mathbf{p}}_t$ be the update. It holds that

$$\min_{\mathbf{p}} \mathcal{L}_t(\mathbf{p}) \leq \mathcal{L}_t(\hat{\mathbf{p}}_t) + (1 + \frac{1}{\rho}) \min_{\mathbf{p}} \mathcal{L}_t(\mathbf{p});$$

where $\rho = (\frac{\rho}{m} + \frac{1}{\rho})$ and $\frac{1}{\rho} = \frac{\max(\mathbf{A} \succ \mathbf{A})}{\max(\mathbf{A} \succ \mathbf{A}) + n} - 1$.

Now we prove similar guarantee for the update according to COMRADE in Algorithm 4.

Lemma 14. Let $\{\mathbf{S}_i\}_{i=1}^m \subset \mathbb{R}^{n \times s}$ be sketching matrices based on Lemma 11. Let \mathcal{L}_t be defined in (4.10) and $\hat{\mathbf{p}}_t$ be defined in Algorithm 4 ($\epsilon = 0$)

$$\min_{\mathbf{p}} \mathcal{L}_t(\mathbf{p}) \leq \mathcal{L}_t(\hat{\mathbf{p}}_t) + \frac{1}{\rho} \min_{\mathbf{p}} \mathcal{L}_t(\mathbf{p});$$

where $\rho = \frac{1}{1 - \frac{\rho}{m}} \frac{1}{\min(\mathbf{H}_t)} (1 + \frac{1}{2} \ln(\frac{m}{s})) \frac{1}{s} \max_i \|\mathbf{b}_i\|_2$ and $\frac{1}{\rho} = (\frac{\rho}{m} + \frac{1}{\rho})$ and $\frac{1}{\rho} = \frac{\max(\mathbf{A} \succ \mathbf{A})}{\max(\mathbf{A} \succ \mathbf{A}) + n}$.

Proof. First consider the quadratic function (4.10)

$$\begin{aligned} \mathcal{L}_t(\hat{\mathbf{p}}_t) - \mathcal{L}_t(\mathbf{p}) &= \frac{1}{2} \|\mathbf{H}_t^{\frac{1}{2}}(\hat{\mathbf{p}}_t - \mathbf{p})\|_2^2 \\ &= \underbrace{\|\mathbf{H}_t^{\frac{1}{2}}(\hat{\mathbf{p}}_t - \mathbf{p}_t)\|_2^2}_{\text{Term1}} + \underbrace{\|\mathbf{H}_t^{\frac{1}{2}}(\mathbf{p}_t - \mathbf{p})\|_2^2}_{\text{Term2}}; \end{aligned} \quad (4.12)$$

where $\mathbf{p}_t = \frac{1}{m} \sum_{i=1}^m (\mathbf{H}_{i;t})^{-1} \mathbf{g}_t$. First we bound the Term 2 of (4.12) using the quadratic function and Lemma 13

$$\begin{aligned} \frac{1}{2} \|\mathbf{H}_t^{\frac{1}{2}}(\mathbf{p}_t - \mathbf{p})\|_2^2 &\leq \frac{1}{2} \|\mathbf{H}_t^{\frac{1}{2}} \mathbf{p}\|_2^2 \quad (\text{Using Lemma 13}) \\ &= \frac{1}{2} \mathcal{L}_t(\mathbf{p}); \end{aligned} \quad (4.13)$$

The step in equation (4.13) is from the definition of the function \mathcal{L}_t and \mathbf{p} . It can be shown that

$$\mathcal{L}_t(\mathbf{p}) = \frac{1}{2} \|\mathbf{H}_t^{\frac{1}{2}} \mathbf{p}\|_2^2;$$

Now we bound the Term 1 in (4.12). By Lemma 11, we have $(1 - \frac{\epsilon}{2})\mathbf{A}_t^>\mathbf{A}_t - \mathbf{A}_t^>\mathbf{S}_i\mathbf{S}_i^>\mathbf{A}_t$ $(1 + \frac{\epsilon}{2})\mathbf{A}_t^>\mathbf{A}_t$. Following we have $(1 - \frac{\epsilon}{2})\mathbf{H}_t - \mathbf{H}_{i;t} \leq (1 + \frac{\epsilon}{2})\mathbf{H}_t$. Thus there exists matrix \mathbf{H}_t^{-1} satisfying

$$\mathbf{H}_t^{\frac{1}{2}}\mathbf{H}_{i;t}^{-1}\mathbf{H}_t^{\frac{1}{2}} = \mathbf{I} + \frac{\epsilon}{2}\mathbf{I} \quad \text{and} \quad \frac{1}{1 + \frac{\epsilon}{2}} \leq \frac{1}{1 - \frac{\epsilon}{2}};$$

So we have,

$$\mathbf{H}_t^{\frac{1}{2}}\mathbf{H}_{i;t}^{-1}\mathbf{H}_t^{\frac{1}{2}} \leq \frac{1}{1 - \frac{\epsilon}{2}} = \frac{1}{1 - \frac{\epsilon}{2}}. \quad (4.14)$$

Now we have

$$\begin{aligned} \mathbf{H}_t^{\frac{1}{2}}(\hat{\mathbf{p}}_t - \mathbf{p}_t) &= \mathbf{H}_t^{\frac{1}{2}} \frac{1}{m} \sum_{i=1}^m (\hat{\mathbf{p}}_{i;t} - \mathbf{p}_{i;t}) \\ &= \frac{1}{m} \sum_{i=1}^m \mathbf{H}_t^{\frac{1}{2}} (\hat{\mathbf{p}}_{i;t} - \mathbf{p}_{i;t}) \\ &= \frac{1}{m} \sum_{i=1}^m \mathbf{H}_t^{\frac{1}{2}} \mathbf{H}_{i;t}^{-1} (\mathbf{g}_{i;t} - \mathbf{g}_t) \\ &= \frac{1}{m} \sum_{i=1}^m \mathbf{H}_t^{\frac{1}{2}} \mathbf{H}_{i;t}^{-1} \mathbf{H}_t^{\frac{1}{2}} \mathbf{H}_t^{-\frac{1}{2}} (\mathbf{g}_{i;t} - \mathbf{g}_t) \\ &= \frac{1}{m} \sum_{i=1}^m \mathbf{H}_t^{\frac{1}{2}} \mathbf{H}_{i;t}^{-1} \mathbf{H}_t^{\frac{1}{2}} \mathbf{H}_t^{-\frac{1}{2}} (\mathbf{g}_{i;t} - \mathbf{g}_t) \\ &= \frac{1}{1 - \frac{\epsilon}{2}} \frac{1}{m} \sum_{i=1}^m \mathbf{H}_t^{-\frac{1}{2}} (\mathbf{g}_{i;t} - \mathbf{g}_t) \quad (\text{Using (4.14)}) \\ &= \frac{1}{1 - \frac{\epsilon}{2}} \frac{1}{\rho \frac{1}{\min(\mathbf{H}_t)}} \frac{1}{m} \sum_{i=1}^m k(\mathbf{g}_{i;t} - \mathbf{g}_t)k. \end{aligned} \quad (4.15)$$

Now we bound $k(\mathbf{g}_{i;t} - \mathbf{g}_t)k$ using Lemma 12,

$$k(\mathbf{g}_{i;t} - \mathbf{g}_t)k = k \frac{1}{n} \mathbf{B} \mathbf{S} \mathbf{S}^> \mathbf{1} - \frac{1}{n} \mathbf{B} \mathbf{1} k \leq (1 + \frac{\epsilon}{2}) \frac{1}{2 \ln(\frac{m}{\epsilon})} \frac{1}{S} \max_i k \mathbf{b}_i k;$$

Plugging it into equation (4.15) we get,

$$\begin{aligned} \mathbf{H}_t^{\frac{1}{2}}(\hat{\mathbf{p}}_t - \mathbf{p}_t) &= \frac{1}{1 - \frac{\epsilon}{2}} \frac{1}{\rho \frac{1}{\min(\mathbf{H}_t)}} \frac{1}{m} \sum_{i=1}^m k(\mathbf{g}_{i;t} - \mathbf{g}_t)k \\ &= \frac{1}{1 - \frac{\epsilon}{2}} \frac{1}{\rho \frac{1}{\min(\mathbf{H}_t)}} (1 + \frac{\epsilon}{2}) \frac{1}{2 \ln(\frac{m}{\epsilon})} \frac{1}{S} \max_i k \mathbf{b}_i k. \end{aligned} \quad (4.16)$$

Now collecting the terms of (4.16) and (4.13) and plugging them into (4.12) we have

$$f_t(\hat{\mathbf{p}}_t) - f_t(\mathbf{p}^*) \leq \frac{1}{2} \mathbf{g}_t^\top \mathbf{H}_t \mathbf{g}_t + \frac{L}{2} \|\Delta_t\|^2 + (1 - \frac{1}{2}) f_t(\mathbf{p}^*);$$

where \mathbf{g}_t is as defined in (4.4). □

Lemma 15. Let $\alpha \in (0, 1)$; be any fixed parameter. And $\hat{\mathbf{p}}_t$ satisfies $f_t(\hat{\mathbf{p}}_t) \leq \frac{1}{2} \mathbf{g}_t^\top \mathbf{H}_t \mathbf{g}_t + (1 - \alpha) \min_{\mathbf{p}} f_t(\mathbf{p})$. Under the Assumption 15 (Hessian L-Lipschitz) and $\Delta_t = w_t - w^*$ satisfies

$$\Delta_{t+1}^\top \mathbf{H}_t \Delta_{t+1} \leq L k \Delta_{t+1} k k \Delta_t k^2 + \frac{1}{2} \Delta_t^\top \mathbf{H}_t \Delta_t + 2 \alpha^2;$$

Proof. We have $w_{t+1} = w_t + \alpha \hat{\mathbf{p}}_t$; $\Delta_t = w_t - w^*$ and $\Delta_{t+1} = w_{t+1} - w^*$. Also $\hat{\mathbf{p}}_t = w_t - w_{t+1} = \Delta_t - \Delta_{t+1}$. From the definition of \mathbf{g}_t we have,

$$f_t(\hat{\mathbf{p}}_t) = \frac{1}{2} (\Delta_t - \Delta_{t+1})^\top \mathbf{H}_t (\Delta_t - \Delta_{t+1}) - (\Delta_t - \Delta_{t+1})^\top \mathbf{g}_t;$$

$$(1 - \alpha) \min_{\mathbf{p}} f_t(\mathbf{p}) \leq \frac{1}{2} \Delta_t^\top \mathbf{H}_t \Delta_t - \Delta_t^\top \mathbf{g}_t;$$

From the above two equation we have

$$f_t(\hat{\mathbf{p}}_t) - (1 - \alpha) \min_{\mathbf{p}} f_t(\mathbf{p}) \leq \frac{1}{2} \Delta_t^\top \mathbf{H}_t \Delta_t - \Delta_t^\top \mathbf{g}_t - \frac{1}{2} (\Delta_t - \Delta_{t+1})^\top \mathbf{H}_t (\Delta_t - \Delta_{t+1}) + (\Delta_t - \Delta_{t+1})^\top \mathbf{g}_t;$$

$$= \frac{1}{2} \Delta_{t+1}^\top \mathbf{H}_t \Delta_{t+1} - \frac{1}{2} \Delta_t^\top \mathbf{H}_t \Delta_{t+1} + \frac{1}{2} \Delta_{t+1}^\top \mathbf{g}_t - \frac{1}{2} \Delta_t^\top \mathbf{H}_t \Delta_t;$$

From Lemma 14 the following holds

$$f_t(\hat{\mathbf{p}}_t) - \frac{1}{2} \Delta_{t+1}^\top \mathbf{H}_t \Delta_{t+1} + \frac{1}{2} \Delta_{t+1}^\top \mathbf{g}_t \leq \frac{1}{2} \Delta_t^\top \mathbf{H}_t \Delta_t + (1 - \alpha) \min_{\mathbf{p}} f_t(\mathbf{p}) - \frac{1}{2} \Delta_t^\top \mathbf{H}_t \Delta_{t+1};$$

So we have

$$\frac{1}{2} \Delta_{t+1}^\top \mathbf{H}_t \Delta_{t+1} - \Delta_t^\top \mathbf{H}_t \Delta_{t+1} + \Delta_{t+1}^\top \mathbf{g}_t \leq \frac{1}{2} \Delta_t^\top \mathbf{H}_t \Delta_t + 2 \alpha^2; \tag{4.17}$$

Consider $\mathbf{g}_t = \mathbf{g}(w_t)$

$$\mathbf{g}(w_t) = \mathbf{g}(w) + \int_0^1 r^2 f''(w + z(w_t - w)) dz (w_t - w)$$

$$= \int_0^1 r^2 f''(w + z(w_t - w)) dz \Delta_t \quad (\text{as } \mathbf{g}(w) = 0);$$

Now we bound the following

$$\begin{aligned}
 k\mathbf{H}_t\Delta_t - \mathbf{g}(w_t)k &\leq k\Delta_tk \int_{-1}^1 [r^2 f(w_t) - r^2 f(w_t + z(w_t - w))] dz \\
 &\leq k\Delta_tk \int_{-1}^1 [r^2 f(w_t) - r^2 f(w_t + z(w_t - w))] dz \quad (\text{By Jensen}) \\
 &\leq k\Delta_tk \int_{-1}^1 (1 - |z|)L |w_t - w| dz \quad (\text{by } L\text{-Lipschitz assumption}) \\
 &= \frac{L}{2} k\Delta_tk^2 :
 \end{aligned}$$

Plugging it into (4.17) we have

$$\begin{aligned}
 \Delta_{t+1}^{\geq} \mathbf{H}_t \Delta_{t+1} &\leq 2\Delta_{t+1}^{\geq} (\mathbf{H}_t \Delta_t - \mathbf{g}_t) + \frac{2}{(1 - \frac{2}{\mu})} \Delta_t^{\geq} \mathbf{H}_t \Delta_t + 2^2 \\
 &\leq 2k\Delta_{t+1}k k\mathbf{H}_t \Delta_t - \mathbf{g}_t k + \frac{2}{(1 - \frac{2}{\mu})} \Delta_t^{\geq} \mathbf{H}_t \Delta_t + 2^2 \\
 &\leq Lk\Delta_{t+1}k k\Delta_tk^2 + \frac{2}{(1 - \frac{2}{\mu})} \Delta_t^{\geq} \mathbf{H}_t \Delta_t + 2^2 :
 \end{aligned}$$

□

Proof of Theorem 6

Proof. From the Lemma 15 with probability 1

$$\begin{aligned}
 \Delta_{t+1}^{\geq} \mathbf{H}_t \Delta_{t+1} &\leq Lk\Delta_{t+1}k k\Delta_tk^2 + \frac{2}{(1 - \frac{2}{\mu})} \Delta_t^{\geq} \mathbf{H}_t \Delta_t + 2^2 \\
 &\leq Lk\Delta_{t+1}k k\Delta_tk^2 + (\frac{2}{1 - \frac{2}{\mu}} \max(\mathbf{H}_t)) k\Delta_tk^2 + 2^2 :
 \end{aligned}$$

So we have,

$$k\Delta_{t+1}k \leq \max \left\{ \frac{L}{\min(\mathbf{H}_t)} (\frac{2}{1 - \frac{2}{\mu}}) k\Delta_tk; \frac{L}{\min(\mathbf{H}_t)} k\Delta_tk^2 + \frac{2}{\min(\mathbf{H}_t)} \right\} :$$

□

4.9 Analysis of Section 4.4

In this section we provide the theoretical analysis of the Byzantine robust method explained in Section 4.4 and prove the statistical guarantee. In any iteration t the following holds

$$\begin{aligned}
 |U_t| &= |U_t \setminus M_t| + |U_t \setminus B_t| \\
 |M_t| &= |U_t \setminus M_t| + |M_t \setminus T_t| :
 \end{aligned}$$

Combining both we have

$$jU_t j = jM_t j - j(M_t \setminus T_t)j + j(U_t \setminus B_t)j;$$

Lemma 16. Let $\{S_i\}_{i=1}^m \subset \mathbb{R}^{n \times s}$ be sketching matrices based on Lemma 11. Let t be defined in (4.10) and $\hat{\mathbf{p}}_t$ be defined in Algorithm 4. It holds that

$$\min_{\mathbf{p}} t(\mathbf{p}) = t(\hat{\mathbf{p}}_t) + \frac{1}{2} \text{byz} + (1 - \frac{1}{2} \text{byz}) \|\mathbf{p}\|^2;$$

where byz and $\frac{1}{2} \text{byz}$ is defined in (4.5) and (4.6) respectively.

Proof. In the following analysis we omit the subscript 't'. From the definition of the quadratic function (4.10) we know that

$$t(\hat{\mathbf{p}}) - t(\mathbf{p}) = \frac{1}{2} k\mathbf{H}^{\frac{1}{2}} (\hat{\mathbf{p}} - \mathbf{p})^2 k^2.$$

Now we consider

$$\begin{aligned} \frac{1}{2} k\mathbf{H}^{\frac{1}{2}} (\hat{\mathbf{p}} - \mathbf{p})^2 k^2 &= \frac{1}{2} k\mathbf{H}^{\frac{1}{2}} \left(\frac{1}{jU_j} \sum_{i \in M} \hat{\mathbf{p}}_i - \mathbf{p} \right)^2 k^2 \\ &= \frac{1}{2} k\mathbf{H}^{\frac{1}{2}} \frac{1}{jU_j} \left(\sum_{i \in M} (\hat{\mathbf{p}}_i - \mathbf{p}) \right)^2 k^2 + \frac{1}{2} k\mathbf{H}^{\frac{1}{2}} \frac{1}{jU_j} \left(\sum_{i \in (U \setminus B)} (\hat{\mathbf{p}}_i - \mathbf{p}) \right)^2 k^2 \\ &= \underbrace{\frac{1}{2} k\mathbf{H}^{\frac{1}{2}} \frac{1}{jU_j} \left(\sum_{i \in M} (\hat{\mathbf{p}}_i - \mathbf{p}) \right)^2 k^2}_{\text{Term1}} + \underbrace{\frac{1}{2} k\mathbf{H}^{\frac{1}{2}} \frac{1}{jU_j} \left(\sum_{i \in (M \setminus T)} (\hat{\mathbf{p}}_i - \mathbf{p}) \right)^2 k^2}_{\text{Term2}} \\ &\quad + \underbrace{\frac{1}{2} k\mathbf{H}^{\frac{1}{2}} \frac{1}{jU_j} \left(\sum_{i \in (U \setminus B)} (\hat{\mathbf{p}}_i - \mathbf{p}) \right)^2 k^2}_{\text{Term3}} \end{aligned}$$

Now we bound each term separately and use the result of the Lemma 14 to bound each term.

$$\begin{aligned} \text{Term1} &= k\mathbf{H}^{\frac{1}{2}} \frac{1}{jU_j} \left(\sum_{i \in M} (\hat{\mathbf{p}}_i - \mathbf{p}) \right)^2 k^2 \\ &= \left(\frac{1}{1} \right)^2 k\mathbf{H}^{\frac{1}{2}} \frac{1}{jM_j} \left(\sum_{i \in M} (\hat{\mathbf{p}}_i - \mathbf{p}) \right)^2 k^2 \\ &= \left(\frac{1}{1} \right)^2 \left[\frac{1}{m} + \frac{1}{M} k\mathbf{H}^{\frac{1}{2}} \mathbf{p}^2 k^2 \right]; \end{aligned}$$

where $\frac{1}{M} = \left(\frac{1}{jM_j} + \frac{1}{1} \right) = \left(\frac{1}{(1)_m} + \frac{1}{1} \right)$.

Similarly the Term 2 can be bonded as it is a bound on good machines

$$\begin{aligned}
 Term2 &= 2k\mathbf{H}^{\frac{1}{2}} \frac{1}{jUj} \times_{i2(M \setminus T)} (\hat{\mathbf{p}}_i - \mathbf{p})^2 k^2 \\
 &= 2\left(\frac{1}{1}\right)^2 k\mathbf{H}^{\frac{1}{2}} \frac{1}{jM \setminus Tj} \times_{i2(M \setminus T)} (\hat{\mathbf{p}}_i - \mathbf{p})^2 k^2 \\
 &= 2\left(\frac{1}{1}\right)^2 [\quad^2 + \quad_{M \setminus T}^2 k\mathbf{H}^{\frac{1}{2}} \mathbf{p} \quad^2];
 \end{aligned}$$

where $\quad_{M \setminus T} = \left(\frac{\rho}{jM \setminus Tj} + \frac{1}{1}\right)$ and $\quad_{(1)M} = \left(\frac{\rho}{(1)M} + \frac{1}{1}\right)$.

For the Term 3 we know that $\quad > \quad$ so all the untrimmed worker norm is bounded by a good machine as at least one good machine gets trimmed.

$$\begin{aligned}
 Term3 &= 2k\mathbf{H}^{\frac{1}{2}} \frac{1}{jUj} \times_{i2(U \setminus B)} (\hat{\mathbf{p}}_i - \mathbf{p})^2 k^2 \\
 &= 2 \max(\mathbf{H}) \left(\frac{jU \setminus Bj}{jUj}\right)^2 k \frac{1}{jU \setminus Bj} \times_{i2(U \setminus B)} (\hat{\mathbf{p}}_i - \mathbf{p})^2 k^2 \\
 &= 2 \max(\mathbf{H}) \left(\frac{jU \setminus Bj}{jUj}\right)^2 \frac{1}{jU \setminus Bj} \times_{i2(U \setminus B)} k(\hat{\mathbf{p}}_i - \mathbf{p})^2 k^2 \\
 &= 4 \max(\mathbf{H}) \left(\frac{jU \setminus Bj}{jUj}\right)^2 \frac{1}{jU \setminus Bj} \times_{i2(U \setminus B)} (k\hat{\mathbf{p}}_i k^2 + k\mathbf{p} \quad^2) \\
 &= 4 \max(\mathbf{H}) \left(\frac{jU \setminus Bj}{jUj}\right)^2 \max_{i2M} (k\hat{\mathbf{p}}_i k^2 + k\mathbf{p} \quad^2) \\
 &= 4 \max(\mathbf{H}) \left(\frac{jU \setminus Bj}{jUj}\right)^2 \max_{i2M} (k\hat{\mathbf{p}}_i - \mathbf{p} \quad^2 + 2k\mathbf{p} \quad^2) \\
 &= 4 \left(\frac{jU \setminus Bj}{jUj}\right)^2 \max_{i2M} (k\mathbf{H}^{\frac{1}{2}} (\hat{\mathbf{p}}_i - \mathbf{p}) \quad^2 + 2k\mathbf{H}^{\frac{1}{2}} \mathbf{p} \quad^2) \\
 &= 4 \left(\frac{jU \setminus Bj}{jUj}\right)^2 (\quad^2 + (2 + \frac{1}{1}) k\mathbf{H}^{\frac{1}{2}} \mathbf{p} \quad^2) \\
 &= 4 \left(\frac{1}{1}\right)^2 (\quad^2 + (2 + \frac{1}{1}) k\mathbf{H}^{\frac{1}{2}} \mathbf{p} \quad^2);
 \end{aligned}$$

where $\quad_1 = \left(\frac{1}{1} + \frac{1}{1}\right) = \frac{1}{1}$ and $\quad = \frac{\max(\mathbf{H})}{\min(\mathbf{H})}$.

Combining all the bounds on Term1, Term2 and Term3 we have

$$\frac{1}{2} k\mathbf{H}^{\frac{1}{2}} (\hat{\mathbf{p}} - \mathbf{p})^2 k^2 + \quad_{byz}^2 + \quad_{byz}^2 k\mathbf{H}^{\frac{1}{2}} \mathbf{p} \quad^2 k^2;$$

where

$$\begin{aligned} \frac{2}{\text{byz}} &= 3 \frac{1}{1} + 4 \frac{1}{1}^2; \\ \frac{2}{\text{byz}} &= 2 \frac{1}{1}^2 + \frac{1}{1}^2 + 4 \frac{1}{1} (2 + \frac{2}{1}): \end{aligned}$$

Finally we have

$$\begin{aligned} & \left(\hat{\mathbf{p}} \right) \left(\mathbf{p} \right) \frac{2}{\text{byz}} \frac{2}{\text{byz}} \left(\mathbf{p} \right) \\ & \left. \right) \left(\hat{\mathbf{p}} \right) \frac{2}{\text{byz}} + \left(1 \frac{2}{\text{byz}} \right) \left(\mathbf{p} \right): \end{aligned}$$

□

Lemma 17. Let $\frac{2}{\text{byz}} \in (0; 1)$; $\frac{2}{\text{byz}}$ be any fixed parameter. And $\hat{\mathbf{p}}_t$ satisfies $t(\hat{\mathbf{p}}_t) \leq \frac{2}{\text{byz}} + (1 - \frac{2}{\text{byz}}) \min_{\mathbf{p}} t(\mathbf{p})$. Under the Assumption 15 (Hessian L-Lipschitz) and $\Delta_t = w_t \mathbf{w}$ satisfies

$$\Delta_{t+1}^T \mathbf{H}_t \Delta_{t+1} \leq L \|\Delta_{t+1}\| \|\Delta_t\|^2 + \frac{2}{1 - \frac{2}{\text{byz}}} \Delta_t^T \mathbf{H}_t \Delta_t + 2 \frac{2}{\text{byz}};$$

Proof. We choose $\frac{2}{\text{byz}}$ and $\frac{2}{\text{byz}}$ from the Lemma 16 and follow the proof of Lemma 15 to obtain the desired bound. □

Proof of Theorem 7

Proof. We get the desired bound by developing from the result of the Lemma 17 and following the proof of Theorem 6 □

4.10 Auxiliary Lemmas

Lemma 18. Let $f \mathbf{S}_i g_{i=1}^m \in \mathbb{R}^{n \times s}$ be sketching matrices satisfies the Lemma 11. Let t be defined in (4.10) and $\hat{\mathbf{p}}_t$ be defined in Algorithm 4. It holds that

$$\min_{\mathbf{p}} t(\mathbf{p}) \leq t(\hat{\mathbf{p}}_t) \leq \frac{2}{\text{byz}} + \left(1 - \frac{2}{\text{byz}} \right) \left(\mathbf{p} \right)$$

where

Proof. In the following analysis we omit the subscript 't'. From the definition of the quadratic function (4.10) we know that

$$\begin{aligned} (Q(\hat{\mathbf{p}})) \left(\mathbf{p} \right) &= \frac{1}{2} \|\mathbf{H}^{\frac{1}{2}}(Q(\hat{\mathbf{p}}) - \mathbf{p})\|^2 \\ &= \underbrace{\|\mathbf{H}^{\frac{1}{2}}(Q(\hat{\mathbf{p}}) - \hat{\mathbf{p}})\|^2}_{\text{Term1}} + \underbrace{\|\mathbf{H}^{\frac{1}{2}}(\hat{\mathbf{p}} - \mathbf{p})\|^2}_{\text{Term2}} \end{aligned}$$

First we bound the Term 1.

$$\begin{aligned} k\mathbf{H}^{\frac{1}{2}}(Q(\hat{\mathbf{p}}) - \mathbf{p})k^2 &= \frac{\max(\mathbf{H}_t)(1 - \eta)}{\max(\mathbf{H}_t)(1 - \eta)} [k\hat{\mathbf{p}} - \mathbf{p}k^2 + k\mathbf{p}k^2] \\ &= \frac{\max(\mathbf{H}_t)}{\min(\mathbf{H}_t)} (1 - \eta) [k\mathbf{H}^{\frac{1}{2}}(\hat{\mathbf{p}} - \mathbf{p})k^2 + k\mathbf{H}^{\frac{1}{2}}\mathbf{p}k^2] \\ &= (1 - \eta) [k\mathbf{H}^{\frac{1}{2}}(\hat{\mathbf{p}} - \mathbf{p})k^2 + k\mathbf{H}^{\frac{1}{2}}\mathbf{p}k^2] \end{aligned}$$

Now plugging back the bound of Term 1, we get

$$\begin{aligned} (Q(\hat{\mathbf{p}}) - \mathbf{p}) &= (1 - \eta) [k\mathbf{H}^{\frac{1}{2}}(\hat{\mathbf{p}} - \mathbf{p})k^2 + k\mathbf{H}^{\frac{1}{2}}\mathbf{p}k^2] + k\mathbf{H}^{\frac{1}{2}}(\hat{\mathbf{p}} - \mathbf{p})k^2 \\ &= (1 + (1 - \eta)) [k\mathbf{H}^{\frac{1}{2}}(\hat{\mathbf{p}} - \mathbf{p})k^2] + (1 - \eta) k\mathbf{H}^{\frac{1}{2}}\mathbf{p}k^2 \end{aligned}$$

Now we use Lemma 14 to bound the term $k\mathbf{H}^{\frac{1}{2}}(\hat{\mathbf{p}} - \mathbf{p})k^2$ and we get,

$$\begin{aligned} (Q(\hat{\mathbf{p}}) - \mathbf{p}) &= (1 + (1 - \eta)) (k^2 + k\mathbf{H}^{\frac{1}{2}}\mathbf{p}k^2) + (1 - \eta) k\mathbf{H}^{\frac{1}{2}}\mathbf{p}k^2 \\ &= (1 + (1 - \eta)) k^2 + [(1 + (1 - \eta)) k^2 + (1 - \eta)] k\mathbf{H}^{\frac{1}{2}}\mathbf{p}k^2 \\ &= \frac{2}{b} + (1 - \frac{2}{b}) (\mathbf{p}) \end{aligned}$$

where

$$\begin{aligned} \frac{2}{b} &= (1 + (1 - \eta)) k^2 \\ \frac{2}{b} &= (1 + (1 - \eta)) k^2 + (1 - \eta) \end{aligned}$$

□

Lemma 19. $\Delta_t = w_t$ satisfies

$$\Delta_{t+1}^T \mathbf{H}_t \Delta_{t+1} \leq L k\Delta_{t+1}k k\Delta_tk^2 + \frac{2}{1 - \frac{2}{b}} \Delta_t^T \mathbf{H}_t \Delta_t + 2 \frac{2}{b}$$

Theorem 9. Let $\eta \geq 1; \frac{\eta}{d}$ be the coherence of \mathbf{A}_t and m be the number of partitions. For some $\eta \in (0; 1)$, with probability 1

$$k\Delta_{t+1}k \leq \frac{S}{\min(\mathbf{H}_t)} \left(\frac{\max(\mathbf{H}_t)}{\min(\mathbf{H}_t)} \left(\frac{2}{1 - \frac{2}{b}} k\Delta_tk; \frac{L}{\min(\mathbf{H}_t)} k\Delta_tk^2 \right) + \frac{b}{\min(\mathbf{H}_t)} \right)$$

4.11 Analysis of Section 4.5

First we prove the following lemma that will be useful in our subsequent calculations. Consider that $Q(\hat{\mathbf{p}}) = \frac{1}{jB_j} \sum_{i \in B} Q(\hat{\mathbf{p}}_i)$. And also we use the following notation $\beta = \left(\frac{\rho}{jB_j} + \frac{1}{\gamma}\right)$, $\alpha = \frac{\max(\mathbf{A}^T \mathbf{A})}{\max(\mathbf{A}^T \mathbf{A}) + n} < 1$.

Lemma 20. *If $Q(\hat{\mathbf{p}}_i)$ is the local update direction and \mathbf{p}^* is the optimal solution to the quadratic function then*

$$\mathbf{H}^1(Q(\hat{\mathbf{p}}) - \mathbf{p}^*)^2 \leq (1 + \alpha) \beta^2 + \left(\frac{2}{B} + (1 - \alpha)\beta\right) \mathbf{H}^1 \mathbf{p}^{*2};$$

where \mathbf{H} is the exact Hessian and

$$\beta = \frac{\rho}{(1 + \alpha)\gamma};$$

$$\alpha_{comp;B} = \left(\frac{2}{B} + (1 - \alpha)\beta\right);$$

is defined in equation (4.4) and

Proof.

$$\begin{aligned} \mathbf{H}^1(Q(\hat{\mathbf{p}}) - \mathbf{p}^*)^2 &= \mathbf{H}^1(Q(\hat{\mathbf{p}}) - \hat{\mathbf{p}} + \hat{\mathbf{p}} - \mathbf{p}^*)^2 \\ &= \underbrace{\mathbf{H}^1(Q(\hat{\mathbf{p}}) - \hat{\mathbf{p}})^2}_{Term1} + \underbrace{\mathbf{H}^1(\hat{\mathbf{p}} - \mathbf{p}^*)^2}_{Term2} \end{aligned} \tag{4.18}$$

Following the proof of Lemma 14 we get

$$\mathbf{H}^1(\hat{\mathbf{p}}_i - \mathbf{p}^*)^2 \leq \alpha + \beta \mathbf{H}^1 \mathbf{p}^{*2}; \tag{4.19}$$

where α is as defined in (4.4). Now we consider the term

$$\begin{aligned} \mathbf{H}^1(Q(\hat{\mathbf{p}}) - \hat{\mathbf{p}})^2 &\leq \max(\mathbf{H})(1 - \alpha) k \hat{\mathbf{p}}_i k^2 \\ &\leq \max(\mathbf{H})(1 - \alpha) k \hat{\mathbf{p}}_i - \mathbf{p}^* k^2 + k \mathbf{p}^* k^2 \\ &\leq \frac{\max}{\min}(1 - \alpha) \mathbf{H}^1(\hat{\mathbf{p}}_i - \mathbf{p}^*)^2 + \mathbf{H}^1 \mathbf{p}^{*2} \\ &= (1 - \alpha) \mathbf{H}^1(\hat{\mathbf{p}}_i - \mathbf{p}^*)^2 + \mathbf{H}^1 \mathbf{p}^{*2} \\ &\leq (1 - \alpha) \beta^2 + (1 + \alpha) \mathbf{H}^1 \mathbf{p}^{*2} \quad \text{Using (4.19):} \end{aligned}$$

Now we use the above calculation and bound Term1

$$\mathbf{H}^{\frac{1}{2}}(Q(\hat{\mathbf{p}}) - \hat{\mathbf{p}})^2 \leq \frac{1}{jB_j} \sum_{i \geq 2B} \mathbf{H}^{\frac{1}{2}}(Q(\hat{\mathbf{p}}_i) - \hat{\mathbf{p}}_i)^2 + (1 + \frac{2}{j}) \mathbf{H}^{\frac{1}{2}} \mathbf{p}^2 : \quad (4.20)$$

We can bound the Term2 directly using the proof of Lemma 14

$$\mathbf{H}^{\frac{1}{2}}(\hat{\mathbf{p}} - \mathbf{p})^2 \leq \frac{2}{B} \mathbf{H}^{\frac{1}{2}} \mathbf{p}^2 : \quad (4.21)$$

Now we use (4.20) and (4.21) and plug them in (4.18)

$$\mathbf{H}^{\frac{1}{2}}(Q(\hat{\mathbf{p}}) - \mathbf{p})^2 \leq (1 + (1 + \frac{2}{j}))^2 + (\frac{2}{B} + (1 + \frac{2}{j}))((1 + \frac{2}{j})) \mathbf{H}^{\frac{1}{2}} \mathbf{p}^2 :$$

Now we define

$$\begin{aligned} \alpha_1 &= \frac{1}{(1 + (1 + \frac{2}{j}))} \\ \alpha_{comp;B}^2 &= (\frac{2}{B} + (1 + \frac{2}{j}))((1 + \frac{2}{j})) : \end{aligned}$$

□

Now we have the robust update in iteration t to be $Q(\hat{\mathbf{p}}) = \frac{1}{jU_j} \sum_{i \geq 2U_t} Q(\hat{\mathbf{p}}_{i;t})$.

Lemma 21. Let $\{S_i, g_{i=1}^m\} \subseteq \mathbb{R}^{n \times s}$ be sketching matrices based on Lemma 11. Let α_t be defined in (4.10) and $Q(\hat{\mathbf{p}}_t)$ be the update with Q being α_t -approximate compressor. It holds that

$$\min_{\mathbf{p}} \alpha_t(\mathbf{p}) \leq \alpha_t(Q(\hat{\mathbf{p}}_t)) \leq \alpha_{comp;byz}^2 + (1 + \alpha_{comp;byz}^2) \alpha_t(\mathbf{p}) ;$$

where $\alpha_{comp;byz}$ and $\alpha_{comp;byz}^2$ is as defined in (4.7) and (4.8) respectively.

Proof. In the following analysis we omit the subscript ' t '. From the definition of the quadratic function (4.10) we know that

$$\alpha(Q(\hat{\mathbf{p}}))(\mathbf{p}) = \frac{1}{2} k \mathbf{H}^{\frac{1}{2}}(Q(\hat{\mathbf{p}}) - \mathbf{p})^2 :$$

Now we consider

$$\begin{aligned}
 \frac{1}{2}k\mathbf{H}^{\frac{1}{2}}(Q(\hat{\mathbf{p}}) - \mathbf{p})^2 &= \frac{1}{2}k\mathbf{H}^{\frac{1}{2}}\left(\frac{1}{jU_j} \sum_{i \in 2U} Q(\hat{\mathbf{p}}_i) - \mathbf{p}\right)^2 \\
 &= \frac{1}{2}k\mathbf{H}^{\frac{1}{2}}\frac{1}{jU_j} \left(\sum_{i \in 2M} (Q(\hat{\mathbf{p}}_i) - \mathbf{p}) + \sum_{i \in 2(M \setminus T)} (Q(\hat{\mathbf{p}}_i) - \mathbf{p}) \right. \\
 &\quad \left. + \sum_{i \in 2(U \setminus B)} (Q(\hat{\mathbf{p}}_i) - \mathbf{p}) \right)^2 \\
 &= \underbrace{\frac{1}{2}k\mathbf{H}^{\frac{1}{2}}\frac{1}{jU_j} \sum_{i \in 2M} (Q(\hat{\mathbf{p}}_i) - \mathbf{p})^2}_{Term1} + \underbrace{2k\mathbf{H}^{\frac{1}{2}}\frac{1}{jU_j} \sum_{i \in 2(M \setminus T)} (Q(\hat{\mathbf{p}}_i) - \mathbf{p})^2}_{Term2} \\
 &\quad + \underbrace{2k\mathbf{H}^{\frac{1}{2}}\frac{1}{jU_j} \sum_{i \in 2(U \setminus B)} (Q(\hat{\mathbf{p}}_i) - \mathbf{p})^2}_{Term3}
 \end{aligned}$$

Now we bound each term separately and use the Lemma 20

$$\begin{aligned}
 Term1 &= k\mathbf{H}^{\frac{1}{2}}\frac{1}{jU_j} \sum_{i \in 2M} (Q(\hat{\mathbf{p}}_i) - \mathbf{p})^2 \\
 &= \left(\frac{1}{j}\right)^2 k\mathbf{H}^{\frac{1}{2}}\frac{1}{jM_j} \sum_{i \in 2M} (Q(\hat{\mathbf{p}}_i) - \mathbf{p})^2 \\
 &= \left(\frac{1}{j}\right)^2 \left[\frac{2}{1} + \frac{2}{comp;M} k\mathbf{H}^{\frac{1}{2}}\mathbf{p}^2 \right];
 \end{aligned}$$

where $\frac{2}{comp;M} = \left(\frac{2}{M} + (1 - \frac{1}{M})\right)\left(1 + \frac{1}{j}\right)$. Similarly the Term 2 can be bounded as it is a bound on good machines

$$\begin{aligned}
 Term2 &= 2k\mathbf{H}^{\frac{1}{2}}\frac{1}{jU_j} \sum_{i \in 2(M \setminus T)} (Q(\hat{\mathbf{p}}_i) - \mathbf{p})^2 \\
 &= 2\left(\frac{1}{j}\right)^2 k\mathbf{H}^{\frac{1}{2}}\frac{1}{jM \setminus T_j} \sum_{i \in 2(M \setminus T)} (Q(\hat{\mathbf{p}}_i) - \mathbf{p})^2 \\
 &= 2\left(\frac{1}{j}\right)^2 \left[\frac{2}{1} + \frac{2}{comp;M \setminus T} k\mathbf{H}^{\frac{1}{2}}\mathbf{p}^2 \right];
 \end{aligned}$$

For the Term 3 we know that $\frac{1}{j} > \frac{1}{M}$ so all the untrimmed worker norm is bounded by a good

machine as at least one good machine gets trimmed.

$$\begin{aligned}
 Term3 &= 2k\mathbf{H}^{\frac{1}{2}} \frac{1}{jUj} \times_{i2(U \setminus B)} (Q(\hat{\mathbf{p}}_i) - \mathbf{p})^2 k^2 \\
 &\leq 2 \max(\mathbf{H}) \left(\frac{jU \setminus Bj}{jUj}\right)^2 k \frac{1}{jU \setminus Bj} \times_{i2(U \setminus B)} (Q(\hat{\mathbf{p}}_i) - \mathbf{p})^2 k^2 \\
 &\leq 2 \max(\mathbf{H}) \left(\frac{jU \setminus Bj}{jUj}\right)^2 \frac{1}{jU \setminus Bj} \times_{i2(U \setminus B)} k(Q(\hat{\mathbf{p}}_i) - \mathbf{p})^2 k^2 \\
 &\leq 4 \max(\mathbf{H}) \left(\frac{jU \setminus Bj}{jUj}\right)^2 \frac{1}{jU \setminus Bj} \times_{i2(U \setminus B)} (kQ(\hat{\mathbf{p}}_i)k^2 + k\mathbf{p}k^2) \\
 &\leq 4 \max(\mathbf{H}) \left(\frac{jU \setminus Bj}{jUj}\right)^2 \max_{i2M} (kQ(\hat{\mathbf{p}}_i)k^2 + k\mathbf{p}k^2) \\
 &\leq 4 \max(\mathbf{H}) \left(\frac{jU \setminus Bj}{jUj}\right)^2 \max_{i2M} (kQ(\hat{\mathbf{p}}_i) - \mathbf{p}k^2 + 2k\mathbf{p}k^2) \\
 &\leq 4 \left(\frac{jU \setminus Bj}{jUj}\right)^2 \max_{i2M} (k\mathbf{H}^{\frac{1}{2}}(Q(\hat{\mathbf{p}}_i) - \mathbf{p})k^2 + 2k\mathbf{H}^{\frac{1}{2}}\mathbf{p}k^2) \\
 &\leq 4 \left(\frac{jU \setminus Bj}{jUj}\right)^2 \left(\frac{2}{1} + (2 + \frac{2}{1})k\mathbf{H}^{\frac{1}{2}}\mathbf{p}k^2\right) \\
 &\leq 4 \left(\frac{1}{1}\right)^2 \left(\frac{2}{1} + (2 + \frac{2}{1})k\mathbf{H}^{\frac{1}{2}}\mathbf{p}k^2\right):
 \end{aligned}$$

Combining all the bounds on Term1 , Term2 and Term3 we have

$$\frac{1}{2}k\mathbf{H}^{\frac{1}{2}}(\hat{\mathbf{p}} - \mathbf{p})^2 \leq \frac{2}{byz} + \frac{2}{byz}k\mathbf{H}^{\frac{1}{2}}\mathbf{p}k^2;$$

where

$$\begin{aligned}
 \frac{2}{comp;byz} &= 3 \frac{1}{1} + 4 \frac{1}{1} \\
 \frac{2}{comp;byz} &= 2 \frac{1}{1} + \frac{2}{comp;M \setminus T} + \frac{1}{1} + \frac{2}{comp;M} + 4 \frac{1}{1} (2 + \frac{2}{comp;1}):
 \end{aligned}$$

Finally we have

$$\begin{aligned}
 &(\hat{\mathbf{p}} - \mathbf{p})^2 \leq \frac{2}{comp;byz} + \frac{2}{comp;byz} k\mathbf{H}^{\frac{1}{2}}\mathbf{p}k^2 \\
 &+ \frac{2}{comp;byz} + (1 + \frac{2}{comp;byz}) k\mathbf{H}^{\frac{1}{2}}\mathbf{p}k^2:
 \end{aligned}$$

□

Lemma 22. Let $\alpha \in (0, 1)$; β be any fixed parameter. And $Q(\hat{\mathbf{p}}_t)$ satisfies $\|Q(\hat{\mathbf{p}}_t) - \min_{\mathbf{p}} Q(\mathbf{p})\| \leq \beta + (1 - \beta) \min_{\mathbf{p}} Q(\mathbf{p})$. Under the Assumption 15 (Hessian L -Lipschitz) and $\Delta_t = w_t - w$ satisfies

$$\Delta_{t+1}^T \mathbf{H}_t \Delta_{t+1} \leq L \|\Delta_{t+1}\| \|\Delta_t\|^2 + \frac{\beta^2}{1 - \beta} \Delta_t^T \mathbf{H}_t \Delta_t + 2\beta^2.$$

Proof. We choose $\alpha = \beta$ and $\gamma = \beta$ from the Lemma 21 and follow the proof of Lemma 15 to obtain the desired bound. \square

Proof of Theorem 8

Proof. We get the desired bound by developing from the result of the Lemma 22 and following the proof of Theorem 6 \square

Additional Experiment

In addition to the experimental results in Section 4.6, we provide some more experiments supporting the robustness of the COMRADE in two different types of attacks : 1. ‘Gaussian attack’: where the Byzantine workers add Gaussian Noise ($\mathcal{N}(\cdot; \sigma^2)$) to the update and 2. ‘random label attack’: where the Byzantine worker machines learns based on random labels instead of proper labels.

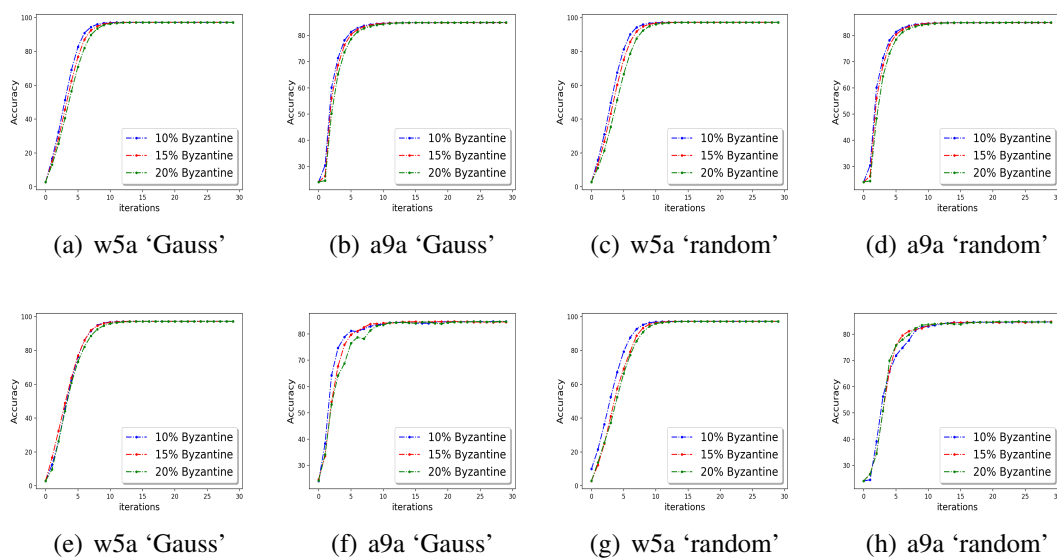


Figure 4.3: (First row) Accuracy of COMRADE with 10%;15%;20% Byzantine workers with 'Gaussian' attack for (a). w5a (b). a9a and 'random label' attack for (c). w5a (d).a9a. (Second row) Accuracy of COMRADE with -approximate compressor (Section 4.5) with 10%;15%;20% Byzantine workers with 'Gaussian' attack for (a). w5a (b). a9a and 'random label' attack for (c). w5a (d).a9a.

Chapter 5

Escaping Saddle Points in FL: Distributed Cubic Regularized Newton Method

We study the problem of optimizing a non-convex loss function (with saddle points) in a Federated Learning framework in the presence of Byzantine machines. Our proposed algorithm is a variant of the celebrated cubic-regularized Newton method of Nesterov and Polyak [43], which avoids saddle points efficiently and converges to local minima. Furthermore, our algorithm resists the presence of Byzantine machines, which may create *fake local minima* near the saddle points of the loss function, also known as saddle-point attack. We robustify the cubic-regularized Newton algorithm such that it avoids the saddle points and the fake local minimas efficiently. Furthermore, being a second order algorithm, the iteration complexity is much lower than its first order counterparts, and thus our algorithm communicates little with the parameter server. We obtain theoretical guarantees for our proposed scheme under several settings including approximate (sub-sampled) gradients and Hessians. Moreover, we validate our theoretical findings with experiments using standard datasets and several types of Byzantine attacks.

5.1 Introduction

In this chapter we propose provable and efficient algorithms for distributed learning that are communication efficient and Byzantine resilient. In particular, we focus on optimizing a non-convex loss function $f(\cdot)$ in a distributed optimization framework. We have m worker machines, out of which β fraction may behave in a Byzantine fashion, where $\beta < \frac{1}{2}$. Optimizing a loss function in a distributed setup has gained a lot of attention in recent years [17, 19, 30, 33]. However, most of these approaches either work when $f(\cdot)$ is convex, or provide weak guarantees in the non-convex case (ex: zero gradient points, maybe a saddle point).

On the other hand, in order to fit complex machine learning models, one often requires to find local minima of a non-convex loss $f(\cdot)$, instead of critical points only, which may include several saddle points. Training deep neural networks and other high-capacity learning architectures [35, 36] are some of the examples where finding local minima is crucial. [36, 37] shows that the stationary

points of these problems are in fact saddle points and far away from any local minimum, and hence designing efficient algorithm that escapes saddle points is of interest. Moreover, in [38, 39], it is argued that saddle points can lead to highly sub-optimal solutions in many problems of interest. This issue is amplified in high dimension as shown in [40], and becomes the main bottleneck in training deep neural nets.

Furthermore, a line of recent work [39, 41, 42], shows that for many non-convex problems, it is sufficient to find a local minimum. In fact, in many problems of interest, all local minima are global minima (e.g., dictionary learning [42], phase retrieval [39], matrix sensing and completion [36, 41], and some of neural nets [37]). Also, in [137], it is argued that for more general neural nets, the local minima are as good as global minima.

The issue of saddle point avoidance becomes non-trivial in the presence of Byzantine workers. Since we do not assume anything on the behavior of the Byzantine workers, it is certainly conceivable that by appropriately modifying their messages to the center, they can create *fake local minima* that are close to the saddle point of the loss function $f(\cdot)$, and these are far away from the true local minima of $f(\cdot)$. This is popularly known as the *saddle-point attack* (see [31]), and it can arbitrarily destroy the performance of any non-robust learning algorithm. Hence, our goal is to design an algorithm that escapes saddle points of $f(\cdot)$ in an efficient manner as well as resists the saddle-point attack simultaneously. The complexity of such an algorithm emerges from the the interplay between non-convexity of the loss function and the behavior of the Byzantine machines.

The problem of saddle point avoidance in the context of non-convex optimization has received considerable attention in the past few years. In the seminal paper of Jin et al. [46], a gradient descent based approach is proposed. By defining a certain *perturbation condition* and adding Gaussian noise to the iterates of gradient descent, the algorithm of [46] provably escapes the saddle points of the non-convex loss function. A few papers [138, 139] following the above use various modifications to obtain saddle point avoidance guarantees. However, these algorithms are non-robust. A Byzantine robust saddle point avoidance algorithm is proposed by Yin et al. [31], and probably is the closest to this work. In [31], the authors propose a repeated check-and-escape type of first order gradient descent based algorithm. First of all, being a first order algorithm, the convergence rate is quite slow (the rate for gradient decay is $1 = \frac{1}{T}$, where T is the number of iterations). Moreover, implementation-wise, the algorithm presented in [31] is computation heavy, and takes potentially many iterations between the center and the worker machines. Hence, this algorithm is not efficient in terms of the communication cost.

In this work, we consider a variation of the famous cubic-regularized Newton algorithm of Nesterov and Polyak [43]. It is theoretically proved in [43] that a cubic-regularized Newton method with proper choice of parameters like step size always outperforms the gradient based first order schemes (like [31]) in all situations under consideration. Indeed, in Theorem 10, we observe that the rate of gradient decay is $\frac{1}{T^{2-3}}$, which is strictly better than the first order gradient based methods. In Section 5.6, we experimentally show that our scheme outperforms that of [31], in terms of iteration complexity and hence communication cost. Also, our algorithm is easy to implement whereas a range of hyper-parameter choice and tuning makes the implementation of ByzantinePGD [31] difficult.

In [43–45], it is shown that cubic-regularized Newton can efficiently escape the saddle points

of a non-convex function. Assuming the loss function has a Lipschitz continuous Hessian (see Assumption 19), the cubic-regularized Newton optimizes an auxiliary function (detailed in Section 5.3), which is an upper second order approximation of the original loss function. It is shown in [43] that the cubic regularized term in the auxiliary function pushes the Hessian towards a positive semi-definite matrix.

A point w is said to satisfy the ρ -second order stationary condition of the loss function $f(\cdot)$ if,

$$\| \nabla f(w) \| \leq \rho \cdot \min(\lambda_2(\nabla^2 f(w)))$$

$\nabla f(w)$ denotes the gradient of the function and $\lambda_2(\nabla^2 f(w))$ denotes the minimum eigenvalue of the Hessian of the function. Hence, under the assumption (which is standard in the literature, see [31, 46]) that all saddle points are strict (i.e., $\lambda_2(\nabla^2 f(w_s)) < 0$ for any saddle point w_s), all second order stationary points (with $\rho = 0$) are local minima, and hence converging to a stationary point is equivalent to converging to a local minima.

We consider a distributed variant of the cubic regularized Newton algorithm. In this scheme, the center machine asks the workers to solve an auxiliary function and return the result. Note that the complexity of the problem is partially transferred to the worker machines. It is worth mentioning that in most distributed optimization paradigm, including Federated Learning, the workers possess sufficient compute power to handle this partial transfer of compute load, and in most cases, this is desirable [3]. The center machine aggregates the solution of the worker machines and takes a descent step. Note that, unlike gradient aggregation, the aggregation of the solutions of the local optimization problems is a highly non-linear operation. Hence, it is quite non-trivial to extend the centralized cubic regularized algorithm to a distributed one. The solution to the cubic regularization even lacks a closed form solution unlike the second order Hessian based update or the first order gradient based update. The analysis is carried out by leveraging the first order and second order stationary conditions of the auxiliary function solved in each worker machines.

In addition to this, we use a simple norm-based thresholding approach to robustify the distributed cubic-regularized Newton method. In [31], the authors use computation-heavy schemes like coordinate-wise median, trimmed mean and iterative filtering. In contrast to these approaches, our scheme is computationally efficient. Norm based thresholding is a standard trick for Byzantine resilience as featured in [140, 141]. However, since the local optimization problem lacks a closed form solution, using norm-based trimming is also technical challenging in this case. Handling the Byzantine worker machines becomes a bit more complicated as those stationary conditions of the good machines (non-Byzantine machine) do not hold for the Byzantine worker machines.

5.1.1 Our Contributions

1. We propose a novel distributed and robust cubic regularized Newton algorithm, that escapes saddle point efficiently. We prove that the algorithm convergence at a rate of $\frac{1}{T^{2-3\rho}}$, which is faster than the first order methods (which converge at $1 = \frac{1}{T}$ rate, see [31]). Hence, the number of iterations (and hence the communication cost) required to achieve a target accuracy is much fewer than the first order methods. A simple simulation in Section 5.6, shows that the algorithm of [31] requires 36x more steps than ours, showing a huge communication gain.

2. The computation complexity of our algorithm is also much less than the existing schemes [31]. Part of computation is deferred to the worker machines, which is desirable in paradigm like Federated Learning.
3. We use norm-based thresholding to resist Byzantine workers. In previous works, computation heavy techniques like coordinate-wise median, coordinate-wise trimmed mean and spectral filtering are used to resist Byzantine workers. In contrast, our norm based thresholding in computation friendly.
4. We work with inexact gradients and Hessians, which is quite common in distributed setup like Federated Learning.
5. In Section 5.6, we verify our theoretical findings via experiments. We use benchmark LIBSVM ([47]) datasets for logistic regression and non-convex robust regression and show convergence results for both non-Byzantine and several different Byzantine attacks.

5.2 Related Work

In the recent years, there are handful first order algorithms [142–144] that focus on the escaping saddle points and convergence to local minima. The critical algorithmic aspect is running gradient based algorithm and adding perturbation to the iterates when the gradient is small. ByzantinePGD [31], PGD [46], Neon+GD[138], Neon2+GD [139] are examples of such algorithms. For faster convergence rate, second order Hessian based algorithms are developed for saddle point avoidance. The work of Nesterov and Polyak [43] first proposes the cubic regularized Newton method and provides analysis for the second order stationary condition. An algorithm called Adaptive Regularization with Cubics (ARC) was developed by [145, 146] where cubic regularized Newton method with access to inexact Hessian was studied. The inexactness of Hessians for the ARC algorithm is adaptive over iterations. Cubic regularization with both the gradient and Hessian being inexact was studied in [147]. In [44], a cubic regularized Newton with sub-sampled Hessian and gradient was proposed, but for analysis, the batch size of the sample changes in adaptive manner to provide guarantees for the inexactness of the Hessian and gradient. In this work, we also take a similar approach as [44], but we relax the adaptive nature of the sample size. Momentum based cubic regularized algorithm was studied in [45]. A variance reduced cubic regularized algorithm was proposed in [148, 149]. In terms of solving the cubic sub-problem, [150] proposes a gradient based algorithm and [151] provides a Hessian-vector product technique.

Furthermore, as mentioned in Chapters 3,4, the Byzantine resilience algorithms in FL have received significant interest (see [19, 22, 23, 30, 140]. Also, in Chapters 3 and 4, a norm based thresholding approach for Byzantine resilience for distributed first and second order algorithms respectively was proposed. However, none of these works provide saddle point avoidance guarantees (they only show convergence to a small gradient point). The work [31] is the only one that provides second order guarantee (Hessian positive semi-definite) under Byzantine attack, and we compare our algorithm with that of [31].

5.3 Problem Formulation

In this section, we formally set up the problem. Similar to Chapter 4, we minimize a loss function additive over losses of worker machines. The loss function takes the following form:

$$f(w) = \frac{1}{m} \sum_{i=1}^m f_i(w); \tag{5.1}$$

where the function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is twice differentiable and non-convex. Continuing with the notation from the previous chapters, we consider distributed optimization framework with m worker machines and one center machine where the worker machines communicate to the center machine. Each worker machine is associated with a local loss function f_i minimized over i.i.d. data points drawn from some unknown distribution. In addition to that, we also consider the case where fraction of the worker machines are Byzantine for some $\epsilon < \frac{1}{2}$. The Byzantine machines can send arbitrary updates to the central machine which can disrupt the learning. Furthermore, the Byzantine machines can collude with each other, create *fake local minima* or attack maliciously by gaining information about the learning algorithm and other workers.

In the rest of the chapter, the norm $\| \cdot \|_k$ will refer to ℓ_2 norm or spectral norm when the argument is a vector or a matrix respectively.

Algorithm 5: Byzantine Robust Distributed Cubic Regularized Newton Algorithm

- 1: **Input:** Step size κ , parameter $\epsilon > 0; \mu > 0; M > 0$
 - 2: **Initialize:** Initial iterate $w_0 \in \mathbb{R}^d$
 - 3: **for** $k = 0; 1; \dots; T - 1$ **do**
 - 4: **Central machine:** broadcasts w_k
 for $i \in [m]$ **do in parallel**
 - 5: i -th worker machine:
 - Non-Byzantine: Computes local gradient $g_{i;k}$ and local Hessian $\mathbf{H}_{i;k}$; locally solves the problem described in equation (5.2) and sends $s_{i;k+1}$ to the central machine,
 - Byzantine: Generates $s_{i;k+1}$ (arbitrary), and sends it to the center machine
 - end for**
 - 6: **Center Machine:**
 - Sort the worker machines in a non decreasing order according to norm of updates $\|s_{i;k+1}\|_2$ from the local machines
 - Return the indices of the first $(1 - \epsilon)$ fraction of machines as U_t ,
 - Update parameter: $w_{k+1} = w_k + \frac{1}{\kappa} \sum_{i \in U_t} s_{i;k+1}$
 - 7: **end for**
-

5.4 Distributed Cubic Regularized Newton

We first focus on the non-Byzantine setup ($\alpha = 0$; $\beta = 0$ in Algorithm 5) of distributed cubic regularized Newton algorithm. Byzantine resilience attribute of Algorithm 5 is deferred to Section 5.5. As mentioned before, the data is drawn independently across worker machines from some unknown distribution. The local data at the i th machine is denoted by S_i . Starting with initialization w_0 , the central machine broadcasts the parameter to the worker machines. At k -th iteration, the i -th worker machine solves a cubic-regularized auxiliary loss function based on its local data:

$$s_{i;k+1} = \underset{s}{\operatorname{argmin}} \mathbf{g}_{i;k}^T s + \frac{1}{2} s^T \mathbf{H}_{i;k} s + \frac{M}{6} \|s\|^3; \quad (5.2)$$

where $M > 0$; $\gamma > 0$ are parameter and $\mathbf{g}_{i;t}, \mathbf{H}_{i;t}$ are the gradient and Hessian of the local loss function f_i computed on the independently sampled data (S_i) stored in the worker machine.

$$\begin{aligned} \mathbf{g}_{i;k} &= \nabla f_i(w_k) = \frac{1}{|S_i|} \sum_{z_i \in S_i} \nabla f_i(w_k; z_i); \\ \mathbf{H}_{i;k} &= \nabla^2 f_i(w_k) = \frac{1}{|S_i|} \sum_{z_i \in S_i} \nabla^2 f_i(w_k; z_i); \end{aligned}$$

After receiving the update $s_{i;k+1}$, the central machine updates the parameter in the following way

$$s_{k+1} = s_k + \frac{\gamma}{m} \sum_{i=1}^m s_{i;k+1}; \quad (5.3)$$

where γ is the step-size.

Remark 23. Note that, we introduce the parameter γ in the cubic regularized sub-problem. The parameter γ emphasizes the effect of the second and third order terms in the sub-problem. The choice of γ plays an important role in our analysis in handling the non-linear update from different worker machines. Such non-linearity vanishes if we choose $\gamma = 0$ and the distributed cubic Newton becomes distributed gradient descent algorithm.

5.4.1 Theoretical Guarantees

We have the following standard assumptions:

Assumption 16. The non-convex loss function $f(\cdot)$ is twice continuously-differentiable and bounded below, i.e., $f = \inf_{w \in \mathbb{R}^d} f(x) > -\infty$.

Assumption 17. The loss $f(\cdot)$ is L -Lipschitz continuous ($\|f(\mathbf{z}) - f(\mathbf{y})\| \leq L\|\mathbf{z} - \mathbf{y}\|$), has L_1 -Lipschitz gradients ($\|\nabla f(\mathbf{z}) - \nabla f(\mathbf{y})\| \leq L_1\|\mathbf{z} - \mathbf{y}\|$) and L_2 -Lipschitz Hessian ($\|\nabla^2 f(\mathbf{z}) - \nabla^2 f(\mathbf{y})\| \leq L_2\|\mathbf{z} - \mathbf{y}\|$).

The above assumption states that the loss and the gradient and Hessian of the loss do not drastically change in the local neighborhood. These assumptions are standard in the analysis of the saddle point escape for cubic regularization (see [44, 147]).

In this work, each worker machine solves the cubic sub-problem as described in the equation (5.2). The gradient and Hessian used in the equation are inexact in nature as they are computed using the sub-sampled data.

Assumption 18. For a given $\rho_g > 0$ and for all $k; i$,

$$\| \nabla f(w_k) - \mathbf{g}_{i;k} \| \leq \rho_g \tag{5.4}$$

Assumption 19. For a given $\rho_H > 0$ and for all $k; i$,

$$\| \nabla^2 f(w_k) - \mathbf{H}_{i;k} \| \leq \rho_H \tag{5.5}$$

In the following section, we provide an exact characterization of ρ_g and ρ_H to justify the assumptions.

5.4.2 Inexact Gradient and Hessian

In this work, we assume that each worker machine solve the sub-problem described in equation (5.2) with the data sampled independently from some unknown distribution. So, in each iteration, the gradient and Hessian computed by each worker machine are actually sub-sampled gradient and Hessian of the objective function f and inexact in nature. In the following lemmas, we described the deviation conditions given the size of the sampled data and provide probabilistic deviation bound that ensure the deviation defined in Assumption 18 and 19.

Lemma 23. (Gradient deviation bound) Given S iid data sample, under the Assumption 17, we have $\| \mathbf{g}_i - \nabla f(w_k) \| \leq c \frac{L \sqrt{\log(2d)}}{\sqrt{J_j}}$, with probability exceeding $1 - \beta$, where c is a constant and \mathbf{g}_i is the gradient computed in i -th worker machine.

Lemma 24. (Hessian deviation bound) Given S iid data sample, under the Assumption 17, we have $\| \mathbf{H}_i - \nabla^2 f(w_k) \| \leq c_1 \frac{L_1 \sqrt{\log(2d)}}{\sqrt{J_j}}$, with probability at least $1 - \beta$, where c_1 is a constant and \mathbf{H}_i is the Hessian computed in i -th worker machine.

Remark 24. In the Assumptions 18 and 19, the deviation bounds are in ℓ_2 norm for the gradient and in spectral norm for the Hessian. In the previous works, in centralized model, [44, 45, 147] study cubic regularization with sub-sampled and inexact gradient and Hessian. In the central model, the motivation of the sub-sampled Hessian and gradient is for the ease of the computation. Here, we use the deviation bounds as each of the worker machine only have access to a fraction of the data.

Also, in contrast to the sub-sampled analysis of [44], we choose the deviation of both the gradient and Hessian in the Assumptions 18 and 19 to be independent of the update s which is the solution of the sub-problem (5.2).

The analysis of the deviation bounds follows from the vector and matrix Bernstein inequalities. The assumption of the independent data in each worker can be relaxed. The analysis can be easily extended for data partition (non iid data), following an analysis of [141] the bound of $\frac{1}{\sqrt{JSj}}$ holds.

Theorem 10. *Under the Assumptions 16,17,18, 19, and $\beta = 0$, after T iterations, the sequence $\{w_i g_{i=1}^T\}$ generated by the Algorithm 5 with $\beta = 0$, contains a point w such that*

$$\begin{aligned} \|r f(w)\| &\leq \frac{1}{T^{\frac{2}{3}}} + \frac{2}{T} + (g + H); \\ \min \lambda(r^2 f(x)) &\geq \frac{3}{T^{\frac{1}{3}}} - H; \end{aligned} \quad (5.6)$$

where, $\lambda(\cdot)$ denotes the minimum eigenvalue and

$$\begin{aligned} \beta_1 &= \frac{L_2}{2} + \frac{M}{2} \quad \beta_2 = H \quad \beta_3 = \frac{M}{2} + L_2 \\ &= \frac{f(w_0) - f^*}{2m^2} + \frac{\sum_{k=0}^{T-1} \beta_k}{2m^2} ((m-1)L_1 + H) + \frac{\sum_{k=0}^{T-1} \beta_k g}{m} \\ &= \frac{M}{4m} - \frac{L_2}{6} - \frac{1}{2km} ((m-1)L_1 + H) - \frac{g}{2k} \end{aligned}$$

Remark 25. We choose the step sizes $\beta_k g_{k=0}^T$ such way that $\sum_{i=0}^T \beta_k$ and $\sum_{i=0}^T \frac{\beta_k}{k}$ is bounded. For the ease of choice, we can choose $\beta_k = \frac{c}{k}$, for some constant $c > 0$. Also, we choose $\beta_k = \dots$

Remark 26. Both the gradient and the minimum eigenvalue of the Hessian in the Theorem 10 have two parts. The first part decreases with the number iterations T . The gradient and the minimum eigenvalue of the Hessian have the rate of $O\left(\frac{1}{T^{\frac{2}{3}}}\right)$ and $O\left(\frac{1}{T^{\frac{1}{3}}}\right)$, respectively. Both of these rates match the rates of the centralized version of the cubic regularized Newton. In the second parts of the gradient bound and the minimum eigenvalue of the Hessian have the error floor of $g + H$ and H , respectively. Both the terms g and H decrease at the rate of $\frac{1}{\sqrt{JSj}}$, where JSj is the number of data in each of the worker machines.

Remark 27 (Two rounds of communication $g = 0$). We can improve the bound in the Theorem 10, with the calculation of the actual gradient which requires one more round of communication in each iteration. In the first iteration, all the worker machines compute the gradient based on the stored data and send it to the center machine. The center machine averages them and then broadcast the global gradient $r f(w_k) = \frac{1}{m} \sum_{i=1}^m g_{i:k}$ at iteration k . In this manner, the worker machines solve the sub-problem (5.2) with the actual gradient. The analysis follows same as that of the Theorem 10 with $g = 0$. This improves the gradient bound while the communication remains $O(d)$ in each iteration.

5.4.3 Solving cubic sub-problem

We use a gradient based approach for solving the cubic sub-problem (5.2) in each worker machines (see [147]). The worker machines computes the gradient and Hessian based on the local data stored in the machines and perform the following gradient descent algorithm to yields a solution withing certain tolerance.

Algorithm 6: Gradient based Cubic solver

- 1: **Input:** Step size η , local gradient g and Hessian \mathbf{H} and tolerance $\epsilon > 0$ and M ; \mathcal{M} .
 - 2: **Initialize:** $s = 0$; $G = g$
 - 3: While $\|G\| > \epsilon$;
 - $s = s + \eta G$
 - $G = g + \mathbf{H}s + \frac{M^2}{2} \|s\|^2 s$
 - 4: Return s
-

5.5 Byzantine Resilience

In this section, we analyze our algorithm’s resilience against Byzantine workers. We consider that $(\frac{1}{2})$ fraction of the worker machines are Byzantine in nature. We denote the set of Byzantine worker machines by B and the set of the rest of the good machines as \mathcal{M} . In each iteration, the good machines send the solution of the sub-cubic problem described in equation (5.2) and the Byzantine machines can send any arbitrary values or intentionally disrupt the learning algorithm with malicious updates. Moreover, in the non-convex optimization problems, one of the more complicated and important issue is to avoid saddle points which can yield highly sub-optimal results. In the presence of Byzantine worker machines, they can be in cohort to create a *fake local minima* and drive the algorithm into sub-optimal region. Lack of any robust measure towards these type of intentional and unintentional attacks can be catastrophic to the learning procedure as the learning algorithm can get stuck in such sub-optimal point. To tackle such Byzantine worker machines, we employ a simple process called *norm based thresholding*.

After receiving all the updates from the worker machines, the central machine outputs a set U which consists of the indexes of the worker machines with smallest norm. We choose the size of the set U to be $(1 - \epsilon)m$. Hence, we ‘trim’ ϵ fraction of the worker machine so that we can control the iterated update by not letting the worker machines with large norm participate and diverge the learning process. We denote the set of trimmed machine as T . We choose $\epsilon > 0$ so that at least one of the good machines gets trimmed. In this way, the norm of the all the updates in the set U is bounded by at least the largest norm of the good machines.

Theorem 11. For $0 < \epsilon < \frac{1}{2}$ and under the Assumptions 16,17,18, 19, after T iterations, the sequence $\{w_i\}_{i=1}^T$ generated by the Algorithm 5 contains a point w such that

$$\| \nabla f(w) \| \leq \frac{1;byz}{T^{\frac{2}{3}}} + \frac{2;byz}{T} + g + \frac{(1 - \epsilon)}{(1 + m)} (1 + m) H$$

$$\min_{k=0, \dots, T-1} \|\nabla^2 f(w_k)\| \geq \frac{3;byz}{T^{\frac{1}{3}}} H \quad \text{where,} \quad (5.7)$$

$$1;byz = \frac{L_2(1 + m)(1 - \epsilon)}{2(1 - \epsilon)} + \frac{M(1 + m)^2}{2} \frac{(1 - \epsilon)^2}{(1 - \epsilon)} \frac{2}{byz}$$

$$2;byz = \frac{(1 - \epsilon)}{(1 - \epsilon)} (1 + m) H \frac{3}{byz}$$

$$3;byz = \frac{M(1 - \epsilon)}{2(1 - \epsilon)} (1 + m) + L_2 \frac{(1 + m)(1 - \epsilon)}{(1 - \epsilon)} \quad byz$$

$$byz = \frac{f(w_0) - f}{byz} + \frac{\sum_{k=0}^{T-1} \|\nabla f(w_k)\|^{\frac{1}{3}}}{byz}$$

$$byz = \frac{M(1 - \epsilon)}{4k(1 - \epsilon)^2 m} \frac{(L(1 + m) + g)(1 - \epsilon)}{2(1 - \epsilon)} \frac{(1 - \epsilon) mL_2}{6(1 - \epsilon)}$$

$$\frac{(1 - \epsilon)}{2k(1 - \epsilon)^2 m} (1 - \epsilon) mL_1 + (1 - \epsilon) m^2 + H$$

$$f_{floor} = \frac{k g(1 - \epsilon)}{(1 - \epsilon)} + \frac{kL}{(1 - \epsilon)} + \frac{2}{2(1 - \epsilon)^2 m^2}$$

$$(2(1 - \epsilon)m + 1)L_1 + H)(1 - \epsilon)m + (1 - \epsilon)m^2 L_1$$

Remark 28. Compared to the non-Byzantine part described in Theorem 10, the rate of remains same except for the error floor of the gradient bound suffering a small constant factor.

Remark 29. The condition for the step-size k remains same as described in the Remark 26 and we choose $k = \frac{k(1 - \epsilon)}{(1 - \epsilon)} (1 + m)$.

Remark 30. In the previous work, [31] first provides a *perturbed gradient based algorithm* to escape the saddle point in non-convex optimization in the presence of Byzantine worker machines. Also, in that paper, the Byzantine resilience is achieved using techniques such as trimmed mean, median and collaborative filtering. These methods require additional assumptions (coordinate of the gradient being sub-exponential etc.) for the purpose of analysis. In this work, we perform a simple *norm based thresholding* that provides robustness towards any sorts of adversarial attacks. Also the perturbed gradient descent (PGD) actually requires multiple rounds of communications between the central machine and the worker machines whenever the norm of the gradient is small as this is an indication of either a local minima or a saddle point. In contrast to that, our method does not require any additional communication for *escaping* the saddle points. Our method provides such ability by virtue of cubic regularization.

Remark 31. Since our algorithm is second order in nature, it requires less number of iterations compared to the first order gradient based algorithms. Our algorithm achieves a superior rate of $O\left(\frac{1}{T^{\frac{2}{3}}}\right)$ compared to the gradient based approach of rate $O\left(\frac{1}{T}\right)$. Our algorithm dominates ByzantinePGD [23] in terms of convergence, communication rounds and simplicity and efficiency of Byzantine resilience.

5.6 Experimental Results

In this section, we validate our algorithm in both Byzantine and non-Byzantine setup on benchmark LIBSVM ([47]) dataset in both convex and non-convex problems. We choose the following problems for our experiment.

1. Logistic regression:

$$\min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_i \log(1 + \exp(-y_i \mathbf{x}_i^T w)) + \frac{\lambda}{2n} \|w\|^2; \quad (5.8)$$

2. Non-convex robust linear regression:

$$\min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_i \log\left(\frac{(y_i - w^T \mathbf{x}_i)^2}{2} + 1\right); \quad (5.9)$$

where $w \in \mathbb{R}^d$ is the parameter, $\mathbf{x}_i, \mathbf{g}_{i=1}^n \in \mathbb{R}^d$ are the feature vectors and $y_i, g_{i=1}^n \in \{-1, 1\}$ are the corresponding labels. We choose ‘a9a’ ($d = 123; n = 32K$, we split the data into 70=30 and use as training/testing purpose) and ‘w8a’ (training data $d = 300; n = 50K$ and testing data $d = 300; n = 15K$) classification datasets and partition the data in 20 different worker machines.

In Figure 5.3, we show the performance of our algorithm in non-Byzantine setup ($\epsilon = 0$). In the top row of Figure 5.3, we plot the classification accuracy on test data of both ‘a9a’ and ‘w8a’ datasets for logistic regression problem and in the bottom row of Figure 5.3, we plot the function value of the non-convex robust linear regression problem defined in equation (5.9) for training data of ‘a9a’ and ‘w8a’ datasets. We choose the learning rate $\eta = 1$ and the parameter $\lambda = 1$ and $M = \{10; 15; 20\}$.

Next, we show the effectiveness of our algorithm in Byzantine setup. In this work, we consider the following four Byzantine attacks: (1) ‘Gaussian Noise attack’: where the Byzantine worker machines add Gaussian noise to the update. (2) ‘Random label attack’: where the Byzantine worker machines train and learn based on random labels instead of the proper labels. (3) ‘Flipped label attack’: where (for Binary classification) the Byzantine worker machines flip the labels of the data and learn based on wrong labels. (4) ‘Negative update attack’: where the Byzantine workers computes the update s (here solves the sub-problem in Eq. (5.2)) and communicates c ’s with $c \in (0; 1)$ making the direction of the update opposite of the actual one.

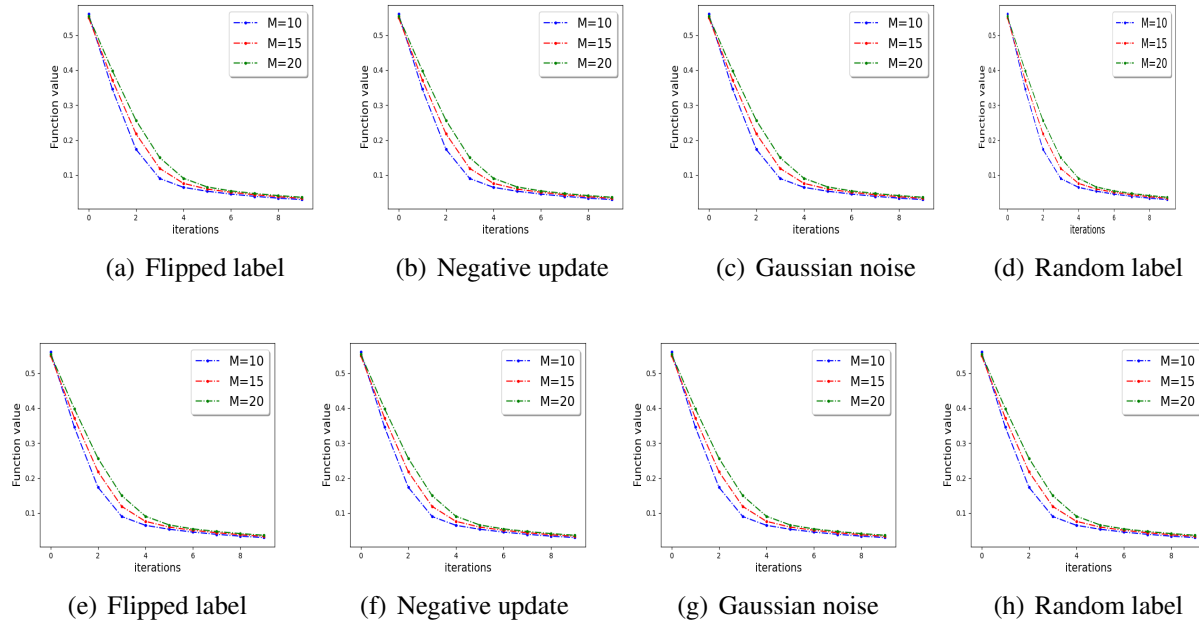


Figure 5.1: Function loss of the training data ‘a9a’ dataset (first row) and ‘w8a’ dataset (second row) with 10%; 15%; 20% Byzantine worker machines for (a,e). Flipped label attack (b,f). Negative Update attack (c,g). Gaussian noise attack and (d,h). Random label attack for non-convex robust linear regression problem.

We show the classification accuracy on testing data of ‘a9a’ and ‘w8a’ dataset for logistic regression problem in Figure 5.2 and training function loss of ‘a9a’ and ‘w8a’ dataset for robust linear regression problem in the Figure 5.1. It is evident from the plots that a simple *norm based thresholding* makes the learning algorithm robust. We choose the parameters $\epsilon = 1; M = 10$, learning rate $\eta = 1$, fraction of the Byzantine machines $f = 1; 15; 20$ and $\delta = \epsilon + \frac{2}{m}$.

We also compare our algorithm with ByzantinePGD [23]. For both the algorithms, we choose $\|\cdot\|_2$ norm of the gradient as a stopping criteria and compute the number of times the worker machines communicate with the center machine as a measure of communication cost and convergence rate. We choose $R = 10; r = 5; Q = 10; T_{th} = 10$ and ‘co-ordinate wise Trimmed mean’ for Byzantine resilience (see the algorithm in [23]). For our algorithm, we choose $M = 10; \epsilon = 1$ in the non-Byzantine setup. The ByzantinePGD algorithm requires 257 rounds of communications (157 rounds for reaching the stopping criteria and 100 rounds for the ‘Escape’ sub-routine to check whether it is a ‘saddle point’) where our algorithm requires only 7 rounds of communication. We choose ‘w8a’ dataset for non-convex robust linear regression problem. Eventhough ByzantinePGD does not require the computation of Hessian and cubic sub-problem solving, our algorithm outperforms by a lot (36x) in terms of communication rounds. In the Appendix, we provide results in Byzantine settings.

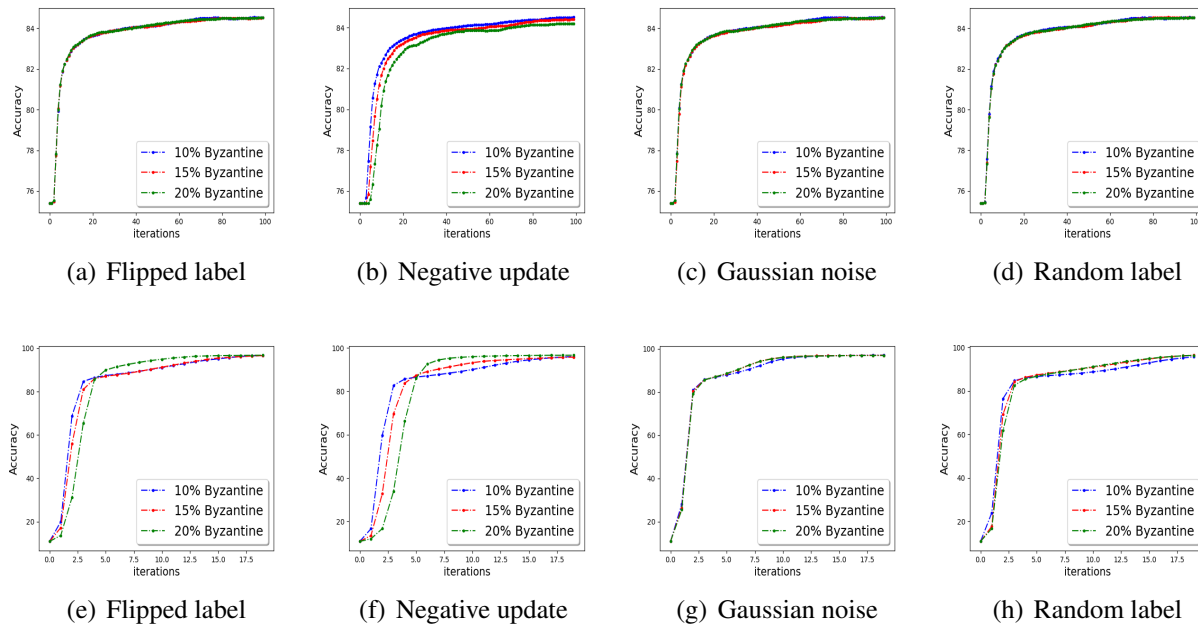


Figure 5.2: Classification accuracy of the testing data ‘a9a’ dataset (first row) and ‘w8a’ dataset (second row) with 10%;15%;20% Byzantine worker machines for (a,e). Flipped label attack.(b,f). Negative Update attack (c,g). Gaussian noise attack and (d,h). Random label attack for logistic regression problem.

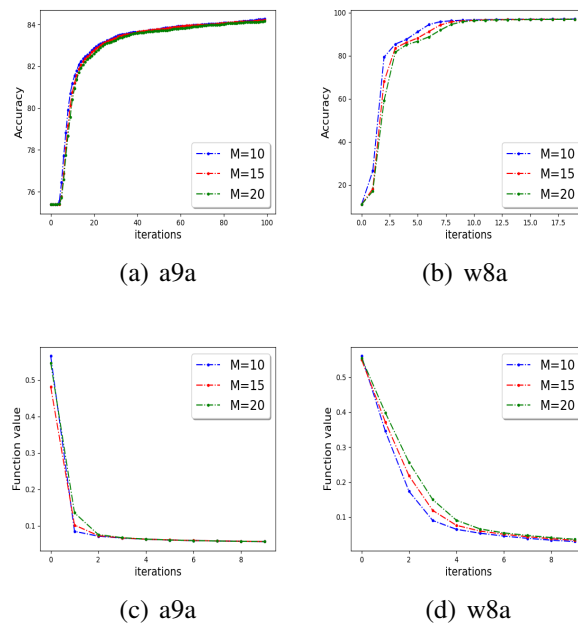


Figure 5.3: (First row) Accuracy of the algorithm for logistic regression on test data of (a). a9a and (b). w8a dataset. (Second row). Function value of the non-convex robust linear regression on the training data of (a). a9a and (b). w8a dataset.

5.7 Proofs

In this part, first, we establish some useful facts and lemmas. Next, we provide the missing proofs and analysis of Theorems 10 and 11 and additional experiments comparing ByzantinePGD [31] in Byzantine setup.

5.7.1 Some useful facts

For the purpose of analysis we use the following sets of inequalities.

Fact 1. For a_1, \dots, a_n we have the following inequality

$$k \sum_{i=1}^n a_i^3 \leq \sum_{i=1}^n ka_i k^3 \leq n^2 \sum_{i=1}^n ka_i k^3 \quad (5.10)$$

$$k \sum_{i=1}^n a_i^2 \leq \sum_{i=1}^n ka_i k^2 \leq n \sum_{i=1}^n ka_i k^2 \quad (5.11)$$

Fact 2. For $a_1, \dots, a_n > 0$ and $r < s$

$$\frac{1}{n} \sum_{i=1}^n a_i^{1-r} \leq \frac{1}{n} \sum_{i=1}^n a_i^{1-s} \quad (5.12)$$

Lemma 25 ([43]). *Under Assumption 17, i.e., the Hessian of the function is L_2 -Lipschitz continuous, for any $w, y \in \mathbb{R}^d$, we have*

$$k r f(w) - r f(y) - r^2 f(w)(y - w) k \leq \frac{L_2}{2} k y - w k^2 \quad (5.13)$$

$$f(y) - f(w) - r f(w)^T (y - w) \leq \frac{1}{2} (y - w)^T r^2 f(w)(y - w) \leq \frac{L_2}{6} k y - w k^2 \quad (5.14)$$

Next, we establish the following Lemma that provides some nice properties of the cubic sub-problem.

Lemma 26. *Let $M > 0$; $\gamma > 0$; $\mathbf{g} \in \mathbb{R}^d$; $\mathbf{H} \in \mathbb{R}^{d \times d}$, and*

$$\mathbf{s} = \underset{w}{\operatorname{argmin}} \mathbf{g}^T w + \frac{\gamma}{2} w^T \mathbf{H} w + \frac{M}{6} k w k^3; \quad (5.15)$$

The following holds

$$\mathbf{g} + \mathbf{H} \mathbf{s} + \frac{M}{2} k \mathbf{s} k \mathbf{s} = \mathbf{0}; \quad (5.16)$$

$$\mathbf{H} + \frac{M}{2} k \mathbf{s} k \mathbf{I} \preceq \mathbf{0}; \quad (5.17)$$

$$\mathbf{g}^T \mathbf{s} + \frac{\gamma}{2} \mathbf{s}^T \mathbf{H} \mathbf{s} \leq \frac{M}{4} k \mathbf{s} k^3; \quad (5.18)$$

Proof. The equations (5.16) and (5.17) are from the first and second order optimal condition. We proof (5.18), by using the conditions of (5.16) and (5.17).

$$\mathbf{g}^T \mathbf{s} + \frac{M}{2} \mathbf{s}^T \mathbf{H} \mathbf{s} = \mathbf{H} \mathbf{s} + \frac{M}{2} \|\mathbf{s}\|^2 \mathbf{s} + \frac{M}{2} \mathbf{s}^T \mathbf{H} \mathbf{s} \quad (5.19)$$

$$\begin{aligned} &= \mathbf{s}^T \mathbf{H} \mathbf{s} + \frac{M}{2} \|\mathbf{s}\|^3 + \frac{M}{2} \mathbf{s}^T \mathbf{H} \mathbf{s} \\ &= \frac{M}{4} \|\mathbf{s}\|^3 + \frac{M}{2} \|\mathbf{s}\|^3 \\ &= \frac{M}{4} \|\mathbf{s}\|^3. \end{aligned} \quad (5.20)$$

In (5.19), we substitute the expression \mathbf{g} from the equation (5.16). In (5.20), we use the fact that $\mathbf{s}^T \mathbf{H} \mathbf{s} + \frac{M}{2} \|\mathbf{s}\|^3 > 0$ from the equation (5.17). □

5.7.2 Proof of Theorem 10

First we state the results of Lemma 26 for each worker node in iteration k ,

$$\mathbf{g}_{i;k} + \mathbf{H}_{i;k} \mathbf{s}_{i;k+1} + \frac{M}{2} \|\mathbf{s}_{i;k+1}\|^2 \mathbf{s}_{i;k+1} = 0 \quad (5.21)$$

$$\mathbf{H}_{i;k} + \frac{M}{2} \|\mathbf{s}_{i;k+1}\|^2 \mathbf{I} \succeq \mathbf{0} \quad (5.22)$$

$$\mathbf{g}_{i;k}^T \mathbf{s}_{i;k+1} + \frac{M}{2} \mathbf{s}_{i;k+1}^T \mathbf{H}_{i;k} \mathbf{s}_{i;k+1} + \frac{M}{4} \|\mathbf{s}_{i;k+1}\|^3 \quad (5.23)$$

At iteration k , we have,

$$\begin{aligned}
 & f(w_{k+1}) - f(w_k) \\
 & \quad r f(w_k)^T (w_{k+1} - w_k) + \frac{1}{2} (w_{k+1} - w_k)^T r^2 f(w_k) (w_{k+1} - w_k) + \frac{L_2}{6} k w_{k+1} - w_k k^3 \\
 & \quad k r f(w_k)^T s_{k+1} + \frac{2}{k^2} s_{k+1}^T r^2 f(w_k) s_{k+1} + \frac{L_2}{6} k k s_{k+1} k^3 \\
 = & \frac{k}{m} r f(w_k)^T \left(\sum_i s_{i;k+1} \right) + \frac{2}{2} \left(\frac{1}{m} \sum_i s_{i;k+1} \right)^T r^2 f(w_k) \left(\frac{1}{m} \sum_i s_{i;k+1} \right) + \frac{L_2}{6} k^3 k s_{k+1} k^3 \quad (5.24)
 \end{aligned}$$

$$\begin{aligned}
 & \frac{k}{m} \sum_i r f(w_k)^T s_{i;k+1} + \frac{2}{2m^2} \sum_i s_{i;k+1}^T r^2 f(w_k) s_{i;k+1} + \sum_{i \neq j} s_{i;k+1}^T r^2 f(w_k) s_{j;k+1} \\
 & + \frac{L_2}{6m} k^3 \sum_i k s_{i;k+1} k^3 \\
 & \frac{k}{m} \sum_i \mathbf{g}_{i;k}^T s_{i;k+1} + \frac{2}{2} s_{i;k+1}^T \mathbf{H}_{i;k} s_{i;k+1} + \frac{k}{m} \sum_i (r f(w_k) - \mathbf{g}_{i;k+1})^T s_{i;k+1} \\
 & \frac{k}{2m} \sum_i s_{i;k+1}^T \mathbf{H}_{i;k} s_{i;k+1} \\
 & + \frac{2}{2m^2} \sum_i s_{i;k+1}^T r^2 f(w_k) s_{i;k+1} + \sum_{i \neq j} s_{i;k+1}^T r^2 f(w_k) s_{j;k+1} + \frac{L_2}{6m} k^3 \sum_i k s_{i;k+1} k^3 \\
 & \frac{k}{m} \sum_i \frac{M}{4} k^2 s_{i;k+1} k^3 + \frac{k}{m} \sum_i g k s_{i;k+1} k \quad \frac{k}{2m} \quad \frac{2}{2m^2} \sum_{i \neq j} s_{i;k+1}^T \mathbf{H}_{i;k} s_{i;k+1} \\
 & + \frac{2}{2m^2} \sum_i s_{i;k+1}^T r^2 f(w_k) \mathbf{H}_{i;k} s_{i;k+1} + \sum_{i \neq j} s_{i;k+1}^T r^2 f(w_k) s_{j;k+1} + \frac{L_2}{6m} k^3 \sum_i k s_{i;k+1} k^3 \quad (5.25)
 \end{aligned}$$

$$\begin{aligned}
 & \frac{M}{4m} k^2 \sum_i k s_{i;k+1} k^3 + \frac{L_2}{6m} k^3 \sum_i k s_{i;k+1} k^3 + \frac{k}{m} \sum_i g k s_{i;k+1} k \\
 & + \frac{k}{2m} \sum_i \frac{2}{2m^2} \sum_i \frac{M}{2} k s_{i;k+1} k^3 \\
 & + \frac{2}{2m^2} \sum_i \mathbf{H}_{i;k} k s_{i;k+1} k^2 + L_1 \sum_{i \neq j} k s_{i;k+1} k k s_{j;k+1} k \quad (5.26)
 \end{aligned}$$

$$\begin{aligned}
 = & \frac{M}{4m^2} k^2 \sum_i k s_{i;k+1} k^3 + \frac{L_2}{6m} k^3 \sum_i k s_{i;k+1} k^3 + \frac{k}{m} \sum_i g k s_{i;k+1} k \\
 & \underbrace{\hspace{10em}}_{Term1} \\
 & + \frac{2}{2m^2} \sum_i \mathbf{H}_{i;k} k s_{i;k+1} k^2 + L_1 \sum_{i \neq j} k s_{i;k+1} k k s_{j;k+1} k \quad (5.27) \\
 & \underbrace{\hspace{10em}}_{Term2}
 \end{aligned}$$

In (5.24), we apply the inequality (5.20) on $k \frac{1}{m} \sum_i \mathbf{s}_{i;k+1}^T \mathbf{P} \mathbf{s}_{i;k+1} k^3$. In line (5.25), we use the gradient approximation from the Assumption 18. In line (5.25), we apply the fact that $\mathbf{s}_{i;k+1}^T \mathbf{H}_{i;k} \mathbf{s}_{i;k+1} + \frac{M}{2} k \mathbf{s}_{i;k+1}^T k^3 > 0$ from the equation (5.22) and assume that $\frac{M}{2} k > \frac{g}{m}$. In line (5.26), we use the Assumption 19 and the fact that the Hessian of the objective function is bounded as the gradient is L_1 -Lipschitz continuous.

Now we bound the Term 1 and Term 2 in equation (5.27).

$$\text{Term 1} = \frac{k g}{m} \sum_i \mathbf{s}_{i;k+1}^T k \mathbf{P} \mathbf{s}_{i;k+1} k^3 = \frac{k g}{m} \sum_i (1 + k \mathbf{s}_{i;k+1}^T k^3) \quad (5.28)$$

In line (5.28), we use the fact $a^2 b \leq a^3 + b^3$ where $a, b > 0$. We choose $a = 1$ and $b = k \mathbf{s}_{i;k+1}^T k$.

$$\begin{aligned} \text{Term 2} &= \frac{2}{2m^2} \sum_i \mathbf{H}_{i;k} \mathbf{s}_{i;k+1}^T k^2 + L_1 \sum_i k \mathbf{s}_{i;k+1}^T k^2 + \sum_i k \mathbf{s}_{i;k+1}^T k^2 \\ &= \frac{2}{2m^2} ((m-1)L_1 + \mathbf{H}) \sum_i k \mathbf{s}_{i;k+1}^T k^2 \end{aligned} \quad (5.29)$$

$$\frac{2}{2m^2} ((m-1)L_1 + \mathbf{H}) \sum_i k \mathbf{s}_{i;k+1}^T k^3 + 1 \quad (5.30)$$

In line (5.29), we use the inequality (5.11) to bound $k \sum_i \mathbf{s}_{i;k+1}^T k^2$ and in line (5.30), we choose $a = k \sum_i \mathbf{s}_{i;k+1}^T k^2$ and $b = 1$ and use $a^2 b \leq a^3 + b^3$.

Combining the result of (5.28) and (5.30) in equation (5.27), we have

$$\begin{aligned} f(W_{k+1}) - f(W_k) &= \frac{M}{4m^2} k^2 + \frac{L_2}{6m} k^3 + \frac{k g}{m} + \frac{2}{2m^2} ((m-1)L_1 + \mathbf{H}) \sum_i k \mathbf{s}_{i;k+1}^T k^3 \\ &+ \frac{k g}{m} + \frac{2}{2m^2} ((m-1)L_1 + \mathbf{H}) \end{aligned}$$

Now we consider that $\frac{M}{4m^2} k^2 = \frac{M}{4km} \frac{L_2}{6} + \frac{1}{2km} ((m-1)L_1 + \mathbf{H}) \frac{g}{k}$. We can assure > 0 by choosing $M \geq \frac{4km}{L_2} \frac{L_2}{6} + \frac{1}{2km} ((m-1)L_1 + \mathbf{H}) + \frac{g}{k}$.

Now we have

$$\frac{1}{m} \sum_i k \mathbf{s}_{i;k+1}^T k^3 \leq \frac{1}{m} (f(W_k) - f(W_{k+1})) + \frac{k g}{m} + \frac{2}{2m^2} ((m-1)L_1 + \mathbf{H})$$

Now we consider the step k_0 , where $k_0 = \arg \min_{0 \leq k \leq T-1} \|kw_{k+1} - w_k\| = \arg \min_{0 \leq k \leq T-1} \sum_{i=1}^n k \|s_{i;k_0+1}\|^3$.

$$\begin{aligned} \min_{0 \leq k \leq T} \|kw_{k+1} - w_k\|^3 &= \min_{0 \leq k \leq T} \sum_{i=1}^n k \|s_{i;k_0+1}\|^3 \\ &= \frac{1}{T} \sum_{k=0}^{T-1} \sum_{i=1}^n k \|s_{i;k+1}\|^3 \\ &= \frac{1}{T} \sum_{k=0}^{T-1} \left(f(w_k) - f(w_{k+1}) + \frac{2}{2m} \sum_{k=0}^{T-1} ((m-1)L_1 + H) + \frac{k}{m} g \right) \\ &= \frac{1}{T} \left(\frac{f(w_0) - f(w_T)}{2m} + \sum_{k=0}^{T-1} \frac{2}{2m} ((m-1)L_1 + H) + \sum_{k=0}^{T-1} \frac{k}{m} g \right) \\ &= \frac{1}{T} \left(\frac{f(w_0) - f(w_T)}{2m^2} + \sum_{k=0}^{T-1} \frac{2}{2m^2} ((m-1)L_1 + H) + \sum_{k=0}^{T-1} \frac{k}{m} g \right) \end{aligned}$$

Based on the calculation above, we have

$$\|kw_{k_0+1} - w_{k_0}\|^3 \leq \frac{1}{T^{\frac{1}{3}}} \sum_{i=1}^n k \|s_{i;k_0+1}\|^3 + \frac{1}{T^{\frac{1}{3}}} \left(\frac{f(w_0) - f(w_T)}{2m^2} + \sum_{k=0}^{T-1} \frac{2}{2m^2} ((m-1)L_1 + H) + \sum_{k=0}^{T-1} \frac{k}{m} g \right)^{\frac{1}{3}}$$

With the proper choice step-size k , we choose

$$k = \frac{1}{T^{\frac{1}{3}}} \left(\frac{f(w_0) - f(w_T)}{2m^2} + \sum_{k=0}^{T-1} \frac{2}{2m^2} ((m-1)L_1 + H) + \sum_{k=0}^{T-1} \frac{k}{m} g \right)^{\frac{1}{3}} \text{ and have}$$

$$\|kw_{k_0+1} - w_{k_0}\|^3 \leq \frac{1}{T^{\frac{1}{3}}} \sum_{i=1}^n k \|s_{i;k_0+1}\|^3 + \frac{1}{T^{\frac{1}{3}}} \left(\frac{f(w_0) - f(w_T)}{2m^2} + \sum_{k=0}^{T-1} \frac{2}{2m^2} ((m-1)L_1 + H) + \sum_{k=0}^{T-1} \frac{k}{m} g \right)^{\frac{1}{3}} \quad (5.31)$$

Now the gradient condition

$$kr f(w_{k+1})k = r f(w_{k+1}) \frac{1}{m} \sum_{i=1}^n \mathbf{g}_{i;k} - \frac{1}{m} \sum_{i=1}^n \left(\mathbf{H}_{i;k+1} \mathbf{s}_{i;k+1} - \frac{M^2}{2} k \mathbf{s}_{i;k+1} k \mathbf{s}_{i;k+1} \right) \quad (5.32)$$

$$\begin{aligned} & r f(w_{k+1}) - r f(w_k) - r^2 f(w_k) (x_{k+1} - x_k) + \frac{1}{m} \sum_{i=1}^n (\mathbf{g}_{i;k} - r f(w_k)) \\ & + r^2 f(w_k) (x_{k+1} - x_k) - \frac{1}{m} \sum_{i=1}^n \mathbf{H}_{i;k} \mathbf{s}_{i;k+1} + \frac{1}{m} \sum_{i=1}^n \frac{M^2}{2} k \mathbf{s}_{i;k+1} k \mathbf{s}_{i;k+1} \\ & \frac{L_2}{2} k k \mathbf{s}_{k+1} k^2 + \frac{k}{m} \sum_{i=1}^n r^2 f(w_k) \mathbf{s}_{i;k+1} - \frac{1}{m} \sum_{i=1}^n \mathbf{H}_{i;k} \mathbf{s}_{i;k+1} + \frac{M^2}{2m} \sum_i k \mathbf{s}_{i;k+1} k^2 + g \end{aligned} \quad (5.33)$$

$$\begin{aligned} & \frac{L_2}{2m} k^2 + \frac{M^2}{2m} \sum_i k \mathbf{s}_{i;k+1} k^2 + \frac{k}{m} \sum_{i=1}^n r^2 f(w_k) \mathbf{s}_{i;k+1} - \frac{1}{m} \sum_{i=1}^n r^2 f(w_k) \mathbf{s}_{i;k+1} \\ & + \frac{1}{m} \sum_{i=1}^n r^2 f(w_k) \mathbf{s}_{i;k+1} - \frac{1}{m} \sum_{i=1}^n \mathbf{H}_{i;k} \mathbf{s}_{i;k+1} + g \\ & \frac{L_2}{2m} k^2 + \frac{M^2}{2m} \sum_i k \mathbf{s}_{i;k+1} k^2 + \frac{(k)}{m} L_1 \sum_i k \mathbf{s}_{i;k+1} k + \frac{H}{m} \sum_i k \mathbf{s}_{i;k+1} k + g \end{aligned} \quad (5.34)$$

$$\begin{aligned} & \frac{L_2}{2} k^2 + \frac{M^2}{2} \frac{1}{m} \sum_i k \mathbf{s}_{i;k+1} k^2 + (j_k - j L_1 + H) \frac{1}{m} \sum_i k \mathbf{s}_{i;k+1} k + g \\ & \frac{L_2}{2} + \frac{M^2}{2} \frac{1}{m} \sum_i k \mathbf{s}_{i;k+1} k^2 + j_1 - \frac{j L_1}{k} + \frac{H}{k} \frac{1}{m} \sum_i k \mathbf{s}_{i;k+1} k + g \\ & \frac{L_2}{2} + \frac{M^2}{2} \frac{1}{m} \sum_i k \mathbf{s}_{i;k+1} k^2 + j_1 - \frac{j L_1}{k} + \frac{H}{k} \frac{1}{m} \sum_i k \mathbf{s}_{i;k+1} k^3 \end{aligned} \quad (5.35)$$

$$\begin{aligned} & + g + j_1 - \frac{j L_1}{k} + \frac{H}{k} \\ & \frac{L_2}{2} + \frac{M^2}{2} \frac{1}{m} \sum_i k \mathbf{s}_{i;k+1} k^3 + j_1 - \frac{j L_1}{k} + \frac{H}{k} \frac{1}{m} \sum_i k \mathbf{s}_{i;k+1} k^3 \end{aligned} \quad (5.36)$$

$$+ g + j_1 - \frac{j L_1}{k} + \frac{H}{k} \quad (5.37)$$

In line (5.32), we use first order condition described in equation (5.21). In line (5.33), we apply the result (5.13) from Lemma 25 and the approximate gradient condition from Assumption 18. In line (5.34), we apply the approximate Hessian condition from Assumption 19. We apply the

inequality of (5.12) in line (5.37). At step k_0 , by choosing $\delta_k = \frac{1}{T^{\frac{1}{3}}}$, we have

$$\begin{aligned} \min(r^2 f(w_{k_0+1})) &\leq \frac{L_2}{2} + \frac{M}{2} \frac{1}{T^{\frac{2}{3}}} + H \frac{1}{T} + (g + H) \\ &= \frac{1}{T^{\frac{2}{3}}} + \frac{2}{T} + (g + H) \end{aligned} \quad (5.38)$$

where, $\delta_1 = \frac{L_2}{2} + \frac{M}{2} \frac{1}{T^{\frac{2}{3}}}$ and $\delta_2 = H \frac{1}{T}$. The Hessian bound

$$\begin{aligned} \min(r^2 f(w_{k+1})) &= \frac{1}{m} \sum_{i=1}^m \min(r^2 f(w_{k+1})) \\ &= \frac{1}{m} \sum_{i=1}^m \min(\mathbf{H}_{i;k}) (\mathbf{H}_{i;k}^{-1} r^2 f(w_{k+1})) \\ &\leq \frac{1}{m} \sum_{i=1}^m \min(\mathbf{H}_{i;k}) \|\mathbf{H}_{i;k}^{-1}\| \|r^2 f(w_{k+1})\| \\ &\leq \frac{1}{m} \sum_{i=1}^m \min(\mathbf{H}_{i;k}) \frac{1}{m} \sum_{i=1}^m \|\mathbf{H}_{i;k}^{-1}\| \|r^2 f(w_{k+1})\| \end{aligned} \quad (5.39)$$

$$\begin{aligned} &\leq \frac{1}{m} \sum_{i=1}^m \frac{M}{2} \|\mathbf{s}_{i;k+1}\| \frac{1}{m} \sum_{i=1}^m \|\mathbf{H}_{i;k}^{-1}\| \|r^2 f(w_k) - r^2 f(w_{k+1})\| \\ &\leq \frac{1}{m} \sum_{i=1}^m \frac{M}{2} \|\mathbf{s}_{i;k+1}\| \frac{1}{m} \sum_{i=1}^m \|\mathbf{H}_{i;k}^{-1}\| \|r^2 f(w_k) - r^2 f(w_{k+1})\| \\ &\leq \frac{1}{m} \sum_{i=1}^m \frac{M}{2} \|\mathbf{s}_{i;k+1}\| \frac{1}{m} \sum_{i=1}^m \|\mathbf{H}_{i;k}^{-1}\| \|L_2 \|w_k - w_{k+1}\| \end{aligned} \quad (5.40)$$

$$\begin{aligned} &\leq \frac{M}{2} L_2 \frac{1}{m} \sum_{i=1}^m \|\mathbf{s}_{i;k+1}\| \frac{1}{m} \sum_{i=1}^m \|\mathbf{H}_{i;k}^{-1}\| \|L_2 \|w_k - w_{k+1}\| \\ &\leq \frac{M}{2} L_2 \frac{1}{m} \sum_{i=1}^m \|\mathbf{s}_{i;k+1}\|^3 \frac{1}{m} \sum_{i=1}^m \|\mathbf{H}_{i;k}^{-1}\| \|L_2 \|w_k - w_{k+1}\| \end{aligned} \quad (5.41)$$

Equation (5.39) follows from Weyl's inequality. We apply the Hessian approximation from the Assumption 19 in equation (5.40). In equation (5.41), we apply the result described in (5.12).

At step k_0 , by choosing $\delta_k = \frac{1}{T^{\frac{1}{3}}}$, we have

$$\begin{aligned} \min(r^2 f(w_{k_0+1})) &\leq \frac{M}{2} + L_2 \frac{1}{T^{\frac{1}{3}}} + H \\ &= \frac{3}{T^{\frac{1}{3}}} + H \end{aligned} \quad (5.42)$$

where, $\beta_3 = \frac{M}{2} + L_2$.

5.7.3 Proof of Theorem 11

We consider the following

$$\begin{aligned}
 & f(W_{k+1}) - f(W_k) \\
 & r f(W_k)^T (W_{k+1} - W_k) + \frac{1}{2} (W_{k+1} - W_k)^T r^2 f(W_k) (W_{k+1} - W_k) + \frac{L_2}{6} \|W_{k+1} - W_k\|^3 \\
 = & \underbrace{\frac{k}{jU_j} r f(W_k)^T \sum_{i \in 2U} \mathbf{s}_{i;k+1}}_{\text{Term1}} + \underbrace{\frac{k^2}{2jU_j^2} \sum_{i \in 2U} \mathbf{s}_{i;k+1}^T r^2 f(W_k) \sum_{i \in 2U} \mathbf{s}_{i;k+1}}_{\text{Term2}} + \underbrace{\frac{L_2}{6} \sum_{i \in 2U} \|\mathbf{s}_{i;k+1}\|^3}_{\text{Term3}}
 \end{aligned} \tag{5.43}$$

In line (5.43), we expand the update $W_{k+1} - W_k = \sum_{i \in 2U} \mathbf{p}_{i;k+1}$. Also we use the following fact.

$$jU_j = jU \setminus M_j + jU \setminus B_j \tag{5.44}$$

$$jM_j = jU \setminus M_j + jT \setminus M_j \tag{5.45}$$

Combining both the equations (5.44) and (5.45), we have

$$jU_j = jM_j \cup jT \setminus M_j + jU \setminus B_j \tag{5.46}$$

We use the fact of (5.46) to bound each term in equation (5.43). First, consider the Term 1,

$$\begin{aligned}
 & \frac{k}{jU_j} r f(W_k)^T \sum_{i \in 2U} \mathbf{s}_{i;k+1} \\
 = & \frac{k}{(1 - \frac{M}{2})m} r f(W_k)^T \sum_{i \in 2M} \mathbf{s}_{i;k+1} + \frac{k}{(1 - \frac{M}{2})m} r f(W_k)^T \sum_{i \in 2M \setminus T} \mathbf{s}_{i;k+1} + \frac{k}{(1 - \frac{M}{2})m} r f(W_k)^T \sum_{i \in 2U \setminus B} \mathbf{s}_{i;k+1} \\
 = & \frac{k}{(1 - \frac{M}{2})m} \left[\sum_{i \in 2M} \mathbf{g}_{i;k}^T \mathbf{s}_{i;k+1} + \sum_{i \in 2M \setminus T} r f(W_k)^T \mathbf{s}_{i;k+1} + \sum_{i \in 2U \setminus B} r f(W_k)^T \mathbf{s}_{i;k+1} \right] \\
 + & \frac{k}{2} \sum_{i \in 2M} \mathbf{s}_{i;k+1}^T \mathbf{H}_{i;k} \mathbf{s}_{i;k+1} \\
 + & \frac{k}{(1 - \frac{M}{2})m} \sum_{i \in 2M} (r f(W_k) \mathbf{g}_{i;k})^T \mathbf{s}_{i;k+1}
 \end{aligned} \tag{5.47}$$

$$\begin{aligned}
 & \frac{M^2 k}{4(1 - \frac{M}{2})m} \sum_{i \in 2M} \|\mathbf{s}_{i;k+1}\|^3 + \frac{k g}{(1 - \frac{M}{2})m} \sum_{i \in 2M} \|\mathbf{s}_{i;k+1}\|^k + \frac{k}{2(1 - \frac{M}{2})m} \sum_{i \in 2M} \mathbf{s}_{i;k+1}^T \mathbf{H}_{i;k} \mathbf{s}_{i;k+1} \\
 + & \frac{kL}{(1 - \frac{M}{2})m} \sum_{i \in 2M \setminus T} \|\mathbf{s}_{i;k+1}\|^k + \frac{kL}{(1 - \frac{M}{2})m} \sum_{i \in 2U \setminus B} \|\mathbf{s}_{i;k+1}\|^k
 \end{aligned} \tag{5.48}$$

We use the following facts in (5.48).

- $\mathbf{g}_{i;k}^T \mathbf{s}_{i;k+1} + \frac{1}{2} \mathbf{s}_{i;k+1}^T \mathbf{H}_{i;k} \mathbf{s}_{i;k+1} - \frac{M}{4} \|\mathbf{s}_{i;k+1}\|_k^3$ and sum over the set \mathcal{M} .
- The gradient approximation described in Assumption 18.
- As the function f is L -Lipschitz, the gradient is bounded.

Now we bound Term 3 as follows,

$$\begin{aligned} & \frac{L_2}{6} \frac{\|\mathbf{s}_{i;k+1}\|_k^3}{jUj} \times_{i \in 2U} \quad \frac{L_2}{6(1-\frac{1}{m})} \frac{\|\mathbf{s}_{i;k+1}\|_k^3}{m} \times_{i \in 2U} \quad \# \\ & \frac{L_2}{6(1-\frac{1}{m})} \frac{\|\mathbf{s}_{i;k+1}\|_k^3}{m} \times_{i \in 2M} \quad \times_{i \in 2M \setminus T} \quad \times_{i \in 2U \setminus B} \quad \# \end{aligned} \quad (5.49)$$

$$\frac{L_2}{6(1-\frac{1}{m})} \frac{\|\mathbf{s}_{i;k+1}\|_k^3}{m} \times_{i \in 2M} \quad \times_{i \in 2M \setminus T} \quad \times_{i \in 2U \setminus B} \quad \# \quad (5.50)$$

In line (5.49), we use the inequality described in (5.10) and in line (5.50), we split the sum using (5.46).

Finally, we bound Term 2

$$\begin{aligned} & \frac{\|\mathbf{s}_{i;k+1}\|_k^2}{2jUj^2} \times_{i \in 2U} \quad \times_{i \in 2U} \quad \times_{i \in 2U} \\ & = \frac{\|\mathbf{s}_{i;k+1}\|_k^2}{2(1-\frac{1}{m})^2 m^2} \left(\times_{i \in 2U} \mathbf{s}_{i;k+1}^T r^2 f(W_k) \mathbf{s}_{i;k+1} + \times_{i \notin j2U} \mathbf{s}_{i;k+1}^T r^2 f(W_k) \mathbf{s}_{j;k+1} \right) \\ & = \frac{\|\mathbf{s}_{i;k+1}\|_k^2}{2(1-\frac{1}{m})^2 m^2} \left(\times_{i \in 2M} \mathbf{s}_{i;k+1}^T (r^2 f(W_k) - \mathbf{H}_{i;k}) \mathbf{s}_{i;k+1} \quad \times_{i \in 2M \setminus T} \mathbf{s}_{i;k+1}^T r^2 f(W_k) \mathbf{s}_{i;k+1} \right. \\ & \quad \left. + \times_{i \in 2U \setminus B} \mathbf{s}_{i;k+1}^T r^2 f(W_k) \mathbf{s}_{i;k+1} \right) \\ & \quad + \frac{\|\mathbf{s}_{i;k+1}\|_k^2}{2(1-\frac{1}{m})^2 m^2} \times_{i \notin j2U} \mathbf{s}_{i;k+1}^T r^2 f(W_k) \mathbf{s}_{j;k+1} + \frac{\|\mathbf{s}_{i;k+1}\|_k^2}{2(1-\frac{1}{m})^2 m^2} \times_{i \in 2M} \mathbf{s}_{i;k+1}^T \mathbf{H}_{i;k} \mathbf{s}_{i;k+1} \end{aligned} \quad (5.51)$$

$$\begin{aligned} & \frac{\|\mathbf{s}_{i;k+1}\|_k^2}{2(1-\frac{1}{m})^2 m^2} \times_{i \in 2M} \left(\|\mathbf{s}_{i;k+1}\|_k^2 + L_1 \|\mathbf{s}_{i;k+1}\|_k^2 + L_1 \|\mathbf{s}_{i;k+1}\|_k^2 \right) \times_{i \in 2U \setminus B} \|\mathbf{s}_{i;k+1}\|_k^2 \\ & \quad + \frac{\|\mathbf{s}_{i;k+1}\|_k^2}{2(1-\frac{1}{m})^2 m^2} L_1 \|\mathbf{s}_{i;k+1}\|_k^2 \times_{i \in 2U} \quad \times_{i \in 2U} \|\mathbf{s}_{i;k+1}\|_k^2 + \frac{\|\mathbf{s}_{i;k+1}\|_k^2}{2(1-\frac{1}{m})^2 m^2} \times_{i \in 2M} \mathbf{s}_{i;k+1}^T \mathbf{H}_{i;k} \mathbf{s}_{i;k+1} \end{aligned} \quad (5.52)$$

$$\begin{aligned} & \frac{\|\mathbf{s}_{i;k+1}\|_k^2}{2(1-\frac{1}{m})^2 m^2} \left[\left((1-\frac{1}{m})L_1 + L_1 \right) \times_{i \in 2M} \|\mathbf{s}_{i;k+1}\|_k^2 + \left((1-\frac{1}{m})L_1 + L_1 \right) \times_{i \in 2M \setminus T} \|\mathbf{s}_{i;k+1}\|_k^2 \right. \\ & \quad \left. + \left((1-\frac{1}{m})L_1 \right) \times_{i \in 2U \setminus B} \|\mathbf{s}_{i;k+1}\|_k^2 \right] + \frac{\|\mathbf{s}_{i;k+1}\|_k^2}{2(1-\frac{1}{m})^2 m^2} \times_{i \in 2M} \mathbf{s}_{i;k+1}^T \mathbf{H}_{i;k} \mathbf{s}_{i;k+1} \end{aligned} \quad (5.53)$$

Now we collect the terms from equations (5.48), (5.50) and (5.53). First we focus on the terms that are summed over the set \mathcal{M} .

$$\begin{aligned}
 & \left(\frac{M^2}{4(1-k)m} + \frac{L_2^3}{6(1-k)m} \right) \sum_{i \in \mathcal{M}} [ks_{i;k+1}k^3] + \frac{k}{2(1-k)m} \left(\frac{k}{(1-k)m} \right) \sum_{i \in \mathcal{M}} \mathbf{s}_{i;k+1}^T \mathbf{H}_{i;k} \mathbf{s}_{i;k+1} \\
 & + \frac{k}{(1-k)m} \sum_{i \in \mathcal{M}} ks_{i;k+1}k + \frac{k}{2(1-k)^2m^2} ((1-k)m-1)L_1 + H \sum_{i \in \mathcal{M}} ks_{i;k+1}k^2 \\
 & \frac{M^2}{4(1-k)m} + \frac{L_2^3}{6(1-k)m} \sum_{i \in \mathcal{M}} ks_{i;k+1}k^3 \\
 & + \frac{k}{(1-k)m} + \frac{k}{2(1-k)^2m^2} ((1-k)m-1)L_1 + H \sum_{i \in \mathcal{M}} ks_{i;k+1}k^3 + 1 \tag{5.54} \\
 = & \frac{M^2}{4(1-k)m} + \frac{L_2^3}{6(1-k)m} + \frac{k}{(1-k)m} + \frac{k}{2(1-k)^2m^2} ((1-k)m-1)L_1 + H \\
 & \sum_{i \in \mathcal{M}} ks_{i;k+1}k^3 + \frac{k}{(1-k)} + \frac{k}{2(1-k)^2m} ((1-k)m-1)L_1 + H(1-k) \tag{5.55}
 \end{aligned}$$

In line (5.54), we use the fact of 5.22 and $ks_{i;k+1}k \leq ks_{i;k+1}k^3 + 1$ and $ks_{i;k+1}k^2 \leq ks_{i;k+1}k^3 + 1$ following the inequality $a^b \leq a^3 + b^3$. Also, we use the fact that $|j \setminus \mathcal{M}| \leq (1-k)m$.

Now we consider the terms with the set $\mathcal{M} \setminus \mathcal{T}$ and $U \setminus B$.

$$\begin{aligned}
 & \frac{kL}{(1-k)m} \left(\sum_{i \in \mathcal{M} \setminus \mathcal{T}} ks_{i;k+1}k + \sum_{i \in U \setminus B} ks_{i;k+1}k \right) \tag{5.56} \\
 & + \frac{L_2^3}{6(1-k)m} \left[\sum_{i \in \mathcal{M} \setminus \mathcal{T}} ks_{i;k+1}k^3 + \sum_{i \in U \setminus B} ks_{i;k+1}k^3 \right] \\
 & + \frac{k}{2(1-k)^2m^2} ((1-k)m+2)L_1 \sum_{i \in \mathcal{M} \setminus \mathcal{T}} ks_{i;k+1}k^2 + (1-k)mL_1 \sum_{i \in U \setminus B} ks_{i;k+1}k^2 \\
 & \frac{kL}{(1-k)m} + \frac{L_2^3}{6(1-k)m} + \frac{k}{2(1-k)^2m^2} ((1-k)m+2)L_1 \sum_{i \in \mathcal{M} \setminus \mathcal{T}} ks_{i;k+1}k^3 \\
 & + \frac{kL}{(1-k)m} + \frac{L_2^3}{6(1-k)m} + \frac{k}{2(1-k)^2m^2} (1-k)mL_1 \sum_{i \in U \setminus B} ks_{i;k+1}k^3 \\
 & + \frac{kL}{(1-k)m} + \frac{k}{2(1-k)^2m} [(1-k)m+2(1-k)]L_1 \tag{5.57}
 \end{aligned}$$

Now as $\epsilon > \epsilon_0$, at least one good machine is trimmed. So the norm of all the machine update in the set U is upper bounded by the maximum norm of the good machine. We upper bound the terms as

follows,

$$\begin{aligned} & \times \prod_{i \in 2U \setminus B} k s_{i;k+1} k^3 \quad m \max_{i \in 2M} k s_{i;k+1} k^3 \quad m \times \prod_{i \in 2M} k s_{i;k+1} k^3 \\ \text{and} & \quad \prod_{i \in 2M \setminus T} k s_{i;k+1} k^3 \quad \prod_{i \in 2M} k s_{i;k+1} k^3 \end{aligned}$$

We have

$$\begin{aligned} & \frac{kL(1+m)}{(1-k)m} + \frac{(m-1)L_2}{6(1-k)m} \frac{k^3}{k} + \frac{k^2}{2(1-k)^2 m^2} ((1-k)m + (1-k)m^2 + 2)L_1 \\ & \times \prod_{i \in 2M} k s_{i;k+1} k^3 + \frac{kL}{(1-k)m} + \frac{k^2}{2(1-k)^2 m} [(1-k)m + 2(1-k)] L_1 \end{aligned} \quad (5.58)$$

We combine the results (5.58) and (5.55). We have

$$f(w_{k+1}) - f(w_k) \leq \frac{1}{\text{byz} (1-k)m} \times \prod_{i \in 2M} k s_{i;k+1} k^3 + f_{\text{floor}} \quad (5.59)$$

where

$$\begin{aligned} \text{byz} &= \left[\frac{M}{4k(1-k)^2 m^2} \frac{(L(1+m) + g)}{k(1-k)m} - \frac{mL_2}{6(1-k)m} - \frac{1}{2k(1-k)^2 m^2} (1-k)mL_1 \right. \\ & \left. + (1-k)m^2 + H \right] (1-k)m \\ f_{\text{floor}} &= \frac{k^3 g(1-k)}{(1-k)} + \frac{kL}{(1-k)} \\ & + \frac{k^2}{2(1-k)^2 m^2} (2(1-k)m + 1)L_1 + H(1-k)m + (1-k)m^2 L_1 \end{aligned}$$

We maintain $\text{byz} > 0$ by choosing

$$\begin{aligned} M &> \frac{4k(1-k)m}{k} \left[\frac{(L(1+m) + g)}{k} + \frac{mL_2}{6} \right. \\ & \left. + \frac{1}{2k(1-k)m} ((1-k)mL_1 + (1-k)m^2 + H) \right] \end{aligned}$$

Now we can have the following results from the proof of Theorem 10 for step

$$k_0 = \arg \min_{0 \leq k \leq T-1} \|w_{k+1} - w_k\|$$

$$\begin{aligned} \|w_{k_0+1} - w_{k_0}\| & \leq \frac{1}{(1-k_0)m} \times \prod_{i \in 2M} k_{k_0} s_{i;k_0+1} k^3 \quad \# \frac{1}{3} \\ & \leq \frac{1}{T^{\frac{1}{3}}} \frac{f(w_0) - f}{\text{byz}} + \frac{\sum_{k=0}^{T-1} f_{\text{floor}}}{\text{byz}} \quad \# \frac{1}{3} \end{aligned} \quad (5.60)$$

$$= \frac{\text{byz}}{T^{\frac{1}{3}}} \quad (5.61)$$

where, $\text{byz} = \frac{h}{\text{byz}} \frac{f(w_0) - f}{\text{byz}} + \frac{P_{k=0}^T - 1}{\text{byz}} \frac{i}{\text{byz}} \frac{1}{3}$.

The gradient condition

$$\| \nabla r f(w_{k+1}) \|_k = \frac{1}{jMj} \sum_{i \in 2M} \mathbf{g}_{i:k} - \frac{1}{jMj} \sum_{i \in 2M} \mathbf{H}_{i:k+1} \mathbf{s}_{i:k+1} - \frac{M^2}{2} \|\mathbf{s}_{i:k+1}\|_k \|\mathbf{s}_{i:k+1}\| \quad (5.62)$$

$$\begin{aligned} & r f(w_{k+1}) - r f(w_k) - r^2 f(w_k)(x_{k+1} - x_k) + \frac{1}{jMj} \sum_{i \in 2M} (\mathbf{g}_{i:k} - r f(w_k)) \\ & + r^2 f(w_k)(x_{k+1} - x_k) - \frac{1}{jMj} \sum_{i \in 2M} \mathbf{H}_{i:k} \mathbf{s}_{i:k+1} + \frac{1}{jMj} \sum_{i \in 2M} \frac{M^2}{2} \|\mathbf{s}_{i:k+1}\|_k \|\mathbf{s}_{i:k+1}\| \\ & + \frac{L_2^2}{2} \frac{1}{jUj} \sum_{i \in 2U} \|\mathbf{s}_{i:k+1}\|^2 + g + \frac{M^2}{2} \frac{1}{jMj} \sum_{i \in 2M} \|\mathbf{s}_{i:k+1}\|_k^2 \\ & + \frac{k}{jUj} \sum_{i \in 2U} r^2 f(w_k) \mathbf{s}_{i:k+1} - \frac{1}{jMj} \sum_{i \in 2M} r^2 f(w_k) \mathbf{s}_{i:k+1} \\ & + \frac{1}{jMj} \sum_{i \in 2M} r^2 f(w_k) \mathbf{s}_{i:k+1} - \frac{1}{jMj} \sum_{i \in 2M} \mathbf{H}_{i:k} \mathbf{s}_{i:k+1} \end{aligned} \quad (5.63)$$

In line (5.62), we use the fact the first order optimal condition (5.21) holds for the good machines in the set \mathcal{M} . And in (5.63), we use the in exact gradient condition from Assumption 18 and the condition (5.23). Consider the term

$$\begin{aligned} & \frac{k}{jUj} \sum_{i \in 2U} r^2 f(w_k) \mathbf{s}_{i:k+1} - \frac{1}{jMj} \sum_{i \in 2M} r^2 f(w_k) \mathbf{s}_{i:k+1} \\ & = k \frac{k}{(1+m)m} \sum_{i \in 2M} r^2 f(w_k) \mathbf{s}_{i:k+1} - \frac{1}{jMj} \sum_{i \in 2M \setminus T} r^2 f(w_k) \mathbf{s}_{i:k+1} + \frac{1}{jMj} \sum_{i \in 2B \setminus U} r^2 f(w_k) \mathbf{s}_{i:k+1} \\ & \quad - \frac{1}{(1+m)m} \sum_{i \in 2M} r^2 f(w_k) \mathbf{s}_{i:k+1} k \\ & \quad + \frac{k(1+m)}{(1+m)m} \frac{1}{(1+m)m} \sum_{i \in 2M} r^2 f(w_k) \mathbf{s}_{i:k+1} \\ & \quad - \frac{k(1+m)}{(1+m)m} \frac{1}{(1+m)m} L_1 \sum_{i \in 2M} r^2 \|\mathbf{s}_{i:k+1}\|_k \end{aligned} \quad (5.64)$$

We choose $\alpha = \frac{k(1+m)}{2}$ making the term in equation (5.64) equals to 0. Now we have,

$$\frac{L_2}{2} \frac{1}{jUj} \times_{i2U} s_{i;k+1}^2 + \frac{L_2}{2(1+m)} \frac{1}{m} \times_{i2U} k s_{i;k+1} k^2 + \frac{L_2(1+m)}{2(1+m)} \frac{1}{m} \times_{i2M} k s_{i;k+1} k^2 \quad (5.65)$$

Putting the calculation of (5.64) and (5.65), in (5.63), we have,

$$\|r f(w_{k+1})\| \leq \frac{L_2(1+m)}{2(1+m)} \frac{1}{m} \times_{i2M} k s_{i;k+1} k^2 + g + \frac{H}{(1+m)} \frac{1}{m} \times_{i2M} k s_{i;k+1} k \quad (5.66)$$

$$\frac{L_2(1+m)}{2(1+m)} \frac{1}{m} \times_{i2M} k s_{i;k+1} k^2 + g + \frac{H}{(1+m)} \frac{1}{m} \times_{i2M} k s_{i;k+1} k \quad (5.67)$$

$$\begin{aligned} & \frac{L_2(1+m)(1+m)}{2(1+m)} + \frac{M}{2} \frac{1}{k} \times_{i2M} k s_{i;k+1} k^2 + g \\ & + \frac{H}{k} \frac{1}{(1+m)} \times_{i2M} k s_{i;k+1} k \\ & \frac{L_2(1+m)(1+m)}{2(1+m)} + \frac{M(1+m)^2}{2} \frac{(1+m)^2}{(1+m)} + \frac{1}{(1+m)} \times_{i2M} k s_{i;k+1} k^3 \\ & + \frac{(1+m)}{(1+m)} (1+m) H \frac{1}{(1+m)} \times_{i2M} k s_{i;k+1} k^3 + g + \frac{(1+m)}{(1+m)} (1+m) H \end{aligned} \quad (5.68)$$

We use the power mean inequality described in (5.12) in line (5.68). Then at step k_0 , we have,

$$\|r f(w_{k_0+1})\| \leq \frac{1_{:byz}}{T^{\frac{2}{3}}} + \frac{2_{:byz}}{T} + g + \frac{(1+m)}{(1+m)} (1+m) H \quad (5.69)$$

where

$$\begin{aligned} 1_{:byz} &= \frac{L_2(1+m)(1+m)}{2(1+m)} + \frac{M(1+m)^2}{2} \frac{(1+m)^2}{(1+m)} \\ 2_{:byz} &= \frac{(1+m)}{(1+m)} (1+m) H \end{aligned}$$

The Hessian bound is

$$\begin{aligned}
 & \min(r^2 f(w_{k+1})) \\
 &= \frac{1}{(1-\frac{1}{m})^{i2M}} \min r^2 f(w_{k+1}) \\
 &= \frac{1}{(1-\frac{1}{m})^{i2M}} \min \mathbf{H}_{i;k} \quad (\mathbf{H}_{i;k} \quad r^2 f(w_{k+1})) \\
 & \frac{1}{(1-\frac{1}{m})^{i2M}} \min(\mathbf{H}_{i;k}) \quad k\mathbf{H}_{i;k} \quad r^2 f(w_{k+1})k \tag{5.70}
 \end{aligned}$$

$$\begin{aligned}
 & \frac{1}{(1-\frac{1}{m})^{i2M}} \min(\mathbf{H}_{i;k}) \quad \frac{1}{(1-\frac{1}{m})^{i2M}} \quad k\mathbf{H}_{i;k} \quad r^2 f(w_{k+1})k \\
 & \frac{1}{(1-\frac{1}{m})^{i2M}} \quad \frac{M}{2} k\mathbf{s}_{i;k+1}k \quad \frac{1}{(1-\frac{1}{m})^{i2M}} \quad k\mathbf{H}_{i;k} \quad r^2 f(w_k)k \\
 & \frac{1}{(1-\frac{1}{m})^{i2M}} \quad k r^2 f(w_k) \quad r^2 f(w_{k+1})k \tag{5.71}
 \end{aligned}$$

$$\frac{1}{(1-\frac{1}{m})^{i2M}} \quad \frac{M}{2} k\mathbf{s}_{i;k+1}k \quad H \quad \frac{1}{(1-\frac{1}{m})^{i2M}} \quad L_2 k w_k \quad w_{k+1}k \tag{5.72}$$

$$\begin{aligned}
 & \frac{M}{2(1-\frac{1}{m})^{i2M}} \quad k\mathbf{s}_{i;k+1}k \quad H \quad L_2 k \frac{k}{(1-\frac{1}{m})^{i2U}} \quad \mathbf{s}_{i;k+1}k \\
 & \frac{M}{2(1-\frac{1}{m})^{i2M}} \quad k\mathbf{s}_{i;k+1}k \quad H \quad L_2 \frac{1}{(1-\frac{1}{m})^{i2U}} \quad k \quad k\mathbf{s}_{i;k+1}k \\
 & \frac{M}{2(1-\frac{1}{m})^{i2M}} \quad k\mathbf{s}_{i;k+1}k \quad H \quad L_2 \frac{(1+m)}{(1-\frac{1}{m})^{i2M}} \quad k \quad k\mathbf{s}_{i;k+1}k \\
 & \frac{M}{2} \frac{1}{k(1-\frac{1}{m})^{i2M}} + L_2 \frac{(1+m)}{(1-\frac{1}{m})^{i2M}} \quad k \quad k\mathbf{s}_{i;k+1}k \quad H \\
 & \frac{M}{2} \frac{1}{k} + L_2 \frac{(1+m)(1-\frac{1}{m})}{(1-\frac{1}{m})^{i2M}} \quad \frac{1}{(1-\frac{1}{m})^{i2M}} \quad k \quad k\mathbf{s}_{i;k+1}k^{\frac{3}{2}} \quad H \tag{5.73}
 \end{aligned}$$

In (5.70), we use the Weyl's inequality. In (5.72), we use the fact that Hessian is Lipschitz continuous. In (5.73), we use the power mean inequality described in (5.12) At step k_0 , we have

$$\min(r^2 f(w_{k_0+1})) \geq \frac{3:byz}{T^{\frac{1}{3}}} \quad H \tag{5.74}$$

where

$$3:byz = \frac{M(1-\frac{1}{m})}{2(1-\frac{1}{m})^{i2M}} (1+m) + L_2 \frac{(1+m)(1-\frac{1}{m})}{(1-\frac{1}{m})^{i2M}} \quad byz$$

5.7.4 Additional Experiments

Here we compare our algorithm with the ByzantinePGD [31] in the byzantine setup. In the Table 5.1, we show the number iterations that is required by each algorithm to reach the stopping criteria based on the gradient norm. We choose the four adversarial attacks and the experimental setup that are described in Section 5.6.

	Gaussian Noise			Flipped Label		
	10%	15%	20%	10%	15%	20%
[31]	199.2	198.3	211.2	199.7	198.6	211.6
Our method	2.0	10.1	13.5	10.3	14.1	16.0

Table 5.1: Number of iterations required by ByzantinePGD [23] and our method when 10%, 15%, and 20% of the worker machines are Byzantine in nature.

	Negative Update			Random Label		
	10%	15%	20%	10%	15%	20%
[31]	200.7	197.6	210.5	199.9	198	209.7
Our method	8.7	10.1	13.5	8.8	10.0	10.7

Table 5.2: Number of iterations required by ByzantinePGD [23] and our method when 10%, 15%, and 20% of the worker machines are Byzantine in nature.

Part II

Efficient and Tractable Algorithms for Non-Convex Batch Learning

In this part of the thesis, we study some statistical and algorithmic aspects of non-convex optimization in the standard supervised batch setting. In particular we study max-affine regression, which is a generalization of several classical non-convex problems including the real phase retrieval.

- **Chapter 6:** We study the max affine regression with Gaussian covariates
- **Chapter 7:** We study the max-affine regression problem for small-ball covariate design.

Chapter 6

Max Affine Regression with Gaussian Design

Max-affine regression refers to a model where the unknown regression function is modeled as a maximum of k unknown affine functions for a fixed $k \geq 1$. This generalizes linear regression and (real) phase retrieval, and is closely related to convex regression. Working within a non-asymptotic framework, we study this problem in the high-dimensional setting assuming that k is a fixed constant, and focus on the estimation of the unknown coefficients of the affine functions underlying the model. We analyze a natural alternating minimization (AM) algorithm for the non-convex least squares objective when the design is Gaussian. We show that the AM algorithm, when initialized suitably, converges with high probability and at a geometric rate to a small ball around the optimal coefficients. In order to initialize the algorithm, we propose and analyze a combination of a spectral method and a random search algorithm in a low-dimensional space, which may be of independent interest. The final rate that we obtain is near-parametric and minimax optimal (up to a polylogarithmic factor) as a function of the dimension, sample size, and noise variance. In that sense, our approach should be viewed as a *direct* and implementable method of enforcing regularization to alleviate the curse of dimensionality in problems of the convex regression type. Numerical experiments illustrate the sharpness of our bounds in the various problem parameters.

6.1 Introduction

Max-affine regression refers to the regression model

$$Y = \max_{1 \leq j \leq k} hX;_j^i + b_j + \epsilon \tag{6.1}$$

where Y is a univariate response, X is a d -dimensional vector of covariates and ϵ models zero-mean noise that is independent of X . We assume that $k \geq 1$ is a known integer and study the problem of estimating the unknown parameters $\beta_1, \dots, \beta_k \in \mathbb{R}^d$ and $b_1, \dots, b_k \in \mathbb{R}$ from independent observations $(x_1; y_1), \dots, (x_n; y_n)$ drawn according to the model (6.1). Furthermore,

we assume for concreteness¹ in this chapter that the covariate distribution is standard Gaussian, with $x_i \stackrel{i.i.d.}{\sim} N(0; I_d)$.

Let us provide some motivation for studying the model (6.1). When $k = 1$, equation (6.1) corresponds to the classical linear regression model. When $k = 2$, the intercepts $b_2 = b_1 = 0$, and $\beta_2 = \beta_1 = \beta$, the model (6.1) reduces to

$$Y = \beta^T X; \quad \beta \in \mathbb{R}^d \tag{6.2}$$

The problem of recovering β from observations drawn according to the above model is known as (real) phase retrieval—variants of which arise in a diverse array of science and engineering applications [48–51]—and has associated with it an extensive statistical and algorithmic literature.

To motivate the model (6.1) for general k , note that the function $x \mapsto \max_{1 \leq j \leq k} (hx; \beta_j^T x + b_j)$ is always a convex function and, thus, estimation under the model (6.1) can be used to fit convex functions to the observed data. Indeed, the model (6.1) serves as a parametric approximation to the non-parametric convex regression model

$$Y = f(X) + \epsilon; \tag{6.3}$$

where $f: \mathbb{R}^d \rightarrow \mathbb{R}$ is an unknown convex function. It is well-known that convex regression suffers from the curse of dimensionality unless d is small, which is basically a consequence of the fact that the metric entropy of natural totally bounded sub-classes of convex functions grows exponentially in d (see, e.g., [52–54]). To overcome this curse of dimensionality, one would need to work with more structured sub-classes of convex functions. Since convex functions can be approximated to arbitrary accuracy by maxima of affine functions, it is reasonable to regularize the problem by considering only those convex functions that can be written as a maximum of a fixed number of affine functions. Constraining the number of affine pieces in the function therefore presents a simple method to enforce structure, and such function classes have been introduced and studied in the convex regression literature (see e.g., [152]). This assumption directly leads to our model (6.1), and it has been argued by [55, 153, 154] that the parametric model (6.1) is a tractable alternative to the full non-parametric convex regression model (6.3) in common applications of convex regression to data arising in economics, finance and operations research where d is often moderate to large.

Another motivation for the model (6.1) comes from the problem of estimating convex sets from support function measurements. The support function of a compact convex set $K \subset \mathbb{R}^d$ is defined by $h_K(x) := \sup_{u \in K} \langle x, u \rangle$ for d -dimensional unit vectors x . The problem of estimating an unknown compact, convex set K from noisy measurements of $h_K(\cdot)$ arises in certain engineering applications such as robotic tactile sensing and projection magnetic resonance imaging (see, e.g., [155–157]). Specifically, the model considered here is

$$Y = h_K(X) + \epsilon;$$

and the goal is to estimate the set $K \subset \mathbb{R}^d$. As in convex regression, this problem suffers from a curse of dimensionality unless d is small, as is evident from known minimax lower bounds [158].

¹In Chapter 7 weakens distributional assumptions on the covariates, but this requires significantly more technical effort.

To alleviate this curse, it is natural to restrict \mathcal{K} to the class of all polytopes with at most k extreme points for a fixed k ; such a restriction has been studied as a special case of enforcing structure in these problems by Soh and Chandrasekharan [159]. Under this restriction, one is led to the model (6.1) with $b_1 = \dots = b_k = 0$, since if \mathcal{K} is the polytope given by the convex hull of $x_1, \dots, x_k \in \mathbb{R}^d$, then its support function is equal to $x \mapsto \max_{1 \leq j \leq k} h(x; x_j)$.

Equipped with these motivating examples, our goal is to study a computationally efficient estimation methodology for the unknown parameters of the model (6.1) from i.i.d observations $(x_i; y_i)_{i=1}^n$. Before presenting our contributions, let us first rewrite the observation model (6.1) by using more convenient notation, and use it to describe existing estimation procedures for this model. Denote the unknown parameters by $\beta_j := (x_j; b_j) \in \mathbb{R}^{d+1}$ for $j = 1, \dots, k$ and the observations by $(x_i; y_i)$ for $i = 1, \dots, n$, where $x_i := (x_i; 1) \in \mathbb{R}^{d+1}$. In this notation, the observation model takes the form

$$y_i = \max_{1 \leq j \leq k} h(x_i; \beta_j) + \epsilon_i \quad \text{for } i = 1, 2, \dots, n. \tag{6.4}$$

We assume that in addition to the covariates being i.i.d. standard Gaussian, the noise variables $\epsilon_1, \dots, \epsilon_n$ are independent random variables drawn from a (univariate) distribution that is zero-mean and sub-Gaussian, with unknown sub-Gaussian parameter σ .

Let us now describe existing estimation procedures for max-affine regression. The most obvious approach is the global least squares estimator, defined as any minimizer of the least squares criterion

$$(\beta_1^{(ls)}, \dots, \beta_k^{(ls)}) \in \underset{\beta_1, \dots, \beta_k \in \mathbb{R}^{d+1}}{\operatorname{argmin}} \sum_{i=1}^n \left(y_i - \max_{1 \leq j \leq k} h(x_i; \beta_j) \right)^2. \tag{6.5}$$

It is easy to see (see Lemma 27 to follow) that a global minimizer of the least squares criterion above always exists but it will not—at least in general—be unique, since any relabeling of the indices of a minimizer will also be a minimizer. While the least squares estimator has appealing statistical properties (see, e.g. [158–160]), the optimization problem (6.5) is non-convex and, in general, NP-hard [103, 161] without further assumptions on the covariate matrix and response vector.²

It is interesting to compare (6.5) to the optimization problem used to compute the least squares estimator in the more general convex regression model (6.3), given by

$$b^{(ls)} \in \underset{b \in \mathcal{C}}{\operatorname{argmin}} \sum_{i=1}^n (y_i - b(x_i))^2; \tag{6.6}$$

where the minimization is over all convex functions b . In sharp contrast to the problem (6.5), the optimization problem (6.6) is convex [162, 163] and can be solved efficiently for fairly large values of the pair $(d; n)$ [164]. Unfortunately however, the utility of $b^{(ls)}$ in estimating the parameters of the max-affine model is debatable, as it is unclear how one may obtain estimates of the true parameters β_1, \dots, β_k from $b^{(ls)}$, which typically will *not* be a maximum of only k affine functions.

²We provide a proof of this in Appendix 6.9.14 for completeness.

Three heuristic techniques for solving the non-convex optimization problem (6.5) were empirically evaluated by Balázs [154, Chapters 6 and 7], who compared running times and performance of these techniques on a wide variety of real and synthetic datasets for convex regression. The first technique is the alternating minimization algorithm of Magnani and Boyd [55], the second technique is the convex adaptive partitioning (or CAP) algorithm of Hannah and Dunson [153], and the third is the adaptive max-affine partitioning algorithm proposed by Balázs himself [154]. The simplest and most intuitive of these three methods is the first alternating minimization (AM) algorithm, which is an iterative algorithm for estimating the parameters β_1, \dots, β_k and forms the focus of our study. In the t -th iteration of the algorithm, the current estimates $\beta_1^{(t)}, \dots, \beta_k^{(t)}$ are used to partition the observation indices $1, \dots, n$ into k sets $S_1^{(t)}, \dots, S_k^{(t)}$ such that $j \geq \arg\max_{u \in [k]} h_i(\beta_u^{(t)})$ for every $i \in S_j^{(t)}$. For each $1 \leq j \leq k$, the next estimate $\beta_j^{(t+1)}$ is then obtained by performing a least squares fit (or equivalently, linear regression) to the data $(x_i, y_i); i \in S_j^{(t)}$. More intuition and a formal description of the algorithm are provided in Section 6.2. Balázs found that when this algorithm was run on a variety of datasets with multiple random initializations, it compared favorably with the state of the art in terms of its final predictive performance—see, for example, Figures 7.4 and 7.5 in the thesis [154], which show encouraging results when the algorithm is used to fit convex functions to datasets of average wages and aircraft profile drag data, respectively. In the context of fitting convex sets to support function measurements, Soh and Chandrasekaran [159] recently proposed and empirically evaluated a similar algorithm in the case of isotropic covariates. However, to the best of our knowledge, no theoretical results exist to support the performance of such a technique.

In this chapter, we present a theoretical analysis of the AM algorithm for recovering the parameters of the max-affine regression model when the covariate distribution is Gaussian. This assumption forms a natural starting point for the study of many iterative algorithms in related problems [59–61, 99], and is also quite standard in theoretical investigations of multidimensional regression problems. Note that the AM algorithm described above can be seen as a generalization of classical AM algorithms for (real) phase retrieval [95, 165], which have recently been theoretically analyzed in a series of papers [59–61] for Gaussian designs. The AM—and the closely related expectation maximization³, or EM—methodology is widely used for parameter estimation in missing data problems [166, 167] and mixture models [91], including those with covariates such as mixtures-of-experts [168] and mixtures-of-regressions [169] models. Theoretical guarantees for such algorithms have been established in multiple statistical contexts [99, 170–172]; in the case when the likelihood is not unimodal, these are typically of the local convergence type. In particular, algorithms of the EM type return, for many such latent variable models, minimax-optimal parameter estimates when initialized in a neighborhood of the optimal solution (e.g., [169, 173, 174]); conversely, these algorithms can get stuck at spurious fixed points when initialized at random [175]. In some specific applications of EM to mixtures of two Gaussians [176, 177] and mixtures of two regressions [178], however, it has been shown that randomly initializing the EM algorithm suffices in order to obtain consistent parameter estimates. Here, we establish guarantees on the AM algorithm for max-affine regression that are of the former type: we prove local geometric convergence of the

³Indeed, for many problems, the EM algorithm reduces to AM in the noiseless limit, and AM should thus be viewed as a variant of EM that uses hard-thresholding to determine values of the latent variables.

AM iterates when initialized in a neighborhood of the optimal solution. We analyze the practical variant of the algorithm in which the steps are performed without sample-splitting. As in the case of mixture models [169, 179], we use spectral methods to obtain such an initialization.

Contributions Let us now describe our results in more detail. To simplify the exposition, we state simplified corollaries of our theorems; for precise statements, see Section 6.3. We prove in Theorem 12 that for each $\epsilon > 0$, the parameter estimates $\theta_1^{(t)}; \dots; \theta_k^{(t)}$ returned by the AM algorithm at iteration t satisfy, with high probability, the inequality

$$\sum_{j=1}^k \|\theta_j^{(t)} - \theta_j^*\|^2 \leq C(\theta_1^*; \dots; \theta_k^*) \frac{2kd}{n} \log(kd) \log \frac{n}{kd} \quad (6.7)$$

for every $t \geq \log_{4/3} \frac{P_{j=1}^k \|\theta_j^{(0)} - \theta_j^*\|^2}{\epsilon^2}$, provided that the sample size n is sufficiently large and that the initial estimates satisfy the condition

$$\min_{c>0} \max_{1 \leq j \leq k} c \|\theta_j^{(0)} - \theta_j^*\|^2 \leq \frac{1}{k} C(\theta_1^*; \dots; \theta_k^*) \quad (6.8)$$

Here $C(\theta_1^*; \dots; \theta_k^*)$ and $c(\theta_1^*; \dots; \theta_k^*)$ are constants depending only on the true parameters $\theta_1^*; \dots; \theta_k^*$, and their explicit values are given in Theorem 12. The constant c in equation (6.8) endows the initialization with a scale-invariance property: indeed, scaling all parameters $\theta_1^{(0)}; \dots; \theta_k^{(0)}$ by the same positive constant c produces the same initial partition of subsets $S_1^{(0)}; \dots; S_k^{(0)}$, from which the algorithm proceeds identically.

Treating k as a fixed constant, inequality (6.7) implies, under the initialization condition (6.8), that the parameter estimates returned by AM converge geometrically to within a small ball of the true parameters, and that this error term is nearly the parametric risk $\frac{2d}{n}$ up to a logarithmic factor. The initialization condition (6.8) requires the distance between the initial estimates and the true parameters to be at most a specific (k -dependent) constant. It has been empirically observed that there exist bad initializations under which the AM algorithm behaves poorly (see, e.g., [55, 154]) and assumption (6.8) is one way to rule these out.

A natural question based on our Theorem 12 is whether it is possible to produce preliminary estimates $\theta_1^{(0)}; \dots; \theta_k^{(0)}$ satisfying the initialization condition (6.8). Indeed, one such method is to repeatedly initialize parameters (uniformly) at random within the unit ball \mathbb{B}^{d+1} ; Balázs empirically observed in a close relative of such a scheme (see Figure 6.6 in his thesis [154]) that increasing the number of random initializations is often sufficient to get the AM algorithm to succeed. However, reasoning heuristically, the number of repetitions required to ensure that one such random initialization generates parameters that satisfy condition (6.8) increases exponentially in the ambient dimension d , and so it is reasonable to ask if, in large dimensions, there is some natural form of dimensionality reduction that allows us to perform this step in a lower-dimensional space.

When⁴ $k < d$, we show that a natural spectral method (described formally in Algorithm 8) is able to reduce the dimensionality of our problem from d to k . In particular, this method returns an orthonormal basis of vectors $\vartheta_1, \dots, \vartheta_k$ such that the k -dimensional linear subspace spanned by these vectors accurately estimates the subspace spanned by the vectors $\varphi_1, \dots, \varphi_k$. We form the matrix $\mathcal{V} := [\vartheta_1 \ \dots \ \vartheta_k]$ by collecting these vectors as its columns, and in order to account for the intercepts, further append such a matrix to form the matrix $\tilde{\mathcal{V}} := \begin{bmatrix} \mathcal{V} & 0 \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{(d+1) \times (k+1)}$.

We then choose M random initializations in $(k + 1)$ dimensions—the ℓ -th such initialization is given by a set of vectors $\tilde{g}_1, \dots, \tilde{g}_k \in \mathbb{R}^{k+1}$ each chosen uniformly at random from the $(k + 1)$ -dimensional unit ball—so that the collection of k vectors $\tilde{\mathcal{V}} \tilde{g}_{j=1}^k$ serves as our ℓ -th guess of the true parameters. In order to decide which of these random points to choose, we evaluate (on an independent set of samples) the goodness-of-fit statistic $\min_c \sum_i (y_i - c \max_{j=1, \dots, k} h_i(\tilde{\mathcal{V}} \tilde{g}_j))^2$ for each $1 \leq \ell \leq M$, where the minimization over the constant c accounts for the scale-invariance property alluded to above. Letting ℓ^* denote the index with the smallest loss, we then return the initialization $\tilde{g}_j^{(0)} = \tilde{\mathcal{V}} \tilde{g}_j$ for $j = 1, \dots, k$.

Our algorithm can thus be viewed as a variant of the repeated random initialization evaluated by Balázs [154], but incurs significantly smaller computational cost, since we only run the full-blown iterative AM algorithm once. Note that our algorithm treats the number of initializations M as a tuning parameter to be chosen by the statistician, similar to Balázs [154], but we show a concrete upper bound on M that is sufficient to guarantee convergence. In particular, we show that in order to produce an initialization satisfying condition (6.8) with high probability, it suffices to choose M as a function *only* of the number of affine pieces k and other geometric parameters of the problem (and independently of the sample size n and ambient dimension d).

To produce our overall guarantee, we combine the initialization with the AM algorithm in Corollary 2, showing that provided the sample size scales linearly in the dimension (with a multiplicative pre-factor that depends polynomially on k and other problem-dependent parameters), we obtain estimates that are accurate up to the parametric risk. Our algorithm is also computationally efficient when k is treated as a fixed constant.

From a technical standpoint, our results for the AM algorithm are significantly more challenging to establish than related results in the literature [59, 99, 180, 181]. First, it is technically very challenging to compute the *population operator* [99]—corresponding to running the AM update in the infinite sample limit—in this setting, since the \max function introduces intricate geometry in the problem that is difficult to reason about in closed form. Second, we are interested in analyzing the AM update without sample-splitting, and so cannot assume that the iterates are independent of the covariates; the latter assumption has been used fruitfully in the literature to simplify analyses of such algorithms [60, 61, 180]. Third, and unlike algorithms for phase retrieval [59, 181], our algorithm performs least squares using sub-matrices of the covariate matrix that are chosen *depending* on our random iterates. Accordingly, a key technical difficulty of the proof, which may be of independent interest, is to control the spectrum of these random matrices, rows of which are

⁴If $k = d$, then this dimensionality reduction step can be done away with and one can implement the random search routine directly.

drawn from (randomly) truncated variants of the Gaussian distribution.

Our spectral initialization algorithm is also a natural estimator based on the method-of-moments, and has been used in a variety of non-convex problems [169, 173, 174]. However, our guarantees for this step are once again non-trivial to establish. In particular, the eigengap of the population moment (on which the rates of the estimator depend) is difficult to compute in our case since the \max function is not differentiable, and so it is not clear that higher order moments return reasonable estimates even in the infinite sample limit (see Section 6.2). However, since we operate exclusively with Gaussian covariates, we are able to use some classical moment calculations for truncated Gaussian distributions [182] in order to bound the eigengap. Translating these calculations into an eigengap is quite technical, and involves the isolation of many properties of the population moments that may be of independent interest.

Finally, it is important to note that owing to the scale invariance of our initialization condition (6.8) and goodness-of-fit statistic, our random search scheme does not require a bound on the size of the parameters; it suffices to initialize parameters uniformly within the *unit* ball. This is in contrast to other search procedures employed for similar problems [103, 183], which are based on covering arguments and require a bound on the maximum norm of the unknown parameters.

Notation For a positive integer n , let $[n] := \{1, 2, \dots, n\}$. For a finite set S , we use $|S|$ to denote its cardinality. All logarithms are to the natural base unless otherwise mentioned. For two sequences $\{a_n\}_{n=1}^{\infty}$ and $\{b_n\}_{n=1}^{\infty}$, we write $a_n \sim b_n$ if there is a universal constant C such that $a_n \leq Cb_n$ for all $n \geq 1$. The relation $a_n \asymp b_n$ is defined analogously, and we use $a_n \approx b_n$ to indicate that both $a_n \asymp b_n$ and $a_n \sim b_n$ hold simultaneously. We use $c; C; c_1; c_2; \dots$ to denote universal constants that may change from line to line. For a pair of vectors $(u; v)$, we let $u \cdot v := uv^T$ denote their inner product. We use $\|k\|$ to denote the ℓ_2 norm unless otherwise stated. Denote by I_d the $d \times d$ identity matrix. We let $\mathbf{1}\{E\}$ denote the indicator of an event E . Let $\text{sgn}(t)$ denote the sign of a scalar t , with the convention that $\text{sgn}(0) = 1$. Let $\lambda_i(\cdot)$ denote the i -th largest eigenvalue of a symmetric matrix \cdot . Let $S^{d-1} := \{v \in \mathbb{R}^d : \|v\| = 1\}$ denote the unit sphere in d -dimensions, and use $B^d := \{v \in \mathbb{R}^d : \|v\| \leq 1\}$ to denote the d -dimensional unit ball.

6.2 Background and problem formulation

In this section, we formally introduce the geometric parameters underlying the max-affine regression model, as well as the methodology we use to perform parameter estimation.

6.2.1 Model and Geometric Parameters

We work throughout with the observation model defined in equation (6.4); recall that our covariates are drawn i.i.d. from a standard Gaussian distribution, and that our noise is σ -sub-Gaussian. We let $X \in \mathbb{R}^{n \times d}$ denote the covariate matrix with row i given by the vector x_i , and collect the responses in a vector $y \in \mathbb{R}^n$.

Recall that $x_i = (x_i; 1) \in \mathbb{R}^{d+1}$ for each $i \in [n]$; the matrix of appended covariates $X \in \mathbb{R}^{n \times (d+1)}$ is defined by appending a vector of ones to the right of the matrix X . Our primary goal is to use the data $(X; y)$ —or equivalently, the pair $(X; y)$ —to estimate the underlying parameters $\beta_j, j=1, \dots, k$.

An important consideration in achieving such a goal is the “effective” sample size with which we observe the parameter β_j . Toward that end, for $X \sim \mathcal{N}(0; I_d)$, let

$$j(\beta_1, \dots, \beta_k) := \Pr \left(\max_{j \in [k]} hX; \beta_j = \max_{j^0 \in [k]} hX; \beta_{j^0} \right) \tag{6.9}$$

denote the probability with which the j -th parameter $\beta_j = (\beta_j; b_j)$ attains the maximum. Note that the event on which more than one of the parameters attains the maximum has measure zero, except in the case where $\beta_i = \beta_j$ for some $i \neq j$. We explicitly disallow this case and assume that the parameters β_1, \dots, β_k are distinct. Let

$$\min(\beta_1, \dots, \beta_k) := \min_{j \in [k]} j(\beta_1, \dots, \beta_k); \tag{6.10}$$

and assume that we have $\min(\beta_1, \dots, \beta_k) > 0$; in other words, we ignore vacuous cases in which some parameter is never observed. Roughly speaking, the sample size of the parameter that is observed most rarely is given by $\min_{j \in [k]} j n \approx n \min(\beta_1, \dots, \beta_k)$; and so the error in estimating this parameter should naturally depend on $\min(\beta_1, \dots, \beta_k)$. By definition, we always have $\min(\beta_1, \dots, \beta_k) \leq 1/k$.

Since we are interested in performing parameter estimation under the max-affine regression model, a few geometric quantities also appear in our bounds, and serve as natural notions of “signal strength” and “condition number” of the estimation problem. The signal strength is given by the minimum separation

$$(\beta_1, \dots, \beta_k) = \min_{j, j^0: j \neq j^0} \|\beta_j - \beta_{j^0}\|^2;$$

we also assume that $(\beta_1, \dots, \beta_k)$ is strictly positive, since otherwise, a particular parameter is never observed. To denote a natural form of conditioning, define the quantities

$$j(\beta_1, \dots, \beta_k) = \frac{\max_{j^0 \in [k]} \|\beta_j - \beta_{j^0}\|^2}{\min_{j^0 \in [k]} \|\beta_j - \beta_{j^0}\|^2}; \quad \text{with} \quad (\beta_1, \dots, \beta_k) = \max_{j \in [k]} j(\beta_1, \dots, \beta_k);$$

Finally, let $B_{\max}(\beta_1, \dots, \beta_k) := \max_{j \in [k]} \|\beta_j\|$ denote the maximum norm of any unknown parameter. We often use the shorthand

$$\begin{aligned} \min &= \min(\beta_1, \dots, \beta_k); & (\beta_1, \dots, \beta_k) &= (\beta_1, \dots, \beta_k); \\ &= (\beta_1, \dots, \beta_k); & \text{and} & \quad B_{\max} = B_{\max}(\beta_1, \dots, \beta_k) \end{aligned}$$

when the true parameters β_1, \dots, β_k are clear from context.

6.2.2 Methodology

As discussed in the introduction, the most natural estimation procedure from i.i.d. samples $(x_i; y_i)_{i=1}^n$ of the model (6.4) is the least squares estimator (6.5). The following lemma (which does not seem to have been explicitly stated previously in the literature, except in the case $k = 2$ [160, 184]) proves that the least squares estimator $(b_1^{(ls)}; \dots; b_k^{(ls)})$ always exists. Note, however, that it will not be unique in general since any relabeling of a minimizer is also a minimizer.

Lemma 27. *The least squares estimator $(b_1^{(ls)}; \dots; b_k^{(ls)})$ exists for every dataset $(x; y)$.*

We postpone the proof of Lemma 27 to Appendix 6.5.1. In spite of the fact that the least squares estimator always exists, the problem (6.5) is non-convex and NP-hard in general. The AM algorithm presents a tractable approach towards solving it in the statistical setting that we consider.

Alternating Minimization

We now formally describe the AM algorithm proposed by Magnani and Boyd [55]. For each $j = 1; \dots; k$, define the sets

$$S_j(x; y) := \{i \in [n] : j = \min_{u \in [k]} \arg \max_{i \in [n]} (h_{i; u}^T y)\} \tag{6.11}$$

for $j = 1; \dots; k$. In words, the set $S_j(x; y)$ contains the indices of samples on which parameter β_j attains the maximum; in the case of a tie, samples having multiple parameters attaining the maximum are assigned to the set with the smallest corresponding index (i.e., ties are broken in the lexicographic order⁵). Thus, the sets $\{S_j(x; y)\}_{j=1}^k$ define a partition of $[n]$. The AM algorithm employs an iterative scheme where one first constructs the partition $S_j^{(t)}; \dots; S_k^{(t)}$ based on the current iterates $\beta_1^{(t)}; \dots; \beta_k^{(t)}$ and then calculates the next parameter estimate $\beta_j^{(t+1)}$ by a least squares fit to the dataset $\{(x_i; y_i) : i \in S_j^{(t)}\}_{j=1}^k$. The algorithm (also described below as Algorithm 7) is, clearly, quite intuitive and presents a natural approach to solving (6.5).

As a sanity check, Lemma 28 (stated and proved in Appendix 6.5.1) shows that the global least squares estimator (6.5) is a fixed-point of this iterative scheme under a mild technical assumption.

We also note that the AM algorithm was proposed by Soh [185] in the context of estimating structured convex sets from support function measurements. It should be viewed as a generalization of a classical algorithm for (real) phase retrieval due to Fienup [95], which has been more recently analyzed in a series of papers [59, 61] for Gaussian designs. While some analyses of AM algorithms assume sample-splitting across iterations (e.g. [60, 61, 180]), we consider the more practical variant of AM run without sample-splitting, since the update (6.12a)-(6.12b) is run on the full data $(x; y)$ in every iteration.

⁵In principle, it is sufficient to define the sets $S_j(x; y); j \in [k]$ as any partition of $[n]$ having the property that $h_{i; j}^T y = \max_{u \in [k]} h_{i; u}^T y$ for every $j \in [k]$ and $i \in S_j(x; y)$; here “any” means that ties can be broken according to an arbitrary rule, and we have chosen this rule to be the lexicographic order in equation (6.11).

Algorithm 7: Alternating minimization for estimating maximum of k affine functions

Input: Data $f_i; y_i g_{i=1}^n$; initial parameter estimates $\begin{pmatrix} 0 \\ 1 \end{pmatrix}; \dots; \begin{pmatrix} 0 \\ k \end{pmatrix}$; number of iterations T .

Output: Final estimator of parameters $b_1; \dots; b_k$.

2 Initialize $t = 0$.

3 repeat

5 Compute maximizing index sets

$$S_j^{(t)} = S_j \left(\begin{pmatrix} t \\ 1 \end{pmatrix}; \dots; \begin{pmatrix} t \\ k \end{pmatrix} \right); \quad (6.12a)$$

for each $j \geq [k]$, according to equation (6.11).

7 Update

$$b_j^{(t+1)} \geq \underset{\mathbb{R}^{d+1}}{\operatorname{argmin}} \times_{i \in S_j^{(t)}} (y_i - h_i; i)^2; \quad (6.12b)$$

for each $j \geq [k]$.

8 until $t = T$;

10 Return $b_j = b_j^{(T)}$ for each $j \geq [k]$.

Initialization

The alternating minimization algorithm described above requires an initialization. While the algorithm was proposed to be run from a random initialization with restarts [55, 159], we propose to initialize the algorithm from parameter estimates that are sufficiently close to the optimal parameters. This is similar to multiple procedures to solve non-convex optimization problems in statistical settings (e.g., [99, 186]), that are based on iterative algorithms that exhibit *local* convergence to the unknown parameters. Such algorithms are typically initialized by using a moment method, which (under various covariate assumptions) returns useful parameter estimates.

Our approach to the initialization problem is similar, in that we combine a moment method with random search in a lower-dimensional space. For convenience of analysis, we split the n samples into two equal parts—assume that n is even without loss of generality—and perform each of the above steps on different samples so as to maintain independence between the two steps. The formal algorithm is presented in two parts as Algorithms 8 and 9.

In related problems [98, 169, 173, 187], a combination of a second order and third order method (involving tensor decomposition) is employed to obtain parameter estimates in one shot. Take the problem of learning generalized linear models [187] as an example; here, the analysis of the moment method relies on the link function being (at least) three times differentiable so that the population moment quantities can be explicitly computed. After showing that these expectations are closed form functions of the unknown parameters, matrix/tensor perturbation tools are then applied to show that the empirical moments concentrate about their population counterparts. However, in our setting, the max function is not differentiable, and so it is not clear that higher order moments

Algorithm 8: PCA for k -dimensional subspace initialization

Input: Data $f_j, y_i, g_{j=1}^n$.

Output: Matrix $\mathcal{U} \in \mathbb{R}^d \times k$ having orthonormal columns that (approximately) span the k dimensional subspace spanned by the vectors g_1, \dots, g_k .

2 Compute the quantities

$$\mathcal{M}_1 = \frac{2}{n} \sum_{i=1}^n y_i x_i \quad \text{and} \quad \mathcal{M}_2 = \frac{2}{n} \sum_{i=1}^n y_i x_i x_i^T - I_d; \quad (6.13)$$

and let $\mathcal{M} = \mathcal{M}_1 + \mathcal{M}_2$; here, I_d denotes the $d \times d$ identity matrix and \cdot denotes the outer product.

4 Perform the eigendecomposition $\mathcal{M} = \mathcal{P} \mathcal{B} \mathcal{P}^T$, and use the first k columns of \mathcal{P} (corresponding to the k largest eigenvalues) to form the matrix $\mathcal{U} \in \mathbb{R}^d \times k$. Return \mathcal{U} .

return reasonable estimates even in expectation since Stein’s lemma (on which many of these results rely) is not applicable⁶ in this setting. Nevertheless, we show that the second order moment returns a k -dimensional subspace that is close to the true span of the parameters $f_j, g_{j=1}^k$; the degree of closeness depends only on the geometric properties of these parameters.

Let us also briefly discuss Algorithm 9, which corresponds to performing random search in $(k + 1) \times k$ dimensional space to obtain the final initialization. In addition to the random initialization employed in step 1 of this algorithm, we also use the mean squared error on a holdout set (corresponding to samples $n=2 + 1$ through n) to select the final parameter estimates. In particular, we evaluate the error in a scale-invariant fashion; the computation of the optimal constant c in step 2 of the algorithm can be performed in closed form for each fixed index j , since for a pair of vectors $(u; v)$ having equal dimension, we have

$$\operatorname{argmin}_{c \geq 0} \|cu - vk\|^2 = \max_{\|u\| = \|v\|} \frac{\langle u, v \rangle}{\|u\| \|v\|}; 0 \leq$$

A key parameter that governs the performance of our search procedure is the number of initializations M ; we show in the sequel that it suffices to take M to be a quantity that depends only on the number of affine pieces k , and on other geometric parameters in the problem.

Our overall algorithm should be viewed as a slight variation of the AM algorithm with random restarts. It inherits similar empirical performance (see panel (b) of Figure 6.2 to follow), while significantly reducing the computational cost, since operations are now performed in ambient dimension $k + 1$, and the iterative AM algorithm is run only once overall. It also produces *provable* parameter estimates, and as we show in the sequel, the number of random initializations M can be set independently of the pair $(n; d)$. Having stated the necessary background and described our methodology, we now proceed to statements and discussions of our main results.

⁶A natural workaround is to use Stein’s lemma on the infinitely differentiable “softmax” surrogate function, but our approach to this involved balancing the estimation error (which, in turn, involves derivatives of the softmax function) and approximation error terms, and led to suboptimal dependence on the dimension.

Algorithm 9: Low-dimensional random search

Input: Data $\{x_i; y_i\}_{i=1}^n$, subspace estimate $\mathcal{V} \subseteq \mathbb{R}^{d+k}$ having orthonormal columns that (approximately) span the k dimensional subspace spanned by the vectors v_1, \dots, v_k , and number of random initializations $M \geq N$.

Output: Initial estimator of parameters $\beta_1^{(0)}, \dots, \beta_k^{(0)}$.

- Choose M random points x_j i.i.d. for $i \in [M]$ and $j \in [k]$, each uniformly from the $(k+1)$ -dimensional unit ball B^{k+1} . Let

$$x_j = \begin{bmatrix} v_j \\ 0 \end{bmatrix}$$

be a matrix in $\mathbb{R}^{(d+1) \times (k+1)}$ having orthonormal columns.

- Compute the index

$$i^* = \underset{i \in [M]}{\operatorname{argmin}} \frac{1}{n} \sum_{i=2+1}^n \min_{j \in [k]} \left(y_i - c \max_{j \in [k]} |x_{i,j}| \right)^2$$

- Return the $(d+1)$ -dimensional parameters

$$\beta_j^{(0)} = x_{i^*,j} \quad \text{for each } j \in [k].$$

6.3 Main results

In this section, we present our main theoretical results for the methodology introduced in Section 6.2.

6.3.1 Local geometric convergence of alternating minimization

We now establish local convergence results for the AM algorithm. Recall the definition of the parameters $(\beta_{\min}; \beta_{\max})$ introduced in Section 6.2, and the assumption that the covariates $\{x_i\}_{i=1}^n$ are drawn i.i.d. from the standard Gaussian distribution $\mathcal{N}(0; I_d)$. Throughout the paper, we assume that the true parameters v_1, \dots, v_k are fixed.

Theorem 12. *There exists a tuple of universal constants $(c_1; c_2)$ such that if the sample size satisfies the bound*

$$n \geq c_1 \max\{fd; 10 \log ng \max\left\{\frac{k}{3 \min}; 2 \frac{k^5}{15 \min}\right\}\};$$

then for all initializations $\beta_1^{(0)}; \dots; \beta_k^{(0)}$ satisfying the bound

$$\min_{c>0} \max_{j \neq j^0} \frac{c \beta_j^{(0)} - \beta_{j^0}^{(0)}}{k_j - j^0 k} \leq c_2 \frac{\beta_{\min}^6}{k^2} \log^{3=2} \frac{k^2}{\beta_{\min}}; \quad (6.14a)$$

the estimation error at all iterations $t \geq 1$ is simultaneously bounded as

$$\sum_{j=1}^k \|\beta_j^{(t)} - \beta_j^*\|_2 \leq \frac{3}{4} \sum_{j=1}^k \|\beta_j^{(0)} - \beta_j^*\|_2 + c_1 \frac{k d}{\beta_{\min}^3 n} \log(kd) \log(n=kd) \quad (6.14b)$$

with probability exceeding $1 - c_2 k \exp(-c_1 n \frac{\beta_{\min}^6}{k^2} + \frac{k^2}{n^t})$. Here, the positive scalar c minimizes the LHS of inequality (6.14a).

See Appendix 6.6 for a concise mathematical statement of the probability bound.

Let us interpret the various facets of Theorem 12. As mentioned before, it is a local convergence result, which requires the initialization $\beta_1^{(0)}; \dots; \beta_k^{(0)}$ to satisfy condition (6.14a). In the well-balanced case (with $\beta_{\min} = 1=k$) and treating k as a fixed constant, the initialization condition (6.14a) posits that the parameters are a constant “distance” from the true parameters. Notably, closeness is measured in a relative sense, and between pairwise differences of the parameter estimates as opposed to the parameters themselves; the intuition for this is that the initialization $\beta_1^{(0)}; \dots; \beta_k^{(0)}$ induces the initial partition of samples $S_1(\beta_1^{(0)}; \dots; \beta_k^{(0)}); \dots; S_k(\beta_1^{(0)}; \dots; \beta_k^{(0)})$, whose closeness to the true partition depends only on the relative pairwise differences between parameters, and is also invariant to a global scaling of the parameters. It is also worth noting that local geometric convergence of the AM algorithm is guaranteed uniformly from *all* initializations satisfying condition (6.14a). In particular, the initialization parameters are not additionally required to be independent of the covariates or noise, and this allows us to use the same n samples for initialization of the parameters.

Let us now turn our attention to the bound (6.14b), which consists of two terms. In the limit $t \rightarrow \infty$, the final parameters provide an estimate of the true parameters that is accurate to within the second term of the bound (6.14b). Up to a constant, this is the statistical error term

$$\sigma_{n; (d; k; \beta_{\min})} = \frac{k d}{\beta_{\min}^3 n} \log(kd) \log(n=kd) \quad (6.15)$$

that converges to 0 as $n \rightarrow \infty$, thereby providing a consistent estimate in the large sample limit. Notice that the dependence of $\sigma_{n; (d; k; \beta_{\min})}$ on the tuple $(k; d; n)$ is minimax-optimal up to the logarithmic factor $\log(n=d)$, since a matching lower bound can be proved for the linear regression problem when $k = 1$. In Proposition 16, (see Appendix 6.9.1) we also show a parametric lower bound on the minimax estimation error for general k , of the order $\frac{k d}{\beta_{\min}^3 n}$. Panel (c) of Figure 6.1 verifies in a simulation that the statistical error depends linearly on $d=n$. The dependence of the statistical error on the pair $(k; \beta_{\min})$ is more involved, and we do not yet know if these are optimal. As discussed before, a linear dependence of β_{\min} is immediate from a sample-size argument; the

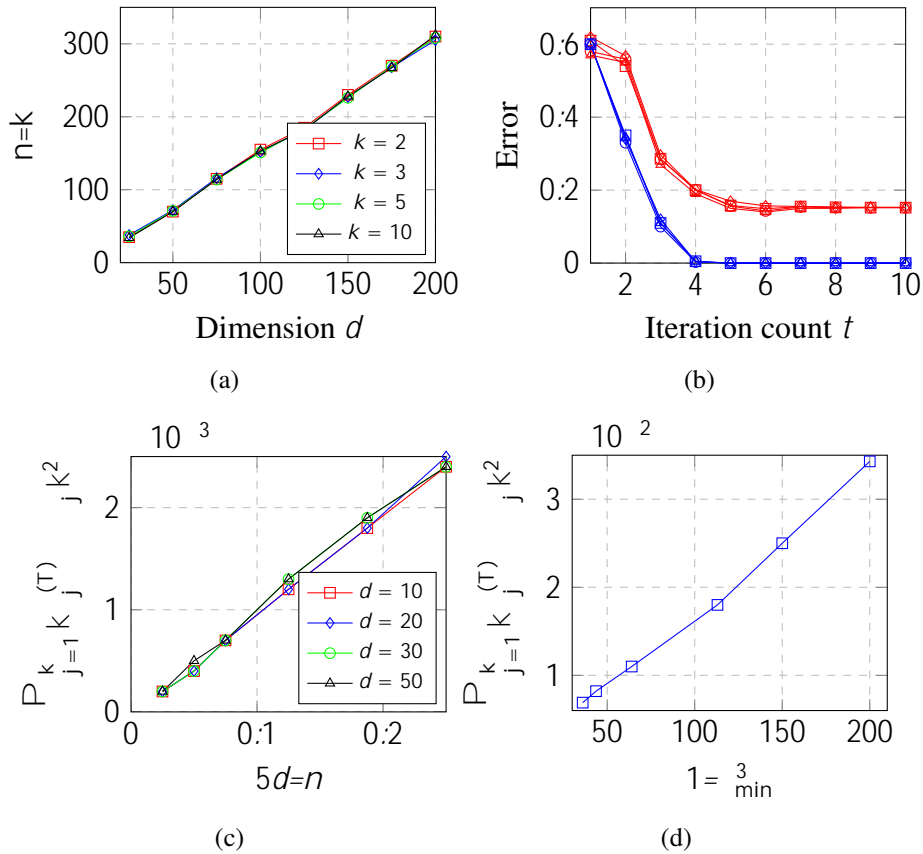


Figure 6.1: Convergence of the AM with Gaussian covariates— in panel (a), we plot the noiseless sample complexity of AM; we fix $k_i, k = 1$ for all $i \geq [k]$, $\epsilon = 0$ and $1_{\min} = 1=k$. We say θ_i is recovered if $\|\hat{\theta}_i^{(t)} - \theta_i\| \leq 0.01$. For a fixed dimension d , we run a linear search on the number of samples n , such that the empirical probability of success over 100 trials is more than 0.95, and output the least such n . In panel (b), we plot the optimization error (in blue) $\sum_{j=1}^k \|\hat{\theta}_j^{(t)} - \theta_j^{(T)}\|^2$ and the deviation from the true parameters (in red) $\sum_{j=1}^k \|\hat{\theta}_j^{(t)} - \theta_j\|^2$ over iterations t for different $d \in \{0.15; 0.25; 0.4; 0.5\}$, with $k = 5$, $d = 100$, $T = 50$ and $n = 5d$, and averaged over 50 trials. Panel (c) shows that the estimation error at $T = 50$ scales at the parametric rate $d=n$, where we have chosen a fixed $k = 5$ and $\epsilon = 0.25$. Panel (d) shows the variation of this error as a function of 1_{\min} where we fix $k = 3; d = 2; n = 10^3; \epsilon = 0.4$.

cubic dependence arises because the sub-matrices of Σ chosen over the course of the algorithm are not always well-conditioned, and their condition number scales (at most) as $\frac{2}{1_{\min}}$. In Appendix 6.9.3, we show a low-dimensional example (with $d = 2$ and $k = 3$) in which the least squares estimator incurs a parameter estimation error of the order $\frac{1}{1_{\min}^3 n}$ even when provided with the *true* partition of covariates $f_S(\theta_1; \dots; \theta_k)_{j=1}^3$. While this does not constitute an information theoretic lower bound, it provides strong evidence to suggest that our dependence on 1_{\min} is optimal at least when viewed in isolation. We verify this intuition via simulation: in panel (d) of Figure 6.1, we observe

that on this example, the error of the final AM iterate varies linearly with the quantity $1 - \frac{3}{\mu_{\min}}$.

The first term of the bound (6.14b) is an optimization error that is best interpreted in the noiseless case $\sigma = 0$, wherein the parameters $\theta_1^{(t)}; \dots; \theta_k^{(t)}$ converge at a geometric rate to the true parameters $\theta_1; \dots; \theta_k$, as verified in panel (a) of Figure 6.1. In particular, in the noiseless case, we obtain exact recovery of the parameters provided $n \geq C \frac{kd}{\mu_{\min}} \log(n=d)$. Thus, the ‘‘sample complexity’’ of parameter recovery is linear in the dimension d , which is optimal (panel (a) verifies this fact). In the well-balanced case, the dependence on k is quartic, but lower bounds based on parameter counting suggest that the true dependence ought to be linear. Again, we are not aware of whether the dependence on μ_{\min} in the noiseless case is optimal; our simulations shown in panel (a) suggests that the sample complexity depends inversely on μ_{\min} , and so closing this gap is an interesting open problem. When $\sigma > 0$, we have an overall sample size requirement

$$n \geq c \max\{fd; 10 \log ng\} \max\left\{\frac{k}{\mu_{\min}}; 2 \frac{k^5}{\mu_{\min}^2}\right\} := n_{\text{AM}}(c) \tag{6.16}$$

As a final remark, note that Theorem 12 holds under Gaussian covariates and when the true parameters $\theta_1; \dots; \theta_k$ are fixed independently of the covariates. In Chapter 7, it is shown that both of these features of the result can be relaxed, i.e., AM converges geometrically even under a milder covariate assumption, and this convergence occurs *for all* true parameters that are geometrically similar.

6.3.2 Initialization

In this section, we provide guarantees on the initialization method described in Algorithms 8 and 9 in Theorems 13 and 14, respectively.

Consider the matrices \mathcal{Y} and \mathcal{M} defined in Algorithm 8. Algorithm 8 is a moment method: we extract the top k principal components of a carefully chosen moment statistic of the data to obtain a subspace estimate $\hat{\mathcal{Y}}$. Spectral algorithms such as these have been used to obtain initializations in a wide variety of non-convex problems [56, 98, 188] to obtain an accurate estimate of the subspace spanned by the unknown parameters. It is well-known that the performance of the algorithm in recovering a k -dimensional subspace depends on $\lambda_k(\mathbb{E}[\mathcal{M}])$, which is the k -th largest eigenvalue of the population moment $\mathbb{E}[\mathcal{M}]$. We show in the proof (see the discussion following Lemma 33) that there is a strictly positive scalar γ such that

$$\lambda_k(\mathbb{E}[\mathcal{M}]) \geq \gamma \tag{6.17}$$

It should be stressed that we obtain an explicit expression for γ as a function of the various problem parameters (in equation (6.48) of the proof) that is, a priori⁷, independent of the ambient dimension d .

⁷While this may seem surprising—after all, the unknown parameters $\theta_1; \dots; \theta_k$ live in dimension d —all the interesting action is confined to the k dimensional subspace spanned by these parameters and γ is a function of the geometry induced by the parameters on this subspace.

This characterization is the main novelty of our contribution, and allows us to establish the following guarantee on the PCA algorithm. We let $U \in \mathbb{R}^{d \times k}$ denote a matrix whose orthonormal columns span the linear subspace spanned by the vectors v_1, \dots, v_k , and define the quantity

$$\delta := \max_{j \in [k]} \|v_j\|_1 + b_j \quad (6.18)$$

Theorem 13. *There is a universal constant C such that \hat{U} satisfies the bound*

$$\|\hat{U} - U\|_F^2 \leq C \frac{\delta^2 + \delta}{2} \frac{kd \log^3(nk)}{n}$$

with probability greater than $1 - Cn^{-10}$.

The proof of Theorem 13 is provided in Appendix 6.7. We have thus shown that the projection matrix $U(U)^\top$ onto the true subspace spanned by the vectors v_1, \dots, v_k can be estimated at the parametric rate via our PCA procedure. This guarantee is illustrated via simulation in panel (a) of Figure 6.2.

Let us now turn to establishing a guarantee on Algorithm 9 when it is given a (generic) subspace estimate \hat{U} as input. Since the model (6.1) is only identifiable up to a relabeling of the individual parameters, we can only hope to show that a suitably permuted set of the initial parameters is close to the true parameters. Toward that end, let P_k denote the set of all permutations from $[k] \rightarrow [k]$, and let

$$\text{dist} \left(\sum_{j=1}^n \sum_{j=1}^k \theta_j^{(0)} \mathbf{v}_j; \sum_{j=1}^k \theta_j \mathbf{v}_j \right) := \min_{P \in P_k} \sum_{j=1}^k \theta_{P(j)}^{(0)} \|\mathbf{v}_j\|^2 \quad (6.19)$$

denote the minimum distance attainable via a relabeling of the parameters. With this notation in place, we are now ready to state our result for parameter initialization. In it, we assume that the input matrix \hat{U} is fixed independently of the samples used to carry out the random search.

Theorem 14. *Let $\delta \in (0, 1)$ and $0 < r \leq \frac{5-2\delta \log^{-1}(k-\min)}{8k^3}$ denote two positive scalars. Further suppose that*

$$\|\hat{U} - U\|_{op} \leq \frac{3-2\delta}{8B_{\max} k^2}; \quad \text{and that} \quad M \leq 1 + \frac{B_{\max}}{r} \delta^2 \log(1/\delta);$$

Then there is a tuple of universal constants $(c_1; c_2)$ such that if

$$n \geq c_1 \max \left\{ d \frac{k^3}{\min} \log^2(\frac{1}{\min}); \frac{k^3}{\min} \log M \right\};$$

then

$$\min_{c>0} \text{dist} \left(\sum_{j=1}^n \sum_{j=1}^k c_j \theta_j^{(0)} \mathbf{v}_j; \sum_{j=1}^k \theta_j \mathbf{v}_j \right) \leq c_1 \frac{k}{\min} \left(4k r^2 + B_{\max}^2 \|\hat{U} - U\|_{op}^2 + \frac{2 \log M}{n} \right)$$

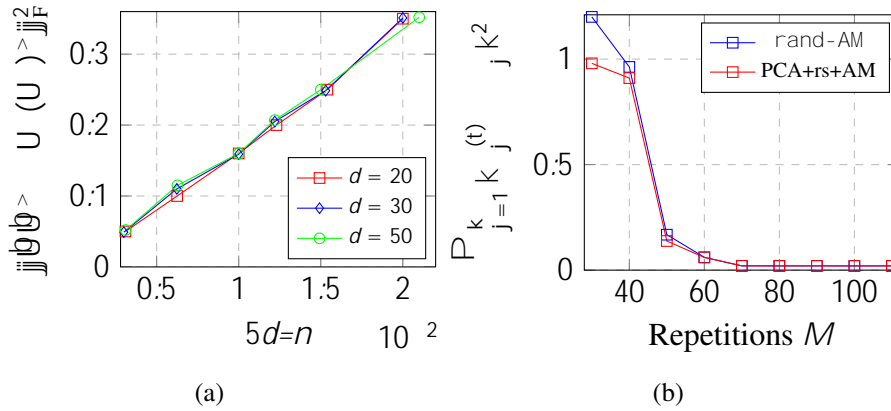


Figure 6.2: Simulation of the PCA and overall guarantees. We assume that the true parameter matrix $A = A(U)^>$ for a $\mathbb{R}^d \times k$ matrix U and an invertible $A \in \mathbb{R}^k \times k$, and that Algorithm 8 returns a subspace estimate \hat{U} . Panel (a) reveals the subspace estimation error as a function of $d=n$, which is corroborated by Theorem 13. In panel (b), we compare the performance of our overall algorithm (in red) with that of AM with repeated random initialization [154] (in blue) averaged over 50 trials. We fix $k = 3$, $d = 50$, $n = 35kd$ and $\epsilon = 0.1$. For a sufficiently large M , both schemes perform in a similar fashion.

with probability exceeding $1 - c_1 k M \exp(-c_2 n^{\frac{4}{k^4 \log^2(k=\min)}})$.

We prove Theorem 14 in Appendix 6.8. Combining Theorems 13 and 14 with some algebra then allows us to prove a guarantee for the initialization procedure that combines Algorithms 8 and 9 in sequence. In particular, fix a pair of positive scalars ϵ and $\delta \in (0, 1)$. Then combining the theorems shows that if (for an appropriately large universal constant c), we have

$M \geq 1 + c \frac{B_{\max} k^3 \log^{1=2}(k=\min)}{\min} \log(1=)$, and the sample size n is greater than

$$n_{\text{init}}(\epsilon; M; c) := c \max \left\{ \frac{k}{\min}, \frac{k^5}{\frac{5}{\min}^2} \log(k=\min) \log(M=); \frac{d \log^3(nk) \log(k=\min)}{2} \frac{k^7 B_{\max}^2}{\frac{5}{\min}^2} (\epsilon^2 + \delta^2) g \right\}; \tag{6.20}$$

then $\min_{c>0} \text{dist} \left(\hat{c}^{(0)}_{j=1}^n, \text{O}^k_{j=1}^k \right); \quad \frac{k}{j} \quad \frac{k}{j=1} \quad \epsilon^2$ with probability greater than $1 - c n^{-10}$.

Equipped with this guarantee on our initialization step, we are now in a position to state an end-to-end guarantee on our overall methodology in the next section.

6.3.3 Overall algorithmic guarantee

Assume without loss of generality that the identity permutation minimizes the distance measure dist , so that $\hat{c}^{(0)}_j$ is the estimate of the parameter c_j for each $j \in [k]$. Recall the statistical error

$n; (d; k; \min)$ defined in equation (6.15), which is, up to a constant factor, the final (squared) radius of the ball to which the AM update converges when initialized suitably, and the notation $n_{AM}(c)$ and $n_{init}(; M; c)$ from equations (6.16) and (6.20), respectively. We now state a guarantee for our overall procedure that runs Algorithms 8, 9, and 7 in that sequence; we omit the proof since it follows by simply putting together the pieces from Theorem 12 and the discussion above.

Corollary 2. *There exist universal constants c_1 and c_2 such that for each $\geq 2(0; 1)$, if*

$$M \geq 1 + c_1 \frac{B_{\max} k^4 \log^{1+2}(k = \min)}{\min} \log(1 =); n \geq \max \{ n_{init} \geq c_2 \frac{6}{k^2}; c_1; M \}; n_{AM}(c_1)$$

and $T_0 = c_1 \log \frac{1}{n; (d; k; \min)} ;$

then the combined algorithm satisfies, simultaneously for all $T \geq T_0$, the bound

$$\Pr \left(\sum_{j=1}^k \|\hat{x}_j^{(T)}\|_2^2 \leq c_1 n; (d; k; \min) + c_1 n^{10} + k \exp \left(-c_2 n \frac{6}{k^2} \right) + \frac{k^2}{n^7} \right) \geq 1 - \epsilon$$

We thus obtain, an algorithm that when given a number of samples that is near-linear in the ambient dimension, achieves the rate $n; (d; k; \min) = \frac{2kd}{3 \min n} \log(kd) \log(n=kd)$ of estimation of all kd parameters in squared ℓ_2 norm. This convergence is illustrated in simulation in Figure 6.2, in which we choose $k = 3, d = 50$ and $n = 35kd$. Interestingly, panel (b) of this figure shows that our provable multi-step algorithm has performance similar to the algorithm that runs AM with repeated random initializations.

The computational complexity of our overall algorithm (with exact matrix inversions) is given by $O(knd^2 \log \frac{1}{n; (d; k; \min)} + Mnd)$, where we also assume that the k top eigenvectors of the matrix \mathcal{M} are computed exactly in Algorithm 8. This guarantee can also be extended to the case where the linear system is solved up to some numerical precision by (say) a conjugate gradient method and the eigenvectors of \mathcal{M} are computed using the power method, thereby reducing the computational complexity. Such an extension is standard and we do not detail it here.

6.3.4 Proof sketch and technical challenges

Let us first sketch, at a high level, the ideas required to establish guarantees on the AM algorithm. We need to control the iterates of the AM algorithm without sample-splitting across iterations, and so the iterates themselves are random and depend on the sequence of random variables $(\epsilon_i; \delta_i)_{i=1}^n$. A popular and recent approach to handling this issue in related iterative algorithms (e.g., [99]) goes through two steps: first, the population update, corresponding to running (6.12a)-(6.12b) in the case $n \rightarrow \infty$, is analyzed, after which the random iterates in the finite-sample case are shown to be close to their (non-random) population counterparts by using concentration bounds for the associated empirical process. The main challenge in our setting is that the population update is quite non-trivial

to write down since it involves a delicate understanding of the geometry of the covariate distribution induced by the maxima of affine functions. We thus resort to handling the random iterates directly, thereby sidestepping the calculation of the population operator entirely.

In order to convey the principal difficulties associated with our approach, let us present a bound on the error obtained after running a single step of the algorithm, starting at the parameters β_1, \dots, β_k and obtaining, as a result of one step of the algorithm, the parameters $\beta_1^+, \dots, \beta_k^+$. We use the shorthand notation $S_j := S_j(\beta_1, \dots, \beta_k)$; and let $P_{J(\beta_1, \dots, \beta_k)}$ denote the projection matrix onto the range of the matrix S_j .

Let y_i denote the vector with entry i given by $\max_{u \in [2[k]]} h_{i; u} \cdot i$. We have

$$\begin{aligned} k \|S_j(\beta_1^+, \dots, \beta_k^+) - S_j\|^2 &= k P_{J(\beta_1, \dots, \beta_k)} Y_{S_j} - S_j \|^2 \\ &= k P_{J(\beta_1, \dots, \beta_k)} Y_{S_j} + P_{J(\beta_1, \dots, \beta_k)} S_j - S_j \|^2 \\ &\leq 2k P_{J(\beta_1, \dots, \beta_k)} (Y_{S_j} - S_j)^2 + 2k P_{J(\beta_1, \dots, \beta_k)} S_j^2 \\ &\leq 2k Y_{S_j} - S_j \|^2 + 2k P_{J(\beta_1, \dots, \beta_k)} S_j^2; \end{aligned} \tag{6.21}$$

where we have used the fact that the projection operator is non-expansive on a convex set.

Let

$$f_{h_{i; \cdot} \cdot i} = \max_{u \in [2[k]]} g := h_{i; u} \cdot i = \max_{u \in [2[k]]} h_{i; u} \cdot i; \text{ for each } i \in [n]; \cdot \in [k]$$

denote a convenient shorthand for these events. The first term on the RHS of inequality (6.21) can be written as

$$\sum_{i \in S_j} (y_i - h_{i; j} \cdot i)^2 \leq \sum_{i=1}^n \sum_{j^0 \in J} \mathbf{1}_{h_{i; j} \cdot i = \max} \text{ and } h_{i; j^0} \cdot i = \max_{j^0} h_{i; j^0} \cdot i^2;$$

where the inequality accounts for ties. Each indicator random variable is bounded, in turn, as

$$\mathbf{1}_{h_{i; j} \cdot i = \max} \text{ and } h_{i; j^0} \cdot i = \max_{j^0} \leq \mathbf{1}_{h_{i; j} \cdot i \geq h_{i; j^0} \cdot i} \text{ and } h_{i; j^0} \cdot i \geq h_{i; j} \cdot i \\ = \mathbf{1}_{h_{i; j} \cdot i \geq h_{i; j^0} \cdot i} \geq 0;$$

Switching the order of summation yields the bound

$$\sum_{i \in S_j} (y_i - h_{i; j} \cdot i)^2 \leq \sum_{j^0 \in J} \sum_{i=1}^n \mathbf{1}_{h_{i; j} \cdot i \geq h_{i; j^0} \cdot i} \geq 0 \cdot h_{i; j} \cdot i^2;$$

Recalling our notation for the minimum eigenvalue of a symmetric matrix, the LHS of inequality (6.21) can be bounded as

$$k \|S_j(\beta_1^+, \dots, \beta_k^+) - S_j\|^2 \leq \min_{j \in J} \lambda_{\min}(S_j) \|k \beta_j^+ - \beta_j\|^2;$$

Putting together the pieces yields, for each $j \geq [k]$, the pointwise bound

$$\frac{1}{2} \min_{\substack{\tilde{S}_j \succeq S_j \\ \times \times^n}} \left(k \sum_{j^0 \in J} \sum_{i=1}^n \mathbf{1} \left(h_{i; j} - j^0 i \right) \left(h_{i; j} - j^0 i \right) + k P_{J(\{1, \dots, k\})} S_j k^2 \right) \tag{6.22}$$

Up to this point, note that all steps of the proof were deterministic. Observe from equation (6.22) that in order to obtain an error bound on the next parameter, we need to control three distinct quantities: (a) the noise term $k P_{J(\{1, \dots, k\})} S_j k^2$, (b) the prediction error of the noiseless problem, given by a pairwise sum of terms of the form $\mathbf{1} \left(h_{i; j} - j^0 i \right) \left(h_{i; j} - j^0 i \right)$, and (c) the minimum eigen value of the covariate matrix restricted to the set S_j , denoted by $\min_{\tilde{S}_j \succeq S_j} \lambda$. Since the set S_j is in itself random and depends on the pair $(j; \cdot)$ (since the current parameters were obtained over the course of running the algorithm), obtaining such a bound is especially challenging.

For step (a)—handled by Lemma 29—we apply standard concentration bounds for quadratic forms of sub-Gaussian random variables in conjunction with bounds on the *growth functions* of multi-class classifiers [189]. Crucially, this affords a uniform bound on the noise irrespective of which iterate the alternating minimization update is run from. To show step (b)—in Lemma 30—we generalize a result of Waldspurger [59]. Finally, the key difficulty in step (c) is to control the spectrum of random matrices, rows of which are drawn from (randomly) truncated variants of the Gaussian distribution. The expectation of such a random matrix can be characterized by appealing to tail bounds on the non-central χ^2 distribution, and the Gaussian covariate assumption additionally allows us to show that an analogous result holds for the random matrix with high probability (see Lemma 31). Here, our initialization condition is crucial: the aforementioned singular value control suffices for the sub-matrices formed by the *true* parameters, and we translate these bounds to the sub-matrices generated by random parameters by appealing to the fact that the initialization is sufficiently close to the truth.

Having discussed our proof of the AM update in some detail, let us now turn to a brief discussion of the techniques used to prove Theorems 13 and 14. As mentioned before, our proof of Theorem 13 relies on a lower bound on the eigengap of the population moment. We obtain such a lower bound by appealing to classical moment calculations for suitably truncated Gaussian distributions [182]. Translating these calculations into an eigengap is quite technical, and involves the isolation of many properties of the population moments that may be of independent interest. As briefly alluded to in Section 6.2, the heart of the technical difficulty is due to the fact that that \max function is not differentiable, and so moments cannot be calculated by repeated applications of Stein’s lemma like in related problems [98, 188, 190].

In order to establish Theorem 14, we crucially use the scale-invariance property of the initialization along with some arguments involving empirical process theory to show that the goodness-of-fit statistic employed in the algorithm is able to isolate a good initialization. Establishing these bounds requires us to relate the prediction and estimation errors in the problem (in Lemma 43), which may be of independent interest.

6.4 Discussion

We conclude with short discussions of prediction error guarantees, a comparison with adaptivity in convex regression, related models, and future directions.

6.4.1 Guarantees on prediction error

While our principal focus in this paper was on estimation of the unknown parameters $f_j; b_j g_{j=1}^k$, the complementary question of prediction error is also interesting and important. In particular, suppose that we produce the max-affine function estimate $b^{(MA)}$ given by $b^{(MA)}(x) := \max_{j \in [k]} (hx; b_j i + \theta_j)$ for each $x \in \mathbb{R}^d$, and measure its performance via the prediction error

$$\frac{1}{n} \sum_{i=1}^n (b^{(MA)}(x_i) - (x_i))^2; \tag{6.23}$$

where $(x) = \max_{j \in [k]} (hx; b_j i + b_j)$ denotes the ‘‘true’’ function. When $(; b)$ belongs to the subclass of k -piece affine functions induced by parameters in the set $B_{vol}(\min; ;)$ and the covariates are drawn from a Gaussian distribution, our results imply (via Theorem 12, and by using Lemma 43 to translate our estimation error guarantee into a prediction error guarantee) the rate

$$\frac{1}{n} \sum_{i=1}^n (b^{(MA)}(x_i) - (x_i))^2 \leq C(\min; ;) \frac{kd}{n} \log(kd) \log(n=kd); \tag{6.24}$$

At least in principle, an explicit dependence on \min should not be expected in the prediction error, since if a particular pair of parameters $(; b)$ attains the maximum extremely rarely (resulting in a small value of \min), then we may simply drop these parameters from the estimate (and estimate the function with the maximum of the remaining $k - 1$ pieces) without affecting the prediction error significantly. Indeed the minimax risk of prediction (without any requirements of computational efficiency) is known to be independent of the geometry of the problem instance (see, e.g., [158]).

We also note that polynomial-time algorithms with small prediction error are known, without any dependence on \min . In particular, [191, Theorem 1.8] shows that the sample complexity for obtaining ϵ -accurate estimates in prediction error is bounded by $n \leq \exp(c_1(k=)^{\log k} d^{c_2})$ for absolute constants c_1 and c_2 . While the dependence on both \min and d can likely be improved⁸, these results provide additional evidence that the prediction error is much less sensitive to the geometry of the instance than the estimation error considered in this paper.

6.4.2 Comparison with algorithms for convex regression

As mentioned earlier, the most standard estimator in convex regression is the convex least squares estimator $b^{(ls)}$ defined as in (6.6) which can be computed efficiently as shown in [162, 164]. The

⁸Note that unlike our paper, this work makes only boundedness assumptions on the covariates, and their focus is not on achieving the optimal dimension/samoke size dependence.

performance of $\hat{b}^{(ls)}$ in the max-affine regression model (6.1) has been the subject of some interest in the literature on adaptivity of shape-constrained estimators (see [192] for an overview of results of this type). These results mainly focus on the prediction error:

$$\frac{1}{n} \sum_{i=1}^n (\hat{b}^{(ls)}(x_i) - f(x_i))^2$$

as opposed to estimation of the parameters $\beta_j; j = 1, \dots, k$ which is our main focus. There is actually no natural way of obtaining parameter estimates from $\hat{b}^{(ls)}$ as $\hat{b}^{(ls)}$ will typically be a maximum of a strictly larger than k number of affine functions. Let us now compare our results with the existing results on the prediction error of the convex least squares estimator. When $d = 1$, it has been showed by [193, 194] that

$$\frac{1}{n} \sum_{i=1}^n (\hat{b}^{(ls)}(x_i) - f(x_i))^2 \leq \frac{k \log n}{n}$$

with high probability assuming that $f, x_i, g_{i=1}^n$ are uniformly spaced on the interval $[0; 1]$. For $d \geq 2$, Han and Wellner [152] studied the adaptivity properties of $\hat{b}^{(bls)}$ which is the least squares estimator over the class of *bounded* convex functions which is different from $\hat{b}^{(ls)}$ and computationally tricky to compute. However, [152] showed that unless $d \geq 4$,

$$\frac{1}{n} \sum_{i=1}^n (\hat{b}^{(bls)}(x_i) - f(x_i))^2 \leq C_m n^{-4+d} (\log n)^{d+4}$$

with high probability assuming that the covariates are drawn from a distribution supported on a convex polytope with m simplices (the constant pre-factor C_m depends on m). Comparing these two results with our result on the prediction error (6.24), we see that when $d = 1$, our results in the prediction error are strictly weaker than those of prior work [193, 194], but as soon as $d \geq 2$, they are significantly stronger than existing adaptivity results [152], at least for a sub-class of k -affine functions. We emphasize once again that the focus of the body of work differs from ours, and so the comparison presented above is necessarily incomplete.

A parallel line of work (including our own) eschews the c-LSE (and its variants) entirely and pursues a different avenue to alleviate the curse of dimensionality⁹, by directly fitting convex functions consisting of a certain number of affine pieces [55], or more broadly, by treating the number of affine functions as a tuning parameter to be chosen in a data-dependent fashion via cross-validation [153]. Hannah and Dunson [153] showed that performing estimation under a carefully chosen sequence of models of the form (6.1) via their “convex adaptive partitioning”, or CAP estimator is able to obtain consistent prediction rates for general convex regression problems. However, it is unclear if the CAP estimator is able to avoid the curse of dimensionality in the special case when the true function is k -piece affine.

⁹Note that setting $k = n^{d-(4+d)}$, we can (essentially) recover the entire class of convex functions from the maxima of k affine functions (see, e.g., Balázs [154]), so interesting parametric structure is only expected to emerge when k is essentially constant, or grows very slowly with n .

6.4.3 Related models

Models closely related to (6.1) also appear in second price auctions, where an item having d features is bid on and sold to the highest bidder at the second highest bid [195, 196]. Assuming that each of k user groups bids on an item and that each bid is a linear function of the features, one can use a variant of the model (6.1) with the \max function replaced by the second order statistic to estimate the individual bids of the user groups based on historical data. Another related problem is that of multi-class classification [189], in which one of k labels is assigned to each sample based on the argmax function, i.e., for a class of functions F , we have the model $Y = \operatorname{argmax}_{1 \leq j \leq k} f_j(X)$ for j distinct functions $f_1, \dots, f_k \in F$. When F is the class of linear functions based on d features, this can be viewed as the “classification” variant of our regression problem.

The model (6.1) can also be seen as a special case of multi-index models [197, 198] as well as mixture-of-experts models [199, 200]. Multi-index models are of the form $Y = g(h_1(X); \dots; h_k(X)) + \epsilon$ for an *unknown* function g and this function g is taken to be the $\max(\cdot)$ function in the model (6.1). In the mixture-of-experts model, the covariate space is partitioned into k regions via certain *gating functions*, and the observation model is given by k distinct regression functions: one on each region. The model (6.1) is clearly a member of this class, since the $\max(\cdot)$ function implicitly defines a partition of \mathbb{R}^d depending on which of the k linear functions of X attains the maximum, and on each of these partitions, the regression function is linear in X .

6.4.4 Open Problems

In this paper, we analyzed a natural alternating minimization algorithm for estimating the maximum of unknown affine functions, and established that it enjoys local linear convergence to a ball around the optimal parameters. We also proposed an initialization based on PCA followed by random search in a lower-dimensional space. An interesting open question is if there are other efficient methods besides random search that work just as well post dimensionality reduction. Another interesting question has to do with the necessity of dimensionality reduction: in simulations (see, e.g., Figure 6.2), we have observed that if the AM algorithm is repeatedly initialized in $(d + 1)$ -dimensional space without dimensionality reduction, then the number of repetitions required to obtain an initialization from which it succeeds (with high probability) is similar to the number of repetitions required after dimensionality reduction. This suggests that our (sufficient) initialization condition (6.14a) may be too stringent, and that the necessary conditions on the initialization to ensure convergence of the AM algorithm are actually much weaker. We leave such a characterization for future work, but note that some such conditions must exist: the AM algorithm when run from a single random initialization, for instance, fails with constant probability when $k \geq 3$. Understanding the behavior of the randomly initialized AM algorithm is also an open problem in the context of phase retrieval [59, 201].

We note that once again that the Gaussian assumption made in this paper for convenience of analysis can be relaxed to allow (for instance) log-concave covariate distributions, which includes the uniform distribution on $[-1, 1]^d$ common in nonparametric statistics. Such an extension requires

significant technical effort and the structure of the proof also changes slightly; simultaneously, the dependence of the eventual error bounds on the parameter γ_{\min} is also different in the more general setting. In particular, Lemmas 30-32 in the current paper must be extended, and this requires, among other things, an analysis of random matrices whose rows are drawn from a (truncated) small-ball distribution. Our companion paper [202] is also concerned with the *universal* setting in which guarantees are proved uniformly over all choices of parameters once the covariates have been drawn, in contrast to the setting of the current paper in which parameters are fixed in advance. Universal guarantees are commonly sought out in statistical signal processing applications, including phase retrieval [56].

In the broader context of max-affine estimation, it is also interesting to analyze other non-convex procedures (e.g. gradient descent) to obtain conditions under which they obtain accurate parameter estimates. The CAP estimator of Hannah and Dunson [153] and the adaptive max-affine partitioning algorithm of Balázs [154] are also interesting procedures for estimation under these models, and it would be interesting to analyze their performance when the number of affine pieces k is fixed and known. For applications in which the dimension d is very large, it is also interesting to study the model with additional restrictions of sparsity on the unknown parameters—such problems are known to exhibit interesting statistical-computational gaps even in the special case of sparse phase retrieval (see, e.g., Cai et al. [203]). We also note that in practice, and especially for convex regression, the parameter k would be unknown and must be estimated (say) via cross-validation. Understanding the behavior of such a two-stage estimator is an important direction of future work.

Appendix

We now present proofs of our main results. We assume throughout that the sample size n is larger than some universal constant. Values of constants $c; c_1; c^d; \dots$ may change from line to line. Statements of our theorems, for instance, minimize the number of constants by typically using one of these to denote a large enough constant, and another to denote a small enough constant.

6.5 Proofs

6.5.1 Technical results concerning the global LSE

In this section, we provide a proof of the existence of the global least squares estimator that was stated in the main text. We also state and prove a lemma that shows that the global LSE is a fixed point of the AM update under a mild technical condition.

6.5.2 Proof of Lemma 27

Fix data $(x_1, y_1), \dots, (x_n, y_n)$ and let

$$L(\alpha_1, \dots, \alpha_k) := \sum_{i=1}^n y_i \max_{j \in [k]} h_{i; j} \alpha_j^2$$

denote the objective function in (6.5) with $\alpha_j := (x_j, 1)$. The goal is to show that a global minimizer of $L(\alpha_1, \dots, \alpha_k)$ over $\alpha_1, \dots, \alpha_k \in \mathbb{R}^{d+1}$ exists. For $\alpha_1, \dots, \alpha_k \in \mathbb{R}^{d+1}$, let S_1, \dots, S_k denote a fixed partition of $[n]$ having the property that

$$h_{i; j} \alpha_j = \max_{u \in [k]} h_{i; u} \alpha_u \quad \text{for every } j \in [k] \text{ and } i \in S_j.$$

Also, let b_1, \dots, b_k denote the solution to the following constrained least squares problem:

$$\begin{aligned} & \text{minimize} && \sum_{j=1}^k \sum_{i \in S_j} (y_i - h_{i; j} \alpha_j)^2 \\ & \text{subject to} && h_{i; j} \alpha_j = h_{i; u} \alpha_u; u, j \in [k]; i \in S_j. \end{aligned}$$

Note that the above quadratic problem is feasible as $\alpha_1, \dots, \alpha_k$ satisfies the constraint and, consequently, b_1, \dots, b_k exists uniquely for every $\alpha_1, \dots, \alpha_k \in \mathbb{R}^{d+1}$. Note further that, by construction,

$$L(b_1, \dots, b_k) = L(\alpha_1, \dots, \alpha_k)$$

and that the set

$$\mathcal{O} := \{(b_1, \dots, b_k) : \alpha_1, \dots, \alpha_k \in \mathbb{R}^{d+1}\}$$

is finite because b_1, \dots, b_k depends on $\alpha_1, \dots, \alpha_k$ only through the partition S_1, \dots, S_k and the number of possible such partitions of $[n]$ is obviously finite. Finally, it is evident that

$$(b_1^{(s)}, \dots, b_k^{(s)}) = \operatorname{argmin}_{(\alpha_1, \dots, \alpha_k) \in \mathcal{O}} L(\alpha_1, \dots, \alpha_k)$$

is a global minimizer of $L(\alpha_1, \dots, \alpha_k)$ as

$$L(b_1^{(s)}, \dots, b_k^{(s)}) = L(b_1, \dots, b_k) = L(\alpha_1, \dots, \alpha_k)$$

for every $\alpha_1, \dots, \alpha_k$. This concludes the proof of Lemma 27.

6.5.3 Fixed point of AM update

The following lemma establishes that the global LSE is a fixed point of the AM update under a mild technical condition.

Lemma 28. *Consider the global least squares estimator (6.5). Suppose that the k values $h_{i,j}^{(s)}$ for $j = 1, \dots, k$ are distinct for each $i \geq [n]$. Then*

$$b_j^{(s)} \geq \operatorname{argmin}_{\mathbb{R}^{d+1}} \sum_{i \in S_j(b_1^{(s)}, \dots, b_k^{(s)})} (y_i - h_{i,j}^{(s)})^2 \quad \text{for every } j \in [k]; \quad (6.25)$$

Proof. It is clearly enough to prove (6.25) for $j = 1$. Suppose that $b_1^{(s)}$ does not minimize the least squares criterion over $S_1(b_1^{(s)}, \dots, b_k^{(s)})$. Let

$$b_1^{(s)} \geq \operatorname{argmin}_{\mathbb{R}^{d+1}} \sum_{i \in S_1(b_1^{(s)}, \dots, b_k^{(s)})} (y_i - h_{i,1}^{(s)})^2$$

be any other least squares minimizer over $S_1(b_1^{(s)}, \dots, b_k^{(s)})$ and let, for $\epsilon > 0$,

$$e_1 := b_1^{(s)} + \epsilon \frac{b_1^{(s)} - b_1^{(s)}}{\|b_1^{(s)} - b_1^{(s)}\|}$$

When $\epsilon > 0$ is sufficiently small, we have

$$S_j(e_1, b_2^{(s)}, \dots, b_k^{(s)}) = S_j(b_1^{(s)}, \dots, b_k^{(s)}) \quad \text{for every } j \in [k]$$

due to the no ties assumption and the fact that e_1 and $b_1^{(s)}$ can be made arbitrarily close as ϵ becomes small. Thus, if

$$U(1, \dots, k) := \sum_{i=1}^n y_i \max_{j \in [k]} h_{i,j}^{(s)} j^2 = \sum_{j \in [k]} \sum_{i \in S_j(1, \dots, k)} (y_i - h_{i,j}^{(s)})^2;$$

then

$$\begin{aligned} & U(e_1, b_2^{(s)}, \dots, b_k^{(s)}) \\ &= \sum_{i \in S_1(e_1, b_2^{(s)}, \dots, b_k^{(s)})} (y_i - h_{i,1}^{(s)} e_1)^2 + \sum_{j \in [k]} \sum_{i \in S_j(e_1, b_2^{(s)}, \dots, b_k^{(s)})} (y_i - h_{i,j}^{(s)} b_j^{(s)})^2 \\ &= \sum_{i \in S_1(b_1^{(s)}, b_2^{(s)}, \dots, b_k^{(s)})} (y_i - h_{i,1}^{(s)} e_1)^2 + \sum_{j \in [k]} \sum_{i \in S_j(b_1^{(s)}, b_2^{(s)}, \dots, b_k^{(s)})} (y_i - h_{i,j}^{(s)} b_j^{(s)})^2 \\ &< \sum_{i \in S_1(b_1^{(s)}, b_2^{(s)}, \dots, b_k^{(s)})} (y_i - h_{i,1}^{(s)} b_1^{(s)})^2 + \sum_{j \in [k]} \sum_{i \in S_j(b_1^{(s)}, b_2^{(s)}, \dots, b_k^{(s)})} (y_i - h_{i,j}^{(s)} b_j^{(s)})^2 \\ &= U(b_1^{(s)}, b_2^{(s)}, \dots, b_k^{(s)}) \end{aligned}$$

where the strict inequality above comes from the fact that e_1 is closer to the least squares solution $b_1^{(s)}$ compared to $b_1^{(s)}$. This leads to a contradiction as the criterion function is smaller than its value at a global minimizer, thereby concluding the proof. \square

6.6 Proof of Theorem 12

Let us begin by introducing some shorthand notation, and providing a formal statement of the probability bound guaranteed by the theorem. For a scalar w , vectors $u \in \mathbb{R}^d$ and $v = (u; w) \in \mathbb{R}^{d+1}$, and a positive scalar r , let $B_v(r) = \{v' \in \mathbb{R}^{d+1} : \frac{kv'v'k}{ku'k} \leq r\}$; and let

$$I(r; j_{j=1}^k) = \{1, \dots, k \in \mathbb{R}^{d+1} : \exists c > 0 : c(i, j) \in B_{i, j}(r) \text{ for all } 1 \leq i \neq j \leq k\}$$

Also, use the shorthand

$$\#_t(r; j_{j=1}^k) := \sup_{\substack{(0) \dots (t) \\ k \in I(r; j_{j=1}^k)}} \sum_{j=1}^k k_j^{(t)} j^2 \leq \frac{3}{4} \sum_{j=1}^k kc_j^{(0)} j^2; \text{ and}$$

$$N_n(d; k; \min) := 2 \frac{kd}{\min n} \log(kd) \log(n=kd)$$

to denote the error tracked over iterations (with c denoting the smallest $c > 0$ such that $c(i, j) \in B_{i, j}(r)$ for all $1 \leq i \neq j \leq k$), and a proxy for the final statistical rate, respectively.

Theorem 12 states that there are universal constants c_1 and c_2 such that if the sample size obeys the condition $n \geq n_{AM}(c_1)$, then we have

$$\Pr \max_{t=1} \#_t \leq c_2 \frac{6}{k^2}; \quad j_{j=1}^k \leq c_1 N_n(d; k; \min) \leq c_2 k \exp \left(-c_1 n \frac{6}{k^2} \right) + \frac{k^2}{n^7}; \tag{6.26}$$

Let us assume, without loss of generality, that the scalar c above is equal to 1. It is convenient to state and prove another result that guarantees a one-step contraction, from which Theorem 12 follows as a corollary. In order to state this result, we assume that one step of the alternating minimization update (6.12a)-(6.12b) is run starting from the parameters $f_{j, j=1}^k$ to produce the next iterate $f_{j, j=1}^{+k}$. We use the shorthand

$$V_{ij} = |i - j|;$$

$$V_{ij} = |i - j|; \text{ and}$$

$$V_{ij}^+ = |i^+ - j^+|;$$

Also recall the definitions of the geometric quantities $(;)$. The following proposition guarantees the one step contraction bound.

Proposition 15. *There exist universal constants c_1 and c_2 such that*

(a) *If the sample size satisfies the bound $n \geq c_1 \max \frac{dk}{\min}; \log n \geq \frac{k^2}{\min}$, then for all parameters $f_{j, j=1}^k$ satisfying*

$$\max_{1 \leq j \neq j^0 \leq k} \frac{V_{j, j^0}}{k_j} \frac{V_{j, j^0}}{j^0 k} \log^{3=2} \frac{k_j}{V_{j, j^0}} \frac{j^0 k}{V_{j, j^0}} \leq c_2 \frac{6}{k^2}; \tag{6.27a}$$

we have, simultaneously for all pairs $1 \leq j \leq k$, the bound

$$\frac{V_{j,\cdot}^+ \cdot V_{j,\cdot}^2}{k_j \cdot k^2} \leq \max_{j^0=1} \frac{d}{3 \min n} \cdot \frac{1}{4k} \sum_{j^0=1}^k \frac{V_{j,j^0} \cdot V_{j,j^0}^2}{k_j \cdot j^0 k^2} + \frac{V_{\cdot,j^0} \cdot V_{\cdot,j^0}^2}{k \cdot j^0 k^2} + c_1 \frac{k d}{3 \min n} \log(n=d) \tag{6.27b}$$

with probability exceeding $1 - c_1 k \exp(-c_2 n \frac{k^2}{6 \min} + \frac{k^2}{n^7})$.

(b) If the sample size satisfies the bound $n \geq c_1 \max_{j^0=1} \frac{dk}{3 \min} \cdot \log n \frac{k^2}{6 \min}$, then for all parameters $f_j, g_{j=1}^k$ satisfying

$$\max_{1 \leq j \leq j^0 \leq k} \frac{V_{j,j^0} \cdot V_{j,j^0}}{k_j \cdot j^0 k} \log^{3=2} \frac{k_j \cdot j^0 k}{V_{j,j^0} \cdot V_{j,j^0}} \leq c_2 \frac{6 \min}{k^2}; \tag{6.28a}$$

we have the overall estimation error bound

$$\sum_{i=1}^k \frac{V_{j,\cdot}^+ \cdot V_{j,\cdot}^2}{k_j \cdot k^2} \leq \frac{3}{4} \sum_{i=1}^k \frac{V_{j,\cdot}^+ \cdot V_{j,\cdot}^2}{k_j \cdot j^0 k^2} + c_1 \frac{k d}{3 \min n} \log(k) \log(n=dk) \tag{6.28b}$$

with probability exceeding $1 - c_1 k \exp(-c_2 n \frac{k^2}{6 \min} + \frac{k^2}{n^7})$.

Let us briefly comment on why Proposition 15 implies Theorem 12 as a corollary. Clearly, equations (6.28a) and (6.28b) in conjunction show that the estimation error decays geometrically after running one step of the algorithm. The only remaining detail to be verified is that the next iterates $f_j^+, g_{j=1}^k$ also satisfy condition (6.27a) provided the sample size is large enough; in that case, the one step estimation bound (6.28b) can be applied recursively to obtain the final bound (6.14b).

With the constant c_2 from the proposition, let r_b be the largest value in the interval $[0; e^{-3=2}]$ such that $r_b \log^{3=2}(1=r_b) \leq c_2 \frac{6 \min}{k^2}$. Similarly, let r_a be the largest value in the interval $[0; e^{-3=2}]$ such that $r_a \log^{3=2}(1=r_a) \leq c_2 \frac{6 \min}{k^2}$. Bounds on both of these values will be used repeatedly later on.

Assume that the current parameters satisfy the bound (6.27a). Choosing $n \geq 4 \frac{d}{3 \min}$ and applying inequality (6.27b), we have, for each pair $1 \leq j \leq k$, the bound

$$\frac{V_{j,\cdot}^+ \cdot V_{j,\cdot}^2}{k_j \cdot k^2} \leq \frac{1}{4k} \sum_{j^0=1}^k \frac{V_{j,j^0} \cdot V_{j,j^0}^2}{k_j \cdot j^0 k^2} + \frac{V_{\cdot,j^0} \cdot V_{\cdot,j^0}^2}{k \cdot j^0 k^2} + c_1 \frac{k d}{3 \min n} \log(n=d) + \frac{1}{2} r_a^2 + c_1 \frac{k d}{3 \min n} \log(n=d);$$

Further, if $n \geq C \frac{k d}{3 \min r_a^2}$ for a sufficiently large constant C , we have

$$\frac{V_{j,\cdot}^+ \cdot V_{j,\cdot}^2}{k_j \cdot k^2} \leq r_a^2.$$

Thus, the parameters $\{s_j\}_{j=1}^k$ satisfy inequality (6.27a) for the sample size choice required by Theorem 12. Finally, noting, for a pair of small enough scalars $(a; b)$, the implication

$$a \leq \frac{b}{2} \log^{3=2}(1=b) \Rightarrow a \log^{3=2}(1=a) \leq b;$$

and adjusting the constants appropriately to simplify the probability statement completes the proof of the theorem. It now remains to establish Proposition 15.

6.6.1 Proof of Proposition 15

Recall that we denote by

$$S_j(\{1; \dots; k\}) := \{1 \leq i \leq n : h_{i,j} = \max_{1 \leq u \leq k} (h_{i,u})\};$$

the indices of the rows for which s_j attains the maximum, and we additionally keep these sets disjoint by breaking ties lexicographically. To lighten notation, we use the shorthand

$$J(\{1; \dots; k\}) := \bigcup_{j=1}^k S_j(\{1; \dots; k\});$$

Recall the notation

$$B_v(r) = \{v \in \mathbb{R}^{d+1} : \frac{kv - v \cdot k}{ku} \leq r\}$$

introduced before, and the definitions of the pair of scalars $(r_a; r_b)$. To be agnostic to the scale invariance of the problem, we set $c = 1$ and define the set of parameters

$$I(r) = \{1 \leq i \leq n : v_{i,j} \in B_{v_{i,j}}(r) \text{ for all } 1 \leq j \leq k\};$$

and use the shorthand $I_a := I(r_a)$ and $I_b := I(r_b)$, to denote the set of parameters satisfying conditions (6.27a) and (6.28a), respectively,

Finally, recall the deterministic bound (6.22) established in Section 6.3.4, restated below for convenience.

$$\frac{1}{2} \min_{\{s_j\}_{j=1}^k} \sum_{j=1}^k s_j^2 \sum_{j^0: j^0 \in J} \sum_{i=1}^n \mathbf{1}_{\{h_{i,j^0} \geq s_{j^0}\}} \sum_{i=1}^n \mathbf{1}_{\{h_{i,j^0} \leq 0\}} + kP_{J(\{1; \dots; k\})} \sum_{j=1}^k s_j^2;$$

It suffices to show high probability bounds on the various quantities appearing in this bound. First, we claim that the noise terms are uniformly bounded as

$$\Pr \left(\sup_{1 \leq j \leq k} \max_{s_j \in \mathbb{R}^{d+1}} \sum_{i=1}^n |P_{j^0}(\cdot) - s_j(\cdot)|^2 \leq 2^2 k(d+1) \log(kd) \log(n=kd) \frac{n}{kd} \right) \quad (6.29a.I)$$

$$\Pr \left(\sup_{1 \leq j \leq k} \max_{s_j \in \mathbb{R}^{d+1}} \sum_{i=1}^n |P_{j^0}(\cdot) - s_j(\cdot)|^2 \leq 2^2 k(d+1) \log(n=d) \frac{n}{d} \text{ for each } j \geq [k] \right) \quad (6.29a.II)$$

Second, we show that the indicator quantities are simultaneously bounded for all $j; j^0$ pairs. In particular, we claim that there exists a tuple of universal constants $(C; c_1; c_2; c^0)$ such that for each positive scalar $r = 1/24$, we have

$$\Pr \left(\prod_{j \notin j^0} \mathbf{1}_{\{v_{j;j^0} \geq B_{v_{j;j^0}}(r)\}} \geq \frac{1}{2} \prod_{j^0: j^0 \notin j} \mathbf{1}_{\{h_{i;j^0} \geq h_{i;j^0}^0\}} \geq \frac{1}{2} \prod_{j^0: j^0 \notin j} \mathbf{1}_{\{h_{i;j^0} \geq h_{i;j^0}^0\}} \right) \geq C \max_{j^0: j^0 \notin j} \left(\frac{dk}{3} \log n \frac{k^2}{6} \right)^{-1} \exp \left(-c_1 \frac{k}{2} n e^{-c_2 n} + e^{-c^0 \max_{j^0: j^0 \notin j} dk} \log n \right) \quad (6.29b)$$

Finally, we show a bound on the LHS of the bound (6.22) by handling the singular values of (random) sub-matrices of \mathbf{A} with a uniform bound. In particular, we claim that there are universal constants $(C; c; c^0)$ such that if $n \geq C \max_{\min} \frac{dk}{3}; \log n \geq \frac{k^2}{6}$, then for each $j \geq [k]$, we have

$$\Pr \left(\inf_{1 \leq j \leq k} \min_{s_j \in \mathbb{R}^{d+1}} \sum_{i=1}^n |P_{j^0}(\cdot) - s_j(\cdot)|^2 \leq C \frac{3}{\min} n \exp \left(-cn \frac{6}{k^2} \right) + c^0 n^{-10} \right) \quad (6.29c)$$

Notice that claim (6.29a.I) implicitly defines a high probability event $E^{(a:I)}$, claim (6.29a.II) defines high probability events $E_j^{(a:II)}$, claim (6.29b) defines a high probability event $E^{(b)}(r)$, and claim (6.29c) defines high probability events $E_j^{(c)}$. Define the intersection of these events as

$$E(r) := \bigcap_{j \geq [k]} \left(E^{(a:I)} \cap E_j^{(a:II)} \cap E^{(b)}(r) \cap E_j^{(c)} \right)$$

and note that the claims in conjunction with the union bound guarantee that if the condition on the sample size $n \geq c_1 \max_{\min} \frac{dk}{3}; \log n \geq \frac{k^2}{6}$ holds, then for all $r \geq r_b$, we have

$$\Pr \left(\mathbb{1}_{E(r)} \geq 1 - c_1 k \exp \left(-c_2 n \frac{6}{k^2} \right) + \frac{k^2}{n^7} \right)$$

where we have adjusted constants appropriately in stating the bound. We are now ready to prove the two parts of the proposition.

Proof of part (a): Work on the event $E(r_a)$. Normalizing inequality (6.22) by n and using claims (6.29a.II), (6.29b), and (6.29c) with $r = r_a$ then yields, simultaneously for all $j \geq [k]$, the bound

$$\begin{aligned}
 k_j^+ - j k^2 &\leq C \max \left\{ \frac{d}{\min n}, \frac{r_a}{\min} \log^{3=2}(1=r_a) \right\} \times_{j^0: j^0 \notin j} k v_{j:j^0} - v_{j:j^0} k^2 + C^0 \frac{kd}{\min n} \log(n=d) \\
 &\stackrel{(i)}{\leq} \max \left\{ \frac{Cd}{\min n}, \frac{1}{4k} \right\} \times_{j^0: j^0 \notin j} k v_{j:j^0} - v_{j:j^0} k^2 + C^0 \frac{kd}{\min n} \log(n=d);
 \end{aligned}$$

where in step (i), we have used the definition of the quantity r_a . Using this bound for the indices $j; \cdot$ in conjunction with the definition of the quantity \cdot proves inequality (6.27b). \square

Proof of part (b): We now work on the event $E(r_b)$ and proceed again (see equation (6.22)) from the bound

$$\begin{aligned}
 k_j^+ - j k^2 &\leq C \max \left\{ \frac{d}{\min n}, \frac{r_b}{\min} \log^{3=2}(1=r_b) \right\} \times_{j^0: j^0 \notin j} k v_{j:j^0} - v_{j:j^0} k^2 \\
 &\quad + \frac{C}{\min n} k P_{j(1, \dots, k)} s_j k^2;
 \end{aligned}$$

Summing over $j \geq [k]$ and using the fact that $ka + bk^2 \leq 2kak^2 + 2k bk^2$, we obtain

$$\begin{aligned}
 \sum_{j=1}^k k_j^+ - j k^2 &\leq C \max \left\{ \frac{kd}{\min n}, \frac{kr_b}{\min} \log^{3=2}(1=r_b) \right\} \sum_{j=1}^k k_j - j k^2 \\
 &\quad + \frac{C}{\min n} \sum_{j \geq [k]} k P_{j(1, \dots, k)} s_j k^2 \\
 &\stackrel{(ii)}{\leq} \frac{3}{4} \sum_{j=1}^k k_j - j k^2 + C^0 \frac{kd}{\min n} \log(k) \log(n=kd);
 \end{aligned}$$

where in step (ii), we have used the definition of the quantity r_b , the bound $n \geq Ckd = \frac{3}{\min}$, and claim (6.29a.I). This completes the proof. \square

We now prove each of the claims in turn. This constitutes the technical meat of our proof, and involves multiple technical lemmas whose proofs are postponed to the end of the section.

Proof of claims (6.29a.I) and (6.29a.II): We begin by stating a general lemma about concentration properties of the noise.

Lemma 29. Consider a random variable $z \in \mathbb{R}^n$ with i.i.d. σ -sub-Gaussian entries, and a fixed matrix $A \in \mathbb{R}^{n \times (d+1)}$. Then, we have

$$\sup_{1 \leq j \leq k} \sum_{i=1}^n |P_{j(1:k)} z_i|^2 \leq 2k(d+1) \log(kd) \log(n=kd) \quad (6.30a)$$

with probability greater than $1 - \frac{1}{kd}$ and

$$\sup_{1 \leq j \leq k} \max_{S \subseteq [k]} |P_{j(S)} z_S|^2 \leq 2k(d+1) \log(n=d) \quad (6.30b)$$

with probability greater than $1 - \frac{1}{d}$.

The proof of the claims follows directly from Lemma 29, since the noise vector z is independent of the matrix A , and $A \in \mathbb{R}^{n \times (d+1)}$. \square

Proof of claim (6.29b): We now state a lemma that directly handles indicator functions as they appear in the claim.

Lemma 30. Let $u \in \mathbb{R}^d$ and $w \in \mathbb{R}$, and consider a fixed parameter $v = (u; w) \in \mathbb{R}^{d+1}$. Then there are universal constants $(c_1; c_2; c_3; c_4)$ such that for all positive scalars $r \geq 1$, we have

$$\sup_{v \in B_v(r)} \frac{1}{n} \sum_{i=1}^n \mathbf{1}_{\{f(h_i; v) - h_i v \geq 0\}} \leq c_1 \max\left\{\frac{d}{n}; r \log^3\left(\frac{1}{r}\right)\right\}$$

with probability exceeding $1 - c_1 e^{-c_2 \max\{fd; 10 \log ng\}} - c_3 n e^{-c_4 n}$. Here, we adopt the convention that $0/0 = 0$.

Applying Lemma 30 with $v = v_{j,j^0}$ and $v = v_{j,j^0}$ for all pairs $(j; j^0)$ and using a union bound directly yields the claim. \square

Proof of claim (6.29c): For this claim, we state three technical lemmas pertaining to the singular values of random matrices whose rows are formed by truncated Gaussian random vectors. We let $\text{vol}(K)$ denote the volume of a set $K \subseteq \mathbb{R}^d$ with respect to d -dimensional standard Gaussian measure, i.e., with $\text{vol}(K) = \Pr\{Z \in K\}$ for $Z \sim N(0; I_d)$.

Lemma 31. Suppose n vectors $x_i, i=1, \dots, n$ are drawn i.i.d. from $N(0; I_d)$, and $K \subseteq \mathbb{R}^d$ is a fixed convex set. Then there exists a tuple of universal constants $(c_1; c_2)$ such that if $\text{vol}^3(K) n \geq c_1 d \log^2(1/\text{vol}(K))$, then

$$\min_{i: x_i \in K} \sum_{j=1}^n |x_i \cdot x_j| \geq c_2 \text{vol}^3(K) n$$

with probability greater than $1 - c_1 \exp\left(-c_2 n \frac{\text{vol}^4(K)}{\log^2(1/\text{vol}(K))}\right) - c_1 \exp(-c_2 n \text{vol}(K))$.

For a pair of scalars $(w; w^\flat)$ and d -dimensional vectors $(u; u^\flat)$, define the *wedge* formed by the $d + 1$ -dimensional vectors $v = (u; w)$ and $v^\flat = (u^\flat; w^\flat)$ as the region

$$W(v; v^\flat) = \{x \in \mathbb{R}^d : (hx; ui + w) \geq (hx; u^\flat i + w^\flat) \geq 0\};$$

and let $\mathcal{W} = \{W = W(v; v^\flat) : \text{vol}(W) \leq g\}$ denote the set of all wedges with Gaussian volume less than g . The next lemma bounds the maximum singular value of a sub-matrix formed by any such wedge.

Lemma 32. *There is a tuple of universal constants $(c_1; c_2)$ such that if $n \geq c_1 \max\{d; \frac{\log n}{2}\}$, then*

$$\sup_{W \in \mathcal{W}} \max_{i: x_i \in W} \sum_{i: x_i \in W} \lambda_i \leq c_1 n^{\rho_-}$$

with probability greater than $1 - \exp(-c_2 n^2) - n^{-10}$.

We are now ready to proceed to a proof of claim (6.29c). For convenience, introduce the shorthand notation

$$S_j := S_j(x_1, \dots, x_k)$$

to denote the set of indices corresponding to observations generated by the true parameter θ_j . Letting $A \Delta B := (A \setminus B) \cup (B \setminus A)$ denote the symmetric difference between two sets A and B , we have

$$\min_{j \in [k]} |S_j \Delta \hat{S}_j| \leq \min_{j \in [k]} |S_j \Delta \tilde{S}_j| + \max_{j \in [k]} |\tilde{S}_j \Delta S_j| + |S_j \Delta \hat{S}_j|.$$

Recall that by definition, we have

$$\begin{aligned} S_j \Delta \hat{S}_j &= \{i: h_i; \theta_j i \geq g\} \Delta \{i: h_i; \hat{\theta}_j i \geq g\} = \max\{ \{i: h_i; \theta_j i \geq g\} \setminus \{i: h_i; \hat{\theta}_j i \geq g\} \} \cup \{i: h_i; \hat{\theta}_j i \geq g\} \setminus \{i: h_i; \theta_j i \geq g\} \} \\ &= \{i: h_i; v_{j,j^0} i \geq h_i; v_{j,j^0} i < 0\} \cup \{i: h_i; v_{j,j^0} i < 0\} \\ &= \{i: x_i \in W_{j,j^0} \setminus \tilde{W}_{j,j^0}\} \cup \{i: x_i \in \tilde{W}_{j,j^0} \setminus W_{j,j^0}\} \end{aligned} \tag{6.31}$$

Putting together the pieces, we have

$$\min_{j \in [k]} |S_j \Delta \hat{S}_j| \leq \min_{j \in [k]} |S_j \Delta \tilde{S}_j| + \max_{j \in [k]} \sum_{i: x_i \in W_{j,j^0} \setminus \tilde{W}_{j,j^0}} \lambda_i + \max_{j \in [k]} \sum_{i: x_i \in \tilde{W}_{j,j^0} \setminus W_{j,j^0}} \lambda_i \tag{6.32}$$

Now by Lemma 36, the definition of the set I_b , and the definition of r_b , we have

$$\text{vol}(W_{j,j^0} \setminus \tilde{W}_{j,j^0}) \leq n r_b \log^{1+2}(1/r_b) \leq C \frac{6}{k^2} \min_{j \in [k]} \lambda_j$$

Owing to the sample size assumption $n \geq C \max\{d, k^2 \frac{\log n}{\min}\}$, the conditions of Lemma 32 are satisfied, and applying it yields

$$\sup_{\substack{V_{j,j^0} \in \mathcal{B}_{V_{j,j^0}}(r_a) \\ \forall j \in [k]}} \max_{\substack{X \\ i \in [d]}} \sum_{j \in [k]} \sum_{j^0 \in [k]} |V_{j,j^0}| \leq n C \frac{\min}{k}$$

with probability exceeding $1 - n^{-10} \exp(-cn \frac{\min}{k^2})$. Moreover, Lemma 31 guarantees the bound $\min_j \hat{s}_j \geq s_j - c_2 n^{-3} \min$, so that putting together the pieces, we have

$$\inf_{1 \leq j \leq k} \min_{j \in [k]} \hat{s}_j \geq s_j - c_2 n^{-3} \min - C n k \frac{\min}{k} \geq C \min^3 n; \tag{6.33}$$

with probability greater than $1 - c \exp(-cn \frac{\min}{k^2}) - n^{-10}$. These assertions hold provided

$$n \geq C \max\left\{d, \frac{k}{\min}, \frac{k^2 \log n}{\min}\right\};$$

and this completes the proof. □
 Having proved the claims, we turn to proofs of our technical lemmas.

Proof of Lemma 29

In this proof, we assume that $\beta = 1$; our bounds can finally be scaled by β^2 .

It is natural to prove the bound (6.30b) first followed by bound (6.30a). First, consider a fixed set of parameters $f_{1, \dots, k} \in \mathcal{G}$. Then, we have

$$P_{J(1, \dots, k)} \sum_{j \in [k]} Z_{S_j}^2 = \sum_{j \in [k]} U U^T Z_{S_j}^2;$$

where $U \in \mathbb{R}^{J \times (d+1)}$ denotes a matrix with orthonormal columns that span the range of $J(1, \dots, k)$.

Applying the Hanson-Wright inequality for independent sub-Gaussians (see [112, Theorem 2.1]) and noting that $\sum_{j \in [k]} \sum_{i \in S_j} U_{ij}^2 \leq \frac{\rho}{d+1}$ we obtain

$$\Pr \left(\sum_{j \in [k]} U U^T Z_{S_j}^2 \geq (d+1) + t \right) \leq e^{-ct};$$

for each $t \geq 0$. In particular, this implies that the random variable $\sum_{j \in [k]} U U^T Z_{S_j}^2$ is sub-exponential.

This tail bound holds for a fixed partition of the rows of J ; we now take a union bound over all possible partitions. Toward that end, define the sets

$$S_j = \{S_j(1, \dots, k) : 1, \dots, k \in \mathbb{R}^{d+1}\}; \text{ for each } j \in [k];$$

From Lemma 38, we have the bound $\sum_{j=1}^k |S_j| \leq 2^{ckd \log(en=d)}$. Thus, applying the union bound, we obtain

$$\Pr \left(\sup_{\substack{1, \dots, k \in \mathbb{R}^{d+1}}} \sum_{j=1}^k P_{J(1, \dots, k)} Z_{S_j}^2 \geq (d+1) + t \sum_{j=1}^k |S_j| e^{-ct} \right)$$

and substituting $t = ck(d+1) \log(n=d)$ and performing some algebra establishes bound (6.30b).

In order to establish bound (6.30a), we once again consider the random variable $\sum_{j=1}^k P_{J(1, \dots, k)} Z_{S_j}^2$ for a fixed set of parameters f_1, \dots, f_k, g . Note that this is the sum of k independent sub-exponential random variables and can be thought of as a quadratic form of the entire vector Z . So once again from the Hanson-Wright inequality, we have

$$\Pr \left(\sup_{\substack{1, \dots, k \in \mathbb{R}^{d+1}}} \sum_{j=1}^k P_{J(1, \dots, k)} Z_{S_j}^2 \geq k(d+1) + t \sum_{j=1}^k |S_j| e^{-ct} \right)$$

for all $t \geq 0$.

Also define the set of all possible partitions of the n points via the max-affine function; we have the set

$$S = \{S_1(1, \dots, k), \dots, S_k(1, \dots, k) : 1, \dots, k \in \mathbb{R}^{d+1}\}$$

Lemma 39 yields the bound $\sum_{j=1}^k |S_j| \leq 2^{ckd \log(kd) \log(n=kd)}$, and combining a union bound with the high probability bound above establishes bound (6.30a) after some algebraic manipulation. \square

Proof of Lemma 30

Let $v = \sum_{i=1}^n v_i$; we have

$$\begin{aligned} \mathbf{1} \{f h_i; v_i - h_i; v_i \leq 0\} & \leq \mathbf{1} \{f h_i; v_i - h_i; v_i \leq 0\} \\ \mathbf{1} \{h_i; v_i^2 - h_i; v_i^2 - h_i; v_i^2\} & \leq \mathbf{1} \{h_i; v_i^2 - h_i; v_i^2 - h_i; v_i^2\} \end{aligned}$$

Define the (random) set $K_v = \{i : h_i; v_i^2 > h_i; v_i^2\}$; we have the bound

$$\frac{1}{n} \sum_{i=1}^n \mathbf{1} \{f h_i; v_i - h_i; v_i \leq 0\} \leq \frac{1}{n} |K_v| v^2$$

We now show that the quantity $|K_v| v^2$ is bounded uniformly for all $v \in B_v(r)$ for small enough r . Recall that u is the “linear” portion of v , and let $m = \max\{d, 10 \log n, n^{16r} \log(1+r)\}$

(note that m depends implicitly on r). We claim that for all $r \in (0, 1/24]$, we have

$$\Pr \sup_{v \in 2B_v(r)} \| \sum_{j \in T} K_{vj} \| > m \leq 4e^{-c \max\{fd, 10 \log ng\}} + cne^{-c'n}; \text{ and} \tag{6.34a}$$

$$\Pr \sup_{\substack{T \subseteq [n]: \\ |T| \geq m}} \sup_{\substack{! \in \mathbb{R}^{d+1} \\ ! \neq 0}} \frac{\| \sum_{j \in T} ! k_j^2 \|}{\| ! \|^2} \leq (2d + 20m \log(n/m)) e^{-c \max\{fd, 10 \log ng\}}; \tag{6.34b}$$

Taking these claims as given, the proof of the lemma is immediate, since $\frac{n}{m} \leq \frac{1}{16r \log(1/r)}$, so that $\log(n/m) \leq C \log(1/r)$.

Proof of claim (6.34a): By definition of the set K_v , we have

$$\begin{aligned} \Pr \sup_{v \in 2B_v(r)} \| \sum_{j \in T} K_{vj} \| > mg &= \Pr \exists v \in 2B_v(r) : \left\| \sum_{j \in T} v k_j^2 \right\| > k_{TV} k^2 \\ &= \Pr \exists v \in 2B_v(r) : \frac{\| \sum_{j \in T} v k_j^2 \|}{\| v \|^2} > \frac{k_{TV} k^2}{\| v \|^2} \\ &= \Pr \exists v \in 2B_v(r) : r^2 \frac{\| \sum_{j \in T} v k_j^2 \|}{\| v \|^2} > \frac{k_{TV} k^2}{\| v \|^2} \\ &= \Pr \exists v \in 2B_v(r) : \frac{\| \sum_{j \in T} v k_j^2 \|}{\| v \|^2} > \left(\frac{r}{\| v \|^2} \right)^2 \left(\frac{k_{TV} k^2}{\| v \|^2} \right) \\ &\quad + \Pr \frac{\| \sum_{j \in T} v k_j^2 \|}{\| v \|^2} > r^2 \left(\frac{r}{\| v \|^2} \right)^2 \left(\frac{k_{TV} k^2}{\| v \|^2} \right); \end{aligned}$$

where the final step follows by the union bound and holds for all positive scalars $t, t_T, g_T \in [n]$. For some fixed subset T of size n , we have the tail bounds

$$\Pr \sup_{\substack{T \subseteq [n]: \\ |T| \geq m}} \sup_{\substack{! \in \mathbb{R}^{d+1} \\ ! \neq 0}} \frac{\| \sum_{j \in T} ! k_j^2 \|}{\| ! \|^2} > \left(\frac{r}{\| v \|^2} \right)^2 \left(\frac{k_{TV} k^2}{\| v \|^2} \right) \geq e^{-t^2/2}; \text{ for all } t \geq 0; \text{ and} \tag{6.35a}$$

$$\Pr \frac{\| \sum_{j \in T} v k_j^2 \|}{\| v \|^2} > r^2 \left(\frac{r}{\| v \|^2} \right)^2 \left(\frac{k_{TV} k^2}{\| v \|^2} \right) \geq e^{-t^2/2} \text{ for all } t \geq 0; \tag{6.35b}$$

where step (i) follows from the sub-Gaussianity of the covariate matrix (see Lemma 40), and step (ii) from a tail bound for the non-central χ^2 distribution (see Lemma 41).

Substituting these bounds yields

$$\Pr f \sup_{v \in B_V(r)} |K_{v,j}| > mg \leq \sum_{j=m+1}^n \left(42e^{-t^2/2} + e^{-t^2} \frac{(\rho_{-d} + \rho_{-} + t)^2}{\rho_{-}} \right)^3$$

$$\leq \sum_{j=m+1}^n \left(42e^{-t^2/2} + 2r \frac{(\rho_{-d} + \rho_{-} + t)^2}{\rho_{-}} \right)^3$$

Recall that t was a free (non-negative) variable to be chosen. We now split the proof into two cases and choose this parameter differently for the two cases.

Case 1, $m \leq n/2$: Substituting the choice $t = 4\sqrt{\rho_{-} \log(n/m)}$, we obtain

$$\sum_{j=m+1}^n \left(42e^{-t^2/2} + 2r \frac{(\rho_{-d} + \rho_{-} + t)^2}{\rho_{-}} \right)^3$$

$$\leq \sum_{j=m+1}^n e^{-c} + \sum_{j=m+1}^n 2r \frac{(\rho_{-d} + 5\sqrt{\rho_{-} \log(n/m)})^2}{\rho_{-}}$$

$$(i) \quad \sum_{j=m+1}^n e^{-c} + \sum_{j=m+1}^n 2r (1 + 5\sqrt{\rho_{-} \log(n/m)})^2$$

$$(ii) \quad \sum_{j=m+1}^n e^{-c} + \sum_{j=m+1}^n 12r \sqrt{\rho_{-} \log(n/m)}$$

$$\leq \sum_{j=m+1}^n e^{-c} + 12 \frac{en}{r} \sqrt{\rho_{-} \log(n/m)}$$

where step (i) follows from the bound $m \leq n/2$, and step (ii) from the bound $m \leq n/2$.

Now note that the second term is only problematic for small n . For all $n \geq m = n/2$ (16r $\sqrt{\rho_{-} \log(1+r)}$), we have

$$12 \frac{en}{r} \sqrt{\rho_{-} \log(n/m)} \leq (3/4) \sum_{j=m+1}^n e^{-c} \tag{3-4}$$

The first term, on the other hand, satisfies the bound $\sum_{j=m+1}^n e^{-c} \leq (3/4) \sum_{j=m+1}^n e^{-c}$ for sufficiently large n .

Case 2, $m > n/2$: In this case, setting $t = 2\sqrt{\rho_{-} n}$ for each j yields the bound

$$\sum_{j=m+1}^n \left(42e^{-t^2/2} + 2r \frac{(\rho_{-d} + \rho_{-} + t)^2}{\rho_{-}} \right)^3 \leq 2 \sum_{n=2}^n e^{-2n} + (12r)^3$$

$$\leq ce^{-cn};$$

where we have used the fact that $d \leq n-2$ and $r \leq 1-24$.

Putting together the pieces from both cases, we have shown that for all $r \geq (0; 1-24]$, we have

$$\Pr \sup_{v \in B_{\infty}^d(r)} |K_{v,j}| > mg \leq cne^{d/n} + \frac{e^{-e}}{r^{m+1}} \quad (3.4)$$

$$cne^{d/n} + 4(3/4)^{\max\{d, 10 \log ng\}},$$

thus completing the proof of the claim.

Proof of claim (6.34b): The proof of this claim follows immediately from the steps used to establish the previous claim. In particular, writing

$$\Pr \left[\sum_{j=1}^m \sum_{T \subseteq [n]: |T|=k} \sum_{k=1}^m k \tau! k^2 \leq 2d + 20m \log(n=m) \right]$$

$$\Pr \left[\sum_{j=1}^m \sum_{T \subseteq [n]: |T|=k} \sum_{k=1}^m k \tau! k^2 \leq \rho_{d+} + \rho_{m+} + \rho_{\frac{4m \log(n=m)}{2}} \right]$$

$$\Pr \left[\sum_{j=1}^m \sum_{T \subseteq [n]: |T|=k} \sum_{k=1}^m k \tau! k^2 \leq \rho_{d+} + \rho_{m+} + \rho_{\frac{4m \log(n=m)}{2}} \right]$$

$$(iv) \frac{e^n}{2} \frac{n}{m} \exp\{-2m \log(n=m)g\}$$

$$\frac{e^n}{2} \frac{n}{m} \leq 2e^{c \max\{d, 10 \log ng\}},$$

where step (iv) follows from the tail bound (6.35a). □

Proof of Lemma 31

The lemma follows from some structural results on the truncated Gaussian distribution. Using the shorthand $\text{vol} := \text{vol}(K)$ and letting ϕ denote the d -dimensional Gaussian density, consider a random vector y drawn from the distribution having density $h(y) = \frac{1}{\text{vol}} \mathbf{1}_{\{y \in K\}}$, and denote its mean and second moment matrix by μ and Σ , respectively. Also denote the recentered random variable by $e = y - \mu$. We claim that

$$\|K\|_2 \leq C \log(1/\text{vol}); \tag{6.36a}$$

$$\Sigma \preceq (1 + C \log(1/\text{vol})) I; \tag{6.36b}$$

$$e \text{ is } c\text{-sub-Gaussian for a universal constant } c; \tag{6.36c}$$

Taking these claims as given for the moment, let us prove the lemma.

The claims (6.36a) and (6.36c) taken together imply that the random variable $\mathbb{E} \sum_{i=1}^m \frac{1}{m} X_i^n$ is sub-Gaussian with parameter $\sqrt{2c^2 + 2C \log(1/\text{vol})}$. Now consider m i.i.d. draws of $\mathbb{E} \sum_{i=1}^m \frac{1}{m} X_i^n$ given by $f_i g_{i=1}^m$; standard results (see, e.g., Vershynin [111, Remark 5.40], or Wainwright [13, Theorem 6.2]) yield the bound

$$\Pr \left(\sum_{i=1}^m \frac{1}{m} X_i^n > \mathbb{E} \sum_{i=1}^m \frac{1}{m} X_i^n + \sqrt{2 \frac{d}{m} + \frac{r}{m}} \right) \leq 2 \exp(-cn \min f; 2g)$$

Using this bound along with claim (6.36b) and Weyl's inequality yields

$$\min \sum_{i=1}^m \frac{1}{m} X_i^n \geq C \text{vol}^2 - \sqrt{2 \frac{d}{m} + \frac{r}{m}} \tag{6.37}$$

with probability greater than $1 - 2 \exp(-cn \min f; 2g)$.

Furthermore, when n samples are drawn from a standard Gaussian distribution, the number m of them that fall in the set K satisfies $m \geq \frac{1}{2} n \text{vol}$ with high probability. In particular, this follows from a straightforward binomial tail bound, which yields

$$\Pr \left(m < \frac{n \text{vol}}{2} \right) \leq \exp(-cn \text{vol}) \tag{6.38}$$

Recall our choice $n = Cd \frac{\log^2(1/\text{vol})}{\text{vol}^3}$, which in conjunction with the bound (6.38) ensures that $C \text{vol}^2 - \sqrt{\frac{1}{8} \frac{d}{m}}$ with high probability. Setting $\epsilon = C \text{vol}^2 - \sqrt{\frac{1}{8} \frac{d}{m}}$ in inequality (6.37), we have

$$\min \sum_{i=1}^m \frac{1}{m} X_i^n \geq \frac{C}{2} \text{vol}^2$$

with probability greater than $1 - 2 \exp(-cn \text{vol}^4) = 4^{-n}$. Putting together the pieces thus proves the lemma. It remains to show the various claims. □

Proof of claim (6.36a) Let X_A denote a random variable formed as a result of truncating the Gaussian distribution to a (general) set A with volume vol . Letting μ_A denote its mean, the dual norm definition of the ℓ_2 norm yields

$$\|k_A\| = \sup_{v \in S^{d-1}} \langle v, k_A \rangle = \sup_{v \in S^{d-1}} \mathbb{E} \langle v, X_A \rangle$$

Let us now evaluate an upper bound on the quantity $\mathbb{E} \langle v, X_A \rangle$. In the calculation, for any d -dimensional vector y , we use the shorthand $y_v := v^T y$ and $y_{nv} := U_{nv}^T y$ for a matrix $U_{nv} \in \mathbb{R}^{d \times d}$

having orthonormal columns that span the subspace orthogonal to v . Letting $A_v \in \mathbb{R}^{d \times (d-1)}$ denote the projection of A onto the direction v , define the set $A_{nv}(w) \subset \mathbb{R}^{d-1}$ via

$$A_{nv}(w) = \{y_{nv} \in \mathbb{R}^{d-1} : y \in A \text{ and } y_v = wg\}$$

Letting ϕ_d denote the d -dimensional standard Gaussian pdf, we have

$$\begin{aligned} E_j h v; A_{ij} &= \frac{1}{\text{vol}} \int_{y \in A} |jy^> v| \phi_d(y) dy \\ &= \frac{1}{\text{vol}} \int_{y \in A} |jy_v| \phi_{d-1}(y_{nv}) dy \\ &= \frac{1}{\text{vol}} \int_{y_v \in A_v} |jy_v| \phi_{d-1}(y_v) \underbrace{\int_{y_{nv} \in A_{nv}(y_v)} \phi_{d-1}(y_{nv}) dy_{nv}}_{\int_{y_v \in A_v} \phi_{d-1}(y_v) dy_v} dy_v \\ &\stackrel{(i)}{=} \frac{1}{\text{vol}} \int_{y_v \in A_v} |jy_v| \phi_{d-1}(y_v) dy_v; \end{aligned} \tag{6.39}$$

where step (i) follows since $\phi_{d-1}(y_v) = 1$ point-wise. On the other hand, we have

$$\text{vol} = \int_{y_v \in A_v} \phi_{d-1}(y_v) \int_{y_{nv} \in A_{nv}(y_v)} \phi_{d-1}(y_{nv}) dy_{nv} dy_v = \int_{y_v \in A_v} \phi_{d-1}(y_v) dy_v; \tag{6.40}$$

Combining inequalities (6.39) and (6.40) and letting $w = y_v$, an upper bound on k can be obtained by solving the one-dimensional problem given by

$$\begin{aligned} k &\leq \sup_{S \subset \mathbb{R}} \frac{1}{\text{vol}} \int_{w \in S} |jw| \phi_{d-1}(w) dw \\ \text{s.t. } &\int_{w \in S} \phi_{d-1}(w) dw = \text{vol}; \end{aligned}$$

It can be verified that the optimal solution to the problem above is given by choosing the truncation set $S = (-\tau; \tau) \cap [-1; 1]$ for some threshold $\tau > 0$. With this choice, the constraint can be written as

$$\text{vol} = \int_{|w| \leq \tau} \phi_{d-1}(w) dw = \frac{1}{\sqrt{2\pi}} e^{-\frac{\tau^2}{2}};$$

where we have used a standard Gaussian tail bound. Simplifying yields the bound

$$k \leq \frac{1}{\sqrt{2\pi}} \sqrt{\log(C/\text{vol})};$$

Furthermore, we have

$$\frac{1}{\text{vol}} \int_{\mathcal{K}} \langle w, z \rangle^2 dw = \frac{C}{\text{vol}} e^{-\frac{1}{2}\|z\|^2} \int_{\mathcal{K}} e^{\frac{1}{2}\|z\|^2} \langle w, z \rangle^2 dw$$

$$\stackrel{(ii)}{\leq} \frac{C}{\text{vol}} \int_{\mathcal{K}} e^{\frac{1}{2}\|z\|^2} \frac{1}{\|z\|^2} dw$$

where step (ii) follows from the bound $\Pr\{Z \leq z\} \leq \frac{1}{\|z\|^2}$ valid for a standard Gaussian variate Z . Putting together the pieces, we have

$$k \leq C \log(1/\text{vol});$$

□

Proof of claim (6.36b) Let us first show the upper bound. Writing $\text{cov}(\cdot)$ for the covariance matrix, we have

$$\text{tr}(\text{cov}(\cdot)) \leq \text{tr}(\text{cov}(Z)) + k \|z\|^2$$

$$\stackrel{(iii)}{\leq} \text{tr}(\text{cov}(Z)) + C \log(1/\text{vol});$$

where step (iii) follows from the fact that $\text{cov}(\cdot) \preceq \text{cov}(Z)$, since truncating a Gaussian to a convex set reduces its variance along all directions [204, 205].

We now proceed to the lower bound. Let P_K denote the Gaussian distribution truncated to the set K . Recall that we denoted the probability that a Gaussian random variable falls in the set K by $\text{vol}(K)$; use the shorthand $\text{vol} = \text{vol}(K)$. Define the polynomial

$$p_u(x) = \langle x, u \rangle^2 - \mathbb{E}_{X \sim P_K}[\langle X, u \rangle^2];$$

note that we are interested in a lower bound on $\inf_{\|u\|=1} \mathbb{E}_{X \sim P_K}[p_u(X)]$.

For $\epsilon > 0$, define the set

$$S := \{x \in \mathbb{R}^d : p_u(x) \geq \epsilon\};$$

Letting Z denote a d -dimensional standard Gaussian random vector and using the shorthand $\mu := \mathbb{E}_{X \sim P_K}[X]$, we have

$$\Pr\{Z \in S\} \geq \Pr\{\langle Z, u \rangle^2 \geq \epsilon\} \tag{6.41}$$

$$= \Pr\{\langle Z, u \rangle \geq \sqrt{\epsilon}\} + \Pr\{\langle Z, u \rangle \leq -\sqrt{\epsilon}\} \tag{6.42}$$

$$= \int_{\sqrt{\epsilon}}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}} dt + \int_{-\infty}^{-\sqrt{\epsilon}} \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}} dt \tag{6.43}$$

where in the final step, we have used the fact that $p_u(x) \leq \frac{1}{2}$ for all $x \in \mathbb{R}^d$.

Consequently, we have

$$\begin{aligned} \mathbb{E}_{X \sim P_K} [p_u(X)] &= \frac{1}{\text{vol}(K)} \mathbb{E}_Z [p_u(Z) \mathbf{1}_{\{Z \in K\}}] \\ &= \frac{1}{\text{vol}(K)} \mathbb{E}_Z [p_u(Z) \mathbf{1}_{\{Z \in K \setminus S^c\}}] \\ &\stackrel{(iv)}{=} \frac{1}{\text{vol}(K)} \mathbb{E}_Z [\mathbf{1}_{\{Z \in K \setminus S^c\}}] \\ &= \frac{\Pr\{Z \in K \setminus S^c\}}{\text{vol}(K)} \\ &\stackrel{(v)}{=} \frac{\Pr\{Z \in K\} - \Pr\{Z \in S\}}{\text{vol}(K)} \end{aligned}$$

Here, step (iv) follows from the definition of the set S , which ensures that $p_u(x) \leq \frac{1}{2}$ for all $x \in S^c$. Step (v) follows as a consequence of equation (6.43), since

$$\Pr\{Z \in K \setminus S^c\} = \Pr\{Z \in K\} - \Pr\{Z \in S\} \leq \frac{c}{2} \text{vol}(K)$$

Finally, choosing $c = c \text{vol}(K)^2$ for a suitably small constant c , we have $\mathbb{E}_{X \sim P_K} [p_u(X)] \leq C \text{vol}(K)^2$ for a fixed $u \in S^{d-1}$. Since u was chosen arbitrarily, this proves the claim. \square

Proof of claim (6.36c) Since the random variable \mathcal{E} is obtained by truncating a Gaussian random variable to a convex set, it is 1-strongly log-concave. Thus, standard results [206, Theorem 2.15] show that the random variable \mathcal{E} is c -sub-Gaussian. \square

Proof of Lemma 32

For a pair of $d + 1$ -dimensional vectors $(v; v^\wedge)$, denote by

$$n_{W(v; v^\wedge)} = \#\{i : x_i \in W(v; v^\wedge)\} \tag{6.44}$$

the random variable that counts the number of points that fall within the wedge $W(v; v^\wedge)$; recall our notation \mathcal{W} for the set of all wedges with Gaussian volume less than $\frac{1}{2}$. Since each wedge is formed by the intersection of two hyperplanes, applying Lemmas 37 and 38 in conjunction yields that there are universal constants $(c; c^d; C)$ such that

$$\sup_{W \in \mathcal{W}} n_W \leq c n \tag{6.45}$$

with probability exceeding $1 - \exp(-c^d n^2)$, provided $n \geq \frac{C}{c}$. In words, the maximum number of points that fall in *any* wedge of volume $\frac{1}{2}$ is linear in n with high probability.

It thus suffices to bound, simultaneously, the maximum singular value of every sub-matrix of having (at most) $c n$ rows. Applying [181, Theorem 5.7] yields the bound¹⁰

$$\Pr \left(\max_{S:|S| \leq c n} \max_{i \in S} \sum_{j \in S} |X_{ij}| \leq c_1 n^{\rho_-} n^{-10}; \right)$$

where we have used the lower bound $n \geq c \max\{d; \log n\} = g$ on the sample size.

Putting together the pieces, we have that if $n \geq c \max\{d; \frac{\log n}{2}\}$, then

$$\sup_{W \subseteq W} \max_{i: x_i \in W} \sum_{j \in W} |X_{ij}| \leq c_1 n^{\rho_-}$$

with probability exceeding $1 - n^{-10} \exp(-c^d n^2)$. □

6.7 Proof of Theorem 13

We dedicate the first portion of the proof to a precise definition of the quantity ρ .

Let $X \in \mathbb{R}^{k \times d}$ denote a matrix with rows $(x_j)^T; j = 1; \dots; k$ and let $\mathbf{1} = (1; \dots; 1)^T \in \mathbb{R}^k$. We employ the decomposition $X = A(U)^T$, where $A \in \mathbb{R}^{k \times k}$ is the invertible matrix of coefficients and $U \in \mathbb{R}^{d \times k}$ is a matrix of orthonormal columns. Note that for $X \sim N(0; I_d)$, the vector in \mathbb{R}^k with j -th component $\langle x_j, \mathbf{1} \rangle + b_j$ is distributed as $Z + b$ where $Z \sim N(0; \mathbf{1})$ and the vector $b \in \mathbb{R}^k$ collects the scalars b_j in its entries. For $Z \sim N(0; \mathbf{1})$, let

$$\rho := \frac{\mathbb{E} \max(Z + b)^T \mathbf{1}}{\mathbb{E} [(\max(Z + b)^T \mathbf{1})^2]} \tag{6.46}$$

denote the correlation coefficient between the maximum and a particular linear combination of a multivariate Gaussian distribution. Variants of such quantities have been studied extensively in the statistical literature (see, e.g., James [207]). For our purposes, the fact that $\max(Z + b)^T \mathbf{1} \neq 0$ for any finite b , coupled with a full-rank A , ensure that $\rho \neq 0$ for any fixed k . Also define the positive scalar $\mu := \frac{\mathbb{E} [(\max(Z + b)^T \mathbf{1})^2]}{\mathbb{E} [(\max(Z + b)^T \mathbf{1})]}$ which tracks the average size of our observations. Also recall the quantity δ defined in the main section.

For each $j \in [k]$ consider the zero-mean Gaussian random vector with covariance $(\mathbf{1} - e_j e_j^T) A (A)^T (\mathbf{1} - e_j e_j^T)$. This is effectively a Gaussian that lives in $k - 1$ dimensions, with density that we denote by $\phi_j(x_1; x_2; \dots; x_{j-1}; 0; x_{j+1}; \dots; x_k)$ at point $(x_1; x_2; \dots; x_{j-1}; 0; x_{j+1}; \dots; x_k)$ (the

¹⁰Strictly speaking, [181, Theorem 5.7] applies to Gaussian random matrices, i.e., without the appended column of ones. By multiplying each row of X with an independent Rademacher RV (see the proof of Lemma 40) to obtain a sub-Gaussian random matrix with the same singular values, and noting also that the proof technique of [181, Theorem 5.7] relies on chaining and holds for a sub-Gaussian random matrix, one can show that the same result also holds for the matrix X .

density is not defined elsewhere). Truncate this random vector to the region $f_{X_i} = b_i - b_j : i \in [k]g$; this results in the *truncated* Gaussian density $f_j(x_1; x_2; \dots; x_{j-1}; 0; x_{j+1}; \dots; x_k)$ for each $j \in [k]$.

For any $x \in \mathbb{R}^k$ such that $x_j = 0$, define

$$F_i^j(x) = \int_{b_1}^{z_1} \dots \int_{b_{i-1}}^{z_{i-1}} \int_{b_{i+1}}^{z_{i+1}} \dots \int_{b_k}^{z_k} f_j(x_1; \dots; x_{i-1}; x; x_{i+1}; \dots; x_k) dx_k \dots dx_{i+1} dx_{i-1} \dots dx_1 \tag{6.47}$$

to be the i -th marginal density of this truncated Gaussian evaluated at the point x , with the convention that $F_i^j(\cdot) = 0$ everywhere. Also define the vector F^j by setting its i -th entry to $(F^j)_i = F_i^j(b_i - b_j)$.

Now let P denote the matrix with entries

$$P_{i,j} = \begin{cases} (F^j)_i = \sum_{k \notin j} (F^j)_k & \text{if } i \notin j \\ 0 & \text{otherwise.} \end{cases}$$

Note that the matrix P is the transition matrix of an irreducible, aperiodic Markov chain, with one eigenvalue equal to 1. Consequently, the matrix $I - P$ is rank $k - 1$. With this setup in place, let

$$\gamma := \min_{j \in [k]} \sum_{i \in [k]} (F^j)_i = \min_{j \in [k]} \sum_{k \notin j} (F^j)_k = \frac{1}{k-1} \sum_{k \in [k]} ((I - P)^{-1} (I - P)) \tag{6.48}$$

denote a positive scalar that will serve as a bound on our eigengap.

Let $M_1 = E[\max(X + b)X]$ and $M_2 = E[\max(X + b)(XX^T - I_d)]$ denote the expectations of the first and second moment estimators, respectively.

For a random variable $W \sim N(b; \cdot)$, we often use the shorthand

$$fW_j = \max g := fW_j \quad W_i \text{ for all } 1 \leq i \leq k g:$$

Finally, collect the probabilities $f_j g_{j=1}^k$ defined in equation (6.9) in a vector $\in \mathbb{R}^k$. We use $\mathbf{1}$ to denote the all-ones vector in k dimensions.

We are ready to state our two main lemmas.

Lemma 33. (a) *The first moment satisfies*

$$M_1 = \sum_{j \in [k]} (F^j)_i \quad \text{and} \quad hM_1; \sum_{i \in [k]} \mathbf{1}_i = \sum_{j \in [k]} \mathbf{1} :$$

(b) *The second moment satisfies*

$$M_2 \succeq 0; \quad M_2 \sum_{i \in [k]} \mathbf{1} = 0; \quad \text{rank}(M_2) = k - 1 \quad \text{and}$$

$$\lambda_{k-1}(M_2) = \min_{j \in [k]} \sum_{k \notin j} (F^j)_k = \frac{1}{k-1} \sum_{k \in [k]} ((I - P)^{-1} (I - P)) :$$

We combine this lemma with a result that shows that the empirical moments concentrate about their expectations.

Lemma 34. *For an absolute constant C , we have*

$$\Pr \left(\|\mathcal{M}_1 - M_1\|_F^2 \leq C \left(\frac{2 + \delta^2}{n} \frac{d \log^2(nk)}{n} \right) \leq 5dn^{-12}; \text{ and} \right) \quad (6.49a)$$

$$\Pr \left(\|\mathcal{M}_2 - M_2\|_{\text{op}}^2 \leq C \left(\frac{2 + \delta^2}{n} \frac{d \log^3(nk)}{n} \right) \leq 5dn^{-12}; \right) \quad (6.49b)$$

Lemma 33 is proved at the end of this section, and Lemma 34 is proved in Appendix 6.9.10. For now, we take both lemmas as given and proceed to a proof of Theorem 13.

Recall the matrix $\mathcal{M} = \mathcal{M}_1 + \mathcal{M}_2$ and let $M = M_1 + M_2$. By Lemma 33, the matrix M is positive semidefinite with k non-zero eigenvalues. In particular, using the shorthand $\mathbf{1} := (\mathbf{1})^T$, we have $\mathbf{1} \in \text{nullspace}(M_2)$, and so

$$\mathbf{1}^T M \mathbf{1} = \mathbf{1}^T M_1 \mathbf{1} = \frac{2}{n} k^2;$$

where the final inequality follows by part (a) of Lemma 33.

Thus, there is a k -dimensional subspace orthogonal to the nullspace of M (and so the range of M is k dimensional). For any unit vector v in this subspace, we have

$$v^T M v \geq \frac{2}{n} k^2; \quad \lambda_k(M) \geq \frac{2}{n} k^2.$$

Thus, the k th eigenvalue of M satisfies

$$\lambda_k(M) \geq \frac{2}{n} k^2; \quad \min_{j \geq k} \lambda_j(M) \geq \frac{2}{n} k^2 \prod_{k \leq j} \frac{\lambda_j(M)}{\lambda_k(M)} \geq \frac{2}{n} k^2 \frac{\lambda_k(M)}{\lambda_1(M)} = \frac{2}{n} k^2 \frac{\lambda_k(M)}{\lambda_1(M)};$$

where the equality follows by definition (6.48). By Lemma 34, we have

$$\begin{aligned} \|\mathcal{M} - M\|_{\text{op}}^2 &\leq 2\|\mathcal{M}_2 - M_2\|_{\text{op}}^2 + 2\|\mathcal{M}_1 - M_1\|_{\text{op}}^2 \\ &\leq 2C \left(\frac{2 + \delta^2}{n} \log^2(nk) \right) \frac{d \log(nk)}{n} + 16 \|\mathcal{M}_1 - M_1\|_{\text{op}}^2 \leq 4 \|\mathcal{M}_1 - M_1\|_{\text{op}}^2 \\ &\leq C \left(\frac{2 + \delta^2}{n} \log^2(nk) \right) \frac{d \log(nk)}{n}; \end{aligned}$$

where the last two inequalities each hold with probability greater than $1 - 2n^{-10}$.

We denote the estimated and true eigenspaces by \mathcal{U} and U , respectively. Applying [208, Theorem 2] yields the bound

$$\|\mathcal{U} - U\|_F \leq C \frac{2 + \delta^2}{2} \frac{k d \log^3(nk)}{n};$$

thereby proving the required result. □

We now proceed to a proof of Lemma 33.

6.7.1 Proof of Lemma 33

Recall our decomposition $X = A(U)^>$, where $U \in \mathbb{R}^{d \times k}$ is a matrix of orthonormal columns, and $A \in \mathbb{R}^{k \times k}$ is an invertible matrix of coefficients. Since we are always concerned with random variables of the form X with X Gaussian, we may assume without loss of generality by the rotation invariance of the Gaussian distribution that $U = [e_1^d \ e_2^d \ \dots \ e_k^d]$, where e_i^d denotes the i th standard basis vector in \mathbb{R}^d .

We let $X_i^j = (X_i; X_{i+1}; \dots; X_j)$ denote a sub-vector of the random vector X , so that by the above argument, we have $X \stackrel{d}{=} A X_1^k$.

Calculating M_1 : Using the shorthand $Z = A X_1^k$, we have

$$\begin{aligned} M_1 &= E[\max(X + b)X] \\ &= U E[\max(A X_1^k + b)X] \\ &= U (A)^{-1} E[\max(Z + b)Z]. \end{aligned}$$

Now using Stein's lemma¹¹, by a calculation similar to the one performed also in Seigel [209] and Liu [210], we have

$$E[\max(Z + b)Z] = \dots;$$

where $\dots \in \mathbb{R}^k$ is the vector of probabilities, the j -th of which is given by equation (6.9), and we have used $\Sigma = A (A)^> = (\dots)(\dots)^>$ to denote the covariance matrix of Z .

Therefore, we have the first moment

$$M_1 = U (A)^{-1} A (A)^> = (\dots)^> ;$$

Correlation bound: By computation, we have

$$\begin{aligned} hM_1; (\dots)^>^{-1} \mathbf{1} i &= E[\max(Z + b)hZ; \dots \mathbf{1} i] \stackrel{(i)}{=} \frac{\mathbb{P}[\max(Z + b) > 0]}{E[(\max(Z + b))^2] E[hZ; \dots \mathbf{1} i]^2} \\ &\stackrel{(ii)}{=} \dots (\dots)^>^{-1} \mathbf{1} ; \end{aligned}$$

where step (i) follows from the definition (6.46) of the quantity \dots , and step (ii) from explicitly calculating the expectation and recalling the definition of \dots .

Positive semidefiniteness of M_2 : For some $u \in \mathbb{R}^d$, let $f(X) = \max(X + b)$ and $g_u(X) = hu; X^2$. Since g_u is an even function, we have $E[g_u(X)X] = 0$. Furthermore, since both f and g_u are convex, applying Lemma 42 (see Appendix 6.9.9) yields the bound

$$E[f(X)g_u(X)] \geq E[f(X)]E[g_u(X)];$$

¹¹One can also derive $M_1 = (\dots)^>$ directly applying Stein's lemma $E[Xf(X)] = E[r f(X)]$ to $f(x) := \max(X + b)$ so that $r f(x)$ equals \dots_j whenever x belongs to the region when j is maximized.

so that substituting yields the bound

$$u^T E[\max(X + b)XX^T]u = u^T E[\max(X + b)I]u:$$

Since this holds for all $u \in \mathbb{R}^d$, we have shown that the matrix $E[\max(X + b)(XX^T - I)]$ is positive semidefinite.

Calculating M_2 : We now use Stein's lemma to compute an explicit expression for the moment M_2 . By the preceding substitution, we have

$$\begin{aligned} M_2 &= E \left[\max(A X_1^k + b) \begin{matrix} X_1^k(X_1^k)^T & I_k \\ X_{k+1}^d(X_1^k)^T & X_{k+1}^d(X_{k+1}^d)^T \end{matrix} \right] \\ &= E \begin{matrix} \max(A X_1^k + b)(X_1^k(X_1^k)^T & I_k) & 0 \\ 0 & 0 & 0 \end{matrix} \end{aligned}$$

Once again using the substitution $Z = A X_1^k$ and $\Sigma = A(A)^T$, we have

$$M_2 = U(A)^T E[\max(Z + b)(ZZ^T - \Sigma)](A)^T(U)^T;$$

and applying Stein's lemma yields

$$E[\max(Z + b)(ZZ^T - \Sigma)] = \Sigma - \mathbf{1}\mathbf{1}^T;$$

where $\Sigma \in \mathbb{R}^{k \times k}$ denotes a matrix with entry i,j given by $\Sigma_{ij} = E[Z_i \mathbf{1}_{Z_j + b_j = \max}]$, and the final equality follows by symmetry of the matrix.

Simplifying further, we have

$$M_2 = U(A)^T A(U)^T:$$

Nullspace of M_2 : Notice that $\mathbf{1} = E[Z] = 0$, so that

$$M_2(\mathbf{1})^T = U(A)^T A(U)^T \mathbf{1} = 0:$$

Rank of M_2 and bound on $\lambda_{k-1}(M_2)$: By the previous claim, we have $\text{rank}(M_2) = k - 1$. Furthermore, the matrix M_2 has $d - k$ eigenvalues equal to zero, and the other k of its eigenvalues equal to those of Σ , all of which are positive (by the PSD property of M_2), and at least one of which is zero. Thus, it suffices to work with the eigenvalues of Σ ; in particular, a lower bound on $\lambda_{k-1}(\Sigma)$ directly implies a lower bound on $\lambda_{k-1}(M_2)$.

Let us first show that $\lambda_{k-1}(\Sigma) > 0$. Since we know that a zero-eigenvector of Σ is the all-ones vector $\mathbf{1}$, it suffices to show that $x^T \Sigma x \neq 0$ when $\mathbf{1}^T x = 0$. We use the shorthand $x \perp \mathbf{1}$ to denote any such vector.

We now explicitly evaluate the entries of the matrix Σ . We denote the j th column of this matrix by Σ_j . We have

$$\begin{aligned} \Sigma_j &= E[Z\mathbf{1} - Z_j + b_j = \max] \\ &= E[\mathbf{1} - Z_j\mathbf{1} - Z_j + b_j = \max] = E[(\mathbf{1} - Z_j - Z)\mathbf{1} - Z_j + b_j = \max] \\ &= \mathbf{1} - E[Z_j\mathbf{1} - Z_j + b_j = \max] = E[(\mathbf{1} - Z_j - Z)\mathbf{1} - Z_j + b_j = \max]: \end{aligned} \tag{6.50}$$

For any $x \succeq \mathbf{1}$, we have $x^T E[Z\mathbf{1} - fZ + b = \max] \succeq \mathbf{1} = 0$, so that in order to show that $x^T \Sigma_j \preceq 0$, it suffices to consider just the second term in the expression (6.50).

In order to focus on this term, consider the matrix Σ_j with column j given by

$$\Sigma_j = E[(\mathbf{1} - Z_j - Z)\mathbf{1} - Z_j - Z - b - b_j]:$$

where the indicator random variable above is computed element-wise. We are interested in evaluating the eigenvalues of the matrix Σ_j .

The quantity Σ_j can be viewed as the first moment of a (lower) truncated, multivariate Gaussian with (original) covariance matrix

$$\Sigma_j = (\mathbf{1} - e_j^T - \mathbf{1})A(A)^T(\mathbf{1} - e_j^T - \mathbf{1})^T:$$

Recalling the column vectors F^j defined (in equation (6.47)) for each $j \in [k]$ and applying [211, (11)] (see also Tallis [182] for a similar classical result), we may explicitly evaluate the vector Σ_j , as

$$\begin{aligned} \Sigma_j &= -\Sigma_j F^j \\ &\stackrel{(iii)}{=} (\mathbf{1} - e_j^T - \mathbf{1})A(A)^T G_j \end{aligned}$$

where in step (iii), we have let G_j denote a vector in \mathbb{R}^k with entry i given by

$$(G_j)_i = \begin{cases} (F^j)_i & \text{if } j \neq i \\ \sum_{k \neq j} (F^j)_k & \text{otherwise.} \end{cases}$$

Letting $G \in \mathbb{R}^{k \times k}$ denote the matrix with G_j as its j th column, and for $x \succeq \mathbf{1}$, we have

$$x^T (\Sigma_j) x = x^T G x;$$

since once again, for each $x \succeq \mathbf{1}$, we have $x^T (\mathbf{1} - e_j^T - \mathbf{1})A(A)^T(\mathbf{1} - e_j^T - \mathbf{1})^T x = 0$.

Now consider the matrix $\Sigma_j G$. In order to show the claimed bound, it suffices to show that $x^T \Sigma_j G x \preceq 0$ if $x \succeq \mathbf{1}$. We show this by combining two claims:

Claim 1: The nullspace of G is one-dimensional.

Claim 2: Both the left and right eigenvectors of $\Sigma_j G$ that correspond to this nullspace are not orthogonal to the $\mathbf{1}$ vector.

We show both claims concurrently. The nullspace of G is clearly non-trivial, since $\mathbf{1}^T G = 0$. Let us first show, by contradiction, that the left eigenvector corresponding to this nullspace dimension is

not orthogonal to the all-ones vector. Toward that end, let x_L denote the aforementioned left eigenvector which also satisfies $\sum_i x_{L,i} = 0$. By virtue of being a left eigenvector, x_L satisfies $x_L^T G = \lambda x_L^T$, or in other words, $x_L^T = \lambda^{-1} x_L^T G$. Since $x_L^T \mathbf{1} = 0$, we have $\lambda^{-1} x_L^T \mathbf{1} = 0$, but this contradicts the positive definiteness of G .

It remains to establish that the null-space of G is in fact only one-dimensional, and that its right eigenvector is not orthogonal to the all-ones vector. Notice that we may write the matrix as

$$G = (I - P^>) \text{diag}(G);$$

where we recall that the matrix P is defined with entries

$$P_{i,j} = \begin{cases} \sum_{k \notin j} P_{i,k} (F^j)_k & \text{if } i \notin j \\ 0 & \text{otherwise;} \end{cases}$$

Since all of the entries of P are positive and sum to 1 along the rows, the matrix P can be viewed as the transition matrix of a Markov chain. Furthermore, since this Markov chain communicates, it is irreducible and aperiodic, with only one eigenvalue equal to 1. Thus, the matrix $I - P^>$ is rank $k - 1$, thereby establishing that the nullspace of G is one-dimensional. Furthermore, the right eigenvector x_R of G is a non-negative vector by the Perron-Frobenius theorem, so that it cannot satisfy $\sum_i x_{R,i} = 0$.

We have thus established both claims, which together show that $\lambda_{k-1}(M_2) \notin 0$. Further noting that the matrix M_2 is positive semi-definite, we have

$$\lambda_{k-1}(M_2) \geq \min_{j \in [k]} G_{j,j} \geq \min_{i,j} (I - P^>)_{i,j}^{-1} \frac{1}{\sum_{k \in [k]} [(I - P^>)(I - P)]_{i,k}};$$

and this completes the proof of the claim, and consequently, the lemma. □

6.8 Proof of Theorem 14

Recall the matrix \mathcal{V} formed by appending a standard basis vector to \mathcal{U} . First, we show that there is a point among the randomly chosen initializations that is sufficiently close to the true parameters. Toward that end, let $c_0 := r + B_{\max}$ and define $\tilde{v}_j = \mathcal{V}^{-1} v_j$ for each $j \in [k]$ and $\tilde{u}_\ell \in [M]$. Let

$$\tilde{v}_j^\# := \operatorname{argmin}_{\tilde{u} \in [M]} \max_{j \in [k]} \|c_0 \tilde{u} - \tilde{v}_j\|_2;$$

and define the event

$$E_1(M; r) := \max_{j \in [k]} \|c_0 \tilde{v}_j^\# - \tilde{v}_j\|_2 \leq r + B_{\max} \|\mathcal{U} \mathcal{U}^>\|_{\text{op}};$$

in words $E_1(M; r)$ is the event that *none* of the randomly initialized points (when scaled by a fixed constant c_0) is close to the true parameters. The following lemma bounds the probability of such an event provided M is sufficiently large.

Lemma 35. *If $M \geq 1 + \frac{B_{\max}}{r} k^2 \log(1/\epsilon)$, then $\Pr \{f_{E_1}(M; r) \geq g\} \leq \epsilon$.*

Taking the lemma as given, let us now proceed to the proof of the theorem. Define the shorthand

$$P(\beta_1, \dots, \beta_k) := \frac{2}{n} \sum_{i=n-2+1}^n \max_{j \in [k]} h_{i; j}^2$$

for each set of parameters $\beta_1, \dots, \beta_k \in \mathbb{R}^{d+1}$.

For each $\beta \in [M]$, let

$$c_\beta := \operatorname{argmin}_{c \geq 0} \frac{2}{n} \sum_{i=n-2+1}^n y_i - c \max_{j \in [k]} h_{i; j}^2$$

and recall that $\hat{\beta}$ is the index returned by the algorithm. Also note that trivially, we have $c_{\hat{\beta}} > 0$ with probability tending to 1 exponentially in n , so that this pathological case in which the initial partition is random can be ignored.

Due to sample splitting, the parameters β_j are independent of the noise sequence $\{g_{i=n-2+1}^n\}$. Thus, applying Lemma 44 from Appendix 6.9.13 yields the bound

$$\Pr_{\beta} \left[P(c_{\hat{\beta}_1}, \dots, c_{\hat{\beta}_k}) \leq c_1 \right] \leq \min_{\beta \in [M]} P(c_{\beta_1}, \dots, c_{\beta_k}) + \frac{2t^{p \log M + c_1}}{n} e^{-c_2 n t^{p \log M + c_1}}$$

valid for all $t \geq \frac{p \log M + c_1}{c_2}$ and suitable universal constants c_1 and c_2 . Setting $t = \frac{p \log M + c_1}{c_2}$ and on this event, we have

$$P(c_{\hat{\beta}_1}, \dots, c_{\hat{\beta}_k}) \leq c_1 P(c_{\beta_1^\#}, \dots, c_{\beta_k^\#}) + c_1 \frac{2 \log M}{n}$$

with probability greater than $1 - e^{-c_2 n}$.

To complete the proof, let $C(\min; k) := c_2 \frac{k}{\min}^3$ for a suitable constant c_2 and apply

Lemma 43 twice (note that here we use the assumption $n \geq Cd \frac{k^3}{\min} \log^2(k = \min)$) in order to obtain

$$\begin{aligned} \times \min_{j \geq 2[k]} k_j &\leq c_1 j^0 k^2 = C(\min; k) P(c_1 \hat{\cdot}_1; \dots; \hat{\cdot}_k) \\ &\leq C_1 C(\min; k) \left(P(c_0 \hat{\cdot}_1^{\#}; \dots; c_0 \hat{\cdot}_k^{\#}) + \frac{2 \log M}{n} \right) \\ &\leq C_1 C(\min; k) \left(2 \sum_{j=1}^k k c_0 j^{\#} j k^2 + \frac{2 \log M}{n} \right) \\ &\leq C_1 C(\min; k) \left(2k \max_{j \geq 2[k]} k c_0 j^{\#} j k^2 + \frac{2 \log M}{n} \right) \\ (ii) \quad &\leq C_1 C(\min; k) \left(4k r^2 + B_{\max}^2 \sum_{j \geq 2[k]} \mathbf{b} \mathbf{b}^{\top} \right) U(U)^{\sum_{j \geq 2[k]} \mathbf{b} \mathbf{b}^{\top}} + \frac{2 \log M}{n} \end{aligned}$$

on an event of suitably high probability, where step (ii) follows from Lemma 35 and on the event $E_1^c(M; r)$. Note that implicitly, we have also used a union bound over all M choices of our parameters, which leads to the overall probability bound of $1 - c_1 k M \exp - c_2 \frac{4}{k^4 \log^2(k = \min)}$.

Finally, note that provided the RHS above is less than $\epsilon^2 = 4$, each minimum on the LHS is attained for a unique index j^0 . This condition is ensured by the sample size assumption of the theorem; thus, we have

$$\begin{aligned} \min_{c > 0} \text{dist} \quad & \sum_{j=1}^n c_j^{(0)} \mathbf{O}_k; \quad j \quad j=1}^k \quad c_1 C(\min; k) \\ & 4k r^2 + B_{\max}^2 \sum_{j \geq 2[k]} \mathbf{b} \mathbf{b}^{\top} \quad U(U)^{\sum_{j \geq 2[k]} \mathbf{b} \mathbf{b}^{\top}} + \frac{2 \log M}{n} \end{aligned}$$

Combining the various probability bounds then completes the proof. □

6.8.1 Proof of Lemma 35

Recall that U is a matrix of orthonormal columns spanning the k -dimensional subspace spanned by the vectors $f_1; \dots; f_k$. Define the matrix

$$V = \begin{pmatrix} U & 0 \\ 0 & 1 \end{pmatrix};$$

for each $j \geq 2[k]$, we have $j = V_j$ for some vector $j \in \mathbb{R}^{k+1}$. Also define the rotation matrix

$$O = \begin{pmatrix} \mathbf{b}^{\top} U & 0 \\ 0 & 1 \end{pmatrix};$$

so that $\forall O \quad V = \begin{pmatrix} \theta\theta^T U \\ 0 \end{pmatrix} \begin{pmatrix} U \\ 0 \end{pmatrix}$ and we have $\| \begin{pmatrix} \theta\theta^T U \\ 0 \end{pmatrix} \|_k = \| \theta\theta^T U \|_k$ for any unitarily invariant norm $\|\cdot\|_k$.

Now for each $j \in [k]$ and $\cdot \in [M]$, applying the triangle inequality yields

$$\begin{aligned} \|c_{0j} \cdot - \cdot\|_k &\leq \|c_{0j} \theta\theta^T \cdot\|_k + \| \theta\theta^T \cdot \|_k + \| \theta\theta^T \cdot \|_k \\ &\leq \|c_{0j} \cdot\|_k + k_j \| \theta\theta^T \cdot \|_k \\ &\leq \|c_{0j} \cdot\|_k + B_{\max} \| \theta\theta^T \cdot \|_k \end{aligned}$$

For each pair $(\cdot; j)$, define the event

$$E_j^{\cdot}(r) := \|c_{0j} \cdot - \cdot\|_k \leq r$$

We claim that if $M \geq 1 + \frac{B_{\max}}{r} k^2 \log(1/\rho)$, we have

$$\Pr [\cdot \in [M] \setminus \bigcup_{j \in [k]} E_j^{\cdot}(r)] \leq \rho \tag{6.51}$$

Indeed, such a claim suffices, since it implies that

$$\min_{\cdot \in [M]} \max_{j \in [k]} \|c_{0j} \cdot - \cdot\|_k \leq r + B_{\max} \| \theta\theta^T \cdot \|_k$$

with probability exceeding $1 - \rho$, thereby proving the theorem. It remains to establish claim (6.51).

Denote by ρ the probability with which for a fixed pair $(\cdot; j)$, we have $\|c_{0j} \cdot - \cdot\|_k \leq r$. This is the ratio of the volume of the ℓ_2 -ball of radius r and the ℓ_2 -ball of radius c_0 , and so we have $\rho = \frac{r^k}{r + B_{\max}}^k$. Thus, we have

$$\begin{aligned} \Pr [\cdot \in [M] \setminus \bigcup_{j \in [k]} E_j^{\cdot}(r)] &\leq (1 - \rho^k)^M \\ &\leq e^{-\rho^k M} \end{aligned} \tag{i}$$

where step (i) holds provided $M \geq \frac{1}{\rho^k} \log(1/\rho)$. Putting together the pieces completes the proof. □

6.9 Technical Lemmas and Background

6.9.1 Fundamental limits

In this section, we present two lower bounds: one on the minimax risk of parameter estimation, and another on the risk of the least squares estimator with side-information.

6.9.2 Minimax lower bounds

Recall our notation X for the matrix whose columns consist of the parameters β_1, \dots, β_k . Assume that the intercepts b_1, \dots, b_k are identically zero, so that $\beta_j = x_j$ and $\beta = X$. For a fixed matrix X , consider the observation model

$$y = \max(X\beta) + \epsilon; \tag{6.52}$$

where $y \in \mathbb{R}^n$, the noise $\epsilon \sim N(0; \sigma^2 I_n)$ is chosen independently of X , and the max function is computed row-wise.

Proposition 16. *There is an absolute constant C such that the minimax risk of estimation satisfies*

$$\inf_{\hat{\beta}} \sup_{X \in \mathbb{R}^{n \times d}} \mathbb{E} \frac{1}{n} \sum_{i=1}^n \|X(\hat{\beta} - \beta^*)\|_F^2 \geq C \frac{\sigma^2 kd}{n}.$$

Here, the expectation is taken over the noise ϵ , and infimum is over all measurable functions of the observations $(X; y)$. Indeed, when X is a random Gaussian matrix, it is well conditioned and has singular values of the order $\sqrt{\frac{d}{n}}$, so that this bound immediately yields

$$\inf_{\hat{\beta}} \sup_{X \in \mathbb{R}^{n \times d}} \mathbb{E} \frac{1}{n} \sum_{i=1}^n \|\hat{\beta} - \beta^*\|_F^2 \geq C \frac{\sigma^2 kd}{n}.$$

Let us now provide a proof of the proposition.

Proof. The proof is based on a standard application of Fano’s inequality (see, e.g., Wainwright [13, Chapter 15] and Tsybakov [212, Chapter 2]). For a tolerance level $\epsilon > 0$ to be chosen, we choose the local set

$$F = \{X \in \mathbb{R}^{n \times k} \mid \|X\|_F \leq 4 \sqrt{\frac{d}{kn}}\}$$

and let X^1, \dots, X^M be a $2\sqrt{\frac{d}{kn}}$ -packing of the set in the Frobenius norm. This can be achieved by packing the j -th column $Q_j := X_{j \cdot}^T \in \mathbb{R}^n$ at level $2\sqrt{\frac{d}{n}}$ in ℓ_2 norm for all $j \in [k]$. Standard results yield the bound $\log M \leq C_1 kd \log 2$.

For each $i \neq j$, we have

$$2\sqrt{\frac{d}{k}} \leq \frac{\|X(\beta^i - \beta^j)\|_F}{\sqrt{n}} \leq 8\sqrt{\frac{d}{k}}; \tag{6.53}$$

Let $P_j = N(\max(X(\beta^j)); \sigma^2 I_n)$ denote the distribution of the observation vector y when the true parameter is β^j . We thus obtain

$$D_{\text{KL}}(P_j \parallel P_i) = \frac{1}{2\sigma^2} \|\max(X(\beta^j)) - \max(X(\beta^i))\|_2^2 \geq \frac{1}{2\sigma^2} \frac{\|X(\beta^j - \beta^i)\|_F^2}{n}.$$

where the inequality follows since the max function is 1-Lipschitz in ℓ_2 norm. Putting together the pieces yields

$$D_{\text{KL}}(P_j \parallel P_i) \leq \frac{32k^2 n}{2};$$

so that the condition

$$\frac{\frac{1}{M^2} \sum_{i,j} D_{\text{KL}}(P_j \parallel P_i) + \log 2}{\log M} \leq \frac{1}{2}$$

is satisfied with the choice $\epsilon^2 = C \frac{2d}{n}$. Finally, applying Fano's inequality (see, e.g., [13, Proposition 15.2]) yields the minimax lower bound

$$\inf_b \sup P \frac{1}{n} \sum_{j=1}^k X(b_j) \geq C \frac{2kd}{n}; \tag{6.54}$$

□

6.9.3 Performance of unconstrained least squares with side-information

In this section, we perform an explicit computation when $k = 3$ and $d = 2$ to illustrate the cubic \min dependence of the error incurred by the unconstrained least squares estimator, even when provided access to the true partition $\mathcal{S}_j = \{x_1, \dots, x_3\}_{j=1}^3$.

We begin by defining our unknown parameters. For a scalar $\theta \in (0, \pi/4)$, let

$$x_1 = \sin(\theta) e_1; \quad x_2 = \cos(\theta) e_2; \quad \text{and} \quad x_3 = \cos(\theta) e_2;$$

and set $b_j = 0$ for $j = 1; 2; 3$.

Now an explicit computation yields that the cone on which x_1 attains the maximum is given by

$$C_1 := \{x \in \mathbb{R}^2 : \langle x, x_1 \rangle \geq \max_{j \in [k]} \langle x, x_j \rangle\} = \{x \in \mathbb{R}^2 : x_1 \geq 0; x_2 \geq x_1 \tan(\theta)\};$$

Now consider a Gaussian random vector in \mathbb{R}^2 truncated to that cone. In particular, consider a two-dimensional random variable W with density $\phi(x) \mathbf{1}_{x \in C_1} / \text{vol}(C_1)$, where ϕ is the two-dimensional standard Gaussian density and $\text{vol}(S)$ denotes the Gaussian volume of a set S . Note that we have $\text{vol}(C_1) = \frac{\pi}{2} \cos^2(\theta)$ by construction.

Let us now compute the second order statistics of W , using polar coordinates with R^2 denoting a χ^2_2 random variable. The individual second moments take the form

$$E[W_1^2] = \frac{1}{2} E[R^2] \int_0^{\pi/2} \cos^2(\theta) d\theta = 1;$$

and

$$E[W_2^2] = -E[R^2] \frac{1}{2} \sin^2 d = \frac{1}{2} (\sin(2\theta))^2;$$

On the other hand, the cross terms are given by

$$E[W_1 W_2] = -E[R^2] \frac{1}{2} \sin(\theta) \cos(\theta) d = 0;$$

Thus, it can be verified that for all $\theta \in [0; \pi/4]$, the second moment matrix of W has a tuple of singular values $(1; c^2)$ for an absolute constant c .

Let us now use this calculation to reason about the least squares estimator. Drawing n samples from the Gaussian distribution on \mathbb{R}^2 , we expect $n_1 \approx n$ of them to fall in the set C_1 with high probability. Collect these samples as rows of a matrix X_1 . When n is large enough, i.e., on the order of n^3 , standard bounds (as in Section 6.6.1) can be applied to explicitly evaluate the singular values of the matrix $\frac{1}{n_1} X_1^T X_1$. In particular, we have

$$\lambda_1 \left(\frac{1}{n_1} X_1^T X_1 \right) = c^d \quad \text{and} \quad \lambda_2 \left(\frac{1}{n_1} X_1^T X_1 \right) = c^{-2};$$

We now provide the $n_1 \times 2$ matrix X_1 as side information to a procedure whose goal is to estimate the unknown parameters. Clearly, given this matrix, a natural procedure to run in order to estimate θ_1 is the (unconstrained) least squares estimator on these samples, which we denote by $\hat{\theta}_1$. As is well known, the rate obtained (in the fixed design setting) by this estimator with σ -sub-Gaussian noise is given by

$$\begin{aligned} E \|\hat{\theta}_1 - \theta_1\|^2 &= \frac{2}{n_1} \text{tr}(X_1^T X_1)^{-1} \\ &= \frac{2}{n_1} (c^{-2} + c^d) \\ &= \frac{2}{3n}; \end{aligned}$$

where the last two relations hold with exponentially high probability in n . We have thus shown that the unconstrained least squares estimator (even when provided with additional side information) attains an error having cubic dependence on σ_{\min} . While this does not constitute an information theoretic lower bound, our calculation provides some evidence for the fact that, at least when viewed in isolation, the dependence of our statistical error bound (6.15) on σ_{\min} is optimal for Gaussian covariates.

6.9.4 Background and technical lemmas used in the proof of Theorem 12

In this section, we collect statements and proofs of some technical lemmas used in the proofs of our results concerning the AM algorithm.

6.9.5 Bounds on the “volumes” of wedges in \mathbb{R}^d

For a pair of scalars $(w; w^\flat)$ and d -dimensional vectors $(u; u^\flat)$, recall that we define the *wedge* formed by the $d + 1$ -dimensional vectors $v = (u; w)$ and $v^\flat = (u^\flat; w^\flat)$ as the region

$$W(v; v^\flat) = \{x \in \mathbb{R}^d : (hx; ui + w) \leq (hx; u^\flat i + w^\flat) \leq 0\}$$

Note that the wedge is a purely geometric object.

For any set $C \subset \mathbb{R}^d$, let

$$\text{vol}(C) = \int_{x \sim N(0; I_d)} \mathbb{1}_{x \in C} dx$$

denote the volume of the set under the measure corresponding to the covariate distribution.

We now bound the volume of a wedge for the Gaussian distribution.

Lemma 36. *Suppose that for a pair of scalars $(w; w^\flat)$, d -dimensional vectors $(u; u^\flat)$, and $v = (u; w)$ and $v^\flat = (u^\flat; w^\flat)$, we have $\frac{\|v\| \|v^\flat\|}{\|u\| \|u^\flat\|} < 1/2$. Then, there is a positive constant C such that*

$$\text{vol}(W(v; v^\flat)) \geq C \frac{\|v\| \|v^\flat\|}{\|u\| \|u^\flat\|} \log^{1/2} \frac{2\|u\| \|u^\flat\|}{\|v\| \|v^\flat\|} ;$$

Proof of Lemma 36

Using the notation $x = (x; 1) \in \mathbb{R}^{d+1}$ to denote the appended covariate, we have

$$\text{vol}(W(v; v^\flat)) = \Pr \{ (hx; u; w) \leq (hx; u^\flat; w^\flat) \leq 0 \}$$

where the probability is computed with respect to Gaussian measure.

In order to prove a bound on this probability, we begin by bounding the associated indicator random variable as

$$\mathbb{1}_{(hx; u; w) \leq (hx; u^\flat; w^\flat) \leq 0} \leq \mathbb{1}_{(hx; u; w) \leq (hx; u^\flat; w^\flat) \leq 0} + \mathbb{1}_{(hx; u; w) \leq (hx; u^\flat; w^\flat) \leq 0} \leq \mathbb{1}_{(hx; u; w) \leq (hx; u^\flat; w^\flat) \leq 0} + \mathbb{1}_{(hx; u; w) \leq (hx; u^\flat; w^\flat) \leq 0} ; \tag{6.55}$$

where inequality (6.55) holds for all $t \geq 0$. In order to bound the expectation of the second term, we write

$$\Pr \{ (hx; u; w) \leq (hx; u^\flat; w^\flat) \leq 0 \} \leq \Pr \{ \chi_{nc}^2 \geq t \} \stackrel{(i)}{\leq} \frac{e^{-t}}{t}$$

where χ_{nc}^2 is a non-central chi-square random variable centered at $\frac{w}{\|u\|}$, and step (i) follows from standard χ^2 tail bounds (see Lemma 41).

It remains to control the expectation of the first term on the RHS of inequality (6.55). We have

$$\Pr \left\{ \sum_{i=1}^n v_i^2 \geq t \right\} = \Pr \left\{ \sum_{i=1}^n u_i^2 + 2(w^T w)^2 \geq t \right\}$$

$$\Pr \left\{ \sum_{i=1}^n u_i^2 \geq \frac{t}{2} \right\} \leq \Pr \left\{ \sum_{i=1}^n v_i^2 \geq \frac{t}{2} \right\}.$$

Now, invoking a standard sub-exponential tail bound on the upper tail of a χ^2 random variable yields

$$\Pr \left\{ \sum_{i=1}^n v_i^2 \geq t \right\} \leq c_1 \exp \left(-\frac{c_2}{k} \frac{t}{n} \right) \leq \frac{c_1}{k} \exp \left(-\frac{c_2}{k} \frac{t}{n} \right)$$

$$\leq c_1 \exp \left(-\frac{c_2}{k} \frac{t}{n} \right) \leq \frac{c_1}{k} \exp \left(-\frac{c_2}{k} \frac{t}{n} \right).$$

Putting all the pieces together, we obtain

$$\text{vol}(W(v; v^{\otimes 2})) \leq c_1 \exp \left(-\frac{c_2}{k} \frac{t}{n} \right) + \frac{et}{k} \exp(-t/2).$$

Substituting $t = 2c_k \log(2k) = 2c_k \log(2k)$, which is a valid choice provided $\frac{2c_k}{k} < 1/2$, yields the desired result. \square

6.9.6 Growth Functions and Uniform Empirical Concentration

We now briefly introduce growth functions and uniform laws derived from them, and refer the interested reader to Mohri et al. [213] for a more in-depth exposition on these topics.

We define growth functions in the general multi-class setting [189]. Let X denote a set, and let F denote a family of functions mapping $X \rightarrow \{0, 1, \dots, k-1\}$. The *growth function* $F : \mathbb{N} \rightarrow \mathbb{R}$ of F is defined via

$$F(n) := \max_{x_1, \dots, x_n \in X} \text{abs} \{ f(x_1); f(x_2); \dots; f(x_n) \} : f \in F$$

In words, it is the cardinality of all possible labelings of n points in the set X by functions in the family F .

A widely studied special case arises in the case $k = 2$, with the class of binary functions. In this case, a natural function class F is formed by defining \mathcal{C} to be a family of subsets of X , and identifying each set $C \in \mathcal{C}$ with its indicator function $f_C := 1_C : X \rightarrow \{0, 1\}$. In this case, define $F_C = \{f_C : C \in \mathcal{C}\}$. A bound on the growth function for such binary function provides following guarantee for the uniform convergence for the empirical measures of sets belonging to \mathcal{C} .

Lemma 37 (Theorem 2 in [214]) *Let \mathcal{C} be a family of subsets of a set X . Let μ be a probability measure on X , and let $\hat{\mu}_m := \frac{1}{m} \sum_{i=1}^m \mu_i$ be the empirical measure obtained from m independent copies of a random variable X with distribution μ . For every u such that $m \geq 2/u^2$, we have*

$$\Pr \left\{ \sup_{C \in \mathcal{C}} \hat{\mu}_m(C) - \mu(C) \geq u \right\} \leq 4 \exp(-2mu^2) \quad (6.56)$$

We conclude this section by collecting some results on the growth functions of various function classes. For our development, it will be specialized to the case $X = \mathbb{R}^d$.

Define the class of binary functions F_H as the set of all functions of the form

$$f_{i,b}(x) := \frac{\text{sgn}(hx; i + b) + 1}{2};$$

specifically, let $F_H := \{f_{i,b} : i \in \mathbb{R}^d; b \in \mathbb{R}\}$. In particular, these are all functions that can be formed by a d -dimensional hyperplane.

Using the shorthand $B_1^k = \{f_{i,b} : i \in \mathbb{R}^d; b \in \mathbb{R}\}$, define the binary function

$$g_{i_1, b_1^k}(x) := \bigwedge_{i=1}^k f_{i, b_i}(x);$$

and the binary function class corresponding to the intersection of k hyperplanes

$$G_{H^k} := \{g_{i_1, b_1^k} : i_1 \in \mathbb{R}^d; b_1 \in \mathbb{R}; \dots; i_k \in \mathbb{R}^d; b_k \in \mathbb{R}\};$$

Finally, we are interested in the argmax function over hyperplanes. Here, define the function

$$m_{i_1, b_1^k}(x) := \underset{j \in [k]}{\text{argmax}} (h_j; x_i + b_j) - 1;$$

mapping $\mathbb{R}^d \rightarrow \{0, \dots, k-1\}$. The function class that collects all such functions is given by

$$\mathcal{M}_k := \{m_{i_1, b_1^k} : i_1 \in \mathbb{R}^d; b_1 \in \mathbb{R}; \dots; i_k \in \mathbb{R}^d; b_k \in \mathbb{R}\};$$

The following results bound the growth functions of each of these function classes. We first consider the function classes F_H and G_{H^k} , for which bounds on the VC dimension directly yield bounds on the growth function.

Lemma 38 (Sauer-Shelah (e.g. Section 3 of Mohri et al. [213])). *We have*

$$F_H(n) \leq \frac{en}{d+1}^{d+1}; \text{ and} \tag{6.57}$$

$$G_{H^k}(n) \leq \frac{en}{d+1}^{k(d+1)}; \tag{6.58}$$

The second bound can be improved (see, e.g. [215]), but we state the version obtained by a trivial composition of individual halfspaces.

The following bound on the growth function of the class \mathcal{M}_k is also known.

Lemma 39 (Theorem 3.1 of Daniely et al. [189]). *For an absolute constant C , we have*

$$\mathcal{M}_k(n) \leq \frac{en}{Ck(d+1) \log(kd)}^{Ck(d+1) \log(kd)};$$

6.9.7 Singular value bound

We now state and prove a technical lemma that bound the maximum singular value of a matrix whose rows are drawn from a sub-Gaussian distribution.

Lemma 40. *Suppose that the covariates are drawn i.i.d. from a ψ -sub-Gaussian distribution. Then for a fixed subset $S \subseteq [n]$ of size $|S| = m$ and each $t \geq 0$, we have*

$$\Pr \left(\max_{S \subseteq [n]} \sum_{i \in S} \lambda_i^2 \geq t \right) \leq e^{-2} \left(\frac{e^{\psi^2}}{d} + d + t \right)^m \leq 2e^{-\min\{ft; t^2g\}}$$

where $e = \max\{f; 1/g\}$.

Proof of Lemma 40

Let $\{z_i g_{i=1}^d$ denote i.i.d. Rademacher variables, and collect these in an d -dimensional vector Z . Let $D = \text{diag}(z)$ denote a diagonal matrix, and note that by unitary invariance of the singular values, the singular values of the matrix $e_S = D \cdot S$ are the same as those of S .

By construction, the matrix e_S has i.i.d. rows, and the i -th row is given by $z_i(x_i; 1)$. For a $d + 1$ dimensional vector $e = (h; w)$ with $h \in \mathbb{R}^d$ and $w \in \mathbb{R}$, we have

$$\begin{aligned} \mathbb{E} \exp(h^T z_i(x_i; 1)) &= \frac{e^w}{2} \mathbb{E} [\exp(h^T x_i)] + \frac{e^{-w}}{2} \mathbb{E} [\exp(-h^T x_i)] \\ &= \exp\left(\frac{1}{2} \|h\|^2\right) \frac{1}{2} (e^w + e^{-w}) \\ &= \exp\left(\frac{1}{2} \|h\|^2\right) \exp\left(\frac{w^2}{2}\right) \exp\left(\frac{1}{2} \|h\|^2\right) \end{aligned}$$

where we have used the fact that x_i is zero-mean and ψ -sub-Gaussian.

Since the rows of e_S are i.i.d., zero-mean, and ψ -sub-Gaussian, applying [13, Theorem 6.2] immediately yields the lemma. \square

6.9.8 Anti-concentration of χ^2 random variable

The following lemma shows the anti-concentration of the central and non-central χ^2 random variable.

Lemma 41. *Let Z and Z^0 denote central and non-central χ^2 random variables with d degrees of freedom, respectively. Then for all $p \in [0, 1]$, we have*

$$\Pr\{Z^0 \leq pg\} \leq \Pr\{Z \leq pg\} \frac{p}{1-p} \exp\left(-\frac{p}{1-p}\right) = \exp\left(-\frac{p}{1-p} \log\left(\frac{p}{1-p} + \frac{p}{1-p}\right)\right) \quad (6.59)$$

Proof of Lemma 41

The fact that $Z^0 \stackrel{st.}{\preceq} Z$ follows from standard results that guarantee that central χ^2 random variables stochastically dominate their non-central counterparts.

The tail bound is a simple consequence of the Chernoff bound. In particular, we have for all $\rho > 0$ that

$$\begin{aligned} \Pr f(Z) - \rho g &= \Pr \left[\exp(-\rho Z) \geq \exp(-\rho g) \right] \\ &= \exp(\rho) E[\exp(-\rho Z)] \\ &= \exp(\rho) (1 + 2^{-\rho})^{\frac{1}{2}} \end{aligned} \tag{6.60}$$

where in the last step, we have used $E[\exp(-\rho Z)] = (1 + 2^{-\rho})^{\frac{1}{2}}$, which is valid for all $\rho > 1=2$. Minimizing the last expression over $\rho > 0$ then yields the choice $\rho = \frac{1}{2} \ln \frac{1}{1-\epsilon}$, which is greater than 0 for all $0 < \epsilon < 1$. Substituting this choice back into equation (6.60) proves the lemma. \square

6.9.9 Background and technical lemmas used in the proof of Theorem 13

We begin by stating a result of Hargreaves [216, Theorem 1.2] (see also Hu [217]) that guarantees that convex functions of a Gaussian random vector are positively correlated. We state it below in the notation of the current paper.

Lemma 42 ([216]). *Let f and g be two convex functions on \mathbb{R}^d , and let X be a standard d -dimensional Gaussian vector. Then*

$$E[f(X)g(X)] \geq (1 + hm(g); m(f)) E[f(X)]E[g(X)] \tag{6.61}$$

where for any d -variate function h , we have $m(h) = \frac{E[Xh(X)]}{E[h(X)]}$.

We also prove Lemma 34, which was used in the proof of Theorem 13.

6.9.10 Proof of Lemma 34

We prove each bound separately. First, by the rotation invariance of the Gaussian distribution, we may assume that $U = [e_1^d \dots e_k^d]$, so that the max is computed as a function of the k coordinates X_1, \dots, X_k .

We also define some events that we make use of repeatedly in the proofs. For each $i \in [n]$, define the events

$$\begin{aligned} E_i &= \{ |X_{ij}| \leq \sqrt{\frac{\log(2nk)}{5}} \text{ for all } 1 \leq j \leq k; \text{ and} \\ F_i &= \{ |X_{ij}| \leq \sqrt{\frac{\log(2n)}{5}} \} \end{aligned}$$

Note that by standard sub-Gaussian tail bounds, we have $\Pr F_i^c \leq 2n^{-12}$ and $\Pr E_i^c \leq 2n^{-12}$ for each $i \in [n]$. For notational convenience, define for each i the modified covariate $Z_i = X_i - \mathbb{1} F_i^c$.

We have

$$\| \max_{j \geq [k]} (z_j + b) \|_2 \leq C \max_{j \geq [k]} k_j k_1^p \overline{\log(nk)} + \|b\|_2 \leq C^p \overline{\log(nk)} \ \&$$

almost surely, where in the second bound, we have used the shorthand $\& = \max_{j \geq [k]} k_j k_1 + \|b\|_2 k_1$ as defined in equation (6.18). With this setup in place, we are now ready to prove both deviation bounds.

Proof of bound (6.49a)

Let us first bound the deviation of the first moment. We work with the decomposition

$$\mathbb{M}_1 \quad M_1 = \frac{2}{n} \sum_{i=1}^{\mathfrak{X}^2} \max(x_i + b) x_i \underbrace{\left\{ \frac{E[\max(X + b) X]}{T_i^1} \right\}}_{T_i^1} + \frac{2}{n} \sum_{i=1}^{\mathfrak{X}^2} \max(z_i + b) z_i \underbrace{\left\{ \frac{E[\max(Z + b) Z]}{T_i^2} \right\}}_{T_i^2} :$$

By triangle inequality, it suffices to bound the norms of each of the two sums separately. We now use the further decomposition

$$T_i^1 = \underbrace{\max(x_i + b) x_i}_{P_i} \underbrace{\max(z_i + b) z_i}_{Q_i} + \underbrace{\max(z_i + b) z_i}_{Q_i} \underbrace{\left\{ \frac{E[\max(Z + b) Z]}{T_i^2} \right\}}_{R_i} + \underbrace{\left\{ \frac{E[\max(Z + b) Z]}{T_i^2} \right\}}_{R_i} \underbrace{\left\{ \frac{E[\max(X + b) X]}{T_i^1} \right\}}_{R_i} :$$

Since $Z_i = X_i$ with probability greater than $1 - 2n^{-12}$, the term $P_i = 0$ on this event.

Also, for each fixed $j \geq [k]$, applying the Hoeffding inequality yields the bound

$$\Pr \left\{ \frac{2}{n} \sum_{i=1}^{\mathfrak{X}^2} Q_{i,j} \geq t \right\} \leq 2 \exp \left\{ - \frac{nt^2}{8C^2 \&^2 (\log(nk))^2} \right\} :$$

On the other hand, for $j \geq [d] n [k]$, we have

$$\begin{aligned} \frac{2}{n} \sum_{i=1}^{\mathfrak{X}^2} Q_{i,j} &\leq \frac{2}{n} \sum_{i=1}^{\mathfrak{X}^2} Z_{i,j} \\ &= \& \frac{2}{n} \sum_{i=1}^{\mathfrak{X}^2} X_{i,j} : \end{aligned}$$

Standard Gaussian tail bounds then yield

$$\Pr \left\{ \frac{2}{n} \sum_{i=1}^{\mathfrak{X}^2} Q_{i,j} \geq t \right\} \leq \frac{2}{t^p \overline{\log(nk)}} \exp \left\{ - \frac{nt^2}{8} \right\}$$

for each $t \geq 0$. Putting together the pieces with a union bound and choosing constants appropriately, we then have

$$\Pr \left\{ \sum_{i=1}^8 Q_i \geq \frac{1}{n} Ck^2 (\log(nk))^2 + \frac{1}{n} C^{\theta} (d-k)^2 \log(nk) \right\} \leq 2dn^{-12};$$

It remains to handle the final terms $fR_i g_{i=1}^n$. Note that when $j \geq [k]$, we have $R_{i,j} = 0$. It therefore suffices to bound the various $R_{i,j}$ terms when $j \leq [k]$. We have

$$\begin{aligned} jR_{i,j} &= jE[\max(z_i + b)z_{i,j}] - E[\max(x_i + b)x_{i,j} \mathbf{1}_{fE_i g}] - E[\max(x_i + b)x_{i,j} \mathbf{1}_{fE_i^c g}] \\ &= jE[\max(x_i + b)x_{i,j} \mathbf{1}_{fE_i^c g}] \end{aligned}$$

Expanding this further, we have

$$\begin{aligned} jR_{i,j} &= E[\max(jh_{i,j} + x_{i,j} + jb)jx_{i,j} \mathbf{1}_{fE_i^c g}] \\ &= E[jx_{i,j} jkx_{i,j} (k - k_{1,j} + kb - k_1) \mathbf{1}_{fE_i^c g}] \\ &= E[jx_{i,j} jkx_{i,j} \mathbf{1}_{fE_i^c g}] \\ &\leq \sum_{j=1}^k E[jx_{i,j} jx_{i,j} \mathbf{1}_{fE_i^c g}]. \end{aligned}$$

Note that for a pair (X_1, X_2) of i.i.d. random variables, Jensen's inequality yields the bounds

$$\begin{aligned} E[jX_1 X_2 \mathbf{1}_{fX_1; X_2 \leq g}] &\leq E[X_1^2 \mathbf{1}_{fjX_1 \leq g}] \text{ for all } 0; \text{ and} \\ E[jX_1 \mathbf{1}_{fjX_1 \leq g}] &\leq E[X_1^2 \mathbf{1}_{fjX_1 \leq g}] \text{ for all } 1: \end{aligned}$$

Furthermore, if X is a standard Gaussian random variable, then a simple calculation (see also Burkardt [218]) yields the bound

$$E[X^2 j jX_j \leq g] \leq \frac{1}{2} e^{-\frac{g^2}{2}}; \text{ for all } g \geq 0.$$

Putting together the pieces with with $\rho = 5 \sqrt{\log(2nk)}$, we have

$$jR_{i,j}^2 \leq Ck^2 \log(nk)(nk)^{-24};$$

and summing over $j \leq [k]$ yields the bound

$$\sum_{i=1}^8 R_i \leq Ck^2 \log(nk)(nk)^{-24};$$

Finally, putting together the pieces with a union bound yields the desired bound on the random variable $\sum_{i=1}^8 T_i$.

The second term can be bounded more easily; in particular, on the intersection of the events $\mathcal{F}_i \mathcal{G}_{i=1}^n$, we have

$$\frac{2}{n} \sum_{i=1}^n T_i^2 \leq C^2 \log n \frac{2}{n} \sum_{i=1}^n x_i^2 \leq C^2 \frac{(d + \log n) \log n}{n};$$

where the final bound holds with probability greater than $1 - cn^{-10}$. Finally, putting the bounds together yields the result.

Proof of bound (6.49b)

Once again, we decompose the required term as

$$\mathcal{M}_2 = M_2 = \frac{2}{n} \sum_{i=1}^n \max\left(\frac{x_i + b}{Z_i}, \frac{x_i x_i^>}{Z_i^2} l_d\right) + \frac{2}{n} \sum_{i=1}^n \max\left(\frac{x_i x_i^>}{Z_i^2} l_d, \frac{x_i x_i^>}{Z_i^2} l_d\right);$$

We use the further decomposition

$$\begin{aligned} \frac{x_i + b}{Z_i} &= \frac{\max(x_i + b, x_i x_i^> l_d)}{Z_i} \frac{\max(z_i + b, z_i z_i^> l_d)}{Z_i} \\ &+ \frac{\max(z_i + b, z_i z_i^> l_d)}{Z_i} \frac{E[\max(x_i + b, x_i x_i^> l_d)]}{Z_i} \\ &+ \frac{E[\max(x_i + b, x_i x_i^> l_d)]}{Z_i} \frac{E[\max(z_i + b, z_i z_i^> l_d)]}{Z_i}; \end{aligned}$$

As before, since $Z_i = x_i$ with probability greater than $1 - 2n^{-12}$, the term $\frac{x_i + b}{Z_i} = 0$ on this event.

Let us further decompose $\frac{x_i + b}{Z_i}$ as

$$\begin{aligned} \frac{x_i + b}{Z_i} &= \frac{\max(x_i + b, x_i x_i^> l_d) + \frac{1}{\log(nk)} \frac{x_i x_i^>}{Z_i}}{Z_i} \frac{E[\max(x_i + b, x_i x_i^> l_d) + \frac{1}{\log(nk)} \frac{x_i x_i^>}{Z_i}]}{Z_i} \\ &+ \frac{\frac{1}{\log(nk)} \frac{x_i x_i^>}{Z_i}}{Z_i} l_d + l_d \frac{(E[\max(x_i + b, x_i x_i^> l_d)] \max(x_i + b))}{Z_i}; \end{aligned}$$

so that

$$\frac{2}{n} \sum_{i=1}^n \mathcal{J}_{\text{op}}^{(1)} \leq \frac{2}{n} \sum_{i=1}^n \mathcal{J}_{\text{op}}^{(2)} + \frac{2}{n} \sum_{i=1}^n \mathcal{J}_{\text{op}}^{(3)} + \frac{2}{n} \sum_{i=1}^n \mathcal{J}_{\text{op}}^{(3)};$$

Since $\max_i (z_i + b) \leq C\sqrt{\log(nk)}$, the random vector $\max_i (z_i + b) + C\sqrt{\log(nk)}z_i$ is well-defined and bounded; sub-Gaussian concentration bounds [13] can therefore be applied to obtain

$$\mathbb{P} \left[\sum_{i=1}^n \frac{1}{n} \mathbb{X}_i^{(1)} \right]_{\text{op}} \leq c_1 \sqrt{\log(nk)}^2 \left(\frac{d}{n} + \frac{d}{n} + \dots \right) + c_2 \exp(-n \min(\dots)^2)$$

where $\sqrt{\log(nk)} = \max_i (z_i + b) + \sqrt{\log(nk)} \leq 2\sqrt{\log(nk)}$. Reasoning similarly for the second term, we have

$$\mathbb{P} \left[\sum_{i=1}^n \frac{1}{n} \mathbb{X}_i^{(2)} \right]_{\text{op}} \leq c_1 \sqrt{\log(nk)}^2 \left(\frac{d}{n} + \frac{d}{n} + \dots \right) + c_2 \exp(-n \min(\dots)^2) :$$

Combining these bounds setting $\dots = c_1 \frac{d}{n}$, we have

$$\sum_{i=1}^n \frac{2}{n} \mathbb{X}_i^{(1)} + \sum_{i=1}^n \frac{2}{n} \mathbb{X}_i^{(2)} \leq C\sqrt{\log(nk)}^2 \left(\frac{d}{n} + \frac{d}{n} \right)$$

with probability at least $1 - \exp(-c'd)$.

The term $\mathbb{X}_i^{(3)}$, on the other hand, can be controlled directly via Hoeffding's inequality. Since $\max_i (z_i + b)$ is $C\sqrt{\log(nk)}$ sub-Gaussian, we obtain

$$\mathbb{P} \left[\sum_{i=1}^n \frac{2}{n} \mathbb{X}_i^{(3)} \right] \leq \sqrt{\log(nk)} t^5 \exp\left(-\frac{nt^2}{32}\right) :$$

Choosing $t = c\sqrt{\frac{d+\log n}{n}}$ and putting together all the pieces, we obtain

$$\sum_{i=1}^n \frac{2}{n} \mathbb{X}_i \leq C\sqrt{\log(nk)}^2 \left(\frac{d+\log n}{n} + \frac{d+\log n}{n} \right) + c\sqrt{\log(nk)} \frac{d}{n}$$

with probability at least $1 - cn^{-12}$.

It remains to handle the terms $\sum_{i=1}^{n-2} g_i$, and to do so, we use a similar argument to before. We first bound the absolute value of the $(p; q)$ th entry of each matrix as

$$\begin{aligned} |g_i(p; q)| &= \mathbb{E}[\max_i (z_i + b) z_i z_i^{\geq}(p; q)] - \mathbb{E}[\max_i (x_i + b) x_i x_i^{\geq}(p; q) \mathbf{1}_{FE_i g}] \\ &\quad + \mathbb{E}[\max_i (x_i + b) x_i x_i^{\geq}(p; q) \mathbf{1}_{FE_i^c g}] = \mathbb{E}[\max_i (x_i + b) x_i x_i^{\geq}(p; q) \mathbf{1}_{FE_i^c g}] \end{aligned}$$

Expanding this further, we have

$$\begin{aligned} |g_i(p; q)| &\leq \mathbb{E}[\max_{j \in [2^k]} (j h_{\cdot, j} + j b_{\cdot, j}) x_{i,p} x_{i,q} \mathbf{1}_{FE_i^c g}] \\ &\leq \mathbb{E}[\sum_{j \in [2^k]} |j x_{i,p} x_{i,q}| k x_{i,k} \mathbf{1}_{FE_i^c g}] \leq 3 \\ &\leq 4 \sum_{j \in [2^k]} |j x_{i,p} x_{i,q}| \sum_{j \in [2^k]} |j x_{i,j}| \mathbf{1}_{FE_i^c g}^5 : \end{aligned}$$

Also note that $\mathbb{E} [j X^p] = 0$ unless $p \geq [k]; q \geq [k]$. Hence we finally need to control the terms of the form $\mathbb{E} [j X^p j^q]$ for a standard Gaussian X . Substituting $\mathbb{E} [j X^p j^q] = 5^{\frac{p+q}{2}} \sqrt{\log(nk)}$, a simple calculation of truncated third moment of standard Gaussian ([218]) yields

$$j^i (p; q) j^i \leq \log^2(nk) (nk)^{-10};$$

and proceeding as before provides a strictly lower order bound on $\sum_{i=1}^d \mathbb{E} [j^i]_{\text{op}}$ than the remaining terms.

The term $\sum_{i=1}^d \mathbb{E} [j^i]_{\text{op}}$ can be bounded more easily. Specifically, on the intersection of the events $\mathcal{F} \cap \mathcal{G}_{i=1}^{n=2}$, applying [13, Lemma 6.2], we have

$$\sum_{i=1}^d \frac{2}{n} \mathbb{E} [j^i]_{\text{op}}^2 \leq C^2 \log n \sum_{i=1}^d \frac{2}{n} \mathbb{E} [X_i^2] \leq \mathbb{E} [j^2]_{\text{op}} \leq C^2 \log n \left(\frac{d + \log n}{n} + \frac{(d + \log n)^2}{n^2} \right)$$

where the final bound holds with probability greater than $1 - cn^{-12}$. Finally combining all the terms yield the desired result. \square

6.9.11 Background and technical lemmas used in the proof of Theorem 14

In this section, we collect two technical lemmas that were used to prove Theorem 14.

6.9.12 Prediction and estimation error

Here, we connect the prediction error to the estimation error, which may be of independent interest. Recall our notation dist for the minimum distance between parameters obtainable after relabeling.

Lemma 43. *There exists a tuple of universal constants $(c_1; c_2)$ such that for each set of parameters $\theta_1; \dots; \theta_k \in \mathbb{R}^{d+1}$:*

1. *If $n \geq c_1 d$, then we have*

$$\frac{1}{n} \sum_{i=1}^n \max_{j \in [k]} h_{i; j} \leq \max_{j \in [k]} \mathbb{E} [h_{i; j}]^2 \leq c_1 \text{dist}(\mathcal{F}_{j \in [k]}^k; \mathcal{F}_{j=1}^k)$$

with probability exceeding $1 - c_1 \exp(-c_2 n)$.

2. *If $n \geq c_1 d \frac{k^3}{\min}$ $\log^2(k = \min)$, then we have*

$$c_2 \frac{\min}{k} \sum_{j \in [k]} \min_{j' \in [k]} k_j \leq \mathbb{E} [j^0 k^2] \leq \frac{1}{n} \sum_{i=1}^n \max_{j \in [k]} h_{i; j} \leq \max_{j \in [k]} \mathbb{E} [h_{i; j}]^2$$

with probability exceeding $1 - c_1 k \exp(-c_2 n \frac{4}{k^4 \log^2(k = \min)})$.

Proof. To prove the part 1 of the lemma, we leverage the fact that the max function is 1-Lipschitz with respect to the ℓ_2 -norm. Consequently, we obtain

$$\frac{1}{n} \sum_{i=1}^n \max_{j \in [k]} |h_{ij} - j^i| \leq \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^k |h_{ij} - j^i| \leq \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^k (|h_{ij} - j^i|)^2;$$

where we have ordered the parameters such that $\text{dist}(f_{j \in [k]}^k; j_{j=1}^k)$ is minimized. We now use the fact that the rows of \mathbf{H} are 1-sub-Gaussian (this is restatement of the conclusion of Lemma 40) to complete the proof.

We now proceed to a proof of part 2 of the lemma. Recall the setup of Appendix 6.7 along with notation $(f_{X_i}^n; g_{i=1}^n; b; \cdot; \cdot)$. Specifically, we have $j = (j; b_j)$ and $(\cdot)^> = [1 \ 2 \ \dots \ k]$. Similarly let $j = (j; b_j) \in \mathbb{R}^{d+1}$ and $\cdot^> = [1 \ 2 \ \dots \ k]$. In the notation of Section 6.6, we define for each pair $(\cdot; b)$, the sets

$$S_j(\cdot; b) = \{i \in [n] : h_{X_i} \cdot j + b_j = \max_{j^0 \in [k]} (h_{X_i} \cdot j^0 + b_{j^0}) \quad ; \quad j \in [k];$$

We use the shorthand $S_j = S_j(\cdot; b)$ and $\mathfrak{S}_j = S_j(\cdot; b)$ for the rest of the proof. By definition, we have

$$\begin{aligned} & \frac{1}{n} \sum_{i=1}^n (\max_{j \in [k]} |h_{X_i} \cdot j + b_j| - \max_{j^0 \in [k]} |h_{X_i} \cdot j^0 + b_{j^0}|)^2 \\ &= \frac{1}{n} \sum_{\substack{j \in [k] \\ m \in [k] \\ i \in S_j \setminus \mathfrak{S}_m}} (h_{X_i} \cdot j + b_j - (h_{X_i} \cdot m + b_m))^2 \\ &= \frac{1}{n} \sum_{\substack{j \in [k] \\ m \in [k] \\ i \in S_j \setminus \mathfrak{S}_m}} (h_{X_i} \cdot j - h_{X_i} \cdot m)^2 \\ &= \frac{1}{n} \sum_{\substack{j \in [k] \\ m \in [k]}} \mathbf{e}_{j,m}(\cdot, \cdot) k^2; \end{aligned}$$

where we have let $\mathbf{e}_{j,m}$ denote the sub-matrix of \mathbf{H} with rows indexed by the set $S_j \setminus \mathfrak{S}_m$. It is also useful to define the convex sets

$$K_{\cdot} := \{x \in \mathbb{R}^d : h_{X_i} \cdot x + b_{\cdot} = \max_{j^0 \in [k]} (h_{X_i} \cdot j^0 + b_{j^0}) \quad ; \quad \text{and}$$

$$K_m := \{x : h_{X_i} \cdot m + b_m = \max_{j^0 \in [k]} (h_{X_i} \cdot j^0 + b_{j^0})$$

for each pair $(\cdot; m) \geq k = \min$. By definition, for each $\cdot \geq [k]$, there exists a corresponding index m such that $\text{vol}(K_{\cdot} \setminus K_{m_{\cdot}}) \leq \frac{\min}{k}$. Proceeding from above, we have

$$\frac{1}{n} \sum_{i=1}^n (\max(x_i + b) - \max(x_i - b))^2 \leq \frac{1}{n} \sum_{\cdot \geq [k]} e_{\cdot; m_{\cdot}} (\cdot - m_{\cdot})^2$$

$$\leq \frac{1}{n} \sum_{\cdot \geq [k]} \min_{m} e_{\cdot; m} e_{\cdot; m} \leq k \cdot \min_{\cdot \geq [k]} m \cdot k^2$$

Finally, applying Lemma 31 in conjunction with the bound $\text{vol}(K_{\cdot} \setminus K_{m_{\cdot}}) \leq \frac{\min}{k}$, we obtain that provided $n \geq c_1 d \frac{k^3}{\min} \log^2(k = \min)$, we have

$$\min_{\cdot \geq [k]} e_{\cdot; m_{\cdot}} e_{\cdot; m_{\cdot}} \leq \frac{\min}{k} n^3$$

with probability exceeding $1 - c_1 \exp(-c_2 n \frac{\min^4}{k^4 \log^2(k = \min)})$ for each index $\cdot \geq [k]$. Taking a union bound over the k indices and combining the pieces completes the proof. \square

6.9.13 Projection onto a finite collection of rays

Consider a vector $y \in \mathbb{R}^n$ observed via the observation model

$$y = \beta + \epsilon;$$

where ϵ has independent, zero-mean, σ -sub-Gaussian entries. For a fixed set of M vectors $f_1, \dots, f_M \in \mathbb{R}^n$, denote by $\mathcal{C} := \{c \in \mathbb{R}^n : c = \alpha f_i, \alpha \geq 0, i \in [M]\}$ the set of all one-sided rays obtainable with these vectors.

Now consider the projection estimate

$$P_{\mathcal{C}}(y) = \operatorname{argmin}_{c \in \mathcal{C}} \|y - c\|_2^2;$$

which exists since the projection onto each ray exists. The following lemma proves an oracle inequality on the error of such an estimate.

Lemma 44. *There are universal constants c, C, c_1 and c_2 such that*

$$\Pr(\|y - P_{\mathcal{C}}(y)\|_2^2 \geq c \min_{c \in \mathcal{C}} \|y - c\|_2^2 + c_2 t(\log M + c_1)) \leq c_2 e^{-nt(\log M + c_1)};$$

for all $t \geq C(\log M + c_1)$.

Proof. We follow the standard technique for bounding the error for non-parametric least squares estimators. From the definition, we have

$$P_{\mathcal{C}}(y) = \operatorname{argmin}_{c \in \mathcal{C}} \|y - c\|_2^2;$$

We substitute the expression for y and obtain

$$P_C(y) = \operatorname{argmax}_{2C} 2h; \quad i \quad k \quad k^2 :$$

To obtain an upper bound on $kP_C(y)$ k^2 , it is sufficient to control the following quantity (e.g. see [219, Chapter 3], [13, Chapter 13]):

$$E \sup_{2C:k} h; \quad i \quad \#$$

for some $\epsilon > 0$ to be chosen later. Since ϵ is ϵ -sub-Gaussian, we use Dudley's entropy integral to control the term above. We obtain

$$E \sup_{2C:k} h; \quad i \quad \# \leq C \int_0^\infty \sqrt{\log N(\epsilon; f_{2C;k} \quad k \quad g; \epsilon_2)} d\epsilon;$$

where $N(\epsilon; S; \epsilon_2)$ is the ϵ -covering number of a compact set S in ϵ_2 norm. Note that C contains scaled versions of M fixed vectors f_1, \dots, f_M . For a fixed i , with $i \in [M]$, the covering number $N(\epsilon; f_{2C;k} \quad k \quad g; \epsilon_2)$ is equivalent to the covering number of a bounded interval (in 1 dimension). Using [220], this is $(1 + \frac{2}{\epsilon})$. Since there are M such fixed vectors, we obtain

$$N(\epsilon; f_{2C;k} \quad k \quad g; \epsilon_2) \leq C_1 M (1 + \frac{2}{\epsilon});$$

Substituting, we obtain

$$E \sup_{2C:k} h; \quad i \quad \# \leq C \sqrt{\log M + C_1} :$$

Now, the critical inequality ([13, Chapter 13]) takes the form

$$(\sqrt{\log M + C_1}) \cdot \epsilon^2;$$

Hence we can choose $\epsilon = C_2 \sqrt{\log M + C_1}$. Now, for any t , invoking [13, Theorem 13.2] yields the oracle inequality

$$\begin{aligned} kP_C(y) &\leq k^2 + c k P_C(\cdot) k^2 + \epsilon^2 t (\log M + c_1) \\ &= c \min_{2C} k^2 + \epsilon^2 t (\log M + c_1) ; \end{aligned}$$

with probability exceeding $1 - c_2 e^{-nt(\log M + c_1)}$, which proves the lemma. □

6.9.14 NP-hardness of real phase retrieval

Our discussion borrows from a similar proof established in [103, Proposition 1] for mixtures of linear regressions. Recall that with n i.i.d observations $f(x_i; y_i)g_{i=1}^n$, the max-affine model takes the form

$$y_i = \max_{1 \leq j \leq k} (hx_i; j^i + b_j) + \epsilon_i;$$

where $f_i g_{i=1}^n$ is a sequence of i.i.d zero mean sub-Gaussian noise.

We now consider a special case, where $k = 2$, $b_1 = b_2 = 0$, and $\epsilon_1 = \epsilon_2$, corresponding to the real phase retrieval problem. Furthermore, we consider the noiseless case $\epsilon = 0$. Our covariate matrix is given by $X \in \mathbb{R}^{n \times d}$ and the response vector by $y \in \mathbb{R}^n$. We now show that even in this special case, there is family of instances $(X; y)$ such that solving the least squares problem (6.5) is NP-hard. In particular, we say that a “solution” to the noiseless phase retrieval problem exists on an instance $(X; y)$ if the least squares objective in equation (6.5) has minimum value zero.

Proposition 17. *Deciding whether a problem instance $(X; y)$ has a solution to the noiseless phase retrieval problem is NP-hard.*

Proof. The proof follows from a reduction to the subset-sum problem, the decision version of which is stated as follows: given ρ numbers $a_1; \dots; a_\rho \in \mathbb{R}$, we must decide if there exists a partition $S \subseteq [\rho]$ such that

$$\sum_{i \in S} a_i = \sum_{j \in S^c} a_j;$$

For each ρ -dimensional vector a , we design a problem instance $(X; y)$ such that solving the noiseless (real) phase retrieval problem on $(X; y)$ implies deciding on the subset sum problem specified by a .

To accomplish this, take $n = 2\rho + 1$ and $d = \rho$, and define the instance

$$X = \begin{bmatrix} I_\rho & 0 \\ 0 & I_\rho \end{bmatrix} \quad \text{and} \quad y = \begin{bmatrix} a \\ 0 \end{bmatrix};$$

where I_ρ denotes the $\rho \times \rho$ identity matrix. By construction, finding a solution to the noiseless (real) phase retrieval problem on this instance corresponds to finding a subset $S \subseteq [2\rho + 1]$ and a pair of vectors $(x_1; x_2)$ with $x_1 = x_2$, such that $X_S x_1 = y_S$, and $X_{S^c} x_2 = y_{S^c}$. Here X_S and y_S are the sub-matrix and sub-vector of X and y respectively restricted to the set S . Note that in general, the set S cannot contain the index i and $\rho + i$, since they correspond to two mutually exclusive equations. From this observation, we have $x_1(i) = \sum_{j \in S} a_j g_j$, and $x_1(i) = x_2(i)$, where $x_1(i)$ and $x_2(i)$ denote the i -th coordinate of x_1 and x_2 , respectively.

As a consequence, if x_1 (and $x_2 = x_1$) satisfies the first 2ρ equations in this system, then the final equation demands that

$$\sum_{i \in S} x_1(i) = 0 = \sum_{j \in S^c} x_2(j);$$

By construction, note that this is accomplished if and only if

$$\prod_{i \in S} a_i = \prod_{j \in S^c} a_j;$$

and so a solution to the noiseless (real) phase retrieval problem on $(X; y)$ yields a solution to the subset-sum problem, as desired. \square

Chapter 7

Max-affine Regression Beyond Gaussian Design—Small Ball covariates

In this chapter, we show that the Alternating Minimization (AM) algorithm is significantly robust. In particular, it converges locally under a weaker, “small-ball” design assumption (accommodating, for instance, and bounded log-concave distribution), and even when the underlying parameters are chosen with knowledge of the realized covariates. Once again, the final rate obtained by the procedure is near-parametric and minimax optimal (up to a polylogarithmic factor) as a function of the dimension, sample size, and noise variance. As a by-product of our analysis, we obtain convergence guarantees on a classical algorithm for the (real) phase retrieval problem in the presence of noise, and under considerably weaker assumptions on the design distribution than was previously known.

7.1 Introduction

Similar to Chapter 6, we show that AM converges under significantly weaker statistical assumptions. In particular, we allow the distribution of the covariates to come from the larger class of sub-Gaussian distributions that satisfy a *small-ball* condition. In addition, we also consider the scenario of *universal* parameter estimation, meaning that our guarantees hold uniformly over all $\theta_1, \dots, \theta_k$. This allows the parameters to be chosen with knowledge of the realized covariates, a robust setting that is commonly studied in signal processing applications like phase retrieval [56]. In contrast, our prior work [221] only handled the case where the parameters are fixed.

More precisely, our covariate assumption relies on the following definition.

Definition 12. (*Small-ball*) A distribution P_X satisfies a $(\epsilon; c_s)$ -small-ball property if, for $X \sim P_X$ and each $\epsilon > 0$, we have

$$\sup_{u \in \mathbb{S}^{d-1}; w \in \mathbb{R}} \Pr \left(|hX; u| + w \right)^2 \leq c_s \epsilon^2 \quad (c_s) : \quad (7.1)$$

The small-ball properties of various classes of distributions have been studied extensively in the probability literature [57, 58], and many natural distributions possess this property provided they are not too “peaky”. For instance, a simple calculation yields that provided the density of $hX; ui$ is bounded by $\frac{1}{\bar{c}}$ for each $u \in S^{d-1}$, the distribution P_X satisfies the $(1=2; \bar{c})$ -small ball property. We now present our assumption on the covariate distribution; recall that $X \in \mathbb{R}^d$ is said to be γ -sub-Gaussian if

$$\sup_{u \in S^{d-1}} \mathbb{E}[\exp(hX; ui)] \leq \exp\left(\frac{\gamma^2}{2}\right) \text{ for each } u \in \mathbb{R}^d.$$

Assumption 20. *The distribution P_X is isotropic, γ -sub-Gaussian, and satisfies a $(\gamma; c_s)$ small-ball condition.*

Let us briefly state a few examples where Assumption 20 is satisfied with particular values of the tuple $(\gamma; c_s)$. The first is the class of compactly supported log-concave random vectors, which satisfy the small ball conditions with $(\gamma; c_s) = (1=2; C)$ for an absolute constant C (see [222, Appendix G.1]). Boundedness further implies sub-Gaussianity. As a specific example, consider X with each entry drawn i.i.d. according to the distribution $\text{Unif}[-\frac{1}{3}; \frac{1}{3}]$, which is commonly used as a random design in investigations of non-parametric regression problems [223]. The associated distribution P_X is isotropic by definition, and has $(\gamma; c_s) = (12; 1=2; C)$. Similarly, any other uniform distribution on a bounded, isotropic convex set would also satisfy Assumption 20. The second (canonical) example for which Assumption 20 is satisfied is the standard Gaussian distribution. As we verify in Appendix 7.6.1 with γ^2 tail bounds, the standard Gaussian satisfies $(\gamma; c_s) = (1; 1=2; e)$. Also, while we have only presented examples in which $\gamma = 1=2$, there are distributions that satisfy the small ball condition for other values of γ : for example, any random variable with density $f(x) \propto e^{-k|x|^c}$ for a positive constant c .

In order to make our guarantees on universal parameter estimation more clear, let us define a few geometric quantities induced by the max affine regression model (6.1). For $X \sim P_X$, let

$$g_j(\beta_1; \dots; \beta_k) := \Pr\{hX; j^0 i + b_j = \max_{j^0 \in [k]} (hX; j^0 i + b_{j^0}) g_j\}$$

and define

$$\min(\beta_1; \dots; \beta_k) := \min_{j^0 \in [k]} g_j(\beta_1; \dots; \beta_k)$$

For a fixed set of true parameters, the quantity $\min(\beta_1; \dots; \beta_k) n$ is the expected number of samples that are noisy linear combinations of one of these parameters. Thus, even if the underlying parameters are fixed, we can only hope to estimate these parameters when $\min(\beta_1; \dots; \beta_k)$ is sufficiently large.

The signal strength of the problem is the minimum separation

$$(\beta_1; \dots; \beta_k) = \min_{j^0: j \notin j^0} |j - j^0|^2$$

We also define a notion of condition number, given by

$$\kappa(\theta_1, \dots, \theta_k) = \max_{j \in [k]} \frac{\max_{j^0 \in \mathcal{J}_j} \frac{\theta_j}{\theta_{j^0}}}{\min_{j^0 \in \mathcal{J}_j} \frac{\theta_j}{\theta_{j^0}}},$$

We often use the shorthand

$$\kappa_{\min} = \kappa(\theta_1, \dots, \theta_k); \quad \kappa_{\max} = \kappa(\theta_1, \dots, \theta_k); \quad \text{and} \quad \kappa = \kappa(\theta_1, \dots, \theta_k);$$

when the true parameters $\theta_1, \dots, \theta_k$ are clear from context.

Recall that our goal was to prove a result that holds uniformly for all true parameters $f_j, g_{j=1}^k$. However, this is clearly impossible in a general sense, since we cannot hope to obtain consistent estimates if some parameters are never observed in the sample. A workaround is to hold certain geometric quantities fixed while sweeping over all possible allowable parameters $\theta_j, j = 1, \dots, k$. Accordingly, for each triple of positive scalars (δ, ϵ, η) , we define the set of “admissible” true parameters as

$$\mathcal{B}_{\text{vol}}(\delta, \epsilon, \eta) = \{ \theta_1, \dots, \theta_k : \kappa_{\min} \leq \kappa(\theta_1, \dots, \theta_k) \leq \kappa_{\max} \leq \kappa, \text{ and } \theta_j \geq \delta, \theta_j \leq \epsilon, \theta_j \geq \eta \}$$

We let the true parameters $\theta_1, \dots, \theta_k$ take values in the set $\mathcal{B}_{\text{vol}}(\delta, \epsilon, \eta)$, and prove guarantees uniformly over all such $\theta_1, \dots, \theta_k$.

With these definitions at hand, we are now ready to discuss our contributions.

Contributions: Suppose Assumption 20 holds, and $f_j^{(t)}, g_{j=1}^k$ are the parameter estimates returned by the AM algorithm at the t -th iteration. In Theorem 18, we show that for any $\epsilon > 0$, there is a sufficiently large iteration t such that

$$\sum_{i=1}^k \frac{\| \theta_i^{(t)} - \theta_i \|^2}{\theta_i^2} \leq C_{\delta, \epsilon, \eta}^{(1)} \frac{2kd}{n^{\frac{1+2}{\min}}} \log(kd) \log \frac{n}{kd}.$$

Such a result holds *simultaneously* for all $(\theta_1, \dots, \theta_k) \in \mathcal{B}_{\text{vol}}(\delta, \epsilon, \eta)$ with high probability provided the sample size is large enough and the initialization is chosen close enough to the true parameters. All of these aspects of the result are quantitative, and clarified in the statements of the formal theorems to follow.

In Corollary 3, we specialize Theorem 18 to the phase retrieval problem, showing that a slight variant of the same AM algorithm [95] exhibits linear convergence provided the covariates (called “measurements” in the signal processing literature) satisfy Assumption 20. Since our result holds for universal parameter estimation, we allow the underlying “signal” to be chosen adversarially, with knowledge of the measurements. Such a robust setting is common for phase retrieval problems. However, to the best of our knowledge, all previous results on the AM algorithm for phase retrieval [59, 60] only held under the assumptions of Gaussian covariates and noiseless observations, and/or required resampling of the measurements [61]. Ours is thus the first work to

handle non-Gaussian covariates in the presence of noise, while also analyzing the algorithm without resampling.

In Section 6.3, we validate the local convergence of AM with $\text{Unif}[\rho_{\bar{3}}; \rho_{\bar{3}}]$ d covariates. We observe that, when $\epsilon = 0$, AM recovers the true parameters $f_i, g_{i=1}^k$ within 4–5 iterations. Furthermore, when $\epsilon \neq 0$, the iterates of AM plateaus at the statistical error of the problem, which has a linear dependence on $d=n$. Moreover, in Section 7.2.4, we empirically show the performance of two initialization techniques; (a) random initialization with multiple restarts and (b) heuristic Principal Component Analysis (PCA) based initialization. For (a), we randomly initialize AM with multiple seeds, and observe that around 80–100 restarts is sufficient for a moderate dimensional problem. Furthermore, for (b), we use an appropriately defined second moment matrix (a function of $\{y_i; x_i, g_{i=1}^n\}$), and running PCA in conjunction with a random search method (Algorithm 3 of [221]), we obtain the initialization required for AM. Hence, the requirement of ‘good’ initialization is not a deterrent in the practical usage of AM.

7.2 Main results

Let us now state and discuss our results in precise terms.

7.2.1 Local geometric convergence of alternating minimization

For each pair $1 \leq i \neq j \leq k$ and $t \geq 0$, we use the shorthand $v_{ij} = \rho_i - \rho_j$ and $v_{ij}^{(t)} = \rho_i^{(t)} - \rho_j^{(t)}$ to denote the pairwise differences between parameters.

Theorem 18. *Suppose that Assumption 20 holds. Then there exists a pair of universal positive constants $(c_1; c_2)$ and positive constants $(C_{\rho; \rho_s}^{(1)}; C_{\rho; \rho_s}^{(2)})$ depending only on the triple $(\rho; \rho_s; c_s)$ such that if the sample size satisfies the bound*

$$n \geq C_{\rho; \rho_s}^{(1)} \max\{fd; 10 \log ng\} \max_{\min} \frac{k}{1+2^{-1}} \log(n=d); \quad (7.2)$$

then simultaneously for all true parameters $\rho_1; \dots; \rho_k \geq B_{\text{Vol}}(\min; \rho_s)$ and all initializations satisfying

$$\min_{c>0} \max_{j \neq j^0} \frac{c v_{j:j^0}^{(0)} - v_{j:j^0}}{k_j - j^0 k} \leq C_{\rho; \rho_s}^{(2)} \left(\frac{1+2^{-1}}{k}\right)^{-1} \log^{(1+2^{-1})} \left(\frac{k}{\min}\right); \quad (7.3a)$$

the estimation error at all iterations $t \geq 1$ satisfies

$$\sum_{j=1}^k \rho_j^{(t)} - \rho_j^2 \leq \frac{3}{4} \sum_{j=1}^k \rho_j^{(0)} - \rho_j^2 + C_{\rho; \rho_s}^{(1)} \frac{kd}{n} \frac{1+2^{-1}}{\min} \log(kd) \log(n=kd) \quad (7.3b)$$

with probability exceeding $1 - c_1 \frac{n}{k^2} + \exp(-c_2 n \frac{1}{\min})$. Here, $c (> 0)$ minimizes the LHS of inequality (7.3a).

Let us discuss the initialization conditions of the theorem in more detail. We require the initialization $\theta_1^{(0)}; \dots; \theta_k^{(0)}$ to satisfy condition (7.3a). In the well-balanced case (with $\min = 1=k$) and treating k as constant, the initialization condition (7.3a) posits that the parameters are a constant “distance” from the true parameters. Closeness here is measured in a relative sense, i.e., between pairwise differences of the parameter estimates as opposed to the parameters themselves. The intuition for this is that $\theta_1^{(0)}; \dots; \theta_k^{(0)}$ induces a partition of samples $S_1(\theta_1^{(0)}; \dots; \theta_k^{(0)}); \dots; S_1(\theta_1^{(0)}; \dots; \theta_k^{(0)})$, and the closeness to this to the true partition depends only on the relative pairwise differences between parameters. Furthermore, the initialization condition is also invariant to a global scaling of the parameters, since scaling does not change the initial partition of samples. Also note that the geometric convergence guarantee (7.3b) holds uniformly for all initializations satisfying condition (7.3a). Hence, the initialization parameters are not additionally required to be independent of the covariates or noise.

Let us now turn our attention to the bound (7.3b), which consists of two terms. When $t \rightarrow 1$, the second term of the bound (7.3b) provides an estimate of the closeness of the final parameters to the true parameters. Up to a constant, this is the statistical error term

$$sb_{n;}(d; k; \min) = 2 \frac{kd}{\frac{1+2}{\min} - 1} \frac{1}{n} \log(kd) \log(n=kd) \tag{7.4}$$

that converges to 0 as $n \rightarrow 1$, thereby providing a consistent estimate in the large sample limit. The dependence on \min is discussed shortly.

The first term of (7.3b) is an optimization error that is best interpreted in the noiseless case $\epsilon = 0$, wherein the parameters $\theta_1^{(t)}; \dots; \theta_k^{(t)}$ converge at a geometric rate to the true parameters $\theta_1; \dots; \theta_k$. In the noiseless case, we obtain exact recovery of the parameters provided

$$n \geq C \frac{kd}{\frac{1+2}{\min} - 1} \log(n=d):$$

Thus, the “sample complexity” of parameter recovery is linear in the dimension d , which is optimal. In the well-balanced case, we require $n \geq k^{2+2} \frac{1}{\min} d$, but lower bounds based on parameter counting suggest that the true dependence ought to be linear. We are not aware of whether the dependence on \min in the noiseless case is optimal; our simulations in panel (a) suggest that the sample complexity depends inversely on \min , and so closing this gap is an interesting open problem.

In Figure 7.1, we verify that for independent, isotropic covariates chosen uniformly from a symmetric interval¹, initializing the AM algorithm in a neighborhood of the true parameters suffices to ensure that it converges to the true parameters. Furthermore, both the sample size requirement and final error of the algorithm exhibit the behaviors predicted by Theorem 18.

We now compare Theorem 18 with Chapter 6 (Theorem 1) for the special case of Gaussian covariates, where $\rho = 1$ and $\sigma = 1=2$. In this case, all terms of the form $\frac{3}{\min}$ in [221, Theorem 1]

¹Such a distribution is compactly supported and log-concave, and therefore satisfies Assumption 20.

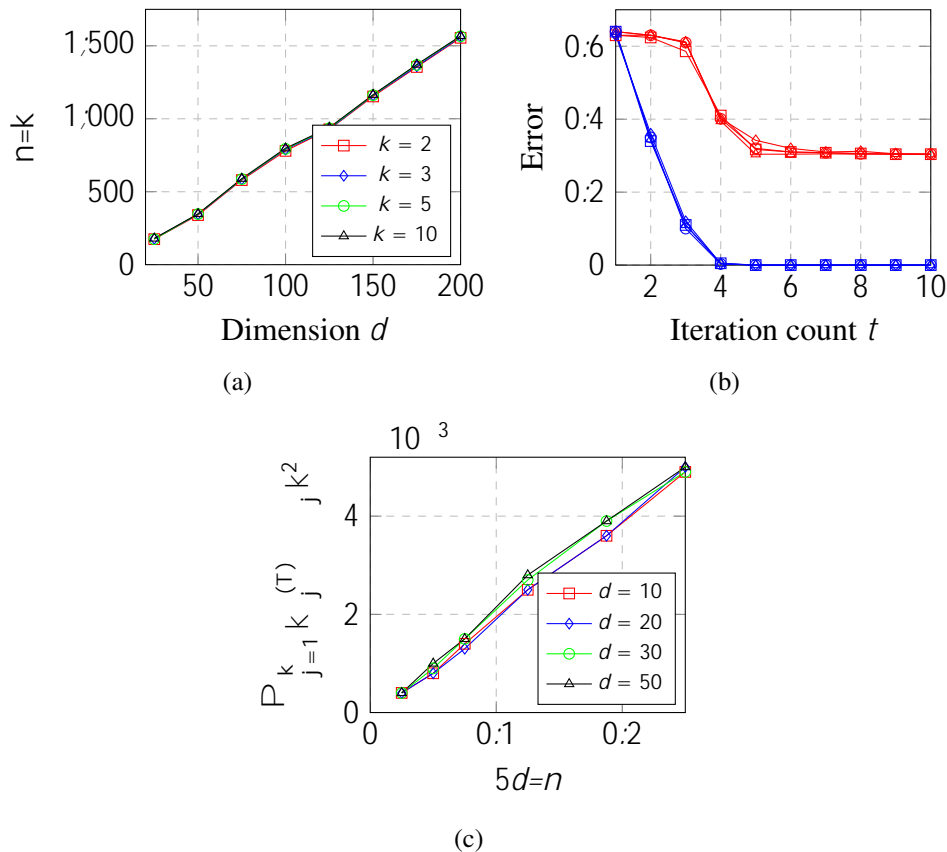


Figure 7.1: Convergence of AM when the covariates are drawn i.i.d. from the distribution $\text{Unif}[\frac{1}{3}, \frac{2}{3}]^d$. In panel (a), we plot the noiseless sample complexity of AM; we fix $k_i, k = 1$ for all $i \geq [k]$, $\beta_i = 0$ and $\beta_{\min} = 1 = k$. We say β_i is recovered if $|\beta_i^{(t)} - \beta_i| \leq 0.01$. For a fixed dimension d , we run a linear search on the number of samples n , such that the empirical probability of success over 100 trials is more than 0.95, and output the least such n . In panel (b), we plot the optimization error (in blue) $\sum_{j=1}^k \|\beta_j^{(t)} - \beta_j\|^2$ and the deviation from the true parameters (in red) $\sum_{j=1}^k \|\beta_j^{(t)} - \beta_j\|^2$ over iterations t for various values of β in the tuple $(0.15; 0.25; 0.4; 0.5)$, with $k = 5$, $d = 100$, $T = 50$ and $n = 5d$. The resulting error curves are averaged over 50 trials. Panel (c) shows that the estimation error at $T = 50$ scales at the parametric rate $d=n$, where we have chosen a fixed $k = 5$ and $\beta = 0.25$.

are replaced by terms of the form $\frac{5}{\min}$. In particular, we see that the initialization condition (7.3a) is more stringent and the final statistical rate of the estimate (corresponding to the limit $t \rightarrow \infty$) now attains an estimation error that is a factor $\frac{2}{\min}$ higher than the corresponding rate of Chapter 6 (Theorem 1). The sample size requirement is similarly affected. On the other hand, the geometric convergence result (7.3b) now holds uniformly for all true parameters $\beta_1, \dots, \beta_k \in \mathcal{B}_{\text{vol}}(\frac{1}{\min}, \frac{1}{\max})$, as opposed to Chapter 6 (Theorem 1), which holds only when the true parameters are held fixed. The more stringent initialization condition and sample size requirements can be viewed as the price

Algorithm 10: Alternating minimization for real phase retrieval

Input: Data $\{x_i; y_i\}_{i=1}^n$; initial parameter estimate $b^{(0)} \in \mathbb{R}^d$; number of iterations T .

Output: Final estimator \hat{b} .

1 Initialize $t = 0$.

2 **repeat**

5 Compute sign vector $s^{(t)}$ with i -th entry as

$$s_i^{(t)} = \text{sgn}(hx_i; s^{(t)}) \quad \text{for each } i \in [n]: \tag{7.5a}$$

6 Update

$$b^{(t+1)} = \arg \min_{\mathbb{R}^d} \sum_{i=1}^n (y_i - s_i^{(t)} hx_i; i)^2: \tag{7.5b}$$

7 **until** $t = T$;

8 **Return** $\hat{b} = b^{(T)}$.

to pay for the more robust convergence of the AM algorithm. Notably, the dependence on $k; n; d$ remains unchanged.

7.2.2 Consequences for phase retrieval

A notable consequence of Theorem 18 is that it can be applied to the phase retrieval model—in which results are usually proved uniformly over all true parameters [56, 224]—to yield a convergence result under general distributional assumptions on the covariates. In particular, setting $\mu_{\min} = 1/2$ and $k = 2$ yields a local linear convergence result for the AM algorithm of the Gershberg-Saxton-Fienup type (presented for completeness as Algorithm 10) uniformly *for all* provided the covariates satisfy a small-ball condition. We note that other algorithms for phase retrieval have also been shown to succeed under such small-ball assumptions [225, 226].

Corollary 3. *Suppose that Assumption 20 holds, and that $b^{(t)}$ is the t -th iterate of Algorithm 10. There exists a universal positive constant c_1 and a pair of constants $(C_{\mu; c_s}^{(1)}; C_{\mu; c_s}^{(2)})$ depending only on $(\mu; c_s)$ such that if*

$$n \geq C_{\mu; c_s}^{(1)} \max\{d; 10 \log n\} \log(n=d);$$

then simultaneously for all true parameters $\theta \in \mathbb{R}^d$ and all initializations $b^{(0)}$ satisfying

$$\min_{c>0} \min_{s \in \mathbb{S}^{d-1}} \frac{c}{k} \frac{s}{k} \leq C_{\mu; c_s}^{(2)}; \tag{7.6a}$$

the estimation error for all $t \geq 1$ satisfies

$$\min_{s \in \mathcal{S}} \|k^{(t)} - s\|_2^2 \leq \frac{3}{4} \min_{s \in \mathcal{S}} \|k^{(0)} - s\|_2^2 + \frac{c_1 \sigma^2 d \log(n=d)}{n}$$

with probability exceeding $1 - c_1 n^{-\gamma}$.

Let us now compare this with the sharpest existing local convergence result of AM for phase retrieval due to Waldspurger [59], which holds for Gaussian covariates and in the noiseless setting². Specializing Corollary 3 to the noiseless setting, we observe that provided the ratio $n=d$ is larger than a fixed constant (that depends only on $(\sigma; C_S)$), we obtain exact recovery of the underlying parameter with high probability, up to a global sign, provided the measurement vectors are sub-Gaussian and satisfy a small-ball condition. To the best of our knowledge, prior work on the AM algorithm had not established provable guarantees for non-Gaussian covariates even in the noiseless setting. In the noisy case, Corollary 3 guarantees convergence of the iterates to a small neighborhood around either s or $-s$, and the size of this neighborhood is within a logarithmic factor of being minimax optimal [56, 184]. Once again, to the best of our knowledge, guarantees for the AM algorithm as applied to noisy phase retrieval did not exist in the literature.

7.2.3 Proof ideas and technical challenges

We now sketch the high level proof ideas required to establish guarantees on the AM algorithm. Some parts of the proof here is similar to that of Chapter 6. For example, Similar to Chapter 6, here also, the proof consists of 3 steps: (a) controlling the behavior of noise, (b) analyzing the prediction error of the noiseless problem and (c) translating the prediction error to estimation error by inverting a specifically chosen sub-matrix of the covariate matrix. The analysis of (a) parallels to that of Chapter 6. However, for part (b) and (c), the weakened assumption on covariates poses non-trivial technical difficulties.

In order to control the prediction error of the noiseless problem, we crucially use the anti concentration property of the small-ball distribution. Using the good initialization condition in conjunction with the above mentioned lemmas, we upper-bound the noiseless prediction error of the AM iterates. Finally, in order to translate our bounds on the prediction error into bounds on parameter estimation, we invert specifically chosen sub-matrices of the covariate matrix over the course of the algorithm. Our bounds naturally depend on how these sub-matrices are conditioned. A key technical difficulty of the proof is to control the spectrum of these random matrices, rows of which are drawn from (randomly) truncated variants of the covariate distribution. Our techniques for controlling the spectrum of these matrices is more broadly applicable, and we expect this result to be of broader interest.

The technical meat of the paper crucially depends on the two technical lemmas we obtain that bound the maximum and minimum singular values of a matrix whose rows are drawn from a

²Waldspurger [59] deals with the complex phase retrieval, whose analysis is significantly more complicated than real phase retrieval considered here.

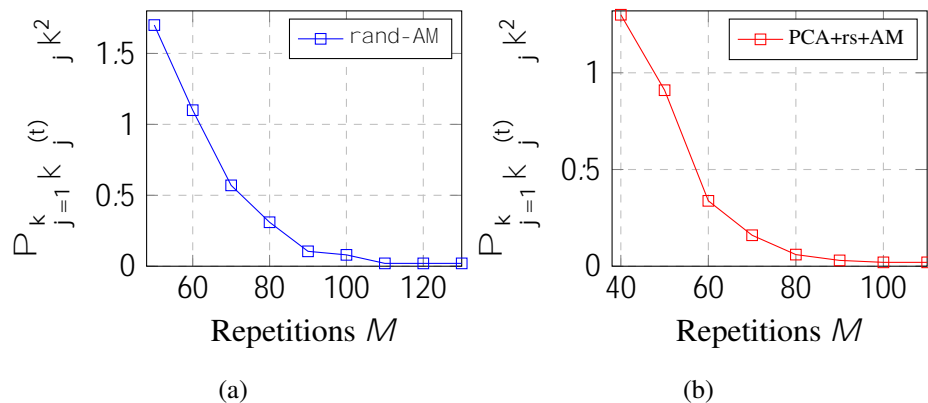


Figure 7.2: Initialization and overall guarantee with bounded uniform covariates drawn according to the distribution $\text{Unif}[\frac{p}{3}, \frac{p}{3}]^d$. In panel (a) we fix $k = 3$, $d = 10$, $n = 30kd$, $\sigma = 0.1$ and $\rho_{\min} = 1/3$. We evaluate the performance of AM with repeated random initialization [154] and plot the estimation error with M , the number of restarts. In panel (b), we run Algorithm 2 from our companion paper [221] in order to initialize the AM algorithm. We fix $k = 3$, $d = 50$, $n = 30kd$, $\sigma = 0.1$ and $\rho_{\min} = 1/3$. All the data points are obtained as a result of averaging 25 trials of the experiment.

sub-Gaussian distribution obeying the small-ball property. Note that these results hold uniformly for any matrix that has sufficiently many rows. Our results on the minimum singular value are similar in spirit to those of Rudelson and Vershynin [58], but proved under a slightly different set of assumptions. We apply these results on carefully chosen sub-matrices of the covariate matrix \mathbf{A} .

7.2.4 Initialization and overall guarantee

We conclude our discussion of the main results with some simulations demonstrating that the ‘good’ initialization condition required in Theorem 18 for the convergence of the AM algorithm is indeed achievable in practice. In particular, we consider two approaches. We first evaluate the random initialization scheme with multiple restarts proposed by Balasz [154]. We also simulate the PCA-based approach proposed in Chapter 6 for Gaussian covariates, wherein we use a moment method to convert the regression problem in d -dimensions into one in $k + 1$ dimensions, followed by a random initialization scheme. This method is particularly useful in the high dimensional regime $d \gg k$, and is described formally in Chapter 6 (Algorithms 2 and 3).

In Figure 7.2 (a), we initialize AM randomly, with multiple restarts, following [154]. We observe that with sufficiently many restarts, AM converges. Hence, the requirement of ‘good’ initialization can be replaced by random initialization with multiple restarts. This is how AM is used in practice as seen in [154]. However, we observe that with increasing dimension, the number of restarts M increases rapidly, and hence we get hit by the ‘curse of dimensionality’. For this reason, we consider a low dimensional setting with $d = 10$ and observe that around 110 restarts suffices. Understanding this repeated random initialization algorithm and obtaining explicit dependence on the number of restarts (as a function of dimension d , and the number of affine pieces k) is an interesting open

problem.

We attempt to resolve the curse of dimensionality via a heuristic PCA algorithm. Our algorithm matches exactly Algorithm 2 of Chapter 6, except the definition of \mathcal{M}_2 (the second moment used in the spectral method, see [221] for details). We define $\mathcal{M}_2 = \frac{2}{n} \sum_{i=1}^{n-2} (y_i^2=2) x_i x_i^\top / d$. After obtaining the subspace spanned the true parameters, we run the random search method (Algorithm 3 of Chapter 6) to obtain the initialization, and finally run AM with it. We consider a moderate dimensional problem ($d = 50$), and show (in Figure 7.2(b)) that 100 restarts are sufficient. Note that as expected, this algorithm can handle a high dimensional problem; thanks to the PCA algorithm. This implies that, with a slight variation in the definition of \mathcal{M}_2 , the PCA algorithm is indeed robust with respect to the covariate distribution: it works (at least empirically) even when the covariates are sub-Gaussian satisfying a small-ball condition. Obtaining provable guarantees for PCA with non-Gaussian covariates is kept as an interesting future work.

7.3 Discussion and Open Problems

We analyze a natural alternating minimization algorithm for estimating the maximum of unknown affine functions, when the covariates come from sub-Gaussian distribution with small-ball condition. Additionally, we also consider the setting of universal parameter estimation, meaning that our results hold uniformly for all true parameters. In particular, this also allows the parameters be chosen with the knowledge of the covariates. We establish that AM enjoys local linear convergence to a ball around the optimal parameters. We characterize the statistical error of AM, and empirically verify the dependence on problem parameters like d and n . As a corollary, specializing to the (real) phase retrieval setting, we obtain several new results; in particular we obtain provable guarantees for the phase retrieval problem with non-Gaussian covariates and in the presence of noise.

Moreover, we experimentally show two techniques for the initializing AM using random initialization (with restarts) and PCA. However, we do not have a provable initialization guarantee. An immediate future work would to obtain a provable initialization. An important question that remains to be addressed is whether the heuristic of random initialization with multiple restarts, that has been employed both by us and in prior work can be shown to provably converge with an explicit bound on the number of repetitions. Understanding the behavior of the randomly initialized AM algorithm is also an open problem in the context of phase retrieval [59, 201].

Also, it would be interesting to study this problem in very large dimension with a sparsity assumption on the true parameters. Furthermore, another interesting problem is to obtain a minimax lower bound on estimation error in order to understand the (actual) dependence on several problem parameters. We keep these as our future endeavors.

Appendix

We now present proofs of our main results. We assume throughout that the sample size n is larger than some universal constant. Also, we assume that the values of constants $C; C_1; C^d; \dots$ may change

from line to line.

7.4 Proof of Theorem 18

Throughout the proof, we make references to the proof of [221, Theorem 1]. Recall our observation model (6.1). We use the notation $\Sigma \in \mathbb{R}^{n \times (d+1)}$ for the covariate matrix, $y \in \mathbb{R}^n$ for the response vector. Recall that the noise is sub-Gaussian. Let us begin by introducing some shorthand notation. For a scalar w , vectors $u \in \mathbb{R}^d$ and $v = (u; w) \in \mathbb{R}^{d+1}$, and a positive scalar r , let $B_v(r) = \{u' \in \mathbb{R}^d : \frac{kv - v'k}{ku - k} \leq r\}$. Define the set

$$I(r; \Sigma) = \{1, \dots, k\} \times \{1, \dots, k\} : \exists c > 0 : c(\Sigma_{i,j}) \geq B_{i,j}(r) \text{ for all } 1 \leq i \neq j \leq k\}$$

Also, use

$$\#_t(r; \Sigma) := \sup_{\substack{(0) \\ 1, \dots, k}} \sum_{j=1}^k k_j^{(t)} \frac{3}{4} \sum_{j=1}^k k_j^{(0)}$$

to denote the error tracked over iterations (with c denoting the smallest $c > 0$ such that $c(\Sigma_{i,j}) \geq B_{i,j}(r)$ for all $1 \leq i \neq j \leq k$). Finally, we use the shorthand

$$\text{sb}_{n; (d; k; \min)} := \frac{2}{n} \frac{kd}{1+2^{-1}} \log(kd) \log(n=kd):$$

to denote the final statistical rate. With the rest of the notation remaining the same as before, the theorem claims that there exist constants such that if condition (7.2) is satisfied, then we have

$$\Pr \left(\max_{t=1}^{\infty} \sup_{\substack{(0) \\ 1, \dots, k}} \#_t @ C_{; \mathcal{C}_s}^{(2)} \frac{1+2^{-1}}{k} ; \Sigma_{j=1}^k A_{; \mathcal{C}_s}^{(1)} \text{sb}_{n; (d; k; \min)} \right) \leq 1 - c_1 \frac{k^2}{n^7} + \exp(-c_2 n^2_{\min}) \tag{7.7}$$

We dedicate the rest of the proof to establishing (7.7). Without loss of generality, we assume that the scalar c is equal to 1. It is convenient to state and prove another result that guarantees a one-step contraction, from which Theorem 18 follows as a corollary. In order to state this result, we assume that one step of the alternating minimization update is run starting from the parameters $\{f_j\}_{j=1}^k$ to produce the next iterate $\{f_j^+\}_{j=1}^k$. In the statement of the proposition, we use the shorthand

$$\begin{aligned} V_{i,j} &= \Sigma_{i,j} \\ V_{i,j} &= \Sigma_{i,j} \text{ and} \\ V_{i,j}^+ &= \Sigma_{i,j}^+ \end{aligned}$$

Also recall the definitions of the geometric quantities $(\Sigma_{i,j})$. The following proposition guarantees the one step contraction bound.

Proposition 19. *Suppose that Assumption 20 holds. Then there exists a tuple of universal constants $(c_1; c_2)$ and another tuple of constants $(C^{(1)}; ; ; c_s; C^{(2)}; ; ; c_s)$ depending only on the tuple $(; ; ; c_s)$ such that*

(a) *If the sample size satisfies the bound $n \geq c_1 \max_j f_j d; 10 \log ng \max_{j \in \{1, \dots, k\}} \frac{1 + \frac{1}{\min_j \lambda_j}}{2} \frac{k}{\min_j \lambda_j} \log(n=d)$, then for any set of parameters $\{f_j\}_{j=1}^k \geq B_{\text{vol}}(\min_j \lambda_j; ; ;)$ and $f_j g_{j=1}^k$ satisfying*

$$\max_{1 \leq j \in j^0 \leq k} \frac{V_{j:j^0}}{k_j} \frac{V_{j:j^0}}{j^0 k} \log^{1+} \frac{k_j}{V_{j:j^0}} \frac{j^0 k}{V_{j:j^0}} C^{(2)}; ; ; c_s \frac{1+2}{\min} \frac{1}{k}; \quad (7.8a)$$

we have, simultaneously for all pairs $1 \leq j \in j^0 \leq k$, the bound

$$\frac{V_{j:j^0}^+}{k_j} \frac{V_{j:j^0}^2}{j^0 k^2} \leq C^{(1)}; ; ; c_s \max \left(\frac{d}{\min} \frac{1}{n}; \frac{1}{4k} \right) \times \frac{V_{j:j^0}}{k_j} \frac{V_{j:j^0}^2}{j^0 k^2} + \frac{V_{j:j^0}}{k_j} \frac{V_{j:j^0}^2}{j^0 k^2} + C^{(1)}; ; ; c_s \frac{2}{\min} \frac{kd}{\frac{1+2}{\min} n} \log(n=d) \quad (7.8b)$$

with probability exceeding $1 - c_1 \frac{k^2}{n^2} + \exp(-c_2 n \frac{2}{\min})$.

(b) *If the sample size satisfies the bound*

$$n \geq c_1 \max_j \max_{j \in \{1, \dots, k\}} f_j d; 10 \log ng \max_{j \in \{1, \dots, k\}} \frac{1 + \frac{1}{\min_j \lambda_j}}{2} \frac{k}{\min_j \lambda_j} \log(n=d); C^{(1)}; ; ; c_s \frac{kd}{\frac{1+2}{\min} n};$$

then for any set of parameters $\{f_j\}_{j=1}^k \geq B_{\text{vol}}(\min_j \lambda_j; ; ;)$ and $f_j g_{j=1}^k$ satisfying

$$\max_{1 \leq j \in j^0 \leq k} \frac{V_{j:j^0}}{k_j} \frac{V_{j:j^0}}{j^0 k} \log^{1+} \frac{k_j}{V_{j:j^0}} \frac{j^0 k}{V_{j:j^0}} C^{(2)}; ; ; c_s \frac{1+2}{\min} \frac{1}{k}; \quad (7.9a)$$

we have the overall estimation error bound

$$\sum_{i=1}^k \frac{V_{j:j^0}^+}{k_j} \frac{V_{j:j^0}^2}{j^0 k^2} \leq \frac{3}{4} \sum_{i=1}^k \frac{V_{j:j^0}}{k_j} \frac{V_{j:j^0}^2}{j^0 k^2} + C^{(1)}; ; ; c_s \frac{2}{\min} \frac{kd}{\frac{1+2}{\min} n} \log(kd) \log(n=kd) \quad (7.9b)$$

with probability exceeding $1 - c_1 \frac{k^2}{n^2} + \exp(-c_2 n \frac{2}{\min})$.

Let us briefly comment on how Theorem 18 follows from Proposition 19. Note that the equations (7.9a) and (7.9b) in conjunction clearly show that geometric decay of the estimation error after running one step of the algorithm. We only need to verify that the next iterates $\{f_j\}_{j=1}^k$ also satisfy condition (7.8a) provided the sample size n is large enough. Hence, the one step estimation bound (7.9b) can be applied recursively to obtain the final bound (7.3b).

For the constant $C_{\gamma; c_S}^{(2)}$ in the proposition, let r_a be the largest scalar in the interval $[0; e^{-(1+\gamma)}]$ such that $r_a \log^{1+\gamma}(1/r_a) \leq C_{\gamma; c_S}^{(2)} \frac{1+\gamma}{k}$, and let r_b be the largest scalar in the interval $[0; e^{-(1+\gamma)}]$ with $r_b \log^{1+\gamma}(1/r_b) = C_{\gamma; c_S}^{(2)} \frac{1+\gamma}{k}$.

Assume that the current parameters satisfy the bound (7.8a). Choosing $n \geq 4 \frac{d}{\min_{j \in \mathcal{S}} d_j}^{1+\gamma}$ and applying inequality (7.8b), we have, for each pair $1 \leq j \in \mathcal{S} \subseteq \mathcal{K}$, the bound

$$\begin{aligned} \frac{V_{j^*}^+}{k_j} \frac{V_{j^*}^2}{k^2} &\leq C_{\gamma; c_S}^{(1)} \frac{1}{4k} \sum_{j^0=1}^k \frac{V_{j^*j^0}}{k_j} \frac{V_{j^*j^0}^2}{j^0 k^2} + \frac{V_{j^*j^0}}{k_j} \frac{V_{j^*j^0}^2}{j^0 k^2} \\ &+ C_{\gamma; c_S}^{(2)} \frac{2}{\min_{j \in \mathcal{S}} d_j} \frac{kd}{n} \log(n/d) \\ &\leq \frac{1}{2} r_a^2 + C_{\gamma; c_S}^{(2)} \frac{2}{\min_{j \in \mathcal{S}} d_j} \frac{kd}{n} \log(n/d) \end{aligned}$$

Further, if

$$n \geq C \frac{2}{r_a^2} \frac{k^{1+\gamma}}{\min_{j \in \mathcal{S}} d_j^{1+\gamma}} \log^{2+\gamma}(k) \frac{k}{\min_{j \in \mathcal{S}} d_j} \log(kd);$$

for a sufficiently large constant C , we have

$$\frac{V_{j^*}^+}{k_j} \frac{V_{j^*}^2}{k^2} \leq r_a^2.$$

Thus, the parameters $\{j^*_{j=1}^k\}$ satisfy inequality (7.8a) for the sample size choice required by Theorem 18. Finally, adjusting constants and simplifying the probability statement completes the proof of the theorem.

7.4.1 Proof of Proposition 19

We use the shorthand notation $S_j := S_j(\mathcal{S}; \mathcal{K})$; and let P_{S_j} denote the projection matrix onto the range of the matrix S_j . Recall our notation for the difference vectors.

Let y denote the vector with entry i given by $\max_{\mathcal{S} \subseteq \mathcal{K}} h_{i; \mathcal{S}}$. We have

$$\begin{aligned} k \sum_{j \in \mathcal{S}} (y_j^+ - y_j)^2 &= k P_{S_j} y_{S_j} - \sum_{j \in \mathcal{S}} y_j^2 k^2 \\ &= k P_{S_j} y_{S_j} + P_{S_j} y_{S_j} - \sum_{j \in \mathcal{S}} y_j^2 k^2 \\ &= 2k P_{S_j} (y_{S_j} - y_j)^2 + 2k P_{S_j} y_j^2 k^2 \\ &= 2k y_{S_j} - \sum_{j \in \mathcal{S}} y_j^2 k^2 + 2k P_{S_j} y_j^2 k^2; \end{aligned} \tag{7.10}$$

where we have used the fact that the projection operator is non-expansive on a convex set.

Let

$$f_{h_i; \cdot i} = \max_{u \in \mathbb{R}^k} g := \max_{u \in \mathbb{R}^k} h_i; \cdot i = \max_{u \in \mathbb{R}^k} h_i; \cdot i \quad ; \text{ for each } i \in [n]; \cdot \in [k]$$

denote a convenient shorthand for these events. The first term on the RHS of inequality (7.10) can be written as

$$\sum_{i \in S_j} \sum_{j^0: j^0 \in j} \mathbf{1}_{h_i; \cdot j^0 i = \max} \text{ and } h_i; \cdot j^0 i = \max_{j^0 \in j} h_i; \cdot j^0 i^2;$$

where the inequality accounts for ties. Each indicator random variable is bounded, in turn, as

$$\mathbf{1}_{h_i; \cdot j^0 i = \max} \text{ and } h_i; \cdot j^0 i = \max_{j^0 \in j} h_i; \cdot j^0 i \leq \mathbf{1}_{h_i; \cdot j^0 i \leq h_i; \cdot j^0 i} \text{ and } h_i; \cdot j^0 i \leq h_i; \cdot j^0 i \\ = \mathbf{1}_{h_i; \cdot j^0 i \leq h_i; \cdot j^0 i} \leq 1$$

Switching the order of summation yields the bound

$$\sum_{i \in S_j} \sum_{j^0: j^0 \in j} \mathbf{1}_{h_i; \cdot j^0 i \leq h_i; \cdot j^0 i} \text{ and } h_i; \cdot j^0 i \leq h_i; \cdot j^0 i \leq h_i; \cdot j^0 i^2$$

Recalling our notation for the minimum eigenvalue of a symmetric matrix, the LHS of inequality (7.10) can be bounded as

$$k \sum_j (\lambda_j^+)^2 \leq \min_j \lambda_j^+ \sum_j k \lambda_j^+ k^2$$

Hence, we arrive at the bound

$$\min_j \lambda_j^+ \sum_j k \lambda_j^+ k^2 \leq \sum_{j^0: j^0 \in j} \mathbf{1}_{h_i; \cdot j^0 i \leq h_i; \cdot j^0 i} \text{ and } h_i; \cdot j^0 i \leq h_i; \cdot j^0 i \leq h_i; \cdot j^0 i^2 + k \sum_j \lambda_j^+ k^2$$

For the second term (noise) on the RHS, we apply [221, Lemma 3]. We obtain

$$\sup_{1, \dots, k \in \mathbb{R}^{d+1}} \max_{j \in [k]} k \sum_j \lambda_j^+ k^2 \leq 2^2 k(d+1) \log(n=d)$$

with probability greater than $1 - \frac{n}{d}$.

We now make two claims to bound the remaining two terms in the bound. Recall that Assumption 20 holds, and use the shorthand $e := \max_{j \in [k]} \lambda_j^+$. Our first claim bounds the indicator quantities under this assumption. In particular, we claim that there exists a tuple of universal

constants $(C; c_1; C^0; c^0)$ such that for any $r \geq 1=24$, we have

$$\Pr \left[\exists \mathbf{g}_1, \dots, \mathbf{g}_k \in \mathbb{R}^{d+1} \text{ and } \mathbf{v}_1, \dots, \mathbf{v}_k \text{ such that } v_{j,j^0} \geq B_V(r) \text{ for all } 1 \leq j \leq k : \right. \\ \left. \times \prod_{i=1}^n \mathbf{1} \left[\sum_{j^0: j^0 \in j} h_{i; v_{j,j^0}} - h_{i; v_{j^0}} \leq 0 \right] \right] \\ \leq C e^2 \prod_{j^0: j^0 \in j} k v_{j,j^0} \prod_{j^0: j^0 \in j} v_{j,j^0} k^2 \max_{(1+\epsilon)^d; n} (1+\epsilon) (er) \log^{+1}(1+er) \\ \leq c_1 \frac{k}{2} n e^{C^0 n} + c e^{c^0 \max\{fd; 10 \log ng\}} \quad (7.11a)$$

The singular values of sub-matrices of \mathbf{A} are then handled by the following claim: that there exist universal constants $(C; c; c^0)$ and a constant $C^0; c_s$ that depends only the tuple $(\epsilon; \epsilon; c_s)$, such that if $n \geq C d \max\left\{\frac{1+\epsilon}{\min}; \frac{k}{\min}\right\} \log(n=kd)$, then

$$\Pr \left[\inf_{\mathbf{g}_1, \dots, \mathbf{g}_k \in \mathbb{B}_{\text{vol}}(\min; \epsilon)} \inf_{\substack{\mathbf{v}_1, \dots, \mathbf{v}_k \in \mathbb{R}^{d+1}; \\ v_{j,j^0} \geq 2B_{V_{j^0}}(r_B)}} \min_{j \in [k]} \lambda_j(\mathbf{A}_{\mathbf{g}_1, \dots, \mathbf{g}_k; \mathbf{v}_1, \dots, \mathbf{v}_k}) > \lambda_j(\mathbf{A}_{\mathbf{g}_1, \dots, \mathbf{g}_k}) \right] \\ \leq C^0; c_s \frac{1+2\epsilon}{\min} n e^{c^0 n} \exp(-c^0 n \frac{2}{\min}) \quad (7.11b)$$

The proof of the proposition follows directly from these claims. □
We now prove both the given claims in turn.

Proof of claim (7.11a): The proof of the claim hinges on the following lemma.

Lemma 45. *Suppose that Assumption 20 holds. Then, there exist universal constants $(C; c_1; c_2)$ such for all positive scalars $r \geq 1=24$, we have*

$$\sup_{\substack{\mathbf{v} \in \mathbb{R}^{d+1} \\ \mathbf{v} \in 2B_V(r)}} \frac{1}{n} \prod_{i=1}^n \mathbf{1} \left[\sum_{j^0} h_{i; v_{j,j^0}} - h_{i; v_{j^0}} \leq 0 \right] \leq C e^2 \max_{(1+\epsilon)^d; n} \frac{\max\{fd; 10 \log ng\}}{n}; (1+\epsilon) (er) \log^{+1} \frac{1}{er}$$

with probability exceeding $1 - c_1 n^{-8}$, where again, we adopt the convention that $0=0 = 0$.

This lemma immediately establishes the claim in conjunction with a union bound over all $\frac{k}{2}$ pairs of parameters. □

Proof of claim (7.11b): For this claim, we require a uniform bound on singular values uniformly over all choices of the true parameters $\mathbf{f} \in \mathbb{B}_{\text{vol}}^k(\min; \epsilon)$. We proceed by bounding the minimum singular values of *all* sub-matrices of \mathbf{A} of a certain size, and then show that each sub-matrix \mathbf{A}^j encountered over the course of the algorithm has a certain size with high probability.

In the following lemma, we use the shorthand $C; c_s := 4c_s^2 \max\{f^9(e)^2; 1\}g$.

Lemma 46. *Suppose that Assumption 20 holds, and that for a scalar $\epsilon \in (0, 1)$, the sample size obeys the lower bound $n \geq \frac{1}{\epsilon^2} \max\{4d, \frac{d+1}{\epsilon}\}$. Then we have*

$$\min_{S: |S|=n} \min_{\tilde{s} \geq s} \tilde{s} \leq \frac{1}{C_{\tilde{c}_S} \log C_{\tilde{c}_S} + 2C_{\tilde{c}_S}^{-1} \log(e^2 - \epsilon)} \frac{1}{\epsilon^2} n$$

with probability greater than $1 - 3 \exp(-n)$.

We combine this lemma with a lower bound on the size of each subset.

Lemma 47. *Suppose that $n \geq 4 \frac{kd}{\min} \log(n=kd)$. Then we have*

$$\inf_{1 \leq j \leq k} \inf_{\substack{B_{\text{vol}}(\min; \cdot) \\ \nu_{j,j} \geq 2B_{\nu_{j,j}}(r_b)}} \min_{j \in [k]} |S_j(\cdot; \cdot)| \geq n \frac{\min}{4}$$

with probability exceeding $1 - 2 \exp(-cn \frac{2}{\min})$.

We are now ready to proceed to a proof of claim (7.11b). Note that the condition of the theorem guarantees that we have $n \geq \frac{1}{\epsilon^2} \max\{4d, \frac{d+1}{\epsilon}\} \geq 4 \frac{kd}{\min} \log(n=kd)$ provided $C_{\min} \geq \frac{1}{\epsilon^2}$.

Choosing $\epsilon = \frac{1}{\min}$ and conditioning on the intersection of the pair of events guaranteed by Lemmas 46 and 47, we have

$$\frac{\inf_{1 \leq j \leq k} \inf_{\substack{B_{\text{vol}}(\min; \cdot) \\ \nu_{j,j} \geq 2B_{\nu_{j,j}}(r_b)}} \min_{j \in [k]} |S_j(\cdot; \cdot)|}{C_{\tilde{c}_S} \log C_{\tilde{c}_S} + 2C_{\tilde{c}_S}^{-1} \log(4e^2 - \frac{1}{\min})} \geq \frac{\min}{4e^2} \frac{1}{4} n$$

Combining this bound with the various high probability statements completes the proof of the claim. □

Having proved the claims, we turn to proofs of the three technical lemmas.

Proof of Lemma 45

Consider a fixed pair $(v; v)$, and as before, let $v = v - v$; we have

$$\begin{aligned} \mathbf{1} \{f h_i; v_i - h_i; v_i \geq 0\} & \leq \mathbf{1} \{f h_i; v_i - h_i; v_i \geq 0\} \\ \mathbf{1} \{h_i; v_i^2 - h_i; v_i^2 \geq h_i; v_i^2\} & \end{aligned}$$

Define the (random) set $K_v = \{i : h_i; v_i^2 \geq h_i; v_i^2\}$, and note that this quantity implicitly depends on v as well. We have the bound

$$\frac{1}{n} \sum_{i=1}^n \mathbf{1} \{f h_i; v_i - h_i; v_i \geq 0\} \leq \frac{1}{n} |K_v| \frac{1}{v^2}$$

We now show that the quantity $k_{K_v} v^k$ is bounded as desired for all pairs $(v; v)$ such that $v \in B_v(r)$. Recall that $v = (u w)$, and let

$$m = \max_{v \in B_v(r)} \left(4d; (d+1) \frac{1}{\epsilon} \log n; n^{C \log \frac{1}{\epsilon}} \right);$$

which implicitly depends on r . We claim that for all $0 < \epsilon < 1/24$, we have

$$\Pr_{v \in B_v(r)} \left[\sup_{v \in B_v(r)} |K_{v,j}| > m \right] \leq \frac{3}{1 - (3/4)^{\max\{d, 10 \log n\}}} + c n \epsilon^{c^n}; \text{ and} \quad (7.12a)$$

$$\Pr_{\substack{T \in [n]: \\ |T| > m}} \left[\sup_{v \in B_v(r)} \frac{|K_{T,v}|}{|K_{T,v}|} > d + 4m \epsilon^2 \log(n+m) \right] \leq \frac{n}{m}; \quad (7.12b)$$

Taking these claims as given, the proof of the lemma is immediate, since $\frac{n}{m} \leq (C \log(1/\epsilon))^{-1}$, so that $\log(n+m) \leq C \log(1/\epsilon)$.

We now proceed to the proofs of the two claims, (7.12a) and (7.12b). The proof of claim (7.12b) follows from claim (7.12a) by using Lemma 40 in Appendix 6.9.5, so we dedicate the rest of the proof to establishing claim (7.12a).

Proof of claim (7.12a): By definition of the set K_v , we have

$$\begin{aligned} \Pr_{v \in B_v(r)} \left[\sup_{v \in B_v(r)} |K_{v,j}| > m \right] &= \Pr_{\substack{T \in [n]: \\ |T| > m}} \left[\Pr_{v \in B_v(r)} \left[\frac{|K_{T,v}|}{|K_{T,v}|} > m \right] \right] \\ &= \Pr_{\substack{T \in [n]: \\ |T| > m}} \left[\Pr_{v \in B_v(r)} \left[\frac{|K_{T,v}|}{|K_{T,v}|} > \frac{|K_{T,v}|}{|K_{T,v}|} \right] \right] \\ &= \Pr_{\substack{T \in [n]: \\ |T| > m}} \left[\Pr_{v \in B_v(r)} \left[r^2 \frac{|K_{T,v}|}{|K_{T,v}|} > \frac{|K_{T,v}|}{|K_{T,v}|} \right] \right] \\ &= \Pr_{\substack{T \in [n]: \\ |T| > m}} \left[\Pr_{v \in B_v(r)} \left[\frac{|K_{T,v}|}{|K_{T,v}|} > \frac{|K_{T,v}|}{|K_{T,v}|} \right] \right] \\ &\quad + \Pr_{\substack{T \in [n]: \\ |T| > m}} \left[\frac{|K_{T,v}|}{|K_{T,v}|} > r^2 \frac{|K_{T,v}|}{|K_{T,v}|} \right] \end{aligned}$$

where the final step follows from the union bound and holds for all positive scalars t, g_T, n . For a fixed subset T of size $|T| = m$, we have the tail bounds

$$\Pr \left(\sup_{k \in 2R^{d+1}} \frac{k \cdot T! k^2}{k! k^2} \leq \rho + e^2 \left(\rho \sqrt{d} + d + t \right) \right) \geq 2e^{-\min\{t, t^2 g\}}, \text{ and} \tag{7.13a}$$

$$\Pr \left(\inf_{v \in 2R^{d+1}} \frac{k \cdot T v k^2}{k u k^2} \leq 3 \cdot 4c_s^2 \max\{9(e)^2, 1\} \log(1 + \frac{t}{\rho}) \right) \leq e^{-2}; \tag{7.13b}$$

where inequality (7.13a) follows from Lemma 40, and inequality (7.13b) from Lemma 53 since we have $m \leq \max\{\frac{d+1}{4}, d\}$.

Now use the shorthand $C_{\rho, c_s} := 4c_s^2 \max\{9(e)^2, 1\} g$ as before. Substituting these bounds and letting $t = 1$ be a parameter to be chosen, we have

$$\Pr \{ |K_{\rho, j}| > mg \} \leq \sum_{|T|=m+1} \left[2e^{-t} + 3(C_{\rho, c_s} r^2 \frac{\rho + e^2(\rho \sqrt{d} + d + t)}{\rho} \log(\frac{\rho + e^2(\rho \sqrt{d} + d + t)}{r^2 \rho + r^2 e^2(\rho \sqrt{d} + d + t)})) \right] e^{-2};$$

We now split the proof into two cases and choose t differently for the two cases.

Case 1, $m \leq n \leq e$: Substituting the choice $t = 4 \log(n/\rho)$, we obtain

$$\frac{\rho + e^2(\rho \sqrt{d} + d + t)}{\rho} \stackrel{(i)}{\leq} \frac{\rho + 5e \rho \sqrt{\log(n/\rho)} + \rho d}{\rho} \stackrel{(ii)}{\leq} 7e \rho \sqrt{\log(n/\rho)};$$

where step (i) follows from the bound $m \leq d$, and step (ii) from the bound $e \geq 1$. Putting together the pieces and noting that the map $x \mapsto x \log(1+x)$ is an increasing on the interval $(0; e^{-1})$, we have, for each ρ in this set, the bound

$$\Pr \{ |K_{\rho, j}| = \rho \} \leq \frac{en}{\rho} e^{-c} + 3 \sum_{|T|} 14C_{\rho, c_s}^{1+2} r e \log(n/\rho) \log \frac{1}{7re \rho \sqrt{\log(n/\rho)}};$$

For a sufficiently large constant C , and $\rho \leq n(Cre \log(1+re))$, the second term is bounded by $3(3+4) \dots$

Case 2, $\delta = n = e$: The argument for this case is identical to before (with the choice $t = 2n = \delta$). Setting $C_{\delta; C_S}^0$ to be a sufficiently small δ -dependent constant we obtain, for each δ in this range and for all $r \leq C_{\delta; C_S}^0$,

$$3C_{\delta; C_S}^0 r^2 \frac{\delta + e^2(\delta^{-\rho} d + d + \delta t)}{\delta} \leq 1 \quad (3=4)$$

where we also use the fact that $d \leq n = 2$. Once again, using the non-decreasing nature of the map $x \mapsto x \log(1+x)$ on the interval $(0; e^{-1})$, we have

$$\begin{aligned} n &\leq 42e^{-\delta t} + 3 C_{\delta; C_S}^0 r^2 \frac{\delta + e^2(\delta^{-\rho} d + d + \delta t)}{\delta} \log \frac{\delta}{r^2 \delta + r^2 e^2(\delta^{-\rho} d + d + \delta t)} \\ &\leq 2e^{-cn} + 3 \frac{r}{\frac{1}{3}} \end{aligned}$$

Putting together the pieces from both cases, we have shown that for all $r \leq C_{\delta; C_S}^0$, we obtain

$$\Pr \{f_j K_{V_j} > mg\} \leq 2ne^{-cn} + 3 \frac{\delta^{m+1}}{2ne^{-cn} + n\rho^{10^{-1} \log n}}, \quad (3=4)$$

where $\rho = (3=4)$. This completes the proof of the claim. □

Proof of Lemma 46

The proof of this lemma follows from Lemma 53 in Appendix 6.9.5 in conjunction with the union bound. In particular, we have

$$\Pr \left\{ \min_{S_j: |S_j|=n} \min_{S \supseteq S} \frac{1}{S} \leq \frac{1}{3} \right\} \leq 3 \frac{n}{n} (C_{\delta; C_S}^0 \log(1 = \delta)) \frac{n}{3} e^{\frac{(C_{\delta; C_S}^0 \log(1 = \delta))}{3}}$$

Finally, setting $\delta = \frac{1}{C_{\delta; C_S}^0 \log C_{\delta; C_S}^0 + 2C_{\delta; C_S}^0 \log(e = \delta)}$ and performing some algebraic manipulation yields the claimed bound. □

Proof of Lemma 47

Note that we have $n \geq 4 \frac{kd}{\min} \log(n=kd)$. Noting that the S_j can be thought of as the indicator vector corresponding to the intersection of k halfspaces, applying [221, Lemmas 11 and 12] in conjunction

yields the bound

$$\inf_{1 \leq j \leq k} \frac{1}{2 \text{Bvol}(\min_{j^0} \dots)} j S_j(\dots) \geq n \frac{\min}{2}$$

with probability exceeding $1 - ce^{-c'n^2_{\min}}$. Furthermore, we have

$$j S_j(\dots) \leq j S_j(\dots) \leq j S_j(\dots) \leq S_j(\dots)$$

Using the notation W for a wedge and the notation n_W for the number of points in the wedge W , we have

$$j S_j(\dots) \leq S_j(\dots) \stackrel{(i)}{\leq} \sum_{j^0=1}^{(i)} n_{W(v_{j:j^0}, v_{j:j^0})} \stackrel{(ii)}{\leq} \sum_{j^0=1}^{(ii)} \frac{1}{2} n \text{vol}(W(v_{j:j^0}, v_{j:j^0}))$$

where step (i) follows from [221, equation (33)], and step (ii) follows once again from [221, Lemma 11]. The volume of a wedge under a distribution satisfying Assumption 20 is upper bounded in Appendix 6.9.5. Applying Lemma 51, we have

$$j S_j(\dots) \leq n \frac{\min}{2} \sum_{j^0=1}^{(iii)} \frac{v_{j:j^0}}{j} \frac{v_{j:j^0}}{j^0} \log \frac{j}{v_{j:j^0}} \frac{j^0}{v_{j:j^0}} \stackrel{(iii)}{\leq} n \frac{\min}{2} k \leq n \frac{\min}{4k}$$

where in step (iii), we have used condition (7.9a) satisfied by all the parameters $v_{j:j^0}$, by which we have

$$\frac{v_{j:j^0}}{k_j} \frac{v_{j:j^0}}{j^0 k} \log \frac{k_j}{v_{j:j^0}} \frac{j^0 k}{v_{j:j^0}} \leq C_{\dots}^{(1)} \frac{1+2}{\min} \dots$$

This further implies, for a sufficiently small constant $C_{\dots}^{(1)}$, that

$$\frac{v_{j:j^0}}{k_j} \frac{v_{j:j^0}}{j^0 k} \log^{1=2} \frac{k_j}{v_{j:j^0}} \frac{j^0 k}{v_{j:j^0}} \leq \frac{\min}{4k} \dots$$

Since these steps held for an arbitrary index j , the proof of the lemma is complete. □

7.5 Proof of Corollary 3

Since the proof is more or less subsumed by Theorem 18, we only sketch the details. Let X^k denote the current iterate and X^{k+1} denote the next iterate. Proceeding as before, we have the deterministic bound

$$\|X^{k+1} - X^k\|_F^2 \leq \sum_{i=1}^n \mathbb{1}_{\{f(x_i) > i\}} \|x_i\|_2^2 + k P_X \|k\|_F^2;$$

and so this case is even simpler than the max-affine setting, since we no longer select specific sub-matrices of X on which to invert a linear system. Standard bounds on sub-Gaussian random matrices yield

$$\Pr \left[\sum_{i=1}^n X_i X_i^T \succ X \right] \leq n \exp(-c_1 \frac{d}{\log n}) \leq 2n^{-8}; \tag{7.14}$$

By assumption, we have $n \geq 2(d + 4 \sqrt{\log n})$. Thus, applying Lemmas 45 and 3 (of [221]) along with the bound (7.14) to obtain that simultaneously for all pairs (i, j) satisfying $k_i \leq k_j$, $r \geq 1/24$, we have

$$\|k_i - k_j\|_2^2 \leq c_1 \frac{d}{n} \max \left\{ (1 + \frac{1}{r}) \frac{\max\{fd, 10 \log ng\}}{n}; (1 + \frac{1}{r}) \log^{+1} \frac{1}{r} \right\};$$

$$\|k_i - k_j\|_2^2 \leq c_1 \frac{d}{n} \log \frac{n}{d};$$

with probability exceeding $1 - c_1 n^{-7}$. Choosing a small enough scalar r (depending on the tuple (c_1, c_2, c_3)) and a large enough n to make the quantity within the braces less than $3/4$ completes the proof.

7.6 Technical Lemmas and Background

7.6.1 Small ball properties and examples

We begin with a technical lemma taken from Rudelson and Vershynin [58, Corollary 1.4] that shows that product measures of small-ball distributions also satisfy the small-ball condition.

Lemma 48 ([58]). *For P_X satisfying the (c_s) -small-ball property (7.1) and $x_1, \dots, x_m \stackrel{i.i.d.}{\sim} P_X$, there is a universal constant C such that we have*

$$\sup_{\substack{u \in \mathbb{S}^{d-1} \\ w \in \mathbb{R}^d}} \Pr \left[\sum_{i=1}^m (x_i; u + w)^2 \leq m (C c_s)^m \text{ for all } w > 0 \right] < 1; \tag{7.15}$$

We also verify that log-concave distributions and the standard Gaussian distribution satisfy the small-ball condition (7.1).

7.6.2 Log-concave distribution

The following result, taken from Carbery and Wright [227, Theorem 8], provides a small-ball bound for log-concave distributions almost directly.

Lemma 49 ([227]). *Let $p : \mathbb{R}^d \rightarrow \mathbb{R}$ denote polynomial of degree (at most) ℓ , let X denote a d -dimensional random vector drawn from a log-concave distribution, and let $S = \{x \in \mathbb{R}^d : |p(x)| \geq g\}$ for each $g > 0$. Then for each $q > 0$, we have*

$$\Pr\{X \in S\} \geq g \mathbb{E}_X |p(X)|^q \geq C q^{-\ell}.$$

For a unit norm vector $u \in S^{d-1}$, setting $p(x) = \langle x, u \rangle + w$ and $q = 2$, we obtain

$$\Pr\{\langle X, u \rangle + w \geq 0\} \geq C \frac{1}{1+w^2} \geq C^{-1} w^{-2},$$

where we have used the fact that $\mathbb{E}[\langle X, u \rangle^2] = 1$ since P_X is isotropic. Since this holds for each pair $(u, w) \in S^{d-1} \times \mathbb{R}$, we have verified that X satisfies small ball condition with $(c_1; c_2) = (1/2; C^2)$.

7.6.3 Standard Gaussian distribution

This is the canonical example of a sub-Gaussian distribution satisfying a small-ball condition. Suppose X is a standard Gaussian random variable. For a unit vector $u \in S^{d-1}$, this implies that $\langle X, u \rangle + w$ is a central χ^2 random variable with 1 degree of freedom.

Lemma 50. *Let Z_ν and Z_ν^ℓ denote central and non-central χ^2 random variables with ν degrees of freedom, respectively. Then for all $p \in [0, 1]$, we have*

$$\Pr\{Z_\nu^\ell \geq p\} \geq \Pr\{Z_\nu \geq p\} \geq \frac{p}{\nu} \exp\left(-\frac{p}{\nu}\right) = \exp\left(-\frac{p}{\nu} \log\left(\frac{p}{\nu} + \frac{\nu}{p}\right)\right) \quad (7.16)$$

Applying the above lemma for $\nu = 1$ and $t = p$, we obtain

$$\Pr\{\langle X, u \rangle + w \geq 0\} \geq (e^{-1})^{1/2}.$$

Hence the standard Gaussian satisfies the small-ball condition with $(c_1; c_2) = (1/2; e)$.

For completeness, we provide a proof of Lemma 50 below.

Proof of Lemma 50

The fact that $Z_\nu^\ell \stackrel{st.}{\geq} Z_\nu$ follows from standard results that guarantee that central χ^2 random variables stochastically dominate their non-central counterparts.

The tail bound is a simple consequence of the Chernoff bound. In particular, we have for all $\rho > 0$ that

$$\begin{aligned} \Pr fZ \cdot \rho g &= \Pr f \exp(\rho Z) \cdot \exp(-\rho) g \\ &= \exp(-\rho) E[\exp(\rho Z)] \\ &= \exp(-\rho) (1 + 2^{-\rho})^{\frac{1}{2}}; \end{aligned} \tag{7.17}$$

where in the last step, we have used $E[\exp(\rho Z)] = (1 + 2^{-\rho})^{\frac{1}{2}}$, which is valid for all $\rho > 0$. Minimizing the last expression over $\rho > 0$ then yields the choice $\rho = \frac{1}{2} \frac{1}{\rho} - 1$, which is greater than 0 for all $0 < \rho < \frac{1}{2}$. Substituting this choice back into equation (7.17) proves the lemma. \square

7.6.4 Background and technical lemmas used in the proofs of Theorem 18

In this section, we collect statements and proofs of some technical lemmas used in the proofs of our results concerning the AM algorithm.

7.6.5 Bounds on the “volumes” of wedges in \mathbb{R}^d

For a pair of scalars $(w; w')$ and d -dimensional vectors $(u; u')$, recall that we define the *wedge* formed by the $d + 1$ -dimensional vectors $v = (u; w)$ and $v' = (u'; w')$ as the region

$$W(v; v') = \{x \in \mathbb{R}^d : (hx; u) + w \leq (hx; u') + w'\} \cap \{0\}^d;$$

Note that the wedge is a purely geometric object.

For any set $C \subset \mathbb{R}^d$, let

$$\text{vol}_{P_X}(C) = \Pr_{X \sim P_X} \{X \in C\}$$

denote the volume of the set under the measure corresponding to the covariate distribution.

We now bound the volume of a wedge for distributions satisfying the small-ball condition. When the distribution P_X is ϵ -sub-Gaussian and satisfies a $(\epsilon; c_s)$ -small-ball condition, we have the following result.

Lemma 51. *Suppose that Assumption 20 holds, and that for a pair of scalars $(w; w')$, d -dimensional vectors $(u; u')$, and $v = (u; w)$ and $v' = (u'; w')$, we have $\frac{kv \cdot v'k}{kuk} < 1 - \epsilon$. Then, there is a positive constant C such that*

$$\text{vol}_{P_X}(W(v; v')) \leq C \cdot C_{\epsilon; c_s}^0 \frac{kv \cdot v'k^2}{kuk^2} \log \frac{kuk}{kv \cdot v'k};$$

where $C_{\epsilon; c_s}^0$ is a constant that depends only on the tuple $(\epsilon; c_s)$.

Proof of Lemma 51

Using the notation $x = (x; 1) \in \mathbb{R}^{d+1}$ to denote the appended covariate, we have

$$\text{vol}(W(v; v^\theta)) = \Pr \{ \langle x, v \rangle \geq \langle x, v^\theta \rangle \mid \langle x, v \rangle \geq 0 \};$$

where the probability is computed with respect to Gaussian measure.

In order to prove a bound on this probability, we begin by bounding the associated indicator random variable as

$$\begin{aligned} \mathbf{1}_{\langle x, v \rangle \geq \langle x, v^\theta \rangle \mid \langle x, v \rangle \geq 0} &= \mathbf{1}_{\langle x, v \rangle \geq \langle x, v^\theta \rangle \mid \langle x, v \rangle \geq 0} \\ &\leq \mathbf{1}_{\langle x, v \rangle \geq \langle x, v^\theta \rangle \mid \langle x, v \rangle \geq 0} + \mathbf{1}_{\langle x, v \rangle < \langle x, v^\theta \rangle \mid \langle x, v \rangle \geq 0}; \end{aligned} \tag{7.18}$$

where inequality (7.18) holds for all $t \geq 0$.

We now have

$$\Pr \{ \langle x, v \rangle \geq \langle x, v^\theta \rangle \mid \langle x, v \rangle \geq 0 \} \leq \frac{c_2 t}{k \|v\|^2};$$

since P_X satisfies a small-ball condition.

Furthermore, the sub-Gaussianity of the covariate distribution yields the upper tail bound

$$\begin{aligned} \Pr \{ \langle x, v \rangle \geq \langle x, v^\theta \rangle \mid \langle x, v \rangle \geq 0 \} &\leq c_1 \exp \left(-\frac{c_2}{2 \cdot k \|v\|^2} \frac{t}{2} \right) \\ &\leq c_1 \exp \left(-\frac{c_2}{2 \cdot k \|v\|^2} \frac{t}{2} \right); \end{aligned}$$

Putting all the pieces together, we obtain

$$\text{vol}_{P_X}(W(v; v^\theta)) \leq c_1 \exp \left(-\frac{c_2}{2 \cdot k \|v\|^2} \frac{t}{2} \right) + \frac{c_2 t}{k \|v\|^2};$$

Substituting $t = 2c_2^{-2} k \|v\|^2 (\log(k \|v\|) + \log(k \|v^\theta\|))$ yields the desired result. □

7.6.6 Uniform bounds on singular values of (sub-)matrices

We now state and prove two technical lemmas that bound the maximum and minimum singular values of a matrix whose rows are drawn from a sub-Gaussian distribution obeying the small-ball property. Our results on the minimum singular value are similar in spirit to those of Rudelson and Vershynin [228], but proved under a slightly different set of assumptions.

Lemma 52. *Suppose that the covariates are drawn i.i.d. from a ψ_2 -sub-Gaussian distribution. Then for a fixed subset $S \subseteq [n]$ of size $|S| = m$ and each $t \geq 0$, we have*

$$\Pr \left(\max_{S \subseteq [n], |S|=m} \lambda_{\min}(A_S) \leq \frac{e}{\sqrt{m}} + e^2 \left(\frac{1}{\sqrt{m}} + \frac{t}{\sqrt{m}} \right) \right) \leq 2e^{-\min\{ft, t^2g\}};$$

where $e = \max \{f, 1\}g$.

The second lemma controls the minimum singular value of any sub-matrix of Σ that has sufficiently many rows. Recall that distribution P_X is isotropic and ϵ -sub-Gaussian, and satisfies the $(\epsilon; c_S)$ small-ball condition (7.1).

Lemma 53. *Suppose that Assumption 20 holds, and that $\epsilon < \min\{4d, \frac{d+1}{2}\}$. Let $e = \max\{f, 1/g\}$. Then for a fixed subset $S \subseteq [n]$ of size $|S| \geq \frac{1}{\epsilon^2} \log(1/\delta)$ and for each positive $\delta < \min\{f, g\}$, we have*

$$\Pr \left(\min_{S \subseteq [n], |S| \geq \frac{1}{\epsilon^2} \log(1/\delta)} \lambda_{\min}(\Sigma_S) \geq \frac{\delta}{3} \right) \geq 1 - \delta$$

Proof of Lemma 52

Let $\{z_i\}_{i=1}^n$ denote i.i.d. Rademacher variables, and collect these in an n -dimensional vector Z . Let $D = \text{diag}(Z)$ denote a diagonal matrix, and note that by unitary invariance of the singular values, the singular values of the matrix $\Sigma_S = D^T \Sigma D$ are the same as those of Σ_S .

By construction, the matrix Σ_S has i.i.d. rows, and the i -th row is given by $z_i(x_i; 1)$. For a $d + 1$ dimensional vector $e = (z; w)$ with $z \in \mathbb{R}^d$ and $w \in \mathbb{R}$, we have

$$\begin{aligned} \mathbb{E} \exp(h e^T z_i(x_i; 1)) &= \frac{e^w}{2} \mathbb{E} [\exp(h z^T x_i)] + \frac{e^{-w}}{2} \mathbb{E} [\exp(-h z^T x_i)] \\ &= \exp\left(\frac{h^2}{2}\right) \frac{1}{2} (e^w + e^{-w}) \\ &= \exp\left(\frac{h^2}{2}\right) \exp\left(\frac{w^2}{2}\right) \exp\left(\frac{h^2 w^2}{2}\right) \end{aligned}$$

where we have used the fact that x_i is zero-mean and ϵ -sub-Gaussian.

Since the rows of Σ_S are i.i.d., zero-mean, and ϵ -sub-Gaussian, applying [13, Theorem 6.2] immediately yields the lemma. □

Proof of Lemma 53

We let M denote the $n \times (d + 1)$ matrix Σ_S . By the variational characterization of the minimum eigenvalue, we have

$$\lambda_{\min}(M^T M) = \inf_{v \in \mathbb{S}^d} \|Mv\|^2 = \inf_{v \in \mathbb{S}^d} \|Mv\|$$

Let $Z_v = \|Mv\|$ denote a random process indexed by v , and let $Z = \inf_{v \in \mathbb{S}^d} Z_v$; we are interested bounding the lower tail of Z . Consider a ϵ -covering $\{v^1, \dots, v^N\}$ of the set \mathbb{S}^d in ℓ_2 norm, with $N \leq (1 + 2/\epsilon)^{d+1}$. Letting v^j be the closest element of the cover to v , we have

$$Z_v \leq Z_{v^j} \leq \|Z_v - Z_{v^j}\| + Z_{v^j} \leq \epsilon + Z_{v^j}$$

so that we have the bound $Z \geq \min_{j \in [N]} Z_{v^j} - \epsilon$. We have thus reduced the infimum over the unit shell to a finite minimum.

We thus have

$$\begin{aligned} \Pr \left(\min_{j \in [N]} M_j > M \right) &= \Pr \left(\max_{j \in [N]} Z_j^2 > \rho_{\cdot}^0 \right) \\ &= \Pr \left(\max_{j \in [N]} Z_j^2 > 2 \rho_{\cdot}^0 \right) + \Pr \left(\max_{j \in [N]} Z_j^2 \leq 2 \rho_{\cdot}^0 \text{ and } \max_{j \in [N]} M_j > M \right) \\ &\leq \Pr \left(\max_{j \in [N]} Z_j^2 > 2 \rho_{\cdot}^0 \right) + \Pr \left(\max_{j \in [N]} M_j > M \mid \max_{j \in [N]} Z_j^2 \leq 2 \rho_{\cdot}^0 \right); \end{aligned}$$

where we have used the union bound in each of the last two steps.

Now note that for each $j \in [N]$, the small ball condition yields the bound

$$\Pr \left(Z_j^2 > 2 \rho_{\cdot}^0 \right) = \Pr \left(\|M_j\|^2 > 2 \rho_{\cdot}^0 \right) \leq (2C_S)^d;$$

where we have used Lemma 48 to reason about the product measure of small-ball distributions.

Furthermore, since M has e sub-Gaussian rows, we may apply Lemma 52 to obtain

$$\Pr \left(\max_{j \in [N]} M_j > M \mid \max_{j \in [N]} Z_j^2 \leq 2 \rho_{\cdot}^0 \right) \leq 2 \exp \left(- \frac{e^2 \rho_{\cdot}^0}{2e^2} \left(\frac{d + \rho_{\cdot}^0}{d} \right)^2 \right);$$

which holds provided $\frac{e^2 \rho_{\cdot}^0}{2e^2} \left(\frac{d + \rho_{\cdot}^0}{d} \right)^2 \geq 1$. When $e^2 \rho_{\cdot}^0 \geq 4$, the choice $\rho_{\cdot}^0 := \frac{4}{e^2} \log^{-1}(1/e)$ ensures that

$$\frac{e^2 \rho_{\cdot}^0}{2e^2} \left(\frac{d + \rho_{\cdot}^0}{d} \right)^2 \geq \log(1/e) \geq \frac{1}{2} \log(1/e);$$

where in step (i) we have used the properties $e \leq 1$ and $\rho_{\cdot}^0 \leq 4d$. This yields the bound

$$\Pr \left(\max_{j \in [N]} M_j > M \mid \max_{j \in [N]} Z_j^2 \leq 2 \rho_{\cdot}^0 \right) \leq 2^{-2};$$

and putting together the pieces by substituting $N = 1 + \frac{2}{\rho_{\cdot}^0} e^{d+1}$, we have

$$\Pr \left(\min_{j \in [N]} M_j > M \right) \leq \left(1 + \frac{2}{\rho_{\cdot}^0} e^{d+1} \right) (2C_S)^d + 2^{-2};$$

Now note that we have $\rho_{\cdot}^0 < \min\{1, e^2\} / e$, so that $\rho_{\cdot}^0 \leq 1$, so that $1 + \frac{2}{\rho_{\cdot}^0} e^{d+1} \leq 3e^{d+1}$. Therefore, we have

$$\begin{aligned} \Pr \left(\min_{j \in [N]} M_j > M \right) &\leq (9e^2)^{d+1} (2C_S)^d + (d+1) \log^{d+1}(1/e) + 2^{-2} \\ &\leq 3 \cdot 4C_S^2 \max\{9e^2, 1\} \log^{d+1}(1/e) + 2^{-2}; \end{aligned}$$

where we have used the fact that $\rho_{\cdot}^0 \leq d + 1$. This completes the proof. □

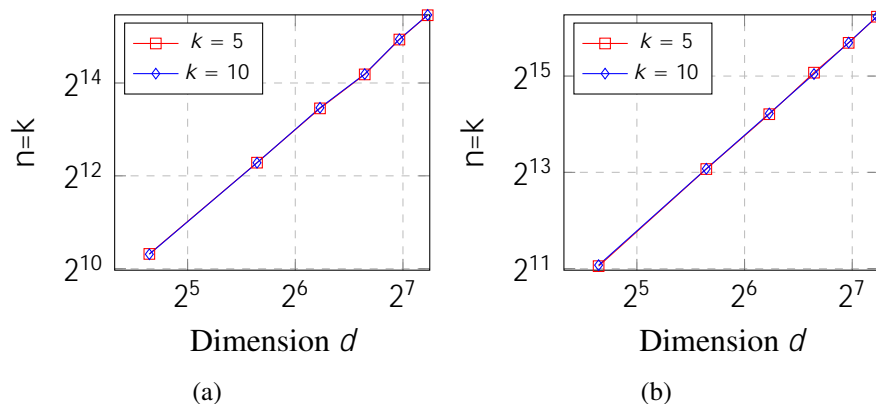


Figure 7.3: AM with Rademacher and $\mathcal{P}_{2.4}^1(\text{Bin}(10;0.4) - 4)$ covariates in log-log scale; the best-fit line to plot (a) has slope 1.94 and the best-fit line to plot (b) has slope 1.96, and hence the sample complexity scales linearly with k but quadratically with dimension d .

7.6.7 Numerical experiments: Noiseless Sample Complexity

In this section, we present some numerical experiments to illustrate the noiseless sample complexity of the AM algorithm under different covariate assumptions. We choose $\beta_i = e_i$ for $i \geq [k]$, where e_i is the i -th canonical basis. Hence, $\|\beta\|_1 = k$ and $\beta_{\min} = 1/k$. We also choose the noiseless setting, i.e., $\sigma = 0$. We say β_i is recovered if $|\hat{\beta}_i^{(t)} - \beta_i| \leq 0.01$. For a fixed dimension d , we run a linear search on the number of samples n , such that the empirical probability of success over 100 trials is more than 0.95, and output the smallest such n .

Here, we consider the distributions that do not satisfy the small ball condition. It is observed that for these distributions, the sample complexity for perfect parameter recovery has a quadratic dependence on dimension d . In particular, we draw the covariates i.i.d from Rademacher and $\mathcal{P}_{2.4}^1(\text{Bin}(10;0.4) - 4)$ distributions. In Figure 7.3, we plot the noiseless sample complexity and dimension on a log scale. Since the slope of the (best-fit) lines in Figure 7.3 is very close to 2, we conclude that the number of samples required for perfect recovery of the parameters is linear in k , but quadratic in d .

Part III

Learning and Adaptation in Bandit and RL Framework

In this part of the thesis, we study the problem of adaptation (or model selection) in bandit and Reinforcement Learning framework.

- **Chapter 8:** We study the problem of adaptation/model selection for classical stochastic linear bandits.
- **Chapter 9:** We extend our study of model selection to generic contextual bandits beyond linear structure.
- **Chapter 10:** We study the problem of model selection for Reinforcement Learning with function approximation.

Chapter 8

Problem-Complexity Adaptive Model Selection for Stochastic Linear Bandits

We consider the problem of *model selection* for two popular stochastic linear bandit settings, and propose algorithms that adapt to the *unknown* problem complexity. In the first setting, we consider the K armed mixture bandits, where the mean reward of arm $i \in [K]$, is $\mu_i + h_{i;t} \theta_i$, with $h_{i;t} \in \mathbb{R}^d$ being the known context vector and $\theta_i \in [-1; 1]$ and μ_i are unknown parameters. We define $k \in [K]$ as the problem complexity and we propose Adaptive Linear Bandit (ALB), a novel phase based algorithm that adapts to the true problem complexity, k , and achieves regret of $\mathcal{O}(k \sqrt{T})$, where k is a priori unknown. In the second setting, we consider the standard linear bandit problem (with possibly an infinite number of arms) where the sparsity of θ , denoted by $d \in \mathbb{N}$, is unknown to the algorithm. Defining d as the problem complexity (similar to [70]), we show that ALB achieves $\mathcal{O}(d \sqrt{T})$ regret, matching that of an oracle who knew the true sparsity level. This methodology is then extended to the case of finitely many arms and similar results are proven. We further verify through synthetic and real-data experiments that the performance gains are fundamental and not artifacts of mathematical bounds. In particular, we show 1.5–3x drop in cumulative regret over non-adaptive algorithms.

8.1 Introduction

We consider two canonical settings for the stochastic MAB problem. The first is the K armed mixture MAB setting, in which the mean reward from any arm $i \in [K]$ is given by $\mu_i + h_{i;t} \theta_i$, where $h_{i;t} \in \mathbb{R}^d$ is the known context vector of arm i at time t , and $\theta_i \in \mathbb{R}$, $\mu_i \in \mathbb{R}^d$ are unknown and needs to be estimated. This setting also contains the standard MAB [62, 63] when $\theta_i = 0$. Popular linear bandit algorithms, like LinUCB, OFUL (see [64–66]) handle the case with no bias ($\theta_i = 0$), while OSOM [67], the recent improvement can handle arm-bias. Implicitly, all the above algorithms assume an upper bound on the norm of θ , $\|\theta\| \leq L$, which is supplied as an input. Crucially however, the regret guarantees scale linearly in the upper bound L . In contrast, we choose $k \in [K]$ as the problem complexity, and provide a novel phase based algorithm, that, without any upper

bound on the norm $k \leq k$, adapts to the true complexity of the problem instance, and achieves a regret scaling linearly in the true norm $k \leq k$. As a corollary, our algorithm's performance matches the minimax regret of simple MAB when $\epsilon = 0$, even though the algorithm did not apriori know that $\epsilon = 0$. Formally, we consider a continuum of hypothesis classes, with each class positing a different upper bound on the norm $k \leq k$, where the complexity of a class is the upper bound posited. As our regret bound scales linearly in $k \leq k$ (the complexity of the smallest hypothesis class containing the instance) as opposed to an upper bound on $k \leq k$, our algorithm achieves model selection guarantees.

The second setting we consider is the standard linear stochastic bandit [66] with possibly an infinite number of arms, where the mean reward of any arm $x \in \mathbb{R}^d$ (arms are vectors in this case) given by $\langle \mu, x \rangle$, where $\mu \in \mathbb{R}^d$ is unknown. For this setting, we consider model selection from among a total of d different hypothesis classes, with each class positing a different cardinality for the support of μ . We exhibit a novel algorithm, where the regret scales linearly in the unknown cardinality of the support of μ . The regret scaling of our algorithm matches that of an oracle that has knowledge of the optimal support cardinality [68],[69], thereby achieving model selection guarantees. Our algorithm is the first known algorithm to obtain regret scaling matching that of an oracle that has knowledge of the true support. This is in contrast to standard linear bandit algorithms such as [66], where the regret scales linearly in d . We also extend this methodology to the case when the number of arms is finite and obtain similar regret rates matching the oracle. Model selection with dimension as a measure of complexity was also recently studied by [70], in which the classical contextual bandit [64] with a finite number of arms was considered. We clarify here that although our results for the finite arm setting yields a better (optimal) regret scaling with respect to the time horizon T and the support of μ (denoted by d), our guarantee depends on a problem dependent parameter and thus not uniform over all instances. In contrast, the results of [70], although sub-optimal in d and T , is uniform over all problem instances. Closing this gap is an interesting future direction.

8.1.1 Our Contributions

1. Successive Refinement Algorithms for Stochastic Linear Bandit - We present two novel epoch based algorithms, ALB (Adaptive Linear Bandit) - Norm and ALB - Dim, that achieve model selection guarantees for both families of hypothesis respectively. For the K armed mixture MAB setting, ALB-Norm, at the beginning of each phase, estimates an *upper bound* on the norm of $k \leq k$. Subsequently, the algorithm assumes this bound to be true during the phase, and the upper bound is re-estimated at the end of a phase. Similarly for the linear bandit setting, ALB-Dim estimates the support of μ at the beginning of each phase and subsequently only plays from this estimated support during the phase. In both settings, we show the estimates converge to the true underlying value—in the first case, the estimate of norm $\| \mu \|$ converges to the true norm, and in the second case, for all time after a random time with finite expectation, the estimated support equals the true support. Our algorithms are reminiscent of successive rejects algorithm [229] for standard MAB, with the crucial difference being that our algorithm is *non-monotone*. Once rejected, an arm is never pulled in the classical successive rejects. In contrast, our algorithm is successive

refinement and is not necessarily monotone —a hypothesis class discarded earlier can be considered at a later point of time.

2. Regret depending on the Complexity of the smallest Hypothesis Class - In the K armed mixture MAB setting, ALB-Norm’s regret scale as $\Theta(k \sqrt{T})$, which is superior compared to state of art algorithms such as OSOM [67], whose regret scales as $\Theta(L \sqrt{T})$, where L is an upper bound on k that is supplied as an input. As a corollary, we get the ‘best of both worlds’ guarantee of [67], where if $\epsilon = 0$, our regret bound recovers known minimax regret guarantee of simple MAB. Similarly, for the linear bandit setting with unknown support, ALB-Dim achieves a regret of $\Theta(d \sqrt{T})$, where d is the true sparsity of μ . This matches the regret obtained by oracle algorithms that know of the true sparsity d [68, 69]. We also apply our methodology to the case when there is a finite number of arms and obtain similar regret scaling as the oracle. ALB-Dim is the first algorithm to obtain such model selection guarantees. Prior state of art algorithm ModCB for model selection with dimension as a measure of complexity was proposed in [70], with a finite set of arms, where the regret guarantee was sub-optimal compared to the oracle. However, our regret bounds for dimension, though matches the oracle, depends on the minimum non-zero coordinate value and is thus not uniform over μ . Obtaining regret rates in this case that matches the oracle and is uniform over all μ is an interesting future work.

3. Empirical Validation - We conduct synthetic and real data experiments that demonstrate superior performance of ALB compared to state of art methods such as OSOM [67] in the mixture K armed MAB setting and OFUL [66] in the linear bandit setting. We further observe, that the performance of ALB is close to that of the oracle algorithms that know the true complexity. This indicates that the performance gains from ALB is fundamental, and not artifacts of mathematical bounds.

Motivating Example: Our model selection framework is applicable to personalized news recommendation platforms, that recommend one of K news outlets, to each of its users. The recommendation decisions to any fixed user, can be modeled as an instance of a MAB; the arms are the K different news outlets, and the platforms recommendation decision (to this user) on day t is the arm played at time t . On each day t , each news outlet i reports a story, that can be modeled by the vectors $\mu_{i;t}$, which can be obtained by embedding the stories into a fixed dimension vector space by some common embedding schemes. The reward obtained by the platform in recommending news outlet i to this user on day t can be modeled as $\mu_i + h_{i;t} \cdot \mu_i$, where μ_i captures the preference of this user to news outlet i and the vector $h_{i;t}$ captures the “interest” of the user. Thus, if a channel i on day t , publishes a news article $\mu_{i;t}$, that this user “likes”, then most likely the content $\mu_{i;t}$ is “aligned” to μ_i and have a large inner product $h_{i;t} \cdot \mu_i$. Different users on the platform however may have different biases and μ_i . Some users have strong preference towards certain topics and will read content written by any outlet on this topic (these users will have a large value of μ_i). Other users may be agnostic to topics, but may prefer a particular news outlet a lot (for ex. some users like fox news exclusively or CNN exclusively, regardless of the topic). These users will have low μ_i .

In such a multi-user recommendation application, we show that our algorithm ALB-Norm that tailors the model class for each user separately is more effective (lesser regret), than to, employ

a (non-adaptive) linear bandit algorithm for each user. We further show that our algorithms are also more effective than state of art model selection algorithms such as OSOM [67], which posits a ‘binary’ model - users either assign a 0 weight to topic or assign a potentially large weight to topic. Furthermore the heterogeneous complexity in this application can also be captured by the cardinality of the support of θ ; different people are interested in different sub-vectors of θ which the recommendation platform is not aware of a priori. In this context, our adaptive algorithm ALB-Dim that tailors to the interest of the individual user achieves better performance compared to non-adaptive linear bandit algorithms.

8.2 Related Work

Model selection for MAB are only recently being studied [230, 231], with [67], [70] being the closest to our work. OSOM was proposed in [67] for model selection in the K armed mixture MAB from two hypothesis classes —a “simple model” where $k = k = 0$, or a “complex model”, where $0 < k = k = L$. OSOM was shown to obtain a regret guarantee of $O(\log(T))$ when the instance is simple and $\Theta(L \sqrt{T})$ otherwise. We refine this to consider a *continuum* of hypothesis classes and propose ALB-Norm, which achieves regret $\Theta(k \sqrt{T})$, a superior guarantee (which we also empirically verify) compared to OSOM. Model selection with dimension as a measure of complexity was recently initiated in [70], where an algorithm ModCB was proposed. The setup considered in [70] was that of contextual bandits [64] with a fixed and finite number of arms. ModCB in this setting was shown to achieve a regret scaling that is sub-optimal compared to the oracle. In contrast, we consider the linear bandit setting with a continuum of arms [66], and ALB-Dim achieves a regret scaling matching that of an oracle. The continuum of arms allows ALB-Dim a finer exploration of arms, that enables it to learn the support of θ reliably and thus obtain regret matching that of the oracle. However, our regret bounds depend on the magnitude of the minimum non-zero value of θ and is thus not uniform over all θ . Obtaining regret rates matching the oracle that holds uniformly over all θ is an interesting future work.

Corral was proposed in [230], by casting the optimal algorithm for each hypothesis class as an expert, with the forecaster’s performance having low regret with respect to the best expert (best model class). However, Corral can only handle finitely many hypothesis classes and is not suited to our setting with continuum hypothesis classes.

Adaptive algorithms for linear bandits have also been studied in different contexts from ours. The papers of [232, 233] consider problems where the arms have an unknown structure, and propose algorithms adapting to this structure to yield low regret. The paper [234] proposes an algorithm in the adversarial bandit setup that adapt to an unknown structure in the adversary’s loss sequence, to obtain low regret. The paper of [235] consider adaptive algorithms, when the distribution changes over time. In the context of online learning with full feedback, there have been several works addressing model selection [236–239]. In the context of statistical learning, model selection has a long line of work (for eg. [240], [241], [242], [243], [244] [245]). However, the bandit feedback in our setups is much more challenging and a straightforward adaptation of algorithms developed for either statistical learning or full information to the setting with bandit feedback is not feasible.

8.3 Norm as a measure of Complexity

8.3.1 Problem Formulation

In this section, we formally define the problem. At each round $t \in [T]$, the player chooses one of the K available arms. Each arm has a context $f_{i,t} \in \mathbb{R}^d$ that changes over time t . Similar to the standard stochastic contextual bandit framework, the context vectors for each arm is chosen independently of all other arms and of the past time instances.

We assume that there exists an underlying parameter $\mu \in \mathbb{R}^d$ and biases $f_1, \dots, f_K \in \mathbb{R}^d$ each taking value in $[-1, 1]$ such that the mean reward of an arm is a linear function of the context of the arm. The reward for playing arm i at time t is given by, $g_{i,t} = \mu^\top f_{i,t} + \eta_{i,t}$, where $f_{i,t} \in \mathbb{R}^d$ are i.i.d zero mean and $\eta_{i,t}$ sub-Gaussian noise. The context vector satisfies

$$\mathbb{E}[\eta_{i,t} | f_{j,s}; j,s \in [K]; s \in [t-1]] = 0;$$

and

$$\mathbb{E}[\eta_{i,t} | f_{j,s}; j,s \in [K]; s \in [t-1]] < \frac{1}{\min_i \|\mu_i\|}.$$

The above setting is popularly known as stochastic contextual bandit [67]. In the special case of $\mu = 0$, the above model reduces to $g_{i,t} = \mu_i + \eta_{i,t}$. Note that in this setting, the mean reward of arms are fixed, and not dependent on the context. Hence, this corresponds to a simple *multi-armed bandit* setup and standard algorithms (like UCB [63]) can be used as a learning rule. At round t , we define $i_t = \arg\max_{j \in [K]} [\mu_j + \eta_{j,t}]$ as the best arm. Also let an algorithm play arm A_t at round t . The regret of the algorithm upto time T is given by,

$$R(T) = \sum_{s=1}^T \mu_{i_s} + \eta_{i_s; s} - \sum_{s=1}^T \mu_{A_s} + \eta_{A_s; s}.$$

Throughout the paper, we use $C; C_1; \dots; C_1; \dots$ to denote positive universal constants, the value of which may differ in different instances.

We define a new notion of complexity for stochastic linear bandits; and propose an algorithm that adapts to it. We define k as the problem complexity for the linear bandit instance. Note that if $k = 0$, the linear bandit model reduces to the simple multi-armed bandit setting. Furthermore, the cumulative regret $R(T)$ of linear bandit algorithms (like OFUL [66] and OSOM [67]) scales linearly with k ([67]). Hence, k constitutes a natural notion of model complexity. In Algorithm 11, we propose an adaptive scheme which adapts to the true complexity of the problem, k . Instead of assuming an upper-bound on k , we use an initial exploration phase to obtain a rough estimate of k and then successively refine it over multiple epochs. The cumulative regret of our proposed algorithm actually scales linearly with k .

8.3.2 Adaptive Linear Bandit (norm)—(ALB-Norm algorithm)

We present the adaptive scheme in Algorithm 11. Note that Algorithm 11 depends on the subroutine OFUL⁺. Observe that at each iteration, we estimate the bias f_1, \dots, f_K and μ separately. The

Algorithm 11: Adaptive Linear Bandit (Norm)

- 1: **Input:** Initial exploration period τ , the phase length T_1 , $\tau > 0$, $\tau_s > 0$.
 - 2: Select an arm at random, sample rewards τ times
 - 3: Obtain initial estimate (b_1) of k according to Section 8.3.3
 - 4: **for** $t = 1; 2; \dots; K$ **do**
 - 5: Play arm t , receive reward $g_{t,t}$
 - 6: **end for**
 - 7: Define $S = \frac{1}{\tau} \sum_{i=1}^{\tau} g_{i,t}$
 - 8: **for** epochs $i = 1; 2; \dots; N$ **do**
 - 9: Use S as pure-exploration reward
 - 10: Play $\text{OFUL}^+(b_i)$ until the end of epoch i (denoted by E_i)
 - 11: At $t = E_i$, refine estimate of k as, $b_{i+1} = \max_{C_{E_i}} k$
 - 12: Set $T_{i+1} = 2T_i$, $\tau_{i+1} = \frac{\tau}{2}$.
 - 13: **end for**
 - 14: $\text{OFUL}^+(b)$:
 - 15: **Input:** Parameters b , $\tau > 0$, number of rounds \bar{T}
 - 16: **for** $t = 1; 2; \dots; \bar{T}$ **do**
 - 17: Select the best arm estimate as $j_t = \operatorname{argmax}_{i \in [K]} \max_{C_{t-1}} \tau_{i,t}^{-1} + h_{i,t} \tau_i g$,
where $\tau_{i,t}$ and C_t are given in Section 8.3.2.
 - 18: Play arm j_t , and update $\frac{1}{\tau} \sum_{i=1}^{\tau} g_{i,t}$ and C_t
 - 19: **end for**
-

estimation of the bias involves a simple sample mean estimate with upper confidence level, and the estimation of τ involves building a confidence set that shrinks over time.

In order to estimate τ , we use a variant of the popular OFUL [66] algorithm with arm bias. We refer to the algorithm as OFUL^+ . Algorithm 11 is epoch based, and over multiple epochs, we successively refine the estimate of k . We start with a rough over-estimate of k (obtained from a pure exploration phase), and based on the confidence set constructed at the end of the epoch, we update the estimate of k . We argue that this approach indeed correctly estimates k with high probability over a sufficiently large time horizon T .

We now discuss the algorithm OFUL^+ . A variation of this was proposed in [67] in the context of model selection between linear and standard multi-armed bandits. We use $\tau_{i,t}$ to address the bias term, which we define shortly. The parameters b and τ are used in the construction of the confidence set C_t . Suppose OFUL^+ is run for a total of \bar{T} rounds and plays arm A_s at time s . Let $T_i(t)$ be the number of times OFUL^+ plays arm i until time t . Also, let b be the current estimate of k . We define,

$$g_{i;t} = \frac{1}{T_i(t)} \sum_{s=1}^t g_{i;s} \mathbf{1}_{A_s = i}$$

With this, we have ¹

$$\tilde{r}_{i,t} = g_{i,t} + c \left(\frac{1}{\sqrt{t}} + b \right) \frac{d}{T_i(t)} \log \frac{1}{\delta} ;$$

The confidence interval C_t , is defined as

$$C_t = \left\{ \hat{\theta}_t \in \mathbb{R}^d : \|\hat{\theta}_t - \theta\| \leq K(b; t; \mathcal{F})g \right\}$$

where $\hat{\theta}_t$ is the least squares estimate defined as

$$\hat{\theta}_t = \left(\sum_{k=1}^t A_{k+1:k+1} + I \right)^{-1} \sum_{k=1}^t A_{k+1:k+1} G_{k+1:t}$$

with $\sum_{k=1}^t A_{k+1:k+1}$ as a matrix having rows $\sum_{k=1}^t A_{k+1:k+1}^{(1)}, \dots, \sum_{k=1}^t A_{k+1:k+1}^{(d)}$ and $G_{k+1:t} = [g_{A_{k+1:k+1}}, \dots, g_{A_t:t}]^T$. The radius of C_t is given by (see Section 8.8.1 for complete expression),

$$K(b; t; \mathcal{F}) = c \frac{\left(\frac{\rho_{\bar{d}}}{\min_t} + b \right)^d}{\sqrt{t}} \log(K \mathcal{F} =) ;$$

Lemma 2 of [67] shows that $\mathbb{P} \left(\sum_{t=1}^T C_t \right)$ with probability ² at least $1 - 4\delta$.

8.3.3 Construction of initial estimate b_1

We select an arm at random (without loss of generality, assume that this is arm 1), and sample rewards (in an i.i.d fashion) for $2/\delta$ times, where $\delta > 0$ is a parameter to be fed to the Algorithm 11. In order to kill the bias of arm 1, we take pairwise differences and form: $y(1) = g_{1,1} - g_{1,2}; y(2) = g_{1,3} - g_{1,4}$ and so on. Augmenting $y(\cdot)$, we obtain: $Y = X \theta + \tilde{r}$; where the i -th row of X is $(1, 2i+1, \dots, 2i+2)$, the i -th element of \tilde{r} is $g_{1,2i+1} - g_{1,2i+2}$. Hence, the least squares estimate, $\hat{\theta}^{(s)}$ satisfies $\|\hat{\theta}^{(s)} - \theta\| \leq \frac{\rho_{\bar{d}}}{\sqrt{2}} \frac{d}{\log(1/\delta)}$, with probability exceeding $1 - \delta$ ([13]). We set the initial estimate

$$b_1 = \max_{k \in [K]} \left\{ \frac{\rho_{\bar{d}}}{\sqrt{2}} \frac{d}{\log(1/\delta)}; 1/g \right\}$$

and this satisfies $b_1 \geq \theta_k$ and $b_1 \geq 1$ with probability at least $1 - \delta$.

8.3.4 Regret Guarantee of Algorithm 11

We now obtain an upper bound on the cumulative $R(T)$ with Algorithm 11 with high probability. For theoretical tractability, we assume that OFUL⁺ restarts at the start of each epoch. We have the following lemma regarding the sequence $\{b_i g_{i=1}^1\}$ of estimates of k : b_i :

¹For complete expression, see Section 8.8.1

²There is a typo in the proof of regret in [67]. We correct the typo, and modify the definition of $\tilde{r}_{i,t}$ and $K(b; t; \mathcal{F})$. As a consequence, the high probability bounds change a little.

Lemma 54. *With probability exceeding $1 - 8^{-1} \epsilon$, the sequence $\{b_i g_{i=1}^1\}$ converges to k at a rate $O(\frac{1}{\sqrt{T}})$, and we obtain $b_i \in (c_1 k - k + c_2)$ for all i , provided $T_1 \geq C_1 (\max\{p; q\} b_1)^2 d$, where $C_1 > 9$, and $p = \lfloor \frac{14 \log(\frac{2K T_1}{1})}{\rho_{\min}} \rfloor$; $q = \lfloor \frac{2C \log(\frac{2K T_1}{1})}{\rho_{\min}} \rfloor$:*

Hence, the sequence converges to k at an exponential rate. We have the following guarantee on the cumulative regret $R(T)$:

Theorem 20. *Suppose $T_1 > \max\{T_{\min}(\cdot; T); C_1 (\max\{p; q\} b_1)^2 d\}$, where $C_1 > 9$ and $T_{\min}(\cdot; T) = (\frac{16}{2 \rho_{\min}} + \frac{8}{3 \rho_{\min}}) \log(\frac{2dT}{1})$. Then, with probability at least $1 - 8^{-1} \epsilon$, we have*

$$R(T) \leq C_1 (2 + K) k + C(k + 1) (\frac{\rho_{\min}}{K} + \frac{\rho_{\min}}{d}) \frac{\rho_{\min}}{T} \log(K T_1) \log(T/T_1):$$

Remark 32. Note that the regret bound depends on the problem complexity k , and we prove that Algorithm 11 adapts to this complexity. Ignoring the log factors, Algorithm 11 has a regret of $O((1 + k) (\frac{\rho_{\min}}{K} + \frac{\rho_{\min}}{d}) \frac{\rho_{\min}}{T})$ with high probability.

Remark 33. (Matches Linear Bandit algorithm) Note that the above bound matches the regret guarantee of the linear bandit algorithm with bias as presented in [67].

Remark 34. (Matches UCB when $\epsilon = 0$) When $\epsilon = 0$ (the simplest model, without any contextual information), Algorithm 1 recovers the minimax regret of UCB algorithm. Indeed, substituting $k = 0$ in the above regret bound yields $R(T) = O(\frac{\rho_{\min}}{K T})$, with high probability, provided $K > d$. Hence, we obtain the “best of both worlds” results with simple model ($\epsilon = 0$) and contextual bandit model ($\epsilon \neq 0$).

8.4 Dimension as a Measure of Complexity - Continuum Armed Setting

In this section, we consider the standard stochastic linear bandit model in d dimensions [66], with the dimension as a measure of complexity. The setup in this section is almost identical to that in Section 8.3.1, with the 0 arm biases and a continuum collection of arms denoted by the set $A := \{x \in \mathbb{R}^d : \|x\| \leq 1\}$.³ Thus, the mean reward from any arm $x \in A$ is $\langle \mu, x \rangle$, where $\mu \in \mathbb{R}^d$. We assume that μ is d -sparse, where d is a priori unknown to the algorithm. Thus, unlike in Section 8.3, there is no i.i.d. context sampling in this section. We consider a sequence of d nested hypothesis classes, where each hypothesis class $i \in [d]$ models μ as a i -sparse vector. The goal of the forecaster is to minimize the regret, namely $R(T) := \sum_{t=1}^T [\langle \mu, x_t^* \rangle - \langle \mu, x_t \rangle]$, where at any time t , x_t is the action recommended by an algorithm and $x_t^* = \arg\max_{x \in A} \langle \mu, x \rangle$. The regret $R(T)$ measures the loss in reward of the forecaster with that of an oracle that knows μ and thus can compute x_t^* at each time.

³Our algorithm can be applied to any compact set $A \subset \mathbb{R}^d$, including the finite set as shown in Section 8.8.6.

Algorithm 12: Adaptive Linear Bandit (Dimension)

```

1: Input: Initial Phase length  $T_0$  and slack  $\epsilon > 0$ .
2:  $\mathbf{b}_0 = \mathbf{1}$ ,  $T_{-1} = 0$ 
3: for Each epoch  $i \in \{0; 1; 2; \dots\}$  do
4:    $T_i = 25^i T_0$ ,  $\epsilon_i = \frac{\epsilon}{2^i}$ ,  $\delta_i = \frac{\epsilon}{2^i}$ 
5:    $D_i := \{j : \mathbf{b}_{ij} \geq \frac{\epsilon_i}{2} g\}$ 
6:   for Times  $t \in \{T_{i-1} + 1; \dots; T_i\}$  do
7:     Play OFUL( $1; \epsilon_i$ ) only restricted to coordinates in  $D_i$ . Here  $\epsilon_i$  is the probability slack parameter and 1 represents  $\mathbf{1}_{k \in D_i}$ .
8:   end for
9:   for Times  $t \in \{T_i + 1; \dots; T_i + 5^i d \frac{T_0}{\epsilon}\}$  do
10:    Play an arm from the action set  $A$  chosen uniformly and independently at random.
11:   end for
12:    $\mathbf{S}_i \in \mathbb{R}^{S_i \times d}$  with each row being the arm played during all random explorations in the past.
13:    $\mathbf{y}_i \in \mathbb{R}^{S_i}$  with  $i$ -th entry being the observed reward at the  $i$ -th random exploration in the past
14:    $\mathbf{b}_{i+1} = (\sum_{i=0}^i \mathbf{S}_i \mathbf{y}_i)^{-1} \sum_{i=0}^i \mathbf{y}_i$ , is a  $d$  dimensional vector
15: end for

```

8.4.1 ALB-Dim Algorithm

The algorithm is parametrized by $T_0 \in \mathbb{N}$, which is given in Equation (8.1) in the sequel and slack $\epsilon \in (0; 1)$. As in the previous case, ALB-Dim proceeds in phases numbered $0; 1; \dots$ which are non-decreasing with time. At the beginning of each phase, ALB-Dim makes an estimate of the set of non-zero coordinates of \mathbf{b} , which is kept fixed throughout the phase. Concretely, each phase i is divided into two blocks - (i) a regret minimization block lasting $25^i T_0$ time slots, (ii) followed by a random exploration phase lasting $5^i d \frac{T_0}{\epsilon}$ time slots. Thus, each phase i lasts for a total of $25^i T_0 + 5^i d \frac{T_0}{\epsilon}$ time slots. At the beginning of each phase $i = 0$, $D_i \subseteq [d]$ denotes the set of ‘active coordinates’, namely the estimate of the non-zero coordinates of \mathbf{b} . Subsequently, in the regret minimization block of phase i , a fresh instance of OFUL [66] is spawned, with the dimensions restricted only to the set D_i and probability parameter $\epsilon_i := \frac{\epsilon}{2^i}$. In the random exploration phase, at each time, one of the possible arms from the set A is played chosen uniformly and independently at random. At the end of each phase $i = 0$, ALB-Dim forms an estimate \mathbf{b}_{i+1} of \mathbf{b} , by solving a least squares problem using all the random exploration samples collected till the end of phase i . The active coordinate set D_{i+1} , is then the coordinates of \mathbf{b}_{i+1} with magnitude exceeding $2^{-(i+1)}$. The pseudo-code is provided in Algorithm 12, where, $S_i = 0$, S_i in lines 15 and 16 is the total number of random-exploration samples in all phases upto and including i .

8.4.2 Main Result

We first specify, how to set the input parameter T_0 , as function of ϵ . For any $N \geq d$, denote by A_N to be the $N \times d$ random matrix with each row being a vector sampled uniformly and independently from the unit sphere in d dimensions. Denote by $M_N := \frac{1}{N} E[A_N^T A_N]$, and by $\lambda_{\max}^{(N)}$; $\lambda_{\min}^{(N)}$, to be the largest and smallest eigenvalues of M_N . Observe that as M_N is positive semi-definite ($0 \leq \lambda_{\min}^{(N)} \leq \lambda_{\max}^{(N)}$) and almost-surely full rank, i.e., $P[\lambda_{\min}^{(N)} > 0] = 1$. The constant T_0 is the smallest integer such that

$$P_{T_0} = \max \left\{ \frac{32}{(\lambda_{\min}^{(N)})^2} \ln(2d); \frac{4}{3} \frac{(6 \lambda_{\max}^{(N)} + \lambda_{\min}^{(N)}) (d + \lambda_{\max}^{(N)})}{(\lambda_{\min}^{(N)})^2} \ln(2d) \right\} \quad (8.1)$$

Remark 35. T_0 in Equation (8.1) is chosen such that, at the end of phase 0, $P[\|j\|_0 \leq \|j\|_1 - 1] \geq 1 - \epsilon$.

A formal statement of the Remark is provided in Lemma 55 in Appendix 8.8.2.

Theorem 21. *Suppose Algorithm 12 is run with input parameters $\epsilon \in (0; 1)$, and T_0 as given in Equation (8.1), then with probability at-least $1 - \epsilon$, the regret after a total of T arm-pulls satisfies*

$$R_T \leq \frac{50}{4.65} T_0 + 25 \sqrt{\frac{T}{d}} \left[1 + 4 \sqrt{\frac{T}{d} \ln\left(1 + \frac{25T}{d}\right)} \left(1 + \sqrt{2 \ln\left(\frac{T}{T_0}\right) + d \ln\left(1 + \frac{25T}{d}\right)} \right) \right]$$

The parameter $\epsilon > 0$ is the minimum magnitude of the non-zero coordinate of θ , i.e., $\epsilon = \min_{j: \theta_j \neq 0} |\theta_j|$ and d the sparsity of θ , i.e., $d = \#\{j: \theta_j \neq 0\}$.

In order to parse this result, we give the following corollary.

Corollary 4. *Suppose Algorithm 12 is run with input $\epsilon \in (0; 1)$, and $T_0 = \Theta(d^2 \ln^2 \frac{1}{\epsilon})$ given in Equation (8.1), then with probability at-least $1 - \epsilon$, the regret after T times satisfies*

$$R_T = O\left(\frac{d^2}{4.65} \ln^2(d)\right) + \Theta\left(d \sqrt{\frac{T}{d}}\right)$$

Remark 36. The constants in the Theorem are not optimized. In particular, the exponent of ϵ can be made arbitrarily close to 4, by setting $\epsilon_i = C^{-i}$ in Line 4 of Algorithm 12, for some appropriately large constant $C > 1$, and increasing $T_i = (C^i)^4 T_0$, for appropriately large C^i ($C^i \geq C^4$).

Discussion - The regret of an oracle algorithm that knows the true complexity d scales as $\Theta(d \sqrt{\frac{T}{d}})$ [68, 69], matching ALB-Di m's regret, upto an additive constant independent of time. ALB-Di m is the first algorithm to achieve such model selection guarantees. On the other hand, standard linear bandit algorithms such as OFUL achieve a regret scaling $\Theta(d \sqrt{\frac{T}{d}})$, which is much larger compared to that of ALB-Di m, especially when $d \ll d$, and ϵ is a constant. Numerical simulations further confirms this deduction, thereby indicating that our improvements are fundamental and not from mathematical bounds. Corollary 4 also indicates that ALB-Di m has higher regret if ϵ is lower. A

small value of ϵ makes it harder to distinguish a non-zero coordinate from a zero coordinate, which is reflected in the regret scaling. Nevertheless, this only affects the *second order term as a constant*, and the dominant scaling term only depends on the true complexity d , and not on the underlying dimension d . However, the regret guarantee is not uniform over all ϵ as it depends on ϵ . Obtaining regret rates matching the oracles and that hold uniformly over all ϵ is an interesting avenue of future work.

8.5 Dimension as a Measure of Complexity - Finite Armed Setting

8.5.1 Problem Setup

In this section, we consider the model selection problem for the setting with finitely many arms in the framework studied in [70]. At each time $t \geq [T]$, the forecaster is shown a context $X_t \in X$, where X is some arbitrary ‘feature space’. The set of contexts $(X_t)_{t=1}^T$ are i.i.d. with $X_t \in D$, a probability distribution over X that is known to the forecaster. Subsequently, the forecaster chooses an action $A_t \in A$, where the set $A := \{1, \dots, K\}$ are the K possible actions chosen by the forecaster. The forecaster then receives a reward $Y_t := h(X_t; A_t) + \epsilon_t$. Here $(\epsilon_t)_{t=1}^T$ is an i.i.d. sequence of 0 mean sub-gaussian random variables with sub-gaussian parameter σ^2 that is known to the forecaster. The function⁴ $h : X \times A \rightarrow \mathbb{R}^d$ is a known feature map, and $\mu \in \mathbb{R}^d$ is an unknown vector. The goal of the forecaster is to minimize its regret, namely $R(T) := \sum_{t=1}^T \mathbb{E}[h(X_t; A_t) - i]$, where at any time t , conditional on the context X_t , $A_t \in \arg\max_{a \in A} h(X_t; a)$. Thus, A_t is a random variable as X_t is random.

To describe the model selection, we consider a sequence of M dimensions $1 \leq d_1 < d_2 < \dots < d_M := d$ and an associated set of feature maps $(h^m)_{m=1}^M$, where for any $m \in [M]$, $h^m : X \times A \rightarrow \mathbb{R}^{d_m}$, is a feature map embedding into d_m dimensions. Moreover, these feature maps are nested, namely, for all $m \in [M-1]$, for all $x \in X$ and $a \in A$, the first d_m coordinates of $h^{m+1}(x; a)$ equals $h^m(x; a)$. The forecaster is assumed to have knowledge of these feature maps. The unknown vector μ is such that its first d_m coordinates are non-zero, while the rest are 0. The forecaster does not know the true dimension d_m . If this were known, then standard contextual bandit algorithms such as LinUCB [64] can guarantee a regret scaling as $\mathcal{O}(\sqrt{d_m T})$. In this section, we provide an algorithm in which, even when the forecaster is unaware of d_m , the regret scales as $\mathcal{O}(\sqrt{d_m T})$. However, this result is non uniform over all ϵ as, we will show, depends on the minimum non-zero coordinate value in μ .

Model Assumptions We will require some assumptions identical to the ones stated in [70]. Let $k \leq k_2 \leq 1$, which is known to the forecaster. The distribution D is assumed to be known to the forecaster. Associated with the distribution D is a matrix $M := \frac{1}{K} \sum_{a \in A} \mathbb{E} [h^M(x; a) h^M(x; a)^T]$ (where $x \in D$), where we assume its minimum eigen value $\min(\lambda(M)) > 0$ is strictly positive.

⁴Superscript M will become clear shortly

Further, we assume that, for all $a \in A$, the random variable $M(x; a)$ (where $x \in D$ is random) is a sub-gaussian random variable with (known) parameter σ^2 .

8.5.2 ALB-Di m Algorithm

The algorithm in this case is identical to that of Algorithm 12, except with the difference that in place of OFUL, we use SupLinRel of [246] as the black-box. The full details of the Algorithm are provided in Appendix 8.8.6.

8.5.3 Main Result

For brevity, we only state the Corollary of our main Theorem (Theorem 22) which is stated in Appendix 8.8.6.

Corollary 5. *Suppose Algorithm 13 is run with input parameters $\beta \in (0; 1)$, and $T_0 = \Theta(d^2 \ln^2 \frac{1}{\beta})$ given in Equation (8.15), then with probability at-least $1 - \beta$, the regret after T times satisfies*

$$R_T \leq O\left(\frac{d^2}{4.65} \ln^2(d) \ln \frac{TK}{T} + \Theta\left(\frac{d}{T}\right)\right);$$

where $\sigma = \min_{i, j: i \neq j} \sigma_{ij}$ and d the sparsity of Σ .

Discussion - Our regret scaling (in time) matches that of an oracle that knows the true problem complexity and thus obtains a regret scaling of $\Theta\left(\frac{d}{T}\right)$. This, thus improves on the rate compared to that obtained in [70], whose regret scaling is sub-optimal compared to the oracle. On the other hand however, our regret bound depends on σ and is thus not uniform over all Σ , unlike the bound in [70] that is uniform over Σ . Thus, in general, our results are not directly comparable to that of [70]. It is an interesting future work to close the gap and in particular, obtain the regret matching that of an oracle to hold uniformly over all Σ .

8.6 Simulations

Synthetic Experiments We compare ALB-Norm with the (non-adaptive) OFUL⁺ and an *oracle* that knows the problem complexity apriori. The oracle just runs OFUL⁺ with the known problem complexity. We choose the bias $\beta \in [0.1; 1]$, and the additive noise to be zero-mean Gaussian random variable with variance 0.5. At each round of the learning algorithm, we sample the context vectors from a d -dimensional standard Gaussian, $N(0; I_d)$. We select $d = 50$, the number of arms, $K = 75$, and the initial epoch length as 100. In particular, we generate the true Σ in 2 different ways: (i) $k = 0.1$, but the initial estimate $b_1 = 10$, and (ii) $k = 1$, with the initial estimate $b_1 = 10$.

In panel (a) and (b) of Figure 8.1, we observe that, in setting (i), OFUL⁺ performs poorly owing to the gap between k and b_1 . On the other hand, ALB-Norm is sandwiched between the OFUL⁺

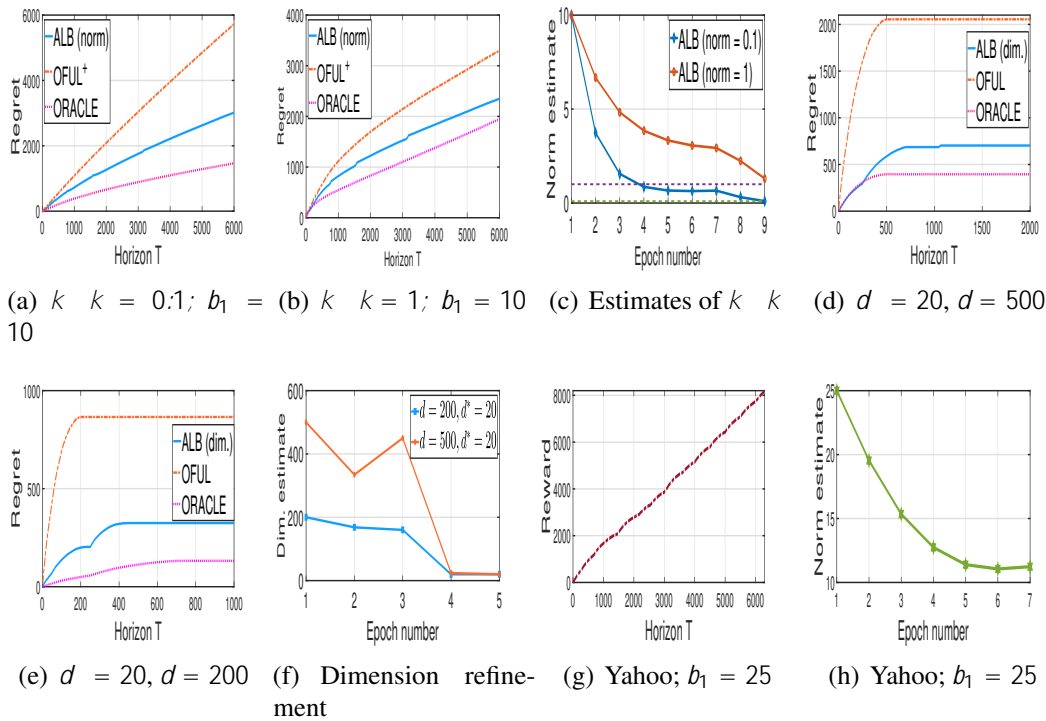


Figure 8.1: Synthetic and real-data experiments, validating the effectiveness of Algorithm 1 and 12. All the results are averaged over 25 trials.

and the oracle. Similar things happen in setting (ii). In panel (c), we show that the norm estimates of ALB-Norm improves over epochs, and converges to the true norm very quickly.

In panel (d)-(f) of Figure 8.1, we compare the performance of ALB-Dim with the OFUL ([66]) algorithm and an *oracle* who knows the true support of \mathbf{a} priori. For computational ease, we set $a_i = 2^{-i}$ in simulations. We select \mathbf{a} to be $d = 20$ -sparse, with the smallest non-zero component, $a_d = 0.12$. We have 2 settings: (i) $d = 500$ and (ii) $d = 200$. In panel (d) and (e), we observe a huge gap in cumulative regret between ALB-Dim and OFUL, thus showing the effectiveness of dimension adaptation. In panel (f), we plot the successive dimension refinement over epochs. We observe that within 4–5 epochs, ALB-Dim finds the sparsity of \mathbf{a} .

Real-data experiment Here, we evaluate the performance of ALB-Norm on Yahoo! ‘Learning to Rank Challenge’ dataset ([247]). In particular, we use the file `set2_test.txt`, which consists of 103174 rows and 702 columns. The first column denotes the rating, $f_0; 1; \dots; 4g$ given by the user (which is taken as reward); the second column denotes the user id, and the rest 700 columns denote the context of the user. After selecting 20,000 rows and 50 columns at random (several other random selections yield similar results), we cluster the data by running k means algorithm with $k = 500$. We treat each cluster as a bandit arm with mean reward as the empirical mean of the individual rating in the cluster, and the context as the centroid of the cluster. This way, we obtain a bandit setting with $K = 500$ and $d = 50$.

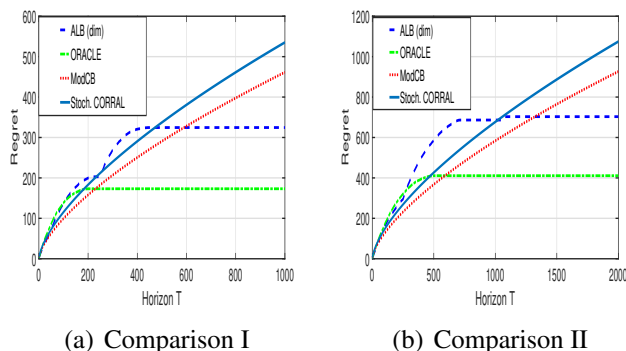


Figure 8.2: Comparison with Stochastic Corral and ModCB

Assuming (reward, context) coming from a linear model (with bias, see Section 6.2), we use ALB-Norm to estimate the bias and k simultaneously. In panel (g), we plot the cumulative reward accumulated over time. We observe that the reward is accumulated over time in an almost linear fashion. We also plot the norm estimate, k over epochs in panel (h), starting with an initial estimate of 25. We observe that within 6 epochs the estimate stabilizes to a value of 11.1. This shows that ALB-Norm adapts to the actual k .

Comparison of ALB (dim): When Σ is sparse, we compare ALB-Dim with 3 baselines: (i) the ModCB algorithm of [70] (ii) the Stochastic Corral algorithm of [248] and (iii) an oracle which knows the support of Σ . We select Σ to be $d = 20$ sparse, with dimension $d = 200$ and $d = 500$. The smallest non-zero component of Σ is 0.12. For ModCB, we use ILOVETOCONBANDITS algorithm, similar to [70]. We select the cardinality of action set as 2 and select the sub-Gaussian parameter of the embedding as unity. In Figure 8.2, we observe that, the regret of ALB (dim) is better than ModCB and Stochastic Corral. The theoretical regret bound for ModCB scales as $O(T^{2=3})$ (which is much larger than the ALB-Dim algorithm we propose), and Figure 1(c), validates this. The Stochastic Corral algorithm treats the base algorithms as bandit arms (with bandit feedback), as opposed to ALB-Dim which, at each arm-pull, updates the information about all the base algorithms. Thus, (Figs 1(d), 1(e)), ALB-Dim has a superior performance compared to Stochastic Corral.

8.7 Conclusion

In this paper, we considered refined model selection for linear bandits, by defining new notions of complexity. We gave two novel algorithms ALB-Norm and ALB-Dim that successively refines the hypothesis class and achieves model selection guarantees; regret scaling in the complexity of the smallest class containing the true model. This is the first such algorithm to achieve regret scaling similar to an oracle that knew the problem complexity. An interesting direction of future work is to derive regret bounds for the case when the dimension is a measure of complexity, that hold uniformly over all Σ , i.e., have no explicit dependence on d .

8.8 Proofs

8.8.1 Detailed Description of OFUL⁺

We now discuss the algorithm OFUL⁺. A variation of this was proposed in [67] in the context of model selection between linear and standard multi-armed bandits. As seen in the OFUL⁺ subroutine of Algorithm 11, we use $\tilde{r}_{i,t}$ to address the bias term in the observation, which we define shortly. The parameters b and β appears in the construction of the confidence set and the regret guarantee. Furthermore, assume that the algorithm OFUL⁺ is run for T rounds.

Let A_s be the arm index played at time instant s and $T_i(t)$ be the number of times we play arm i until time t . Hence $T_i(t) = \sum_{s=1}^t \mathbf{1}_{A_s = i}$. Also, let \hat{k} be the current estimate of k . Also define,

$$g_{i,t} = \frac{1}{T_i(t)} \sum_{s=1}^t g_{i;s} \mathbf{1}_{A_s = i}$$

With this, we have

$$\tilde{r}_{i,t} = g_{i,t} + \frac{1 + T_i(t)}{T_i^2(t)} \left(1 + 2 \log \frac{K(1 + T_i(t))^{1-2\beta}}{T_i^{1-2\beta}} \right) + b \frac{2d}{T_i(t)} \log \frac{1}{\beta} \quad (8.2)$$

In order to specify the confidence interval C_t , we first talk about the least squares estimate $\hat{\theta}_t$ first. Using the notation of [67], we define

$$\hat{\theta}_t = \left(\sum_{K+1:t} \mathbf{A}_{K+1:t} + I \right)^{-1} \sum_{K+1:t} \mathbf{A}_{K+1:t} \mathbf{G}_{K+1:t}$$

where $\mathbf{A}_{K+1:t}$ is a matrix with rows $\mathbf{A}_{K+1:K+1}, \dots, \mathbf{A}_{t:t}$ and $\mathbf{G}_{K+1:t} = [g_{A_{K+1:K+1}}, \dots, g_{A_{t:t}}]$. With this, the confidence interval is defined as

$$C_t = \left\{ \theta \in \mathbb{R}^d : \|\hat{\theta}_t - \theta\|_{\mathbf{A}_{K+1:t}} \leq \beta \right\} \quad (8.3)$$

and Lemma 2 of [67] shows that $\theta \in C_t$ with probability at least $1 - 4\beta$.

We now define the quantity $\mathcal{K}(b; t; T)$. Note that we track the dependence on the complexity

parameter k . We have

$$T_{\min}(; \mathcal{F}) = \frac{16}{2_{\min}} + \frac{8}{3_{\min}} \log \frac{2d\mathcal{F}}{S};$$

$$M(b; t) = b + \frac{d}{2} \log \left(1 + \frac{t}{d} \right) + \log \frac{1}{t}; \quad (8.4)$$

$$(b; t; \mathcal{F}) = \frac{10}{3} @ b + 2 + \frac{\sqrt{t}}{t} \frac{1}{1 + 2 \log \frac{2K\mathcal{F}}{A}}$$

$$4 \log \frac{2K\mathcal{F}}{t} + \frac{\sqrt{t}}{t} \frac{1}{t \log \frac{2K\mathcal{F}}{t} + \log^2 \frac{2K\mathcal{F}}{t}}; \quad (8.5)$$

$$K(b; t; \mathcal{F}) = \begin{cases} < M(b; t) + (b; t; \mathcal{F}) & \text{if } 1 < t < T_{\min}; \\ \frac{M(b; t)}{1 + \min_{t=2}} + \frac{(b; t; \mathcal{F})}{1 + \min_{t=2}} & \text{if } t > T_{\min}; \end{cases} \quad (8.6)$$

8.8.2 Proofs of the main results

In this section, we collect the proof of our main results. We start with the norm-based complexity measure.

8.8.3 Proof of Theorem 20

We first take Lemma 54 for granted and conclude the proof of Theorem 20 using the lemma. Suppose we play Algorithm 11 for N epochs. The cumulative regret is given by

$$R(T) = C_1(2 + K)k + \sum_{i=1}^N R(i; b_i)(T_i);$$

where $R(i; b_i)(T_i)$ is the cumulative regret of the $\text{OFUL}_i^+(b_i)$ in the i -th epoch. As seen (by tracking the dependence on k) in [67], the cumulative regret of $\text{OFUL}_i^+(b_i)$ scales linearly with b_i . Hence, we obtain

$$R(T) = \sum_{i=1}^N b_i R(i; 1)(T_i);$$

Using Lemma 1, we obtain, with probability at least $1 - \delta$,

$$R(T) = C_1(2 + K)k + (c_1 k + c_2) \sum_{i=1}^N R(i; 1)(T_i)$$

Theorem 3 of [67] gives,

$$R(i;1)(T_i) \leq C \left(\frac{\rho_{\overline{K}}}{K} + \frac{\rho_{\overline{d}}}{d} \right)^{\rho} \frac{1}{T_i} \log \frac{KT_i}{i} \quad (8.7)$$

with probability exceeding $1 - 5^{-i}$. With the doubling trick, we have

$$T_i = 2^{i-1} T_1; \quad i = \frac{1}{2^{i-1}}$$

Substituting, we obtain

$$R(i;1)(T_i) \leq C_1 \left(\frac{\rho_{\overline{K}}}{K} + \frac{\rho_{\overline{d}}}{d} \right)^{\rho} \frac{1}{T_i} \log \frac{KT_i}{i} = (2i-2) \log \frac{KT_1}{1}$$

with probability at least $1 - 5^{-i}$.

Using the above expression, we obtain

$$R(T) \leq C_1(2 + K)k + (C_2k + C_3) \sum_{i=1}^N \left(\frac{\rho_{\overline{K}}}{K} + \frac{\rho_{\overline{d}}}{d} \right)^{\rho} \frac{1}{T_i} (2i-2) \log \frac{KT_1}{1}$$

with probability

$$\begin{aligned} & 1 - 8^{-1} - 5^{-1} - 1 + \frac{1}{2} + \dots \text{N-th term} \\ & 1 - 8^{-1} - 5^{-1} - 1 + \frac{1}{2} + \dots \\ & = 1 - 8^{-1} - 10^{-1} \\ & = 1 - 18^{-1}; \end{aligned}$$

where the term 8^{-1} comes from Lemma 54. Also, from the doubling principle, we obtain

$$\sum_{i=1}^N 2^{i-1} T_1 = T \implies N = \log_2 \left(1 + \frac{T}{T_1} \right)$$

Using the above expression, we obtain

$$\begin{aligned}
 R(T) & \leq C_1(2 + K)k + k + (C_2k + k + C_3) \sum_{i=1}^N (\frac{\rho_{\bar{K}}}{K} + \frac{\rho_{\bar{d}}}{d}) \frac{\rho_{\bar{T}_i}}{T_i} (2i - 2) \log \frac{KT_1}{1} \\
 & \leq C_1(2 + K)k + k + 2(C_2k + k + C_3) (\frac{\rho_{\bar{K}}}{K} + \frac{\rho_{\bar{d}}}{d}) \log \frac{KT_1}{1} \sum_{i=1}^N \frac{\rho_{\bar{T}_i}}{T_i} \\
 & \leq C_1(2 + K)k + k + 2(C_2k + k + C_3) (\frac{\rho_{\bar{K}}}{K} + \frac{\rho_{\bar{d}}}{d}) \log \frac{KT_1}{1} N \sum_{i=1}^N \frac{\rho_{\bar{T}_i}}{T_i} \\
 & \leq C_1(2 + K)k + k + 2(C_2k + k + C_3) (\frac{\rho_{\bar{K}}}{K} + \frac{\rho_{\bar{d}}}{d}) \log \frac{KT_1}{1} \log \frac{T}{T_1} \sum_{i=1}^N \frac{\rho_{\bar{T}_i}}{T_i} \\
 & \leq C_1(2 + K)k + k + C(k + k + 1) (\frac{\rho_{\bar{K}}}{K} + \frac{\rho_{\bar{d}}}{d}) \log \frac{KT_1}{1} \log \frac{T}{T_1} \sum_{i=1}^N \frac{\rho_{\bar{T}_i}}{T_i};
 \end{aligned}$$

where the last inequality follows from the fact that

$$\begin{aligned}
 \sum_{i=1}^N \frac{\rho_{\bar{T}_i}}{T_i} & = \frac{\rho_{\bar{T}_N}}{T_N} \left(1 + \frac{1}{\rho_{\bar{T}_2}} + \frac{1}{2} + \dots \right) \text{N-th term} \\
 & = \frac{\rho_{\bar{T}_N}}{T_N} \left(1 + \frac{1}{\rho_{\bar{T}_2}} + \frac{1}{2} + \dots \right) \\
 & = \frac{\rho_{\bar{T}_2}}{\rho_{\bar{T}_2} - 1} \frac{\rho_{\bar{T}_N}}{T_N} \\
 & = \frac{\rho_{\bar{T}_2}}{\rho_{\bar{T}_2} - 1} \frac{\rho_{\bar{T}_N}}{T_N};
 \end{aligned}$$

The above regret bound holds with probability at least $1 - 18^{-1}$.

8.8.4 Proof of Lemma 54

Let us consider the i -th epoch, and let \hat{E}_i be the least square estimate of E at the end of epoch i . From the above section, the confidence interval at the end of epoch i , is given by

$$C_{E_i} = \left\{ \theta \in \mathbb{R}^d : \|\theta - \hat{E}_i\| \leq K_i(b_i; T_i; T_i) \right\}$$

where we play $\text{OFUL}_i^+(b_i)$ during the i -th epoch, and T_i is the number of total rounds in the i -th epoch. By choosing $T_1 > T_{\min}(\cdot; T)$, we ensure that $T_i \geq T_{\min}(\cdot; T_i)$. From equation (8.6), and ignoring the non-dominant terms, we obtain

$$K_i(b_i; T_i; T_i) = \frac{M_i(b_i; T_i)}{1 + \min_{T_i=2} T_i} + \frac{M_i(b_i; T_i; T_i)}{1 + \min_{T_i=2} T_i}.$$

with

$$M_i(b_i; T_i) = b_i + c_1 \rho_{-d} \log \frac{T_i}{d_i}$$

and

$$K_i(b_i; T_i; T_i) = 4b_i \rho_{-d} \log \frac{2KT_i}{i} + c_2 \rho_{-d} \log \frac{2KT_i}{i}$$

Substituting the values, considering the dominating terms, and for a sufficiently large T_i , we obtain

$$K_i(b_i; T_i; T_i) \approx \frac{7b_i \log \frac{2KT_i}{i}}{\rho_{-\frac{d}{\min T_i}}} + C \rho_{-\frac{d}{\min T_i}} \log \frac{2KT_i}{i}$$

$$\approx \frac{7b_i \log \frac{2KT}{i}}{\rho_{-\frac{d}{\min T_i}}} + C \rho_{-\frac{d}{\min T_i}} \log \frac{2KT_i}{i}$$

where C is an universal constant. From Lemma 2 of [67], we know that $\rho_{-d} \geq C_{E_i}$ with probability at least $1 - 4^{-i}$. Hence, we obtain

$$\hat{k}_{E_i} \leq k + 2K_i(b_i; T_i) \leq k + \frac{14b_i \log \frac{2KT_i}{i}}{\rho_{-\frac{d}{\min T_i}}} + 2C \rho_{-\frac{d}{\min T_i}} \log \frac{2KT_i}{i}$$

Recall from Algorithm 11 that at the end of the i -th epoch, we set the length $T_{i+1} = 2T_i$, and the estimate of k is set to

$$b_{i+1} = \max_{k \in \mathcal{K}} k$$

From the definition of C_{E_i} , we obtain

$$b_{i+1} = \hat{k}_{E_i} \leq k + K_i(b_i; T_i) \leq \frac{7b_i \log \frac{2KT_i}{i}}{\rho_{-\frac{d}{\min T_i}}} + C \rho_{-\frac{d}{\min T_i}} \log \frac{2KT_i}{i} :$$

Re-writing the above expression, with probability at least $1 - 4^{-i}$, we obtain

$$b_{i+1} \leq k + \frac{7 \log \frac{2KT_i}{i}}{\rho_{-\frac{d}{\min T_i}}} \frac{b_i}{\rho_{-\frac{d}{T_i}}} + \frac{C \log \frac{2KT_i}{i}}{\rho_{-\frac{d}{\min T_i}}} \frac{\rho_{-d}}{\rho_{-\frac{d}{T_i}}}$$

$$\leq k + ip \frac{b_i}{\rho_{-\frac{d}{T_i}}} + iq \frac{\rho_{-d}}{\rho_{-\frac{d}{T_i}}}$$

$$\leq k + ip \frac{b_i}{2^{\frac{i-1}{2}} \rho_{-\frac{d}{T_1}}} + iq \frac{\rho_{-d}}{2^{\frac{i-1}{2}} \rho_{-\frac{d}{T_1}}} \tag{8.8}$$

where we use the fact that $i = \frac{1}{2^{i-1}}$ and $T_i = 2^{\frac{i-1}{2}} T_1$, and we have

$$p = \frac{14 \log \frac{2KT_1}{\rho_{\min}}}{\rho_{\min}} A$$

and

$$q = \frac{2C \log \frac{2KT_1}{\rho_{\min}}}{\rho_{\min}} A :$$

Hence, we obtain

$$b_{i+1} = b_i + k + iq \frac{\rho_{\min}}{2^{\frac{i-1}{2}} T_1} + ip \frac{1}{2^{\frac{i-1}{2}} T_1} b_i :$$

From the construction of b_i , we have $b_i \leq k + k$. Hence provided

$$T_1 \geq \frac{\rho_{\min}^2}{2^{i-1}}$$

which is equivalent to the condition $T_1 \geq 3\rho_{\min}^2$ (using the fact that $\frac{\rho_{\min}^2}{2^{i-1}} \geq 3$ for $i \geq 1$), we obtain

$$b_{i+1} = b_i + ip \frac{1}{2^{\frac{i-1}{2}} T_1} + k + iq \frac{\rho_{\min}}{2^{\frac{i-1}{2}} T_1} :$$

From the above expression, we obtain

$$\sup_i b_i < 1 :$$

with probability

$$1 - 4 \sum_{i=1}^{\infty} \frac{1}{2^i} + \frac{1}{4} + \dots = 1 - 8 \sum_{i=1}^{\infty} \frac{1}{2^i} :$$

Invoking Equation (8.8) and using the above fact in conjunction yield (with probability at least $1 - 8 \sum_{i=1}^{\infty} \frac{1}{2^i}$)

$$\lim_{i \rightarrow \infty} b_i = k + k :$$

However, from construction $b_i \leq k + k$. Using this, along with the above equation, we obtain

$$\lim_{i \rightarrow \infty} b_i = k + k :$$

with probability exceeding $1 - 8 \sum_{i=1}^{\infty} \frac{1}{2^i}$. So, the sequence (b_0, b_1, \dots) converges to $k + k$ with high probability, and hence our successive refinement algorithm is consistent.

Rate of Convergence: Since

$$b_i - b_{i-1} = \mathcal{O}\left(\frac{1}{2^i}\right); \quad (8.9)$$

with probability greater than $1 - 4^{-i}$, the rate of convergence of the sequence $\sum_{i=0}^T b_i g_{i=0}^1$ is exponential in the number of epochs.

Uniform upper bound on b_i for all i : We now compute a uniform upper bound on b_i for all i .

Consider the sequence $\frac{1}{2^{i/2}}$, and let t_j denote the j -th term of the sequence. It is easy to check that $\sum_{i=1}^{\infty} t_i = 1.5$, and that the sequence $\sum_{i=1}^{\infty} t_i g_{i=1}^1$ is convergent. With this new notation, we have

$$b_2 = k + t_1 \frac{pb_1}{T_1} + t_1 \frac{g_{i=1}^1 d}{T_1};$$

with probability exceeding $1 - 4^{-1}$. Similarly, for b_3 , we have

$$b_3 = k + t_2 \frac{pb_1}{T_1} + t_2 \frac{g_{i=1}^1 d}{T_1} + t_2 \frac{p}{T_1} \left(k + t_1 \frac{pb_1}{T_1} + t_1 \frac{g_{i=1}^1 d}{T_1} \right);$$

with probability at least $1 - 4^{-1} - 4^{-2} = 1 - 6^{-1}$. Similarly, we write expressions for b_4, b_5, \dots . Now, provided $T_1 \geq C_1 (\max\{p, q, g_{i=1}^1\})^2 d$, where $C_1 > 9$ is a sufficiently large constant, the expression for b_i can be upper-bounded as

$$b_i \leq (c_1 k + c_2); \quad (8.10)$$

with probability

$$\begin{aligned} & 1 - 4^{-1} - 4^{-2} - \dots \text{ upto } i\text{-th term} \\ & 1 - 4^{-1} - 4^{-2} - 4^{-3} - \dots \\ & = 1 - 8^{-1}; \end{aligned}$$

Here c_1 and c_2 are constants, and are obtained from summing an infinite geometric series with decaying step size. We also use the fact that $b_1 \leq 1$, and the fact that $\frac{1}{2^i} = \frac{1}{2^{i-1}}$.

8.8.5 Proof of Theorem 21

We shall need the following lemma from [249], on the behaviour of linear regression estimates.

Lemma 55. *If $M \geq d$ and satisfies $M = O\left(\frac{1}{\alpha} + d \ln \frac{1}{\alpha}\right)$, and $\hat{b}^{(M)}$ is the least-squares estimate of β , using the M random samples for feature, where each feature is chosen uniformly and independently on the unit sphere in d dimensions, then with probability $1 - \alpha$, $\hat{b}^{(M)}$ is well defined (the least squares regression has a unique solution). Furthermore,*

$$\mathbb{P}[\|\hat{b}^{(M)} - \beta\|_2 \leq \alpha] \geq 1 - \alpha.$$

We shall now apply the theorem as follows. Denote by \hat{b}_i to be the estimate of β at the beginning of any phase i , using all the samples from random explorations in all phases less than or equal to $i - 1$.

Remark 37. The choice $T_0 := O(d^2 \ln^2 \frac{1}{\alpha})$ in Equation (8.1) is chosen such that from Lemma 56, we have that

$$\mathbb{P}[\|\hat{b}^{(d^2 T_0 e)} - \beta\|_2 \leq \frac{1}{2}] \geq \frac{1}{2}$$

Lemma 56. *Suppose $T_0 = O(d^2 \ln^2 \frac{1}{\alpha})$ is set according to Equation (8.1). Then, for all phases $i \geq 4$,*

$$\mathbb{P}[\|\hat{b}_i - \beta\|_2 \leq \frac{1}{2^i}] \geq \frac{1}{2^i} \tag{8.11}$$

where \hat{b}_i is the estimate of β obtained by solving the least squares estimate using all random exploration samples until the beginning of phase i .

Proof. The above lemma follows directly from Lemma 55. Lemma 55 gives that if \hat{b}_i is formed by solving the least squares estimate with at-least $M_i := O((4^i + d) \ln \frac{1}{\alpha})$ samples, then the guarantee in Equation (8.11) holds. However, as $T_0 = O(d^2 \ln^2 \frac{1}{\alpha})$, we have naturally that $M_i \leq 4^i T_0$. The proof is concluded if we show that at the beginning of phase $i \geq 4$, the total number of random explorations performed by the algorithm exceeds $i 4^i d T_0 e$. Notice that at the beginning of any phase $i \geq 4$, the total number of random explorations that have been performed is

$$\sum_{j=0}^{i-1} 5^j d T_0 e = d T_0 e \frac{5^i - 1}{4};$$

$$i 4^i d T_0 e;$$

where the last inequality holds for all $i \geq 4$. □

The following corollary follows from a straightforward union bound.

Corollary 6.

$$\mathbb{P} \left(\bigcup_{i=4}^n \{j_i^b \neq j_i^o\} \right) \leq \sum_{i=4}^n \frac{1}{2^i} \leq \frac{1}{2^3} = \frac{1}{8}.$$

Proof. This follows from a simple union bound as follows.

$$\begin{aligned} \mathbb{P} \left(\bigcup_{i=4}^n \{j_i^b \neq j_i^o\} \right) &= \mathbb{P} \left[\bigcup_{i=4}^n \{j_i^b \neq j_i^o\} \right] \\ &\leq \sum_{i=4}^n \mathbb{P} \left(\{j_i^b \neq j_i^o\} \right) \\ &\leq \sum_{i=4}^n \frac{1}{2^i} \\ &\leq \frac{1}{2^3} \\ &= \frac{1}{8}. \end{aligned}$$

□

We are now ready to conclude the proof of Theorem 21.

Proof of Theorem 21. We know from Corollary 6, that with probability at-least $1 - \frac{1}{8}$, for all phases $i \geq 4$, we have $j_i^b = j_i^o$. Call this event E . Now, consider the phase $i(\cdot) := \max\{4; \log_2 \frac{T}{d}\}$. Now, when event E holds, then for all phases $i \geq i(\cdot)$, D_i is the correct set of d non-zero coordinates of θ . Thus, with probability at-least $1 - \frac{1}{8}$, the total regret upto time T can be upper bounded as follows

$$\begin{aligned} R_T &\leq \sum_{j=0}^{i(\cdot)-1} 25^j T_0 + 5^j d^j \frac{T}{T_0} e + \sum_{j=i(\cdot)}^{\log_{25} \frac{T}{T_0}} \text{Regret}(\text{OFUL}(1; j; 25^j T_0)) \\ &\quad + \sum_{j=i(\cdot)}^{\log_{25} \frac{T}{T_0}} 5^j d^j \frac{T}{T_0} e. \end{aligned} \tag{8.12}$$

The term $\text{Regret}(\text{OFUL}(L; \cdot; T))$ denotes the regret of the OFUL algorithm [66], when run with parameters $L \geq \mathbb{R}_+$, such that $k \leq L$, and $\delta \in (0; 1)$ denotes the probability slack and T is the time horizon. Equation (8.12) follows, since the total number of phases is at-most $\log_{25} \frac{T}{T_0}$. Standard result from [66] give us that, with probability at-least $1 - \delta$, we have

$$\text{Regret}(\text{OFUL}(1; \cdot; T)) \leq 4 \frac{\sum_{i=0}^{\log_{25} \frac{T}{T_0}} T d \ln \left(1 + \frac{T}{d}\right)}{1 + \frac{\sum_{i=0}^{\log_{25} \frac{T}{T_0}} T d \ln \left(1 + \frac{T}{d}\right)}{2 \ln \frac{1}{\delta} + d \ln \left(1 + \frac{T}{d}\right)}} :$$

Thus, we know that with probability at-least $1 - \delta$, for all phases $i \in [0; \log_{25} \frac{T}{T_0})$, the regret in the exploration phase satisfies

$$\text{Regret}(\text{OFUL}(1; \cdot; 25^i T_0)) \leq 4 \frac{\sum_{i=0}^{\log_{25} \frac{T}{T_0}} d 25^i T_0 \ln \left(1 + \frac{25^i T_0}{d}\right)}{1 + \frac{\sum_{i=0}^{\log_{25} \frac{T}{T_0}} d 25^i T_0 \ln \left(1 + \frac{25^i T_0}{d}\right)}{2 \ln \frac{1}{\delta} + d \ln \left(1 + \frac{25^i T_0}{d}\right)}} : \quad (8.13)$$

In particular, for all phases $i \in [\log_{25} \frac{T}{T_0}; \log_{25} \frac{T}{T_0}]$, with probability at-least $1 - \delta$, we have

$$\begin{aligned} \text{Regret}(\text{OFUL}(1; \cdot; 25^i T_0)) &\leq 4 \frac{\sum_{i=0}^{\log_{25} \frac{T}{T_0}} d 25^i T_0 \ln \left(1 + \frac{25^i T_0}{d}\right)}{1 + \frac{\sum_{i=0}^{\log_{25} \frac{T}{T_0}} d 25^i T_0 \ln \left(1 + \frac{25^i T_0}{d}\right)}{2 \ln \frac{1}{\delta} + d \ln \left(1 + \frac{25^i T_0}{d}\right)}} ; \\ &= C(T; \cdot; d) \frac{\sum_{i=0}^{\log_{25} \frac{T}{T_0}} d 25^i T_0 \ln \left(1 + \frac{25^i T_0}{d}\right)}{2 \ln \frac{1}{\delta} + d \ln \left(1 + \frac{25^i T_0}{d}\right)} ; \end{aligned} \quad (8.14)$$

where the constant captures all the terms that only depend on T , δ and d . We can write that constant as

$$C(T; \cdot; d) = 4 \frac{\sum_{i=0}^{\log_{25} \frac{T}{T_0}} d 25^i T_0 \ln \left(1 + \frac{25^i T_0}{d}\right)}{2 \ln \frac{1}{\delta} + d \ln \left(1 + \frac{25^i T_0}{d}\right)} :$$

Equation (8.14) follows, by substituting $i = \log_{25} \frac{T}{T_0}$ in all terms except the first 25^i term in Equation (8.13). As Equations (8.14) and (8.12) each hold with probability at-least $1 - \delta$, we can

combine them to get that with probability at-least $1 - \epsilon$,

$$\begin{aligned}
 R_T &\leq 2T_0 25^{i^*} + \sum_{j=0}^{\log_{25} \frac{T}{T_0} + 1} C(T; d) \rho \frac{1}{25^j T_0} + 25^j d \rho \frac{1}{T_0} e^{5 \log_{25} \frac{T}{T_0}}; \\
 &\leq 2T_0 25^{i^*} + 25^{\rho} \bar{T} + C(T; d) \sum_{j=0}^{\log_{25} \frac{T}{T_0} + 1} \rho \frac{1}{25^j T_0}; \\
 &\stackrel{(a)}{\leq} 50T_0 \frac{2}{4.65} + 25^{\rho} \bar{T} + 25^{\rho} \bar{T} C(T; d); \\
 &= O \left(\frac{d^2}{4.65} \ln^2 \frac{1}{\epsilon} + \Theta \left(d \sqrt{T} \ln \frac{1}{\epsilon} \right) \right);
 \end{aligned}$$

Step (a) follows from $25^{i^*} \leq 2^{4.65}$.

□

8.8.6 ALB-Dim for Stochastic Contextual Bandits with Finite Arms

8.8.7 ALB-Dim Algorithm for the Finite Armed Case

The algorithm given in Algorithm 13 is identical to the earlier Algorithm 12, except in Line 8, this algorithm uses SupLinRel [246] as opposed to OFUL used in the previous algorithm. In practice, one could also use LinUCB [64] in place of SupLinRel. However, we choose to present the theoretical argument using SupLinRel, as unlike LinUCB, has an explicit closed form regret bound [246]. The pseudocode is provided in Algorithm 13.

In phase $i \geq N$, the SupLinRel algorithm is instantiated with input parameter $25^i T_0$ denoting the time horizon, slack parameter $\gamma \in (0, 1)$, dimension d_{M_i} and feature scaling $b(\cdot)$. We explain the role of these input parameters. The dimension ensures that SupLinRel plays from the restricted dimension d_{M_i} . The feature scaling implies that when a context $x \in X$ is presented to the algorithm, the set of K feature vectors, each of which is d_{M_i} dimensional are $\frac{d_{M_i}(x;1)}{b(\cdot)}; \dots; \frac{d_{M_i}(x;K)}{b(\cdot)}$. The constant $b(\cdot) := O \left(\sqrt{\log \frac{TK}{\epsilon}} \right)$ is chosen such that

$$\mathbb{P} \left(\sup_{t \in [0; T]; a \in A} \sum_{k=1}^K \langle M(x_t; a), k \rangle \leq b(\cdot) \sqrt{4} \right) \geq 1 - \epsilon;$$

Such a constant exists since $(x_t)_{t \in [0; T]}$ are i.i.d. and $M(x; a)$ is a sub-gaussian random variable with parameter $4 \sqrt{2}$, for all $a \in A$. Similar idea was used in [70].

Algorithm 13: Adaptive Linear Bandit (Dimension) with Finitely Many arms

- 1: **Input:** Initial Phase length T_0 and slack $\epsilon > 0$.
 - 2: $b_0 = \mathbf{1}, T_1 = 0$
 - 3: **for** Each epoch $i \geq 1; 2; g$ **do**
 - 4: $T_i = 25^i T_0, \quad \alpha_i = \frac{1}{2^i}, \quad \beta_i = \frac{\epsilon}{2^i}$
 - 5: $D_i := \{f_j : j \in [K]\} \cdot \frac{\alpha_i}{2} g$
 - 6: $M_i := \inf_{f \in M} \max_{g \in D_i} f \cdot g$.
 - 7: **for** Times $t \geq \lceil T_{i-1} \rceil + 1; \dots; T_i g$ **do**
 - 8: Play according to SupLinRel of [246] with time horizon of $25^i T_0$ with parameters $\alpha_i \in (0; 1)$, dimension d_{M_i} and feature scaling $b(\cdot) := O \left(\frac{1}{\log \frac{TK}{\epsilon}} \right)$.
 - 9: **end for**
 - 10: **for** Times $t \geq \lceil T_i \rceil + 1; \dots; T_i + 5^i \beta_i T_0 g$ **do**
 - 11: Play an arm from the action set A chosen uniformly and independently at random.
 - 12: **end for**
 - 13: $R_i \in \mathbb{R}^{S_i \times d}$ with each row being the arm played during all random explorations in the past.
 - 14: $y_i \in \mathbb{R}^{S_i}$ with i -th entry being the observed reward at the i -th random exploration in the past
 - 15: $b_{i+1} = \left(\frac{T_i}{i} \right)^{-1} y_i$, is a d dimensional vector
 - 16: **end for**
-

8.8.8 Regret Guarantee for Algorithm 13

In order to specify a regret guarantee, we will need to specify the value of T_0 . We do so as before. For any N , denote by $\lambda_{\max}^{(N)}$ and $\lambda_{\min}^{(N)}$ to be the maximum and minimum eigen values of the following matrix: $\Sigma^N := \mathbb{E} \frac{1}{K} \sum_{j=1}^K \sum_{t=1}^N M(x_t; j) M(x_t; j)^T$, where the expectation is with respect to $(x_t)_{t \in [T]}$ which is an i.i.d. sequence with distribution D . First, given the distribution of $x \sim D$, one can (in principle) compute $\lambda_{\max}^{(N)}$ and $\lambda_{\min}^{(N)}$ for any $N \geq 1$. Furthermore, from the assumption on D , $\lambda_{\min}^{(N)} = \Theta \left(\frac{1}{d} \right) > 0$ for all $N \geq 1$. Choose $T_0 \geq \mathbb{N}$ to be the smallest integer such that

$$\frac{1}{T_0} b(\cdot) \max \left\{ \frac{32}{\left(\frac{\lambda_{\min}^{(T_0 \epsilon)} \right)^2} \ln(2d)}, \frac{4}{3} \frac{\left(6 \frac{\lambda_{\max}^{(T_0 \epsilon)}}{\lambda_{\min}^{(T_0 \epsilon)}} + \frac{\lambda_{\max}^{(T_0 \epsilon)}}{\lambda_{\min}^{(T_0 \epsilon)}} \right) \left(d + \frac{\lambda_{\max}^{(T_0 \epsilon)}}{\lambda_{\min}^{(T_0 \epsilon)}} \right)}{\left(\frac{\lambda_{\min}^{(T_0 \epsilon)}}{\lambda_{\max}^{(T_0 \epsilon)}} \right)^2} \ln(2d) \right\} \leq \epsilon \quad (8.15)$$

As before, it is easy to see that

$$T_0 = O \left(d^2 \ln^2 \frac{1}{\epsilon} \ln \frac{TK}{\epsilon} \right)$$

Furthermore, following the same reasoning as in Lemmas 56 and 55, one can verify that for all $i \geq 4, \lambda_{\min}^{(T_i \epsilon)} \geq \frac{1}{2^i} \lambda_{\min}^{(T_0 \epsilon)}$.

Theorem 22. Suppose Algorithm 13 is run with input parameters $\gamma \in (0; 1)$, and T_0 as given in Equation (8.15), then with probability at-least $1 - \gamma$, the regret after a total of T arm-pulls satisfies

$$R_T \leq 2T_0 \max \left\{ 25^4; \frac{2}{4.65} \right\} + 308(1 + \ln(2KT \ln T))^{3=2} \frac{P}{Td_m} + 100 \frac{P}{T}.$$

The parameter $\gamma > 0$ is the minimum magnitude of the non-zero coordinate of θ , i.e., $\gamma = \min_j \theta_j : \theta_j \neq 0$.

In order to parse the above theorem, the following corollary is presented.

Corollary 7. Suppose Algorithm 13 is run with input parameters $\gamma \in (0; 1)$, and $T_0 = \Theta \left(\frac{d^2 \ln^2 \frac{1}{\gamma}}{\gamma} \right)$ given in Equation (8.15), then with probability at-least $1 - \gamma$, the regret after T times satisfies

$$R_T \leq O \left(\frac{d^2}{4.65} \ln^2(d) \right) \ln \left(\frac{TK}{\gamma} \right) + \Theta \left(\frac{P}{Td_m} \right).$$

Proof of Theorem 22. The proof proceeds identical to that of Theorem 21. Observe from Lemmas 55 and 56, that the choice of T_0 is such that for all phases $i \geq 1$, the estimate $P \leq k_{i-1} \leq k_1 \cdot 2^{-i} \leq \frac{1}{2}$. Thus, from an union bound, we can conclude that

$$P \leq \sum_{i=1}^h k_{i-1} \leq k_1 \sum_{i=1}^h 2^{-i} \leq \frac{1}{4}.$$

Thus at this stage, with probability at-least $1 - \gamma$, the following events holds.

- $\sup_{t \in [0; T]; a \in A} |k^M(x_t; a) - k_2| \leq \gamma$
- $k_{i-1} \leq k_1 \cdot 2^{-i}$, for all $i \geq 4$.

Call these events as E . As before, let $\gamma > 0$ be the smallest value of the non-zero coordinate of θ . Denote by the phase $i(\gamma) := \max \{4; \log_2 \frac{2}{\gamma}\}$. Thus, under the event E , for all phases $i \geq i(\gamma)$, the dimension $d_{M_i} = d_m$, i.e., the SupLinRel is run with the correct set of dimensions.

It thus remains to bound the error by summing over the phases, which is done identical to that in Theorem 21. With probability, at-least $1 - \gamma \sum_{i=4}^h 2^{-i} \geq 1 - \gamma$,

$$R_T \leq \sum_{j=0}^{i(\gamma)-1} 25^j T_0 + 5^j \frac{P}{T_0} + \sum_{j=i(\gamma)}^{\log_{25} \frac{T}{T_0}} \text{Regret}(\text{SupLinRel})(25^j T_0; i; d_{M_i}; \gamma) + \sum_{j=i(\gamma)}^{\log_{25} \frac{T}{T_0}} 5^j \frac{P}{T_0}.$$

where $\text{Regret}(\text{SupLinRel})(25^j T_0; i; d_{M_i}; b(\cdot)) \leq 44(1 + \ln(2K25^j T_0 \ln 25^j T_0))^{3-2} \mathbb{P} \frac{1}{25^j T_0 d_{M_i}} + 2^{\mathbb{P}} \frac{1}{25^j T_0}$. This expression follows from Theorem 6 in [246]. We now use this to bound each of the three terms in the display above. Notice from straightforward calculations that the first term is bounded by $2T_0 25^{i(\cdot)}$ and the last term is bounded above by $25 d_{M_i}^{\mathbb{P}} T_0 e^{5 \log_{25} \frac{T}{T_0}}$ respectively. We now bound the middle term as

$$\sum_{j=i(\cdot)}^{\log_{25} \frac{T}{T_0}} \text{Reg}(\text{SupLinRel})(25^j T_0; i; d_{M_i}; b(\cdot)) \leq \sum_{j=i(\cdot)}^{\log_{25} \frac{T}{T_0}} 44(1 + \ln(2K25^j T_0 \ln 25^j T_0))^{3-2} \mathbb{P} \frac{1}{25^j T_0 d_{M_i}} + 2^{\mathbb{P}} \frac{1}{25^j T_0}.$$

The first summation can be bounded as

$$\begin{aligned} & \sum_{j=i(\cdot)}^{\log_{25} \frac{T}{T_0}} 44(1 + \ln(2K25^j T_0 \ln 25^j T_0))^{3-2} \mathbb{P} \frac{1}{25^j T_0 d_{M_i}} \\ & \sum_{j=i(\cdot)}^{\log_{25} \frac{T}{T_0}} 44(1 + \ln(2KT \ln T))^{3-2} \mathbb{P} \frac{1}{25^j T_0 d_{M_i}} \\ & 44(1 + \ln(2KT \ln T))^{3-2} 75^{\log_{25} \frac{T}{T_0}} \mathbb{P} \frac{1}{T_0 d_{M_i}} \\ & = 308(1 + \ln(2KT \ln T))^{3-2} \mathbb{P} \frac{1}{T d_{M_i}} \end{aligned}$$

and the second by

$$\sum_{j=i(\cdot)}^{\log_{25} \frac{T}{T_0}} 2^{\mathbb{P}} \frac{1}{25^j T_0} \leq 50^{\mathbb{P}} \frac{1}{T}.$$

Thus, with probability at-least $1 - \delta$, the regret of Algorithm 13 satisfies

$$R_T \leq 2T_0 25^{i(\cdot)} + 308(1 + \ln(2KT \ln T))^{3-2} \mathbb{P} \frac{1}{T d_m} + 100^{\mathbb{P}} \frac{1}{T};$$

where $i(\cdot) := \max\{4; \log_2 \frac{T}{25}\}$. Thus,

$$R_T \leq 2T_0 \max \left\{ 25^4; \frac{2}{4.65} \right\} + 308(1 + \ln(2KT \ln T))^{3-2\rho} \frac{\rho}{Td_m} + 100^{\rho} T;$$

as $25 \leq 2^{4.65}$

□

Chapter 9

Model Selection for Contextual Bandits Beyond Linear Structure

In this chapter, we focus on a provable model selection guarantee for the (generic) stochastic contextual bandit problem. The results of Chapter 8 holds only for linear contextual bandits. The linear parameterization is quite restrictive. here, we remove the linear assumption and solve a non parametric problem. In particular we assume that we are given a set of M nested function classes and the reward function lies in one of them. Via careful exploration, we show that with high probability the correct model class identification is possible with minimal cost (an additive constant in the regret term). We first discuss the contextual model selection problem.

9.1 Problem formulation

Let A be the set of K actions, and let X be the set of d dimensional contexts. At time t , Nature picks (x_t, r_t) in an i.i.d fashion, where $x_t \in X$ and a context dependent $r_t \in [0; 1]^K$. Upon observing the context, the agent takes action $a_t \in A$, and obtains the reward of $r_t(a_t)$. Note that, the reward $r_t(a_t)$ depends on the context x_t and the action a_t . Furthermore, we have the following realizability assumption:

Assumption 21. (Realizability:) *There exists a predictor $f \in F$, such that $E[r_t(a)|x] = f(x; a)$, for all x and a .*

Note that the realizability assumption is standard in the literature ([70, 71]. Also, in the contextual bandit literature ([71, 72]) it is assumed that the true regression function f^* is unknown, but we know the function class F where it belongs. Hence, we pay some price (denoted by the regret) for this. To set up notation, for any $f \in F$, we define a policy induced by the function, $\pi_f(x) = \operatorname{argmin}_{a \in A} f(x; a)$. So, we need to compete with the policy induced by the true regressor

f . We define the regret over T rounds as the following:

$$R(T) = \sum_{t=1}^T [r_t(f(x_t)) - r_t(a_t)]:$$

[71] proposed and analyzed a contextual bandit algorithm, namely FALCON (stands for FAst Least-squares-regression-oracle CONtextual bandits), which gives provable guarantees for contextual bandits beyond linear structure. FALCON is an epoch based algorithm, and depends only on an *offline regression oracle*, which outputs an estimate \hat{f} of the regression function f at the beginning of each epoch. Using a randomization scheme, that depends on the inverse gap with respect to the estimate of the best action, it is showed that under the realizability assumption, with probability $1 - \epsilon$, FALCON yields a regret of

$$R(T) = O\left(\frac{1}{\epsilon} \sqrt{KT \log(KT)}\right):$$

Although the above result makes sense only for the finite F , an extension to the infinite F is possible and was addressed in the same paper (see [71]).

We study the problem of model selection for contextual bandits. Hence, in contrast to the standard setting we do not know F . Instead, we are given a nested class of M function classes, $F_1 \subset F_2 \subset \dots \subset F_M$. The smallest function class where the true regressor lies is denoted by F_d , where $d \in [M]$. Hence, the regret of the contextual bandit algorithm should depend on F_d . However, we do not know d , and our goal is to propose adaptive algorithms such that the regret depends on the *actual* problem complexity F_d . First, let us rewrite the realizability assumption with the nested hypothesis class feature.

Assumption 22. *There exists a predictor $f_d \in F_d$, such that $E[r_t(a)|x] = f_d(x; a)$, for all x and a . Furthermore, the conditional variance of $r_t(\cdot)$ is bounded, i.e., given $x \in X$,*

$$E[r_t(a) - f_d(x; a)]^2 = \sigma^2 < 1:$$

for all $a \in A$.

Note that, the bounded second moment is also quite mild and is equivalent conditions appear in the stochastic optimization (SGD) literature.

We also have a separability condition as given in the following assumption. As we will see subsequently (in Section 9.4.1, this is equivalent to the condition on the minimum non-zero value of the optimal parameter in the linear contextual bandit setting.

Assumption 23. *For any $f \in F_{d-1}$, we have*

$$\inf_f E_{x;a}[f(x; a) - f_d(x; a)]^2 \geq \gamma;$$

for all pairs $(x; a) \in X \times A$, where $\gamma > 0$ is the minimum separation across the function classes.

Note that separability condition is quite standard in statistics, specially in the area of clustering and latent variable modelling (see [99, 250]). Model selection without separability condition is kept as an interesting future work.

Algorithm 14: Adaptive Contextual Bandits (ACB)

- 1: **Input:** epochs $0 = t_0 < t_1 < t_2 < \dots$; confidence parameter δ , Function classes F_1, F_2, \dots, F_M , threshold ϵ
 - 2: **for** epoch $m = 1; 2; \dots; M$ **do**
 - 3: $n_m = 2^m$
 - 4: **for** function classes $j = 1; 2; \dots; M$ **do**
 - 5: Compute $\hat{f}_j^m = \operatorname{argmin}_{f \in F_j} \sum_{t=t_{m-1}+1}^{t_m-1} (f(x_t; a_t) - r_t(a_t))^2$ via offline regression oracle
 - 6: Construct $T_j^m = \frac{1}{2^{m-2}} \sum_{t=t_{m-1}+1}^{t_m-1} (\hat{f}_j^m(x_t; a_t) - r_t(a_t))^2$
 - 7: **end for**
 - 8: **Model Selection:** Find the minimum index $j^* \in [M]$ such that $T_{j^*}^m \leq \epsilon$. Let this class be denoted by F_{j^*} .
 - 9: Set learning rate $\eta_m = \frac{\epsilon}{K \sum_{j=1}^M \log(jF_j) \sum_{m=1}^M \eta_m}$
 - 10: **for** round $t = t_{m-1} + 1; \dots; t_m$ **do**
 - 11: Observe context $x_t \in X$
 - 12: Compute $\hat{f}^m(a)$ for all action $a \in A$, set $\hat{a}_t = \operatorname{argmax}_{a \in A} \hat{f}^m(a)$
 - 13: Define $p_t(a) = \frac{1}{K + \sum_{m=1}^M (\hat{f}^m(x_t; \hat{a}_t) - \hat{f}^m(x_t; a))}$ $\forall a \notin \hat{a}_t$, $p_t(\hat{a}_t) = 1 - \sum_{a \notin \hat{a}_t} p_t(a)$.
 - 14: Sample $a_t \sim p_t(\cdot)$ and observe reward $r_t(a_t)$.
 - 15: **end for**
 - 16: **end for**
-

9.2 Algorithm—Adaptive Contextual Bandits (ACB)

In this section, we provide a novel model selection algorithm that use successive refinements over epochs. We use the FALCON algorithm of [71], and add a model selection phase at the beginning of each epoch. In other words, over multiple epochs, we successively refine our estimates of the *proper* model class where the true regressor function f^* lies. The details are provided in Algorithm 14.

We assume that epochs have doubling epoch length. Let t_0, t_1, \dots be epoch instances, with $t_0 = 0$, and $t_m = 2^m$. Before the beginning of the m -th epoch, using all the data of the $m-1$ -th epoch, we add a model selection module, as shown in Algorithm 14.

Note that we feed the samples of the $m-1$ -th epoch to the offline regression oracle. Moreover, we split the samples in 2 equal halves. We use the first half to compute the regression estimate \hat{f}_j^m , and then use the rest to construct T_j^m . We do not use the same set of samples to remove any dependence issues with \hat{f}_j^m and the samples $(f(x_t; a_t), r_t(a_t))_{t=t_{m-1}+1}^{t_m}$.

We compare T_j^m to a pre-calculated threshold, and pick the model class where T_j^m falls below such threshold (with smallest j , see Algorithm 23). After selecting the regressor function, the randomization follows exactly that of FALCON, i.e., we follow the inverse gap probability distribution $p_t(\cdot)$, and sample action $a_t \sim p_t(\cdot)$ and henceforth observe reward $r_t(a_t)$.

9.2.1 Regret Guarantee

We now analyze the performance of the model selection procedure of Algorithm 14. We have the doubling epochs, i.e., $m = 2^m$. Without loss of generality, we simply assume $\epsilon_1 = 2$. Also, assume that we are at the beginning of epoch m , and hence we have the samples from epoch $m - 1$. So, we have total of 2^{m-1} samples, out of which, we use 2^{m-2} to construct the regression functions and the rest 2^{m-2} to obtain the testing function T_j^m . Furthermore, we want the model selection procedure to succeed with probability at least $1 - \epsilon = 2^{-m}$, since we want a guarantee that holds for all m , and a simple application of the union bound yields that. We have the following Lemma.

Lemma 57. *Suppose we run Algorithm 14 with the threshold $\epsilon = \epsilon^2 + c_0$, where c_0 is a universal (small) constant. Then, provided*

$$2^m \geq \max_f \frac{\log(1 - \epsilon)}{c_0^2}; \frac{1}{c_0^2} \max_f \log(j F_M j = \epsilon); \log(1 - \epsilon) \geq \epsilon$$

and $\epsilon > c_0$, Algorithm 14 identifies the correct model class F_d , with probability exceeding

$$1 - \sum_{m=1}^M \epsilon \geq 1 - 6M \epsilon$$

With the above lemma, we obtain the following regret bound for Algorithm 14.

Theorem 23. *Suppose Assumptions 2 and 3 hold and $\epsilon > c_0$. Then with probability at least $1 - 6M \epsilon$, running Algorithm 14 for T iterations yield*

$$R(T) \leq C \max_f \frac{\log(1 - \epsilon)}{c_0^2}; \frac{1}{c_0^2} \max_f \log(j F_M j = \epsilon); \log(1 - \epsilon) \geq \epsilon + O\left(\frac{1}{KT \log(j F_d j T = \epsilon)}\right)$$

Remark 38. The first term of the regret does not scale with T . Hence, the regret scaling (with respect to T) is $O\left(\frac{1}{KT \log(j F_d j T = \epsilon)}\right)$, with high probability, which is exactly the model selection guarantee.

Remark 39. The first term can be interpreted as the cost of model selection. Hence, the model selection procedure only adds an additive constant to the regret performance of FALCON.

Remark 40. Note that the first term is a problem dependent quantity and implies the complexity of the problem. If ϵ is small, the model selection problem is hard, and the regret will be worse. Note that this is reflected via $\epsilon = \epsilon^2$ dependence.

Remark 41. Note that the constant c_0 in the threshold can be chosen as a small enough constant, and so the condition on the gap, given by $\epsilon > c_0$ is a mild condition. Since we do not know the gap ϵ , from an implementation point of view, the value of c_0 should be kept sufficiently small. Of course, this results in a cost of $O(1/c_0^2)$ in the constant (independent of T) term in the regret.

Remark 42. Note that if an oracle provides an estimate of the gap ϵ , one can avoid the condition $\epsilon > c_0$ by choosing the threshold ϵ as a function of ϵ .

Algorithm 15: ETC for Model Selection for Contextual Bandits

- 1: **Input:** Action set A , Function classes F_1, F_2, \dots, F_M , time horizon T , confidence parameter δ , threshold ϵ
 - 2: **Explore:**
 - 3: **for** $t = 1; 2; \dots; C \rho_{\overline{T}}$ **do**
 - 4: Observe context reward pair $(x_t; r_t)$
 - 5: Select action a_t uniformly at random from A , independent of x_t
 - 6: Observe reward $r_t(a_t)$
 - 7: **end for**
 - 8: Compute regression estimator $\hat{f}_j = \operatorname{argmin}_{f \in F_j} \frac{1}{C \rho_{\overline{T}}} \sum_{t=1}^{C \rho_{\overline{T}}} [f(x_t; a_t) - r_t(a_t)]^2$ (via offline regression oracle) for all $j \in [M]$
 - 9: **Model Selection test:**
 - 10: Obtain another set of $C \rho_{\overline{T}}$ fresh samples of $(x_t; r_t; a_t)$ via pure exploration (similar to line 4-6)
 - 11: Construct the test statistic $T_j = \frac{1}{C \rho_{\overline{T}}} \sum_{t=1}^{C \rho_{\overline{T}}} (\hat{f}_j(x_t; a_t) - r_t(a_t))^2$ for all $j \in [M]$
 - 12: **Thresholding:** Find the minimum index $\hat{j} \in [M]$ such that $T_{\hat{j}} \leq \epsilon$. We obtain the regressor $\hat{f} \in F_{\hat{j}}$.
 - 13: **Commit:**
 - 14: **for** $t = 2C \rho_{\overline{T}} + 1; \dots; T$ **do**
 - 15: Observe context reward pair $(x_t; r_t)$
 - 16: Select action a_t according to FALCON algorithm of [71] with the function class $F_{\hat{j}}$.
 - 17: Observe reward $r_t(a_t)$
 - 18: **end for**
-

9.3 Explore-then-Commit algorithm for model selection

In the previous section, we analyze FALCON with model selection, and obtain high probability regret bounds. Here, instead, we use a simple explore-then-commit (ETC) algorithm for selecting the correct model, and then commit to it during the exploitation phase. After a round of exploration, we do a (one-time) threshold based testing to estimate the function class, and after that, exploit the estimated function class for the rest of the iterations. We show that this simple strategy finds the optimal function class F_d with high probability. The details are given in Algorithm 15. We now explain the exploration and exploitation phases of this algorithm.

Exploration: Let the time horizon be T . For now, we assume that we know the value of T . We keep $2C \rho_{\overline{T}}$ time for exploration, where C is a universal constant. The exploration works in 2 phases.

Collect samples For $2C \rho_{\overline{T}}$ time epochs, we sample from the Nature randomly. Precisely, the context-reward pair $(x_t; r_t)$ is being sampled by Nature in an i.i.d fashion, and the action the agent

takes is chosen uniformly at random from the action set A . In particular, the action is chosen independent of the context x_t . Hence, this is a pure exploration strategy.

Obtain regression estimate Based on the samples of the first $C^{\rho} \bar{T}$ rounds, we estimate the regression function \hat{f} for all the (hypothesis) function classes $F_1; \dots; F_M$. More precisely, we call the offline regression oracle (as mentioned in [71]) and compute the following function estimate:

$$\hat{f}_j = \operatorname{argmin}_{f \in F_j} \left(\sum_{t=1}^{C^{\rho} \bar{T}} f(x_t; a_t) - r_t(a_t) \right)^2$$

for all $j \in [M]$.

Testing on fresh samples To remove dependence issues, we use the remaining $C^{\rho} \bar{T}$ samples obtained from the sampling phase. Here we actually compute the following test statistic for all hypothesis classes:

$$T_j = \frac{1}{C^{\rho} \bar{T}} \sum_{t=1}^{C^{\rho} \bar{T}} (\hat{f}_j(x_t; a_t) - r_t(a_t))^2$$

for all $j \in [M]$. The idea is to perform a thresholding on T_j . Intuitively, since the (conditional) expectation of $r(a)$ is $f_d(x; a)$, we can expect T_j to be small for hypothesis classes that contain f_d . Otherwise, thanks to the separation condition, we expect T_j to be large. So, a simple thresholding on T_j is sufficient to identify the smallest hypothesis class where f_d lies and we show that this procedure succeeds with high probability.

As mentioned earlier, we provide guarantees with ETC under 2 settings—(a) gap constrained and (b) long horizon.

ETC with (large) Gap constraint

Recall from Assumption 23 that, γ is the minimum separation across the function classes. Here, we analyze Algorithm 15 with the following assumption on the gap γ .

Assumption 24. We assume that gap satisfies

$$\gamma^2 + C T^{-1/2} \log(j F_M j) + T^{-1/4} \sqrt{\log(1/\delta)}$$

where T is the time horizon, $\delta \in (0; 1)$ and C is a universal constant.

Note that the above assumption suggests that for large T , the gap satisfies $\gamma^2 + \text{small-term}$. Armed with the above assumption, we have the following result:

Lemma 58. *Suppose we set the threshold in Algorithm 15 as*

$$\delta = \frac{1}{2} + C^0 T^{-1/2} \log(jF_M j) + T^{-1/4} \sqrt{\log(1/\delta)} ;$$

for a universal constant C^0 . Then, with probability at least $1 - 2M^{-1}$, Algorithm 15 identifies the correct model class F_d .

Regret Guarantee We now analyze the regret performance of Algorithm 15. The regret $R(T)$ is comprised of 2 stages; (a) exploration and (b) commit (exploitation). We have the following result.

Theorem 24. *Suppose Assumptions 2 and 3 hold. Then with probability at least $1 - (2M + 1)^{-1}$, running Algorithm 15 for T iterations yield*

$$R(T) = O\left(\sqrt{KT \log(jF_d jT)}\right);$$

ETC with long horizon

One of the potential issues of the previous section is the assumption on the gap. In this section, we remove this restriction. In other words, we trade off the requirement on the gap with the time horizon T . We show that for any gap Δ (as long as $\Delta > c_0$ for some small constant c_0), our proposed model selection procedure succeeds provided $T \gg \Delta^{-4}$.

Guarantee We have the following result.

Theorem 25. *Suppose we choose the threshold $\delta = \frac{1}{2} + c_0$ in Algorithm 15. The model selection procedure succeeds with probability exceeding $1 - 2(M + 1)^{-1}$ provided,*

$$T \geq C \max\left\{ \frac{\log^2(1/\delta)}{c_0^4}, \frac{1}{c_0^4} \log^2(jF_M j); \log^2(1/\delta) \right\};$$

and running Algorithm 15 for T iterations yield

$$R(T) = O\left(\sqrt{KT \log(jF_d jT)}\right);$$

Remark 43. Note that if the separation $\Delta \gg \Delta^{-2}$ (high SNR regime), the first condition is equivalent to $T \gg \Delta^{-4}$, and we get the setting with large gap.

Remark 44. Note that we still require a gap constraint, $\Delta > c_0$. But this is a mild condition compared to Assumption 24, as one can choose c_0 sufficiently small constant.

9.4 Model Selection with infinite function classes

The results in Section 9 holds for finite function classes, since the regret bound depends on the cardinality of the function class. However, as shown in [71], it can be easily extended to the infinite function class setting. Exploiting the notion of the complexity of infinite function classes, the reduction is done.

Like before, we consider a nested sequence of M function classes $F_1 \subseteq \dots \subseteq F_M$. The reward is sampled from an unknown function f_d lying in the (smallest) function class indexed by $d \geq [M]$, which is unknown. Given the function classes, our job is to find the function class F_d , and subsequently exploit the class to obtain sub-linear regret. Let us first rewrite the separability assumption.

We assume that the function classes $F_1 \subseteq \dots \subseteq F_M$ are compact. This, in conjunction with the extreme value theorem ensures the existence of the following minimizers: for $j < d$, we define

$$f_j = \operatorname{arginf}_{f \in F_j} \mathbb{E}_{x;a} [f(x; a) - f_d(x; a)]^2 \quad (9.1)$$

for all pairs $(x; a)$. For $j = d$, we know that this minimizer is indeed f_d . This comes directly from the realizability assumption. Note that we require the existence of the minimizer (regression function) in order to use it for selecting actions in the contextual bandit framework (see [71])

Having defined the minimizers, we have the following separability assumption.

Assumption 25. For any f_j , where $j < d$, we have

$$\mathbb{E}_{x;a} [f_j(x; a) - f_d(x; a)]^2 \geq \gamma_j;$$

for all pairs $(x; a) \in X \times A$. Here we interpret γ_j as the minimum separation across function classes.

Similar to [71], here, we are not worried about the explicit form of the regression functions f_j . Rather, we assume the following performance guarantee of the offline regressor. For $j = d$ (meaning, the class containing the true regressor f_d), we have the following assumption.

Assumption 26. Given n i.i.d data samples $(x_1; a_1; r_1(a_1)); (x_2; a_2; r_2(a_2)); \dots; (x_n; a_n; r_n(a_n))$, the offline regression oracle returns a function \hat{f}_j , such that for $\epsilon > 0$, with probability at least $1 - \epsilon$,

$$\mathbb{E}_{x;a} [\hat{f}_j(x; a) - f_d(x; a)] \leq \epsilon_{F_j}(n)$$

This assumption is taken from [71], and it is used to obtain the regret guarantee of the FALCON algorithm. As discussed in the above-mentioned paper, the quantity $\epsilon_{F_j}(n)$ is a decreasing function of n , e.g., $\epsilon_{F_j}(n) = \mathcal{O}(1/n)$. As an instance, consider the class of all linear regressors in \mathbb{R}^d . In that case, $\epsilon_{F_j}(n) = \mathcal{O}(d/n)$. For function classes with finite VC dimension (or related quantities like VC-sub graph or fat-shattering dimension; pseudo dimension in general, denoted by d), we have $\epsilon_{F_j}(n) = \mathcal{O}(d/n)$.

Here, to avoid repetition, we do not present all our previous results in the infinite function class setting. We consider two instances:

1. The ETC algorithm with large gap condition
2. The adaptive contextual bandit (ACB) algorithm

The model-selection algorithm remains more-or-less the same overall. For Option I, we explore for the first $2C/\bar{T}$ rounds. The first C/\bar{T} rounds are used to collect samples (x_t, r_t, a_t) via pure exploration. Feeding this samples to the offline regression oracle, and focusing on the individual function classes $f_{F_j} g_{j=1}^M$ separately, we obtain $(\hat{f}_j; F_j; (C/\bar{T}))$ for all $j \in [M]$. Thereafter, we perform another round of pure exploration, and obtain C/\bar{T} fresh samples. Like in the finite case, we construct statistic T_j for all $j \in [M]$.

For Option II, we collect all the samples from the previous epoch of the FALCON algorithm, split the samples, to obtain the regression estimate \hat{f}_j^m and similarly construct test statistic T_j^m for all $j \in [M]$.

Similar to Algorithm 15 and 14, we choose the correct model based on a threshold on the test statistic T_j (for Option II, it is T_j^m). Let us assume that the threshold is δ . We show that for all $j = d$, $T_j > \delta$, and for all $j < d$, $T_j < \delta$ with high probability. We also obtain an explicit form of δ . Once this is shown, the model selection procedure follows exactly as Algorithm 15, i.e., we find the smallest index $\hat{d} \in [M]$, for which $T_{\hat{d}} > \delta$. With high probability, we show that $\hat{d} = d$.

Regret Guarantee We first show the guarantee for Option I, and then Option II.

Theorem 26. (ETC) Suppose Assumptions 2, 6 and 5 hold and the gap satisfies

$$\Delta^2 + C_1 \sum_{F_M} (C/\bar{T}) + T^{-1/4} \Delta \sqrt{\log(1/\delta)} > \delta;$$

Then running Algorithm 15 for T iterations with threshold

$$\delta = \Delta^2 + C_1 \sum_{F_M} (C/\bar{T}) + T^{-1/4} \Delta \sqrt{\log(1/\delta)} > \delta;$$

yields, with probability at least $1 - (2M + 1)\delta$, the regret

$$R(T) = O\left(\frac{1}{K_{F_d} \Delta^2 T}\right);$$

Theorem 27. (ACB) Suppose Assumptions 2 and 3 hold, and $\Delta > c_0$. Then with probability at least $1 - 6M\delta$, running Algorithm 14 for T iterations yield

$$R(T) = C \max_f \frac{\log(1/\delta)}{(c_0)^2} \frac{1}{c_0^2} \max_{f \in \mathcal{F}} \log(j_{F_M} f); \log(1/\delta) \Delta + O\left(\frac{1}{K_{F_d} \Delta^2 T}\right);$$

Remark 45. The proof of these theorems parallels exactly similar to the finite function class setting. The only difference is that instead of upper-bounding the prediction error using [72], we use the definition of $\Delta(\cdot)$ to accomplish this.

9.4.1 Special Case—Stochastic Linear Bandits

Here we consider the specific instance of the above-discussed generic contextual bandits—the linear setup. We see that order-wise, the generic Algorithm 14 recovers the regret guarantees of the linear bandit setup (with finite number of arms).

Recall the problem setup of Chapter 8 for finite set of actions. Furthermore, assume that the context embeddings $\phi_j(x; a)$ is a unit d_j dimensional Gaussian random variable. Note that this is stronger than the sub-Gaussian assumption.

Rewriting the nested hypothesis class, this corresponds to setting the function classes F_j to be the linear class as the following:

$$F_j = \{f(x; a) = \sum_{i=1}^j \theta_i \phi_i(x; a) \mid \theta_i \in \mathbb{R}^{d_i}, \|\theta\|_2 \leq 1\}$$

Also, m is the smallest index such that the optimal regressor is realized, i.e.,

$$f_m(x; a) = \sum_{i=1}^m \theta_i^* \phi_i(x; a)$$

So, this can be cast-ed as a model selection problem in our framework.

Let us look at the separability assumption of 23. For $j < m$, we first compute

$$f_j = \operatorname{arginf}_{f \in F_j} \mathbb{E}_{x;a} [f(x; a) - f_m(x; a)]^2;$$

and then compute the quantity

$$\mathbb{E}_{x;a} [f_j(x; a) - f_m(x; a)]^2;$$

Substituting $f_m(x; a) = \sum_{i=1}^m \theta_i^* \phi_i(x; a)$ and optimizing over f_j , we obtain

$$\mathbb{E}_{x;a} [f_j(x; a) - f_m(x; a)]^2 = \sum_{i=1}^j \frac{\sigma_i^2}{d_m - j + 1};$$

where σ_i is the minimum magnitude of the non-zero coordinate of θ^* , i.e., $\sigma_i = \min_{i: \theta_i^* \neq 0} |\theta_i^*|$.

Note that we exploit the Gaussian distribution of $\phi_i(\cdot)$ to obtain the above calculation.

Hence for linear stochastic bandits, one may take

$$f_j = \sum_{i=1}^j \theta_i^* \phi_i(x; a);$$

where σ_i is the minimum magnitude of the non-zero coordinate of θ^* , i.e., $\sigma_i = \min_{i: \theta_i^* \neq 0} |\theta_i^*|$.

Substituting in the regret expression of Theorem 27, with $F_{d_m} := (T) = \mathcal{O}(d_m = T)$ (linear class of d_m dimensional functions, VC-dimension is $d_m + 1$), we obtain

$$R_{lin}(T) = \mathcal{O}\left(\frac{d}{4} + \sqrt{\frac{KTd_m}{T}}\right);$$

with high probability. Several remarks are in order:

Remark 46. If the number of actions $K = \mathcal{O}(1)$, the above regret scaling matches to that of Theorem 22. Hence, in this setting we recover the performance of Algorithm 13.

Remark 47. Note that in general, the performance of Algorithm 14 specialized to the linear setting is worse than Theorem 22. In particular, there is no K dependence in Theorem 22, but we have a \sqrt{K} term in the leading factor here. This matches our intuition. Algorithm 14 is applicable for any generic contextual bandit problem, whereas Algorithm 13 is specialized to the linear case only. The price in regret can be viewed as the cost of generalization.

Remark 48. Let us now focus on the additive (minor) term with no T dependence. It scales as $1 = \sqrt{4}$ here, whereas in Theorem 22 it scales as $1 = \sqrt{4.65}$. Note that, we remarked (after Theorem 13) that via carefully choosing the problem constants, the dependence in Theorem 22 can be made arbitrarily close to $1 = \sqrt{4}$. So, we have the same dependence on $\sqrt{4}$ in both the settings.

Remark 49. Finally, note that the additive term is linearly dependent (d) here, whereas it has a quadratic dependence (d^2) in Theorem 22. We believe this stems from the analysis of Algorithm 13. In Algorithm 13, we are successively estimating the support of the underlying true parameter, and it is not clear whether support recovery is indeed required to ensure low regret.

9.5 Proofs

9.5.1 Proof of Lemma 57

Let us first show that T_j concentrates around its expectation. We show it via a simple application of the Hoeffdings inequality.

Fix a particular m and $j \geq [M]$. Note that \hat{f}_j^m is computed based on 2^{m-2} samples. Also, in the testing phase, we use a fresh set of 2^{m-2} samples, and so \hat{f}_j^m is independent of the second set of samples, used in constructing T_j . Since we have $r(\cdot) \geq [0; 1]$. Furthermore, we have $\hat{f}_j^m(\cdot) \geq [0; 1]$. Note that this assumption is justified since our goal is obtain an estimate of the reward function via regression function, and this assumption also features in [71]. So the random variable $(\hat{f}_j^m(x_t; a_t) - r_t(a_t))^2$ is upper-bounded by 4, and hence sub-Gaussian with a constant parameter.

Note that we are using only the samples from the previous epoch. In the, FALCON algorithm, the regression estimate actually remains fixed over an entire epoch. Hence, conditioning on the filtration consisting of (context, action, reward) triplet upto the end of the $m-2$ -th epoch, the random variables $(\hat{f}_j^m(x_t; a_t) - r_t(a_t))^2$ $g_{t=m-1}^{m-1}$ (a total of 2^{m-2} samples) are independent. Note that similar argument is given in [71, Section 4.1] (the FALCON+ algorithm) to argue the independence of the (context, action, reward) triplet, accumulated over just the previous epoch.

Hence using Hoeffdings inequality for sub-Gaussian random variables, we have

$$P(T_j - \mathbb{E}T_j \geq \epsilon) \leq \exp(-\epsilon^2/32):$$

Let us look at the expression $\mathbb{E}T_j$.

Realizable classes: Fix m and consider $j \geq 2$ such that $j \leq d$. So, for this realizable setting, we obtain the excess risk as:

$$\begin{aligned} & \mathbb{E}_{x;r;a}[\hat{f}_j^m(x; a) - r(a)]^2 = \inf_{f \in F_j} \mathbb{E}_{x;r;a}[f(x; a) - r(a)]^2 \\ & = \mathbb{E}_{x;r;a}[\hat{f}_j^m(x; a) - r(a)]^2 = \mathbb{E}_{x;r;a}[f_d(x; a) - r(a)]^2 \\ & = \mathbb{E}_{x;a}[\hat{f}_j^m(x; a) - f_d(x; a)]^2: \end{aligned}$$

So, we have, for the realizable function class,

$$\begin{aligned} \mathbb{E} T_j^m &= \frac{1}{2^{m-2}} \mathbb{E}_{x_t;r_t;a_t} \sum_{t=1}^{2^{m-2}} [\hat{f}_j^m(x_t; a_t) - r_t(a_t)]^2 \\ &= \frac{1}{2^{m-2}} \sum_{t=1}^{2^{m-2}} \mathbb{E}_{x_t;r_t;a_t} [f_d(x_t; a_t) - r_t(a_t)]^2 + \frac{1}{2^{m-2}} \sum_{t=1}^{2^{m-2}} \mathbb{E}_{x_t;a_t} [\hat{f}_j^m(x; a) - f_d(x; a)]^2 \\ &\quad + C_1 \log(2^m j F_j) = (2^{m-2}); \end{aligned}$$

with probability at least $1 - \epsilon = 2^{-m}$. Here, the first term comes from the second moment bound of r^2 , and the second term comes from [72, Lemma 4.1]. So, by applying Hoeffding's inequality, we finally have.

$$T_j^m \leq (2^{m-2})^2 + C \frac{\log(j F_j)}{2^m} + C_1 \frac{m}{2^m} + C_3 \frac{\sqrt{\log(1/\epsilon)}}{2^{m-2}} + C_4 \frac{\sqrt{m}}{2^{m-2}}$$

with probability at least $1 - \epsilon = 2^{-m}$. Note that, provided

$$2^m \geq \frac{1}{C_0} \max\{\log(j F_j), \log(1/\epsilon)\} \tag{9.2}$$

with carefully chosen constants, we have

$$T_j^m \leq (2^{m-2})^2 + C_0$$

with probability at least $1 - \epsilon = 2^{-m}$, where C_0 is a small universal constant.

Non-Realizable classes: For the non realizable classes, we have the following calculation. For any $f \in F_j$, where $j < d$, we have

$$\begin{aligned} & \mathbb{E}_{x;r;a}[f(x; a) - r(a)]^2 = \mathbb{E}_{x;r;a}[r(a) - f_d(x; a)]^2 \\ & = \mathbb{E}_{x;a;r}[(f(x; a) - f_d(x; a))(f(x; a) + f_d(x; a) - 2r(a))] \\ & = \mathbb{E}_{x;a} \mathbb{E}_{r|x}[(f(x; a) - f_d(x; a))(f(x; a) + f_d(x; a) - 2r(a))] \\ & = \mathbb{E}_{x;a}[(f(x; a) - f_d(x; a))(f(x; a) + f_d(x; a) - 2\mathbb{E}_{r|x}r(a))] \\ & = \mathbb{E}_{x;a}[f(x; a) - f_d(x; a)]^2; \end{aligned}$$

where the third inequality follows from the fact that given context x , the distribution of r is independent of a (see [72, Lemma 4.1]).

So, we have

$$\mathbb{E}_{x;r;a}[f(x; a) - r(a)]^2 = \mathbb{E}_{x;r;a}[r(a) - f_d(x; a)]^2 + \mathbb{E}_{x;a}[f(x; a) - f_d(x; a)]^2 + \sigma^2;$$

where the last inequality comes from the separability assumption along with the assumption on the second moment. Since the regressor $\hat{f}_j^m \geq F_j$, we have

$$\mathbb{E}_{x;r;a}[\hat{f}_j^m(x; a) - r(a)]^2 = \mathbb{E}_{x;r;a}[r(a) - f_d(x; a)]^2 + \mathbb{E}_{x;a}[f(x; a) - f_d(x; a)]^2 + \sigma^2;$$

Now, using 2^{m-2} samples, we obtain

$$T_j^m + \sigma^2 \leq C_3 \frac{\rho \log(1/\delta)}{2^{m-2}} + C_4 \frac{\rho}{2^{m-2}}$$

with probability at least $1 - \delta/2^m$. Now, suppose m is such that

$$\frac{\rho \log(1/\delta)}{2^{m-2}} + \frac{\rho}{2^{m-2}} \leq c_0;$$

where c_0 is a small universal constant (see equation (9.2)) such that $\delta > c_0$. Observe that, provided

$$2^m \geq \frac{\log(1/\delta)}{c_0^2};$$

and with appropriately chosen constants, we have

$$T_j^m \leq \sigma^2 + c_0;$$

with probability at least $1 - \delta/2^m$.

Based on the above calculation, we choose the threshold $\tau = \sigma^2 + c_0$. Finally, we say that provided

$$2^m \geq \max\left\{ \frac{\log(1/\delta)}{c_0^2}, \frac{1}{c_0^2} \max_{j \in \mathcal{M}} \log(j F_{Mj}); \log(1/\delta) \right\};$$

the model selection procedure succeeds with probability exceeding

$$1 - \sum_{m=1}^{\infty} \delta/2^m \geq 1 - \delta;$$

9.5.2 Proof of Theorem 23

The above calculation shows that as soon as

$$2^m \leq \max_{f \in \mathcal{F}} \frac{\log(1 - \epsilon)}{(\frac{c_0}{2})^2}; \frac{1}{c_0^2} \max_{f \in \mathcal{F}} \log(j F_M j = \epsilon); \log(1 - \epsilon) \geq \epsilon;$$

the model selection procedure will succeed with high probability. Until the above condition is satisfied, we do not have any handle on the regret and hence the regret in that phase will be linear. This corresponds the first term in the regret expression. After the condition is satisfied, the regret is given by (see [71])

$$\sum_{m=1}^M \mathbb{P} \left[\frac{K(m-1) \log(j F_d j(m-1) = \epsilon)}{KT \log(j F_d j T = \epsilon)} \right];$$

with probability exceeding $1 - \epsilon$. Finally, a union bound on the error probability yields the theorem.

9.5.3 Proof of Lemma 58

Since, we have samples from pure exploration, let us first show that T_j concentrates around its expectation. We show it via a simple application of the Hoeffdings inequality.

Fix a particular $j \in [M]$. Note that \hat{f}_j is computed based on the first set of $C^{\rho} \bar{T}$ samples. Also, in the testing phase, we again sample $C^{\rho} \bar{T}$ samples, and so \hat{f} is independent of the second set of $C^{\rho} \bar{T}$ samples, used in constructing T_j . Note that we have $r(\cdot) \in [0; 1]$. Furthermore, we have $\hat{f}(\cdot) \in [0; 1]$. Note that this assumption is justified since our goal is obtain an estimate of the reward function via regression function. This assumption also features in [71]. So the random variable $(\hat{f}_j(x_t; a_t) - r_t(a_t))^2$ is upper-bounded by 4, and hence sub-Gaussian with a constant parameter. Also, note that since we are choosing an action independent of the context, the random variables $f(\hat{f}_j(x_t; a_t) - r_t(a_t))^2_{t=1}^{C^{\rho} \bar{T}}$ are independent. Hence using Hoeffdings inequality for sub-Gaussian random variables, we have

$$\mathbb{P}(T_j - \mathbb{E} T_j \geq \epsilon) \leq \exp(-\frac{\epsilon^2}{32 C^{\rho} \bar{T}});$$

Re-writing the above, we obtain

$$T_j - \mathbb{E} T_j \leq \frac{\epsilon}{C^{\rho} \bar{T}}$$

with probability at least $1 - \epsilon$. Let us look at the expression $\mathbb{E} T_j$.

$$\mathbb{E} T_j = \mathbb{E} \left[\frac{1}{C^{\rho} \bar{T}} \sum_{t=1}^{C^{\rho} \bar{T}} (\hat{f}_j(x_t; a_t) - r_t(a_t))^2 \right];$$

Case I: Realizable Class First consider the case that $j = d$, meaning that $f_d \in F_j$. So, for this realizable setting, we obtain the excess risk as (using [72])

$$\begin{aligned} & \mathbb{E}_{x;r;a} [\hat{f}_j(x; a) - r(a)]^2 = \inf_{f \in F_j} \mathbb{E}_{x;r;a} [f(x; a) - r(a)]^2 \\ & = \mathbb{E}_{x;r;a} [\hat{f}_j(x; a) - r(a)]^2 = \mathbb{E}_{x;r;a} [f_d(x; a) - r(a)]^2 \\ & = \mathbb{E}_{x;a} [\hat{f}_j(x; a) - f_d(x; a)]^2: \end{aligned}$$

So, we have, for the realizable function class,

$$\begin{aligned} \mathbb{E} T_j &= \frac{1}{C} \frac{1}{T} \sum_{t=1}^T \mathbb{E}_{x_t;r_t;a_t} [\hat{f}_j(x_t; a_t) - r_t(a_t)]^2 \\ &= \frac{1}{C} \frac{1}{T} \sum_{t=1}^T \mathbb{E}_{x_t;r_t;a_t} [f_d(x_t; a_t) - r_t(a_t)]^2 + \frac{1}{C} \frac{1}{T} \sum_{t=1}^T \mathbb{E}_{x_t;a_t} [\hat{f}_j(x; a) - f_d(x; a)]^2 \\ & \leq \frac{1}{C} \frac{1}{T} \sum_{t=1}^T \mathbb{E}_{x_t;r_t;a_t} [f_d(x_t; a_t) - r_t(a_t)]^2 + C_1 \log(j F_j) = (C \frac{1}{T})^{-1}; \end{aligned}$$

with probability at least $1 - \frac{\delta}{2}$, where C_1 is a known universal constant. Here, we also use the fact that the second moment bound is $\frac{1}{C} \frac{1}{T} \sum_{t=1}^T \mathbb{E}_{x_t;r_t;a_t} [f_d(x_t; a_t) - r_t(a_t)]^2$. So, we finally have.

$$T_j \leq \frac{1}{C} \frac{1}{T} \sum_{t=1}^T \mathbb{E}_{x_t;r_t;a_t} [f_d(x_t; a_t) - r_t(a_t)]^2 + C_2 T^{-1} \log(j F_j) + C_3 T^{-1} \frac{1}{\log(1 - \frac{\delta}{2})}$$

with probability at least $1 - \frac{\delta}{2}$.

Case II: Non-realizable class We now consider the case when $j < d$, meaning that f_d does not lie in F_j . We have

$$\mathbb{E}_{x;r;a} [\hat{f}_j(x; a) - r(a)]^2 = \inf_{f \in F_j} \mathbb{E}_{x;r;a} [f(x; a) - r(a)]^2 > 0:$$

Continuing, we obtain

$$\begin{aligned} \mathbb{E} T_j &= \frac{1}{C} \frac{1}{T} \sum_{t=1}^T \mathbb{E}_{x_t;r_t;a_t} [\hat{f}_j(x_t; a_t) - r_t(a_t)]^2 \\ & \geq \inf_{f \in F_j} \mathbb{E}_{x;a} [f(x; a) - f_d(x; a)]^2; \\ & \geq \dots \end{aligned}$$

So, in this setting,

$$T_j \leq \mathbb{E} T_j \leq \frac{S}{C} \frac{32 \log(1 - \frac{\delta}{2})}{T} + \frac{S}{C} \frac{32 \log(1 - \frac{\delta}{2})}{T};$$

with probability at least $1 - \frac{\delta}{2}$.

Choice of threshold If we can ensure the lower bound (on T_j) in the non-realizable setting is larger than the upper bound in the realizable case, we can ensure identifiability. Based on the above calculation, we fix the following threshold on the test statistic T_j ,

$$2 + C^{\rho} T^{1-2} \log(jF_{j=}) + T^{1-4} \frac{\rho}{\log(1=)}$$

$$2 + C^{\rho} T^{1-2} \log(jF_{Mj=}) + T^{1-4} \frac{\rho}{\log(1=)} =$$

For all $j \geq [M]$, we compare T_j with this threshold . From the above calculation, it is clear that

$$T_j$$

with high probability for all $j < d$. Furthermore, recall that from Assumption 24). For sufficiently large T , it is ensured that

$$T_j >$$

for all $j < d$ (with high probability). This implies that we can identify whether a function class is realizable or not simply based on the threshold . Finally, a simple union bound yields the result.

9.5.4 Proof of Theorem 24

The regret $R(T)$ can be decomposed in 2 stages, namely exploration and exploitation.

$$R(T) = R_{\text{explore}} + R_{\text{exploit}}$$

Since we spend $2C^{\rho} T$ time steps in exploration, and $r_t(\cdot) \in [0;1]$, the regret incurred in this stage

$$R_{\text{explore}} \leq C_1 \frac{\rho}{T}$$

Now, at the end of the explore stage, provided Assumptions 2 and 3, we know, with probability at least $1 - 2^{-\rho}$, we obtain the true function class F_d . The threshold is set in such a way that we obtain the above result. Now, we would just commit to the function class and use the contextual bandit algorithm, namely FALCON. From [71], the regret guarantee of FALCON is

$$R_{\text{exploit}} \leq O \left(\frac{\rho}{K(T - 2C^{\rho} T) \log(jF_d j(T - 2C^{\rho} T))} \right)$$

$$O \left(\frac{\rho}{KT \log(jF_d jT)} \right) ;$$

with probability exceeding $1 - 2^{-\rho}$. Combining the above expressions yield the result.

9.5.5 Proof of Lemma 25

As usual, we consider 2 cases:

Relizable class: The calculation here remains the same as before,

$$T_j \leq \frac{2}{\epsilon} + C_2 T^{1-2} \log(j F_j) + C_3 T^{1-4} \frac{1}{\log(1-\epsilon)}$$

with probability at least $1 - 2\epsilon$. We now choose T such that

$$C_2 T^{1-2} \log(j F_j) + C_3 T^{1-4} \frac{1}{\log(1-\epsilon)} \leq c_0$$

$$T \geq \frac{1}{c_0^4} \max_j \log^2(j F_j); \log^2(1-\epsilon);$$

for appropriate choice of constants. Using this, we get

$$T_j \leq \frac{2}{\epsilon} + c_0$$

with probability at least $1 - 2\epsilon$.

Non-realizable class For any $f \notin F_j$, where $j < d$, we have

$$\begin{aligned} & \mathbb{E}_{x;r;a} [f(x;a) - r(a)]^2 = \mathbb{E}_{x;r;a} [r(a) - f_d(x;a)]^2 \\ & = \mathbb{E}_{x;a;r} [(f(x;a) - f_d(x;a))(f(x;a) + f_d(x;a) - 2r(a))] \\ & = \mathbb{E}_{x;a} \mathbb{E}_{r|x} [(f(x;a) - f_d(x;a))(f(x;a) + f_d(x;a) - 2r(a))] \\ & = \mathbb{E}_{x;a} [(f(x;a) - f_d(x;a))(f(x;a) + f_d(x;a) - 2\mathbb{E}_{r|x} r(a))] \\ & = \mathbb{E}_{x;a} [f(x;a) - f_d(x;a)]^2; \end{aligned}$$

where the third inequality follows from the fact that given context x , the distribution of r is independent of a . This comes from the basic model of contextual bandits.

So, we have

$$\mathbb{E}_{x;r;a} [f(x;a) - r(a)]^2 \leq \mathbb{E}_{x;r;a} [r(a) - f_d(x;a)]^2 + \mathbb{E}_{x;a} [f(x;a) - f_d(x;a)]^2 + \frac{2}{\epsilon};$$

where the last inequality comes from the separability assumption along with the assumption on the variance. Since the regressor $\hat{f}_j \notin F_j$, we have

$$\mathbb{E}_{x;r;a} [\hat{f}_j(x;a) - r(a)]^2 \leq \mathbb{E}_{x;r;a} [r(a) - f_d(x;a)]^2 + \mathbb{E}_{x;a} [f(x;a) - f_d(x;a)]^2 + \frac{2}{\epsilon};$$

Now, using $C \frac{1}{\epsilon}$ samples, we obtain

$$T_j \leq \frac{2}{\epsilon} + \frac{2}{\epsilon} + C_3 T^{1-4} \frac{1}{\log(1-\epsilon)}$$

with high probability. Note that, provided $\frac{2}{\epsilon} > c_0$ and

$$T > C \frac{\log^2(1-\epsilon)}{(\frac{2}{\epsilon})^4};$$

for a sufficiently large constant C , we have

$$T_j \leq \frac{1}{C} + c_0;$$

So we have

$$T_j > \frac{1}{C} + c_0$$

with high probability.

Choice of threshold: Based on the above calculation, we choose the threshold $\tau = \frac{1}{C} + c_0$. Finally we can say that our model selection procedure succeeds with high probability provided,

$$T \leq C \max_{f \in F_j} \frac{\log^2(1/\delta)}{c_0^4}; \frac{1}{c_0^4} \log^2(j|F_j|); \log^2(1/\delta)g;$$

9.5.6 Proof of Theorem 26

Case I: Realizable Class Consider $j = d$. Using calculations similar to the finite cardinality setting, we obtain

$$E T_j \leq \frac{1}{C} + \frac{1}{F_j}; (C^{\rho} \bar{T});$$

Hence, invoking Hoeffding's inequality, we obtain

$$T_j \leq \frac{1}{C} + \frac{1}{F_j}; (C^{\rho} \bar{T}) + C_1 T^{-1/4} \sqrt{\log(1/\delta)}$$

with probability at least $1 - \delta$.

Case II: Non-realizable Class We now consider the setting where $j < d$, meaning that f_d does not lie in F_j . In this case, we have

$$E_{x;r;a} [\hat{f}_j(x;a) - r(a)]^2 \geq \inf_{f \in F_j} E_{x;r;a} [f(x;a) - r(a)]^2 > 0;$$

Continuing, we obtain

$$E T_j = \frac{1}{C} + \frac{1}{T} E_{x_t;r_t;a_t} \sum_{t=1}^{\rho \bar{T}} [\hat{f}_j(x_t;a_t) - r_t(a_t)]^2 \\ \geq \inf_{f \in F_j} E_{x;a} [f(x;a) - f_d(x;a)]^2;$$

and hence

$$T_j \leq E T_j \leq \frac{1}{C} + \frac{32 \log(1/\delta)}{C^{\rho} \bar{T}};$$

Fixing a Threshold: Based on the above calculation, if we select the threshold

$$= \epsilon^2 + C_{FM} (C^{\rho} \bar{T}) + 2C_1 T^{1-4} \rho \overline{\log(1-\epsilon)}.$$

With the gap condition

$$\epsilon^2 + C_1 C_{FM} (C^{\rho} \bar{T}) + T^{1-4} \rho \overline{\log(1-\epsilon)} ;$$

we are guaranteed that $T_j > \bar{T}$ for the non realizable classes and $T_j \leq \bar{T}$ for the realizable classes with high probability. Hence exploiting the nested hypothesis class structure, the smallest index where $T_j \leq \bar{T}$ corresponds to selecting the correct model with high probability.

After obtaining the correct model class, the regret expression comes directly from [71] in the infinite function class setting.

9.5.7 Proof of Theorem 27

The proof directly follows by combining the proof of Theorem 23 and 26, and hence omitted.

Chapter 10

Model Selection for Reinforcement Learning with Function Approximation

We study the problem of model selection in Reinforcement Learning with function approximation. We restrict our attention to the framework of model based episodic *linear* MDP. In particular, similar to Chapter 8, we study both the *norm* and *sparsity/dimension* of the problem parameter as problem complexity and obtain model selection guarantees, meaning that the performance (regret) of our algorithm depends on the complexity of the problem instance, as compared to some predefined upper-bound on the complexity parameters. We first set the problem up, in the next section, and in the subsequent sections, take norm and sparsity as complexity measures and propose algorithms with regret guarantee.

10.1 Formulation and Linear Model

In this chapter, we consider an heterogeneous, episodic Markov decision process denoted by a tuple $M(S; A; H; \{P_h\}_{h=1}^H; \{r_h\}_{h=1}^H)$, where S is the state space, A is the action space, H is the length of each episode, P_h is the transition probability at step h such that $P_h(j|s; a)$ is the distribution over states provided action a is taken for state s at step h , and $r_h : S \times A \rightarrow [0; 1]$ is the deterministic reward function at step h . A policy π is a collection of H functions $f_h : S \rightarrow A$, where each of them maps a state s to an action a .

In each episode, an initial state s_1 is first picked by the environment. Then, at each step $h \in [H]$, the agent observes the state s_h , picks an action a_h according a certain policy π , receives a reward $r_h(s_h; a_h)$, and then transitions to the next state s_{h+1} , which is drawn from the distribution $P_h(j|s_h; a_h)$. The episode ends when s_{H+1} is reached. For each $(s; a) \in S \times A$, we define action values $Q_h(s; a)$ and state values $V_h(s)$ as

$$Q_h(s; a) := r_h(s; a) + \mathbb{E} \sum_{h^0=h+1}^H r_{h^0}(S_{h^0}; \pi^{h^0}(S_{h^0})) \mathbb{1}_{S_h = s; a_h = a}$$

$$V_h(s) := Q_h(s; \pi_h(s))$$

with $V_{H+1}(s) = 0$. The optimal value function $V_h(\cdot)$ and the optimal action-value function $Q_h(\cdot; \cdot)$ are given by $V_h(s) = \sup_a V_h(s; a)$ and $Q_h(s; a) = \sup_{a'} Q_h(s; a')$. For brevity, we denote $[P_h V_{h+1}](s; a) := \mathbb{E}_{s' \sim P(\cdot|s; a)} V_{h+1}(s')$. With this notation, Bellman equations for policy π can be written as

$$Q_h(s; a) = r_h + P_h V_{h+1}(s; a)$$

and the Bellman optimality equation is given by

$$Q_h(s; a) = r_h + P_h V_{h+1}(s; a)$$

In the online learning setting, the agent interacts with the environment for K episodes to learn the unknown transition kernels $\{P_h\}$ and hence to improve its policy. For each $k = 1, \dots, K$, the environment picks a starting state s_1^k , and the agent chooses a policy π^k that will be followed through the whole k -th episode. The objective is to design a learning algorithm that constructs a sequence $\{\pi^k\}$ that minimizes the total regret

$$R(K) = \sum_{k=1}^K \sum_{i=1}^H |V_1^k(s_1^k) - V_1^*(s_1^k)|$$

In this chapter, we consider a special class of MDPs called *linear kernel MDPs* (a.k.a., linear mixture MDPs) [73]. Roughly speaking, it means that the transition kernel $\{P_h\}$ can be represented as a linear function of a given feature map $\phi: S \times A \rightarrow \mathbb{R}^d$. Formally, we have the following definition.

Definition 13 (Linear kernel MDP). *$M = \{S; A; H; \{P_h\}_{h=1}^H; \{r_h\}_{h=1}^H\}$ is called an heterogeneous, episodic B -bounded linear kernel MDP if there exists a known feature mapping $\phi: S \times A \rightarrow \mathbb{R}^d$ and an unknown vector $\theta_h \in \mathbb{R}^d$ with $\|\theta_h\|_2 \leq B$ such that*

(i) *For any state-action-state triplet $(s; a; s') \in S \times A \times S$ and step $h \in [H]$, $P(s'|s; a) = \sum_{i=1}^d \theta_{hi} \phi(s; a)_i \phi(s')_i$.*

(ii) *For any bounded function $V: S \rightarrow [0; 1]$ and any tuple $(s; a) \in S \times A$, we have $\|V_h(s; a) - V^*(s; a)\|_2 \leq 1$, where $V_h(s; a) := \sum_{s' \in S} P(s'|s; a) V(s')$.*

In the learning problem, the vectors $\{\theta_h\}_{h=1}^H$ are unknown to the learner. The episodic MDP is parameterized by $\theta = \{\theta_h\}_{h=1}^H$ and we denote the MDP by M_θ . In what follows, we define two natural complexity measures of M_θ : (a) $\sum_{h=1}^H \|\theta_h\|_2$ for all $h \in [H]$, and (b) $\sum_{h=1}^H \|\theta_h\|_2^2$ for all $h \in [H]$.

Notation We keep the notation consistent with the previous chapters. For positive integer r , by $[r]$, we denote the set of integers $\{1, 2, \dots, r\}$. Also $\|\cdot\|_2$ denotes ℓ_2 norm unless otherwise specified. Moreover, $\lambda_{\min}(A)$ denotes the minimum eigenvalue of the matrix A .

Algorithm 16: Adaptive Reinforcement Learning (norm)-ARL-Norm

- 1: **Input:** Over estimate of norm B , the initial phase length K_1 , Slack $\epsilon_1 = \epsilon > 0$
 - 2: Initial estimates: $b_h^{(1)} = \max_{f \in \mathcal{F}_h} \|f\|_B$, for all $h \in [H]$.
 - 3: **for** epochs $i = 1; 2; \dots; N$ **do**
 - 4: Play UCRL-VTR⁺ with norm estimates $\{b_h^{(i)}\}_{h=1}^H$, until the end of epoch i (denoted by E_i)
 - 5: At $k = E_i$, refine estimate of $\|k_h\|_k$ as, $b_h^{(i+1)} = \max_{\mathcal{C}_{E_i, h}} \|k_h\|_k$
 - 6: Set $K_{i+1} = 2K_i$, $\epsilon_{i+1} = \frac{\epsilon}{2}$.
 - 7: **end for**
-

10.2 Norm as Complexity measure

In this section, we define $\|k_h\|_k$ as a natural measure of complexity of the problem. Recall that the transition kernel is parameterized by θ_h , i.e., $P(s^j | s; a) = \theta_h(s^j | s; a)$. Hence, if θ_h is close to 0 (hence $\|k_h\|_k$ is very small), the set of states s^j for the next step will have a small cardinality. Similarly, when θ_h is high (with $\|k_h\|_k$ is large), the above-mentioned cardinality will be quite large. We mention here that, via the same set of arguments the sparsity (number of non-zero coordinates) of θ_h also serves a natural measure of complexity. We consider this in the subsequent section.

Here, we propose and analyze an algorithm, that adapts to the problem complexity $\|k_h\|_k$, and as a result, the regret obtained will depend on $\|k_h\|_k$. In prior works, usually it is assumed that $f_h \in \mathcal{F}_h$ lies in a norm ball with known radius, i.e., $\|f_h\|_B \leq B$ (see [73]). This is a non-adaptive algorithm and the algorithm uses B as a proxy for the problem complexity, which can be a huge over-estimate. In sharp contrast, we start with this over estimate of $\|k_h\|_k$, and successively refine this estimate over multiple epochs. We show that this refinement strategy yields a consistent sequence of estimates of $\|k_h\|_k$, and as a consequence, our regret bound depends on $\|k_h\|_k$, not B .

Our idea: We take the UCRL-VTR⁺ of [74] as a blackbox, and successively refine the estimates of $f_h \in \mathcal{F}_h$ over epochs. Note that we take UCRL-VTR⁺ as our baseline algorithm since the regret of the same is near optimal. In particular UCRL-VTR⁺ tracks the variance of the value function (via updating a parameter $\theta_{k,h}$) along with a careful estimation of covariance matrix corresponding to the embeddings $\phi_h(\cdot)$.

We now proceed for the successive model refinement based model selection algorithm. The details are given in Algorithm 16. In particular we choose the doubling epoch. As mentioned, we use the model based RL algorithm of [74], namely UCRL-VTR⁺. We begin with some over estimate of $f_h \in \mathcal{F}_h$ (in particular, we assume $\|k_h\|_k \leq B$ for all $h \in [H]$), and over epochs, we refine this estimate. In particular, as seen in Algorithm 16, at the end of the epoch, based on the confidence set built, we choose the new estimate of $f_h \in \mathcal{F}_h$. We argue that this sequence of estimates is indeed consistent, and as a result, the regret depends on $f_h \in \mathcal{F}_h$.

Let us have a look how UCRL-VTR⁺ constructs the confidence ellipsoid $\hat{\mathcal{C}}_{k,h}$ for episode k .

From [74], we obtain the following

$$\hat{C}_{k;h} = \frac{1}{k} \sum_{t=1}^k \hat{C}_{k;h}^t$$

where $\hat{C}_{k;h}^t$ is an estimate of $C_{k;h}$ (a regularized least square estimator with adjusted variance) computed by UCRL-VTR⁺, and

$$\hat{C}_{k;h}^t = 8 \frac{\rho}{d \log(1 + k) \log(4k^2 H)} + 4 \frac{\rho}{d \log(4k^2 H)} + b_h$$

where b_h is the current estimate of k and ρ is the regularization in the least squares problem. Also, we have

$$\hat{C}_{k;h}^t = I + \sum_{s=1}^k \frac{1}{v_{t,h+1}(s^t; a_h^t)} \sum_{a=1}^k v_{t,h+1}(s^t; a_h^t) \hat{C}_{k;h}^t(s^t; a_h^t)$$

Then, [74] shows that

$$\mathbb{P} \left(\sum_{h=1}^H \hat{C}_{k;h} \leq \frac{1}{3} \right) \geq 1 - \frac{1}{3}$$

Now, let us present the main result of this section. We show that the sequence of estimators build via Algorithm 16 actually converges to the true norm estimates. We first make the following assumption on the embeddings:

Assumption 27. We have, for all $h \geq [H]$,

$$\mathbb{E}_{k-1} \left[\sum_{s=1}^k \frac{1}{v_{k,h+1}(s^k; a_h^k)} \sum_{a=1}^k v_{k,h+1}(s^k; a_h^k) = c < \min I \right]$$

where, $\min > 0$, and $\mathbb{E}_{k-1}[\cdot]$ is a conditional expectation, conditioned on the observation upto episode $k-1$.

We emphasize that similar assumptions have featured in existing literature, e.g., see [67, 70]. Note that we had similar assumptions in Chapter 8. In fact removing such spectral assumption is an interesting open problem (see COLT open problem, [251]). Model selection without the above assumption is an interesting future direction.

Armed with the above assumption, we obtain the following result, which is crucial in understanding the rate at which the confidence ellipsoid of $f_{h, g_{h=1}^H}$ shrinks.

Lemma 59. Consider the matrix $M_{k;h} = I + \sum_{t=1}^k \frac{1}{v_{t,h+1}(s^t; a_h^t)} \sum_{a=1}^k v_{t,h+1}(s^t; a_h^t) \hat{C}_{k;h}^t$. For all $k \geq \frac{1}{\min} \log(2dK)$ and all $h \geq [H]$, we have

$$\min(M_{k;h}) \geq I + \sum_{t=1}^k \frac{1}{v_{t,h+1}(s^t; a_h^t)} \sum_{a=1}^k v_{t,h+1}(s^t; a_h^t) \hat{C}_{k;h}^t + \frac{1}{\min k=2}$$

with probability at least $1 - \frac{1}{3}$, where

$$\min(\cdot) = \frac{16d}{2 \min H^2} + \frac{8 \rho}{3 H \min} \log(2dK)$$

We now show that the norm estimates produced by ARL-norm indeed converges to the true norm $\|f_h^k g_{h=1}^H\|$ at an exponential rate with high probability.

Lemma 60. *With probability exceeding $1 - 8H^{-1}$, the sequence $\|f_h^{(i)} g_{i=1}^H\|$ converges to $\|f_h^k g_{h=1}^H\|$ at a rate $O(\frac{1}{2^i})$, and we obtain $\|f_h^{(i)} g_{i=1}^H\| \leq (c_1 k + c_2) \|f_h^k g_{h=1}^H\|$ for all i , provided the length of the initial episode K_1 satisfies the following*

$$K_1 > \max_{h \in [H]} \frac{1}{\min_{h \in [H]} \rho_h} \log_2 \left(1 + \frac{K}{K_1} \right); C_1 \max_{h \in [H]} \rho_h; q g b_h^{(1)} \leq \frac{1}{2}$$

where C_1 , and $p = 3 \frac{q}{\min_{h \in [H]} \rho_h}$ and $q = \frac{C}{\min_{h \in [H]} \rho_h} \frac{1}{\log(K_1) \log(K_1 H)} + \log(K_1 H)$

Hence, the sequence converges to $\|f_h^k g_{h=1}^H\|$ at an exponential rate. Armed with the above lemma, we finally focus on the regret of ARL (norm). We have the following:

Theorem 28. *Suppose the initial epoch length K_1 satisfies,*

$$K_1 > \max_{h \in [H]} \frac{1}{\min_{h \in [H]} \rho_h} \log_2 \left(1 + \frac{K}{K_1} \right); C_1 \max_{h \in [H]} \rho_h; q g b_h^{(1)} \leq \frac{1}{2}$$

Then, with probability exceeding $1 - 10H^{-1}$, we have

$$R(K) \leq (c_1 k + c_2) \frac{1}{\min_{h \in [H]} \rho_h} \frac{1}{(d^2 H^3 + d H^4)^{1/2}} \text{polylog}(K) \text{polylog}\left(\frac{K}{K_1}\right)^{\frac{1}{2}}$$

where $k = \max_{h \in [H]} \|f_h^k g_{h=1}^H\|$.

Remark 50. If the algorithm is run for T rounds, with $T = KH$, we have

$$R(T) \leq \frac{1}{\min_{h \in [H]} \rho_h} \frac{1}{(d^2 H^2 + d H^3)^{1/2}} \text{polylog}(K) \text{polylog}\left(\frac{K}{K_1}\right)^{\frac{1}{2}}$$

with probability at least $1 - 8H^{-2}$.

Remark 51. The regret depends on $\|f_h^k g_{h=1}^H\|$, instead of an upper-bound on the norm. Hence, our algorithm adapts to the problem complexity.

Remark 52. (Cost of model selection) Note that the regret bound here matches to that of the near-optimal regret of UCRL-VTR+, and so the model selection is (order-wise) free, in terms of regret.

10.3 Dimension as complexity measure

In this section, we consider the linear MDP, with dimension as a measure of complexity. We propose and analyze an adaptive algorithm that tailors to the sparsity of $\|f_h^k g_{h=1}^H\|$. We denote the sparsity of the parameters as $\|f_h^k g_{h=1}^H\|$, and define $d = \max_h d_h$. In what follows, we consider estimating the

Algorithm 17: Adaptive Reinforcement Learning (dim)-ARL-Dim

```

1: Input: Initial Phase length  $k_0$ , slack  $\epsilon > 0$ .
2:  $b_0 = \mathbf{1}$ ,  $T_1 = 0$ 
3: for epochs  $i = 0; 1; 2; \dots$  do
4:    $k_i = 4^i k_0$ ,  $\epsilon_i = \frac{\epsilon}{2^i}$ 
5:    $D_i := \{j : b_{ij} \geq (0.9)^{i+1} g\}$ 
6:   for times  $t \geq \lceil k_{i-1} + 1 \rceil; \dots; k_i g$  do
7:     Play UCRL-VTR+( $\epsilon = 2^i$ ) only restricted to  $(D_h^i)_{h \in [H]}$  co-ordinates.
       Here,  $\epsilon = 2^i$  is the slack parameter and 1 represents  $k_h$  for all  $h \in [H]$ .
8:   end for
9:   for times  $t \geq \lceil k_{i-1} + 1 \rceil; \dots; k_i + 2^i d^p \overline{k_0} g$  do
10:    Play UCRL-VTR+( $\epsilon = 2^i$ ) starting from where we left of in epoch  $i - 1$ .
11:  end for
12: end for

```

sparsity parameter, d_h of the h -th parameter θ_h . For notational convenience, we drop the subscript h from d_h and θ_h .

As seen in Algorithm 17, our proposed scheme works over multiple epochs, and we use diminishing thresholds to estimate the support of θ . The algorithm is parameterized by k_0 , and the slack $\epsilon \in (0, 1)$. ARL-Dim proceeds in phases numbered $0; 1; \dots$, increasing with time. Each phase starts with a support estimation routine of D_i , and is kept fixed throughout the phase.

Each phase i is divided into two blocks - (i) a regret minimization block lasting $4^i k_0$ time slots, (ii) followed by a support estimation phase lasting $2^i d^p \overline{k_0} g$ time slots. Thus, each phase i lasts for a total of $4^i k_0 + 2^i d^p \overline{k_0} g$ time slots.

At the beginning of phase $i = 0$, $D_i \subseteq [d]$ denotes the set of ‘active coordinates’, namely the estimate of the non-zero coordinates of θ , and by notation, $D_0 = [d]$. In the regret minimization block of phase i , a fresh instance of UCRL-VTR⁺ is spawned, with the dimensions restricted only to the set D_i and probability parameter $\epsilon_i := \frac{\epsilon}{2^i}$.

On the other hand, in the support estimation phase, we continue running the UCRL-VTR⁺ algorithm in full d dimension, from the point where we left of in epoch $i - 1$. Concretely, one should think of the support estimation phases over epochs as a single run of UCRL-VTR⁺ in the full d dimension. We choose the support estimation phases length in such a way that this does not affect the final regret of ARL-Dim. Furthermore, choosing the error (slack) probability halving over epochs, we ensure that final regret bound holds with high probability. At the end of each phase $i = 0$, ARL-Dim forms an estimate b_{i+1} of θ . The active coordinate set D_{i+1} , is then the coordinates of b_{i+1} with magnitude exceeding $(0.9)^{(i+1)}$. The pseudo-code is provided in Algorithm 17. By this careful choice of exploration periods and thresholds, we show that the estimated support of θ is equal to the true support, for all but finitely many phases. Thus after a finite number of epochs, the true support of θ locks-in, and thereafter the agent incurs the optimal regret that an oracle knowing the true support would incur. Hence, the extra regret we incur with respect to an oracle (which knows the support of θ) is the additive constant independent of time (depending on

the smallest non-zero value of ϵ).

10.3.1 Regret Guarantee

Suppose we choose the parameter

$$k_0 = 2 \frac{1}{\min(\epsilon)} + O \left(\frac{d^2 \log^4(K^2 H \epsilon)}{\min(\epsilon)^4} \right) ;$$

This ensures that the estimate of θ_h after the 0-th epoch, $\hat{\theta}_1$ satisfy

$$\mathbb{P} \left(\|\hat{\theta}_1 - \theta_h\| \leq \frac{2}{(0.9)^2} \right)$$

Furthermore, for all epochs $i \geq 2$, we have

$$\mathbb{P} \left(\|\hat{\theta}_i - \theta_h\| \leq \frac{2}{(0.9)^i} \right) \geq \frac{2}{2^i} ;$$

where $\hat{\theta}_i$ is the estimate of θ_h obtained by collecting all samples of the support estimation phase of UCRL-VTR⁺ algorithm until the beginning of phase i . A formal statement of the above is deferred to the proof section.

We are now ready to present the main theorem of this section. Recall that $d = \max_h d_h$.

Theorem 29. *Suppose Algorithm 17 is run with parameter k_0 chosen above and with slack $\epsilon > 0$ for K episodes. Then, with probability at least $1 - 3H \epsilon$, the regret upper bound is given by*

$$R(K) \leq \frac{1}{16} \frac{1}{\min(\epsilon)} + \frac{d^2 \log^4(K^2 H \epsilon)}{\min(\epsilon)^4} + \frac{1}{(d)^2 H^3 + d H^4 \text{polylog}(d(\frac{K}{k_0})e)} \text{polylog}(K \epsilon) \frac{1}{K}$$

where $\epsilon = \min_{h \in [H]} \min_{j: \theta_h(j) \neq 0} \theta_h(j)$, where $\theta_h(j)$ denotes the j -th coordinate of θ_h .

Remark 53. Note that we maintain a $\frac{1}{K}$ regret dependence, which is near optimal (see [74]). However, the regret depends on ϵ , and hence it is a problem instance dependent bound.

Remark 54. (Dependence on ϵ .) In the above regret bound, we haven't optimized over the dependence on ϵ . Note that the $1/\epsilon^4$ dependence can be improved significantly by choosing an aggressive growing epoch length. For example, if we choose the support estimation period as $6^i d \frac{1}{k_0} e$, the regret minimization period as $36^i k_0$ and the threshold as $(0.5)^i$, the dependence on ϵ can be significantly reduced to $1/\epsilon^{5.12}$. The support estimation period, regret minimization period and threshold selection may be kept as tuning parameters.

Remark 55. (Cost of model selection:) Note that the cost of model selection is the first term in the regret expression, which is independent of K , and hence a constant term.

10.4 Conclusion

We address the model selection problem for episodic MDP with linear function approximation. In particular, similar to the contextual bandits, we take norm and dimension as complexity parameter and propose algorithms that adapts to these. An immediate future work is to relax the linearity structure for RL, and obtain model selection for classes beyond linearity.

10.5 Proofs

10.5.1 Proof of Lemma 59

We use matrix Freedman inequality, and we critically use the bounded ness of $v_{t,h+1}(s_h^t; a_h^t)$. Note that a similar proof technique is used in [67] in the problem of contextual linear bandits.

For notational simplicity, we denote $v_{t,h+1}(s_h^t; a_h^t)$ by v_t . In particular we have the bound $\|v_t\| \leq 1$ for all t , and from construction $v_{t,h} \leq \frac{H}{d}$ for all t and h . With this, similar to [67], we construct the following matrix martingale,

$$Z_k = \sum_{t=1}^k \sum_{h=1}^d v_t v_t^\top - k c I$$

with $Z_0 = 0$, and consider the martingale difference sequence, $Y_k = Z_k - Z_{k-1}$. Since, $\|v_{t,h}\| \leq \frac{H}{d}$, we have

$$\|Y_k\|_{op} = \mathbb{E} \|Y_k\|_{op} \leq \sum_{h=1}^d \|v_k\|_{h,h} \leq \frac{H}{d};$$

and as a result

$$\|Y_k\|_{op} \leq \frac{H}{d} + \frac{H}{d} \leq 2 \frac{H}{d};$$

Furthermore, a simple calculation yields

$$\mathbb{E} \|Y_k\|_{op}^2 = \mathbb{E} \|Y_k\|_{op}^2 \leq 2 \frac{d}{H^2};$$

Now, applying matrix Freedman inequality (Theorem 13 of [67]) with $R = 2 \frac{H}{d}; u = \frac{d}{H^2}$, we obtain

$$\mathbb{P} \|Z_k\|_{op} \geq \frac{\min_k}{2};$$

for all k satisfying $k \leq \frac{16d}{2 \min H^2} + \frac{8 \frac{H}{d}}{3H \min} \log(2dK)$. Finally, using Weyl's inequality, we obtain the result.

10.5.2 Proof of Lemma 60

We consider doubling epochs, with initial epoch length K_1 and $K_i = 2^{i-1}K_1$ for $i \in \{1, \dots, N\}$. The number of epochs is N .

Let us consider the i -th epoch, and let $\hat{b}_{E_i;h}$ be the least square estimate of b_h at the end of epoch i . The confidence interval at the end of epoch i , is given by

$$C_{E_i;h} = \{k \in \mathbb{R}^d : k \in \hat{b}_{E_i;h} \pm \sqrt{\frac{\text{tr}(\hat{C}_{E_i;h})}{K_i}}\}$$

Using Lemma 59, one can rewrite $C_{E_i;h}$ as

$$C_{E_i;h} = \{k \in \mathbb{R}^d : k \in \hat{b}_{E_i;h} \pm \sqrt{\frac{\text{tr}(\hat{C}_{E_i;h})}{1 + \frac{K_i}{\min_{j=2} K_j}}}\}$$

with probability at least $1 - \frac{1}{K_i} = 2^{i-1}$. Here we use the fact that $\min_{j=2} (M_{K_j;h}) \leq 1 + \frac{K_i}{\min_{j=2} K_j}$, provided $K_i \geq \min_{j=2} (2^j)$. To ensure this condition, we take (the sufficient condition) $K_1 \geq \frac{1}{\min_{j=2} (2^j)} N$, where N is the number of epochs. Also, from the doubling principle, we obtain

$$\sum_{i=1}^N 2^{i-1} K_1 = K_1 \sum_{i=1}^N 2^{i-1} = K_1 (2^N - 1) \leq 2^N K_1$$

Hence, with K_1 satisfying $K_1 \geq \frac{1}{\min_{j=2} (2^j)} \log_2 (1 + \frac{K}{K_1})$, we ensure that $\min_{j=2} (M_{K_j;h}) \leq 1 + \frac{K}{\min_{j=2} K_j}$.

Also, we know that $b_h \in C_{E_i;h}$ with probability at least $1 - \frac{1}{K_i}$. Hence, we obtain

$$k \in \hat{b}_{E_i;h} \pm \sqrt{\frac{\text{tr}(\hat{C}_{E_i;h})}{1 + \frac{K_i}{\min_{j=2} K_j}}}$$

with probability at least $1 - \frac{1}{K_i}$. Recall from Algorithm 16 that at the end of the i -th epoch, we set the length $K_{i+1} = 2K_i$, and the estimate of k is set to

$$b_h^{(i+1)} = \max_{k \in C_{E_i;h}} k$$

From the definition of $C_{E_i;h}$, we obtain

$$b_h^{(i+1)} = \hat{b}_{E_i;h} + \sqrt{\frac{\text{tr}(\hat{C}_{E_i;h})}{1 + \frac{K_i}{\min_{j=2} K_j}}}$$

$$k \in \hat{b}_{E_i;h} \pm \sqrt{\frac{\text{tr}(\hat{C}_{E_i;h})}{1 + \frac{K_i}{\min_{j=2} K_j}}}$$

with probability exceeding $1 - 4^{-i}$. Let us look at

$$\hat{E}_{i,h} = 8 \frac{C}{d \log(1 + K_i) \log(4K_i^2 H_i)} + 4 \frac{\rho}{d \log(4K_i^2 H_i)} + \rho b_h^{(i)}$$

We now substitute $K_i = 2^{i-1} K_1$ and $H_i = \frac{1}{2^{i-1}}$. We obtain

$$\frac{\rho}{1 + \min K_i} = \frac{\rho}{2^{i-1} K_1} = 2^{\frac{i-2}{2}} \frac{\rho}{\min K_1}$$

and

$$\begin{aligned} \frac{\hat{E}_{i,h}}{1 + \min K_i} &= \frac{\rho b_h^{(i)}}{2^{\frac{i-2}{2}} \frac{\rho}{\min K_1}} \\ &+ C \frac{i}{2^{\frac{i-2}{2}} \frac{\rho}{\min K_1}} \left(\frac{\rho}{d \log(K_i) \log(K_i H_i)} + \rho \frac{1}{d \log(K_i H_i)} \right) \end{aligned}$$

Using this, we obtain

$$\begin{aligned} b_h^{(i+1)} &\leq k_h k + 3 \frac{\rho b_h^{(i)}}{2^{\frac{i-2}{2}} \frac{\rho}{\min K_1}} \\ &+ C \frac{i}{2^{\frac{i-2}{2}} \frac{\rho}{\min K_1}} \left(\frac{\rho}{d \log(K_i) \log(K_i H_i)} + \rho \frac{1}{d \log(K_i H_i)} \right) \\ &= k_h k + b_h^{(i)} \frac{\rho}{2^{\frac{i-2}{2}}} \frac{1}{K_1} + \frac{i q}{2^{\frac{i-2}{2}}} \frac{d}{K_1} \end{aligned}$$

with probability at least $1 - 4^{-i}$, where

$$p = 3 \frac{C}{\min} \quad \text{and} \quad q = \frac{C}{\min} \left(\frac{\rho}{\log(K_i) \log(K_i H_i)} + \log(K_i H_i) \right)$$

Hence, we obtain,

$$b_h^{(i+1)} \leq b_h^{(i)} k_h k + \left(1 + \frac{p}{2^{\frac{i-2}{2}}} \frac{1}{K_1} \right) b_h^{(i)} + \frac{i q}{2^{\frac{i-2}{2}}} \frac{d}{K_1}$$

By construction, $b_h^{(i)} \leq k_h k$. Hence, provided $K_1 > \frac{4p^2}{2^i} > 2p^2$, we have

$$b_h^{(i+1)} \leq b_h^{(i)} \frac{p}{2^{\frac{i-2}{2}}} \frac{1}{K_1} k_h k + \frac{i q}{2^{\frac{i-2}{2}}} \frac{d}{K_1}$$

From the above expression,

$$\sup_i b_h^{(i)} < 1$$

with probability greater than or equal to

$$1 - \prod_{i=1}^H \frac{1}{4} - \prod_{i=1}^H \frac{1}{4} = 1 - \frac{1}{4^H} - \frac{1}{4^H} > 1 - \frac{2}{4^H} > 1 - \frac{2}{8} > \frac{3}{4}.$$

. From the expression of $b_h^{(i)}$ and using the above fact in conjunction yield

$$\lim_{i \rightarrow \infty} b_h^{(i)} = k_h k.$$

However, from construction $b_h^{(i)} = k_h k$. Using this, along with the above equation, we obtain

$$\lim_{i \rightarrow \infty} b_h^{(i)} = k_h k.$$

with probability exceeding $1 - \frac{2}{8}$. So, the sequence $b_h^{(1)}; b_h^{(2)}; \dots; b_h^{(i)}$ converges to $k_h k$ with probability at least $1 - \frac{2}{8H}$ for all $h \in [H]$, and hence our successive refinement algorithm is consistent.

Rate of Convergence: Since

$$b_h^{(i)} - b_h^{(i-1)} = O\left(\frac{1}{2^i}\right); \tag{10.1}$$

with high probability, the rate of convergence of the sequence $b_h^{(i)}; g_{i=1}^1$ is exponential in the number of epochs.

Uniform upper bound on $b_h^{(i)}$ for all i : (We now compute a uniform upper bound on $b_h^{(i)}$ for all i .)

Consider the sequences $\frac{1}{2^{i-2}}$ and $\frac{1}{2^{i-2}}$, and let t_j and u_j denote the j -th term of the sequences respectively. It is easy to check that $\sup_i t_i = \frac{1}{2}$ and $\sup_i u_i = 2$, and that the sequences $\sum_{i=1}^j t_i$ and $\sum_{i=1}^j u_i$ are convergent. With this new notation, we have

$$b_h^{(2)} = k_h k + u_1 \frac{p b_h^{(1)}}{K_1} + t_1 \frac{q^{\rho-} d}{K_1};$$

with probability exceeding $1 - \frac{1}{4}$. Similarly, for $b_h^{(3)}$, we have

$$b_h^{(3)} = k_h k + u_2 \frac{p b_h^{(2)}}{K_1} + t_2 \frac{q^{\rho-} d}{K_1} \\ = \left(1 + u_2 \frac{p}{K_1}\right) k_h k + u_1 u_2 \frac{p^2}{K_1^2} b_h^{(1)} + t_1 u_2 \frac{p}{K_1} \frac{q^{\rho-} d}{K_1} + t_2 \frac{q^{\rho-} d}{K_1};$$

with probability at least $1 - 4^{-2} = 1 - 6^{-2}$. Similarly, we write expressions for $b_h^{(4)}; b_h^{(5)}; \dots$. Now, provided $K_1 \leq C_1 \max_{p,q} b_h^{(1)} d$, where C_1 is a sufficiently large constant, the expression for $b_h^{(i)}$ can be upper-bounded as

$$b_i \leq (c_1 k_h k + c_2); \tag{10.2}$$

with probability

$$\begin{aligned} & 1 - 4^{-2} - \left(\frac{1}{2} + \frac{1}{4} + \dots \text{upto } i\text{-th term} \right) \\ & 1 - 4^{-2} - \left(\frac{1}{2} + \frac{1}{4} + \dots \right) \\ & = 1 - 8^{-2}; \end{aligned}$$

Here c_1 and c_2 are constants, and are obtained from summing an infinite geometric series with decaying step size. We also use the fact that $b_1 \leq 1$, and the fact that $\frac{1}{2^i} = \frac{1}{2^{i-1}}$.

10.5.3 Proof of Theorem 28

Suppose we play Algorithm 16 for N epochs. The cumulative regret is given by

$$R(T) = \sum_{i=1}^N R(i; b^{(i)})(K_i);$$

where $R(i; b^{(i)})(K_i)$ is the cumulative regret of the UCRL-VTR⁺ in the i -th epoch. As seen (by tracking the dependence on $\frac{1}{k_h} \sum_{h=1}^H$) in UCRL-VTR⁺, the cumulative regret of UCRL-VTR⁺ is given by

$$R(i; b^{(i)})(K_i) = \mathcal{O} \left(\frac{1}{(d^2 H^3 + d H^4)^{1-2b^{(i)}}} \right) \rho \frac{1}{K_i} \text{polylog}(K_i);$$

with probability at least $1 - \frac{1}{i}$, where $b^{(i)} = \max_{h \in [H]} b_h^{(i)}$. Hence, we have

$$R(T) \leq \sum_{i=1}^N \rho \frac{1}{b^{(i)}} R(i; 1)(K_i);$$

Using Lemma 60, we obtain, with probability at least $1 - \frac{1}{8H}$,

$$R(T) \leq (c_1 k_h k + c_2) \sum_{i=1}^N R(i; 1)(K_i);$$

where $k = \max_{h \in [H]} k_h$. With the doubling trick, we have

$$K_i = 2^{i-1} K_1; \quad i = \overline{1, \dots, N}$$

Substituting, we obtain

$$R(i; 1)(K_i) \leq \frac{c_1}{(d^2 H^2 + dH^3)^{1-2}} \sum_{i=1}^N \frac{1}{K_i} \text{poly}(i) \text{polylog}(K_1)$$

with probability greater than $1 - \frac{1}{i}$.

Using the above expression, we obtain

$$R(K) \leq \frac{c_1}{(c_1 k + c_2)} \sum_{i=1}^N \frac{1}{(d^2 H^2 + dH^3)^{1-2}} \sum_{i=1}^N \frac{1}{K_i} \text{poly}(i) \text{polylog}(K_1)$$

with probability at least $1 - \frac{8H}{2}$.

Also, from the doubling principle, we obtain

$$\sum_{i=1}^N 2^{i-1} K_1 = K \implies N = \log_2 \left(1 + \frac{K}{K_1} \right)$$

Using the above expression, we obtain

$$\begin{aligned} R(K) &\leq \frac{c_1}{(c_1 k + c_2)} \sum_{i=1}^N \frac{1}{(d^2 H^3 + dH^4)^{1-2}} \sum_{i=1}^N \frac{1}{K_i} \text{poly}(i) \text{polylog}(K_1) \\ &\leq \frac{c_1}{(c_1 k + c_2)} \sum_{i=1}^N \frac{1}{(d^2 H^3 + dH^4)^{1-2}} \text{polylog}(K_1) \text{poly}(i) \sum_{i=1}^N \frac{1}{K_i} \\ &\leq \frac{c_1}{(c_1 k + c_2)} \sum_{i=1}^N \frac{1}{(d^2 H^3 + dH^4)^{1-2}} \text{polylog}(K_1) \text{poly}(N) \sum_{i=1}^N \frac{1}{K_i} \\ &\leq \frac{c_1}{(c_1 k + c_2)} \sum_{i=1}^N \frac{1}{(d^2 H^3 + dH^4)^{1-2}} \text{polylog}(K_1) \text{polylog}\left(\frac{K}{K_1}\right) \sum_{i=1}^N \frac{1}{K_i} \\ &\leq \frac{c_1}{(c_1 k + c_2)} \sum_{i=1}^N \frac{1}{(d^2 H^3 + dH^4)^{1-2}} \text{polylog}(K_1) \text{polylog}\left(\frac{K}{K_1}\right) \sum_{i=1}^N \frac{1}{K_i} \end{aligned}$$

where the last inequality follows from the fact that

$$\begin{aligned} \sum_{i=1}^N \rho \frac{1}{K_i} &= \rho \frac{1}{K_N} \left(1 + \frac{1}{2} + \frac{1}{2} + \dots \right) \\ &= \rho \frac{1}{K_N} \left(1 + \frac{1}{2} + \frac{1}{2} + \dots \right) \\ &= \frac{\rho}{2} \frac{1}{K_N} \\ &= \frac{\rho}{2} \frac{1}{K} \end{aligned}$$

The above regret bound holds with probability greater than or equal to $1 - 8H^{-2}$.

10.5.4 Proof of Theorem 29

First, we need the following set of results. Throughout the calculation, we take $\rho = \frac{1}{H}$ (1).

Lemma 61. Suppose UCRL-VTR⁺ is run for M episodes, where $\min(\frac{1}{H}, \frac{1}{K}) \leq M < K$. If $M = O\left(\frac{d \log^2(K^2 H)}{\min(\frac{1}{H}, \frac{1}{K})^2}\right)$, and $\hat{\mu}^{(M)}$ is the estimate of μ , then we have

$$P\left[\|\hat{\mu}^{(M)} - \mu\| \leq \frac{1}{2}\right] \geq 1 - 2e^{-M}$$

Proof. The proof of this comes from the analysis of UCRL-VTR⁺. Note that with Assumption 27 (and effectively Lemma 1), if we let UCRL-VTR⁺ run for M episodes, we obtain

$$\|\hat{\mu}^{(M)} - \mu\| \leq \frac{8 \rho \frac{d \log(1 + M)}{\min(\frac{1}{H}, \frac{1}{K})} \log(4K^2 H) + 4 \rho \frac{d \log(4K^2 H)}{\min(\frac{1}{H}, \frac{1}{K})} + \rho}{\min(\frac{1}{H}, \frac{1}{K})} \quad (10.3)$$

with probability at least $1 - 2e^{-M}$. Here $\min(\frac{1}{H}, \frac{1}{K}) = \frac{16d}{2 \min(H, K)^2} + \frac{8 \rho d}{3H \min(\frac{1}{H}, \frac{1}{K})} \log(2dK)$. Hence, with the choice of M , and using the fact that $M < K$, we get the lemma. \square

We shall now apply the lemma as follows. Denote by $\hat{\mu}_i$ to be the estimate of μ at the beginning of any phase i , using all the samples from random explorations in all phases less than or equal to $i - 1$.

Remark 56. The choice $\hat{\mu}_0 = \mu + O\left(\frac{d \log^2(K^2 H)}{\min(0.9)^2}\right)$ (we have added the $\min(\frac{1}{H}, \frac{1}{K})$ to make the calculations easier), ensures that

$$P\left[\|\hat{\mu}_i - \mu\| \leq 0.9 \frac{1}{2}\right] \geq 1 - 2e^{-M}$$

□

Proof of Theorem 29. We know from Corollary 8, that with probability at-least $1 - \delta$, for all phases $i \geq 2$, we have $\|j_i^0 - j_{i-1}\| \leq (0.9)^i$. Call this event E . Now, consider the phase $i(i) := \max\{2; \log_{(1-0.9)} \frac{1}{\delta}\}$. Now, when event E holds, then for all phases $i \geq i(i)$, D_i is the correct set of d non-zero coordinates of θ . Thus, with probability at-least $1 - \delta$, the total regret upto time K episodes can be upper bounded as follows

$$\begin{aligned}
 R(K) & \leq \sum_{j=0}^{i(i)-1} 4^j k_0 + 2^i d^{\rho} \frac{K}{k_0} e + \sum_{j=i(i)}^{\log_4 \frac{K}{k_0}} \text{Regret}(\text{UCRL-VTR}^+(1; i; 4^j k_0)) \\
 & \quad + \sum_{j=i(i)}^{\log_4 \frac{K}{k_0}} \text{Regret}(\text{UCRL-VTR}^+(1; i; 2^j d^{\rho} \frac{K}{k_0} e)): \quad (10.5)
 \end{aligned}$$

The term $\text{Regret}(\text{UCRL-VTR}^+(1; i; T))$ denotes the regret of the UCRL-VTR⁺ algorithm, when run with $k = k_0 4^j$, and $\delta = (0.1)^j$ denotes the probability slack and K is the time horizon. Equation (10.5) follows, since the total number of phases is at-most $O(\log_4 \frac{K}{k_0})$. Also, note that the third term in Equation (10.5) is dominated by the second term, and hence handling the first two terms is sufficient.

We now use the near optimal regret bound of [74] to upper bound the second term of Equation (10.5). In particular, for all phases $i \geq [i(i); d \log_4 \frac{K}{k_0}]$, we have, with probability at-least $1 - \delta$,

$$\text{Regret}(\text{UCRL-VTR}^+(1; i; K)) \leq O\left(\frac{\rho}{(d)^2 H^3 + d H^4} \text{polylog}(K) \right)^{\rho} \frac{K}{k_0} :$$

Thus, we know that with probability at-least $1 - \delta$, for all phases $i \geq i(i)$, the regret in the exploration phase satisfies

$$\text{Regret}(\text{UCRL-VTR}^+(1; i; 4^i k_0)) \quad (10.6)$$

$$\begin{aligned}
 & \leq O\left(\frac{\rho}{(d)^2 H^3 + d H^4}\right)^{\rho} \frac{K}{4^i k_0} \text{poly}(i) \text{polylog}(K) \\
 & \leq O\left(\frac{\rho}{(d)^2 H^3 + d H^4}\right)^{\rho} \frac{K}{4^i k_0} \text{poly}(d \log_4(\frac{K}{k_0}) e) \text{polylog}(K) \\
 & \leq C(H; K; i; d) \frac{K}{4^i k_0} \quad (10.7)
 \end{aligned}$$

$$\begin{aligned}
 C(H; K; i; d) & = O\left(\frac{\rho}{(d)^2 H^3 + d H^4} \text{poly}(d \log_4(\frac{K}{k_0}) e) \text{polylog}(K) \right)^{\rho} \\
 & = O\left(\frac{\rho}{(d)^2 H^3 + d H^4} \text{polylog}(d \log_4(\frac{K}{k_0}) e) \text{polylog}(K) \right)^{\rho}
 \end{aligned}$$

Equation the above follows, by substituting $i = O(\log_4 \frac{K}{k_0})$. We have with high probability, the (order-wise) regret is

$$\begin{aligned}
 R(K) &= 2k_0 4^{i(\cdot)} + 2 \sum_{j=0}^{\log_4 \frac{K}{k_0} + 1} C(H; K; ; d) \rho_{4^j k_0}; \\
 &= 2k_0 4^{i(\cdot)} + C(H; K; ; d) \sum_{j=0}^{\log_4 \frac{K}{k_0} + 1} \rho_{4^j k_0}; \\
 (a) \quad &= 2k_0 \frac{1}{16} + 4 \frac{\rho_{\overline{K}}}{K} + 4 \frac{\rho_{\overline{K}}}{K} C(H; K; ; d); \\
 &= \frac{1}{16} \frac{2}{\min(\cdot)} + \frac{d^2 \log^4(K^2 H =)}{\frac{2}{\min}(0.9)^4} + C(H; K; ; d) \frac{\rho_{\overline{K}}}{K} \\
 &= \frac{1}{16} \frac{2}{\min(\cdot)} + \frac{d^2 \log^4(K^2 H =)}{\frac{2}{\min}(0.9)^4} \\
 &+ \frac{\rho_{\overline{K}}}{(d)^2 H^3 + d H^4} \text{polylog}(d(\frac{K}{k_0})e) \text{polylog}(K =) \frac{\rho_{\overline{K}}}{K}
 \end{aligned}$$

Step (a) follows from 4.1.16.

Let us now compute the probability with which the above holds. Note that the support estimation procedure succeeds with probability at least $1 - H$. Furthermore, we have 2 parallel runs of UCRL-VTR+, each of which succeeds with probability at least $1 - \frac{1}{2} - \frac{1}{2} = 1 - 1$. Hence, with probability at least $1 - 3H$, the above regret bound holds.

□

Bibliography

- [1] H Brendan McMahan et al. “Communication-efficient learning of deep networks from decentralized data”. In: *arXiv preprint arXiv:1602.05629* (2016).
- [2] Jakub Konečný et al. “Federated optimization: distributed machine learning for on-device intelligence”. In: *arXiv preprint arXiv:1610.02527* (2016).
- [3] Brendan McMahan and Daniel Ramage. *Federated Learning: Collaborative Machine Learning without Centralized Training Data*. <https://research.google.com/2017/04/federated-learning-collaborative.html>. 2017.
- [4] Jacob Devlin et al. “BERT: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).
- [5] Leslie Lamport, Robert Shostak, and Marshall Pease. “The Byzantine Generals Problem”. In: *ACM Trans. Program. Lang. Syst.* 4.3 (July 1982), pp. 382–401.
- [6] Virginia Smith et al. “Federated multi-task learning”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 4424–4434.
- [7] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. “Clustered Federated Learning: Model-Agnostic Distributed Multi-Task Optimization under Privacy Constraints”. In: *arXiv preprint arXiv:1910.01991* (2019).
- [8] Yihan Jiang et al. “Improving federated learning personalization via model agnostic meta learning”. In: *arXiv preprint arXiv:1909.12488* (2019).
- [9] Yishay Mansour et al. “Three approaches for personalization with applications to federated learning”. In: *arXiv preprint arXiv:2002.10619* (2020).
- [10] Badrul M Sarwar et al. “Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering”. In: *Proceedings of the fifth international conference on computer and information technology*. Vol. 1. 2002, pp. 291–324.
- [11] Qing Li and Byeong Man Kim. “Clustering approach for hybrid recommender system”. In: *Proceedings IEEE/WIC International Conference on Web Intelligence (WI 2003)*. IEEE. 2003, pp. 33–38.
- [12] Rich Caruana. “Multitask learning”. In: *Machine learning* 28.1 (1997), pp. 41–75.
- [13] Martin J Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*. Vol. 48. Cambridge University Press, 2019.

- [14] Dan Alistarh et al. “The convergence of sparsified gradient methods”. In: *Advances in Neural Information Processing Systems*. 2018, pp. 5973–5983.
- [15] Sebastian U Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. “Sparsified sgd with memory”. In: *Advances in Neural Information Processing Systems*. 2018, pp. 4447–4458.
- [16] Nikita Ivkin et al. “Communication-efficient distributed SGD with Sketching”. In: *arXiv preprint arXiv:1903.04488* (2019).
- [17] Dan Alistarh et al. “Communication-efficient stochastic gradient descent, with applications to neural networks”. In: (2017).
- [18] Lili Su and Nitin H Vaidya. “Fault-tolerant multi-agent optimization: optimal iterative distributed algorithms”. In: *Proceedings of the 2016 ACM symposium on principles of distributed computing*. ACM. 2016, pp. 425–434.
- [19] Jiashi Feng, Huan Xu, and Shie Mannor. “Distributed robust learning”. In: *arXiv preprint arXiv:1409.5937* (2014).
- [20] Sai Praneeth Karimireddy et al. “Error feedback fixes signsgd and other gradient compression schemes”. In: *International Conference on Machine Learning*. PMLR. 2019, pp. 3252–3261.
- [21] Dan Alistarh et al. “QSGD: Communication-efficient SGD via gradient quantization and encoding”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 1709–1720.
- [22] Jeremy Bernstein et al. “signsgd with majority vote is communication efficient and Byzantine fault tolerant”. In: *arXiv preprint arXiv:1810.05291* (2018).
- [23] Dong Yin et al. “Byzantine-Robust Distributed Learning: Towards Optimal Statistical Rates”. In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. Stockholmsmässan, Stockholm Sweden: PMLR, Oct. 2018, pp. 5650–5659.
- [24] Ohad Shamir, Nati Srebro, and Tong Zhang. “Communication-efficient distributed optimization using an approximate newton-type method”. In: *International conference on machine learning*. 2014, pp. 1000–1008.
- [25] Yuchen Zhang and Xiao Lin. “Disco: Distributed optimization for self-concordant empirical loss”. In: *International conference on machine learning*. 2015, pp. 362–370.
- [26] Shusen Wang et al. “GIANT: Globally improved approximate Newton method for distributed optimization”. In: *Advances in Neural Information Processing Systems*. 2018, pp. 2332–2342.
- [27] Rixon Crane and Fred Roosta. “DINGO: Distributed Newton-Type Method for Gradient-Norm Optimization”. In: *Advances in Neural Information Processing Systems*. 2019, pp. 9494–9504.
- [28] Fred Roosta et al. “Newton-MR: Newton’s Method Without Smoothness or Convexity”. In: *arXiv preprint arXiv:1810.00303* (2018).

- [29] Sashank J Reddi et al. “Aide: Fast and communication efficient distributed optimization”. In: *arXiv preprint arXiv:1608.06879* (2016).
- [30] Yudong Chen, Lili Su, and Jiaming Xu. “Distributed statistical machine learning in adversarial settings: Byzantine gradient descent”. In: *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 1.2 (2017), p. 44.
- [31] Dong Yin et al. “Defending Against Saddle Point Attack in Byzantine-Robust Distributed Learning”. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, Sept. 2019, pp. 7074–7084.
- [32] Avishek Ghosh et al. “Robust federated learning in a heterogeneous environment”. In: *arXiv preprint arXiv:1906.06629* (2019).
- [33] Peva Blanchard et al. “Byzantine-tolerant machine learning”. In: *arXiv:1703.02757* (2017).
- [34] Michal Derezhinski and Michael W Mahoney. “Distributed estimation of the inverse Hessian by determinantal averaging”. In: *Advances in Neural Information Processing Systems*. 2019, pp. 11401–11411.
- [35] Daniel Soudry and Yair Carmon. *No bad local minima: Data independent training error guarantees for multilayer neural networks*. 2016. arXiv: 1605. 08361 [stat. ML].
- [36] Rong Ge, Chi Jin, and Yi Zheng. “No Spurious Local Minima in Nonconvex Low Rank Problems: A Unified Geometric Analysis”. In: *Proceedings of the 34th International Conference on Machine Learning*. Vol. 70. Proceedings of Machine Learning Research. International Convention Centre, Sydney, Australia: PMLR, June 2017, pp. 1233–1242.
- [37] Kenji Kawaguchi. “Deep Learning without Poor Local Minima”. In: *Advances in Neural Information Processing Systems*. Ed. by D. Lee et al. Vol. 29. Curran Associates, Inc., 2016, pp. 586–594.
- [38] Prateek Jain et al. *Global Convergence of Non-Convex Gradient Descent for Computing Matrix Squareroot*. 2017. arXiv: 1507. 05854 [math. NA].
- [39] Ju Sun, Qing Qu, and John Wright. “A Geometric Analysis of Phase Retrieval”. In: *CoRR* abs/1602.06664 (2016). arXiv: 1602. 06664.
- [40] Yann N Dauphin et al. “Identifying and attacking the saddle point problem in high-dimensional non-convex optimization”. In: *Advances in Neural Information Processing Systems*. Vol. 27. 2014, pp. 2933–2941.
- [41] Srinadh Bhojanapalli, Behnam Neyshabur, and Nathan Srebro. *Global Optimality of Local Search for Low Rank Matrix Recovery*. 2016. arXiv: 1605. 07221 [stat. ML].
- [42] Ju Sun, Qing Qu, and John Wright. “Complete Dictionary Recovery Over the Sphere I: Overview and the Geometric Picture”. In: *IEEE Transactions on Information Theory* 63.2 (Feb. 2017), pp. 853–884.
- [43] Yurii Nesterov and Boris T Polyak. “Cubic regularization of Newton method and its global performance”. In: *Mathematical Programming* 108.1 (2006), pp. 177–205.

- [44] Jonas Moritz Kohler and Aurelien Lucchi. “Sub-sampled cubic regularization for non-convex optimization”. In: *International Conference on Machine Learning*. PMLR. 2017, pp. 1895–1904.
- [45] Zhe Wang et al. “Cubic regularization with momentum for nonconvex optimization”. In: *Uncertainty in Artificial Intelligence*. PMLR. 2020, pp. 313–322.
- [46] Chi Jin et al. “How to escape saddle points efficiently”. In: *International Conference on Machine Learning*. PMLR. 2017, pp. 1724–1732.
- [47] Chih-Chung Chang and Chih-Jen Lin. “LIBSVM: A library for support vector machines”. In: *ACM transactions on intelligent systems and technology (TIST)* 2.3 (2011), p. 27.
- [48] Robert W Harrison. “Phase problem in crystallography”. In: *JOSA a* 10.5 (1993), pp. 1046–1055.
- [49] C Fienup and J Dainty. “Phase retrieval and image reconstruction for astronomy”. In: *Image Recovery: Theory and Application* 231 (1987), p. 275.
- [50] Anwei Chai, Miguel Moscoso, and George Papanicolaou. “Array imaging using intensity-only measurements”. In: *Inverse Problems* 27.1 (2010), p. 015005.
- [51] Fajwel Fogel, Irène Waldspurger, and Alexandre d’Aspremont. “Phase retrieval for imaging problems”. In: *Mathematical programming computation* 8.3 (2016), pp. 311–335.
- [52] EM Bronshtein. “ ϵ -entropy of convex sets and functions”. In: *Siberian Mathematical Journal* 17.3 (1976), pp. 393–398.
- [53] Adityanand Guntuboyina and Bodhisattva Sen. “Covering numbers for convex functions”. In: *IEEE Transactions on Information Theory* 59.4 (2013), pp. 1957–1965.
- [54] Fuchang Gao and Jon A Wellner. “Entropy of Convex Functions on \mathbb{R}^d ”. In: *Constructive approximation* 46.3 (2017), pp. 565–592.
- [55] Alessandro Magnani and Stephen P Boyd. “Convex piecewise-linear fitting”. In: *Optimization and Engineering* 10.1 (2009), pp. 1–17.
- [56] Yuxin Chen and Emmanuel Candes. “Solving random quadratic systems of equations is nearly as easy as solving linear systems”. In: *Advances in Neural Information Processing Systems*. 2015, pp. 739–747.
- [57] Grigoris Paouris. “Small ball probability estimates for log-concave measures”. In: *Transactions of the American Mathematical Society* 364.1 (2012), pp. 287–308.
- [58] Mark Rudelson and Roman Vershynin. “Small ball probabilities for linear images of high-dimensional distributions”. In: *International Mathematics Research Notices* 2015.19 (2014), pp. 9594–9617.
- [59] Irene Waldspurger. “Phase Retrieval With Random Gaussian Sensing Vectors by Alternating Projections”. In: *IEEE Transactions on Information Theory* 64.5 (May 2018), pp. 3301–3312.

- [60] Teng Zhang. “Phase retrieval using alternating minimization in a batch setting”. In: *Applied and Computational Harmonic Analysis* (2019).
- [61] Praneeth Netrapalli, Prateek Jain, and Sujay Sanghavi. “Phase retrieval using alternating minimization”. In: *Advances in Neural Information Processing Systems*. 2013, pp. 2796–2804.
- [62] Tze Leung Lai and Herbert Robbins. “Asymptotically efficient adaptive allocation rules”. In: *Advances in applied mathematics* 6.1 (1985), pp. 4–22.
- [63] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. “Finite-time analysis of the multiarmed bandit problem”. In: *Machine learning* 47.2-3 (2002), pp. 235–256.
- [64] Wei Chu et al. “Contextual bandits with linear payoff functions”. In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. 2011, pp. 208–214.
- [65] Varsha Dani, Thomas P Hayes, and Sham M Kakade. “Stochastic linear optimization under bandit feedback”. In: (2008).
- [66] Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. “Improved algorithms for linear stochastic bandits”. In: *Advances in Neural Information Processing Systems*. 2011, pp. 2312–2320.
- [67] Niladri S Chatterji, Vidya Muthukumar, and Peter L Bartlett. “Osom: A simultaneously optimal algorithm for multi-armed and linear contextual bandits”. In: *arXiv preprint arXiv:1905.10040* (2019).
- [68] Alexandra Carpentier and Rémi Munos. “Bandit theory meets compressed sensing for high dimensional stochastic linear bandit”. In: *Artificial Intelligence and Statistics*. 2012, pp. 190–198.
- [69] Hamsa Bastani and Mohsen Bayati. “Online decision making with high-dimensional covariates”. In: *Operations Research* 68.1 (2020), pp. 276–294.
- [70] Dylan J Foster, Akshay Krishnamurthy, and Haipeng Luo. “Model selection for contextual bandits”. In: *Advances in Neural Information Processing Systems*. 2019, pp. 14714–14725.
- [71] David Simchi-Levi and Yunzong Xu. *Bypassing the Monster: A Faster and Simpler Optimal Algorithm for Contextual Bandits under Realizability*. 2020. arXiv: 2003. 12699 [cs. LG].
- [72] Alekh Agarwal et al. “Contextual bandit learning with predictable rewards”. In: *Artificial Intelligence and Statistics*. PMLR. 2012, pp. 19–26.
- [73] Zeyu Jia et al. “Model-based reinforcement learning with value-targeted regression”. In: *Learning for Dynamics and Control*. PMLR. 2020, pp. 666–686.
- [74] Dongruo Zhou, Quanquan Gu, and Csaba Szepesvari. “Nearly Minimax Optimal Reinforcement Learning for Linear Mixture Markov Decision Processes”. In: *arXiv preprint arXiv:2012.08507* (2020).

- [75] Jonathan Lee et al. “Online model selection for reinforcement learning with function approximation”. In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2021, pp. 3340–3348.
- [76] Sebastian Caldas et al. “Leaf: A benchmark for federated settings”. In: *arXiv preprint arXiv:1812.01097* (2018).
- [77] Wayne S DeSarbo and William L Cron. “A maximum likelihood methodology for cluster-wise linear regression”. In: *Journal of classification* 5.2 (1988), pp. 249–282.
- [78] Yuekai Sun, Stratis Ioannidis, and Andrea Montanari. “Learning Mixtures of Linear Classifiers.” In: *ICML*. 2014, pp. 721–729.
- [79] Martin Zinkevich et al. “Parallelized stochastic gradient descent”. In: *Advances in neural information processing systems*. 2010, pp. 2595–2603.
- [80] Benjamin Recht et al. “Hogwild: A lock-free approach to parallelizing stochastic gradient descent”. In: *Advances in neural information processing systems*. 2011, pp. 693–701.
- [81] Mu Li et al. “Scaling distributed machine learning with the parameter server”. In: *11th USENIX Symposium on Operating Systems Design and Implementation (fOSDIg 14)*. 2014, pp. 583–598.
- [82] Andrew Hard et al. “Federated learning for mobile keyboard prediction”. In: *arXiv preprint arXiv:1811.03604* (2018).
- [83] Yue Zhao et al. “Federated Learning with Non-IID Data”. In: *arXiv:1806.00582* (2018).
- [84] Anit Kumar Sahu et al. “On the Convergence of Federated Optimization in Heterogeneous Networks”. In: *CoRR abs/1812.06127* (2018). arXiv: 1812. 06127.
- [85] Liping Li et al. “RSA: Byzantine-Robust Stochastic Aggregation Methods for Distributed Learning from Heterogeneous Datasets”. In: *CoRR abs/1811.03761* (2018). arXiv: 1811. 03761.
- [86] Felix Sattler et al. “Robust and Communication-Efficient Federated Learning from Non-IID Data”. In: *CoRR abs/1903.02891* (2019). arXiv: 1903. 02891.
- [87] Xiang Li et al. “On the convergence of fedavg on non-iid data”. In: *arXiv:1907.02189* (2019).
- [88] Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. “Agnostic Federated Learning”. In: *CoRR abs/1902.00146* (2019). arXiv: 1902. 00146.
- [89] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. “Personalized federated learning: A meta-learning approach”. In: *arXiv preprint arXiv:2002.07948* (2020).
- [90] Fei Chen et al. “Federated Meta-Learning with Fast Convergence and Efficient Communication”. In: *arXiv preprint arXiv:1802.07876* (2018).
- [91] Lei Xu and Michael I Jordan. “On convergence properties of the EM algorithm for Gaussian mixtures”. In: *Neural computation* 8.1 (1996), pp. 129–151.

- [92] Dar-Shyang Lee. “Effective Gaussian mixture learning for video background subtraction”. In: *IEEE transactions on pattern analysis and machine intelligence* 27.5 (2005), pp. 827–832.
- [93] Michel Wedel and Wagner A Kamakura. “Mixture regression models”. In: *Market segmentation*. Springer, 2000, pp. 101–124.
- [94] Dong Yin et al. “Learning mixtures of sparse linear regressions using sparse graph codes”. In: *IEEE Transactions on Information Theory* 65.3 (2018), pp. 1430–1451.
- [95] James R Fienup. “Phase retrieval algorithms: a comparison”. In: *Applied optics* 21.15 (1982), pp. 2758–2769.
- [96] Rick P Millane. “Phase retrieval in crystallography and optics”. In: *JOSA A* 7.3 (1990), pp. 394–411.
- [97] Constantinos Daskalakis, Christos Tzamos, and Manolis Zampetakis. “Ten steps of EM suffice for mixtures of two Gaussians”. In: *arXiv preprint arXiv:1609.00368* (2016).
- [98] Xinyang Yi, Constantine Caramanis, and Sujay Sanghavi. “Solving a mixture of many random linear equations by tensor decomposition and alternating minimization”. In: *arXiv preprint arXiv:1608.05749* (2016).
- [99] Sivaraman Balakrishnan, Martin J Wainwright, and Bin Yu. “Statistical guarantees for the EM algorithm: From population to sample-based analysis”. In: *The Annals of Statistics* 45.1 (2017), pp. 77–120.
- [100] Irene Waldspurger. “Phase retrieval with random gaussian sensing vectors by alternating projections”. In: *IEEE Transactions on Information Theory* 64.5 (2018), pp. 3301–3312.
- [101] Avishek Ghosh and Kannan Ramchandran. “Alternating Minimization Converges Super-Linearly for Mixed Linear Regression”. In: *arXiv preprint arXiv:2004.10914* (2020).
- [102] Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. “Low-rank matrix completion using alternating minimization”. In: *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*. ACM, 2013, pp. 665–674.
- [103] Xinyang Yi, Constantine Caramanis, and Sujay Sanghavi. “Alternating minimization for mixed linear regression”. In: *International Conference on Machine Learning*. 2014, pp. 613–621.
- [104] Bowei Yan, Mingzhang Yin, and Purnamrita Sarkar. “Convergence of gradient EM on multi-component mixture of Gaussians”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 6956–6966.
- [105] Ofer Dekel et al. “Optimal distributed online prediction using mini-batches”. In: *Journal of Machine Learning Research* 13.Jan (2012), pp. 165–202.
- [106] Yann LeCun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

- [107] Alex Krizhevsky and Geoffrey Hinton. “Learning multiple layers of features from tiny images”. In: *Technical Report* (2009).
- [108] David Lopez-Paz and Marc’Aurelio Ranzato. “Gradient episodic memory for continual learning”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 6467–6476.
- [109] Ian J Goodfellow et al. “An empirical investigation of catastrophic forgetting in gradient-based neural networks”. In: *arXiv preprint arXiv:1312.6211* (2013).
- [110] James Kirkpatrick et al. “Overcoming catastrophic forgetting in neural networks”. In: *Proceedings of the national academy of sciences* 114.13 (2017), pp. 3521–3526.
- [111] Roman Vershynin. “Introduction to the non-asymptotic analysis of random matrices”. In: *arXiv preprint arXiv:1011.3027* (2010).
- [112] Mark Rudelson and Roman Vershynin. “Hanson-Wright inequality and sub-Gaussian concentration”. In: *Electronic Communications in Probability* 18 (2013).
- [113] Ananda Theertha Suresh et al. “Distributed mean estimation with limited communication”. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 3329–3337.
- [114] Hongyi Wang et al. “Atomo: Communication-efficient learning via atomic sparsification”. In: *Advances in Neural Information Processing Systems*. 2018, pp. 9850–9861.
- [115] Wei Wen et al. “Terngrad: Ternary gradients to reduce communication in distributed deep learning”. In: *Advances in neural information processing systems*. 2017, pp. 1509–1519.
- [116] Venkata Gandikota, Raj Kumar Maity, and Arya Mazumdar. “vqSGD: Vector Quantized Stochastic Gradient Descent”. In: *arXiv preprint arXiv:1911.07971* (2019).
- [117] Jeremy Bernstein et al. “signSGD: Compressed optimisation for non-convex problems”. In: *arXiv preprint arXiv:1802.04434* (2018).
- [118] Nikko Strom. “Scalable distributed DNN training using commodity GPU cloud computing”. In: *Sixteenth Annual Conference of the International Speech Communication Association*. 2015.
- [119] Frank Seide et al. “1-bit stochastic gradient descent and its application to data-parallel distributed training of speech dnns”. In: *Fifteenth Annual Conference of the International Speech Communication Association*. 2014.
- [120] Yuchen Zhang et al. “Information-theoretic lower bounds for distributed statistical estimation with communication constraints”. In: *Advances in Neural Information Processing Systems*. 2013, pp. 2328–2336.
- [121] Samuel Horvath et al. “Stochastic Distributed Learning with Gradient Quantization and Variance Reduction”. In: *arXiv e-prints*, arXiv:1904.05115 (Apr. 2019), arXiv:1904.05115. arXiv: 1904. 05115.
- [122] El Mahdi El Mhamdi, Rachid Guerraoui, and Sébastien Rouault. “The hidden vulnerability of distributed learning in byzantium”. In: *arXiv preprint arXiv:1802.07927* (2018).

- [123] Georgios Damaskinos et al. “AGGREGATHOR: Byzantine Machine Learning via Robust Gradient Aggregation”. In: (2019). Published in the Conference on Systems and Machine Learning (SysML) 2019, Stanford, CA, USA., p. 19.
- [124] *AWS News Blog*. <https://tinyurl.com/yxe4hu4w>. Accessed: 2019-10-08.
- [125] K. Lee et al. “Coded computation for multicore setups”. In: *2017 IEEE International Symposium on Information Theory (ISIT)*. June 2017, pp. 2413–2417.
- [126] Paolo Costa et al. “R2C2: A Network Stack for Rack-scale Computers”. In: *SIGCOMM 2015*. ACM - Association for Computing Machinery, Aug. 2015.
- [127] Simon Haykin. *An introduction to analog and digital communication*. John Wiley, 1994.
- [128] Sai Praneeth Karimireddy et al. “SCAFFOLD: Stochastic controlled averaging for on-device federated learning”. In: *arXiv preprint arXiv:1910.06378* (2019).
- [129] Prathamesh Mayekar and Himanshu Tyagi. “Limits on Gradient Compression for Stochastic Optimization”. In: *arXiv preprint arXiv:2001.09032* (2020).
- [130] Shusen Wang, Alex Gittens, and Michael W Mahoney. “Sketched ridge regression: Optimization perspective, statistical perspective, and model averaging”. In: *The Journal of Machine Learning Research* 18.1 (2017), pp. 8039–8088.
- [131] Vipul Gupta et al. “Oversketching newton: Fast convex optimization for serverless systems”. In: *arXiv preprint arXiv:1903.08857* (2019).
- [132] Mert Pilanci and Martin J Wainwright. “Newton sketch: A near linear-time optimization algorithm with linear-quadratic convergence”. In: *SIAM Journal on Optimization* 27.1 (2017), pp. 205–245.
- [133] Petros Drineas et al. “Fast approximation of matrix coherence and statistical leverage”. In: *Journal of Machine Learning Research* 13.Dec (2012), pp. 3475–3506.
- [134] Petros Drineas and Michael W Mahoney. “RandNLA: randomized numerical linear algebra”. In: *Communications of the ACM* 59.6 (2016), pp. 80–90.
- [135] Michael W Mahoney. “Randomized algorithms for matrices and data”. In: *Foundations and Trends® in Machine Learning* 3.2 (2011), pp. 123–224.
- [136] Swarm2. *Swarm User Documentation*. <https://people.cs.umass.edu/~swarm/index.php?n=Main.NewSwarmDoc>. Accessed: 2018-01-05. 2018.
- [137] Anna Choromanska et al. *The Loss Surfaces of Multilayer Networks*. 2015. arXiv: 1412.0233 [cs. LG].
- [138] Yi Xu, Rong Jin, and Tianbao Yang. “First-order stochastic algorithms for escaping from saddle points in almost linear time”. In: *arXiv preprint arXiv:1711.01944* (2017).
- [139] Zeyuan Allen-Zhu and Yuanzhi Li. “Neon2: Finding local minima via first-order oracles”. In: *arXiv preprint arXiv:1711.06673* (2017).
- [140] Avishek Ghosh et al. “Communication-efficient and byzantine-robust distributed learning”. In: *2020 Information Theory and Applications Workshop (ITA)*. IEEE. 2020, pp. 1–28.

- [141] Avishek Ghosh, Raj Kumar Maity, and Arya Mazumdar. “Distributed Newton Can Communicate Less and Resist Byzantine Workers”. In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. 2020.
- [142] Jason D Lee et al. “Gradient descent converges to minimizers”. In: *arXiv:1602.04915* (2016).
- [143] Jason D Lee et al. “First-order methods almost always avoid saddle points”. In: *arXiv preprint arXiv:1710.07406* (2017).
- [144] Simon S Du et al. “Gradient descent can take exponential time to escape saddle points”. In: *arXiv preprint arXiv:1705.10412* (2017).
- [145] Coralia Cartis, Nicholas IM Gould, and Philippe L Toint. “Adaptive cubic regularisation methods for unconstrained optimization. Part I: motivation, convergence and numerical results”. In: *Mathematical Programming* 127.2 (2011), pp. 245–295.
- [146] Coralia Cartis, Nicholas IM Gould, and Philippe L Toint. “Adaptive cubic regularisation methods for unconstrained optimization. Part II: worst-case function-and derivative-evaluation complexity”. In: *Mathematical programming* 130.2 (2011), pp. 295–319.
- [147] N Tripuraneni et al. “Stochastic cubic regularization for fast nonconvex optimization”. In: *Advances in Neural Information Processing Systems*. 2018, pp. 2899–2908.
- [148] Dongruo Zhou, Pan Xu, and Quanquan Gu. “Stochastic variance-reduced cubic regularized Newton methods”. In: *International Conference on Machine Learning*. PMLR. 2018, pp. 5990–5999.
- [149] Zhe Wang et al. “Stochastic variance-reduced cubic regularization for nonconvex optimization”. In: *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR. 2019, pp. 2731–2740.
- [150] Yair Carmon and John C Duchi. “Gradient descent efficiently finds the cubic-regularized non-convex newton step”. In: *arXiv preprint arXiv:1612.00547* (2016).
- [151] Naman Agarwal et al. “Finding approximate local minima faster than gradient descent”. In: *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*. 2017, pp. 1195–1199.
- [152] Qiyang Han and Jon A Wellner. “Multivariate convex regression: global risk bounds and adaptation”. In: *arXiv preprint arXiv:1601.06844* (2016).
- [153] Lauren A Hannah and David B Dunson. “Multivariate convex regression with adaptive partitioning”. In: *The Journal of Machine Learning Research* 14.1 (2013), pp. 3261–3294.
- [154] Gábor Balázs. “Convex regression: theory, practice, and applications”. PhD thesis. University of Alberta, 2016.
- [155] Jerry Ladd Prince and Alan S Willsky. “Reconstructing convex sets from support line measurements”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 12.4 (1990), pp. 377–389.

- [156] Jens Gregor and Fernando R Rannou. “Three-dimensional support function estimation and application for projection magnetic resonance imaging”. In: *International journal of imaging systems and technology* 12.1 (2002), pp. 43–50.
- [157] Richard J. Gardner. *Geometric Tomography*. 2nd ed. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2006.
- [158] Adityanand Guntuboyina. “Optimal rates of convergence for convex set estimation from support functions”. In: *The Annals of Statistics* 40.1 (2012), pp. 385–411.
- [159] Yong Sheng Soh and Venkat Chandrasekaran. “Fitting Tractable Convex Sets to Support Function Evaluations”. In: *arXiv preprint arXiv:1903.04194* (2019).
- [160] Sara A van de Geer. *Regression analysis and empirical processes*. Vol. 45. CWI Tract. Stichting Mathematisch Centrum, Centrum voor Wiskunde en Informatica, Amsterdam, 1988, pp. vi+161.
- [161] Matthew Fickus et al. “Phase retrieval from very few measurements”. In: *Linear Algebra and its Applications* 449 (2014), pp. 475–499.
- [162] Emilio Seijo and Bodhisattva Sen. “Nonparametric least squares estimation of a multivariate convex regression function”. In: *The Annals of Statistics* 39.3 (2011), pp. 1633–1657.
- [163] Eunji Lim and Peter W Glynn. “Consistency of multidimensional convex regression”. In: *Operations Research* 60.1 (2012), pp. 196–208.
- [164] Rahul Mazumder et al. “A computational framework for multivariate convex regression and its variants”. In: *Journal of the American Statistical Association* 114.525 (2019), pp. 318–331.
- [165] R Gerchberg and W Saxton. “A practical algorithm for the determination of phase from image and diffraction plane pictures”. In: *Optik* 35 (1972), pp. 237–246.
- [166] Evelyn ML Beale and Roderick JA Little. “Missing values in multivariate analysis”. In: *Journal of the Royal Statistical Society: Series B (Methodological)* 37.1 (1975), pp. 129–145.
- [167] Herman O Hartley. “Maximum likelihood estimation from incomplete data”. In: *Biometrics* 14.2 (1958), pp. 174–194.
- [168] Michael I Jordan and Robert A Jacobs. “Hierarchical mixtures of experts and the EM algorithm”. In: *Neural computation* 6.2 (1994), pp. 181–214.
- [169] Arun Tejasvi Chaganty and Percy Liang. “Spectral experts for estimating mixtures of linear regressions”. In: *International Conference on Machine Learning*. 2013, pp. 1040–1048.
- [170] CF Jeff Wu. “On the convergence properties of the EM algorithm”. In: *The Annals of statistics* 11.1 (1983), pp. 95–103.
- [171] Paul Tseng. “An analysis of the EM algorithm and entropy-like proximal point methods”. In: *Mathematics of Operations Research* 29.1 (2004), pp. 27–44.

- [172] Stéphane Chrétien and Alfred O Hero. “On EM algorithms and their proximal generalizations”. In: *ESAIM: Probability and Statistics* 12 (2008), pp. 308–326.
- [173] Yuchen Zhang et al. “Spectral methods meet EM: A provably optimal algorithm for crowd-sourcing”. In: *Advances in neural information processing systems*. 2014, pp. 1260–1268.
- [174] Kai Zhong, Prateek Jain, and Inderjit S Dhillon. “Mixed linear regression with multiple components”. In: *Advances in neural information processing systems*. 2016, pp. 2190–2198.
- [175] Chi Jin et al. “Local maxima in the likelihood of Gaussian mixture models: Structural results and algorithmic consequences”. In: *Advances in neural information processing systems*. 2016, pp. 4116–4124.
- [176] Constantinos Daskalakis, Christos Tzamos, and Manolis Zampetakis. “Ten Steps of EM Suffice for Mixtures of Two Gaussians”. In: *30th Annual Conference on Learning Theory*. 2017.
- [177] Ji Xu, Daniel J Hsu, and Arian Maleki. “Global analysis of expectation maximization for mixtures of two Gaussians”. In: *Advances in Neural Information Processing Systems*. 2016, pp. 2676–2684.
- [178] Jeongyeol Kwon et al. “Global Convergence of EM Algorithm for Mixtures of Two Component Linear Regression”. In: *arXiv preprint arXiv:1810.05752v3* (2018).
- [179] Daniel Hsu and Sham M Kakade. “Learning mixtures of spherical Gaussians: moment methods and spectral decompositions”. In: *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*. ACM. 2013, pp. 11–20.
- [180] Constantine Caramanis Xinyang Yi and Sujay Sanghavi. “Solving a Mixture of Many Random Linear Equations by Tensor Decomposition and Alternating Minimization”. In: (2016).
- [181] Yan Shuo Tan and Roman Vershynin. “Phase Retrieval via Randomized Kaczmarz: Theoretical Guarantees”. In: *arXiv e-prints*, arXiv:1706.09993 (June 2017), arXiv:1706.09993. arXiv: 1706. 09993 [math. NA].
- [182] Georges M Tallis. “The moment generating function of the truncated multi-normal distribution”. In: *Journal of the Royal Statistical Society: Series B (Methodological)* 23.1 (1961), pp. 223–229.
- [183] Yanyao Shen and Sujay Sanghavi. “Iterative Least Trimmed Squares for Mixed Linear Regression”. In: *arXiv preprint arXiv:1902.03653* (2019).
- [184] Guillaume Lecué and Shahar Mendelson. “Minimax rate of convergence and the performance of ERM in phase recovery”. In: *arXiv preprint arXiv:1311.5024* (2013).
- [185] Yong Sheng Soh. “Fitting Convex Sets to Data: Algorithms and Applications”. PhD thesis. California Institute of Technology, 2019.
- [186] Raghunandan H Keshavan, Andrea Montanari, and Sewoong Oh. “Matrix completion from a few entries”. In: *IEEE transactions on information theory* 56.6 (2010), pp. 2980–2998.

- [187] Hanie Sedghi, Majid Janzamin, and Anima Anandkumar. “Provable tensor methods for learning mixtures of generalized linear models”. In: *Proceedings of Machine Learning Research* 51 (2014), pp. 1223–1231.
- [188] Kai Zhong et al. “Recovery guarantees for one-hidden-layer neural networks”. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 4140–4149.
- [189] Amit Daniely, Sivan Sabato, and Shai S Shwartz. “Multiclass learning approaches: A theoretical comparison with implications”. In: *Advances in Neural Information Processing Systems*. 2012, pp. 485–493.
- [190] Dmitry Babichev and Francis Bach. “Slice inverse regression with score functions”. In: *Electronic Journal of Statistics* 12.1 (2018), pp. 1507–1543.
- [191] Surbhi Goel et al. “Reliably learning the ReLU in polynomial time”. In: *Conference on Learning Theory*. PMLR. 2017, pp. 1004–1042.
- [192] Adityanand Guntuboyina and Bodhisattva Sen. “Nonparametric shape-restricted regression”. In: *Statistical Science* 33.4 (2018), pp. 568–594.
- [193] Adityanand Guntuboyina and Bodhisattva Sen. “Global risk bounds and adaptation in univariate convex regression”. In: *Probability Theory and Related Fields* 163.1-2 (2015), pp. 379–411.
- [194] Pierre C. Bellec. “Sharp oracle inequalities for Least Squares estimators in shape restricted regression”. In: *Ann. Statist.* 46.2 (Apr. 2018), pp. 745–780.
- [195] Andres M Medina and Mehryar Mohri. “Learning theory and algorithms for revenue optimization in second price auctions with reserve”. In: *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. 2014, pp. 262–270.
- [196] Jamie Morgenstern and Tim Roughgarden. “Learning simple auctions”. In: *Conference on Learning Theory*. 2016, pp. 1298–1318.
- [197] Ker-Chau Li. “Sliced inverse regression for dimension reduction”. In: *Journal of the American Statistical Association* 86.414 (1991), pp. 316–327.
- [198] Joel L Horowitz. *Semiparametric and nonparametric methods in econometrics*. Vol. 12. Springer, 2009.
- [199] Robert A Jacobs et al. “Adaptive mixtures of local experts.” In: *Neural computation* 3.1 (1991), pp. 79–87.
- [200] Seniha Esen Yuksel, Joseph N Wilson, and Paul D Gader. “Twenty years of mixture of experts”. In: *IEEE transactions on neural networks and learning systems* 23.8 (2012), pp. 1177–1193.
- [201] Teng Zhang. “Phase Retrieval by Alternating Minimization with Random Initialization”. In: *arXiv preprint arXiv:1812.01255* (2018).

- [202] Avishek Ghosh et al. “Max-Affine Regression II: Universal parameter estimation for small-ball designs [Online] <https://tinyurl.com/tae7z5r>”. In: 2019.
- [203] T Tony Cai, Xiaodong Li, and Zongming Ma. “Optimal rates of convergence for noisy sparse phase retrieval via thresholded Wirtinger flow”. In: *The Annals of Statistics* 44.5 (2016), pp. 2221–2251.
- [204] Marek Kanter and Harold Propp. “Reduction of variance for Gaussian densities via restriction to convex sets”. In: *Journal of Multivariate Analysis* 7.1 (1977), pp. 74–81.
- [205] Santosh S Vempala. “Learning convex concepts from Gaussian distributions with PCA”. In: *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*. IEEE. 2010, pp. 124–130.
- [206] Michel Ledoux. *The concentration of measure phenomenon*. 89. American Mathematical Soc., 2001.
- [207] Barry James, Kang James, and Yongcheng Qi. “Limit distribution of the sum and maximum from multivariate Gaussian sequences”. In: *Journal of multivariate analysis* 98.3 (2007), pp. 517–532.
- [208] Yi Yu, Tengyao Wang, and Richard J Samworth. “A useful variant of the Davis–Kahan theorem for statisticians”. In: *Biometrika* 102.2 (2014), pp. 315–323.
- [209] Andrew F Siegel. “A surprising covariance involving the minimum of multivariate normal variables”. In: *Journal of the American Statistical Association* 88.421 (1993), pp. 77–80.
- [210] Jun S Liu. “Siegel’s formula via Stein’s identities”. In: *Statistics & Probability Letters* 21.3 (1994), pp. 247–251.
- [211] BG Manjunath and Stefan Wilhelm. *Moments calculation for the double truncated multivariate normal density*. 2009.
- [212] Alexandre B. Tsybakov. *Introduction to Nonparametric Estimation*. 1st. Springer Publishing Company, Incorporated, 2008.
- [213] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2018.
- [214] Vladimir N Vapnik and Aleksei Yakovlevich Chervonenkis. “The uniform convergence of frequencies of the appearance of events to their probabilities”. In: *Doklady Akademii Nauk*. Vol. 181. 4. Russian Academy of Sciences. 1968, pp. 781–783.
- [215] Monika Csikos, Andrey Kupavskii, and Nabil H Mustafa. “Optimal bounds on the VC-dimension”. In: *arXiv preprint arXiv:1807.07924* (2018).
- [216] Gilles Hargé. “A convex/log-concave correlation inequality for Gaussian measure and an application to abstract Wiener spaces”. In: *Probability theory and related fields* 130.3 (2004), pp. 415–440.
- [217] Yaozhong Hu. “Itô-Wiener chaos expansion with exact residual and correlation, variance inequalities”. In: *Journal of Theoretical Probability* 10.4 (1997), pp. 835–848.

- [218] John Burkardt. “The truncated normal distribution”. In: (2014).
- [219] Aad W van der Vaart and Jon A. Wellner. *Weak convergence and empirical processes*. Springer Series in Statistics. With applications to statistics. Springer-Verlag, New York, 1996, pp. xvi+508.
- [220] Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*. Vol. 47. Cambridge University Press, 2018.
- [221] Avishek Ghosh et al. “Max-Affine Regression I: Parameter Estimation for Gaussian Designs”. In: <https://tinyurl.com/spyjmgv> (2019).
- [222] Avishek Ghosh et al. “Max-Affine Regression: Provable, Tractable, and Near-Optimal Statistical Estimation”. In: *arXiv preprint arxiv:1906.09255* (2019).
- [223] Larry Wasserman. *All of nonparametric statistics*. Springer Science & Business Media, 2006.
- [224] Emmanuel J Candes, Xiaodong Li, and Mahdi Soltanolkotabi. “Phase retrieval via Wirtinger flow: Theory and algorithms”. In: *IEEE Transactions on Information Theory* 61.4 (2015), pp. 1985–2007.
- [225] Yonina C Eldar and Shahar Mendelson. “Phase retrieval: Stability and recovery guarantees”. In: *Applied and Computational Harmonic Analysis* 36.3 (2014), pp. 473–494.
- [226] John C Duchi and Feng Ruan. “Solving (most) of a set of quadratic equalities: Composite optimization for robust phase retrieval”. In: *arXiv preprint arXiv:1705.02356* (2017).
- [227] Anthony Carbery and James Wright. “Distributional and L^q norm inequalities for polynomials over convex bodies in \mathbb{R}^n ”. In: *Mathematical research letters* 8.3 (2001), pp. 233–248.
- [228] Mark Rudelson and Roman Vershynin. “The Littlewood–Offord problem and invertibility of random matrices”. In: *Advances in Mathematics* 218.2 (2008), pp. 600–633.
- [229] Jean-Yves Audibert and Sébastien Bubeck. “Best arm identification in multi-armed bandits”. In: 2010.
- [230] Alekh Agarwal et al. “Corralling a band of bandit algorithms”. In: *arXiv:1612.06246* (2016).
- [231] Avishek Ghosh, Sayak Ray Chowdhury, and Aditya Gopalan. “Misspecified linear bandits”. In: *Thirty-First AAAI Conference on Artificial Intelligence*. 2017.
- [232] Andrea Locatelli and Alexandra Carpentier. “Adaptivity to Smoothness in X-armed bandits”. In: *Conference on Learning Theory*. 2018, pp. 1463–1492.
- [233] Akshay Krishnamurthy, Zhiwei Steven Wu, and Vasilis Syrgkanis. “Semiparametric contextual bandits”. In: *arXiv preprint arXiv:1803.04204* (2018).
- [234] Thodoris Lykouris, Karthik Sridharan, and Éva Tardos. “Small-loss bounds for online learning with partial information”. In: *arXiv preprint arXiv:1711.03639* (2017).

- [235] Peter Auer, Pratik Gajane, and Ronald Ortner. “Adaptively tracking the best arm with an unknown number of distribution changes”. In: *European Workshop on Reinforcement Learning*. Vol. 14. 2018, p. 375.
- [236] Haipeng Luo and Robert E Schapire. “Achieving all with no parameters: Adanormalhedge”. In: *Conference on Learning Theory*. 2015, pp. 1286–1304.
- [237] Brendan McMahan and Jacob Abernethy. “Minimax optimal algorithms for unconstrained linear optimization”. In: *Advances in Neural Information Processing Systems*. 2013, pp. 2724–2732.
- [238] Francesco Orabona. “Simultaneous model selection and optimization through parameter-free stochastic learning”. In: *Advances in Neural Information Processing Systems*. 2014, pp. 1116–1124.
- [239] Ashok Cutkosky and Kwabena Boahen. “Online learning without prior information”. In: *arXiv preprint arXiv:1703.02629* (2017).
- [240] Vladimir Vapnik. *Estimation of dependences based on empirical data*. Springer Science & Business Media, 2006.
- [241] Lucien Birgé, Pascal Massart, et al. “Minimum contrast estimators on sieves: exponential bounds and rates of convergence”. In: *Bernoulli* 4.3 (1998), pp. 329–375.
- [242] Gábor Lugosi, Andrew B Nobel, et al. “Adaptive model selection using empirical complexities”. In: *The Annals of Statistics* 27.6 (1999), pp. 1830–1864.
- [243] Sylvain Arlot, Peter L Bartlett, et al. “Margin-adaptive model selection in statistical learning”. In: *Bernoulli* 17.2 (2011), pp. 687–713.
- [244] Vladimir Cherkassky. “Model complexity control and statistical learning theory”. In: *Natural computing* 1.1 (2002), pp. 109–133.
- [245] Luc Devroye, László Györfi, and Gábor Lugosi. *A probabilistic theory of pattern recognition*. Vol. 31. Springer Science & Business Media, 2013.
- [246] Peter Auer. “Using confidence bounds for exploitation-exploration trade-offs”. In: *Journal of Machine Learning Research* 3.Nov (2002), pp. 397–422.
- [247] Olivier Chapelle and Yi Chang. “Yahoo! Learning to Rank Challenge Overview”. In: *Proceedings of the 2010 International Conference on Yahoo! Learning to Rank Challenge - Volume 14*. YLRC’10. Haifa, Israel: JMLR.org, 2010, pp. 1–24.
- [248] Aldo Pacchiano et al. “Model selection in contextual stochastic bandit problems”. In: *arXiv preprint arXiv:2003.01704* (2020).
- [249] Michael Krikheli and Amir Leshem. “Finite sample performance of linear least squares estimators under sub-Gaussian martingale difference noise”. In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2018, pp. 4444–4448.

- [250] Jeongyeol Kwon and Constantine Caramanis. “The EM Algorithm gives Sample-Optimality for Learning Mixtures of Well-Separated Gaussians”. In: *Conference on Learning Theory*. PMLR. 2020, pp. 2425–2487.
- [251] Dylan J Foster, Akshay Krishnamurthy, and Haipeng Luo. “Open Problem: Model Selection for Contextual Bandits”. In: *Conference on Learning Theory*. PMLR. 2020, pp. 3842–3846.