Algorithms for Robust and Memory-Efficient Learning

By

Fred Zhang

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Summer 2024

Algorithms for Robust and Memory-Efficient Learning

Abstract

Algorithms for Robust and Memory-Efficient Learning

By

Fred Zhang

Doctor of Philosophy in Computer Science

University of California, Berkeley

Professor Jelani Nelson, Chair

Modern machine learning (ML) processes massive data. The thesis tackles two algorithmic challenges arising from large-scale ML—robustness to noisy training data and memory-efficiency of the learning algorithms. Motivated by the first, I propose (i) the fastest algorithm for learning the mean of high-dimensional heavy-tailed distribution, (ii) a unified analysis framework for robust estimation, and (iii) efficient and robust algorithm for privately estimating high dimensional Gaussian. For memory-efficiency, I give the first sub-linear space algorithm for online prediction, the most classic problem in sequential learning.

Contents

Contents i

List of Figures iii

List of Tables iv

1 Introduction 1
- 1.1 Algorithmic robust statistics . . . 1
- 1.2 Memory-efficient online learning . . . 2

2 Fast Algorithm for Heavy-Tailed Mean Estimation 3
- 2.1 Introduction . . . 3
- 2.2 Preliminaries and Assumptions . . . 9
- 2.3 Descent Procedure . . . 9
- 2.4 Inner Maximization and its Two-Sided Relaxation . . . 11
- 2.5 Approximating the Inner Maximization . . . 11
- 2.6 Conclusion and Discussion . . . 17

3 Unifying Robust and Heavy-Tailed Mean Estimation 18
- 3.1 Introduction . . . 18
- 3.2 Preliminaries . . . 23
- 3.3 The Meta-Problem and a Meta-Algorithm . . . 24
- 3.4 Estimation under Corruptions . . . 29
- 3.5 Equivalent Notions of Centrality . . . 32
- 3.6 Estimation under Heavy-Tails . . . 35
- 3.7 Discussion . . . 39

4 Robust and Private Estimation of Gaussian 40
- 4.1 Introduction . . . 40
- 4.2 Techniques . . . 45
- 4.3 Preliminaries . . . 46
- 4.4 Pure DP Covariance Estimation . . . 48
- 4.5 Lower Bounds Pure DP Covariance Estimate . . . 58

5 Memory-Efficient Online Learning **63**
5.1 Introduction . . . 63
5.2 Preliminaries . . . 71
5.3 The building block . . . 73
5.4 Full algorithm and analysis . . . 81
5.5 Lower bound against adaptive adversary . . . 91
5.6 Conclusion . . . 93

Bibliography **95**

A Technical details for Chapter 2 **107**
A.1 Main algorithm description . . . 107
A.2 Technical facts . . . 107
A.3 Omitted proofs from Section 2.3 . . . 109
A.4 Omitted proof from Section 2.4 . . . 110
A.5 Omitted proof from Section 2.5 . . . 113
A.6 Omitted proofs from Section 2.5 . . . 115
A.7 Full proof of main theorem . . . 116
A.8 Interpretation of FHP algorithm [101] as regret minimization . . . 117

B Technical details for Chapter 3 **120**
B.1 Technical Lemmas and Proofs . . . 120
B.2 Extension to sub-gaussian distributions . . . 124
B.3 Sample reweighing via Matrix Multiplicative Update . . . 127
B.4 Sample reweighing via Online Gradient Descent . . . 131
B.5 Optimal Breakdown Point Analysis . . . 136

List of Figures

2.1 Main algorithm—Descent . . . 10
2.2 Prune . . . 12
2.3 Approximate inner maximization via Bregman projection—ApproxBregman . 14

4.1 PureDPMatrixMean (based on PureDPMean in Theorem 4.3.4) . . . 49
4.2 One round, weak private preconditioning algorithm . . . 50
4.3 Recursive private preconditioning algorithm . . . 52
4.4 Private covariance estimation algorithm . . . 53
4.5 Private mean estimation algorithm . . . 55

5.1 Pool at the end of an epoch. If expert 3 is removed, then we no longer store $\mathcal{L}_{I_3}(1), \mathcal{L}_{I_3}(2)$. . . . 75

A.1 Distance estimation—DistEst . . . 107
A.2 Gradient estimation—GradEst . . . 108
A.3 Final algorithm . . . 108
A.4 Rounding algorithm—Round . . . 115
A.5 Iterative MWU procedure . . . 117

List of Tables

4.1 Computationally-efficient sample complexity upper bounds for DP covariance estimators of $\mathcal{N}(0, \Sigma)$, with the utility guarantee $\left\|\widehat{\Sigma} - \Sigma\right\|_{\Sigma} \leq \alpha$. Robustness refers to additional error when the sample is η-corrupted. Here, d is the dimension, κ is an a priori bound such that $\mathbb{I} \preceq \Sigma \preceq \kappa\mathbb{I}$. The bounds suppress polylogarithmic factors in d, $\frac{1}{\eta}$, $\frac{1}{\alpha}$, and $\frac{1}{\delta}$. 42
4.2 Sample complexity lower bounds for DP covariance estimators of $\mathcal{N}(0, \Sigma)$, with the utility guarantee $\left\|\widehat{\Sigma} - \Sigma\right\|_{\Sigma} \leq \alpha$. Here, d is the dimension, κ is an a priori bound such that $\mathbb{I} \preceq \Sigma \preceq \kappa\mathbb{I}$. The bounds suppress polylogarithmic factors in d, $\frac{1}{\alpha}$, and $\frac{1}{\delta}$. 43

Acknowledgments

First and foremost, I would like to thank my advisor, Jelani Nelson. Jelani has been extraordinarily supportive throughout the journey, serving as an endless source of ideas and optimism. As a research supervisor, he ensured that I stayed on track while providing me with the freedom to pursue my own interests. I hope to continue seek his advice and wisdom after I cease to be his PhD advisee.

I am grateful to Prasad Raghavendra and Nikhil Srivastava for being on the thesis committee. The dissertation work would not have been fruitful without their support.

Throughout graduate school, I have interacted with Jacob Steinhardt and his research group at various points, both socially and academically. Although I took no formal part in his lab, Jacob has been always been kind to me and offered me advice multiple times when I need them—and I am truly grateful for that.

In the past few years, I have received mentorship from the postdocs in Jelani's group, including Rasmus Kyng, Kyle Luh, Paris Syminelakis, Shay Golan, and Samson Zhou. I learned a tremendous amount from each of them.

In Summer 2022 and 2023, I was fortunate to work with David Woodruff, Richard Zhang, Matthew Fahrbach, Peilin Zhong, and Neel Nanda during two productive research internships. Special thanks to Neel who mentored me on a mechanistic interpretability project which significantly steered my research direction. I would not have ventured into empirical ML research without the experience and his support.

Thanks are due to Pankaj Agarwal, Emily Fox, Rong Ge, and Debmalya Panigrahi at Duke. They provided me with much advice and mentorship during my formative years as an undergraduate.

The results of this thesis have been an outcome of fruitful collaborations with Daniel Alabi, Sam Hopkins, Pravesh Kothari, Zhixian Lei, Jerry Li, Kyle Luh, Binghui Peng, Pranay Tankala, and Prayaag Venkat. In addition to them. I would like to express my gratitude to all of my collaborators over the years.

Many thanks to the staff and students in the Berkeley and Harvard theory groups. The journey would not been as enjoyable without the friends I made here.

Finally, I thank my family for their support.

Chapter 1

Introduction

Modern machine learning (ML) practice turns vast amounts of data into large models. This thesis addresses two central challenges in this pipeline. First, the massive, high-dimensional datasets pivotal to ML's success often contain noise or are susceptible to malicious tampering. Learning from such data efficiently and reliably presents a major computational challenge. Second, modern ML requires processing massive, Internet-scale data and training multi-billion parameter models. Both present challenges to the memory efficiency of the learning algorithms.

This thesis studies fundamental questions motivated by the challenges. I give an informal overview of the results as follows.

1.1 Algorithmic robust statistics

Recent progress in AI, especially language models, relies on noisy datasets sourced from across the Internet. These datasets, susceptible to outlying or even malicious samples, pose a serious challenge to the robustness of the training procedures. However, this concern is not novel; robust statistics—learning in presence of outliers—has been a central field in statistics since the 1960s.

New algorithms and unified analysis My work in the area is driven by the quest for computational efficiency. This has led to several contributions to the algorithmic guarantees in robust statistics.

In Chapter 2, I study mean estimation from high-dimensional heavy-tailed distributions, where outliers are a natural occurrence. I design the fastest known algorithm that achieves the optimal statistical accuracy. All prior works were based on semi-definite programming (SDP) and thus had prohibitive runtimes. In contrast, our algorithm is fully spectral and only requires eigenvector computations, resulting in better computational efficiency.

Taking this step further in Chapter 3, I show that essentially the same algorithm is robust against adversarial corruptions. This directly leads to a novel analysis of the fastest algorithms of this setting as well. A central innovation of this work is to recast most existing procedures for robust estimation under regret minimization, providing a unified view through the lens of online optimization.

Connections to other fields The research in algorithmic robust statistics has also led to new developments in adjacent fields, including privacy and mechanism design.

In Chapter 4, I give the first polynomial-time algorithms for learning a high-dimensional Gaussian subject to pure differential privacy (DP).

Finally, I mention that beyond this thesis, my work [159] applies high-dimensional median constructions from robust statistics to strategy-proof facility location, a classic problem in mechanism design and social choice theory. With this tool, the work derives new computational and game-theoretic guarantees for the problem in a Bayesian setting.

1.2 Memory-efficient online learning

As present-day deep learning produces massive billion-parameter models, memory constraint has become a severe hurdle to both their training and serving. Motivated by this challenge, I design the first online learning algorithm in sub-linear space in Chapter 5. Traditional methods, such as multiplicative weights update, require memory proportional to the dimension of the problem. Our work sidesteps this limitation. I introduce a novel framework for selecting a competitive subset of experts and a recursive scheme for boosting their performance. Both techniques have since sparked several follow-up works.

My later research [158] initiates the study of robust algorithms for online learning, where the adversary generates inputs adaptively based on the past decisions of the algorithm. Capitalizing on connections with differential privacy, I give the first robust and memory-efficient algorithm for this problem, as well as tight space bound against deterministic schemes.

Chapter 2

Fast Algorithm for Heavy-Tailed Mean Estimation

In this chapter, we study the algorithmic problem of estimating the mean of a heavy-tailed random vector in $\mathbb{R}^d$, given n i.i.d. samples. The goal is to design an efficient estimator that attains the optimal sub-gaussian error bound, only assuming that the random vector has bounded mean and covariance. Polynomial-time solutions to this problem are known but have high runtime due to their use of semi-definite programming (SDP). Moreover, conceptually, it remains open whether convex relaxation is truly necessary for this problem.

We show that it is possible to go beyond SDP and achieve better computational efficiency. In particular, we provide a *spectral* algorithm that achieves the optimal statistical performance and runs in time $\widetilde{O}\left(n^2 d\right)$, improving upon the previous fastest runtime $\widetilde{O}\left(n^{3.5} + n^2 d\right)$ by Cherapanamjeri *et al.* (COLT '19). Our algorithm is spectral in that it only requires (approximate) eigenvector computations, which can be implemented very efficiently by, for example, power iteration or the Lanczos method.

At the core of our algorithm is a novel connection between the furthest hyperplane problem introduced by Karnin *et al.* (COLT '12) and a structural lemma on heavy-tailed distributions by Lugosi and Mendelson (Ann. Stat. '19). This allows us to iteratively reduce the estimation error at a geometric rate using only the information derived from the top singular vector of the data matrix, leading to a significantly faster running time.

2.1 Introduction

Estimating the mean of a multivariate distribution from samples is among the most fundamental statistical problems. Surprisingly, it was only recently that a line of works in the statistics literature culminated in an estimator achieving the optimal statistical error under minimal assumptions ([125]). However, from an algorithmic point of view, computation of this estimator appears to be intractable. On the other hand, fast estimators, such as the empirical average, tend to achieve sub-optimal statistical performance. The following question remains open:

Can we provide simple, fast algorithm that computes a statistically optimal mean estimator in high dimensions, under minimal assumptions?

In this chapter, we make progress towards this goal, under the classic setting where only finite mean and covariance are assumed. Formally, our problem is defined as follows. Given n i.i.d. copies $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n$ of a random vector $\boldsymbol{X} \in \mathbb{R}^d$ with bounded mean $\boldsymbol{\mu} = \mathbb{E}\,\boldsymbol{X}$ and covariance $\boldsymbol{\Sigma} = \mathbb{E}(\boldsymbol{X} - \boldsymbol{\mu})(\boldsymbol{X} - \boldsymbol{\mu})^T$, compute an estimate $\widehat{\boldsymbol{\mu}} = \widehat{\boldsymbol{\mu}}(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n)$ of the mean $\boldsymbol{\mu}$. Our goal is to show that for any failure probability $\delta \in (0, 1]$,

$$\Pr\left(\|\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}\| > r_\delta\right) \leq \delta,$$

for as small a radius r_δ as possible. Moreover, we would like to compute $\widehat{\boldsymbol{\mu}}$ efficiently. The naïve estimator is simply the empirical mean $\overline{\boldsymbol{\mu}} = \frac{1}{n}\sum_{i=1}^n \boldsymbol{X}_i$. It is well known that among all estimators, the empirical mean minimizes mean squared error. However, if we instead use the size of the deviations to quantify the quality of the estimator, the empirical mean is only optimal for sub-gaussian random variables ([31]). When $\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ we have with probability at least $1 - \delta$,

$$\|\overline{\boldsymbol{\mu}} - \boldsymbol{\mu}\| \leq \sqrt{\frac{\operatorname{Tr}(\boldsymbol{\Sigma})}{n}} + \sqrt{\frac{2\|\boldsymbol{\Sigma}\| \log(1/\delta)}{n}} \tag{2.1.1}$$

An estimator that achieves above is said to have *sub-gaussian performance* or *sub-gaussian rate.*

In practical settings, assuming that the samples obey a Gaussian distribution may be unrealistic. In an effort to design robust estimators, it is natural to study the mean estimation problem under very weak assumptions on the data. A recent line of works ([31, 132, 52, 94, 124, 125, 113]) study the mean estimation problem when the samples obey a heavy-tailed distribution.

For heavy-tailed distributions the performance of the empirical mean is abysmal. If we only assume that $\boldsymbol{X}$ has finite mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$, then by Chebyshev's inequality, the empirical mean only achieves error of order $\sqrt{\operatorname{Tr}(\boldsymbol{\Sigma})/\delta n}$, which is worse than the sub-gaussian rate in two ways. First, its dependence on $\frac{1}{\delta}$ is exponentially worse. Second, the $\operatorname{Tr}(\boldsymbol{\Sigma})$ term, which may grow with the dimension d, is multiplied the dimension-independent term $\sqrt{1/\delta n}$, whereas in the Gaussian case, the two are separate.

Median-of-means paradigm Surprisingly, recent work has shown that it is possible to improve on the performance of the empirical mean using the *median-of-means* approach. For $d = 1$, the following construction, originally due to [134, 92, 9], achieves sub-gaussian performance:

(i) First, bucket the data into $k = \lceil 10 \log(1/\delta) \rceil$ disjoint groups and compute their means $Z_1, Z_2, \cdots, Z_k$.

(ii) Then, output the median $\widehat{\mu}$ of $Z_1, Z_2, \cdots, Z_k$.

A long line of work has followed this paradigm and generalized it to higher dimensions ([31, 52, 94, 124, 125]). The key challenge is to correctly define a notion of median for a collection of points in $\mathbb{R}^d$. [132] considered $\widehat{\boldsymbol{\mu}}_{GM}$ defined to be the *geometric median* of the bucket means $\boldsymbol{Z}_1, \ldots, \boldsymbol{Z}_k$. For some constant c_{GM}, with probability at least $1-\delta$, it satisfies

$$\|\widehat{\boldsymbol{\mu}}_{GM} - \boldsymbol{\mu}\| \leq c_{GM}\sqrt{\frac{\operatorname{Tr}\boldsymbol{\Sigma}\cdot\log(1/\delta)}{n}}. \tag{2.1.2}$$

This achieves the correct dependence on δ, but the dimension dependent and independent terms are still not separated. Following this work, [124] described a tournament-based estimator, which finally achieved the optimal sub-gaussian radius. The idea behind their construction is to consider *all* 1*-dimensional projections* of the bucket means and try to find an estimate that is close to the median of the means of all projections. This construction has been further simplified by [86]. Formally, it was shown that the following estimator achieves the optimal, sub-gaussian error:

$$\widehat{\boldsymbol{\mu}}_{LM} = \arg\min_{\boldsymbol{x}\in\mathbb{R}^d}\max_{\boldsymbol{u}\in\mathbb{S}^{d-1}}\left|\text{median}\left\{\langle\boldsymbol{Z}_i,\boldsymbol{u}\rangle\right\}_{i=1}^k - \langle\boldsymbol{x},\boldsymbol{u}\rangle\right|. \tag{2.1.3}$$

Clearly, searching over all directions in $\mathbb{S}^{d-1}$ requires exponential time. The key question, therefore, is whether one can achieve both computational and statistical efficiency simutaneously.

Computational considerations A priori, it is unclear that the Lugosi-Mendelson estimator can be computed in polynomial time as a direct approach involves solving an intractable optimization problem. Moreover, the Lugosi-Mendelson analysis seems to suggest that estimation in the heavy-tailed model is conceptually harder than under (adversarial) corruptions. In the latter, each sample can be classified as either an inlier or an outlier. In the heavy-tailed setting, Lugosi-Mendelson shows that there is a majority of the bucket means that cluster around the true mean along any projection. However, a given sample may be an inlier by being close to the mean when projected onto one direction, but an outlier when projected onto another. In other words, the *set* of inliers may change from one direction to another.

Surprisingly, a recent line of works have established the polynomial-time computability of Lugosi-Mendelson estimator. [86] formulates $\widehat{\boldsymbol{\mu}}_{LM}$ as the solution of a low-degree polynomial optimization problem and showed that using the Sum-of-Squares SDP hierarchy to relax this problem yields a sub-gaussian estimator. While the run-time of this algorithm is polynomial, it involves solving a large SDP. Soon after, [42] provided an iterative method in which each iteration involves solving a smaller, explicit SDP, leading to a run-time of $\widetilde{O}\left(n^{3.5}+n^2d\right)$[1]. Even more recently, a concurrent and independent work by [110] gave an estimator with sub-gaussian performance that can be computed in time $\widetilde{O}(n^2 d)$. The construction is inspired by

[1]Throughout we use $\widetilde{O}(\cdot)$ to hide polylogarithmic factors (in n, d and $\log(1/\delta)$).

a near-linear time algorithm for robust mean estimation under adversarial corruptions due to [38]. The algorithm requires solving (covering) SDPs.

We note, however, that a common technique in these algorithms is SDP, which tends to be impractical for large sample sizes and in high dimensions. In contrast, our algorithm only requires approximate eigenvector computations. For a problem as fundamental as mean estimation, it is desirable to obtain simple and ideally practical solutions. A key conceptual message of this chapter is that SDP is indeed unnecessary and can be replaced by simple spectral techniques.

Our result In this chapter, we demonstrate for the first time that mean estimation with sub-gaussian rates can be achieved efficiently *without* SDP. The runtime of the algorithm matches the independent work of [110]. In addition, our algorithm enjoys robustness against (additive) corruptions, where the number of adversarial points is a small fraction of k.

It is known that there exists an information-theoretic requirement for achieving such rates—that is, $\delta \geq 2^{-O(n)}$ ([52]). Under this assumption, we give an efficient spectral algorithm.

Theorem 2.1.1. *Let $\delta \geq Ae^{-n}$ for a constant A and $k = \lceil 3600 \log(1/\delta) \rceil$. Given n points $\mathcal{G} \cup \mathcal{B}$, where $\mathcal{G}$ are i.i.d. samples from a distribution over $\mathbb{R}^d$ with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ and $\mathcal{B}$ a set of arbitrary points with $|\mathcal{B}| \leq k/200$, there is an efficient algorithm that outputs an estimate $\widehat{\boldsymbol{\mu}} \in \mathbb{R}^d$ such that with probability at least $1-\delta$,*

$$\|\boldsymbol{\mu} - \widehat{\boldsymbol{\mu}}\| \leq C \left(\sqrt{\frac{\mathrm{Tr}(\boldsymbol{\Sigma})}{n}} + \sqrt{\frac{\|\boldsymbol{\Sigma}\| \log(1/\delta)}{n}} \right),$$

for a constant C. Furthermore, the algorithm runs in time $O\left(nd + k^2 d \operatorname{polylog}(k, d)\right)$.

The algorithm is iterative. Each iteration only requires an (approximate) eigenvector computation, which can be implemented in nearly linear time by power iteration or the Lanczos algorithm. We believe that our algorithm can be fairly practical.

Other related works Recently, [138] established a formal connection between the Huber contamination model and the heavy-tailed model we study in this chapter. They leverage this connection to use an existing $\widetilde{O}(nd^2)$-time mean estimation algorithm of [59] to design estimators for the heavy-tailed model. Under moment assumptions, their estimator achieves performance better than geometric median (2.1.2), yet worse than sub-gaussian.

In addition, algorithmic robust statistics has gained much attention in the theoretical computer science community in recent years. A large body of works have studied the mean estimation problem with *adversarially* corrupted samples, with the focus on providing efficient algorithms ([59, 108, 38, 64]). For a more complete survey, see [56]

Going beyond mean estimation, there has been a recent spate of works on other statistical problems under heavy-tailed distributions. We refer the readers to [123] for a survey.

Technical overview Our main algorithm builds upon the iterative approach of [42]. For simplicity, assume there is no adversarial point. At a high level, for each iteration t, the algorithm will maintain a current guess $\boldsymbol{x}_t$ of the true mean. To update, Cherapanamjeri *et al.* study the inner maximization of $\widehat{\boldsymbol{\mu}}_{LM}$ (2.1.3) with $\boldsymbol{x} = \boldsymbol{x}_t$. They showed that under Lugosi-Mendelson structral condition, the problem is essentially equivalent of following program, which we call $\mathcal{M}(\boldsymbol{x}_t, \boldsymbol{Z})$:

$$\begin{aligned} \max \quad & \theta \\ \text{subject to} \quad & b_i \langle \boldsymbol{Z}_i - \boldsymbol{x}_t, \boldsymbol{u} \rangle \geq b_i \theta \text{ for } i = 1, \ldots, k \\ & \sum_{i=1}^{k} b_i \geq 0.95k \\ & \boldsymbol{b} \in \{0,1\}^k, \boldsymbol{u} \in \mathbb{S}^{d-1}. \end{aligned}$$

It can be shown that an optimal solution $\boldsymbol{u} \in \mathbb{S}^{d-1}$ will align with the unit vector in the direction of $\boldsymbol{\mu} - \boldsymbol{x}_t$, and θ approximates $\|\boldsymbol{\mu} - \boldsymbol{x}_t\|$. Hence, one can perform the update $\boldsymbol{x}_{t+1} \leftarrow \boldsymbol{x}_t + \gamma\theta\boldsymbol{u}$, for some appropriate constant γ, to geometrically decrease the distance of $\boldsymbol{x}_t$ to $\boldsymbol{\mu}$.

In this chapter, we start by drawing a connection between the above program and the furthest hyperplane problem (FHP) of [101]. This allows us to avoid the SDP approach in [42]. The problem can be formulated as the following:

$$\begin{aligned} \max \quad & \theta && \text{(FHP)} \\ \text{subject to} \quad & |\langle \boldsymbol{Z}_i - \boldsymbol{x}_t, \boldsymbol{u} \rangle| \geq \theta \text{ for } i = 1, \ldots, k && (2.1.4) \\ & \boldsymbol{u} \in \mathbb{S}^{d-1}. \end{aligned}$$

In the original formulation due to Karnin *et al.*, the goal is to find a maximum margin linear classifier for a collection of points, where the margin is *two-sided*. Notice that any feasible solution to $\mathcal{M}(\boldsymbol{x}_t, \boldsymbol{Z})$ satisfies at least $0,95k$ constraints of FHP as well. For an arbitrary dataset, the two-sided margin requirement indeed provides a relaxation. One technical observation of this chapter is that it is *not* a significant one, for the random data we care about—if a major fraction of the constraint (2.1.4) are satisfied, then most constraints of $\mathcal{M}(\boldsymbol{x}_t, \boldsymbol{Z})$ are satisfied as well.

Unfortunately, the algorithm of Karnin *et al.* cannot directly apply, as it only works under a strong promise that there exists a feasible solution that satisfies *all* of the constraints (2.1.4). In our setting, there may not be such a feasible solution; we can only guarantee that there exists a unit vector (namely, the one in the direction of $\boldsymbol{\mu} - \boldsymbol{x}_t$) that satisfies most of constraints with large margin.

Our main contribution is to provide an algorithm that works even under this weak promise. We now briefly review the algorithm of Karnin *et al.*, show why it fails for our purpose, and explain how we address the issues that arise. Suppose that there exists a unit vector $\boldsymbol{u}^*$ and θ^* which are feasible for the FHP problem. Then, averaging the constraints tells us that

$$\frac{1}{k}\sum_{i=1}^{k}\langle \boldsymbol{Z}_i, \boldsymbol{u}^*\rangle^2 \geq \theta^{*2}.$$

Hence, if we define $\boldsymbol{u}$ to be the top right singular vector of the matrix $\boldsymbol{A}$ whose rows are $\boldsymbol{Z}_i$, then

$$\|\boldsymbol{A}\boldsymbol{u}\|_2^2 = \sum_{i=1}^{k}\langle \boldsymbol{Z}_i, \boldsymbol{u}\rangle^2 \geq \sum_{i=1}^{k}\langle \boldsymbol{Z}_i, \boldsymbol{u}^*\rangle^2 \geq k\theta^{*2},$$

so $\boldsymbol{u}$ satisfies the constraints in (FHP) *on average*. However, the distribution of the quantities $\langle \boldsymbol{Z}_i, \boldsymbol{u}\rangle^2$ may be extremely skewed, so that $\boldsymbol{u}$ only satisfies a few of the constraints with large margin. If this happens, however, we can downweight those constraints which are satisfied by $\boldsymbol{u}$ with large slack to encourage it to satisfy more constraints. This reweighting procedure is repeated several times, and at the end we use a simple rounding scheme to yield a single output vector with the desired properties from all the repetitions. In particular, this weighting scheme is essentially the same as the classic *multiplicative weights update* (MWU) method ([13]) for regret minimization, as we show in Appendix A.8.

If we are only guaranteed that $\boldsymbol{u}^*$ satisfies most, but not all, of the constraints, then the inequality $\sum_{i=1}^{k}\langle \boldsymbol{Z}_i, \boldsymbol{u}^*\rangle^2 \geq k\theta^{*2}$ may no longer hold when the points $\boldsymbol{Z}_i$ get re-weighted and the algorithm of Karnin *et al.* cannot be guaranteed to converge. To illustrate this point, consider the following extreme case. Suppose that after the first iteration, the algorithm finds the vector $\boldsymbol{u}^*$ as the top right singular vector of $\boldsymbol{A}$. In the re-weighting procedure, the constraints i for which $\langle \boldsymbol{Z}_i, \boldsymbol{u}^*\rangle^2 \geq \theta^{*2}$ may be down-weighted significantly, whereas the remaining constraints may be unaffected. This may result in most of the weight being concentrated on the constraints i where $\langle \boldsymbol{Z}_i, \boldsymbol{u}^*\rangle^2 \ll \theta^{*2}$. In the second iteration, we have no guarantee of the behavior of the top singular vector of the re-weighted matrix because all the weight is concentrated on a small set consisting of these "bad" constraints.

To address this scenario, our key technical idea is to project the weights onto the set of *smooth distributions* after each update. Informally, the notion of smooth distribution enforces that no point can take too much probability mass—say, more than $4/k$. This prevents the weights from ever being concentrated on too small a subset and allows us to guarantee that $\sum_{i=1}^{k}\langle \boldsymbol{Z}_i, \boldsymbol{u}^*\rangle^2 \geq k\theta^{*2}$ still holds approximately. Moreover, the appropriate notion of projection here is that of a Bregman projection. Leveraging our earlier MWU interpretation of the algorithm (Section A.8), we apply a classic regret bound for MWU under Bregman projection ([13]), and this yields the same guarantee of the original algorithm. Finally, we remark that the projection can be computed quickly. Combining all these ideas together, we manage to bypass the barrier of having bad points, under the much weaker assumption on $\boldsymbol{u}^*$.

Organization The remainder of this article is organized as follows. In Section 2.2, we set up the notations and specify assumptions on the data. In Section 2.3, we explain the high level approach based on an iterative descent procedure from [42]. The procedure requires us to approximately maximize a (non-convex) objective, and we discuss its properties in Section 2.4. Section 2.5 contains the main technical innovations of this chapter, where we design and analyze a faster algorithm for the aforementioned optimization problem.

2.2 Preliminaries and Assumptions

In the following, we use $r_\delta = \sqrt{\operatorname{Tr}(\boldsymbol{\Sigma})/n} + \sqrt{\|\boldsymbol{\Sigma}\| \log(1/\delta)/n}$ to denote the optimal, subgaussian error rate and $k = \lceil 3200 \log(8/\delta) \rceil$. The input data $\{\boldsymbol{X}_i\}_{i=1}^n$ consist of $\mathcal{G}$, a set of i.i.d. points, and $\mathcal{B}$, a set of adversarial points, with $|B| \le k/200$. Our algorithm preprocesses the data $\boldsymbol{X}_i$ into the bucket means $\boldsymbol{Z}_1, \boldsymbol{Z}_2, \cdots, \boldsymbol{Z}_{2k} \in \mathbb{R}^d$.[2] Let $\mathcal{B}_j$ be the set of $\boldsymbol{X}_i$ in bucket j. We say that a bucket mean $\boldsymbol{Z}_j$ is *contaminated* if B_j contains an adversarial $X_i \in B$ and *uncontaminated* otherwise. Note that the number of contaminated bucket means is at most $k/200$.

Our argument is built on the Lugosi-Mendelson condition. It states that under any one-dimensional projection, most of the (uncontaminated) bucket means are close to the true mean, by an additive factor of $O(r_\delta)$. Throughout, we pessimistically assume all contaminated bucket means do not satisfy this property (under any projection) and condition on the following event.

Assumption 2.2.1 (Lugosi-Mendelson condition). *Under the setting above, for all unit $\boldsymbol{v}$, we have*

$$|\{i : \langle \boldsymbol{v}, \boldsymbol{Z}_i \rangle - \langle \boldsymbol{v}, \boldsymbol{\mu} \rangle \ge 600 r_\delta\}| \le 0.05k.$$

Lemma 2.2.1 ([125]). *Assumption 2.2.1 holds with probability at least $1 - \delta/8$.*

2.3 Descent Procedure

At a high level, our algorithm builds upon the iterative descent paradigm of [42]. It maintains a sequence of estimates and updates via distance and gradient estimate.

Definition 2.3.1 (distance estimate). *We say that d_t is a distance estimate (with respect to $\boldsymbol{x}_t$) if*

(i) when $\|\boldsymbol{\mu} - \boldsymbol{x}_t\| \le 14000 r_\delta$, we have $d_t \le 28000 r_\delta$; and

(ii) when $\|\boldsymbol{\mu} - \boldsymbol{x}_t\| > 14000 r_\delta$, we have

$$\frac{1}{21}\|\boldsymbol{\mu} - \boldsymbol{x}_t\| \le d_t \le 2\|\boldsymbol{\mu} - \boldsymbol{x}_t\| \tag{2.3.1}$$

[2] We assume δ is such that $k \le n/2$; as we mentioned in the introduction, this is information-theoretically necessary, up to a constant ([52]).

Definition 2.3.2 (gradient estimate)**.** *We say that $\boldsymbol{g}_t$ is a gradient estimate (with respect to $\boldsymbol{x}_t$) if*

$$\left\langle \boldsymbol{g}_t, \frac{\boldsymbol{\mu}-\boldsymbol{x}_t}{\|\boldsymbol{\mu}-\boldsymbol{x}_t\|} \right\rangle \geq \frac{1}{200} \tag{2.3.2}$$

whenever $\|\boldsymbol{\mu}-\boldsymbol{x}_t\| > 14000 r_\delta$.

```
1. Input: Buckets means $\boldsymbol{Z}_1, \ldots, \boldsymbol{Z}_k \in \mathbb{R}^d$, initial estimate $\boldsymbol{x}_0$, iteration
   count $T_{\mathsf{des}}$, and step size $\eta$.
2. For $t = 1, \ldots, T_{\mathsf{des}}$:
    a) Compute $d_t = \textsc{DistEst}(\boldsymbol{Z}', \boldsymbol{x}_t)$.
    b) Compute $\boldsymbol{g}_t = \textsc{GradEst}(\boldsymbol{Z}', \boldsymbol{x}_t)$.
    c) Update $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \eta d_t \boldsymbol{g}_t$.
3. Output: $\boldsymbol{x}_{t^*}$, where $t^* = \arg\min_t d_t$.
```

Figure 2.1: Main algorithm—Descent

Suppose we initialize the estimate with coordinate-wise median-of-means which achieves an error rate $\sqrt{\|\boldsymbol{\Sigma}\| kd/n}$ (lemma A.2.2). The following lemma states that if DistEst and GradEst provide distance and gradient estimate, then the algorithm Descent succeeds in logarithmic iterations. The lemma has essentially appeared in [42], albeit with a general initialization and a different set of constants. We give a proof in Appendix A.3 for completeness.

Lemma 2.3.1 (convergence rate; see [42])**.** *Assume that for all $t \leq T_{\mathsf{des}}$, d_t is a distance estimate and $\boldsymbol{g}_t$ is a gradient estimate (with respect to $\boldsymbol{x}_t$). Suppose $\|\boldsymbol{\mu}-\boldsymbol{x}_0\| \leq O\left(\sqrt{\|\boldsymbol{\Sigma}\| kd/n}\right)$. Then the output of Algorithm 2.1 Descent instantiated with $T_{\mathsf{des}} = \Theta(\log d)$ and $\eta = 1/8000$ satisfies $\|\boldsymbol{x}_{t^*} - \boldsymbol{\mu}\| \leq O(r_\delta)$.*

2.4 Inner Maximization and its Two-Sided Relaxation

[42] obtains gradient and distance estimates by solving the *inner maximization* problem of the Lugosi-Mendelson estimator, denoted by $\mathcal{M}(\boldsymbol{x}, \boldsymbol{Z})$:

$$\begin{aligned} \max \quad & \theta \\ \text{subject to} \quad & b_i \langle \boldsymbol{Z}_i - \boldsymbol{x}, \boldsymbol{w} \rangle \geq b_i \theta \text{ for } i = 1, \dots, k \\ & \sum_{i=1}^{k} b_i \geq 0.95k \\ & \boldsymbol{b} \in \{0,1\}^k, \boldsymbol{w} \in \mathbb{S}^{d-1}. \end{aligned}$$

We also denote its feasibility version for a fixed θ by $\mathcal{M}(\theta, \boldsymbol{x}, \boldsymbol{Z})$. Note that the constraint of $\mathcal{M}(\boldsymbol{x}, \boldsymbol{Z})$ dictates that 0.95 fraction of the data must lie on *one* side of the hyperplane $\boldsymbol{w}$ with a margin θ. As discussed in the introduction, we relax it by allowing a two-sided margin: $\mathcal{M}_2(\boldsymbol{x}, \boldsymbol{Z})$.

$$\begin{aligned} \max \quad & \theta \\ \text{subject to} \quad & b_i | \langle \boldsymbol{Z}_i - \boldsymbol{x}, \boldsymbol{w} \rangle | \geq b_i \theta \text{ for } i = 1, \dots, k \\ & \sum_{i=1}^{k} b_i \geq 0.95k \\ & \boldsymbol{b} \in \{0,1\}^k, \boldsymbol{w} \in \mathbb{S}^{D-1}. \end{aligned}$$

One technical observation here is that under the Lugosi-Mendelson condition, this relaxation is insignificant. Indeed, approximately solving the problem suffices for gradient and distance estimates.

Lemma 2.4.1. *Let θ^* be the optimal value of $\mathcal{M}(\boldsymbol{x}, \boldsymbol{Z})$ and $\boldsymbol{w}$ be a unit vector such that for at least $k/8$ of the $\boldsymbol{Z}_i$, we have $| \langle \boldsymbol{w}, \boldsymbol{Z}_i - \boldsymbol{x} \rangle | \geq \theta$, where $\theta = 0.1\theta^*$. We have that (i) θ is a distance estimate and (ii) either $\boldsymbol{w}$ or $-\boldsymbol{w}$ is a gradient estimate.*

We give a proof in Appendix A.4. The intuition here is simple. If $\|\boldsymbol{x} - \boldsymbol{\mu}\| \ll r_\delta$, then the Lugosi-Mendelson condition ensures at most $0.05k$ points are far from $\boldsymbol{x}$ by $O(r_\delta)$ (under any projection), so $\theta = O(r_\delta)$. On the other hand, if $\|\boldsymbol{x} - \boldsymbol{\mu}\| \gg r_\delta$, along the gradient direction, a majority of data lie only on *one* side of the hyperplane, the side that contains the true mean, so the two-sided constraint does not make a difference.

2.5 Approximating the Inner Maximization

We now give an algorithm that efficiently computes a approximate solution to the relaxation of the inner maximization. This will provide gradient and distance estimates for each iteration of the main Descent algorithm (Algorithm 2.1).

The run-time of the algorithm is proportional $1/\theta^2$. For technical reasons, we need to ensure that $\|\boldsymbol{Z}_i - \boldsymbol{x}\| \le 1$ for all i. However, naïvely scaling all the data would decrease θ, thereby blowing up the running time. Hence, as a preprocessing step, we prune out a small fraction of points $\boldsymbol{Z}_i - \boldsymbol{x}$ with large norm before scaling.

Pruning and scaling

The preprocessing step (Algorithm 2.2) will be executed *only once* in the algorithm. After the pruning step and an appropriate scaling, we may assume the following structures on the data.

Assumption 2.5.1. *Given a current estimate $\boldsymbol{x}$, the pruned dataset $\boldsymbol{Z} \in \mathbb{R}^{k' \times d}$ of size k', let $\boldsymbol{Z}_i' = \frac{1}{B}(\boldsymbol{Z}_i - \boldsymbol{x})$, where $B = \max_i \|\boldsymbol{Z}_i' - \boldsymbol{x}\|$. We assume (i) $\|\boldsymbol{Z}_i'\| \le 1$; (ii) $k' \ge 0.9k$; and (iii) there exists $\theta = \Omega(1/\sqrt{d})$ and a unit vector $\boldsymbol{w}$ such that for at least $0.8k$ points $|\langle \boldsymbol{Z}_i', \boldsymbol{w}\rangle| \ge \theta$.*

We analyze the subroutine and prove the following lemma in Appendix A.5.

Lemma 2.5.1. *With probability at least $1-\delta/8$, Assumption 2.5.1 holds for any $\boldsymbol{x}$ such that $\|\boldsymbol{x} - \boldsymbol{\mu}\| \le O\left(\sqrt{\|\boldsymbol{\Sigma}\| kd/n}\right)$ and $\|\boldsymbol{x} - \boldsymbol{\mu}\| \ge \Omega(r_\delta)$.*

```
1. Input: Dataset $Z_1, Z_2, \cdots, Z_k \in \mathbb{R}^d$, initial estimate $x_0$
2. Compute the distances $d_i = \|Z_i - x_0\|$.
3. Sort the points by $d_i$ in decreasing order.
4. Remove the top $1/10$ fraction of them. Let $Z_1, \cdots, Z_{k'}$ be the remaining
   data.
5. Output: $Z_1, \cdots, Z_{k'}$
```

Figure 2.2: Prune

In the remainder of the section, given a current estimate $\boldsymbol{x}$, we work with the pruned and scaled data, centered at $\boldsymbol{x}$, which we call $\boldsymbol{Z}' \in \mathbb{R}^{k' \times d}$.

We will aim at proving the following lemma, under Assumption 2.5.1.

Lemma 2.5.2 (key lemma)**.** *Assume Assumption 2.5.1. Let $\delta \in (0,1)$ and $T_{des} = \Theta(\log d)$. Suppose that there exists $\boldsymbol{w}^* \in \mathbb{S}^{d-1}$ which satisfies $|\langle \boldsymbol{Z}_i', \boldsymbol{w}^*\rangle| \ge \theta^*$ for $0.8k$ points in $\{\boldsymbol{Z}_i'\}$. Then there is an algorithm* ApproxBregman *which, with probability at least $1 - \delta/4T_{des}$,* *outputs* $\boldsymbol{w} \in \mathbb{S}^{d-1}$ *such that for at least* 0.45 *fraction of the points* $\boldsymbol{Z}_i'$*, it holds that* $|\langle \boldsymbol{Z}_i', \boldsymbol{w}\rangle| \geq 0.1\theta^*$*.*

Further, ApproxBregman *runs in time* $\widetilde{O}(k^2 d)$*.*

Approximation via Bregman Projection

In this section, we give the main algorithm for approximating $\mathcal{M}_2$. Suppose (by binary search) that we know the optimal margin θ in Lemma 2.5.2. The goal is to find a unit vector $\boldsymbol{w}$ such that a constant fraction of $\boldsymbol{Z}_i'$ has margin $|\langle \boldsymbol{Z}_i', \boldsymbol{w}\rangle| \geq \theta$. The intuition is that we can start by computing the top singular vector of $\boldsymbol{Z}'$. Then the margin would be large on average: certain points may overly satisfy the margin demand while other may under-satisfy it. Hence, we would downweight those data poitns that achieve large margin and compute the top singular vector of the weighted matrix again.

However, it may stop making progress if it puts too much weight on the points that do not satisfy the margin bound. In this section, we show how to prevent this scenario from occurring. The key idea is that at every iteration, we "smooth" the weight vector τ_t so that we can guarantee progress is being made. We will formulate our algorithm in the well-studied regret-minimization framework and appeal to existing machinery ([13]) to derive the desired approximation guarantees.

First, we define what type of distribution we would like τ_t to be.

Definition 2.5.1 (Smooth distributions)**.** *The set of* smooth distributions *on* $[k']$ *is defined to be*

$$\mathcal{K} = \left\{ p \in \Delta_{k'} : p(i) \leq \frac{4}{k'} \text{ for every } i \in [k'] \right\},$$

where $\Delta_{k'}$ *is the set of probability distributions on* $[k']$*,*

$$\Delta_{k'} = \left\{ p : [k'] \to [0,1] : \sum_{i \in [k']} p(i) = 1 \right\}.$$

In the course of the algorithm, after updating τ_t as in the previous section, it may no longer be smooth. Hence, we will replace it by the closest smooth weight vector (under KL divergence). The following fact confirms that finding this closest smooth weight vector can be done quickly.

Fact 2.5.1 ([18])**.** *For any* $p \in \Delta_k$ *with support size at least* $k'/2$*, computing*

$$\Pi_{\mathcal{K}}(p) = \arg\min_{q \in \mathcal{K}} KL(p||q)$$

can be done in $\tilde{O}(k')$ *time, where* $KL(\cdot||\cdot)$ *denotes the Kullback-Leibler divergence.*

```
1. Input: Buckets means $\boldsymbol{Z}' \in \mathbb{R}^{k' \times d}$, margin $\theta$, iteration count $T \in \mathbb{N}$
2. Initialize weights: $\boldsymbol{\tau}_1 = \frac{1}{k'}(1, \ldots, 1) \in \mathbb{R}^{k'}$.
3. For $t = 1, \ldots, T$, repeat:
    a) Let $\boldsymbol{A}_t$ be the $k' \times d$ matrix whose $i$th row is $\sqrt{\boldsymbol{\tau}_t(i)}(\boldsymbol{Z}'_i)$ and $\boldsymbol{w}_t$ be
       its approximate top right singular vector .
    b) Set $\sigma_t(i) = |\langle \boldsymbol{Z}'_i, \boldsymbol{w}_t \rangle|$.
    c) Reweight: If $\|\boldsymbol{A}_t \boldsymbol{w}_t\|_2^2 \geq \frac{\theta^2}{10}$, then $\boldsymbol{\tau}_{t+1}(i) = \boldsymbol{\tau}_t(i)\,(1 - \boldsymbol{\sigma}_t(i)^2/2)$ for
       $i \in [k']$. Otherwise, do not change the weights.
    d) Normalize: Let $Z = \sum_{i \in [k']} \boldsymbol{\tau}_{t+1}(i)$ and redefine $\boldsymbol{\tau}_{t+1} \leftarrow \frac{1}{Z}\boldsymbol{\tau}_{t+1}$.
    e) Compute the Bregman projection: $\boldsymbol{\tau}_{t+1} \leftarrow \Pi_{\mathcal{K}}(\boldsymbol{\tau}_{t+1})$.
4. Output: $\boldsymbol{w} \leftarrow$ROUND$\big(\boldsymbol{Z}', \{\boldsymbol{w}_j\}_{t=1}^{T}, \theta\big)$ (or report FAIL if ROUND fails).
```

Figure 2.3: Approximate inner maximization via Bregman projection—APPROXBREGMAN

Remark 2.5.1. *In our algorithm, we will only compute Bregman projections of distributions of support size at least $k'/2$. This is because neither our reweighting procedure nor the actual projection algorithm of [18] sets any coordinates to 0 and the initial weight is uniform.*

Since Algorithm 2.3 is the MWU method with Bregman projections onto the set $\mathcal{K}$, we will apply the following regret guarantee.[3]

Theorem 2.5.1 (Theorem 2.4 of [13])**.** *Suppose that for $\sigma_t^2(i) \in [0,1]$ for all $i \in [k']$ and $t \in [T]$. Then after T iterations of Algorithm 2.3, for any $p \in \mathcal{K}$, it holds that:*

$$\sum_{t=1}^{T} \left\langle \boldsymbol{\tau}_t, \boldsymbol{\sigma}_t^2 \right\rangle \leq \frac{3}{2} \sum_{t=1}^{T} \left\langle \boldsymbol{p}, \boldsymbol{\sigma}_t^2 \right\rangle + 2KL(\boldsymbol{p}||\boldsymbol{\tau}_1).$$

Finally, we comment that we cannot naïvely apply the power method for the singular vector computation. The power method has failure probability of $\frac{1}{10}$, whereas our algorithm should fail with probability at most $\delta = O(\exp(-k))$ that is exponentially low. However, we note that the algorithm computes the top singular vectors of a sequence of matrices $\boldsymbol{A}_1, \boldsymbol{A}_2, \ldots, \boldsymbol{A}_T$. Observe that as long as $T = \Omega(\log(1/\delta)) = \Omega(k)$, with probability at least

[3]To be more precise, the iterations t in which $\|\boldsymbol{A}_t\boldsymbol{w}_t\|_2^2 \geq \frac{\theta^2}{10}$ behave according to the MWU method. Whenever $\|\boldsymbol{A}_t\boldsymbol{w}_t\|_2^2 < \frac{\theta^2}{10}$, the algorithm does not update the weights, which has no effect on the other iterations.

$1 - \delta/8$, the power method will succeed for $0.9T$ of the matrices. We will show that this many successes suffice to guarantee correctness of our algorithm.

We first prove the following lemma, a requirement for the rounding algorithm to succeed.

Lemma 2.5.3 (regret analysis)**.** *After* $T = O\left(\max\left(\frac{\log k'}{\theta^2}, \log(T_{\mathsf{des}}/\delta)\right)\right)$ *iterations of Algorithm 2.3, for all but a* $1/4$ *fraction of* $i \in [k']$*:*

$$\sum_{t=1}^{T} \langle \boldsymbol{Z}'_i, \boldsymbol{w}_t \rangle^2 \geq 100 \log k'.$$

Proof. Let $S = \{i \in [k'] : |\langle \boldsymbol{Z}'_i, \boldsymbol{w}^* \rangle| \geq \theta\}$ be the set of constraints satisfied by the unit vector $\boldsymbol{w}^*$ whose existence is guaranteed in the hypothesis of Lemma 2.5.2. By assumption, we have that $|S| \geq 0.8k'$. We simply calculate each of the terms in Theorem 2.5.1.

First, let $\mathcal{I} = \{t \in [T] : \boldsymbol{w}_t$ is a $1/2$-approximate top singular vector of $\boldsymbol{A}_t\}$. Then we have for any $t \in \mathcal{I}$:

$$\begin{aligned}
\left\langle \boldsymbol{\tau}_t, \boldsymbol{\sigma}_t^2 \right\rangle &= \sum_{i=1}^{k'} \boldsymbol{\tau}_t(i) \langle \boldsymbol{Z}'_i, \boldsymbol{w}_t \rangle^2 && \text{(by definition)} \\
&\geq \frac{1}{2} \sum_{i=1}^{k'} \boldsymbol{\tau}_t(i) \langle \boldsymbol{Z}'_i, \boldsymbol{w}^* \rangle^2 && \text{(because } \boldsymbol{w}_t \text{ is an approximate top eigenvector)} \\
&\geq \frac{1}{2} \sum_{i \in S} \boldsymbol{\tau}_t(i) \langle \boldsymbol{Z}'_i, \boldsymbol{w}^* \rangle^2 \\
&\geq \frac{1}{2} \sum_{i \in S} \boldsymbol{\tau}_t(i) \theta^2 && \text{(by definition of } S\text{)} \\
&\geq \frac{1}{2} \cdot \frac{1}{5} \theta^2 = \frac{\theta^2}{10} && \text{(because } |S| \geq 0.8k' \text{ and } \boldsymbol{\tau}_t \in \mathcal{K}\text{)}.
\end{aligned}$$

Summing this inequality over $t \in [T]$, we have that

$$\sum_{t=1}^{T} \left\langle \boldsymbol{\tau}_t, \boldsymbol{\sigma}_t^2 \right\rangle \geq \sum_{t \in \mathcal{I}} \left\langle \boldsymbol{\tau}_t, \boldsymbol{\sigma}_t^2 \right\rangle \geq \frac{|\mathcal{I}|}{10} \theta^2.$$

By Chernoff-Hoeffding bound combined with the guarantee of power iteration (Fact A.2.1), as long as $T = \Omega(\log(T_{\mathsf{des}}/\delta))$, then with probability at least $1 - \frac{\delta}{8T_{\mathsf{des}}}$, for at least $\frac{4}{5}T$ iterations, it will be the case that $\boldsymbol{w}_t$ is an approximate top singular vector. In other words, $|\mathcal{I}| \geq \frac{4}{5}T$, so that we have:

$$\sum_{t=1}^{T} \left\langle \boldsymbol{\tau}_t, \boldsymbol{\sigma}_t^2 \right\rangle \geq \frac{2T}{25} \theta^2.$$

Next, note that if we choose $\boldsymbol{p} = \boldsymbol{e}_i$, then

$$\sum_{t=1}^{T} \left\langle \boldsymbol{p}, \boldsymbol{\sigma}_t^2 \right\rangle = \sum_{t=1}^{T} \left\langle \boldsymbol{Z}_i', \boldsymbol{w}_t \right\rangle^2 .$$

Because $\boldsymbol{\tau}_1$ is uniform, the relative entropy term in Theorem 2.5.1 is at most $\log k'$. Let's pretend for a moment that $\boldsymbol{e}_i \in \mathcal{K}$ (it is not). Then after plugging in the above calculations to Theorem 2.5.1 and rearranging, we have that for every $i \in [k']$

$$\sum_{t=1}^{T} \left\langle \boldsymbol{Z}_i', \boldsymbol{w}_t \right\rangle^2 \geq \frac{2T}{25}\theta^2 - 2\log k' \geq 100 \log k',$$

by setting $T \geq \frac{10^5 \log k'}{\theta^2}$. This gives the bound claimed in the statement of the lemma, but it remains to fix the invalid assumption that $\boldsymbol{e}_i \in \mathcal{K}$. To do so, we will construct, for most $i \in [k']$, another distribution $\boldsymbol{p}' \in \mathcal{K}$ such that

$$\sum_{t=1}^{T} \left\langle \boldsymbol{e}_i, \boldsymbol{\sigma}_t^2 \right\rangle \geq \sum_{t=1}^{T} \left\langle \boldsymbol{p}', \boldsymbol{\sigma}_t^2 \right\rangle .$$

Combining this with $\sum_{t=1}^{T} \left\langle \boldsymbol{p}', \boldsymbol{\sigma}_t^2 \right\rangle \geq 100 \log k'$ gives the desired lower bound, for most i. Write $\boldsymbol{\alpha} = \sum_{t=1}^{T} \boldsymbol{\sigma}_t^2$, and without loss of generality assume that

$$\boldsymbol{\alpha}_1 \geq \boldsymbol{\alpha}_2 \geq \ldots \geq \boldsymbol{\alpha}_{k'} .$$

For $i = 1, \ldots, 4k'/5$, take $\boldsymbol{p}'$ to be uniform on those $j \in [k']$ such that $\boldsymbol{\alpha}_i \geq \boldsymbol{\alpha}_j$ (there are at least $k'/5$ such i). By construction, we have that $\langle \boldsymbol{\alpha}, \boldsymbol{e}_i \rangle \geq \langle \boldsymbol{\alpha}, \boldsymbol{p}' \rangle$. Finally, observe that $\boldsymbol{p}' \in \mathcal{K}$ because $\boldsymbol{p}'$ is uniform on a set of size at least $k'/5$. □

Observe that the ApproxBregman produces a sequence of vectors by the end. [101] provides a rounding algorithm that combines them into one with the desired margin bound. We describe the algorithm and prove the following lemma in Appendix A.6.

Lemma 2.5.4. *The algorithm* Round *(Algorithm A.4) outputs $\boldsymbol{w}$ that satisfies $|\langle \boldsymbol{Z}_i', \boldsymbol{w} \rangle| \geq 0.1\theta$ for $0.45k$ of the points, with probability at least $1 - \delta/4T_{\mathsf{des}}$.*

Finally, we are now ready to prove the key lemma using ApproxBregman.

Proof of Lemma 2.5.2. The correctness follows from Lemma 2.5.4. We focus on run-time. By Assumption 2.5.1, we have that $1/\theta^2 = O(d)$. By projecting onto the subspace spanned by the bucket means, we can assume $d \leq k$. Hence, Lemma 2.5.3 implies that the iteration count is $\widetilde{O}(k')$. The runtime of each iteration is bounded by the cost of computing an approximate top singular vector of a k' by d matrix via the power method, which is $\widetilde{O}(k'd)$ by Fact A.2.1. Finally, each repetition of the rounding algorithm Round takes time $\widetilde{O}(k'd)$, and the number of trials is at most $O(\log(1/\delta'))$ by definition. Thus, the runtime of the rounding algorithm is $\widetilde{O}(k^2 d)$. □

Putting it Together

Our main algorithm begins with the initial guess as the coordinate-wise median-of-means of $\{\boldsymbol{Z}_i\}_{i=k+1}^{2k}$. Then it proceeds via the Descent procedure, where the gradient and distance estimates are given by ApproxBregman. To ensure independence, we only use the $\{\boldsymbol{Z}_i\}_{i=1}^{k+1}$ for the descent part. We provide the full description in Appendix A.1.

We now give a proof sketch our main theorem. The formal proof is found in Appendix A.7.

Proof sketch of Theorem 2.1.1. Our argument is conditioned on (i) that the Lugosi-Mendelson condition holds, (ii) that the initial guess $\boldsymbol{x}_0$ satisfies an error bound $\sqrt{kd\|\boldsymbol{\Sigma}\|/n}$, and (iii) that the Prune procedure succeeds. Each fails with probability at most $\delta/8$.

The guarantee of ApproxBregman, along with Lemma 2.4.1, implies that GradEst and DistEst succeed with probability at least $1-\delta/4T_{\mathsf{des}}$ each iteration. Taking union bound over all above events, the failure probability of the final algorithm is at most δ. Applying the guarantee of the Descent procedure and error bound of the initial guess finishes the proof. □

2.6 Conclusion and Discussion

In this paper, we provided a faster algorithm for estimating the mean of a heavy-tailed random vector that achieves subgaussian performance. Unlike previous algorithms, our faster running time is achieved by the use of a simple spectral method that iteratively updates the current estimate of the mean until it is sufficiently close to the true mean.

Our work suggests two natural directions for future research. First, is it possible to achieve subgaussian performance for heavy-tailed *covariance* estimation in polynomial time? Currently, the best polynomial-time covariance estimators do not achieve the optimal statistical rate (see [123, 43]), while a natural generalization of the (computationally intractable) Lugosi-Mendelson estimator is known to achieve subgaussian performance. One approach would be to build on our framework; the key technical challenge is to design an efficient subroutine for producing bi-criteria approximate solutions to the natural generalization of the inner maximization problem to the covariance setting.

Another direction is to achieve a truly linear-time algorithm for the mean estimation problem. Our iterative procedure for solving the inner maximization problem take $\widetilde{O}(k)$ iterations; is it possible to reduce this to a constant?

Chapter 3

Unifying Robust and Heavy-Tailed Mean Estimation

In this chapter, we study the problem of estimating the mean of a distribution in high dimensions when either the samples are adversarially corrupted or the distribution is heavy-tailed. Recent developments in robust statistics have established efficient and (near) optimal procedures for both settings. However, the algorithms developed on each side tend to be sophisticated and do not directly transfer to the other, with many of them having ad-hoc or complicated analyses.

We provide a meta-problem and a duality theorem that lead to a new unified view on robust and heavy-tailed mean estimation in high dimensions. We show that the meta-problem can be solved either by a variant of the Filter algorithm from the recent literature on robust estimation or by the quantum entropy scoring scheme (QUE), due to Dong, Hopkins and Li (NeurIPS '19). By leveraging our duality theorem, these results translate into simple and efficient algorithms for both robust and heavy-tailed settings. Furthermore, the QUE-based procedure has run-time that matches the fastest known algorithms on both fronts.

Our analysis of Filter is through the classic regret bound of the multiplicative weights update method. This connection allows us to avoid the technical complications in previous works and improve upon the run-time analysis of a gradient-descent-based algorithm for robust mean estimation by Cheng, Diakonikolas, Ge and Soltanolkotabi (ICML '20).

3.1 Introduction

Learning from high-dimensional data in the presence of outliers is a central task in modern statistics and machine learning. Outliers have many sources. Modern data sets can be exposed to random corruptions or even malicious tampering, as in data poison attacks. Data drawn from heavy-tailed distributions can naturally contain outlying samples—heavy-tailed data are found often in network science, biology, and beyond [69, 114, 17, 6]. Minimizing the effect of outliers on the performance of learning algorithms is therefore a key challenge for statistics and computer science.

Robust statistics—that is, statistics in the presence of outliers—has been studied formally since at least the 1960s, and informally since long before [91, 154]. However, handling outliers in high dimensions presents significant computational challenges. Classical robust estimators (such as the Tukey median) suffer from worst-case computational hardness, while naïve computationally-efficient algorithms (e.g., throwing out atypical-looking samples) have far-from-optimal rates of error. In the last five years, however, numerous works have developed sophisticated, efficient algorithms with optimal error rates for a variety of problems in high-dimensional robust statistics. Despite significant recent progress, many basic algorithmic questions remain unanswered, and many algorithms and rigorous approaches to analyzing them remain complex and *ad hoc.*

In this work, we revisit the most fundamental high-dimensional estimation problem, estimating the mean of a distribution from samples, in the following two basic and widely-studied robust settings. In each case, $X_1, \ldots, X_n \in \mathbb{R}^d$ are independent samples from an unknown d-dimensional distribution D with mean $\mu \in \mathbb{R}^d$ and (finite) covariance $\Sigma \in \mathbb{R}^{d \times d}$.

- *Robust mean estimation:* Given $Y_1, \ldots, Y_n \in \mathbb{R}^d$ such that $Y_i = X_i$ except for εn choices of i, estimate the mean μ. We interpret the εn *contaminated* samples $Y_i \neq X_i$ as corruptions introduced by a malicious adversary. Naïve estimators such as the empirical mean can suffer arbitrarily-high inaccuracy as a result of these malicious samples.

- *Heavy-tailed mean estimation:* Given $X_1, \ldots, X_n$, estimate μ by an estimator $\hat{\mu}$ such that $\|\mu - \hat{\mu}\|$ is small with high probability (or equivalently, estimate μ with optimal confidence intervals). Since our only assumption about D is that it has finite covariance, D may have heavy tails. Standard estimators such as the empirical mean can therefore be poorly concentrated.

A significant amount of recent work in statistics and computer science has led to an array of algorithms for both problems with provably-optimal rates of error and increasingly-fast running times, both in theory and experiments [108, 60, 64, 39, 89, 42, 110, 112]. However, several questions remain, which we address in this work.

First, the relationship between heavy-tailed and robust mean estimation is still murky: while algorithms are known which simultaneously solve both problems to information-theoretic optimality [110], we lack general conditions under which algorithms for one problem also solve the other. This suggests:

Question 1: Is there a formal connection between robust mean estimation and heavy-tailed mean estimation which can be exploited by efficient algorithms?

Second, iterated *sample downweighting* (or pruning) is arguably the most natural approach to statistics with outliers—indeed, the *filter*, one of the first computationally efficient algorithms for optimal robust mean estimation [60]) takes this approach—but rigorous analyses of filter-style algorithms remain *ad hoc.* Other iterative methods, such as gradient descent, suffer the same fate: they are simple-to-describe algorithms which require significant creativity to analyze [41]. We ask:

> *Question 2: Is there a simple and principled approach to rigorously analyze iterative algorithms for robust and heavy-tailed mean estimation?*

Our Results

Our main contribution in this work is a simple and unified treatment of iterative methods for robust and heavy-tailed mean estimation.

Addressing Question 1, we begin by distilling a simple meta-problem, which we call *spectral sample reweighing*. While several variants of spectral sample reweighing are implicit in recent algorithmic robust statistics literature, our work is the first to separate the problem from the context of robust mean estimation and show the reduction from heavy-tailed mean estimation. The goal in spectral sample reweighing is to take a dataset $\{x_i\}_{i\in[n]} \subseteq \mathbb{R}^d$, reweigh the vectors x_i according to some weights $w_i \in [0,1]$, and find a center $\nu \in \mathbb{R}^d$ such that after reweighing the maximum eigenvalue of the covariance $\sum_{i\le n} w_i(x_i-\nu)(x_i-\nu)^\top$ is as small as possible.

Definition 3.1.1 ((α,ε) spectral sample reweighing, informal, see Definition 3.3.1)**.** *For $\varepsilon \in (0,1/2)$, let $\mathcal{W}_{n,\varepsilon} = \{w \in \Delta_n : \|w\|_\infty \le \frac{1}{(1-\varepsilon)n}\}$ be the set of probability distributions on $[n]$ with bounded ℓ_∞ norm. Let $\alpha \ge 1$. Given $\{x_i\}_{i=1}^n$ in $\mathbb{R}^d$, an α-approximate spectral sample reweighing algorithm returns a probability distribution $w \in \mathcal{W}_{n,3\varepsilon}$ and a* spectral center *$\nu \in \mathbb{R}^d$ such that*

$$\left\| \sum_{i\le n} w_i(x_i-\nu)(x_i-\nu)^\top \right\| \le \alpha \cdot \min_{w'\in\mathcal{W}_{n,\varepsilon},\nu'\in\mathbb{R}^d} \left\| \sum_{i\le n} w'_i(x_i-\nu')(x_i-\nu')^\top \right\| ,$$

where $\|\cdot\|$ denotes the spectral norm, or maximum eigenvalue.

Note that that spectral sample reweighing is a *worst-case* computational problem. The basic optimization task underlying spectral sample reweighing is to find weights $w \in \mathcal{W}_{n,\varepsilon}$ minimizing the spectral norm of the weighted second moment of $\{x_i-\nu\}_{i\in[n]}$. An α-approximation is allowed to output instead w in the slightly larger set $\mathcal{W}_{n,3\varepsilon}$ and may only minimize the spectral norm up to a multiplicative factor α. The parameter ε should be interpreted as the degree to which $w \in \mathcal{W}_{n,\varepsilon}$ may deviate from the uniform distribution.

Our first result shows that robust and heavy-tailed mean estimation both reduce to spectral sample reweighing.

Theorem 3.1.1 (Informal, see Theorem 3.4.1, Theorem 3.6.1)**.** *Robust and heavy-tailed mean estimation can both be solved with information-theoretically optimal error rates (up to constant factors) by algorithms which make one call to an oracle providing a constant-factor approximation to spectral sample reweighing (with $\varepsilon = \varepsilon_0$ a small universal constant) and run in additional time $\widetilde{O}(nd)$.*

For robust mean estimation this reduction is implicit in [60] and others (see e.g. [64]). For heavy-tailed mean estimation the reduction was not previously known: we analyze it by a simple convex duality argument (borrowing techniques from [39, 110]). Our argument gives a new equivalence between two notions of a *center* for a set of high-dimensional vectors—the spectral center considered in spectral sample reweighing and a more combinatorial notion developed by Lugosi and Mendelson in the context of heavy-tailed mean estimation [126]. We believe this equivalence is of interest in its own right—see Proposition 3.5.1 and Proposition 3.5.2.

We now turn our attention to Question 2. We offer a unified approach to rigorously analyzing several well-studied algorithms by observing that each in fact instantiates a common strategy for *online convex optimization*, and hence can be analyzed by applying a standard regret bound. This leads to the following three theorems. We first demonstrate that the filter, one of the first algorithms proposed for efficient robust mean estimation [60, 117, 58, 53], can be analyzed in this framework. Specifically, we show:

Theorem 3.1.2 ([60], Informal, see Theorem 3.3.1)**.** *There is an algorithm,* FILTER*, based on multiplicative weights, which gives a constant-factor approximation to spectral sample reweighing for sufficiently small ε, in time $\widetilde{O}(nd^2)$*[1]*.*

Previous approaches to analyzing the filter required by-hand construction of potential functions to track the progress of the algorithm. Our novel strategy to prove Theorem 3.1.2 demystifies the analysis of the filter by applying an out-of-the-box regret bound: the result is a significantly simpler proof than in prior work. It allows us to capture robust mean estimation in both bounded covariance and sub-gaussian setting.

Moving on, we also analyze gradient descent, giving the following new result, which we also prove by applying an out-of-the-box regret bound. Although it gives weaker running-time bound than we prove for FILTER, the advantage is that the algorithm is vanilla gradient descent. (By comparison, the multiplicative weights algorithm of Theorem 3.1.2 can be viewed as a more exotic mirror-descent method.)

Theorem 3.1.3 (Informal, see Theorem B.4.1)**.** *There is a gradient-descent based algorithm for spectral sample reweighing which gives a constant-factor approximation to spectral sample reweighing in $O(nd^2/\varepsilon^2)$ iterations and $\widetilde{O}(n^2d^3/\varepsilon^2)$ time.*

Prior work analyzing gradient descent for robust mean estimation required sophisticated tools for studying non-convex iterative methods [41]. Our regret-bound strategy shows for the first time that gradient descent solves heavy-tailed mean estimation, and that it solves robust mean estimation in significantly fewer iterations than previously known (prior work shows a bound of $\widetilde{O}(n^2d^4)$ iterations in the robust mean estimation setting, where our bound gives $O(nd^2)$ iterations [41]).

[1]We use $\widetilde{O}, \widetilde{\Omega}$ notation to hide polylogarithmic factors. Also, we remark that a variant of our main algorithm achieves the optimal breakdown point of 1/2; see Section B.5.

Finally, we demonstrate that the nearly-linear time algorithm for robust mean estimation in [64] fits into this framework as well. Thus, this framework captures state-of-the-art algorithms for robust mean estimation.

Theorem 3.1.4 ([64], Informal, see Theorem B.3.1)**.** *There is an algorithm based on matrix multiplicative weights which gives a constant-factor approximation to spectral sample reweighing for sufficiently small ε, in time $\widetilde{O}(nd\log(1/\varepsilon))$.*

Related work

For robust mean estimation, [60, 108] give the first polynomial-time algorithm with optimal (dimension-independent) error rates. Their results have been further improved and generalized by a number of works [16, 58, 61, 55, 88, 148, 64, 62, 40, 57]. See [53] for a complete survey.

The first (computationally inefficient) estimator to obtain optimal confidence intervals for heavy-tailed distributions in high dimensions is given by [126]; this construction was first made algorithmic by [89], using the Sum-of-Squares method. Later works [42, 110, 112] significantly improve the run-time: the fastest known algorithm runs in time $\widetilde{O}(n^2 d)$.

Analyses of the Filter algorithm are scattered around the literature [60, 117, 58, 53]. The variant of Filter we present here is based on a soft downweighting procedure first proposed by [147]. However, no prior work analyzes Filter through the lens of regret minimization or points out a connection with the heavy-tailed setting.

Prior works [110, 138, 122] have proposed unified approaches to heavy-tailed and robust mean estimation. In particular, [110] observes a robustness guarantee of [126], originally designed for the heavy-tailed setting. However, these works do not distill a meta-problem or obtain the analysis via duality. In addition, although it matches the fastest-known running time in theory, the algorithm of [110] is based on semidefinite programming, rendering it relatively impractical. Some constructions from [138, 122] are not known to be computationally tractable.

In a concurrent and independent work, [160] also studies the spectral sample reweighing problem (in the context of robust mean estimation), and provides an analysis of filter-type algorithms based on a regret bounds. The argument of [160] relies on a technical optimization landscape analysis, which our arguments avoid. The framework of [160] can be extended to robust linear regression and covariance estimation; it is unclear our techniques extend similarly. Their work also proves an optimal breakdown point analysis of filter-type algorithm for robust mean estimation. We obtain the same result (see Section B.5) with an arguably less sophisticated proof. Lastly, [160] does not discuss the heavy-tailed setting.

Another concurrent and independent work, [54], also shows that the filter and non-convex gradient descent obtain optimal rates in the robust and heavy-tailed settings. The authors also identify a general stability-based condition under which robust mean estimation algorithms achieve optimal rates in the heavy-tailed setting.

Organization

We formally introduce the spectral sample reweighing problem and analyze an algorithm based on the Filter algorithm in Section 3.3. We show how this primitive can be immediately used to solve the robust mean estimation problem in Section 3.4. Then in Section 3.5 we introduce the duality theorem that connect two notions of centrality. The result is used further in Section 3.6, where we show how to leverage the duality for heavy-tailed mean estimation.

3.2 Preliminaries

For a set of n real values α_i, we let $\mathsf{median}\left(\{\alpha_i\}_{i=1}^n\right)$ to denote its median. For a matrix A, we use $\|A\|, \|A\|_2$ to denote the spectral norm of A and $\mathrm{Tr}(A)$ its trace. For a vector v, $\|v\|_p$ denotes the ℓ_p norm. We denote the all-one vector of dimension k by $\mathbb{1}_k$. For vectors u, v, we denote the entrywise product by $u \odot v$; that is, the vector such that $w_i = u_i \cdot v_i$ for each i. For PSD matrices A, B, we write $A \preceq B$ if $B - A$ is PSD. Density matrices refer to the set of PSD matrices with unit trace. For any symmetric matrix $A \in \mathbb{R}^{d \times d}$, let $\exp(A)$ denote the matrix exponential of A. For a weight vector w such that $0 \le w_i \le 1$ and point set $\{x_i\}_{i=1}^n$, we define $\mu(w) = \sum_{i=1}^n w_i x_i$ and $M(w) = \Sigma_w = \sum_{i=1}^n w_i (x_i - \mu(w))(x - \mu(w))^\top$.

Definition 3.2.1 (approximate top eigenvector). *For any PSD matrix M and $c \in (0,1)$, we say that a unit vector v is a c-approximate largest eigenvector of M if $v^T M v \ge c\|M\|_2$.*

For a PSD matrix M, we let ApproxTopEigenvector(M, c, α) to denote an approximation scheme that outputs a (unit-norm) c-approximate largest eigenvector of M with a failure probability of at most α. The classic power method achieves such guarantee with a run-time of $O\left(\frac{1}{1-c} nd \log(1/\alpha)\right)$, when M is given in a factored form $M = X^\top X$, for $X \in \mathbb{R}^{n \times d}$.

Definition 3.2.2 (Kullback–Leibler divergence). *For probability distributions p, q over $[n]$, the KL divergence from q to p is defined as $KL(p||q) = \sum_{i=1}^n p(i) \log \frac{p(i)}{q(i)}$.*

Definition 3.2.3 (total variation distance). *For probability distributions p, q, the total variation distance is defined as $TV(p,q) = \sup_E |p(E) - q(E)| = \frac{1}{2}\|p - q\|_1$, where the supremum is over the set of measurable events.*

We use Δ_n to denote the set of probability distributions over $[n]$ and write $\mathcal{U}_n$ for the uniform distribution over $[n]$. We use $i \sim I$ to denote i drawn uniformly from an index set $I \subseteq [n]$. Throughout, we define $\mathcal{W}_{n,\varepsilon} = \{w \in \Delta_n : w_i \le \frac{1}{(1-\varepsilon)n}\}$ to be the discrete distributions over $[n]$ with bounded ℓ_∞ norm; we call the set *good weights*.

3.3 The Meta-Problem and a Meta-Algorithm

We now define the meta-problem, which we call *spectral sample reweighing*, that underlies both the adversarial and heavy-tailed models. We put it as a promise problem.

Definition 3.3.1 ((α, ε)-spectral sample reweighing)**.** *Let $\varepsilon \in (0, 1/2)$. The spectral sample reweighing problem is specified by the following.*

- *Input: n points $\{x_i\}_{i=1}^n$ in $\mathbb{R}^d$ and $\lambda \in \mathbb{R}$.*
- *Promise: There exists a $\nu \in \mathbb{R}^d$ and a set of good weights $w \in \mathcal{W}_{n,\varepsilon}$ such that*

$$\sum_{i=1}^{n} w_i \left(x_i - \nu\right) \left(x_i - \nu\right)^\top \preceq \lambda I. \tag{†}$$

- *Output: A set of good weights $w' \in \mathcal{W}_{n,3\varepsilon}$ and $\nu' \in \mathbb{R}^d$ that satisfies the condition above, up to the factor of $\alpha \geq 1$:*

$$\sum_{i=1}^{n} w'_i \left(x_i - \nu'\right) \left(x_i - \nu'\right)^\top \preceq \alpha\lambda I. \tag{3.3.1}$$

To provide some intuition, the goal here is to find a set of weights $\{w_i\}_{i=1}^n$, close to the uniform distribution on $[n]$, and a center ν such that by weighting by w and centering by ν, the covariance is bounded, under the promise that such a set of weights exists. We will refer to our promise (†) as a *spectral centrality* assumption.

Solving spectral sample reweighing The main result of this section is an efficient algorithm that achieves a constant factor approximation for the spectral sample reweighing problem.

Theorem 3.3.1 ([60] spectral sample reweighing via filter)**.** *Let $\{x_i\}_{i=1}^n$ be n points in $\mathbb{R}^d$ and $\varepsilon \in (0, 1/10]$. Suppose there exists $\nu \in \mathbb{R}^d$ and $w \in \mathcal{W}_{n,\varepsilon}$ such that*

$$\sum_{i=1}^{n} w_i \left(x_i - \nu\right) \left(x_i - \nu\right)^\top \preceq \lambda I$$

for some $\lambda > 0$. Then, given $\{x_i\}_{i=1}^n, \lambda, \varepsilon$ and a failure rate δ, there is an algorithm that finds $w' \in \mathcal{W}_{n,3\varepsilon}$ and $\nu' \in \mathbb{R}^d$ such that

$$\sum_{i=1}^{n} w'_i \left(x_i - \nu'\right) \left(x_i - \nu'\right)^\top \preceq 60\lambda I,$$

with probability at least $1 - \delta$.

The algorithm runs in $O(d)$ iterations and $\widetilde{O}\left(nd^2 \log(1/\delta)\right)$ time in total.

Our algorithm is a a multiplicative weights-style procedure. In particular, the output center ν' will be a weighted average of the points $\{x_i\}_{i=1}^n$. The algorithm starts with the uniform weighting and iteratively downweights points which are causing the empirical covariance to have a large eigenvalue. To ensure that we always maintain a set of good weights, we project the weights onto the set $\mathcal{W}_{n,\varepsilon}$ at the end of each iteration, according to KL divergence. For technical reasons, the algorithm also requires a *width* parameter ρ. It suffices to set it as the squared diameter of the input points $\{x_i\}_{i=1}^n$, and it can be bounded by $O(d\lambda/\varepsilon)$ by a simple pruning argument (Lemma 3.3.2 and Lemma 3.3.3).

The algorithm should be seen as a variant of the FILTER algorithm, due to Diakonikolas, Kamath, Kane, Li, Moitra, and Stewart [60]. The procedure we present here most resembles a more streamlined version later by Steinhart [147]. However, neither formulated the problem quite this way or gave this analysis. Instead, we will re-analyze the algorithm for the purpose of spectral sample reweighing and in a different manner than previously done in the literature.

Algorithm 1: Multiplicative weights for spectral sample reweighing (Definition 3.3.1)

```
Input: A set of points $\{x_i\}_{i=1}^n$, an iteration count $T$, and parameter $\rho, \delta$
Output: A point $\nu \in \mathbb{R}^d$ and weights $w \in \mathcal{W}_{n,\varepsilon}$.
1 Let $w^{(1)} = \frac{1}{n}\mathbb{1}_n$ and $\eta = 1/2$.
2 For $t$ from 1 to $T$
3     Let $\nu^{(t)} = \sum_i w_i^{(t)} x_i$, $M^{(t)} = \sum_i w_i^{(t)}(x_i - \nu^{(t)})(x_i - \nu^{(t)})^T$.
4     Compute $v^{(t)} = \textsc{ApproxTopEigenvector}(M^{(t)}, 7/8, \delta/T)$.
5     Compute $\tau_i^{(t)} = \left\langle v^{(t)}, x_i - \nu^{(t)} \right\rangle^2$ for each $i$.
6     Set $w_i^{(t+1)} \leftarrow w_i^{(t)} \left(1 - \eta\tau_i^{(t)}/\rho\right)$ for each $i$.
7     Project $w^{(t+1)}$ onto the set of good weights $\mathcal{W}_{n,\varepsilon}$ (under KL divergence):
          $$w^{(t+1)} \leftarrow \arg\min_{w \in \mathcal{W}_{n,\varepsilon}} \mathrm{KL}\left(w \| w^{(t)}\right).$$
8 Return $\nu^{(t^*)}, w^{(t^*)}$, where $t^* = \arg\min_t \|M^{(t)}\|$.
```

Lemma 3.3.1 (analysis of filter)**.** *Let $\varepsilon \in (0, 1/10]$ and $\{x_i\}_{i=1}^n$ be n points in $\mathbb{R}^d$. Suppose there exists $\nu \in \mathbb{R}^d$ and $w \in \mathcal{W}_{n,\varepsilon}$ such that*

$$\sum_{i=1}^n w_i (x_i - \nu)(x_i - \nu)^\top \preceq \lambda I$$

for some $\lambda > 0$. Then, given $\{x_i\}_{i=1}^n$, a failure rate δ and ρ such that $\rho \geq \tau_i^{(t)}$ for all i and

t, Algorithm 1 finds $w' \in \mathcal{W}_{n,\varepsilon}$ and $\nu' \in \mathbb{R}^d$ such that

$$\sum_{i=1}^{n} w'_i \left(x_i - \nu'\right)\left(x_i - \nu'\right)^\top \preceq 60\lambda I, \tag{3.3.2}$$

with probability at least $1-\delta$.

The algorithm terminates in $T = O(\rho\varepsilon/\lambda)$ iterations. Further, if $T = O(poly(n,d))$, then each iteration takes $\widetilde{O}(nd\log(1/\delta))$ time.

We first see how to prove Theorem 3.3.1 via Lemma 3.3.1. Note that it requires to bound the width parameter ρ. To ensure the condition $\rho \geq \tau_i^{(t)}$ for all i and t, observe that as $\|v^{(t)}\| = 1$, we have

$$\tau_i^{(t)} = \left\langle v^{(t)}, x_i - \nu^{(t)} \right\rangle^2 \leq \|x_i - \nu^{(t)}\|^2.$$

Also, since $\nu^{(t)}$ is a convex combination of $\{x_i\}_{i=1}^n$, we can set ρ to be the squared diameter of the input data $\{x_i\}_{i=1}^n$. As the first step, we show that a $(1-2\varepsilon)$ fraction of the points lie within a ball of radius $\sqrt{d\lambda/\varepsilon}$ under the spectral centrality condition. Then a (folklore) pruning procedure can be used to extract such set.

Lemma 3.3.2 (diameter bound)**.** *Let $\{x_i\}_{i=1}^n$ be n points in $\mathbb{R}^d$. Suppose there exists $\nu \in \mathbb{R}^d$ and $w \in \mathcal{W}_{n,\varepsilon}$ such that $\sum_{i=1}^n w_i \left(x_i - \nu\right)\left(x_i - \nu\right)^\top \preceq \lambda I$ for some $\lambda > 0$ and $\varepsilon \in (0, 1/2)$. Then there exists a ball of radius $\sqrt{d\lambda/\varepsilon}$ that contains at least $r = (1-2\varepsilon)n$ points of $\{x_i\}_{i=1}^n$.*

The proof of the lemma can be found in Appendix B.1

Lemma 3.3.3 (folklore; see [64])**.** *Let $\varepsilon < 1/2$ and $\delta > 0$. Let $S \subset \mathbb{R}^d$ be a set of n points. Assume there exists a ball B of radius r and a subset $S' \subseteq S$ such that $|S'| \geq (1-\varepsilon)n$ and $S' \subset B$. Then there is an algorithm* Prune(S, r, δ) *that runs in time $O(nd\log 1/\delta)$ and with probability $1-\delta$ outputs a set $R \subseteq S$ so that $S' \subseteq R$, and R is contained in a ball of radius $4r$.*

Using the lemmas above, we can immediately prove the main theorem.

Proof of Theorem 3.3.1. Given $S = \{x_i\}_{i=1}^n$, λ and ε, we first run the Prune$(S, r, \delta/2)$ algorithm, with $r = \sqrt{d\lambda/\varepsilon}$. By Lemma 3.3.2, the spectral centrality condition (†) implies there exists a ball of radius r containing at least $(1-2\varepsilon)n$ points of S. Therefore, Lemma 3.3.3 guarantees that it will return a set $R \subseteq S$ of at least $(1-2\varepsilon)n$ points contained in a ball of radius $4r$. Hence by Lemma 3.3.1, given R, $\rho = 16d\lambda/\varepsilon$ and failure rate $\delta/2$, Algorithm 1 finds $w' \in \mathcal{W}_{|R|,\varepsilon}$ and $\nu' \in \mathbb{R}^d$ such that

$$\sum_{i \in R} w'_i \left(x_i - \nu'\right)\left(x_i - \nu'\right)^\top \preceq 60\lambda I,$$

with probability at least $1-\delta/2$. Let $w''_i = w'_i$ if $i \in R$ and $w''_i = 0$ otherwise. since $\frac{1}{(1-\varepsilon)(1-2\varepsilon)} \leq \frac{1}{1-3\varepsilon}$ for $\varepsilon < 1/3$, we have $w'' \in \mathcal{W}_{n,3\varepsilon}$ Moreover, $\sum_{i=1}^n w''_i \left(x_i - \nu'\right)\left(x_i - \nu'\right)^\top \preceq 60\lambda I$, as desired.

The overall procedure succeeds with probability at least $1-\delta$ by a union bound, since Algorithm 1 and PRUNE are both set up to have a failure rate at most $\delta/2$. Now for the run-time, $\text{PRUNE}(S, r, \delta)$ takes $O(nd\log(1/\delta))$ by Lemma 3.3.3. Moreover, by Lemma 3.3.1, Algorithm 1 runs in time $\widetilde{O}(nd\log(1/\delta)\cdot T)$ time, with $T = O(\rho\varepsilon/\lambda)$ being the iteration count. Since $\rho = 16d\lambda/\varepsilon$, we have $T = O(d)$, and this immediately yields the desired runtime. □

Analysis via regret minimization Now it remains to analyze Algorithm 1 (Lemma 3.3.1). We will cast the algorithm under the framework of regret minimization using multiplicative weights update (MWU). To see that, we consider $\{x_i\}_{i=1}^n$ as the set of actions, $w^{(t)}$ as our probability distribution over the actions at time t, and we receive a loss vector $\tau^{(t)}$ each round. The weights are updated in a standard fashion. Then, to ensure that the weights lie in the constraint set $\mathcal{W}_{n,\varepsilon}$, we perform a projection step. (Note that the algorithm is implementing both the player and the adversary.) The following is a classic regret bound of MWU for the online linear optimization problem.

Lemma 3.3.4 (regret bound [13])**.** *Suppose $\rho \geq \tau_i^{(t)}$ for every t and i. Then for any weight $w \in \mathcal{W}_{n,\varepsilon}$, Algorithm 1 satisfies that*

$$\frac{1}{T}\sum_{t=1}^{T}\left\langle w^{(t)}, \tau^{(t)}\right\rangle \leq \frac{1}{T}(1+\eta)\sum_{t=1}^{T}\left\langle w, \tau^{(t)}\right\rangle + \frac{\rho \cdot KL(w\|w^{(1)})}{T\eta}, \tag{3.3.3}$$

for any choice of step size $\eta \leq 1/2$.

In addition, we claim the following lemma and delay its proof to the appendix (Lemma B.1.1).

Lemma 3.3.5. *Under the centrality promise (†), for any $w' \in \mathcal{W}_{n,\varepsilon}$,*

$$\|\nu - \nu(w')\| \leq \frac{1}{1-\sqrt{2\varepsilon}}\left(\sqrt{2\lambda} + \sqrt{2\varepsilon\|M(w')\|}\right), \tag{3.3.4}$$

where $\nu(w') = \sum_i w_i' x_i$ and $M(w') = \sum_i w_i'(x_i - \nu(w'))(x_i - \nu(w'))^\top$.

This type of inequality is generally known as the *spectral signature* lemma from the recent algorithmic robust statistics literature; see [117, 53].

With these technical ingredients, we are now ready to analyze the algorithm.

Proof of Lemma 3.3.1. Notice first that since $v^{(t)}$ is a 7/8-approximate largest eigenvector of $M^{(t)} = \sum_i w_i^{(t)}(x_i - \nu^{(t)})(x_i - \nu^{(t)})^T$, then for all t,

$$\sum_i w_i^{(t)}\tau_i^{(t)} = \sum_i w_i \left\langle v^{(t)}, x_i - \nu^{(t)}\right\rangle^2 = v^{(t)\top}M^{(t)}v^{(t)} \geq \frac{7}{8}\left\|M^{(t)}\right\|_2. \tag{3.3.5}$$

Let w be the good weights that satisfies our centrality promise (†). Summing over the T rounds and applying the the regret bound (Lemma 3.3.4), we obtain that

$$\frac{7}{8T}\sum_{t=1}^{T}\left\|M^{(t)}\right\|_2 \leq \frac{1}{T}\sum_{t=1}^{T}\left\langle w^{(t)},\tau^{(t)}\right\rangle \leq (1+\eta)\frac{1}{T}\sum_{t=1}^{T}\left\langle w,\tau^{(t)}\right\rangle + \frac{\rho\cdot \mathrm{KL}(w\|w^{(1)})}{T\eta}.$$

The KL term can be bounded because w and $w^{(1)}$ are both close to uniform. Indeed, it is a simple calculation to verify that $\mathrm{KL}(w\|w^{(1)}) \leq 5\varepsilon$, using the fact $w_i \leq 1/(1-\varepsilon)n$ (Lemma B.1.3). Plugging in $\eta = 1/2$, we get

$$\frac{7}{8T}\sum_{t=1}^{T}\left\|M^{(t)}\right\|_2 \leq \frac{3}{2T}\sum_{t=1}^{T}\left\langle w,\tau^{(t)}\right\rangle + \frac{10\varepsilon\rho}{T}. \tag{3.3.6}$$

Our eventual goal is to bound this by $O(\lambda)$. Note that the second term is easy to control—just set $T = \Omega(\rho\varepsilon/\lambda)$, and this will determine the iteration count and thus the runtime.

The remaining is mostly tedious calculations to bound the first term. The reader can simply skip forward to (3.3.12). For those interested: we proceed by expanding the first term on the right-hand side,

$$\frac{3}{2T}\sum_{t=1}^{T}\left\langle w,\tau^{(t)}\right\rangle = \frac{3}{2T}\sum_{t=1}^{T}\sum_{i=1}^{n} w_i\left\langle x_i-\nu^{(t)},v^{(t)}\right\rangle^2 \tag{3.3.7}$$

$$= \frac{3}{2T}\sum_{t=1}^{T}\sum_{i=1}^{n} w_i\left(\left\langle x_i-\nu,v^{(t)}\right\rangle^2 + \left\langle \nu-\nu^{(t)},v^{(t)}\right\rangle^2\right) \tag{3.3.8}$$

$$\leq \frac{3}{2}\lambda + \frac{3}{2T}\sum_{t=1}^{T}\left\langle \nu-\nu^{(t)},v^{(t)}\right\rangle^2 \tag{3.3.9}$$

$$\leq \frac{3}{2}\lambda + \frac{3}{2T}\sum_{t=1}^{T}\left\|\nu-\nu^{(t)}\right\|_2^2, \tag{3.3.10}$$

where (3.3.7) is by the definition that $\tau_i^{(t)} = \left\langle v^{(t)},x_i-\nu^{(t)}\right\rangle^2$, (3.3.8) uses the definition of $\nu^{(t)}$, (3.3.9) follows from the spectral centrality assumption (†), and (3.3.10) is by the fact that $\|v^{(t)}\| = 1$. Since $\nu^{(t)} = \sum_{i=1}^{n} w_i^{(t)}x_i$, we can apply Lemma 3.3.5 to bound $\|\nu-\nu^{(t)}\|$ and it follows that

$$\frac{3}{2T}\sum_{t=1}^{T}\left\|\nu-\nu^{(t)}\right\|_2^2 \leq \frac{3}{2T}\left(\sum_{t=1}^{T}\frac{25}{2}\lambda + \frac{1}{3}\left\|M^{(t)}\right\|_2\right),$$

for $\varepsilon \leq 1/10$. Plugging the bound into (3.3.10), we obtain

$$\frac{3}{2T}\sum_{t=1}^{T}\langle w,\tau^{(t)}\rangle \leq \frac{3}{2}\lambda + \frac{3}{2T}\left(\sum_{t=1}^{T}\frac{25}{2}\lambda + \frac{1}{3}\left\|M^{(t)}\right\|_2\right) = \frac{81}{4}\lambda + \frac{1}{2T}\sum_{t=1}^{T}\left\|M^{(t)}\right\|_2. \tag{3.3.11}$$

Finally, substituting this back into (3.3.6), we see that

$$\frac{7}{8T}\sum_{t=1}^{T}\left\|M^{(t)}\right\|_2 \le \frac{81}{4}\lambda + \frac{1}{2T}\sum_{t=1}^{T}\left\|M^{(t)}\right\|_2 + \frac{10\varepsilon\rho}{T}. \tag{3.3.12}$$

Now if we set $T = 10\rho\varepsilon/\lambda$, then the last term is λ. Rearranging yields that $\frac{1}{T}\sum_{t=1}^{T}\left\|M^{(t)}\right\|_2 \le 60\lambda$. This shows that within $T = O(\rho\varepsilon/\lambda)$ iterations we have achieved our goal (3.3.2).

Now it remain to argue the cost of each iteration. For approximating the largest eigenvector, the well-known power method computes a constant-approximation in $O(nd\log(1/\alpha))$ time with a failure probability at most α [107]. We set $\alpha = \delta/T$, and an application of union bound implies that all the T calls to the power method jointly succeed with probability at least $1-\delta$. This gives a total run-time of $\widetilde{O}(nd\log(1/\delta))$, since $T = O(\text{poly}(n,d))$, and bounds the overall failure probability of the algorithm by δ. Finally, we remark that the KL projection onto $\mathcal{W}_{n,\varepsilon}$ can be computed exactly in $O(n)$ time, by the deterministic procedures in [84, 157]. This completes the run-time analysis. $\square$

Faster algorithm Under the same assumptions, the spectral sample reweighing problem can be solved in $\widetilde{O}(nd\log(1/\delta))$ time, by adapting a *matrix* multiplicative weight scheme, due to Dong, Hopkins and Li [64]. The algorithm and its analysis generally follow from the proofs therein. The details can be found in Appendix B.3.

As we will see soon, applying this procedure directly match the fastest known algorithms for both robust and heavy-tailed settings.

Gradient descent analysis As we argued, Algorithm 1 is essentially an online linear optimization scheme, with the objective of minimizing $\sum_{t=1}^{T}\langle w^{(t)}, \tau^t\rangle$. It is known that the multiplicative weights rule employed here can be seen an entropic mirror descent update [150]. Therefore, it is natural to ask whether an additive update/gradient descent procedure would solve the problem as well. In Appendix B.4, we provide such an analysis (Theorem B.4.1). More importantly, the resulting scheme is equivalent of the gradient descent algorithm analyzed by [41]. Our analysis improves upon the iteration complexity from their work (in the concrete settings of robust mean estimation, under bounded second moment and sub-gaussian distributions).

3.4 Estimation under Corruptions

We now apply Algorithm 1 for the robust mean estimation problem. We focus on the bounded second moment distributions, where Algorithm 1 can be invoked in a black-box fashion. A slight variant of it can be used for the sub-gaussian setting, where we achieve a more refined analysis; see Appendix B.2.

The problem is formally defined below.

Definition 3.4.1 (robust mean estimation). *Given a distribution D over $\mathbb{R}^d$ with bounded covariance and a parameter $0 \leq \varepsilon < 1/2$, the adversary draws n i.i.d. samples D, inspects the samples, then removes at most εn points and replaces them with arbitrary points. We call the resulting dataset ε-corrupted (by an adaptive adversary).*

The goal is to estimate the mean of D only given the ε-corrupted set of samples.

Using a meta-algorithm for approximating the spectral sample reweighing problem, we will show the following. In particular, using Algorithm 1 matches the run-time and statistical guarantee of the original Filter algorithm.

Theorem 3.4.1 (robust mean estimation via sample reweighing). *Let D be a distribution over $\mathbb{R}^d$ with mean μ and covariance $\Sigma \preceq \sigma^2 I$ and $\varepsilon \leq 1/10$. Given an ε-corrupted set of $n = \Omega(d \log d/\varepsilon)$ samples, there is an algorithm that runs in time $\widetilde{O}(nd^2)$ that with constant probability outputs an estimate $\widehat{\mu}$ such that $\|\widehat{\mu} - \mu\| \leq O(\sigma\sqrt{\varepsilon})$.*

Further, the algorithm is via a black-box application of Algorithm 1, which can be replaced by any constant approximation algorithm for the spectral sample reweighing problem (Definition 3.3.1).

Information-theoretically, Theorem 3.4.1 is near optimal. It is known that the sample complexity of $d \log d/\varepsilon$ is tight, only up to the log factor. The estimation error $O(\sqrt{\varepsilon})$ is tight up to constant factor.

Our analysis requires a set of deterministic conditions to hold for the input, which follow from Lemma A.18 of [58]. This is meant to obtain the desired spectral centrality condition and to bound the final estimation error.

Lemma 3.4.1 (deterministic conditions [58]). *Let S be an ε-corrupted set of $\Omega(d \log d/\varepsilon)$ samples from D with mean μ and covariance $\Sigma \preceq I$. With high constant probability, S contains a subset G of size at least $(1-\varepsilon)n$ such that*

$$\|\mu - \mu_G\| \leq O(\sqrt{\varepsilon}) \tag{3.4.1}$$

$$\left\| \frac{1}{|G|} \sum_{i \in G} (x_i - \mu_G)(x_i - \mu_G)^\top \right\|_2 \leq O(1), \tag{3.4.2}$$

where $\mu_G = \frac{1}{|G|}\sum_{i \in G} x_i$.

We now prove the main result of this section—using the meta-algorithm to solve the robust mean estimation problem. Observe that it suffices to prove the theorem with $\sigma^2 = 1$. Without loss of generality, we can first divide every input sample by σ, execute the algorithm and then multiply the output by σ.

Proof of Theorem 3.4.1. First, we check that the centrality promise (†) is satisfied. This would ensure that we are in the setting of the spectral sample reweighing problem so that the meta-algorithm applies. Assume the conditions from Lemma 3.4.1. Then suppose we let $w_i = 1/|G|$ if $x_i \in G$ and $w_i = 0$ otherwise, so we have that $w \in \mathcal{W}_{n,\varepsilon}$, and let $\nu = \mu_G$ and $\lambda = O(1)$. Observe that (3.4.2) is exactly the spectral centrality condition (†) . Then we can apply Theorem 3.3.1 and obtain that the algorithm will find $\nu' \in \mathbb{R}^d$ and $w' \in \mathcal{W}_{n,3\varepsilon}$ such that

$$M(w') := \sum_{i=1}^{n} w'_i \left(x_i - \nu'\right)\left(x_i - \nu'\right)^\top \preceq O(1) \cdot I$$

Furthermore, by definition of the algorithm, ν' is a weighted average of the points $\{x_i\}_{i=1}^n$; that is, $\nu' = \nu(w') = \sum_{i=1}^n w'_i x_i$. This allows us again to apply the spectral signature lemma. In particular, Lemma B.1.2 implies

$$\|\mu_G - \nu'\| \le \frac{1}{1-6\varepsilon}\left(\sqrt{6\varepsilon\lambda} + \sqrt{3\varepsilon\|M(w')\|}\right) = O\left(\sqrt{\varepsilon}\right)$$

since $\lambda = O(1)$ and $\|M(w')\| = O(1)$. Finally, by triangle inequality and (3.4.1),

$$\|\mu - \nu'\| \le \|\mu_G - \nu'\| + \|\mu - \mu_G\| \le O(\sqrt{\varepsilon}).$$

Therefore, the output ν' estimates the true mean up to an error of $O(\sqrt{\varepsilon})$, as desired.

Finally, the run-time guarantee follows directly from the statement of Theorem 3.3.1, since we apply the meta-algorithm in a black-box fashion. This completes the proof. □

Optimal breakdown point In Section B.5, we show that a variant of the filter algorithm can be used to achieve the optimal breakdown point of 1/2. The result also appeared in a concurrent work [160], with an argubly more sophisticated proof.

Other algorithms To improve the computational efficiency, applying the same argument and using the matrix multiplicative weight algorithm (Theorem B.3.1), we can obtain a near linear time algorithm, which matches the fastest known algorithm for robust mean estimation [64, 40].

Corollary 3.4.1 (faster robust mean estimation [64])**.** *Let D be a distribution over $\mathbb{R}^d$ with mean μ and covariance $\Sigma \preceq \sigma^2 I$ and ε be a sufficiently small constant. Given an ε-corrupted set of $n = \Omega(d \log d/\varepsilon)$ samples, there is a matrix multiplicative update algorithm that runs in time $\widetilde{O}(nd)$ and with constant probability computes an estimate of error $O(\sigma\sqrt{\varepsilon})$.*

Since $\lambda = O(1)$ in the robust mean estimation problem under bounded covariance (Lemma 3.4.1), our analysis of the gradient descent algorithm (Theorem B.4.1) implies the following.

Corollary 3.4.2 (robust mean estimation via gradient descent)**.** *Let D be a distribution over $\mathbb{R}^d$ with mean μ and covariance $\Sigma \preceq \sigma^2 I$ and ε be a sufficiently small constant. Given an*

ε-corrupted set of $n = \Omega(d \log d/\varepsilon)$ samples, there is a gradient-descent based algorithm that computes an estimate of error $O(\sigma\sqrt{\varepsilon})$ with constant probability in $\widetilde{O}(nd^2/\varepsilon^2)$ iterations.[2]

A variant of the gradient descent-based algorithm can be used for robust mean estimation in the sub-gaussian setting as well; see Appendix 8.

3.5 Equivalent Notions of Centrality

In this section, we prove a duality statement that connects the setting of heavy-tailed and robust estimation. In particular, we will show that the following two (deterministic) notions of a *center* ν for points $\{x_i\}_{i=1}^k$ are essentially equivalent. We call them *spectral* and *combinatorial* center. The former is the requirement that showed up first in the original formulation of the spectral sample reweighing problem (Definition 3.3.1) and then in dealing with adversarial corruptions. The latter will yield the right notion of high-dimensional median for estimating the mean of heavy tailed data, now known as the *Lugosi-Mendelson estimator*, due to [126].

In the following, let $\{x_i\}_{i=1}^k$ be a set of k points in $\mathbb{R}^d$.

Spectral center Recall that our meta-problem of spectral sample reweighing (Definition 3.3.1) requires the assumption:

$$\min_{w \in \mathcal{W}_{k,\varepsilon}} \left\| \sum_{i=1}^{k} w_i \left(x_i - \nu\right)\left(x_i - \nu\right)^\top \right\| \leq \lambda. \tag{3.5.1}$$

Intuitively, this says that the data are roughly clustered around ν and no bad point significantly corrupts its shape. Note that by linearity, the objective can be rewritten as a minimax one, and this leads to the following definition

Definition 3.5.1 ((ε, λ)-spectral center)**.** *A point $\nu \in \mathbb{R}^d$ is a (ε, λ)-spectral center of $\{x_i\}_{i=1}^k$ if*

$$\min_{w \in \mathcal{W}_{k,\varepsilon}} \max_{M \succeq 0, Tr(M)=1} \sum_{i=1}^{k} w_i \left\langle (x_i - \nu)(x_i - \nu)^\top, M \right\rangle \leq \lambda. \tag{spectral center}$$

In the robust mean estimation setting, the deterministic conditions (Lemma 3.4.1) imply that the true mean is a $(\varepsilon, O(1))$-spectral center.

[2]The $1/\varepsilon$ dependence in the run-time can be removed by a simple bucketing trick due to [110]; also see Lemma B.1 of [64].

Combinatorial center On other hand, there is another natural way of saying that the data are centered around ν, which proves to be more useful in the heavy-tailed setting. We call it *combinatorial centrality* condition. It roughly says that when we project the data onto *any* one-dimensional direction, a majority of them will be close to ν.

Definition 3.5.2 ((ε, λ)-combinatorial center). *A point ν is a (ε, λ)-combinatorial center of $\{x_i\}_{i=1}^k$ if for all unit $v \in \mathbb{R}^d$.*

$$\sum_{i=1}^{k} \mathbb{1}\left\{\langle x_i - \nu, v\rangle \geq \sqrt{\lambda}\right\} \leq \varepsilon k, \qquad \text{(combinatorial center)}$$

Lugosi and Mendelson [126] show that optimal confidence intervals for mean estimation can be obtained in the heavy-tailed model by finding combinatorial centers. (We elaborate in the next section.)

Duality It turns out that for constant ε these two conditions are equivalent (up to some minor gaps in constants). To pave way for the proofs, a key observation, first made by [39], is that the left-side of (spectral center) is an SDP objective. (This is because it is simply minimizing the maximum eigenvalue of $\sum_i w_i(x_i-\nu)(x_i-\nu)^\top$.) And strong duality allows us to swap the min and max, so

$$\min_{w\in\mathcal{W}_{k,\varepsilon}} \max_{M} \sum_{i=1}^{k} w_i \left\langle (x_i-\nu)(x_i-\nu)^\top, M\right\rangle = \max_{M} \min_{w\in\mathcal{W}_{k,\varepsilon}} \sum_{i=1}^{k} w_i \left\langle (x_i-\nu)(x_i-\nu)^\top, M\right\rangle, \tag{3.5.2}$$

where the maximization is over the set of density matrices. Using this, we prove the following two propositions, showing (by contrapositives) that the two notions of centrality are equivalent. The constants in the statements are chosen only to serve the purpose of heavy-tailed mean estimation, and they can be tweaked easily by the same arguments.

We consider the easy direction first.

Proposition 3.5.1 (spectral center $\Longrightarrow$ combinatorial center). *If for some unit $v \in \mathbb{R}^d$*

$$\sum_{i=1}^{k} \mathbb{1}\left\{|\langle x_i - \nu, v\rangle| \geq 10\sqrt{\lambda}\right\} \geq 0.4k, \tag{3.5.3}$$

then we have that for $\varepsilon = 0.3$,

$$\min_{w\in\mathcal{W}_{k,\varepsilon}} \max_{M\succeq 0, Tr(M)=1} \sum_{i=1}^{k} w_i \left\langle (x_i-\nu)(x_i-\nu)^\top, M\right\rangle \geq \lambda.$$

Proof. The assumption (3.5.3) immediately implies that

$$\sum_{i=1}^{k} \mathbb{1}\left\{\langle x_i - \nu, v\rangle^2 \geq 100\lambda\right\} \geq 0.4k$$

This means that there are (at least) $0.4k$ points in $\{x_i\}_{i=1}^k$ such that

$$t_i := \left\langle (x_i - \nu)(x_i - \nu)^\top, M \right\rangle \geq 100\lambda$$

, where $M = vv^\top$. We call them outliers.

Now by the SDP duality (3.5.2), we only need to show that for *any* feasible w the objective is at least λ. Observe first that for a fixed M, the optimal w^* for the max-min objective is to put weight $1/(1-\varepsilon)k$ on the $(1-\varepsilon)k$ points with the smallest t_i. Recall we set $\varepsilon = 0.3$. Hence, by pigeonhole principle, the support of w^* must have an overlap of size $0.1k$ with the outliers. It follows that

$$\sum_{i=1}^{k} w_i^* \left\langle (x_i - \nu)(x_i - \nu)^\top, M \right\rangle \geq 0.1k \cdot \frac{1}{(1-0.3)k} \cdot 100\lambda \geq 10\lambda.$$

Since w^* is the optimal choice, this completes the proof. □

The other direction is a bit more involved. The key idea is to round the maximizing PSD matrix M into a single vector v, via gaussian sampling, and this part of the argument is due to [110].

Proposition 3.5.2 (combinatorial center $\Longrightarrow$ spectral center)**.** *Let $\varepsilon = 0.1$. If for some $\nu \in \mathbb{R}^d$*

$$\min_{w \in \mathcal{W}_{k,\varepsilon}} \max_{M \succeq 0, Tr(M)=1} \sum_{i=1}^{k} w_i \left\langle (x_i - \nu)(x_i - \nu)^\top, M \right\rangle \geq \lambda$$

then we have for some unit v,

$$\sum_{i=1}^{k} \mathbb{1}\left\{ |\langle x_i - \nu, v\rangle| \geq 0.1\sqrt{\lambda} \right\} \geq 0.01k.$$

Proof. Strong duality (3.5.2) implies that there exists PSD M of unit trace such that

$$\sum_{i=1}^{k} w_i \left\langle (x_i - \nu)(x_i - \nu)^\top, M \right\rangle \geq \lambda$$

for all $w \in \mathcal{W}_{k,\varepsilon}$. As we observed, the optimal w^* for a fixed M would put weights on the points with smallest value of $t_i = \left\langle (x_i - \nu)(x_i - \nu)^\top, M \right\rangle$. The fact that the objective is large implies that there must be more than $\varepsilon k = 0.1k$ points with $t_i \geq \lambda$. Let B be this set of points such that $t_i \geq \lambda$.

It remains to demonstrate a vector v such that

$$\sum_{i=1}^{k} \mathbb{1}\left\{|\langle x_i - \nu, v\rangle| \geq 0.1\sqrt{\lambda}\right\} \geq 0.01k. \tag{3.5.4}$$

The idea is to round the PSD matrix M to a single vector v that achieves this inequality. The right rounding method is simply *gaussian sampling*. Namely, if we draw $v_M \sim \mathcal{N}(0, M)$, then it can be shown that with constant probability $v = v_M/\|v_M\|$ satisfies the property above.

For that, we apply the argument from [110]. First let $g_i = \langle x_i - \nu, v_M\rangle$ for each $i \in [k]$. Note that g_i is a mean-zero Gaussian random variable with variance $\sigma_i^2 = t_i$. A standard anti-concentration calculation shows that for any $i \in B$, $\Pr(|g_i| \geq 0.5\sqrt{\lambda}) \geq 1/2$. Therefore, if we define

$$Y = \sum_{i=1}^{k} \mathbb{1}\left\{|\langle x_i - \nu, v\rangle| \geq 0.5\sqrt{\lambda}\right\},$$

then by linearity of expectations we have $\mathbb{E}\, Y \geq 0.05k$. It follows from the Payley-Zigmund inequality that $\Pr(Y \geq 0.01k) \geq 0.0018$. Moreover, by Borell-TIS inequality (Theorem 7.1 of [111]), we can bound that with probability at least 0.999,

$$\|v_M\| \leq \mathbb{E}\,\|v_M\| + 4\sqrt{\|M\|} \leq \sqrt{\mathrm{Tr}(M)} + 4\sqrt{\mathrm{Tr}(M)} \leq 5,$$

since $\mathrm{Tr}(M) = 1$. Combining these facts immediately proves (3.5.4). □

3.6 Estimation under Heavy-Tails

We now come to the heavy-tailed mean estimation problem and show how to solve it using the machinery developed in the last sections. The setting is very simple

Definition 3.6.1 (heavy-tailed mean estimation with optimal rates)**.** *Given n random vectors $\{X_i\}_{i=1}^n$ drawn i.i.d. from a distribution D over $\mathbb{R}^d$ with mean μ and (finite) covariance Σ and a desired confidence $2^{-O(n)} \leq \delta < 1$, compute an estimate $\widehat{\mu}$ such that with probability at least $1-\delta$,*

$$\|\widehat{\mu} - \mu\| \lesssim r_\delta \overset{\text{def}}{=} \sqrt{\frac{\mathrm{Tr}(\Sigma)}{n}} + \sqrt{\frac{\|\Sigma\|\log(1/\delta)}{n}}. \tag{3.6.1}$$

We note that the error rate (3.6.1) is information-theoretically optimal, up to a constant. The bound is known as *sub-gaussian* error, since when D is sub-gaussian, the empirical average obtains the guarantee. Moreover, in general, the estimator needs to depend on the parameter δ, and the requirement that $\delta \geq 2^{-O(n)}$ is necessary [32, 52]. In the following, we will aim only at a computationally efficient, δ-*dependent* construction that attains the optimal error r_δ.

Lugosi-Mendelson Estimator. In one dimension, the well-known *median-of-means* construction, due to [134, 92, 9], provides such strong guarantee:

(i) Bucket the data into $k = \lceil 8 \log(1/\delta) \rceil$ disjoint groups and compute their means Z_i.

(ii) Output the median $\widehat{\mu}$ of $\{Z_1, Z_2, \cdots, Z_k\}$.

In high dimensions, however, the question is a lot more subtle, with the correct notion of median being elusive. A long line of work culminated in the celebrated work of Lugosi and Mendelson [126]. The estimator follows the median-of-means paradigm by first bucketing the data into k groups and taking the means $\{Z_i\}_{i=1}^k$. The key structural lemma of their work is that the true mean is a $(0.01, O\left(r_\delta^2\right))$-combinatorial center of the bucket means, where r_δ is the sub-gaussian error rate (3.6.1). Recall that it means that if we consider projecting the bucket means to a one-dimensional direction, a majority of them are close to the true mean.

Lemma 3.6.1 (Lugosi-Mendelson structural lemma [126])**.** *Consider the setting of heavy-tailed mean estimation (Definition 3.6.1). Let $\{Z_i\}_{i=1}^k$ be the k bucket means with $k = \lceil 800 \log(1/\delta) \rceil$. Then with probability at least $1-\delta$, for all unit $v \in \mathbb{R}^d$,*

$$|\langle Z_i - \mu, v\rangle| \leq 3000 \left(\sqrt{\frac{\operatorname{Tr}(\Sigma)}{n}} + \sqrt{\frac{\|\Sigma\| \log(1/\delta)}{n}} \right), \tag{E_v}$$

for $0.99k$ of the bucket means $\{Z_i\}_{i=1}^k$.

This is exactly the combinatorial centrality condition (Definition 3.5.2) we introduced in Section 3.5. To build more intuition, we should visualize it as a clustering property. That is, under any one-dimensional projection, the bucket means are clustered around the true mean, and the width of the cluster is precisely the optimal sub-gaussian error $O(r_\delta)$.

This enables a natural estimator/algorithm—we can search for a point $\widehat{\mu}$ that is a $(0.01, r_\delta^2)$-combinatorial center for $\{Z_i\}_{i=1}^k$. Of course, such $\widehat{\mu}$ exists, since Lugosi-Mendelson (Lemma 3.6.1) showed that μ itself satisfies the condition (with probability at least $1-\delta$). Furthermore, one can check any valid $(\varepsilon, O(r_\delta^2))$-combinatorial center (Definition 3.5.2) $\widehat{\mu}$ with $\varepsilon < 1/2$ is indeed an estimator with sub-gaussian error rate $O(r_\delta)$, by a simple "pigeonhole + triangle inequality" argument.

Lemma 3.6.2 (combinatorial center has sub-gaussian rate)**.** *Let $\{Z_i\}_{i=1}^k$ be defined as above and $\varepsilon < 1/2$. Suppose that the condition in the Lugosi-Mendelson structural lemma (Lemma 3.6.1) holds. Then any $(\varepsilon, O\left(r_\delta^2\right))$-combinatorial center $\widehat{\mu}$ of $\{Z_i\}_{i=1}^k$ attains the sub-gaussian error (3.6.1) (up to constant).*

Proof. Let v be the unit vector in the direction of $\mu - \widehat{\mu}$. Then since $\widehat{\mu}$ is an $(\varepsilon, O(r_\delta^2))$-combinatorial center with $\varepsilon < 1/2$, we have $|\langle Z_i - \widehat{\mu}, v\rangle| \leq r_\delta$ for most Z_i. Also, $|\langle Z_i - \mu, v\rangle| \leq O(r_\delta)$ for most $\{Z_i\}_{i=1}^k$ by our assumption from Lugosi-Mendelson lemma. By the pigeonhole principle, there must be a Z_j such that $|\langle Z_j - \widehat{\mu}, v\rangle| \le O(r_\delta)$ and $|\langle Z_j - \mu, v\rangle| \le O(r_\delta)$. By triangle inequality,

$$\|\widehat{\mu} - \mu\| = \langle \mu - \widehat{\mu}, v\rangle \le |\langle Z_i - \mu, v\rangle| + |\langle Z_i - \widehat{\mu}, v\rangle| \le O(r_\delta).$$

as desired, and this completes the proof. □

However, the problem of efficiently finding a combinatorial center appears difficult. If one sticks to its definition, it is required to ensure that for *all* unit vector v, the clustering property (E_v) holds. It seems that even just *certifying* this condition would naïvely take exponential time (say, by enumerating a 1/2-net of unit sphere). Yet, we can actually resort to duality, to avoid the pain of designing a new algorithm from scratch. As we showed, a combinatorial center is just a spectral center, which our meta-algorithm can find for us.

Theorem 3.6.1 (heavy-tailed mean estimation via spectral sample reweighing)**.** *Given $\{X_i\}_{i=1}^n$ and δ, any constant-factor approximation algorithm for the spectral sample reweighing problem (Definition 3.3.1) can be used to compute an estimate $\widehat{\mu}$ that obtains the subgaussian error rate for heavy-tailed mean estimation (Definition 3.6.1), with probability at least $1-\delta$.*

Proof. Let $\{Z_i\}_{i=1}^k$ be the bucket means with $k = \lceil 800\log(1/\delta)\rceil$ and let $\lambda = 3000r_\delta$. We assume that the true mean μ is a $(0.01, \lambda^2)$-combinatorial center of $\{Z_i\}_{i=1}^k$. Suppose that we can obtain an α-factor approximation the spectral sample reweighing, with the input being $\{Z_i\}_{i=1}^k$.

- *Promise*: First let's check the spectral centrality condition holds. Since, by assumption, μ is a $(0.01, \lambda^2)$-combinatorial center of $\{Z_i\}_{i=1}^k$, we have that for all unit v

$$\sum_{i=1}^k \mathbb{1}\{|\langle x_i - \mu, v\rangle| \ge \lambda\} \le 0.01k.$$

  Thus, Proposition 3.5.2 (with $\nu = \mu$) implies that

$$\min_{w\in\mathcal{W}_{k,\varepsilon}} \max_{M\succeq 0, \mathrm{Tr}(M)=1} \sum_{i=1}^k w_i \left\langle (x_i-\mu)(x_i-\mu)^T, M\right\rangle \le 100\lambda^2,$$

  where $\varepsilon = 0.1$. This means that there exists $w \in \mathcal{W}_{k,\varepsilon}$ such that

$$\left\|\sum_{i=1}^n w_i \left(x_i - \mu\right)\left(x_i - \mu\right)^T\right\| \le 100\lambda^2.$$

- *Output*: Now the guarantee of an α-factor approximation for spectral sample reweighing (Definition 3.3.1) is that we have $\widehat{\mu} \in \mathbb{R}^d$ and $w' \in \mathcal{W}_{k,3\varepsilon}$ such that

$$\left\|\sum_{i=1}^n w_i' \left(x_i - \widehat{\mu}\right)\left(x_i - \widehat{\mu}\right)^T\right\| \le 100\alpha\lambda^2.$$

It immediately follows that

$$\min_{w \in \mathcal{W}_{k,3\varepsilon}} \max_{M \succeq 0, \mathrm{Tr}(M)=1} \sum_{i=1}^{k} w_i \left\langle (x_i - \widehat{\mu})(x_i - \widehat{\mu})^T, M \right\rangle \leq 100\alpha\lambda^2.$$

Now we can apply Proposition 3.5.1. Since α is a constant by assumption, we obtain that for all unit v,

$$\sum_{i=1}^{k} \mathbb{1}\left\{ |\langle x_i - \widehat{\mu}, v\rangle| \geq C(\alpha) \cdot \lambda \right\} \leq 0.4k, \tag{3.6.2}$$

for some constant $C(\alpha) = O(1)$ that depends on α. Therefore, we get that a majority of the points cluster around $\widehat{\mu}$, along any direction v, so it is a $(0.4, O(\lambda))$-combinatorial center. It follows from Lemma 3.6.2 that $\|\widehat{\mu} - \mu\| \leq O(r_\delta)$, as $\lambda = O(r_\delta^2)$.

Finally, note that the only condition of the argument is that the true mean is a combinatorial center, which occurs with probability at least $1 - \delta$, by Lemma 3.6.1. □

We remark that the exact constants we choose in the proof are immaterial, and no efforts have been given in optimizing them.

The theorem implies that the filter algorithm (Algorithm 1) combined with a simple pruning step from [112]) can be used for heavy-tailed mean estimation as well.

Corollary 3.6.1 (filter for heavy-tailed mean estimation)**.** *Given $\{X_i\}_{i=1}^n$ drawn i.i.d. from a distribution with mean μ and covariance Σ and a failure probability $2^{-O(n)} \leq \delta < 1$, there is an efficient algorithm that outputs $\widehat{\mu}$ such that with probability at least $1-\delta$, $\|\widehat{\mu}-\mu\| \leq O(r_\delta)$.*

Further, the algorithm is a black-box application of Algorithm 1 and runs in time $O(k^2d^2 + nd)$.

Proof. Given the input, we first compute the bucket means $\{Z_i\}_{i=1}^{2k}$, which takes $O(nd)$ time. Assume that the condition of the Lugosi-Mendelson structural lemma (Lemma 3.6.1) holds; that is, μ is a $(0.01, \lambda^2)$-combinatorial center of $\{Z_i\}_{i=1}^k$, where $\lambda = 3000r_\delta$. We use the filter algorithm (Algorithm 1) with the input being a pruned subset of $\{Z_i\}_{i=1}^k$ and apply its guarantees.

Here, we will not use the pruning step (Lemma 3.3.3), since it requires the knowledge of λ. Instead, we first compute the coordinate-wise median-of-means $\widehat{\mu}_0$ of $\{Z_i\}_{i=k+1}^{2k}$ and the distances $d_i = \|Z_i - \widehat{\mu}_0\|$ for each $i \in [k]$. We then sort the points by d_i (in descending order) and remove the top $0.01k$ points in $\{Z_i\}_{i=1}^k$ with large d_i. It can be shown that the remaining points has diameter at most $O(\sqrt{d}r_\delta)$; see Lemma E.1 of [112]. Let S the remaining points in $\{Z_i\}_{i=1}^k$.

For the run-time, we can apply the guarantee of the filter algorithm (Lemma 3.3.1), given the input S and a failure probability $\delta/3$. Since the squared diameter is $\rho = O(dr_\delta^2)$ and $\lambda = O(r_\delta^2)$, this gives a run-time of $\widetilde{O}(k^2d^2)$, since $k = O(\log(1/\delta))$.

We now have a constant-factor approximation for the spectral sample reweighing problem. By Theorem 3.6.1, this gives an estimate with the sub-gaussian error (3.6.1). Finally, the procedure's success depends on the condition of Lugosi-Mendelson (Theorem 3.6.1), success of the pruning procedure, and the guarantees of constant-approximation of spectral sample reweighing (Theorem 3.3.1). The failure probability of each event can be bounded by $\delta/3$. Applying union bound completes the proof. □

Other algorithms for heavy-tailed mean estimation This argument also enables us to solve the heavy-tailed mean estimation problem using other approximation algorithms for the spectral sample reweighing problem. Let $\lambda = 3000 r_\delta$. Recall that the argument for Theorem 3.6.1 shows that there is a $(0.1, O(\lambda^2))$-spectral center (which is the true mean μ). Moreover, the pruning step in the proof of Corollary 3.6.1 allows us to bound the squared diameter of a large subset of $\{Z_i\}_{i=1}^k$ by $\rho = O(d\lambda^2)$.

This implies that the gradient descent-based algorithm that we analyze in Appendix B.4 solves the heavy-tailed setting in $O\left(kd^2\right)$ iterations.

Corollary 3.6.2 (heavy-tailed mean estimation via gradient descent)**.** *Assume the setting of Corollary 3.6.1. A black-box application of the gradient descent-based algorithm (Algorithm 9, Appendix B.4) solves the heavy-tailed mean estimation problem with optimal error rate within $O(nd^2)$ iterations and $\widetilde{O}(n^2d^3)$ time.*

The quantum entropy scoring scheme (Appendix B.3), however, runs in $\widetilde{O}(\log(\rho/\lambda))$ number of iterations. Setting its failure probability to be $\delta/3$, we obtain the following, which matches the fastest-known algorithm for the problem [110, 112].

Corollary 3.6.3 (heavy-tailed mean estimation via quantum entropy scoring)**.** *Assume the setting of Corollary 3.6.1. A black-box application of the matrix multiplicative update algorithm (Algorithm 8, Appendix B.3) solves the heavy-tailed mean estimation problem with optimal error rate, in $\widetilde{O}(1)$ iterations and $\widetilde{O}(k^2d)$ total run-time.*

3.7 Discussion

Estimating the mean of a distribution is arguably the most fundamental problem in statistics. We showed that in robust and heavy-tailed settings, the problem can be approached by techniques from regret minimization and online learning. We believe the ideas we present here may be more broadly applicable to other problems in high-dimensional robust statistics, such regression and covariance estimation.

Chapter 4

Robust and Private Estimation of Gaussian

In this chapter, we give computationally-efficient algorithms for privately estimating a Gaussian distribution with optimal dependence on the dimension in the sample complexity.

In the pure DP setting, we give a polynomial-time algorithm that estimates an unknown d-dimensional Gaussian distribution up to an arbitrary tiny total variation error using $\widetilde{O}(d^2 \log \kappa)$ samples while tolerating a constant fraction of adversarial outliers. Here, κ is the condition number of the target covariance matrix. The sample bound matches best non-private estimators in the dependence on the dimension (up to a polylogarithmic factor). Prior to this work, only identifiability results (yielding inefficient super-polynomial time algorithms) were known for the problem. We prove a new lower bound on differentially private covariance estimation to show that the dependence on the condition number κ in the above sample bound is also tight.

4.1 Introduction

Learning a high-dimensional Gaussian distribution is arguably the most basic task in statistical estimation. A long line of work has focused on finding algorithms for this fundamental problem that satisfy additional constraints such as robustness to adversarial outliers [108, 61, 60, 53, 106] and differential privacy [103, 99, 97, 27, 121, 30, 2, 153, 105, 120, 14, 98]. An overarching goal in this line of work is to investigate the *cost of privacy*—the overhead in sample complexity and running time that one must incur over and above the setting without the privacy constraints. Minimizing the cost of differential privacy—and ideally, achieving the same asymptotic dependence on the underlying dimension in sample complexity—is a challenging goal.

To appreciate this goal, note that differentially private algorithms must necessarily have low sensitivity. That is, switching a single sample point in the input should lead to a small change to the estimate output by the algorithm. Importantly, this is a *worst-case* guarantee:

it must hold regardless of whether the input data satisfies the modeling assumption of being independent Gaussian samples. Natural and simple estimators such as the empirical mean and covariance of the input data have an unbounded sensitivity and result in no privacy guarantees.

Without privacy constraints, standard concentration inequalities imply that the mean of an unknown Gaussian distribution can be estimated up to an ℓ_2 error (more stringently, the "Mahalanobis" error) of $\sim d/n$ with $n \gg d$ samples. For covariance estimation, a similar analysis yields an error rate of $\sim d^2/n$ for $n \gg d^2$.[1] The last few years have seen remarkable progress in finding private estimators that come close to the above benchmarks. A recent work [87] provides an optimal estimator in ℓ_2 norm for the mean of a distribution with covariance bounded in spectral norm while satisfying *pure* differential privacy—the strongest guarantee investigated in this setting. On the other hand, in the less stringent model of approximate differential privacy, an essentially optimal algorithm for estimating a Gaussian distribution was recently found in [14].

This chapter The main contribution of this chapter is the first polynomial-time sample-optimal algorithms for estimating a Gaussian distribution in pure approximate differential privacy model. For pure differential privacy, the main challenge is the task of covariance estimation for which no computationally efficient algorithm was known. We resolve this challenge with dimension-dependence that asymptotically matches the optimal non-private bounds above. Our algorithms incur a logarithmic dependence on κ, the condition number of the unknown covariance. We improve prior lower bounds for the problem to show that such a dependence is in fact necessary for any pure differentially private algorithm thus concluding that our guarantees are asymptotically optimal.

For the algorithms in the pure differential privacy model, the main technique is a pre-conditioning scheme for covariance estimation that iteratively constructs a rough estimate of the large eigenvalues of the unknown covariance. Previously, such a subroutine was proposed in [99] with the weaker guarantees of approximate differential privacy. Our subroutine relies on the new sum-of-squares exponential mechanism introduced in the recent work of [87].

Our Results

We now describe our results in more detail. We start by formally defining differential privacy.

Definition 4.1.1 (Differential Privacy [67, 68])**.** *For $\varepsilon \geq 0$ and $\delta \in [0,1]$, a (randomized) algorithm $\mathcal{M}$ is (ε,δ)-differentially private if for all pairs of neighboring databases x, x' that differ in exactly one row, and for all output subsets S of the range of $\mathcal{M}$, the following holds:*

$$\mathbb{P}[\mathcal{M}(x) \in S] \leq e^{\varepsilon} \cdot \mathbb{P}[\mathcal{M}(x') \in S] + \delta,$$

[1]Note that for obtaining vanishing total variation error guarantees, we need to estimate the covariance of the unknown Gaussian in the relative Frobenius distance.

Reference	Sample Complexity	Privacy	Robustness
Naïve estimator	$\frac{d^2}{\alpha^2} + \frac{\kappa d^2}{\alpha\varepsilon}$	(ε, δ)	Not robust
[99]	$\frac{d^2}{\alpha^2} + \frac{d^2}{\alpha\varepsilon} + \frac{d^{3/2}\log^{1/2}\kappa}{\varepsilon}$	(ε, δ)	Not robust
[98]	$\frac{d^2}{\alpha^2} + \frac{d^2}{\alpha\varepsilon} + \frac{d^{5/2}}{\varepsilon}$	(ε, δ)	Not robust
[14] and [153]	$\frac{d^2}{\alpha^2} + \frac{d^2}{\alpha\varepsilon}$	(ε, δ)	Not robust
[14]	$\frac{d^{3.5}}{\alpha^3\varepsilon}$	(ε, δ)	η
[105]	$\frac{d^8}{\alpha^4\varepsilon^8}$	(ε, δ)	η
Our result (Theorem 4.1.1)	$\frac{d^2\log\kappa}{\varepsilon} + \frac{d^2}{\alpha^2\varepsilon}$	$(\varepsilon, 0)$	$\sqrt{\eta}$

Table 4.1: Computationally-efficient sample complexity upper bounds for DP covariance estimators of $\mathcal{N}(0, \Sigma)$, with the utility guarantee $\left\|\widehat{\Sigma} - \Sigma\right\|_\Sigma \leq \alpha$. Robustness refers to additional error when the sample is η-corrupted. Here, d is the dimension, κ is an a priori bound such that $\mathbb{I} \preceq \Sigma \preceq \kappa\mathbb{I}$. The bounds suppress polylogarithmic factors in $d, \frac{1}{\eta}, \frac{1}{\alpha}$, and $\frac{1}{\delta}$.

where the probability is over the coin flips of the algorithm $\mathcal{M}$. We say that an algorithm satisfies ε-pure DP if it satisfies (ε, δ)-DP with $\delta = 0$.

Our main contribution is finding sample-optimal private and outlier-robust algorithms for estimating a high-dimensional Gaussian distribution. In Table 4.1, we summarize the sample complexity, privacy, and robustness guarantees of our algorithms in relation to prior work.

Specifically, in the setting of pure DP, we prove:

Theorem 4.1.1 (Pure DP Gaussian estimation; see Theorem 4.4.3 and Theorem 4.4.5)**.** *Fix $\alpha, \varepsilon, \eta > 0$. Let $\mathbb{I} \preceq \Sigma_* \preceq \kappa\mathbb{I}$ and $\|\mu_*\|_2 \leq R$. There is a polynomial-time ε-pure DP algorithm that takes input an η-corrupted sample of size n from $\mathcal{N}(\mu_*, \Sigma_*)$ and with probability at least* 0.99 *over the draw of the sample and the randomness of the algorithm, outputs estimates $\widehat{\mu}$, $\widehat{\Sigma}$ such that $TV(\mathcal{N}(\mu_*, \Sigma_*), \mathcal{N}(\widehat{\mu}, \hat{\Sigma})) \leq \alpha + O(\sqrt{\eta})$ so long as $n \geq \widetilde{O}\left(d^2\log(\kappa)/\varepsilon + d^2/\alpha^2\varepsilon + d\log R/\varepsilon\right)$.*

As discussed earlier, the case when $\Sigma = I$ (i.e., mean estimation) was resolved in the recent work of [87] with essentially optimal $\widetilde{O}(d)$ sample complexity. For high dimensional covariance estimation all prior computationally efficient algorithms only satisfied an approximate DP guarantee. We note that an identifiability argument was presented in [2, 30]. The conceptual barrier here is that the exponential mechanism, a canonical mechanism for ensuring pure DP, does not admit a straightforward computationally efficient implementation for private estimation.

Reference	Sample Complexity	Privacy
[99]	$\frac{d^2}{\alpha\varepsilon}$	$(\varepsilon, 0)$
[96]	$\frac{d^2}{\alpha\varepsilon}$	(ε, δ)
Our result (Theorem 4.1.2)	$\frac{d^2 \log(\kappa)}{\varepsilon} + \frac{d^2}{\alpha\varepsilon}$	$(\varepsilon, 0)$

Table 4.2: Sample complexity lower bounds for DP covariance estimators of $\mathcal{N}(0, \Sigma)$, with the utility guarantee $\left\|\widehat{\Sigma} - \Sigma\right\|_\Sigma \le \alpha$. Here, d is the dimension, κ is an a priori bound such that $\mathbb{I} \preceq \Sigma \preceq \kappa\mathbb{I}$. The bounds suppress polylogarithmic factors in d, $\frac{1}{\alpha}$, and $\frac{1}{\delta}$.

Our work provides the first computationally-efficient algorithm for covariance estimation and achieves near optimal sample complexity as a function of the dimension. We note that the dependence on η (the fraction of outliers) in our result above is likely suboptimal and may be improvable to $\widetilde{O}(\eta)$. Our key idea is exploiting the pure DP mean estimation algorithm of [87] to build a preconditioning scheme for covariances. Such a scheme was previously used in [99] for covariance estimation with approximate DP.

Pure DP Lower Bound We complement our algorithmic result by proving a lower bound on the sample complexity of pure DP covariance estimation that scales with $\log(\kappa)$, where κ is the condition number of the unknown covariance. This matches our algorithmic guarantees and improves on the $\Omega\left(d^2/(\alpha\varepsilon)\right)$ sample lower bound for ε-pure DP covariance estimation from prior work [99]. Our lower bound is based on a packing-type analysis building on [99, 30].

Theorem 4.1.2 (Pure DP lower bound for covariance estimation; see Theorem 4.5.1)**.** *Let $\varepsilon, \alpha \in (0,1)$, $d \ge 2$. Any ε-DP algorithm that, given n i.i.d. samples $\mathcal{X} = \{X_1, \ldots, X_n\}$ from $\mathcal{N}(0, \Sigma)$ for an unknown $\Sigma \in \mathbb{R}^{d \times d}$ satisfying $I \preceq \Sigma \preceq \kappa I$, outputs $\widehat{\Sigma} = \widehat{\Sigma}(\mathcal{X})$ such that, with probability 0.9,*

$$TV\left(\mathcal{N}(0, \Sigma), \mathcal{N}\left(0, \widehat{\Sigma}\right)\right) < O(\alpha),$$

requires

$$n = \widetilde{\Omega}\left(\frac{d^2 \log(\kappa)}{\varepsilon} + \frac{d^2}{\alpha\varepsilon}\right).$$

Comparing this with our upper bound (Theorem 4.1.1) shows that our pure DP algorithmic result achieves nearly optimal sample complexity, only up to a $1/\alpha$ factor in its second term.

Concurrent Work In a concurrent work, Hopkins, Kamath, Majid and Narayanan [90] obtained near-optimal polynomial-time algorithms for learning a Gaussian subject to pure and approximate DP and robustness constraints.

Additional Related Work

Private Estimation Assuming the data is drawn from a one-dimensional normal distribution, Karwa and Vadhan [103] obtain the first (finite-sample) guarantees for confidence interval estimation of the mean—whether or not the variance of the population is known. Their bounds are tight up to logarithmic factors. In the high-dimensional cases, Kamath, Li, Singhal, and Ullman [99] give the first (ε, δ)-DP algorithm for learning a Gaussian. Their sample complexity has a logarithmic dependence on the condition number of the unknown covariance. This result has since been improved and refined by multiple follow-up papers, including [21, 27, 14, 98, 105], with a focus on removing the dependence on conditioning. We summarize these upper bounds in Table 4.1. We stress that all of them study (ε, δ)-DP. In fact, our work is the first sample- and computationally-efficient algorithm for learning a multivariate Gaussian under the stronger ε-DP.

For pure DP, the most relevant work to us is Hopkins, Kamath and Majid [87], which gives a computationally efficient pure DP procedure for mean estimation of bounded second moment distributions. Our pure DP algorithm builds upon their main result. In addition, Bun et al. [30] gives a general (computationally inefficient) cover-based approach to pure DP estimation.

Robust Statistics and Connections with Privacy We refer the reader to [53] for a survey. For robustly learning the mean of a Gaussian distribution subject to privacy, Liu, Kong, Kakade, and Oh provide polynomial-time algorithms with sub-optimal sample complexity [121]. However, with optimal sample complexity, they only provide an exponential-time algorithm. For robustly learning a general Gaussian distribution subject to DP, Kothari, Manurangsi, and Velingker provide a polynomial-time approximate DP algorithm with sample complexity $\widetilde{O}(d^8)$ [105]. On the other hand, Ashtiani and Liaw obtain polynomial-time private and robust algorithms with sample complexity $\tilde{O}(d^{3.5})$ [14]. Liu, Kong, and Oh [120] rely on the observation that one-dimensional robust statistical estimators have low sensitivity on resilient datasets. Resilience, defined by Steinhardt, Charikar, and Valiant [149], measures how stable the empirical mean is to deletion of a constant fraction of a dataset. This definition is, intuitively, similar in spirit, but not equivalent to DP guarantees. Using this framework and the exponential mechanism, Liu, Kong, and Oh [120] design computationally inefficient algorithms for DP statistical estimation.

More relevant to us, Kothari, Manurangsi and Velingker [105] showed how to transform robust algorithms into private ones. Finally, a very recent work by Georgiev and Hopkins [76] provides a generic meta-theorem for obtaining robustness from DP.

Lower Bounds for DP In the pure DP case, geometric packing-style arguments are often used to prove lower bounds [131, 19, 80]. Our pure DP lower bound arguments build upon previous works [99, 30]. For the approximate DP case, the lower bounds are based on fingerprinting codes, introduced by Boneh and Shaw [25]. The existence of such codes imply lower bounds for approximate DP [29]. The technique can be applied to prove sample complexity lower bounds for Gaussian estimation under approximate DP [96].

4.2 Techniques

We will focus on private covariance estimation where we are given samples from $\mathcal{N}(0, \Sigma)$ with $\mathbb{I} \preceq \Sigma \preceq \kappa\mathbb{I}$. The sensitivity of the naive empirical covariance is clearly unbounded but can be fixed (see [99]) by first removing input points with $\|y_i\|_2^2 > \kappa d$ ("clamping"). This yields a private covariance estimator with sample complexity growing linearly in κ.

To improve the dependence on κ, [99] proposes a natural iterative private conditioning scheme:

(i) By applying Gaussian mechanism to clamped samples, obtain a rough estimate of the covariance with quadratic forms along large eigenvectors preserved multiplicatively and a subset of these can also be identified from the estimate.

(ii) Via a linear transformation, shrink the covariance in the above large directions and repeat.

This scheme uses the Gaussian mechanism so only satisfies approximate (and concentrated) DP guarantees. We obtain a pure DP version of this scheme by replacing the first step with the pure DP *mean estimation* algorithm of [87] (on bounded second moment distributions) — let $\{x_i\}_{i=1}^n$ be the i.i.d. samples from $\mathcal{N}(0, \Sigma)$. If we form the samples $\mathcal{Y} = \left\{x_i^{\otimes 2}/\kappa : i \in [n]\right\}$ for $n \gg d^2/\varepsilon$, then [87] guarantees that we can get $\widehat{\Sigma}$ such that $\left\|\widehat{\Sigma} - \Sigma\right\|_F \leq 0.01\kappa$. This estimate already allows an approximation of quadratic forms up to an additive 0.01κ. This means that the large quadratic forms (i.e., $\geq \kappa/2$) are well preserved multiplicatively. This allows building an iterative preconditioning scheme as above that allows reducing to the case when the unknown covariance as $O(1)$ condition number.

At that point, we can reduce to mean estimation rather naturally: when Σ has largest eigenvalue κ, for a Gaussian random variable $x \sim \mathcal{N}(0, \Sigma)$ $x^{\otimes 2}$ has covariance with spectral norm at most $2\kappa^2$. Thus, we can apply the pure DP algorithm of [87] we can obtain an estimate with a small relative Frobenius error. This algorithm also inherits handling outliers from [87].

4.3 Preliminaries

Notation Throughout this chapter, we will use X to denote an i.i.d. (uncorrupted) sample of n points in $\mathbb{R}^d$ and Y to denote its $(1-\alpha)$-corruption. For any finite set S of points, we will use $\mathbb{E}_{s\sim S} f(s)$ to denote the empirical average of $f(s)$ as s varies uniformly over S. We denote the d-by-d identity matrix by I_d. Let $\mathbb{S}_+^d$ denote the set of d-by-d positive semidefinite (PSD) matrices. For a matrix A, we use $\|A\|, \|A\|_2$ to denote the spectral norm of A and $\|A\|_F$ denotes its Frobenius norm. For PSD matrices A, B, we write $A \preceq B$ if $B - A$ is PSD. For a PSD matrix Σ and vector x, define $\|x\|_\Sigma = \left\|\Sigma^{-1/2}x\right\|_2$. For a matrix X, define $\|X\|_\Sigma = \left\|\Sigma^{-1/2}X\Sigma^{-1/2}\right\|_F$. For $M \in \mathbb{R}^{m\times n}$ and $N \in \mathbb{R}^{m'\times n'}$, we define $M \otimes N$ to be the standard $mm' \times nn'$ matrix given by the Kronecker product of M and N. Let $\Delta_d = \left\{p \in \mathbb{R}_+^d, \sum_{i=1}^d p_i = 1\right\}$ denote the probability simplex in $\mathbb{R}^d$. Let $\mathbb{1}_n$ denote the all-one vector of n dimensions.

Differential Privacy

Theorem 4.3.1 (Basic Composition [67]). *For every $\varepsilon, \delta \geq 0$ and $k \in \mathbb{N}$, the class of (ε, δ)-DP mechanisms is $(k\varepsilon, k\delta)$-DP under k-fold adaptive compositions.*

Theorem 4.3.2 (A variant of Parallel Composition [130]). *Let $\mathcal{M} : \mathcal{O} \times \mathcal{Y} \to \mathcal{O}$ be an ε-DP mechanism that takes as input some parameters $w \in \mathcal{O}$ and a dataset $Y \in \mathcal{Y}$ and outputs $\mathcal{M}(w, Y) \in \mathcal{O}$. For any $k \in \mathbb{N}$, let $\mathcal{M}_k : \mathcal{O} \times \mathcal{Y}^k \to \mathcal{O}^{k+1}$ be a mechanism that takes as input parameters $w_0 \in \mathcal{O}$ and a dataset $Y = (Y_1, \dots, Y_k) \in \mathcal{Y}^k$ partitioned into k disjoint components where k does not depend on Y.*

Given $w_0 \in \mathcal{O}$, the mechanism $\mathcal{M}_k$ (adaptively) computes $w_i = \mathcal{M}(w_{i-1}, Y_i)$ for each $i \in [k]$, and outputs $(w_0, w_1, \dots, w_k)$. Then $\mathcal{M}_k$ satisfies ε-DP.

Proof. Fix any sequence $w = (w_0, w_1, \dots, w_k) \in \mathcal{O}^{k+1}$ and let $Y = (Y_1, \dots, Y_k)$ and $Y' = (Y_1', \dots, Y_k')$ be neighboring datasets in $\mathcal{Y}^k$. We need to show that the ratio $\frac{\Pr[\mathcal{M}_k(w_0,Y)=w]}{\Pr[\mathcal{M}_k(w_0,Y')=w]}$ lies between $e^{-\varepsilon}$ and e^{ε}.

Since k does not depend on any dataset and Y is partitioned into disjoint subsets, there must exist at most one partition that is neighboring. To this end and without loss of generality, let $j \in [k]$ be an index such that $Y_j, Y_j' \in \mathcal{Y}$ are neighboring datasets and $Y_i = Y_i'$ for all $i \neq j$. Therefore, by independence,

$$\frac{\Pr[\mathcal{M}_k(w_0, Y) = w]}{\Pr[\mathcal{M}_k(w_0, Y') = w]} = \prod_{i=1}^{k} \frac{\Pr[\mathcal{M}(w_{i-1}, Y_i) = w_i]}{\Pr[\mathcal{M}(w_{i-1}, Y_i') = w_i]}.$$

Thus, in the rightmost product above, only the j^{th} of the k factors may differ from 1. Since $\mathcal{M}$ is ε-DP, this j^{th} factor lies between $e^{-\varepsilon}$ and e^{ε}. Thus, $\mathcal{M}_k$ is also ε-DP. $\square$

One of the most generic mechanisms used to satisfy pure differential privacy is the *Exponential Mechanism*:

Theorem 4.3.3 (Exponential Mechanism [131])**.** *Let $\mathcal{X} \sim \mathcal{X}' \in \mathcal{O}^k$ denote two neighboring datasets. Consider any arbitrary utility function $u : \mathcal{O}^k \times \mathcal{R} \to \mathbb{R}$ with global sensitivity $\Delta_u = \max_{\mathcal{X}\sim\mathcal{X}',r} |u(\mathcal{X}, r) - u(\mathcal{X}', r)|$. For any dataset $\mathcal{X}$, the exponential mechanism outputs $r \in \mathcal{R}$ with probability $\propto \exp\left(\frac{\varepsilon \cdot u(\mathcal{X},r)}{2\Delta_u}\right)$.*

Furthermore, the exponential mechanism satisfies ε-DP.

In general, the exponential mechanism is not computationally efficient to implement. We cite a recent result on pure DP mean estimation (on bounded second moment distributions), due to [87], that presents computationally efficient implementations of the exponential mechanism via a Sum-of-Squares approach. Their main procedure is outlier-robust with a corruption level of η, at the cost of increasing the estimation error by an additive $O(\sqrt{\eta})$.

Theorem 4.3.4 (Pure DP mean estimation, Theorem 1.2 of [87])**.** *For every $n, d \in \mathbb{N}$ and $R, \alpha, \varepsilon, \beta > 0$ there is a polynomial-time ε-DP algorithm* PureDPMean *such that for every distribution D on $\mathbb{R}^d$ such that $\|\mathbb{E}_{X\sim D} X\|_2 \le R$ and $Cov_{X\sim D}(X) \preceq I_d$, given $X_1, \ldots, X_n \sim D$, with probability at least $1 - \beta$ the algorithm outputs $\widehat{\mu}$ such that $\|\widehat{\mu} - \mathbb{E}_{X\sim D} X\|_2 \le \alpha$ so long as*

$$n \ge \widetilde{O}\left(\frac{d + \log(1/\beta)}{\alpha^2\varepsilon} + \frac{d\log(R) + \min(d, \log(R)) \cdot \log(1/\beta)}{\varepsilon}\right).$$

Furthermore, if an η-fraction of the samples are adversarially corrupted, the algorithm maintains the same guarantee, at the cost now that $\|\widehat{\mu} - \mathbb{E}_{X\sim D} X\| \le \alpha + O(\sqrt{\eta})$.

Basic Tools from Probability Theory

Definition 4.3.1 (Total Variation Distance)**.** *For any two distributions P, Q over $\mathbb{R}^d$, the total variation distance TV is defined as*

$$TV(P, Q) = \sup_{S\subseteq\mathbb{R}^d} |P(S) - Q(S)|.$$

Moreover, it can be verified that

$$TV(P, Q) = \frac{1}{2}\int_{\mathbb{R}^d} |p(x) - q(x)|dx.$$

The following lemma shows how to convert parameter estimation to distribution estimation (in total variation distance), for the multidimensional Gaussian distribution.

Lemma 4.3.1 (Parameter closeness implies distribution closeness; see Lemma 2.9 of [99])**.** *Let $\alpha \ge 0$, $\mu, \widehat{\mu} \in \mathbb{R}^d$ and $\Sigma, \widehat{\Sigma} \in \mathbb{R}^{d\times d}$ be PSD. Suppose that*

$$\left\|\Sigma^{-1/2}(\mu - \widehat{\mu})\right\|_2 \le \alpha \text{ and } \left\|\Sigma^{-1/2}\widehat{\Sigma}\Sigma^{-1/2} - I\right\|_F \le \alpha.$$

Then $TV\left(\mathcal{N}(\mu, \Sigma), \mathcal{N}\left(\widehat{\mu}, \widehat{\Sigma}\right)\right) \le O(\alpha)$.

The fact below follows from Theorem 4.12 of [60].

Fact 4.3.1. *Let $X \sim \mathcal{N}(0,\Sigma)$ and $Y = XX^T$. Then $Cov(Y) \preceq 3\Sigma \otimes \Sigma$.*

4.4 Pure DP Covariance Estimation

In this section, we give an efficient algorithm for Gaussian covariance estimation under pure differential privacy.

High-level overview First, we iteratively use the mean estimation algorithm of [87] applied to the 2nd-tensor powers precondition the unknown covariance matrix to reduce the condition number of the unknown Σ by a constant factor and thus after $O(\log \kappa)$ steps assume WLOG that we have a well-conditioned covariance. We can now appeal to pure DP mean estimation algorithm applied to the 2nd-tensor powers of the input points to obtain the desired estimate of the covariance. Our estimator works even under η-fraction outliers (since the mean estimation procedure of [87] does) and obtains an additional TV error of $O(\sqrt{\eta})$.

Weak Private Preconditioning

Given the samples, our goal is to output a preconditioning matrix A such that $I \preceq A\Sigma A \preceq 0.99\kappa I$, where κ is the condition number of the known covariance Σ — improving the condition number of $A\Sigma A$ improves over that of Σ by a constant factor. This guarantee is similar to what appears in the previous literature on private covariance estimation and subspace recovery [99, 144, 98]. However, the algorithms from prior work crucially rely upon the Gaussian mechanism, which only ensure *approximate* (or concentrated) DP.

In this section, we describe and analyze a weak pure DP algorithm for preconditioning the covariance. The procedure reduces the condition number of the covariance (multiplicatively) by a constant factor. Towards this goal, a simple observation is that the algorithm for pure DP mean estimation from [87], applied naïvely, can be used for covariance estimation with an absolute Frobenius norm error guarantee.

Theorem 4.4.1 (Pure DP covariance estimation in absolute Frobenius norm)**.** *Let $\alpha > 0$ be an error parameter, $\varepsilon > 0$ be a privacy parameter and $d \in \mathbb{N}$. There is a polynomial-time ε-DP algorithm* PureDPMatrixMean *that, given ε, α and*

$$n \geq \widetilde{O}\left(\frac{d^2 + \log(1/\beta)}{\alpha^2 \varepsilon}\right)$$

i.i.d. samples $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$ from $\mathcal{N}(0,\Sigma)$ for an unknown $\Sigma \in \mathbb{R}^{d\times d}$ satisfying $\Sigma \preceq \kappa I$, outputs $\widehat{\Sigma} = \widehat{\Sigma}(\mathcal{X})$ satisfying

$$\left\|\widehat{\Sigma} - \Sigma\right\|_F \leq \alpha\kappa$$

1. **Input:** Samples $\mathcal{X} = \{X_1, \dots, X_n\} \subset \mathbb{R}^d$, condition number $\kappa \geq 1$, accuracy parameter $\alpha > 0$, failure probability $\beta > 0$, privacy parameter $\varepsilon > 0$.
2. Form the set of samples $\mathcal{Y} = \left\{ Y_i = \frac{X_i \otimes X_i}{\sqrt{3}\kappa} : i \in [n] \right\}$.
3. Run the algorithm PureDPMean in Theorem 4.3.4 (the main result of [87]) on input $\mathcal{Y}$ with $R = \sqrt{d/3}$ and $\alpha/\sqrt{3}, \beta, \varepsilon > 0$ to obtain an estimate $\tilde{\Sigma}$.
4. **Output:** Covariance matrix estimate $\widehat{\Sigma} = \sqrt{3}\kappa\tilde{\Sigma}$.

Figure 4.1: PureDPMatrixMean (based on PureDPMean in Theorem 4.3.4)

with probability at least $1 - \beta$.

Proof. Observe that

$$\|\mathbb{E}\, Y_i\|_F \leq \frac{1}{\sqrt{3}\kappa} \|\Sigma\|_F \leq \sqrt{\frac{d}{3}} = R$$

by the definition of Y_i, and

$$\mathrm{Cov}(Y_i) \preceq \frac{1}{\kappa^2} \Sigma \otimes \Sigma \preceq I$$

by Fact 4.3.1. By the choice of n and the error guarantee of PureDPMean, we have that the matrix $\widehat{\Sigma}$ output by Algorithm 4.1 (PureDPMatrixMean) satisfies $\left\| \widehat{\Sigma} - \Sigma \right\|_F \leq \alpha\kappa$ with probability at least $1-\beta$. Finally, the algorithm runs in polynomial time, since PureDPMean is in polynomial time and it takes linear time to form and rescale the samples. □

We leverage the above observation to design a weak preconditioning algorithm (Algorithm 4.2) that reduces the condition number of Σ by a constant factor. Algorithm 4.2 first runs Algorithm 4.1 with an error parameter $\alpha = 0.01$. Then from the error guarantee of Theorem 4.4.1, we can privately estimate all eigenvalues of Σ up to an additive factor of 0.01κ. Finally, we run a *partial projection* step, a technique from [99]. Informally speaking, the algorithm partially projects out the eigenvectors associated with large eigenvalues. Intuitively, this shrinks the directions of large variance more so than those of small variance, and thus reduces conditioning number.

The algorithm is formally described by Algorithm 4.2 and its guarantees given below.

Lemma 4.4.1 (Private preconditioning, one round)**.** *Let* $\varepsilon > 0$ *be a privacy parameter,* $d \in \mathbb{N}$*,* $\kappa \geq 20$*, and* $\beta > 0$ *be a failure probability. There is a polynomial-time* ε*-DP*

1. **Input:** Samples $\mathcal{X} = \{X_1, \ldots, X_n\} \subset \mathbb{R}^d$, condition number $\kappa \geq 1$, accuracy parameter $\alpha > 0$, failure probability $\beta > 0$, privacy parameter $\varepsilon > 0$.
2. Run Algorithm 4.1 PureDPMatrixMean on input $\mathcal{X}$ and with parameters $(\alpha = 0.01, \beta, \varepsilon, \kappa)$ to obtain an estimate $\widehat{\Sigma}$ of the covariance matrix.
3. Let V be the span of all eigenvectors of $\widehat{\Sigma}$ attaining eigenvalue at least $\kappa/2$, Π be the projector onto V and $\Pi_\perp$ be the projector onto the orthogonal complement of V.
4. **Output:** Weak preconditioner $A = 1.19 \cdot (0.9\Pi + \Pi_\perp)$.

Figure 4.2: One round, weak private preconditioning algorithm

algorithm that, given ε, β and

$$n \geq \widetilde{O}\left(\frac{d^2 + \log(1/\beta)}{\varepsilon}\right)$$

i.i.d. samples $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$ from $\mathcal{N}(0, \Sigma)$ for an unknown $\Sigma \in \mathbb{R}^{d\times d}$ satisfying $I \preceq \Sigma \preceq \kappa I$, outputs $A \in \mathbb{R}^{d\times d}$ such that

$$I \preceq A\Sigma A^T \preceq 0.99\kappa I$$

with probability at least $1 - \beta$.

Proof. We will show that Algorithm 4.2 satisfies the claims. First, note that it is ε-DP because the algorithm, based on PureDPMean, in Theorem 4.4.1 is ε-DP and Algorithm 4.2 post-preprocesses its output. By our distributional assumptions, with probability $1 - \beta$ it holds that $\widehat{\Sigma}$, computed in the second step of Algorithm 4.2, satisfies

$$\left\|\widehat{\Sigma} - \Sigma\right\|_F \leq 0.01\kappa. \tag{4.4.1}$$

This implies that, for any unit vectors u, v it holds that $\left|u^T\widehat{\Sigma}v - u^T\Sigma v\right| \leq 0.01\kappa$. Let V be the subspace spanned by all eigenvectors of $\widehat{\Sigma}$ with corresponding eigenvalue at least $\kappa/2$, Π be the projector onto V and $\Pi_\perp$ be the projector onto $V^\perp$, the orthogonal complement of V. We will now show that the matrix $A = \gamma\Pi + \Pi_\perp$ satisfies $0.85I \preceq A\Sigma A \preceq 0.83\kappa I$ for $\gamma = 0.9$. Then rescaling A by 1.19 ensures that the conclusion of the lemma holds.

For the upper bound, we have that

$$
\begin{aligned}
\|A\Sigma A\|_2 &\le \left\|A\widehat{\Sigma}A\right\|_2 + \left\|A\left(\Sigma - \widehat{\Sigma}\right)A\right\|_2 \le \left\|A\widehat{\Sigma}A\right\|_2 + \left\|\Sigma - \widehat{\Sigma}\right\|_2 \|A\|_2^2 \\
&\le \max\left(\kappa/2, \gamma^2 \cdot \left\|\widehat{\Sigma}\right\|_2\right) + \left\|\Sigma - \widehat{\Sigma}\right\|_2 \|A\|_2^2 \\
&\le \max\left(\kappa/2, \gamma^2 \cdot 1.01\kappa\right) + 0.01\kappa \\
&\le \left(1.01\gamma^2 + 0.01\right)\kappa \le 0.83\kappa,
\end{aligned}
$$

where the first line involves an application of the triangle inequality, the second follows since the spectral norm is sub-multiplicative, the third is by the choice of A, the fourth line by the error guarantee of $\widehat{\Sigma}$ (Equation 4.4.1) and $\|A\|_2 \le 1$, and the last inequality uses $\gamma = 0.9$.

For the lower bound, consider any unit vector $u \in \mathbb{R}^d$. We will lower bound $u^T A\Sigma A u$ in two different ways and maximize over the two. First, because $\Sigma \succeq I$, we have that

$$u^T A\Sigma A u \ge u^T A^2 u = \gamma^2 \|\Pi u\|_2^2 + \|\Pi_\perp u\|_2^2 \ge \|\Pi_\perp u\|_2^2 .$$

For the second lower bound, we have

$$u^T A\Sigma A u \ge \gamma^2 u^T \Pi\Sigma\Pi u + \gamma u^T \Pi_\perp \Sigma \Pi u + \gamma u^T \Pi\Sigma\Pi_\perp u.$$

Since $\Pi u \in V$, the first term is lower bounded by $.49\gamma^2\kappa \|\Pi u\|_2^2$. For the second (and similarly for the third) term, we have that

$$\left|\gamma u^T \Pi_\perp \Sigma \Pi u\right| \le 0.01\gamma\kappa \|\Pi u\|_2 \|\Pi_\perp u\|_2 \le 0.01\gamma\kappa \|\Pi u\|_2 .$$

Aggregating these bounds, we have that for any unit vector $u \in \mathbb{R}^d$, it holds that

$$u^T A\Sigma A u \ge \max\left(\|\Pi_\perp u\|_2^2, \left(0.49\gamma^2 \|\Pi u\|_2^2 - 0.02\gamma \|\Pi u\|_2\right)\kappa\right).$$

Using the facts that $\kappa \ge 20$ and $\|\Pi u\|_2^2 + \|\Pi_\perp u\|_2^2 = 1$, it is straightforward to verify that this lower bound is always at least 0.85. This completes the proof. □

Recursive Private Preconditioning

Given the weak conditioning algorithm, the natural next step is to recurse. Applying the weak preconditioner for $O(\log\kappa)$ times suffices to put the covariance nearly into identity.

Specifically, we show that Algorithm 4.3 satisfies the following guarantees.

Theorem 4.4.2 (Private preconditioning, recursive). *Let $\varepsilon > 0$ be a privacy parameter and $d \in \mathbb{N}$. There is a polynomial-time ε-DP algorithm that, given*

$$n \ge \widetilde{O}\left(\frac{d^2 \log(\kappa)}{\varepsilon}\right)$$

```
1. Input: Samples $\mathcal{X} = \{X_1, X_2, \dots, X_n\} \subset \mathbb{R}^d$, condition number $\kappa \geq 1$,
   privacy parameter $\varepsilon > 0$.
2. Set $L = O(\log \kappa)$ and partition $\mathcal{X} = \mathcal{X}_1 \sqcup \cdots \sqcup \mathcal{X}_L$ into $L$ subsets of size
   $n/L$ each.
3. Set $A_0 = I_d$ and $\kappa_1 = \kappa$.
4. For each $j \in [L]$:
   a) Set $A_{<j} = \prod_{k=0}^{j-1} A_k$.
   b) Run Algorithm 4.2 on samples $\{A_{<j}X : X \in \mathcal{X}_j\}$ with parameters
      $(\varepsilon, \beta = 1/1000L, \kappa_j)$ to obtain a matrix $A_j \in \mathbb{R}^{d \times d}$.
   c) Set $\kappa_{j+1} = 0.99\kappa_j$.
5. Let $A = \prod_{i=j}^{L} A_i$
6. Output: the preconditioning matrix $A$.
```

Figure 4.3: Recursive private preconditioning algorithm

i.i.d. samples $\mathcal{X} = \{X_1, \dots, X_n\}$ *from* $\mathcal{N}(0, \Sigma)$ *for an unknown* $\Sigma \in \mathbb{R}^{d \times d}$ *satisfying* $I \preceq \Sigma \preceq \kappa I$*, outputs a matrix* $A \in \mathbb{R}^{d \times d}$ *such that*

$$I \preceq A\Sigma A \preceq 20I$$

with probability at least 0.99.

Proof. We will show that Algorithm 4.3 satisfies the claims. It follows from parallel composition (Theorem 4.3.2) and the privacy of the weak preconditioner (Lemma 4.4.1) that the algorithm is ε-DP. Assume without loss of generality that $\kappa \geq 20$. By the choice of $L = O(\log(\kappa))$ and $\beta = 1/1000L$ and an application of union bound, with probability as least 0.999, all L invocations of Algorithm 4.2, in step 4(b), succeed. We now condition on this success event.

Let $\Sigma_1 = \Sigma$ and $\Sigma_j = A_{<j}\Sigma_j A_{<j}$ for every $j \leq L$. By the guarantee of Algorithm 4.2, in the j-th iteration we get a preconditioning matrix A_j such that

$$I \preceq A_j \Sigma_j A_j \preceq 0.99\kappa_j I. \tag{4.4.2}$$

By induction on j and the choice of L, we have that A satisfies $I \preceq A\Sigma A \preceq 20I$. □

Putting it Together

We can now put everything together and prove one of our primary results, the main statement of Theorem 4.1.1. For convenience, we restate it below as Theorem 4.4.3.

1. **Input:** Samples $\mathcal{X} = \{X_1, \dots, X_n\} \subset \mathbb{R}^d$, condition number $\kappa \geq 1$, privacy parameter $\varepsilon > 0$.
2. Compute the preconditioning matrix $A \in \mathbb{R}^{d \times d}$ using $\{X_1, \dots, X_{n/2}\}$ as input to Algorithm 4.3 and privacy parameter $\varepsilon/2$.
3. Run Algorithm 4.1 PureDPMatrixMean on samples $\{AX_{n/2+1}, \dots, AX_n\}$ with privacy parameter $\varepsilon/2$ and error parameter $\alpha/20$ to obtain a covariance matrix $\Sigma_1 \in \mathbb{R}^{d \times d}$.
4. **Output:** Covariance estimate $\widehat{\Sigma} = A^{-1}\Sigma_1 A^{-1}$.

Figure 4.4: Private covariance estimation algorithm

Theorem 4.4.3 (Pure DP covariance estimation)**.** *Let $\alpha > 0$ be an error parameter, $\varepsilon > 0$ be a privacy parameter, and $d \in \mathbb{N}$. There is a polynomial-time ε-DP algorithm that, given ε and*

$$n \geq \widetilde{O}\left(\frac{d^2 \log(\kappa)}{\varepsilon} + \frac{d^2}{\alpha^2 \varepsilon} + \frac{d \log(R)}{\varepsilon}\right)$$

i.i.d. samples $\mathcal{X} = \{X_1, \dots, X_n\}$ from $\mathcal{N}(0, \Sigma)$ for an unknown $\Sigma \in \mathbb{R}^{d \times d}$ satisfying $I \preceq \Sigma \preceq \kappa I$, outputs $\widehat{\Sigma} = \widehat{\Sigma}(\mathcal{X})$ satisfying

$$\left\|\Sigma^{-1/2}\widehat{\Sigma}\Sigma^{-1/2} - I\right\|_F \leq \alpha$$

with probability at least 0.99.

Proof. We will show that Algorithm 4.4 satisfies the claims. First, note that the algorithm is ε-DP by applying basic composition (Theorem 4.3.1) to the privacy guarantees of Theorem 4.4.2 and Theorem 4.4.1. Moreover, the algorithm is in polynomial time, since Algorithms 4.1 and 4.3 both run in polynomial time.

It now suffices to prove the utility guarantees. Let $\mathcal{X}$ consist of n i.i.d. samples from $\mathcal{N}(0, \Sigma)$, for an unknown Σ satisfying $I \preceq \Sigma \preceq \kappa I$. By Theorem 4.4.2, if

$$n \geq \widetilde{O}\left(\frac{d^2 \log(\kappa)}{\varepsilon}\right)$$

the preconditioner A computed in step 2 of Algorithm 4.4 satisfies $I \preceq A\Sigma A \preceq 20I$ with probability 0.99. Conditioned on this event, the samples $\{AX_{n/2+1}, \ldots, AX_n\}$ are i.i.d. according to $\mathcal{N}(0, A\Sigma A)$. Since $A\Sigma A \preceq 20I$ and $n \geq \widetilde{O}\left(\frac{d^2}{\alpha^2\varepsilon}\right)$, Algorithm 4.1 will return a covariance matrix Σ_1 satisfying

$$\|\Sigma_1 - A\Sigma A\|_F \leq \alpha,$$

with probability at least 0.99. We translate this into a relative Frobenius distance guarantee as follows. Notice that

$$\begin{aligned}\alpha \geq \|\Sigma_1 - A\Sigma A\|_F &= \left\|A\Sigma^{1/2}\left(\Sigma^{-1/2}A^{-1}\Sigma_1A^{-1}\Sigma^{-1/2} - I\right)\Sigma^{1/2}A\right\|_F \\ &\geq \lambda_{\min}(A\Sigma A)\left\|\Sigma^{-1/2}A^{-1}\Sigma_1A^{-1}\Sigma^{-1/2} - I\right\|_F,\end{aligned}$$

where the last step uses the sub-multiplicativity of the Frobenius norm. By definition of the algorithm, the final estimate of Σ is $\widehat{\Sigma} = A^{-1}\Sigma_1A^{-1}$. Plugging this into the inequality above, we have

$$\alpha \geq \|\Sigma_1 - A\Sigma A\|_F \geq \lambda_{\min}(A\Sigma A)\left\|\Sigma^{-1/2}\widehat{\Sigma}\Sigma^{-1/2} - I\right\|_F. \tag{4.4.3}$$

Observe that $\lambda_{\min}(A\Sigma A) \geq 1$, since $I \preceq A\Sigma A$. Applying this fact and rearranging the inequality 4.4.3, we get that

$$\left\|\Sigma^{-1/2}\widehat{\Sigma}\Sigma^{-1/2} - I\right\|_F \leq \alpha, \tag{4.4.4}$$

completing the proof. □

Application: General Pure DP Gaussian Estimation

We now show how to estimate a high-dimensional Gaussian with unknown mean and covariance in statistical distance, under pure DP. This is by combining our result on private covaraince estimation and the prior work on mean estimation [87]. The argument is standard: we simply estimate the mean and covariance separately and apply Lemma 4.3.1 that converts closeness in parameters to closeness in distribution.

Pure DP mean estimation The first step of the algorithm is to privately estimate the mean. The idea is simple and similar to [99]. If Σ were known, then we can apply $\Sigma^{-1/2}$ to the samples and run the mean estimation algorithm (PureDPMean) of [87] on input $\left\{\Sigma^{-1/2}X_i\right\}_{i=1}^n$. For $X_i \sim \mathcal{N}(\mu, \Sigma)$, we have $\Sigma^{-1/2}X_i \sim \mathcal{N}\left(\Sigma^{-1/2}\mu, I\right)$. Thus, the output $\widehat{\mu}$ of PureDPMean satisfies that $\left\|\Sigma^{-1/2}(\mu - \widehat{\mu})\right\|_2 \leq \alpha$, which is what we need for distribution estimation (Lemma 4.3.1).

In the setting when Σ is unknown, we apply our our preconditioning algorithm to privately learn a matrix A that is spectrally close to $\Sigma^{-1/2}$. This effectively sets the samples to have near identity covariance. We show it suffices for our purpose.

Specifically, our private mean estimation procedure is described by Algorithm 4.5 and its guarantees given below.

1. **Input:** Samples $\mathcal{X} = \{X_1, X_2, \ldots, X_{3n}\} \subset \mathbb{R}^d$, privacy parameter $\varepsilon > 0$.
2. For each $i = 1, 2, \ldots, n$, let $Y_i = \frac{1}{\sqrt{2}}(X_{2i} - X_{2i-1})$.
3. Compute the preconditioning matrix $A \in \mathbb{R}^{d\times d}$ using $\{Y_1, \ldots, Y_n\}$ as input to Algorithm 4.3 and privacy parameter $\varepsilon/2$.
4. Run the algorithm PureDPMean in Theorem 4.3.4 [87] on samples $\left\{AX_{2n+1}/\sqrt{20}, \ldots, AX_{3n}/\sqrt{20}\right\}$ with privacy parameter $\varepsilon/2$, error parameter $\alpha/\sqrt{20}$, and failure rate $\beta = 0.01$ to obtain a mean estimate $\tilde{\mu} \in \mathbb{R}^d$.
5. **Output:** Mean estimate $\widehat{\mu} = \sqrt{20}A^{-1}\tilde{\mu}$.

Figure 4.5: Private mean estimation algorithm

Lemma 4.4.2 (Pure DP Gaussian mean estimation). *Let $\alpha > 0$ be an error parameter, $\varepsilon > 0$ be a privacy parameter, $R \in \mathbb{R}$ and $d \in \mathbb{N}$. There is a polynomial-time ε-DP algorithm that, given ε and*

$$n \geq \widetilde{O}\left(\frac{d^2\log(\kappa)}{\varepsilon} + \frac{d}{\alpha^2\varepsilon} + \frac{d\log(R)}{\varepsilon}\right)$$

i.i.d. samples $\mathcal{X} = \{X_1, \ldots, X_n\}$ from $\mathcal{N}(\mu, \Sigma)$ for an unknown μ satisfying $\|\mu\|_2 \leq R$ and an unknown $\Sigma \in \mathbb{R}^{d\times d}$ satisfying $I \preceq \Sigma \preceq \kappa I$, outputs $\widehat{\mu}$ satisfying

$$\left\|\Sigma^{-1/2}(\mu - \widehat{\mu})\right\|_2 \leq \alpha$$

with probability at least 0.9.

Proof. The privacy follows from basic composition of the privacy property of Algorithm 4.3 and PureDPMean. We focus on the utility analysis proving that $\left\|\Sigma^{-1/2}(\mu - \widehat{\mu})\right\|_2 \leq \alpha$.

Since input samples $\{X_1, X_2, \ldots, X_{2n}\}$ are i.i.d. from $\mathcal{N}(\mu, \Sigma)$, the random vectors Y_i are i.i.d. according to $\mathcal{N}(0, \Sigma)$. By Theorem 4.4.2, the choice of n and our assumption on Σ, step 2 of Algorithm 4.5 outputs a preconditioning matrix $A \in \mathbb{R}^{d\times d}$ such that

$$I \preceq A\Sigma A \preceq 20I \tag{4.4.5}$$

with probability at least 0.99. Since $I \preceq \Sigma$ and $A\Sigma A \preceq 20I$, we have $\|A\|_2 \leq \sqrt{20}$. Since $\{X_{2n+1}, \ldots, X_{3n}\}$ are i.i.d. from $\mathcal{N}(\mu, \Sigma)$, then in step 3, $\{AX_{2n+1}/\sqrt{20}, \ldots, AX_{3n}/\sqrt{20}\}$ are i.i.d. according to $\mathcal{N}(A\mu/\sqrt{20}, A\Sigma A/20)$. Recall that the guarantee from A ensures $A\Sigma A/20 \preceq I$. Moreover, $\|A\mu/\sqrt{20}\|_2 \leq \|A\|_2\|\mu\|_2/\sqrt{20} \leq R$, since $\|A\|_2 \leq \sqrt{20}$.

Therefore, conditioned on these events, Theorem 4.3.4 implies that the mean estimate $\tilde{\mu}$ in step 3 satisfies that $\|A\mu/\sqrt{20} - \tilde{\mu}\|_2 \leq \alpha/\sqrt{20}$, and hence $\|A(\mu - \widehat{\mu})\|_2 \leq \alpha$, with probability at least 0.99. Now since $I \preceq A\Sigma A$, we have $\|\Sigma^{-1/2}A^{-1}\|_2 \le 1$. Hence, $\left\|\Sigma^{-1/2}(\mu - \widehat{\mu})\right\|_2 \le \|\Sigma^{-1/2}A^{-1}\|_2 \cdot \|A(\mu - \widehat{\mu})\|_2 \le \alpha$, with probability at least 0.99. The proof follows by applying a union bound over the failure probability of step 2 and 3 of Algorithm 4.5. □

Putting it Together We now put together Lemma 4.4.2 on mean estimation and Theorem 4.4.3 on covariance estimation to show:

Theorem 4.4.4 (Pure DP Gaussian estimation)**.** *Let $\alpha > 0$ be an error parameter, $\varepsilon > 0$ be a privacy parameter, $R \in \mathbb{R}$ and $d \in \mathbb{N}$. There is a polynomial-time ε-DP algorithm that, given ε and*

$$n \ge \widetilde{O}\left(\frac{d^2\log(\kappa)}{\varepsilon} + \frac{d^2}{\alpha^2\varepsilon} + \frac{d\log(R)}{\varepsilon}\right)$$

i.i.d. samples $\mathcal{X} = \{X_1, \dots, X_n\}$ from $\mathcal{N}(\mu, \Sigma)$ for an unknown μ satisfying $\|\mu\|_2 \le R$ and an unknown $\Sigma \in \mathbb{R}^{d\times d}$ satisfying $I \preceq \Sigma \preceq \kappa I$, outputs $\widehat{\mu}, \widehat{\Sigma}$ such that

$$TV\left(\mathcal{N}(\mu, \Sigma), \mathcal{N}\left(\widehat{\mu}, \widehat{\Sigma}\right)\right) \le O(\alpha) \tag{4.4.6}$$

with probability at least 0.8.

Proof. For simplicity, assume that n is even. We use $n/2$ samples as input to the private mean estimation algorithm (Algorithm 4.5) and another $n/2$ samples for covariance estimation (Algorithm 4.4), both with a privacy parameter $\varepsilon/2$. Privacy follows from basic composition (Theorem 4.3.1). For utility, by our choice of n, Lemma 4.4.2 implies that $\left\|\Sigma^{-1/2}(\mu - \widehat{\mu})\right\|_2 \le \alpha$ with probability 0.9, and Theorem 4.4.3 implies that $\left\|\Sigma^{-1/2}\widehat{\Sigma}\Sigma^{-1/2} - I\right\|_F \le \alpha$ with probability 0.9. By union bound with probability 0.8, both steps succeed and thus Lemma 4.3.1 yields (4.4.6). □

Robustness

We now argue that our algorithms are robust to adversarial corruptions, with the cost that estimation error generally is worsened to $\alpha + O(\sqrt{\eta})$, where η is the fraction of corrupted samples. As we discussed, our algorithm for learning Gaussian under pure DP is by reducing the problem to black-box applications of the main procedure from [87], namely, PureDPMean in Theorem 4.3.4. We exploit the robustness property of PureDPMean to show:

Theorem 4.4.5 (Robust pure DP Gaussian estimation)**.** *Let $\alpha > 0$ be an error parameter, $\varepsilon > 0$ be a privacy parameter, $R \in \mathbb{R}$ and $d \in \mathbb{N}$. For a sufficiently small constant η, there is a polynomial-time ε-DP algorithm that, given ε and*

$$n \ge \widetilde{O}\left(\frac{d^2\log(\kappa)}{\varepsilon} + \frac{d^2}{\alpha^2\varepsilon} + \frac{d\log(R)}{\varepsilon}\right)$$

η-corrupted samples $\mathcal{X} = \{X_1, \dots, X_n\}$ from $\mathcal{N}(\mu, \Sigma)$ for an unknown μ satisfying $\|\mu\|_2 \leq R$ and an unknown $\Sigma \in \mathbb{R}^{d\times d}$ satisfying $I \preceq \Sigma \preceq \kappa I$, outputs $\widehat{\mu}$ and $\widehat{\Sigma}$ such that

$$TV\left(\mathcal{N}(\mu, \Sigma), \mathcal{N}\left(\widehat{\mu}, \widehat{\Sigma}\right)\right) \leq O(\alpha + \sqrt{\eta}) \tag{4.4.7}$$

with probability at least 0.8.

To give a proof sketch, the key step is to observe that our recursive preconditioning algorithm (Algorithm 4.3) is robust. Recall that the algorithm simply calls our weak preconditioning scheme (Algorithm 4.2) recursively (for $\log\kappa$ times). This weak scheme, in turn, runs PureDPMean to roughly estimate Σ, up to an additive error of 0.01κ in the (absolute) Frobenius norm. We observe that the robustness property of PureDPMean suffices to yield the same error guarantee even under η-corruption. Hence, the recursive preconditioning algorithm retains its performance under corruption. Finally, the remaining steps of our algorithms for mean and covariance estimation simply calls PureDPMean on the preconditioned samples. We lose the extra factor of $\sqrt{\eta}$ from there.

We remark that in the analysis we make no attempt to optimize the breakdown point of the algorithm. (In fact, it depends on the hidden constant in the $O(\sqrt{\eta})$ error term of PureDPMean.)

Proof of Theorem 4.4.5. We start by modifying step 2 of Algorithm 4.2 to invoke Algorithm 4.1 with $\alpha = 0.0099$ instead. By the choice of n, α and for a sufficiently small η, Theorem 4.3.4 implies that the rough estimate $\widehat{\Sigma}$, computed in the second step of Algorithm 4.2, satisfies

$$\left\|\widehat{\Sigma} - \Sigma\right\|_F \leq 0.01\kappa \tag{4.4.8}$$

with probability $1 - \beta$. Observe that with the error bound above, the rest of the proof of Lemma 4.4.1 remains valid. Inspecting the analysis of the recursively preconditioning algorithm, we note that the error guarantee of Lemma 4.4.1 suffices to imply Theorem 4.4.2.

We now argue for the error rate on covariance and mean estimation, separately.

- For covariance estimation, consider Algorithm 4.4 and its guarantees Theorem 4.1.1. Step 2, the preconditioning step, of Algorithm 4.4 retains its performance exactly. In step 3, we instead get an estimate $\widehat{\Sigma}$ such that $\left\|\Sigma^{-1/2}\widehat{\Sigma}\Sigma^{-1/2} - I\right\|_F \leq \alpha + O(\sqrt{\eta})$, due to the robustness property of PureDPMean.

- For mean estimation, consider Algorithm 4.5 and its guarantees of Lemma 4.4.2. Similarly, step 3 of the algorithm retains its performance exactly, and we lose an extra $O(\sqrt{\eta})$ factor in step 4. Hence, the algorithm outputs an $\widehat{\mu}$ such that $\|\mu - \widehat{\mu}\| \leq \alpha + O(\sqrt{\eta})$.

Applying the Lemma 4.3.1 converts the parameter closeness to distribution closeness. □

4.5 Lower Bounds Pure DP Covariance Estimate

In this section, we detail an information-theoretic lower bound for Gaussian covariance estimation under differential privacy constraints. Like our upper bound, our lower bound has a logarithmic dependence on κ. Our proof builds on previous packing-style lower bound arguments for pure DP estimation [80, 2, 30, 99].

High-level Overview For any integer $d \geq 2$, given any covariance matrix $\Sigma \in \mathbb{R}^{d\times d}$ that satisfies $I \preceq \Sigma \preceq \kappa I$, we show that to learn $\mathcal{N}(0,\Sigma)$ within total variation $O(\alpha)$, we must require sample complexity $\Omega\left(\frac{d^2}{\varepsilon}\log\left(\frac{d\kappa}{\alpha}\right)+\frac{d^2}{\varepsilon\alpha}\right)$. We proceed in two steps: first we show a lower bound of $\Omega\left(\frac{d^2}{\varepsilon}\log\left(\frac{d\kappa}{\alpha}\right)\right)$ which does not depend polynomially on $1/\alpha$. Then we combine it with a previous lower bound of $\Omega\left(\frac{d^2}{\varepsilon\alpha}\right)$, due to [99]. Taken together, this gives us a lower bound of $\Omega\left(\frac{d^2}{\varepsilon}\log\left(\frac{d\kappa}{\alpha}\right)+\frac{d^2}{\varepsilon\alpha}\right)$:

Theorem 4.5.1. *Let $\varepsilon, \alpha \in (0,1)$, $d \geq 2$. Any ε-DP algorithm that, given n i.i.d. samples $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$ from $\mathcal{N}(0,\Sigma)$ for an unknown $\Sigma \in \mathbb{R}^{d\times d}$ satisfying $I \preceq \Sigma \preceq \kappa I$, outputs $\widehat{\Sigma} = \widehat{\Sigma}(\mathcal{X})$ such that, with probability at least* 0.9*,*

$$TV\Big(\mathcal{N}(0,\Sigma), \mathcal{N}(0,\widehat{\Sigma})\Big) < O(\alpha),$$

must require

$$n = \Omega\left(\frac{d^2}{\varepsilon}\log\left(\frac{d\kappa}{\alpha}\right)+\frac{d^2}{\varepsilon\alpha}\right).$$

Condition Number Lower Bound

We rely on previous work on differentially private hypothesis selection [30]: given samples from some unknown distribution P (e.g., defined by $\mathcal{N}(0,\Sigma)$ for some $\Sigma \in \mathbb{R}^{d\times d}$), what is the closest distribution to P in some set $\mathcal{H}$?

Crucial to the derivation of the lower bound of $\Omega\left(\frac{d^2}{\varepsilon}\log\left(\frac{d\kappa}{\alpha}\right)\right)$ is the notion of covers and packings:

Definition 4.5.1 (γ-Cover)**.** *A γ-cover of a set of distributions $\mathcal{H}$ is a set of distributions $\mathcal{C}_\gamma$, such that for every $H \in \mathcal{H}$ there exists $P \in \mathcal{C}_\gamma$ with the following property: $TV(P,H) \leq \gamma$.*

Definition 4.5.2 (γ-Packing)**.** *A γ-packing of a set of distributions $\mathcal{H}$ is a set of distributions $\mathcal{P}_\gamma \subseteq \mathcal{H}$, such that for every pair $P, Q \in \mathcal{P}_\gamma$, $TV(P,Q) > \gamma$.*

The following lemma states that, provided we can find an α-packing, we can get a sample complexity lower bound for pure DP:

Lemma 4.5.1 (Lemma 5.1 in [30]). *Let $\mathcal{P}_\alpha$ be an α-packing of a set of distributions $\mathcal{H}$. Then for any $P \in \mathcal{H}$, any ε-differentially private algorithm that takes sample $X_1, \ldots, X_n \sim P$ produces, with probability at least 0.99, a distribution $\widehat{H}$ such that $TV(P, \widehat{H}) \leq \alpha/2$ requires*

$$n = \Omega\left(\frac{\log |\mathcal{P}_\alpha|}{\varepsilon}\right).$$

Note that the sample complexity lower bound in Lemma 4.5.1 is not of the form $\Omega\left(\frac{\log|\mathcal{P}_\alpha|}{\varepsilon\alpha}\right)$ since such a lower bound would contradict already-existing upper bounds.

Lemma 4.5.2 shows the existence of an α-cover of a set of d-dimensional Gaussian distributions:

Lemma 4.5.2 (Lemma 6.8 in [30]). *Let $\mu \in \mathbb{R}^d, \Sigma \in \mathbb{R}^{d\times d}$ such that $\|\mu\|_2 \leq R$ and $I \preceq \Sigma \preceq \kappa I$. Then there exists an α-cover of the set of Gaussian distributions $\mathcal{N}(\mu, \Sigma)$ of size*

$$O\left(\frac{dR}{\alpha}\right)^d \cdot O\left(\frac{d\kappa}{\alpha}\right)^{d(d+1)/2}.$$

To use the lower bound from Lemma 4.5.1, we need an α-packing and not an α-cover. The following lemma relates the size of the largest α-packing to the smallest α-cover:

Lemma 4.5.3 (Lemma 5.2 in [30]). *Let $\mathcal{H}$ be a set of distributions. If p_α and c_α are the size of the largest α-packing and smallest α-cover of $\mathcal{H}$, respectively, then*

$$p_{2\alpha} \leq c_\alpha \leq p_\alpha.$$

We can now obtain the following corollary of a sample complexity lower bound that depends on κ:

Corollary 4.5.1. *Fix $\varepsilon, \alpha \in (0,1)$. For any integer $d \geq 2$ and covariance matrix $\Sigma \in \mathbb{R}^{d\times d}$ satisfying $I \preceq \Sigma \preceq \kappa I$, let $\hat{\Sigma} = \hat{\Sigma}(\mathcal{X})$ be any ε-DP algorithm such that with probability at least 0.99, given n i.i.d. samples $\mathcal{X} = \{X_1, \ldots, X_n\}$ from $\mathcal{N}(0, \Sigma)$, the algorithm has the following guarantee:*

$$TV(\mathcal{N}(0,\Sigma), \mathcal{N}(0,\hat{\Sigma})) < \alpha/2.$$

Then

$$n = \Omega\left(\frac{d^2}{\varepsilon}\log\left(\frac{d\kappa}{\alpha}\right)\right).$$

Proof. Set $R = O(\alpha/d)$ since the mean of the Gaussian is a constant. By Lemma 4.5.3 and Lemma 4.5.2, there exists an α-packing of size $O((\frac{d\kappa}{\alpha})^{d^2})$. And by Lemma 4.5.1, this gives us a sample complexity lower bound of $n = \Omega(\frac{d^2}{\varepsilon}\log(\frac{d\kappa}{\alpha}))$.

□

Precision Matrix Lower Bound

We now proceed to show the lower bound of $\Omega(\frac{d^2}{\varepsilon\alpha})$. The proof of the theorem relies on the following technical lemma on the TV distance between two mean-zero Gaussians with different covariance. By Theorem 4.5.4, it suffices to derive a lower bound on $\left\|\Sigma_1^{1/2}\Sigma_2^{-1}\Sigma_1^{1/2} - I_d\right\|_F$ where Σ_1 and Σ_2 are the unknown covariance matrix and the output from the ε-DP algorithm, respectively.

Lemma 4.5.4 (Lemma 3.5 in [50], Theorem 1.1 in [51])**.** *Let $\mu \in \mathbb{R}^d$ and let Σ_1, Σ_2 be positive definite symmetric $d \times d$ matrices. Use $\lambda_1, \ldots, \lambda_d$ to denote the eigenvalues of $\Sigma_1^{-1}\Sigma_2 - I_d$. Then,*

$$0.01 \le \frac{TV(\mathcal{N}(\mu, \Sigma_1), \mathcal{N}(\mu, \Sigma_2))}{\min\left\{1, \sqrt{\sum_{i=1}^d \lambda_i^2}\right\}} \le 1.5.$$

Also,

$$TV(\mathcal{N}(0, \Sigma_1), \mathcal{N}(0, \Sigma_2)) \ge \frac{1}{100} \min\left\{1, \left\|\Sigma_1^{1/2}\Sigma_2^{-1}\Sigma_1^{1/2} - I_d\right\|_F\right\}.$$

For two Gaussians with the same mean, [51] gives closed-form lower and upper bounds in TV distance. The upper and lower bounds are within (small) constants of each other.

Note that since $\Sigma_1^{-1}\Sigma_2$ and $\Sigma_1^{-1/2}\Sigma_2\Sigma_1^{-1/2}$ have the same spectrum, we have

$$\sum_{i=1}^d \lambda_i^2 = \left\|\Sigma_1^{-1/2}\Sigma_2\Sigma_1^{-1/2} - I_d\right\|_F^2.$$

The following statement, although not explicitly stated in [99], can be inferred from [99]. We combine the statement with cleaner versions of [50] to derive our lower bound.

Proposition 4.5.1 (see also [99])**.** *Let $d \ge 2$. Let $\widehat{\Sigma}$ be an ε-DP algorithm that outputs an approximation of the covariance matrix Σ, where $\frac{1}{2}I \preceq \Sigma \preceq 2I$, such that*

$$\mathop{\mathbb{E}}_{X \sim \mathcal{N}(0,\Sigma)^{\otimes n}} \left[\left\|\left[\widehat{\Sigma}(X)\right]^{-1} - \Sigma^{-1}\right\|_F^2\right] \le \frac{\alpha^2}{64}.$$

Then $n = \Omega\left(\frac{d^2}{\varepsilon\alpha}\right)$.

Proof. We can equivalently prove that $\mathbb{E}_{X\sim\mathcal{N}(0,\Sigma)^{\otimes n}}\left[\left\|\widehat{\Sigma}(X) - \Sigma\right\|_F^2\right] \le \frac{\alpha^2}{64}$. We assume that $n = O(\frac{d^2}{\varepsilon\alpha})$ and will reach a contradiction. Let $\mathcal{M}_d$ denote the set of d-by-d real symmetric matrices. Consider

$$\mathcal{S}_n = \left\{S \in \mathcal{M}_d \,:\, S \text{ has diagonal entries 0 and non-diagonals either } -\frac{\alpha}{2d} \text{ or } +\frac{\alpha}{2d}\right\}.$$

Clearly,

$$|\mathcal{S}_n| = 2^{(d^2-d)/2}.$$

Let $U \sim \mathrm{Unif}(\mathcal{S}_n)$ be the uniform distribution over $\mathcal{S}_n$. For any $V \in \mathcal{S}_n$, define $\Sigma = \Sigma(V) = I + V$. The ε-DP algorithm $\widehat{\Sigma}$ aims to output a matrix as close to Σ as possible.

Define Z and Z' such that

$$Z = \left\langle \widehat{\Sigma}(X), V \right\rangle = 2\sum_{i<j} \widehat{\Sigma}(X)_{ij} \cdot V_{ij}, \tag{4.5.1}$$

and

$$Z' = \left\langle \widehat{\Sigma}(X'), V \right\rangle = 2\sum_{i<j} \widehat{\Sigma}(X')_{ij} \cdot V_{ij}. \tag{4.5.2}$$

Let V, V' be independent samples from $\mathrm{Unif}(\mathcal{S}_n)$. Then sample $X \sim \mathcal{N}(0, \Sigma(V))^{\otimes n}$ and $X' \sim \mathcal{N}(0, \Sigma(V'))^{\otimes n}$. Also, note that $\left\|\widehat{\Sigma}(X) - \Sigma(V)\right\|_F^2 = \sum_{i<j} 2(\widehat{\Sigma}_{ij}(X) - \Sigma_{ij}(V))^2$.

Then, by Lemma 4.5.5, $\mathbb{E}[Z] \geq \frac{\alpha^2}{16} - \frac{1}{2}\left\|\widehat{\Sigma}(X) - \Sigma(V)\right\|_F^2 \geq \frac{7\alpha^2}{128}$.

Finally, observe that by Lemma 4.5.6 and Lemma 4.5.5, $\mathbb{P}[Z > \alpha^2/32] = \Omega(1)$ and $\mathbb{P}[Z' > \alpha^2/32] \leq \exp(-\Omega(d^2))$ but Lemma 4.5.6 leads to $\Omega(1) \leq \exp(4\alpha\varepsilon n) - \Omega(d^2)$ which would require $n = \Omega(\frac{d^2}{\varepsilon\alpha})$. □

The following lemma is used to lower bound the expected value of $Z = \left\langle \widehat{\Sigma}(X), V \right\rangle$ in terms of $\left\|\widehat{\Sigma}(X) - \Sigma(V)\right\|_F^2$:

Lemma 4.5.5 (Claim 6.12 of [99]). *For any $d \geq 2$,*

$$\mathbb{E}[Z] \geq \frac{\alpha^2}{16} - \frac{1}{2}\left\|\widehat{\Sigma}(X) - \Sigma(V)\right\|_F^2 \geq \frac{7\alpha^2}{128}.$$

The following lemma relates Z to Z':

Lemma 4.5.6 (Claim 6.13 and 6.14 of [99]). *For Z, Z' as defined in Equations (4.5.1) and (4.5.2), we have:*

1. $\mathbb{P}[Z > \alpha^2/32] \leq \exp(4(\alpha\varepsilon n)) \cdot \mathbb{P}[Z' > \alpha^2/32]$.
2. $\mathbb{P}[Z' > \alpha^2/32] \leq \exp(-\Omega(d^2))$.

Putting it Together

We now prove Theorem 4.5.1:

Proof of Theorem 4.5.1. By Lemma 4.5.4, to prove Theorem 4.1.2, it suffices to show that it is impossible to have both $\left\|\Sigma^{1/2}\widehat{\Sigma}^{-1}\Sigma^{1/2} - I\right\|_F \leq \alpha$ and $n = O\left(\frac{d^2}{\varepsilon\alpha}\right)$ using an ε-DP algorithm to compute $\widehat{\Sigma} = \widehat{\Sigma}(\mathcal{X})$.

First note that

$$\left\|\Sigma^{1/2}\widehat{\Sigma}^{-1}\Sigma^{1/2} - I\right\|_F = \left\|\Sigma^{1/2}(\widehat{\Sigma}^{-1} - \Sigma^{-1})\Sigma^{1/2}\right\|_F \tag{4.5.3}$$

$$\geq \sigma_d(\Sigma^{1/2})^2 \left\|\widehat{\Sigma}^{-1} - \Sigma^{-1}\right\|_F, \tag{4.5.4}$$

where $\sigma_d(\Sigma^{1/2})$ denotes the smallest singular value of $\Sigma^{1/2}$. Because $\Sigma^{1/2}$ is symmetric and positive definite, we know that eigenvalues coincide with its singular values. Also, the eigenvalues of $\Sigma^{1/2}$ are larger than $1/\sqrt{2}$ so that

$$\left\|\Sigma^{1/2}\widehat{\Sigma}^{-1}\Sigma^{1/2} - I\right\|_F \geq \frac{1}{2}\left\|\widehat{\Sigma}^{-1} - \Sigma^{-1}\right\|_F.$$

By Proposition 4.5.1, $\left\|\widehat{\Sigma}^{-1} - \Sigma^{-1}\right\|_F^2 \geq \frac{\alpha^2}{64}$. As a result, $\left\|\Sigma^{1/2}\widehat{\Sigma}^{-1}\Sigma^{1/2} - I\right\|_F \geq \alpha/16$, contradicting the assumption that

$$\mathrm{TV}\left(\mathcal{N}(0,\Sigma), \mathcal{N}(0,\widehat{\Sigma})\right) < \alpha/1600.$$

Thus, $n = \Omega\left(\frac{d^2}{\varepsilon\alpha}\right)$. By Corollary 4.5.1, we obtain a lower bound of $n = \Omega(\frac{d^2}{\varepsilon}\log(\frac{d\kappa}{\alpha}))$. As a result, the lower bound is a max of $n = \Omega\left(\frac{d^2}{\varepsilon\alpha}\right)$ and $n = \Omega\left(\frac{d^2}{\varepsilon}\log(d\kappa/\alpha)\right)$ which asymptotically is

$$n = \Omega\left(\frac{d^2}{\varepsilon}\log\left(\frac{d\kappa}{\alpha}\right) + \frac{d^2}{\varepsilon\alpha}\right).$$

This completes the proof. □

Chapter 5

Memory-Efficient Online Learning

We provide the first sub-linear space and sub-linear regret algorithm for online learning with expert advice (against an oblivious adversary), addressing an open question raised recently by Srinivas, Woodruff, Xu and Zhou (STOC 2022). We also demonstrate a separation between oblivious and (strong) adaptive adversaries by proving a linear memory lower bound of any sub-linear regret algorithm against an adaptive adversary.

Our algorithm is based on a novel pool selection procedure that bypasses the traditional wisdom of leader selection for online learning, and a generic reduction that transforms any weakly sub-linear regret $o(T)$ algorithm to $T^{1-\alpha}$ regret algorithm, which may be of independent interest. Our lower bound utilizes the connection of no-regret learning and equilibrium computation in zero-sum games, leading to a proof of a strong lower bound against an adaptive adversary.

5.1 Introduction

Online prediction with expert advice is a fundamental task in sequential decision making and is the backbone of optimization [81], bandit learning [109], control theory [65], among many other fields. The problem is usually formulated as an online forecasting process that repeats for T days. On each day, the algorithm is asked to provide a prediction of an outcome, given the advice from n experts on the current day and all the previous information. After announcing a decision, it then receives feedback on the outcome and the loss of the expert predictions, normalized to $[0,1]^n$. The objective in online learning is typically to compete against the best expert in hindsight, and the performance of an algorithm is measured in terms of *regret*, which is defined as the additional cost of the algorithm over the best expert.

The celebrated Multiplicative Weights Update (MWU) method solves this online prediction problem with an optimal $O\left(\sqrt{T \log n}\right)$ regret [13]. Similar regret bounds have also been attained by weighted majority vote [119], follow-the-perturbed-leader (FTPL) scheme [95] and online mirror descent (OMD) [81]. Besides the original motivation of online prediction, MWU and the follow-up variants find broader applications in algorithmic design, game theory and machine learning, with notable examples including efficient algorithms for linear programming and semi-definite programming [73], approximate solution for max flow [44], equilibrium computation [33, 72] and boosting [71].

Despite a long history of research since 80's, the question of *space complexity* has been little explored in the online learning literature. All existing approaches explicitly track the cumulative cost of every expert and follow the advice of a (regularized or perturbed) leading expert, thus requiring a memory of size $\Omega(n)$. Motivated by this lack of understanding, a very recent work by Srinivas, Woodruff, Xu and Zhou [146] initiates the study of memory complexity of expert learning. They prove a lower bound of $\Omega(\sqrt{nT/S})$ regret for any algorithm using $O(S)$ memory, even when the loss sequence are i.i.d distributed. This implies that $\Omega(n)$ space is necessary to get the *optimal* $\sqrt{T}$ regret bound. On the positive side, they show that sub-linear regret is achievable in sub-linear space, but only in random-order streams or when the best expert itself incurs sub-linear loss. These structural assumptions are arguably rather strong. This leaves open the general question of attaining sub-linear regret in a memory-efficient fashion.

In this work, we revisit the classic online learning problem with an eye towards an better understanding its space complexity. On the algorithmic front, we ask, for constant $\alpha, \beta \in (0, 1)$:

Can we achieve sub-linear $O(T^{1-\alpha})$ regret in online learning, using sub-linear $O(n^{\beta})$ space?

We resolve this open question by designing the first sub-linear space online learning algorithm, in general worst-case settings. The algorithm assumes an *oblivious adversary* that fixes the loss sequence in advance. To complement this, we prove a lower bound showing that against an (strong) adaptive adversary, no sub-linear space algorithm can achieve sub-linear regret.

Our results

To introduce the results, we specify our problem setting (see section 2.2 for a formal description). We consider a general setup of the expert problem with T days and n experts. On each day $t \in [T]$, the algorithm decides to play one of the experts $i_t \in [n]$. Subsequently, nature reveals the loss $\ell_t(i)$ for all $i \in [n]$ and the algorithm incurs a loss of $\ell_t(i_t)$. We assume that $\ell_t(i) \in [0, 1]$ for all $t \in [T]$ and $i \in [n]$. The goal is to design an algorithm such that the (total) regret $\textsc{Regret}(T) = \mathbb{E}[\sum_{t\in[T]} \ell_t(i_t)] - \min_{i^* \in [n]} \sum_{t\in[T]} \ell_t(i^*)$ is sub-linear in T.[1] An oblivious adversary (randomly) chooses the loss vectors independent of the algorithm's decisions.

Our main result is the following.

[1]We call $T^{1-\alpha}$ sub-linear regret (for constant α), and $o(T)$ weakly sub-linear. We use $\widetilde{O}(\cdot)$ to hide polylogarithmic factors in n, T and $O_n(\cdot)$ to hide polynomial dependence on n. High probability refers to probability at least $1 - 1/\operatorname{poly}(T)$.

Theorem 5.1.1 (Informal, see Theorem 5.4.1)**.** *Let $n, T \geq 1$, $\delta \in (0, 1]$. There exists an online learning algorithm that achieves a total regret of $O_n\left(T^{\frac{2}{2+\delta}}\right)$ with high probability against an oblivious adversary, using $\widetilde{O}(n^{\delta})$ memory.*

The theorem provides a general memory-regret trade-off. To give some concrete examples:

- We can get $O_n\left(T^{4/5}\right)$ regret in $\widetilde{O}\left(\sqrt{n}\right)$ memory (by setting $\delta = 0.5$).
- We can get $O_n(T^{0.67})$ regret in $\widetilde{O}\left(n^{0.99}\right)$ memory (by setting $\delta = 0.99$).

On a conceptual level, our algorithm breaks the following traditional wisdom in online prediction. We recall that the classic regret-minimization algorithms, such as MWU and FTPL, all track the loss of *all* experts and follow a (regularized or perturbed) leader. Alternatively, one may assume an oracle that outputs the leading expert (a.k.a. oracle-efficient online learning [82, 66]). Therefore, the task of identifying a leading expert is usually believed to be a necessary sub-routine of regret minimization. Indeed, with $\Omega(n)$ memory, identifying a leader and maintaining its cumulative loss is trivial, while in principle online prediction is a much harder task. Perhaps surprisingly, in the sub-linear space regime, our results suggest the opposite. It is known that under space constraint, one cannot identify a leader, or even constant approximate its loss [146]. On the other hand, theorem 5.1.1 implies that achieving sub-linear regret is possible with arbitrarily small polynomial space. Therefore, conceptually this work shows that *online prediction is easier than and does not require tracking the leader, in the low space regime.*

In the case that only polylogarithmic space is allowed, we also give a weakly sub-linear regret algorithm. In particular, by setting $\varepsilon = 1/\operatorname{poly}\log(nT)$ in the following theorem, the algorithm achieves $o(T)$ regret in $\widetilde{O}(1)$ memory (for sufficiently large T).

Theorem 5.1.2 (Informal, see theorem 5.3.1)**.** *Let $n \geq 1$, $\varepsilon \in (1/\sqrt{n}, 1/2)$ and $T \geq \Omega(\varepsilon^2 n)$. There exists an online learning algorithm that achieves a total regret of $\widetilde{O}\left(\varepsilon T + T^{2/3}\left(\varepsilon^2 n\right)^{1/3}\right)$ with high probability against an oblivious adversary, using $\widetilde{O}(\varepsilon^{-2})$ memory.*

Our algorithms assume an oblivious adversary, a standard model in online learning as well as its cousin fields like online, streaming and dynamic algorithms [102, 10]. In a variety of applications, however, algorithms are required to work against adaptive adversaries [133, 145, 155, 20, 11].

To complement our algorithmic results, we consider a strong adaptive adversary model, where the costs may be chosen adversarially that depend on the algorithm's prior randomness and decisions; see definition 5.2.2 for a formal definition. In this setting, MWU still achieves $\widetilde{O}(\sqrt{T})$ regret but uses linear memory. We prove that $\Omega(n)$ memory is indeed necessary to obtain any sub-linear regret at all:

Theorem 5.1.3 (Informal, see theorem 5.5.1)**.** *Let $0 < \varepsilon < 1/40$. Any algorithm that achieves εT total regret against a strong adaptive adversary requires at least*

$$\Omega\left(\min\left\{\varepsilon^{-1}\log_2 n, n\right\}\right)$$

bits of memory.

The theorem states that $\Omega(n)$ memory is necessary even to get any sub-linear, say, $O\left(T^{0.99}\right)$ regret, for sufficiently large $T = \Omega\left(n^{100}\right)$. In contrast, under the oblivious adversary model, our upper bound (theorem 5.1.1) can attain such regret guarantee in $o(n)$ space. Therefore, this exhibits a separation between oblivious and adaptive adversary model in the low space regime.

Technical overview

We provide a streamlined overview of our approach. For notational convenience, we omit polylogarithmic factors and one should think of $T = \text{poly}(n)$.

A baseline algorithm for weakly sub-linear regret

We first present an algorithm that achieves weakly sub-linear regret $O(\varepsilon T)$ using space $O(\varepsilon^{-2})$, then provide a novel width reduction procedure to make it sub-linear in section 5.1.

A natural idea is to carefully maintain a small pool of experts and run MWU over them. To begin with, we divide the T days into T/B epochs with B contiguous days in each epoch. At the beginning of each epoch, we sample a random set of new experts and add them into the pool, then executing MWU over the pool (starting from uniform weight). MWU guarantees that the algorithm is always competitive with the best expert(s) *in the pool*, and this turns online prediction into the task of maintaining a good pool of experts. The immediate hurdle is that, due to space constraint, the pool must be kept small and hence the best expert is likely to be outside of the pool. Therefore, we need to maintain the pool so that it consists of good experts with respect to each particular epoch, but without knowing the performance of the experts outside the pool.

Maintaining the pool The algorithm samples a small number of experts into its pool at the beginning of each epoch. After a few epochs, the pool size would grow and exceed the memory budget. The key algorithmic task is to design a rule of evicting experts.

The first and natural idea is track the cumulative loss of each expert in the pool, and intuitively, only an expert with low average loss should be reserved. However, it is easy to come up with instances, where (1) the best expert i^* has low error in every epoch; but (2) random new experts have even lower error in 1/3 epochs (though higher in the rest). If these 1/3 epochs are evenly spaced, then i^* can be easily kicked out by the new expert. Subsequently, it would take a long time to get back, since our sampling rate has to be low to respect the space constraint. This is clearly undesirable.

Intuitively, consider two experts in the pool with equally low average loss since their joining. The one that has stayed longer should be treated differently from the other, since the former is more stable against further loss. Hence, the second idea is to keep experts that have lived a long time in the pool. In other words, an expert has no reason to be evicted if it is "Pareto-optimal": it has either stayed long or achieved little loss. Formally, we say an expert j dominates another expert i, if j joins the pool earlier than i and has achieved at most ε more average loss than i since expert i joins. At the end of every epoch, our algorithm evaluates all experts. Any expert i being dominated by another is evicted.

Bounding memory Notice that the algorithm does not dictate any explicit bound on the pool size, let alone memory. This is the challenge that we now resolve.

For that, the key observation is the following. Consider any surviving expert i that joins the pool later than some other j. It has much smaller, in fact at least ε, loss on its interval than j, by our eviction rule. Then, we claim that either (1) expert j has $\varepsilon/2$ larger average loss than expert i, or (2) j lives $(1+\varepsilon/2)$ longer than i. The reason is that j has loss at least 0 everywhere, so if (1) doesn't hold, then it takes extra length to catch up with the loss difference. A straightforward proof would bound the pool size by $O(1/\varepsilon^2)$ as both events can happen for at most $O(1/\varepsilon)$ times consecutively. We derive a refined one of $O(1/\varepsilon)$ via a potential function argument. This leads to a memory bound of $O(1/\varepsilon^2)$ because we need to track the performance of each expert over every interval.

Bounding regret To bound the regret, our plan is to show that there exist experts in pool that are competitive to i^* (even if i^* may not be in the pool), except for at most $O(n)$ *unlucky* epochs. For simplicity, assume the algorithm only samples and adds one new expert per epoch. For an epoch t, imagine i^* gets sampled, and it remains alive until the end of epoch $t \le t' \le T/B$. To this end, there must exist an expert $i(t)$ that already lies in the pool, stays alive during $[t, t']$ and outperforms i^*. The reason is that the expert i^* can only be evicted by an older expert, and the eviction time is independent of future randomness (again, we assume the loss sequence is fixed in advance). Hence, $i(t)$ is competitive with i^* during the epochs $[t, t']$ (note this is independent of whether i^* is actually sampled or not). Then we can proceed to $(t'+1)$-th epoch. There is one exception: if i^* is sampled and would not be evicted by the end, then $i(t)$ simply does not exist. On one hand, with probability $1/n$, we would sample i^* and it stays until the end. Otherwise, with probability $1-1/n$, we just lose this epoch and proceed to the next. The latter event should not happen for more than $O(n)$ times with high probability.

In summary, the baseline algorithm is always competitive with best expert in pool, up to a total regret of $T/B \cdot O\left(\sqrt{B}\right)$, by MWU. Moreover, there is always some expert in the pool competitive with the best expert i^* up to an $O(\varepsilon T)$ regret, except in those unlucky epochs which incur $O(nB)$ regret. To further optimize the algorithm, we sample $1/\varepsilon^2$ new experts instead of 1. It turns out that allows us to bound the total regret over the unlucky epochs by $O(\varepsilon^2 nB)$. Putting everything together, the total regret is $O\left(\varepsilon T + T/\sqrt{B} + \varepsilon^2 nB\right) \approx O(\varepsilon T)$.

Bootstrap the baseline and width reduction

The above baseline algorithm has a total regret of $O(\varepsilon T)$, and the bottleneck lies in the following. First, the eviction rule essentially discretizes the loss into multiples of ε and we need to perform more refined division to reduce the loss. On the other hand, one can construct examples showing the pool size grows at least $O(1/\varepsilon)$, since the loss takes range from $[0, 1]$. The idea is to bootstrap the baseline algorithm and to reduce the width of experts.[2]

Precondition the experts The idea is fairly simple: we just run MWU over the original expert i and the baseline algorithm, and take its output prediction as the new expert e_i. Let Δ denote the loss of the baseline algorithm. Roughly speaking, it is guaranteed that the performance of e_i is at least as good as i and the baseline, and since the baseline has average regret of at most ε, e_i takes loss in $[\Delta - \varepsilon, \Delta]$.[3] Therefore, the width of the loss is significantly reduced—from $[0, 1]$ to $[0, \varepsilon]$. This allows us to discretize with an additive factor ε^2 instead of ε. The idea of preconditioning is ubiquitous and powerful in modern algorithm design, but as far as we know, it is the first time applied to online prediction.

Putting things together The above bootstrapping procedure reduces the regret from $O(\varepsilon T)$ to $O(\varepsilon^2 T)$, up to some lower order terms. There is no reason to stop here, and in fact we repeatedly perform the bootstrapping for roughly $\frac{\log T}{\log n}$ times. Carefully balancing with the lower order terms yields a final regret bound of $O_n\left(T^{\frac{2}{2+\delta}}\right)$. At the end, the scheme would maintain a hierarchical pool of experts. The algorithm plays a mixed strategy over them, instead of simply applying MWU. Our approach of transforming a weakly sub-linear regret $O(\varepsilon T)$ algorithm to a sub-linear regret $O(T^{1-\alpha})$ algorithm is general. We believe it could have broad applications in the area of online prediction.

Lower bound via learning in games

Our lower bound draws close connection with learning in games. It is well known that one can use no-regret learning algorithm to compute Nash equilibria of a zero-sum game [72], via the following template: Alice follows a no-regret algorithm with each of her action as an expert, and Bob (the adversary) best responds to it. We construct a family of zero-sum games whose equilibrium (or minmax) strategies are far apart, and any single strategy can only achieve ε-approximate minmax value for a few of them. The construction is by randomly

[2] An expert with loss in $[a, b]$ is said to have width $(b - a)$, see [13] for formal definition.

[3] For technical reasons, we also need to truncate the loss because it is possible that e_i performs much better than i and the baseline.

embedding a generalized matching penny game. Via a counting argument, one can prove that algorithms using sub-linear space simply cannot achieve ε-approximate minmax in all its states for most of the games in the family. This contradicts with the fact that no-regret dynamics achieve minmax value and therefore establishes the lower bound.

Compared with Srinivas, Woodruff, Xu & Zhou

Before we survey other related work, [146] is the most relevant to us. We discuss our findings in light of the main results therein.

Lower bound On the hardness side, [146] shows that $\Omega(S)$ memory is necessary for achieving $O(\sqrt{nT/S})$ regret even for i.i.d. loss sequence, whereas our lower bound indicates that $\Omega(S)$ is necessary for $O(T/S)$ regret against an *adaptive* adversary. These two results are incomparable in general, since our lower bound is quantitatively stronger but under a much stronger adversary model. On the technical side, the lower bounds of [146] leverage communication complexity techniques, whereas ours (theorem 5.1.3) exploits the connection between no-regret learning and zero-sum games.

Upper bound [146] gives an $O(S)$-space algorithm that obtains $\widetilde{O}(\sqrt{nT/S})$ regret assuming that the loss sequence arrives in random order. This matches their aforementioned lower bound. Unsurprisingly, the design and analysis of the algorithm hinges heavily upon the random order assumption and does not have any implication for the (standard) worst-case loss.

For worst-case loss sequence, [146] gives an $O(\frac{n}{\delta T})$-space algorithm with δT regret (for any $\delta T = \widetilde{\Omega}(\sqrt{T})$), assuming that the best expert receives a total loss of at most $M = O\left(\frac{\delta^2 T}{\log^2 n}\right)$. The assumption on the best expert is rather strong: for their result to be meaningful, one needs $\delta < 1$ and so the best expert already has sub-linear loss. As explained earlier, our theorem 5.1.1 does not require any such condition. In fact, under the assumption, the algorithmic design is conceptually simple. A naïve algorithm is to follow the advice of a single expert until its cumulative loss exceeds M, then switches to a new one, and repeats. This procedure uses $O(1)$ memory and is worse than the best expert in total loss by at most an $O(n)$ multiplicative factor. Instead of tracking a single expert, [146] designs a more general scheme by sampling a pool of experts and running majority vote. Improving upon the naïve idea, their algorithm achieves a performance of $O(\delta^{-1}\log^2 n)$ multiplicatively worse than the best expert (in terms of total loss). We stress again that our algorithm works beyond this low mistake regime.

Related work

Identifying best expert is hard We first mention that identifying even an approximately best expert requires $\Omega(n)$ memory in stream. This is matched by the trivial algorithm of tracking the cumulative loss of all experts. The proof is by a simple reduction from the well-studied SET-DISJOINTNESS problem; see the survey [143] and references therein. We refer interested reader to the prior work [146] for a detailed argument.

Prior work on expert learning Forms of the classic MWU algorithm for online learning can be dated back to 1950's [28]. The algorithm has been analyzed in a variety of settings and shown to achieve (nearly) optimal regret [119, 137]. The algorithm also finds a wide range of applications in algorithm design and optimization [33, 71, 44, 73, 104, 85, 4]. See the survey [13] for a complete treatment.

There are other online learning algorithm that are less computationally expensive than MWU. In particular, a line of work initiated by [95, 82] equips an online learner with an (offline) optimization oracle. The goal is to minimize oracle calls, a proxy for time complexity. Strong regret and oracle complexity guarantees can be achieved under this framework [66, 79, 22, 47]. Moreover, under various structural assumptions, one can improve upon the time complexity of MWU [83, 127, 152, 95]. These lines of work, however, generally do not consider space bounds. Finally, in terms of technique, the algorithm of [82] also uses the idea of random sub-sampling of the experts.

Memory-efficient online learning There has been a recent spate of work on memory-bounded online learning, mostly in (stochastic) multi-arm bandit settings. This includes the study of both regret minimization and pure exploration [118, 35, 15, 93, 128]. We mention that a recent work [3] considers a multi-pass setting, where the algorithm may take several passes over the data. They show that the regret-memory trade-off can be significantly improved when this is allowed. This chapter is focused on single-pass algorithm. Space usage is also considered by [100] in the analysis of pairwise losses in online learning, but under a restricted memory model.

A related line of work is on the memory-sample lower bounds for statistical and computational learning problems [151, 140, 141, 74, 142, 77, 75, 129], such as learning sparse parities and linear regression. Our problem is not statistical in nature, since we assume a worst-case loss sequence, and therefore does not lie in their setting. Other lines of work also study learning problems in data streams, including continual learning [36], entropy estimation [1, 7], detecting correlations [46], robust estimation [63], learning simple classifiers [26] and matrix rank estimation [45].

Adversary models Our algorithm assumes an oblivious adversary, which is standard in the literature. We remark that our lower bound considers a notion of adaptivity stronger than the typical ones in the literature, where the adversary cannot access the internal randomness of the algorithm; see, e.g., [49, 34]. Finally, several recent works show that certain lower bounds under adaptive adversary can be circumvented by smoothed analysis [139, 22, 78, 79].

5.2 Preliminaries

Notations Throughout the paper, we use T to denote the number of rounds and n to denote the number of experts. We write $[n] := \{1, \dots, n\}$ and $[n_1, n_2] := \{n_1, \dots, n_2\}$. Let Δ_n denote the collection of probability distributions over $[n]$. Unless specified otherwise, all logarithms are base e. We refer to a word of memory as $O(\log(nT))$ bits.

Online learning with expert advice

We study the classic online prediction with expert problem under the (standard) oblivious adversary model. In this problem, an algorithm makes prediction every round with the advice from experts and with the goal of minimizing its regret with respect to the best expert in hindsight. Formally,

Definition 5.2.1 (Online learning with oblivious adversary)**.** *An algorithm is initiated with memory M_1 and makes prediction for T rounds. At the t-th iteration ($t \in [T]$),*

- *The algorithm chooses an expert $i_t \in [n]$ based on its memory M_t*
- *The nature reveals the loss vector $\ell_t \in [0, 1]^n$,*
- *The algorithm receives loss $\ell_t(i_t) \in [0, 1]$ and updates its memory state M_{t+1}.*

An adversary is said to be oblivious if the sequence of loss vectors $\ell_1, \dots, \ell_T \in [0, 1]^n$ are chosen independent of the algorithm's decision. Equivalently, the nature fixes the loss vectors in advance (possibly randomly) and they are unknown to the algorithm. An algorithm is said to use up to M bits of space if $\max_{t\in[T]} |M_t| \le M$.

Remark 5.2.1. *Strictly speaking, the algorithm can not store the loss vector ℓ_t into its memory in the second step, as it already takes $\Omega(n)$ bits. Instead, we allow the algorithm to query the entry of the loss vector ℓ_t.*

The goal of the algorithm is to minimize the total *regret* over T rounds, defined as

$$R(T) := \mathbb{E}\left[\sum_{t\in[T]} \ell_t(i_t)\right] - \min_{i^*\in[n]} \sum_{t\in[T]} \ell_t(i^*), \tag{5.2.1}$$

where the expectation is taken over the randomness of the algorithm. We also consider the average regret, defined as $R(T)/T$.

We also consider the adaptive adversary model and prove a linear memory lower bound.

Definition 5.2.2 (Online learning with strong adaptive adversary)**.** *An algorithm is initiated with memory M_1 and makes prediction for T rounds. At the t-th iteration (for $t \in [T]$),*

- *The algorithm commits a distribution $p_t \in \Delta_n$ over the experts $[n]$ based on its memory state M_t;*
- *The adversary reveals the loss vector $\ell_t \in [0,1]^n$ after observing the distribution p_t;*
- *The algorithm receives loss $\langle p_t, \ell_t \rangle$ and updates its memory state M_{t+1}.*

Note we assume the algorithm commits a probability distribution over the experts $[n]$ (instead of a single expert) but the realization is unknown to the adversary. Equivalently, it can be seen that the adversary can choose loss vector ℓ_t based on the *decisions* as well as the *internal randomness* of the algorithm up to round $t-1$.

This notion of adaptivity here is stronger than the traditional one in the online learning literature, where the adversary does not have access to the algorithm's internal states. Rather, definition 5.2.2 closely resembles white-box adversary for adversarially robust streaming algorithm, proposed recently by [5]. We remark that the standard implementation of MWU takes $\widetilde{O}(n)$ space and achieves $O\left(\sqrt{T \log n}\right)$ regret against strong adaptive adversary.

Algorithmic and mathematical tools

Multiplicative weights update Our algorithm will use the classic MWU scheme as a subroutine. We state its update and decision rule, and the formal regret guarantees. See [13] for a standard exposition.

Algorithm 2: Multiplicative weight update (MWU)

```
Input: Learning rate $\eta$, expert $[n]$
1 Initialize $p_t \in \Delta_n$ to be the uniform distribution over $[n]$, the set of experts.
2 For $t$ from 1 to $T$
3     Compute $p_t \in \Delta_n$ over experts such that $p_t(i) \propto \exp\left(-\eta \sum_{\tau=1}^{t-1} \ell_\tau(i)\right)$
4     Sample an expert $i_t \sim p_t$ and observe the loss vector $\ell_t \in [0,1]^n$
```

Lemma 5.2.1 (MWU regret, [13]). *Suppose $n, T, \eta > 0$ and the loss $\ell_t \in [0,1]^n$ ($t \in [T]$), then the multiplicative weight update algorithm satisfies*

$$\sum_{t=1}^{T} \langle p_t, \ell_t \rangle - \min_{i^* \in [n]} \ell_t(i^*) \leq \frac{\log n}{\eta} + \eta T,$$

and with probability at least $1-\delta$,

$$\sum_{t=1}^{T} \ell_t(i_t) - \min_{i^* \in [n]} \ell_t(i^*) \leq \frac{\log n}{\eta} + \eta T + O\left(\sqrt{T \log(n/\delta)}\right).$$

Taking $\eta = \sqrt{\frac{\log n}{T}}$, the MWU algorithm has a total regret of $O\left(\sqrt{T \log(nT)}\right)$ with probability at least $1 - 1/\operatorname{poly}(T)$ and a standard implementation takes $O(n \log T)$ bits of memory.

Probability and concentration inequalities We state the standard concentration inequality.

Lemma 5.2.2 (Azuma-Hoeffding bound)**.** *Let $X_0, \ldots, X_n$ be a martingale with respect to the filter $F_0 \subseteq F_1 \cdots \subseteq F_n$ such that for $Y_i = X_i - X_{i-1}$, $i \in [n]$, we have that $|Y_i| = |X_i - X_{i-1}| \leq c_i$. Then*

$$\Pr[|X_t - Y_0| \geq t] \leq 2\exp\left(-\frac{t^2}{2\sum_{i=1}^n c_i^2}\right).$$

Minmax theorem It is well known that the equilibrium value of a zero sum game is unique, and it equals to the minmax or maxmin value.

Lemma 5.2.3 (Minmax Theorem [135])**.** *For any $A \in \mathbb{R}^{n\times n}$, the minmax theorem guarantees that*

$$\min_{x\in\Delta_n}\max_{y\in\Delta_n} x^\top Ay = \max_{x\in\Delta_n}\min_{y\in\Delta_n} x^\top Ay.$$

5.3 The building block

We first give an online learning algorithm that achieves $\widetilde{O}(\varepsilon T + TB^{-1/2} + \varepsilon^2 nB)$ total regret over T days in $S = \widetilde{O}(\varepsilon^{-2})$ space, where $B \ll T$ is a parameter specified later. This procedure apparently does not achieve our end goal, but instead it will serve as a building block for the our full algorithm in section 5.4.

Theorem 5.3.1. *Let T, n, B be positive integer, $\varepsilon \in (0, 1/2)$, there exists an online learning algorithm that achieves $O\left(\varepsilon T + TB^{-1/2}\log^{1/2}(nT) + \varepsilon^2 nB\log T\right)$ regret with probability at least $1 - 1/\operatorname{poly}(T)$ and uses $O(\varepsilon^{-2}\log^3(nT))$ bits of memory.*

For the sake of simplicity, we assume ε^{-1} and T/B are integers in the rest of section.

Baseline algorithm

Algorithm description The Baseline (Algorithm 3) maintains a pool of experts $\mathcal{P}$ at every step. It divides the whole sequence into T/B epochs, where each epoch consists of B (contiguous) days. Baseline proceeds epoch by epoch. A random set of experts R_t of size ε^{-2} is sampled uniformly without replacement from $[n]$ and enters the pool at the beginning of each epoch. For now, assume an expert's cumulative loss within each epoch is tracked and stored, ever since it joins the pool. Within an epoch, we maintain the same pool of experts and run the MWU algorithm only on them, starting with the uniform weights, and produces a (random) decision every round. Naïvely, the pool size grows by one every epoch, which is unacceptable for large T. To address the issue and bound the pool size, we evict experts at the very end of each epoch by comparing their average losses.

Intuition Intuitively, if any expert performs poorly relative to the others in the pool, it makes sense to evict it. However, care needs to be taken when comparing a long-surviving expert with a recently joined one. Due to the worst-case nature of the input, a new expert may start off by receiving significantly less loss than an old one. Nevertheless, it is yet unclear that it will continue to excel in the long run. Therefore, we design the algorithm so that an expert cannot be deleted simply because it is outperformed by a newer expert. It turns out that this rule is crucial in proving our memory bounds as well.

Eviction rule More formally, for any epoch $t \in [T/B]$, let $\mathcal{P}_t \subseteq [n]$ be the pool of experts at the beginning of t-th epoch (after adding R_t). The pool remains unchanged throughout the t-th epoch, and the algorithm considers removing certain experts from the pool at the end of the epoch. For experts in R_t, the rule is simple and we just keep the best expert. For each remaining expert i (including the one that survives in R_t), let $\alpha(t,i) \leq t$ be the epoch when expert i enters the pool. Let $\Gamma_{t,i} := [\alpha(t,i) : t]$ be the period of time from the $\alpha(t,i)$-th epoch to the t-th epoch. For simplicity we assume that Baseline explicitly tracks and updates the total loss value of all experts $i \in \mathcal{P}_t$, over each epoch since their entrance. (We will discuss later how to optimize the space usage.)

Let $\mathcal{L}_t(i) = \frac{1}{B}\sum_{b=1}^{B} \ell_{(t-1)B+b}(i)$ be the average loss for expert i in epoch t. For any time interval $\mathcal{I} \subseteq \Gamma_{t,i}$, let $\mathcal{L}_I(i) = \frac{1}{|I|}\sum_{t\in I} \mathcal{L}_t(i)$ be the average loss of expert i over I. The algorithm compares i with every other expert $j \in \mathcal{P}_t$. The expert i is evicted at the end of epoch t if and only if

(i) The expert j entered the pool $\mathcal{P}_t$ before the expert i; and

(ii) The average loss $\mathcal{L}_{\Gamma_{t,i}}(i)$ of i over $\Gamma_{t,i}$ is at least that of expert j up to an additive factor of ε:

$$\mathcal{L}_{\Gamma_{t,i}}(i) \geq \mathcal{L}_{\Gamma_{t,i}}(j) - \varepsilon. \tag{5.3.1}$$

Simply put, condition (i) ensures that an older expert in the pool cannot be kicked out due to a younger one. Notice that (ii) effectively requires (i), since our algorithm only keeps track of the loss of the experts within the pool. If j entered the pool later than i, we cannot even compute the right-hand side of (5.3.1).

We run the comparison-based pruning procedure at the very end of each epoch and use $\widetilde{P}_t$ to denote the set of experts that survive pruning. We will argue that (1) the size of the pool is bounded and (2) overall the pool contains good experts such that our algorithm is competitive against the best expert, albeit it can easily be outside the pool.

Algorithm 3: Baseline expert learning algorithm (BASELINE)

```
   Input: Parameter $T, B$ and $\varepsilon$, experts $[n]$
1  For each epoch $t = 1, 2, \ldots, T/B$
2      Sample a random set $R$ of $\varepsilon^{-2}$ experts without replacement from $[n]$ and add
        them to $\mathcal{P}$
3      Initialize MWU (with uniform weights) over experts in $\mathcal{P}$
4      For each day $b = 1, 2, \ldots, B$
5          Sample and play the MWU decision over experts in $\mathcal{P}$
       /* Evict expert                                                    */
6      Remove all except the best expert from $R$
7      For every pair $\{i, j\} \in \mathcal{P}$
8          Remove $i$ from $\mathcal{P}$ if $j$ entered $\mathcal{P}$ before $i$ and condition (5.3.1) holds
```

Implementation details Naïvely, the algorithm stores the cumulative loss of each expert in the pool, over every epoch since it entered. This may cause large memory usage. However, observe that, to execute the eviction rule, it is only required that for each j, we have the data $\mathcal{L}_{\Gamma_{i,t}}(j)$ for all i that entered later than j, in addition to its own cumulative loss. Therefore, the algorithm can explicitly track these values only; and if the pool size is S, then this takes $O(S^2)$ words of memory. See fig. 5.1 for an illustration.

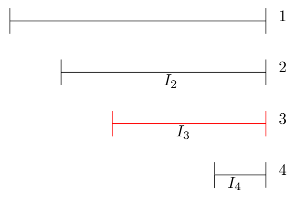

Figure 5.1: Pool at the end of an epoch. If expert 3 is removed, then we no longer store $\mathcal{L}_{I_3}(1), \mathcal{L}_{I_3}(2)$.

Analysis of the Baseline Algorithm

We now provide a formal analysis of BASELINE and argue its memory and regret guarantees.

Memory bound

The algorithm does not dictate an explicit bound on the memory, and in particular, on the size of pool. First, we present a key technical lemma that insists a *loss vs. length* structure on the surviving experts $\widetilde{\mathcal{P}}_t$. For a fixed t and any $i, j \in \widetilde{\mathcal{P}}_t$, write $i \prec j$ if $\alpha(t,i) > \alpha(t,j)$—namely, i joined the pool later than j, so j is older—and $i \succ j$ otherwise. Note that expert i and j must join the pool at different time due to our eviction rule. Roughly speaking, the lemma states that if i survives the pruning (after Line 10 of **??** 3), then for any $j \succ i$, one of the following must happen:

(i) expert j suffers significantly more average loss over $\Gamma_{t,j}$ than i suffers overs its interval $\Gamma_{t,i}$; or

(ii) expert j has resided in the pool significantly longer than i.

For notational convenience, for any $i, j \in \widetilde{\mathcal{P}}_t$ such that $j \succ i$, define $L_{i,i} = \mathcal{L}_{\Gamma_{t,i}}(i)$, $L_{i,j} = \mathcal{L}_{\Gamma_{t,i}}(j)$ and accordingly $L_{j,j} = \mathcal{L}_{\Gamma_{t,j}}(j)$. Then we have

Lemma 5.3.1 (loss vs. length)**.** *For any epoch $t \in [T/B]$, suppose experts $i, j \in \widetilde{\mathcal{P}}_t$ and $j \succ i$. Let $\alpha \in (0,1)$, then at least one of the following must hold:*

(i) $L_{j,j} \geq L_{i,i} + \varepsilon - \alpha$;

(ii) $|\Gamma_{t,j}| \geq \left(1 + \frac{\alpha}{1-\alpha}\right) |\Gamma_{t,i}|$.

The underlying intuition is simple: Since i survives the pruning and j joins the pool earlier than i, we know that j achieves ε worse average performance than i over i's own interval. Now consider the opposite of condition (i)—it asks j to be at most $\varepsilon - \alpha$ worse average loss overall than i. For that to happen, j needs to have low error on the days before i enters the pool. However, even if j gets 0 loss over this prior period, it still requires time to bring the average loss down by at least ε. Therefore, j must have entered the pool quite earlier than i, which is condition (ii).

Proof of lemma 5.3.1. By definition of the algorithm, any such expert i must have survived the comparison-based pruning against $j \succ i$. Therefore, the condition (5.3.1) must fail for i, j, and so we have

$$L_{i,j} > L_{i,i} + \varepsilon. \tag{5.3.2}$$

Now fix $\alpha \in (0,1)$ and let's assume that $L_{j,j} < L_{i,i} + \varepsilon - \alpha$. It suffices to show that $|\Gamma_{t,j}| \geq (1 + \frac{\alpha}{1-\alpha})|\Gamma_{t,i}|$. Let $T_1 = |\Gamma_{t,j}| - |\Gamma_{t,i}| > 0$ be the number of extra days j lies in the pool than i, and $T_2 = |\Gamma_{t,i}|$. Also let $L_1 = \mathcal{L}_{\Gamma_{t,j} \setminus \Gamma_{t,i}}(j)$ be the average loss of j over the T_1 days. Simply rewriting the assumption that $L_{j,j} < L_{i,i} + \varepsilon - \alpha$:

$$L_{j,j} = \frac{L_1 T_1 + L_{i,j} T_2}{T_1 + T_2} < L_{i,i} + \varepsilon - \alpha. \tag{5.3.3}$$

Since $L_1 \geq 0$, we get

$$\frac{L_{i,j} T_2}{T_1 + T_2} < L_{i,i} + \varepsilon - \alpha. \tag{5.3.4}$$

Substituting the inequality $L_{i,i} < L_{i,j} - \varepsilon$ (5.3.2) to the right-side:

$$\frac{L_{i,j}T_2}{T_1 + T_2} < L_{i,j} - \alpha \tag{5.3.5}$$

Rearranging, we get

$$T_1 > \left(\frac{L_{i,j}}{L_{i,j} - \alpha} - 1\right) T_2 = \frac{\alpha}{L_{i,j} - \alpha} T_2 > \frac{\alpha}{1 - \alpha} T_2, \tag{5.3.6}$$

where the last inequality follows since $L_{i,j} \leq 1$. Equivalently, we have that $|\Gamma_{t,j}| - |\Gamma_{t,i}| > \frac{\alpha}{1-\alpha}|\Gamma_{t,i}|$, and this completes the proof. □

Using the above lemma, we can bound the size of the pool via a potential function argument. The potential takes both situations of lemma 5.3.1 into account, where either length or loss gets larger.

Lemma 5.3.2 (pool size). *For any epoch $t \in [T/B]$, the size of the pool $\widetilde{P}_t$ is at most $S = \frac{4}{\varepsilon}\log T$.*

Proof. Fix t and let S be the size of the pool. We sort the experts in the pool in ascending order of their entering times: $i_1 \prec i_2 \prec \cdots \prec i_S$. Define the potential function $\Phi : [S] \to \mathbb{R}_{\geq 0}$, where

$$\Phi(\tau) = 2\log|\Gamma_{t,i_\tau}| + L_{i_\tau,i_\tau}, \quad \tau \in [S]. \tag{5.3.7}$$

We note that $\Phi(1) \geq 0$ and $\Phi(S) \leq 2\log T + 1$. The goal is to prove

$$\Phi(\tau + 1) - \Phi(\tau) \geq \varepsilon, \quad \forall \tau \in [S - 1]. \tag{5.3.8}$$

This would imply the pool size $S \leq 4\varepsilon^{-1}\log T$.

We observe that eq. (5.3.8) is simply true whenever $L_{i_{\tau+1},i_{\tau+1}} \geq L_{i_\tau,i_\tau} + \varepsilon$ since $i_{\tau+1}$ enters the pool before i_τ by our assumption. Now, it is safe to write $L_{i_{\tau+1},i_{\tau+1}} = L_{i_\tau,i_\tau} + \varepsilon - \alpha$, for some $\alpha \in (0, 1)$ (α would not exceed 1 as one can easily show $L_{i_\tau,i_\tau} + \varepsilon < 1$, i.e., i_τ survives at $\widetilde{P}_t$). Then we have

$$\begin{aligned}
\Phi(\tau + 1) - \Phi(\tau) &= 2\log(|\Gamma_{t,i_{\tau+1}}|/|\Gamma_{t,i_\tau}|) + L_{i_{\tau+1},i_{\tau+1}} - L_{i_\tau,i_\tau} \\
&= 2\log\left(1 + \frac{\alpha}{1-\alpha}\right) + \varepsilon - \alpha \\
&\geq \min\left\{\frac{\alpha}{1-\alpha}, 2\log 2\right\} + \varepsilon - \alpha \geq \varepsilon,
\end{aligned}$$

where the second step follows from lemma 5.3.1, the third step follows from $\log(1 + x) \geq \frac{x}{2}$ whenever $x < 1$ and the last step holds since $\alpha \in (0, 1)$. We have proven eq. (5.3.8) and completed the proof. □

We can now wrap up with a memory bound.

Proposition 5.3.1 (memory bound)**.** *At any time during the execution of* Baseline*, the memory usage is at most* $O\left(\varepsilon^{-2}\log^3(nT)\right)$ *bits.*

Proof. Fix an epoch t, as we observed earlier, for each expert $j \in \widetilde{\mathcal{P}}_{t-1}$, the algorithm only needs to keep track of their loss over the intervals $\Gamma_{t-1,i}$ for all $i \in \mathcal{P}_{t-1}$ such that $i \prec j$. In particular, for each $j \in \mathcal{P}_{t-1}$, the algorithm stores $\mathcal{L}_{\Gamma_{t-1,i}}(j)$ for all $i \prec j$. This is sufficient for executing the eviction rule (eq. (5.3.1)). Storing each $\mathcal{L}_{\Gamma_{t-1,i}}(j)$ takes $O(\log nT)$ bits. By lemma 5.3.2, there are at most $S = \frac{4}{\varepsilon}\log T$ experts in $\mathcal{P}_t$. This leads to $S^2 \cdot O(\log nT) = O(\varepsilon^{-2}\log^3(nT))$ bits of memory usage. We sample $|R_t| = \varepsilon^{-2}$ new experts at the beginning of t-th epoch and keep track of them within the epoch, these takes $O(\varepsilon^{-2}\log nT)$ extra bits. □

Regret bound

We now prove a regret bound of Baseline. On a high level, the MWU procedure only guarantees that Baseline is always competitive with the best expert *in the pool*. The key challenge here therefore is to argue that our pool is competitive against even the best expert among $[n]$, which may lie outside the pool. Indeed, next we will prove that the the experts in the pool is nearly competitive against the best expert, except in $\widetilde{O}(\varepsilon^2 n)$ epochs. Formally, we show:

Proposition 5.3.2 (regret bound)**.** *Given the parameter* $\varepsilon \in (0, 1/2)$ *and positive integer* $B \ll T$, Baseline *achieves a total regret of* $O\left(\varepsilon T + TB^{-1/2}\log^{1/2}(nT) + \varepsilon^2 nB\log T\right)$ *with probability at least* $1 - 1/\operatorname{poly}(T)$.

Let i^* denote the best expert in hindsight. Since the adversary is oblivious to the algorithm's decision, it suffices to fix a loss sequence $\ell_1, \ell_2, \ldots, \ell_T \in [0,1]^n$ and prove the algorithm achieves low regret on it. Let ξ_t denote the random bits used by Baseline during the t-th epoch, which includes both the randomness of sampling R_t and the random bits used by MWU within the epoch.

Initialize $\mathcal{B} = \emptyset$ and $i(t) = \mathsf{nil}$ for each $t \in [T/B]$. We will build up the set $\mathcal{B} \subseteq [T/B]$ over time. Intuitively, it contains "unlucky" epochs that we have no control over the regret. On the other hand, for any lucky epoch $t \in [T/B]\backslash\mathcal{B}$, we would associate the t-th epoch with an expert $i(t) \in [n]$ that (1) is competitive with i^*, and (2) lies in the pool $\mathcal{P}_t$. Formally, the value of $\mathcal{B}$ and $\{i(t)\}_{t\in[T/B]}$ are assigned by the following stochastic process.

Stochastic process Starting with $\beta(1) = 1$ and $\tau = 1$, define a (discrete) stochastic process by realizing the randomness $\xi_1, \ldots, \xi_{T/B}$ epoch by epoch. Suppose the process proceeds to step τ and the randomness $\xi_1, \ldots, \xi_{\beta(\tau)-1}$ are realized up to the $(\beta(\tau)-1)$-th epoch. Then the pool $\widetilde{\mathcal{P}}_{\beta(\tau)-1}$ is also fixed by definition. Let $t(i^*, \tau)$ denote the epoch when i^* gets evicted, conditioned on $i^* \in R_{\beta(\tau)}$ due to the randomness $\xi_{\beta(\tau)}$ and that it survives the competition among $R_{\beta(\tau)}$ (Line 7 of **??** 3). (If i^* never gets evicted, then we set $t(i^*, \tau) = \infty$.) The (conditional) eviction time $t(i^*, \tau)$ of expert i^* is determined by the fixed pool $\widetilde{\mathcal{P}}_{\beta(\tau)-1}$ and the loss sequence $\ell_1, \dots, \ell_T$. In other words, we observe that $t(i^*, \tau) \in [T/B]$ is only a function of $\ell_1, \dots, \ell_T$ and $\widetilde{\mathcal{P}}_{\beta(\tau)-1}$: it is independent of $\xi_{\beta(\tau)+1}, \dots, \xi_{T/B}$ because i^* enters at epoch $\beta(\tau)$ and can only be kicked out by experts joining before it, i.e., the experts in $\widetilde{\mathcal{P}}_{\beta(\tau)-1}$; and it is independent of $\xi_{\beta(\tau)}$ because we already condition on the event of i^* surviving among $R_{\beta(\tau)}$.

We now continue to define the stochastic process and consider the following cases.

- Case 1. Suppose $t(i^*, \tau) \neq \infty$, i.e., expert i^* would be evicted at the end of epoch $t(i^*, \tau) \in [\beta(\tau), T/B]$. Suppose it is removed by expert i^*_τ. Then we set $\beta(\tau+1) = t(i^*, \tau) + 1$ and assign $i(\beta(\tau)) = \cdots = i(\beta(\tau+1) - 1) = i^*_\tau$. We then realize the randomness $\xi_{\beta(\tau)}, \dots, \xi_{\beta(\tau+1)-1}$ and proceed to step $\tau + 1$.
- Case 2. Suppose $t(i^*, \tau) = \infty$, i.e., expert i^* would not be kicked out of the pool once it is sampled in $R_{\beta(\tau)}$ and survives the competition among $R_{\beta(\tau)}$. We then realize the randomness of $\xi_{\beta(\tau)}$ and further divide into two cases based on it.
  - Case 2-1. If $R_{\beta(\tau)}$ contains an expert $i^*_\tau \in [n]$ that receives less loss than i^* during epoch $\beta(\tau)$, then we set $\beta(\tau+1) = \beta(\tau) + 1$ and assign $i(\beta(\tau)) = i^*_\tau$. We proceed to step $\tau + 1$.
  - Case 2-2. Suppose $R_{\beta(\tau)}$ does not contain any expert that gets less loss than i^* during epoch $\beta(\tau)$, then
    * Case 2-2-1. If expert i^* has been sampled, i.e., $i^* \in R_{\beta(\tau)}$, then assign $i(\beta(\tau)) = \cdots = i(T/B) = i^*$ and terminate the process. We note that by the definition of $t(i^*, \tau)$ and the condition, i^* will survive till the end.
    * Case 2-2-2. If expert i^* has not been sampled, i.e., $i^* \notin R_{\beta(\tau)}$, then add $\beta(\tau)$ into $\mathcal{B}$ and set $\beta(\tau+1) = \beta(\tau) + 1$. We proceed to step $\tau + 1$.

Having defined the stochastic process, we proceed with our regret analysis. The following two lemmas are critical to the proof. First, we show the size of $\mathcal{B}$ is small with high probability:

Lemma 5.3.3 (unlucky epoch). *With probability at least* $1 - 1/\operatorname{poly}(T)$*, the stochastic process ends with* $|\mathcal{B}| \leq O(\varepsilon^2 n \log T)$*.*

Proof. We count the total number of steps that the stochastic process come with Case 2-2. Whenever the process falls into Case 2-2, i.e., at some step τ, $R_{\beta(\tau)}$ does not contain any expert better than i^* during epoch $\beta(\tau)$, we know that

$$\Pr\left[i^* \in R_{\beta(\tau)} | R_{\beta(\tau)} \text{ has no expert better than } i^* \text{ during epoch } \beta(\tau)\right] \geq \frac{1}{n} \cdot \varepsilon^{-2}, \quad (5.3.9)$$

since there are ε^{-2} experts sampled without replacement from $[n]$. Consider the following two cases: (1) if $i^* \in R_{\beta(\tau)}$, then the process would terminate immediately and there will be no more Case 2-2 in the future. This situation happens with probability at least $\frac{1}{n} \cdot \varepsilon^{-2}$ by eq. (5.3.9); and (2) if $i^* \notin R_{\beta(\tau)}$, then the process would still continue. Since we do not augment $\mathcal{B}$ in Case 1 and Case 2-1, the size of $\mathcal{B}$ is bounded by total number of steps of Case 2-2, and we have,

$$\Pr\left[|\mathcal{B}| \geq c\varepsilon^2 n \log T\right] \leq \left(1 - 1/\varepsilon^2 n\right)^{c\varepsilon^2 n \log T} \leq T^{-c}.$$

This concludes the proof. □

We then prove for epoch $t \in [T]$, the expert $i(t)$ resides in pool $\mathcal{P}_t$ and is competitive with i^* (over certain time period).

Lemma 5.3.4 (cover)**.** *For any epoch $t \in [T/B] \backslash \mathcal{B}$, suppose $\beta(\tau) \leq t < \beta(\tau+1)$, then we have*

(i) $i(\beta(\tau)) = \cdots = i(t) = \cdots = i(\beta(\tau+1)-1) = i_\tau^* \neq \mathsf{nil}$;

(ii) $i_\tau^* \in \mathcal{P}_\nu$ *for any* $\nu \in [\beta(\tau) : \beta(\tau+1)-1]$;

(iii) $\sum_{\nu=\beta(\tau)}^{\beta(\tau+1)-1} \mathcal{L}_\nu(i(t)) < \sum_{\nu=\beta(\tau)}^{\beta(\tau+1)-1} \mathcal{L}_\nu(i^*) + \varepsilon(\beta(\tau+1) - \beta(\tau))$.

Proof. The first claim is straightforward from the assignment process. Since $t \in [\beta(\tau) : \beta(\tau+1)-1]$ and $t \notin \mathcal{B}$, we note in the $\beta(\tau)$-th epoch, the process falls into Case 1, Case 2-1 or Case 2-2-1.

In Case 1. If i^* is sampled in the $R_{\beta(\tau)}$ and happens to survive the comparison among R_t, then i^* wound be evicted at the end of $(\beta(\tau+1)-1)$-th epoch when comparing with expert i_τ^*. First of all, this indicates that i_τ^* enters the pool before epoch $\beta(\tau)$ (an expert can only be evicted by older expert), and therefore, the eviction time of i_τ^* is already determined given $\xi_1, \ldots, \xi_{\beta(\tau)-1}$ and $\ell_1, \ldots, \ell_{T/B}$ and it is no earlier than $\beta(\tau+1)-1$ because otherwise, it could not kick i^* out. The third claims follows directly from the eviction rule (see eq. (5.3.1)).

In Case 2-1, we know that $\beta(\tau) = t$, $\beta(\tau+1) = t+1$ and $i_\tau^* \in R_t$ is an expert that has better performance than i^*. The second and last claims are then straightforward.

Finally, in Case 2-2-1, we note that $i_\tau^* = i^*$ and i^* survives till the end. The last two claims are straightforward and we finish the proof. □

Now we can finish the proof of proposition 5.3.2.

Proof of proposition 5.3.2. Conditioned on the high probability event of lemma 5.3.3, the total regret of Baseline is controlled as follows:

$$
\begin{aligned}
&\sum_{t=1}^{T/B}\sum_{b=1}^{B}\ell_{(t-1)B+b}\left(i_{(t-1)B+b}\right)-\ell_{(t-1)B+b}(i^*)\\
=&\sum_{t\in[T/B]\backslash\mathcal{B}}\sum_{b=1}^{B}\left(\ell_{(t-1)B+b}\left(i_{(t-1)B+b}\right)-\ell_{(t-1)B+b}(i^*)\right)\\
&\quad+\sum_{t\in\mathcal{B}}\sum_{b=1}^{B}\ell_{(t-1)B+b}\left(i_{(t-1)B+b}\right)-\ell_{(t-1)B+b}(i^*)\\
\le&\sum_{t\in[T/B]\backslash\mathcal{B}}\sum_{b=1}^{B}(\ell_{(t-1)B+b}(i_{(t-1)B+b})-\ell_{(t-1)B+b}(i^*))+O(\varepsilon^2 n\log T)\cdot B\\
\le&\sum_{t\in[T/B]\backslash\mathcal{B}}B(\mathcal{L}_t(i(t))-\mathcal{L}_t(i^*))+\frac{T}{B}\cdot O\left(\sqrt{B\log(nT)}\right)+O\left(\varepsilon^2 nB\log T\right)\\
=&\sum_{\tau}\sum_{t\in[\beta(\tau):\beta(\tau+1)-1],t\notin\mathcal{B}}B(\mathcal{L}_t(i(t))-\mathcal{L}_t(i^*))+O\left(TB^{-1/2}\log^{1/2}(nT)\right)+O\left(\varepsilon^2 nB\log T\right)\\
\le&\sum_{\tau}\varepsilon B(\beta(\tau+1)-\beta(\tau))+O\left(TB^{-1/2}\log^{1/2}(nT)\right)+O\left(\varepsilon^2 nB\log T\right)\\
=&\,O\left(\varepsilon T+TB^{-1/2}\log^{1/2}(nT)+\varepsilon^2 nB\log T\right).
\end{aligned}
$$

We split the regret based on whether t belongs to $\mathcal{B}$ in the first step. The second step follows from $|\mathcal{B}|\le O(\varepsilon^2 n\log T)$ (lemma 5.3.3) and $\ell_t\in[0,1]^n$. The third step follows from the guarantee of MWU (lemma 5.2.1) and the fact that the expert $i(t)$ is contained in the pool $\mathcal{P}_t$ (second claim of lemma 5.3.4). We split $[T/B]$ according to $\beta(\tau)$ in the fourth step and make use of the first claim of lemma 5.3.4. The fifth step follows from the third claim of lemma 5.3.4. We conclude the regret analysis. □

Combining proposition 5.3.1 and proposition 5.3.2, we conclude the proof of theorem 5.3.1. Moreover, balancing the last two regret terms by taking $B=(T/\varepsilon^2 n)^{2/3}$, we get:

Corollary 5.3.1. *Let T,n be positive integer, $\varepsilon\in(0,1/2)$, $T=\Omega(\varepsilon^2 n)$, there exists an online learning algorithm that achieves $\widetilde{O}(\varepsilon T+T^{2/3}(\varepsilon^2 n)^{1/3})$ regret with probability at least $1-1/\operatorname{poly}(T)$ and uses $O(\varepsilon^{-2}\log^3(nT))$ bits of memory.*

5.4 Full algorithm and analysis

Building upon Baseline, we can state our main result.

Theorem 5.4.1 (Main algorithmic result)**.** *Let T, n be positive integers and $\delta \in (0,1]$. There exists an online learning algorithm that achieves a total regret of $\widetilde{O}\left(n^2 T^{\frac{2}{2+\delta}}\right)$ with probability at least $1 - \mathrm{poly}(T)$ and uses $O\left(n^{\delta}\log^4(nT)\right)$ bits of memory.*

Full Algorithm

Parameters Let $\varepsilon = n^{-\delta/2}$ and $T = n(n/\varepsilon)^K$. For simplicity, we assume ε^{-1}, εn and K are positive integers for now. Define $T_k = n(n/\varepsilon)^k$ for each $k \in [K]$, and the epoch length is fixed as $B = 1/\varepsilon^2$.

Algorithm description The FullAlgo (pseudocode in ?? 6) aggregates Baseline$_+$ (?? 4) by levels. We first describe the algorithm Baseline$_+$, which takes in a level parameter k. At the bottom level ($k = 1$), the algorithm repeatedly runs Baseline for $T/T_1 = (n/\varepsilon)^{K-1}$ episodes. Within each episode, the algorithm starts with a fresh run of Baseline and continues for $T_1 = n^2/\varepsilon$ days. The T_1 days are split into εn^2 epochs, and each epoch contains $B = 1/\varepsilon^2$ days. We will later see that this guarantees that each n^2/ε days, the algorithm gets a total regret of $O(n^2 \log(nT))$ compared with the best expert.

Baseline$_+$ differs significantly from Baseline starting from the second level ($k \geq 2$). Instead of potentially playing a different decision every day, Baseline$_+$ joins every $T_{k-1} = n(n/\varepsilon)^{k-1}$ consecutive days as one *decision day* and plays the same decision on all of them. There are T/T_k episodes and n/ε decision days within each episode. Again, they are divided into εn epochs with $1/\varepsilon^2$ days each epoch. The algorithm restarts every episode.

The key point is that instead of directly following the advice of expert i, the algorithm follows from the combination of expert i and Baseline$_+(k-1)$. In particular, MergeExp (pseudocode in ?? 5) takes an expert i and Baseline$_+(k-1)$ and runs MWU over them. We take it as the new expert $e_{k,i}$ for level k. The advantage is that the loss of $e_{k,i}$ is roughly the minimum of Baseline$_+(k-1)$ and i (by the regret guarantee of MWU) and thus has small width. This motivates the modified eviction rule.

Eviction rule We reload the notations from section 5.3 and introduce a few more. For any level $k \in [K]$, episode $r \in [T/T_k]$, epoch $t \in [\varepsilon n]$, let $P_{k,r,t} \subseteq \{e_{k,i}\}_{i\in[n]}$ be the pool of experts at the beginning of t-th epoch (after adding $R_{k,r,t}$) and the pool remains unchanged during the t-th epoch. Let $T_{k,r,t,b} = (r-1)T_k + (t-1)BT_{k-1} + (b-1)T_{k-1}$. For each decision day $b \in [B]$, we update the MWU and the cumulative loss according to the *truncated loss*

$$\widehat{\mathcal{L}}_{k,r,t,b}(e_{k,i}) = \max\left\{\mathcal{L}_{k,r,t,b}(e_{k,i}) - \mathcal{L}_{k,r,t,b}\left(\textsc{Baseline}_+(k-1)\right), -\varepsilon^{k-1}\log^{2k-1}(nT)\right\} \quad (5.4.1)$$

where

$$\mathcal{L}_{k,r,t,b}(e_{k,i}) = \frac{1}{T_{k-1}}\sum_{\tau=1}^{T_{k-1}} \ell_{T_{k,r,t,b}+\tau}(e_{k,i}) \quad (5.4.2)$$

is the average loss on the b-th decision day, Note that without the truncation, $\widehat{\mathcal{L}}_{k,r,t,b}$ would simply be a shift of $\mathcal{L}_{k,r,t,b} \in [0,1]^n$. Looking ahead, the truncation guarantees the width of $\widehat{\mathcal{L}}_{k,r,t,b}$ to be $2\varepsilon^{k-1}\log^{2k-1}(nT)$, since it is possible that $e_{k,i}$ performs much better than expert i and Baseline$_+(k-1)$.

Now we can state the eviction rule as follows. For experts in $R_{k,r,t}$, we just keep the best expert. For each remaining expert $e_{k,i}$ (include the one that survives in $R_{k,r,t}$), let $\Gamma_{k,r,t,i}$ be the period of time that $e_{k,i}$ resides in the pool. For any time interval $I \subseteq \Gamma_{k,r,t,i}$, recall that $\mathcal{L}_I(e_{k,i}) = \frac{1}{|I|}\sum_{t\in I}\mathcal{L}_t(e_{k,i})$ is the average loss of expert j over I. Let $\widehat{\mathcal{L}}_I(e_{k,i}) = \frac{1}{|I|}\sum_{t\in I}\widehat{\mathcal{L}}_t(e_{k,i})$ be the cumulative truncated loss defined similarly. The algorithm compares $e_{k,i}$ with every other expert $e_{k,j} \in \mathcal{P}_{k,r,t}$, and the expert $e_{k,i}$ is evicted at the end of epoch t if and only if

(i) The expert $e_{k,j}$ entered the pool $\mathcal{P}_{k,r,t}$ before the expert $e_{k,i}$; and

(ii) The average loss $\widehat{\mathcal{L}}_{\Gamma_{k,r,t,i}}(e_{k,i})$ of $e_{k,i}$ over $\Gamma_{k,r,t,i}$ is at least that of expert $e_{k,j}$ up to an additive factor of $\varepsilon^k\log^{2k-1}(nT)$:

$$\widehat{\mathcal{L}}_{\Gamma_{k,r,t,i}}(e_{k,i}) \geq \widehat{\mathcal{L}}_{\Gamma_{k,r,t,i}}(e_{k,j}) - \varepsilon^k\log^{2k-1}(nT). \tag{5.4.3}$$

In summary, Baseline$_+(k)$ (for $k \in [K]$) differs from Baseline in three ways:

- Baseline$_+(k)$ restarts every T_k days, and within each episode, it regards T_{k-1} days as one decision day;
- Baseline$_+(k)$ follows the decision of MergeExp instead of directly using the original experts $[n]$, and crucially it considers the truncated loss for eviction and MWU update;
- The eviction threshold changes from ε to $\varepsilon^k\log^{2k-1}(nT)$.

Finally, we note that FullAlgo outputs the decision of Baseline$_+(K)$. The pseudocode of these procedures are given below.

Algorithm 4: BASELINE$_+$

```
Input: Parameter $k$
1  For each episode $r = 1, 2, \ldots, T/T_k$
       /* if $k = 1$, then loop from 1 to $\varepsilon n^2$ */
2      For each epoch $t = 1, 2, \ldots, \varepsilon n$
3          For all $i \in [n]$, let $e_{k,i} = \textsc{MergeExp}(k, i)$.
4          Sample a random set $R$ of $\varepsilon^{-2}$ experts without replacement from $\{e_{k,i}\}_{i\in[n]}$,
             add them to $\mathcal{P}$.
5          For each decision day $b = 1, 2, \ldots, B$
6              Compute $p \propto \exp\left(-\eta \sum_{\tau=1}^{b-1} \widehat{\mathcal{L}}_{k,r,t,\tau}(e_{k,i})\right)$ for $e_{k,i} \in \mathcal{P}$
7              Sample an expert $i_{k,r,t,b} \sim p$ and follow $e_{k,i_{k,r,t,b}}$ for $T_{k-1}$ days
8          Remove all except the best expert from $R$
9          For every pair $\{e_{k,i}, e_{k,j}\} \in \mathcal{P}$
10             Remove $e_{k,i}$ from $\mathcal{P}$ if $e_{k,j}$ entered $\mathcal{P}$ before $e_{k,i}$ and condition (5.4.3) holds
11     Clear the pool $\mathcal{P}$ and restart
```

Algorithm 5: Merge expert (MERGEEXP)

```
Input: Parameter $k$, expert $i$
Output: Expert $e_{k,i}$
1  For each episode $r = 1, 2, \ldots, T/T_{k-1}$
2      Initiate with uniform weight over expert $i$ and BASELINE$_+(k-1)$
3      For $t = 1, 2, \ldots, T_{k-1}$
4          Run MWU over expert $i$ and BASELINE$_+(k-1)$, and play the decision
```

Algorithm 6: Full expert learning algorithm (FULLALGO)

```
Input: Parameter $T$, $\varepsilon$, experts $[n]$
Maintain BASELINE$_+(k)$ (for each $k \in [K]$) and play the decision of BASELINE$_+(K)$
```

Analysis of FullAlgo

We provide a formal analysis of FULLALGO and prove its memory and regret guarantees.

Regret analysis

We start with the regret analysis first, since the memory analysis depends on it. Formally, we aim to show:

Proposition 5.4.1 (regret bound)**.** *For any level $k \in [K]$ and episode $r \in [T/T_k]$, the* BASELINE$_+(k)$ *has a total regret of $O\left(n^{k+1}\log^{2k}(nT)\right)$ with probability at least $1-1/\operatorname{poly}(T)$.*

We prove the claim by an induction on k. The case $k = 1$ follows directly from proposition 5.3.2 by taking $T = T_1 = n^2/\varepsilon$ and $B = 1/\varepsilon^2$. Suppose the induction holds up to level

$k-1$ ($k \geq 2$), i.e.,

$$\mathcal{L}_{r,k,t,b}(\textsc{Baseline}_{+}(k-1)) - \mathcal{L}_{r,k,t,b}(i) \leq \varepsilon^{k-1}\log^{2k-2}(nT). \tag{5.4.4}$$

We proceed for level k and prove the claim for any episode $r \in [T/T_k]$.

We first state some basic properties on $\mathcal{L}_{k,r,t,b}(e_{k,i})$ and $\widehat{\mathcal{L}}_{k,r,t,b}(e_{k,i})$. The first claim states $e_{k,i}$ is relatively good on each decision day (since it runs MWU over expert i and $\textsc{Baseline}_{+}(k-1)$), and the second claim states $\widehat{\mathcal{L}}_{k,r,t,b}(e_{k,i})$ has small width.

Lemma 5.4.1. *For epoch $t \in [\varepsilon n]$ and decision day $b \in [B]$, with probability at least $1 - 1/\operatorname{poly}(T)$, one has*

- $\mathcal{L}_{k,r,t,b}(e_{k,i}) \leq \min\{\mathcal{L}_{k,r,t,b}(i), \mathcal{L}_{k,r,t,b}(\textsc{Baseline}_{+}(k-1))\} + O\left(\sqrt{\log(nT)/T_{k-1}}\right)$*; and*
- $\widehat{\mathcal{L}}_{k,r,t,b}(e_{k,i}) \in \left[-\varepsilon^{k-1}\log^{2k-1}(nT), \varepsilon^{k-1}\log^{2k-1}(nT)\right]$.

Proof. The first claim follows directly from the MWU guarantee of MergeExpert, in particular, with probability at least $1-1/\operatorname{poly}(T)$,

$$\mathcal{L}_{k,r,t,b}(e_{k,i}) \leq \min\{\mathcal{L}_{k,r,t,b}(i), \mathcal{L}_{k,r,t,b}(\textsc{Baseline}_{+}(k-1))\} + O\left(\sqrt{\log(nT)/T_{k-1}}\right). \tag{5.4.5}$$

For the second claim, we have

$$\begin{aligned}\mathcal{L}_{k,r,t,b}(e_{k,i}) - \mathcal{L}_{k,r,t,b}(\textsc{Baseline}_{+}(k-1)) &\leq \mathcal{L}_{k,r,t,b}(i) - \mathcal{L}_{k,r,t,b}(\textsc{Baseline}_{+}(k-1)) \\ &\quad + O\left(\sqrt{\log(nT)/T_{k-1}}\right) \\ &\leq 2\varepsilon^{k-1}\log^{2k-2}(nT).\end{aligned}$$

where the first step follows from eq. (5.4.5), the second step holds due to induction hypothesis (eq. (5.4.4)) and $T_{k-1} = n(n/\varepsilon)^{k-1}$. Therefore,

$$\begin{aligned}\widehat{\mathcal{L}}_{k,r,t,b}(e_{k,i}) &= \max\left\{\mathcal{L}_{k,r,t,b}(e_{k,i}) - \mathcal{L}_{k,r,t,b}(\textsc{Baseline}_{+}(k-1)), -\varepsilon^{k-1}\log^{2k-1}(nT)\right\} \\ &\in \left[-\varepsilon^{k-1}\log^{2k-1}(nT)), \varepsilon^{k-1}\log^{2k-1}(nT)\right].\end{aligned}$$

We finish the proof here. $\square$

Next we show that even though we update MWU in $\textsc{Baseline}_{+}(k)$ using the truncated loss, the regret with respect to the original expert $[n]$ can be still be bounded.

Lemma 5.4.2. *For any epoch $t \in [\varepsilon n]$, suppose $e_{k,i} \in \mathcal{P}_{k,r,t}$, then with probability at least $1 - 1/\operatorname{poly}(T)$,*

$$\sum_{b=1}^{B}\mathcal{L}_{k,r,t,b}(e_{k,i_{k,r,t,b}}) \leq \sum_{b=1}^{B}\widehat{\mathcal{L}}_{k,r,t,b}(e_{k,i}) + \mathcal{L}_{k,r,t,b}(\textsc{Baseline}_{+}(k-1)) + \frac{1}{4}\varepsilon^{k-2}\log^{2k}(nT).$$

Proof. By lemma 5.4.1, we note that $\widehat{\mathcal{L}}_{k,r,t}(e_{k,j}) \in [-\varepsilon^{k-1}\log^{2k-1}(nT)), \varepsilon^{k-1}\log^{2k-1}(nT)]$ for any $j \in [n]$. Recall that Baseline$_+(k)$ runs MWU in epoch t with $B = 1/\varepsilon^2$ decision days, hence, with probability at least $1 - \mathrm{poly}(T)$,

$$\begin{aligned}\sum_{b=1}^{B} \widehat{\mathcal{L}}_{k,r,t,b}(e_{k,i_{k,r,t,b}}) &\leq \sum_{b=1}^{B} \widehat{\mathcal{L}}_{k,r,t,b}(e_{k,i}) + \varepsilon^{k-1}\log^{2k-1}(nT)\cdot O\left(\sqrt{\log(nT)}/\varepsilon\right) \\ &\leq \sum_{b=1}^{B} \widehat{\mathcal{L}}_{k,r,t,b}(e_{k,i}) + \frac{1}{4}\varepsilon^{k-2}\log^{2k}(nT). \qquad (5.4.6)\end{aligned}$$

The LHS satisfies

$$\begin{aligned}\sum_{b=1}^{B} \widehat{\mathcal{L}}_{k,r,t,b}(e_{k,i_{k,r,t,b}}) &= \sum_{b=1}^{B} \max\left\{\mathcal{L}_{k,r,t,b}(e_{k,i_{k,r,t,b}}) - \mathcal{L}_{k,r,t,b}(\textsc{Baseline}_+(k-1)), -\varepsilon^{k-1}\log^{2k-1}(nT)\right\} \\ &\geq \mathcal{L}_{k,r,t,b}(e_{k,i_{k,r,t,b}}) \qquad (5.4.7) \\ &\quad - \mathcal{L}_{k,r,t,b}(\textsc{Baseline}_+(k-1)). \qquad (5.4.8)\end{aligned}$$

Combining eq. (5.4.6) and eq. (5.4.8), we conclude the proof. □

Lemma 5.4.3. *For any $i \in [n]$ and epoch $t \in [\varepsilon n]$, with probability at least $1 - 1/\mathrm{poly}(T)$,*

$$\sum_{b=1}^{B} \mathcal{L}_{k,r,t,b}(e_{k,i_{k,r,t,b}}) \leq \sum_{b=1}^{B} \mathcal{L}_{k,r,t,b}(i) + O\left(\varepsilon^{k-3}\log^{2k-2}(nT)\right).$$

Proof. For any expert $e_{k,i}$, any decision day $b \in [B]$, since MergeExp runs MWU over Baseline$_+(k-1)$ and expert i, then with probability at least $1 - 1/\mathrm{poly}(T)$,

$$\begin{aligned}\mathcal{L}_{k,r,t,b}(e_{k,i}) &\leq \mathcal{L}_{k,r,t,b}(\textsc{Baseline}_+(k-1)) + O\left(\sqrt{\log(nT)/T_{k-1}}\right) \\ &\leq \mathcal{L}_{k,r,t,b}(i) + \varepsilon^{k-1}\log^{2k-2}(nT) + O\left(\sqrt{\log(nT)/T_{k-1}}\right) \\ &\leq \mathcal{L}_{k,r,t,b}(i) + O\left(\varepsilon^{k-1}\log^{2k-2}(nT)\right).\end{aligned}$$

The first step follows from the guarantee of MWU, the second step follows from induction hypothesis (eq. (5.4.4)) and the last step follows from the choice of T_{k-1}. Summing over $b \in [B]$, we get the desired bound. □

Let $i^*_{k,r} \in [n]$ be the optimal expert in the r-th episode. We use the same stochastic process of section 5.3 to define the unlucky epoch $\mathcal{B}_{k,r} \subseteq [\varepsilon n]$ and assign $i(t) \in [n] \cup \{\mathsf{nil}\}$ ($t \in [\varepsilon n]$). Note that we fix all the randomness used at level $1, 2, \ldots, k-1$ and episode $1, 2, \ldots, r-1$ in advance, and so the stochastic process depends only on the randomness of Baseline$_+(k)$ inside episode r.

We can similarly bound the number of unlucky epochs.

Lemma 5.4.4 (unlucky epoch)**.** *With probability at least* $1 - 1/\operatorname{poly}(T)$*,*

(i) $|\mathcal{B}_{k,r}| \le O(\varepsilon^2 n \log T)$ *and*

(ii) $\sum_{b=1}^{B} \mathcal{L}_{k,r,t,b}(e_{k,i_{k,r,t,b}}) - \sum_{b=1}^{B} \mathcal{L}_{k,r,t,b}(i^*_{k,r}) \le O\left(\varepsilon^{k-3}\log^{2k-2}(nT)\right)$ *for any* $t \in \mathcal{B}_{k,r}$*.*

Proof. The first claim follows directly from lemma 5.3.3, the second claim follows from lemma 5.4.3. □

The covering property holds similarly:

Lemma 5.4.5 (cover)**.** *For any epoch* $t \in [\varepsilon n]\backslash\mathcal{B}_{k,r}$*, suppose* $\beta(\tau) \le t < \beta(\tau+1)$*, then with probability at least* $1 - 1/\operatorname{poly}(T)$*, we have*

(i) $i(\beta(\tau)) = \cdots = i(t) = \cdots = i(\beta(\tau+1)-1) = i^*_{k,r,\tau} \neq \mathsf{nil}$*;*

(ii) $i^*_{k,r,\tau} \in \mathcal{P}_{k,r,\nu}$ *for any* $\nu \in [\beta(\tau) : \beta(\tau+1)-1]$*; and*

(iii)

$$\begin{aligned}&\sum_{\nu=\beta(\tau)}^{\beta(\tau+1)-1}\sum_{b=1}^{B}\widehat{\mathcal{L}}_{k,r,t,b}(e_{k,i(t)}) + \mathcal{L}_{k,r,t,b}(\textsc{Baseline}_+(k-1))\\ <&\sum_{\nu=\beta(\tau)}^{\beta(\tau+1)-1}\sum_{b=1}^{B}\mathcal{L}_{k,r,t,b}(i^*_{k,r}) + 2\varepsilon^{k-2}\log^{2k-1}(nT)(\beta(\tau+1)-\beta(\tau)).\end{aligned}$$

Proof. The first two claims follow exactly from lemma 5.3.4. For the last claim, first we have

$$\begin{aligned}\sum_{\nu=\beta(\tau)}^{\beta(\tau+1)-1}\sum_{b=1}^{B}\widehat{\mathcal{L}}_{k,r,t,b}(e_{k,i(t)}) &< \sum_{\nu=\beta(\tau)}^{\beta(\tau+1)-1}\sum_{b=1}^{B}\widehat{\mathcal{L}}_{k,r,t,b}(e_{k,i^*_{k,r}}) + \varepsilon^k\log^{2k-1}(nT)B(\beta(\tau+1)-\beta(\tau))\\ &= \sum_{\nu=\beta(\tau)}^{\beta(\tau+1)-1}\sum_{b=1}^{B}\widehat{\mathcal{L}}_{k,r,t,b}(e_{k,i^*_{k,r}}) + \varepsilon^{k-2}\log^{2k-1}(nT)(\beta(\tau+1)-\beta(\tau)),\end{aligned} \tag{5.4.9}$$

where the first step follows from the third claim of lemma 5.4.5 by replacing ε with

$$\varepsilon^k\log^{2k-1}(nT),$$

and the second step comes from the choice of $B = 1/\varepsilon^2$.

With probability at least $1 - 1/\operatorname{poly}(T)$, we have

$$
\begin{aligned}
&\sum_{b=1}^{B} \widehat{\mathcal{L}}_{k,r,t,b}(e_{k,i^*_{k,r}}) \\
&= \sum_{b=1}^{B} \max\{\mathcal{L}_{k,r,t,b}(e_{k,i^*_{k,r}}) - \mathcal{L}_{k,r,t,b}(\textsc{Baseline}_{+}(k-1)), -\varepsilon^{k-1}\log^{2k-1}(nT)\} \\
&\leq \sum_{b=1}^{B} \max\{\mathcal{L}_{k,r,t,b}(i^*_{k,r}) - \mathcal{L}_{k,r,t,b}(\textsc{Baseline}_{+}(k-1)) \qquad (5.4.10)\\
&\quad + O\left(\sqrt{\log(nT)/T_{k-1}}\right), -\varepsilon^{k-1}\log^{2k-1}(nT)\} \\
&\leq \sum_{b=1}^{B} \mathcal{L}_{k,r,t,b}(i^*_{k,r}) - \mathcal{L}_{k,r,t,b}(\textsc{Baseline}_{+}(k-1)) + O(\sqrt{\log(nT)/T_{k-1}}) \cdot (1/\varepsilon^2) \\
&\leq \sum_{b=1}^{B} \mathcal{L}_{k,r,t,b}(i^*_{k,r}) - \mathcal{L}_{k,r,t,b}(\textsc{Baseline}_{+}(k-1)) + \varepsilon^{k-2}\log^{2k-1}(nT). \qquad (5.4.11)
\end{aligned}
$$

The first step follows from the definition, the second step follows from the first claim in lemma 5.4.1, the third step holds due to the induction hypothesis (eq. (5.4.4)), the last step follows from the choice of T_{k-1}.

Combining eq. (5.4.9) and eq. (5.4.11), we complete the proof. □

We can now wrap up the proof of the regret guarantee (proposition 5.4.1):

Proof of proposition 5.4.1. With probability at least $1 - 1/\operatorname{poly}(T)$, we have

$$
\begin{aligned}
&\sum_{t=1}^{\varepsilon n}\sum_{b=1}^{B}\mathcal{L}_{k,r,t,b}\left(e_{k,i_{k,r,t,b}}\right) - \mathcal{L}_{k,r,t,b}(i^*_{k,r})\\
= &\sum_{t\in[\varepsilon n]\backslash\mathcal{B}_{k,r}}\sum_{b=1}^{B}\left(\mathcal{L}_{k,r,t,b}\left(e_{k,i_{k,r,t,b}}\right) - \mathcal{L}_{k,r,t,b}(i^*_{k,r})\right) + \sum_{t\in\mathcal{B}_{k,r}}\sum_{b=1}^{B}\left(\mathcal{L}_{k,r,t,b}\left(e_{k,i_{k,r,t,b}}\right) - \mathcal{L}_{k,r,t,b}(i^*_{k,r})\right)\\
\leq &\sum_{t\in[\varepsilon n]\backslash\mathcal{B}_{k,r}}\sum_{b=1}^{B}\left(\mathcal{L}_{k,r,t,b}\left(e_{k,i_{k,r,t,b}}\right) - \mathcal{L}_{k,r,t,b}(i^*_{k,r})\right) + O(\varepsilon^2 n\log T)\cdot O(\varepsilon^{k-3}\log^{2k-2}(nT))\\
\leq &\sum_{t\in[\varepsilon n]\backslash\mathcal{B}_{k,r}}\sum_{b=1}^{B}\left(\widehat{\mathcal{L}}_{k,r,t,b}(e_{k,i(t)}) + \mathcal{L}_{k,r,t,b}(\textsc{Baseline}_+(k-1)) - \mathcal{L}_{k,r,t,b}(i^*_{k,r})\right)\\
&+\frac{1}{4}\varepsilon^{k-2}\log^{2k}(nT)\cdot\varepsilon n + O(\varepsilon^{k-1}n\log^{2k-1}(nT))\\
= &\sum_{\tau}\sum_{t\in[\beta(\tau):\beta(\tau+1)-1],t\notin\mathcal{B}}\sum_{b=1}^{B}\left(\widehat{\mathcal{L}}_{k,r,t,b}(e_{k,i(t)}) + \mathcal{L}_{k,r,t,b}(\textsc{Baseline}_+(k-1)) - \mathcal{L}_{k,r,t,b}(i^*_{k,r})\right)\\
&+\frac{1}{2}\varepsilon^{k-1}n\log^{2k}(nT)\\
\leq &\sum_{\tau}2\varepsilon^{k-2}\log^{2k-1}(nT)(\beta(\tau+1)-\beta(\tau)) + \frac{1}{2}\varepsilon^{k-1}n\log^{2k}(nT) \leq \varepsilon^{k-1}n\log^{2k}(nT).
\end{aligned}
$$

The second step follows from lemma 5.4.4, the third step is via lemma 5.4.2 and the fact that $i(t)\in\mathcal{P}_{k,r,t}$ for any $t\in[\varepsilon n]\backslash\mathcal{B}_{k,r}$ (see the second claim of lemma 5.4.5), the fourth step follows from lemma 5.4.5, and the fifth step from the third claim of lemma 5.4.5.

Hence, the average regret of the r-th episode equals $\varepsilon^{k-1}n\log^{2k}(nT)/(\varepsilon nB) = \varepsilon^k n\log^{2k}(nT)$. We finish the induction and complete the proof here. □

Memory

We bound the memory requirement of $\textsc{Baseline}_+(k)$ (for each $k\in[K]$). The proof essentially inherits from section 5.3, with the key observation that

$$\widehat{\mathcal{L}}_{k,r,t,b}\in[-\varepsilon^{k-1}\log^{2k-1}(nT), \varepsilon^{k-1}\log^{2k-1}(nT)]$$

instead of $[0,1]^n$. This allows one to perform a fine-grained division like eq. (5.4.3). In the remaining of this section, we always condition on the high probability event of the previous section.

The following lemma is similar to lemma 5.3.1.

Lemma 5.4.6. *For any level $k\in[2:K]$, episode $r\in T_k$ and epoch $t\in[\varepsilon n]$, suppose experts $e_{k,i}, e_{k,j}\in\widetilde{\mathcal{P}}_{k,r,t}$ and $e_{k,j}\succ e_{k,i}$. Let $\alpha\in(0,1)$, then at least one of the following must hold:*

(i) $\widehat{\mathcal{L}}_{j,j} \geq \widehat{\mathcal{L}}_{i,i} + 2\varepsilon^{k-1}\log^{2k-1}(nT)(\varepsilon/2 - \alpha)$;

(ii) $|\Gamma_{k,r,t,j}| \geq \left(1 + \frac{\alpha}{1-\alpha}\right)|\Gamma_{k,r,t,i}|$.

Here $\widehat{\mathcal{L}}_{i,i} = \widehat{\mathcal{L}}_{\Gamma_{k,r,t,i}}(e_{k,i})$, $\widehat{\mathcal{L}}_{i,j} = \mathcal{L}_{\Gamma_{k,r,t,i}}(e_{k,j})$ *and accordingly* $\widehat{\mathcal{L}}_{j,j} = \mathcal{L}_{\Gamma_{k,r,t,j}}(e_{k,j})$

Proof. Normalizing $\widehat{\mathcal{L}}_{k,r,t,b}$ by a factor of $2\varepsilon^{k-1}\log^{2k-1}(nT)$, by lemma 5.4.1, we know that the truncated loss $\widehat{\mathcal{L}}_{k,r,t,b}/2\varepsilon^{k-1}\log^{2k-1}(nT) \in [-1/2, 1/2]^n$. The new eviction rule (eq. (5.4.3)) reduces to the old one (eq. (5.3.1)) with ε replaced by $\varepsilon/2$. Then we can apply lemma 5.3.1 and get the desired. □

Similar to lemma 5.3.2, it follows that the pool size is small. The proof is analogous to lemma 5.3.2, by applying the same potential function using the conditions from lemma 5.4.6.

Lemma 5.4.7 (pool size)**.** *For any level* $k \in [K]$, *episode* $r \in T_k$ *and epoch* $t \in [\varepsilon n]$, *the size of the pool* $\widetilde{P}_{k,r,t}$ *is at most* $S_k = O(\varepsilon^{-1}\log T)$.

Now we can wrap up the memory requirement.

Proposition 5.4.2 (memory bound)**.** *At any time during the execution of* FullAlgo, *the memory usage is at most* $O\left(\frac{1}{\varepsilon^2}\log^4(nT)\right)$ *bits.*

Proof. At any time step, FullAlgo maintains Baseline$_+(k)$ for each $k \in K$ and experts $\{e_{k,i}\}_{k\in[K],i\in[n]}$. For Baseline$_+(k)$, by lemma 5.4.7, the size of the pool never exceeds $O(\varepsilon^{-1}\log T)$ and therefore it takes $O(\varepsilon^{-2}\log^3 nT)\cdot K = O(\varepsilon^{-2}\log^4 nT)$ bits of memory in total. Note that Baseline$_+(k)$ tracks $\{e_{k,i}\}_{i\in[n]}$ instead of the original expert. This does not take extra memory, since we always maintain Baseline$_+(k-1)$ and perform MWU on Baseline$_+(k-1)$ and expert i does not take extra memory. □

Combining proposition 5.4.1 and proposition 5.4.2, we can prove theorem 5.4.1.

Proof of theorem 5.4.1. Taking $\varepsilon^{-1} = n^{-\delta/2}$, by proposition 5.4.1, the memory never exceeds $O(n^\delta \log^4(nT))$. For regret analysis, suppose $n(n/\varepsilon)^K \leq T < n(n/\varepsilon)^{K+1}$ for some integer $K \geq 0$. The proposition 5.4.1 states that within $n(n/\varepsilon)^{K+1}$ days, the total regret is at most $O\left(n^{K+2}\log^{2K+2}(nT)\right) \leq \widetilde{O}\left(n^2 T^{\frac{2}{2+\delta}}\right)$. We conclude the proof here. □

Remark 5.4.1. *Our results extend easily to the case that* T *is unknown in advance: One can apply the common doubling trick and obtain the same result.*

5.5 Lower bound against adaptive adversary

We prove no algorithm can achieve sub-linear regret using sub-linear space when facing an adaptive adversary (see definition 5.2.2).

Theorem 5.5.1 (Lower bound against adaptive adversary)**.** *Let $n, T > 0$ be sufficiently large, $0 < \varepsilon < 1/40$. Any algorithm that achieves $O(\varepsilon T)$ regret against an adaptive adversary requires at least $\Omega(\min\{\varepsilon^{-1}\log_2 n, n\})$ bits of memory.*

Our lower bound construction utilizes the well-established connection of no-regret learning and zero-sum games. We first construct a family of zero-sum games whose equilibria are far apart (lemma 5.5.1 and lemma 5.5.2). We then prove it serves as a hard distribution for the online learning task. Throughout the proof, we assume $\varepsilon^{-1} \leq \frac{n}{20\log_2 n}$ and $k = 1/(2\varepsilon)$ is an integer.

Hard distribution We construct a family of zero-sum games. For any set $S \subseteq [n]$ of size k, the game matrix $A_S \in [0,4]^{n\times n}$ determines the loss of the first player (Alice):

$$A_S[i,j] = \begin{cases} 4 & i \notin S \\ 0 & i \in S, i \neq j \\ 1 & i \in S, i = j. \end{cases}$$

To summarize, Alice receives loss 4 if she plays any action outside of the support of S, and they are strictly dominated by actions in S. For any action pair $(i,j) \in S \times S$, the game is constructed as a generalized matching penny game: Alice receives loss 1 if her action is matched by Bob, and she receives loss 0 otherwise. The (hard) distribution $\mathcal{D}_k$ is defined as the uniform distribution over the above family of zero-sum games $\{A_S\}_{S\subseteq[n],|S|=k}$.

We first make a simple observation about the equilibrium strategy.

Lemma 5.5.1. *For any $S \subseteq [n]$ of size k, the minmax value of game A_S equals $1/k$. Furthermore, Alice's equilibrium strategy is $\frac{1}{k}\cdot \mathbf{1}_S$, where $\mathbf{1}_S$ is the indicator vector whose i-th entry equals $\mathbf{1}\{i \in S\}$.*

Proof. It is easy to verify that $(\mathbf{1}_S, \mathbf{1}_S)$ is the unique equilibrium of the game, and by definition, Alice receives $\frac{1}{k}$ loss in the equilibrium. □

We then prove that no single strategy can (approximately) cover the minmax strategy of a large number of games. For any strategy $p \in \Delta_n$ and game A_S, define $\ell(p,S)$ as the worst case loss received by Alice when playing p in game A_S, i.e., $\ell(p,S) = \max_{i\in[n]} p^\top A_S \mathbf{1}_i$.

Lemma 5.5.2. *For any fixed strategy $p \in \Delta_n$, there are at most $\binom{n}{3k/4}$ number of sets $S \subseteq [n]$ ($|S| = k$) such that $\ell(p,S) < 2/k$.*

Proof. Without loss of generality, we assume $p_1 \geq p_2 \geq \cdots \geq p_n$.

Case 1. Suppose $p_1 \geq 2/k$. Then for any set $S \subseteq [n]$ ($|S| = k$), there are two cases: either (1) if $1 \in S$, then Bob plays action 1 and Alice receives at least $p_1 \geq 2/k$ loss; or (2) if $1 \notin S$, then Alice receives at least $4p_1 \geq 8/k$ loss. Therefore, in this case, there is no S satisfies $\ell(p, S) \leq 2/k$.

Case 2. Suppose $p_1 < 2/k$. Let $i^* \in [n]$ be the largest integer such that $p_{i^*} \geq 1/2k$ and let $I = [i^*]$. Note if $p_1 < 1/2k$, we simply take $I = \emptyset$.

For any S satisfies $\ell(p, S) < 2/k$, we first prove $I \subseteq S$. Otherwise, suppose that there exists an index $i \in I$ such that $i \notin S$, then by having Bob play action i, Alice receives at least $4p_i \geq 2/k$ loss.

Case 2-1. Suppose $|I| < k/4$, then we claim that there is no set S satisfies $\ell(p, S) < 2/k$. To see this, for any set $S \subseteq [n]$ of size k, we note that

$$\sum_{i\in S} p_i = \sum_{i \in I} p_i + \sum_{i \in S\backslash I} p_i \leq \frac{k}{4} \cdot \frac{2}{k} + \frac{3k}{4} \cdot \frac{1}{2k} = \frac{7}{8},$$

where the second step holds as $|I| < k/4$, and for $i \in I, p_i < p_1 = 2/k$, for $i \in S\backslash I$, $p_i < 1/2k$. Hence, we have $\sum_{i \notin S} p_i \geq 1/8$ and Alice receives at least $1/8 \cdot 4 = 1/2$ loss.

Case 2-2. Suppose $|I| > 1/4k$. We have already proved $I \subseteq S$, and therefore, there are at most $3k/4$ indices in $S\backslash I$ and they can be chosen from $[n]\backslash I$, which can be upper bounded by $\binom{n}{3k/4}$.

We conclude the lemma here. □

Now we can prove the lower bound against adaptive adversary. The high-level idea is that one can use a lower memory no-regret algorithm to approximate the minmax value of a zero-sum game (where the opponent always plays the best response). Meanwhile, by a counting argument, the number of different zero sum games constructed above are too large and can not be "covered" by a low memory algorithm.

Proof of theorem 5.5.1. Let ALG be any algorithm with asymptotic regret of $O(\varepsilon T)$ when facing a strong adaptive adversary (definition 5.2.2). Suppose ALG uses M bits of memory and we shall prove $M \geq \Omega\left(\varepsilon^{-1} \log_2 n\right)$ (note we assume $\varepsilon^{-1} \leq \frac{n}{\log_2 n}$ at the very beginning). Consider the following scenario.

- A game A_S is sampled from the distribution $\mathcal{D}_k$.
- Alice and Bob play the game A_S repeatedly for T rounds. In the t-th round ($t \in [T]$)
  - Alice consults with ALG and commits a distribution $p_t \in \Delta_n$ over her n actions;
  - Bob receives p_t and best responds to Alice with $y_t = \arg\max_{i\in[n]} \left(p_t^\top A\right)_i$.

Slightly abuse of notation, we would also use y_t as an indicator vector. At the end, Alice would achieve ε-approximate to its minmax value:

$$\frac{1}{T}\sum_{t=1}^{T}\langle p_t, A_S y_t\rangle \le \frac{1}{T}\min_{i^*}\sum_{t=1}^{T}\langle i^*, A_S y_t\rangle + \varepsilon \le \frac{1}{k} + \varepsilon \le \frac{3}{2k}. \tag{5.5.1}$$

The first step follows from the regret guarantee of ALG, the second step follows from the min-max Theorem (lemma 5.2.3) and the minmax value of A_S always equals to $1/k$ (lemma 5.5.1), the last step follows from the choice of k.

We prove by contradiction and assume that ALG uses at most $M = \frac{1}{10}\varepsilon^{-1}\log_2 n$ bits of memory. We aim to prove that Alice has loss at least $3/2k$ for every iteration with high probability (over the choice of S). The proof is via the following counting argument.

As the algorithm uses at most $\frac{1}{10}\varepsilon^{-1}\log_2 n$ bits of memory, there are $2^M = n^{k/5}$ possible memory states X in total. For each state $x \in X$, suppose that the algorithm outputs a strategy $p_x \in \Delta_n$ (note p_x could be a random variable). Define

$$T_x := \{S : \Pr[\ell(p_x, S) < 2/k] \ge 0.1, S \in [n], |S| = k\}.$$

By lemma 5.5.2, we know that $|T_x| \le 10 \cdot \binom{n}{3k/4}$. Taking an union over all states $x \in X$, one has

$$\left|\bigcup_{x\in X} T_x\right| \le n^{k/5}\cdot 10\binom{n}{3k/4} \le \frac{1}{100}\binom{n}{k}.$$

The last step follows from the choice of parameters.

Hence, we conclude that with probability at least 0.99, the nature draws a game A_S such that $S \notin \bigcup_{x\in X} T_x$. This means that in each round of the game, the algorithm receives at least $\frac{1}{10}\frac{1}{k} + \frac{9}{10}\frac{2}{k} = \frac{19}{10k}$ loss in expectation since Bob always plays best response. This contradicts with eq. (5.5.1) and we conclude the proof. □

5.6 Conclusion

In this paper, we provide the first sub-linear space online learning algorithm that achieves sub-linear regret when facing an oblivious adversary. A separation has also been established between oblivious and strong adaptive adversaries, where a linear memory lower bound is shown to be necessary to achieve sub-linear regret.

Our work opens up a variety of exciting future research directions:

(i) First, an immediately interesting question is to close the gap between the space upper and lower bound.

(ii) Second, the adaptive adversary model considered in the paper is relatively strong for some applications, where the adversary only sees the prior decisions but not the mixed strategy of current round. A natural open question is to investigate this model, often called black-box adversary in the adversarially robust streaming literature.

(iii) Third, other problem-specific notions of regret have been studied under different settings, such as dynamic environments [70] and game theory [24, 37, 12]. A natural follow-up question is whether sub-linear space is achievable there.

(iv) Finally, the MWU algorithm has numerous applications in game theory [72] and machine learning (including boosting [71] and reinforcement learning [48]). Our paper opens up opportunities of deriving sub-linear space algorithms for these applications.

Bibliography

[1] Jayadev Acharya et al. "Estimating entropy of distributions in constant space". In: *Advances in Neural Information Processing Systems (NeurIPS)* (2019).

[2] Ishaq Aden-Ali, Hassan Ashtiani, and Gautam Kamath. "On the sample complexity of privately learning unbounded high-dimensional gaussians". In: *Algorithmic Learning Theory (ALT)*. 2021.

[3] Arpit Agarwal, Sanjeev Khanna, and Prathamesh Patil. "A Sharp Memory-Regret Trade-Off for Multi-Pass Streaming Bandits". In: *Conference on Learning Theory (COLT)* (2022).

[4] Sara Ahmadian et al. "Robust load balancing with machine learned advice". In: *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 2022.

[5] Miklós Ajtai et al. "The White-Box Adversarial Data Stream Model". In: *Proceedings of the 41st ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS)*. 2022. ISBN: 9781450392600.

[6] Reka Albert. "Scale-free networks in cell biology". In: *Journal of Cell Science* 118.21 (2005), pp. 4947–4957.

[7] Maryam Aliakbarpour et al. "Estimation of Entropy in Constant Space with Improved Sample Complexity". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2022.

[8] Zeyuan Allen-Zhu, Zhenyu Liao, and Lorenzo Orecchia. "Spectral sparsification and regret minimization beyond matrix multiplicative updates". In: *ACM Symposium on Theory of Computing (STOC '15)*. 2015.

[9] Noga Alon, Yossi Matias, and Mario Szegedy. "The space complexity of approximating the frequency moments". In: *J. Comput. System Sci.* 58.1, part 2 (1999), pp. 137–147. ISSN: 0022-0000. DOI: 10.1006/jcss.1997.1545.

[10] Noga Alon, Yossi Matias, and Mario Szegedy. "The space complexity of approximating the frequency moments". In: *Journal of Computer and System Sciences* 58.1 (1999), pp. 137–147.

[11] Noga Alon et al. “Adversarial laws of large numbers and optimal regret in online classification”. In: *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing (STOC)*. 2021.

[12] Ioannis Anagnostides et al. “Near-optimal no-regret learning for correlated equilibria in multi-player general-sum games”. In: *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*. 2022.

[13] Sanjeev Arora, Elad Hazan, and Satyen Kale. “The multiplicative weights update method: a meta-algorithm and applications”. In: *Theory of Computing* 8.1 (2012), pp. 121–164. DOI: 10.4086/toc.2012.v008a006.

[14] Hassan Ashtiani and Christopher Liaw. “Private and polynomial time algorithms for learning Gaussians and beyond”. In: *Conference on Learning Theory (COLT)*. 2022.

[15] Sepehr Assadi and Chen Wang. “Exploration with limited memory: streaming algorithms for coin tossing, noisy comparisons, and multi-armed bandits”. In: *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing (STOC)*. 2020.

[16] Sivaraman Balakrishnan et al. “Computationally efficient robust sparse estimation in high dimensions”. In: *Conference on Learning Theory (COLT ’17)*. 2017.

[17] Albert-Laszlo Barabasi. “The origin of bursts and heavy tails in human dynamics”. In: *Nature* 435.7039 (2005), pp. 207–211.

[18] Boaz Barak, Moritz Hardt, and Satyen Kale. “The uniform hardcore lemma via approximate Bregman projections”. In: *ACM-SIAM Symposium on Discrete Algorithms (SODA ’09)*. 2009, pp. 1193–1200. URL: http://dl.acm.org/citation.cfm?id=1496770.1496899.

[19] Amos Beimel, Shiva Prasad Kasiviswanathan, and Kobbi Nissim. “Bounds on the Sample Complexity for Private Learning and Private Data Release”. In: *Conference on Theory of Cryptography (TCC)*. Lecture Notes in Computer Science. 2010.

[20] Omri Ben-Eliezer et al. “A Framework for Adversarially Robust Streaming Algorithms”. In: *Journal of the ACM (JACM)* 69.2 (2022), pp. 1–33.

[21] Sourav Biswas et al. “Coinpress: Practical private mean and covariance estimation”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2020.

[22] Adam Block et al. “Smoothed online learning is as easy as statistical learning”. In: *Conference on Learning Theory (COLT)*. 2022.

[23] Avrim Blum, John Hopcroft, and Ravindran Kannan. *Foundations of Data Science*. Cambridge University Press, 2019.

[24] Avrim Blum and Yishay Mansour. “From external to internal regret.” In: *Journal of Machine Learning Research* 8.6 (2007).

[25] Dan Boneh and James Shaw. “Collusion-Secure Fingerprinting for Digital Data”. In: *IEEE Trans. Inf. Theory* 44.5 (1998), pp. 1897–1905.

[26] Gavin Brown, Mark Bun, and Adam Smith. "Strong Memory Lower Bounds for Learning Natural Models". In: *Conference on Learning Theory (COLT)* (2022).

[27] Gavin Brown et al. "Covariance-aware private mean estimation without private covariance estimation". In: *Advances in Neural Information Processing Systems*. 2021.

[28] George W Brown. "Iterative solution of games by fictitious play". In: *Act. Anal. Prod Allocation* 13.1 (1951), p. 374.

[29] Mark Bun, Jonathan Ullman, and Salil Vadhan. "Fingerprinting Codes and the Price of Approximate Differential Privacy". In: *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC)*. 2014.

[30] Mark Bun et al. "Private Hypothesis Selection". In: *IEEE Transactions on Information Theory* 67.3 (2021), pp. 1981–2000. URL: https://doi.org/10.1109/TIT.2021.3049802.

[31] Olivier Catoni. "Challenging the empirical mean and empirical variance: a deviation study". In: *Ann. Inst. Henri Poincaré Probab. Stat.* 48.4 (2012), pp. 1148–1185. ISSN: 0246-0203. DOI: 10.1214/11-AIHP454.

[32] Olivier Catoni. "Challenging the empirical mean and empirical variance: a deviation study". In: *Annales de l'IHP Probabilités et statistiques* 48.4 (2012), pp. 1148–1185.

[33] Nicolo Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge university press, 2006.

[34] Nicolò Cesa-Bianchi, Ofer Dekel, and Ohad Shamir. "Online Learning with Switching Costs and Other Adaptive Adversaries". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2013.

[35] Arghya Roy Chaudhuri and Shivaram Kalyanakrishnan. "Regret minimisation in multi-armed bandits using bounded arm memory". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*. 2020.

[36] Xi Chen, Christos Papadimitriou, and Binghui Peng. "Memory Bounds for Continual Learning". In: *2022 IEEE 63th Annual Symposium on Foundations of Computer Science (FOCS)*. 2022.

[37] Xi Chen and Binghui Peng. "Hedging in games: Faster convergence of external and swap regrets". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2020.

[38] Yu Cheng, Ilias Diakonikolas, and Rong Ge. "High-dimensional robust mean estimation in nearly-linear time". In: *ACM-SIAM Symposium on Discrete Algorithms (SODA '19)*. 2019, pp. 2755–2771. DOI: 10.1137/1.9781611975482.171.

[39] Yu Cheng, Ilias Diakonikolas, and Rong Ge. "High-dimensional robust mean estimation in nearly-linear time". In: *ACM-SIAM Symposium on Discrete Algorithms (SODA '19)*. 2019.

[40] Yu Cheng et al. "Faster Algorithms for High-Dimensional Robust Covariance Estimation". In: *Conference on Learning Theory (COLT '19)*. 2019.

[41] Yu Cheng et al. "High-Dimensional Robust Mean Estimation via Gradient Descent". In: *International Conference on Machine Learning (ICML '20)*. 2020.

[42] Yeshwanth Cherapanamjeri, Nicolas Flammarion, and Peter L Bartlett. "Fast Mean Estimation with Sub-Gaussian Rates". In: *Conference on Learning Theory (COLT '19)*. 2019, pp. 786–806. URL: https://arxiv.org/abs/1902.01998.

[43] Yeshwanth Cherapanamjeri et al. "Algorithms for Heavy-Tailed Statistics: Regression, Covariance Estimation, and Beyond". In: *arXiv preprint arXiv:1912.11071* (2019).

[44] Paul Christiano et al. "Electrical flows, laplacian systems, and faster approximation of maximum flow in undirected graphs". In: *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing (STOC)*. 2011, pp. 273–282.

[45] Michael Crouch et al. "Stochastic streams: Sample complexity vs. space complexity". In: *24th Annual European Symposium on Algorithms (ESA)*. 2016.

[46] Yuval Dagan and Ohad Shamir. "Detecting correlations with little memory and communication". In: *Conference On Learning Theory (COLT)*. 2018.

[47] Yan Dai, Haipeng Luo, and Liyu Chen. "Follow-the-Perturbed-Leader for Adversarial Markov Decision Processes with Bandit Feedback". In: *Advances in Neural Information Processing Systems (NeurIPS)* (2022).

[48] Constantinos Daskalakis, Noah Golowich, and Kaiqing Zhang. "The complexity of Markov equilibrium in stochastic games". In: *arXiv preprint arXiv:2204.03991* (2022).

[49] Ofer Dekel, Ambuj Tewari, and Raman Arora. "Online Bandit Learning against an Adaptive Adversary: from Regret to Policy Regret". In: *Proceedings of the 29th International Conference on Machine Learning (ICML)*. 2012.

[50] Luc Devroye, Abbas Mehrabian, and Tommy Reddad. "The minimax learning rates of normal and Ising undirected graphical models". In: *Electronic Journal of Statistics* 14.1 (2020), pp. 2338–2361.

[51] Luc Devroye, Abbas Mehrabian, and Tommy Reddad. "The total variation distance between high-dimensional Gaussians". In: *arXiv preprint arXiv:1810.08693* (2018).

[52] Luc Devroye et al. "Sub-Gaussian mean estimators". In: *Annals of Statistics* 44.6 (2016), pp. 2695–2725. DOI: 10.1214/16-AOS1440.

[53] Ilias Diakonikolas and Daniel M Kane. "Recent Advances in Algorithmic High-Dimensional Robust Statistics". In: *arXiv preprint arXiv:1911.05911* (2019).

[54] Ilias Diakonikolas, Daniel M Kane, and Ankit Pensia. "Outlier robust mean estimation with subgaussian rates via stability". In: *arXiv preprint arXiv:2007.15618* (2020).

[55] Ilias Diakonikolas, Daniel M Kane, and Alistair Stewart. "List-decodable robust mean estimation and learning mixtures of spherical Gaussians". In: *ACM Symposium on Theory of Computing (STOC '18)*. 2018.

[56] Ilias Diakonikolas and Daniel M. Kane. "Recent Advances in Algorithmic High-Dimensional Robust Statistics". In: *arXiv preprint arXiv:1911.05911* (2019). URL: https://arxiv.org/abs/1911.05911.

[57] Ilias Diakonikolas, Weihao Kong, and Alistair Stewart. "Efficient algorithms and lower bounds for robust linear regression". In: *ACM-SIAM Symposium on Discrete Algorithms (SODA '19)*. 2019.

[58] Ilias Diakonikolas et al. "Being robust (in high dimensions) can be practical". In: *International Conference on Machine Learning (ICML '17)*. 2017.

[59] Ilias Diakonikolas et al. "Robust Estimators in High-Dimensions Without the Computational Intractability". In: *SIAM Journal on Computing* 48.2 (2019), pp. 742–864. DOI: 10.1137/17M1126680.

[60] Ilias Diakonikolas et al. "Robust estimators in high-dimensions without the computational intractability". In: *SIAM Journal on Computing* 48.2 (2019), pp. 742–864.

[61] Ilias Diakonikolas et al. "Robustly learning a gaussian: Getting optimal error, efficiently". In: *ACM-SIAM Symposium on Discrete Algorithms (SODA '18)*. SIAM. 2018.

[62] Ilias Diakonikolas et al. "Sever: A Robust Meta-Algorithm for Stochastic Optimization". In: *International Conference on Machine Learning (ICML '19)*. 2019.

[63] Ilias Diakonikolas et al. "Streaming Algorithms for High-Dimensional Robust Statistics". In: *International Conference on Machine Learning (ICML)*. 2022.

[64] Yihe Dong, Samuel B Hopkins, and Jerry Li. "Quantum Entropy Scoring for Fast Robust Mean Estimation and Improved Outlier Detection". In: *Neural Information Processing Systems (NeurIPS '19)*. 2019.

[65] John C Doyle, Bruce A Francis, and Allen R Tannenbaum. *Feedback control theory*. Courier Corporation, 2013.

[66] Miroslav Dudík et al. "Oracle-efficient online learning and auction design". In: *Journal of the ACM (JACM)* 67.5 (2020), pp. 1–57.

[67] Cynthia Dwork et al. "Calibrating Noise to Sensitivity in Private Data Analysis". In: *Third Theory of Cryptography Conference on Theory of Cryptography (TCC)*. 2006.

[68] Cynthia Dwork et al. "Our Data, Ourselves: Privacy Via Distributed Noise Generation". In: *Annual International Conference on Theory and Application of Cryptographic Techniques (EUROCRYPT)*. 2006.

[69] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. "On power-law relationships of the internet topology". In: *ACM SIGCOMM Computer Communication Review* 29.4 (1999), pp. 251–262.

[70] Dylan J Foster, Alexander Rakhlin, and Karthik Sridharan. "Adaptive online learning". In: *Advances in Neural Information Processing Systems (NIPS)* (2015).

[71] Yoav Freund and Robert E Schapire. “A decision-theoretic generalization of on-line learning and an application to boosting”. In: *Journal of Computer and System Sciences* 55.1 (1997), pp. 119–139.

[72] Yoav Freund and Robert E Schapire. “Adaptive game playing using multiplicative weights”. In: *Games and Economic Behavior* 29.1-2 (1999), pp. 79–103.

[73] Dan Garber and Elad Hazan. “Sublinear time algorithms for approximate semidefinite programming”. In: *Mathematical Programming* 158.1 (2016), pp. 329–361.

[74] Sumegha Garg, Ran Raz, and Avishay Tal. “Extractor-based time-space lower bounds for learning”. In: *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*. 2018.

[75] Sumegha Garg et al. “Memory-sample lower bounds for learning parity with noise”. In: *24th International Conference on Approximation Algorithms for Combinatorial Optimization Problems (APPROX) and 25th International Conference on Randomization and Computation (RANDOM)*. 2021.

[76] Kristian Georgiev and Samuel B Hopkins. “Privacy Induces Robustness: Information-Computation Gaps and Sparse Mean Estimation”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2022.

[77] Alon Gonen, Shachar Lovett, and Michal Moshkovitz. “Towards a combinatorial characterization of bounded-memory learning”. In: *Advances in Neural Information Processing Systems (NeurIPS)* (2020).

[78] Nika Haghtalab, Tim Roughgarden, and Abhishek Shetty. “Smoothed analysis with adaptive adversaries”. In: *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*. 2022.

[79] Nika Haghtalab et al. “Oracle-Efficient Online Learning for Beyond Worst-Case Adversaries”. In: *Advances in Neural Information Processing Systems (NeurIPS)* (2022).

[80] Moritz Hardt and Kunal Talwar. “On the geometry of differential privacy”. In: *Proceedings of the 2010 ACM International Symposium on Theory of Computing (STOC)*. 2010.

[81] Elad Hazan. “Introduction to Online Convex Optimization”. In: *Foundations and Trends in Optimization* 2.3-4 (2016), pp. 157–325.

[82] Elad Hazan and Tomer Koren. “The computational power of optimization in online learning”. In: *Proceedings of the Forty-eighth Annual ACM Symposium on Theory of Computing (STOC)*. 2016, pp. 128–141.

[83] David P Helmbold and Robert E Schapire. “Predicting nearly as well as the best pruning of a decision tree”. In: *Machine Learning* 27.1 (1997), pp. 51–68.

[84] Mark Herbster and Manfred K Warmuth. “Tracking the best linear predictor”. In: *Journal of Machine Learning Research* 1.Sep (2001), pp. 281–309.

[85] Sam Hopkins, Jerry Li, and Fred Zhang. "Robust and heavy-tailed mean estimation made simple, via regret minimization". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2020.

[86] Samuel B Hopkins. "Sub-Gaussian mean estimation in polynomial time". In: *arXiv preprint arXiv:1809.07425* (2018). URL: https://arxiv.org/abs/1809.07425.

[87] Samuel B Hopkins, Gautam Kamath, and Mahbod Majid. "Efficient mean estimation with pure differential privacy via a sum-of-squares exponential mechanism". In: *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*. 2022.

[88] Samuel B Hopkins and Jerry Li. "Mixture models, robustness, and sum of squares proofs". In: *ACM SIGACT Symposium on Theory of Computing (STOC '18)*. 2018.

[89] Samuel B. Hopkins. "Mean estimation with sub-Gaussian rates in polynomial time". In: *Annals of Statistics* 48.2 (Apr. 2020), pp. 1193–1213.

[90] Samuel B. Hopkins et al. *Robustness Implies Privacy in Statistical Estimation*. 2022. DOI: 10.48550/ARXIV.2212.05015. URL: https://arxiv.org/abs/2212.05015.

[91] Peter J Huber. "Robust Estimation of a Location Parameter". In: *The Annals of Mathematical Statistics* 35.1 (1964), pp. 73–101.

[92] Mark R. Jerrum, Leslie G. Valiant, and Vijay V. Vazirani. "Random generation of combinatorial structures from a uniform distribution". In: *Theoret. Comput. Sci.* 43.2-3 (1986), pp. 169–188. ISSN: 0304-3975. DOI: 10.1016/0304-3975(86)90174-X.

[93] Tianyuan Jin et al. "Optimal streaming algorithms for multi-armed bandits". In: *International Conference on Machine Learning (ICML)*. 2021.

[94] Emilien Joly, Gábor Lugosi, and Roberto Imbuzeiro Oliveira. "On the estimation of the mean of a random vector". In: *Electronic Journal of Statistics* 11.1 (2017), pp. 440–451. DOI: 10.1214/17-EJS1228.

[95] Adam Kalai and Santosh Vempala. "Efficient algorithms for online decision problems". In: *Journal of Computer and System Sciences* 71.3 (2005), pp. 291–307.

[96] Gautam Kamath, Argyris Mouzakis, and Vikrant Singhal. "New Lower Bounds for Private Estimation and a Generalized Fingerprinting Lemma". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2022.

[97] Gautam Kamath, Vikrant Singhal, and Jonathan Ullman. "Private mean estimation of heavy-tailed distributions". In: *Conference on Learning Theory (COLT)*. 2020.

[98] Gautam Kamath et al. "A private and computationally-efficient estimator for unbounded gaussians". In: *Conference on Learning Theory (COLT)*. 2022.

[99] Gautam Kamath et al. "Privately learning high-dimensional distributions". In: *Conference on Learning Theory (COLT)*. 2019.

[100] Purushottam Kar et al. “On the Generalization Ability of Online Learning Algorithms for Pairwise Loss Functions”. In: *International Conference on Machine Learning (ICML)*. 2013.

[101] Zohar Karnin et al. “Unsupervised SVMs: On the complexity of the furthest hyperplane problem”. In: *Conference on Learning Theory (COLT ’12)*. 2012, pp. 1–17. URL: http://proceedings.mlr.press/v23/karnin12.html.

[102] Richard M Karp, Umesh V Vazirani, and Vijay V Vazirani. “An optimal algorithm for on-line bipartite matching”. In: *Proceedings of the Twenty-second Annual ACM Symposium on Theory of Computing (STOC)*. 1990.

[103] Vishesh Karwa and Salil P. Vadhan. “Finite Sample Differentially Private Confidence Intervals”. In: *Innovations in Theoretical Computer Science Conference (ITCS)*. 2018.

[104] Adam Klivans and Raghu Meka. “Learning graphical models using multiplicative weights”. In: *IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. 2017.

[105] Pravesh Kothari, Pasin Manurangsi, and Ameya Velingker. “Private robust estimation by stabilizing convex relaxations”. In: *Conference on Learning Theory (COLT)*. 2022.

[106] Pravesh K Kothari, Peter Manohar, and Brian Hu Zhang. “Polynomial-Time Sum-of-Squares Can Robustly Estimate Mean and Covariance of Gaussians Optimally”. In: *Algorithmic Learning Theory (ALT)*. 2022.

[107] Jacek Kuczyński and Henryk Woźniakowski. “Estimating the largest eigenvalue by the power and Lanczos algorithms with a random start”. In: *SIAM Journal on Matrix Analysis and Applications* 13.4 (1992), pp. 1094–1122.

[108] Kevin A Lai, Anup B Rao, and Santosh Vempala. “Agnostic estimation of mean and covariance”. In: *Symposium on Foundations of Computer Science (FOCS ’16)*. 2016, pp. 665–674. DOI: 10.1109/FOCS.2016.76.

[109] Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

[110] Guillaume Lecué and Jules Depersin. “Robust subgaussian estimation of a mean vector in nearly linear time”. In: *arXiv preprint arXiv:1906.03058* (2019). URL: https://arxiv.org/abs/1906.03058.

[111] Michel Ledoux. *The concentration of measure phenomenon*. American Mathematical Society, 2001.

[112] Zhixian Lei et al. “A fast spectral algorithm for mean estimation with sub-gaussian rates”. In: *Conference on Learning Theory (COLT ’20)*. 2020.

[113] Matthieu Lerasle et al. “MONK Outlier-Robust Mean Embedding Estimation by Median-of-Means”. In: *International Conference on Machine Learning*. 2019, pp. 3782–3793.

[114] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. "Graphs over time: densification laws, shrinking diameters and possible explanations". In: *ACM SIGKDD International Conference on Knowledge Discovery in Data Mining (KDD '05)*. 2005.

[115] Jerry Li. *Lecture 4: Spectral signatures and efficient certifiability*. https://jerryzli.github.io/robust-ml-fall19/lec4.pdf. 2019.

[116] Jerry Li. *Lecture 5: Filtering from spectral signatures*. https://jerryzli.github.io/robust-ml-fall19/lec5.pdf. 2019.

[117] Jerry Zheng Li. "Principled approaches to robust machine learning and beyond". PhD thesis. Massachusetts Institute of Technology, 2018.

[118] David Liau et al. "Stochastic multi-armed bandits in constant space". In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*. 2018.

[119] N Littlestone and MK Warmuth. "The weighted majority algorithm". In: *30th Annual Symposium on Foundations of Computer Science (FOCS)*. 1989.

[120] Xiyang Liu, Weihao Kong, and Sewoong Oh. "Differential privacy and robust statistics in high dimensions". In: *Conference on Learning Theory (COLT)*. 2022.

[121] Xiyang Liu et al. "Robust and differentially private mean estimation". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2021.

[122] Gabor Lugosi and Shahar Mendelson. "Robust multivariate mean estimation: the optimality of trimmed mean". In: *Annals of Statistics* (2020).

[123] Gábor Lugosi and Shahar Mendelson. "Mean Estimation and Regression Under Heavy-Tailed Distributions: A Survey". In: *Foundations of Computational Mathematics* (Aug. 2019). DOI: 10.1007/s10208-019-09427-x.

[124] Gábor Lugosi and Shahar Mendelson. "Near-optimal mean estimators with respect to general norms". In: *Probability Theory and Related Fields* (2019). ISSN: 1432-2064. DOI: 10.1007/s00440-019-00906-4.

[125] Gábor Lugosi and Shahar Mendelson. "Sub-Gaussian estimators of the mean of a random vector". In: *Annals of Statistics* 47.2 (2019), pp. 783–794. ISSN: 0090-5364. DOI: 10.1214/17-AOS1639.

[126] Gábor Lugosi and Shahar Mendelson. "Sub-gaussian estimators of the mean of a random vector". In: *Annals of Statistics* 47.2 (2019), pp. 783–794.

[127] Wolfgang Maass and Manfred K Warmuth. "Efficient learning with virtual threshold gates". In: *Information and Computation* 141.1 (1998), pp. 66–83.

[128] Arnab Maiti, Vishakha Patil, and Arindam Khan. "Multi-Armed Bandits with Bounded Arm-Memory: Near-Optimal Guarantees for Best-Arm Identification and Regret Minimization". In: *Advances in Neural Information Processing Systems (NeurIPS)* (2021).

[129] Annie Marsden et al. "Efficient Convex Optimization Requires Superlinear Memory". In: *Conference on Learning Theory (COLT)*. 2022.

[130] Frank McSherry. “Privacy integrated queries: an extensible platform for privacy-preserving data analysis”. In: *Commun. ACM* 53.9 (2010), pp. 89–97. DOI: 10.1145/1810891.1810916. URL: https://doi.org/10.1145/1810891.1810916.

[131] Frank McSherry and Kunal Talwar. “Mechanism Design via Differential Privacy”. In: *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. 2007.

[132] Stanislav Minsker. “Geometric median and robust estimation in Banach spaces”. In: *Bernoulli* 21.4 (2015), pp. 2308–2335. DOI: 10.3150/14-BEJ645.

[133] Rajeev Motwani and Prabhakar Raghavan. *Randomized algorithms.* Cambridge University Press, 1995.

[134] A. S. Nemirovsky and D. B. and Yudin. *Problem complexity and method efficiency in optimization.* A Wiley-Interscience Publication. Translated from the Russian and with a preface by E. R. Dawson, Wiley-Interscience Series in Discrete Mathematics. John Wiley & Sons, Inc., New York, 1983, pp. xv+388. ISBN: 0-471-10345-4.

[135] J. v. Neumann. “Zur theorie der gesellschaftsspiele”. In: *Mathematische Annalen* 100.1 (1928), pp. 295–320.

[136] Masashi Okamoto. “Some inequalities relating to the partial sum of binomial probabilities”. In: *Annals of the institute of Statistical Mathematics* 10.1 (1959), pp. 29–35.

[137] Erik Ordentlich and Thomas M Cover. “The cost of achieving the best portfolio in hindsight”. In: *Mathematics of Operations Research* 23.4 (1998), pp. 960–982.

[138] Adarsh Prasad, Sivaraman Balakrishnan, and Pradeep Ravikumar. “A Unified Approach to Robust Mean Estimation”. In: *arXiv preprint arXiv:1907.00927* (2019). URL: https://arxiv.org/abs/1907.00927.

[139] Alexander Rakhlin, Karthik Sridharan, and Ambuj Tewari. “Online Learning: Stochastic, Constrained, and Smoothed Adversaries”. In: *Advances in Neural Information Processing Systems (NIPS)*. 2011.

[140] Ran Raz. “A time-space lower bound for a large class of learning problems”. In: *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. 2017.

[141] Ran Raz. “Fast learning requires good memory: A time-space lower bound for parity learning”. In: *Journal of the ACM (JACM)* 66.1 (2018), pp. 1–18.

[142] Vatsal Sharan, Aaron Sidford, and Gregory Valiant. “Memory-sample tradeoffs for linear regression with small error”. In: *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing (STOC)*. 2019.

[143] Alexander A Sherstov. “Communication complexity theory: Thirty-five years of set disjointness”. In: *International Symposium on Mathematical Foundations of Computer Science (MFCS)*. 2014.

[144] Vikrant Singhal and Thomas Steinke. “Privately learning subspaces”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2021.

[145] Aleksandrs Slivkins. “Introduction to multi-armed bandits”. In: *Foundations and Trends® in Machine Learning* 12.1-2 (2019), pp. 1–286.

[146] Vaidehi Srinivas et al. “Memory Bounds for the Experts Problem”. In: *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*. 2022.

[147] Jacob Steinhardt. “Robust Learning: Information Theory and Algorithms”. PhD thesis. Stanford University, 2018.

[148] Jacob Steinhardt, Moses Charikar, and Gregory Valiant. “Resilience: A Criterion for Learning in the Presence of Arbitrary Outliers”. In: *Innovations in Theoretical Computer Science Conference (ITCS ’18)*. 2018.

[149] Jacob Steinhardt, Moses Charikar, and Gregory Valiant. “Resilience: A Criterion for Learning in the Presence of Arbitrary Outliers”. In: *9th Innovations in Theoretical Computer Science Conference (ITCS)*. 2018.

[150] Jacob Steinhardt and Percy Liang. “Adaptivity and Optimism: An Improved Exponentiated Gradient Algorithm”. In: *International Conference on Machine Learning (ICML ’14)*. 2014.

[151] Jacob Steinhardt, Gregory Valiant, and Stefan Wager. “Memory, communication, and statistical queries”. In: *Conference on Learning Theory (COLT)*. 2016.

[152] Eiji Takimoto, Akira Maruoka, and Volodya Vovk. “Predicting nearly as well as the best pruning of a decision tree through dynamic programming scheme”. In: *Theoretical Computer Science* 261.1 (2001), pp. 179–209.

[153] Eliad Tsfadia et al. “Friendlycore: Practical differentially private aggregation”. In: *International Conference on Machine Learning (ICML)*. 2022.

[154] John W. Tukey. “A survey of sampling from contaminated distributions”. In: *Contributions to probability and statistics* 2 (1960), pp. 448–485.

[155] David Wajc. “Rounding dynamic matchings against an adaptive adversary”. In: *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing (STOC)*. 2020.

[156] Weiran Wang and Canyi Lu. “Projection onto the capped simplex”. In: *arXiv preprint arXiv:1503.01002* (2015).

[157] Manfred K Warmuth and Dima Kuzmin. “Randomized online PCA algorithms with regret bounds that are logarithmic in the dimension”. In: *Journal of Machine Learning Research* 9.Oct (2008), pp. 2287–2320.

[158] David Woodruff, Fred Zhang, and Samson Zhou. “On robust streaming for learning with experts: algorithms and lower bounds”. In: *Advances in Neural Information Processing Systems (NeurIPS)* (2023).

[159] Emmanouil Zampetakis and Fred Zhang. “Bayesian Strategy-Proof Facility Location via Robust Estimation”. In: *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics (AISTATS)*. 2023.

[160] Banghua Zhu, Jiantao Jiao, and Jacob Steinhardt. “Robust estimation via generalized quasi-gradients”. In: *arXiv preprint arXiv:2005.14073* (2020).

[161] Martin Zinkevich. “Online convex programming and generalized infinitesimal gradient ascent”. In: *International Conference on Machine Learning (ICML ’03)*. 2003.

Appendix A

Technical details for Chapter 2

A.1 Main algorithm description

1. **Input:** Dataset $\boldsymbol{Z}'$ and current estimate $\boldsymbol{x}_t$
2. $\boldsymbol{Z}'_i \leftarrow (\boldsymbol{Z}'_i - \boldsymbol{x}_t)/B$ by scaling each point by $B = \max_i \|\boldsymbol{Z}'_i - \boldsymbol{x}_t\|$.
3. $\theta \leftarrow$ the largest margin θ such that ApproxBregman$(\boldsymbol{Z}', \theta, T)$ does not Fail, where $T = O(\log k/\theta^2)$.
4. **Output:** $\widehat{d} = \frac{B}{10}\theta$.

Figure A.1: Distance estimation—DistEst

A.2 Technical facts

We formally state the statistical guarantee of empirical average and coordinate-wise median-of-means. The former is an application of the Chebyshev's inequality. The latter is folklore but can follow easily from the Lugosi-Mendelson condition by considering the projections onto standard basis vectors.

Lemma A.2.1 (empirical mean)**.** *Let $\delta \in (0,1)$. Given n i.i.d. copies $\boldsymbol{X}_1, \dots, \boldsymbol{X}_n$ of a random vector $\boldsymbol{X} \in \mathbb{R}^d$ with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$, let $\overline{\boldsymbol{\mu}} = \frac{1}{n}\sum_{i=1}^n \boldsymbol{X}_i$. Then with probability at least $1-\delta$,*

$$\|\overline{\boldsymbol{\mu}} - \boldsymbol{\mu}\| \leq \sqrt{\frac{\mathrm{Tr}(\boldsymbol{\Sigma})}{\delta n}}.$$

```
1. Input: Dataset $\boldsymbol{Z}'$ and current estimate $\boldsymbol{x}_t$
2. $\boldsymbol{Z}'_i \leftarrow (\boldsymbol{Z}'_i - \boldsymbol{x}_t)/B$ by scaling each point by $B = \max_i \|\boldsymbol{Z}'_i - \boldsymbol{x}_t\|$.
3. $\theta \leftarrow$ the largest margin $\theta$ such that ApproxBregman$(\boldsymbol{Z}', \theta, T)$ does not
   Fail, where $T = O(\log k/\theta^2)$.
4. $\widehat{\boldsymbol{g}} \leftarrow$ ApproxBregman$(\boldsymbol{Z}', \theta, T)$
5. If $\langle \widehat{\boldsymbol{g}}, \boldsymbol{Z}'_i \rangle \geq 0.1\theta$ for at least $0.5k$ of the $\boldsymbol{Z}'_i$, output $\widehat{\boldsymbol{g}}$; otherwise, output
   $-\widehat{\boldsymbol{g}}$.
```

Figure A.2: Gradient estimation—GradEst

```
1. Input: Dataset $\boldsymbol{X}_1, \boldsymbol{X}_2, \cdots, \boldsymbol{X}_n \in \mathbb{R}^d$
2. Let $k = 3600 \log(1/\delta)$. Divide the data into $2k$ groups.
3. Compute the bucket mean of each group: $\boldsymbol{Z}_1, \boldsymbol{Z}_2, \cdots, \boldsymbol{Z}_{2k} \in \mathbb{R}^d$.
4. Compute the coordinate-wise median-of-means of the second half of
   bucket means:
       $\boldsymbol{x}_0 \leftarrow$ MedianOfMeans$(\{\boldsymbol{Z}_{k+1}, \cdots, \boldsymbol{Z}_{2k}\})$.
5. Prune the first half of bucket means, where $\boldsymbol{Z}$ is the data matrix of
   $\{\boldsymbol{Z}_i\}_{i=1}^{k}7$:
       $\boldsymbol{Z}' \leftarrow$ Prune$(\boldsymbol{Z}, \boldsymbol{x}_0)$.
6. $T_{\mathsf{des}} \leftarrow \Theta(\log d), \eta \leftarrow 1/8000$
7. Run the main descent procedure: $\widehat{\boldsymbol{\mu}} \leftarrow$ Descent$(\boldsymbol{Z}', \boldsymbol{x}_0, T_{\mathsf{des}}, \eta)$, using
   DistEst and GradEst as above.
8. Output: $\widehat{\boldsymbol{\mu}}$
```

Figure A.3: Final algorithm

Lemma A.2.2 (coordinate-wise median-of-means)**.** *Assume the Lugosi-Mendelson condition (Assumption 2.2.1). Let $\{\boldsymbol{Z}_i\}_{i=1}^k$ be the bucket means from n points (with at most $k/200$ contaminated) and $\widehat{\boldsymbol{\mu}}$ be their coordinate-wise median-of-means. Then with probability at least $1-\delta/8$,*

$$\|\widehat{\boldsymbol{\mu}}-\boldsymbol{\mu}\| \le 600\sqrt{d}r_\delta \lesssim \sqrt{\frac{d\|\boldsymbol{\Sigma}\|\log(1/\delta)}{n}}.$$

Our algorithm requires computing an approximation of the top (right) singular vector of a matrix $\boldsymbol{A}\in\mathbb{R}^{m\times n}$. The classic power method is efficient for this task.

Fact A.2.1 (power iteration; see Theorem 3.1 of [23])**.** *Let $\lambda(\boldsymbol{A})=\max_{\boldsymbol{x}\in\mathbb{S}^{n-1}}\|\boldsymbol{A}\boldsymbol{x}\|_2^2$. With probability at least $9/10$, the power method (with random initialization) outputs a unit vector $\boldsymbol{w}$ such that $\|\boldsymbol{A}\boldsymbol{w}\|_2^2 \ge \frac{\lambda(\boldsymbol{A})}{2}$ in $O(\log n)$ iterations. Moreover, each iteration can be performed in $O(mn)$ time.*

The following is a standard bound on binomial tail.

Lemma A.2.3 ([136])**.** *Let $H(n,p)$ be a binomial random variable. Then*

$$\Pr\left(H(n,p)\ge 2np\right)\le \exp\left(-np/3\right).$$

A.3 Omitted proofs from Section 2.3

Proof of Lemma 2.3.1. First, suppose that in some iteration t it holds that $\|\boldsymbol{\mu}-\boldsymbol{x}_t\|\le 14000r_\delta$. Then

$$\frac{1}{21}\|\boldsymbol{\mu}-\boldsymbol{x}_{t^*}\|\le d_{t^*}\le d_t\le 2\|\boldsymbol{\mu}-\boldsymbol{x}_t\|\le 28000r_\delta,$$

so that we may conclude $\|\boldsymbol{\mu}-\boldsymbol{x}_{t^*}\|\le 588000r_\delta$. Second, suppose that in all iterations t it holds that $\|\boldsymbol{\mu}-\boldsymbol{x}_t\|>14000r_\delta$. Then by the update rule with $\eta={}^1\!/_{8000}$,

$$\begin{aligned}\|\boldsymbol{x}_{t+1}-\boldsymbol{\mu}\|^2 &= \|\boldsymbol{x}_t-\boldsymbol{\mu}\|^2+2\eta d_t\langle \boldsymbol{x}_t-\boldsymbol{\mu},\boldsymbol{g}_t\rangle+\eta^2 d_t^2\|\boldsymbol{g}_t\|^2\\ &\le \|\boldsymbol{x}_t-\boldsymbol{\mu}\|^2-\frac{1}{800000}d_t\|\boldsymbol{\mu}-\boldsymbol{x}_t\|+\frac{1}{16000000}\|\boldsymbol{x}_t-\boldsymbol{\mu}\|^2\\ &\le \|\boldsymbol{x}_t-\boldsymbol{\mu}\|^2-\frac{1}{1680000}\|\boldsymbol{\mu}-\boldsymbol{x}_t\|^2+\frac{1}{16000000}\|\boldsymbol{x}_t-\boldsymbol{\mu}\|^2\\ &= \left(1-\frac{179}{336000000}\right)\|\boldsymbol{x}_t-\boldsymbol{\mu}\|^2\end{aligned}$$

Hence, the error bound drops at a geometric rate. The conclusion follows since $\|\boldsymbol{\mu}-\boldsymbol{x}_0\|\le O(\sqrt{kd\|\boldsymbol{\Sigma}\|/n})\le O(\sqrt{d}r_\delta)$. □

A.4 Omitted proof from Section 2.4

First recall that [42] showed that the optimal solution to $\mathcal{M}(\boldsymbol{x}_t, \boldsymbol{Z})$ satisfies the property that θ is a valid distance estimate (definition 2.3.1) and $\boldsymbol{w}$ a gradient estimate (definition 2.3.2).

Lemma A.4.1 (Lemma 1 of [42])**.** *For all $t = 1, 2, \cdots, T$, let $d_t = \theta^*$ be the optimal value of $\mathcal{M}(\boldsymbol{x}_t, \boldsymbol{Z})$. Then $|d_t - \|\boldsymbol{\mu} - \boldsymbol{x}_t\|| \leq 600r_\delta$, so d_t is a distance estimate with respect to $\boldsymbol{x}_t$.*

Lemma A.4.2 (Lemma 2 of [42])**.** *For all $t = 1, 2, \cdots, T$, let $(\theta^*, \boldsymbol{b}^*, \boldsymbol{w}^*)$ be the optimal solution of $\mathcal{M}(\boldsymbol{x}_t, \boldsymbol{Z})$. Then $\boldsymbol{g}_t$ is a distance estimate with respect to $\boldsymbol{x}_t$.*

We now start by proving a generic claim that any reasonably good *bicriteria approximation* of $\mathcal{M}(\boldsymbol{x}_t, \boldsymbol{Z})$ suffices to provide gradient and distance estimates.

Definition A.4.1 (bicriteria solution)**.** *Let θ^* be the optimal value of $\mathcal{M}(\boldsymbol{x}, \boldsymbol{Z})$. We say that $(\theta, \boldsymbol{b}, \boldsymbol{w})$ is a (α, β)-bicriteria solution to $\mathcal{M}(\boldsymbol{x}, \boldsymbol{Z})$ if $\sum_i b_i \geq \alpha k$ and $b_i \langle \boldsymbol{Z}_i - \boldsymbol{x}, \boldsymbol{w} \rangle \geq b_i\theta$ for all i, where $\theta = \beta\theta^*$.*

Lemma A.4.3 (distance estimate)**.** *Let $(\theta, \boldsymbol{b}, \boldsymbol{w})$ be a $(1/10, 1/20)$-bicriteria solution to $\mathcal{M}(\boldsymbol{x}_t, \boldsymbol{Z})$. Then $d_t = \theta$ is a distance estimate with respect to $\boldsymbol{x}_t$.*

Proof of Lemma A.4.3. By Lemma A.4.1, the optimal value θ^* lies in the range

$$[\|\boldsymbol{\mu} - \boldsymbol{x}_t\| - 600r_\delta, \|\boldsymbol{\mu} - \boldsymbol{x}_t\| + 600r_\delta]\,.$$

Moreover, since $\theta^*/20 \leq \theta \leq \theta^*$, we have that

$$\frac{\|\boldsymbol{\mu} - \boldsymbol{x}\|}{20} - 30r_\delta \leq \theta \leq \frac{\|\boldsymbol{\mu} - \boldsymbol{x}\|}{20} + 30r_\delta. \tag{A.4.1}$$

- When $\|\boldsymbol{\mu} - \boldsymbol{x}\| \geq 14000r_\delta$, we get from the inequality (A.4.1) that
$$\frac{\|\boldsymbol{\mu} - \boldsymbol{x}\|}{21} \leq \theta \leq \frac{\|\boldsymbol{\mu} - \boldsymbol{x}\|}{19}.$$
- When $\|\boldsymbol{\mu} - \boldsymbol{x}\| \leq 14000r_\delta$, $\theta \leq 730r_\delta < 28000r_\delta$, again by (A.4.1).

□

Lemma A.4.4 (gradient estimate)**.** *Let $(\theta, \boldsymbol{b}, \boldsymbol{w})$ be a $(1/10, 1/20)$-bicriteria solution to $\mathcal{M}(\boldsymbol{x}_t, \boldsymbol{Z})$. Then $\boldsymbol{g}_t = \boldsymbol{w}$ is a gradient estimate with respect to $\boldsymbol{x}_t$.*

Proof of Lemma A.4.4. Let $\boldsymbol{g}^* = (\boldsymbol{\mu} - \boldsymbol{x}_t)/\|\boldsymbol{\mu} - \boldsymbol{x}_t\|$ be the true gradient. We need to show that $\langle \boldsymbol{g}^*, \boldsymbol{g}_t \rangle \geq 1/20$. On the one hand, by Lemma A.4.1, we have

$$d_t = \theta \geq \frac{1}{20}(\|\boldsymbol{u} - \boldsymbol{x}_t\| - 600r_\delta). \tag{A.4.2}$$

On the other hand, for at least $k/10$ points, we have $\langle \boldsymbol{Z}_i - \boldsymbol{x}_t, \boldsymbol{g}_t \rangle \geq d_t$ and for at least $0.95k$ points, we have $\langle \boldsymbol{Z}_i - \boldsymbol{\mu}, \boldsymbol{g}_t \rangle \leq 600 r_\delta$ by Assumption 2.2.1. Hence, there must be a point $\boldsymbol{Z}_j$ that satisfies both inequalities, so it follows that

$$d_t \leq \langle \boldsymbol{Z}_j - \boldsymbol{x}_t, \boldsymbol{g}_t \rangle = \langle \boldsymbol{Z}_j - \boldsymbol{\mu}, \boldsymbol{g}_t \rangle + \langle \boldsymbol{\mu} - \boldsymbol{x}_t, \boldsymbol{g}_t \rangle \leq 600 r_\delta + \|\boldsymbol{\mu} - \boldsymbol{x}_t\| \langle \boldsymbol{g}^*, \boldsymbol{g}_t \rangle . \tag{A.4.3}$$

Using (A.4.2) and (A.4.3) and rearranging,

$$\langle \boldsymbol{g}^*, \boldsymbol{g}_t \rangle \geq \frac{1}{20} - \frac{630 r_\delta}{\|\boldsymbol{\mu} - \boldsymbol{x}_t\|} \geq \frac{1}{200},$$

where we use $\|\boldsymbol{\mu} - \boldsymbol{x}_t\| \geq 14000 r_\delta$. □

Now we show that the optimal solution to the two-sided relaxation give distance and gradient estimate.

Lemma A.4.5. *Let $(\theta', \boldsymbol{b}', \boldsymbol{w}')$ be an optimal solution of $\mathcal{M}_2(\boldsymbol{x}, \boldsymbol{Z})$. We have that*

(i) the value θ' lies in $[\|\boldsymbol{\mu} - \boldsymbol{x}\| - 600 r_\delta, \|\boldsymbol{\mu} - \boldsymbol{x}\| + 600 r_\delta]$; and

(ii) one of the following two statements must hold, if $\|\boldsymbol{\mu} - \boldsymbol{x}\| \geq 14000 r_\delta$:

- *there is a set $\mathcal{C}$ of at least $0.9k$ points such that $\langle \boldsymbol{Z}_i - \boldsymbol{x}, \boldsymbol{w}' \rangle \geq \theta'$ for all $i \in \mathcal{C}$; or*
- *there is a set $\mathcal{C}$ of at least $0.9k$ points such that $\langle \boldsymbol{Z}_i - \boldsymbol{x}, -\boldsymbol{w}' \rangle \geq \theta'$ for all $i \in \mathcal{C}$.*

Proof of Lemma A.4.5. Let θ be the optimal value of $\mathcal{M}(\boldsymbol{x}, \boldsymbol{Z})$. To prove (i), first recall that Lemma A.4.1 states that $\theta \geq \|\boldsymbol{\mu} - \boldsymbol{x}\| - 600 r_\delta$. Therefore, we get that $\theta' \geq \|\boldsymbol{\mu} - \boldsymbol{x}\| - 600 r_\delta$, as $\theta' \geq \theta$. For the upper bound, assume for the sake of a contradiction that $\theta' > \|\boldsymbol{\mu} - \boldsymbol{x}\| + 600 r_\delta$. Then one side of the hyperplane defined by $\boldsymbol{w}'$ must contain at least 19/40 fraction of points, so let's suppose without loss of generality that

$$\langle \boldsymbol{Z}_i - \boldsymbol{x}, \boldsymbol{w}' \rangle \geq \theta' > \|\boldsymbol{\mu} - \boldsymbol{x}\| + 600 r_\delta \tag{A.4.4}$$

for at least $19k/40$ $\boldsymbol{Z}_i$'s. Also, note that

$$\langle \boldsymbol{Z}_i - \boldsymbol{x}, \boldsymbol{w}' \rangle = \langle \boldsymbol{Z}_i - \boldsymbol{\mu}, \boldsymbol{w}' \rangle + \langle \boldsymbol{\mu} - \boldsymbol{x}, \boldsymbol{w}' \rangle \leq \|\boldsymbol{\mu} - \boldsymbol{x}\| + \langle \boldsymbol{Z}_i - \boldsymbol{\mu}, \boldsymbol{w}' \rangle . \tag{A.4.5}$$

Combining (A.4.4) and (A.4.5), it follows that for at least $19k/40$ $\boldsymbol{Z}_i$'s we have

$$\langle \boldsymbol{Z}_i - \boldsymbol{\mu}, \boldsymbol{w}' \rangle > 600 r_\delta. \tag{A.4.6}$$

On the other hand, consider projections of all bucket means $\boldsymbol{Z}_i$ onto $\boldsymbol{w}'$. Assumption 2.2.1 implies that

$$|\{i : \langle \boldsymbol{w}', \boldsymbol{Z}_i \rangle - \langle \boldsymbol{w}', \boldsymbol{\mu} \rangle \geq 600 r_\delta)\}| \leq 0.05k.$$

This means that at most $k/20$ points satisfy $\langle \boldsymbol{Z}_i - \boldsymbol{\mu}, \boldsymbol{w}' \rangle \geq 600 r_\delta$, contradicting (A.4.6).

To prove (ii), let $S^+ = \{i : \langle \boldsymbol{Z}_i - \boldsymbol{x}, \boldsymbol{w}' \rangle \geq \theta'\}$ and $S^- = \{i : \langle \boldsymbol{Z}_i - \boldsymbol{x}, -\boldsymbol{w}' \rangle \geq \theta'\}$.

Notice that since $\|\boldsymbol{\mu} - \boldsymbol{x}\| \geq 14000 r_\delta$, S^+ and S^- are disjoint. Now let

$$B = \{i : | \langle \boldsymbol{w}', \boldsymbol{Z}_i - \boldsymbol{\mu} \rangle | \leq 600 r_\delta\} = \{i : | \langle \boldsymbol{w}', \boldsymbol{Z}_i - \boldsymbol{x} \rangle - \langle \boldsymbol{w}', \boldsymbol{\mu} - \boldsymbol{x} \rangle | \leq 600 r_\delta\} .$$

By Assumption 2.2.1, $|B| \geq 19k/20$. Consider the two cases.

- If $\langle \boldsymbol{w}', \boldsymbol{\mu} - \boldsymbol{x} \rangle \geq 0$, observe that B must intersect S^+ but not S^-. This implies that $|S^-| \leq k/20$, so $|S^+| \geq 9k/10$, since $|S^+| + |S^-| = 19k/20$ and they are disjoint.
- If $\langle \boldsymbol{w}', \boldsymbol{\mu} - \boldsymbol{x} \rangle < 0$, by the same argument, we have $|S^-| \geq 9k/10$.

□

Next, we show that approximating $\mathcal{M}_2$ in a bicriteria manner achieves a similar guarantee.

Lemma A.4.6. *Let θ^* be the optimal value of $\mathcal{M}(\boldsymbol{x}, \boldsymbol{Z})$ and $\boldsymbol{w}'$ be a unit vector such that for at least $k/8$ of the $\boldsymbol{Z}_i$, we have $|\langle \boldsymbol{w}', \boldsymbol{Z}_i - \boldsymbol{x} \rangle| \geq \theta'$, where $\theta' = 0.1\theta^*$. One of the following two statements must hold if $\|\boldsymbol{\mu} - \boldsymbol{x}\| \geq 14000 r_\delta$.*

- *there is a set $\mathcal{C}$ of at least $0.95k$ points such that $\langle \boldsymbol{Z}_i - \boldsymbol{x}, \boldsymbol{w}' \rangle \geq \theta' - 600 r_\delta$ for all $i \in \mathcal{C}$;*
- *there is a set $\mathcal{C}$ of at least $0.95k$ points such that $\langle \boldsymbol{Z}_i - \boldsymbol{x}, -\boldsymbol{w}' \rangle \geq \theta' - 600 r_\delta$ for all $i \in \mathcal{C}$.*

Proof of Lemma A.4.6. Let $\mathcal{C} = \{i : |\langle \boldsymbol{w}', \boldsymbol{Z}_i - \boldsymbol{\mu} \rangle| \leq 600 r_\delta\}$ be the set of "good" points with respect to direction $\boldsymbol{w}'$. By Assumption 2.2.1, $|\mathcal{C}| \geq 19k/20$. Further, let $S = \{|\langle \boldsymbol{w}', \boldsymbol{Z}_i - \boldsymbol{x} \rangle| \geq \theta'\}$, which we assume has size at least $k/8$. Thus, by pigeonhole principle, there must be a point, say $\boldsymbol{Z}_j$, that is in both sets. There are two cases.

- Suppose $\langle \boldsymbol{w}', \boldsymbol{\mu} - \boldsymbol{x} \rangle \geq 0$. Since $j \in S$ and $\theta^* \geq 13400 r_\delta$ by Lemma A.4.5, we have $|\langle \boldsymbol{w}', \boldsymbol{Z}_i - \boldsymbol{x} \rangle| \geq 1340 r_\delta$. On the other hand, since $j \in \mathcal{C}$,
$$|\langle \boldsymbol{w}', \boldsymbol{Z}_j - \boldsymbol{\mu} \rangle| = |\langle \boldsymbol{w}', \boldsymbol{Z}_j - \boldsymbol{x} \rangle - \langle \boldsymbol{w}', \boldsymbol{\mu} - \boldsymbol{x} \rangle| \leq 600 r_\delta. \tag{A.4.7}$$
Hence, we observe that $\langle \boldsymbol{w}', \boldsymbol{Z}_j - \boldsymbol{x} \rangle \geq \theta' \geq 1340 r_\delta$. By definition of $\mathcal{C}$, all its points cluster around $\boldsymbol{Z}_j$ by an additive factor of $600 r_\delta$.
- Suppose $\langle \boldsymbol{w}', \boldsymbol{\mu} - \boldsymbol{x} \rangle \leq 0$. We get the second case in the claim by the same argument.

□

Finally, we are ready to prove Lemma 2.4.1.

Proof of Lemma 2.4.1. Let's first check the distance estimate (Definition 2.3.1) guarantee.

- If $\|\boldsymbol{\mu} - \boldsymbol{x}\| \geq 14000 r_\delta$, we have
$$\theta' \geq \frac{1}{10}\|\boldsymbol{\mu} - \boldsymbol{x}\| - 60 r_\delta \geq \frac{2}{35}\|\boldsymbol{\mu} - \boldsymbol{x}\|,$$
since $\theta' = 0.1\theta^*$ and $\theta^* \geq \|\boldsymbol{\mu} - \boldsymbol{x}\| - 600 r_\delta$. The upper bound of (2.3.1) obviously holds.
- If $\|\boldsymbol{\mu} - \boldsymbol{x}\| \leq 14000 r_\delta$, we have $\theta' \leq 1460 r_\delta$ by Lemma A.4.5.

For gradient estimate, we appeal to Lemma A.4.6 and get that if $\|\boldsymbol{\mu} - \boldsymbol{x}\| \geq 14000 r_\delta$, then either $(\theta', \boldsymbol{b}', \boldsymbol{w}')$ or $(\theta', \boldsymbol{b}', -\boldsymbol{w}')$ is a $(19/20, 1/20)$-bicriteria approximation of $\mathcal{M}(\boldsymbol{x}, \boldsymbol{Z})$, where $\boldsymbol{b}'$ is the indicator vector of $\mathcal{C}$. Thus, we can apply Lemma A.4.4, and this completes the proof. □

A.5 Omitted proof from Section 2.5

We remark that under Lugosi-Mendelson condition, the assumption $\|\boldsymbol{\mu} - \boldsymbol{x}_0\| \lesssim \sqrt{kd\|\boldsymbol{\Sigma}\|/n}$ can be easily achieved by initializing $\boldsymbol{x}_0$ to be the coordinate-wise median-of-means (with a failure probability at most $\delta/8$).

Lemma A.5.1 (pruning)**.** *Let $\beta = 600\sqrt{kd\|\boldsymbol{\Sigma}\|/n}$, and suppose $\|\boldsymbol{\mu} - \boldsymbol{x}_0\| \leq \beta$. Given the bucket means $\boldsymbol{Z} \in \mathbb{R}^{k\times d}$ such that at most $k/200$ points are contaminated, the algorithm* PRUNE *removes $k/10$ of the points and guarantees that with probability at least $1 - \delta/8$, among the remaining data,*

$$\max_i \|\boldsymbol{Z}_i - \boldsymbol{\mu}\| \leq O(\beta).$$

Further, PRUNE$(\boldsymbol{Z}, \boldsymbol{x}_0)$ *can be implemented in $\widetilde{O}(kd)$ time.*

Proof of Lemma A.5.1. For correctness, consider $\|\boldsymbol{Z}_i - \boldsymbol{\mu}\|$, and by triangle inequality,

$$\|\boldsymbol{Z}_i - \boldsymbol{x}_0\| - \|\boldsymbol{\mu} - \boldsymbol{x}_0\| \leq \|\boldsymbol{Z}_i - \boldsymbol{\mu}\| \leq \|\boldsymbol{Z}_i - \boldsymbol{x}_0\| + \|\boldsymbol{\mu} - \boldsymbol{x}_0\|.$$

Since $\|\boldsymbol{\mu} - \boldsymbol{x}_0\| \leq \beta$ by our assumption,

$$\|\boldsymbol{Z}_i - \boldsymbol{x}_0\| - \beta \leq \|\boldsymbol{Z}_i - \boldsymbol{\mu}\| \leq \|\boldsymbol{Z}_i - \boldsymbol{x}_0\| + \beta. \tag{A.5.1}$$

Let $\mathcal{S}_{\mathsf{good}} = \{i : \|\boldsymbol{Z}_i - \boldsymbol{\mu}\| \leq \beta\}$ and $\mathcal{S}_{\mathsf{bad}} = \{i : \|\boldsymbol{Z}_i - \boldsymbol{\mu}\| \geq 20\beta\}$. It suffices to show that with probability at least $1 - \delta/8$ all the points in $\mathcal{S}_{\mathsf{bad}}$ are removed. We first lower bound the number of good points. Each uncontaminated $\boldsymbol{Z}_i$ is an average of $\lfloor n/k \rfloor$ i.i.d. random vectors. Applying Lemma A.2.1 on estimation error of empirical mean, we obtain that for each uncontaminated i, with probability at least $1 - 1/1000$,

$$\|\boldsymbol{Z}_i - \boldsymbol{\mu}\| \leq \sqrt{1000 \cdot \operatorname{Tr}(\boldsymbol{\Sigma}) k/n} \leq \beta.$$

Therefore, each uncontaminated $\boldsymbol{Z}_i$ is in $\mathcal{S}_{\mathsf{good}}$ with probability at least $1 - 1/1000$. Let H be the number of uncontaminated points not in $\mathcal{S}_{\mathsf{good}}$ and $p = 1/1000$. Since there are at most $k/200$ contaminated points and each uncontaminated point is independent, by a binomial tail bound (Lemma A.2.3)

$$\begin{aligned}
\Pr\left(H \leq 2p \cdot (199/200)k\right) &\geq 1 - \exp\left(-p \cdot (199/200)k/3\right) \\
&\geq 1 - \exp\left(-\log\left(8/\delta\right)\right) \\
&= 1 - \delta/8,
\end{aligned}$$

where we used $k = \lceil 3600 \log(8/\delta) \rceil$. Hence, with probability at least $1-\delta/8$, $\mathcal{S}_{\mathsf{good}}$ contains at least $(399/400)k$ (uncontaminated) points. We condition on this event for the rest of the proof.

Now observe that

$$\|\boldsymbol{Z}_i - \boldsymbol{x}_0\| < \|\boldsymbol{Z}_j - \boldsymbol{x}_0\| \quad \text{for each } j \in \mathcal{S}_{\mathsf{bad}} \text{ and } i \in \mathcal{S}_{\mathsf{good}} \tag{A.5.2}$$

by (A.5.1). Suppose for a contradiction that $j \in \mathcal{S}_{\mathsf{bad}}$ is not removed by line 4. Then it means that $d_j \le d_i$ for $k/10$ of the $\boldsymbol{Z}_i'$'s. By pigeonhole principle, this implies $d_j \le d_i$ for some $i \in \mathcal{S}_{\mathsf{good}}$, since $|S_{\mathsf{good}}| \ge (399/400)k$. This contradicts condition (A.5.2).

Computing the distances takes $O(kd)$ time and sorting takes $O(k \log k)$ time. Thus, the algorithm Prune runs in time $O(kd + k\log k)$ and succeeds with probability at least $1-\delta/8$. □

Pruning allows us to bound the norms of the points $\boldsymbol{Z}_i - \boldsymbol{x}_t$ for each iteration t.

Corollary A.5.1 (scaling and margin)**.** *Suppose* $\|\boldsymbol{\mu} - \boldsymbol{x}\| \le O\left(\sqrt{\|\boldsymbol{\Sigma}\| kd/n}\right)$ *and* $\|\boldsymbol{\mu} - \boldsymbol{x}\| \ge \Omega\left(r_\delta\right)$. *Let* $\mathcal{S}$ *be the pruned dataset of size* $k' \ge 9k/10$ *such that* $\|\boldsymbol{Z}_i - \boldsymbol{\mu}\| \le O\left(\sqrt{\|\boldsymbol{\Sigma}\| kd/n}\right)$ *for each* $i \in \mathcal{S}$. *There exists a scaling factor* B, $\theta > 0$ *and unit vector* $\boldsymbol{w}$ *such that for at least* $4k/5$ *points in* $\mathcal{S}$,

$$\left|\left\langle \tfrac{1}{B}(\boldsymbol{Z}_i - \boldsymbol{x}), \boldsymbol{w} \right\rangle\right| \ge \theta.$$

Further, we have that $1/\theta^2 = O(d)$.

Proof of Corollary A.5.1. Let $B = \max_{i \in \mathcal{S}} \|\boldsymbol{Z}_i - \boldsymbol{x}\|$. Then B is bounded by

$$\|\boldsymbol{Z}_i - \boldsymbol{x}\| \le \|\boldsymbol{Z}_i - \boldsymbol{\mu}\| + \|\boldsymbol{\mu} - \boldsymbol{x}\| \le O\left(\sqrt{\|\boldsymbol{\Sigma}\| kd/n}\right). \tag{A.5.3}$$

By Lemma A.4.5, there exists a unit vector $\boldsymbol{w}$ such that for at least $0.8k$ points in $\mathcal{S}$, $\langle \boldsymbol{Z}_i - \boldsymbol{x}, \boldsymbol{w} \rangle \ge \theta'$ and $\theta' = \Omega(r_\delta)$. Hence, we get that

$$\theta = \Omega\left(\frac{r_\delta}{B}\right) = \Omega\left(\frac{\sqrt{k\|\boldsymbol{\Sigma}\|/n} + \sqrt{\operatorname{Tr}\boldsymbol{\Sigma}/n}}{\sqrt{\|\boldsymbol{\Sigma}\| \cdot kd/n}}\right) = \Omega\left(1/\sqrt{d}\right).$$

□

Proof of Lemma 2.5.1. The lemma follows directly from Lemma A.5.1 and Corollary A.5.1. □

```
1. Input: Buckets means $\boldsymbol{Z}'$, unit vectors $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_T \in \mathbb{R}^d$, margin $\theta$,
2. Round to a single vector: $\boldsymbol{w} = \frac{\boldsymbol{w}'}{\|\boldsymbol{w}'\|}$, where $\boldsymbol{w}' = \sum_{t=1}^T g_t \boldsymbol{w}_t$ and $g_t \sim \mathcal{N}(0,1)$, for $t = 1, \ldots, T$.
3. Repeat until $|\langle \boldsymbol{Z}_i', \boldsymbol{w} \rangle| \geq \frac{1}{10}\theta$ for at least $0.6k'$ of $\boldsymbol{Z}_i'$:
    a) Sample $g_t \sim \mathcal{N}(0,1)$, for $t = 1, \ldots, T$.
    b) Recompute $\boldsymbol{w} = \boldsymbol{w}'/\|\boldsymbol{w}'\|$, where $\boldsymbol{w}' = \sum_{t=1}^T g_t \boldsymbol{w}_t$.
    c) Report FAIL if more than $\Omega(\log(T_{\mathsf{des}}/\delta))$ trials have been performed.
4. Output: $\boldsymbol{w}$
```

Figure A.4: Rounding algorithm—ROUND

A.6 Omitted proofs from Section 2.5

[101] provides a rounding scheme that combines the sequence of vectors produced by APPROXBREGMAN into one vector that satisfies the desired margin bound. The original routine succeeds with constant probability. We simply perform independent trials to boost the rate. We now analyze the algorithm. We cite the following lemma for the guarantee of the rounding algorithm (Algorithm A.4).

Lemma A.6.1 (Lemma 6 of [101]). *Suppose that for at least $\frac{3}{4}$ fraction of $i \in [k']$, it holds that*

$$\sum_{t=1}^{T} \langle \boldsymbol{Z}_i', \boldsymbol{w}_t \rangle^2 \geq \log k'. \tag{A.6.1}$$

Let $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_T$ be the unit vectors satisfying the above condition. Then with constant probability, the vector $\boldsymbol{w}$ in each repetition of the step 3 of the ROUND *algorithm (Algorithm A.4) satisfies $|\langle \boldsymbol{Z}_i, \boldsymbol{w} \rangle| \geq \theta/10$ for at least a 0.45 fraction of $i \in [k']$.*

Now we prove the guarantee of ROUND.

Proof of Lemma 2.5.4. By Lemma A.6.1, it suffices to prove inequality (A.6.1) holds for at least a 3/4 fraction of the points. By the regret analysis (Lemma 2.5.3), the vectors $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_T$ produced during the iterations of Algorithm 2.3 satisfy the hypothesis of Lemma A.6.1. Hence, the guarantee of Lemma A.6.1 holds with constant probability. Moreover, we can test that this guarantee holds in time $O(Tk'd)$. To boost the success probability to $1 - \delta'$ (with $\delta' = \delta/4T_{\mathsf{des}}$), ROUND algorithm performs $O(\log(1/\delta'))$ independent trials. Hence, it reports FAIL with probably at most δ'. Otherwise, by its definition, the output $\boldsymbol{w}$ satisfies desired bound $|\langle \boldsymbol{Z}_i', \boldsymbol{w} \rangle| \geq 0.1\theta$ for $0.45k$ of the points. □

A.7 Full proof of main theorem

Proof of Theorem 2.1.1. Our argument assumes the following global events.

(i) The Lugosi-Mendelson condition (Assumption 2.2.1) holds.

(ii) The initial estimate $\boldsymbol{x}_0$ satisfies $\|\boldsymbol{\mu} - \boldsymbol{x}_0\| \leq 600\sqrt{\|\boldsymbol{\Sigma}\| kd/n}$.

(iii) The pruning step succeeds: $\|\boldsymbol{Z}'_i - \boldsymbol{\mu}\| \leq O\left(\sqrt{\|\boldsymbol{\Sigma}\| kd/n}\right)$

We consider our main algorithm (Algorithm A.3) and first prove the correctness of DistEst and GradEst. Let $\boldsymbol{Z}'$ be defined as in line 2 of DistEst and GradEst. Lemma A.4.5 states that there exists a margin θ^* in $[\|\boldsymbol{\mu} - \boldsymbol{x}\| - 600 r_\delta, \|\boldsymbol{\mu} - \boldsymbol{x}\| + 600 r_\delta]$. When $\|\boldsymbol{\mu} - \boldsymbol{x}_t\| \geq 14000 r_\delta$, we have that for at least $0.8k$ points $\boldsymbol{Z}'_i$ it holds $B \cdot |\langle \boldsymbol{Z}'_i, \boldsymbol{w}^* \rangle| \geq \theta^*$ for some unit vector $\boldsymbol{w}^*$, since the data are scaled by B. Furthermore, when the pruning step succeeds, Assumption 2.2.1 holds. This allows us to apply the key lemma (Lemma 2.5.2).

(i) For GradEst, we use binary search in line 3 to find $\theta = \theta^*/B$. By the guarantee of Lemma 2.5.2, $|\langle \boldsymbol{w}, \boldsymbol{Z}'_i \rangle| \geq \frac{\theta}{10}$ for at least $^k/_8$ of the $\boldsymbol{Z}_i$. It follows that $|\langle \boldsymbol{w}, \boldsymbol{Z}_i - \boldsymbol{x}_t \rangle| \geq \frac{B\theta}{10}$ for at least $^k/_8$ of the $\boldsymbol{Z}_i$. Thus, Lemma 2.4.1 implies that the output $\boldsymbol{g}_t$ is a gradient estimate.

(ii) By the same argument, we apply Lemma 2.4.1 and conclude that $\widehat{d_t}$ of DistEst is a distance estimate.

Finally, we apply Lemma 2.3.1 for the guarantee of Descent.

Next we bound several failure probabilities of the algorithm. The first three correspond to the global conditions.

- By Lemma 2.2.1, the Lugosi-Mendelson condition Assumption 2.2.1 fails with probability at most $\delta/8$.
- By Lemma A.2.2, the coordinate-wise median-of-means error bound fails with probability at most $\delta/8$
- By Lemma 2.5.1, the guarantee of our pruning and scaling procedure fails with probability at most $\delta/8$.
- Conditioned on above, the ApproxBregman satisfies the guarantee of the key lemma (Lemma 2.5.2). The failure probability is at most $\delta/4T_{\mathsf{des}}$ each iteration. We take union bound over all these iterations.

Overall, the failure probability of the entire algorithm (Algorithm A.3) is bounded by δ via union bound.

The runtime follows from Lemma 2.5.2 which claims each iteration takes time $\widetilde{O}(k^2 d)$ and the fact that $T_{\mathsf{des}} = \widetilde{O}(1)$. □

A.8 Interpretation of FHP algorithm [101] as regret minimization

Here, we review the bicriteria approximation algorithm of Karnin *et al.* [101] and show how it can be interpreted in the multiplicative weights update (MWU) framework for regret minimization. Given $\boldsymbol{Z}_1, \ldots, \boldsymbol{Z}_k \in \mathbb{R}^d$ such that $\|\boldsymbol{Z}_i\| \leq 1$, we study the following *furthest hyperplane problem*:

$$\begin{aligned} \text{Find} \quad & \boldsymbol{w} \in S^{d-1} \\ \text{subject to} \quad & |\langle \boldsymbol{Z}_i, \boldsymbol{w} \rangle| \geq r \text{ for } i = 1, \ldots, k, \end{aligned}$$

where we are promised that there does indeed exist a feasible solution $\boldsymbol{w}^*$. Since this problem is (provably) hard (even to approximate) we will settle for *bicriteria approximate* solutions. By this, we simply mean that we require the algorithm to output some $\boldsymbol{w}$ such that $|\langle \boldsymbol{Z}_i, \boldsymbol{w} \rangle| \geq \frac{r}{10}$ for most of the $i \in [k]$. For our applications, the particular constants will not matter much, as long as they are actually constants.

See Algorithm A.5 for a formal description. First we give some intuition and then we sketch the important steps in the analysis.

```
1. Input: $\boldsymbol{Z}_1, \ldots, \boldsymbol{Z}_k \in \mathbb{R}^d$ and iteration count $T \in \mathbb{N}$.
2. Initialize weights: $\boldsymbol{\tau}_1 = \frac{1}{k}(1, \ldots, 1) \in \mathbb{R}^k$.
3. For $t = 1, \ldots, T$, repeat:
    a) Let $\boldsymbol{A}_t$ be the $k \times d$ matrix whose $i$th row is $\sqrt{\boldsymbol{\tau}_t(i)}\boldsymbol{Z}_i$ and $\boldsymbol{w}_t$ be
       the top right unit singular vector of $\boldsymbol{A}_t$.
    b) Set $\boldsymbol{\sigma}_t(i) = |\langle \boldsymbol{Z}_i, \boldsymbol{w}_t \rangle|$.
    c) Reweight: $\boldsymbol{\tau}_{t+1}(i) = \boldsymbol{\tau}_t(i)\eta^{-\boldsymbol{\sigma}_t^2(i)}$ for $i \in [k]$ for an appropriately
       chosen constant $\eta$. In MWU language, $\boldsymbol{\sigma}_t^2$ is the loss vector at time $t$.
    d) Normalize: Let $Z = \sum_{i \in [k]} \boldsymbol{\tau}_{t+1}(i)$ and redefine $\boldsymbol{\tau}_{t+1} \leftarrow \frac{1}{Z}\boldsymbol{\tau}_{t+1}$.
4. Output: $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_T \in S^{d-1}$.
```

Figure A.5: Iterative MWU procedure

Intuition

Because we are promised that $\boldsymbol{w}^*$ exists, averaging the constraints yields:

$$\frac{1}{k}\sum_{i=1}^{k}\langle \boldsymbol{Z}_i, \boldsymbol{w}^*\rangle^2 \geq r^2.$$

Note that if we define $\boldsymbol{A}_1$ as in Algorithm A.5, then the definition of singular vector tells us that:

$$\max_{\boldsymbol{w}\in S^{d-1}} \|\boldsymbol{A}_1\boldsymbol{w}\|^2 = \frac{1}{k}\sum_{i=1}^{k}\langle \boldsymbol{Z}_i, \boldsymbol{w}\rangle^2 \geq \frac{1}{k}\sum_{i=1}^{k}\langle \boldsymbol{Z}_i, \boldsymbol{w}^*\rangle^2 \geq r^2.$$

Thus, $\boldsymbol{w}_1$, the top singular vector as defined in Algorithm A.5, satisfies the constraints *on average*. It could be the case that $\langle \boldsymbol{Z}_4, \boldsymbol{w}_1\rangle^2 \gg r^2$ but $\langle \boldsymbol{Z}_i, \boldsymbol{w}\rangle^2 \ll r^2$ for all $i \neq 4$. To fix this issue, we would simply down-weight $\boldsymbol{Z}_4$ in the next iteration, so that $\boldsymbol{w}_2$ aligns more with $\boldsymbol{Z}_i$ for $i \neq 4$. We repeat this several times, with each $\boldsymbol{w}_t$ improving upon $\boldsymbol{w}_{t-1}$.

At the end, the algorithm produces a collection of vectors $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_T$ which each satisfy a certain property. While it seems natural to just output $\boldsymbol{w}_T$ as the final answer, it turns out that this will not work. Instead, we need to apply a randomized rounding procedure to extract a single vector $\boldsymbol{w}$ from $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_T$.

Analysis

Lemma A.8.1. *When Algorithm A.5 terminates after $T = O(\frac{\log k}{r^2})$ iterations, for every $i \in [k]$ it holds that:*

$$\sum_{t=1}^{T}\langle \boldsymbol{Z}_i, \boldsymbol{w}_t\rangle^2 \geq \frac{\log k}{\log \eta}.$$

Proof. Algorithm A.5 is simply the MWU algorithm with the experts corresponding to the k constraints and the loss of expert i at time t being $\boldsymbol{\sigma}_t^2(i)$. Using the regret guarantee from Theorem 2.1 in [13] with respect to the fixed expert $\boldsymbol{e}_i$ and step size η:

$$\sum_{t=1}^{T}\left\langle \boldsymbol{\tau}_t, \boldsymbol{\sigma}_t^2\right\rangle - (1+\eta)\sum_{t=1}^{T}\left\langle \boldsymbol{e}_i, \boldsymbol{\sigma}_t^2\right\rangle \leq \frac{\log k}{\eta}. \tag{A.8.1}$$

Note that

$$\sum_{t=1}^{T}\left\langle \boldsymbol{e}_i, \boldsymbol{\sigma}_t^2\right\rangle = \sum_{t=1}^{T}\langle \boldsymbol{Z}_i, \boldsymbol{w}_t\rangle^2$$

and

$$\begin{aligned}
\sum_{t=1}^{T} \left\langle \boldsymbol{\tau}_t, \boldsymbol{\sigma}_t^2 \right\rangle &= \sum_{t=1}^{T} \sum_{i=1}^{k} \boldsymbol{\tau}_t(i) \boldsymbol{\sigma}_t^2(i) \\
&= \sum_{t=1}^{T} \sum_{i=1}^{k} \boldsymbol{\tau}_t(i) \left\langle \boldsymbol{Z}_i, \boldsymbol{w}_t \right\rangle^2 && \text{(by definition of the algorithm)} \\
&\geq \sum_{t=1}^{T} \sum_{i=1}^{k} \boldsymbol{\tau}_t(i) \left\langle \boldsymbol{Z}_i, \boldsymbol{w}^* \right\rangle^2 && \text{(since } \boldsymbol{w}^* \text{ is the top eigenvector)} \\
&\geq \sum_{t=1}^{T} \sum_{i=1}^{k} \boldsymbol{\tau}_t(i) r^2 \\
&= T r^2.
\end{aligned}$$

Substituting into and simplifying the regret formula and taking $\eta = 1/3$ gives the claim. □

Given the previous lemma, we can just apply the rounding algorithm as a black-box to the output of Algorithm A.5.

Lemma A.8.2 ([101]). *Let $\alpha \in (0,1)$ and $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_T$ be unit vectors satisfying the conclusion of the previous lemma. Then with probability at least $1/147$, the output $\boldsymbol{w}$ of the Rounding Algorithm A.4 satisfies $|\langle \boldsymbol{Z}_i, \boldsymbol{w} \rangle| \geq \alpha r$ for at least a $1 - 3\alpha$ fraction of $i \in [k]$.*

Appendix B

Technical details for Chapter 3

B.1 Technical Lemmas and Proofs

Spectral Signatures

Lemma B.1.1. *Let $\{x_i\}_{i=1}^n$ be n points in $\mathbb{R}^d$. Suppose there exists $\nu \in \mathbb{R}^d$ and a set of good weights $w \in \mathcal{W}_{n,\varepsilon}$ such that*

$$\sum_{i=1}^{n} w_i \left(x_i - \nu\right)\left(x_i - \nu\right)^\top \preceq \lambda I. \tag{B.1.1}$$

for some $\lambda > 0$. Then for any $w' \in \mathcal{W}_{n,\varepsilon}$,

$$\|\nu - \nu(w')\| \leq \frac{1}{1-\sqrt{2\varepsilon}}\left(\sqrt{\lambda} + \sqrt{2\varepsilon\lambda} + \sqrt{2\varepsilon\|M(w')\|}\right), \tag{B.1.2}$$

where $\nu(w') = \sum_i w'_i x_i$ and $M(w') = \sum_i w'_i(x_i - \nu(w'))(x_i - \nu(w'))^\top$.

The lemma and its proof strategy is similar to the spectral signature lemma in robust statistics and is now somewhat standard in the literature; see, *e.g.*, [64, 117].

Proof. To bound $\|\nu - \nu(w')\|$, we note

$$\|\nu(w') - \nu\|_2^2 = \sum_i w'_i \left\langle \nu(w') - \nu, x_i - \nu \right\rangle. \tag{B.1.3}$$

In bounding this sum, we may assume without loss of generality that $w'_i > 0$ for all i. Now observe that we can decompose the sum as

$$\begin{aligned}\sum_i w'_i \left\langle \nu(w') - \nu, x_i - \nu \right\rangle = \sum_i w_i \left\langle \nu(w') - \nu, x_i - \nu \right\rangle + \sum_{i: w_i > w'_i} \left(w'_i - w_i\right)\left\langle \nu(w') - \nu, x_i - \nu \right\rangle \\ + \sum_{i: w'_i > w_i} \left(w'_i - w_i\right)\left\langle \nu(w') - \nu, x_i - \nu \right\rangle.\end{aligned} \tag{B.1.4}$$

We bound the three terms respectively as follows.

(i) For the first term, by Cauchy-Schwarz,

$$\sum_i w_i \langle \nu(w') - \nu, x_i - \nu \rangle = \langle \nu(w') - \nu, \nu(w) - \nu \rangle \le \|\nu(w') - \nu\| \cdot \|\nu(w) - \nu\|.$$

By Jensen's inequality and the spectral centrality assumption, we have for all unit u,

$$\langle \nu(w) - \nu, u \rangle^2 = \left\langle \sum_i w_i x_i - \nu, u \right\rangle^2 \le \sum_i w_i \langle x_i - \nu, u \rangle^2 \le \lambda.$$

Thus, $\|\nu(w) - \nu\| \le \sqrt{\lambda}$.

(ii) For the second term, let $\alpha_i = w_i' - w_i$. Then note that if $w_i' < w_i$,

$$\left| \frac{\alpha_i}{w_i} \right| = \left| \frac{w_i'}{w_i} - 1 \right| \le 1 \tag{B.1.5}$$

Hence,

$$\left(\sum_{i:w_i > w_i'} \alpha_i \langle \nu(w') - \nu, x_i - \nu \rangle \right)^2 \le \left(\sum_{i:w_i > w_i'} \frac{1}{w_i} \alpha_i^2 \right) \cdot \sum_{i:w_i > w_i'} w_i \langle \nu(w') - \nu, x_i - \nu \rangle^2 \tag{B.1.6}$$

$$\le \left(\sum_{i:w_i > w_i'} \frac{1}{w_i} \alpha_i^2 \right) \cdot \lambda \cdot \|\nu(w') - \nu\|_2^2 \tag{B.1.7}$$

$$\le \left(\sum_{i:w_i > w_i'} |\alpha_i| \right) \cdot \lambda \cdot \|\nu(w') - \nu\|_2^2 \tag{B.1.8}$$

$$\le 2\varepsilon\lambda \|\nu(w') - \nu\|_2^2 \tag{B.1.9}$$

where (B.1.6) follows from Cauchy-Schwarz, (B.1.7) follows from the spectral centrality assumption, and (B.1.8) follows from (B.1.5).

(iii) For the third term, again let $\alpha_i = w_i' - w_i$. Similarly, if $w_i' > w_i$,

$$\left| \frac{\alpha_i}{w_i'} \right| = \left| \frac{w_i}{w_i'} - 1 \right| \le 1 \tag{B.1.10}$$

It follows that

$$\left(\sum_{i:w_i'>w_i} \alpha_i \left\langle \nu(w') - \nu, x_i - \nu \right\rangle\right)^2 \le \left(\sum_{i:w_i'>w_i} \frac{1}{w_i'}\alpha_i^2\right) \cdot \sum_{i:w_i'>w_i} w_i' \left\langle \nu(w') - \nu, x_i - \nu \right\rangle^2 \tag{B.1.11}$$

$$\le \left(\sum_{i:w_i'>w_i} \frac{1}{w_i'}\alpha_i^2\right) \cdot \sum_{i:w_i'>w_i} w_i' \left\langle \nu(w') - \nu, x_i - \nu \right\rangle^2 \tag{B.1.12}$$

$$\le \left(\sum_{i:w_i'>w_i} |\alpha_i|\right) \cdot \sum_{i:w_i'>w_i} w_i' \left\langle \nu(w') - \nu, x_i - \nu \right\rangle^2 \tag{B.1.13}$$

$$\le 2\varepsilon \cdot \sum_i w_i' \left\langle \nu(w') - \nu, x_i - \nu \right\rangle^2 \tag{B.1.14}$$

For the sum, we have

$$\sum_i w_i' \left\langle \nu(w') - \nu, x_i - \nu \right\rangle^2 = \sum_i w_i' \left\langle \nu(w') - \nu, x_i - \nu(w') \right\rangle^2 + \|\nu(w') - \nu\|^4 \tag{B.1.15}$$

$$\le \|M(w')\|_2 \cdot \|\nu(w') - \nu\|^2 + \|\nu(w') - \nu\|^4. \tag{B.1.16}$$

Putting everything together and rearranging finishes the proof. □

Lemma B.1.2. *Let $\{x_i\}_{i=1}^n$ be n points. Suppose there exists a subset $G \subset [n]$ of size $(1-\varepsilon)$ such that $\frac{1}{|G|}\sum_{i\in G}(x_i - \mu_G)(x_i - \mu_G)^\top \preceq \lambda I$ for some $\lambda > 0$, where $\mu_G = \frac{1}{|G|}\sum_{i\in G} x_i$. Then for any $w \in \mathcal{W}_{n,\varepsilon}$,*

$$\|\mu_G - \mu(w)\| \le \frac{1}{1-2\varepsilon}\left(\sqrt{2\varepsilon\lambda} + \sqrt{\varepsilon\|M(w)\|}\right). \tag{B.1.17}$$

Proof. The proof follows from the same argument of Lemma B.1.1, with $\nu = \mu_G$. □

A KL Divergence Bound

Lemma B.1.3. *Let $p \in \mathcal{W}_{n,\varepsilon}$ and q be the uniform distribution over n points. Then $KL(p||q) \le 5\varepsilon$.*

Proof. The lemma follows from direct calculations. By definition of KL divergence,

$$\begin{aligned}\mathrm{KL}(p||q) &= \sum_i p_i \log \frac{p_i}{q_i} \\ &= \sum_i p_i \log(np_i) \\ &\leq \sum_i \frac{1}{(1-\varepsilon)n} \log \frac{1}{(1-\varepsilon)} \\ &= \frac{1}{1-\varepsilon} \log \frac{1}{1-\varepsilon} \\ &\leq 5\varepsilon.\end{aligned}$$

where the last inequality holds when $0 < \varepsilon \leq 1/2$. □

Proof of Lemma 3.3.2

Proof of Lemma 3.3.2. We show that there exists a ball of radius $\sqrt{d\lambda/\varepsilon}$ that contains at least $(1-3\varepsilon)n$ points. Note that the spectral centrality condition $\sum_{i=1}^n w_i (x_i - \nu)(x_i - \nu)^\top \preceq \lambda I$ implies that

$$\sum_{i=1}^n \mathrm{Tr}\left(w_i (x_i - \nu)(x_i - \nu)^\top\right) \leq d\lambda.$$

By the cyclic property of trace, we get

$$\sum_{i=1}^n w_i \|x_i - \nu\|^2 \leq d\lambda.$$

Therefore, by Markov's inequality,

$$\Pr_{i \sim w}\left(\|x_i - \nu\|^2 \geq d\lambda/\varepsilon\right) \leq \varepsilon, \tag{B.1.18}$$

where $i \sim w$ denotes i drawn from the discrete distribution defined by w. Observe that since $\mathcal{W}_{n,\varepsilon}$ is the convex hull of all uniform distributions over a subset of size $(1-\varepsilon)n$, we have $\|w - \mathcal{U}_n\|_1 \leq 2\varepsilon$. Thus, $\mathrm{TV}(w, \mathcal{U}_n) \leq \varepsilon$. Hence, using the definition of total variation distance, (B.1.18) implies that

$$\Pr_{i \sim \mathcal{U}_n}\left(\|x_i - \nu\|^2 \geq d\lambda/\varepsilon\right) \leq 2\varepsilon, \tag{B.1.19}$$

as desired. □

B.2 Extension to sub-gaussian distributions

We now consider a variant of the filter algorithm (Algorithm 1) analyzed in Section 3.3. The difference is that instead of fixing the step size to be $\eta = 1/2$, we set it as ε. That is, we will perform the multiplicative update less aggressively when there are few bad points. In addition, we require a stronger approximation for the largest eigenvector computation. This increases the the run-time by an $O(\text{poly}(1/\varepsilon))$ factor. For technical reasons, we also ask the algorithm to stop early if the weighted covariance has been reduced to a desired value. Formally, the algorithm is described by the pseudo-code below (Algorithm 7).

Algorithm 7: Multiplicative weights for sub-gaussian robust mean estimation

```
Input: A set of points $\{x_i\}_{i=1}^n$, an iteration count $T$, and parameter $\rho, \delta$
Output: A set of weights $w \in \mathcal{W}_{n,\varepsilon}$.

1 Let $w^{(1)} = \frac{1}{n}\mathbb{1}_n$.
2 For $t$ from 1 to $T$
3     Let $\nu^{(t)} = \sum_i w_i^{(t)} x_i$, $M^{(t)} = \sum_i w_i^{(t)}(x_i - \nu^{(t)})(x_i - \nu^{(t)})^T$.
4     Compute $v^{(t)} = \textsc{ApproxTopEigenvector}(M^{(t)}, 1-\varepsilon^2, \delta/T)$.
5     If $\lambda^{(t)} = v^{(t)\top} M^{(t)} v^{(t)} \le 1$, return $w^{(t)}$.
6     Compute $\tau_i^{(t)} = \langle v^{(t)}, x_i - \nu^{(t)} \rangle^2$.
7     Set $w_i^{(t+1)} \leftarrow w_i^{(t)} \left(1 - \varepsilon \tau_i^{(t)}/\rho\right)$ for each $i$.
8     Project $w^{(t+1)}$ onto the set of good weights $\mathcal{W}_{n,\varepsilon}$ (under KL divergence).
9 Return $w^{(t^*)}$, where $t^* = \arg\min_t \|M^{(t)}\|$.
```

First, we need a stronger spectral signature lemma.

Lemma B.2.1 ([64]). *Let $S = \{x_i\}_{i=1}^n$ be an ε-corrupted set of n samples from a sub-gaussian distribution over $\mathbb{R}^d$, with mean μ and identity covariance. Suppose $n \ge \widetilde{\Omega}(d/\varepsilon^2)$. If $\|M(w)\| \le 1 + \lambda$, for some $\lambda \ge 0$, then for any $w \in \mathcal{W}_{n,2\varepsilon}$,*

$$\|\mu - \mu(w)\| \le \frac{1}{1-\varepsilon}\left(\sqrt{\varepsilon\lambda} + C\varepsilon\sqrt{\log(1/\varepsilon)}\right),$$

for some universal constant $C > 0$.

Moreover, we assume that for all $w \in \mathcal{W}_{n,2\varepsilon}$ we have

$$\left\|\sum_{i \in G} w_i (x_i - \mu)(x_i - \mu)^\top - I\right\| \le \lambda = O(\varepsilon \log(1/\varepsilon)). \tag{B.2.1}$$

This condition holds with high probability over the draws of samples [60].

Lemma B.2.2 (analysis of sub-gaussian filter)**.** *Let ε be a sufficiently small constant and $\{x_i\}_{i=1}^n$ be n points in $\mathbb{R}^d$. Assume the following (deterministic) conditions hold.*

(i) There exists $\nu \in \mathbb{R}^d$ and $w \in \mathcal{W}_{n,\varepsilon}$ such that

$$\left\| \sum_{i=1}^n w_i \left(x_i - \nu\right) \left(x_i - \nu\right)^\top \right\| \leq 1 + O\left(\varepsilon \log\left(1/\varepsilon\right)\right). \tag{B.2.2}$$

(ii) If $\|M(w)\| \leq 1 + \lambda$, for some $\lambda \geq 0$, then for any $w \in \mathcal{W}_{n,\varepsilon}$,

$$\|\nu - \mu(w)\| \leq \frac{1}{1-\varepsilon} \left(\sqrt{\varepsilon\lambda} + C\varepsilon\sqrt{\log(1/\varepsilon)} \right), \tag{B.2.3}$$

Then, given $\{x_i\}_{i=1}^n$, a failure rate δ and ρ such that $\rho \geq \tau_i^{(t)}$ for all i and t, Algorithm 7 finds $w' \in \mathcal{W}_{n,\varepsilon}$ such that

$$\|M(w')\| \leq 1 + O\left(\varepsilon \log\left(1/\varepsilon\right)\right), \tag{B.2.4}$$

with probability at least $1-\delta$.

The algorithm terminates in $T = O(\rho/\varepsilon)$ iterations. Further, if $T = O(poly(n,d))$, then each iteration takes $\widetilde{O}(nd \log\left(1/\delta\right)/\varepsilon^2)$ time.

Proof of Lemma B.2.2. If the algorithm gets stopped early (at Line 5), then it means that

$$\|M^{(t)}\| \leq \lambda^{(t)} / \left(1 - \varepsilon^2\right) \leq 1 / \left(1 - \varepsilon^2\right) \leq 1 + O(\varepsilon^2),$$

since $v^{(t)}$ is a $(1-\varepsilon^2)$ approximate largest eigenvector of $M^{(t)}$. Hence, in this case, we immediately achieves the goal (B.2.4).

Now assume the algorithm did not stop early and so $\|M^{(t)}\| > 1$ for all t. Then we have

$$\sum_i w_i^{(t)} \tau_i^{(t)} = \sum_i w_i^{(t)} \left\langle v^{(t)}, x_i - \nu^{(t)} \right\rangle^2 = v^{(t)\top} M^{(t)} v^{(t)} \geq \left(1 - \varepsilon^2\right) \left\| M^{(t)} \right\|_2, \tag{B.2.5}$$

for all t. Since the step size $\varepsilon < 1/2$ and $\rho \geq \tau_i^{(t)}$ for all i,t by assumption, we can apply the regret bound of MWU (Lemma 3.3.4) and conclude that for w that satifies assumption (B.2.2),

$$\frac{1-\varepsilon^2}{T} \sum_{t=1}^T \left\| M^{(t)} \right\|_2 \leq \frac{1}{T} \sum_{t=1}^T \left\langle w^{(t)}, \tau^{(t)} \right\rangle \leq (1+\varepsilon) \frac{1}{T} \sum_{t=1}^T \left\langle w, \tau^{(t)} \right\rangle + \frac{\rho \cdot \mathrm{KL}(w \| w^{(1)})}{T\varepsilon}. \tag{B.2.6}$$

We now focus on bounding $\frac{1}{T}\sum_{t=1}^T \left\langle w, \tau^{(t)} \right\rangle$.

Claim B.2.1. *In the setting above, we have*

$$\frac{1}{T} \sum_{t=1}^T \left\langle w, \tau^{(t)} \right\rangle \leq 1 + O\left(\varepsilon \log(1/\varepsilon)\right) + \frac{2\varepsilon}{(1-\varepsilon)^2} \frac{1}{T} \sum_{t=1}^T \left\| M^{(t)} \right\| - \frac{2\varepsilon}{(1-\varepsilon)^2}$$

Proof. Note that

$$\begin{aligned}\frac{1}{T}\sum_{t=1}^{T}\left\langle w,\tau^{(t)}\right\rangle &= \frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{n} w_i \left\langle x_i-\nu^{(t)},v^{(t)}\right\rangle \\ &= \frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{n} w_i \left(\left\langle x_i-\nu,v^{(t)}\right\rangle^2+\left\langle \nu-\nu^{(t)},v^{(t)}\right\rangle^2\right) \\ &\le 1+O\left(\varepsilon\log(1/\varepsilon)\right)+\frac{1}{T}\sum_{t=1}^{T}\left\langle \nu-\nu^{(t)},v^{(t)}\right\rangle^2 \qquad \text{(B.2.7)} \\ &\le 1+O\left(\varepsilon\log(1/\varepsilon)\right)+\frac{1}{T}\sum_{t=1}^{T}\left\|\nu-\nu^{(t)}\right\|_2^2, \qquad \text{(B.2.8)}\end{aligned}$$

where (B.2.7) follows from the assumption (B.2.4). Now we apply assumption (B.2.3) to bound $\left\|\nu-\nu^{(t)}\right\|_2^2$. Since we may assume $\|M^{(t)}\|\ge 1$ by the early stopping of Line 5, we have

$$\begin{aligned}\left\|\nu-\nu^{(t)}\right\|^2 &\le \frac{2}{(1-\varepsilon)^2}\left(\varepsilon\left(\left\|M^{(t)}\right\|-1\right)+C^2\varepsilon^2\log(1/\varepsilon)\right) \\ &= \frac{2\varepsilon}{(1-\varepsilon)^2}\left\|M^{(t)}\right\|-\frac{2\varepsilon}{(1-\varepsilon)^2}+O(\varepsilon^2\log(1/\varepsilon)).\end{aligned}$$

Substituting the bound back into (B.2.8) completes the proof. □

Using Claim B.2.1, the KL bound (Lemma B.1.3) and (B.2.6), we have

$$\frac{1-\varepsilon^2}{T}\sum_{t=1}^{T}\left\|M^{(t)}\right\|_2 \le \frac{2(1+\varepsilon)\varepsilon}{(1-\varepsilon)^2}\frac{1}{T}\sum_{t=1}^{T}\left\|M^{(t)}\right\|+1-\frac{2(1+\varepsilon)\varepsilon}{(1-\varepsilon)^2}+O(\varepsilon\log(1/\varepsilon))+\frac{5\rho}{T}.$$

For sufficiently small ε, we rearrange and divide through to obtain

$$\frac{1}{T}\sum_{t=1}^{T}\left\|M^{(t)}\right\|_2 \le 1+O(\varepsilon\log(1/\varepsilon))+O(\varepsilon)+\frac{O(\rho)}{T}.$$

Setting $T=O(\rho/\varepsilon)$ completes the correctness proof. Finally, the per-iteration cost follows from the run-time of using power method to approximate the largest eigenvector. □

Using the lemma we can prove our main theorem.

Theorem B.2.1 (sub-gaussian robust mean estimation, [60]). *Let $S=\{x_i\}_{i=1}^n$ be an ε-corrupted set of n samples from a sub-gaussian distribution over $\mathbb{R}^d$, with mean μ and identity covariance. Suppose $n\ge\widetilde{\Omega}(d/\varepsilon^2)$. Given S, there is an algorithm that outputs $\widehat{\mu}$ such that $\|\widehat{\mu}-\mu\|\le O\left(\varepsilon\log\left(1/\varepsilon\right)\right)$ with high constant probability. The algorithm runs in time $\widetilde{O}\left(nd^2/\varepsilon^3\right)$*

Proof. Let $\delta = 0.01$. We apply Algorithm 7 with a simple pruning procedure as a pre-processing. By standard concentration of sub-gaussian random vectors, with high constant probability, $\|x_i - \mu\| \le r = O(\sqrt{d \log n})$ for all $i \in G$. Hence, we apply $\textsc{Prune}(S, r, \delta)$, and by Lemma 3.3.3 it guarantees to terminate in $O(nd)$ time and removes at most εn (bad) points.

We feed the remaining (at least) $(1-\varepsilon)n$ points $R \supseteq G$ into Algorithm 7 with $\rho = r^2$. Notice that $\frac{1}{(1-\varepsilon)(1-\varepsilon)} \le \frac{1}{1-2\varepsilon}$ for $\varepsilon \le 1/2$, so assumptions (i)-(ii) of Lemma B.2.2 are satisfied by the claim of (B.2.1) and Lemma B.2.1, respectively.

It then follows from Lemma B.2.2 that Algorithm 7 outputs $w' \in \mathcal{W}_{|R|,\varepsilon}$ such that

$$\left\| \sum_{i \in R} (x_i - \mu(w'))(x_i - \mu(w'))^\top \right\| \le 1 + O\left(\varepsilon \log(1/\varepsilon)\right),$$

where $\mu(w') = \sum_{i \in R} w'_i x_i$. Let $w''_i = w'_i$ if $i \in R$ and $w''_i = 0$ otherwise. We obtain $w'' \in \mathcal{W}_{n,2\varepsilon}$ such that $\|M(w'')\| \le 1 + O\left(\varepsilon \log\left(1/\varepsilon\right)\right)$. Applying the spectral signature (Lemma B.2.1) proves that $\mu(w'')$ attains the desired estimation error. Moreover, the run-time simply follows from Lemma B.2.2. $\square$

B.3 Sample reweighing via Matrix Multiplicative Update

We now show that the spectral sample reweighing problem (Definition 3.3.1) can be solved in near linear time via a matrix multiplicative update scheme from the recent work of [64], analyzed there for the robust mean estimation setting. Our analysis will closely resemble the arguments therein.

Theorem B.3.1. *Let $\{x_i\}_{i=1}^n$ be n points in $\mathbb{R}^d$. Suppose there exists $\nu \in \mathbb{R}^d$ and $w^* \in \mathbb{R}_n$ such that $|w^*| = 1 - \varepsilon$, $\|w^*\|_\infty \le 1/n$ and $\sum_{i=1}^n w_i^* \left(x_i - \nu\right)\left(x_i - \nu\right)^\top \preceq \lambda I$ for some $\lambda > 0$ and a sufficiently small ε. Then, given $\{x_i\}_{i=1}^n, \lambda$, the squared diameter ρ of the points and a failure rate δ, there is a matrix multiplicative weights-based algorithm (Algorithm 8) that, with probability at least $1-\delta$, finds $w \in \mathcal{W}_{n,\varepsilon}$ and $\nu' \in \mathbb{R}^d$ such that*

$$\sum_{i=1}^n w_i \left(x_i - \nu'\right)\left(x_i - \nu'\right)^\top \preceq O(\lambda) I.$$

Further, the algorithm terminates in $O(\log(\rho/\lambda))$ iterations, where ρ is the squared diameter of the input points $\{x_i\}_{i=1}^n$, and each iteration can be implemented in $\widetilde{O}(nd \log(1/\delta))$ time.

Remark B.3.1. *In the following, we will consider an idealized version of the algorithm and omit the detail of implementing the numerical linear algebra primitives in $\widetilde{O}(nd \log(1/\delta))$ time each iteration. The exact details can be found in [64].*

The algorithm is based on the matrix multiplicative weights update. For a sequence of PSD matrices $M_1 \succeq M_2 \succeq \cdots \succeq M_{t-1}$, we will apply the matrix multiplicative weight (MMW) update, given by

$$\mathsf{MMW}(M_0, M_1, \cdots, M_{t-1}) = \exp\left(\frac{1}{\|M_0\|_2}\sum_{k=1}^{t-1} M_k\right) / \operatorname{tr}\exp\left(\frac{1}{\|M_0\|_2}\sum_{k=1}^{t-1} M_k\right). \tag{B.3.1}$$

For technical reasons, we will not maintain a set of weights that is a probability distribution throughout. Instead, we will initiate from uniform weights and monotonically downweight each point. The key invariant we will maintain is the following, which is a weighted extension of the notion of "mostly-good weights" from [64].

Definition B.3.1 (mostly-good weight)**.** *Suppose that $w^* \in [0,1]^n$ satisfies $\|w^*\|_\infty \leq 1/n$. The set of mostly-good weight vectors (with respect to w^*) is*

$$\mathcal{C}(w^*) = \left\{ w \in \mathbb{R}^n : 0 \leq w_i \leq \frac{1}{n} \quad \textit{and} \quad \sum_{i=1}^n w_i^* \left(\frac{1}{n} - w_i\right) \leq \sum_{i=1}^n \left(\frac{1}{n} - w_i^*\right)\left(\frac{1}{n} - w_i\right) \right\}$$

Lemma B.3.1. *Suppose that $w^* \in [0,1]^n$ satisfies $\|w^*\|_\infty \leq 1/n$ and $|w^*| = 1 - \varepsilon$. Then for any mostly-good weight $w \in \mathcal{C}(w^*)$ (with respect to w^*), we have that $|w| \geq 1 - 2\varepsilon$.*

Proof. By rearranging the condition of mostly-good weight, we get that

$$\frac{1}{n}\sum_{i=1}^n \frac{1}{n} - w_i \geq 2\sum_{i=1}^n w_i^*\left(\frac{1}{n} - w_i\right).$$

Since $\sum_{i=1}^n w_i^* = 1 - \varepsilon$, it follows that

$$\frac{1}{n} - \frac{1}{n}\sum_{i=1}^n w_i \geq \frac{2 - 2\varepsilon}{n} - 2\sum_{i=1}^n w_i^* w_i.$$

By assumption, $w_i^* \leq \frac{1}{n}$, so

$$\frac{1}{n} - \frac{1}{n}\sum_{i=1}^n w_i \geq \frac{2 - 2\varepsilon}{n} - \frac{2}{n}\sum_{i=1}^n w_i.$$

Multiplying n on both sides and rearranging we get $\sum_{i=1}^n w_i \geq 1 - 2\varepsilon$. □

A crucial subroutine we use is a deterministic down-weighting scheme, from [64], that maintains the mostly-good property of the input weights.

Lemma B.3.2 (1D Filter [64]). *Let $\eta \in (0, 1/2)$, let $b \geq 2\eta$, and let $w_1, \dots, w_m$ and $\tau_1, \dots, \tau_m$ be non-negative numbers so that $\sum_{i=1}^m w_i \leq 1$. Let $\tau_{\max} = \max_{i \in [m]} \tau_i$. Suppose there exists w^* such that $\|w\|_\infty \leq \frac{1}{n}$*

$$\sum_{i=1}^n w_i^* w_i \tau_i \leq \eta\sigma \text{ , where } \sigma = \frac{1}{n}\sum_{i=1}^n w_i \tau_i \text{ .}$$

Then $\textsc{1DFilter}(w, \tau, b)$ *runs in time $O((1 + \log \frac{\tau_{\max}}{b\sigma})m)$ and outputs $0 \leq w' \leq w$ so that*

- $\sum w_i^*(w_i - w_i') \leq \sum(1/n - w_i^*)(w_i - w_i')$, *and*
- $\frac{1}{n}\sum_{i=1}^n w_i' \tau_i \leq b\sigma$.

The algorithm is formally described in Algorithm 8. Throughout let $M^{(s)} = M(w^{(s)})$ and $M_t^{(s)} = M(w_t^{(s)})$, where $M(w) = \sum_{i=1}^n w_i(x_i - \mu(w))(x_i - \mu(w))^\top$. The procedure runs by epochs, where each epoch s reduces the largest eigenvalue of $M^{(s)}$ by a constant factor. We will show that the inner loop achieves the reduction within $O(\log d)$ iterations while maintaining the invariant that the weights are mostly-good (Definition B.3.1). Similar to

Algorithm 8: Matrix multiplicative update for spectral sample reweighing (Definition 3.3.1)

```
Input: A set of points $x_1, \dots, x_n$, $\lambda, \rho$ and a failure rate $\delta$
Output: A point $\nu' \in \mathbb{R}^d$ and weights $w' \in \mathcal{W}_{n,\varepsilon}$ that satisfy (3.3.1) up to a
        constant factor.

1  Let $w^{(0)} = \frac{1}{n}(1, 1, \cdots, 1)$.
2  For $s$ from 0 to $O(\log \rho)$
3      Compute $\lambda^{(s)} = \|M^{(s)}\|$.
4      If $\lambda^{(s)} \leq 300\lambda$
5          Return $w^{(s)}/\|w^{(s)}\|_1, \mu(w^{(s)})$.
6      For $t$ from 0 to $O(\log d)$
7          Compute $\lambda_t^{(s)} = \|M_t^{(s)}\|$ and terminate epoch if $\lambda_t^{(s)} \leq \frac{2}{3}\lambda_0^{(s)}$.
8          Compute $U_t^{(s)} = \mathsf{MMW}(M_1^{(s)}, M_2^{(s)}, \cdots, M_{t-1}^{(s)})$.
9          Compute
               $$\tau_{t,i}^{(s)} = \left(x_i - \mu\left(w_t^{(s)}\right)\right)^\top U_t^{(s)} \left(x_i - \mu\left(w_t^{(s)}\right)\right) \quad \text{(B.3.2)}$$
10         Let $w_{t+1}^{(s)} = w_t^{(s)}$ if $\sum_i w_{t,i}^{(s)}\tau_{t,i}^{(s)} \leq \frac{1}{4}\lambda_1^{(s)}$; otherwise
            $w_{t+1}^{(s)} = \textsc{1DFilter}(w_t^{(s)}, \tau_t^{(s)})$.
11     Let $w^{(s+1)} = w_t^{(s)}$.
```

our MWU analysis, our argument relies on a spectral signature lemma.

Lemma B.3.3 (spectral signature for mostly-good weights)**.** *Let $\{x_i\}_{i=1}^n$ be n points in $\mathbb{R}^d$. Suppose there exists $\nu \in \mathbb{R}^d$ and $w^* \in \mathbb{R}_n$ such that $|w^*| = 1-\varepsilon$ for sufficiently small ε, $\|w^*\|_\infty \le 1/n$ and*

$$\sum_{i=1}^n w_i^* (x_i - \nu)(x_i - \nu)^\top \preceq \lambda I$$

for some $\lambda > 0$. Then for any $w \in \mathcal{C}(w^*)$,

$$\|\nu - \nu(w)\| \le \frac{1}{1-\sqrt{2\varepsilon}} \left(3\sqrt{\lambda} + 2\sqrt{\varepsilon \|M(w)\|}\right), \tag{B.3.3}$$

where $\nu(w) = \sum_i w_i x_i$ and $M(w) = \sum_i w_i (x_i - \nu(w))(x_i - \nu(w'))^\top$.

Proof. This directly follows from Lemma B.1.1 and scaling. □

Using this, we establish a key invariant of the inner loop of the algorithm.

Lemma B.3.4. *Let $w \in \mathcal{C}(w^*)$ be such that $\beta = \|M(w)\|_2 \ge 300\lambda$ and U be a density matrix. Let $\tau_i = (x_i - \mu(w))^\top U (x_i - \mu(w))$. If $w' = 1\text{DF}\textsc{ilter}(w, \tau, 1/4)$, then we have $w' \in \mathcal{C}(w^*)$ and $\langle M(w'), U\rangle \le \frac{1}{4}\langle M(w), U\rangle$.*

Proof. Let $\mu(w^*) = \sum_i w_i^* x_i$. Then for any unit vector u, we have that by Jensen's inequality

$$\langle \mu(w^*) - \nu, u\rangle^2 \le \left\langle \sum_{i=1}^n w_i^* x_i - \nu, u \right\rangle^2 \le \sum_{i=1}^n w_i^* \langle x_i - \nu, u\rangle^2 \le \lambda.$$

Thus, $\|\mu(w^*) - \nu\|_2^2 \le \lambda$. Expanding the definition of τ_i, we get

$$\begin{aligned}
\sum_{i=1}^n w_i^* w_i \tau_i &= \left\langle \sum_{i=1}^n w_i^* w_i (x_i - \mu(w))(x_i - \mu(w))^\top, U \right\rangle \\
&\le \frac{1}{n} \left\langle \sum_{i=1}^n w_i^* (x_i - \mu(w^*))(x_i - \mu(w^*))^\top, U \right\rangle \\
&\qquad + \frac{1}{n}\|w^*\|_1 \cdot (\mu(w^*) - \mu(w))^\top U (\mu(w^*) - \mu(w)) \\
&\le \frac{1}{n}\langle M(w^*), U\rangle + \frac{1}{n}(1-\varepsilon)\|\mu(w^*) - \mu(w)\|_2^2 \\
&\le \frac{1}{n}\lambda + \frac{2}{n}\|\mu(w^*) - \nu\|_2^2 + \frac{6}{n}\|\mu(w) - \nu\|_2^2 && \text{(B.3.4)} \\
&\le \frac{1}{n}\lambda + \left(\frac{4}{n}\lambda + \frac{2}{n}\varepsilon\|M(w)\|\right) && \text{(B.3.5)} \\
&\le \frac{1}{30n}\|M(w)\| = \frac{1}{30n}\sum_{i=1}^n w_i \tau_i, && \text{(B.3.6)}
\end{aligned}$$

where (B.3.4) follows from the spectral centrality condition and triangle inequality, (B.3.5) follows from Lemma B.3.3, and (B.3.6) uses our assumption that $\|M(w)\| \geq 300\lambda$, the definition of τ_i and ε is sufficiently small. This allows us to apply the guarantee of the 1D filter procedure (Lemma B.3.2) and get that

$$\begin{aligned}\langle M(w'), U\rangle = \left\langle \sum_{i=1}^{n} w_i' (X_i - \mu(w'))(X_i - \mu(w')), U \right\rangle = \sum_{i=1}^{n} w_i'\tau_i \leq \frac{1}{4}\sum_{i=1}^{n} w_i\tau_i \\ = \frac{1}{4}\langle M(w), U\rangle.\end{aligned}$$

Furthermore, $w' \in \mathcal{C}(w^*)$. This completes the proof. □

We are now ready to prove the main theorem of this section.

Proof of Theorem B.3.1. Consider a fixed epoch and drop the super script for simplicity of notation. It is not hard to observe that $M(w_{t+1}) \preceq M(w_t)$ (see Lemma 3.4 [64]). Let $\alpha = 1/\|M(w_0)\|$. A regret bound for matrix multiplicative weights [8] implies that

$$\begin{aligned}\left\|\sum_{t=0}^{T-1} M(w_{t+1})\right\|_2 &\leq \sum_{t=0}^{T-1} \langle M(w_{t+1}), U_t\rangle + \alpha \sum_{t=0}^{T-1} \langle U_t, M(w_{t+1})\rangle \|M(w_{t+1})\|_2 + \frac{\log d}{\alpha} \\ &\leq 2\sum_{t=0}^{T-1} \langle M(w_{t+1}), U_t\rangle + \|M(w_0)\|_2 \cdot \log d\end{aligned}$$

Now by definition of Line 10, we have $\langle M(w_{t+1}), U_t\rangle \leq \frac{1}{4}\|M(w_0)\|_2$. Hence,

$$T\|M(w_T)\|_2 \leq \left\|\sum_{t=0}^{T-1} M(w_t)\right\|_2 \leq T \cdot \frac{1}{2}\|M(w_0)\|_2 + \|M(w_0)\|_2 \cdot \log d.$$

Setting $T \gg \log d$ shows that the inner loop terminates in $O(\log d)$ iterations and reduces the largest eigenvalue of the covariance by, say, $4/5$.

Finally, to bound the number of epochs, we simply note that $\|M^{(0)}\| \leq \rho$. Therefore, $O(\log(\rho/\lambda))$ epochs suffice drive the largest eigenvalue of $\|M^{(s)}\|$ down to $O(\lambda)$, since it is reduced geometrically each epoch. □

B.4 Sample reweighing via Online Gradient Descent

Regret analysis of gradient descent

We now consider a gradient updated-based algorithm for solving the spectral sample reweighing problem (Definition 3.3.1). The analysis will be through the classic regret guarantee of online gradient descent for convex optimization [161]. Though the resulting run-time is higher than the MWU scheme we analyzed in Section 3.3, it nonetheless betters the recent work of [41], where essentially the same gradient descent-based algorithm is studied.

We will leverage the following regret guarantee of online gradient descent; the definition of the algorithm in the general setting can be found in standard text [81].

Lemma B.4.1 (Theorem 3.1 [81], originally due to [161])**.** *Let $f_t : \mathcal{K} \to \mathbb{R}$ be the convex cost function revealed at iteration t, where $\mathcal{K}$ is a convex feasible set. Suppose f_t is L-Lipschitz (in ℓ_2 norm) and $\|x_0 - x^*\|_2 \le R$ for some $x^* \in \arg\min_{x\in\mathcal{K}} \sum_t f_t(x)$. The online gradient descent algorithm with step sizes $\eta_t = \frac{R}{L\sqrt{t}}$ achieves*

$$\sum_{t=1}^{T} f_t(x_t) - \min_{x\in\mathcal{K}} \sum_{t=1}^{T} f_t(x) \le \frac{3}{2} LR\sqrt{T}. \tag{B.4.1}$$

Our algorithm implicitly defines the cost functions $f_t(w) = \left\langle w, \tau^{(t)} \right\rangle$, where the feasible set is $\mathcal{W}_{n,\varepsilon}$, and implements the online gradient descent algorithm for the linear objective. Note that $\nabla f_t(w) = \tau^{(t)}$, and the main difference of this algorithm from the MWU scheme (Algorithm 1) is that we use an additive/gradient-descent update, in lieu of the multiplicative update.

Algorithm 9: Gradient descent for spectral sample reweighing (Definition 3.3.1)

```
Input: A set of points $\{x_i\}_{i=1}^n$, an iteration count $T$, and step sizes $\eta_t$
Output: A point $\nu \in \mathbb{R}^d$ and weights $w \in \mathcal{W}_{n,\varepsilon}$.
1 Let $w^{(1)} = \frac{1}{n}(1, 1, \cdots, 1)$.
2 For $t$ from 1 to $T$
3     Let $\nu^{(t)} = \sum_i w_i^{(t)} x_i$, $M^{(t)} = \sum_i w_i^{(t)} (x_i - \nu^{(t)})(x_i - \nu^{(t)})^T$.
4     Let $v^{(t)}$ be the top eigenvector of $M^{(t)}$ (with $\|v^{(t)}\| = 1$).
5     Compute $\tau_i^{(t)} = \left\langle v^{(t)}, x_i - \nu^{(t)} \right\rangle^2$.
6     Set $w_i \leftarrow w_i - \eta_t \tau^{(t)}$.
7     Project $w^{(t+1)}$ onto the set of good weights $\mathcal{W}_{n,\varepsilon}$ (under $\ell_2$ distance).
8 Return $\nu^{(t^*)}, w^{(t^*)}$, where $t^* = \arg\min_t \|M^{(t)}\|$.
```

Lemma B.4.2. *Let ρ be the squared diameter of the inputs points $\{x_i\}_{i=1}^n$. The cost function $f_t(\cdot)$ is $\sqrt{n}\rho$-Lipschitz (in ℓ_2 norm), for all t.*

Proof. Since f_t is differentiable, we only need the bound $\|\nabla f_t\|$. We have that for all t and i,

$$\tau_i^{(t)} = \left\langle v^{(t)}, x_i - \nu^{(t)} \right\rangle^2 \le \|x_i - \nu^{(t)}\|_2^2 \le \rho.$$

Therefore, $\|\nabla f_t\| = \|\tau^{(t)}\| \le \sqrt{n}\rho$. □

Theorem B.4.1. *Given $\{x_i\}_{i=1}^n$ and $\eta_t = R/L\sqrt{t}$ with $L = \sqrt{n}\rho, R = \sqrt{2}$, the online gradient descent algorithm (based on Algorithm 9) yields a constant-factor approximation for the spectral sample reweighing problem (Definition 3.3.1) in $O(nd^2/\varepsilon^2)$ iterations and $O(n^2d^3/\varepsilon^2)$ total run-time.*

Proof. We first apply the PRUNE procedure of Lemma 3.3.3 to bound the diameter.

By Lemma 3.3.2 and the guarantee of PRUNE, we can have $\rho = 16d\lambda/\varepsilon$. Then we apply Algorithm 9.

We will use Lemma B.4.1 to analyze Algorithm 9. First, by Lemma B.4.2, we have $L = \sqrt{n}\rho$, and further, since the ℓ_2 diameter of the probability simplex can be (trivially) bounded by $\sqrt{2}$, $R = \sqrt{2}$. Moreover, observe for any t,

$$f_t\left(w^{(t)}\right) = \left\langle w^{(t)}, \tau^{(t)}\right\rangle = \sum_i w_i^{(t)} \left\langle v^{(t)}, x_i - \nu^{(t)}\right\rangle^2 = v^{(t)T} M^{(t)} v^{(t)} = \left\|M^{(t)}\right\|_2.$$

Let $w \in \mathcal{W}_{n,\varepsilon}$ be a weight that satisfies the spectral centrality condition. Then, from the regret guarantee (B.4.1),

$$\frac{1}{T}\sum_{t=1}^{T} \left\|M^{(t)}\right\|_2 \le \frac{1}{T}\sum_{t=1}^{T} \left\langle w, \tau^{(t)}\right\rangle + \frac{3LR}{2\sqrt{T}} \tag{B.4.2}$$

We bound the two terms on the right side individually.

(i) A bound on the first term follows exactly from the calculations we did in the analysis of MWU algorithm (Algorithm 1). In particular, from (3.3.11) we have

$$\frac{1}{T}\sum_{t=1}^{T}\langle w, \tau^{(t)}\rangle \le 15\lambda + \frac{1}{3T}\sum_{t=1}^{T} \left\|M^{(t)}\right\|_2.$$

(ii) Observe that it suffices to set $T = 3L^2R^2/\lambda^2$ to bound the second term by λ.

Substituting the two bounds back into (B.4.2),

$$\frac{1}{T}\sum_{t=1}^{T} \left\|M^{(t)}\right\|_2 \le 16\lambda + \frac{1}{3T}\sum_{t=1}^{T} \left\|M^{(t)}\right\|_2. \tag{B.4.3}$$

Rearranging and dividing through immediately yields the desired guarantee.

Given that $L = \sqrt{n}\rho, R = \sqrt{2}$, we have that the iteration count $T = 6n\rho^2/\lambda^2$. Since $\rho = 16d\lambda/\varepsilon$, $T = O(nd^2/\varepsilon^2)$. For the run-time, note that instead of computing the exact largest eigenvector, we can use power method to find an 7/8-approximate one. Observe that this suffices for our analysis of the method above. Finally, the Euclidean projection onto $\mathcal{W}_{n,\varepsilon}$ can be computed in $O(n \log n)$ time [156]. This yields the desired run-time. □

Extension to sub-gaussian setting

Theorem B.4.1 implies that a gradient descent-based algorithm (Algorithm 9) can be used for robust mean estimation under bounded covariance. We now extend the result to the sub-gaussian setting, showing that the same iteration and run-time complexity holds. The optimal estimation error we will aim for is $O(\varepsilon\sqrt{\log(1/\varepsilon)})$. We assume the spectral signature Lemma B.2.1 and the deterministic condition (B.2.1).

Algorithm 10: Gradient descent for sub-gaussian robust mean estimation

```
Input: A set of points $\{x_i\}_{i=1}^n$, step sizes $\eta_t$, an iteration count $T$, and parameter $\rho$
Output: A set of weights $w \in \mathcal{W}_{n,\varepsilon}$.
1 Let $w^{(1)} = \frac{1}{n}\mathbb{1}_n$.
2 For $t$ from 1 to $T$
3     Let $\nu^{(t)} = \sum_i w_i^{(t)} x_i$, $M^{(t)} = \sum_i w_i^{(t)}(x_i - \nu^{(t)})(x_i - \nu^{(t)})^T$.
4     Compute $v^{(t)} = \textsc{ApproxTopEigenvector}(M^{(t)}, 1-\varepsilon^2, \delta/T)$.
5     If $\lambda^{(t)} = v^{(t)\top} M^{(t)} v^{(t)} \le 1$, return $w^{(t)}$.
6     Compute $\tau_i^{(t)} = \left\langle v^{(t)}, x_i - \nu^{(t)} \right\rangle^2$.
7     Set $w_i \leftarrow w_i - \eta_t \tau^{(t)}$.
8     Project $w^{(t+1)}$ onto the set of good weights $\mathcal{W}_{n,\varepsilon}$ (under $\ell_2$ distance).
9 Return $w^{(t^*)}$, where $t^* = \arg\min_t \|M^{(t)}\|$.
```

In particular, we will analyze Algorithm 10 and prove the following set of guarantees.

Lemma B.4.3. *Let ε be a sufficiently small constant and $\{x_i\}_{i=1}^n$ be n points in $\mathbb{R}^d$. Assume the following (deterministic) conditions hold.*

(i) There exists $\nu \in \mathbb{R}^d$ and $w \in \mathcal{W}_{n,\varepsilon}$ such that

$$\left\| \sum_{i=1}^n w_i \left(x_i - \nu\right)\left(x_i - \nu\right)^\top \right\| \le 1 + O\left(\varepsilon \log\left(1/\varepsilon\right)\right). \tag{B.4.4}$$

(ii) If $\|M(w)\| \le 1 + \lambda$, for some $\lambda \ge 0$, then for any $w \in \mathcal{W}_{n,\varepsilon}$,

$$\|\nu - \mu(w)\| \le \frac{1}{1-\varepsilon}\left(\sqrt{\varepsilon\lambda} + C\varepsilon\sqrt{\log(1/\varepsilon)}\right), \tag{B.4.5}$$

Then, given $\{x_i\}_{i=1}^n$, a failure rate δ and ρ such that $\rho \ge \tau_i^{(t)}$ for all i and t, Algorithm 10 finds $w' \in \mathcal{W}_{n,\varepsilon}$ such that

$$\|M(w')\| \le 1 + O\left(\varepsilon \log\left(1/\varepsilon\right)\right), \tag{B.4.6}$$

with probability at least $1-\delta$.

The algorithm terminates in $T = O(n\rho^2/\varepsilon^2)$ iterations. Further, if $T = O(poly(n,d))$, then each iteration takes $\widetilde{O}(nd\log(1/\delta)/\varepsilon^2)$ time.

Proof. If the algorithm gets early stopped, then $\|M^{(t)}\| \le 1 + O(\varepsilon^2)$, so assumption (B.4.4) guarantees that $\mu(w^{(t)})$ achieves the desired bound (B.4.6). We now assume that $\|M^{(t)}\| > 1$ for any t.

By the regret bound (Lemma B.4.1) and the inequality $\left\langle w^{(t)}, \tau^{(t)} \right\rangle \ge (1-\varepsilon^2) \left\|M^{(t)}\right\|_2$, for a w that satisfies assumption (B.4.4)

$$\frac{1-\varepsilon^2}{T} \sum_{t=1}^{T} \left\|M^{(t)}\right\|_2 \le \frac{1}{T} \sum_{t=1}^{T} \left\langle w, \tau^{(t)} \right\rangle + \frac{3LR}{2\sqrt{T}}, \tag{B.4.7}$$

where $L = \sqrt{n}\rho$ and $R = \sqrt{2}$. For the first term, note that we may apply Claim B.2.1 and obtain

$$\frac{1}{T} \sum_{t=1}^{T} \left\langle w, \tau^{(t)} \right\rangle \le 1 + O\left(\varepsilon \log(1/\varepsilon)\right) + \frac{2\varepsilon}{(1-\varepsilon)^2} \frac{1}{T} \sum_{t=1}^{T} \left\|M^{(t)}\right\| - \frac{2\varepsilon}{(1-\varepsilon)^2}$$

By setting $T = 3L^2R^2/\varepsilon^2 = O(n\rho^2/\varepsilon^2)$, we can bound the second term by $O(\varepsilon)$

Substituting the bounds back into (B.4.7), we obtain

$$\frac{1-\varepsilon^2}{T} \sum_{t=1}^{T} \left\|M^{(t)}\right\|_2 \le 1 - \frac{2\varepsilon}{(1-\varepsilon)^2} + O(\varepsilon \log(1/\varepsilon)) + \frac{1}{T} \sum_{t=1}^{T} \frac{2\varepsilon}{(1-\varepsilon)^2} \left\|M^{(t)}\right\|$$

For sufficiently small ε, we can move the last term to the left side and divide through. This immediately yields that

$$\frac{1}{T} \sum_{t=1}^{T} \left\|M^{(t)}\right\|_2 \le 1 + O(\varepsilon \log(1/\varepsilon)).$$

The run-time follows from the cost of computing $(1-\varepsilon^2)$-approximate largest eigenvector via power iteration. $\square$

Using the same argument for Theorem B.2.1, Lemma B.4.3 implies the following theorem.

Theorem B.4.2. *Let $S = \{x_i\}_{i=1}^n$ be an ε-corrupted set of n samples from a sub-gaussian distribution over $\mathbb{R}^d$, with mean μ and identity covariance. Suppose $n \ge \widetilde{\Omega}(d/\varepsilon^2)$. Then given S, there is an algorithm (based on Algorithm 10) that finds $\widehat{\mu}$ such that with high constant probability $\|\widehat{\mu} - \mu\| \le O\left(\varepsilon\sqrt{\log(1/\varepsilon)}\right)$.*

The algorithm runs in $\widetilde{O}(nd^2/\varepsilon^2)$ iterations and $\widetilde{O}(n^2d^3/\varepsilon^2)$ total time.

Equivalence with [41]

The recent work of Cheng, Diakonikolas, Ge and Soltanolkotabi [41] studies a gradient-descent-based algorithm for solving the following non-convex formulation of robust mean estimation.

$$\min \; \|\Sigma_w\| \quad \text{such that } w \in \mathcal{W}_{n,\varepsilon}.$$

where $\Sigma_w = \sum_{i=1}^n w_i(x_i - \mu(w))(x - \mu(w))^\top$. This is equivalent to

$$\min_{w} \max_{u \in \mathbb{S}^{d-1}} F(w, u) = u^\top \Sigma_w u \quad \text{such that } w \in \mathcal{W}_{n,\varepsilon}.$$

The sub-gradient of $F(w, u)$ with respect to w (for a fixed u) is given by

$$\nabla_w F(w, u) = Xu \odot Xu - 2\left(w^\top X u\right) Xu, \tag{B.4.8}$$

where $X \in \mathbb{R}^{n\times d}$ is the data matrix whose the ith row is x_i.

Based on the observation, they consider and analyze an algorithm that computes a (approximately) maximizing u and performs a projected gradient descent on w each iteration.

Since Algorithm 9 can be directly applied to the same robust setting (Corollary 3.4.2), it is natural to consider the relationships between the two algorithms. Indeed, one can argue that they are essentially the same. First, we unpack our gradient update (*i.e.*, the spectral scores) of iteration t. Note that

$$\begin{aligned}
\nabla_i f_t(w^{(t)}) = \tau_i^{(t)} &= \left\langle v^{(t)}, x_i - \nu^{(t)} \right\rangle^2 \\
&= \left\langle v^{(t)}, x_i \right\rangle^2 + \left\langle v^{(t)}, \nu^{(t)} \right\rangle^2 - 2\left\langle v^{(t)}, x_i \right\rangle \left\langle v^{(t)}, \nu^{(t)} \right\rangle \\
&= \left(Xv^{(t)} \odot Xv^{(t)}\right)_i + \left(w^{(t)\top} X v^{(t)}\right)^2 - 2\left(w^{(t)\top} X v^{(t)}\right)\left(Xv^{(t)}\right)_i
\end{aligned}$$

since $\nu^{(t)} = \sum_i w_i^{(t)} x_i = X^T w^{(t)}$, where $\odot$ denotes entrywise product of vectors.

Let $C_t = w^{(t)\top} X v^{(t)}$. Therefore, we can rewrite the gradient as

$$\nabla f_t(w^{(t)}) = C_t^2 \cdot \mathbb{1}_n + Xv^{(t)} \odot Xv^{(t)} - 2C_t \cdot Xv^{(t)}$$

Note that the gradient (B.4.8) used in [41] is exactly the same as above, except without the term of all-one vector $C_t^2 \cdot \mathbb{1}_n$. In the gradient update step, the additional term reduces the weight of every point uniformly by the same quantity C_t^2. However, observe that by Pythagorean theorem, the (Euclidean) projection onto $\mathcal{W}_{n,\varepsilon}$ can be decomposed into two (sequential) steps: (1) first an orthogonal projection onto the affine subspace containing $\mathcal{W}_{n,\varepsilon}$, and then (2) a projection onto $\mathcal{W}_{n,\varepsilon}$ itself. Note that reducing each coordinate by the same quantity or not results in the same vector by the first step. Therefore, the two algorithms yield the same sequence of iterates $(w^{(t)})_t$.

B.5 Optimal Breakdown Point Analysis

We now consider a slight variant of the filter algorithm and show that it achieves the optimal breakdown point of $\varepsilon = 1/2$, for the robust mean estimation problem. Recall that both the classic filter algorithm and our Algorithm 7 work with the spectral scores defined as $\tau_i = \left(\left\langle v^{(t)}, x_i \right\rangle - \left\langle v^{(t)}, \nu^{(t)} \right\rangle\right)^2$, where the second term is the (weighted) average of the first. Instead, the following variant replaces that by the median.

Algorithm 11: Optimal filter for spectral sample reweighing (Definition 3.3.1)

```
Input: A set of points $\{x_i\}_{i=1}^n$, an iteration count $T$, and parameter $\delta$
Output: A point $\nu \in \mathbb{R}^d$ and weights $w \in \mathcal{C}_{n,\varepsilon}$.
1 Let $w^{(1)} = \frac{1}{n}\mathbb{1}_n$.
2 While $\|M^{(t)}\| \geq \frac{16}{7}\lambda\left(1 + \frac{1}{1/2-\varepsilon}\right)$
3     Compute $v^{(t)} = \textsc{ApproxTopEigenvector}(M^{(t)}, 7/8, \delta/T)$.
4     Compute $\alpha_i^{(t)} = \langle v^{(t)}, x_i\rangle$ for each $i$ and let $m^{(t)} = \mathsf{median}\left(\{\alpha_i^{(t)}\}_{i=1}^n\right)$
5     Compute $\tau_i^{(t)} = \left(\alpha_i^{(t)} - m^{(t)}\right)^2$ for each $i$ and $\tau_{\max} = \max_{i:w_i>0}\tau_i^{(t)}$.
6     Set $w_i^{(t+1)} \leftarrow w_i^{(t)}\left(1 - \tau_i^{(t)}/\tau_{\max}\right)$ for each $i$, and $t \leftarrow t+1$.
7 Return $\nu^{(t)}, w^{(t)}$.
```

Throughout we let $\nu^{(t)} = \sum_i w_i^{(t)} x_i$, $M^{(t)} = \sum_i w_i^{(t)}(x_i - \nu^{(t)})(x_i - \nu^{(t)})^T$.

Our proof follows by tracing the argument of the soft down-weighting filter [116]. First, we assume that there exists a good set$G \subseteq [n]$ such that $|G| \geq (1-\varepsilon)n$ and

$$\frac{1}{(1-\varepsilon)n}\sum_{i\in G}(x_i - \nu)(x_i - \nu)^\top \preceq \lambda I. \tag{B.5.1}$$

Let $B = [n] \setminus G$, and we first establish a technical condition on $m^{(t)}$.

Lemma B.5.1. *Let $\beta^{(t)} = \frac{1}{n}\sum_{i\in G}\alpha_i^{(t)}$. Then we have $|m^{(t)} - \beta^{(t)}|^2 \leq \frac{\lambda}{1/2-\varepsilon}$.*

Proof. We fix one iteration and drop the superscript. let $\mu_G = \frac{1}{n}\sum_{i\in G} x_i$. First, observe that by (B.5.1), we have

$$\frac{1}{(1-\varepsilon)n}\sum_{i\in G}(x_i - \mu_G)(x_i - \mu_G)^\top \preceq \lambda I. \tag{B.5.2}$$

Therefore, $\mathbb{E}_{i\sim G}\left[(\alpha_i - \beta)^2\right] = \mathbb{E}_{i\sim G}\langle v, \mu_G - x_i\rangle^2 \leq \lambda$. By Chebyshev's inequality,

$$\Pr_{i\sim G}\left(|\alpha_i - \beta| > \sqrt{\frac{\lambda}{1/2-\varepsilon}}\right) \leq \frac{1}{2} - \varepsilon. \tag{B.5.3}$$

This means that we have $|G|\cdot(1/2+\varepsilon)$ points $i \in [n]$ that satisfy $|\alpha_i - \beta|^2 \leq \frac{\lambda}{1/2-\varepsilon}$. Our claim now follows since $|G| \geq (1-\varepsilon)$ and $(1-\varepsilon)(1/2+\varepsilon) > 1/2$ for any $\varepsilon \in (0, 1/2)$. □

This allows us to establish the key invariant of the algorithm.

Lemma B.5.2. *Suppose at iteration s, we have that*

$$\left\|M^{(s)}\right\| \geq \frac{16}{7}\lambda\left(1+\frac{1}{1/2-\varepsilon}\right). \tag{B.5.4}$$

and for $t = s$

$$\sum_{i\in G}\frac{1}{n}-w_i^{(t)} < \sum_{i\in B}\frac{1}{n}-w_i^{(t)} \tag{B.5.5}$$

Then the condition (B.5.5) *continues to hold for $t = s+1$.*

Proof. Observe that to prove the claim inductively, it suffices to show that for any s,

$$\sum_{i\in G} w_i^{(s)}-w_i^{(s+1)} < \sum_{i\in B} w_i^{(s)}-w_i^{(s+1)}. \tag{B.5.6}$$

We now just focus on these two iterations, drop the superscript and denote $w^{(s+1)}$ by w'. By definition of the update step (line 7 of Algorithm 11), we just need to prove that

$$\sum_{i\in G} w_i\tau_i < \sum_{i\in B} w_i\tau_i. \tag{B.5.7}$$

Now note that since $\nu = \mu(w) = \sum_{i=1}^n w_i x_i$, we have

$$\begin{aligned}\sum_{i=1}^n w_i\tau_i &= \sum_{i=1}^n w_i(\langle v, x_i\rangle - m)^2\\ &= \sum_{i=1}^n w_i\left(\langle v, x_i-\nu\rangle + \langle \nu, v\rangle - m\right)^2\\ &= \sum_{i=1}^n w_i\left(\langle v, x_i-\nu\rangle^2 + (m-\langle v,\nu\rangle)^2\right)\\ &\geq \sum_{i=1}^n w_i\left\langle v, x_i-\nu\right\rangle^2\\ &= v^\top M v \geq \frac{7}{8}\|M\|_2.\end{aligned}$$

Hence, to establish invariant (B.5.7), we proceed by showing that

$$\sum_{i\in G} w_i\tau_i \leq \frac{7}{16}\|M\|_2. \tag{B.5.8}$$

Since $w_i \leq \frac{1}{n}$, we have $\sum_{i \in G} w_i \tau_i \leq \sum_{i \in G} \frac{1}{n}(\langle v, x_i \rangle - m)^2$. On the other hand, let $\mu_G = \frac{1}{n} \sum_{i \in G} x_i$, and so by condition (B.5.1) and Lemma B.5.1,

$$\begin{aligned}\sum_{i \in G} \frac{1}{n} \left(\langle v, x_i \rangle - m\right)^2 &= \frac{1}{n} \sum_{i \in G} \langle v, x_i - \mu_G \rangle^2 + \left|\langle \mu_G, v \rangle - m\right|^2 \\ &\leq \lambda + \frac{\lambda}{1/2 - \varepsilon} \\ &\leq \frac{7}{16} \|M\|,\end{aligned}$$

by our assumption (B.5.4). This completes the proof. □

Theorem B.5.1. *For any $\varepsilon \in (0, 1/2)$, Algorithm 11 gives a constant approximation to the spectral sample reweighting problem (Definition 3.3.1). The algorithm terminates in $T = O(n)$ iterations.*

Proof. The run-time follows from the invariant Lemma B.5.2, which guarantees weights on bad points are removed more than those on good points. Hence, after $2\varepsilon n$ iterations, the algorithm must terminate. Moreover, when the algorithm terminates, we have

$$\left\|M^{(t)}\right\| \leq \frac{16}{7} \lambda \left(1 + \frac{1}{1/2 - \varepsilon}\right). \tag{B.5.9}$$

For any constant $\varepsilon \leq 1/2 - O(1)$, the bound is $O(\lambda)$. □

Robust mean estimation. Our reduction from spectral sample reweighting to robust mean estimation is not sufficiently tight for the purpose of attaining optimal breakdown point. Instead, we need to appeal to the following more refined spectral signature.

Claim B.5.1 (refined spectral signature [115])**.** *Let $S = S_g \cup S_b \setminus S_r$ be n points with $|S_b| = |S_r| = \varepsilon n$. Define $\mu_g = \frac{1}{n} \sum_{i \in S_g} x_i$ and $\Sigma = \frac{1}{n} \sum_{i \in S_g} (x_i - \mu)(x_i - \mu)^\top$. Let $w(S)$ be the uniform distribution on S and $\mathcal{C}_{n,\varepsilon} = \{w : \|w - w(S)\|_1 \leq \varepsilon, 0 \leq w_i \leq 1/n \text{ for } i \in [n]\}$. Then for any $w \in \mathcal{C}_{n,\varepsilon}$,*

$$\left(\sum_{i \in S \cap S_g} w_i\right) \|\mu - \mu(w)\| \leq \sqrt{2\varepsilon \|\Sigma\|} + \sqrt{\varepsilon \, \|\Sigma(w)\|}.$$

Theorem B.5.2. *For the problem of robust mean estimation (under bounded second moment), Algorithm 11 attains the optimal estimation error $O(\sqrt{\varepsilon})$ for any $\varepsilon < 1/2$.*

Proof. By Lemma B.5.2, our algorithm always removes more weights from bad points than from good points. Thus, $w^{(t)} \in \mathcal{C}_{n,2\varepsilon}$, as there are at most εn bad points. Moreover, $\sum_{i \in S \cap S_g} w_i \geq 1 - 2\varepsilon$.

For robust mean estimation, if we have $n = \Omega(d \log d/\varepsilon)$ samples, then $\|\mu_g - \mu\| \leq O(\sqrt{\varepsilon})$ and $\lambda = \|\Sigma\| \leq 2$ [60]. Hence, applying Claim B.5.1 and the guarantee that $\|M^{(t)}\| \leq \frac{16}{7}\lambda\left(1 + \frac{1}{1/2-\varepsilon}\right)$,

$$\left\|\mu_g - \nu^{(t)}\right\| \leq \frac{1}{1-2\varepsilon}\left(2\sqrt{\varepsilon\|\Sigma\|} + \sqrt{2\varepsilon\left\|M^{(t)}\right\|}\right) \leq O(\sqrt{\varepsilon}),$$

for any $\varepsilon < 1/2$. Finally, triangle inequality implies that $\|\mu - \nu^{(t)}\| = O(\sqrt{\varepsilon})$. □