NORTHWESTERN
UNIVERSITY

Computer Science Department

**Technical Report**
**Number: NU-CS-2023-18**

December, 2023

# New Directions for Learning Mixture Models

**Aidao Chen**

## Abstract

Mixture models are a powerful way to model data coming from a heterogeneous population. Over the years, various mixture models and learning techniques have been proposed.
In this dissertation, we explore new directions in the realm of learning mixture models. We study various mixture models and present results and insights which were unknown previously.
In the first part of the thesis, we study the problem of learning a mixture of linear classifiers, which is an important, yet poorly understood problem. In a natural model, we present the first polynomial time algorithm that recovers the parameters when the direction vectors of linear classifiers are linearly independent. We also show a quasi-polynomial time algorithm under provably minimal assumptions.
In the second part of the thesis, we consider the problem of learning a mixture of two subspaces over GF(2). This problem is computationally challenging in the worst case, as it captures the notorious problem of "Learning Parity with Noise" in the degenerate setting. We show that one can design a polynomial time learning algorithm in the non-degenerate setting.

NORTHWESTERN UNIVERSITY

# New Directions for Learning Mixture Models

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY

Field of Computer Science

By

Aidao Chen

EVANSTON, ILLINOIS

December 2023

# Abstract

New Directions for Learning Mixture Models

Aidao Chen

Mixture models are a powerful way to model data coming from a heterogeneous population. Over the years, various mixture models and learning techniques have been proposed.

In this dissertation, we explore new directions in the realm of learning mixture models. We study various mixture models and present results and insights which were unknown previously.

In the first part of the thesis, we study the problem of learning a mixture of linear classifiers, which is an important, yet poorly understood problem. In a natural model, we present the first polynomial time algorithm that recovers the parameters when the direction vectors of linear classifiers are linearly independent. We also show a quasi-polynomial time algorithm under provably minimal assumptions.

In the second part of the thesis, we consider the problem of learning a mixture of two subspaces $A_0, A_1$ over $\mathbb{F}_2^n$. This problem is computationally challenging in the worst case, as it captures the notorious problem of " Learning Parity with Noise" in the degenerate setting when $A_1 \subseteq A_0$ and $\mathsf{dim}(A_1) = \mathsf{dim}(A_0) - 1$. We show that one can design

a polynomial time learning algorithm when either of the two obstacles ($A_1 \subseteq A_0$ and $\mathsf{dim}(A_1) = \mathsf{dim}(A_0) - 1$) are absent.

## Acknowledgements

I owe appreciation to a lot of people for their generous help along my PhD journey. First and foremost, I would like to thank my advisors Anindya De and Aravindan Vijayaraghavan, for their invaluable support and guidance throughout my PhD. I always told my friends that it is very challenging to find anyone better than them. I consider myself incredibly fortunate and deeply grateful to have the distinct honor of receiving guidance and advice from them. From an academic standpoint, I have had the privilege of learning an abundance of intriguing technical insights and problem-solving techniques from my esteemed advisors, in addition to refining my ability to write and communicate scientific work tailored to diverse audiences. The comprehensive knowledge and experience imparted by my mentors have proved invaluable to my academic and professional development, and I am honored to have been exposed to their exceptional guidance and expertise. From a personal perspective, it is truly remarkable to have witnessed such extraordinary care, patience, and consideration from individuals as distinguished and occupied as Anindya and Aravindan. Their exemplary conduct has served as a beacon of inspiration and guidance to me and countless other students, and I am grateful for the remarkable example they have set for us all.

I am immensely grateful to the distinguished members of my thesis committee – Aditya Bhaskara, Anindya De, Konstantin Makarychev, and Aravindan Vijayaraghavan – for their unwavering and selfless support throughout my PhD journey. Their sage counsel,

insightful feedback, and constructive suggestions have been invaluable in refining the quality of my thesis and deepening my understanding of this research direction. I am honored and privileged to have been mentored by such distinguished individuals, whose expertise and guidance have been pivotal to my intellectual and academic growth.

I also would like to express my sincere gratitude to all of my letter writers, including Huxley Bennett, Aditya Bhaskara, Anindya De, Stefano Ermon, Haibin Kan, Feifei Li, Konstantin Makarychev, Weiwei Sun, Aravindan Vijayaraghavan, and Wei Zhang. Their support has been invaluable in creating a truly fantastic environment for my studies and research. I am deeply grateful for their time, effort, and encouragement.

Being a part of the Northwestern CS Theory Group has been an absolutely wonderful experience. I am immensely grateful to the faculties - Anindya De, Jason Hartline, Samir Khuller, Konstantin Makarychev, and Aravindan Vijayaraghavan - for their outstanding commitment to the field and for providing an exceptional learning environment. Their expertise and guidance have been invaluable in shaping my academic journey. I also extend my heartfelt thanks to my colleagues, who have made my time here so memorable. Special shoutouts to Saba Ahmadi, Huck Bennett, Xue Chen, Dunwei Cheng, Charles Cui, Jinshuo Dong, Abhratanu Dutta, Yiding Feng, Aleck Johnsen, Nirmit Joshi, Sanchit Kalhan, Yiduo Ke, Yingkai Li, Sheng Long, Michail Mamakos, Aidan Perreault, Shravas Rao, Leif Rasmussen, Aravind Reddy, Anant Shah, Liren Shan, Vaidehi Srinivas, Pattara Sukprasert, Sam Taggart, Alex Tang, Matthew VonAllmen, Yifan Wu, Sheng Yang and Chenhao Zhang. The Theory Group is truly a wonderful place because of the collective efforts of all its members, and I feel privileged to have been a part of it. I would like to extend my appreciation to my friends: Weixin Jiang, Xiaohu Lei, Yilun Lin, Yin Xia,

Haoming Xing, and Zheng Yuan. Life would certainly be less vibrant and exciting without your presence in it.

In addition, I thank my parents and my brother for their unconditional support.

Last but not least, I am grateful for Yuanyuan's unwavering support and constant companionship throughout my PhD journey. Thank Yuanyuan for being there for me every step of the way.

## Dedication

To Yuanyuan.

# Table of Contents

CHAPTER 1

# Introduction

In the world of machine learning, mixture models have been fundamental tools for the analysis of heterogeneous populations. In particular, the population is assumed to consist of $k$ subgroups (assumed to be homogeneous) and the data in each subgroup follows a parametric model. Given data from the overall population, the typical challenge involves determining the parameters of each component, as well as their respective proportions within the population. Through the years, these powerful probabilistic models have revolutionized the approach to recognize the patterns within seemingly chaotic data.

This thesis serves as a stepping stone towards the future of mixture models, providing results and insights that were unexplored previously. In this chapter, we will give a brief overview of the results which will be discussed in more detail later.

The first part of this thesis studies the problem of learning a mixture of linear classifiers. Over the years, there has been significant research in statistics and computer science aimed at developing efficient polynomial time algorithms for learning many types of mixture models [Feldman et al., 2006, Kalai et al., 2010, Moitra and Valiant, 2010, Belkin and Sinha, 2010, Awasthi et al., 2010, Rabani et al., 2014, Li et al., 2015, Liu and Moitra, 2018, Chen and Moitra, 2019]. Much of this work has focused on the unsupervised setting, where the data is unlabeled. However, there has been growing interest in recent years in studying mixture models in the supervised setting, where the data is labeled [Viele and Tong, 2002, Chaganty and Liang, 2013, Sun et al., 2014, Gandikota

et al., 2020, Diakonikolas and Kane, 2020, Chen et al., 2020]. This direction involves incorporating labeled information into the learning process. Although this line of work is still in its early stages, it shows promise for enhancing the usefulness of mixture models in a variety of applications. Among models in supervised learning, linear classifiers are one of the most fundamental and well-studied models. We study the problem of learning a mixture of linear classifiers. We will describe a natural model for studying this problem later. Basic statistical and algorithmic questions about this natural model remained open before our work. As an example, the identifiability of this model was unresolved until this work.

The second part of this thesis studies the problem of learning a mixture of two subspaces over $\mathbb{F}_2^n$. A common assumption in high-dimensional data analysis is to assume that the given data belongs to a collection of lower dimensional subspaces. In a line of work in machine learning, computer vision and computational geometry [Vidal, 2003, Elhamifar and Vidal, 2013, Soltanolkotabi et al., 2014, Park et al., 2014], this intuition is formalized through the problem of learning a mixture of subspaces, or subspace clustering. Given a set of points in $n$ dimensions that belong to a union of $k$ ($k \geq 2$) subspaces, the goal is to recover the individual subspaces. When the points belong to $\mathbb{R}^n$, Vidal [2003] shows that for any mixture of $k$ subspaces, under some mild general-position assumption of the points in the subspaces, there exists an algorithm that recovers the $k$ individual subspaces. However, the guarantee is specific to the real domain. Therefore, a natural question is whether such algorithmic guarantee can extend to other domains such as $\mathbb{F}_2$.

The study of mixture models can also help the study of other related fields. In our algorithm for the previous problem, we use the idea: use tensor power to blow up the

dimensional difference between two subspaces. This idea serves as a cornerstone for us to tackle the problem of robust subspace recovery. The third part of this thesis studies the problem of robust subspace recovery in a smoothed analysis setting. Robust subspace recovery is a basic problem in unsupervised learning where we are given $m$ points $x_1, \ldots, x_m \in \mathbb{R}^n$, an $\alpha \in (0, 1)$ fraction of which lies on (or close to) a $d$-dimensional subspace $T$. When can we find the subspace $T$, and hence the "inliers", that belong to this subspace? [Hardt and Moitra, 2013] gave the first algorithm for this problem that is both computationally efficient and robust. Their algorithm successfully estimates the subspace $T$ when $\alpha > d/n$, assuming a certain non-degeneracy condition about both the inliers and outliers. Borrowing an idea from Chapter 3, we give a simple algorithm that for any constants $\ell \in \mathbb{N}, \delta > 0$ runs in $\text{poly}(mn^\ell)$ time and in a smoothed analysis setting, provably recovers the subspace $T$ with high probability, when $\alpha \geq (1 + \delta)(d/n)^\ell$. Note that this is significantly smaller than the bound of $(d/n)$ from Hardt and Moitra [2013] when $\ell > 1$.

### 1.1. Mixture of Linear Classifiers

We will describe a natural model for a mixture of linear classifiers. $k$ is the number of linear classifiers. There are $k$ (unknown) strictly positive probability weights $w_1, w_2, \ldots, w_k \in \mathbb{R}$ that sum up to 1. There are $k$ (unknown) unit vectors $v_1, v_2, \ldots, v_k \in \mathbb{R}^n$ representing coefficients of each linear classifier. A sample is drawn as the following: the sample oracle samples $i \in [k]$ with respect to the probability weights $w_1, w_2, \ldots, w_k$, then we receive $(\mathbf{x}, \mathbb{1}_{\langle v_i, \mathbf{x} \rangle \geq 0})$ where $\mathbf{x}$ is sampled from the standard Gaussian $\mathcal{N}(0, I_n)$.

The $i$ is unknown to us. The goal is to recover the unknown parameters $w_1, w_2, \ldots, w_k$ and $v_1, v_2, \ldots, v_k$ approximately.

We would like to point out that our model and the broader topic of learning mixtures of (supervised) linear models have been extensively studied in the literature [Chaganty and Liang, 2013, Gandikota et al., 2020, Chen et al., 2020], including in the context of neural networks [Jacobs et al., 1991, Jordan and Jacobs, 1994, Bishop, 1998]. Despite this interest, several fundamental statistical and algorithmic questions about this model have remained unresolved. For instance, prior to our work, the identifiability of this model, even for $k = 3$, had not been resolved to the best of our knowledge.

Our first result is a polynomial-time algorithm that learns the parameters under the assumption that the unit vectors $v_1, v_2, \ldots, v_k$ are linearly independent.

Furthermore, we give two algorithms that recover the parameters under a less stringent assumption: the vectors $v_1, v_2, \ldots, v_k$ are not parallel to each other. In order to explain our findings, we define $\Delta := \min_{j \neq j'} \min(\|v_j - v_{j'}\|, \|v_j + v_{j'}\|)$. Both algorithms depend polynomially on the ambient dimension $n$. The first algorithm (Theorem 2.2) achieves a quasipolynomial dependence on the number of vectors $k$ with an exponential dependence on $1/\Delta$. The second algorithm (Theorem 2.3) achieves a polynomial dependence on $1/\Delta$, but has an exponential dependence on $k$. Our results imply that as long as $\Delta > 0$, the model is identifiable. It is worth noting that if $\Delta = 0$ (e.g., when $k = 2$), the model is no longer identifiable. Therefore, a dependence on $1/\Delta$ is qualitatively necessary for identifiability and, as a result, for algorithmic implications such as ours.

## 1.2. Mixture of Two Subspaces over $\mathbb{F}_2$

Given samples from a (weighted) mixture of samples drawn uniformly from $k$ subspaces of $\mathbb{F}_2^n$, the goal is to recover each individual subspace. The simplest setting of $k = 2$ already captures the notoriously hard problem Learning Parity with Noise (LPN) as a special case. The best-known algorithm for LPN takes $\exp(O(n/\log n))$ time. One can encode LPN by an instance of a mixture of two subspaces $A_0, A_1$ where $A_1 \subseteq A_0$ and $\dim(A_1) = \dim(A_0) - 1$ (see Proposition 3.24 and Proposition 3.23). We show that one can design a polynomial time algorithm when either of the two obstacles ($A_1 \subseteq A_0$ and $\mathsf{dim}(A_1) = \mathsf{dim}(A_0) - 1$) are absent. First, we show that there is a polynomial time algorithm that outputs the subspaces $A_0, A_1$ when $A_0 \nsubseteq A_1$ and $A_1 \nsubseteq A_0$. The key idea for this algorithm is adaptive dimension reduction: a careful procedure for dimension reduction that reduces the subspace learning problem to $O(1)$ dimension. When $n = O(1)$, this problem becomes easy, as a brute force algorithm will only take $O(1)$ time. Second, we show that there is a polynomial time algorithm that outputs the subspaces $A_0, A_1$ when $\mathsf{dim}(A_1) \leq 0.99\mathsf{dim}(A_0)$. For illustration purposes, let us assume the mixing weights are 0.5 and 0.5 for now. Borrowing an idea from Hardt and Moitra [2013], we show that there is a polynomial time algorithm which solves this problem when $\mathsf{dim}(A_1)/\mathsf{dim}(A_0) \leq 0.49$. When $\mathsf{dim}(A_1)/\mathsf{dim}(A_0) > 0.5$, we will adopt a *dimension gap amplification strategy*. Specifically, we introduce a non-linear map $\phi : \mathbb{F}_2^{d_0} \to \mathbb{F}_2^{d_0'}$ where $d_0' = \sum_{j=0}^{\ell} \binom{d_0}{j}$ for an appropriately chosen $\ell$. The function $\phi$ serves a purpose akin to that of a kernel function within the context of Support Vector Machines. For some carefully chosen $\ell$, we will have $\mathsf{dim}(\mathsf{span}(\phi(A_1)))/\mathsf{dim}(\mathsf{span}(\phi(A_0))) < 1/2$. We can then apply the large-gap-case strategy to recover both subspaces.

### 1.3. Robust Subspace Recovery

Here, we will provide a concise overview of our results concerning robust subspace recovery in a smoothed analysis setting. We introduce the following smoothed analysis framework for studying robust subspace recovery (we refer the reader to Section 4.3.1 for the formal definition of the framework).

In what follows, $\alpha, \varepsilon_0, \rho \in (0,1)$ are parameters.

(1) An adversary chooses a hidden subspace $T$ of dimension $d$ in $\mathbb{R}^n$, and then chooses $\alpha m$ points from $T$ and $(1-\alpha)m$ points from $\mathbb{R}^n$. We denote these points inliers and outliers respectively. Then the adversary mixes them in arbitrary order. Denote these points $a_1, a_2, \ldots, a_m$. Let $A = (a_1, a_2, \ldots, a_m)$.

(2) Each inlier is $\rho$-perturbed with respect to $T$. (Formally, this means considering an orthonormal basis $B_T$ for $T$ and adding $B_T v$, where $v \sim \mathcal{N}(0, \rho^2/d)^d$.) Each outlier is $\rho$-perturbed with respect to $\mathbb{R}^n$. Let $G$ denote the perturbations, and let us write $\widetilde{A} = A + G$.

(3) With the constraint $\|E\|_F \leq \varepsilon_0$, the adversary adds noise $E \in \mathbb{R}^{n \times m}$ to $\widetilde{A}$, yielding $\widetilde{A}' = \widetilde{A} + E$.

(4) We are given $\widetilde{A}'$.

The goal of the subspace recovery problem is to return a subspace $T'$ close to $T$.

We give a simple algorithm that for any constants $\ell \geq 1, \delta > 0$ runs in $\mathrm{poly}(mn^\ell)$ time and in this smoothed analysis setting, provably recovers the subspace $T$ with high probability, when $\alpha \geq (1+\delta)(d/n)^\ell$.

The algorithm for robust subspace recovery at a high level follows the same approach as Hardt and Moitra [2013]. Their main insight was that if we sample a set of size slightly

less than $n$ from the input, and if the fraction of inliers is $> (1+\delta)d/n$, then there is a good probability of obtaining $> d$ inliers, and thus there exist points that are in the linear span of the others. Further, since we sampled fewer than $n$ points and the outliers are also in general position, one can conclude that the only points that are in the linear span of the other points are the inliers.

Our algorithm for handling smaller $\alpha$ is simple and is also tolerant to an inverse polynomial amount of adversarial noise in the points. Borrowing an idea from learning a mixture of two subspaces over $\mathbb{F}_2$ (Section 3.5), we can use a similar way to look for linear dependencies, but with tensored vectors!

Let us illustrate in the case $\ell = 2$. Suppose that the fraction of inliers is $> (1+\delta)\binom{d+1}{2}/\binom{n+1}{2}$. Suppose we take a sample of size slightly less than $\binom{n+1}{2}$ points from the input, and consider the flattened vectors $x \otimes x$ of these points. As long as we have more than $\binom{d+1}{2}$ inliers, we expect to find linear dependencies among the tensored inlier vectors. However, we need to account for the adversarial error in the points (this error could depend on the random perturbations as well). For each point, we will look for "bounded" linear combinations that are close to the given point. We can show that such dependencies cannot involve the outliers. This in turn allows us to recover the subspaces.

## 1.4. Organization of the Thesis

In Chapter 2, we introduce the problem of learning a mixture of linear classifiers over Gaussian marginals; and discuss the technique and results for this problem. In Chapter 3, we study the problem of learning a mixture of two subspaces over $\mathbb{F}_2^n$. In Chapter 4, we shift our attention to a closely related problem: robust subspace recovery.

## 1.5. Bibliographic Notes

The content of this thesis is mainly based on three research papers with co-authors Bhaskara et al. [2019], Chen et al. [2021, 2022].

CHAPTER 2

# Learning a Mixture of Linear Classifiers

## 2.1. Introduction

Mixture models are a widely used method for modeling data that originate from a heterogeneous population. This approach involves assuming that the population is comprised of k subgroups. It is assumed that the data within each subgroup follows a specific parametric model.

The main task of learning mixture models is to determine the parameters of each of the components, as well as their relative proportions within the overall population, using available data. This process can provide insight into the underlying structure of the population and help people better understand the factors that contribute to observed patterns in the data.

Mixture models are used to represent data across a wide range of domains. The examples include Gaussian mixture models, mixtures of product distributions, mixtures of ranking models and mixtures of subcubes for discrete data. Over the years, there has been significant research in statistics and computer science aimed at developing efficient polynomial time algorithms for learning many types of mixture models [Feldman et al., 2006, Kalai et al., 2010, Moitra and Valiant, 2010, Belkin and Sinha, 2010, Awasthi et al., 2010, Rabani et al., 2014, Li et al., 2015, Liu and Moitra, 2018, Chen and Moitra, 2019]. Much of this work has focused on the unsupervised setting, where the data is unlabeled.

However, there has been growing interest in recent years in studying mixture models in the supervised setting, where the data is labeled [Viele and Tong, 2002, Chaganty and Liang, 2013, Sun et al., 2014, Gandikota et al., 2020, Diakonikolas and Kane, 2020, Chen et al., 2020]. This direction involves incorporating labeled information into the learning process. Although this line of work is still in its early stages, it shows promise for enhancing the usefulness of mixture models in a variety of applications.

Linear classifiers are the most basic and well-studied supervised learning models due to their simplicity, effectiveness, and wide range of applications. These models use a linear function to map the input features to the output, making them easy to understand and computationally efficient. Despite their simplicity, linear classifiers have been shown to work well for many real-world problems, including text classification, image recognition, speech recognition and quantitative trading. Additionally, they are often robust to noisy data and can still perform well even when there are some errors in the input data. The strong theoretical foundation of linear classifiers has led to the development of a variety of algorithms and techniques for training and optimizing these models, making them a popular and effective choice for many supervised learning problems.

Therefore, it is both natural and crucial to investigate the problem of learning mixtures of linear classifiers. Within this chapter, we will present various results related to learning a mixture of linear classifiers.

We will describe a natural model for a mixture of linear classifiers. $k$ is the number of linear classifiers. There are $k$ (unknown) strictly positive probability weights $w_1, w_2, \ldots, w_k \in \mathbb{R}$ that sum up to 1. There are $k$ (unknown) unit vectors $v_1, v_2, \ldots, v_k \in \mathbb{R}^n$ representing coefficients of each linear classifier. A sample is drawn as the following:

the sample oracle sample $i \in [k]$ with respect to the probability weights $w_1, w_2, \ldots, w_k$, then we receive $(\mathbf{x}, \mathbb{1}_{\langle v_i, \mathbf{x} \rangle \geq 0})$ where $\mathbf{x}$ is sampled from the standard Gaussian $\mathcal{N}(0, I_n)$. The $i$ is unknown to us. The goal is to recover the unknown parameters $w_1, w_2, \ldots, w_k$ and $v_1, v_2, \ldots, v_k$ approximately.

We would like to point out that our model [Sun et al., 2014] and the broader topic of learning mixtures of (supervised) linear models have been extensively studied in the literature [Chaganty and Liang, 2013, Gandikota et al., 2020, Chen et al., 2020], including in the context of neural networks [Jacobs et al., 1991, Jordan and Jacobs, 1994, Bishop, 1998]. Despite this interest, several fundamental statistical and algorithmic questions about this model have remained unresolved. For instance, prior to our work, the identifiability of this model, even for $k = 3$, had not been resolved to the best of our knowledge.

### 2.1.1. Our results

Our first result is a polynomial-time algorithm that learns the parameters under the assumption that the unit vectors $v_1, v_2, \ldots, v_k$ are linearly independent.

It is worth noting that the model is not identifiable if no separation condition is assumed on the vectors $v_1, v_2, \ldots, v_k \in \mathbb{R}^n$. This is due to the fact that the distribution resulting from an equal weight mixture of two linear classifiers $v_1 = u$ and $v_2 = -u$ is the same for every $u \in \mathbb{R}^n$. Furthermore, when $v_i = v_j$, non-identifiability occurs since the weights of the two components can be redistributed arbitrarily.

In this model, Sun et al. [2014] shows that: if the vectors $v_1, v_2, \ldots, v_k$ are linearly independent, there exists an efficient (polynomial time) algorithm that recovers the subspace spanned by $v_1, v_2, \ldots, v_k$. To the best of our knowledge, there is no parameter

recovery result before our result. Under the assumption that $v_1, v_2, \ldots, v_k$ are linearly independent, we show that there exists an efficient algorithm that recovers the parameters of this model. A formal statement is as follows.

**Theorem 2.1.** *Let $U \in \mathbb{R}^{n\times k}$ be the matrix whose $j$th column is $v_j$. Suppose $\sigma_{\min}(U) \geq 1/\tau$, where $\tau > 0$. Given parameters $\varepsilon, \delta > 0$, $k \in \mathbb{N}$ and $w_{\min} > 0$ satisfying $w_{\min} \leq \min\{w_1, \ldots, w_k\}$, there is an algorithm* Estimate-Parameter *that given samples from the model has the following guarantees:*

(1) *The algorithm runs in sample complexity and time complexity*

$$\log^2(1/\delta)\varepsilon^{-2}\mathrm{poly}(n, \tau, 1/w_{\min}).$$

(2) *With probability $1-\delta$, the algorithm returns estimates $\{\hat{w}_j, \hat{v}_j : j \in [k]\}$ such that*

$$\min_{\pi\in\mathrm{Perm}([k])} \left(\max\{\|\hat{v}_j - v_{\pi(j)}\| : j \in [k]\} + \max\{\left|\hat{w}_j - w_{\pi(j)}\right| : j \in [k]\}\right) \leq \varepsilon.$$

*where the* min *is over permutations $\pi$ on $[k]$.*

Furthermore, we give two algorithms that recover the parameters under a less stringent assumption: that is, assuming that the vectors $v_1, v_2, \ldots, v_k$ are non-parallel.

In order to explain our findings, we define $\Delta := \min_{j\neq j'} \min \|v_j - v_{j'}\|, \|v_j + v_{j'}\|$. At a high level, we provide two algorithms to solve this problem. Both algorithms depend polynomially on the ambient dimension $n$. The first algorithm (Theorem 2.2) achieves a quasipolynomial dependence on the number of vectors $k$ with an exponential dependence on $1/\Delta$. The second algorithm (Theorem 2.3) achieves a polynomial dependence on $1/\Delta$, but has an exponential dependence on $k$. Our results imply that as long as $\Delta > 0$, the

model is identifiable. It is worth noting that if $\Delta = 0$ (e.g., when $k = 2$), the model is no longer identifiable. Therefore, a dependence on $1/\Delta$ is qualitatively necessary for identifiability and, as a result, for algorithmic implications such as ours.

Our next result achieves a running time (and sample complexity) guarantee of the form $n^{O(\log k)/\Delta^2}$.

**Theorem 2.2.** *Given parameters $\varepsilon, \delta > 0$, $k \in \mathbb{N}$ and $w_{\min} > 0$ satisfying $w_{\min} \leq \min\{w_1, \ldots, w_k\}$, there is an algorithm that given samples from the model has the following guarantees:*

(1) *The algorithm runs in sample complexity and time complexity*

$$\log^2(1/\delta)\varepsilon^{-2}\mathrm{poly}(n^{(\log k)/\Delta^2}, ((\log k)/\Delta^2)^{(\log k)/\Delta^2}, 1/w_{\min}).$$

(2) *With probability $1-\delta$, the algorithm returns estimates $\{\hat{w}_j, \hat{v}_j : j \in [k]\}$ such that*

$$\min_{\pi \in \mathrm{Perm}([k])} \left(\max\{\|\hat{v}_j - v_{\pi(j)}\| : j \in [k]\} + \max\{\left|\hat{w}_j - w_{\pi(j)}\right| : j \in [k]\}\right) \leq \varepsilon,$$

*where the* min *is the minimum is over permutations $\pi$ on $[k]$.*

The runtime guarantee stated above is quasi-polynomial, as long as the minimum separation parameter $\Delta = \Omega(1)$. Additionally, the algorithm runs in polynomial time when the number of mixture components $k = O(1)$. The algorithm recovers all unknown parameters up to an error of $\varepsilon > 0$, up to an ambiguity in the relabeling of the $k$ mixture components (this is captured by the permutation $\pi$).

Our next result gives a $\mathrm{poly}((n/\Delta)^k)$ running time and sample complexity guarantee.

**Theorem 2.3.** *Given parameters $\varepsilon, \delta > 0$, $k \in \mathbb{N}$ and $w_{\min} > 0$ satisfying $w_{\min} \leq \min\{w_1, \ldots, w_k\}$, there is an algorithm that given samples from the model has the following guarantees:*

(1) *The algorithm runs in sample complexity and time complexity*

$$\log^2(1/\delta)\varepsilon^{-2}\mathrm{poly}(n^k, k^k, \Delta^{-k}, 1/w_{\min})$$

(2) *With probability $1-\delta$, the algorithm returns estimates $\{\hat{w}_j, \hat{v}_j : j \in [k]\}$ such that*

$$\min_{\pi \in \mathrm{Perm}([k])} \left(\max\{\|\hat{v}_j - v_{\pi(j)}\| : j \in [k]\} + \max\{\left|\hat{w}_j - w_{\pi(j)}\right| : j \in [k]\}\right) \leq \varepsilon.$$

*where the* min *is over permutations $\pi$ on $[k]$.*

The aforementioned theorem provides polynomial time guarantee as long as the value of $k$ remains a constant. The two algorithmic results mentioned above (Theorem 2.2 and Theorem 2.3) both have a polynomial dependence on the ambient dimension $n$, but they trade off different exponential dependencies on $k$ and the separation $\Delta$.

Specifically, when $\Delta = \omega(\sqrt{\log k/k})$, Theorem 2.2 provides a faster algorithm that becomes quasi-polynomial time when $\Delta = \Omega(1)$. On the other hand, Theorem 2.3 gives a faster algorithm when $\Delta = o(\sqrt{\log k/k})$. Even when $\Delta$ has an inverse polynomial dependence on $n$, the algorithm remains polynomial time for $k = O(1)$.

**Identifiability.** The algorithmic results presented above successfully achieve the unique identification and recovery of individual parameters, as opposed to simply finding a distribution that fits the data. In statistical terms, our algorithm recovers the underlying model (sometimes referred to as *parameter estimation*), rather than just doing *density*

*estimation*. The identifiability of results is guaranteed as long as $\Delta > 0$. However, it should be noted that the model is not identifiable when $\Delta = 0$. This is because the distribution induced by an equal weight mixture of two linear classifiers, where $v_1 = u$ and $v_2 = -u$, is the same for every $u \in \mathbb{R}^n$. Moreover, when $v_i = v_j$, non-identifiability arises due to the arbitrary redistribution of the weights of the two components. Thus, our results demonstrate identifiability with provable minimal assumptions.

**Comparison to Prior work.** Mixtures of supervised learning models, such as linear classifiers and other linear models, have been extensively studied in machine learning literature [Jacobs et al., 1991, Chaganty and Liang, 2013, Gandikota et al., 2020, Chen et al., 2020]. This line of research has included the study of hierarchical mixtures of experts [Jacobs et al., 1991, Jordan and Jacobs, 1994, Bishop, 1998]. A result closely related to ours is that of Sun et al. [2014], whose model is the same as ours. Their main result shows that if the vectors $v_1, \ldots, v_k$ are linearly independent, then there is a polynomial time algorithm that recovers the $k$-dimensional subspace spanned by the vectors $v_1, \ldots, v_k$. However, their algorithm does not recover the parameters of the model. To the best of our knowledge, there were no identifiability results for the model for general $k$ until our work.

Our algorithms successfully recover all unknown parameters of the mixture, thus implying identifiability. The running time of the algorithms is either $n^{O(\log k)/\Delta^2}$ or $n^{O(k)}$. Furthermore, when the vectors $v_1, \ldots, v_k$ are linearly independent, as in [Sun et al., 2014], our Theorem 2.1 algorithm successfully recovers all parameters in polynomial time.

**Overview of techniques.**

We now briefly describe the algorithmic ideas and techniques that we will need to establish Theorem 2.2 and Theorem 2.3. Our algorithms are based on the method-of-moments framework and use tensor decompositions to recover the parameters of the model. Our algorithmic results all use the same algorithmic framework, that consists of two main parts:

(i) *Extracting the low-rank tensor:* We first design a procedure that gives a good estimate for any $\ell \in \mathbb{N}$,

$$T = \sum_{j=1}^{k} w_j v_j^{\otimes(2\ell+1)}, \tag{2.1}$$

which is an order $2\ell + 1$ tensor with a rank-$k$ decomposition with one rank-1 term for each component.

(ii) *Parameter recovery through tensor decomposition:* We use an off-the-shelf algorithm for low-rank tensor decomposition, and show that they recover the parameters successfully.

**(i) Extracting the low-rank tensor.** Unlike latent variable models like mixtures of Gaussians [Moitra and Valiant, 2010, Janzamin et al., 2019], it is challenging to obtain a low-rank tensor by simply estimating the moments, for a linear threshold function (linear classifier). In order to estimate $\sum_{j=1}^{k} w_j v_j^{\otimes(2\ell+1)}$, we will instead use Hermite polynomials and consider coefficients of linear threshold functions in the Hermite basis. For a unit vector $v \in \mathbb{R}^n$, define $\mathbf{D}(v)$ be the distribution corresponding to $\mathcal{N}(0, I_n)$ conditioned on $\{x : \mathbb{1}_{\langle v,x\rangle \geq 0}\}$. The key observation is that:

$$\mathbb{E}_{\mathbf{x}\sim\mathbf{D}(v)}[\mathsf{He}^{(2\ell+1)}(\mathbf{x})] \propto v^{\otimes(2\ell+1)},$$

where $\mathsf{He}^{(2\ell+1)}(\cdot)$ is the $(2\ell+1)$th order n-variable Hermite tensor (and the constant of proportionality is non-zero). See Definition 2.9 for a formal definition. We remark that Hermite polynomials have been used in a similar vein in the context of other learning problems like depth-2 neural networks [Janzamin et al., 2015, Ge et al., 2018, Awasthi et al., 2021].

Let $\mathbf{D}$ be the distribution of the positively-labeled samples. Note that $\mathbf{D}$ is just convex combination of $\mathbf{D}(v_1), \ldots, \mathbf{D}(v_k)$. Hence

$$\mathbb{E}_{\mathbf{x}\sim\mathbf{D}}[\mathsf{He}^{(2\ell+1)}(\mathbf{x})] \propto \sum_{j=1}^{k} w_j v_j^{\otimes(2\ell+1)}.$$

The above relation naturally suggests the following meta-algorithm. We acquire a few i.i.d. positive-labeled samples, the output will be the (rescaled) empirical mean of $(2\ell+1)$th order Hermite tensor evaluation. This is described in the Algorithm 1. We prove the following guarantee for estimating the tensor in Section 2.3.

**Theorem 2.4.** *There is an algorithm* Extracting the low-rank tensor *that for a given $k$, $\ell$, error tolerance parameters $\varepsilon, \delta > 0$ $\mathcal{O}(v_1, \cdots, v_k, w_1, \cdots, w_k)$, and access to samples from the model has the following guarantees:*

(1) *The algorithm runs in sample complexity and time complexity* $\log^2(1/\delta)/(\varepsilon^2) \cdot n^{O(\ell)}\ell^{O(\ell)}$.

(2) *With probability* $1-\delta$*, the algorithm returns estimates* $\mathbf{T} \in (\mathbb{R}^n)^{\otimes\ell}$ *such that*

$$\|\mathbf{T} - \left(\sum_{j=1}^{k} w_j v_j^{\otimes(2\ell+1)}\right)\|_F \leq \varepsilon.$$

**(ii) Recovering the parameters through tensor decompositions**

Once we have access to the tensor in (2.1), we use an off-the-shelf algorithm for efficient tensor decompositions (see Theorem 2.22). Polynomial algorithms exist for decomposing a rank-$k$ tensor of the form in (2.1) as long as the flattened vectors given by $\{v_i^{\otimes \ell} : i \in [k]\}$ are linearly independent (in a robust sense). This is encapsulated in Theorem 2.28, which establishes Theorem 2.1. To establish Theorem 2.2 and Theorem 2.3, we need to prove that the (robust) linear independence condition holds for a sufficiently large value of $\ell$.

To prove Theorem 2.2, we show that $\ell = O(\log k/\Delta^2)$ suffices for $\{v_i^{\otimes \ell} : i \in [k]\}$ to be linearly independent.

The key observation is that for $i \neq j$, $\langle v_i^{\otimes \ell}, v_j^{\otimes \ell} \rangle = (\langle v_i, v_j \rangle)^\ell$ decreases exponentially as $\ell$ grow. In fact, if the pairwise inner product of $v_1^{\otimes \ell}, \ldots, v_k^{\otimes \ell}$ is at most $1/(2k)$, we can show that $v_1^{\otimes \ell}, \ldots, v_k^{\otimes \ell}$ is robustly linear independent. This gives a running time of $n^{O(\log k)/\Delta^2}$.

To prove Theorem 2.3, we use a different approach to prove that $\ell = k$ suffices for $v_1^{\otimes \ell}, \ldots, v_k^{\otimes \ell}$ to be linear independent in a robust sense. In particular, we use the notion of Kruskal rank to quantify the degree of linear independence with tensoring. The Kruskal rank (or Krank) of a matrix $A$ is the largest $k$ for which every set of $k$ columns are linearly independent. The Khatri-Rao product of $U$ and $V$ which are size $m \times r$ and $n \times r$ respectively is an $mn \times r$ matrix $U \odot V$ whose $i^{th}$ column is flattened $u_i \otimes v_i$. Let $A \in R^{n \times k}$ be the matrix whose $j$th column is $v_j$. Observe that pairwise linear independence implies $\text{Krank}(A) \geq 2$. Let $U \in \mathbb{R}^{n^\ell \times k}$ be the matrix whose $j$th column is flattened $v_j^{\otimes \ell}$. Observe that $U = A^{\odot k}$.

The idea is that Kruskal-rank increases with Khatri–Rao product. As a consequence, $\text{Krank}(U) \geq k$, i.e., $v_1^{\otimes k}, \ldots, v_k^{\otimes k}$ are linear independent, thus establishing Theorem 2.3.

## 2.2. Preliminaries and Notation

We start by defining the Mixture-of-Linear-Classifier problem formally.

**Definition 2.5.** *The Mixture-of-Linear-Classifier is instantiated by $k$ unit vectors $v_1, \cdots, v_k$ in $\mathbb{R}^n$. In addition, we also have $k$ corresponding weights $w_1, \cdots, w_k$ such that $w_1 + \cdots + w_k = 1$.*

*The vectors $v_1, \cdots, v_k$ in $\mathbb{R}^n$ as well as the weights $w_1, \cdots, w_k$ are unknown. For this instance, the sampling oracle $\mathcal{O}(v_1, \cdots, v_k, w_1, \cdots, w_k)$ is defined as follows: sample $\mathbf{x} \sim \mathcal{N}(0, I_n)$, the standard spherical Gaussian in $\mathbb{R}^n$. Sample $\mathbf{z} \in [k]$ where $\mathbb{P}[\mathbf{z} = j] = w_j, \forall j \in [k]$. $\mathcal{O}(v_1, \cdots, v_k, w_1, \cdots, w_k)$ outputs $(\mathbf{x}, \mathbb{1}_{\langle \mathbf{x}, v_{\mathbf{z}} \rangle \geq 0}) \in \mathbb{R}^n \times \{0, 1\}$.*

*In the Mixture-of-Linear-Classifier problem, the algorithm is given access to the number of component $k$, the sample oracle $\mathcal{O}(v_1, \cdots, v_k, w_1, \cdots, w_k)$, an error parameter $\varepsilon$ and a weight parameter $w_{\min} > 0$ with the promise that $w_{\min} \leq \min\{w_1, \cdots, w_k\}$. The goal of the algorithm is to output estimates $\{\hat{w}_j, \hat{v}_j : j \in [k]\}$ such that*

$$\min_{\pi} \left( \max\{\|\hat{v}_j - v_{\pi(j)}\| : j \in [k]\} + \max\{\left|\hat{w}_j - w_{\pi(j)}\right| : j \in [k]\} \right) \leq \varepsilon,$$

*where the* min *is over all permutations on $[k]$.*

**Notation.** We use $\nabla_t^{(d)}$ to denote the $d$-th order differential operator (with respect to $t$).

We next definite Hermite polynomials. We begin with univariate Hermite polynomials.

**Definition 2.6.** *The $d^{th}$ univariate Hermite polynomial $\mathsf{He}_d(x) : \mathbb{R} \to \mathbb{R}$ is the formal polynomial*

$$\mathsf{He}_d(x) = \left( \left( \frac{\partial}{\partial t} \right)^d \exp(xt - t^2/2) \right) \Bigg|_{t=0}.$$

The following observation gives a recursive relation between $\mathsf{He}_d$ and $\mathsf{He}_{d+1}$.

**Observation 2.7.** *For $d \in \mathbb{Z}_{\geq 0}$,*

$$\frac{d}{dx} \left( \mathsf{He}_d(x) e^{-x^2/2} \right) = -\mathsf{He}_{d+1}(x) e^{-x^2/2}$$

**Proof.** We use Rodrigues formula for the Hermite polynomial [see e.g., equation (10) of Patarroyo, 2019], which is:

$$\mathsf{He}_d(x) = (-1)^d e^{\frac{x^2}{2}} \left( \frac{d}{dx} \right)^d \left( e^{\frac{-x^2}{2}} \right).$$

Or equivalently,

$$\mathsf{He}_d(x) e^{-\frac{x^2}{2}} = (-1)^d \left( \frac{d}{dx} \right)^d \left( e^{\frac{-x^2}{2}} \right).$$

The claim now follows easily. □

The next observation gives an explicit formula for univariate Hermite polynomials.

**Observation 2.8.** *[see equation (3) of Patarroyo, 2019] For $d \in \mathbb{Z}_{\geq 0}$, we have*

$$\mathsf{He}_d(x) = d! \sum_{j=0}^{\lfloor \frac{d}{2} \rfloor} \frac{(-1)^j}{2^j (d-2j)! j!} x^{d-2j}$$

Next, we define multivariable Hermite polynomials.

**Definition 2.9.** *For $n \in \mathbb{N}$,$d \in \mathbb{Z}_{\geq 0}$. We use $\mathsf{He}^{(d)}$ to denote the $n$-variable* Hermite Tensor *of order $d$. $\mathsf{He}^{(d)} : \mathbb{R}^n \to (\mathbb{R}^n)^{\otimes d}$ is defined as:*

$$\mathsf{He}^{(d)}(x) = \left(\nabla_t^{(d)} \exp(\langle x, t\rangle - \|t\|^2/2)\right)\Big|_{t=0}.$$

## 2.3. Extracting the Low-rank Tensor

In this section, the main goal is to prove Theorem 2.4. Theorem 2.4 gives an algorithm which given samples from a mixture of $k$ linear classifiers, estimates the order-$\ell$ parameter moment. This result is an important piece in our parameter recovery algorithm. The algorithm Extracting the low-rank tensor is described in Algorithm 1.

**Theorem 2.10.** *There is an algorithm* Extracting the low-rank tensor *that for a given $k$, $\ell$, error tolerance parameters $\varepsilon, \delta > 0$ $\mathcal{O}(v_1, \cdots, v_k, w_1, \cdots, w_k)$, and access to samples from the model has the following guarantees:*

(1) *The algorithm runs in sample complexity and time complexity* $\log^2(1/\delta)/(\varepsilon^2) \cdot n^{O(\ell)}\ell^{O(\ell)}$.

(2) *With probability $1-\delta$, the algorithm returns estimates $\mathbf{T} \in (\mathbb{R}^n)^{\otimes \ell}$ such that*

$$\|\mathbf{T} - \left(\sum_{j=1}^{k} w_j v_j^{\otimes(2\ell+1)}\right)\|_F \leq \varepsilon.$$

The main idea behind the algorithm is to show that over the positive samples, the expectation of the $(2\ell+1)$th-order Hermite Tensor is proportional to $\sum_{j\in[k]} w_j v_j^{\otimes(2\ell+1)}$. Based on this, our algorithm is to just output an empirical estimator for the average $(2\ell+1)$th-order Hermite tensor. The rest of this section is dedicated to proving the correctness of Algorithm 1 (Theorem 2.4). Towards this, we start with some definitions.

**Algorithm 1:** EXTRACTING THE LOW-RANK TENSOR

**Input:**
$k$ – number of components
$\mathcal{O}(v_1, \cdots, v_k, w_1, \cdots, w_k)$ – oracle for random samples from the mixture
$\ell$ – parameter for order of the tensor
$\varepsilon$ – error parameter
**Output:**
$\mathbf{T} \in (\mathbb{R}^n)^{\otimes(2\ell+1)}$ – estimate of $\sum_{j=1}^{k} w_j v_j^{\otimes(2\ell+1)}$

1 Set $t = (\varepsilon^{-2})n^{O(\ell)}\ell^{O(\ell)}$;
2 Use $\mathcal{O}(v_1, \cdots, v_k, w_1, \cdots, w_k)$ to sample $t$ independent vectors $\mathbf{x}_1, \ldots, \mathbf{x}_t$ from $\mathbf{D}$;
3 **return** $\mathbf{T} = 1/t \sum_{j\in[t]} 1/c(\ell)\mathsf{He}^{(2\ell+1)}(\mathbf{x}_j)$, *where* $c(\ell) = \sqrt{2/\pi}(-1)^\ell(2\ell-1)!!$;

**Definition 2.11.** *Let $v \in \mathbb{R}^n$, $\mathbf{x} \sim \mathcal{N}(0, I_n)$. Define $\mathbf{D}(v)$ as the conditional distribution of $\mathbf{x}$ given $\langle v, \mathbf{x}\rangle \geq 0$. Or equivalently, the probability density function of $\mathbf{D}(v)$ is given by*

$$(2\pi)^{-n/2} e^{-\|z\|^2/2} \cdot 2\mathbb{1}_{\langle v,z\rangle \geq 0}.$$

*Define $\mathbf{D}$ to be the distribution corresponding to positive samples from $\mathcal{O}(v_1, \cdots, v_k, w_1, \cdots, w_k)$. Or equivalently, $\mathbf{D} = \sum_{j\in[k]} w_j \mathbf{D}(v_j)$.*

The following observation says that the $\ell$-th order derivative (with respect to $t$) of $f(\langle v, t\rangle)$ is proportional to $v^{\otimes\ell}$. It can be easily derived from the chain rule.

**Observation 2.12.** *Let $f : \mathbb{R} \to \mathbb{R}$ be infinitely differentiable, $v \in \mathbb{R}^n$, $\ell \in \mathbb{N}$. Then,*

$$\nabla_t^{(\ell)}(f(\langle v, t\rangle)) = f^{(\ell)}(\langle v, t\rangle) \cdot v^{\otimes\ell}$$

Next, we define a function $\Psi(t)$ (which is the mass that the Gaussian centered at $t$ puts on $[0, \infty)$) and obtain an explicit formula for its derivatives of odd order.

**Claim 2.13.** *Define* $\Psi : \mathbb{R} \to \mathbb{R}$ *to be*

$$\Psi(t) = \int_{\mathbb{R}} \exp\left(-(x-t)^2/2\right) \mathbb{1}_{x \geq 0} dx.$$

*For all* $\ell \in \mathbb{Z}_{\geq 0}$*, we have*

$$\Psi^{(2\ell+1)}(0) = (-1)^{\ell}(2\ell-1)!!$$

**Proof.** To prove the above equality, we first swap the integration with differentiation and relate the resulting expression to Hermite polynomials.

We know that

$$\Psi(t) = \int_0^{\infty} \exp\left(-(x-t)^2/2\right) dx = \int_0^{\infty} \exp\left(tx - t^2/2\right) \exp\left(-x^2/2\right) dx.$$

$$\text{Hence, } \Psi^{(2\ell+1)}(0) = \left(\left(\frac{\partial}{\partial t}\right)^{2\ell+1}\Psi\right)\Bigg|_{t=0}$$
$$= \left(\int_0^\infty \left(\frac{\partial}{\partial t}\right)^{2\ell+1} \exp\left(tx - t^2/2\right)\exp\left(-x^2/2\right)dx\right)\Bigg|_{t=0}$$

by Leibniz integral rule

$$= \int_0^\infty \left(\left(\frac{\partial}{\partial t}\right)^{2\ell+1} \exp\left(tx - t^2/2\right)\right)\Bigg|_{t=0} \exp\left(-x^2/2\right)dx$$
$$= \int_0^\infty \mathsf{He}_{2\ell+1}(x)\exp\left(-x^2/2\right)dx$$
$$= \int_0^\infty d\left(-\mathsf{He}_{2\ell}(x)\exp\left(-x^2/2\right)\right)$$

see Observation 2.7

$$= \mathsf{He}_{2\ell}(0) = (-1)^\ell(2\ell-1)!!$$

by Observation 2.8

□

The following lemma is crucial in establishing Theorem 2.4. The lemma proves that the expectation of $(2\ell+1)$th-order Hermite Tensor over $\mathbf{D}$ is proportional to $\sum_{j\in[k]} w_j v_j^{\otimes(2\ell+1)}$.

**Lemma 2.14.** *For $\ell \in \mathbb{Z}_{\geq 0}$,*

$$\mathbb{E}_{\mathbf{x}\sim\mathbf{D}}[1/c(\ell)\mathsf{He}^{(2\ell+1)}(\mathbf{x})] = \sum_{j\in[k]} w_j v_j^{\otimes(2\ell+1)},$$

*where $c(\ell) = \sqrt{2/\pi}(-1)^\ell(2\ell-1)!!$.*

**Proof.** The high-level idea is to reduce the problem to the case $k = 1$. In particular, since $\mathbf{D} = \sum_{j\in[k]} w_j \mathbf{D}(v_j)$, it suffices to show that: for all unit vector $v \in \mathbb{R}^n$,

$$\mathbb{E}_{\mathbf{x}\sim\mathbf{D}(v)}[\mathsf{He}^{(2\ell+1)}(\mathbf{x})] = \sqrt{\frac{2}{\pi}}(-1)^\ell(2\ell-1)!! \cdot v^{\otimes(2\ell+1)}.$$

Towards this, recall that,

$$\mathsf{He}^{(2\ell+1)}(x) = \left(\nabla_t^{(2\ell+1)} \exp(\langle x, t\rangle - \|t\|^2/2)\right)\Big|_{t=0}.$$

Then,

$$\begin{aligned}
&\mathbb{E}_{\mathbf{x}\sim\mathbf{D}(v)}[\mathsf{He}^{(2\ell+1)}(\mathbf{x})] \\
&= 2(\frac{1}{\sqrt{2\pi}})^n \int_{\mathbb{R}^n} \mathsf{He}^{(2\ell+1)}(x)\mathbb{1}_{\langle x,v\rangle\geq 0} \exp\left(-\|x\|^2/2\right)dx \quad \text{using Definition 2.11} \\
&= 2(\frac{1}{\sqrt{2\pi}})^n \int_{\mathbb{R}^n} \left(\nabla_t^{(2\ell+1)} \exp(\langle x, t\rangle - \|t\|^2/2)\right)\Big|_{t=0} \mathbb{1}_{\langle x,v\rangle\geq 0} \exp\left(-\|x\|^2/2\right)dx \\
&= 2(\frac{1}{\sqrt{2\pi}})^n \left(\nabla_t^{(2\ell+1)} \int_{\mathbb{R}^n} \exp\left(-\|x-t\|^2/2\right)\mathbb{1}_{\langle x,v\rangle\geq 0}dx\right)\Big|_{t=0} \text{by Leibniz integral rule}
\end{aligned}$$

Let $U$ be an orthonormal matrix such that the first row equals $v^T$. Thus, we have

$$\begin{aligned}
&\mathbb{E}_{\mathbf{x}\sim\mathbf{D}(v)}[\mathsf{He}^{(2\ell+1)}(\mathbf{x})] \\
&= 2(\frac{1}{\sqrt{2\pi}})^n \left(\nabla_t^{(2\ell+1)} \int_{\mathbb{R}^n} \exp\left(-\|y-Ut\|^2/2\right)\mathbb{1}_{y_1\geq 0}dy\right)\Big|_{t=0} && \text{change of variables, } y = Ux \\
&= \frac{2}{\sqrt{2\pi}} \left(\nabla_t^{(2\ell+1)} \int_{\mathbb{R}} \exp\left(-(y_1-\langle v,t\rangle)^2/2\right)\mathbb{1}_{y_1\geq 0}dy_1\right)\Big|_{t=0} \\
&= \frac{2}{\sqrt{2\pi}} \left(\nabla_t^{(2\ell+1)} \Psi(\langle v,t\rangle)\right)\Big|_{t=0} && \text{by Claim 2.13} \\
&= \frac{2}{\sqrt{2\pi}} \left(\Psi^{(2\ell+1)}(\langle v,t\rangle)\cdot v^{\otimes(2\ell+1)}\right)\Big|_{t=0} && \text{by Observation 2.12} \\
&= \sqrt{\frac{2}{\pi}}(-1)^\ell(2\ell-1)!!v^{\otimes(2\ell+1)} && \text{by Claim 2.13}
\end{aligned}$$

□

Our next goal is to bound the variance of our estimator in Algorithm 1. Towards this, we start with the following simple claim.

**Claim 2.15.** *Let $f:\mathbb{R}^n\to\mathbb{R}$ such that $f(x)=f(-x),\forall x\in\mathbb{R}^n$. Then,*

$$\mathbb{E}_{\mathbf{x}\sim\mathbf{D}}[f(\mathbf{x})] = \mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,I_n)}[f(\mathbf{x})]$$

**Proof.** First, using the fact that the distribution $1/2(\mathbf{D}(v_j)+\mathbf{D}(-v_j))$ is $\mathcal{N}(0,I_n)$ and $f$ is even, it follows that

$$\mathbb{E}_{\mathbf{x}\sim\mathbf{D}(v_j)}[f(\mathbf{x})] = \mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,I_n)}[f(\mathbf{x})].$$

We now get the claim by noting that $\mathbf{D}$ is a convex combination of $\mathbf{D}(v_1), \ldots, \mathbf{D}(v_k)$.

□

Next, we upper bound the variance of a polynomial under the distribution $\mathbf{D}$.

**Claim 2.16.** *Let $s \in \mathbb{N}$. Let $m_1, \ldots, m_s : \mathbb{R}^n \to \mathbb{R}$ be monomials of degree at most $t$. Let $\alpha = (\alpha_1, \ldots, \alpha_s) \in \mathbb{R}^s$. Then,*

$$\mathbb{E}_{\mathbf{x}\sim\mathbf{D}}[(\alpha_1 m_1(\mathbf{x}) + \ldots + \alpha_s m_s(\mathbf{x}))^2] \leq s(2t-1)!!\|\alpha\|^2.$$

**Proof.** The idea is to apply Claim 2.15. Claim 2.15 which lets us reduce the problem of computing the variance under $\mathbf{D}$ to that under the standard Gaussian.

$$\begin{aligned}
&\mathbb{E}_{\mathbf{x}\sim\mathbf{D}}[(\alpha_1 m_1(\mathbf{x}) + \ldots + \alpha_s m_s(\mathbf{x}))^2] \\
&\leq \mathbb{E}_{\mathbf{x}\sim\mathbf{D}}[s(\sum_{j\in[s]} \alpha_j^2 m_j(x)^2)] && \text{by Cauchy–Schwarz inequality} \\
&= \mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,I_n)}[s(\sum_{j\in[s]} \alpha_j^2 m_j(x)^2)] && \text{by Claim 2.15} \\
&\leq s(2t-1)!!\|\alpha\|^2.
\end{aligned}$$

The last inequality follows from the fact that $m_j(x)^2$ is a monomial of degree at most $2t$ and thus its expectation under a Gaussian is at most $(2t-1)!!$. It is well-known that $\mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,1)}[x^{2d}] = (2d-1)!!, \forall d \in \mathbb{N}$. A standard induction will give us the above fact. □

The next proposition shows that $\mathrm{Var}_{\mathbf{x}\sim\mathbf{D}}[\mathsf{He}_\alpha^{(2\ell+1)}(\mathbf{x})]$ is at most $\ell^{O(\ell)}$ for any fixed index $\alpha$.

**Proposition 2.17.** *For $\ell \in \mathbb{N}$ with $\ell \geq 2$, fix $\alpha \in [n]^{2\ell+1}$,*

$$\operatorname*{Var}_{\mathbf{x}\sim\mathbf{D}}\left[\tfrac{1}{c(\ell)} \cdot \mathsf{He}_\alpha^{(2\ell+1)}(\mathbf{x})\right] \leq \ell^{c_1\ell},$$

*where $c(\ell) = \sqrt{2/\pi}(-1)^\ell(2\ell-1)!!$ and $c_1$ is an absolute constant.*

**Proof.** We begin by noting that

$$\operatorname*{Var}_{\mathbf{x}\sim\mathbf{D}}[\mathsf{He}_\alpha^{(2\ell+1)}(\mathbf{x})] \leq \mathbb{E}_{\mathbf{x}\sim\mathbf{D}}\left[\left(\mathsf{He}_\alpha^{(2\ell+1)}(\mathbf{x})\right)^2\right]$$

By definition, $\mathsf{He}_\alpha^{(2\ell+1)}(x)$ can be expressed as $\prod_{j=1}^n \mathsf{He}_{s_j}(x_j)$ where:

(1) $s_1, \ldots, s_n \in \mathbb{Z}_{\geq 0}$ depend on $\alpha$.

(2) $\sum_{j=1}^n s_j = 2\ell + 1$.

Hence $\mathsf{He}_\alpha^{(2\ell+1)}(x)$ is a polynomial of degree at most $2\ell+1$ and can be expanded as $\beta_1 m_1(x) + \ldots + \beta_z m_z(x)$ where:

(1) $m_1, \ldots, m_z$ are monomials,

(2) $z \leq (2\ell+1)^{(2\ell+1)}$,

(3) for all $j \in [z]$, $|\beta_j| \leq (2\ell+1)!$.

The second item is true because $\mathsf{He}_s(\cdot)$ has at most $2s+1$ terms (see Observation 2.8) and $\mathsf{He}_\alpha^{(2\ell+1)}(x)$ can be expressed as $\prod_{j=1}^n \mathsf{He}_{s_j}(x_j)$. The last item is true because every coefficient of $\mathsf{He}_s$ is bounded by $s!$ (see Observation 2.8) and $\prod_{j=1}^n s_j! \leq (\sum_{j=1}^n s_j)! = (2\ell+1)!$.

Apply Claim 2.16, we have

$$\mathbb{E}_{\mathbf{x}\sim\mathbf{D}}[\mathsf{He}_\alpha^{(2\ell+1)}(\mathbf{x})^2] \leq z(4\ell+1)!!(z\left((2\ell+1)!\right)^2) \leq \ell^{c'\ell},$$

where $c'$ is an absolute constant. $\square$

We are now ready to finish the proof of Theorem 2.4.

**Proof of Theorem** 2.4**.** Without loss of generality, we can assume $\delta = 0.1$, since we can always boost the success probability at a multiplicative cost of $O(\log(1/\delta)^2)$ via Claim *A*.1. Define $T^* = \sum_{j=1}^{k} w_j v_j^{\otimes(2\ell+1)}$. We will show $\mathbf{T}$ is close to $T^*$ with probability at least 0.9, where $\mathbf{T}$ is the empirical mean of $1/c(\ell)\mathsf{He}^{(2\ell+1)}(\mathbf{x})$. By Lemma 2.14, $T^* = \mathbb{E}_{\mathbf{x}\sim\mathbf{D}}[1/c(\ell)\mathsf{He}^{(2\ell+1)}(\mathbf{x})]$, hence $T^* = \mathbb{E}[\mathbf{T}]$. Fix $\alpha \in [n]^{2\ell+1}$, from Proposition 2.17 we know that

$$\begin{aligned}\mathrm{Var}[\mathbf{T}_\alpha] &= \frac{1}{t}\operatorname*{Var}_{\mathbf{x}\sim\mathbf{D}}[1/c(\ell)\mathsf{He}_\alpha^{(2\ell+1)}(\mathbf{x})] \\ &\le \ell^{c_1\ell}/t,\end{aligned}$$

where $c_1$ is an absolute constant. By Chebyshev's inequality,

$$\mathbb{P}[|\mathbf{T}_\alpha - T^*_\alpha| \ge \frac{\varepsilon}{\sqrt{n^{2\ell+1}}}] \le \frac{\ell^{c_1\ell}/t}{\frac{\varepsilon^2}{n^{2\ell+1}}} = \frac{\ell^{c_1\ell} n^{2\ell+1}}{t\varepsilon^2}$$

Then,

$$\begin{aligned}\mathbb{P}[\vee_{\alpha\in[n]^{2\ell+1}}\{|\mathbf{T}_\alpha - T^*_\alpha| \ge \frac{\varepsilon}{\sqrt{n^{2\ell+1}}}\}] &\le \frac{\ell^{c_1\ell} n^{4\ell+2}}{t\varepsilon^2} \\ &\le 0.01 \qquad\qquad \text{by the choice of } t.\end{aligned}$$

Hence,

$$\mathbb{P}[\wedge_{\alpha\in[n]^{2\ell+1}}\{|\mathbf{T}_\alpha - T^*_\alpha| < \frac{\varepsilon}{\sqrt{n^{2\ell+1}}}\}] \ge 0.99$$

As a result,

$$\mathbb{P}[\|\mathbf{T} - T^*\|_F \leq \varepsilon] \geq 0.99$$

□

## 2.4. Estimation Algorithm for the Parameters of the Mixture of Linear Classifiers

In this section, we prove Theorem 2.1, Theorem 2.2 and Theorem 2.3. Theorem 2.2 shows that there is an algorithm learns a mixture of $k$ linear classifiers in polynomial time, provided that $v_1, v_2, \ldots, v_k$ are linear independent. Recall that $\Delta = \min_{j \neq j'} \min\{\|v_j - v_{j'}\|, \|v_j + v_{j'}\|\}$. Theorem 2.2 shows that there is an algorithm learns a mixture of $k$ linear classifiers in time $\text{poly}(n^{(\log k)/\Delta^2})$, where $\Delta$ is the minimum "separation" between each pair of linear classifiers. Meanwhile, Theorem 2.3 shows that there is an algorithm that does the same thing in time that is roughly $\text{poly}((n/\Delta)^k)$. When $\Delta = \omega(\sqrt{\log k/k})$, Theorem 2.2 gives a faster algorithm. Meanwhile, Theorem 2.3 gives a faster algorithm when $\Delta = o(\sqrt{\log k/k})$.

**Theorem 2.18** (restatement of Theorem 2.1)**.** *Let $U \in \mathbb{R}^{n \times k}$ be the matrix whose $j$th column is $v_j$. Suppose $\sigma_{\min}(U) \geq 1/\tau$, where $\tau > 0$. Given parameters $\varepsilon, \delta > 0$, $k \in \mathbb{N}$ and $w_{\min} > 0$ satisfying $w_{\min} \leq \min\{w_1, \ldots, w_k\}$, there is an algorithm* Estimate-Parameter *that given samples from the model has the following guarantees:*

(1) *The algorithm runs in sample complexity and time complexity*

$$\log^2(1/\delta)\varepsilon^{-2}\mathrm{poly}(n, \tau, 1/w_{\min}).$$

(2) *With probability* $1-\delta$*, the algorithm returns estimates* $\{\hat{w}_j, \hat{v}_j : j \in [k]\}$ *such that*

$$\min_{\pi \in \mathrm{Perm}([k])} \left(\max\{\|\hat{v}_j - v_{\pi(j)}\| : j \in [k]\} + \max\{\left|\hat{w}_j - w_{\pi(j)}\right| : j \in [k]\}\right) \leq \varepsilon.$$

*where the* min *is over permutations* $\pi$ *on* $[k]$.

**Theorem 2.19** (restatement of Theorem 2.2)**.** *Given parameters* $\varepsilon, \delta > 0$*,* $k \in \mathbb{N}$ *and* $w_{\min} > 0$ *satisfying* $w_{\min} \leq \min\{w_1, \ldots, w_k\}$*, there is an algorithm that given samples from the model has the following guarantees:*

(1) *The algorithm runs in sample complexity and time complexity*

$$\log^2(1/\delta)\varepsilon^{-2}\mathrm{poly}(n^{(\log k)/\Delta^2}, ((\log k)/\Delta^2)^{(\log k)/\Delta^2}, 1/w_{\min}).$$

(2) *With probability* $1-\delta$*, the algorithm returns estimates* $\{\hat{w}_j, \hat{v}_j : j \in [k]\}$ *such that*

$$\min_{\pi \in \mathrm{Perm}([k])} \left(\max\{\|\hat{v}_j - v_{\pi(j)}\| : j \in [k]\} + \max\{\left|\hat{w}_j - w_{\pi(j)}\right| : j \in [k]\}\right) \leq \varepsilon,$$

*where the* min *is the minimum is over permutations* $\pi$ *on* $[k]$.

**Theorem 2.20** (restatement of Theorem 2.3)**.** *Given parameters* $\varepsilon, \delta > 0$*,* $k \in \mathbb{N}$ *and* $w_{\min} > 0$ *satisfying* $w_{\min} \leq \min\{w_1, \ldots, w_k\}$*, there is an algorithm that given samples from the model has the following guarantees:*

(1) *The algorithm runs in sample complexity and time complexity*

$$\log^2(1/\delta)\varepsilon^{-2}\mathrm{poly}(n^k, k^k, \Delta^{-k}, 1/w_{\min})$$

(2) *With probability* $1-\delta$*, the algorithm returns estimates* $\{\hat{w}_j, \hat{v}_j : j \in [k]\}$ *such that*

$$\min_{\pi\in\mathrm{Perm}([k])} \left(\max\{\|\hat{v}_j - v_{\pi(j)}\| : j \in [k]\} + \max\{\left|\hat{w}_j - w_{\pi(j)}\right| : j \in [k]\}\right) \le \varepsilon.$$

*where the* min *is over permutations* $\pi$ *on* $[k]$.

The basic idea of the above theorems are the same. Roughly speaking, Theorem 2.22 (from Bhaskara et al. [2014a]) says we can decompose a noisy third-order low-rank tensor efficiently under some mild non-degeneracy conditions. By Theorem 2.4, we can estimate $T^* = \sum_{j\in[k]} w_j v_j^{\otimes(2\ell+1)}$ accurately. Note that $T^*$ can be viewed as a third-order low-rank tensor $\sum_{j\in[k]}(v_j^{\otimes\ell}) \otimes (v_j^{\otimes\ell}) \otimes (w_j v_j)$. Our approach will be combining Theorem 2.4 and Theorem 2.22.

In order to combine Theorem 2.4 and Theorem 2.22, $\sum_{j\in[k]}(v_j^{\otimes\ell})\otimes(v_j^{\otimes\ell})\otimes(w_j v_j)$ needs to satisfy the conditions of Theorem 2.22. The major challenge is to show that $v_1^{\otimes\ell}, \ldots, v_k^{\otimes\ell}$ are linear independent in a robust sense (that is measured in terms of the least singular value of the $n^\ell \times k$ matrix formed by the flattenings of these $k$ tensored vectors as columns). The assumption of Theorem 2.1 incorporates this condition as a fundamental requirement. Theorem 2.2 and Theorem 2.3 use different approaches to establish this condition. On the one hand, Claim 2.23 shows that $\ell = 10(\log k)/\Delta^2$ suffices. This leads to Theorem 2.2. On the other hand, Claim 2.26 shows that $\ell = k$ suffices (even when $\Delta$ can be a small inverse polynomial in $n$). This leads to Theorem 2.3.

We start by introducing the concept of Kruskal rank for convenience.

**Definition 2.21** (Definition 1.2, Bhaskara et al. [2014a]). *The Kruskal rank (or Krank) of a matrix $A$ is the largest $k$ for which every set of $k$ columns are linearly independent. Also the $\tau$-*robust Krank *is denoted by $Krank_\tau(A)$, and is the largest $k$ for which every $n \times k$ sub-matrix $A_{|S}$ of $A$ has $\sigma_k(A_{|S}) \geq 1/\tau$.*

The following theorem from Bhaskara et al. [2014a] is crucial; see also [Janzamin et al., 2019, Goyal et al., 2014] for related tensor decomposition guarantees. Suppose $U \in \mathbb{R}^{m\times R}, V \in \mathbb{R}^{n\times R}, W \in \mathbb{R}^{p\times R}$. The theorem says that: if U, V are well-conditioned and columns of W are "pairwise well-conditioned", there is an algorithm that can recover all the rank-one terms from $T = \sum_{i=1}^{R} u_i \otimes v_i \otimes w_i$ efficiently. Moreover, the algorithm can also tolerate some inverse polynomial amount of noise.

**Theorem 2.22** (Theorem 2.3, Bhaskara et al. [2014a]). *Suppose $U \in \mathbb{R}^{m\times R}, V \in \mathbb{R}^{n\times R}, W \in \mathbb{R}^{p\times R}$. $u_i, v_i, w_i$ are the $i$th column of $U, V, W$, respectively. Suppose $U, V, W$ satisfy that:*

(1) *The condition numbers $\kappa(U), \kappa(V) \leq \kappa$,*

(2) *The column vectors of $W$ are not close to parallel: $Krank_{1/\delta}(W) \geq 2$,*

(3) *The decompositions are bounded : for all $i$, $\|u_i\|_2, \|v_i\|_2, \|w_i\|_2 \leq C$.*

*Suppose we are given tensor $T + E \in \mathbb{R}^{m\times n\times p}$ with the entries of $E$ being bounded by $\varepsilon \cdot \mathrm{poly}(1/\kappa, 1/m, 1/n, 1/p, \delta, 1/C)$ and moreover $T$ has a decomposition $T = \sum_{i=1}^{R} u_i \otimes v_i \otimes w_i$. There is an algorithm with the following guarantee:*

(1) *The algorithm runs in time complexity* $\mathrm{poly}(m, n, p)$.

(2) *With probability* 0.99*, the algorithm returns* each rank one term *in the decomposition of* $T$ *(up to renaming), within an additive error of* $\varepsilon$*.*

The next claim says the following. We can view $\sum_{j\in[k]} w_j v_j^{\otimes(2\ell+1)}$ as $\sum_{j\in[k]}(v_j^{\otimes\ell})\otimes(v_j^{\otimes\ell})\otimes(w_j v_j)$. If $\ell = 10(\log k)/\Delta^2$, the above tensor satisfy the condition of Theorem 2.22.

**Claim 2.23.** *Define* $\Delta = \min_{j\neq j'}\min\{\|v_j - v_{j'}\|, \|v_j + v_{j'}\|\}$*. Suppose* $\Delta > 0$*. Define* $\ell = 10(\log k)/\Delta^2$*. Consider* $\sum_{j\in[k]} w_j v_j^{\otimes(2\ell+1)} = \sum_{j\in[k]}(v_j^{\otimes\ell})\otimes(v_j^{\otimes\ell})\otimes(w_j v_j)$*. Let* $U \in \mathbb{R}^{n^\ell\times k}$ *be the matrix whose* $j$*th column is flattened* $v_j^{\otimes\ell}$*. Let* $W \in \mathbb{R}^{n\times k}$ *be the matrix whose* $j$*th column is* $w_j v_j$*. Then the following hold:*

(1) *The condition numbers* $\kappa(U) \leq \mathrm{poly}(k)$*,*

(2) *For all* $i \neq j$*, we have* $w_i v_i, w_j v_j$ *are not close to parallel:* $\mathit{Krank}_{2/(w_{\min}\Delta)}(W) \geq 2$*,*

(3) *For all* $j$*, we have* $\|v_j^{\otimes k}\|_F, \|w_j v_j\| \leq 1$*.*

**Proof.** *Proof of part (1):*

The main idea is the following. We have

$$\langle v_i^{\otimes\ell}, v_j^{\otimes\ell}\rangle = \begin{cases} 1 & i = j \\ \langle v_i, v_j\rangle^\ell & i \neq j \end{cases}$$

Since $\langle v_i, v_j\rangle^\ell \to 0$ as $\ell \to \infty$, we have that $U^T U \to I$ as $\ell \to \infty$. Hence we expect $\kappa(U)$ to be small if $\ell$ is sufficiently large.

Using the variational characterization for singular values:

$$\sigma_{\min}(U) = \|\alpha_1 v_1^{\otimes\ell} + \ldots + \alpha_k v_k^{\otimes\ell}\|_F$$

for some unit vector $(\alpha_1, \ldots, \alpha_k)$.

Without loss of generality, we assume

(1) $|\alpha_1|$ is the greatest one among $\{|\alpha_1|, \ldots, |\alpha_k|\}$,

(2) $\alpha_1 \geq 0$.

From Cauchy–Schwarz, we have

$$\begin{aligned} \|\alpha_1 v_1^{\otimes \ell} + \ldots + \alpha_k v_k^{\otimes \ell}\|_F &\geq \langle \alpha_1 v_1^{\otimes \ell} + \ldots + \alpha_k v_k^{\otimes \ell}, v_1^{\otimes \ell} \rangle \\ &= \alpha_1 + \alpha_2 \langle v_2^{\otimes \ell}, v_1^{\otimes \ell} \rangle + \ldots + \alpha_k \langle v_k^{\otimes \ell}, v_1^{\otimes \ell} \rangle \\ &= \alpha_1 \left(1 + \alpha_2/\alpha_1 \langle v_2^{\otimes \ell}, v_1^{\otimes \ell} \rangle + \ldots + \alpha_k/\alpha_1 \langle v_k^{\otimes \ell}, v_1^{\otimes \ell} \rangle\right) \end{aligned} \tag{2.2}$$

For any $j \neq 1$,

$$\begin{aligned} |\alpha_j/\alpha_1 \langle v_j^{\otimes \ell}, v_1^{\otimes \ell} \rangle| &\leq |\langle v_j^{\otimes \ell}, v_1^{\otimes \ell} \rangle| = |\langle v_j, v_1 \rangle|^\ell \\ &\leq (1 - \Delta^2/2)^\ell \qquad \text{since } \Delta = \min_{j \neq j'} \min\{\|v_j - v_{j'}\|, \|v_j + v_{j'}\|\} \\ &\leq \exp(-\Delta^2 \ell/2) \leq \frac{1}{2k}. \end{aligned}$$

Applying the above inequality along with (2.2), we get

$$\sigma_{\min}(U) \geq \alpha_1(1 - (k-1)/2k) \geq \frac{\alpha_1}{2} \geq \frac{1}{2\sqrt{k}}$$

Meanwhile

$$\|U\|^2 \leq \|U\|_F^2 = \sum_{j \in [k]} \|v_j^{\otimes k}\|_F^2 = k.$$

Therefore part (1) is true.

*Proof of part (2):* It follows by the definition $\Delta = \min_{j \neq j'} \min\{\|v_j - v_{j'}\|, \|v_j + v_{j'}\|\}$.

*Proof of part (3):* Recall that $\{v_j\}$ are unit vectors. □

Next, we introduce the concept of Khatri-Rao product for convenience.

**Definition 2.24** (Definition 1.3, Bhaskara et al. [2014a])**.** *The Khatri-Rao product of $U$ and $V$ which are size $m \times r$ and $n \times r$ respectively is an $mn \times r$ matrix $U \odot V$ whose $i^{th}$ column is flattened $u_i \otimes v_i$.*

The next lemma is a robust analogue of the following fact: $\text{Krank}(A \odot B) \geq \min\{\text{Krank}(A) + \text{Krank}(B) - 1, R\}$, where $A, B$ are matrix with $R$ columns. Intuitively, it means that Krank will increase with Khatri-Rao product. It will be used to prove Claim 2.26.

**Lemma 2.25** (Lemma A.4, Bhaskara et al. [2014b])**.** *$A, B$ are matrix with $R$ columns. Say $Krank_{\tau_1}(A) \geq k_A$, $Krank_{\tau_2}(B) \geq k_B$, where $k_A, k_B \in \mathbb{N}$. Let $t = \min\{k_A + k_B - 1, R\}$. Then $Krank_{(\tau_1 \tau_2 \sqrt{t})}(A \odot B) \geq t$.*

We will need the following claim, which shows that: $\sum_{j \in [k]} w_j v_j^{\otimes(2k+1)} = \sum_{j \in [k]} (v_j^{\otimes k}) \otimes (v_j^{\otimes k}) \otimes (w_j v_j)$ satisfies the condition of Theorem 2.22. This means we can recover the rank-one terms from $\sum_{j \in [k]} w_j v_j^{\otimes(2k+1)}$.

**Claim 2.26.** *Define $\Delta = \min_{j \neq j'} \min\{\|v_j - v_{j'}\|, \|v_j + v_{j'}\|\}$. Suppose $\Delta > 0$. Consider $\sum_{j \in [k]} w_j v_j^{\otimes(2k+1)} = \sum_{j \in [k]} (v_j^{\otimes k}) \otimes (v_j^{\otimes k}) \otimes (w_j v_j)$. Let $U \in \mathbb{R}^{n^k \times k}$ be the matrix whose $j$th column is flattened $v_j^{\otimes k}$. Let $W \in \mathbb{R}^{n \times k}$ be the matrix whose $j$th column is $w_j v_j$. Then the following hold:*

(1) *The condition numbers $\kappa(U) \leq (1/\Delta)^{O(k)} k^{O(k)}$,*

(2) *For all $i \neq j$, we have $w_i v_i, w_j v_j$ are not close to parallel: $Krank_{2/(w_{\min}\Delta)}(W) \geq 2$,*

(3) *For all $j$, we have $\|v_j^{\otimes k}\|_F, \|w_j v_j\| \leq 1$.*

**Proof.** *Proof of part (1):* The main idea is to apply Lemma 2.25. Let $A \in R^{n \times k}$ be the matrix whose $j$th column is $v_j$. Observe that $U = A^{\odot k}$. Roughly speaking, note that $\mathrm{Krank}(A) \geq 2$, we have $\mathrm{Krank}(A^{\odot(k-1)}) \geq k$. $A^{\odot(k-1)}$ has full column rank, so as $U$.

Let $A \in R^{n \times k}$ be the matrix whose $j$th column is $v_j$. Observe that $U = A^{\odot k}$. We know that $\mathrm{Krank}_{2/\Delta}(A) \geq 2$ by the definition $\Delta = \min_{j \neq j'} \min\{\|v_j - v_{j'}\|, \|v_j + v_{j'}\|\}$. Apply Lemma 2.25 inductively, we have

$$\mathrm{Krank}_{\left((2/\Delta)^k \sqrt{k! \cdot k}\right)} A^{\odot k} \geq k.$$

In other word,

$$\sigma_{\min}(A^{\odot k}) \geq (\Delta/2)^k / \sqrt{k! \cdot k} = \Delta^{O(k)} k^{-O(k)}.$$

Meanwhile

$$\|U\|^2 \leq \|U\|_F^2 = \sum_{j \in [k]} \|v_j^{\otimes k}\|_F^2 = k.$$

Therefore part (1) is true.

*Proof of part (2):* It follows by the definition $\Delta = \min_{j \neq j'} \min\{\|v_j - v_{j'}\|, \|v_j + v_{j'}\|\}$.

*Proof of part (3):* Recall that $\{v_j\}$ are unit vectors. □

The next claim says that we can get an accurate estimate of $v, w$ from an accurate estimate of $wv^{\otimes(2\ell+1)}$. This is useful since we get $k$ rank-one tensors of the form $wv^{\otimes(2\ell+1)}$ after apply Theorem 2.22 (tensor decomposition) to $\sum_{j \in [k]} w_j v_j^{\otimes(2\ell+1)}$. While it is easy and standard, we include it here for the sake of completeness.

**Claim 2.27.** *Let $\ell \in \mathbb{N}, \varepsilon, w \in (0,1]$. Let $v \in \mathbb{R}^n$ be a unit vector. $\{e_\alpha : \alpha \in [n]\}$ is the standard basis of $\mathbb{R}^n$. Suppose $T \in (\mathbb{R}^n)^{\otimes(2\ell+1)}$ satisfies that*

$$\|wv^{\otimes(2\ell+1)} - T\|_F \leq \frac{\varepsilon w}{4n^\ell}.$$

*Let $\hat{w} \in \mathbb{R}, \hat{v} \in \mathbb{R}^n$ be such that $\hat{w} = \|T\|_F, \hat{v}_\alpha = \frac{\langle T, e_\alpha \otimes I_n^{\otimes\ell}\rangle}{\hat{w}}, \forall \alpha \in [n]$. Then*

$$|w - \hat{w}| \leq \varepsilon \tag{2.3}$$

$$\|v - \hat{v}\| \leq \varepsilon$$

**Proof.** Define

$$\delta = \frac{\varepsilon w}{4n^\ell}.$$

By triangle inequality, $|\|wv^{\otimes(2\ell+1)}\|_F - \|T\|_F| \leq \delta \leq \varepsilon$. Note $\|wv^{\otimes(2\ell+1)}\|_F = w$, hence $|w - \hat{w}| \leq \delta \leq \varepsilon$, i.e., (2.3) is true.

Fix $\alpha \in [n]$. By Cauchy-Schwarz inequality,

$$|\langle wv^{\otimes(2\ell+1)} - T, e_\alpha \otimes I_n^{\otimes\ell}\rangle| \leq \delta \|e_\alpha \otimes I_n^{\otimes\ell}\|_F = \delta n^{\ell/2}.$$

As a consequence,

$$|wv_\alpha - \langle T, e_\alpha \otimes I_n^{\otimes\ell}\rangle| \leq \delta n^{\ell/2}.$$

We know that $|\hat{w}v_\alpha - wv_\alpha| \leq |\hat{w} - w| \leq \delta$. Then,

$$|\hat{w}v_\alpha - \langle T, e_\alpha \otimes I_n^{\otimes\ell}\rangle| \leq \delta(n^{\ell/2} + 1) \leq 2\delta n^{\ell/2}.$$

Hence,

$$|v_\alpha - \frac{\langle T, e_\alpha \otimes I_n^{\otimes \ell}\rangle}{\hat{w}}| \leq \frac{2\delta n^{\ell/2}}{\hat{w}} \leq \frac{4\delta n^{\ell/2}}{w}.$$

The last inequality is due to $|w - \hat{w}| \leq \delta \leq w/2$.

Then we have

$$\|v - \hat{v}\| \leq \frac{4\delta\sqrt{n}n^{\ell/2}}{w} \leq \varepsilon$$

□

Next, we will prove Theorem 2.28. The main steps of the algorithm are:

(1) Get a estimation of $T^* = \sum_{j\in[k]} w_j v_j^{\otimes(2\ell+1)}$ via Theorem 2.4.

(2) Use Theorem 2.22 (tensor decomposition) to recover all the rank-one terms.

(3) Use Claim 2.27 to recover the parameters from the rank-one terms.

**Theorem 2.28.** *Let $\ell \in \mathbb{N}$. Let $U \in \mathbb{R}^{n^\ell \times k}$ be the matrix whose $j$th column is flattened $v_j^{\otimes\ell}$. Suppose $\sigma_{\min}(U) \geq 1/\tau$, where $\tau > 0$. Given parameters $\varepsilon, \delta > 0$, $k \in \mathbb{N}$ and $w_{\min} > 0$ satisfying $w_{\min} \leq \min\{w_1, \ldots, w_k\}$, there is an algorithm* Estimate-Parameter *that given samples from the model has the following guarantees:*

(1) *The algorithm runs in sample complexity and time complexity*

$$\log^2(1/\delta)\varepsilon^{-2}\mathrm{poly}(n^\ell, \ell^\ell, \tau, 1/w_{\min}).$$

(2) *With probability* $1-\delta$*, the algorithm returns estimates* $\{\hat{w}_j, \hat{v}_j : j \in [k]\}$ *such that*

$$\min_{\pi \in \mathrm{Perm}([k])} \left(\max\{\|\hat{v}_j - v_{\pi(j)}\| : j \in [k]\} + \max\{\left|\hat{w}_j - w_{\pi(j)}\right| : j \in [k]\}\right) \leq \varepsilon.$$

*where the* min *is over permutation* $\pi$ *on* $[k]$.

**Algorithm 2:** ESTIMATE-PARAMETER

**Input:**
$k$ – the number of component
$\ell$ – parameter for order of the tensor
$\tau$ – parameter for lower bound on least singular value of $U$
$\mathcal{O}(v_1, \cdots, v_k, w_1, \cdots, w_k)$ – the sample oracle
$\varepsilon$ – error parameter
$w_{\min}$ – weight lower bound
$\delta$ – failure probability
**Output:**
$\{\hat{w}_j, \hat{v}_j : j \in [k]\}$ – estimate of $\{w_j, v_j : j \in [k]\}$

1 Apply Theorem 2.4, get $\mathbf{T}$ that is $\varepsilon \mathrm{poly}(n^{-\ell}, 1/\tau, w_{\min})$-close to $\sum_{j=1}^{k} w_j v_j^{\otimes(2\ell+1)}$ with probability at least 0.99;
2 View $\sum_{j\in[k]} w_j v_j^{\otimes(2\ell+1)}$ as $\sum_{j\in[k]} (v_j^{\otimes\ell}) \otimes (v_j^{\otimes\ell}) \otimes (w_j v_j)$. By Theorem 2.22, with probability at least 0.99, we can estimate each rank one term in the decomposition of $T^*$ (up to renaming), within additive error $\frac{\varepsilon w_{\min}}{8n^{\ell}}$;
3 By Claim 2.27, we can recover the parameters from the rank-one terms. This leads to estimates $\{\hat{w}_j, \hat{v}_j : j \in [k]\}$.;
4 **return** $\{\hat{w}_j, \hat{v}_j : j \in [k]\}$;

**Proof.** The algorithm ESTIMATE-PARAMETER is described in Algorithm 2.

Without loss of generality, we can assume $\delta = 0.1$. This is because we can always boost the success probability of our algorithm at a multiplicative cost of $O(\log^2(1/\delta))$ via Claim $A.1$.

Let $T^* = \sum_{j\in[k]} w_j v_j^{\otimes(2\ell+1)}$. By Theorem 2.4, there is an algorithm such that:

(1) It runs with sample complexity and time complexity

$$\varepsilon^{-2}\mathrm{poly}(n^{\ell}, \ell^{\ell}, \tau, 1/w_{\min})$$

(2) With probability 0.99,

we can estimate $T^*$ within an additive error of $\varepsilon\mathrm{poly}(n^{-\ell}, 1/\tau, w_{\min})$.

View $\sum_{j\in[k]} w_j v_j^{\otimes(2\ell+1)}$ as $\sum_{j\in[k]}(v_j^{\otimes\ell}) \otimes (v_j^{\otimes\ell}) \otimes (w_j v_j)$. We note that the following hold:

(1) The condition numbers $\kappa(U) \leq k\tau$,

(2) For all $i \neq j$, we have $w_i v_i, w_j v_j$ are not close to parallel: $\mathrm{Krank}_{\tau/w_{\min}}(W) \geq 2$,

(3) For all $j$, we have $\|v_j^{\otimes k}\|_F, \|w_j v_j\| \leq 1$.

Hence, $T^*$ satisfies the condition of Theorem 2.22. By Theorem 2.22, with probability at least 0.99, we can estimate each rank one term in the decomposition of $T^*$ (up to renaming), within additive error $\frac{\varepsilon w_{\min}}{8n^{\ell}}$. By Claim 2.27, we can recover the parameters from the rank-one terms. This leads to estimates $\{\hat{w}_j, \hat{v}_j : j \in [k]\}$ such that

$$\min_{\pi} \left(\max\{\|\hat{v}_j - v_{\pi(j)}\| : j \in [k]\} + \max\{\left|\hat{w}_j - w_{\pi(j)}\right| : j \in [k]\}\right) \leq \varepsilon,$$

where the min is over permutation $\pi$ on $[k]$. □

We are now ready to finish the proof of Theorem 2.1, Theorem 2.2 and Theorem 2.3.

**Proof of Theorem** 2.1**.** Note that Theorem 2.1 is just a special case ($\ell = 1$) of Theorem 2.28. □

The proofs of Theorem 2.2 and Theorem 2.3 have the same structure as Theorem 2.28.

**Proof of Theorem** 2.2. Without loss of generality, we can assume $\delta = 0.1$. This is because we can always boost the success probability of our algorithm at a multiplicative cost of $O(\log^2(1/\delta))$ via Claim A.1. Define $\ell = 10(\log k)/\Delta^2$.

Let $T^* = \sum_{j\in[k]} w_j v_j^{\otimes(2\ell+1)}$. By Theorem 2.4, there is an algorithm such that

(1) It runs in sample complexity and time complexity

$$\begin{aligned}&\varepsilon^{-2}\mathrm{poly}(n^\ell, k, \ell^\ell, 1/w_{\min}, 1/\Delta)\\ &= \varepsilon^{-2}\mathrm{poly}(n^{(\log k)/\Delta^2}, ((\log k)/\Delta^2)^{(\log k)/\Delta^2}, 1/w_{\min})\end{aligned}$$

(2) With probability 0.99,

we can estimate $T^*$ within an additive error of $\varepsilon\mathrm{poly}(1/n^\ell, 1/k, 1/w_{\min}, 1/\Delta)$.

By Claim 2.23, $T^*$ satisfies the condition of Theorem 2.22. By Theorem 2.22, with probability at least 0.99, we can estimate each rank one term in the decomposition of $T^*$ (up to renaming), within additive error $\frac{\varepsilon w_{\min}}{8n^\ell}$. By Claim 2.27, we can recover the parameters from the rank-one terms. This leads to estimates $\{\hat{w}_j, \hat{v}_j : j \in [k]\}$ such that

$$\min_\pi \left(\max\{\|\hat{v}_j - v_{\pi(j)}\| : j \in [k]\} + \max\{\left|\hat{w}_j - w_{\pi(j)}\right| : j \in [k]\}\right) \le \varepsilon,$$

where the min is over permutation $\pi$ on $[k]$. □

**Proof of Theorem** 2.3. Without loss of generality, we can assume $\delta = 0.1$. This is because we can always boost the success probability of our algorithm at a multiplicative

cost of $O(\log^2(1/\delta))$ via Claim A.1. Let $T^* = \sum_{j\in[k]} w_j v_j^{\otimes(2k+1)}$. By Theorem 2.4, there is an algorithm such that:

(1) It runs in sample complexity and time complexity $\varepsilon^{-2}\mathrm{poly}(n^k, k^k, \Delta^{-k}, 1/w_{\min})$.

(2) With probability 0.99,

we can estimate $T^*$ within an additive error of $\varepsilon\mathrm{poly}(\Delta^k, k^{-k}, n^{-k}, w_{\min})$.

By Claim 2.26, $T^*$ satisfies the condition of Theorem 2.22. By Theorem 2.22, with probability at least 0.99, we can estimate each rank one term in the decomposition of $T^*$ (up to renaming), within additive error $\frac{\varepsilon w_{\min}}{8n^k}$. By Claim 2.27, we can recover the parameters from the rank-one terms. This leads to estimates $\{\hat{w}_j, \hat{v}_j : j \in [k]\}$ such that

$$\min_{\pi} \left(\max\{\|\hat{v}_j - v_{\pi(j)}\| : j \in [k]\} + \max\{\left|\hat{w}_j - w_{\pi(j)}\right| : j \in [k]\}\right) \le \varepsilon,$$

where the min is over permutation $\pi$ on $[k]$. □

CHAPTER 3

# Learning a Mixture of Two Subspaces over $\mathbb{F}_2$

## 3.1. Introduction

A common assumption in high-dimensional data analysis is to assume that the given data belongs to a collection of lower-dimensional subspaces. In a prominent line of work in machine learning, computer vision, and computational geometry [Vidal, 2003, Elhamifar and Vidal, 2013, Soltanolkotabi et al., 2014, Park et al., 2014], this intuition is formalized through the problem of learning a mixture of subspaces, or subspace clustering. Given a set of points in $n$ dimensions that belong to a union of $k$ ($k \geq 2$) subspaces, the goal is to find the individual subspaces that contain all the points. When the points belong to $\mathbb{R}^n$, a beautiful result by Vidal [2003] shows that for any mixture of $k$ subspaces, under some mild general-position assumption of the points in the subspaces, there exists an algorithm that runs in time $n^{O(k)}$ and recovers the $k$ individual subspaces. Recently, subspace clustering has also been studied with outlier noise, in the special case when the points in each cluster are drawn from a Gaussian supported on a subspace [Raghavendra and Yau, 2020, Bakshi and Kothari, 2020]. However, these guarantees are specific to the real domain. Therefore, a natural question is whether such algorithmic guarantees extend to other domains such as $\mathbb{F}_2$.

*Can we efficiently learn a mixture of subspaces over finite fields?*

The algorithmic problem of recovering subspaces from samples has a distinct flavor over

finite fields, presenting significant computational challenges even in simple settings. In the most basic scenario, we are provided with samples drawn from a mixture of $k = 2$ unknown subspaces $A_0, A_1 \subseteq \mathbb{F}_2^n$ of dimensions $d_0$ and $d_1$ (respectively), with unknown mixing weights $w_0$ and $w_1$ in the interval $[0, 1]$, which add up to 1. Each sample is independently drawn according to the following procedure: with probability $w_0$, the sample is drawn from the uniform distribution $\mathsf{U}_{A_0}$ over $A_0$, and with probability $w_1$, the sample is drawn from the uniform distribution $\mathsf{U}_{A_1}$ over $A_1$. Our objective is to learn the individual subspaces $A_0$ and $A_1$ from independent samples generated by this model. For a precise definition of this model, we refer the reader to Definition 3.4.

The problem of learning mixtures of subspaces over $\mathbb{F}_2$ is a natural generalization of the problem of learning mixtures of subcubes, which was previously studied in [Chen and Moitra, 2019]. Specifically, subcubes can be thought of as (affine) subspaces where the constraints are defined by standard basis vectors. Our work considers arbitrary subspaces of $\mathbb{F}_2^n$, while excluding affine subspaces. Our work can also be framed within the framework of *learning from positive examples* Denis et al. [2005], De et al. [2014], Canonne et al. [2020], Ernst et al. [2015]. This framework investigates the learnability of supervised concept classes (in this case subspaces), when the learning algorithm is provided only with positive samples. More interestingly, the simple setting of $k = 2$ already captures the notorious problem of learning parities with noise (LPN) as a special case. One can encode LPN as learning a mixture of two subspaces $A_0, A_1$ where the subspaces $A_1 \subset A_0 \subseteq \mathbb{F}_2^n$ and $\dim(A_1) = \dim(A_0) - 1$ (see Proposition 3.24 and Proposition 3.23). The best-known algorithm for LPN runs in time $\exp\left(O(n/\log n)\right)$ [Blum et al., 2003]. Moreover, LPN

is also used as an average-case hardness assumption in learning theory and cryptography [Pietrzak, 2012]. To avoid this computational barrier, we will assume that we are not in the degenerate setting when one subspace contains the other. We call the two subspaces $A_0$ and $A_1$ *incomparable* iff $A_0 \nsubseteq A_1$ and $A_1 \nsubseteq A_0$. This leads to the following natural question about the computational complexity of the problem:

**Question.** *Is LPN the only computational obstruction for learning a mixture of two subspaces? Can one design faster algorithms when the subspaces $A_0, A_1$ are incomparable?*

Our first result shows that one can indeed design a polynomial time algorithm when the two subspaces are incomparable.

**Theorem 3.1.** *Suppose $A_0, A_1$ are incomparable. There is an algorithm* Incomparable-Subspace-Recovery *with the following guarantee: given oracle access to $\mathcal{O}(A_0, A_1, w_0, w_1)$ (for unknown $A_0, A_1, w_0, w_1$), $w_{min} > 0$ (such that $w_{min} \leq \min\{w_0, w_1\}$) and confidence parameter $\delta > 0$,*

(1) Incomparable-Subspace-Recovery *runs in sample and time complexity* $\mathrm{poly}(n/w_{min}) \cdot \log(1/\delta)$.

(2) *With probability $1 - \delta$, the algorithm outputs the subspaces $A_0, A_1$, and estimates the weights $w_0, w_1$ up to any desired inverse polynomial accuracy.*

Hence the above result gives a significantly faster polynomial time algorithm if we are *not* in the degenerate *comparable* setting when one subspace contains the other. In contrast, when $A_1 \subset A_0$ and $dim(A_1) = dim(A_0) - 1$ (or vice versa), the best-known algorithm takes $\exp(O(n/\log n))$ time. We remark that the algorithm succeeds in uniquely

identifying and recovering the individual subspaces, as opposed to just finding a mixture of two subspaces that fits the data. In the parlance of statistics, our algorithm recovers the underlying model (sometimes referred to as *parameter estimation*) as opposed to just doing *density estimation*.

Next, observe that the (presumed) hardness of LPN only implies hardness of the subspace recovery problem when (i) $A_1 \subseteq A_0$ and (ii) $\mathsf{dim}(A_1) = \mathsf{dim}(A_0) - 1$. This naturally prompts the question whether subspace recovery remains hard if (say) $A_1 \subseteq A_0$ but $\mathsf{dim}(A_1) \ll \mathsf{dim}(A_0)$. In other words, we ask the following question:

**Question.** *Can we design fast algorithms for subspace recovery when* $\mathsf{dim}(A_0)$ *and* $\mathsf{dim}(A_1)$ *are substantially different? Note that we are not imposing any conditions on the comparability of the hidden subspaces* $A_0$ *and* $A_1$*.*

Our next result provides an affirmative answer to this question.

**Theorem 3.2.** *Let* $w_{min} \geq 1/100$*. Let* $d_0 \geq d_1$ *and suppose* $\alpha := d_1/d_0 < 1 - \frac{\log d_0}{\sqrt{d_0}}$*. There is an algorithm* Subspace-Recover-Large-Diff *with the following guarantee: given oracle access to* $\mathcal{O}(A_0, A_1, w_0, w_1)$ *(for unknown* $A_0, A_1, w_0, w_1$*),* $w_{min} > 0$ *(such that* $w_{min} \leq \min\{w_0, w_1\}$*) and confidence parameter* $\delta > 0$*,*

(1) Subspace-Recover-Large-Diff *runs in sample and time complexity* $\log(1/\delta)\mathrm{poly}(n) \cdot d_0^{O(1)/(1-\alpha)}$.

(2) *With probability* $1 - \delta$*, the algorithm outputs the subspaces* $A_0, A_1$*, and estimates the mixing weights up to any desired inverse polynomial accuracy.*

Informally speaking, if the ratio of dimensions $\alpha$ is bounded away from 1, the running time is polynomial. In general, the running time of the algorithm has a dependence of $O(1/(1-\alpha))$ in the exponent.

### 3.1.1. Overview of Techniques.

We now briefly describe the algorithmic ideas and techniques used to prove our results. The algorithms that establish Theorem 3.1 and Theorem 3.2 use very different ideas. We begin with an overview of Theorem 3.1.

**Incomparable Setting (Theorem 3.1).**

The main component of the polynomial time algorithm in the incomparable setting is a careful procedure for *dimension reduction* that reduces the subspace clustering problem to $O(1)$ dimensions. We will construct a matrix $M \in \mathbb{F}_2^{r \times n}$ where $r = O(1)$ (in the actual proof, we set $r = 10$), and solve the clustering problem given samples of the form $y = Mx$ where $x$ is drawn from the original mixture. Note that a subspace under any linear map $M$ also gives a subspace; hence the samples in $\mathbb{R}^r$ are drawn from a mixture of subspaces $MA_0$ and $MA_1$. Any algorithm for learning a mixture of subspaces in $r = O(1)$ dimensions will allow us to cluster the points, and recover the individual subspaces $A_0, A_1$.

How do we choose the linear map $M$? A key property that we require of $M$ is that if $A_0$ and $A_1$ are incomparable, then $MA_0$ and $MA_1$ should also remain incomparable. While it is not hard to see that such a $M$ exists (even when $r = O(1)$), it is far from clear how to find it given that we do not have $A_0$ and $A_1$ explicitly. A natural choice for $M$ is a random matrix, where every entry is chosen independently from $\mathbb{F}_2$. Random linear maps are often used for dimension reduction in the real domain to approximately preserve inner

products and pairwise distances. However, a random map does not work in our setting, particularly when the target dimension $r \ll d_1$. This is because with high probability the subspaces collapse and $\mathbf{M}A_0 = \mathbf{M}A_1 = \mathbb{F}_2^r$. Therefore, this naive approach of dimension reduction is useless to recover the individual subspaces $A_0, A_1$.

Our approach instead proceeds in multiple rounds, where in each round, we reduce the dimension by one while preserving the property that the projected subspaces remain incomparable. More precisely, one can show that for a random linear map $\mathbf{M}_{n-1} \in \mathbb{F}_2^{(n-1)\times n}$, with *constant probability*, $\mathbf{M}_{n-1}A_0$ and $\mathbf{M}_{n-1}A_1$ are incomparable if $A_0, A_1$ are originally incomparable. However, this does not suffice per se, since we want to apply this for $\Omega(n)$ rounds (and thus, the probability of success becomes exponentially small). The crucial component of our algorithm is a testing procedure that runs in polynomial time, which given samples from a mixture of subspaces $U, V$, w.h.p. outputs whether $U$ and $V$ are *comparable* or *incomparable*. With such a procedure, in every phase we can reduce the dimension by 1, by sampling several random linear maps, running our testing procedure on each of them, and picking one that preserves incomparability of the subspaces. The guarantee of the testing procedure is given below.

**Theorem 3.3.** *There is an algorithm*
Test-Comparability *with the following guarantee: Given oracle access to* $\mathcal{O}(U, V, w_U, w_V)$ *(for unknown* $U, V, w_U, w_V$*),* $w_{min} > 0$ *(such that* $\min\{w_U, w_V\} \geq w_{min}$*) and confidence parameter* $\delta > 0$*,*

(1) Test-comparability *runs in sample and time complexity* $1/{w_{min}}^2 \cdot poly(n) \log(1/\delta)$.

(2) *With probability* $1-\delta$*, the algorithm outputs* True *if* $U$ *and* $V$ *are comparable and* False *otherwise.*

The testing procedure uses the following main insight. Suppose for simplicity the span $\mathsf{span}(U \cup V) = \mathbb{F}_2^n$. We prove that the subspaces $U$ and $V$ are incomparable if and only if there exists a non-zero polynomial $p$ of degree 2 that vanishes on $\mathcal{A} = U \cup V$. In fact, it will suffice to choose $\mathcal{A}$ to be a randomly chosen set of polynomial size sampled from the mixture of subspaces $U$ and $V$. The set of feasible degree-2 polynomials can then be obtained by setting up a system of linear equations where the unknowns correspond to coefficients of $p$.

Let us define $\mathbf{M} \in \mathbb{F}_2^{O(1)\times n}$ as $\mathbf{M} = \mathbf{M}_r \cdot \mathbf{M}_{r+1} \cdot \ldots \cdot \mathbf{M}_{n-1}$ – in other words, $\mathbf{M}$ is the linear map obtained by composing the dimension reduction maps over the $n-r$ rounds. Once the dimension is reduced to $r = O(1)$, we use a brute-force algorithm to recover $\mathbf{M}A_0, \mathbf{M}A_1$. Finally, once we know $\mathbf{M}A_0, \mathbf{M}A_1$, we can draw uniform samples from $A_0 \backslash \{x \in A_0 : \mathbf{M}x \in \mathbf{M}A_1\}$ to recover $A_0$; we can recover $A_1$ similarly (see Lemma 3.18).

**Significant dimension difference (Theorem 3.2).**
When the dimension of the subspaces are substantially different, we use algebraic ideas inspired by techniques in the real domain to recover the subspaces. The main algorithmic idea is by adapting ideas from related problems of subspace recovery over the reals [Hardt and Moitra, 2013, Bhaskara et al., 2019]. To explain the idea, consider the setting with equal mixing weights of $1/2$, $d_0 \approx n$, and suppose $\alpha = 1 - \Omega(1)$. If we consider a random subsample of $d_0$ points from the data set, we expect to have roughly $d_0/2$ points from subspace $A_0$ and $d_0/2$ points from subspace $A_1$. Suppose $\alpha < 1/2$ (referred to as the "large gap case")i.e., $d_1 < d_0/2$, then with high probability there is a linear dependence

in this sub-sample. Further, this linear dependence is (entirely) among points lying in the subspace $A_1$. This can be used to recover the subspace $A_1$ (and consequently, the subspace $A_0$ as well).

To see why this idea does not work in general, consider the case when the weights $w_0 = 0.9, w_1 = 0.1$ and $d_1 = 0.8d_0$. Then, to see a linear dependence among the points in $A_1$, we need to sample at least $d_1$ points from $A_1$. However, on average, this will mean sampling around $(w_0/w_1) \cdot d_1 = 9d_1$ many points from $A_0$. As $9d_1$ is much larger than the ambient dimension and thus, we will find many *spurious* linear dependencies – i.e., dependencies that do not come from points belonging to $A_1$. Thus, this strategy will fail to identify $A_1$.

Instead, when $\alpha \geq 1/2$, we will adopt a *dimension gap amplification strategy.* In particular, we consider a non-linear map $\phi : \mathbb{F}_2^{d_0} \to \mathbb{F}_2^{d_0'}$ where $d_0' = \sum_{j=0}^{\ell} \binom{d_0}{j}$ for an appropriately chosen $\ell$. Further, for a set $B$, let us define $\phi(B)$ as the set $\{\phi(x) : x \in B\}$. Roughly speaking, we want to choose an appropriate $\ell$ such that $\mathsf{dim}(\mathsf{span}(\phi(A_1)))/\mathsf{dim}(\mathsf{span}(\phi(A_0))) < 1/2$. For such an $\ell$, we can now apply the strategy for the large gap case to recover $A_1$ and $A_0$. We note that the idea of such a dimension gap amplification was also applied in the related subspace recovery problem over reals (see Chapter 4) – there, the goal was to recover one subspace $S$ of dimension $d \leq n$ containing $o(d/n)$ fraction of the points, while the rest of the points are drawn in general position from the whole of $\mathbb{R}^n$. While in spirit our idea is similar, it is challenging to get a handle on the dimensions of $\mathsf{span}(\phi(A_1))$ and $\mathsf{span}(\phi(A_0))$. Fortunately for us, some powerful results from additive combinatorics [Keevash and Sudakov, 2005, Ben-Eliezer et al., 2012] let us get precise estimates for $\mathsf{dim}(\mathsf{span}(\phi(A_0)))$ and $\mathsf{dim}(\mathsf{span}(\phi(A_1)))$. Roughly speaking, we show that

for $\ell \approx 1/(1-\alpha)$, $\mathsf{dim}(\mathsf{span}(\phi(A_1)))/\mathsf{dim}(\mathsf{span}(\phi(A_0))) < 1/2$, thus reducing to the large gap case.

## 3.2. Preliminaries

We start by defining the subspace recovery problem formally.

**Definition 3.4.** *The Subspace-Recovery problem is instantiated by two subspaces of $\mathbb{F}_2^n$ - $A_0$ and $A_1$ of dimensions $d_0$ and $d_1$ respectively. In addition, we also have weights $w_0$ and $w_1$ such that $w_0 + w_1 = 1$.*

*The subspaces $A_0$, $A_1$, dimensions $d_0$, $d_1$ as well as the weights $w_0$ and $w_1$ are unknown. For this instance, we define the sampling oracle $\mathcal{O}(A_0, A_1, w_0, w_1)$ is defined as follows: sample $\mathbf{b} \in \{0,1\}$ where $\mathbb{P}[\mathbf{b}=0] = w_0$ and $\mathbb{P}[\mathbf{b}=1] = w_1$. If $\mathbf{b}=0$, $\mathcal{O}(A_0, A_1, w_0, w_1)$ outputs a uniformly random element from $A_0$ and if $\mathbf{b}=1$, $\mathcal{O}(A_0, A_1, w_0, w_1)$ outputs a uniformly random element from $A_1$.*

*In the Subspace-Recovery problem, the algorithm is given access to the sampling oracle $\mathcal{O}(A_0, A_1, w_0, w_1)$, an error parameter $\varepsilon > 0$ and a weight parameter $w_{min} > 0$ with the promise that $w_{min} \leq \min\{w_0, w_1\}$. The goal of the algorithm is to output subspaces $A_0, A_1$ and estimates $\hat{w}_0$, $\hat{w}_1$ such that $|w_0 - \hat{w}_0| + |w_1 - \hat{w}_1| \leq \varepsilon$.*

*Without loss of generality, we will assume $d_0 \geq d_1$ from now on.*

**Remark 3.5.** *Note that once $A_0, A_1$ is found, estimating $w_0, w_1$ is not hard, this is because $\mathbb{P}_{\mathbf{x}\sim\mathcal{O}(A_0,A_1,w_0,w_1)}[\mathbf{x} \in A_0 \setminus A_1] = w_0 \frac{|A_0 \setminus A_1|}{|A_0|}$. Formally, there is an algorithm with the following guarantee: given oracle access to $\mathcal{O}(A_0, A_1, w_0, w_1)$ (for unknown $w_0, w_1$), $A_0, A_1$ and confidence parameter $\delta > 0$,*

(1) *this algorithm runs in sample and time complexity* $\mathrm{poly}(n) \cdot 1/\varepsilon^2 \cdot \log(1/\delta)$

(2) *With probability* $1-\delta$*, the algorithm outputs* $\hat{w}_0$*,* $\hat{w}_1$ *such that* $|w_0 - \hat{w}_0| + |w_1 - \hat{w}_1| \leq \varepsilon$.

*By this observation, we can focus on finding* $A_0, A_1$ *from now on.*

We next define the concept of incomparable subspaces.

**Definition 3.6.** *We define two subspaces* $A, B$ *to be incomparable if and only if* $A \nsubseteq B$ *and* $B \nsubseteq A$.

**3.2.0.1. Some useful notation.**

(1) For any $f : \mathbb{F}_2^n \to \mathbb{F}_2$, we use $\mathsf{zero}(f)$ to denote the set $\{x : f(x) = 0\}$.

(2) For integers $n, d \in \mathbb{N}$, we use $\mathsf{RM}(n, d)$ to denote the set of multilinear polynomials of degree at most $d$ over $\mathbb{F}_2^n$.

(3) For integers $n, k \in \mathbb{N}$ with $n \geq k$, we use $\binom{n}{\leq k}$ to denote $\sum_{i=0}^{k} \binom{n}{i}$.

(4) For a sample oracle $\mathcal{O}$ which return samples in $\mathbb{F}_2^n$, matrix $D \in \mathbb{F}_2^{k \times n}$, we use $D\mathcal{O}$ to denote a new sample oracle which each time returns $D\mathbf{x}$ where $\mathbf{x}$ is sampled from $\mathcal{O}$.

(5) For an index set $S$, we use $x_S$ to denote the set $\{x_i : i \in S\}$.

(6) For a set $S$ of vectors, we use $\mathsf{rank}(S)$ to denote $\dim(\mathsf{span}(S))$.

**3.2.0.2. Some useful facts regarding polynomials.** We next list some useful facts regarding polynomials over the field $\mathbb{F}_2$. While most of these are easy and standard, we list them here for the sake of completeness.

**Claim 3.7.** *Let $p$ be a multilinear polynomial over $\mathbb{F}_2^n$. If the polynomial $p$ is not identically zero (as a formal expression) and its degree is at most $c$, then*

$$\mathbb{P}_{\mathbf{x}\sim\mathbb{F}_2^n}[p(\mathbf{x}) \neq 0] \geq 1/2^c.$$

**Proof.** The proof is by induction on degree. If $c = 0$, then $p$ is identically 1 and thus the claim follows trivially.

Now, as an inductive hypothesis, assume that the claim is true for all polynomials of degree at most $c-1$. Let $p$ be a polynomial of degree $c$. Since $p$ is not identically zero, there exists $i$ such that $p$ can be expressed as

$$p(x_1, \cdots, x_n) = q(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) \cdot x_i + r(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n), \tag{3.1}$$

where degree of $q$ is at most $c-1$ and $q$ is not identically zero. The above formulation uses the fact that polynomials over $\mathbb{F}_2$ are multilinear. Observe that any choice of $\mathbf{x}_{-i} = (\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_n)$ such that $q(\mathbf{x}_{-i}) \neq 0$,

$$\mathbb{P}_{\mathbf{x}_i\sim\mathbb{F}_2}[p(\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_n) \neq 0] \geq \frac{1}{2}. \tag{3.2}$$

Now, applying the induction hypothesis on the polynomial $q(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$, we have that

$$\mathbb{P}_{\mathbf{x}\sim\mathbb{F}_2^n}[q(\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_n) \neq 0] \geq \frac{1}{2^{c-1}}.$$

Combining this with (3.1) and (3.2), we get the claim.

□

**Claim 3.8.** *There is an efficient algorithm* Size-system-polynomial *which given a set of points as input $z_1, \ldots, z_R \in \mathbb{F}_2^n$, determines the size of the set $T = |\{p \in \mathsf{RM}(n,2) : p(z_1) = p(z_2) = \cdots = p(z_r) = 0\}|$.*

**Proof.** Observe that $p$ can be expressed as linear system of equations (i) where the unknowns are the coefficients of $p$ and (ii) the equations are given by the constraints $\{p(z_i) = 0\}_{1 \leq i \leq R}$. Using Gaussian elimination, we can determine the rank $r$ of this system. Observe that the size of $T$ is just $2^r$, thus proving the claim. □

**3.2.0.3. Some useful facts regarding subspaces of $\mathbb{F}_2^n$.** We now list some useful facts about subspaces of $\mathbb{F}_2^n$.

**Claim 3.9.** *Let $k, d, n \in \mathbb{N}$ such that $k \geq 100d$. Let $V \subseteq \mathbb{F}_2^n$ be a subspace of dimension $d$. Let $\mathbf{x}_1, \cdots, \mathbf{x}_k$ be $k$ vectors sampled uniformly at random from $V$. Then,*

$$\mathbb{P}_{\mathbf{x}_1, \cdots, \mathbf{x}_k}[\forall S \subseteq [k] \text{ such that } |S| \geq 0.9k, \text{ we have } \mathsf{span}(\mathbf{x}_S) = V] \geq 1 - 2^{0.4k}. \tag{3.3}$$

**Proof.** We know that there always exists a linear bijection between $V$ and $\mathbb{F}_2^d$. Without loss of generality, we assume $n = d, V = \mathbb{F}_2^d$. Without loss of generality, assume 0.9k is an integer. For a fixed $S$ with $|S| = 0.9k$

$$\begin{aligned}
&\mathbb{P}[\mathsf{span}(\mathbf{x}_S) = \mathbb{F}_2^d] \\
&= \prod_{j=0}^{d-1} \left(1 - 2^{-0.9k+j}\right) && \text{See [Ferreira et al., 2012, Equation (2)]} \\
&\geq 1 - \sum_{j=0}^{d-1} 2^{-0.9k+j} \geq 1 - 2^{-0.9k+d} \geq 1 - 2^{-0.89k}.
\end{aligned}$$

The number of choice of $S$ is at most $\binom{k}{0.1k} \leq (10e)^{0.1k} \leq 2^{0.48k}$. Then the proof is completed by a union bound. $\square$

The next claim says that a union of two proper subspaces of $\mathbb{F}_2^n$ must differ substantially from any subspace of $\mathbb{F}_2^n$.

**Claim 3.10.** *Let $S$ be a subspace of $\mathbb{F}_2^n$ and of dimension $d$. Let $U, V \subsetneq S$ be two proper subspaces. Then $|S \backslash (U \cup V)| \geq 2^{d-2}$.*

**Proof.** Notice that the size of subspace in $\mathbb{F}_2$ is always a power of 2. There are two cases:

Case 1: $\dim(U) = \dim(V) = d - 1$.

Observe that $\dim(U \cap V) \geq d - 2$ and hence $|U \cup V| = |U| + |V| - |U \cap V| \leq 3 \cdot 2^{d-2}$.

Case 2: At least one of $\dim(U)$ or $\dim(V) \leq d - 2$.

In this case, $|U \cup V| \leq |U| + |V| \leq 2^{d-1} + 2^{d-2} \leq 3 \cdot 2^{d-2}$. Thus, in either case, $|U \cup V| \leq 3 \cdot 2^{d-2}$ which implies that $|S \backslash (U \cup V)| \geq 2^{d-2}$. $\square$

**Claim 3.11.** *Let $b_1, \cdots, b_t \in \mathbb{F}_2^n$ be linearly independent. Sample $\mathbf{M} \in \mathbb{F}_2^{m \times n}$ uniformly at random. Then $\mathbf{M}b_1, \cdots, \mathbf{M}b_t$ are independent and identically distributed. In other words, the joint distribution of $\mathbf{M}b_1, \cdots, \mathbf{M}b_t$ is the uniform distribution over $\mathbb{F}_2^{m \times t}$.*

**Proof.** Let us first add vectors $b_{t+1}, \ldots, b_n$ such that $\{b_1, \ldots, b_n\}$ is a basis of $\mathbb{F}_2^n$. Let $B$ be the matrix whose $i^{th}$ column is $b_i$. Now, observe that the map $\Psi : \mathbb{F}_2^{m \times n} \to \mathbb{F}_2^{m \times n}$ defined as $\Psi : M \mapsto M \cdot B$ is a bijection. Thus, if the random variable $\mathbf{M}$ is uniform over $\mathbb{F}_2^{m \times n}$, then so is $\mathbf{M} \cdot B$. Consequently, the first $t$ columns of $\mathbf{M} \cdot B$, namely, $\mathbf{M}b_1, \ldots, \mathbf{M}b_t$ are independent and identically distributed.

□

The following theorem gives a hypothesis testing routine for mixtures of subspaces over $\mathbb{F}_2^n$. The proof of this theorem is deferred to Appendix B.

**Theorem 3.12.** *Let* $\mathbf{D}$ *be a distribution of a mixture of two incomparable subspaces* $A, B \subseteq \mathbb{F}_2^n$ *with mixing weights* $w_A, w_B \geq w_0$*. Let* $\{A_j, B_j\}_{j=1}^N$ *be a collection of* $N$ *sets of hypothesis with the property that there exists* $i$ *such that* $\{A_i, B_i\} = \{A, B\}$*. There is an algorithm* Choose-The-Right-Hypothesis *which is given a confidence parameter* $\delta$*,* $w_0$*,* $\{A_j, B_j\}_{j=1}^N$ *and a sampler for* $\mathbf{D}$*. Every subspace of* $\{A_j, B_j\}_{j=1}^N$ *will be represented by a basis of that subspace, and the algorithm will have access to the basis. This algorithm has the following behavior,*

(1) *It runs in* $\mathrm{poly}(N, 1/w_0)\log(1/\delta)$ *time.*

(2) *With the probability* $1 - \delta$ *outputs the index* $i$ *such that* $\{A_i, B_i\} = \{A, B\}$*.*

### 3.3. Testing Comparability of the Subspaces

In this section, the main goal is to prove Theorem 3.3 (restated below for the convenience of the reader). We recall that Theorem 3.3 gives an efficient algorithm which given samples from a mixture of two subspaces $U, V$, decides whether $U$ and $V$ are comparable. This result in turn is an important piece in our subspace recovery algorithm in the "incomparable" case.

**Theorem 3.13** (restatement of Theorem 3.3)**.** *There is an algorithm* Test-Comparability *with the following guarantee: Given oracle access to* $\mathcal{O}(U, V, w_U, w_V)$ *(for unknown* $U, V, w_U, w_V$*),* $w_{min} > 0$ *(such that* $\min\{w_U, w_V\} \geq w_{min}$*) and confidence parameter* $\delta > 0$*,*

(1) TEST-COMPARABILITY *runs in sample and time complexity* $1/{w_{min}}^2 \cdot poly(n)\log(1/\delta)$.

(2) *With probability* $1-\delta$*, the algorithm outputs* True *if* $U$ *and* $V$ *are comparable and* False *otherwise.*

The main idea of the algorithm is the following. First, we take a few samples from the mixture to get $\mathsf{span}(U \cup V)$. By dimension reduction, it suffices to deal with the case $\mathsf{span}(U \cup V) = \mathbb{F}_2^n$. The crucial property we use is the following: If $\mathsf{span}(U \cup V) = \mathbb{F}_2^n$, $U, V$ are incomparable if there exists non-zero $p \in \mathsf{RM}(n,2)$ such that $p$ vanishes on the entire set $U \cup V$. The proof of Theorem 3.3 is deferred to the end of the section – to start, we prove some auxiliary lemmas.

**Claim 3.14.** *Assume* $s \geq 8n/w_{min}$*. Let* $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_s$ *be sampled from a mixture of two subspaces* $U, V \subseteq \mathbb{F}_2^n$ *(potentially comparable) of dimension at most* $d$ *with mixing weights* $w_U, w_V \geq w_{min}$*. Then, with probability at least* $1 - \exp(-s{w_{min}}^2/32)$*,* $\mathsf{span}(\mathbf{x}_1, \cdots, \mathbf{x}_s) = \mathsf{span}(U \cup V)$.

**Proof.** For fixed $x_1, \cdots, x_i$ such that $\mathsf{span}(x_1, \cdots, x_i) \subsetneq \mathsf{span}(U \cup V)$, we will show

$$\mathbb{P}_{\mathbf{x}_{i+1}}[\mathbf{x}_{i+1} \notin \mathsf{span}(x_1, \cdots, x_i)] \geq w_{min}/2. \tag{3.4}$$

Define $W = \mathsf{span}(x_1, \cdots, x_i)$. By our assumption, either $U \nsubseteq W$ or $V \nsubseteq W$. Let us assume that it is the former (the other case is symmetric). Under this assumption, $U \cap W$ is a proper subset of $U$. Since both are linear subspaces and the size of any linear space

**Algorithm 3:** TEST-COMPARABILITY

**Input:**
$n$ – ambient dimension
$\mathcal{O}(U, V, w_U, w_V)$ – oracle for random samples from a mixture of subspaces.
$w_{min}$ – lower bound of two mixture weights.
**Output:** True (if comparable) or False (if incomparable)

Set $t = 16n/({w_{min}}^2)$;
Sample $\mathbf{x}_1, \cdots, \mathbf{x}_t$ from $\mathcal{O}(U, V, w_U, w_V)$;
Set $S = \mathsf{span}(\mathbf{x}_1, \cdots, \mathbf{x}_t), v = dim(S)$;
Find $y_1, \cdots, y_v$ such that they form a basis of $S = \mathsf{span}(\mathbf{x}_1, \cdots, \mathbf{x}_t)$.;
Find a matrix $D \in \mathbb{F}_2^{v \times n}$ such that $Dy_i = e_i$ for all $i$, where $e_i$ is the $i$th element of the standard basis of $\mathbb{F}_2^v$;
Set $\mathcal{O}' = D\mathcal{O}(U, V, w_U, w_V) = \mathcal{O}(DU, DV, w_U, w_V)$. This is explained in prelims:useful notation;
Set $r = 8n^2/w_{min}$;
Sample $\mathbf{z}_1, \cdots, \mathbf{z}_r$ from $\mathcal{O}' = \mathcal{O}(DU, DV, w_U, w_V)$;
Use algorithm SIZE-SYSTEM-POLYNOMIAL to compute $T = |\{p \in \mathsf{RM}(v, 2) : p(\mathbf{z}_1) = p(\mathbf{z}_2) = \cdots = p(\mathbf{z}_r) = 0\}|$;
// See Claim 3.8
. **if** $T = 1$ **then**
  **return** True;
**else**
  **return** False;

over $\mathbb{F}_2$ is always a power of 2, $|U \cap W| \leq 0.5|U|$. Hence

$$\mathbb{P}[\mathbf{x}_{i+1} \in U \backslash W] \geq w_U \frac{|U \backslash W|}{|U|} \geq w_{min} \cdot 0.5.$$

In other words, $rank(x_1, \cdots, \mathbf{x}_{i+1}) = rank(x_1, \cdots, x_i) + 1$ will hold with probability at least $w_{min}/2$, thus proving (3.4). Define $\mathbf{y}_i = \mathsf{rank}(\mathbf{x}_1, \cdots, \mathbf{x}_i) - \mathsf{rank}(\mathbf{x}_1, \cdots, \mathbf{x}_{i-1})$, then $\mathbf{y}_1, \cdots, \mathbf{y}_s$ satisfy the condition of Lemma $C.1$ with $\gamma = w_{min}/2, d = \mathsf{rank}(U \cup V), k = s$. Claim 3.14 now follows by applying Lemma $C.1$. □

The next (easy) claim says that suppose the distribution $\mathbf{Z}$ (over $\mathbb{F}_2^d$) is *not too concentrated on any single element*. Then, a randomly chosen set of size roughly quadratic in $d$ is a *hitting set* for quadratic polynomials over $\mathbb{F}_2^d$. In other words, any non-zero element of $\mathsf{RM}(d,2)$ is non-zero on at least one element of this set.

**Claim 3.15.** *Let $\mathbf{Z}$ be a distribution over $\mathbb{F}_2^d$ such that the probability weight of every element is at least $w^*/2^d$. Let $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t$ be independent sampled from $\mathbf{Z}$. Then, we have*

$$\mathbb{P}\Big[\forall q \in \mathsf{RM}(d,2) \setminus \{0\}, \exists j \in [t] \text{ s.t. } q(\mathbf{x}_j) \neq 0\Big] \geq 1 - \exp\left(-tw^*/4 + \binom{d}{\leq 2}\log 2\right).$$

**Proof.** Fix $q \in \mathsf{RM}(d,2)$ such that $q \neq 0$. By Claim 3.7,

$$\mathbb{P}_{\mathbf{x}\sim_u \mathbb{F}_2^d}[q(\mathbf{x}) = 1] \geq 1/4.$$

As a consequence,

$$\mathbb{P}_{\mathbf{x}\sim Z}[q(\mathbf{x}) = 0] \leq 1 - \frac{w^*}{4}.$$

Hence

$$\mathbb{P}[q(\mathbf{x}_1) = \cdots = q(\mathbf{x}_t) = 0] \leq (1 - w^*/4)^t \leq \exp(-tw^*/4).$$

Notice that $|\mathsf{RM}(d,2)| = 2^{\binom{d}{\leq 2}}$. Using the union bound, we get the claim. □

We are now ready to finish the proof of Theorem 3.3.

**Proof of Theorem 3.3.** Without loss of generality, we assume $\delta = 0.1$, since we can always boost the probability at a multiplicative cost of $\log(1/\delta)$. By Claim 3.14, we know that $S = \mathsf{span}(U \cup V)$ (defined in Step 3 of the algorithm) with probability 0.999. Henceforth, we assume that $S = \mathsf{span}(U \cup V)$ holds.

By definition, $D$ (defined in Step 5 of the algorithm) is a linear bijection between $S$ and $\mathbb{F}_2^v$. Hence $DU, DV$ are incomparable if and only if $U, V$ are incomparable. Now observe that, $\mathcal{O}' = \mathcal{O}(DU, DV, w_U, w_V)$ will give samples from mixture of two subspaces $DU, DV$ with mixing weights $w_U, w_V \geq w_{min}$. Notice that $\mathsf{span}(DU \cup DV) = \mathbb{F}_2^v$. We divide the rest of the analysis into two cases.

Case 1: $DU, DV$ are comparable.

We have $DU = \mathbb{F}_2^v$ or $DV = \mathbb{F}_2^v$. By Claim 3.15, with probability 0.999, there will only be one polynomial (the zero polynomial) in the set $\{p \in \mathsf{RM}(v, 2) : p(\mathbf{z}_1) = p(\mathbf{z}_2) = \cdots = p(\mathbf{z}_r) = 0\}$. In this case, $T = 1$. Thus, overall, with probability 0.998, the algorithm returns the correct answer in this case.

Case 2: $DU, DV$ are incomparable.

In this case, $dim(DU) \leq v - 1$ (and $dim(DV) \leq v - 1$). Thus, there exists non-zero vector $b_U$ (resp. $b_V$) such that $\langle b_U, DU\rangle = \{0\}$ (resp. $\langle b_V, DV\rangle = \{0\}$). Now, consider the non-zero polynomial $p(x) = \langle b_U, x\rangle\langle b_V, x\rangle$. By definition it satisfies $p(DU \cup DV) = \{0\}$. Thus, in this case, the set $\{p \in \mathsf{RM}(v, 2) : p(\mathbf{z}_1) = p(\mathbf{z}_2) = \cdots = p(\mathbf{z}_r) = 0\}$ has at least two elements. Thus, overall, with probability 0.999, the algorithm returns the correct answer in this case. $\square$

### 3.4. Learning Mixtures of Incomparable Subspaces

In this section, we give a polynomial time algorithm (Algorithm 4: INCOMPARABLE-SUBSPACE-RECOVERY) for recovering the subspaces $A_0, A_1$ when given access to samples from a mixture of two subspaces that are incomparable. We prove the following theorem.

**Theorem 3.16** (restatement of Theorem 3.1)**.** *Suppose $A_0, A_1$ are incomparable. There is an algorithm*

INCOMPARABLE-SUBSPACE-RECOVERY *with the following guarantee: given oracle access to $\mathcal{O}(A_0, A_1, w_0, w_1)$ (for unknown $A_0, A_1, w_0, w_1$), $w_{min} > 0$ (such that $w_{min} \leq \min\{w_0, w_1\}$) and confidence parameter $\delta > 0$,*

(1) INCOMPARABLE-SUBSPACE-RECOVERY *runs in sample and time complexity* $\text{poly}(n/w_{min}) \cdot \log(1/\delta)$.

(2) *With probability $1 - \delta$, the algorithm outputs the subspaces $A_0, A_1$, and estimates the weights $w_0, w_1$ up to any desired inverse polynomial accuracy.*

The main idea is a new procedure for *dimension reduction* that reduces the subspace clustering problem to $O(1)$ dimensions. We will construct a linear map $M \in \mathbb{F}_2^{10 \times n}$ such that after projecting using $M$, the subspaces obtained $MA_0 = \{Mx : x \in A_0\}$ and $MA_1 = \{Mx : x \in A_1\}$ are incomparable. The construction of $M$ involves multiple rounds. In each round, we use Algorithm TEST-COMPARABILITY (and Theorem 3.3) as a black box, and find a projection that brings down the dimension by one with high probability, while maintaining incomparability of the subspaces. Once we recover the subspaces $MA_0, MA_1$ in $O(1)$ dimensions (using a brute force algorithm: enumerate all possible pairs of subspace, then use Theorem 3.12), we can then recover the original

subspaces $A_0, A_1$ by considering samples in $A_0 \cup A_1$ which are not mapped to $MA_0 \cap MA_1$ by $M$. We defer the proof of Theorem 3.1 to the end of section.

**Algorithm 4:** INCOMPARABLE-SUBSPACE-RECOVERY

**Input:**
$n$ – ambient dimension.
$\mathcal{O}(A_0, A_1, w_0, w_1)$ – oracle for random samples from mixture of subspaces.
$w_{min}$ – lower bound of two mixture weights.
**Output:** two subspaces.

1 $M$=FIND-A-GOOD-PROJECTOR$(n, \mathcal{O}(A_0, A_1, w_0, w_1), w_{min})$;
2 Use brute force to solve INCOMPARABLE-SUBSPACE-RECOVERY$(10, M\mathcal{O}(A_0, A_1, w_0, w_1), w_{min})$, let $U, V$ be the output ;
3 Set $t = 100n/w_{min}$;
4 Sample $\mathbf{x}_1, \cdots, \mathbf{x}_t$ from $\mathcal{O}(A_0, A_1, w_0, w_1)$;
5 **return** $\mathsf{span}(\{\mathbf{x}_i : M\mathbf{x}_i \notin V\}), \mathsf{span}(\{\mathbf{x}_i : M\mathbf{x}_i \notin U\})$;

The following lemma is crucial in establishing Theorem 3.1. The lemma proves that with high probability, Algorithm FIND-A-GOOD-PROJECTOR (Algorithm 5) reduces the dimension to $r = 10$ while *preserving the incomparability* of the subspaces. If $M$ is randomly chosen from $\mathbb{F}_2^{10\times n}$, then $MA_1 \subseteq MA_0$ since $MA_0$ collapses to $\mathbb{F}_2^{10}$ with high probability. Algorithm FIND-A-GOOD-PROJECTOR instead proceeds in multiple rounds, and reduces the dimension one per round. If the projector $\mathbf{M}'$ is chosen uniformly at random from $\mathbb{F}_2^{(n-1)\times n}$, with constant probability $\mathbf{M}'A_0, \mathbf{M}'A_1 \in \mathbb{F}_2^{n-1}$ remain incomparable. We can now use Algorithm TEST-COMPARABILITY (and Theorem 3.3) to boost the success probability in each round by repeatedly sampling $M'$ and rejecting it if the resulting subspaces are comparable.

**Lemma 3.17.** *Given samples from a mixture of two incomparable subspaces $A_0, A_1 \subseteq \mathbb{F}_2^n$ with mixing weights $w_0, w_1 \geq w_{min}$. There exists $M \in \mathbb{F}_2^{10\times n}$ such that $MA_0, MA_1$ are*

*incomparable subspaces. Moreover, there is an algorithm* Find-A-Good-Projector *that runs in time* $1/w_{min} \cdot poly(n)$ *and find such a* $M$ *with probability at least* 0.999.

**Algorithm 5:** Find-A-Good-Projector

**Input:**
$n$ – ambient dimension
$\mathcal{O}(A_0, A_1, w_0, w_1)$ – oracle for random samples from mixture of subspaces.
$w_{min}$ – lower bound of two mixture weights.
**Output:** a matrix $M \in \mathbb{F}_2^{10\times n}$.

1 Set $M = I_n$, where $I_n \in \mathbb{F}_2^{n\times n}$ is the identity matrix;
2 **for** $i = n; i > 10; i = i - 1$ **do**
3  Sample $\mathbf{T} \in \mathbb{F}_2^{(i-1)\times i}$ uniformly at random;
4  **while** Test-Comparability$(i, \mathbf{T}M\mathcal{O}(A_0, A_1, w_0, w_1), w_{min}, 1/n^2)$ `// the last parameter is the failure probability we want.`
5  **do**
6  Sample $\mathbf{T} \in \mathbb{F}_2^{(i-1)\times i}$ uniformly at random;
7  $M = \mathbf{T}M$;
8 **return** $M$;

**Proof.** We now show that Algorithm Find-A-Good-Projector runs in polynomial time and finds a required projector $M$ with high probability. Observe that from Theorem 3.3, every call of Test-Comparability (in step 4 of Algorithm 5) fails with probability at most $\delta = O(1/n^2)$. We will prove that at any iteration $i \in \{n, n-1, \ldots, 11\}$, a randomly chosen matrix $\mathbf{T} \in \mathbb{F}_2^{(i-1)\times i}$ (in step 3) succeeds with constant probability in preserving the incomparability of the subspaces. This ensures that it will suffice to sample $O(\log n)$ many random $T$ per round before we succeed in that round (and hence $O(n \log n)$ overall).

Fix an iteration $i \in \{n, n-1, \ldots, 11\}$, and let $M \in \mathbb{F}_2^{i\times n}$ be the current projector. Let $U := MA_0, V := MA_1$, and assume $U, V$ are incomparable. We show the following claim.

**Claim:** *For a random* $\mathbf{T} \in \mathbb{F}_2^{(i-1)\times i}$ *chosen in step 3,*

$$\mathbb{P}_{\mathbf{T}}[\mathbf{T}U, \mathbf{T}V \text{ are incomparable}] \geq 9/128. \tag{3.5}$$

We now prove the claim by considering two cases depending on the rank of $U \cup V$ i.e., the dimension of the span of $U \cup V$.

**Case 1:** $rank(U \cup V) \leq i - 1$**.**

Let $v = rank(U \cup V)$ and $b_1, \cdots, b_v$ be a basis of $\mathsf{span}(U \cup V)$. By Claim 3.11, $\mathbf{T}b_1, \cdots, \mathbf{T}b_v$ can be viewed as being sampled independently from $\mathbb{F}_2^{i-1}$. A uniformly random matrix from $\mathbb{F}_2^{(i-1)\times(i-1)}$ is full-rank with probability at least $\prod_{j\geq 1}(1 - 2^{-j}) \geq 1/4$. Hence,

$$\mathbb{P}[\mathbf{T}b_1, \cdots, \mathbf{T}b_v \text{ are linearly independent}] \geq 1/4.$$

When $\mathbf{T}b_1, \cdots, \mathbf{T}b_v$ are linearly independent, $\mathbf{T}U, \mathbf{T}V$ are incomparable as required. This establishes (3.5) in Case 1.

**Case 2:** $rank(U \cup V) = i$**.**

Let $b_1, \ldots, b_{dim(U\cap V)}$ be a basis of $U \cap V$. We extend the basis such that $b_1, \ldots, b_{dim(U\cap V)}, c_1, \ldots, c_{dim(U)-dim(U\cap V)}$ is a basis of $U$, and similarly, we extend the basis so that $b_1, \ldots, b_{dim(U\cap V)}, d_1, \ldots, d_{dim(V)-dim(U\cap V)}$ is a basis of $V$. Observe that $b_1, \ldots, b_{dim(U\cap V)}, c_1, \ldots, c_{dim(U)-dim(U\cap V)}, d_1, \ldots, d_{dim(V)-dim(U\cap V)}$ is a basis of $\mathsf{span}(U \cup V)$. Reorder this basis to get $a_1, \ldots, a_i$ such that $a_{i-1} = c_1, a_i = d_1$. Let $\mathbf{t}_j$ denote $\mathbf{T}a_j$. By

Claim 3.11, $\mathbf{t}_1, \cdots, \mathbf{t}_i$ are independent and identically distributed. Let $\mathcal{E}$ be the event

$$\mathcal{E} = \begin{cases} \mathbf{t}_j \notin \mathsf{span}(\mathbf{t}_1, \cdots, \mathbf{t}_{j-1}) & \forall 1 \leq j \leq i-3 \\ \mathbf{t}_{i-2} \in \mathsf{span}(\mathbf{t}_1, \cdots, \mathbf{t}_{i-3}) \\ \mathbf{t}_{i-1} \notin \mathsf{span}(\mathbf{t}_1, \cdots, \mathbf{t}_{i-2}) \\ \mathbf{t}_i \notin \mathsf{span}(\mathbf{t}_1, \cdots, \mathbf{t}_{i-1}) \end{cases}$$

Then,

$$\mathbb{P}_{\mathbf{T}}[\mathcal{E}] = (\prod_{j=1}^{i-3}(1 - 2^{j-1}/2^{i-1})) \cdot 1/4 \cdot 3/4 \cdot 1/2 \geq 3/4 \cdot 3/32 = 9/128.$$

Condition on $\mathcal{E}$. We now show that $\mathbf{T}U, \mathbf{T}V$ are incomparable as required. We will show $\mathbf{T}U \nsubseteq \mathbf{T}V$, the other direction is similar. By definition $\mathbf{t}_{i-1} = \mathbf{T}a_{i-1} = Tc_1 \in TU$, and $\mathbf{t}_{i-1} \notin \mathsf{span}(\mathbf{t}_1, \mathbf{t}_2, \cdots, \mathbf{t}_{i-2}, \mathbf{t}_i)$. However $\mathbf{T}V \subseteq \mathsf{span}(\mathbf{t}_1, \mathbf{t}_2, \cdots, \mathbf{t}_{i-2}, \mathbf{t}_i)$, hence $\mathbf{t}_{i-1} \notin \mathbf{T}V$, $\mathbf{T}U \nsubseteq \mathbf{T}V$. This establishes (3.5). Hence the lemma follows. □

The following lemma shows that a few samples drawn uniformly from $S \setminus T$ suffice to recover $S$ with high probability. This will allow us to recover $A_0$ and $A_1$ after clustering the points in $MA_0 \cup MA_1$.

**Lemma 3.18.** *Let $S$ be a subspace of $\mathbb{F}_2^n$ and of dimension $d$. Let $T$ be a proper subspace of $S$. Let $t \geq 8n$ be a integer. $\mathbf{x}_1, \cdots, \mathbf{x}_t$ are independently uniformly sampled from $S \setminus T$. Then,*

$$\mathbb{P}[\mathsf{span}(\mathbf{x}_1, \cdots, \mathbf{x}_t) = S] \geq 1 - e^{-t/128}.$$

**Proof.** Let $V \subsetneq S$ be a fixed subspace. Then by Claim 3.10, $|S\backslash(T \cup V)| \geq 2^{d-2}$, which is at least $1/4$ of $|S|$. We have

$$\mathbb{P}_{\mathbf{x}\sim_u S\backslash T}[\mathbf{x} \notin V] \geq 1/4.$$

In other words, if $\mathsf{span}(\mathbf{x}_1, \cdots, \mathbf{x}_k) \neq S$, then $rank(\mathbf{x}_1, \cdots, \mathbf{x}_{k+1}) = rank(\mathbf{x}_1, \cdots \mathbf{x}_k) + 1$ will hold with probability at least $1/4$. Define the random variables $\mathbf{y}_i = \mathsf{rank}(\mathbf{x}_1, \cdots, \mathbf{x}_i) - \mathsf{rank}(\mathbf{x}_1, \cdots, \mathbf{x}_{i-1})$ for $i \in \{1, 2, \ldots, t\}$. Note that $\mathbf{y}_1, \cdots, \mathbf{y}_t$ are not quite independent (since the probability the rank increases at step $i$ depends on the random choices of $\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}$ in previous iterations). But they satisfy the condition of Lemma $C.1$ with $\gamma = 1/4, d = \dim(S), k = t$. The proof is completed after applying Lemma $C.1$. $\square$

We are now ready to complete the proof of Theorem 3.1.

**Proof of Theorem 3.1.** Without loss of generality, we assume $\delta = 0.1$, since we can always boost the probability at a multiplicative cost of $\log(1/\delta)$. By Lemma 3.17, $M$ satisfies the property that $MA_0, MA_1$ are incomparable with high probability (probability at least 0.999, say). Moreover assuming $MA_0, MA_1$ are incomparable, the brute force algorithm will return them with high probability.

Let $U = MA_0, V = MA_1$. We will show that $\mathsf{span}(\{\mathbf{x}_i : M\mathbf{x}_i \notin V\}) = A_0$ with probability 0.998. Observe that $W = \{x \in A_0 : Mx \in MA_1\}$ is a proper subspace of $A_0$. Hence if $\mathbf{x}$ is drawn uniformly from $A_0$, $\mathbf{x}$ will not be in $W$ with probability at least $1/2$. By Chernoff bound, we expect to see at least $20n$ samples in $\{\mathbf{x}_i : M\mathbf{x}_i \notin V\}$ with probability at least 0.999 and all these samples can be viewed as uniformly drawn from $A_0\backslash W$. By Lemma 3.18, $\mathsf{span}(\{\mathbf{x}_i : M\mathbf{x}_i \notin MA_1\}) = A_0$ with probability 0.998. A similar argument shows that the algorithm also recovers $A_1$ with high probability. Finally, after recovering

$A_0, A_1$ it is also easy to estimate the weights $w_0, w_1$ to inverse polynomial accuracy (see Remark 3.5). □

### 3.5. Mixtures of Two Subspaces with Significant Dimension Difference

In this section, we prove Theorem 3.2 (restated below for convenience of the reader) which shows that there is a computationally efficient algorithm for learning a mixture of two subspaces with significantly different dimensions. Note that the following theorem does *not* assume that the two subspaces are incomparable.

**Theorem 3.19** (restatement of Theorem 3.2)**.** *Let $w_{min} \geq 1/100$. Let $d_0 \geq d_1$ and suppose $\alpha := d_1/d_0 < 1 - \frac{\log d_0}{\sqrt{d_0}}$. There is an algorithm* Subspace-Recover-Large-Diff *with the following guarantee: given oracle access to $\mathcal{O}(A_0, A_1, w_0, w_1)$ (for unknown $A_0, A_1, w_0, w_1$), $w_{min} > 0$ (such that $w_{min} \leq \min\{w_0, w_1\}$) and confidence parameter $\delta > 0$,*

(1) Subspace-Recover-Large-Diff *runs in sample and time complexity* $\log(1/\delta)\text{poly}(n) \cdot d_0^{O(1)/(1-\alpha)}$.

(2) *With probability $1 - \delta$, the algorithm outputs the subspaces $A_0, A_1$, and estimates the mixing weights up to any desired inverse polynomial accuracy.*

The algorithm Subspace-Recover-Large-Diff is described in Figure 6. Before proving Theorem 3.2, we will make some simplifying assumptions (with their justifications given below) followed by some useful notation.

**Remark 3.20.** *Without loss of generality, we can assume*

(1) $n = d_0$. *This is because we can first use Theorem 3.3 to test whether the underlying subspaces are incomparable. If they are incomparable, we can use Theorem 3.1 to recover the subspaces. If not, we can take* $O(n/w_{min})$ *samples from the mixture to get* $\mathsf{span}(A_0 \cup A_1)$ *with high probability (see Claim* 3.14*). We can then construct a linear bijection, say* $D$*, between* $\mathsf{span}(A_0 \cup A_1)$ *and* $\mathbb{F}_2^{d_0}$*. Applying the map* $D$ *to every sample from the mixture, we can now assume that* $n = d_0$.

(2) *The algorithm knows* $d_0, d_1$. *This is because we can enumerate all the possible values of* $d_0, d_1$ *and run the algorithm* Subspace-Recover-Large-Diff *to get a list of candidate hypotheses. We can then use the hypothesis testing algorithm in Theorem 3.12 to identify the correct one with high probability.*

(3) *We set* $\delta = 0.1$. *This is because we can always boost the success probability of our algorithm at a multiplicative cost of* $\log(1/\delta)$.

(4) $d_0$ *is at least a sufficiently large constant (which only depends on* $w_{min}$*). Otherwise, we can always apply a brute force algorithm to recover the subspaces.*

**Notation.**

(1) We will use $\phi_\ell(x) \in \mathbb{F}_2^{\binom{n}{\leq \ell}}$ to represent the vector consisting of all the monomials of degree at most $\ell$ on $x$, including the constant term. As an example, when $\ell = 2$ and $n = 2$, we have $\phi_\ell(x) = (1, x_1, x_2, x_1x_2)$ – note that because the underlying field is $\mathbb{F}_2$, all the monomials are multilinear. We will use $\phi_\ell(A)$ to denote $\{\phi_\ell(x) : x \in A\}$. $\phi_\ell(A)$ is a set of vectors in $\mathbb{F}_2^{\binom{n}{\leq \ell}}$.

(2) We define $t := d_0 - d_1 = (1-\alpha)d_0$ to denote the difference between the dimensions of the underlying subspaces $A_0$ and $A_1$.

(3) For a sequence of vector $x_1, x_2, \cdots, x_k$, we define $x_{-i} := \{x_j : j \neq i\}$.

(4) Let us denote by $y_i := \phi_\ell(x_i)$.

Finally, we note that for any subspace $V$ of dimension $d$ over $\mathbb{F}_2$, $rank(\phi_\ell(V)) = \binom{d}{\leq \ell}$.

**Algorithm 6:** SUBSPACE-RECOVER-LARGE-DIFF

**Input:**
$d_0$ – dimension of the larger subspace
$\alpha \leq 1$ – ratio of the dimensions of two subspaces
$\mathcal{O}(A_0, A_1, w_0, w_1)$ – oracle for random samples from mixture of subspaces.
$w_{min}$ – minimum of two mixture weights.
**Output:** two subspaces $U, V$.

1 Set $\ell = \frac{2\log(100/w_{min})}{1-\alpha}$;
2 Use $\mathcal{O}(A_0, A_1, w_0, w_1)$ to sample $m = \binom{d_0}{\leq \ell}$ vectors $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_m$;
3 Let S be the set of all $i \in [m]$ such that $\mathbf{y}_i \coloneqq \phi_\ell(\mathbf{x}_i)$ can be expressed as linear combination of $\{\phi_\ell(\mathbf{x}_j) : j \neq i\}$;
4 **return** $U = \mathsf{span}(\{\mathbf{x}_i : i \in S\}), V = \mathsf{span}(\{\mathbf{x}_i : \mathbf{x}_i \notin U\})$;

We start with the following crucial lemma from Ben-Eliezer et al. [2012] (stated below). An equivalent version was also proven in [Keevash and Sudakov, 2005, Theorem 1.5].

**Lemma 3.21** (Lemma 4, Ben-Eliezer et al. [2012])**.** *Let $x_1, x_2, \cdots, x_R$ be $R = 2^r$ distinct points in $\mathbb{F}_2^n$. Consider the linear space of degree $d$ polynomials restricted to these points; that is, the space*

$$\{(p(x_1), \cdots, p(x_R)) : p \in \mathsf{RM}(n, d)\}.$$

*The linear dimension of this space is at least $\binom{r}{\leq d}$.*

As an easy corollary, we have the following claim.

**Lemma 3.22.** *Let $x_1, x_2, \cdots, x_R$ be distinct points in $\mathbb{F}_2^n$. If $R \geq 2^r$, then*

$$\mathsf{rank}(\{\phi_\ell(x_1), \cdots, \phi_\ell(x_R)\}) \geq \binom{r}{\leq \ell}.$$

**Proof.** Without loss of generality, we can assume $R = 2^r$, since having more points can only increase the rank. Let $t = |\mathsf{RM}(n, \ell)|$. Say $\mathsf{RM}(n, \ell) = \{p_1, \cdots, p_t\}$. Let $A \in \mathbb{F}_2^{t \times R}$ be defined as $A_{i,j} = p_i(x_j)$. Applying Lemma 3.21 with $d = \ell$, we know the row-rank of $A$ is at least $\binom{r}{\leq \ell}$. Let $B \in \mathbb{F}_2^{\binom{n}{\leq \ell} \times R}$ be the matrix whose $i$th column is $\phi_\ell(x_i)$. Since every polynomial is a linear combination of monomials, there exists $C \in \mathbb{F}_2^{t \times \binom{n}{\leq \ell}}$ such that $A = CB$, hence $\mathsf{rank}(B) \geq \mathsf{rank}(A) \geq \binom{r}{\leq \ell}$. □

**Proof of Theorem 3.2.** Let $I_0$ (resp. $I_1$) be the set of all $i$ such that $\mathbf{x}_i$ was sampled from $A_0$ (resp. $A_1$). We now define the events $\mathcal{E}_1$, $\mathcal{E}_2$, $\mathcal{E}_3$ and $\mathcal{E}_4$ as follows:

(1) $\mathcal{E}_1$: $\forall i \in I_0, \mathbf{y}_i \notin \mathsf{span}(\{\mathbf{y}_{-i}\} \cup \phi_\ell(A_1))$

(2) $\mathcal{E}_2$: $|I_1| \geq 10\binom{\alpha d_0}{\leq \ell}$

(3) $\mathcal{E}_3$: $\forall T \subseteq I_1$ such that $|T| \geq 0.9|I_1|$, we have $\mathsf{span}(\{\mathbf{x}_j\}_{j \in T}) = A_1$

(4) $\mathcal{E}_4$: $\mathsf{span}(\{\mathbf{x}_j\}_{j \in I_0}) = A_0$

Assume $\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3, \mathcal{E}_4$ holds. Note that whenever $\mathcal{E}_1$ holds, it follows that $S$ (defined in line 3 of Subspace-Recover-Large-Diff) is a subset of $I_1$. We now show that $A_1$ can be recovered from the span of the samples corresponding to $S$. Now, consider the set $\{\phi_\ell(\mathbf{x}_i) : i \in I_1 \setminus S\}$. By definition, the elements of this set are linearly independent (otherwise, they will belong in $S$). As $\dim(\mathsf{span}(\phi_\ell(A_1))) \leq \binom{\alpha d_0}{\leq \ell}$, it follows that $|\{\phi_\ell(\mathbf{x}_i) : i \in I_1 \setminus S\}| \leq \binom{\alpha d_0}{\leq \ell}$. As $i \mapsto \phi_\ell(\mathbf{x}_i)$ is a injection on $I_1 \setminus S$ , it follows that $|\{i \in I_1 \setminus S\}| \leq \binom{\alpha d_0}{\leq \ell}$. Since $\mathcal{E}_2$ holds, $|I_1 \setminus S| \leq 0.1|I_1|$, hence $|S| \geq 0.9|I_1|$. Since $\mathcal{E}_3$ holds, $\mathsf{span}\left(\{\mathbf{x}_j\}_{j \in S}\right) = A_1$.

We now argue that the algorithm also recovers $A_0$. We claim $\{j \in [m] : \mathbf{x}_j \notin A_1\} = I_0$. Fix $j \in I_0$. Since $\mathcal{E}_1$ holds, $\phi_\ell(\mathbf{x}_j) = \mathbf{y}_j \notin \phi_\ell(A_1)$, then $\mathbf{x}_j \notin A_1$. Hence $I_0 \subseteq \{j : \mathbf{x}_j \notin A_1\}$. It is not hard to see $\{j : \mathbf{x}_j \notin A_1\} \subseteq I_0$. Finally when $\mathcal{E}_4$ holds, we have $\mathsf{span}(\{\mathbf{x}_j : \mathbf{x}_j \notin A_1\}) = \mathsf{span}(\{\mathbf{x}_j : j \in I_0\}) = A_0$.

Thus, it remains to show that $\mathcal{E}_1$, $\mathcal{E}_2$, $\mathcal{E}_3$ and $\mathcal{E}_4$ hold simultaneously with probability 0.99.

**Proof of $\mathbb{P}[\mathcal{E}_1] \geq 0.999$:** First, observe that by definition, $\ell = \frac{2\log(100/w_{min})}{1-\alpha}$. Using the assumption on $d_0$ and $w_{min}$, it follows that

$$\ell = \frac{2\log(100/w_{min})}{1-\alpha} = O\left(\frac{\sqrt{d_0}}{\log d_0}\right); \quad d_0 \geq \frac{2\ell}{(1-\alpha)}. \tag{3.6}$$

From this, applying the constraints on $d_0$ and $\ell$ from (3.6), we get

$$\left(\frac{w_{min}}{100}\right)^{1/\ell} \geq 1 + \frac{1}{\ell} \cdot \log\left(\frac{w_{min}}{100}\right) \geq \frac{(1+\alpha)}{2} \geq \alpha + \frac{\ell}{d_0}. \tag{3.7}$$

Now, it is not difficult to see that $\binom{\alpha d_0}{\leq \ell} \leq \binom{\alpha d_0 + \ell}{\ell}$ – it easily follows from the combinatorial interpretation of binomial coefficients. Now, using this and (3.7), we get

$$\frac{\binom{\alpha d_0}{\leq \ell}}{\binom{d_0}{\leq \ell}} \leq \frac{\binom{\alpha d_0 + \ell}{\ell}}{\binom{d_0}{\ell}} \leq \left(\alpha + \frac{\ell}{d_0}\right)^{\ell} \leq \frac{w_{min}}{100}. \tag{3.8}$$

We now have,

(3.9)

$$\begin{aligned}
&\mathbb{P}\Big[\dim(\mathsf{span}(\{\mathbf{y}_{-i}\}\cup\phi_\ell(A_1)))\le(1-0.4w_{min})\binom{d_0}{\le\ell}\Big]\\
\ge&\mathbb{P}\Big[\dim(\mathsf{span}(\{\mathbf{y}_{-i}\}\cup\phi_\ell(A_1)))\le(1-0.5w_{min})\binom{d_0}{\le\ell}+\binom{\alpha d_0}{\le\ell}\Big]\\
&\qquad\qquad\text{using (3.8),}\\
\ge&\mathbb{P}\Big[\dim(\mathsf{span}(\{\mathbf{y}_{-i}\}))\le(1-0.5w_{min})\binom{d_0}{\le\ell}\Big]\\
&\qquad\qquad\text{using }\dim(\mathsf{span}(\phi_\ell(A_1)))=\binom{\alpha d_0}{\le\ell},\\
\ge&\mathbb{P}[|I_0|\le(1-0.5w_{min})\binom{d_0}{\le\ell}]\\
&\qquad\qquad\text{using }|I_0|\ge|\{\mathbf{y}_{-i}\}|\ge\dim(\mathsf{span}(\{\mathbf{y}_{-i}\})),
\end{aligned}$$

(3.10)

$$\begin{aligned}
\ge&1-e^{-\frac{w_{min}{}^2}{24}\binom{d_0}{\le\ell}}\\
&\qquad\qquad\text{from a standard Chernoff bound.}
\end{aligned}$$

Let us now define the event $\mathcal{B}_i$ as the event that $i\in I_0$ and $\dim(\mathsf{span}(\{\mathbf{y}_{-i}\}\cup\phi_\ell(A_1)))\le(1-0.4w_{min})\binom{d_0}{\le\ell}$. Let $r\coloneqq\lceil(1-0.4w_{min}/\ell)d_0+\ell\rceil$. Using reasoning similar to (3.8), we have

$$\frac{\binom{r}{\le\ell}}{\binom{d_0}{\le\ell}}\ge\frac{\binom{r}{\ell}}{\binom{d_0+\ell}{\ell}}\ge\left(\frac{r-\ell}{d_0}\right)^\ell\ge\left(1-\frac{0.4w_{min}}{\ell}\right)^\ell\ge1-0.4w_{min}.$$

Thus, it follows that if the event $\mathcal{B}_i$ holds, $\dim(\mathsf{span}(\{\mathbf{y}_{-i}\}\cup\phi_\ell(A_1)))\le\binom{r}{\le\ell}$. Now, let us define the set $\mathcal{H}_i=\{x\in\mathbb{F}_2^{d_0}:\phi_\ell(x)\in\mathsf{span}(\{\mathbf{y}_{-i}\}\cup\phi_\ell(A_1))\}$. By Lemma 3.22, we get

that $|\mathcal{H}_i| \leq 2^{r+1}$. Thus, we now have

$$\mathbb{P}[\mathbf{y}_i \in \mathsf{span}(\{\mathbf{y}_{-i}\} \cup \phi_\ell(A_1))|\mathcal{B}_i] = \frac{|\mathcal{H}_i|}{2^{d_0}} \leq \frac{2^{r+1}}{2^{d_0}} \leq 2^{-\frac{0.35 w_{min} d_0}{\ell}}. \tag{3.11}$$

Applying the above inequality along with (3.10), we get

$$\mathbb{P}[\mathbf{y}_i \notin \mathsf{span}(\{\mathbf{y}_{-i}\} \cup \phi_\ell(A_1))|i \in I_0] \geq 1 - 2^{\frac{-0.35 w_{min} d_0}{\ell}} - e^{-\frac{{w_{min}}^2}{24}\binom{d_0}{\leq \ell}} \geq 1 - 2^{\frac{-0.3 w_{min} d_0}{\ell}}. \tag{3.12}$$

By taking a union bound, it follows that

$$\mathbb{P}[\forall i \in I_0, \mathbf{y}_i \notin \mathsf{span}(\{\mathbf{y}_{-i}\} \cup \phi_\ell(A_1))] \geq 1 - \binom{d_0}{\leq \ell} 2^{\frac{-0.3 w_{min} d_0}{\ell}} \geq 1 - 2^{\frac{-0.2 w_{min} d_0}{\ell}}. \tag{3.13}$$

As we have chosen $d_0$ to be sufficiently large, the right-hand side is at least 0.999 showing that $\mathbb{P}[\mathcal{E}_1] \geq 0.999$.

**Proof of $\mathbb{P}[\mathcal{E}_2] \geq 0.999$:** This follows from a straightforward Chernoff bound on the sampling process defining $I_1$.

**Proof of $\mathbb{P}[\mathcal{E}_3] \geq 0.999$:** This is a direct application of Claim 3.9.

**Proof of $\mathbb{P}[\mathcal{E}_4] \geq 0.999$:** This also follows from Claim 3.9. □

### 3.6. Reduction from Learning Parity with Noise

In this section, we show how the problem of learning a mixture of two (comparable) subspaces captures the notorious hard problem of *learning parity with noise* (LPN).

Given $n \in \mathbb{N}$, the $(n, \varepsilon)$-LPN problem is instantiated by an (unknown) parity function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ and a noise parameter $\varepsilon \in (0, 1/2)$. The samples are generated i.i.d. by a sampling oracle $\mathcal{O} = \mathcal{O}(f, \varepsilon)$ as follows. First, $\mathbf{x} \sim_u \mathbb{F}_2^n$ is sampled uniformly at random from $\mathbb{F}_2^n$. Then $\mathbf{b} \in \{0, 1\}$ is sampled such that $\mathbb{P}[\mathbf{b} = 0] = 1 - \varepsilon$ and $\mathbb{P}[\mathbf{b} = 1] = \varepsilon$. If $\mathbf{b} = 0$, $\mathcal{O}$ outputs $(\mathbf{x}, f(\mathbf{x}))$ and if $\mathbf{b} = 1$, outputs $(\mathbf{x}, 1 - f(\mathbf{x}))$. Given samples generated i.i.d. by the sampling oracle $\mathcal{O}(f, \varepsilon)$, the goal is to learn the unknown parity function $f$.

The following simple proposition reduces LPN to learning mixtures of (comparable) subspaces in $\mathbb{F}_2^{n+1}$, where the subspaces have dimensions $n + 1$ and $n$ respectively.

**Proposition 3.23.** *Suppose there exists an algorithm* ALG *that given samples from a mixture of two subspaces $A_0 = \mathbb{F}_2^{n+1}, A_1 \subseteq \mathbb{F}_2^{n+1}$ of dimensions $n + 1, n$ respectively, with mixing weights $2\varepsilon, 1 - 2\varepsilon$, runs in time $T = T(n, \delta)$ and solves this problem with probability $1 - \delta$. Then there is an algorithm that solves $(n, \varepsilon)$-*LPN *with probability $1 - \delta$ and running time $O(T) + \mathrm{poly}(n)$.*

**Proof.** Consider a sample $(\mathbf{x}, \mathbf{y}) \in \mathbb{F}_2^{n+1}$ (with $\mathbf{x} \in \mathbb{F}_2^n$) drawn from a sampling oracle $\mathcal{O}(f, \varepsilon)$ for the $(n, \varepsilon)$-LPN problem. We can view $(\mathbf{x}, \mathbf{y})$ as a sample from a mixture of two subspace $\mathbb{F}_2^{n+1}, A_1 \subseteq \mathbb{F}_2^{n+1}$ of dimension $n + 1, n$ (respectively) with mixing weights $2\varepsilon, (1 - 2\varepsilon)$ as follows. Let $A_1$ be the subspace of dimension $n$ defined by the linear equation $f(\mathbf{x}) + \mathbf{y} = 0$ over $\mathbb{F}_2$. On the one hand, if $\mathbf{b} = 1$, then $(\mathbf{x}, \mathbf{y}) \in \mathbb{F}_2^{n+1}$ does not belong to $A_1$; it is drawn from $A_0 \setminus A_1$. On the other hand when $\mathbf{b} = 0$, $(\mathbf{x}, \mathbf{y}) \in \mathbb{F}_2^{n+1}$

lies in the subspace $A_1$. But this could correspond to a sample drawn from $A_1$ or to the portion of $A_0$ that overlaps with $A_1$ (recall that $A_1 \subset A_0$ and $|A_0 \cap A_1| = |A_0|/2$ in our case). Hence by setting the mixing weights of the subspaces $A_0 = \mathbb{F}_2^{n+1}, A_1$ to be $2\varepsilon, 1-2\varepsilon$ respectively, we can view a sample $(\mathbf{x}, \mathbf{y})$ drawn from the LPN problem as being drawn from the mixture of subspaces $A_0, A_1$.

Our goal is then to recover $A_0, A_1$ from i.i.d. samples of the form $(\mathbf{x}, \mathbf{y})$ drawn from the LPN problem. If the algorithm ALG succeeds in finding $A_1$, then this provides a parity function $f$ (corresponding to the constraint defining $A_1$) that satisfies the LPN problem. □

The next proposition shows that learning mixtures of two subspaces $A_0, A_1$ in $\mathbb{F}_2^{n+1}$ where $A_0 = \mathbb{F}_2^{n+1}$ and $\dim(A_1) = n$ is in fact equivalent to the LPN problem.

**Proposition 3.24.** *Suppose there is an algorithm ALG that solves $(n, \varepsilon)$-LPN with probability $1 - \delta$ and running time $T = T(n, \delta)$. Then, there is an algorithm that gives samples from a mixture of two subspaces $\mathbb{F}_2^{n+1}, A_1 \subseteq \mathbb{F}_2^{n+1}$ of dimension $n+1, n$ respectively with mixing weights $2\varepsilon, 1-2\varepsilon$, runs in time $O(nT) + \text{poly}(n)$ and recovers $A_1$ with probability $1 - \delta - \exp(-n)$.*

**Proof.** We start with a simple observation. Suppose (*) $x_{i_1} + x_{i_2} + \cdots + x_{i_k} = 0$ be the constraint defining subspace $A_1$, and suppose $j \in \{i_1, i_2, \cdots, i_k\}$. Consider the parity

$$f : \mathbb{F}_2^{\{1,2,\ldots,n+1\}\setminus\{j\}} \to \mathbb{F}_2, \text{ where } f(x) = \sum_{\ell \in \{i_1, i_2, \ldots, i_k\}\setminus\{j\}} x_\ell.$$

On one hand, if $(\mathbf{x}_1, \ldots, \mathbf{x}_{n+1})$ is drawn from $A_1$ (this is with probability $1-2\varepsilon$), then the pair $(\mathbf{x}_{-j}, \mathbf{x}_j)$ satisfies the parity $f$ by definition of $A_1$. On the other hand, if $(\mathbf{x}_1, \ldots, \mathbf{x}_{n+1})$

is drawn from $A_0$ (this is with probability $2\varepsilon$), it satisfies parity $f$ with probability $1/2$. In total, the parity $f$ is satisfied with probability $1 - 2\varepsilon + \frac{1}{2}(2\varepsilon) = 1 - \varepsilon$. Hence, a sample $(\mathbf{x}_1, \ldots, \mathbf{x}_{n+1})$ from the mixture of subspaces with weights $2\varepsilon, 1-\varepsilon$, $(\mathbf{x}_{-j}, \mathbf{x}_j)$ can be viewed as a sample of $(n, \varepsilon)$-LPN with unknown parity $f$.

We do not know $\{i_1, i_2, \ldots, i_k\}$. However, we can guess and try out $j = 1, \cdots, j = n+1$ and get at most $n+1$ candidate hypotheses. We can then use the well-known hypothesis testing result from Proposition B.2 to filter and find the correct subspace $A_1$ with high probability. $\square$

CHAPTER 4

# Robust Subspace Recovery in a Smoothed Analysis Setting

## 4.1. Introduction

Robust subspace recovery is a basic problem in unsupervised learning where we are given $m$ points $x_1, \ldots, x_m \in \mathbb{R}^n$, an $\alpha \in (0, 1)$ fraction of which lie on (or close to) a $d$-dimensional subspace $T$. When can we find the subspace $T$, and hence the "inliers", that belong to this subspace? This problem is closely related to designing a robust estimator for subspace recovery: a $\beta$-robust estimator for subspace recovery approximately recovers the subspace even when a $\beta$ fraction of the points are corrupted arbitrarily (think of $\beta = 1 - \alpha$). The largest value of $\beta$ that an estimator tolerates is called the breakdown point of the estimator. This problem has attracted significant attention in the robust statistics community Rousseeuw [1984], Rousseeuw and Leroy [2005], Donoho and Huber [1983], yet many of these estimators are not computationally efficient in high dimensions. On the other hand, the singular value decomposition is not robust to outliers. Hardt and Moitra Hardt and Moitra [2013] gave the first algorithm for this problem that is both computationally efficient and robust. Their algorithm successfully estimates the subspace $T$ when $\alpha > d/n$, assuming a certain non-degeneracy condition about both the inliers and outliers.[1] This algorithm is also robust to some small amount of noise in each point i.e., the inliers need not lie exactly on the subspace $T$. They complemented their result with a

[1]This general position condition holds in a smoothed analysis setting

computational hardness in the worst-case (based on the Small Set Expansion hypothesis) for finding the subspace when $\alpha < d/n$.

We give a simple algorithm that for any constants $\ell \geq 1, \delta > 0$ runs in $\text{poly}(mn^{\ell})$ time and in a smoothed analysis setting, provably recovers the subspace $T$ with high probability, when $\alpha \geq (1+\delta)(d/n)^{\ell}$. Note that this is significantly smaller than the bound of $(d/n)$ from Hardt and Moitra [2013] when $\ell > 1$. For instance in the setting when $d = (1-\eta)n$ for some constant $\eta > 0$ (say $\eta = 1/2$), our algorithms recover the subspace when the fraction of inliers is *any constant* $\alpha > 0$ by choosing $\ell = O(\log(\alpha)/\log(1-\eta))$, while the previous result requires that at least $\alpha > 1 - \eta$ of the points are inliers. On the other hand, when $d/n = n^{-\Omega(1)}$ the algorithm can tolerate any inverse polynomially small $\alpha$, in polynomial time. In our smoothed analysis setting, each point is given a small random perturbation – each outlier is perturbed with a $n$-variate Gaussian $N(0, \rho^2)^n$ (think of $\rho = 1/\text{poly}(n)$), and each inlier is perturbed with a projection of a $n$-variate Gaussian $N(0, \rho^2)^n$ onto the subspace $T$. Finally, there can be some adversarial noise added to each point (this adversarial noise can in fact depend on the random perturbations).

**Informal Theorem 4.1.** *For any $\delta \in (0,1), \ell \in \mathbb{Z}_+$ and $\rho > 0$. Suppose there are $m = \Omega(n^{\ell} + d/(\delta\alpha))$ points $x_1, \ldots, x_m \in \mathbb{R}^n$ which are randomly $\rho$-perturbed according to the smoothed analysis model described above, with an $\alpha \geq (1+\delta)\binom{d+\ell-1}{\ell}/\binom{n+\ell-1}{\ell}$ fraction of the points being inliers, and total adversarial noise $\varepsilon_0 \leq \text{poly}_{\ell}(\rho/m)$. Then there is an efficient algorithm that returns a subspace $T'$ with $\|sin\Theta(T, T')\|_F \leq \text{poly}_{\ell}(\varepsilon_0, \rho, 1/m)$ with probability at least $1 - \exp\big(-\Omega_{\ell}(\delta n) + 2\log m\big) - \exp(-\Omega(d \log m))$.*

See Section 4.3 for a formal statement, algorithm, and proof. While the above result gives smoothed analysis guarantees when $\alpha$ is at least $(d/n)^\ell < d/n$, the hardness result of Hardt and Moitra [2013] shows that finding a $d$-dimensional subspace that contains an $\alpha < d/n$ fraction of the points is computationally hard assuming the Small Set Expansion conjecture. Hence our result presents a striking contrast between the intractability result in the worst-case and a computationally efficient algorithm in a smoothed analysis setting when $\alpha > (d/n)^\ell$ for some constant $\ell \geq 1$. Further, we remark that the error tolerance of the algorithm (amount of adversarial error $\varepsilon_0$) does not depend on the failure probability.

**Techniques and comparisons.** The algorithm for robust subspace recovery at a high level follows the same approach as Hardt and Moitra [Hardt and Moitra, 2013]. Their main insight was that if we sample a set of size slightly less than $n$ from the input, and if the fraction of inliers is $> (1+\delta)d/n$, then there is a good probability of obtaining $> d$ inliers, and thus there exist points that are in the linear span of the others. Further, since we sampled fewer than $n$ points and the outliers are also in general position, one can conclude that the only points that are in the linear span of the other points are the inliers.

Our algorithm for handling smaller $\alpha$ is simple and is also tolerant to an inverse polynomial amount of adversarial noise in the points. Our first observation is that we can use a similar idea of looking for linear dependencies, but with tensored vectors! Let us illustrate in the case $\ell = 2$. Suppose that the fraction of inliers is $> (1+\delta)\binom{d+1}{2}/\binom{n+1}{2}$. Suppose we take a sample of size slightly less than $\binom{n+1}{2}$ points from the input, and consider the flattened vectors $x \otimes x$ of these points. As long as we have more than $\binom{d+1}{2}$ inliers, we expect to find linear dependencies among the tensored inlier vectors. However,

we need to account for the adversarial error in the points (this error could depend on the random perturbations as well). For each point, we will look for "bounded" linear combinations that are close to the given point. Using Theorem 4.6, we can show that such dependencies cannot involve the outliers. This in turn allows us to recover the subspace even when $\alpha > (d/n)^\ell$ for any constant $\ell$ in a smoothed analysis sense.

We remark that the earlier least singular value bounds of Bhaskara et al. [2014a] can be used to show a weaker guarantee about robust linear independence of the matrix formed by columns $\tilde{x}_i^{\otimes \ell}$ with a $c^\ell$ factor loss in the number of columns (for a constant $c \approx e$). This translates to an improvement over Hardt and Moitra [2013] only in the regime when $d < n/c$. The tight characterization in Theorem 4.6 is crucial for our algorithm to beat the $d/n$ threshold of Hardt and Moitra [2013] for any dimension $d < n$.

## 4.2. Preliminaries

In this section, we introduce notation and preliminary results that will be used throughout the rest of the chapter.

Given a vector $a \in \mathbb{R}^n$ and a $\rho$ (typically a small inverse polynomial in $n$), a *$\rho$-perturbation* of $a$ is obtained by adding independent Gaussian random variables $x_i \sim N(0, \rho^2/n)$ to each coordinate of $a$. The result of this perturbation is denoted by $\widetilde{a}$.

We will denote the singular values of a matrix $M$ by $\sigma_1(M), \sigma_2(M), \ldots$, in decreasing order. We will usually use $k$ or $R$ to represent the number of columns of the matrix. The maximum and minimum (nonzero) singular values are also sometimes written $\sigma_{max}(M)$ and $\sigma_{min}(M)$.

While estimating the minimum singular value of a matrix can be difficult to do directly, it is closely related to the *leave-one-out distance* of a matrix, which is often much easier to calculate.

**Definition 4.2.** *Given a matrix $M \in \mathbb{R}^{n\times k}$ with columns $M_1, \ldots, M_k$, the leave-one-out distance of $M$ is*

$$\ell(M) = \min_i dist(M_i, \text{Span}\{M_j : j \neq i\}). \tag{4.1}$$

The leave-one-out distance is closely related to the minimum singular value, up to a factor polynomial in the number of columns of $M$ Rudelson and Vershynin [2008].

**Lemma 4.3.** *For any matrix $M \in \mathbb{R}^{n\times k}$, we have*

$$\frac{\ell(M)}{\sqrt{k}} \leq \sigma_{min}(M) \leq \ell(M). \tag{4.2}$$

**Tensors and multivariate polynomials.** An order-$\ell$ tensor $T \in \mathbb{R}^{n\times n\times\cdots\times n}$ has $\ell$ modes each of dimension $n$. Given vectors $u, v \in \mathbb{R}^n$ we will denote by $u \otimes v \in \mathbb{R}^{n\times n}$ the outer product between the vectors $u, v$, and by $u^{\otimes\ell}$ the outer product of $u$ with itself $\ell$ times i.e., $u \otimes u \otimes \cdots \otimes u$.

We will often identify an $\ell$th order tensor $T$ (with dimension $n$ in each mode) with the vector in $\mathbb{R}^{n^\ell}$ obtained by flattening the tensor into a vector. For the sake of convenience, we will sometimes abuse notation (when the context is clear) and use $T$ to represent both the tensor and flattened vector interchangeably. Given two $\ell$th order tensors $T_1, T_2$ the inner product $\langle T_1, T_2\rangle$ denotes the inner product of the corresponding flattened vectors in $\mathbb{R}^{n^\ell}$.

A symmetric tensor $T$ of order $\ell$ satisfies $T(i_1, i_2, \ldots, i_\ell) = T(i_{\pi(1)}, \ldots, i_{\pi(\ell)})$ for any $i_1, \ldots, i_\ell \in [n]$ and any permutation $\pi$ of the elements in $[\ell]$. It is easy to see that the set of symmetric tensors is a linear subspace of $\mathbb{R}^{n^{\otimes \ell}}$, and has a dimension equal to $\binom{n+\ell-1}{\ell}$. Given any $n$-variate degree $\ell$ homogenous polynomial $g \in \mathbb{R}^n \to \mathbb{R}$, we can associate with $g$ the unique symmetric tensor $T$ of order $\ell$ such that $g(x) = \langle T, x^{\otimes \ell} \rangle$.

## 4.3. Robust Subspace Recovery

We introduce the following smoothed analysis framework for studying robust subspace recovery. The following model also tolerates some small amount of error in each point i.e., inliers need not lie exactly on the subspace, but just close to it.

### 4.3.1. Input model

In what follows, $\alpha, \varepsilon_0, \rho \in (0, 1)$ are parameters.

(1) An adversary chooses a hidden subspace $T$ of dimension $d$ in $\mathbb{R}^n$, and then chooses $\alpha m$ points from $T$ and $(1-\alpha)m$ points from $\mathbb{R}^n$. We denote these points inliers and outliers respectively. Then the adversary mixes them in arbitrary order. Denote these points $a_1, a_2, \ldots, a_m$. Let $A = (a_1, a_2, \ldots, a_m)$, and $I_{in}, I_{out}$ be the set of indices of inliers and outliers respectively. For convenience, we assume that all the points have lengths in the range $[1/2, 1]$.[2]

(2) Each inlier is $\rho$-perturbed with respect to $T$. (Formally, this means considering an orthonormal basis $B_T$ for $T$ and adding $B_T v$, where $v \sim \mathcal{N}(0, \rho^2/d)^d$.) Each

[2]If the perturbations in step (2) are done proportional to the norm, this assumption can be made without loss of generality. (Since the algorithm can scale the lengths of each of the points.)

outlier is $\rho$-perturbed with respect to $\mathbb{R}^n$. Let $G$ denote the perturbations, and let us write $\widetilde{A} = A + G$.

(3) With the constraint $\|E\|_F \leq \varepsilon_0$, the adversary adds noise $E \in \mathbb{R}^{n\times m}$ to $A$, yielding $\widetilde{A}' = \widetilde{A} + E = (\widetilde{a}'_1, \widetilde{a}'_2, \cdots)$. Note that this adversarial noise can depend on the random perturbations in step 2.

(4) We are given $\widetilde{A}'$.

The goal of the subspace recovery problem is to return a subspace $T'$ close to $T$.

**Notation.** As introduced above, $\tilde{A} = A + G$ denotes the perturbed vectors. $\tilde{a}_i$ denotes the $i$'th column of $\tilde{A}$. We also use the notation $A_I$ to denote the sub-matrix of $A$ corresponding to columns in a set $I$.

### 4.3.2. Our result

We show the following theorem about the recoverability of $T$.

**Theorem 4.4.** *Let $\delta \in (0,1)$, $\ell \in \mathbb{Z}_+$ and $\rho > 0$. Suppose we are given $m \geq n^\ell + 8d/(\delta\alpha)$ points $x_1, x_2, \cdots, x_m \in \mathbb{R}^n$ generated as described above, where the fraction of inliers $\alpha$ satisfies $\alpha \geq (1+\delta)\binom{d+\ell-1}{\ell}/\binom{n+\ell-1}{\ell}$. Then there exists $\varepsilon_0 = \mathrm{poly}_\ell(\rho/m)$ such that whenever $\|E\|_F \leq \varepsilon_0$, there is an efficient deterministic algorithm that returns a subspace $T'$ that satisfies*

(4.3)

$$\|sin\Theta(T,T')\|_F \leq \|E\|_F \cdot \mathrm{poly}_\ell(m, 1/\rho),\ w.p.\ \geq 1 - 2m^2[\exp(-\Omega_\ell(\delta n)) + \exp(-\Omega(d\log m))].$$

When $d/n < 1$, the above theorem gives recovery guarantees even when the fraction of inliers is approximately $(d/n)^\ell$. This can be significantly smaller than $d/n$ (shown in Hardt and Moitra [2013]) for any constant $\ell > 1$.

**Algorithm overview.** We start by recalling the approach of Hardt and Moitra [2013]. The main insight here is that if we sample a set of size slightly less than $n$ from the input, and if the fraction of inliers is $> (1+\delta)d/n$, then there is a good probability of obtaining $> d$ inliers, and thus there exist points that are in the linear span of the others. Further, since we sampled fewer than $n$ points and the outliers are also in general position, one can conclude that the only points that are in the linear span of the other points are the inliers! In our algorithm, the key idea is to use the same overall structure, but with tensored vectors. Let us illustrate in the case $\ell = 2$. Suppose that the fraction of inliers is $> (1+\delta)\binom{d+1}{2}/\binom{n+1}{2}$. Suppose we take a sample of size slightly less than $\binom{n+1}{2}$ points from the input, and consider the flattened vectors $x \otimes x$ of these points. As long as we have more than $\binom{d+1}{2}$ inliers, we expect to find linear dependencies among the tensored inlier vectors. Further, using Theorem 4.6 (with some modifications, as we will discuss), we can show that such dependencies cannot involve the outliers. This allows us to find sufficiently many inliers, which in turn allows us to recover the subspace $T$ up to a small error.

Given $m$ points, the algorithm (Algorithm 7) considers several batches of points each of size $b = (1 - \frac{\delta}{3})\binom{n+\ell-1}{\ell}$. Suppose for now that $m$ is a multiple of $b$, and that the $m/b$ batches form an arbitrary partition of the $m$ points. (See the note in Section 4.3.3 for handling the general case.) In every batch, the algorithm does the following: for each point $u$ in the batch, it attempts to represent $u^{\otimes \ell}$ as a "small-coefficient" linear combination

(defined formally below) of the tensor products of the other points in the batch. If the error in this representation is small enough, the point is identified as an inlier.

**Definition 4.5** ($c$-bounded linear combination)**.** *Let $v_1, v_2, \ldots, v_m$ be a set of vectors. A vector $u$ is said to be expressible as a $c$-bounded linear combination of the $\{v_i\}$ if there exist $\{\alpha_i\}_{i=1}^m$ such that $|\alpha_i| \leq c$ for all $i$, and $u = \sum_i \alpha_i v_i$. Further, $u$ is said to be expressible as a $c$-bounded combination of the $\{v_i\}$ with error $\delta$ if there exist $\{\alpha_i\}_{i=1}^m$ as above with $|\alpha_i| \leq c$ for all $i$, and $\|u - \sum_i \alpha_i v_i\|_1 \leq \delta$.*

Notice that in the above definition, the error is measured by $\ell_1$ norm. In the algorithm, we will need a subprocedure to check whether a vector is expressible as a 1-bounded combination of some other vectors with some fixed error. By the choice of $\ell_1$ norm, this subprocedure can be formulated as a Linear Programming problem, hence we can solve it efficiently.

**Algorithm 7:** Robust subspace recovery

1 Set threshold $\tau = \Omega_\ell(\rho^\ell/n^\ell)$(which is the threshold from Theorem 4.6). Set batchsize $b = (1 - (\delta/3))\binom{n+\ell-1}{\ell}$;
2 Let $V_1, V_2, \cdots, V_r$ be the $r \leq m$ batches each of size $b$ as defined above;
3 Initialize $C = \emptyset$;
4 **for** $i = 1, 2, \cdots, r$ **do**
5  Let $S$ be the set of all $u \in V_i$ such that $\tilde{a}_u'^{\otimes \ell}$ can be expressed as 1-bounded combinations of $\{\tilde{a}_v'^{\otimes \ell} : v \in V_i \setminus \{u\}\}$, with error $\leq \tau/2$;
6  $C = C \cup S$;
7 Return the subspace $T'$ corresponding to the top $d$ singular values of $\tilde{A}'_{\overline{C}}$, for any $2d$-sized subset $\overline{C}$ of $C$;

**Proof outline.** The analysis involves two key steps. The first is to prove that none of the outliers are included in $S$ in step 5 of the algorithm. This is where we use 1-bounded linear combinations. If the coefficients were to be unrestricted, then because the

error matrix $E$ is arbitrary, it is possible to have a tensored outlier being expressible as a linear combination of the other tensored vectors in the batch. The second step is to prove that we find enough inliers overall. On average, we expect to find at least $\frac{\delta}{3}\binom{d+\ell-1}{\ell}$ inlier columns in each batch. We "collect" these inliers until we get a total of $2d$ inliers. Finally, we prove that these can be used to obtain $T$ up to a small error.

We will use the following result in Bhaskara et al. [2019], which helps analyze the linear independencies among the tensored vectors.

**Theorem 4.6** (Bhaskara et al. [2019], Theorem 4.1)**.** *Let $\delta \in (0,1)$, and let $V_\ell$ be space of all symmetric order $\ell$ tensors in $\mathbb{R}^{n\times n\times\cdots\times n}$ (dimension is $D = \binom{n+\ell-1}{\ell}$ ), and let $W \subset V_\ell$ be an arbitrary subspace of dimension $\delta D$. Then we have for any $x \in \mathbb{R}^n$ and $\tilde{x} = x + z$ where $z \sim N(0, \rho^2/n)^n$*

$$\mathbb{P}_z\Big[\|\Pi_W \tilde{x}^{\otimes\ell}\|_2 \geq \frac{c_1(\ell)\rho^\ell}{n^\ell}\Big] \geq 1 - \exp\Big(-c_2(\ell)\delta n\Big),$$

*where $c_1(\ell), c_2(\ell)$ are constants that depend only on $\ell$.*

For convenience, let us write $g(n) := \Omega_\ell(\delta n)$ (which is the exponent in the failure probability from Theorem 4.6). Thus the failure probabilities can be written as $\exp(-g(n))$.

**Lemma 4.7.** *With probability at least $1 - \exp(-g(n) + 2\log m)$, none of the outliers are chosen. I.e., $C \cap I_{out} = \emptyset$.*

**Proof.** The proof relies crucially on the choice of the batch size. Let us fix some batch $V_j$. Note that by the way the points are generated, each point in $V_j$ is $\widetilde{a_i}'$, for some $a_i$ that is either an inlier or an outlier.

Let us first consider only the perturbations (i.e., without the noise addition step). Recall that we denoted these vectors by $\widetilde{a}_i$. Let us additionally denote by $B^{(j)}$ the matrix whose columns are $\widetilde{a}_i^{\otimes \ell}$ for all $i$ in the phase $j$. Consider any $i$ corresponding to an outlier. Now, because the batch size is only $(1-\frac{\delta}{3})\binom{n+\ell-1}{\ell}$, we have (using Theorem 4.6) that the projection of the column $B_i^{(j)}$ orthogonal to the span of the remaining columns (which we denote by $B_{-i}^{(j)}$) is large enough, with very high probability. Formally,

$$\mathbb{P}[\mathrm{dist}(B_i^{(j)}, \mathrm{span}(B_{-i}^{(j)})) \geq \tau] \geq 1 - \exp(-g(n)). \tag{4.4}$$

Indeed, taking a union bound, we have that the inequality $\mathrm{dist}(B_i^{(j)}, \mathrm{span}(B_{-i}^{(j)})) \geq \tau$ holds for all outliers $i$ (and their corresponding batch $j$) with probability $\geq 1 - m^2 \exp(-g(n))$.

We need to show that moving from the vectors $\widetilde{a}_i$ to $\widetilde{a}_i'$ maintains the distance. For this, the following simple observation will be useful.

**Observation 4.8.** *If $a_i$ is an outlier, then*

$$\mathbb{P}[\|\widetilde{a}_i\| \geq 1 + 2\rho] \leq \exp(-n/2).$$

*On the other hand if $a_i$ is an inlier,*

$$\mathbb{P}[\|\widetilde{a}_i\| \geq 1 + 4\rho\sqrt{\log m}] \leq \exp(-4d \log m).$$

Both the inequalities are simple consequences of the fact that the vectors $a_i$ were unit length to start with, and are perturbed by $\mathcal{N}(0, \rho^2/n)$ and $\mathcal{N}(0, \rho^2/d)$ respectively.

Now let us consider the vectors with noise added, $\widetilde{a}'_i$. Note that $\|\widetilde{a}_i - \widetilde{a}'_i\| \leq \varepsilon_0$. Since $\|a_i\| \leq 1$ and since $i$ is an outlier, we have (using Observation 4.8), $\|\widetilde{a}'_i\| \leq 1 + 2\rho + \varepsilon_0$, with probability $\geq 1 - \exp(-n/2)$. Thus for the flattened vectors $\widetilde{a}_i^{\otimes \ell}$, with the same probability,

$$\begin{aligned}\|\widetilde{a}_i^{\otimes \ell} - (\widetilde{a}'_i)^{\otimes \ell}\| &= \|\left(\widetilde{a}_i^{\otimes \ell} - \widetilde{a}_i^{\otimes(\ell-1)} \otimes \widetilde{a}'_i\right) + \left(\widetilde{a}_i^{\otimes(\ell-1)} \otimes \widetilde{a}'_i - \widetilde{a}_i^{\otimes(\ell-2)} \otimes \widetilde{a}_i'^{\otimes 2}\right) + \ldots\| \\ &\leq \ell(\max\{\|\widetilde{a}_i\|, \|\widetilde{a}'_i\|\})^{\ell-1}\varepsilon_0 \\ &\leq \ell(1 + 2\rho + \varepsilon_0)^{\ell}\varepsilon_0. \qquad (4.5)\end{aligned}$$

Thus, for any 1-bounded linear combination of the $b$ vectors in the batch (which may contain both inliers and outliers), $\widetilde{a}_i'^{\otimes \ell}$ is at a Euclidean distance $\leq b\ell(1+\varepsilon_0+4\rho\sqrt{\log m})^{\ell}\varepsilon_0$ to the corresponding linear combination of the $\ell$th powers of the vectors in the batch prior to the addition of noise (i.e., the columns of $B_{-i}^{(j)}$). Thus if $b\ell(1+\varepsilon_0+4\rho\sqrt{\log m})^{\ell}\varepsilon_0 < \tau/2$, then $\widetilde{a}_i'^{\otimes \ell}$ cannot be expressed as a 1-bounded combination of the other lifted vectors in the batch with Euclidean error $< \tau/2$, let alone $\ell_1$ error.

This means that none of the outliers are added to the set $S$, with probability at least $1 - m[\exp(-g(n)) - \exp(-4d\log m)]$. □

Next, we turn to proving that sufficiently many inliers are added to $S$. The following simple lemma will help us show that restricting to 1-bounded combinations does not hurt us.

**Lemma 4.9.** *Let $u_1, u_2, \ldots, u_{d+c}$ be vectors that all lie in a $d$-dimensional subspace of $\mathbb{R}^n$. Then at least $c$ of the $u_i$ can be expressed as 1-bounded linear combinations of $\{u_j\}_{j\neq i}$.*

**Proof.** As the vectors lie in a $d$-dimensional subspace, there exists a non-zero linear combination of the vectors that adds up to zero. Suppose $\sum_i \alpha_i u_i = 0$. Choose the $i$ with the largest value of $|\alpha_i|$. This $u_i$ can clearly be expressed as a 1-bounded linear combination of $\{u_j\}_{j \neq i}$.

Now, remove the $u_i$ from the set of vectors. We are left with $d + c - 1$ vectors, and we can use the same argument inductively to show that we can find $c - 1$ other vectors with the desired property. This completes the proof. □

The next lemma now proves that the set $C$ at the end of the algorithm is large enough.

**Lemma 4.10.** *For the values of the parameters chosen above, we have that at the end of the algorithm,*

$$|C| \geq \frac{\delta/3}{1+\delta/3}\alpha m, \text{ with probability at least } 1 - \exp(-4d \log m).$$

**Proof.** We start with the following corollary to Lemma 4.9. Let us consider the $j$th batch.

*Observation.* Let $n_j$ be the number of inliers in the $j$th batch. If $n_j \geq \binom{d+\ell-1}{\ell} + k$, then the size of $S$ found in Step 5 of the algorithm is at least $k$.

Proof of Observation. Define $B^{(j)}$ as in the proof of Lemma 4.7. Now, since the inliers are all perturbed within the target subspace, we have that the vectors $\widetilde{a}_i^{\otimes \ell}$ corresponding to the inliers all live in a space of dimension $\binom{d+\ell-1}{\ell}$. Thus by Lemma 4.9, at least $k$ of the vectors $B_i^{(j)}$ can be written as 1-bounded linear combinations of the vectors $B_{-i}^{(j)}$.

For inliers $i$, using the fact that $a_i$ are perturbed by $\mathcal{N}(0, \rho^2/d)$, we have

$$\mathbb{P}[\|\widetilde{a}_i\| \geq (1 + 4\rho\sqrt{\log m})] \leq \exp(-4d \log m).$$

Using (4.5) again, we have that $\widetilde{a}_i'^{\otimes \ell}$ can be expressed as a 1-bounded linear combination of the other vectors in the batch, with Euclidean error bounded by $b\ell \cdot (1 + 5\rho\sqrt{\log m})^{\ell}\varepsilon_0$. We know $\ell_1$ norm is a $\sqrt{n^{\ell}}$-approximation of $\ell_2$ norm. By assumption, the $\ell_1$ norm of the error is $< \tau/2$, thereby completing the proof of the observation. □

Now, note that we have $\sum_j n_j \geq \alpha m$, by assumption. This implies that

$$\sum_{j=1}^{m/b} \max\left\{0, n_j - \binom{d+\ell-1}{\ell}\right\} \geq \alpha m - \frac{m}{b}\binom{d+\ell-1}{\ell} \geq \frac{\delta/3}{1+\delta/3}\alpha m.$$

The last inequality follows from our choice of $\alpha$ and the batch size $b$. Thus the size of $S$ in the end satisfies the desired lower bound. □

Finally, we prove that using any set of $2d$ inliers, we can obtain a good enough approximation of the space $T$, with high probability (over the choice of the perturbations). The probability will be high enough that we can take a union bound over all $2d$-sized subsets of $[m]$.

**Lemma 4.11.** *Let $I \subseteq I_{in}$ be any (fixed) set of size $2d$. Then if $\|E\|_F \leq \text{poly}(\rho/m)$, the subspace $U$ corresponding to the top $d$ singular value of $\tilde{A}'_I$ will satisfy*

$$\|\sin\Theta(U, T)\|_F \leq \text{poly}(m, 1/\rho) \cdot \|E\|_F$$

*with probability at least $1 - e^{-4d \log m}$.*

**Proof.** We start by considering the matrix $\tilde{A}_I$ (the matrix without addition of error). This matrix has rank $\leq d$ (as all the columns lie in the subspace $T$). The first step is to argue that $\sigma_d(\tilde{A}_I)$ is large enough. This implies that the space of the top $d$ SVD directions is precisely $T$. Then by using Wedin's theorem Wedin [1972], the top $d$ SVD space $U$ of $\tilde{A}'_I$ satisfies

$$\|\sin\Theta(U,T)\|_F \leq \frac{2\sqrt{d}\|E\|_F}{\sigma_d(\tilde{A}_I) - \|E\|_F}. \tag{4.6}$$

Hence it suffices to show $\sigma_d(\tilde{A})$ is at least inverse-polynomial with high probability.

Recall that $\tilde{A}_I = A_I + G_I$, where $G_I$ is a random matrix. Without loss of generality, we can assume that $T$ is spanned by the first $d$ co-ordinate basis; in this case, every non-zero entry of $G_I$ is independently sampled from $\mathcal{N}(0, \frac{\rho^2}{d})$. We can thus regard $A_I, G_I$ as being $d \times 2d$ matrices. Recall that leave-one-out distance is a good approximation of the least singular value, it suffices to show $\ell((A_I + G_I)^T)$ is at least inverse-polynomial with high probability. Let $A_j, G_j$ denote the $j$th row of $A_I, G_I$ correspondingly. Consider $j \in [d]$, and fix all other rows except $j$th. Let $W$ be the subspace of $\mathbb{R}^{2d}$ that is orthogonal to $\text{span}(\{A_k + G_k : k \in [d], k \neq j\})$, and let $w_1, w_2, \ldots, w_{d+1}$ be an orthonormal basis for $W$. Then for any $t > 0$, if the projection of $(A_j + G_j)$ to $W$ is $< t$ (equivalent to the leave-one-out distance $< t$), then for all $1 \leq i \leq d+1$, we must have $|\langle w_i, A_j + G_j\rangle| \leq t$. Using the anti-concentration of a Gaussian and the orthogonality of the $w_i$, this probability can be bounded by $(t/\rho)^{d+1}$. Choosing $t = \rho/m^4$, this can be made $< (1/m^4)^{d+1}$, and thus after taking a union bound over the $m$ choices of $j$, we have that the leave-one-out distance is $> \rho/m^4$ (and thus $\sigma_d > \rho/m^5$) with probability $\geq 1 - \exp(-4d\log m)$ □

We can now complete the proof of the theorem.

PROOF OF THEOREM 4.4. Suppose that $\|E\|_F \leq \varepsilon_0$ is small enough. Now by Lemma 4.7, we have that $C \subseteq I_{in}$ with probability at least $1 - \exp(-g(n) + \log m$. By Lemma 4.10 and our assumption that $m$ is at least $\Omega(d/(\delta\alpha))$, we know $|C| \geq 2d$ with probability $1 - e^{-4d \log m}$. Finally, by Lemma 4.11 and a union bound over all $2d$ sized subsets of $[m]$, we have that with probability at least $1 - \exp(-\Omega(d \log m))$, for any subset of inliers with size $2d$, the subspace $T'$ corresponding to the top-$d$ singular value will satisfy $\|\sin\Theta(T, T')\|_F \leq \|E\|_F/\text{poly}(m)$. □

### 4.3.3. Batches when $m$ is not a multiple of $b$

the case of $m$ not being a multiple of $b$ needs some care because we cannot simply ignore say the last few points (most of the inliers may be in that portion). But we can handle it as follows: let $m'$ be the largest multiple of $b$ that is $< m$. Clearly $m' > m/2$. Now for $1 \leq j \leq n$, define $\mathcal{D}_j = \{x_j, x_{j+1}, \ldots, x_{j+m'-1}\}$ (with the understanding that $x_{n+t} = x_t$). This is a set of $m'$ points for every choice of $j$. Each $\mathcal{D}_j$ is a possible input to the algorithm, and it has at least $m' > m/2$ points, and additionally the property that $b|m'$.

At least one of the $\mathcal{D}_j$ has $\geq \alpha$ fraction of its points being inliers (by averaging). Thus the procedure above (and the guarantees) can be applied to recover the space. To ensure that no outlier is chosen in step 5 of the algorithm (Lemma 4.7), we take an additional union bound to ensure that Lemma 4.7 holds for all $\mathcal{D}_j$.

# References

Pranjal Awasthi, Avrim Blum, and Or Sheffet. Improved guarantees for agnostic learning of disjunctions. In Adam Tauman Kalai and Mehryar Mohri, editors, *COLT*, pages 359–367. Omnipress, 2010. ISBN 978-0-9822529-2-5. URL `http://dblp.uni-trier.de/db/conf/colt/colt2010.html#AwasthiBS10`.

Pranjal Awasthi, Alex Tang, and Aravindan Vijayaraghavan. Efficient algorithms for learning depth-2 neural networks with general relu activations. *ArXiv 2107.10209*, 2021.

Ainesh Bakshi and Pravesh Kothari. List-decodable subspace recovery via sum-of-squares. *ArXiv*, abs/2002.05139, 2020.

Mikhail Belkin and Kaushik Sinha. Polynomial learning of distribution families. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 103–112. IEEE, 2010.

Ido Ben-Eliezer, Rani Hod, and Shachar Lovett. Random low-degree polynomials are hard to approximate. *computational complexity*, 21(1):63–81, 2012.

Aditya Bhaskara, Moses Charikar, Ankur Moitra, and Aravindan Vijayaraghavan. Smoothed analysis of tensor decompositions. In *Proceedings of the 46th Symposium on Theory of Computing (STOC)*. ACM, 2014a.

Aditya Bhaskara, Moses Charikar, and Aravindan Vijayaraghavan. Uniqueness of tensor decompositions with applications to polynomial identifiability. *Proceedings of the*

*Conference on Learning Theory (COLT).*, 2014b.

Aditya Bhaskara, Aidao Chen, Aidan Perreault, and Aravindan Vijayaraghavan. Smoothed analysis in unsupervised learning via decoupling. In *Proceedings of the 60th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2019.

Christopher M Bishop. Latent variable models. In *Learning in graphical models*, pages 371–403. Springer, 1998.

Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *J. ACM*, 50(4):506–519, July 2003. ISSN 0004-5411. doi: 10.1145/792538.792543. URL `http://doi.acm.org/10.1145/792538.792543`.

Clément L Canonne, Anindya De, and Rocco A Servedio. Learning from satisfying assignments under continuous distributions. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 82–101. SIAM, 2020.

Arun Tejasvi Chaganty and Percy Liang. Spectral experts for estimating mixtures of linear regressions. In *International Conference on Machine Learning*, pages 1040–1048. PMLR, 2013.

Aidao Chen, Anindya De, and Aravindan Vijayaraghavan. Learning a mixture of two subspaces over finite fields. In *Algorithmic Learning Theory*, pages 481–504. PMLR, 2021.

Aidao Chen, Anindya De, and Aravindan Vijayaraghavan. Algorithms for learning a mixture of linear classifiers. In *International Conference on Algorithmic Learning Theory*, pages 205–226. PMLR, 2022.

Sitan Chen and Ankur Moitra. Beyond the low-degree algorithm: Mixtures of subcubes and their applications. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2019, page 869–880, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450367059. doi: 10.1145/3313276.3316375. URL https://doi.org/10.1145/3313276.3316375.

Sitan Chen, Jerry Li, and Zhao Song. Learning mixtures of linear regressions in subexponential time via fourier moments. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 587–600, 2020.

Anindya De, Ilias Diakonikolas, and Rocco A Servedio. Learning from satisfying assignments. In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 478–497. SIAM, 2014.

François Denis, Rémi Gilleron, and Fabien Letouzey. Learning from positive and unlabeled examples. *Theoretical Computer Science*, 348(1):70–83, 2005.

Ilias Diakonikolas and Daniel M Kane. Small covers for near-zero sets of polynomials and learning latent variable models. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 184–195. IEEE, 2020.

David L Donoho and Peter J Huber. The notion of breakdown point. *A festschrift for Erich L. Lehmann*, 157184, 1983.

Ehsan Elhamifar and René Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(11):2765–2781, 2013. doi: 10.1109/TPAMI.2013.57. URL http://dx.doi.org/10.1109/TPAMI.2013.57.

Matthias Ernst, Maciej Liśkiewicz, and Rüdiger Reischuk. Algorithmic learning for steganography: proper learning of k-term dnf formulas from positive samples. In *International Symposium on Algorithms and Computation*, pages 151–162. Springer, 2015.

Jon Feldman, Rocco A. Servedio, and Ryan O'Donnell. PAC learning axis-aligned mixtures of Gaussians with no separation assumption. In *Proceedings of the 19th annual conference on Learning Theory*, COLT'06, pages 20–34, Berlin, Heidelberg, 2006. Springer-Verlag. ISBN 3-540-35294-5, 978-3-540-35294-5. doi: 10.1007/11776420_5. URL http://dx.doi.org/10.1007/11776420_5.

Paulo JSG Ferreira, Bruno Jesus, Jose Vieira, and Armando J Pinho. The rank of random binary matrices and distributed storage applications. *IEEE communications letters*, 17 (1):151–154, 2012.

Venkata Gandikota, Arya Mazumdar, and Soumyabrata Pal. Recovery of sparse linear classifiers from mixture of responses. In *Advances in Neural Information Processing Systems*, volume 33, pages 14688–14698, 2020.

Rong Ge, Jason D. Lee, and Tengyu Ma. Learning one-hidden-layer neural networks with landscape design. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=BkwHObbRZ.

Navin Goyal, Santosh Vempala, and Ying Xiao. Fourier PCA and robust tensor decomposition. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 584–593, 2014. doi: 10.1145/2591796.2591875. URL http://doi.acm.org/10.1145/2591796.2591875.

Moritz Hardt and Ankur Moitra. Algorithms and hardness for robust subspace recovery. In *Conference on Learning Theory*, pages 354–375, 2013.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991. doi: 10.1162/neco.1991.3.1.79.

Majid Janzamin, Hanie Sedghi, and Anima Anandkumar. Beating the perils of non-convexity: Guaranteed training of neural networks using tensor methods. *arXiv preprint arXiv:1506.08473*, 2015.

Majid Janzamin, Rong Ge, Jean Kossaifi, and Animashree Anandkumar. Spectral learning on matrices and tensors. *Foundations and Trends in Machine Learning*, 12, 11 2019. doi: 10.1561/2200000057.

Michael I Jordan and Robert A Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural computation*, 6(2):181–214, 1994.

Adam Tauman Kalai, Ankur Moitra, and Gregory Valiant. Efficiently learning mixtures of two Gaussians. In *Proceedings of the 42nd ACM symposium on Theory of computing*, pages 553–562. ACM, 2010.

Peter Keevash and Benny Sudakov. Set systems with restricted cross-intersections and the minimum rank ofinclusion matrices. *SIAM Journal on Discrete Mathematics*, 18 (4):713–727, 2005.

Jian Li, Yuval Rabani, Leonard J. Schulman, and Chaitanya Swamy. Learning arbitrary statistical mixtures of discrete distributions. In *Proceedings of the Forty-Seventh Annual ACM Symposium on Theory of Computing*, STOC '15, page 743–752, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450335362. doi: 10.1145/2746539.2746584. URL https://doi.org/10.1145/2746539.2746584.

A. Liu and A. Moitra. Efficiently learning mixtures of mallows models. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 627–638, 2018.

Ankur Moitra and Gregory Valiant. Settling the polynomial learnability of mixtures of Gaussians. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 93–102. IEEE, 2010.

Dohyung Park, Constantine Caramanis, and Sujay Sanghavi. Greedy subspace clustering. In *Neural Information Processing Systems*, December 2014.

Keith Y Patarroyo. A digression on hermite polynomials. *arXiv preprint arXiv:1901.01648*, 2019.

Krzysztof Pietrzak. Cryptography from learning parity with noise. In *Proceedings of the 38th International Conference on Current Trends in Theory and Practice of Computer Science*, SOFSEM'12, pages 99–114, Berlin, Heidelberg, 2012. Springer-Verlag. ISBN 978-3-642-27659-0. doi: 10.1007/978-3-642-27660-6_9. URL `http://dx.doi.org/10.1007/978-3-642-27660-6_9`.

Yuval Rabani, Leonard J Schulman, and Chaitanya Swamy. Learning mixtures of arbitrary distributions over large discrete domains. In *Proceedings of the 5th conference on Innovations in theoretical computer science*, pages 207–224, 2014.

Prasad Raghavendra and Morris Yau. List decodable subspace recovery. volume 125 of *Proceedings of Machine Learning Research*, pages 3206–3226. PMLR, 09–12 Jul 2020. URL `http://proceedings.mlr.press/v125/raghavendra20a.html`.

Peter J Rousseeuw. Least median of squares regression. *Journal of the American statistical association*, 79(388):871–880, 1984.

Peter J Rousseeuw and Annick M Leroy. *Robust regression and outlier detection*, volume 589. John wiley & sons, 2005.

Mark Rudelson and Roman Vershynin. The littlewood–offord problem and invertibility of random matrices. *Advances in Mathematics*, 218(2):600 – 633, 2008. ISSN 0001-8708. doi: https://doi.org/10.1016/j.aim.2008.01.010. URL `http://www.sciencedirect.com/science/article/pii/S0001870808000224`.

Mahdi Soltanolkotabi, Ehsan Elhamifar, and Emmanuel J. Candès. Robust subspace clustering. *Ann. Statist.*, 42(2):669–699, 04 2014. doi: 10.1214/13-AOS1199. URL `http://dx.doi.org/10.1214/13-AOS1199`.

Yuekai Sun, Stratis Ioannidis, and Andrea Montanari. Learning mixtures of linear classifiers. In *International Conference on Machine Learning*, pages 721–729. PMLR, 2014.

René Esteban Vidal. Generalized principal component analysis (gpca): an algebraic geometric approach to subspace clustering and motion segmentation, 2003.

Kert Viele and Barbara Tong. Modeling with mixtures of linear regressions. *Statistics and Computing*, 12(4):315–330, 2002.

Per-Åke Wedin. Perturbation bounds in connection with singular value decomposition. *BIT Numerical Mathematics*, 12(1):99–111, 1972.

APPENDIX A

# Boost the Success Probability

The following claim is well-known. We include it here for the sake of completeness.

**Claim A.1.** *Let $X$ be a metric space with metric $d$. There is a fixed hidden element $x^* \in X$. $\varepsilon > 0$. Suppose there is a randomized algorithm* ALG *whose output* $\mathbf{x}$ *satisfies*

$$\mathbb{P}[d(\mathbf{x}, x^*) \leq \varepsilon] \geq 0.9.$$

*Then, there is an algorithm* Success-Prob-Booster *with the following guarantee: given access to $d$, independent outputs from the algorithm* ALG *and a confidence parameter $\delta$,*

(1) *With probability $1 - \delta$, the algorithm returns a estimate $\hat{\mathbf{x}}$ such that*

$$d(\hat{\mathbf{x}}, x^*) \leq 3\varepsilon.$$

(2) *The algorithm acquire $O(\log(1/\delta))$ independent outputs from the algorithm* ALG.

(3) *The algorithm makes $O(\log^2(1/\delta))$ calls to $d$.*

(4) *The algorithm runs in time complexity $O(\log^2(1/\delta))$.*

**Proof of Claim** *A*.1**.** The algorithm is described in Algorithm 8.

**Algorithm 8:** SUCCESS-PROB-BOOSTER

**Input:**
$\delta$ – failure probability
**Output:**
$\hat{\mathbf{x}}$ – estimate of $x^*$

1 Set $t = 1000 \log(1/\delta)$;
2 Acquire $t$ independent outputs $\mathbf{x}_1, \ldots, \mathbf{x}_t$ from the algorithm ALG;
3 Use BFPRT algorithm to select the $(0.3t^2)$th smallest element $\tau$ among $\{d(\mathbf{x}_i, \mathbf{x}_j) : 1 \le i < j \le t\}$;
4 Construct undirected graph $G = (V, E)$ where $V = [t]$ and $(i, j) \in E \iff d(\mathbf{x}_i, \mathbf{x}_j) \le \tau$;
5 Find $k \in [t]$ such that the degree of vertex $k$ is the highest in $G$;
6 **return** $\hat{\mathbf{x}} = \mathbf{x}_k$;

Let $\mathcal{E}$ be the event $|\{i \in [t] : d(\mathbf{x}_i, x^*) \le \varepsilon\}| \ge 0.8t$. Use standard Chernoff bound and the fact $t = 1000 \log(1/\delta)$, we have

$$\mathbb{P}[\mathcal{E}] \ge 1 - \delta.$$

Condition on $\mathcal{E}$. We now show that $\mathbf{x}_k$ (in the last line of the algorithm) satisfies $d(\mathbf{x}_k, x^*) \le 3\varepsilon$.

First we claim $\tau \le 2\varepsilon$. Since $\mathcal{E}$ holds, at least $\binom{0.8t}{2}$ pairs of vertices are $2\varepsilon$-close to each other. Since $\binom{0.8t}{2} \ge 0.3t^2$, we know that $\tau \le 2\varepsilon$.

By the definition of $\tau$, we know that $|E| \ge 0.3t^2$. Then the degree of the vertex $k$ is at least $0.6t$. Since $|\{i \in [t] : d(\mathbf{x}_i, x^*) \le \varepsilon\}| \ge 0.8t$ holds, there exist $j \in [t]$ such that

(1) $(k, j) \in E$.

(2) $d(\mathbf{x}_j, x^*) \le \varepsilon$.

Then $d(\mathbf{x}_k, x^*) \le d(\mathbf{x}_k, \mathbf{x}_j) + d(\mathbf{x}_j, x^*) \le \tau + \varepsilon \le 3\varepsilon$. □

APPENDIX B

# Hypothesis Test

We will prove the following theorem.

**Theorem B.1** (restatement of Theorem 3.12)**.** *Let* $\mathbf{D}$ *be a distribution of a mixture of two incomparable subspaces* $A, B \subseteq \mathbb{F}_2^n$ *with mixing weights* $w_A, w_B \geq w_0$*. Let* $\{A_j, B_j\}_{j=1}^N$ *be a collection of* $N$ *sets of hypothesis with the property that there exists* $i$ *such that* $\{A_i, B_i\} = \{A, B\}$*. There is an algorithm* Choose-The-Right-Hypothesis *which is given a confidence parameter* $\delta$*,* $w_0$*,* $\{A_j, B_j\}_{j=1}^N$ *and a sampler for* $\mathbf{D}$*. Every subspace of* $\{A_j, B_j\}_{j=1}^N$ *will be represented by a basis of that subspace, and the algorithm will have access to the basis. This algorithm has the following behavior,*

(1) *It runs in* $\mathrm{poly}(N, 1/w_0)\log(1/\delta)$ *time.*

(2) *With the probability* $1 - \delta$ *outputs the index* $i$ *such that* $\{A_i, B_i\} = \{A, B\}$*.*

We defer the proof to the end of this section.

In order to prove Theorem 3.12, we need a fundamental tool from statistics, namely "hypothesis testing for distributions". There are many equivalent forms of this algorithm — we use the following (convenient) version from De et al. [2014].

**Proposition B.2** (Simplified [De et al., 2014, Proposition 6])**.** *Let* $\mathbf{D}$ *be a distribution over* $W$ *and* $\mathbf{D}_\varepsilon = \{\mathbf{D}_j\}_{j=1}^N$ *be a collection of* $N$ *distribution over* $W$ *with the property that there exists* $i \in [N]$ *such that* $d_{TV}(\mathbf{D}, \mathbf{D}_i) \leq \varepsilon$*. There is an algorithm* $T^D$ *which is given an*

*accuracy parameter $\varepsilon$, a confidence parameter $\delta$, and is provided with access to (i) samplers for $\mathbf{D}$ and $\mathbf{D}_k$, for all $k \in [N]$ (ii) an evaluation oracle $EVAL_{\mathbf{D}_k}$, for all $k \in [N]$, which, on input $w \in W$, output the value $\mathbf{D}_k(w)$. This algorithm has the following behavior: It makes $m = O((1/\varepsilon^2)(\log N + \log(1/\delta)))$ draws from $\mathbf{D}$ and each $\mathbf{D}_k, k \in [N]$, and $O(m)$ calls to each oracle $EVAL_{\mathbf{D}_k}, k \in [N]$, performs $O(mN^2)$ arithmetic operations, and with probability $1 - \delta$ outputs an index $i^* \in [N]$ that satisfies $d_{TV}(\mathbf{D}, \mathbf{D}_{i^*}) \leq 6\varepsilon$.*

**Theorem B.3.** *$\mathbf{D}(A, B, w_A, 1-w_A)$ is defined as the distribution induced by a mixture of two incomparable subspaces $A, B \subseteq \mathbb{F}_2^n$ of dimension at most $d$ with mixing weights $w_A, 1 - w_A$.*

**Lemma B.4.** *Let $A, B, C, D$ be 4 subspaces of $\mathbb{F}_2^n$. Suppose $\{A, B\} \neq \{C, D\}$. Let $\mathbf{D}_1 = \mathbf{D}(A, B, w_A, 1-w_A), \mathbf{D}_2 = \mathbf{D}(C, D, w_C, 1-w_C), w^* = min(w_A, 1-w_A, w_C, 1-w_C)$. Then $d_{TV}(\mathbf{D}_1, \mathbf{D}_2) \geq w^*/8$.*

**Proof.** Without loss of generality, assume $A$ has the largest dimension among all 4 subspaces. We divide the rest of the analysis into a few cases.

- *Case* 1 : $A \neq C$ and $A \neq D$.
- $A = C$ or $A = D$. Assume A=C.[1]
  - *Case* 2 : $A = B$ or $A = D$.
  - $A \neq B$ and $A \neq D$.
    - *Case* 3 : $A, B$ are incomparable.
    - *Case* 4 : $A, D$ are incomparable.
    - *Case* 5 : $B \subsetneq A$ and $D \subsetneq A$.

Case 1:

In this case, $A \cap C$ and $A \cap D$ are two proper subspaces of $A$. By Claim 3.10, $|A \backslash (C \cup D)| \geq |A|/4$, $d_{TV}(\mathbf{D}_1, \mathbf{D}_2) \geq w^*/4$.

Case 2:

Without loss of generality, assume $A = B$. We have $dim(A) \geq dim(D)$ and $D \neq A$. Hence $A \cap D$ is a proper subspace of $A$. $|(\mathbf{D}_1 - \mathbf{D}_2)(A \backslash D)| = (1 - w_C)|A \backslash D|/|A| \geq w^* \cdot 1/2$.

Case 3:

If $B \subseteq D$, we have $B \subsetneq D$. Since $A, B$ are incomparable, $A, D$ are incomparable. $|(\mathbf{D}_1 - \mathbf{D}_2)(D \backslash (A \cup B)| \geq w^*/4$. If $B \nsubseteq D$, $B \cap D$ is a proper subspace of $B$, $|(\mathbf{D}_1 - \mathbf{D}_2)(B \backslash (A \cup D)| \geq w^*/4$.

Case 4: similar to Case 3.

Case 5:

If $|w_A - w_C| \geq w^*/2$, then $|(\mathbf{D}_1 - \mathbf{D}_2)(A \backslash (B \cup D))| = |w_A - w_C| \cdot |A \backslash (B \cup D))|/|A| \geq w^*/2 \cdot 1/4$. If $|w_A - w_C| \leq w^*/2$, without loss of generality, assume $dim(B) \geq dim(D)$. Since $B \neq D$, $B \cap D$ is a proper subspace of $B$. $|(\mathbf{D}_1 - \mathbf{D}_2)(B \backslash D)| = |(w_A - w_C) \cdot |B \backslash D|/|A| + (1 - w_A)|B \backslash D|/|B|| \geq (1 - w_A)|B \backslash D|/|B| - |(w_A - w_C) \cdot |B \backslash D|/|A|| \geq w^*/2 - w^*/2 \cdot 1/2 = w^*/4$. □

Proof of Theorem 3.12. Set $\varepsilon = w_0/100, M = \lceil 1/\varepsilon \rceil, \gamma = (1 - w_0)/M$. Let $\mathbf{D}_\varepsilon = \{\mathbf{D}(A_j, B_j, w_0 + k * \gamma, 1 - w_0 - k * \gamma\}_{j \in [N], k \in [M] \cup \{0\}}$. It is not hard to see that there exist $\mathbf{D}^* \in \mathbf{D}_\varepsilon$ such that $d_{TV}(\mathbf{D}^*, \mathbf{D}) \leq \varepsilon$. By Proposition B.2, we can find $\mathbf{D}' \in \mathbf{D}_\varepsilon$ such that $d_{TV}(\mathbf{D}', \mathbf{D}) \leq 6\varepsilon$ with probability $1 - \delta$. Say $\mathbf{D}' = \mathbf{D}(A', B', w', 1 - w')$. We claim

[1]This is without loss of generality.

$\{A', B'\} = \{A, B\}$. For a contradiction, suppose it is not true. Then by Lemma B.4, $d_{TV}(\mathbf{D}', \mathbf{D}) \geq w_0/8 > 6\varepsilon$, we derive a contradiction. $\square$

APPENDIX C

# Generalized Chernoff Bound

We will prove a variant of Chernoff bound here.

**Lemma C.1.** *Let $\gamma \in (0,1), d, k \in \mathbb{N}$. Let $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_k$ be a sequence of random variables such that for all $i \in [k]$*

$$\mathbb{P}[(\mathbf{x}_i = 1) \vee (\mathbf{x}_1 + \mathbf{x}_2 + \cdots + \mathbf{x}_{i-1} \geq d) | \mathbf{x}_1, \cdots, \mathbf{x}_{i-1}] \geq \gamma.$$

*Assume $k \geq 2d/\gamma$. Then*

$$\mathbb{P}[\mathbf{x}_1 + \cdots + \mathbf{x}_k \geq d] \geq 1 - \exp\left(-k\gamma^2/8\right).$$

**Proof.** We will use the coupling technique. Define

$$\mathbf{y}_i = \begin{cases} 1 & \text{if } \mathbf{x}_1 + \cdots + \mathbf{x}_{i-1} \geq d. \\ \mathbf{x}_i & \text{otherwise.} \end{cases}$$

Then

(1) $\mathbf{x}_1 + \cdots + \mathbf{x}_k \geq d \iff \mathbf{y}_1 + \cdots + \mathbf{y}_k \geq d.$

(2) For all $i \in [k]$,$\mathbb{P}[\mathbf{y}_i = 1 | \mathbf{y}_1, \cdots, \mathbf{y}_{i-1}] \geq \gamma$.

Define a submartingale $\mathbf{Z}_0, \cdots, \mathbf{Z}_k$ by $\mathbf{Z}_0 = 0$ and $\mathbf{Z}_j = \sum_{1 \le l \le j} \mathbf{y}_l - j\gamma$. Then,

$$
\begin{aligned}
&\mathbb{P}[\mathbf{x}_1 + \cdots + \mathbf{x}_k \ge d] \\
&= \mathbb{P}[\mathbf{y}_1 + \cdots + \mathbf{y}_k \ge d] \\
&= 1 - \mathbb{P}[\mathbf{y}_1 + \cdots + \mathbf{y}_k \le d - 1] \\
&\ge 1 - \mathbb{P}[\mathbf{Z}_k - \mathbf{Z}_0 \le d - 1 - k\gamma] \\
&\ge 1 - \exp\left(-\frac{(k\gamma - (d-1))^2}{2k}\right) && \text{by Azuma–Hoeffding inequality} \\
&\ge 1 - \exp\left(-k\gamma^2/8\right). && \text{by } k\gamma \ge 2d
\end{aligned}
$$

□