DISS. ETH NO. ?

HIGH-DIMENSIONAL OPTIMIZATION PROBLEMS IN AVERAGE-CASE AND WORST-CASE SETTINGS

A dissertation submitted to attain the degree of

DOCTOR OF SCIENCES of ETH ZURICH
(Dr. sc. ETH Zurich)

presented by

DANIIL DMITRIEV

accepted on the recommendation of

Prof. Dr. A. S. Bandeira, examiner
Prof. Dr. F. Yang, co-examiner
Prof. Dr. P. Kothari, co-examiner

2025

Daniil Dmitriev: *High-dimensional optimization problems in average-case and worst-case settings*, © 2025

DOI: 10.3929/ethz-a-

ABSTRACT

English abstract here.

ZUSAMMENFASSUNG

Deutsche Zusammenfassung hier.

ACKNOWLEDGEMENTS

I would like to thank ...

CONTENTS

1

INTRODUCTION

This thesis covers several problems in mathematical optimization and learning theory. Optimization is a vast field which plays a crucial role in the technological progress throughout the history. It was first formalized as a separate subfield at the intersection of mathematics and theoretical computer science in the 20th century, and since then stays an incredibly active area of research. The main driver for the development of mathematical optimization back then were practical problems in operational research, such as logistics and resource allocation. Nowadays, optimization methods are the bedrock of the rapid advances in modern machine learning.

Learning theory studies when is it possible to learn from the data, and what is the most optimal way to do so. It provides a theoretial foundation behind the machine learning algorithms and lies at the intersection of mathematics, statistics, and theoretical computer science.

It is well-known that there exist optimization problems that are fundamentally difficult to solve. As an example, finding the largest clique of a graph is a prototipical problem considered to be *hard*, formally defined via *NP-hardness* **DD:** add citation, KARP. Also, intuitively and somewhat orthogonally, the more parameters the optimization problem has (the larger the optimization space is), the harder it is to optimize over them. However, as it turns out if the optimization problem is *random*, and optimization space is high-dimensional, it becomes possible to argue about the optimal solutions and prove that efficient algorithms can achieve or approximate them. This is also known as *blessings of dimensionality*, the phenomenon closely studied by high-dimensional probability and modern theoretical computer science.

First part of the thesis is devoted to two different optimization problems and is guided by the following questions:

1. Can we solve the optimization problem efficiently?
2. How well can we approximate a hard optimization problem with an efficient algorithm?
3. What are the properties of the optimal or approximate solution?

Second part of the thesis covers two problems in learning theory under the assumption and all or part of data is *worst-case*, i.e., is picked in a worst possible (adversarial) way.

The thesis is organized as follows. The remaining part of this chapter contains a brief introduction to the topics discussed in the thesis. **DD:** Add Chapter discusses the random hitting set problem and the possible greedy heuristics. In **DD:** Add Chapter we study a classical graph problem of estimating the Lovász number, and bound its expected value over a class of highly structured graphs. The second part of the thesis, **DD:** add Chapter presents the robust mixture learning problem. Finally, **DD:** add Chapter presents several open problems which arose during my PhD together with the discussion.

1.1 GREEDY ALGORITHMS FOR THE RANDOM HITTING SET PROBLEM

1.1.1 *Hitting set*

`Hitting Set` is a classical problem in combinatorial optimization which, for a given ground set $\mathcal{X} := \{1, ..., n\}$ and a collection $\mathcal{C} := \{\mathcal{S}_1, ..., \mathcal{S}_m\}$ of subsets of $\mathcal{X}$, asks to identify the smallest set $\mathcal{S} \subseteq \mathcal{X}$ that intersects every subset in $\mathcal{C}$. `Hitting Set` arises naturally from the study of *Minimum Vertex Covers on Hypergraphs* (`MVCH`), upon viewing hyperedges as subsets and vertices as elements of the ground set. This is also known as the *Set Cover* problem [1], which has a rich history in worst-case computational complexity theory, including appearing as one of Karp's 21 NP-complete problems. An important question regards the behaviour of natural random instances of `Hitting Set` where each element of the ground set is independently assigned to any subset with probability p. Such problem formulation is motivated, among others, by applications such as group testing [2]. A classical theorem of Lovász [3] gives an upper bound on the integrality gap in this problem which grows with the degree of the underlying hypergraph, i.e., the maximum number of subsets intersecting any one element. This bound was shown to be tight in the worst-case, but leaves much to be desired from an average-case perspective.

In this chapter, we characterize the average-case integrality gap present in random `Hitting Set` and prove that, with high probability, Lovász's greedy algorithm [3] finds the minimal (up to a multiplicative constant) hitting set in polynomial time.

1.1.2 *Random integer and linear programs*

We consider the following integer programming (IP) formulation of the problem,

$$\mathrm{val}_{\mathtt{IP}} := \begin{cases} \min\limits_{x \in \mathbb{R}^n} & \|x\|_1 \\ \text{subject to} & Ax \geqslant \mathbf{1},\ x \in \{0,1\}^n, \end{cases} \tag{1.1.1}$$

where the i-th row of $A \in \{0,1\}^{m \times n}$ provides a binary encoding of the membership of the elements of $\mathcal{X}$ in the set $\mathcal{S}_i$ and $\mathbf{1} := (1, \ldots, 1) \in \mathbb{R}^m$. With the vertex cover formulation of the problem at hand, we note that A consists of the incidence matrix of the underlying hypergraph. In particular, the constraint $Ax \geqslant \mathbf{1}$ ensures that each set in $\mathcal{C}$ is hit by a prescribed candidate solution vector. A natural convex relaxation is obtained by allowing fractional solutions, and may be expressed as the following linear program (LP),

$$\mathrm{val}_{\mathtt{LP}} := \begin{cases} \underset{x}{\text{minimize}} & \|x\|_1 \\ \text{subject to} & Ax \geqslant \mathbf{1},\ x \in [0,1]^n. \end{cases} \tag{1.1.2}$$

Whilst clearly $\mathrm{val}_{\mathtt{LP}} \leqslant \mathrm{val}_{\mathtt{IP}}$, tightness need not hold in general. In fact, for $m = n$ and $A \in \{0,1\}^{n \times n}$ chosen such that each row and column contains exactly k ones, for some fixed $1 < k < n$, an optimal solution is provided by $x^*_{\mathtt{LP}} = (1/k, ..., 1/k)$, which is not integral, thus leading to a strictly smaller objective whenever n/k is not an integer. This evidences the existence of a multiplicative *integrality gap*, as we define next.

Definition 1.1.1. Given solutions $\mathrm{val}_{\mathtt{IP}}$ and $\mathrm{val}_{\mathtt{LP}}$ to eq. (1.1.1) and eq. (1.1.2) respectively, we define *multiplicative integrality gap* as follows:

$$\mathtt{IPGAP} := \frac{\mathrm{val}_{\mathtt{IP}}}{\mathrm{val}_{\mathtt{LP}}}. \tag{1.1.3}$$

In [3], Lovász proved an essentially optimal worst-case upper bound on the Hitting Set multiplicative integrality gap: $\mathtt{IPGAP} \leqslant 1 + \log d_{\mathtt{max}}$, where $d_{\mathtt{max}}$ corresponds to the maximum degree in the underlying hypergraph. This is obtained by analysing the Greedy algorithm (Algorithm 1), which constructs a vertex cover by sequentially adding vertices with the highest degree amongst the uncovered edges, and will be discussed in more detail in the next sections **DD:** in Chapter Bla. However, in many natural examples, the maximum degree $d_{\mathtt{max}}$ grows with the number of vertices in the hypergraph, thus leading to progressively worse bounds for increasingly

Algorithm 1 `Greedy`

$\mathcal{I} \leftarrow \{I_1, \ldots, I_n\}$ ▷ Inclusion sets
$U \leftarrow [m]$
$t \leftarrow 0$
while $|U| > 0$ **do**
    $P \leftarrow \operatorname{argmax}_{I \in \mathcal{I}} |I \cap U|$ ▷ Greedy step
    $\mathcal{I} \leftarrow \mathcal{I} \setminus \{P\}$
    $U \leftarrow U \setminus P$
    $t \leftarrow t + 1$
$\mathrm{val}_{\mathtt{Gr}} \leftarrow t$
return $\mathrm{val}_{\mathtt{Gr}}$

large hypergraphs. Besides being arguably the most natural candidate for solving `Hitting Set`, the greedy algorithm has been shown to be the best possible polynomial time approximation algorithm [4] for the worst-case instances of this classical problem.

Despite extensive work conducted on `Hitting Set` in the last decades, a gap remains in our understanding of the typical performance of linear programming and the greedy algorithm on random problem instances. We hence pose the following questions:

1. Are there integrality gaps in random instances of `Hitting Set`?
2. Can near-optimal solutions be found efficiently?

In the present work **DD:** add Chapter, we provide answers to the above questions *with high probability* (w.h.p.) in a non-asymptotic sense, in the setting where the cardinality n of the ground set $\mathcal{X}$ is large but finite. We prove the absence of integrality gaps up to constants in a wide regime of n, m, p, by conducting an average case analysis of an algorithm that outputs integral covers of matching size to the fractional ones. In addition, a rigorous analysis of the greedy routine will follow by a straightforward reduction.

1.1.3 *Greedy algorithm*

The core principle of `Greedy`, Algorithm 1, is to construct a feasible solution in steps, by sequentially adding to the candidate solution an element which hits the largest number of remaining sets. In the chosen setting, where

elements are added to sets with equal probability and independently of each other, we have precise estimates on the number of subsets hit by an element which is *picked first*. In fact, the size of this set is given by the maximum of independent Binomial random variables, which is analysed in Section 2.4. However, this very first step introduces nontrivial dependencies amongst the remaining matrix columns and significantly complicates keeping track of the marginal gains of each subsequent element addition to the candidate solution.

In order to circumvent this issue, we introduce a modified greedy routine, which we refer to as the `BlockGreedy` algorithm, where the elements of the ground set $[n]$ are split into separate sets of a given size, which we call blocks. At the t-th iteration, the algorithm picks the element hitting the largest number of remaining sets across *the first t blocks only*. By choosing the size of the blocks appropriately, we have that at each iteration t one is guaranteed to find a solution of near-optimal size within the set of newly-included independent columns.

`BlockGreedy` is detailed in Algorithm 3, whilst informally, it works as follows.

1. Let K be the size of the solution (suggested by theoretical analysis);
2. Uniformly at random split n columns into K blocks with n/K columns per block;
3. Start with an empty set of possible choices of columns;
4. At the t-th iteration, first add the columns from the t-th block (Step 6 **DD:** Fix reference here and next). Then, perform one greedy step on the current set of possible choices (Step 7);
5. If after K iterations of the algorithm, some subsets remain uncovered, we use a trivial covering, i.e., covering each subset by a separate column.

Note that the first selection of the element which hits the most number of subsets again introduces dependencies. However, the columns that are in the newly added block are independent of everything else at time t.

1.1.4 *Main contributions*

Given a collection of sets $\mathcal{C} := \{\mathcal{S}_1, ..., \mathcal{S}_m\}$, we define inclusion sets $I_j := \{i \in [m] : j \in \mathcal{S}_i\}$, for $j \in [n]$, which in the `MVCH` formulation of the problem

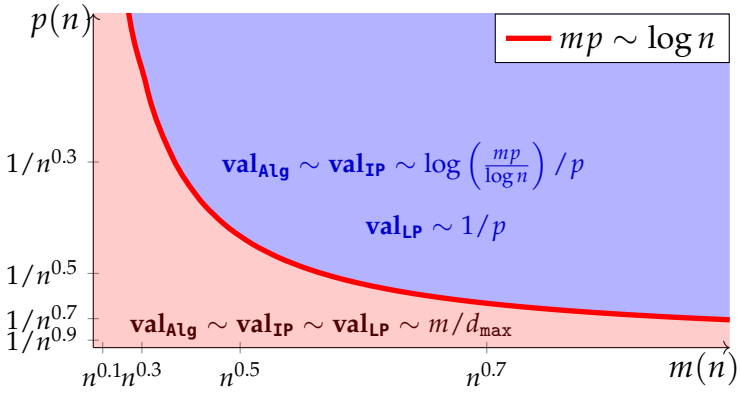

FIGURE 1.1: Transition between the sparse and the dense regime for different values of the average inclusion set size mp.

at hand correspond to the set of hyperedges incident to any given vertex. Furthermore, we let $d_{\max} := \max_{j\in[n]} |I_j|$. Throughout, we use the notation $\mathrm{val}_{\mathtt{Gr}}$, $\mathrm{val}_{\mathtt{Alg}}$ to denote the size of the hitting set returned by `Greedy` and `BlockGreedy` respectively. Below we provide an informal description of the main results (also shown in Figure 1.1) which hold with high probability, where $A(n) \sim B(n)$ denotes that $cA(n) \leqslant B(n) \leqslant CA(n)$ for large enough n and for some constants $c, C > 0$:

sparse regime ($mp \ll \log n$): We show that $\mathtt{IPGAP} \sim 1$ in the sparse regime by proving that the `BlockGreedy` algorithm succeeds in reaching the `LP` lower bound of $\frac{m}{d_{\max}}$.

$$\mathrm{val}_{\mathtt{Alg}} \sim \mathrm{val}_{\mathtt{IP}} \sim \mathrm{val}_{\mathtt{LP}} \sim \frac{m}{d_{\max}}.$$

dense regime ($mp \gg \log n$): We prove that $\mathtt{IPGAP} \sim \log \frac{mp}{\log n}$ in the dense regime. We show that the `BlockGreedy` algorithm performs as well as `IP` in this regime, i.e.

$$\frac{1}{p}\log\left(\frac{mp}{\log n}\right) \sim \mathrm{val}_{\mathtt{Alg}} \sim \mathrm{val}_{\mathtt{IP}} \gg \mathrm{val}_{\mathtt{LP}} \sim \frac{1}{p} \sim \frac{m}{d_{\max}}.$$

threshold regime ($mp \sim \log n$): This regime smoothly interpolates between the sparse and dense ones, with $\mathtt{IPGAP} \sim 1$. The scaling for all quantities of interest is $m/d_{\max} \sim 1/p$.

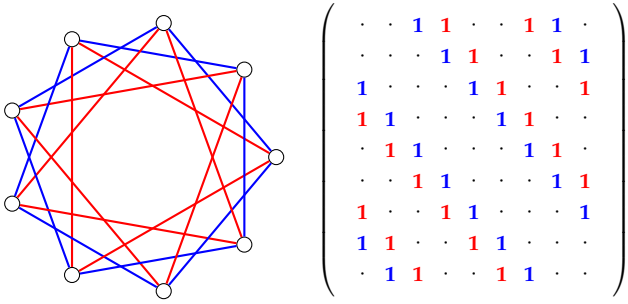

FIGURE 1.2: Circulant graph on 9 vertices and its adjacency matrix (0's replaced by dots). Each vertex i is connected to vertices $i \pm 2$ and $i \pm 3 \mod 9$.

greedy: We prove that $\mathrm{val}_{\mathtt{Gr}} \sim \mathrm{val}_{\mathtt{IP}}$ when $\frac{\log 1/p}{\log n} < 1/2$.

1.2 LOVÁSZ NUMBER FOR RANDOM CIRCULANT GRAPHS

1.2.1 *Lovász number*

The Lovász number ϑ is a well-known statistic of an arbitrary simple undirected graph G. As Lovász first observed in [5], one can define a number $\vartheta(G)$ as the value of a certain semidefinite program (SDP) whose constraints depend on the adjacency matrix of G. The Lovász number provides an upper bound on the Shannon capacity of the graph and satisfies the following inequalities:

$$\omega(G) \leqslant \vartheta(\overline{G}) \leqslant \chi(G), \tag{1.2.1}$$

where $\omega(G)$ is the size of the largest clique in G, $\chi(G)$ is the chromatic number of G, and $\overline{G}$ is the complement of G. This observation is remarkable, since ϑ is computable in polynomial time, while ω and χ are famously NP-hard to compute.

The Lovász number has been studied for a variety of random graph models including the classical Erdős-Rényi (ER) random graph $G(n,p)$. Its expected value was first studied by Juhász [6], who showed that $\mathbb{E}\,\vartheta(G) = \Theta(\sqrt{n/p})$ for $\frac{\log^6 n}{n} \leqslant p \leqslant 1/2$. For $p = 1/2$, Arora and Bhaskara [7] showed that $\vartheta(G)$ concentrates around its median in an interval of polylogarithmic length. In the sparse regime $p < n^{-1/2}$, it has been further shown that $\vartheta(G)$ concentrates around its median in an interval of

constant length [8]. To the best of our knowledge, determining the correct constant in the $\Theta(\sqrt{n/p})$ asymptotic remains an open question.

1.2.2 *Random circulant graphs*

In this chapter, we focus on a class of random *circulant* graphs (RCGs), a family of vertex-transitive graphs with a circulant adjacency matrix; see Figure 1.2 and Theorems 3.1.3 and 3.1.4 **DD:** Why theorems, not definitions???. We emphasize that RCGs are fully determined by the connectivity of any given single vertex. Therefore, a dense RCG can be generated with $\frac{n-1}{2}$ random bits, where each bit affects the presence of n edges, in contrast to the $\frac{n(n-1)}{2}$ random bits in $G(n, 1/2)$, each affecting just one edge. In this sense, RCGs may be viewed as a "partial derandomization" of ER graphs. Indeed, circulant graphs are precisely Cayley graphs on the group $\mathbb{Z}_n$, and general random Cayley graphs have long been studied for similar purposes in theoretical computer science.

It is therefore of interest to understand to what extent the above results for ER graphs also apply to RCGs. For dense RCGs, the asymptotics of the clique number and the chromatic number are well-understood: [9] showed a high-probability upper bound on the clique number $\omega(G) = O(\log n)$, and later [10] proved that the chromatic number is at most $(1 + o(1))\frac{n}{2\log_2 n}$ with high probability. These results imply bounds on the Lovász number through eq. (1.2.1), but the resulting upper and lower bounds are far apart.

Our main result is much sharper upper and lower bounds on the expected Lovász number of a dense RCG.

Theorem 1.2.1. *There exists a constant $C > 0$ such that, for a dense random circulant graph G on n vertices (Theorem 3.1.4),*

$$\sqrt{n} \leqslant \mathbb{E}\,\vartheta(G) \leqslant C\sqrt{n \log\log n}. \tag{1.2.2}$$

1.2.3 *Proof strategy*

Our proof of the upper bound in Theorem 1.2.1 relies on the algebraic structure of circulant graphs. First, following [11], we transform the SDP formulation of $\vartheta(G)$ to a linear program (LP) using the fact that the cir-

culant matrices are diagonalizable by a discrete Fourier transform (DFT) Theorem 3.2.1 gives the resulting LP:

$$\vartheta(G) = \max_{(y_0,\dots,y_{n-1})\in\mathbb{R}^n} \langle y, g\rangle\,,$$
$$\text{subject to } \begin{cases} y_k = y_{n-k} \text{ for } k = 1,\dots,n-1, \\ \|y\|_1 = 1, y \geqslant \mathbf{0}, \\ \langle y, f_k\rangle = 0 \text{ for all edges } (0,k). \end{cases} \tag{1.2.3}$$

Here, f_k is the k-th row of the DFT matrix F, and $g := Fb$ for $b \in \{\pm 1\}^n$ with $b_0 = 1$ and $b_k = 1$ if $(0,k)$ is not an edge, and -1 otherwise, for $1 \leqslant k \leqslant n-1$. We denote $\mathbf{0} := (0,\dots,0)$ and $y \geqslant \mathbf{0}$ stands for entrywise positivity of y.

The last constraint in eq. (1.2.3) requires the Fourier transform of y to have a specific sparsity pattern. Uncertainty principles for the Fourier transform (see, e.g., [12]) then suggest that all feasible vectors y must be dense [13]. A quantitative version of this "density" would be enough to bound the LP. To illustrate, suppose that y is a feasible vector with $\|y\|_1 = 1$ and its mass is spread almost uniformly among its coordinates, i.e., that $\|y\|_2 \leqslant \frac{c}{\sqrt{n}}\|y\|_1 = \frac{c}{\sqrt{n}}$, for some constant $c > 0$. Since $\|g\|_2 = n$, Cauchy-Schwarz inequality would give $\langle y, g\rangle \leqslant \|y\|_2\|g\|_2 \leqslant c\sqrt{n}$, proving upper bound in Theorem 1.2.1 without the extra $\sqrt{\log\log n}$ factor.

The second part of our proof, Theorem 3.2.4, makes the aforementioned intuition rigorous, relying on the *restricted isometry property* (RIP, Theorem 3.1.5). The f_k in our constraints form a so-called *subsampled DFT basis*, which is a random subset of the Fourier basis. The RIP for such bases is in fact a celebrated topic in the compressed sensing literature. RIP was first introduced and studied for subsampled DFT bases in seminal work of Candès and Tao [14], and since then, one of the central questions for compressed sensing is the number of f_k needed for RIP to hold. Theorem 3.4.2 describes a simplified version of the current best bound due to [15] which is sufficient for our purposes. Interestingly, our upper bound proof only uses the fact that feasible solutions of eq. (1.2.3) lie on a (random) nullspace of a subsampled DFT matrix, and omits the positivity constraint $y \geqslant \mathbf{0}$. However, as we discuss in Section 4.4, we believe that this constraint is important for tighter results.

1.3 LOWER BOUNDS FOR ONLINE PRIVATE LEARNING

With the increasing need to protect the privacy of sensitive user data while conducting meaningful data analysis, Differential Privacy (DP) [16] has become a popular solution. DP algorithms ensure that the impact of any single data sample on the output is limited, thus safeguarding individual privacy. Several works have obtained DP learning algorithms for various learning problems in both theory and practice.

However, privacy does not come for free and often leads to a statistical (and sometimes computational) cost. The classical solution for non-private Probably Approximately Correct (PAC) learning [17] is via Empirical Risk Minimisation (ERM) that computes the best solution on the training data. Several works [18, 19] have shown that incorporating DP into ERM incurs a compulsory statistical cost that depends on the dimension of the problem. In the well-known setting of PAC learning with DP, Kasiviswanathan *et al.* [20] provided the first guarantees that all finite VC classes can be learned with a sample size that grows logarithmically in the size of the class. This line of research was advanced by subsequent works [21–23], resulting in the findings of Alon *et al.* [24] which established a surprising equivalence between non-private online learning and Approximate DP-PAC learning.

Unlike the setting of PAC learning, Online learning captures a sequential game between a learner and an adversary. The adversary knows everything about the learner's algorithm except its random bits. In this work we consider a setting where, for a known hypothesis class $\mathcal{H}$, the adversary chooses a sequence of data points $\{x_1, \ldots, x_t\}$ and the target hypothesis $f^* \in \mathcal{H}$ prior to engaging with the learner. Then, the adversary reveals these data points one by one to the learner, who must offer a prediction for each. After each prediction, the adversary reveals the true label for that point. The learner's performance is evaluated by comparing the incurred mistakes against the theoretical minimum that could have been achieved by an optimal hypothesis in hindsight. Known as the *realisable oblivious mistake bound* model, the seminal work of Littlestone [25] showed that i) the number of mistakes incurred by any learner is lower-bounded by the Littlestone dimension (more precisely, $\mathrm{Ldim}(\mathcal{H})/2$) of the target class $\mathcal{H}$ and ii) there is an algorithm that makes at most $\mathrm{Ldim}(\mathcal{H})$ mistakes. This algorithm is commonly referred to as the Standard Optimal Algorithm (SOA).

Recall that certain problem classes possess finite Vapnik-Chervonenkis (VC) dimensions but infinite Littlestone dimensions (such as the one-

dimensional threshold problem). This, together with the equivalence between non-private online learning and DP-PAC learning [24] implies that there exists a fundamental separation between DP-PAC learning and non-private PAC learning. In other words, some learning problems can be solved with vanishing error, as the amount of data increases, in PAC learning but will suffer unbounded error in DP-PAC learning. This implication was first proven for pure DP by Feldman & Xiao [22] and later for approximate DP by Alon *et al.* [26]. With the debate on the sample complexity of approximate DP-PAC learning resolved, we next ask whether a similar gap exists between online learning with DP and non-private online learning. Golowich & Livni [27] addressed this by introducing the Differentially Private Standard Optimal Algorithm (DP-SOA), which suffers a mistake count, that increases logarithmically with the number of rounds T compared to a constant error rate in non-private online learning [25]. This difference suggests a challenge in DP online learning, where errors increase indefinitely as the game continues. The question of whether this growing error rate is an unavoidable aspect of DP-online learning was posed as an open question by Sanyal & Ramponi [28].

MAIN RESULT In this work, we provide evidence that this additional cost is inevitable. Consider any hypothesis class $\mathcal{H}$ and for a learning algorithm $\mathcal{A}$. Let $\mathbb{E}[M_{\mathcal{A}}]$ be the expected number of mistakes incurred by $\mathcal{A}$ and let T be the total number of rounds for which the game is played.

We obtain a lower bound on $\mathbb{E}[M_{\mathcal{A}}]$ under some assumptions on the learning algorithm $\mathcal{A}$. Informally, we say an algorithm $\mathcal{A}$ is β-concentrated (see Theorem 4.3.2 for a formal definition) if there is some output sequence that it outputs with probability at least $1-\beta$ in response to a *non-distinguishing* input sequence. A *non-distinguishing* input sequence is a (possibly repeated) sequence of input data points such that there exists some $f_1, f_2 \in \mathcal{H}$ which cannot be distinguished just by observing their output on the non-distinguishing input sequence. We prove a general statement for any hypothesis class in Theorem 4.3.3 but show a informal corollary below.

Corollary 1.3.1 (Informal Corollary of Theorem 4.3.3)**.** *There exists a hypothesis class $\mathcal{H}$ with* $\mathrm{Ldim}(\mathcal{H}) = 1$ *(see Theorem 4.1.2), such that for any $\varepsilon, \delta > 0, T \leqslant \exp(1/(32\delta))$, and any online learner $\mathcal{A}$ that is (ε, δ)-DP and* 0.1*-concentrated, there is an adversary, such that*

$$\mathbb{E}[M_{\mathcal{A}}] = \widetilde{\Omega}\left(\frac{\log T}{\varepsilon}\right), \tag{1.3.1}$$

where $\widetilde{\Omega}$ *hides logarithmic factors in* ε*. For* $T > \exp(1/(32\delta))$, $\mathbb{E}[M_{\mathcal{A}}] = \widetilde{\Omega}(1/\delta)$.

While the above result uses a hypothesis class of Littlestone dimension one, our main result in Section 4.3.1 also holds for any hypothesis class, even with Littlestone dimension greater than one. Utilising the Point_N hypothesis class (see Theorem 4.1.4) in Theorem 1.3.1, we demonstrate that the minimum number of mistakes a DP online learner must make is bounded below by a term that increases logarithmically with the time horizon T. This holds if the learning algorithm is concentrated and T is less than or equal to $\exp(1/(32\delta))$. This contrasts with non-private online learning, where the number of mistakes does not increase with T in hypothesis classes with bounded Littlestone dimension, even if the learner is concentrated.[1] Our result also shows that the analysis of the algorithm of Golowich & Livni [27], which shows an upper bound of $\Omega(\log(T/\delta))$ for DP-SOA is tight as long as $T \leqslant \exp(1/(32\delta))$. However, as illustrated in Figure 4.1, this is not a limitation as for larger T, since a simple *Name & Shame* algorithm incurs lesser mistake than DP-SOA albeit at vacuous privacy levels (see discussion after Theorem 4.3.3).

In fact, the assumption of concentrated learners is not overly restrictive given that known DP-online learning algorithms exhibit this property, as detailed in Section 4.3.2. Notably, the DP-SOA presented by Golowich & Livni [27] which is the sole DP online learning method known to achieve a mistake bound $O(\log(T))$, is concentrated as shown in Theorem 4.3.6. This suggests that the lower bound holds for all potential DP online learning algorithms.

Additionally, we extend our result to another class of DP online algorithms, which we refer to as *uniform firing* algorithms, that are in essence juxtaposed to concentrated algorithms. These algorithms initially select predictors at random until a certain confidence criterion is met, prompting a switch to a consistent predictor—this transition, or 'firing', is determined by the flip of a biased coin (with bias p_t), where the likelihood of firing increases with each mistake. However, the choice of how p_t increases and the selection of the predictor upon firing depend on the algorithm's design. For this specific type of algorithms, particularly in the context of learning the Point_3 hypothesis class, Proposition 4.4.2 establishes a lower bound on mistakes that also grows logarithmically with T.

Section 4.4.1 discusses Continual Observation [29, 30], another popular task within sequential DP. We show that results on DP Continual Counters

1 All deterministic algorithms are 0-concentrated by definition.

can be used to derive upper bounds in the online learning setting. Nonetheless, it is not clear whether lower bounds for that setting can transferred to DP-Online learning. In addition, these upper bounds suffer a dependence on the hypothesis class size.

Finally, we point out that, to the best of our knowledge we are unaware of any algorithms in the literature for pure DP online learning. Our lower bound in Theorem 1.3.1 immediately provides a lower bound for pure DP. Similarly, DP Continual Counters provide a method for achieving upper bounds, specifically for the Point_N classes, albeit with a linear dependency on N. Obtaining tight upper and lower bounds remains an interesting direction for future research.

1.4 ROBUST MIXTURE LEARNING

1.4.1 *Robust mean estimation*

A classical problem in robust statistics is *robust mean estimation*. It is formulated as follows: For a known family of distributions $\mathcal{P}$, a learner is provided with a dataset of n points, where one part comes from a distribution $P \in \mathcal{P}$, while the rest is generated by an adversarial model. The task of the learner is to reliably and accurately estimate $\mathbb{E}_P X$, the mean of P.

A few clarifications are in order. First, there exist several different adversarial models, some of them listed below. In what follows, $\varepsilon \in (0, 1/2)$ denotes the proportion of adversarial points and n denotes the total dataset size.

1. *Strong adversary.* In this model, in the beginning n samples are generated i.i.d. from P. Adversary removes or replaces up to $\lceil \varepsilon n \rceil$ of them with arbitrary points. The learner then receives the modified dataset.

2. *Weak adversary.* Here, adversary picks an arbitrary distribution Q. The learner receives the dataset of i.i.d. samples from the mixture $(1-\varepsilon)P + \varepsilon Q$.

3. *Mean-shift contamination.* For this model, let $\overline{P}$ denote the centered distribution P. The adversary picks $k := \lceil \varepsilon n \rceil$ points $z_1, \ldots, z_k$. The final dataset consists of $n - k$ i.i.d. samples from P and the remaining k points are obtained by sampling $x_i \overset{\text{i.i.d.}}{\sim} \overline{P}$ and adding z_i to it.

These models are ordered by how restricted the adversary is. Weak adversary model is also known as Huber contamination model. In all models,

adversary has the knowledge of $\mathcal{P}$, and of the algorithm which the learner will be using (apart from the internal randomness of the algorithm).

Second, our goal is to design an algorithm for the learner that is computationally efficient in high dimenions. In particular, its time and sample complexity must be (quasi-)polynomial in the dimension d of the samples.

Finally, *reliable* estimate means that the guarantees of the algorithm must hold with high probability over the randomness of the sampling procedure and the internal randomness of the algorithm. *Accurate* estimate means that the error (usually measured as the ℓ_2 distance between the estimate and the true parameter) does not depend on the dimension d.

In one dimensional case, the median is a great candidate for an efficient and robust mean estimator. Indeed, when the samples come from $\mathcal{N}(\mu, 1)$ in the strong adversary model **DD:** how general is this?, it is easy to show that the median $\hat{\mu}$ satisfies $\|\hat{\mu} - \mu\|_2 = O(\varepsilon)$ with high probability. From straightforward TV distance arguments, $O(\varepsilon)$ is the optimal achievable error.

However, computing median in high dimensions is challenging. Consider for simplicity the case $\mathcal{P} = \{\mathcal{N}(\mu, I), \text{ for } \mu \in \mathbb{R}^d\}$. If we compute median coordinate-wise, then the (in general unimprovable) error is $O(\sqrt{d}\varepsilon)$, which is not accurate, since it depends on the dimension d. There exists another generalization of median to high dimensions, called *Tukey median*, which is defined as follows:

$$\mu_{\text{Tukey}} := \arg\max_{\hat{\mu} \in \mathbb{R}^d} \inf_{v \in \mathbb{R}^d} \Pr(v^\top (X - \hat{\mu}) \geqslant 0). \tag{1.4.1}$$

Tukey median is an accurate estimator: it achieves error $O(\varepsilon)$ with high probability. However, to compute Tukey median is an NP-hard problem, thus it cannot be used as a candidate for an efficient algorithm.

Recent breakthrough results **DD:** add citation show how to achieve error $O(\varepsilon)$ efficiently. One of the ideas is a filtering approach.

1.4.2 *Clustering and list-decodable mean estimation*

Clustering is one of the most commonly known problems in unsupervised learning. The goal of the clustering is to split a given dataset into several non-intersecting groups. The desirable condition of this is split is that, for some similarity score or a metric, samples within the same group (or cluster), are more similar or closer than samples from different groups.

Among well-known solutions for clustering are K-Means (see Algorithm 2) and DBSCAN **DD:** add citation. Both methods are widely used

Algorithm 2 K-Means

Input: Samples $S = \{x_1, \dots, x_n\}$, number of centers k.
Output: Clusters $\mathcal{C} = \{C_1, \dots, C_k\}$.

1: Initialize cluster centers $\mu_1, \dots, \mu_k$. ▷ Can be done at random, or with some heuristic, see K-Means++ **DD:** add citation
2: $\mathcal{C} \leftarrow \emptyset$.
3: Form clusters $\mathcal{C}' = \{C'_1, \dots, C'_k\}$ by assigning each point x_i to the closest cluster center μ_t.
4: **while** $\mathcal{C} \neq \mathcal{C}'$ **do**
5: $\quad \mathcal{C} \leftarrow \mathcal{C}'$.
6: $\quad$ For each $t \in [k]$ set μ_t as the average of C'_t.
7: $\quad$ Recompute clusters $\mathcal{C}' = \{C'_1, \dots, C'_k\}$ as before.
8: **return** $\mathcal{C}$

in practice, and are established methods of unsupervised learning. However, the theoretical guarantees for them are lacking, especially in high dimensions.

Consider the sampling process where the data is generated from the spherical Gaussian mixture model (GMM):

$$X_1, \dots, X_n \overset{\text{i.i.d.}}{\sim} \frac{1}{k} \sum_{i=1}^{k} \mathcal{N}(\mu_i, I) \tag{1.4.2}$$

A natural question is: how large does the separation $\min_{i \neq j} \|\mu_i - \mu_j\|_2$ needs to be, to allow for an efficient clustering procedure? This is an active area of research in theoretical computer science **DD:** add citation, and the state-of-the-art methods are quite distinct from the classical K-Means or DBSCAN. One of the promising approaches to design accurate and efficient clustering algorithms is based on the Sum-of-Squares hierarchy; we refer to **DD:** add citation for detailed survey.

Interestingly, there is a close connection between robust mean estimation and clustering problems. Recall that in the definition of robust mean estimation, we constrained the proportion of adversarial points to $\varepsilon < 1/2$. In *list-decodable mean estimation*, adversarial points constitute instead the majority of points, i.e., proportion of inlier samples $\alpha := 1 - \varepsilon < 1/2$.

Consider the case when the data is sampled from uniform GMM with two mixture components. Clearly, we need to relax our definition of accurate estimate: it is information-theoretically impossible to distinguish which component is a true one (inlier), and which is generated adversarially. To

circumvent this issue, list-decodable paragidm **DD:** add citation allows the algorithm to output a (short) *list of estimators* with the guarantee that with high probability there exists an element in this list close to the true parameter. In the example above, a possible algorithm can output list of size two, containing centers of both mixture components.

Formally speaking, in list-decodable mean estimation data is sampled from the distribution $\alpha P + (1-\alpha)Q$, where $P \in \mathcal{P}$ is the distribution of interest (inlier distribution), and Q is an adversarial distribution. Parameter $\alpha \in (0, 1/2)$ controls the proportion of inlier samples, and the goal as before is to estimate the mean of P.

To see how list-decodable mean estimation is connected to clustering, consider our previous example where data is generated from $\frac{1}{n}\sum_{i=1}^{k}\mathcal{N}(\mu_i, I)$. If we set $\mathcal{P} = \left\{\mathcal{N}(\mu, I) \text{ for } \mu \in \mathbb{R}^d\right\}$, we can feed this data to the algorithm which performs list-decodable mean estimation. Since any $\mathcal{N}(\mu_i, I)$ might be the inlier distribution (and the remaining ones will define Q), the list-decodable mean estimator will output a single list which contains approximation to *all means* μ_i. With some postprocessing **DD:** add citation, this can be used to cluster all samples. Curiously, such methods can be made robust to small adversarial noise. **DD:** add more for the transition.

1.4.3 *Mixture learning with corruptions*

Existing works on robust mixture learning such as [31, 32] consider the problem when the fraction of additive adversarial outliers is smaller than the weight of the smallest subgroup, i.e. $\varepsilon < w_{\text{low}}$. However, for large outlier proportions where $\varepsilon \geqslant w_{\text{low}}$, these algorithms are not guaranteed to recover small clusters with $w_i \leqslant \varepsilon$. In this case, outliers can form additional spurious clusters that are indistinguishable from small inlier groups. As a consequence, generating a list of size equal to the number of components would possibly lead to neglecting the means of small groups. In order to ensure that the output contains a precise estimate for each of the small group means, it is thus necessary for the estimation algorithm to provide a list whose size is strictly larger than the number of components. We call this paradigm *list-decodable mixture learning* (LD-ML), following the footsteps of a long line of work on list-decodable learning (see Sections 5.2 and 5.5).

Specifically, the main challenge in LD-ML is to provide a *short* list that contains good mean estimates for all inlier groups. We first note that there is a minimum list size the algorithm necessarily has to output to guarantee that all groups are recovered. For example, consider an outlier distribution

that includes several copies of the smallest inlier group distribution with means spread out throughout the domain. Since inlier groups are indisntinguishable from spurious outlier ones, the shortest list that includes means of all inlier groups must be of size at least $|L| \geqslant k + \frac{\varepsilon}{\min_i w_i}$. Here, $\frac{\varepsilon}{\min_i w_i}$ can be interpreted as the minimal list-size overhead that is necessary due to "caring" about groups with weight smaller than ε. The key question is hence how good the error guarantees of an LD-ML algorithm can be when the list size overhead stays close to $\frac{\varepsilon}{\min w_i}$, while being agnostic to w_i aside from the knowledge of w_{low}. Furthermore, we are interested in *computationally efficient* algorithms for LD-ML, especially when dealing with high-dimensional data.

DD: paragraph below rewrite, connect better to previous To the best of our knowledge, the only existing efficient algorithms that are guaranteed to recover inlier groups with weights $w_i \leqslant \varepsilon$ are *list-decodable mean estimation* (LD-ME) algorithms. LD-ME algorithms model the data as a mixture of one inlier and outlier distribution with weights $\alpha \leqslant 1/2$ and $1 - \alpha$ respectively. Provided with the weight parameter α, they output a list that contains an estimate close to the inlier mean with high probability. However, for the LD-ML setting, the inlier weights w_i are not known and we would have to use LD-ME algorithms with w_{low} as weight estimates for each group. This leads to suboptimal error in particular for large groups, that hence (somewhat counter intuitively) would have to "pay" for the explicit constraint to recover small groups. Furthermore, even if LD-ME were provided with w_i, by design it would treat inlier points from other components also as outliers, unnecessarily inflating the fraction of outliers to $1 - w_i$ instead of ε.

CONTRIBUTIONS In this paper, we propose an algorithm that (i) correctly estimates the weight of each component only given a lower bound and (ii) does not overestimate proportion of outliers when components are well-separated. In particular, we construct a meta-algorithm that uses mean estimation algorithms as base learners that are designed to deal with adversarial corruptions. This meta-algorithm inherits guarantees from the base learner and any improvement of the latter translates to better results for LD-ML. For example, if the base learner runs in polynomial time, so does our meta-algorithm. Our approach of using the output of weak base learners to achieve better performance is reminiscent of the *boosting* paradigm that is common in machine learning practice.

Our algorithm achieves significant improvements in error and list-size guarantees for multiple settings. For ease of comparison, we summarize

Type of inlier mixture	Best prior work	Ours	Inf.-theor. lower bound
Large groups	$\widetilde{O}(\varepsilon/w_i)$	$\widetilde{O}(\varepsilon/w_i)$	$\Omega(\varepsilon/w_i)$
Small groups	$O\left(\sqrt{\log \frac{1}{w_{\text{low}}}}\right)$	$O\left(\sqrt{\log \frac{\varepsilon+w_i}{w_i}}\right)$	$\Omega\left(\sqrt{\log \frac{\varepsilon+w_i}{w_i}}\right)$
Non-separated groups	$O\left(\sqrt{\log \frac{1}{w_{\text{low}}}}\right)$	$O\left(\sqrt{\log \frac{1}{w_i}}\right)$	$\Omega\left(\sqrt{\log \frac{1}{w_i}}\right)$

TABLE 1.1: For a mixture of Gaussian components $\mathcal{N}(\mu_i, I_d)$, we show upper and lower bounds for the **error of the i-component** given a output list L (of the respective algorithm) $\min_{\hat{\mu}\in L}\|\hat{\mu}-\mu_i\|$. When the error doesn't depend on i, all means have the same error guarantee irrespective of their weight. Note that depending on the type of inlier mixture, different methods in [31] are used as the 'best prior work': robust mixture learning for the first row and list-decodable mean estimation for the rest. 'Large groups' means that $\forall j : \varepsilon \leqslant w_j$, 'small groups' means $\exists j : \varepsilon \geqslant w_j$. In both cases, mixture components are assumed to be separated. For lower bounds, see [33, 34], and Prop. 5.3.5

error improvements for inlier Gaussian mixtures in Table 5.1. The main focus of our contributions is represented in the second row; that is the setting where outliers outnumber some inlier groups with weight $w_j \leqslant \varepsilon$ and the inlier components are *well-separated*, i.e., $\|\mu_i - \mu_j\| \gtrsim^2 \sqrt{\log \frac{1}{w_{\text{low}}}}$, where μ_i's are the inlier component means. As we mentioned before **DD:** is it mentioned before?, robust mixture learning algorithms, such as [32, 35], are not applicable here and the best error guarantees in prior work is achieved by an LD-ME algorithm, e.g. from [31]. While its error bounds are of order $O(\sqrt{\log \frac{1}{w_{\text{low}}}})$ for a list size of $O(\frac{1}{w_{\text{low}}})$, our approach guarantees error $O(\sqrt{\log \frac{\varepsilon}{w_i}})$ for a list size of $k + O(\frac{\varepsilon}{w_{\text{low}}})$. Remarkably, we obtain the same error guarantees as if an oracle would run LD-ME on each inlier group *with the correct weight* w_i separately (with outliers). Hence, the only cost for recovering small groups is the increased list-size overhead of order $O(\frac{\varepsilon}{w_{\text{low}}})$. Further, a sub-routine in our meta-algorithm also obtains novel guarantees under *no* separation assumption, as shown in the third row of Table 5.1.

2 **DD:** remove? We adopt the following standard notation: $f \lesssim g$, $f = O(g)$, and $g = \Omega(f)$ mean that $f \leqslant Cg$ for some universal constant $C > 0$. $\widetilde{O}$-notation hides polylogarithmic terms.

This algorithm achieves the same error guarantees for similar list size as a base learner that knows the correct weights of the inlier components.

Based on a reduction argument from LD-ME to LD-ML, we also provide information-theoretic (IT) lower bounds for LD-ML. If the LD-ME base learners achieve the IT lower bound (possible for inlier Gaussian mixtures), so does our LD-ML algorithm. In synthetic experiments, we implement our meta-algorithm with the LD-ME base learner from [36] and show clear improvements compared to the only prior method with guarantees, while being comparable or better than popular clustering methods such as k-means and DBSCAN for various attack models.

1.5 NOTATION

SET THEORY For integers $k \in \mathbb{N}$, we write[3] $[k] := \{1, ..., k\}$. We let $\mathbb{N}$ and $\mathbb{R}$ denote the set of natural and real numbers respectively.

ASYMPTOTIC NOTATION We adopt the following standard notation: $f \lesssim g$, $f = O(g)$, and $g = \Omega(f)$ mean that $f(n) \leqslant Cg(n)$ for some universal constant $C > 0$ for all n large enough. We let $A \sim B$ or $A = \Theta(B)$ denote that $A \lesssim B \lesssim A$ for large enough n. For deterministic functions $h(n), w(n)$, we let $h \ll w, h \gg w$ denote that $h/w \to 0, w/h \to 0$ respectively, as $n \to \infty$.

LINEAR ALGEBRA We denote vectors, matrices by Roman letters $x, A \in \mathbb{R}^k, \mathbb{R}^{k\times k}$, respectively, for some $k \in \mathbb{N}$.
For a vector $x \in \mathbb{R}^n$ and a scalar $c \in \mathbb{R}$, we write $x \geqslant c$ (resp. $x \leqslant c$) if $x_i \geqslant c$ (resp. $x_i \leqslant c$) for all $i \in [n]$.
For a vector $x \in \mathbb{R}^n$, we denote

$$\|x\|_1 := \sum_{k\in[n]} |x_k|, \quad \|x\|_2 := \left(\sum_{k\in[n]} x_k^2\right)^{1/2}, \text{ and } \|x\|_\infty := \max_{k\in[n]} |x_k| \quad (1.5.1)$$

For a centered random vector $x \in \mathbb{R}^d$ we denote its *sub-Gaussian norm* as

$$\|x\|_{\psi_2} := \inf_{\sigma\geqslant 0}\{\mathbb{E}\exp^{\langle v,x\rangle} \leqslant \exp^{\frac{\|v\|^2\sigma^2}{2}} \ \forall\, v \in \mathbb{R}^d\}. \quad (1.5.2)$$

3 In Chapter 3, with abuse of notation, we write $[n] := \{0, \dots, n-1\}$. Furthermore, we index there vectors and matrices by $[n]$.

For square matrices $A \in \mathbb{R}^{n\times n}$ we denote the averaged trace by $\langle A\rangle := n^{-1}\operatorname{Tr} A$, and for rectangular matrices $A \in \mathbb{R}^{n\times m}$ we denote the Frobenius norm by $\|A\|_F^2 := \sum_{ij}|a_{ij}|^2$, and the operator norm by $\|A\|$.

PROBABILITY We use $\Pr, \mathbb{E}$, and Var to denote probability, expectation, and variance, respectively. For possibly random functions $f(n), g(n)$, we let $\{f \lesssim g\}$ denote a sequence of events $\{f(n) \leqslant Cg(n)\}$ for some constant $C > 0$ independent of n. Consequently, $\Pr(f \lesssim g)$ is viewed as a function of n. We say that a sequence of events $\{A_n\}$ holds *with high probability* (w.h.p.) with respect to a probability measure Pr if there exists a constant $c > 0$, independent of n, such that $\Pr(A_n) \geqslant 1 - n^{-c}$, for large enough values of n.

For families of non-negative random variables $X(n), Y(n)$ we say that X is *stochastically dominated* by Y, and write $X \prec Y$, if for all ε, D it holds that $\Pr(X(n) \geqslant n^{\varepsilon}Y(n)) \leqslant n^{-D}$ for n sufficiently large.

GRAPH THEORY For $n \in \mathbb{N}$, we denote by $G = (V, E)$ a graph with vertex set $V = [n]$ and edge set $E \subseteq (V \times V) \setminus \{(k,k) \text{ for } k \in V\}$. For a graph $G = (V, E)$ we define its complement $\overline{G} = (V, E')$, where $E' = \{(u, v) \text{ s.t. } u \neq v \text{ and } (u, v) \notin E\}$.

2

GREEDY HEURISTICS AND LINEAR RELAXATIONS FOR THE RANDOM HITTING SET PROBLEM

The forthcoming results are valid under the conditions listed below, which will be assumed to hold throughout.

Assumption 2.0.1. We assume that

1. Each element $j \in \mathcal{X}$ is assigned to any subset $\mathcal{S}_i$, $i \in [m]$ with probability $p \equiv p(n)$, independently. That is, $A \in \{0,1\}^{m \times n}$ is such that $A_{ij} \overset{\text{iid}}{\sim} \text{Bernoulli}(p)$;
2. n is intended to be large but finite;
3. $m \equiv m(n) = \text{poly}(n)$, i.e. $\exists c, C > 0$, such that $cn^c \leqslant m \leqslant Cn^C$ for n large enough;
4. There exist $\delta \in (0,1)$, such that $p \equiv p(n)$ satisfies $1/n^\delta \leqslant p \leqslant 1/2$, for all n large enough.

Note that in Assumption 1.1.2.3, the upper bound is chosen to avoid trivial solutions w.h.p. which arise, for example, in the setting where the number of sets grows exponentially in the cardinality of $\mathcal{X}$. In addition, Assumption 1.1.2.4 is by no means restrictive, since one can show that for $m = \text{poly}(n)$ and $np \ll \log n$, we have that A contains an all-zero row w.h.p., yielding an infeasible solution for `IP`. The requirement $p \leqslant 1/2$ is chosen for technical convenience and can be relaxed to any constant p, encompassing the regime in [2].

Our contributions stem from the study of the size of the inclusion sets $I_j := \{i \in [m] \,:\, j \in \mathcal{S}_i\}$, for $j \in [n]$, which in the `MVCH` formulation of the problem at hand correspond to the set of hyperedges incident to any given vertex. The key quantity under study is the average inclusion set size, that is $\mathbb{E}\,|I_j| = mp$, for all j, under the present distributional assumptions. This quantity exhibits two separate regimes of interest, referred to as the *sparse*, $mp \ll \log n$, and *dense*, $mp \gg \log n$, regimes. These, in turn, determine the size of the maximum inclusion set, or maximum degree, $d_{\texttt{max}} := \max_{j \in [n]} |I_j|$. We characterize the integrality gap behaviour up

to multiplicative constants and analyse Lovász's `Greedy` algorithm [3] in these two regimes w.h.p as $n \to \infty$. We do this by proving the success of a simple greedy heuristic, the `BlockGreedy` algorithm (Algorithm 3). Throughout, we use the notation $\mathrm{val}_{\mathtt{Gr}}$, $\mathrm{val}_{\mathtt{Alg}}$ to denote the size of the hitting set returned by `Greedy` and `BlockGreedy` respectively. Below we provide an informal description of the main results which hold with high probability, where $A(n) \sim B(n)$ denotes that $cA(n) \leqslant B(n) \leqslant CA(n)$ for large enough n and for some constants $c, C > 0$:

2.1 MAIN CONTRIBUTIONS

The results above are also depicted in Figure 1.1, and the formal statements are given in Theorem 2.5.3 and Theorem 2.5.4. The rest of the chapter is organized as follows. In Section 2.2, we present relevant notation. In Section 2.3, we outline and discuss related literature. In Section 2.4, we prove a number of preliminary results that will be instrumental in developing the core arguments. Subsequently, in Section 2.5, we delve into the algorithmic aspects of the problem at hand by first providing guarantees for a simple algorithm, `BlockGreedy`. We then analyse `Greedy` by means of a reduction. We conclude in Section 4.4 by summarizing the results and offering indications for future work. We defer the proofs of more technical results to the appendix, in order to streamline the presentation for the reader's convenience.

2.2 NOTATION AND CONVENTIONS

For integers $k \in \mathbb{N}$, we write $[k] := \{1, ..., k\}$. We denote vectors, matrices by bold-faced Roman letters $x, A \in \mathbb{R}^k, \mathbb{R}^{k\times k}$, respectively, for some $k \in \mathbb{N}$. Define the *inclusion set* of an element, or node, $j \in [n]$ as $I_j = \{i \in [m] : j \in \mathcal{S}_i\}$. We denote the ℓ_1 norm of the j-th column of A by X_j, $j \in [n]$, noting that $X_j = |I_j|$ and $X_1, \dots X_n \overset{\text{iid}}{\sim} \text{Binomial}(m, p)$. In addition, we let $d_{\max} \equiv d_{\max}(X_1, \dots, X_n) := \max_{i\in[n]} X_i$. We use $\mathbb{E}$, Var to denote expectation and variance, respectively. By $\lesssim, \gtrsim$ we denote inequalities up to multiplicative constants. We let $A \sim B$ denote that $A \lesssim B \lesssim A$ for large enough n. We let log denote the natural logarithm. For possibly random functions $f(n), g(n)$, we let $\{f \lesssim g\}$ denote a sequence of events $\{f(n) \leqslant Ag(n)\}$ for some constant $A > 0$ independent of n. Consequently, $\Pr(f \lesssim g)$ is viewed as a function of n. For deterministic functions $h(n), w(n)$, we let

$h \ll w, h \gg w$ denote that $h/w \to 0, w/h \to 0$ respectively, as $n \to \infty$. The notation for other inequalities is defined analogously. We say that a sequence of events $\{A_n\}$ holds *with high probability* (w.h.p.) with respect to a probability measure Pr if there exists a constant $c > 0$, independent of n, such that $\Pr(A_n) \geqslant 1 - n^{-c}$, for large enough values of n.

2.3 RELATED WORK

worst-case analysis of greedy Perhaps the most well-known algorithm for solving `Hitting Set`, or equivalently `MVCH`, is the greedy algorithm of Lovász [3], with runtime complexity $O(mn^2)$. This algorithm, which constructs a cover by sequentially adding elements of the ground set which hit the largest number of remaining subsets, was initially studied by Lovász [3] and Johnson [37] independently, for deterministic hypergraphs. Lovász analyses the greedy algorithm to obtain an upper bound on the `Hitting Set` integrality gap of $1 + \log d_{\max}$. Slavik [4] developed the tightest known approximation lower bound for `Greedy`, constructing an instance where `Greedy` finds coverings at least $\log m$ times as large as the minimum one. Importantly, Feige [38] proved that an approximation ratio of $(1 - \varepsilon) \log m$ is not achievable in polynomial time for any $\varepsilon > 0$ unless $NP \subset TIME[n^{O(\log \log n)}]$, certifying `Greedy` as the best possible polynomial-time approximation algorithm for set cover in the worst-case.

random hitting set Little is known about the typical performance of polynomial-time algorithms on random instances of `Hitting Set`. Closing this gap is important from a theoretical standpoint and for applications in combinatorial inference. A prime example of this is found in *group testing*, a classical inference problem where one aims to identify a small subset of defective items within a large population by conducting the smallest number of pooled tests, with applications ranging from the analysis of communication protocols [39] to DNA sequencing [40] and search problems [41]. In [2], Iliopulos and Zadik consider the smallest hitting set as an estimator in the setting of the group testing problem, referring to it as the *Smallest Satisfying Set* estimator. In particular, they provide extensive empirical evidence supporting the claim that the class of instances of the random hitting set problem induced by non-adaptive group testing is tractably solvable by computers.

insights from statistical physics The analysis of a random instance of `Hitting Set` appears in the work of Mézard and Tarzia and relies on nonrigorous techniques from statistical physics [42]. This work considers regular uniform hypergraphs, where the degree of vertices and the size of edges are fixed and assumed to be constant. Depending on these values, they evidence sharp transitions between three different phases, the so-called replica symmetry, 1-replica symmetry breaking, and full replica symmetry breaking phases, which characterize the complexity of the optimization landscape for this problem in the average case setting.

fixed p regime Another instance was studied by Telelis and Zissimopoulos [43] in the setting of random Bernoulli hypergraphs, where elements belong to subsets independently with *fixed* probability $p \in (0, 1)$. Their analysis concerns the asymptotic regime where the size n of the ground set scales to infinity. In this setting, they study the average-case performance of a simple deterministic algorithm which approximates random `Hitting Set` within an *additive* error term at most $o(\log m)$ almost everywhere. This gives an improvement over Lovász's argument in [3] which provides a multiplicative bound. However, the analysis in [43] does not capture the case of sparse hypergraphs, i.e., when $p \to 0$ as $n \to \infty$. The analysis in [43] also does not prove guarantees for the `Greedy` algorithm in the chosen parameter regime.

related problem formulations We bring to the reader's attention a more recent line of work [44, 45], where the authors obtain bounds on (additive) integrality gaps between the value of a random integer program $\max c^T x, Ax \leqslant b, x \in \{0,1\}^n$ with m constraints and that of its linear programming relaxation for a wide range of distributions on (A, b, c), holding w.h.p. as $n \to \infty$. These include the case where the entries of A are uniformly distributed on an integer interval consisting of at least three elements and where the columns of A are distributed according to an isotropic logconcave distribution. However, these fail to capture the setting where A is sparse with entries in $\{0, 1\}$, which is of interest for `Hitting Set`.

2.4 PRELIMINARY BOUNDS

In this section, we outline preliminary bounds on $\mathrm{val}_{\mathtt{LP}}, \mathrm{val}_{\mathtt{IP}}, d_{\mathtt{max}}$ which will prove crucial to analysing `IPGAP` and `Greedy`. We begin by characterizing the value of the linear program:

Lemma 2.4.1. *There exists $c > 0$, independent of n, such that with probability at least $1 - \exp(cn^{1-\delta})$, we have that*

$$\frac{m}{d_{\texttt{max}}} \leqslant val_{LP} \lesssim \frac{1}{p}.$$

The proof is included in Appendix A.1, and follows from a maximum argument and a standard Chernoff bound. We note that the proof also implies $\Pr(\texttt{IP} \text{ is feasible}) \geqslant 1 - \exp\left(-cn^{1-\delta}\right)$. Although Lemma 2.4.1 readily yields $\text{val}_{\texttt{IP}} \geqslant m/d_{\texttt{max}}$, we highlight that this lower bound is not tight whenever $mp \gg \log n$. Indeed, we apply the first moment method to obtain a tighter lower bound on $\text{val}_{\texttt{IP}}$ in this regime:

Lemma 2.4.2. *Let $mp \gg \log n$. For any $D \geqslant 1$ and n large enough, with probability at least $1 - n^{-D}$ we have that*

$$\frac{1}{p} \log\left(\frac{mp}{\log n}\right) \lesssim val_{\texttt{IP}}$$

The proof of Lemma 2.4.2 is provided in Appendix A.1. Lemmas 2.4.1 and 2.4.2 come short of providing a full characterization of `IPGAP`, namely lacking an upper bound on $\text{val}_{\texttt{IP}}$. In this light, we turn our attention to the `Greedy` algorithm, and utilize it to construct a feasible integral solution and hence an upper bound on the value of `IP`. The analysis of `Greedy` crucially relies on characterizing the maximum inclusion set size, $d_{\texttt{max}} := \max_{j\in[n]} |I_j|$. The following lemma offers such a characterization in expectation, and evidences a key difference between the sparse and dense regimes of our problem:

Lemma 2.4.3 (Maximum of Binomials)**.** *Let $X_1, \ldots, X_n \overset{\text{iid}}{\sim} Bin(m, p)$. Under the conditions in Assumption 2.0.1, it holds that*

$$\mathbb{E} d_{\texttt{max}} = \mathbb{E} \max_{i\in[n]} X_i \sim \begin{cases} \frac{\log n}{\log(\log n/mp)} & \text{, if } mp \ll \log n, \\ mp & \text{, if } mp \gtrsim \log n. \end{cases}$$

The proof of Lemma 2.4.3 is provided in Appendix A.1, and involves a straight forward application of Markov's and Jensen's inequalities. Lemma 2.4.3 indicates a sharp transition between two regimes: the sparse regime $mp \ll \log n$, where binomial random variables are known to be well approximated by Poisson random variables, and the dense regime $mp \gg \log n$, where binomial random variables are known to be well approximated by

Gaussian random variables. Importantly, in the sparse (Poisson-like) regime, the expected maximum of binomial random variables exceeds their individual expectations: $\mathbb{E}X_1 \ll \mathbb{E}d_{\texttt{max}}$. Meanwhile in the dense (Gaussian-like) regime, the expected maximum and individual expectations are asymptotically equivalent up to multiplicative constants: $\mathbb{E}X_1 \sim \mathbb{E}d_{\texttt{max}}$. This fine-grained characterization of the maxima of binomial random variables will prove essential to analysing the behaviour of `BlockGreedy` in Section 2.5. Finally, we characterize the asymptotic behaviour of $d_{\texttt{max}}$ and prove that $d_{\texttt{max}} \lesssim \mathbb{E}d_{\texttt{max}}$ with high probability. Whilst this one sided result suffices for the forthcoming analysis, we expect a matching lower bound to hold as well. Additional insights into the concentration of $d_{\texttt{max}}$ may be found in Lemmas A.1.5, A.1.6, in Appendix A.1.

Lemma 2.4.4. *Let $X_1, \dots, X_n \overset{\text{iid}}{\sim} Bin(m, p)$. Then, there exist constants $c, \tilde{c} > 0$, independent of n, such that*

$$\Pr\left(\max_{i\in[n]} X_i \geqslant c \cdot \mathbb{E}\max_{i\in[n]} X_i\right) \leqslant \frac{1}{n^{\tilde{c}}}.$$

The proof of Lemma 2.4.4 is provided in Appendix A.1.

2.5 ALGORITHMIC SOLUTIONS

2.5.1 *Challenges of `Greedy` analysis and `BlockGreedy` algorithm*

The aim of the present section is to conduct a rigorous analysis of the standard `Greedy` algorithm for the `Hitting Set` problem, within the prescribed Bernoulli random setting. In particular, we show that this routine succeeds at constructing hitting sets of optimal size w.h.p., as in the results of Section 2.4, up to multiplicative constants. This is done by first analysing a variation of the greedy heuristic, and subsequently proceeding by a reduction argument.

Let v_t be the element which is picked at the t-th step of `BlockGreedy`, f_t be the number of new subsets that are hit by v_t[1], and $F_t := \sum_{i=1}^{t} f_i$ be the total number of subsets which are hit after t steps. In order to analyse how many elements `BlockGreedy` has picked, we will consider the sequence $f_1, f_2, \dots, f_s$, with $F_t := \sum_{i=1}^{t} f_i$, such that the following holds:

1 It may happen that v_t hits *more* than f_t new subsets. In this case, we still only assume that exactly f_t are covered, and several extra sets will be covered multiple times in subsequent rounds. This overcounting simplifies the analysis and does not result in suboptimal solution.

Algorithm 3 `BlockGreedy`

Let $\mathcal{B}_t \subset \{I_1, ..., I_n\}$ denote the t-th block, i.e. inclusion sets that *become* available at step t.
$\mathcal{I} \leftarrow \varnothing$
$U \leftarrow [m]$
$t \leftarrow 0$
while $|U| > 0$ and $\mathcal{B}_t \neq \varnothing$ **do**
    $\mathcal{I} \leftarrow \mathcal{I} \cup \mathcal{B}_t$ ▷ Adding elements from the new block
    $P \leftarrow \operatorname{argmax}_{I \in \mathcal{I}} |I \cap U|$ ▷ Greedy step
    $\mathcal{I} \leftarrow \mathcal{I} \setminus \{P\}$
    $U \leftarrow U \setminus P$
    $t \leftarrow t + 1$
$\mathrm{val}_{\mathtt{Alg}} \leftarrow t$
if $|U| > 0$ **then** cover the rest of U with a trivial algorithm, $\mathrm{val}_{\mathtt{Alg}} \leftarrow \mathrm{val}_{\mathtt{Alg}} + |U|$
return $\mathrm{val}_{\mathtt{Alg}}$.

1. $F_s = m$;
2. if $mp \lesssim \log n$, then $s \lesssim \mathrm{val}_{\mathtt{LP}}$, otherwise, $s \lesssim \mathrm{val}_{\mathtt{IP}}$.

The first property ensures that `BlockGreedy` picks at most s elements, and the second property gives optimal bounds on s. One way to guarantee that `BlockGreedy` succeeds is to prove that among the choices of `BlockGreedy` at each step t, there was an element $\tilde{v}_t$ which hits at least f_t new subsets w.h.p. We will prove that it is enough to look for $\tilde{v}_t$ in the new block of columns $\mathcal{B}_t$, which are added at step t. Note that unless $F_t = m$, we have that $f_t \geqslant 1$, since each subset is hit by at least one element w.h.p.. Therefore, it will be enough to find a sequence $\{f_1, f_2, \ldots, f_v\}$ such that $F_v \geqslant m - v$, since it implies $F_{2v} = m$. This allows us to reduce the problem of proving the effectiveness of `BlockGreedy` to a key technical lemma. This lemma assumes that before step t, exactly F_{t-1} subsets are hit, and bounds from below the probability that some vertex in the new block will hit at least f_t new subsets. This boils down to computing $\Pr(\mathrm{Bin}(m - F_{t-1}, p) \geqslant f_t)$.

Lemma 2.5.1 (Informal, see Theorem A.2.3)**.** *Let $\varepsilon > 0$ and $mp \lesssim \log n$. For some constants $\tau > 0$, $1 < \alpha < \beta$, and for $t \in \mathbb{N}$, let:*

$$f_t = \left\lceil (\alpha/\beta)^k \tau \mathbb{E} d_{\mathtt{max}} \right\rceil \quad \textit{where } k \textit{ is such that} \quad \beta^{-k-1} m < m - F_{t-1} \leqslant \beta^{-k} m;$$

Then there exists a choice of τ, α, β and K, such that $F_K \geqslant m - K$ and $K \sim val_{LP}$. Furthermore, for this sequence f_t (which depends on ε), for any $t \leqslant K$,

$$\Pr(\mathrm{Bin}(m - F_{t-1}, p) \geqslant f_t) \geqslant n^{-\varepsilon}. \tag{2.5.1}$$

Note that the implicit constants in the statements $K \sim val_{LP}$ depend on ε.

This lemma highlights the crucial dependency of the problem on the relationship between the average degree, mp, and $\log n$. For clarity of exposition, we only state the lemma for the case $mp \lesssim \log n$ and refer to the Theorem A.2.3 in the Appendix for the full version and corresponding proof. Here we comment on the intuition behind the proof.

When $mp \lesssim \log n$, we need to carefully track how the maximum degree changes. We look for an element which (i) covers a large number of subsets, i.e., close to the expected maximum number, $\mathbb{E}d_{\mathtt{max}}$ and (ii) can be found with large enough probability. The second property is important for the reduction to the standard `Greedy` algorithm, whose direct analysis presents substantial difficulties, and is done later in this section. The quantity $\mathbb{E}d_{\mathtt{max}}$ is sensitive to mp whenever the latter is close to $\log n$. Hence, we need to adjust which element we look for accordingly. This is done by setting $f_t = \left\lceil (\alpha/\beta)^k \tau \mathbb{E}d_{\mathtt{max}} \right\rceil$ and increasing the parameter k as the number of remaining rows, $m - F_t$, decreases.

For example, consider the case $mp = \log n$. First, we can only pick a random element, since it will be as good (up to a multiplicative constant) as the maximal element. However, during the execution of the algorithm, the problem becomes more sparse, and if we continue to pick random elements, we will construct a suboptimal solution. Therefore, we gradually increase how much the newly picked element will cover, *with respect to a random element*. This corresponds to the transition between Gaussian-like and Poisson-like behaviour of $\mathrm{Bin}(m - F_{t-1}, p)$.

It is now straightforward to prove the following theorem, which makes rigorous the statements in Section 2.1.

Theorem 2.5.2. *Under Assumption 2.0.1, we have that*

$$\begin{aligned}
&(i) \text{ if } mp \lesssim \log n \text{ then, for any } \varepsilon > 0 \text{ and } n \text{ large enough,} \\
&\qquad \Pr\left(val_{Alg} \lesssim \frac{m}{\mathbb{E}d_{\mathtt{max}}} \right) \geqslant 1 - \exp\left(-n^{1-\delta-\varepsilon}\right); \\
&(ii) \text{ if } mp \gg \log n, \text{ then, for any } \varepsilon > 0 \text{ and } n \text{ large enough,} \\
&\qquad \Pr\left(val_{Alg} \lesssim \frac{1}{p} \log\left(\frac{mp}{\log n} \right) \right) \geqslant 1 - \exp\left(-n^{1-\delta-\varepsilon}\right).
\end{aligned} \tag{2.5.2}$$

Note that if $mp \gtrsim n^{\gamma}$ for some $\gamma > 0$, then $\log \frac{mp}{\log n} \sim \log n$, and the bound in (ii) can be simplified.

Proof. The main idea of the proof is to analyse the distribution of the columns that are added at each step t. These columns are independent, and for each newly added column, the number of additional subsets which it covers is distributed according to $\mathrm{Bin}(m - F_{t-1}, p)$, where F_{t-1} is the number of subsets which are already covered. Lemma A.2.3 (see Theorem 2.5.1 above for an informal version) allows us to lower bound F_t, and we show now that we can do this with high probability.
Fix $\varepsilon > 0$ and let $\varepsilon' := \varepsilon/4$. Let $f_1, f_2, \ldots$ be the sequence from Lemma A.2.3 for ε' and let K be the value for which (A.2.2) is satisfied, i.e. $F_K \geqslant m - K$. Notice that $K \leqslant C \max\left\{\frac{m}{\mathbb{E}\, d_{\mathtt{max}}}, \frac{1}{p}\log(\frac{mp}{\log n})\right\}$ for some constant $C > 0$, for n large enough. We uniformly at random split n elements (columns) into K groups of size n/K each (assuming without loss of generality that K divides n, otherwise we consider groups of size $\lfloor n/K \rfloor$), so that $\mathcal{B}_t$ yields a new set of n/K elements at each iteration $t \leqslant K$ and $\mathcal{B}_t = \varnothing$ for $t > K$. We say that the algorithm fails at step t if before step t, at least F_{t-1} subsets are covered, but after step t less than F_t sets are covered. Using that, for n large enough, (i) columns in each newly added block are independent, (ii) $\Pr\left(\mathrm{Bin}(m - F_{t-1}, p) \geqslant f_t\right) \geqslant n^{-\varepsilon'}$, and (iii) $n/K \geqslant n^{1-\delta-\varepsilon'}$, we get

$$\begin{aligned}
\Pr\left(\texttt{BlockGreedy} \text{ fails at step t}\right) &\overset{(i)}{\leqslant} \left(\Pr\left(\mathrm{Bin}(m - F_{t-1}, p) < f_t\right)\right)^{n/K} \\
&\overset{(ii)}{\leqslant} \left(1 - n^{-\varepsilon'}\right)^{n/K} \\
&\overset{(iii)}{\leqslant} \exp\left(-n^{1-\delta-2\varepsilon'}\right).
\end{aligned}$$

We then proceed by applying a union bound to obtain the result,

$$\begin{aligned}
&\Pr\left(\texttt{BlockGreedy} \text{ fails during first } K \text{ steps}\right) \\
&\quad\leqslant \sum_{t=1}^{K} \Pr\left(\texttt{BlockGreedy} \text{ fails at step t}\right) \\
&\quad\leqslant K \cdot \exp\left(-n^{1-\delta-2\varepsilon'}\right) \\
&\quad\leqslant \exp\left(-n^{1-\delta-3\varepsilon'}\right),
\end{aligned}$$

where the second inequality holds since, by definition, the algorithm runs for K iterations, and the third one holds for n large enough. We proved

that BlockGreedy succeeds in finding at most K elements such that at most $m - F_K$ sets remain uncovered. Since by construction, $m - F_K \leqslant K$, we can cover the remaining rows trivially using that IP is feasible by Lemma A.1.2 with high probability, which proves that

$$\Pr\left(\mathrm{val}_{\mathtt{Alg}} \leqslant 2K\right) \geqslant 1 - \exp\left(-n^{1-\delta-4\varepsilon'}\right) = 1 - \exp\left(-n^{1-\delta-\varepsilon}\right),$$

for n large enough. Recalling that $K \lesssim \mathrm{val}_{\mathtt{LP}}$ for $mp \lesssim \log n$, and that $K \lesssim \mathrm{val}_{\mathtt{IP}}$ for $mp \gg \log n$, finishes the proof. □

Corollary 2.5.3. *Under Theorem 2.0.1, we have that for any $D > 0$,*

$$\begin{aligned}
&(i) \text{ for any } n \text{ large enough,} \\
&\qquad \Pr\left(val_{Alg} \sim val_{IP}\right) \geqslant 1 - n^{-D}; \\
&(ii) \text{ if } mp \lesssim \log n, \text{ then, for any } n \text{ large enough,} \\
&\qquad \Pr\left(\mathit{IPGAP} \sim 1\right) \geqslant 1 - n^{-D}; \\
&(iii) \text{ if } mp \gg \log n, \text{ then, for any } n \text{ large enough,} \\
&\qquad \Pr\left(\mathit{IPGAP} \sim \log\left(\frac{mp}{\log n}\right)\right) \geqslant 1 - n^{-D}.
\end{aligned} \tag{2.5.3}$$

Proof. Proof follows from Theorem 2.4.1, Theorem 2.4.2, and Theorem 2.5.2. □

2.5.2 *Reduction from* BlockGreedy *to* Greedy

With the above results at hand, we now proceed to analyse the Greedy algorithm by means of a suitable reduction. Recall that we denote outputs of BlockGreedy and Greedy as $\mathrm{val}_{\mathtt{Alg}}$ and $\mathrm{val}_{\mathtt{Gr}}$ respectively.

Theorem 2.5.4. *Under Assumption 2.0.1 with $\delta < 1/2$, we have that, for n large enough,*

$$\Pr\left(val_{Gr} \sim val_{IP}\right) \geqslant 1 - \exp\left(-\sqrt{n}\right).$$

Proof. We use Theorem 2.5.2 with $\varepsilon = 1/8 - \delta/4$, and let $K, \mathcal{B}_t$ be as defined in the proof of Theorem 2.5.2. We have that, for n large enough,

$$\Pr\left(\texttt{BlockGreedy fails at any step}\right) \leqslant \exp\left(-n^{\Delta}\right),$$

where $\Delta := 3/4 - \delta/2 > 1/2$.

Given a matrix A, consider running the above definition of `BlockGreedy` for $J := \exp(\sqrt{n})$ times, each time reshuffling the columns. In what follows, we address `BlockGreedy` and `Greedy` defined with the same tie-breaking strategy when it comes to a number of elements hitting the same number of sets, i.e., selecting the left-most column in the associated matrix A. Both $\mathrm{val}_{\texttt{Alg}}$ and $\mathrm{val}_{\texttt{Gr}}$ are random variables, but conditioned on A, $\mathrm{val}_{\texttt{Gr}}$ is deterministic, while $\mathrm{val}_{\texttt{Alg}}$ still depends on the randomness of separating columns into blocks. Using the union bound, we have that

$$\begin{aligned}\Pr\left(\mathrm{val}_{\texttt{Gr}} > 2K\right) \leqslant\ & \Pr\left(\exists \text{ a failed copy of } \texttt{BlockGreedy}\right) \\ & + \Pr\left(\mathrm{val}_{\texttt{Alg}} < \mathrm{val}_{\texttt{Gr}} \text{ over all } J \text{ copies}\right).\end{aligned} \tag{2.5.4}$$

Applying the union bound again, we can upper bound the first term in (2.5.4):

$$\Pr\left(\exists \text{ a failed copy of } \texttt{BlockGreedy}\right) \leqslant J \exp\left(-n^{\Delta}\right) = \exp\left(-n^{\Delta} + n^{1/2}\right). \tag{2.5.5}$$

Now we focus on the second term in (2.5.4). Let $v_1, v_2, \ldots, v_g$ be the ordered sequence of elements picked by `Greedy`. Let $M_t := \{v_1 \in \mathcal{B}_1, v_2 \in \mathcal{B}_1 \cup \mathcal{B}_2, \ldots, v_t \in \mathcal{B}_1 \cup \ldots \cup \mathcal{B}_t\}$. The event $\{\mathrm{val}_{\texttt{Alg}} \geqslant \mathrm{val}_{\texttt{Gr}}\}$ contains the event M_g, since in this case `BlockGreedy` will necessarily pick exactly the same columns $v_1, v_2, \ldots, v_g$. Given that each reshuffling of the columns generates a uniform distribution of $\mathcal{B}_i$'s over possible partitions of n columns, we get that

$$\Pr\left(M_g\right) = \prod_{t=1}^{g} \Pr\left(v_t \in \mathcal{B}_1 \cup \ldots \cup \mathcal{B}_t \mid M_{t-1}\right)$$

The t-th term in the product above is equal to

$$\Pr(v_t \in \mathcal{B}_1 \cup \ldots \cup \mathcal{B}_t \mid M_{t-1}) = \frac{t\frac{n}{K} - (t-1)}{n - (t-1)} \geqslant \frac{t}{K} - \frac{t-1}{n} \geqslant \frac{t}{2(K-1)},$$

where the last inequality holds for $n \geqslant 4K$ (recall that $n \gg K$). Since $M_g \subset \{\mathrm{val}_{\texttt{Alg}} \geqslant \mathrm{val}_{\texttt{Gr}}\}$, we can lower bound the probability of the latter event as follows (note that when $g < K$ there will be less terms in the product, hence, $\Pr(M_g)$ will be even larger),

$$\begin{aligned}&\Pr\left(\mathrm{val}_{\texttt{Alg}} \geqslant \mathrm{val}_{\texttt{Gr}} \text{ for 1 copy}\right) \geqslant \Pr(M_g) \\ &\quad\geqslant \prod_{t=1}^{K-1} \Pr(v_t \in \mathcal{B}_1 \cup \ldots \cup \mathcal{B}_t \mid M_{t-1}) \geqslant \prod_{t=1}^{K-1} \frac{t}{2(K-1)} \geqslant e^{-2K},\end{aligned}$$

where we used that $k! \geqslant (k/e)^k$ in the last inequality. Since $K \leqslant C \max\left\{\frac{m}{\mathbb{E}\, d_{\max}}, \frac{1}{p}\log(\frac{mp}{\log n})\right\}$ and $1/p \leqslant n^{\delta}$, there exists a constant $\tilde{C} > 0$ large enough, such that $K \leqslant \tilde{C} n^{\delta} \log n$. Therefore, using independence of the reshuffling between the copies, we can compute

$$\begin{aligned}\Pr\left(\mathrm{val}_{\mathtt{Alg}} < \mathrm{val}_{\mathtt{Gr}} \text{ over all } J \text{ copies}\right) &= \left(1 - \Pr\left(\mathrm{val}_{\mathtt{Alg}} \geqslant \mathrm{val}_{\mathtt{Gr}} \text{ for 1 copy}\right)\right)^{J} \\ &\leqslant (1 - e^{-2K})^{J} \\ &\leqslant \exp\left(-e^{\sqrt{n} - 2\tilde{C} n^{\delta} \log n}\right).\end{aligned} \tag{2.5.6}$$

Combining (2.5.4), (2.5.5) and (2.5.6), we showed that $\Pr\left(\mathrm{val}_{\mathtt{Gr}} > 2K\right) \leqslant \exp\left(-\sqrt{n}\right)$ for n large enough, which finishes the proof. □

Remark 2.5.5. *We note that the $\delta < 1/2$ condition in Theorem 2.5.4 is likely not optimal, and could be relaxed by reducing to* `BlockGreedy` *with more carefully chosen sets $\mathcal{B}_t$. In particular, the appropriate set sizes $|\mathcal{B}_t|$ may not be identical across $t \leqslant K$. The analysis becomes more technical in this case, and we highlight this as an interesting open direction.*

2.6 DISCUSSION AND OPEN QUESTIONS

Our work characterises multiplicative integrality gaps for the random hitting set problem. In this section, we discuss the intuition behind our main results, together with open questions and conjectures.

2.6.1 *Summary of our results and proof techniques.*

We identified that the nature of integrality gaps depends on the size of the inclusion set, also viewed as the sparsity of the underlying hypergraph. In particular, when the average degree of a vertex is small, i.e., when each element belongs to a small number of subsets, we proved that there exists only a constant gap between linear and integer program solutions, together with a simple algorithmic solution. The situation changes when the hypergraph becomes dense, where we show an increasing integrality gap. This separation stems mostly from the property of the binomial distribution, where the maximum of random variables grows identically to the expected value whenever the expected value is large, but is away from it if $mp \ll \log n$.

In our analysis of `BlockGreedy`, we track this change of behaviour using

a geometric series, which means that the further we are in the execution of the algorithm, the larger the ratio between the element we pick and the average element will be. This picture coincides exactly with how the binomial distribution will behave if we decrease the average degree: for large instances, it will look approximately as a Gaussian, but when the average degree is small, Poisson approximation starts to dominate, the right tail becomes heavier, and the difference between $d_{\texttt{max}}$ and mp increases. Our analysis tracks the transition between Gaussian and Poisson-like behavior.

2.6.2 *Multiplicative vs. additive integrality gaps*

Our result only concerns multiplicative gaps, but the constants in our analysis can be large. This might be a consequence of the generality of the studied problem. For example, if one focuses only on the case of constant p, which immediately implies a very dense instance in our characterization, [43] proves that a simple algorithm is optimal for approximating the integer program up to a small additive error. Proving similar upper bounds on the constant in more general cases is an interesting open problem. Based on numerical experiments, we formulate the following conjectures.

Conjecture 2.6.1 (Very sparse)**.** *For* $mp \ll 1$, $\frac{val_{Gr}}{val_{LP}} \to 1$.

Conjecture 2.6.2 (Sparse)**.** *For* $1 \lesssim mp \ll \log n$, $\frac{val_{Gr}}{val_{IP}} \to 1$, *and* $\frac{val_{IP}}{val_{LP}} \to C_1 \in (1, 1.5)$.

Conjecture 2.6.3 (Dense)**.** *For* $mp \gg \log n$, $\frac{val_{Gr}}{val_{IP}} \to C_2 \in (1, 1.5)$.

2.6.3 *Analysis of a linear program solution.*

One motivation for studying the gaps between the integer and linear programs together with the solutions of linear programs themselves is to construct a rounding scheme which converts a fractional solution to an integer one. We believe this is another interesting direction for future work. In particular, numerical experiments show that entries which have large value in the fractional solution have a strong tendency to correspond to elements that are picked for the integer solution. This supports the claim that a combination of the greedy and linear programming approach might be fruitful in efficiently solving `Hitting Set`. One approach for further study consists of first solving a linear program, initializing x with the largest

elements in the linear solution, and greedily covering the remaining subsets.

3

THE LOVÁSZ NUMBER OF RANDOM CIRCULANT GRAPHS

3.1 PRELIMINARIES

NOTATION For $n \in \mathbb{N}$, let $[n] := \{0, \dots, n-1\}$. We index vectors and matrices by $[n]$: for $x \in \mathbb{R}^n, x = (x_0, \dots, x_{n-1})$. We write $x \geqslant \mathbf{0}$ for entrywise positivity. For $n \in \mathbb{N}$, we denote by $G = (V, E)$ a graph with vertex set $V = [n]$ and edge set $E \subseteq (V \times V) \setminus \{(k,k) \text{ for } k \in V\}$. For a graph $G = (V, E)$ we define its complement $\overline{G} = (V, E')$, where $E' = \{(u, v) \text{ s.t. } u \neq v \text{ and } (u, v) \notin E\}$. We use the standard asymptotic notation, $O(\cdot), \Omega(\cdot)$, and $\Theta(\cdot)$ to describe the order of the growth of functions associated with the limit of the graph dimension n. For $x \in \mathbb{R}^n$, we denote $\|x\|_1 := \sum_{k=0}^{n-1} |x_k|$, $\|x\|_2 := \left(\sum_{k=0}^{n-1} x_k^2\right)^{1/2}$, and $\|x\|_\infty := \max_k |x_k|$.

DISCRETE FOURIER TRANSFORM Let $F \in \mathbb{C}^{n \times n}$ be the discrete Fourier transform matrix: $F_{jk} = \exp(-2\pi ijk/n)$ for $j, k \in [n]$. For $k \in [n]$, let f_k denote the k-th row of F. We associate a matrix $\widetilde{F} \in \mathbb{R}^{m \times n}$ to any RCG G consisting of subsampled rows of F.

Definition 3.1.1. For any RCG G, let $\widetilde{F} \equiv \widetilde{F}(G) \in \mathbb{C}^{m \times n}$ (with m the number of neighbors of 0 in G) be defined as a submatrix of F, including row f_k if $(0, k) \in E(G)$.

Definition 3.1.2. The *Lovász theta number* $\vartheta(G)$ is defined as the solution to the following SDP (J is the all-ones matrix),

$$\begin{aligned} \vartheta(G) := \max_{X \in \mathbb{R}^{n \times n}} \Big\{ \langle X, J \rangle, \text{ such that } X \succeq 0, \operatorname{Tr} X = 1, \\ X_{ij} = 0 \text{ for all } (i,j) \in E(G) \Big\}. \end{aligned} \tag{3.1.1}$$

Definition 3.1.3. A graph on n vertices is called *circulant* if there is an ordering of its vertices such that its adjacency matrix is circulant. Equivalently, a circulant graph is a Cayley graph of a cyclic group $\mathbb{Z}_n$.

This definition implies that a circulant graph is described by listing the neighbors of a single root vertex (say vertex 0), since $(i,j) \in E \iff (0, i-j) \in E$. In this text, we focus on *dense random* circulant graphs.

Definition 3.1.4. For odd n, a *dense random circulant graph (RCG)* is a random Cayley graph of a cyclic group $\mathbb{Z}_n$. It is obtained in the following way: uniformly sample $x \in \{0,1\}^m, m = \frac{n-1}{2}$, and define the first row of the adjacency matrix as

$$R = (0 \;\; x \;\; \overleftarrow{x}), \tag{3.1.2}$$

where $\overleftarrow{x}_i := x_{m-i-1}$. Circulate R to obtain the complete adjacency matrix.

For a circulant graph G we define a vector $g := Fb$, where $b \in \{\pm 1\}^n$ with $b_0 = 1$ and $b_k = 1$ if $(0,k)$ is not an edge, and -1 otherwise, for $1 \leqslant k \leqslant n-1$.

Definition 3.1.5 (Restricted isometry property)**.** A matrix $A \in \mathbb{C}^{q \times n}$ is said to satisfy (k,ε)-restricted isometry property, for $k \leqslant n$ and $\varepsilon \in (0,1)$, if for all k-sparse $x \in \mathbb{C}^n$ we have that

$$(1-\varepsilon)\|x\|_2^2 \leqslant \|Ax\|_2^2 \leqslant (1+\varepsilon)\|x\|_2^2. \tag{3.1.3}$$

3.2 PROOF OF MAIN THEOREM

Let G be a circulant graph. As noted in [11], for circulant graphs the SDP formulation of the Lovász number can be rewritten as the following linear program:

$$\begin{aligned} \vartheta(G) &= \max_{x \in \mathbb{R}^n} \sum_{i \in [n]} x_i, \\ \text{subject to } &\begin{cases} x_k = x_{n-k} \text{ for all } k \in [n] \setminus \{0\}, \\ x_0 = 1, Fx \geqslant \mathbf{0}, \\ x_k = 0 \text{ for all edges } (0,k), \end{cases} \end{aligned} \tag{3.2.1}$$

Table 3.1 shows four equivalent linear programs, arising from strong duality (see, e.g., [46]) and switching between 'time' and 'frequency' domains. For the latter, we perform the change of variables, $y := Fx$ and $t := Fz$ respectively.

All formulations share the same structure: the optimization objective is determinisitic, while the set of feasible solutions is random through the

'time' domain

Primal	**Dual**
$\max_{x\in\mathbb{R}^n} \sum_i x_i$	$\min_{z\in\mathbb{R}^n} 1 + \sum_i z_i$
s.t. $x_k = x_{n-k}$ for all $k \in [n]\setminus\{0\}$,	s.t. $z_k = z_{n-k}$ for all $k \in [n]\setminus\{0\}$,
$x_0 = 1,\ Fx \geqslant \mathbf{0}$,	$z \geqslant \mathbf{0}$,
$x_k = 0$ for all $(0,k) \in E(G)$.	$\langle z, f_k\rangle = -1$ for all $(0,k) \in E(\overline{G})$.

'frequency' domain

Primal	**Dual**
$\max_{y\in\mathbb{R}^n} ny_0$	$\min_{t\in\mathbb{R}^n} 1 + nt_0$
s.t. $y_k = y_{n-k}$ for all $k \in [n]\setminus\{0\}$,	s.t. $t_k = t_{n-k}$ for all $k \in [n]\setminus\{0\}$,
$\|y\|_1 = 1,\ y \geqslant \mathbf{0}$,	$Ft \geqslant \mathbf{0}$,
$\langle y, f_k\rangle = 0$ for all $(0,k) \in E(G)$.	$t_k = -1/n$ for all $(0,k) \in E(\overline{G})$.

TABLE 3.1: Four equivalent LPs for $\vartheta(G)$.

random circulant graph structure. The following proposition introduces randomness to the objective, which is a crucial part of our argument.

Lemma 3.2.1. *Let G be a dense RCG and $\widetilde{F}$ be a subsampled DFT matrix, see Theorem 3.1.1. Let $g := Fb \in \mathbb{R}^n$, for $b \in \{\pm 1\}^n$ with $b_k = 1$ if $(0,k)$ is not an edge and -1 otherwise. Then,*

$$\vartheta(G) = \max_{y\in\mathbb{R}^n} \langle y, g\rangle\,,$$
$$\text{subject to } \begin{cases} y_k = y_{n-k} \text{ for all } k \in [n]\setminus\{0\}, \\ \|y\|_1 = 1, y \geqslant \mathbf{0}, \\ y \in \ker \widetilde{F}, \end{cases} \tag{3.2.2}$$

Proof. We use the primal formulation in the frequency domain and observe that $ny_0 = \left\langle y, \sum_{k\in[n]} f_k \right\rangle$. Since feasible vectors y are orthogonal to $\widetilde{F}$, i.e., $y \in \ker \widetilde{F}$, after subtracting $2\sum_{(0,k)\in E(G)} \langle y, f_k\rangle$ from $\left\langle y, \sum_{k\in[n]} f_k \right\rangle$ we obtain

$$\langle y, \sum_{k\in[n]} f_k\rangle = \langle y, \sum_{(0,k)\notin E(G)} f_k - \sum_{(0,k)\in E(G)} f_k\rangle = \langle y, g\rangle. \tag{3.2.3}$$

□

By the definition of graph G, $b_0 = 1$, and $b_1, b_2, \ldots, b_{\frac{n-1}{2}} \overset{\text{iid}}{\sim} \text{Unif}\{-1, 1\}$. Since $\max_{jk}|F_{jk}| = 1$, we can bound $\|g\|_\infty$, leading to the following upper bound on ϑ.

Lemma 3.2.2. *Let G be a dense RCG. Then,*

$$\Pr(\vartheta(G) \leqslant 1 + 4\sqrt{n\log n}) \geqslant 1 - \frac{2}{n}. \tag{3.2.4}$$

Proof. We show that each entry of g is small with high probability. Indeed, for any $k \in [n]$,

$$\begin{aligned}\Pr(|g_k| > 1 + 4\sqrt{n\log n}) &= \Pr(|\langle f_k, b\rangle| > 1 + 4\sqrt{n\log n}) \\ &\leqslant \Pr\left(|\sum_{j=1}^{(n-1)/2} X_j| > 2\sqrt{n\log n}\right) \leqslant \frac{2}{n^2},\end{aligned} \tag{3.2.5}$$

where $X_j \coloneqq \Re(F_{kj})b_j \in [-1, 1]$, and the last step follows from Hoeffding's inequality (Theorem 3.4.1). Applying union bound over $k \in [n]$, we obtain

$$\Pr(\|g\|_\infty > 1 + 4\sqrt{n\log n}) \leqslant \frac{2}{n}. \tag{3.2.6}$$

Thus, on a complement event, for any feasible vector y of eq. (1.2.3), we can simply upper bound $\langle y, g\rangle \leqslant \|y\|_1\|g\|_\infty \leqslant 1 + 4\sqrt{n\log n}$, which finishes the proof. □

The upper bound in Theorem 1.2.1 would follow if we could show $\max_k g_k = O(\sqrt{n\log\log n})$ with high probability. However, this is too optimistic: since we expect that the coordinates of g behave like standard Gaussian random variables and are uncorrelated, we also expect that $\max_k g_k = \Theta(\sqrt{n\log n})$. Fortunately, as the next lemma shows, only a vanishing fraction of entries is of order at least $\sqrt{n\log\log n}$.

Lemma 3.2.3. *There exists a constant $C > 0$, such that for $\mathcal{I} := \{k \in [n] : |g_k| \geqslant C\sqrt{n \log\log n}\}$, it holds*

$$\Pr\left(|\mathcal{I}| \leqslant \frac{n}{\log^{10} n}\right) \geqslant 1 - \frac{1}{\log^{10} n}. \tag{3.2.7}$$

Proof. We express $|\mathcal{I}| = \sum_{k=0}^{n-1} Y_k$, where $Y_k = \mathbb{I}\{|g_k| \geqslant C\sqrt{n \log\log n}\}$. Using Hoeffding's inequality we obtain, for C large enough,

$$\mathbb{E}|\mathcal{I}| = \sum_{k=0}^{n-1} \Pr(|g_k| \geqslant C\sqrt{n \log\log n}) \leqslant \frac{n}{\log^{20} n}, \tag{3.2.8}$$

where the constant on the right hand side is absorbed into logarithm, and its power is chosen for the technical reasons. Plugging this bound into Markov's inequality we get

$$\Pr\left(|\mathcal{I}| \geqslant \frac{n}{\log^{10} n}\right) \leqslant \frac{1}{\log^{10} n}. \tag{3.2.9}$$

□

The constraint $y \in \ker \widetilde{F}$ was so far only used to change the objective function from ny_0 to $\langle y, g\rangle$. Next lemma highlights another important consequence of this constraint, namely, an upper bound on the $\|y\|_2$.

Lemma 3.2.4. *For large enough n, with probability at least $1 - \frac{1}{n}$ all $x \in \ker \widetilde{F}$ satisfy $\|x\|_2 \leqslant \frac{\log^2 n}{\sqrt{n}}\|x\|_1$.*

Proof. We adapt the existing results in the literature regarding the RIP of the subsampled Fourier basis.

Consider the following coupling: let $\hat{b} \in \{0,1\}^n$ with $\hat{b}_0 = 0$ and $\hat{b}_k \overset{\text{iid}}{\sim} \text{Ber}\left(\frac{\sqrt{2}-1}{\sqrt{2}}\right)$ for $k = 1, \ldots, n-1$. Let $\widetilde{b} \in \{0,1\}^n$ be defined as follows:

$$\widetilde{b}_k = \begin{cases} 0, & \text{for } k = 0, \\ 1, & \text{if } \hat{b}_k = 1 \text{ or } \hat{b}_{n-k} = 1, \text{ for } k \geqslant 1, \\ 0, & \text{otherwise.} \end{cases} \tag{3.2.10}$$

Note that (i) the distribution of $\widetilde{b}$ is the same as the distribution of the adjacency vector for the vertex 0 in the random circulant graph G and (ii)

$\widetilde{b}_i = 0$ implies $\hat{b}_i = 0$. Let $q := \sum_k \hat{b}_k$. Define $\widehat{F} \in \mathbb{C}^{q \times n}$ to be the matrix consisting of subsampled rows of F rescaled by $1/\sqrt{q}$, where the k-th row is included if and only if $\hat{b}_k = 1$.

To show that $\widehat{F}$ satisfies the RIP, we apply Theorem 3.4.2. To ensure its requirements, we condition on the following two events. First, since we do not include row 0 in our construction, we condition on the event that among the uniformly subsampled rows, row 0 is not present; this increases the probability of a bad event by at most a constant factor. Additionally, we condition on a high probability event that $q \geqslant \lceil n/4 \rceil$. Theorem 3.4.2 then implies that there exist constants $c > 0$ and $0 < \varepsilon < 1/3$, such that with probability at least $1 - 1/n$, $\widehat{F}$ satisfies the RIP with parameters $k = \frac{cn}{\log^3 n}$ and ε.

On this event, by Theorem 3.4.3, it follows that

$$\|x\|_2 \leqslant \frac{C(\varepsilon)\log^{3/2} n}{\sqrt{cn}}\|x\|_1 \leqslant \frac{\log^2 n}{\sqrt{n}}\|x\|_1, \tag{3.2.11}$$

for all $x \in \ker \widehat{F}$ and large enough n, where we absorbed the constants in the additional $(\log n)^{1/2}$ factor in the numerator. Since $\widehat{F}$ consists of a subset of rows of $\widetilde{F}$, all $x \in \ker \widetilde{F}$ are also in $\ker \widehat{F}$, so the proof is complete. □

Remark 3.2.5 (Alternative proof technique)**.** *Theorem 3.2.4 also follows from an intermediate step in the proof of RIP of the subsampled Fourier matrix in [15]. More specifically, in our notation [15], Theorem 3.1 implies that* $\|\widehat{F}x\| \geqslant (1 - \varepsilon)\|Fx\|_2^2 - C\varepsilon/k\|x\|_1^2$ *with high probability, and since* $x \in \ker \widehat{F}$*, it follows that* $\|x\|_2 \leqslant \frac{\log^2 n}{\sqrt{n}}$.

Now we present the proof of our main result.

Proof of Theorem 1.2.1. We begin with the lower bound $\mathbb{E}\,\vartheta(G) \geqslant \sqrt{n}$. Since G is vertex-transitive, it holds that $\vartheta(G)\vartheta(\overline{G}) = n$, see [[]Theorem 8]lovasz1979shannon. Therefore,

$$\log n = \mathbb{E}\log \vartheta(G)\vartheta(\overline{G}) = 2\,\mathbb{E}\log \vartheta(G) \leqslant 2\log \mathbb{E}\,\vartheta(G), \tag{3.2.12}$$

where we used the fact that G equals in distribution to $\overline{G}$ together with Jensen's inequality and linearity of the expected value. Upon exponentiating we obtain

$$\mathbb{E}\,\vartheta(G) \geqslant \sqrt{n}. \tag{3.2.13}$$

To prove the upper bound, we use the LP formulation of the Lovász number as in Theorem 3.2.1. Let A denote the intersubsection of the events

of Theorems 3.2.2 and 3.2.4, with $\Pr(A) \geqslant 1 - \frac{3}{n}$ from union bound, and let B denote the event of Theorem 3.2.3. Since $\mathbb{E}[\vartheta | \overline{A} \text{ or } \overline{B}] \Pr(\overline{A} \text{ or } \overline{B}) = O(1)$, we condition on A and B in the following. For constant C defined in Theorem 3.2.3, we split g into two parts, g_{small} and g_{large}, where

$$(g_{\text{small}})_k = \begin{cases} g_k & \text{if } |g_k| < C\sqrt{n \log\log n}, \\ 0 & \text{otherwise,} \end{cases} \tag{3.2.14}$$

and $g_{\text{large}} = g - g_{\text{small}}$. Then, $\langle y, g \rangle = \langle y, g_{\text{small}} \rangle + \left\langle y, g_{\text{large}} \right\rangle$. We bound each term separately: first,

$$\langle y, g_{\text{small}} \rangle \leqslant \|y\|_1 \|g_{\text{small}}\|_\infty = O(\sqrt{n \log\log n}). \tag{3.2.15}$$

On the event B we have that g_{large} is $\frac{n}{\log^{10} n}$-sparse. From eq. (3.2.6) $\|g_{\text{large}}\|_\infty = O(\sqrt{n \log n})$, which implies that $\|g_{\text{large}}\|_2 = O(n/\log^4 n)$. Using Cauchy-Schwartz inequality together with Theorem 3.2.4, we bound the second term as follows:

$$\left\langle y, g_{\text{large}} \right\rangle \leqslant \|y\|_2 \|g_{\text{large}}\|_2 \leqslant \frac{\log^2 n}{\sqrt{n}} \cdot \frac{n}{\log^4 n} = O(\sqrt{n}), \tag{3.2.16}$$

which completes the proof. □

3.3 DISCUSSION

Based on numerical observations, we formulate the following conjecture.

Conjecture 3.3.1. *Let G be a dense random circulant graph. Then,*

$$\mathbb{E}\, \vartheta(G) = (1 + o(1))\sqrt{n}. \tag{3.3.1}$$

Existing lower bounds against RIP (see [12, 47]) do not allow us to use our proof strategy for showing Theorem 3.3.1. Indeed, there exist $\frac{n}{\log n}$-sparse vectors in the kernel of $\widetilde{F}$, which contradicts the desired inequality $\|y\|_2 \leqslant \frac{C}{\sqrt{n}} \|y\|_1$ for $y \in \ker \widetilde{F}$. However, it is still possible that no cn-sparse *entrywise positive* vector exists in the kernel of $\widetilde{F}$, for small enough constant $c > 0$. It is also plausible that constructing a feasible vector for the dual programs in Table 3.1 may lead to tighter upper bounds. We leave these questions for the future work.

PALEY GRAPH A classical example of a circulant graph is *Paley* graph. For a prime $p \equiv 1 \mod 4$, it is defined as the graph on p vertices with vertices i and j connected if and only if $i - j$ is a quadratic residue modulo p, see [48, 49]. Paley graphs are believed to exhibit certain *pseudorandom* properties, and bounding its independence number is a long-standing open problem in number theory and combinatorics [50]. This quantity can be upper bounded by the Lovász number of a certain subgraph called 1-localization which is circulant [51].

Recently, several optimization based approaches were considered, see [51–53]. In [11], a numerical evidence similar to Theorem 3.3.1 regarding subgraphs of Paley graph was observed, which if true, recovers the best known upper bound on the independence number due to [50].

3.4 USEFUL INEQUALITIES

Lemma 3.4.1 (Hoeffding's inequality)**.** *Let $X_1, \dots, X_n$ be independent random variables, such that $\mathbb{E}\, X_i = 0$ and $a \leqslant X_i \leqslant b$ almost surely. Then,*

$$\Pr\left(\left|\sum_{i=1}^{n} X_i\right| \geqslant t\right) \leqslant 2\exp\left(-\frac{2t^2}{n(b-a)^2}\right) \tag{3.4.1}$$

Lemma 3.4.2 (RIP of subsampled DFT matrix, [15])**.** *Let $F \in \mathbb{C}^{n\times n}$ be a DFT matrix: $F_{jk} = \exp(-2\pi ijk/n)$ for $j, k \in [n]$. There exist $c > 0$ and $0 < \varepsilon < 1/3$, such that for all n large enough, a matrix consisting of $q \geqslant \lceil n/4 \rceil$ uniformly subsampled rows of F and rescaled by $1/\sqrt{q}$ has (k, ε)-RIP for $k = \frac{cn}{\log^3 n}$, with probability at least $1 - 2^{\Omega(-\log^2 n)}$.*

Lemma 3.4.3 (e.g. [54], Theorem 11)**.** *If $A \in \mathbb{C}^{m\times n}$ satisfies the RIP with parameters k and $\varepsilon < 1/3$, then there exists $C = C(\varepsilon)$, such that for any $x \in \ker A$ we have that*

$$\|x\|_2 \leqslant \frac{C}{\sqrt{k}}\|x\|_1. \tag{3.4.2}$$

4

ON THE GROWTH OF MISTAKES IN DIFFERENTIALLY PRIVATE ONLINE LEARNING: A LOWER BOUND PERSPECTIVE

4.1 PRELIMINARIES

We provide the necessary definitions for both Online Learning and Differential Privacy.

4.1.1 *Online Learning*

We begin by defining the online learning game between a learner $\mathcal{A}$ and an adversary $\mathcal{B}$. Let $T \in \mathbb{N}_+$ denote number of rounds and let $\mathcal{X}$ be some domain.

Definition 4.1.1 (General game)**.** Let $\mathcal{H} \subseteq \{0,1\}^{\mathcal{X}}$ be a hypothesis class of functions from $\mathcal{X}$ to $\{0,1\}$. The game between a learner $\mathcal{A}$ and adversary $\mathcal{B}$ is played as follows:

- adversary $\mathcal{B}$ picks $f^* \in \mathcal{H}$ and a sequence $x_1, \ldots, x_T \in \mathcal{X}$
- **for** $t = 1, \ldots, T$:
  - learner $\mathcal{A}$ outputs a current prediction $\hat{f}_t \in \{0,1\}^{\mathcal{X}}$ (see Theorem 4.1.3)
  - $\mathcal{A}$ receives $(x_t, f^*(x_t))$
- **let** $M_{\mathcal{A}} = \sum_{t=1}^{T} \mathbb{I}\left\{\hat{f}_t(x_t) \neq f^*(x_t)\right\}$

In this work, we study the following min-max problem:

$$\min_{\mathcal{A}} \max_{\mathcal{B}} \mathbb{E}\left[M_{\mathcal{A}}\right] = \min_{\mathcal{A}} \max_{\mathcal{B}} \sum_{t=1}^{T} \Pr\left[\hat{f}_t(x_t) \neq f^*(x_t)\right], \tag{4.1.1}$$

where probability is taken over the randomness in $\mathcal{A}$. We refer to the random variable $M_{\mathcal{A}}$ as the mistake count of $\mathcal{A}$. The optimal mistake count of this game can be characterised by a combinatorial property of the

hypothesis class $\mathcal{H}$, called *Littlestone dimension*, first shown by Littlestone [25].

LITTLESTONE DIMENSION To define *Littlestone dimension*, we need the concept of *mistake tree*. A mistake tree $\mathcal{T}$ is a complete binary tree, where each internal node v corresponds to some $x_v \in \mathcal{X}$. Each root-to-leaf path of the tree – denoted as $x_1, x_2, \ldots, x_d, x_{\text{leaf}}$ – is associated with a label sequence $y_1, \ldots, y_d$, where $y_i = \mathbb{I}\{x_{i+1} \text{ is the right child of } x_i\}$.

We say that $\mathcal{T}$ is *shattered* by $\mathcal{H}$, if for every possible root-to-leaf labeled path $((x_1, y_1), \ldots, (x_d, y_d))$, there exists $f \in \mathcal{H}$, such that $f(x_i) = y_i$ for all $i \in [d]$. This concept leads us to the formal definition of the Littlestone dimension as follows:

Definition 4.1.2. Littlestone dimension $(\text{Ldim}(\mathcal{H}))$ of a hypothesis class $\mathcal{H}$ is defined as the maximum depth of any mistake tree that can be shattered by $\mathcal{H}$.

Littlestone [25] proved that for any hypothesis class $\mathcal{H}$, there exists a deterministic learner, called Standard Optimal Algorithm (SOA) such that $M_{\text{SOA}} \leqslant \text{Ldim}(\mathcal{H})$. Furthermore, for any learner $\mathcal{A}$ there exists a (possibly random) adversary $\mathcal{B}$, such that $\mathbb{E}[M_{\mathcal{A}}] \geqslant \frac{\text{Ldim}(\mathcal{H})}{2}$, where expectation is taken with respect to the randomness in $\mathcal{B}$. However, the SOA learner is not restricted to output a hypothesis in $\mathcal{H}$ while making its predictions. Such learning algorithms are classified as *improper learners*, which we define formally below:

Definition 4.1.3 (Proper and Improper learner)**.** A learner $\mathcal{A}$ for a hypothesis class $\mathcal{H}$ is called *proper* if its output is restricted to belong to $\mathcal{H}$. Any learner that is not *proper* is called *improper*.

It is worth noting that most learners in online learning are improper learners though their output may only be simple mixtures of hypothesis in $\mathcal{H}$ [55]. We illustrate the importance of improper learners with a simple hypothesis class that we heavily use in the rest of this paper.

Definition 4.1.4 (Point class)**.** For $N \in \mathbb{N}_+$, for domain $\mathcal{X} = [N] := \{1, \ldots, N\}$, define

$$\text{Point}_N := \left\{ f^{(i)}, i \in [N] \right\}, \quad \text{where} \quad f^{(i)}(x) = \mathbb{I}\{i = x\}. \tag{4.1.2}$$

Note that $\text{Ldim}(\text{Point}_N) = 1$ for any $N > 1$. A simple algorithm to learn Point_N predicts 0 for every input until it makes a mistake. The input i

where it incurred the mistake must correspond to the true target concept $f^{(i)}$. This algorithm is improper as no hypothesis f in Point_N predicts 0 universally over the whole domain.

4.1.2 *Differential Privacy*

In this work, our goal is to study learners that satisfy the DP guarantee [16] defined formally below.

Definition 4.1.5 (Approximate differential privacy)**.** An algorithm $\mathcal{A}$ is said to be (ε, δ)-DP, if for any two input sequences $\tau = ((x_1, y_1), \ldots, (x_T, y_T))$ and $\tau' = \left((x'_1, y'_1), \ldots, (x'_T, y'_T)\right)$, such that there exists **only one** t with $(x_t, y_t) \neq (x'_t, y'_t)$, it holds that

$$\Pr(\mathcal{A}(\tau) \in S) \leqslant \exp(\varepsilon) \Pr(\mathcal{A}(\tau') \in S) + \delta, \tag{4.1.3}$$

where S is any set of possible outcomes.

When $\delta = 0$ we recover the definition of *pure differential privacy*, denoted by ε-DP. Note that for online learner $\mathcal{A}$, output at step t depends only on first $t-1$ elements of the input sequence. In the setting of offline learning, the inputs τ, τ' can be thought of as two datasets of length T and $\mathcal{A}$ as the learning algorithm that outputs one hypothesis f (not necessarily in $\mathcal{H}$). If $\mathcal{A}$ simulatanously satisfies DP and is a PAC learner [17], it defines the setting of DP-PAC learning [20]. However, the setting of DP-online learning is more nuanced due to two reasons.

Privacy of Prediction or Privacy of Predictor The first complexity arises from what the privacy adversary observes when altering an input, termed as its *view*. Since $\mathcal{A}$ provides an output hypothesis $\hat{f}_t \in \{0,1\}^{\mathcal{X}}$ at every time step $t \in [T]$ as shown in Theorem 4.1.1, the adversary's view could encompass the entire list of predictors. Our work, like Golowich & Livni [27], focuses on this scenario, where the output set is $S \subseteq \{0,1\}^{\mathcal{X} \times T}$. Nevertheless, certain studies restrict the adversary's view to only the predictions, excluding the predictors themselves [56–58]. In this setting, it is also important to assume that the adversary only observes the predictions on the inputs that it did not change [59, 60]; thus, they have $S \subseteq \{0,1\}^{(T-1)}$.

Oblivious and Adaptive adversary The second complexity is about whether the online adversary $\mathcal{B}$ pre-selects all input points or adaptively chooses the next point based on the learner $\mathcal{A}$'s previous response. Although the former, known as an *oblivious* adversary, seems less potent, this difference

does not manifest itself in non-private learning [61]. However, this distinction becomes significant in the context of DP online learning. Adaptive adversaries, by design, leverage historical data in their decision-making process. While works like Kaplan *et al.* [60] focus on adaptive adversaries, others like Kearns *et al.* [59] concentrate on oblivious ones, and Golowich & Livni [27] examine both. Our contribution lies in setting lower bounds against the simpler scenario of oblivious adversaries.

4.2 RELATED WORK

Understanding which hypothesis classes can be privately learned is an area of vibrant research and was started in the context of Valiant's PAC learning model [17]. A hypothesis class $\mathcal{H}$ is considered PAC-learnable if there exists an algorithm $\mathcal{A}$, which can utilize a polynomial-sized[1], independent, and identically distributed (i.i.d.) sample D from any data distribution to produce a hypothesis $h \in \mathcal{H}$ that achieves a low classification error with high probability on that distribution. In the context of DP-PAC learning, as defined by Kasiviswanathan *et al.* [20], the learner $\mathcal{A}$ must also satisfy DP constraint with respect to the sample D. The overarching objective in this research domain is to find a clear criterion for the private learnability of hypothesis classes, analogous to the way learnability has been characterized in non-private settings—through the Vapnik-Chervonenkis (VC) dimension for offline learning [62] and the Littlestone dimension for online learning [25, 63].

Kasiviswanathan *et al.* [20] started this line of research by showing that the sample complexity of DP-PAC learning a hypothesis class $\mathcal{H}$ is $O(\log(|\mathcal{H}|))$. Beimel *et al.* [23] showed that the VC dimension does not dictate the sample complexity for proper pure DP-PAC learning of the Point_N class. However, they showed that if the setting is relaxed to improper learning then this sample complexity can be improved, thus showing a separation between proper and improper learning, something that is absent in the non-private PAC model. Beimel, Nissim & Stemmer [21] sharpened this result by constructing a new complexity measure called *probabilistic representation dimension* and proving that this measure characterises improper pure DP exactly.

1 The term 'polynomial-sized' refers to a sample size that is polynomial in the PAC parameters, including the error rate, confidence level, size of the hypothesis class, and the dimensionality of the input space.

By leveraging advanced tools from communication complexity theory, they refined the understanding of the probabilistic representation dimension and demonstrated that the sample complexity for learning a notably simple hypothesis class, denoted as Line_p, under approximate improper DP-PAC conditions, is significantly lower than the corresponding lower bound established for pure contexts.

Relaxing the notion of pure DP to approximate DP, Beimel, Nissim & Stemmer [56] showed that the sample complexity for proper approximate DP-PAC learning can be significantly lower than proper pure DP-PAC learning, thereby showing a separation between pure and approximate DP in the context of proper DP-PAC learning. The inquiry into whether a similar discrepancy exists in improper DP-PAC learning was resolved by Feldman & Xiao [22] who proved a separation between pure and approximate DP in the improper DP-PAC learning model. To do this, they first proved a sharper characterisation of the probabilistic representation dimension using concepts from communication complexity. Then, they showed that the sample complexity for learning a notably simple hypothesis class, denoted as Line_p, under approximate improper DP-PAC conditions, is significantly lower than the corresponding lower bound established for pure DP.

Feldman & Xiao [22] were also the first to obtain lower bounds for DP-PAC learning that grows as $\Omega(\text{Ldim}(\mathcal{H}))$, albeit limited to the pure DP setting. Alon *et al.* [26] showed that it is possible to obtain a lower bound for approximate DP that grows as $\Omega(\log^*(\text{Ldim}(\mathcal{H})))$ thus marking a clear distinction between non-private and approximate DP-PAC learning. This finding illustrated that DP-PAC learning's complexity could align with that of online learning, which is similarly governed by the Littlestone dimension. In a series of subsequent works, see [24, 64], a surprising connection was established between private offline learning and non-private online learning. In particular, classes that are privately offline learnable are precisely those with finite Littlestone dimension.

This naturally highlights a similar question of private online learning, in particular whether DP further limits which classes are learnable in the DP-Online learning model. Golowich & Livni [27] provided an algorithm, called DP-SOA, which has expected number of mistakes growing as $O(2^{2^{\text{Ldim}}} \log T)$. Interestingly, unlike SOA in the non-private online setting, DP-SOA's mistake count increases with number of steps T in the online game. When considering adaptive adversaries, the upper bound on mistakes escalates to $O\left(\sqrt{T}\right)$. Under a slightly weaker definition of DP, known as Challenge-DP, where the privacy adversary only sees the predic-

tions and not the whole predictor function, Kaplan *et al.* [60] obtained an upper bound of $O\left(\log^2(T)\right)$ for both adaptive and oblivious adversaries. However, it is not clear from these works, whether the dependence on T is unavoidable. A related setting is that of *continual observation under DP* where such a dependence is indeed unavoidable under the pure DP model. However, the results from continual observation do not immediately transfer to online learning as discussed in Section 4.4.1.

4.3 LOWER BOUND FOR PRIVATE ONLINE LEARNING UNDER CONCENTRATION ASSUMPTION

In this section, we provide the main result of our work along with their proof. Before stating the main result in Theorem 4.3.3, we need to define the concept of *distinguishing tuple* and β-concentrated learners.

Definition 4.3.1. Given $f_0, f_1 \in \mathcal{H}$ and $x_{\text{eq}}, x_{\text{dif}} \in \mathcal{X}$, we call the tuple $(f_0, f_1, x_{\text{eq}}, x_{\text{dif}})$ *distinguishing*, if it satisfies both $f_0(x_{\text{eq}}) = f_1(x_{\text{eq}})$ and $f_0(x_{\text{eq}}) = f_0(x_{\text{dif}}) \neq f_1(x_{\text{dif}})$.

A distinguishing tuple means that there are two functions (f_0, f_1), and two input points $(x_{\text{eq}}, x_{\text{dif}})$, such that only one of these points can effectively differentiate between the two functions. The absence of a distinguishing tuple implies a restricted hypothesis class: either $\mathcal{H}$ is a singleton ($|\mathcal{H}| = 1$), or it contains precisely two inversely related functions ($|\mathcal{H}| = 2$ with $f_1 = 1 - f_0$). In the latter case, *every* input point contains information distinguishing f_1 and f_2. This implies that there is no difference between input sequences, and the mistake bound will not depend on T. For the purposes of our analysis, we proceed under the assumption that a distinguishing tuple always exists.

Let $(f_0, f_1, x_{\text{eq}}, x_{\text{dif}})$ be a distinguishing tuple and suppose that adversary chooses $f^* \in \{f_0, f_1\}$. Knowing only information on $f^*(x_{\text{eq}})$ does not help to tell apart f_0 from f_1. Furthermore, if an algorithm is 'too confident', meaning that it strongly prefers output of f_0 on x_{dif} over output of f_1, it will necessarily make a mistake on x_{dif} if $f^* = f_1$. We will use this basic intuition to obtain our main lower bound and we will call such learners 'concentrated', as defined below.

For input τ, index $t \in [T]$ and an input point $x \in \mathcal{X}$, we denote $\mathcal{A}(\tau)_t[x]$ to be the value of the t-th output function of $\mathcal{A}(\tau)$ evaluated at point x.

Definition 4.3.2. An algorithm $\mathcal{A}$ is called β-concentrated, if there exists a distinguishing tuple $(f_0, f_1, x_{\text{eq}}, x_{\text{dif}})$, such that

$$\Pr\left[\forall t \in [T],\ \mathcal{A}\left(\tau_0\right)_t\left[x_{\text{dif}}\right] = f_0(x_{\text{dif}})\right] \geqslant 1 - \beta, \tag{4.3.1}$$

where $\tau_0 = \left(\left(x_{\text{eq}}, f_0(x_{\text{eq}})\right), \ldots, \left(x_{\text{eq}}, f_0(x_{\text{eq}})\right)\right)$.

Note that τ_0 from Theorem 4.3.2 is a 'dummy', *non-distinguishing* input, as it does not contain any information to distinguish f_0 from f_1.

4.3.1 *Main Result*

We now show that if the learner is both differentially private and concentrated, it will necessarily suffer a large (logarithmic in T) number of mistakes in the game of Theorem 4.1.1.

Theorem 4.3.3. *Let $\mathcal{H}$ be an arbitrary hypothesis class. Let $\varepsilon > 0$, $\delta \leqslant \varepsilon^2$ and $T \leqslant \exp(1/(32\delta))$. If, for some $\delta \leqslant \beta \leqslant 1/10$, $\mathcal{A}$ is a β-concentrated (ε, δ)-DP online learner of $\mathcal{H}$, then there exists an adversary $\mathcal{B}$, such that*

$$\mathbb{E}\left[M_{\mathcal{A}}\right] = \widetilde{\Omega}\left(\frac{\log T/\beta}{\varepsilon}\right), \tag{4.3.2}$$

where $\widetilde{\Omega}$ contains logarithmic in ε factors. For $T > \exp(1/(32\delta))$, $\mathbb{E}\left[M_{\mathcal{A}}\right] = \widetilde{\Omega}\left(1/\delta\right)$.

Before comparing our lower bound with known upper bounds, we first discuss the condition $T \leqslant \exp(1/(32\delta))$. Our bound suggests that for sufficiently large T, specifically when T exceeds $\exp(1/(32\delta))$, the dependency on T is in fact not needed. This can be seen from the following simple *Name & Shame* algorithm: initialize an empty set S; at each step, apply SOA and output $\text{SOA}(S)$; upon receiving a new entry $(x_t, f^*(x_t))$, add it to S with probability δ. Clearly, this algorithm is $(0, \delta)$-DP, since for any fixed input point, it only depends on this point with probability δ. Furthermore, each time the algorithm incurs a mistake, it adds this mistake to S with probability δ. Since the algorithm runs SOA on S, it will make on expectation at most $\text{Ldim}\left(\mathcal{H}\right)/\delta$ mistakes. This algorithm is, however, not 'conventionally' private, since it potentially discloses a δ-fraction of the data, namely the set S.

Now, we compare result of Theorem 4.3.3 with known upper bounds in Figure 4.1. Recall that DP-SOA in Golowich & Livni [27] obtained an

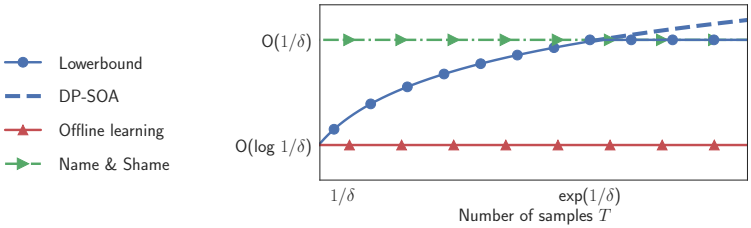

FIGURE 4.1: Lower bound from Theorem 4.3.3 vs. existing upper bounds. We assume that ε, δ are fixed and for simplicity ignore dependence on ε. X-axis corresponds to the number of samples, growing from $T \sim 1/\delta$ to $T \sim \exp(1/\delta)$ and larger. Y-axis shows the expected number of mistakes, $\mathbb{E}[M_{\mathcal{A}}]$.

upper bound which increases logarithmically with the time horizon T. Figure 4.1 shows that for $T \leqslant \exp(1/(32\delta))$, the dependency on T is necessary, thereby showing the tightness of DP-SOA [27]. For larger T, the aforementioned *Name & Shame* actually outperforms DP-SOA, indicating that, for fixed ε, δ, *online algorithm always compromises privacy at very large* T. However, even for smaller T, the plot shows that the number of mistake must grow with increasing T until it matches the *Name & Shame* algorithm.

The main idea behind the construction of the lower bound is the following: assume that there is a distinguishing tuple $(f_0, f_1, x_{\text{eq}}, x_{\text{dif}})$, such that $\mathcal{A}$ on $(x_{\text{eq}}, \ldots, x_{\text{eq}})$ with high probability only outputs functions that are equal to f_0 at x_{dif}. Then, the adversary picks $f^* = f_1$ and its goal is to select time steps t to insert x_{dif}, such that (i) $\mathcal{A}$ with high probability will make a mistake at t, and (ii) $\mathcal{A}$ will not be able to 'extract a lot of information' from this mistake. As we show, both of these conditions can be guaranteed using the DP property and concentration assumption of $\mathcal{A}$. Note that if adversary inserts x_{dif} serially starting from $t = 1$, $\mathcal{A}$ can have regret much smaller than $\log T$. For example, if $\mathcal{A}$ first incurs a constant number of mistakes, the group privacy property allows a considerable change in the output distribution to only output $f_1 = f^*$. But this is possible only if $\mathcal{A}$ can predict in advance, where it will make the mistake in the future.

Therefore, the adversary needs to 'disperse' the points x_{dif} across T time steps, such that $\mathcal{A}$ cannot predict, where it might make the next mistake. We begin our construction by first inserting a point at the beginning. Then, depending on whether the leaner anticipates more points x_{dif} in the first half *or* in the second half of the sequence, we insert x_{dif} in the half where it least expects it. Note that because of the concentration and DP assumption, the learner cannot anticipate points x_{dif} in both halves simultaneously (recall that on input sequence consisting of only x_{eq}, learner does not output a

correct function for x_{dif}). By continuing this construction recursively, we are able to insert $\Omega(\log T)$ points x_{dif}, such that with constant probability, on each of them $\mathcal{A}$ will make a mistake.

of Theorem 4.3.3. Since $\mathcal{A}$ is 0.1-concentrated, there exists a distinguishing tuple $(f_0, f_1, x_{\text{eq}}, x_{\text{dif}})$, such that

$$\Pr\left[\forall t \in [T],\ \mathcal{A}\left(\tau_0\right)_t\left[x_{\text{dif}}\right] = f_0(x_{\text{dif}})\right] \geqslant 0.9, \tag{4.3.3}$$

where $\tau_0 = \left(\left(x_{\text{eq}}, f_0(x_{\text{eq}})\right), \ldots, \left(x_{\text{eq}}, f_0(x_{\text{eq}})\right)\right)$. WLOG assume that $f_0(x_{\text{eq}}) = f_0(x_{\text{dif}}) = 0$. When $T > \exp(1/(32\delta))$, by simply only using the first $\exp(1/(32\delta))$ rounds, we obtain the required lower bound $\widetilde{\Omega}(1/\delta)$. In the remaining, we assume that $T \leqslant \exp(1/(32\delta))$. Furthermore, let k be the largest integer, such that $2^k - 1 \leqslant T$. Note that $\frac{1}{2}\log T \leqslant k \leqslant 2\log T \leqslant 1/(16\delta)$. For simplicity, we also assume that $T = 2^k - 1$. We start with the case $\varepsilon = \varepsilon_0 = \log(3/2)$ and pick $f^* = f_1$ for the adversary.

Note that to show $\mathbb{E}\left[M_{\mathcal{A}}\right] = \Omega\left(\log T/\beta\right)$, we can construct two adversaries, first achieving $\mathbb{E}\left[M_{\mathcal{A}}\right] = \Omega\left(\log 1/\beta\right)$ (**Case I**) and the other $\mathbb{E}\left[M_{\mathcal{A}}\right] = \Omega\left(\log T\right)$ (**Case II**). We now prove each of them.

Case I: We start with showing the adversary for the former bound, i.e., such that $\mathbb{E}\left[M_{\mathcal{A}}\right] = \Omega\left(\log 1/\beta\right)$. The concentration assumption implies that for any $t \in [T]$, $\Pr\left(\mathcal{A}\left(\tau_0\right)_t\left[x_{\text{dif}}\right] = 1\right) \leqslant \beta$. Therefore, if τ_k contains k copies of the point $(x_{\text{dif}}, 1)$, by applying DP property of $\mathcal{A}$ k times, we obtain that for any $t \in [T]$, $\Pr\left(\mathcal{A}\left(\tau_k\right)_t\left[x_{\text{dif}}\right] = 1\right) \leqslant \beta\exp(k\varepsilon_0) + \delta\left(\frac{\exp(k\varepsilon_0)-1}{\exp(\varepsilon_0)-1}\right) \leqslant (k\delta + \beta)\exp(k\varepsilon_0)$. If $k = \frac{1}{8\varepsilon_0}\log(1/\beta)$, we derive that for any $t \in [T]$,

$$\begin{aligned}\Pr(\mathcal{A}(\tau_k)_t\left[x_{\text{dif}}\right] = 1) &\leqslant (k\delta + \beta)\exp(k\varepsilon_0) \\ &\leqslant \frac{\delta\log(1/\delta)}{8\delta^{1/8}\varepsilon_0} \leqslant \frac{1}{3}\delta^{1/2}\log(1/\delta) \leqslant \frac{1}{2}.\end{aligned} \tag{4.3.4}$$

This implies that for the sequence τ_k, expected number of mistakes is $\mathbb{E}\left[M_{\mathcal{A}}\right] \geqslant \frac{1}{16\varepsilon_0}\log(1/\beta)$.

Case II: In the previous case, it did not matter where exactly the points $(x_{\text{dif}}, 1)$ are inserted. However, if we want to prove the lower bound $\Omega(\log T)$, this is no longer true and one needs to be careful with the placement of the inserted points.

In the following, we construct a sequence $\boldsymbol{c} = (x_1, \ldots, x_T)$, such that expected number of mistakes of $\mathcal{A}$ will be large. We proceed iteratively, maintaining scalar sequences $(l_i), (r_i)$, and sequences $\boldsymbol{c}^{(i)}$ such that

1. $\boldsymbol{c}^{(i)}$ contains exactly i points x_{dif} on the prefix $[1, l_i - 1]$,
2. $p_i \coloneqq \Pr\left[\forall t \in [1, l_i - 1],\ \mathcal{A}(\boldsymbol{c}^{(i)})_t\,[x_{\text{dif}}] = 0\right] \geqslant 1/2 - 4i\delta$,
3. $q_i \coloneqq \Pr\begin{bmatrix}\forall t \in [1, l_i - 1],\ \mathcal{A}(\boldsymbol{c}^{(i)})_t\,[x_{\text{dif}}] = 0 \textbf{ and } \\ \exists t \in [l_i, r_i],\ \text{ s.t. } \mathcal{A}(\boldsymbol{c}^{(i)})_t\,[x_{\text{dif}}] = 1\end{bmatrix} \leqslant 2\delta$.

Assume that we obtain these sequences until $i = k \geqslant \frac{1}{2}\log T$. Then on the event $\left\{\forall t \in [1, l_k - 1], \mathcal{A}(\boldsymbol{c}^{(i)})_t\,[x_{\text{dif}}] = 0\right\}$, $\mathcal{A}$ will make k mistakes. Since $k \leqslant 1/(16\delta)$, from the second property above $p_k \geqslant 1/4$. This implies that for the sequence $\boldsymbol{c}^{(k)}$ we have $\mathbb{E}\,[M_{\mathcal{A}}] \geqslant p_k k \geqslant \frac{1}{8}\log T$.

We construct the sequences by induction, starting with $\boldsymbol{c}^{(0)} = (x_{\text{eq}}, \ldots, x_{\text{eq}})$. For each i, $\boldsymbol{c}^{(i)}$ will differ from $\boldsymbol{c}^{(i+1)}$ at exactly one point. This allows us to use DP property of $\mathcal{A}$ in order to compare outputs on $\boldsymbol{c}^{(i)}$ and $\boldsymbol{c}^{(i+1)}$. From 0.1-concentrated assumption, we can pick $l_0 = 1, r_0 = T$ which gives $p_0 = 1$ and $q_0 = 0.1$ (we interpret $\Pr[\forall t \in \varnothing \ldots] = 1$).

Given $\boldsymbol{c}^{(i)}$ we construct $\boldsymbol{c}^{(i+1)}$ by substituting l_i-th input point with x_{dif}:

1. Let $(\boldsymbol{c}^{(i+1)})_j = (\boldsymbol{c}^{(i)})_j$ for all $j \neq l_i$,
2. Set $(\boldsymbol{c}^{(i+1)})_{l_i} = x_{\text{dif}}$.

Now we compute l_{i+1}, r_{i+1} and bound p_{i+1}, q_{i+1}. To do this, we first introduce

$$\begin{aligned} p'_i &\coloneqq \Pr\left[\forall t \in [1, l_i],\ \mathcal{A}(\boldsymbol{c}^{(i)})_t\,[x_{\text{dif}}] = 0\right] \geqslant p_i - q_i, \\ q'_i &\coloneqq \Pr\begin{bmatrix}\forall t \in [1, l_i],\ \mathcal{A}(\boldsymbol{c}^{(i)})_t\,[x_{\text{dif}}] = 0 \textbf{ and } \\ \exists t \in [l_i + 1, r_i],\ \text{ s.t. } \mathcal{A}(\boldsymbol{c}^{(i)})_t\,[x_{\text{dif}}] = 1\end{bmatrix} \leqslant q_i, \end{aligned} \tag{4.3.5}$$

which just account for the shift $l_i \to l_i + 1$. Note that, since the events are nested, we can compute $p'_i - q'_i = \Pr\left[\forall t \in [1, r_i], \mathcal{A}(\boldsymbol{c}^{(i)})_t\,[x_{\text{dif}}] = 0\right] = p_i - q_i$. Next, for $m_i = (l_i + r_i)/2$ and for any $\boldsymbol{x} \in \{x_{\text{eq}}, x_{\text{dif}}\}^T$, define

$$\begin{aligned} Q(\boldsymbol{x}) &\coloneqq \left\{\begin{matrix}\forall t \in [1, l_i],\ \mathcal{A}(\boldsymbol{x})_t\,[x_{\text{dif}}] = 0 \textbf{ and } \\ \exists t \in [l_i + 1, r_i],\ \text{ s.t. } \mathcal{A}(\boldsymbol{x})_t\,[x_{\text{dif}}] = 1\end{matrix}\right\}, \\ Q_1(\boldsymbol{x}) &\coloneqq \left\{\begin{matrix}\forall t \in [1, l_i],\ \mathcal{A}(\boldsymbol{x})_t\,[x_{\text{dif}}] = 0 \textbf{ and } \\ \exists t \in [l_i + 1, m_i],\ \text{ s.t. } \mathcal{A}(\boldsymbol{x})_t\,[x_{\text{dif}}] = 1\end{matrix}\right\}, \\ Q_2(\boldsymbol{x}) &\coloneqq \left\{\begin{matrix}\forall t \in [1, m_i],\ \mathcal{A}(\boldsymbol{x})_t\,[x_{\text{dif}}] = 0 \textbf{ and } \\ \exists t \in [m_i + 1, r_i],\ \text{ s.t. } \mathcal{A}(\boldsymbol{x})_t\,[x_{\text{dif}}] = 1\end{matrix}\right\}. \end{aligned} \tag{4.3.6}$$

Clearly, for any $\boldsymbol{x}$, $Q(\boldsymbol{x}) = Q_1(\boldsymbol{x}) \cup Q_2(\boldsymbol{x})$ with $Q_1(\boldsymbol{x}) \cap Q_2(\boldsymbol{x}) = \emptyset$. Therefore,

$$q_i' = \Pr\left[Q(\boldsymbol{c}^{(i)})\right] = \Pr\left[Q_1(\boldsymbol{c}^{(i)})\right] + \Pr\left[Q_2(\boldsymbol{c}^{(i)})\right]. \tag{4.3.7}$$

We can use DP property of $\mathcal{A}$ when comparing outputs on $\boldsymbol{c}^{(i)}$ and $\boldsymbol{c}^{(i+1)}$ to get

$$\begin{aligned}
&\min\left(\Pr\left[Q_1(\boldsymbol{c}^{(i+1)})\right], \Pr\left[Q_2(\boldsymbol{c}^{(i+1)})\right]\right) \\
&\leqslant \frac{1}{2}(\Pr\left[Q_1(\boldsymbol{c}^{(i+1)})\right] + \Pr\left[Q_2(\boldsymbol{c}^{(i+1)})\right]) \\
&= \frac{1}{2}\Pr\left[Q(\boldsymbol{c}^{(i+1)})\right] \\
&\leqslant \frac{1}{2}\left(\exp(\varepsilon_0)\Pr\left[Q(\boldsymbol{c}^{(i)})\right] + \delta\right) \\
&= \exp(\varepsilon_0)q_i'/2 + \delta/2 \leqslant \frac{3}{4}q_i + \delta/2.
\end{aligned} \tag{4.3.8}$$

If $\Pr\left(Q_1(\boldsymbol{c}^{(i+1)})\right) \leqslant \Pr\left(Q_2(\boldsymbol{c}^{(i+1)})\right)$ we set $l_{i+1} := l_i + 1, r_{i+1} := m_i$, which gives $q_{i+1} = \Pr\left(Q_1(\boldsymbol{c}^{(i+1)})\right)$ and $p_{i+1} = p_i' \geqslant p_i - q_i$. When $\Pr\left(Q_1(\boldsymbol{c}^{(i+1)})\right) > \Pr\left(Q_2(\boldsymbol{c}^{(i+1)})\right)$, we set $l_{i+1} = m_i + 1, r_{i+1} = r_i$, with $q_{i+1} = \Pr\left(Q_2(\boldsymbol{c}^{(i+1)})\right)$. We can bound

$$\begin{aligned}
p_{i+1} &= \Pr\left(\forall t \in [1, m_i],\ \mathcal{A}(\boldsymbol{c}^{(i+1)})_t\left[x_{\text{dif}}\right] = 0\right) \\
&= \Pr\left(\forall t \in [1, l_i],\ \mathcal{A}(\boldsymbol{c}^{(i+1)})_t\left[x_{\text{dif}}\right] = 0\right) \\
&\quad - \Pr\begin{pmatrix}\forall t \in [1, l_i],\ \mathcal{A}(\boldsymbol{c}^{(i+1)})_t\left[x_{\text{dif}}\right] = 0 \ \textbf{and} \\ \exists t \in [l_i + 1, m_i],\ \text{s.t. } \mathcal{A}(\boldsymbol{c}^{(i+1)})_t\left[x_{\text{dif}}\right] = 1\end{pmatrix} \\
&= p_i' - \Pr\begin{pmatrix}\forall t \in [1, l_i],\ \mathcal{A}(\boldsymbol{c}^{(i+1)})_t\left[x_{\text{dif}}\right] = 0 \ \textbf{and} \\ \exists t \in [l_i + 1, m_i],\ \text{s.t. } \mathcal{A}(\boldsymbol{c}^{(i+1)})_t\left[x_{\text{dif}}\right] = 1\end{pmatrix} \\
&\geqslant p_i' - \frac{3}{2}\Pr\begin{pmatrix}\forall t \in [1, l_i],\ \mathcal{A}(\boldsymbol{c}^{(i)})_t\left[x_{\text{dif}}\right] = 0 \ \textbf{and} \\ \exists t \in [l_i + 1, m_i],\ \text{s.t. } \mathcal{A}(\boldsymbol{c}^{(i)})_t\left[x_{\text{dif}}\right] = 1\end{pmatrix} \\
&= p_i' - \frac{3}{2}q_i' = p_i - q_i - \tfrac{1}{2}q_i' \geqslant p_i - \tfrac{3}{2}q_i,
\end{aligned} \tag{4.3.9}$$

where we used the DP property for the first inequality. Overall, we obtain with our choice of l_{i+1}, r_{i+1}:

$$q_{i+1} \leqslant \frac{3}{4}q_i + \delta/2 \quad \text{and} \quad p_{i+1} \geqslant p_i - 2q_i. \tag{4.3.10}$$

By geometric sum properties, we have that $q_i \leqslant \frac{1}{10}(3/4)^i + 2\delta$, and therefore, $p_i \geqslant 1/2 - 4i\delta$. Finally, note that $r_{i+1} - l_{i+1} = (r_i - l_i)/2 - 1$. In the beginning we have $r_0 - l_0 = 2^k - 2$, therefore $r_i - l_i = 2^{k-i} - 2$. Thus, we can repeat this process $\frac{1}{2}\log T \leqslant k \leqslant 2\log T \leqslant 1/(16\delta)$ times. By construction, all inserted points x_{dif} for $\boldsymbol{c}^{(i)}$ lie on the prefix $[1, l_i - 1]$. In order to extend to other values of ε, we either insert more points per step (if $\varepsilon < \varepsilon_0$), or divide the segment into more parts (if $\varepsilon > \varepsilon_0$). We refer to **??** for the argument in these cases. This concludes the proof. □

4.3.2 *Examples of* β*-*concentrated *online learners for* Point_N

Next, we show that several learners that could be used for online learning Point_N, satisfy Theorem 4.3.2. In particular, in Theorem 4.3.5 we prove that any improper learner that learns Point_N using a finite union of hypothesis in Point_N (defined as Multi-point in Theorem 4.3.4) is concentrated. Theorem 4.3.6 implies that the only existing private online learning algorithm DP-SOA is concentrated.

Definition 4.3.4 (Multi-Point class)**.** For $1 \leqslant K \leqslant N \in \mathbb{N}_+$, we define Point_N^K to be the class of functions that are equal to 1 on at most K points.

Lemma 4.3.5. *Let $\beta > 0$ and $K, T \in \mathbb{N}_+$. For any $N \geqslant 3KT^2/\beta$, any learner $\mathcal{A}$ of Point_N that only uses Point_N^K as its output set is β-*concentrated*.*

Proof. Set $x_{\text{eq}} = 1$ and $f_1 = \mathbb{I}\{\cdot = 2\}$. Let $\tau_0 = \left(\left(x_{\text{eq}}, f_1(x_{\text{eq}})\right), \ldots, \left(x_{\text{eq}}, f_1\left(x_{\text{eq}}\right)\right)\right)$ and $\mathcal{A}(\tau_0) = (\hat{f}_1, \ldots, \hat{f}_T)$. For $t \in [T]$, let $Q_t = \left\{x \in [N], \text{ such that } \Pr\left(\hat{f}_t(x) = 1\right) \geqslant \beta/T\right\}$. We can write

$$\begin{aligned}\sum_{x\in[N]} \Pr(\hat{f}_t(x) = 1) &= \sum_{x\in[N]} \sum_{\substack{f\in \text{Point}_N^K \\ f(x)=1}} \Pr(\hat{f}_t = f) \\ &= \sum_{f\in\text{Point}_N^K} \sum_{\substack{x\in[N] \\ f(x)=1}} \Pr(\hat{f}_t = f) \leqslant K \sum_{f\in\text{Point}_N^K} \Pr(\hat{f}_t = f) = K,\end{aligned} \tag{4.3.11}$$

which implies that $|Q_t| \leqslant K\lceil T/\beta\rceil$. Therefore, by union bound, as long as $N \geqslant 3KT^2/\beta$, there exists $x_{\text{dif}} \in \left([N] \setminus \bigcup_{t=1}^T Q_t\right) \setminus \{1, 2\}$. Applying union bound again, we obtain that

$$\Pr(\exists t, \text{ such that } \hat{f}_t(x_{\text{dif}}) = 1) \leqslant \sum_{t=1}^T \Pr(\hat{f}_t(x_{\text{dif}}) = 1) \leqslant \beta. \tag{4.3.12}$$

Thus, $\mathcal{A}$ is β-concentrated and the distinguishing tuple is $(f_1, \mathbb{I}\{\cdot = x_{\text{dif}}\}, x_{\text{eq}}, x_{\text{dif}})$. □

In Theorem 4.3.5 we assumed that K is a constant. Note that as long as $K = o(N)$, by taking N large enough one can obtain the same result. We also remark that the SOA for Point_N, which is an improper learner, uses Point_N^1 as the output hypothesis class. Beimel *et al.* [23] also uses improper learners consisting of union of points for privately learning Point_N in the pure DP-PAC model; their hypothesis class is a subset of the Multi-Point class.[2]

While their motivation was to explicitly construct a smaller hypothesis class that *approximates* Point_N well, on a more general note, it is also common to use larger hypothesis classes in computational learning theory for improper learning. This is particularly useful for benefits like computational efficiency (e.g., 3-DNF vs 3-CNF) and robustness [65]. The purpose of Theorem 4.3.5 is to show that natural improper learners using larger hypothesis classes for Point_N, like Theorem 4.3.4, are also concentrated. Furthermore, as we show next, the only generic private online learner that we are aware of— DP-SOA, is also concentrated.

Corollary 4.3.6. DP-SOA *for* Point_N *is* β-concentrated *for any* $N \geqslant 3T^2/\beta$.

Proof. Output of DP-SOA is equal to the output of SOA algorithm on some input sequence. Note that SOA for Point_N is always equal to either (i) an all-zero function, or (ii) a target function f^*, which implies that it lies in Point_N^1. Therefore, by Theorem 4.3.5, we obtain that DP-SOA is also β-concentrated. □

Independent work In a concurrent and independent work, Cohen *et al.* [66] established that learners that *are not* concentrated must nevertheless suffer large expected regret for the specific case of Point_T. They construct an adversary which strongly relies on the large size of the function class. In particular, generalizing for a smaller function classes, e.g. Point_3 remains an interesting open question. We obtain a partial progress in this direction, see Section 4.4.2.

4.4 DISCUSSION

2 See Section 4.1 in Beimel *et al.* [23] to see the conditions that the hypothesis class needs to satisfy.

4.4.1 *Connection to DP under continual observation*

Continual observation under DP [29] is the process of releasing statistics continuously on a stream of data while preserving DP over the stream. One of simplest problems in this setting is DP-Continual counting where a counter $\mathcal{C} : \{0,1\}^T \to \mathbb{N}_+^T$ is used. We say $\mathcal{C}$ is deemed a $(T, \alpha, \beta, \varepsilon)$-DP continual counter if $\mathcal{C}$ is ε-DP with respect to its input and with probability at least $1 - \beta$ satisfies

$$\max_{t \leqslant T} |\mathcal{C}(\tau)_t - \sum_{i \leqslant t} \tau_i| \leqslant \alpha. \tag{4.4.1}$$

The proof of Theorem 4.4.1 illustrates a straightforward method to convert DP continual counters to DP online learners for the Point_N hypothesis class, thereby transferring upper bounds from DP continual counting to DP online learning. More precisely, the reduction results in a DP online learning algorithm for Point_N with an additional $\sqrt{N}$ factor to the privacy parameter and number of mistakes bounded by α. We believe this argument can also be extended to other finite classes by adjusting the mistake bound with an additional factor that depends on the size of the class.

Proposition 4.4.1. *For sufficiently small $\varepsilon, \beta \geqslant 0$ and any $\alpha \geqslant 0$, let $\mathcal{C}$ be a $(T, \alpha, \beta, \varepsilon)$-DP continual counter. Then, for any $\delta > 0$, an (ε', δ)-DP online learner $\mathcal{A}$ for Point_N exists ensuring $\mathbb{E}[\mathcal{M}_{\mathcal{A}}] \leqslant \alpha$ with $\varepsilon' = \varepsilon\sqrt{3N \log(1/\delta)}$.*

Several works (see Chan, Shi & Song [30]) have identified counters with $\alpha = O\left((\log T)^{1.5}/\varepsilon\right)$ which immediately implies a mistake bound that also scales as $\text{poly}\log(T)/\varepsilon$ using Theorem 4.4.1. Ignoring[3] the dependence on N, this almost matches the lower bound proposed in Theorem 4.3.3. On the other hand, Dwork *et al.* [68] have shown a lower bound of $\Omega\left(\frac{\log T}{\varepsilon}\right)$ for α for any pure DP-continual counter. However it is not clear how to convert the lower bound for DP-continual learning to DP online learning. This is because 1) eq. (4.4.1) asks for a uniformly (across t) accurate counter whereas the mistake bound in eq. (4.1.1) is a global measure and 2) this lower bound is for pure DP whereas our setting is approximate DP. We leave this question for the future work.

3 We do not optimise this as this dependence is not the focus of this work, for this argument consider $N = O(1)$. However, we believe it can be reduced using a DP continual algorithm for MAXSUM; see Jain *et al.* [67]

4.4.2 *Beyond concentration assumption*

In Section 4.3.1, we analyzed a class of algorithms, namely concentrated algorithms, containing the only existing DP online learning algorithm, DP-SOA, and provided a matching logarithmic in T lower bound. A natural question is whether this lower bound can be extended to *any* learner. While we do not provide a general answer, we show that our construction can be made more general. Here, we analyze a different class of algorithms, called *firing algorithms*.

For simplicity of exposition, consider the hypothesis class Point_3, and let $f^* \in \left\{f^{(1)}, f^{(2)}\right\}$. Let x^* be such that $f^* = f^{(x^*)}$ and assume that adversary only considers sequences consisting of $(x^*, 1)$ and $(3, 0)$. Let $\mathcal{D}$ be a distribution on $\left\{\text{all-zero function}, f^{(1)}, f^{(2)}\right\}$. At each point t, a firing algorithm $\mathcal{A}$ computes $p_t = p_t(\#\text{mistakes up until } t) \in [0, 1]$ with $p_t(0) = 0$, and samples a $\xi_t \sim \text{Bern}(p_t)$. If $\xi_t = 1$, the algorithm commits to the correct hypothesis henceforth; otherwise, it outputs $f_t \sim \mathcal{D}$. Note that if the adversary introduces $(3, 0)$ at step t, $\mathcal{A}$ is guaranteed not to err, and a single mistake suffices to identify (non-privately) the correct hypothesis.

When $\mathcal{D}$ has support on only one of $\left\{\text{all-zero function}, f^{(1)}, f^{(2)}\right\}$, it yields a 0-concentrated algorithm. Furthermore, the continual observation algorithm can also be viewed as a firing algorithm with a proper choice of $\mathcal{D}$. We analyze the opposite to the concentrated algorithms, in particular when $\mathcal{D} = \text{Unif}\left\{f^{(1)}, f^{(2)}\right\}$. We call such learners *uniform firing algorithms* and we also obtain a logarithmic in T lower bound for them.

Proposition 4.4.2. *Let $\mathcal{A}$ be an (ε, δ)-DP uniform firing algorithm for class Point_3. Then, if $\log T = O(1/\delta)$, there exists an adversary, such that $\mathbb{E}[M_{\mathcal{A}}] = \Omega(\log T)$.*

Proof is provided in **??** and is similar to the proof of Theorem 4.3.3, but requires a more delicate construction.

4.4.3 *Pure differentially private online learners*

While we have so far focused on Approximate DP with $\delta > 0$, in this section we briefly discuss Online learning under pure DP. Note that Theorem 4.3.5 and Theorem 4.3.3 immediately imply the following lower bound on pure differentially private online learners for Point_N.

Corollary 4.4.3. *Let $\varepsilon > 0$, $\beta > 0$, $K, T \in \mathbb{N}_+$ and $N \geqslant 3KT^2/\beta$. For any ε-DP learner $\mathcal{A}$ which uses only Point_N^K as its output set, there exists an adversary $\mathcal{B}$, such that*

$$\mathbb{E}\left[M_{\mathcal{A}}\right] = \Omega\left(\min\left(\frac{\log T/\beta}{\varepsilon}, T\right)\right). \tag{4.4.2}$$

Proof. Theorem 4.3.5 implies that $\mathcal{A}$ must be β-concentrated. Furthermore, since $\mathcal{A}$ is ε-DP, it is also (ε, β)-DP, and, thus, the existence of adversary with large regret follows from Theorem 4.3.3. □

For (ε, δ)-DP online algorithms, there exists an upper bound provided by Golowich & Livni [27]. However, to the best of our knowledge, not much is known about ε-DP online algorithms. One way to obtain such algorithm is by leveraging existing results from the continual observation literature as done in Theorem 4.4.1. Under the same assumptions as in Theorem 4.4.1, using basic composition instead of advanced composition in the last step results in an ε' scaling as $N\varepsilon$. However, this only works for Point_N and not for more general classes. For arbitrary finite hypothesis classes, it is possible to use a DP continual counter, similar to above to obtain a mistake bound that also scales with the size of the class. We also remark that the AboveThreshold algorithm could be used to design learners for certain function classes. However, the question of whether generic learners can be designed remains open. Another interesting open question is whether the dependence on the size of the class, e.g., N for Point_N, is necessary.

4.4.4 *Open problems*

This work relies on assumptions about properties (Concentrated Assumption in Theorem 4.3.2 or uniform firing Assumption in Section 4.4.2) of the learning algorithms to show a lower bound for any hypothesis class. On the other hand, Cohen *et al.* [66] do not need any assumption on their algorithm but their result only holds for large Point classes (and it cannot immediately be transferred to small Point classes, e.g. Point_3). To fully settle the problem of lower bounds for online DP learners, the limitations of both this work and that of Cohen *et al.* [66] need to be addressed. We propose the following conjecture which we believe is true.

Conjecture 4.4.4 (Lower bound for Approximate DP)**.** *For any $\varepsilon, \delta > 0$ and any (ε, δ)-DP learner $\mathcal{A}$ of Point_3, there exists an adversary $\mathcal{B}$, such that* $\mathbb{E}\left[M_{\mathcal{A}}\right] = \Omega\left(\min\left(\frac{\log T/\delta}{\varepsilon}, 1/\delta\right)\right)$.

A straightforward implication of our main result is that the $\log T$ lower bound also holds for pure DP online learners under the same assumptions on the algorithm. In Section 4.4.1, we showed a generic reduction from DP continual counters to DP online learners for Point_N with regret $\text{poly}\log(T)$. This reduction can be extended to pure DP online learner by using basic composition in Theorem 4.4.1, with an additional cost of $\sqrt{N}$. In Theorem 4.4.5, we raise the question whether for small N, the dependence on T can be lowered to $\log T$ and shown to be tight.

Conjecture 4.4.5 (Upper and lower bounds for Pure DP)**.** *For any $\varepsilon > 0$,*

1. Upper bound: *there exists ε-DP learner of* Point_3*, s.t. for any adversary,* $\mathbb{E}[M] = O\left(\frac{\log T}{\varepsilon}\right)$.
2. Lower bound: *for any ε-DP learner of* Point_3*, there exists an adversary, s.t.* $\mathbb{E}[M] = \Omega\left(\frac{\log T}{\varepsilon}\right)$.

Note that the lower bound part of Theorem 4.4.5 follows from Theorem 4.4.4, but can also be viewed through connection to the continual observation model. In the latter regime, an $\Omega(\log T)$ lower bound was shown in Dwork, Roth, *et al.* [69].

dd

5

ROBUST MIXTURE LEARNING WHEN OUTLIERS OVERWHELM SMALL GROUPS

5.1 INTRODUCTION

Estimating the mean of a distribution from empirical data is one of the most fundamental problems in statistics. The mean often serves as the primary summary statistic of the dataset or is the ultimate quantity of interest that is often not precisely measurable. In practical applications, data frequently originates from a mixture of multiple groups (also called subpopulations) and a natural goal is to estimate the distinct means of each group separately. For example, we might like to use representative individuals to study how a complex decision or procedure would impact different subpopulations. In other applications, such as genetics [70] or astronomy [71] research, finding the means themselves can be a crucial first step towards scientific discovery. In both scenarios, the algorithm should output a list of estimates that are close to the unobservable true means.

However, in practice, the data may also contain outliers, for example due to measurement errors or abnormal events. We would like to find good mean estimates for all inlier groups even when the proportion of such *additive adversarial contaminations* is larger than some smaller groups that we want to properly represent. The central open question that motivates our work is thus:

What is the cost of efficiently recovering small groups that may be outnumbered by outliers?

More specifically, consider a scenario where the practitioner would like to recover the means of small but significant enough inlier groups which constitute at least $w_{\text{low}} \in (0, 1)$ proportion of the (corrupted) data. If k is the number of such inlier groups, for all $i \in [k]$, we then denote by $w_i \geqslant w_{\text{low}}$ the unknown weight of the $i-$th group with mean μ_i. Further, we use ε to refer to the proportion of additive contamination – the data that comes from an unknown adversarial distribution. The goal is to estimate the unknown means μ_i for all $i \in [k]$.

Existing works on robust mixture learning such as [31, 32] consider the problem when the fraction of additive adversarial outliers is smaller than

the weight of the smallest subgroup, i.e. $\varepsilon < w_{\text{low}}$. However, for large outlier proportions where $\varepsilon \geqslant w_{\text{low}}$, these algorithms are not guaranteed to recover small clusters with $w_i \leqslant \varepsilon$. In this case, outliers can form additional spurious clusters that are indistinguishable from small inlier groups. As a consequence, generating a list of size equal to the number of components would possibly lead to neglecting the means of small groups. In order to ensure that the output contains a precise estimate for each of the small group means, it is thus necessary the estimation algorithm to provide a list whose size is strictly larger than the number of components. We call this paradigm *list-decodable mixture learning* (LD-ML), following the footsteps of a long line of work on list-decodable learning (see Sections 5.2 and 5.5).

Specifically, the main challenge in LD-ML is to provide a *short* list that contains good mean estimates for all inlier groups. We first note that there is a minimum list size the algorithm necessarily has to output to guarantee that all groups are recovered. For example, consider an outlier distribution that includes several copies of the smallest inlier group distribution with means spread out throughout the domain. Since inlier groups are indisntinguishable from spurious outlier ones, the shortest list that includes means of all inlier groups must be of size at least $|L| \geqslant k + \frac{\varepsilon}{\min_i w_i}$. Here, $\frac{\varepsilon}{\min_i w_i}$ can be interpreted as the minimal list-size overhead that is necessary due to "caring" about groups with weight smaller than ε. The key question is hence how good the error guarantees of an LD-ML algorithm can be when the list size overhead stays close to $\frac{\varepsilon}{\min w_i}$, while being agnostic to w_i aside from the knowledge of w_{low}. Furthermore, we are interested in *computationally efficient* algorithms for LD-ML, especially when dealing with high-dimensional data.

To the best of our knowledge, the only existing efficient algorithms that are guaranteed to recover inlier groups with weights $w_i \leqslant \varepsilon$ are *list-decodable mean estimation* (LD-ME) algorithms. LD-ME algorithms model the data as a mixture of one inlier and outlier distribution with weights $\alpha \leqslant 1/2$ and $1 - \alpha$ respectively. Provided with the weight parameter α, they output a list that contains an estimate close to the inlier mean with high probability. However, for the LD-ML setting, the inlier weights w_i are not known and we would have to use LD-ME algorithms with w_{low} as weight estimates for each group. This leads to suboptimal error in particular for large groups, that hence (somewhat counter intuitively) would have to "pay" for the explicit constraint to recover small groups. Furthermore, even if LD-ME were provided with w_i, by design it would treat inlier points from other

Type of inlier mixture	Best prior work	Ours	Inf.-theor. lower bound
Large groups	$\widetilde{O}(\varepsilon/w_i)$	$\widetilde{O}(\varepsilon/w_i)$	$\Omega(\varepsilon/w_i)$
Small groups	$O\left(\sqrt{\log \frac{1}{w_{\text{low}}}}\right)$	$O\left(\sqrt{\log \frac{\varepsilon+w_i}{w_i}}\right)$	$\Omega\left(\sqrt{\log \frac{\varepsilon+w_i}{w_i}}\right)$
Non-separated groups	$O\left(\sqrt{\log \frac{1}{w_{\text{low}}}}\right)$	$O\left(\sqrt{\log \frac{1}{w_i}}\right)$	$\Omega\left(\sqrt{\log \frac{1}{w_i}}\right)$

TABLE 5.1: For a mixture of Gaussian components $\mathcal{N}(\mu_i, I_d)$, we show upper and lower bounds for the **error of the i-component** given a output list L (of the respective algorithm) $\min_{\hat{\mu}\in L}\|\hat{\mu}-\mu_i\|$. When the error doesn't depend on i, all means have the same error guarantee irrespective of their weight. Note that depending on the type of inlier mixture, different methods in [31] are used as the 'best prior work': robust mixture learning for the first row and list-decodable mean estimation for the rest. 'Large groups' means that $\forall j : \varepsilon \leqslant w_j$, 'small groups' means $\exists j : \varepsilon \geqslant w_j$. In both cases, mixture components are assumed to be separated. For lower bounds, see [33, 34], and Prop. 5.3.5

components also as outliers, unnecessarily inflating the fraction of outliers to $1 - w_i$ instead of ε.

CONTRIBUTIONS In this paper, we propose an algorithm that (i) correctly estimates the weight of each component only given a lower bound and (ii) does not overestimate proportion of outliers when components are well-separated. In particular, we construct a meta-algorithm that uses mean estimation algorithms as base learners that are designed to deal with adversarial corruptions. This meta-algorithm inherits guarantees from the base learner and any improvement of the latter translates to better results for LD-ML. For example, if the base learner runs in polynomial time, so does our meta-algorithm. Our approach of using the output of weak base learners to achieve better performance is reminiscent of the *boosting* paradigm that is common in machine learning practice.

Our algorithm achieves significant improvements in error and list-size guarantees for multiple settings. For ease of comparison, we summarize error improvements for inlier Gaussian mixtures in Table 5.1. The main focus of our contributions is represented in the second row; that is the setting where outliers outnumber some inlier groups with weight $w_j \leqslant \varepsilon$

and the inlier components are *well-separated*, i.e., $\|\mu_i - \mu_j\| \gtrsim^1 \sqrt{\log \frac{1}{w_{\text{low}}}}$, where μ_i's are the inlier component means. As we mentioned before, robust mixture learning algorithms, such as [32, 35], are not applicable here and the best error guarantees in prior work is achieved by an LD-ME algorithm, e.g. from [31]. While its error bounds are of order $O(\sqrt{\log \frac{1}{w_{\text{low}}}})$ for a list size of $O(\frac{1}{w_{\text{low}}})$, our approach guarantees error $O(\sqrt{\log \frac{\varepsilon}{w_i}})$ for a list size of $k + O(\frac{\varepsilon}{w_{\text{low}}})$. Remarkably, we obtain the same error guarantees as if an oracle would run LD-ME on each inlier group *with the correct weight* w_i separately (with outliers). Hence, the only cost for recovering small groups is the increased list-size overhead of order $O(\frac{\varepsilon}{w_{\text{low}}})$. Further, a sub-routine in our meta-algorithm also obtains novel guarantees under *no* separation assumption, as shown in the third row of Table 5.1. This algorithm achieves the same error guarantees for similar list size as a base learner that knows the correct weights of the inlier components.

Based on a reduction argument from LD-ME to LD-ML, we also provide information-theoretic (IT) lower bounds for LD-ML. If the LD-ME base learners achieve the IT lower bound (possible for inlier Gaussian mixtures), so does our LD-ML algorithm. In synthetic experiments, we implement our meta-algorithm with the LD-ME base learner from [36] and show clear improvements compared to the only prior method with guarantees, while being comparable or better than popular clustering methods such as k-means and DBSCAN for various attack models.

5.2 SETTINGS

We now introduce the learning settings that appear in the paper. Let $d \in \mathbb{N}_+$ be the ambient dimension of the data and $k \in \mathbb{N}_+$ be the number of mixture components (inlier groups/clusters).

5.2.1 *List-decodable mixture learning under adversarial corruptions*

We focus on mixtures that consist of distributions that are sufficiently bounded in the following sense.

1 We adopt the following standard notation: $f \lesssim g$, $f = O(g)$, and $g = \Omega(f)$ mean that $f \leqslant Cg$ for some universal constant $C > 0$. $\widetilde{O}$-notation hides polylogarithmic terms.

Definition 5.2.1. Let $t \in \mathbb{N}_+$ be even and let $D(\mu)$ be a distribution on $\mathbb{R}^d$ with mean μ. We say that $D(\mu)$ has *sub-Gaussian t-th central moments* if for all even $s \leqslant t$ and for every $v \in \mathbb{R}^d$ with $\|v\| = 1$, $\mathbb{E}_{x\sim D} \langle x - \mu, v\rangle^s \leqslant (s-1)!!$.

This class of distributions is closely related to commonly studied distributions in the literature (see, e.g., [33]) with bounded t-th moment. Our requirement for the boundedness of all moments $s \leqslant t$ stems from the fact that our algorithm should adapt to unknown and possibly non-uniform mixture weights.

We assume that we are given samples from a corrupted d-dimensional mixture of k inlier distributions $D_i(\mu_i)$ satisfying Theorem 5.2.1, where the mixture is defined as

$$\mathcal{X} = \sum_{i=1}^{k} w_i D_i(\mu_i) + \varepsilon Q, \tag{5.2.1}$$

and $\sum_{i=1}^{k} w_i + \varepsilon = 1$, where for all $i = 1, \ldots, k$, it holds that $w_i \geqslant w_{\text{low}}$. Further, an $\varepsilon > 0$ proportion of the data comes from an *outlier* distribution Q chosen by the adversary with full knowledge of our algorithm and inlier mixture. Samples drawn from $D_i(\mu_i)$ constitute the i^{th} *inlier cluster*. The goal in mixture learning under corruptions as in eq. (5.2.1), is to design an algorithm that takes in i.i.d. samples from $\mathcal{X}$ and outputs a list L, such that for each $i \in [k]$, there exists $\hat{\mu} \in L$ with small estimation error $\|\mu_i - \hat{\mu}\|$.

To the best of our knowledge, we are the first to study the *list-decodable mixture learning* problem (LD-ML) that considers the case of large fractions of outliers $\varepsilon \geqslant \min_i w_i$ and the goal is to achieve small estimation errors while the list size $|L|$ remains small. While in robust estimation problems, the fractions of inliers and outliers are usually provided to the algorithm, in mixture learning, the mixture proportions are explicit quantities of interest. Throughout the paper, we hence assume that *both* the true weights w_i of the mixture and the fraction of outliers ε are *unknown*. Instead, by definition in eq. (5.2.1), we assume knowledge of a valid lower bound $w_{\text{low}} \leqslant \min_i w_i$.

Note that when $\varepsilon \lesssim \min_i w_i$, the problem is known as robust mixture learning and can be solved with list size $|L| = k$ as discussed in [31, 32, 35]. However, algorithms for robust mixture learning fail when the fraction of outliers becomes comparable to the inlier group size. In the presence of "spurious" adversarial clusters, it is information-theoretically impossible to output a list L, such that (i) $|L| = k$ and (ii) L contains precise estimate for each true mean.

5.2.2 *Mean estimation under adversarial corruptions*

In order to solve LD-ML, we use mean estimation procedures that have provable guarantees under adversarial contamination. Mean estimation can be viewed as a particular case of the mixture learning problem in eq. (5.2.1) with $k = 1$, the fraction of inliers $\alpha = w_1$ and the fraction of outliers $\varepsilon = 1 - \alpha$. The mean estimation algorithms we use to solve LD-ML with w_{low} need to exhibit guarantees under a stronger adversarial model, where the adversary can also replace a small fraction (depending on w_{low}) of the inlier points; see details in Theorem B.2.1. This is a special case of the general contamination model as opposed to the slightly more benign additive contamination model in eq. (5.2.1). For different regimes of α we use black-box learners that solve corresponding regime when *provided with* α.

ROBUST MEAN ESTIMATION When the majority of points are inliers, we are in the RME setting. Robust statistics has studied this setting with different corruption models and efficient algorithms are known to achieve information-theoretically optimal error guarantees (see Section 5.5).

LIST-DECODABLE MEAN ESTIMATION When inliers form a minority, we are in the list-decodable setting and are required to return a list instead of a single estimate. We refer to this setting as cor-kLD (*corrupted known list-decoding*). For mixture learning, α is usually unknown and we need to solve the cor-aLD (*corrupted agnostic list-decoding*) problem (i.e., α is *not provided*, but instead a lower bound $\alpha_{\text{low}} \in [w_{\text{low}}, \alpha]$ is given to the algorithm). Finally, when only additive adversarial contamination is present, as in eq. (5.2.1), we recover the standard list-decoding setting studied in prior works (see Section 5.5) that we call sLD (*simple list-decoding*). In Appendix B.7 we show that two algorithms designed for sLD also exhibit guarantees for cor-kLD for any w_{low}.

5.3 MAIN RESULTS

We now present our main results for list-decodable and robust mixture learning defined in Section 5.2. In Section 5.3.1, we provide algorithmic upper bounds and information-theoretic lower bounds. For the special case of spherical Gaussian mixtures, we show in Section 5.3.1 that we achieve

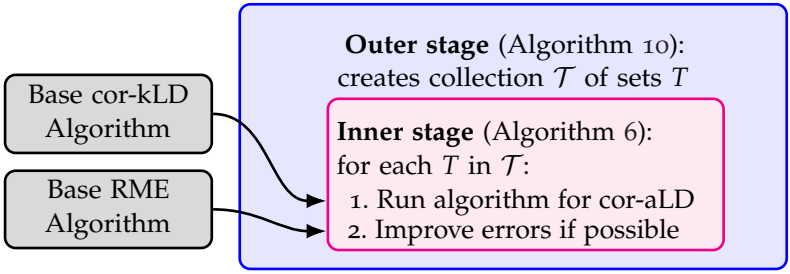

FIGURE 5.1: Schematic of the meta-algorithm (Algorithm 4) underlying Theorem 5.3.3

optimality. Our results are constructive as we provide a meta-algorithm for which these bounds hold.

As depicted in Figure 5.1, our meta-algorithm (Algorithm 4) is a two-stage process. The outer stage (Algorithm 10) reduces the problem to mean estimation by leveraging the mixture structure and splitting the data into a small collection $\mathcal{T}$ of sets T. Each set $T \in \mathcal{T}$ should (i) contain at most one inlier cluster (and few samples from other clusters) and (ii) the total number of outliers across all sets should be at most $O(\varepsilon n)$. We then run the inner stage (Algorithm 6) on sets T, which outputs a mean estimate for the inlier cluster in T. First, a cor-aLD algorithm identifies the weight of the inlier cluster and returns the result of a cor-kLD base learner with this weight. Then, if the weight is large, we improve the error via an RME base learner. A careful filtering procedure in both stages achieves the significantly reduced list size and better error guarantees. We require the base learners to satisfy the following set of assumptions.

Assumption 5.3.1 (Mean-estimation base learners for mixture learning)**.** Let t be an even integer and consider the corruption setting defined in Theorem B.2.1. Further, let the inlier distribution $D(\mu^*) \in \mathcal{D}$ where $\mathcal{D}$ is the the family of distributions satisfying Theorem 5.2.1 for t. We assume that

(a) for $\alpha \in [w_{\text{low}}, 1/3]$ in the cor-kLD regime, there exists an algorithm $\mathcal{A}_{\text{kLD}}$ that uses $N_{LD}(\alpha)$ samples and $T_{LD}(\alpha)$ time to output a list of size bounded by $1/\alpha^{O(1)}$ that with probability at least $1/2$ contains some $\hat{\mu}$ with $\|\hat{\mu} - \mu^*\| \leqslant f(\alpha)$, where f is non-increasing.

(b) for $\alpha \in [1 - \varepsilon_{\text{RME}}, 1]$, with $0 \leqslant \varepsilon_{\text{RME}} \leqslant 1/2 - 2w_{\text{low}}^2$ in the RME regime, there exists an RME algorithm $\mathcal{A}_R$ that uses $N_R(\alpha)$ samples

and $T_R(\alpha)$ time to output with probability at least $1/2$ some $\hat{\mu}$ with $\|\hat{\mu} - \mu^*\| \leqslant g(\alpha)$, where g is non-increasing.

Note that the sample and time-complexity functions such as N_{LD} and T_{LD}, might depend on t, for example growing as d^t. We emphasize that (i) the guarantees of our meta-algorithm depend on the guarantees of the base learners and (ii) we only require the base learners to work in the well-studied setting with *known* fraction of inliers. Corollary 5.3.4 uses known base learners for Gaussian distributions achieving information-theoretically optimal error bounds. There also exists base learners for distributions beyond Gaussians, such as bounded covariance or log-concave distributions, see, e.g. [72].

5.3.1 *Upper bounds for list-decodable mixture learning*

Key quantities that appear in our error bounds are the relative proportion of inliers $\tilde{w}_i$ and outliers $\tilde{\varepsilon}_i$:

$$\tilde{w}_i = \frac{w_i}{w_i + \varepsilon + w_{\text{low}}^2} \quad \text{and} \quad \tilde{\varepsilon}_i = 1 - \tilde{w}_i. \tag{5.3.1}$$

These quantities reflect that each set T in the inner stage contains at most one inlier cluster and a small ($\lesssim w_{\text{low}}^2$) fraction of points from other inlier clusters. We now present a simplified version of our main result in Theorem 5.3.3 (see Theorem B.3.1 for the detailed result) that allows for a more streamlined presentation of the results using the following 'well-behavedness' of f and g.

Assumption 5.3.2. Let f, g be as defined in Theorem 5.3.1. For some $C > 0$, we assume (i) $\varepsilon_{\text{RME}} \geqslant 0.01$, (ii) $\forall x \in (0, 1/3]$, $f(x/2) \leqslant Cf(x)$, and (iii) $\forall x \in [0.99, 1]$, $g(x - (1-x)^2) \leqslant Cg(x)$.

We are now ready to state the main result of the paper.

Theorem 5.3.3. *Let $d, k \in \mathbb{N}_+$, $w_{\text{low}} \in (0, 1/2]$, and t be an even integer. Let $\mathcal{X}$ be a d-dimensional mixture distribution following eq. (5.2.1). Let $\mathcal{A}_{\text{kLD}}$ and $\mathcal{A}_R$ satisfy Theorems 5.3.1 and 5.3.2 for some even t. Further, suppose that $\|\mu_i - \mu_j\| \gtrsim \sqrt{t}(1/w_{\text{low}})^{4/t} + f(w_{\text{low}})$ for all $i \neq j \in [k]$.*

Then there exists an algorithm that, given $\text{poly}(d, 1/w_{\text{low}}) \cdot (N_{LD}(w_{\text{low}}) + N_R(w_{\text{low}}))$ i.i.d. samples from $\mathcal{X}$ as well as d, k, w_{low}, and t, runs in time $\text{poly}(d, 1/w_{\text{low}}) \cdot (T_{LD}(w_{\text{low}}) + T_R(w_{\text{low}}))$ and

with probability at least $1 - w_{\text{low}}^{O(1)}$ *outputs a list* L *of size* $|L| \leqslant k + O(\varepsilon/w_{\text{low}})$ *where, for each* $i \in [k]$*, there exists* $\hat{\mu} \in L$ *such that*

$$\|\hat{\mu} - \mu_i\| = O\left(\min_{1 \leqslant t' \leqslant t} \sqrt{t'}(1/\tilde{w}_i)^{1/t'} + f(\min(\tilde{w}_i, 1/3))\right).$$

If the relative weight of the i-th cluster is large, i.e., $\tilde{\varepsilon}_i \leqslant 0.001$*, then the error is further bounded by*

$$\|\hat{\mu} - \mu_i\| = O\left(g(\tilde{w}_i)\right).$$

The proof together with a more general statement, Theorem B.3.1, can be found in Appendix B.3.

Note that for a mixture setting with $k \geqslant 2$, the assumption $w_{\text{low}} \leqslant 1/k \leqslant 1/2$ is automatically fulfilled. Also, for large weights $\tilde{w}_i$ such that $\log(1/\tilde{w}_i) \ll t$, the t' that minimizes $\sqrt{t'}(1/\tilde{w}_i)^{1/t'}$ is smaller than t, and for small weights the minimizer is $t' = t$.

GAUSSIAN CASE For Gaussian inlier distributions, LD-ME and RME base learners with guarantees for Theorem 5.3.1 have already been developed in prior work. We can thus readily use them in the meta-algorithm to arrive at the following statement with the relative proportions defined in eq. (5.3.1).

Corollary 5.3.4 (Gaussian case)**.** *Let* d, k, w_{low} *and* t *be as in Theorem 5.3.3. Let* $\mathcal{X}$ *be as in eq.* (5.2.1) *with* $D_i(\mu_i) = \mathcal{N}(\mu_i, I_d)$ *with* μ_i*'s satisfying* $\|\mu_i - \mu_j\| \gtrsim \sqrt{\log 1/w_{\text{low}}}$ *for all* $i \neq j \in [k]$*. There exists an algorithm that for* $t = O(\log 1/w_{\text{low}})$*, given* $N = \text{poly}(d^t, (1/w_{\text{low}})^t)$ *i.i.d. samples from* $\mathcal{X}$ *and* w_{low}*, runs in* $\text{poly}(N)$ *time and outputs a list* L *such that with high probability* $|L| = k + O(\varepsilon/w_{\text{low}})$ *and, for all* $i \in [k]$*, there exists* $\hat{\mu} \in L$ *such that*

$$\|\hat{\mu} - \mu_i\| = O\left(\sqrt{\log 1/\tilde{w}_i}\right).$$

If the relative weight of the i-th cluster is large, i.e. $\tilde{\varepsilon}_i \leqslant 0.001$*, then the error is further bounded by*

$$\|\hat{\mu} - \mu_i\| = O\left(\tilde{\varepsilon}_i \sqrt{\log 1/\tilde{\varepsilon}_i}\right).$$

Proof. Theorem 6.12 from [33] provides an LD-ME algorithm $\mathcal{A}_{\text{kLD}}$ achieving error $f(\alpha) \leqslant O(\sqrt{t'}(1/\alpha)^{1/t'})$ for all $t' \leqslant t$. The sample and time complexity scale as $\text{poly}(d^t, (1/\alpha)^t)$. Also, Theorem 5.1 from [73] provides a robust mean estimation algorithm $\mathcal{A}_R$ such that for a small enough constant fraction of outliers $\varepsilon = 1 - \alpha$ it achieves error $g(\alpha) = O((1-\alpha)\sqrt{\log 1/(1-\alpha)})$ with sample complexity $\tilde{\Omega}(d/\varepsilon^2)$. Using these $\mathcal{A}_{\text{kLD}}$ and $\mathcal{A}_R$, we recover the desired bounds. □

COMPARISON WITH PRIOR WORK We now compare our result with the only previous method that can achieve guarantees in the LD-ML setting with unknown w_i. As discussed in [31], algorithms for the simple list-decoding model with $\alpha = w_{\text{low}}$ can be used for LD-ML by viewing a single mixture component as the "ground truth" distribution and effectively treating all other inlier components and original outliers as outliers. Besides requiring a much larger list size of $O(1/w_{\text{low}}) \gg k + O(\varepsilon/w_{\text{low}})$ and error $O(\sqrt{\log 1/w_{\text{low}}})$, this approach has two drawbacks that manifest in the suboptimal guarantees: 1) For larger clusters i with $w_i \gg w_{\text{low}}$, LD-ME only achieves an error $O\left(\sqrt{\log 1/w_{\text{low}}}\right)$. Our result, even without separation assumption, achieves a sharper error bound $O\left(\sqrt{\log 1/w_i}\right)$. 2) When the mixture is separated, LD-ME cannot exploit the structure since it still models the data as $w_{\text{low}}\mathcal{N}(\mu_i, I_d) + (1 - w_{\text{low}})Q$ for each i, so that the algorithm inevitably treats all other true components as outliers. This results in the error $O\left(\sqrt{\log 1/w_{\text{low}}}\right) \gg O\left(\sqrt{\log 1/\tilde{w}_i}\right) = O(1)$ (when $\varepsilon \sim w_i \ll 1$). We refer to Appendix B.1 for further illustrative examples. As a simple example, consider the uniform inlier mixture with $\varepsilon = w_i = 1/(k+1)$, where k is large. In this case, previous results have error guarantees $O(\sqrt{\log k})$, while we obtain error $O(1)$.

5.3.2 *Information-theoretical lower bounds and optimality*

Next, we present information-theoretical lower bounds for list-decodable mixture learning on well-separated distributions $\mathcal{X}$ as defined in eq. (5.2.1). We show that our error is optimal as long as the list size is required to be small. Our proof uses a simple reduction technique and leverages established lower bounds in [31] for the list-decodable mean estimation model (sLD in Section 5.2).

Proposition 5.3.5 (Information-theoretic lower bounds)**.** *Let $\mathcal{A}$ be an algorithm that, given access to $\mathcal{X}$, outputs a list L that, with probability $\geqslant 1/2$, for each $i \in [k]$ contains $\hat{\mu} \in L$ with $\|\hat{\mu} - \mu_i\| \leqslant \beta_i$.*

(a) Consider the case with $\|\mu_i - \mu_j\| \gtrsim (1/w_{\text{low}})^{4/t}$ for $i \neq j \in [k]$, $D_i(\mu_i)$ having t-th bounded sub-Gaussian central moments and $\beta_i \leqslant C(1/w_{\text{low}})^{1/t}$ for each $i \in [k]$. If for some $s \in [k]$ it holds that $w_s \leqslant \varepsilon$, then algorithm $\mathcal{A}$ must either have error bound $\beta_s = \Omega((1/\tilde{w}_i)^{1/t})$ or $|L| \geqslant k + d - 1$.

(b) Consider the case with $\|\mu_i - \mu_j\| \gtrsim \sqrt{\log 1/w_{\text{low}}}$ for $i \neq j \in [k]$, $D_i(\mu_i) = \mathcal{N}(\mu_i, I_d)$ and $\beta_i \leqslant C\sqrt{\log 1/w_{\text{low}}}$ for each $i \in [k]$. If for some $s \in [k]$

it holds that $w_s \leqslant \varepsilon$*, then algorithm* $\mathcal{A}$ *must either have error bound* $\beta_s = \Omega(\sqrt{\log 1/\tilde{w}_i})$ *or* $|L| \geqslant k + \min\{2^{\Omega(d)}, (1/\tilde{w}_i)^{\omega(1)}\}$.

In the Gaussian inlier case, Theorem 5.3.4 together with Theorem 5.3.5 imply optimality of our meta-algorithm. Indeed, if one plugs in optimal base learners (as in the proof of Theorem 5.3.4), we obtain error guarantee that matches lower bound. In particular, "exponentially" larger list size is necessary for asymptotically smaller error. For inlier components with bounded sub-Gaussian moments, [31] obtains information-theoretically (nearly-)optimal LD-ME base learners.

We remark that for the problem of learning mixture models, the separation assumption is common in the literature [31, 72, 74, 75]. Without the separation assumption, even in the *noiseless* uniform case $w_i = 1/k$, [34] shows that no efficient algorithm can obtain error asymptotically better than $\Omega(\sqrt{\log 1/w_i})$. In Theorem B.2.5, we prove that the inner stage Algorithm 6 of our algorithm, without knowledge of w_i and separation assumption, achieves with high probability matching error guarantees $O(\sqrt{\log 1/w_i})$ with a list size upper bound $O(1/w_{\text{low}})$.

Furthermore, in [31], formal evidence of computational hardness was obtained (see their Theorem 5.7, which gives a lower bound in the statistical query model introduced by [76]) that suggests obtaining error $\Omega_t((1/\tilde{w}_s)^{1/t})$ requires running time at least $d^{\Omega(t)}$. This was proved for Gaussian inliers and the running time matches ours up to a constant in the exponent.

5.4 ALGORITHM SKETCH

We now sketch our meta-algorithm specialized to the case of separated Gaussian components $\mathcal{N}(\mu_i, I_d)$ and provide intuition for how it achieves the guarantees in Theorem 5.3.4. In this section, we only discuss how to obtain an error of $O(\sqrt{\log 1/\tilde{w}_i})$ for each mean when $\varepsilon \gtrsim \min_i w_i$. We refer to Appendix B.4 for how to achieve the refined error guarantee of $O(\tilde{\varepsilon}_i \sqrt{\log 1/\tilde{\varepsilon}_i})$ when $\tilde{\varepsilon}_i$ is small.

As discussed in Section 5.3.1, running an out-of-the-box LD-ME algorithm for the sLD problem on our input with parameter $\alpha = w_{\text{low}}$ would give sub-optimal guarantees. In contrast, our two-stage Algorithm 4, equipped with the appropriate cor-kLD and RME base learners as depicted in Figure 5.1, obtains for each component an error guarantee that is as good as if we had access to the samples *only* from this component and from the outliers. We now give more details about the outer stage, Algorithm 5, and inner

Algorithm 4 FullAlgorithm

Input: Samples $S = \{x_1, \dots, x_n\}$, w_{low}, algorithms $\mathcal{A}_{\text{kLD}}$, and $\mathcal{A}_R$.
Output: List L.

1: Run OuterStage (Algorithm 10) on S and let $\mathcal{T}$ be the returned list.
2: $L \leftarrow \emptyset$.
3: **for** $T \in \mathcal{T}$ **do**
4: Run InnerStage (Algorithm 6) on T with $\alpha_{\text{low}} = w_{\text{low}} \cdot \frac{n}{|T|}$.
5: Add the elements of the returned list to L.
6: **return** L.

stage, Algorithm 6, and describe on a high-level how they contribute to a short output list with optimal error bound in Theorem 5.3.4 for large outlier fractions.

5.4.1 *Inner stage: list-decodable mean estimation with unknown inlier fraction*

We now describe how to use a black-box cor-kLD algorithm to obtain a list-decoding algorithm $\mathcal{A}_{\text{aLD}}$ for the cor-aLD mean-estimation setting with access only to $\alpha_{\text{low}} \leqslant \alpha$. $\mathcal{A}_{\text{aLD}}$ is used in the proof of Theorem B.2.5 and plays a crucial role (see Figure 5.1) in our meta-algorithm. In particular, it deals with the unknown weight of the inlier distribution in each set returned by the outer stage. Note that estimating α from the input samples is impossible by nature. Indeed, we cannot distinguish between potential outlier clusters of arbitrary proportion $\leqslant 1 - \alpha$ and the inlier component. Underestimating the size of a large component would inevitably lead to a suboptimal error guarantee. We now show how to overcome this challenge and achieve an error guarantee $O(\sqrt{\log 1/\alpha})$ for a list size $1 + O((1 - \alpha)/\alpha_{\text{low}})$ for the cor-aLD setting. Here we only outline our algorithm and refer to Appendix B.4 for the details.

Algorithm 6 first produces a large list of estimates corresponding to many potential values of α and then prunes it while maintaining a good estimate in the list. In particular, for each $\hat{\alpha} \in A := \{\alpha_{\text{low}}, 2\alpha_{\text{low}}, \dots, \lfloor 1/(3\alpha_{\text{low}}) \rfloor \alpha_{\text{low}}\}$, we run $\mathcal{A}_{\text{kLD}}$ with parameter $\hat{\alpha}$ to obtain a list of means. We append $\hat{\alpha}$ to each mean in the list and obtain a list of pairs $(\hat{\mu}, \hat{\alpha})$. We concatenate these lists of pairs for all $\hat{\alpha}$ and obtain a list L of size $O(1/\alpha_{\text{low}}^2)$. By design, one element of A is close to the true α, so the list L contains at least one $\hat{\mu}$ that is $O(\sqrt{\log 1/\alpha})$-close — the error guarantee

Algorithm 5 Outer stage, informal (see Algorithm 10)

Input: X, w_{low}, Δ, and sLD algorithm $\mathcal{A}_{\text{sLD}}$.
Output: Collection of sets $\mathcal{T}$.

$L \leftarrow (\hat{\mu}_1, \dots, \hat{\mu}_M) \coloneqq \mathcal{A}_{\text{sLD}}(X)$ with w_{low};
while $L \neq \emptyset$ **do**
  for $\hat{\mu} \in L$ **do**
    compute for *an appropriate distance function d*

$$S_{\hat{\mu}}^{(1)} = \{x \in X \mid d(x, \hat{\mu}) \leqslant \Delta\}, \quad S_{\hat{\mu}}^{(2)} = \{x \in X \mid d(x, \hat{\mu}) \leqslant 3\Delta\}$$

  if for all $\hat{\mu}$, $|S_{\hat{\mu}}^{(2)}| > 2|S_{\hat{\mu}}^{(1)}|$ **then** add X to $\mathcal{T}$ and update $L \leftarrow \emptyset$
  else
    $\tilde{\mu} \leftarrow \operatorname{argmax}_{|S_{\hat{\mu}}^{(2)}| \leqslant 2|S_{\hat{\mu}}^{(1)}|} |S_{\hat{\mu}}^{(1)}|$
    add $S_{\tilde{\mu}}^{(2)}$ to $\mathcal{T}$
    $X \leftarrow X \setminus S_{\tilde{\mu}}^{(1)}$
return $\mathcal{T}$

that we aim for — and there is indeed at least an α-fraction of samples near $\hat{\mu}$. We call such a hypothesis "nearby".

Finally, we prune this concatenated list by verifying for each $\hat{\mu}$ whether there is indeed an $\hat{\alpha}$-fraction of samples "not too far" from it. This is similar to pruning procedures with known α proposed in prior work (see Proposition B.1 in [31]). Our procedure (i) never discards a "nearby" hypothesis, and outputs a list where (ii) every hypothesis contains a sufficient number of points close to it and (iii) all hypotheses are separated. Property (i) implies that the final error is $O(\sqrt{\log 1/\alpha})$ and properties (ii) and (iii) imply list size bound $1 + O((1-\alpha)/\alpha_{\text{low}})$. Note that when $\alpha < \alpha_{\text{low}}$, the list size can be simply upper bounded by $O(1/\alpha_{\text{low}})$, see Theorem B.2.4.

5.4.2 *Two-stage meta-algorithm*

Note that even though we could run $\mathcal{A}_{\text{aLD}}$ directly on the entire dataset with $\alpha_{\text{low}} = w_{\text{low}}$, we would only achieve an error for the i^{th} inlier cluster mean of $O(\sqrt{\log 1/w_i})$ – which can be much larger than $O(\sqrt{\log 1/\tilde{w}_i})$ – for a list of size $O(1/w_{\text{low}})$. While $\mathcal{A}_{\text{aLD}}$ takes into account the unknown weight of the clusters, it still treats other inlier clusters as outliers. We now

Algorithm 6 InnerStage

Input: Samples $S = \{x_1, \dots, x_n\}$, $\alpha_{\text{low}} \in [w_{\text{low}}, 1]$, $\mathcal{A}_{\text{kLD}}$, and $\mathcal{A}_R$.
Output: List L.

$\alpha_{\text{low}} \leftarrow \min(1/100, \alpha_{\text{low}})$
$M \leftarrow \emptyset$
for $\hat{\alpha} \in \{\alpha_{\text{low}}, 2\alpha_{\text{low}}, \dots, \lfloor 1/(3\alpha_{\text{low}}) \rfloor \alpha_{\text{low}}\}$ **do**
    run $\mathcal{A}_{\text{kLD}}$ on S with fraction of inliers set to $\hat{\alpha}$
    add the pair $(\hat{\mu}, \hat{\alpha})$ to M for each output $\hat{\mu}$
Let L be the output of ListFilter (Algorithm 8) run on S, α_{low}, and M
for $(\hat{\mu}, \hat{\alpha}) \in L$ **do**
    replace $\hat{\mu}$ by the output of ImproveWithRME (Algorithm 9) run on S, $\hat{\mu}$, $\tau = 40\psi_t(\hat{\alpha}) + 4f(\hat{\alpha})$, and $\mathcal{A}_R$
return L

show that if the outer stage Algorithm 5 of our meta-algorithm Algorithm 4 separates the samples into a not-too-large collection $\mathcal{T}$ of sets with certain properties, running $\mathcal{A}_{\text{aLD}}$ separately on each of the sets can lead to the desired guarantees. In particular, let us assume that $\mathcal{T}$ consists of potentially overlapping sets such that:

(1) For each inlier cluster C^*, there exists one set $T \in \mathcal{T}$ such that T contains (almost) all points from C^* and at most $O(\varepsilon n)$ other points,

(2) It holds that $\sum_{T \in \mathcal{T}} |T| \leqslant n + O(\varepsilon n)$.

By (1), for every inlier cluster C^* with a corresponding true weight w^*, there exists a set T such that the points from C^* constitute at least an $\tilde{w}$-fraction of T with $\tilde{w} := \Omega(w^*/(w^* + \varepsilon))$. By Section 5.4.1, applying $\mathcal{A}_{\text{aLD}}$ with $\alpha_{\text{low}} = w_{\text{low}} \cdot n/|T|$ on such a T then yields a list of size $1 + O((1 - \tilde{w})/w_{\text{low}})$ with an estimation error at most $O(\sqrt{\log 1/\tilde{w}})$. If T contains (almost) no inliers, that is, there is no inlier component that should recovered, then $\mathcal{A}_{\text{kLD}}$ returns a list of size $O(|T|/(w_{\text{low}} n))$.

Now, by the two properties, (almost) all inlier points lie in at most k sets of $\mathcal{T}$, and all other sets of $\mathcal{T}$ contain in total at most $O(\varepsilon n)$ points. Hence, concatenating all lists outputted by $\mathcal{A}_{\text{aLD}}$ applied to all $T \in \mathcal{T}$ leads to a final list size bounded by $k + O(\varepsilon/w_{\text{low}})$.

5.4.3 *Outer stage: separating inlier clusters*

We now informally describe the outer stage that produces the collection of sets $\mathcal{T}$ with the desiderata described in Section 5.4.2, leaving the details to Appendix B.5. The main steps are outlined in pseudocode in Algorithm 5.

Given a set X of $N = \mathrm{poly}(d^t, 1/w_{\mathrm{low}})$ i.i.d. input samples from the distribution eq. (5.2.1) with Gaussian inlier components, the first step of the meta-algorithm is to run Algorithm 5 on X and w_{low} with $\Delta = O(\sqrt{\log 1/w_{\mathrm{low}}})$. Algorithm 5 runs an sLD algorithm on the samples and produces a (large) list of estimates L such that, for each mean, at least one estimate is $O(\sqrt{\log 1/w_{\mathrm{low}}})$-close to it. It then add sets to $\mathcal{T}$ that correspond to these estimates via a dynamic "two-scale" process.

Specifically, for each $\hat{\mu} \in L$, we construct *two sets* $S^{(1)}_{\hat{\mu}} \subseteq S^{(2)}_{\hat{\mu}}$ consisting of samples close to $\hat{\mu}$. By construction, we guarantee that if $S^{(1)}_{\hat{\mu}}$ contains a non-negligible fraction of samples from any inlier cluster C^*, then $S^{(2)}_{\hat{\mu}}$ contains (almost) all samples from C^* (see Theorem B.2.7 (ii)).

Now we very briefly illustrate how this process could be helpful in proving properties (1) and (2). Observe that, as long as there exists some $\hat{\mu}$ with $|S^{(2)}_{\hat{\mu}}| \leqslant 2|S^{(1)}_{\hat{\mu}}|$, we add $S^{(2)}_{\hat{\mu}}$ to $\mathcal{T}$ and remove the samples from $S^{(1)}_{\hat{\mu}}$. Consider one such $\hat{\mu}$. For property (1), we merely note that if $S^{(1)}_{\hat{\mu}}$ contains a part of an inlier cluster C^*, then $S^{(2)}_{\hat{\mu}}$ contains (almost) all of C^*, so we add to $\mathcal{T}$ a set that contains (almost) all of C^*; otherwise, when we remove $S^{(1)}_{\hat{\mu}}$ we remove (almost) no points from C^*, so (almost) all the points from C^* remain in play. For property (2), we merely note that whenever we add $S^{(2)}_{\hat{\mu}}$ to $\mathcal{T}$, increasing the number of points in it by $|S^{(2)}_{\hat{\mu}}|$, we also remove the samples from $S^{(1)}_{\hat{\mu}}$, reducing the number of samples by $|S^{(1)}_{\hat{\mu}}| \geqslant |S^{(2)}_{\hat{\mu}}|/2$. The proof of the properties uses some additional arguments of a similar flavor, and we defer it to Appendix B.5.

5.5 RELATED WORK

LIST-DECODABLE MEAN ESTIMATION Inspired by the list-decoding paradigm that was first introduced for error-correcting codes for large error rates [77], list-decodable mean estimation has become a popular approach for robustly learning the mean of a distribution when the majority of the samples are outliers. A long line of work has proposed efficient algorithms

with theoretical guarantees. These algorithms are either based on convex optimization [72, 78], a filtering approach [31, 79], or low-dimensional projections [80]. Near-linear time algorithms were obtained in [81] and [36]. The list-decoding paradigm is not only used for mean estimation but also other statistical inference problems. Examples include sparse mean estimation [82, 83], linear regression [84–86], subspace recovery [87, 88], clustering [89], stochastic block models and crowd sourcing [78, 90].

ROBUST MEAN ESTIMATION AND MIXTURE LEARNING When the outliers constitute a minority, algorithms typically achieve significantly better error guarantees than in the list-decodable setting. Robust mean estimation algorithms output a single vector close to the mean of the inliers. In a variety of corruption models, efficient algorithms are known to achieve (nearly) optimal error under adversarial corruptions [72–74, 78, 91, 92].

Robust mixture learning tackles the model in eq. (5.2.1) with $\varepsilon \ll \min_i w_i$ and aims to output exactly k vectors with an accurate estimate for the population mean of each component [31, 32, 35, 72, 74, 93, 94]. These algorithms do not enjoy error guarantees for clusters with weights $w_i < \varepsilon$. To the best of our knowledge, our algorithm is the first to achieve non-trivial guarantees in this larger noise regime.

ROBUST CLUSTERING Robust clustering [95] also addresses the presence of small fractions of outliers in a similar spirit to robust mixture learning, conceptually implemented in the celebrated DBScan algorithm [96]. Assuming the output list size is large enough to capture possible outlier clusters, these methods may also be used to tackle list-decodable mixture learning - however, they do not come with an inherent procedure to determine the right choice of hyperparameters that ultimately output a list size that adapts to the problem.

5.6 DISCUSSION AND FUTURE WORK

In this work, we prove that even when small groups are outnumbered by adversarial data points, efficient list-decodable algorithms can provide an accurate estimation of all means with minimal list size. The proof for the upper bound is constructive and analyzes a plug-and-play meta-algorithm (cf. Figure 5.1) that inherits guarantees of the black-box cor-kLD algorithm $\mathcal{A}_{\mathrm{kLD}}$ and RME algorithm $\mathcal{A}_R$, which it uses as base learners. Notably, when the inlier mixture is a mixture of Gaussians with identity covariance,

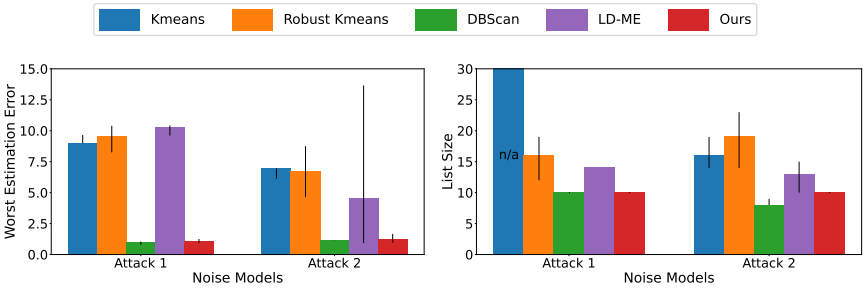

FIGURE 5.2: Comparison of five algorithms with two adversarial noise models. The attack distributions and further experimental details are given in Appendix B.9. On the left we show worst estimation error for constrained list size and on the right the smallest list size for constrained error guarantee. We plot the median of the metrics with the error bars showing 25th and 75th percentile.

we achieve optimality. Furthermore, any new development for the base learners automatically translates to improvements in our bounds.

We would like to end by discussing the possible practical impact of this result. Since an extensive empirical study is out of the scope of this paper, besides the fact that ground-truth means for unsupervised real-world data are hard to come by, we provide preliminary experiments on synthetic data. Specifically, we generate data from a separated $k-$Gaussian mixture with additive contaminations as in eq. (5.2.1) and different types of adversarial distributions (see detailed description in Appendix B.9). We focus on the regime $\varepsilon \sim w_i$ where our algorithms shows the largest theoretical improvements. Precisely, we consider $k = 7$ inlier clusters in $d = 100$ dimensions with cluster weights ranging from 0.3 to 0.02, $w_{\text{low}} = 0.02$, and $\varepsilon = 0.12$. We implement our meta-algorithm with the LD-ME base learner from [36] (omitting the RME base learner improvement for large clusters).

We then compare the output of our algorithm with the vanilla LD-ME algorithm from [36] with $w_{\text{low}} = 0.02$ and (suboptimal) LD-ML guarantees as well as well-known (robust) clustering heuristics without LD-ML guarantees, such as the k-means [97], Robust k-means [98], and DBSCAN [96]. Even though none of these heuristics have LD-ML guarantees, they are commonly used and known to also perform well in practice in noisy settings. The hyper-parameters of the algorithms are tuned beforehand and for the comparative algorithms multiple tuned configurations are selected for

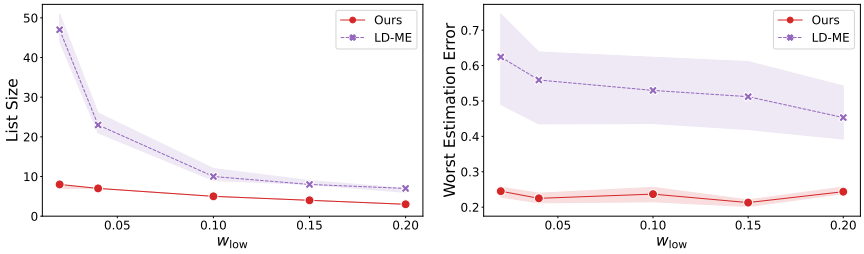

FIGURE 5.3: Comparison of list size and estimation error for large inlier cluster for varying w_{low} inputs. The experimental setup is illustrated in Appendix B.9. We plot the median values with error bars showing 25th and 75th quantiles. As w_{low} decreases, we observe a roughly constant estimation error for our algorithm while the error for LD-ME increases. Further, the decrease in list size is much more severe for LD-ME than for our algorithm.

the experiments. For each algorithm, we run 100 seeds of each parameter setting and record their performance. In Figure 5.2 (left), we fix the list size to 10 and plot the errors for the worst inlier cluster, typically the smallest. We compare the performance of the algorithms by plotting the worst-case estimation errors for a given list size and list sizes that each algorithm requires to achieve a given worst-case estimation error. In Figure 5.2 (right), we fix the error and plot the minimal list size at which competing algorithms reach the same or smaller worst estimation error. Further details on the evaluation metric and parameter tuning are provided in Appendix B.9.

In a different experiment (see Appendix B.9.1 for details), we study the effect of varying w_{low} on the performance of our approach and LD-ME. Figure 5.3 shows the worst estimation error for a cluster with weight of 0.2 that is contaminated by noise. As expected from our theoretical results, we observe that the mean-estimation performance of our algorithm stays roughly constant, regardless of the initial w_{low}. Meanwhile, the estimation error of LD-ME increases as w_{low} decreases further below the true cluster weight. Furthermore, our algorithm consistently outperforms LD-ME in both estimation error and list size, demonstrating significantly lower list size values with only a slight decrease in the same.

Overall, we observe that, in line with our theory, our method significantly outperforms the LD-ME algorithm, and performs better or on par with the heuristic approaches. Additional experimental comparison and implementation details can be found in Appendix B.9. Even though these

experiments do not allow conclusive statements about the improvement of our algorithm for mixture learning for real-world data, they do provide encouraging evidence that effort could be well-spent on follow-up empirical and theoretical work building on our results. For example, it would be interesting to conduct a more extensive empirical study that compares our algorithm against a plethora of robust clustering algorithms. Furthermore, practical data often includes components with different covariances and scalings. One could hence explore an extension of our algorithm that can incorporate different covariances at the same time being agnostic to it.

A

APPENDIX OF GREEDY

A.1 AUXILIARY LEMMAS

Lemma A.1.1 (Lower Bound in Lemma 2.4.1)**.** *We have that*

$$val_{LP} \geqslant \frac{m}{d_{\max}}.$$

Proof. Let $x^*_{\mathsf{LP}} = (x^*_1, x^*_2, \ldots, x^*_m)$ be an optimal solution for (1.1.2). Since $Ax^*_{\mathsf{LP}} \geqslant \mathbf{1}$ entrywise, by summing all entries we obtain that

$$m \leqslant \sum_i x^*_i X_i \leqslant d_{\max} \sum_i x^*_i = d_{\max} \mathrm{val}_{\mathsf{LP}}.$$

which upon rearranging yields the desired result. □

In addition to the above, we have the following elementary upper bound on $\mathrm{val}_{\mathsf{LP}}$, which holds both in the sparse and dense regime.

Lemma A.1.2 (Upper Bound in Lemma 2.4.1)**.** *There exists $c > 0$, independent of n, such that*

$$\Pr\left(val_{LP} \lesssim \frac{1}{p}\right) \geqslant 1 - \exp\left(-cn^{1-\delta}\right).$$

This also implies that $\Pr(\mathtt{IP}$ *is feasible*$) \geqslant 1 - \exp\left(-cn^{1-\delta}\right)$.

Proof. Consider the candidate feasible solution $\hat{x} := \frac{1}{\tilde{C}np}\mathbf{1}$, for some constant $0 < \tilde{C} < 1$. The following results from applying a union bound over constraints and the standard Chernoff bound.

$$\begin{aligned}
\Pr\left(\hat{x} \text{ not feasible}\right) &= \Pr\left(\exists i \in [m] : (A\hat{x})_i < 1\right) \\
&\leqslant m \Pr\left(\mathrm{Bin}(n,p) < \tilde{C}np\right) \\
&\leqslant n^C \exp\left(-\frac{(1-\tilde{C})^2 np}{2}\right) \\
&\leqslant \exp\left(-cn^{1-\delta}\right).
\end{aligned}$$

The desired conclusion follows by considering the complementary event to the one above and noting that $\|\hat{x}\|_1 \sim 1/p$. Note that the event $\{\hat{x}$ is feasible for $\mathtt{LP}\}$ implies the event $\{\mathtt{IP}$ is feasible$\}$. □

Lemma A.1.3 (Lambert W function, [99]). *For any $x \geqslant e$, there holds that*

$$\log x - \log\log x + \frac{\log\log x}{2\log x} \leqslant W_0(x) \leqslant \log x - \log\log x + \frac{e}{e-1}\frac{\log\log x}{\log x}. \tag{A.1.1}$$

In particular,

$$W_0(x) = \log x - \log\log x + o(1), \quad \text{as} \quad x \to \infty. \tag{A.1.2}$$

In addition, for any $x \geqslant 1/e$, the following identity is satisfied

$$W_0(x) = \log\frac{x}{W_0(x)}. \tag{A.1.3}$$

Proof of Lemma 2.4.2. Fix $D \geqslant 1$. Let

$$Z_k := \left|\left\{x \in \{0,1\}^m : Ax \geqslant \mathbf{1}, \|x\|_1 = k\right\}\right|$$

be the number of feasible solutions of norm exactly k. Clearly, $Z_k \leqslant Z_{k+1}$ for any $k \geqslant 0$. We also have that

$$\mathbb{E}Z_k = \sum_{\|x\|=k} \Pr\left((Ax)_i \geqslant 1, \forall i \in [m]\right) = \binom{n}{k}\left(1-(1-p)^k\right)^m.$$

We will now show that for $k \ll \frac{1}{p}\log\left(\frac{mp}{\log n}\right)$, we have $\mathbb{E}Z_k \leqslant n^{-D}$. Using that $p \leqslant 1/2$ from Assumption 4 and that for $x \in (0, \frac{1}{2})$, we have $(1-x)^y \geqslant e^{-2xy}$, we can bound

$$\begin{aligned}\mathbb{E}Z_k = \binom{n}{k}(1-(1-p)^k)^m &\leqslant n^k\left(1-e^{-2pk}\right)^m \\ &\leqslant n^k e^{-me^{-2pk}} = \exp\left\{k\log n - me^{-2pk}\right\}.\end{aligned}$$

Therefore, $\mathbb{E}Z_k \leqslant n^{-D}$ will follow from

$$2pke^{2pk} \leqslant -2Dpe^{2pk} + \frac{2mp}{\log n}. \tag{A.1.4}$$

upon multiplying both sides of the latter inequality by $\frac{\log n}{2pe^{2pk}}$ and exponentiating. Since $k \ll \frac{1}{p}\log\left(\frac{mp}{\log n}\right)$, we also have that $k \leqslant k_* := \frac{1}{2p}W_0\left(\frac{mp}{D\log n}\right)$ for n large enough. For $k = k_*$, the left hand side of (A.1.4) is equal to

$\frac{mp}{D\log n}$, while the right hand side is lower bounded by $\frac{mp}{\log n}$. Since $D \geqslant 1$, we recover that $\mathbb{E}Z_k \leqslant n^{-D}$. Note that for n large enough, $\mathrm{val}_{\mathtt{IP}} \ll \frac{1}{p}\log\frac{mp}{\log n}$ implies that $Z_{k_*} > 0$. Therefore, applying Markov's inequality, we get that

$$\Pr\left(\mathrm{val}_{\mathtt{IP}} \ll \frac{1}{p}\log\left(\frac{mp}{\log n}\right)\right) \leqslant \Pr\left(Z_{k_*} > 0\right) \leqslant \mathbb{E}Z_{k_*} \leqslant n^{-D}, \qquad \text{(A.1.5)}$$

and the proof follows by considering the complementary events. Note that using similar derivations, one can also show that for $k^* := \frac{1}{p}\log\left(\frac{1}{\delta}\frac{mp}{\log n}\right)$, where δ is defined in Assumption 4, we have $\mathbb{E}Z_{k^*} \geqslant 1$. □

Proof of Lemma 2.4.3. For ease of notation, let us define $b_n := \frac{\log n}{mp}$, $b_n^* := \frac{1}{e}(b_n - 1)$, $g_n := \frac{\log n}{\log(\log n/mp)}$. We begin by proving the desired upper bound. By Jensen's inequality and bounding the maximum of positive values by their sum, for any $\lambda > 0$, we obtain

$$\begin{aligned}\mathbb{E}\max_{i\in[n]} X_i &\leqslant \frac{1}{\lambda}\log\mathbb{E}\exp\left(\lambda\max_{i\in[n]} X_i\right)\\ &= \frac{1}{\lambda}\log\mathbb{E}\left(\max_{i\in[n]}\exp\left(\lambda X_i\right)\right)\\ &\leqslant \frac{1}{\lambda}\log\sum_{i\in[n]}\mathbb{E}\exp(\lambda X_i).\end{aligned}$$

Finally, computing the moment generating function of binomial random variables, together with the inequality $1 - x \leqslant e^{-x}$ yields

$$\mathbb{E}\max_{i\in[n]} X_i = \frac{\log n + m\log\left(1 - p(1-e^{\lambda})\right)}{\lambda} \leqslant \frac{\log n - mp(1-e^{\lambda})}{\lambda}.$$

In the regime where $mp \gtrsim \log n$, we may choose $\lambda > 0$ arbitrary, independent of n, from which it immediately follows that $\mathbb{E}\max_{i\in[n]} X_i \lesssim mp$. For $mp \ll \log n$, we proceed by differentiating the last line in the above display and setting the resulting expression to zero. From this, we may choose λ as the solution of the following.

$$e^{\lambda-1}(\lambda - 1) = b_n^*$$

Under the present assumptions, this is expressed in terms of the Lambert W function as $\lambda = 1 + W_0(b_n^*)$, so that by (A.1.3), we obtain

$$\mathbb{E}\max_{i\in[n]} X_i \leqslant \frac{\log n\left(1 - \frac{1}{b_n} + \frac{b_n^*}{b_n}\frac{e}{W_0(b_n^*)}\right)}{1 + W_0(b_n^*)} \sim g_n.$$

In the dense $mp \gtrsim \log n$ regime, a matching lower bound is easily obtained by noting that $\mathbb{E} \max_{i\in[n]} X_i \geqslant \mathbb{E}\, X_1 = mp$.
To deal with the sparse regime, let $\tau = 1/16$. From Markov's inequality,

$$\mathbb{E} \max_{i\in[n]} X_i \geqslant \tau g_n \Pr\left(\max_{i\in[n]} X_i = \lceil \tau g_n \rceil\right) = \tau g_n \left(1 - (1 - \Pr\left(X_1 = \lceil \tau g_n \rceil\right))^n\right).$$

Hence, applying Lemma A.2.2, for n large enough,

$$\mathbb{E} \max_{i\in[n]} X_i \geqslant \tau g_n \left(1 - \left(1 - n^{-1/2}\right)^n\right) \geqslant (\tau/2) g_n,$$

thus providing a matching lower bound for the sparse regime.

In the intermediate threshold regime $mp \sim \log n$, the average and maximum of X_i's become of the same order, that is $mp \sim \mathbb{E} d_{\mathtt{max}} \sim \log n$. The smooth transition follows by noting that in this regime, $b_n, b_n^*, W_0(b_n^*) \sim 1$. □

Lemma A.1.4. *(Chernoff Bound - upper tail) Let $X_1, ..., X_n$ be independent random variables taking values in $\{0, 1\}$, X denote their sum and $\mu = \mathbb{E}\, X$. Then for any $\delta > 0$,*

$$\Pr\left(X \geqslant (1+\delta)\mu\right) \leqslant e^{-\delta^2 \mu/(2+\delta)}.$$

In order to deal with concentration of $d_{\mathtt{max}}$ around its expectation, we state the following useful result on tensorization of variance. We introduce notation Var_i and $\mathbb{E}_i$, where subscript i indicates conditioning on each component of an underlying random vector, except for the i-th one.

Lemma A.1.5 (Theorem 2.3, [100])**.** *Let $X_1, ..., X_n$ be independent random variables and for each function $f : \mathbb{R}^n \to \mathbb{R}$, define*

$$Var_i f(x_1, ..., x_n) := Var\left(x_1, ..., x_{i-1}, X_i, x_{i+1}, ..., x_n\right).$$

Then, there holds that

$$Var\left(f\left(X_1, ..., X_n\right)\right) \leqslant \mathbb{E} \sum_{i=1}^{n} Var_i f\left(X_1, ..., X_n\right)$$

Lemma A.1.6 (Concentration for $d_{\mathtt{max}}$)**.** *Let $X_1, \ldots, X_n \overset{\text{iid}}{\sim} Bin(m, p)$. Then, for any $t > 0$,*

$$\Pr\left(|d_{\mathtt{max}} - \mathbb{E}\, d_{\mathtt{max}}| > t\right) \leqslant \frac{mp}{t^2}.$$

Remark A.1.7. *Note that in all regimes of m, p satisfying Assumption 2.0.1, choosing $t \sim \mathbb{E}\, d_{\max}$ is sufficient to deduce from the previous lemma that $d_{\max} \sim \mathbb{E} d_{\max}$ w.h.p..*

Proof. Proceeding by Chebyschev's inequality, it suffices to show that $\mathrm{Var}(d_{\max} \leqslant mp$. By Lemma A.1.5, we have that

$$\begin{aligned}
\mathrm{Var}(d_{\max}) &\leqslant \mathbb{E} \sum_{i=1}^{n} \mathop{\mathbb{E}}_{i} \left(d_{\max} - \mathop{\mathbb{E}}_{i} d_{\max} \right)^2 \\
&= \mathbb{E} \sum_{i=1}^{n} \mathop{\mathbb{E}}_{i} \left[\left(d_{\max} - \mathop{\mathbb{E}}_{i} d_{\max} \right)^2 \mid d_{\max} = X_i \right] \Pr\left(d_{\max} = X_i \right) \\
&+ \mathbb{E} \sum_{i=1}^{n} \mathop{\mathbb{E}}_{i} \left[\left(d_{\max} - \mathop{\mathbb{E}}_{i} d_{\max} \right)^2 \mid d_{\max} \neq X_i \right] \Pr\left(d_{\max} \neq X_i \right) \\
&= \frac{1}{n} \mathbb{E} \sum_{i=1}^{n} \mathrm{Var} X_i \\
&\leqslant mp,
\end{aligned}$$

which is as required. □

Proof of Lemma 2.4.4. Let us consider the sparse and dense regimes separately.
In the dense regime for $mp \gtrsim \log n$, there exist constants $c_1, c_2, c_3 > 0$ such that $c_1 mp \leqslant \mathbb{E} \max_{i \in [n]} X_i \leqslant c_2 mp$, as argued in Lemma 2.4.3, and $mp \geqslant c_3 \log n$. We apply the union and Chernoff bounds as in Lemma A.1.4 to obtain, for any $t \geqslant 1/c_1$,

$$\begin{aligned}
\Pr\left(\max_{i \in [n]} X_i \geqslant t \cdot \mathbb{E} \max_{i \in [n]} X_i \right) &\leqslant n \Pr\left(X_1 \geqslant t c_1 mp \right) \\
&\leqslant n \exp\left(- \frac{(tc_1 - 1)^2 mp}{1 + tc_1} \right) \\
&\leqslant n \exp\left(- \frac{c_3 (tc_1 - 1)^2 \log n}{1 + tc_1} \right).
\end{aligned}$$

It now suffices to choose t as a function of c_1, c_3 such that $\frac{c_3 (tc_1 - 1)^2}{1 + tc_1} > 1$. By rearranging and solving the resulting quadratic equation, it follows immediately that any $t > \frac{1}{c_1} + \frac{1 + \sqrt{1 + 8c_3}}{2c_3 c_1} > \frac{1}{c_1}$ suffices. Hence, there exist universal constants $c, \tilde{c}$, such that the desired conclusion holds.
We now consider the sparse regime $mp \ll \log n$, where by Lemma 2.4.3

there exists $c_4 > 0$ such that $mp \leqslant c_4 \log n / \log\left(\frac{\log n}{\log mp}\right)$. Notice that for any $\lambda > 0$, $\max_{i\in[n]} X_i \leqslant \frac{1}{\lambda} \log \sum_{i=1}^n e^{\lambda X_i}$. We apply Markov's inequality to obtain, for any $t > 0$,

$$\begin{aligned}
\Pr\left(\max_{i\in[n]} X_i \geqslant t \cdot \mathbb{E}\max_{i\in[n]} X_i\right) &\leqslant \Pr\left(\sum_{i=i}^n e^{\lambda X_i} \geqslant e^{\lambda t\, \mathbb{E}\max_{i\in[n]} X_i}\right) \\
&\leqslant \frac{n\,\mathbb{E}\, e^{\lambda X_1}}{\exp\left(\lambda t\ \mathbb{E}\max_{i\in[n]} X_i\right)} \\
&= \frac{n\left(1-p+pe^{\lambda}\right)^m}{\exp\left(\lambda t\ \mathbb{E}\max_{i\in[n]} X_i\right)} \\
&\leqslant \exp\left(\log n + mp\left(e^{\lambda}-1\right) - \frac{\lambda t\, c_4 \log n}{\log\left(\frac{\log n}{mp}\right)}\right),
\end{aligned}$$

where we used that $1 + x < e^x$ to obtain the last inequality. Finally, by choosing $t = 3/c_4$ and $\lambda = \log(\log n / mp)$, we obtain

$$\Pr\left(\max_{i\in[n]} X_i \geqslant \frac{3}{c_4} \cdot \mathbb{E}\max_{i\in[n]} X_i\right) \leqslant \frac{1}{n}.$$

□

Lemma A.1.8 (Asymptotic expression for binomial probability mass function)**.**
Let $a \equiv a(n)$ and $b \equiv b(n)$ be such that

1. $1 \ll b \ll \sqrt{a}$,
2. $p \ll 1$.

If $b \geqslant Cap$ for $C > 1$, then

$$\log \Pr(\mathrm{Bin}(\lceil a\rceil, p) = \lceil b\rceil) \geqslant -\left(b\log\frac{b}{ap} - b + ap\right)(1+o(1)), \quad \text{(A.1.6)}$$

If also $b \gg ap$, we have that

$$\log \Pr(\mathrm{Bin}(\lceil a\rceil, p) = \lceil b\rceil) \geqslant -\left(b\log\frac{b}{ap}\right)(1+o(1)), \quad \text{(A.1.7)}$$

Furthermore, all bounds remain valid upon replacing $\lceil a\rceil$ to $\lfloor a\rfloor$.

Proof.

$$\begin{aligned}\Pr(\mathrm{Bin}(\lceil a\rceil, p) = \lceil b\rceil) &= \binom{\lceil a\rceil}{\lceil b\rceil} p^{\lceil b\rceil}(1-p)^{\lceil a\rceil-\lceil b\rceil} \\ &\overset{(i)}{\geqslant} \frac{(\lceil a\rceil p)^{\lceil b\rceil}}{4(\lceil b\rceil)!}(1-p)^{\lceil a\rceil-\lceil b\rceil} \\ &\overset{(ii)}{\geqslant} \frac{1}{\lceil b\rceil}\left(\frac{\lceil a\rceil ep}{\lceil b\rceil}\right)^{\lceil b\rceil}(1-p)^{\lceil a\rceil-\lceil b\rceil} \\ &\overset{(iii)}{\geqslant} \frac{1}{\lceil b\rceil}\left(\frac{\lceil a\rceil ep}{\lceil b\rceil}\right)^{\lceil b\rceil} e^{-\lceil a\rceil p}(1-p)^{\lceil a\rceil p-\lceil b\rceil},\end{aligned}$$

where (i) is due to $\binom{n}{k} \geqslant \frac{n^k}{4k!}$ for $0 \leqslant k \leqslant \sqrt{n}$, (ii) is due to $n! \leqslant \frac{n}{4}(n/e)^n$ for n large enough, and (iii) is due to $(1+x/n)^n \geqslant e^x(1-x^2/n)$ for $|x| \leqslant n$. After taking the logarithm, we get

$$\begin{aligned}&\log\Pr(\mathrm{Bin}(\lceil a\rceil, p) = \lceil b\rceil) \\ &\quad\geqslant -\left(\lceil b\rceil \log\frac{\lceil b\rceil}{\lceil a\rceil p} - \lceil b\rceil + \lceil a\rceil p + \log\lceil b\rceil - (\lceil a\rceil p - \lceil b\rceil)\log(1-p)\right).\end{aligned}$$

If $b \geqslant C\lceil a\rceil p$ for $C > 1$, we have that

$$b\log\frac{b}{ap} - b + ap \geqslant \gamma b \gg 1, \qquad \text{for } \gamma := \frac{1}{C} + \log C - 1 > 0.$$

Since $b \gg 1$, we have

$$\frac{\lceil b\rceil \log\frac{\lceil b\rceil}{\lceil a\rceil p} - \lceil b\rceil + \lceil a\rceil p}{b\log\frac{b}{ap} - b + ap} = 1 + o(1).$$

Now, since also $\log\lceil b\rceil \ll b$ and $(\lceil a\rceil p - \lceil b\rceil)\log(1-p) \ll b$ for $p \ll 1$, we have that

$$\log\Pr(\mathrm{Bin}(\lceil a\rceil, p) = \lceil b\rceil) \geqslant -\left(b\log\frac{b}{ap} - b + ap\right)(1+o(1)).$$

If additionally $b \gg ap$, then

$$\frac{b\log\frac{b}{ap} - b + ap}{b\log\frac{b}{ap}} = 1 + o(1),$$

and, finally,

$$\log \Pr(\mathrm{Bin}(\lceil a \rceil, p) = \lceil b \rceil) \geqslant -\left(b \log \frac{b}{ap}\right)(1 + o(1)).$$

Under our assumptions, $1 \ll b \ll \sqrt{a}$, the same bounds hold for $\log \Pr(\mathrm{Bin}(\lfloor a \rfloor, p) = \lfloor b \rfloor)$. □

Lemma A.1.9 (Binomial Monotonicity)**.** *Let $S_m \sim \mathrm{Bin}(m, p)$. Then for $r \geqslant mp$, we have that $\Pr(S_m = r + 1) \leqslant \Pr(S_m = r)$ and $\Pr(S_{m-1} = r) \leqslant \Pr(S_m = r)$.*

Proof. The proof follows a similar argument as that presented in [101].

$$\begin{aligned}
\frac{\Pr(S_m = r+1)}{\Pr(S_m = r)} &= \frac{\binom{m}{r+1} p^{r+1}(1-p)^{m-r-1}}{\binom{m}{r} p^r (1-p)^{m-r}} \\
&= \frac{\frac{m!}{(r+1)!(m-r-1)!} p^{r+1}(1-p)^{m-r-1}}{\frac{m!}{r!(m-r)!} p^r (1-p)^{m-r}} \\
&= \frac{(m-r)p}{(r+1)(1-p)} \leqslant 1.
\end{aligned}$$

Similar arguments show that $\Pr(S_{m-1} = r) \leqslant \Pr(S_m = r)$. □

A.2 MAIN TOOL FOR THE CASE $mp \lesssim \log n$ AND PROOF OF LEMMA A.2.2

Lemma A.2.1. *If $mp \lesssim \log n$, then, for any $\varepsilon > 0$, there exist constants $\tau > 0$ and $1 < \alpha < \beta$, such that, for $k \lesssim \log n$ and for any $\tilde{m}$, satisfying $\beta^{-k-1} m \leqslant \tilde{m} \leqslant \beta^{-k} m$, for all n large enough,*

$$\Pr\left(\mathrm{Bin}(\tilde{m}, p) = \left\lceil (\alpha/\beta)^k \tau \mathbb{E} d_{\mathtt{max}} \right\rceil\right) \geqslant n^{-\varepsilon}.$$

Proof. The proof is essentially a careful application of Lemma A.1.8. Let τ, α, β be constants to be fixed later and $\tilde{m} = \lfloor \beta^{-k-1} m \rfloor$. Depending on whether we have $mp \ll \log n$ or $mp \sim \log n$, different terms will dominate the asymptotic expression from Lemma A.1.8.

We start with the case $mp \ll \log n$. From Lemma 2.4.3, this implies that $mp \ll \mathbb{E}d_{\mathtt{max}} \ll \log n$. Here we can fix $\alpha \equiv 2$ and $\beta \equiv 3$. Applying (A.1.7) for $a = 3^{-k-1}m$ and $b = (2/3)^k \tau \mathbb{E}d_{\mathtt{max}}$, we have:

$$\begin{aligned}
&\log \Pr\left(\mathrm{Bin}(\tilde{m}, p) = \left\lceil (2/3)^k \tau \mathbb{E}d_{\mathtt{max}} \right\rceil\right) \\
&\quad \geqslant -(2/3)^k \tau \mathbb{E}d_{\mathtt{max}} \log\left(\frac{2^k 3 \tau \mathbb{E}d_{\mathtt{max}}}{mp}\right)(1 + o(1))
\end{aligned}$$

Recall that our goal is to show $\log \Pr\left(\mathrm{Bin}(\tilde{m}, p) = \left\lceil (2/3)^k \tau \mathbb{E}d_{\mathtt{max}} \right\rceil\right) \geqslant -\varepsilon \log n$. We first show that there exists $\tau > 0$ satisfying the following two inequalities:

$$\begin{aligned}
&(i) \qquad (2/3)^k \tau (\log 3 + k \log 2) \frac{\mathbb{E}d_{\mathtt{max}}}{\log n} \;\leqslant\; \frac{\varepsilon}{4}, \\
&(ii) \qquad (2/3)^k \tau \frac{\mathbb{E}d_{\mathtt{max}}}{\log n} \log\left(\frac{\mathbb{E}d_{\mathtt{max}}}{mp}\right) \;\leqslant\; \frac{\varepsilon}{4}.
\end{aligned} \tag{A.2.1}$$

Indeed, since $\mathbb{E}d_{\mathtt{max}} \ll \log n$ and $k \ll (3/2)^k$, inequality (i) will be satisfied for any $\tau > 0$ for n large enough. For (ii) we need to use explicit bound for $\mathbb{E}d_{\mathtt{max}}$, in particular from Lemma 2.4.3 we know that there exists $C > 0$, such that $\mathbb{E}d_{\mathtt{max}} \leqslant C \log n / (\log\log n - \log mp)$ for n large enough. Plugging this into (ii), we get for $k = 0$,

$$\begin{aligned}
&\tau \frac{\mathbb{E}d_{\mathtt{max}}}{\log n} \log\left(\frac{\mathbb{E}d_{\mathtt{max}}}{mp}\right) \\
&\quad \leqslant \frac{\tau C (\log C + \log\log n - \log(\log\log n - \log mp) - \log mp)}{\log\log n - \log mp} \\
&\quad = \tau C + o(1).
\end{aligned}$$

For $\tau = \varepsilon/(8C)$, (ii) holds for $k = 0$ for n large enough. By increasing k we only decrease left hand side of (ii), therefore, the same value of τ works for any $k \geqslant 0$.
Finally, by adding (i) and (ii) we showed that, for n large enough,

$$\log \Pr\left(\mathrm{Bin}(\tilde{m}, p) = \left\lceil (\alpha/2)^k \tau \mathbb{E}d_{\mathtt{max}} \right\rceil\right) \geqslant -\frac{\varepsilon}{2} \log n (1 + o(1)) > -\varepsilon \log n,$$

which finishes the proof for the case $mp \ll \log n$.

Now we focus on the case $mp \sim \log n$. Here we apply (A.1.6) for the values $a = \beta^{-k-1}m$ and $b = (\alpha/\beta)^k \tau \mathbb{E} d_{\mathtt{max}}$ keeping in mind the condition $b \geqslant Cap$ with $C > 1$. We have

$$\begin{aligned}
&\log \Pr\Big(\mathrm{Bin}(\tilde{m}, p) = \lceil (\alpha/\beta)^k \tau \mathbb{E} d_{\mathtt{max}} \rceil \Big) \\
&\geqslant -\Big((\alpha/\beta)^k \tau \mathbb{E} d_{\mathtt{max}} \log \big(\frac{\beta \alpha^k \tau \mathbb{E} d_{\mathtt{max}}}{mp} \big) \\
&\qquad - (\alpha/\beta)^k \tau \mathbb{E} d_{\mathtt{max}} + \beta^{-k-1} mp \Big) (1 + o(1))
\end{aligned}$$

We pick $\tau = \gamma mp / \mathbb{E} d_{\mathtt{max}}$, for some constant $\gamma > 1$ to be specified later. Note that this way condition for applying (A.1.6), $\frac{b}{ap} \geqslant C > 1$, is satisfied since $\frac{b}{ap} \geqslant \frac{\tau \mathbb{E} d_{\mathtt{max}}}{mp} = \gamma > 1$. This simplifies the latter expression to the following:

$$\begin{aligned}
&\log \Pr\Big(\mathrm{Bin}(\tilde{m}, p) = \lceil (\alpha/\beta)^k \gamma mp \rceil \Big) \\
&\geqslant -mp \left((\alpha/\beta)^k \gamma \log\big(\beta \gamma \alpha^k\big) - (\alpha/\beta)^k \gamma + \beta^{-k-1} \right) (1 + o(1))
\end{aligned}$$

Since in this regime we have $mp \leqslant D \log n$ for some $D > 0$, for n large enough, it is enough to show

$$(\alpha/\beta)^k \gamma \log\big(\beta \gamma \alpha^k\big) - (\alpha/\beta)^k \gamma + \beta^{-k-1} \leqslant \varepsilon/(2D).$$

We first show that there exist constants $1 < \alpha < \beta$ and $\gamma > 1$, depending on ε and D, satisfying the following two inequalities for any $k \geqslant 0$:

$$\begin{aligned}
&(i) \qquad (\alpha/\beta)^k \left(\gamma \log \beta\gamma - \gamma + \frac{1}{\alpha^k \beta} \right) \leqslant \frac{\varepsilon}{4D}, \\
&(ii) \qquad (\alpha/\beta)^k k \log \alpha \leqslant \frac{\varepsilon}{4D}.
\end{aligned}$$

Note that left hand side of (i) decreases as k increases, therefore, it is enough to look at $k = 0$. We need to show that there exist $\beta, \gamma > 1$, depending on ε, D such that

$$f(\beta, \gamma) := \gamma \log \beta\gamma - \gamma + \frac{1}{\beta} \leqslant \frac{\varepsilon}{4D}.$$

Note that $\frac{\partial f}{\partial \beta} = \gamma/\beta - 1/\beta^2 > 0$ and $\frac{\partial f}{\partial \gamma} = \log \beta\gamma > 0$ as long as $\beta\gamma > 1$. Since $f(1,1) = 0$, we can find $\beta, \gamma > 1$, close enough to 1, such that $f(\beta, \gamma) \leqslant \varepsilon/(4D)$. We use these values of β and γ (or, equivalently, τ). Since

$k \ll (\beta/\alpha)^k$, there exists $\alpha \in (1, \beta)$, such that (ii) holds. Summing (i) and (ii) shows that, for n large enough,

$$\begin{aligned}\log \Pr\Big(\mathrm{Bin}(\tilde{m}, p) = \lceil (\alpha/\beta)^k \gamma mp \rceil\Big) &\geqslant -\frac{\varepsilon mp}{2D}(1 + o(1)) \\ &\geqslant -\frac{\varepsilon \log n}{2}(1 + o(1)) \\ &\geqslant -\varepsilon \log n.\end{aligned}$$

We proved that for $mp \lesssim \log n$, for any $\varepsilon > 0$, for n large enough, there exists τ, α, β, such that

$$\Pr\Big(\mathrm{Bin}(\lfloor \beta^{-k-1} m \rfloor, p) = \lceil (\alpha/\beta)^k \tau \mathbb{E} d_{\mathtt{max}} \rceil\Big) \geqslant n^{-\varepsilon}.$$

Since $\beta^{-k-1} mp < \beta^{-k} mp < \lceil (\alpha/\beta)^k \tau \mathbb{E} d_{\mathtt{max}} \rceil$, from binomial monotonicity, Lemma A.1.9, we have that for any $\tilde{m}$ such that $\beta^{-k-1} m \leqslant \tilde{m} \leqslant \beta^{-k} m$,

$$\Pr\Big(\mathrm{Bin}(\tilde{m}, p) = \lceil (\alpha/\beta)^k \tau \mathbb{E} d_{\mathtt{max}} \rceil\Big) \geqslant n^{-\varepsilon}.$$

In order to deal with the more delicate sparse regime throughout the paper where $mp \ll \log n$, we apply the following technical lemma.

Lemma A.2.2. *For $mp \ll \log n$, $\varepsilon > 0$, and n large enough, we have*

$$\Pr\left(\mathrm{Bin}(m, p) = \left\lceil \frac{\varepsilon}{8} \frac{\log n}{\log(\log n / mp)} \right\rceil\right) \geqslant n^{-\varepsilon}.$$

□

Proof of Lemma A.2.2. We follow the argument in Lemma A.2.1 with $k = 0$ and $\mathbb{E} d_{\mathtt{max}}$ replaced by $\log n / (\log \log n - \log mp)$. Note that in the proof of Lemma A.2.1, in the case $mp \ll \log n$, we only used that $mp \ll \mathbb{E} d_{\mathtt{max}} \ll \log n$ and $\mathbb{E} d_{\mathtt{max}} \leqslant C \log n / (\log \log n - \log mp)$ for some $C > 0$. Since both these properties remain true upon replacing $\mathbb{E} d_{\mathtt{max}}$ with $\log n / (\log \log n - \log mp)$, the proof follows. Since $\tau = \varepsilon/(8C)$, in the setting of Lemma A.2.2, and $C = 1$ in this argument, we pick $\tau = \varepsilon/8$. □

Lemma A.2.3. *Let $\varepsilon > 0$. Consider the following choices of $f_1, f_2, \ldots$:*

(i) *if $mp \lesssim \log n$, for some constants $\tau > 0$ and $1 < \alpha < \beta$,*

$$f_t = \left\lceil (\alpha/\beta)^k \tau \mathbb{E} d_{\mathtt{max}} \right\rceil$$

where k is such that $\beta^{-k-1} m < m - F_{t-1} \leqslant \beta^{-k} m$;

(ii) *if $mp \gg \log n$, and $\log mp \ll \log n$,*

$$f_t = \lceil mp(1-p)^{t-1} \rceil \quad \text{if} \quad t \leqslant t^* := \left\lceil \frac{1}{p} \log \left(\frac{mp}{\log n} \right) \right\rceil,$$

$$f_t = \tilde{f}_{t-t^*}, \quad \text{otherwise,}$$

where $\tilde{f}_t$ is the sequence from the case $mp \lesssim \log n$;

(iii) *otherwise, i.e., when $\log mp \gtrsim \log n$,*

$$f_t = \lceil mp(1-p)^{t-1} \rceil.$$

Then, there exists K, such that

$$\begin{aligned} &(i) && F_K \geqslant m - K; \\ &(ii) && \text{if } mp \lesssim \log n, \text{ then } K \sim val_{\mathtt{LP}}; \\ & && \text{if } mp \gg \log n, \text{ then } K \sim val_{\mathtt{IP}}. \end{aligned} \tag{A.2.2}$$

Furthermore, for this sequence f_t (which depends on ε), for any $t \leqslant K$,

$$\Pr(\mathrm{Bin}(m - F_{t-1}, p) \geqslant f_t) \geqslant n^{-\varepsilon}. \tag{A.2.3}$$

Note that the implicit constants in the statements $K \sim val_{\mathtt{LP}}$ or $K \sim val_{\mathtt{IP}}$ depend on ε.

Proof. We proceed in the proof by first showing that there exists $\tilde{K}$, such that $m - F_{\tilde{K}} \lesssim \tilde{K}$, and then, by increasing $\tilde{K}$ by a multiplicative factor, we find K such that $m - F_K \leqslant K$.

Case $mp \lesssim \log n$

From Lemma A.2.1, there exist constants $\tau > 0$, α, β with $1 < \alpha < \beta$, such that, for any $\tilde{m}$, satisfying $\beta^{-k-1} m \leqslant \tilde{m} \leqslant \beta^{-k} m$, for all n large enough,

$$\Pr\left(\mathrm{Bin}(\tilde{m}, p) = \left\lceil (\alpha/\beta)^k \tau \mathbb{E} d_{\mathtt{max}} \right\rceil \right) \geqslant n^{-\varepsilon}.$$

Recall that in this case $f_t = \left\lceil (\alpha/\beta)^k \tau \mathbb{E} d_{\mathtt{max}} \right\rceil$, where k is such that $\beta^{-k-1} m \leqslant m - F_{t-1} \leqslant \beta^{-k} m$ and $F_t = \sum_{s=1}^{t} f_s$. From Lemma A.2.1 we have that

$\Pr(\mathrm{Bin}(m - F_{t-1}, p) = f_t) \geqslant n^{-\varepsilon}$. Our goal is to prove that there exists $s \lesssim \mathrm{val}_{\mathtt{LP}} \sim m/\mathbb{E}d_{\mathtt{max}}$, such that $m - F_s \lesssim s$.

Lemma A.2.4. *Let* $t^{(k)} := \frac{\beta-1}{\beta\tau}\frac{m}{\mathbb{E}d_{\mathtt{max}}}\alpha^{-k}$.

$$\begin{aligned} &\textit{If} && m - F_{t-1} \leqslant \beta^{-k} m \\ &\textit{then} && m - F_{t+t^{(k)}-1} \leqslant \beta^{-k-1} m. \end{aligned}$$

Informally, if after $t-1$ steps of `BlockGreedy`*, at most $\beta^{-k}m$ subsets are uncovered, then after $t + t^{(k)} - 1$ steps, at most $\beta^{-k-1}m$ subsets remain uncovered.*

Proof. Let $s \geqslant t$. As long as $m - F_{s-1} > \beta^{-k-1}m$, we will always have $f_s = \lceil (\alpha/\beta)^k \tau \mathbb{E}d_{\mathtt{max}} \rceil$. We proceed by contradiction. Assume that $m - F_{t+t^{(k)}-1} > \beta^{-k-1}m$. This implies that for all $s \in [t-1, t+t^{(k)}-1]$, we have $f_s = f := \lceil (\alpha/\beta)^k \tau \mathbb{E}d_{\mathtt{max}} \rceil$. Therefore,

$$F_{t+t^{(k)}-1} - F_{t-1} = t^{(k)} f \geqslant \frac{m(\beta-1)}{\beta^{k+1}} = \beta^{-k}m - \beta^{-k-1}m,$$

and

$$\begin{aligned} m - F_{t+t^{(k)}-1} &= m - F_{t-1} - \left(F_{t+t^{(k)}-1} - F_{t-1}\right) \\ &\leqslant \beta^{-k}m - (\beta^{-k}m - \beta^{-k-1}m) = \beta^{-k-1}m. \end{aligned}$$

Therefore, we must have $m - F_{t+t^{(k)}-1} \leqslant \beta^{-k-1}m$. □

Note that we always have $\beta^{-1}m \leqslant m - F_0 = m$. If we consecutively apply Lemma A.2.4 starting with $k = 0$, then, for $v(k) := \sum_{s=0}^{k} t^{(s)}$ we have $m - F_{v(k)-1} \leqslant \beta^{-k-1}m$. Therefore, for $k := \frac{\log \mathbb{E}d_{\mathtt{max}}}{\log \beta}$, we have $m - F_{v(k)-1} \leqslant \frac{m}{\mathbb{E}d_{\mathtt{max}}}$. We can bound

$$v(k) \leqslant \sum_{s=0}^{\infty} t^{(k)} = \frac{\beta-1}{\beta\tau(\alpha-1)}\frac{m}{\mathbb{E}d_{\mathtt{max}}} \sim \frac{m}{\mathbb{E}d_{\mathtt{max}}}.$$

From Lemma 2.4.4 we have $d_{\mathtt{max}} \lesssim \mathbb{E}d_{\mathtt{max}}$ with high probability. Together with Lemma A.1.1 this implies $\mathrm{val}_{\mathtt{LP}} \geqslant \frac{m}{d_{\mathtt{max}}} \gtrsim \frac{m}{\mathbb{E}d_{\mathtt{max}}}$. Now, if we pick $\tilde{K} := v(k) \lesssim \frac{m}{\mathbb{E}d_{\mathtt{max}}}$, we have that $\mathrm{val}_{\mathtt{Alg}} \lesssim \frac{m}{\mathbb{E}d_{\mathtt{max}}}$. Since $\mathrm{val}_{\mathtt{LP}} \leqslant \mathrm{val}_{\mathtt{Alg}}$, we have that $\tilde{K} \sim \mathrm{val}_{\mathtt{LP}}$ and $m - F_{\tilde{K}} \lesssim \tilde{K}$.

Case $mp \gg \log n$

Here, we have that $\mathbb{E}d_{\max} = mp(1+o(1))$, therefore, picking an element that hits an average number of subsets is approximately the same as picking an element that hits close to maximum number of subsets. From the properties of the mean and the median of the binomial distribution, it follows that $\Pr(\mathrm{Bin}(\tilde{m}, p) \geqslant \lceil \tilde{m}p \rceil) \geqslant 1/3$, for any $\tilde{m}$.

We begin with the case $\log mp \ll \log n$. This means that mp cannot grow polynomially in n, but e.g. $mp \sim \log^2 n$ is possible. In this regime, $\mathrm{val}_{\mathtt{IP}} \sim \frac{1}{p}\log\left(\frac{mp}{\log n}\right)$. Let $K_1 = \lceil \frac{1}{p}\log\left(\frac{mp}{\log n}\right)\rceil$ and $f_1, \ldots, f_{K_1}$ be a sequence such that $f_s = \lceil mp(1-p)^s \rceil$. Then, we have that $m - F_{K_1} \leqslant m(1-p)^{K_1} \leqslant \frac{1}{p}\log n$. Therefore, $(m - F_{K_1})p \sim \log n$, and we can continue with $\tilde{f}_t$ from the previous section $mp \sim \log n$, with $\tilde{F}_t := \sum_{s=1}^{t} \tilde{f}_t$. For this sequence $\tilde{f}_1, \ldots, \tilde{f}_{K_2}$, we have have $K_2 \lesssim \frac{1}{p}$, and $m - F_{K_1} - \tilde{F}_{K_2} \lesssim \frac{1}{p} \ll \frac{1}{p}\log\left(\frac{mp}{\log n}\right)$. The required statement holds for combined sequences f_t and $\tilde{f}_t$ and $\tilde{K} := K_1 + K_2$.

Finally, we study the case $\log mp \gtrsim \log n$, which implies that $\mathrm{val}_{\mathtt{IP}} \sim \frac{1}{p}\log n$. This case is trivial, as one can pick $\tilde{K} = \lceil \frac{1}{p}\log\left(\frac{mp}{\log n}\right)\rceil \lesssim \mathrm{val}_{\mathtt{IP}}$ and $f_1, \ldots, f_{\tilde{K}}$ a sequence such that $f_s = \lceil mp(1-p)^s \rceil$. Then, we have that $m - F_{\tilde{K}} \leqslant m(1-p)^{\tilde{K}} \leqslant \frac{1}{p}\log n \lesssim \mathrm{val}_{\mathtt{IP}}$.

From $m - F_{\tilde{K}} \lesssim \tilde{K}$ to $m - F_K \leqslant K$

Finally, using that $f_t \geqslant 1$ by Lemma A.1.2 unless $F_t = m$, there exists some constant $C > 0$, such that for $K := C\tilde{K}$, $F_K \geqslant m - K$, which finishes the proof. □

B

APPENDIX OF ROBUST

B.1 EXAMPLES

k INLIER CLUSTER, *c* OUTLIER CLUSTERS. One tricky adversarial distribution is the Gaussian mixture model itself. In particular, we consider

$$\mathcal{X}_c = \frac{k}{k+c}\sum_{i=1}^{k}\frac{1}{k}\mathcal{N}(\mu_i, I) + \frac{c}{k+c}\sum_{i=1}^{c}\frac{1}{c}\mathcal{N}(\tilde{\mu}_i, I), \tag{B.1.1}$$

where the first k Gaussian components are inliers and Q is a GMM with c components, which we call *fake* clusters. Since all inlier cluster weights are identical, we denote $w := w_i = 1/(k+c)$. Assume that $1 \ll c \ll k$, which corresponds to $\varepsilon \gg w$. Then, relative weights are $\tilde{w} = 1/(c+1) \approx 1/c$. Due to large adversary, previous results on learning GMMs cannot be applied, leaving vanilla list-decodable learning. However, the latter also cannot guarantee anything better that $\Omega(\sqrt{t}k^{1/t})$ even with the knowledge of k, as long as list size is $O(k+c)$, which can be much worse than our guarantees of $O(\sqrt{t}c^{1/t})$ for the same list size.

Their drawback is that they do not utilize separation between true clusters, i.e., for each i, they model the data as

$$\mathcal{X} = \frac{1}{k+c}\mathcal{N}(\mu_i, I) + \left(1 - \frac{1}{k+c}\right)Q.$$

where Q can be "arbitratily adversarial" for recovering μ_i.

BIG + SMALL INLIER CLUSTERS Consider the mixture

$$\mathcal{X}_b = (1 - w - \varepsilon)\,\mathcal{N}(\mu_1, I_d) + w\mathcal{N}(\mu_2, I_d) + \varepsilon Q, \tag{B.1.2}$$

where $\|\mu_1 - \mu_2\| = \Omega(\sqrt{\log 1/w})$, $w \ll \varepsilon \ll 1$, and Q is chosen adversarially. In this example we have two inlier clusters, one with large weight ≈ 1 and another with small weight w. Adversarial distribution Q has large weight relative to the small cluster, but still negligible weight compared to the large one.

Previous methods would either (i) recover large cluster with optimal error $O(\varepsilon)$ (see, e.g., [102]) but miss out small cluster or (ii) recover both

clusters using list-decodable mean estimation with known $\alpha = w$, but with suboptimal errors $O(\sqrt{\log 1/w})$ and list size $O(1/w)$. In contrast, Theorem 5.3.4 guarantees list size at most $1 + O(\varepsilon/w)$, error $O(\sqrt{\log \varepsilon/w})$ for the small cluster, and error $O(\varepsilon\sqrt{\log 1/\varepsilon})$ for the larger. In general, we achieve (i) optimal errors for both clusters and (ii) optimal (up to constants) list size.

B.2 INNER AND OUTER STAGE ALGORITHMS AND GUARANTEES

Our meta-algorithm Algorithm 4 assumes black-box access to a list-decodable mean estimation algorithm and a robust mean estimation algorithm for sub-Gaussian (up to the t^{th} moment) distributions. From these we obtain stronger mean estimation algorithms when the fraction of outliers is unknown, and finally stronger algorithms for learning separated mixtures when the fraction of outliers can be arbitrarily large. Our algorithm achieves guarantees with polynomial runtime and sample complexity if the black-box learners achieve the guarantees for their corresponding mean estimation setting. In this section we discuss the corruption model and inner and out stage of the meta-algorithm in detail and prove properties needed for the proof of the main Theorem 5.3.3.

B.2.1 *Detailed setting*

In order to achieve these guarantees, our black-box algorithms need to work under a model in which an adversary is allowed to remove a small fraction of the inliers and to add arbitrarily many outliers. In our proofs, for simplicity of exposition, we require the algorithms to have mean estimation guarantees for a small adversarially removed fraction of w_{low}^2. Formally, the corruption model as defined as follows.

Definition B.2.1 (Corruption model)**.** Let $d \in \mathbb{N}_+$, and $\alpha \in [w_{\text{low}}, 1]$. Let D be a d-dimensional distribution. An input of size n according to our corruption model is generated as follows:

- Draw a set C^* of $n_1 = \lceil \alpha n \rceil$ i.i.d. samples from the distribution D.
- An adversary is allowed to arbitrarily remove $\lfloor w_{\text{low}}^2 n_1 \rfloor$ samples from C^*. We refer to the resulting set as S^* with size $n_2 = |S^*|$.
- An adversary is allowed to add $n - n_2$ arbitrary points to S^*. We refer to the resulting set as S_{adv} with size $n_3 = |S_{\text{adv}}|$.

Algorithm 7 InnerStage

Input: Samples $S = \{x_1, \ldots, x_n\}$, $\alpha_{\text{low}} \in [w_{\text{low}}, 1]$, $\mathcal{A}_{\text{kLD}}$, and $\mathcal{A}_R$.
Output: List L.

1: $\alpha_{\text{low}} \leftarrow \min(1/100, \alpha_{\text{low}})$
2: $M \leftarrow \emptyset$
3: **for** $\hat{\alpha} \in \{\alpha_{\text{low}}, 2\alpha_{\text{low}}, \ldots, \lfloor 1/(3\alpha_{\text{low}}) \rfloor \alpha_{\text{low}}\}$ **do**
4: run $\mathcal{A}_{\text{kLD}}$ on S with fraction of inliers set to $\hat{\alpha}$
5: add the pair $(\hat{\mu}, \hat{\alpha})$ to M for each output $\hat{\mu}$
6: Let L be the output of ListFilter (Algorithm 8) run on S, α_{low}, and M
7: **for** $(\hat{\mu}, \hat{\alpha}) \in L$ **do**
8: replace $\hat{\mu}$ by the output of ImproveWithRME (Algorithm 9) run on S, $\hat{\mu}$, $\tau = 40\psi_t(\hat{\alpha}) + 4f(\hat{\alpha})$, and $\mathcal{A}_R$
9: **return** L

- If $n_3 < n$, pad S_{adv} with $n - n_3$ arbitrary points and call the resulting set S.
- Return S.

We call cor-kLD the model when w_{low} and α are given to the algorithm and cor-aLD the model when w_{low} and lower bound $\alpha_{\text{low}} \geqslant w_{\text{low}}$ are given to the algorithm, such that $\alpha \geqslant \alpha_{\text{low}}$. Note that α is **not** provided in cor-aLD model.

Note that in Theorem B.2.1 $|S| = n$ and S^* constitutes at least an $\alpha(1 - w_{\text{low}}^2)$-fraction of S.

B.2.2 *Inner stage algorithm and guarantees*

The algorithm consists of three steps: (1) Constructing a list of hypotheses, (2) Filtering the hypotheses, and (3) Improving the hypotheses if $\alpha \geqslant 1 - \varepsilon_{\text{RME}}$. For convenience, we restate the InnerStage algorithm introduced in the main text.

Theorem B.2.2 (Inner stage guarantees)**.** *Let $d \in \mathbb{N}_+$, $w_{\text{low}} \in (0, 10^{-4}]$, $w_{\text{low}} \leqslant \alpha_{\text{low}} \leqslant \alpha \leqslant 1$, and t be an even integer. Let $D(\mu^*)$ be a d-dimensional distribution with mean $\mu^* \in \mathbb{R}^d$ and sub-Gaussian t-th central moments.*

Consider the cor-aLD *corruption model in Theorem B.2.1 with parameters d, w_{low}, α and distribution $D = D(\mu^*)$. Let $\mathcal{A}_{\text{kLD}}$ and $\mathcal{A}_R$ satisfy Theorem 5.3.1 with high success probability (see Theorem B.2.3).*

Algorithm 8 ListFilter

Input: Samples $S = \{x_1, \dots, x_n\}$, $\alpha_{\text{low}} \in [w_{\text{low}}, 1/100]$, and $M = \{(\hat{\mu}_1, \hat{\alpha}_1), \dots, (\hat{\mu}_m, \hat{\alpha}_m)\}$
Output: List L

define $\beta(\alpha) = 10\psi_t(\alpha) + f(\alpha)$
let v_{ij} be a unit vector in the direction of $\hat{\mu}_i - \hat{\mu}_j$ for $\hat{\mu}_i \neq \hat{\mu}_j \in \{\hat{\mu}, \text{ for } (\hat{\mu}, \hat{\alpha}) \in M\}$
$J \leftarrow \emptyset$
for $(\hat{\mu}_i, \hat{\alpha}_i) \in M$ in decreasing order of $\hat{\alpha}_i$ **do**
    if exists $j \in J$, such that $\|\hat{\mu}_i - \hat{\mu}_j\| \leqslant 4\beta(\hat{\alpha}_i)$ **then continue**
    $T_i \leftarrow \bigcap_{j \in J} \{x \in S, \text{ s.t. } |v_{ij}^\top (x - \hat{\mu}_i)| \leqslant \beta(\hat{\alpha}_i)\}$.
    if $|T_i| < 0.9\hat{\alpha}_i n$ **then** remove $(\hat{\mu}_i, \hat{\alpha}_i)$ from M and **continue**
    add i to J
    for $j \in J \setminus \{i\}$ **do**
        $T_j \leftarrow T_j \bigcap \left\{x \in S, \text{ s.t. } |v_{ij}^\top (x - \hat{\mu}_j)| \leqslant \beta(\hat{\alpha}_i)\right\}$
        if $\left|T_j\right| < 0.9\hat{\alpha}_j n$ **then**:
            remove $(\hat{\mu}_j, \hat{\alpha}_j)$ from M
            rerun ListFilter (Algorithm 8) with the new M
return $\{(\hat{\mu}_i, \hat{\alpha}_i), \text{ for } i \in J\}$

Then InnerStage *(Algorithm 6), given an input of* $\text{poly}(d, 1/w_{\text{low}}) \cdot (N_{LD}(w_{\text{low}}) + N_R(w_{\text{low}}))$ *samples from the* cor-aLD *corruption model, and access to the parameters* d, w_{low}, α_{low}, *and* t, *runs in time* $\text{poly}(d, 1/w_{\text{low}}) \cdot (T_{LD}(w_{\text{low}}) + T_R(w_{\text{low}}))$ *and outputs a list* L *of size* $|L| \leqslant 1 + O((1-\alpha)/\alpha_{\text{low}})$ *such that, with probability* $1 - w_{\text{low}}^{O(1)}$,

1. *There exists* $\hat{\mu} \in L$ *such that*

$$\|\hat{\mu} - \mu^*\| \leqslant O(\psi_t(\alpha/4) + f(\alpha/4)).$$

2. *If* $\alpha \geqslant 1 - \varepsilon_{\text{RME}}$, *then there exists* $\hat{\mu} \in L$ *such that*

$$\|\hat{\mu} - \mu^*\| \leqslant O(g(\alpha - w_{\text{low}}^2)).$$

Proof of Theorem B.2.2 can be found in Appendix B.4.

Remark B.2.3. *For any* $r \in \mathbb{N}$, *we can increase probabilities of success of* $\mathcal{A}_{\text{kLD}}$ *and* $\mathcal{A}_R$ *from* $1/2$ *to* $1 - 2^{-r}$ *in the following way: we increase number of samples*

Algorithm 9 ImproveWithRME

Input: Samples $S = \{x_1, \ldots, x_n\}$, vector $\hat{\mu}$, threshold τ, and $\mathcal{A}_R$
Output: A vector $\tilde{\mu} \in \mathbb{R}^d$

$\tilde{\beta} \leftarrow \tau$
let $\tilde{\alpha}$ be the smallest value in $[1 - \varepsilon_{\text{RME}}, 1]$ that satisfies $g(\tilde{\alpha}) \leqslant \tilde{\beta}/2$. If none exists, **return** $\hat{\mu}$
$\tilde{\mu} \leftarrow \hat{\mu}$ and let μ_{RME} be the output of $\mathcal{A}_R$ run on S with inlier fraction set to $\tilde{\alpha}$.
while $\|\tilde{\mu} - \mu_{\text{RME}}\| \leqslant 3\tilde{\beta}/2$ **do**
    $\tilde{\mu} \leftarrow \mu_{\text{RME}}$
    $\tilde{\beta} \leftarrow g(\tilde{\alpha})$
    let $\tilde{\alpha}'$ be the smallest in $[\tilde{\alpha} + w_{\text{low}}^2, 1]$ such that $g(\tilde{\alpha}') \leqslant \tilde{\beta}/2$. If none exists, **break**
    $\tilde{\alpha} \leftarrow \tilde{\alpha}'$
    let μ_{RME} be the output of $\mathcal{A}_R$ on S with inlier fraction set to $\tilde{\alpha}$
return $\tilde{\mu}$

by a factor of r, randomly split S into r subsets of equal size, apply $\mathcal{A}_{\text{kLD}}$ and $\mathcal{A}_R$ to these subsets and concatenate their outputs. In the proofs we assume that the success probabilities are $1 - w_{\text{low}}^C$ for large enough constant C. This increases the size of the list returned by $\mathcal{A}_{\text{kLD}}$, the number of samples, and the running time by a factor $O(\log(1/w_{\text{low}}))$. In particular, we assume that the size of the list returned by $\mathcal{A}_{\text{kLD}}$ is much smaller than the inverse failure probability.

Remark B.2.4. *In the execution of the meta-algorithm, it may happen that Algorithm 6 is run on set T with* almost no *inliers, i.e., $\alpha < \alpha_{\text{low}}$. We note that from the analysis (see Appendix B.4, or [31], Proposition B.1), we always have upper bound $|L| = O(1/\alpha_{\text{low}})$.*

An immediate consequence of Theorem B.2.2 are the following guarantees of directly applying Algorithm 6 to the mixture learning case with no separation. Here we present upper bounds for Algorithm 6, when no separation assumptions are imposed.

Corollary B.2.5. *Let $d, k \in \mathbb{N}_+$, $w_{\text{low}} \in (0, 10^{-4}]$, and t be an even integer. For all $i = 1, \ldots, k$, let $D_i(\mu_i)$ be a d-dimensional distribution with mean $\mu_i \in \mathbb{R}^d$ and sub-Gaussian t-th central moments. Let $\varepsilon > 0$ and, for all $i = 1, \ldots, k$, let*

$w_i \in [w_{\text{low}}, 1]$, such that $\sum_{i=1}^{k} w_i + \varepsilon = 1$. Let $\mathcal{X}$ be the d-dimensional mixture distribution

$$\mathcal{X} = \sum_{i=1}^{k} w_i D_i(\mu_i) + \varepsilon Q,$$

where Q is an unknown adversarial distribution that can depend on all the other parameters. Let $\mathcal{A}_{\text{kLD}}$ and $\mathcal{A}_R$ satisfy Theorem 5.3.1.

Then there exists an algorithm that, given $\text{poly}(d, 1/w_{\text{low}}) \cdot (N_{LD}(w_{\text{low}}) + N_R(w_{\text{low}}))$ i.i.d. samples from $\mathcal{X}$, and given also d, k, w_{low}, and t, runs in time $\text{poly}(d, 1/w_{\text{low}}) \cdot (T_{LD}(w_{\text{low}}) + T_R(w_{\text{low}}))$ and outputs a list L of size $|L| = O(1/w_{\text{low}})$, such that, with probability at least $1 - w_{\text{low}}^{O(1)}$:

1. *For each $i \in [k]$, there exists $\hat{\mu} \in L$ such that*

$$\|\hat{\mu} - \mu_i\| \leqslant O(\psi_t(w_i/4) + f(w_i/4)).$$

2. *For each $i \in [k]$, if $w_i \geqslant 1 - \varepsilon_{\text{RME}}$, then there exists $\hat{\mu} \in L$ such that*

$$\|\hat{\mu} - \mu_i\| \leqslant O(g(w_i - w_{\text{low}}^2)).$$

Proof. Proof follows by applying Theorem B.2.2 to $\mathcal{X}$ with $\alpha_{\text{low}} = w_{\text{low}}$ and treating each component as a corresponding inlier distribution with $\alpha = w_i$. This gives error upper bound for all inlier components, furthermore, since $\alpha_{\text{low}} = w_{\text{low}}$, list size can be bounded as $|L| \leqslant 1 + O((1-\alpha)/\alpha_{\text{low}}) = O(1/w_{\text{low}})$. □

B.2.3 *Outer stage algorithm and guarantees*

In the outer stage, presented in Algorithm 10, we make use of the list-decodable mean estimation algorithm in Theorem B.2.2 in order to solve list-decodable mixture estimation with separated means. We now present results on the outer stage algorithm. For ease of notation, when it's clear from the context, we drop the indices and refer to elements $\mu_j \in M$ for some $j \in [|M|]$ as μ and their corresponding sets $S_j^{(1)}, S_j^{(2)}$, as defined in lines 6–7 in Algorithm 10, as $S^{(1)}, S^{(2)}$. Further, for $i \in [k]$, let C_i^* denote the set of points corresponding to the i-th inlier component, also called the i-th inlier cluster.

Theorem B.2.6 (Outer stage guarantees, beginning of execution)**.** *Let S consist of n i.i.d. samples from $\mathcal{X}$ as in the statement of Theorem B.3.1.*

Algorithm 10 OuterStage

Input: Samples $S = \{x_1, \dots, x_n\}$, w_{low}, $\mathcal{A}_{\text{sLD}}$
Output: Collection of sets $\mathcal{T}$

run $\mathcal{A}_{\text{sLD}}$ on S with $\alpha = w_{\text{low}}$ and let $M = \{\mu_1, \dots, \mu_{|M|}\}$ be the returned list
let v_{ij} be a unit vector in the direction of $\mu_i - \mu_j$ for $i \neq j \in [|M|]$
$\mathcal{T} \leftarrow \emptyset$ and $R \leftarrow \{1, \dots, |M|\}$
while $R \neq \emptyset$ **do**
  for all $i \in R$ **do**
    $S_i^{(1)} \leftarrow \bigcap_{j \in [|M|], j \neq i} \left\{ x \in S, \text{ s.t. } |v_{ij}^\top (x - \mu_i)| \leqslant \gamma + \gamma' \right\}$
    $S_i^{(2)} \leftarrow \bigcap_{j \in [|M|], j \neq i} \left\{ x \in S, \text{ s.t. } |v_{ij}^\top (x - \mu_i)| \leqslant 3\gamma + 3\gamma' \right\}$
  remove all $i \in R$ for which $|S_i^{(1)}| \leqslant 100 w_{\text{low}}^4 n$
  if $R = \emptyset$ **then break**
  if there exists $i \in R$ such that $|S_i^{(2)}| \leqslant 2|S_i^{(1)}|$ **then**
    select the $i \in R$ with $|S_i^{(2)}| \leqslant 2|S_i^{(1)}|$ for which $|S_i^{(1)}|$ is largest
    $\mathcal{T} \leftarrow \mathcal{T} \cup \left\{ S_i^{(2)} \right\}$
    $S \leftarrow S \setminus S_i^{(1)}$
    $R \leftarrow R \setminus \{i\}$
  else
    $\mathcal{T} \leftarrow \mathcal{T} \cup \{S\}$
    break
return $\mathcal{T}$

Run OuterStage *(Algorithm 10) on S and consider the first iteration of the while-loop and for each $\mu \in M$, denote the corresponding sets as $S^{(1)}, S^{(2)}$. Then, with probability at least $1 - w_{\text{low}}^{O(1)}$, we have that*

(i) the list M that $\mathcal{A}_{\text{sLD}}$ outputs has size $|M| \leqslant 2/w_{\text{low}}$,

(ii) for each $i \in [k]$, there exists $m_i \in [|M|]$ such that $\left|S_{m_i}^{(1)} \cap C_i^*\right| \geqslant (1 - \frac{w_{\text{low}}^2}{2}) \left|C_i^*\right|$,

(iii) for each $i \in [k]$ and $\mu \in M$, we have $\left|S^{(1)} \cap C_i^*\right| < w_{\text{low}}^4 \left|C_i^*\right|$ or $\left|S^{(2)} \cap C_i^*\right| \geqslant (1 - \frac{w_{\text{low}}^2}{2}) \left|C_i^*\right|$,

(iv) for each $i \in [k]$ *and* $\mu \in M$ *such that* $\left|S^{(2)} \cap C_i^*\right| \geqslant w_{\text{low}}^4 \left|C_i^*\right|$, *we have* $\sum_{i' \in [k] \setminus \{i\}} \left|S^{(2)} \cap C_{i'}^*\right| \leqslant w_{\text{low}}^4 n$,

(v) for $i \neq i' \in [k]$ *and for* $j, j' \in [|M|]$, *if* $|S_j^{(2)} \cap C_i^*| \geqslant w_{\text{low}}^4 |C_i^*|$ *and* $|S_{j'}^{(2)} \cap C_{i'}^*| \geqslant w_{\text{low}}^4 |C_{i'}^*|$, *then* $S_j^{(2)} \cap S_{j'}^{(2)} = \emptyset$.

In words, Theorem B.2.6 (ii) states that *at initialization*, OuterStage represents each inlier cluster well, i.e., for each i, the i-th cluster is almost entirely contained in some set $S_j^{(1)}$ for some $j \in [|M|]$. Next, (iii) states that either $S_j^{(1)}$ intersects negligibly some true component, or $S_j^{(2)}$ contains almost entirely the same component. Further, (iv) and (v) state that sets that sufficiently intersect with some true component must be separated from other components and each other.

We now introduce some notation to present the next theorem that establishes further guarantees for the algorithm output during execution. For $\mathcal{T}$, a collection of sets that is the output of Algorithm 10, we define

$$G := \Big\{ i \in [k], \text{ such that there exists } T = S_j^{(2)} \in \mathcal{T} \text{ with } \left|S_j^{(1)} \cap C_i^*\right| \geqslant w_{\text{low}}^4 \left|C_i^*\right|, \text{ for some } j \Big\}. \tag{B.2.1}$$

In words, it is the set of inlier components for which a corresponding set with "sufficiently many" points from the i-th component was added to $\mathcal{T}$. It may happen that for a given index $i \in G$, several $j \in [|M|]$ satisfy $S_j^{(2)} \in \mathcal{T}$ and $\left|S_j^{(1)} \cap C_i^*\right| \geqslant w_{\text{low}}^4 \left|C_i^*\right|$. We define $g_i \in [|M|]$ to denote the index of the *first* such set $S_{g_i}^{(2)}$ added to $\mathcal{T}$.

Further, we define $U_i := (C_i^* \cap S_{g_i}^{(2)}) \setminus S_{g_i}^{(1)}$ to be the set of inlier points from the i-th component, which were *not* removed from S at the iteration corresponding to g_i. Let $U := \cup_{i \in G} U_i$ denote the union of such 'left-over' inlier points.

Theorem B.2.7 (Outer stage guarantees, during execution)**.** *Let S consist of n i.i.d. samples from $\mathcal{X}$ as in the statement of Theorem B.3.1. Run* OuterStage *(Algorithm 10) on S and consider the moment when the sets $S_i^{(2)}$ are added to $\mathcal{T}$. We have that, with probability at least $1 - w_{\text{low}}^{O(1)}$, all of the following are true:*

(i) $|U| \leqslant (2\varepsilon + O(w_{\text{low}}^2))n$,

(ii) for $i \in G$, we have that $\left|S_{g_i}^{(2)} \cap C_i^*\right| \geqslant (1 - \frac{w_{\text{low}}^2}{2} - O(w_{\text{low}}^3))\left|C_i^*\right| \geqslant (1 - w_{\text{low}}^2)w_i n$,

(iii) for $j \in [|M|] \setminus \{g_i \,|\, i \in G\}$, either $\left|S_j^{(2)}\right| \leqslant O(w_{\text{low}}^2)n$, *or at least half of the samples in $S_j^{(1)}$ are either adversarial samples or lie in U,*

(iv) if when the else statement is triggered, $|S| \geqslant 0.1 w_{\text{low}} n$, then at least a 0.4-fraction of the samples in S are adversarial, or equivalently, $|S| \leqslant 2.5\varepsilon n$.

Note that the else statement of OuterStage can only be triggered once, at the end of the execution. In words, Theorem B.2.7 (i) states that, for $i \in G$, samples from i-th cluster that remained in S after $S_{g_i}^{(1)}$ was removed, constitute a small (comparable with ε) fraction. Further, (ii) states that the sets *added to* $\mathcal{T}$, corresponding to $i \in G$, almost entirely contain C_i^*. Finally, (iii) describes the sets that do not correspond to any $g_i, i \in G$. These sets must either be small, or contain a significant amount of outlier points in the neighborhood. The proofs of Theorems B.2.6 and B.2.7 can be found in Appendix B.5.

B.3 PROOF OF THEOREM 5.3.3

In this section, we state and prove a refined version of our main result, Theorem B.3.1, from which the statement of Theorem 5.3.3 directly follows.

B.3.1 *General theorem statement*

We define

$$\psi_t(\alpha) = \begin{cases} \sqrt{t}(1/\alpha)^{1/t} & \text{if } t \leqslant 2\log 1/\alpha, \\ \sqrt{2e\log 1/\alpha} & \text{else,} \end{cases} \tag{B.3.1}$$

which captures a tail decay of a distribution with sub-Gaussian t-th central moments: $\Pr_{x\sim\mathcal{D}}\left(\langle x-\mu, v\rangle^t \geqslant \psi_t(\alpha)\right) \lesssim \alpha$.

We now state our main result for list-decodable mixture learning. Recall that ε_{RME} is defined in Theorem 5.3.1.

Theorem B.3.1 (Main mixture model result). *Let $d, k \in \mathbb{N}_+$, $w_{\text{low}} \in (0, 10^{-4}]$, and t be an even integer. For all $i = 1, \ldots, k$, let $D_i(\mu_i)$ be a d-dimensional distribution with mean $\mu_i \in \mathbb{R}^d$ and sub-Gaussian t-th central moments. Let $\varepsilon > 0$*

and, for all $i = 1, \dots, k$, let $w_i \in [w_{\text{low}}, 1]$, such that $\sum_{i=1}^{k} w_i + \varepsilon = 1$. Let $\mathcal{X}$ be the d-dimensional mixture distribution

$$\mathcal{X} = \sum_{i=1}^{k} w_i D_i(\mu_i) + \varepsilon Q,$$

where Q is an unknown adversarial distribution that can depend on all the other parameters. Let $\mathcal{A}_{\text{kLD}}$ and $\mathcal{A}_R$ satisfy Theorem 5.3.1. Further, suppose that $\|\mu_i - \mu_j\| \geqslant 200\psi_t(w_{\text{low}}^4) + 200f(w_{\text{low}})$ for all $i \neq j \in [k]$.

Then there exists an algorithm (Algorithm 4) that, given $\text{poly}(d, 1/w_{\text{low}}) \cdot (N_{LD}(w_{\text{low}}) + N_R(w_{\text{low}}))$ i.i.d. samples from $\mathcal{X}$, and given also d, k, w_{low}, and t, runs in time $\text{poly}(d, 1/w_{\text{low}}) \cdot (T_{LD}(w_{\text{low}}) + T_R(w_{\text{low}}))$ and with probability at least $1 - w_{\text{low}}^{O(1)}$ outputs a list L of size $|L| \leqslant k + O(\varepsilon/w_{\text{low}})$ such that, for each $i \in [k]$, there exists $\hat{\mu} \in L$ such that:

$$\|\hat{\mu} - \mu_i\| \leqslant O(\psi_t(\tilde{w}_i/10) + f(\tilde{w}_i/10)), \qquad \text{where} \quad \tilde{w}_i = w_i/(w_i + \varepsilon + w_{\text{low}}^2).$$

If the relative weight of the i-th inlier cluster is large, i.e., $\tilde{w}_i \geqslant 1 - \varepsilon_{\text{RME}} + 2w_{\text{low}}^2$, then there exists $\hat{\mu} \in L$ such that

$$\|\hat{\mu} - \mu_i\| \leqslant O(g(\tilde{w}_i - 3w_{\text{low}}^2)).$$

Further, we assume $w_{\text{low}} \in (0, 1/10000]$, since this simplifies some of the proofs. We note that in a mixture with k components we necessarily have $w_{\text{low}} \leqslant 1/k$. Furthermore, when $w_{\text{low}} \in (1/10000, 1/2]$, then we obtain the same result by replacing w_{low} with $w_{\text{low}}/5000$ throughout the statements and the proof. This would only affect both list size and error guarantees by at most a multiplicative constant, which is absorbed in the Big-O notation.

Proof of Theorem 5.3.3. Proof follows directly from Theorem B.3.1, by noticing that Theorem 5.3.2 allows to replace $f(\tilde{w}_i/10)$ by $Cf(\tilde{w}_i)$ and $g(\tilde{w}_i - 3w_{\text{low}}^2)$ by $Cg(\tilde{w}_i)$ for some constant $C > 0$ large enough. □

B.3.2 *Proof of Theorem B.3.1*

We now show how to use the results on the inner and outer stage, Theorem B.2.2 and Theorem B.2.7 respectively, to arrive at the guarantees for the full algorithm Algorithm 4 in Theorem B.3.1. For simplicity of the exposition, we split the proof of Theorem B.3.1 into two separate parts, proving that (i) the output list contains an estimate with small error and that (ii) the size of the output list is small. In what follows we condition on the event E' from the proof of Theorem B.2.7.

(1) PROOF OF ERROR STATEMENT We now prove that, conditioned on the event E, the list L output by Algorithm 4 for each $i \in [k]$ contains an estimate $\hat{\mu} \in L$, such that,

(1) $\|\hat{\mu} - \mu_i\| \leqslant O(\psi_t(\tilde{w}_i/10) + f(\tilde{w}_i/10))$,

(2) if $\tilde{w}_i \geqslant 1 - \varepsilon_{\mathrm{RME}} + 2w_{\mathrm{low}}^2$, then $\|\hat{\mu} - \mu_i\| \leqslant O(g(\tilde{w}_i - 3w_{\mathrm{low}}^2))$.

We start by showing that list-decoding error guarantees as in (1) are achievable for all inlier clusters and proceed by improving the error to (2) with RME base learner. Recall that G is as defined in eq. (B.2.1).

Proof of (1) We now show how the output of the base learner and filtering procedure lead to the error in (1). Fix $i \in [k]$. Recall that C_i denotes the set of $w_i n$ points from i-th inlier component with mean μ_i.

If $i \in G$, then on event E, we have $\left|S_{g_i}^{(2)} \cap C_i^*\right| \geqslant (1 - w_{\mathrm{low}}^2) w_i n$ by Theorem B.2.7 (ii), $\sum_{j \neq i} \left|S_{g_i}^{(2)} \cap C_j^*\right| \leqslant w_{\mathrm{low}}^4 n$ by Theorem B.2.6 (iv), and that the total number of adversarial points is at most $(\varepsilon + w_{\mathrm{low}}^4)n$.

Therefore, the fraction of points from C_i^* in $S_{g_i}^{(2)}$ is at least $\frac{(1-w_{\mathrm{low}}^2)w_i}{w_i+\varepsilon+w_{\mathrm{low}}^3}$, which implies $\alpha \geqslant \tilde{w}_i$ as in Theorem B.2.1. Then, by Theorem B.2.2, the InnerStage algorithm applied to T leads to error $\|\hat{\mu} - \mu_i\| \leqslant O(\psi_t(\tilde{w}_i/4) + f(\tilde{w}_i/4))$. Otherwise, if $i \notin G$, when the OuterStage algorithm reaches the else statement, S contains at least $(1 - O(w_{\mathrm{low}}^3))|C_i^*|$ samples from C_i^*. Indeed, since $i \notin G$, each time we remove points from S, we remove at most $w_{\mathrm{low}}^4 n$ points from C_i^*. By Theorem B.2.6 (i), we do at most $O(1/w_{\mathrm{low}})$ removals, so when the OuterStage algorithm reaches the else statement, S contains at least $(1 - O(w_{\mathrm{low}}^3))\left|C_i^*\right|$ samples from C_i^*.

We showed that samples from C_i^* make up at least a $(1 - w_{\mathrm{low}}^2)w_i n/|S|$ fraction of S. Based on this fact we can then use Theorem B.2.7 (iv) and the assumption on the range of w_{low} to conclude that $|S| \leqslant 2.5\varepsilon n$ and that the fraction of inliers is at least $(1 - w_{\mathrm{low}}^2)w_i/(2.5\varepsilon)$. Therefore, S can be seen as containing samples from the corruption model cor-aLD with α at least $w_i/(2.5\varepsilon) \geqslant w_i/(2.5(w_i + \varepsilon))$. Since S is added to $\mathcal{T}$ in the else statement, applying InnerStage yields the error bound as in (1).

Proof of (2): Next, we prove that for all inlier components i with large weight, i.e., such that $w_i/(w_i + \varepsilon) \geqslant 1 - \varepsilon_{\mathrm{RME}}$, there exists a set $T \in \mathcal{T}$ that consists of samples from the corruption model cor-aLD with $\alpha \geqslant w_i/(w_i + \varepsilon) - 2w_{\mathrm{low}}^2$. Then, running InnerStage, in particular the RME base learner, results in the error bound as in (2) by Theorem B.2.2 (ii). If $i \in$

G, in the previous paragraph we showed that there exists $T \in \mathcal{T}$, such that the corresponding $\alpha \geqslant \frac{w_i}{w_i+\varepsilon+w_{\text{low}}^3} \geqslant \frac{w_i}{w_i+\varepsilon} - 2w_{\text{low}}^2$. We now prove by contradiction that the case $i \notin G$ does not occur. Now assume $i \notin G$ so that as we argued before when the else statement is triggered, S contains at least $(1 - O(w_{\text{low}}^3))|C_i^*|$ samples from C_i^*. By Theorem B.2.6 (ii), for some $m_i \in [|M|]$, we have that $|S_{m_i}^{(1)} \cap C_i^*| \geqslant (1 - w_{\text{low}}^2/2 - O(w_{\text{low}}^3))|C_i^*|$ and by Theorem B.2.6 (iv), $S_{m_i}^{(2)}$ contains at most $w_{\text{low}}^4 n$ samples from other true clusters. Then, since $|S_{m_i}^{(2)}| > 2|S_{m_i}^{(1)}|$, we have that $|S_{m_i}^{(2)}|$ contains at least

$$(1 - w_{\text{low}}^2 - O(w_{\text{low}}^3))|C_i^*| - w_{\text{low}}^4 n \geqslant (1 - 1.5w_{\text{low}}^2)|C_i^*|$$

adversarial samples. Therefore, $\varepsilon \geqslant (1 - 1.5w_{\text{low}}^2)|C_i^*|/n$, and using that $|C_i^*| \geqslant w_i n - w_{\text{low}}^{10} n$, we have $\varepsilon \geqslant (1 - 1.5w_{\text{low}}^2)w_i - w_{\text{low}}^{10}$. However, this contradicts $w_i/(w_i + \varepsilon) \geqslant 1 - \varepsilon_{\text{RME}}$ unless $\varepsilon_{\text{RME}} \geqslant 1/2 - 2w_{\text{low}}^2$, which is prohibited by the assumptions in Theorem B.3.1. Therefore whenever $w_i/(w_i + \varepsilon) \geqslant 1 - \varepsilon_{\text{RME}}$ we are in the case $i \in G$.

(II) PROOF OF SMALL LIST SIZE We now prove that on the set E, we have that $|L| \leqslant k + O(\varepsilon/w_{\text{low}})$. Here, we need to carefully analyze iterations in the while loop where an inlier component is "selected" for the first time in order to obtain a tight list size bound. Recall that g_i corresponds to the index in R that is first selected for the i-th inlier cluster.

First selection of a component: For any $i \in [k]$, if $i \in G$, then Theorem B.2.7 (ii) implies that running InnerStage on $S_{g_i}^{(2)}$ produces a list of size at most $1 + O((|S_{g_i}^{(2)} \setminus C_i^*|)/(w_{\text{low}} n))$. Then, over all $i \in G$, these sets $S_{g_i}^{(2)}$ contribute to the list size $|L|$ at most $k + O\left(\sum_{i=1}^k |S_{g_i}^{(2)} \setminus C_i^*|/(w_{\text{low}} n)\right)$. Furthermore, by Theorem B.2.6 (v), all these sets $S_{g_i}^{(2)}$ are disjoint and each of them contains at most $w_{\text{low}}^4 n$ samples from other true clusters. Therefore $\sum_{i=1}^k |S_{g_i}^{(2)} \setminus C_i^*| \leqslant \varepsilon n + O(w_{\text{low}}^3) n$. Then the contribution to $|L|$ of all these S_i's corresponding to true clusters is at most $k + O\left((\varepsilon + w_{\text{low}}^3)/w_{\text{low}}\right)$. Note that if $\varepsilon \leqslant w_{\text{low}}^3$ and w_{low} is small enough, Algorithm 6 actually produces a list of size 1 in each run considered above, so the contribution is exactly k; otherwise we can bound the contribution by $k + O(\varepsilon/w_{\text{low}})$.

Samples left over from a component: Next, all inlier samples that were not removed, i.e., constituting U, can be considered outlier points for the

future iterations, which, by Theorem B.2.7 (i), only increases the outlier fraction to $\tilde{\varepsilon} = 3\varepsilon + O(w_{\text{low}}^2)$. For the same reason as above, without loss of generality, we can consider $\varepsilon > w_{\text{low}}^2$ since otherwise, the corresponding list size overhead (for small enough w_{low}^2) would again amount to zero.

Clusters of adversarial samples: For iterations where a set $S_j^{(2)}$ was added to the final list, which does not correspond to some $g_i, i \in G$, Theorem B.2.7 (iii), states that either (i) at least half of the samples in $S_j^{(2)}$ were adversarial, or (ii) the cardinality of the set on which Algorithm 6 was executed is small. In both cases the set $S_j^{(2)}$ contributes at most $O(\varepsilon/w_{\text{low}})$ to the final list size $|L|$.

List size in the else statement: Finally, when the algorithm reaches the else statement, as argued in the first part, by Theorem B.2.7 (iv), at that iteration $|S| \leqslant O(\varepsilon)n$. Since Algorithm 6 always produces a list of size bounded by $O(|S| / (w_{\text{low}} n))$ (see Theorem B.2.4), the contribution to $|L|$ at this iteration is bounded by $O(\varepsilon/w_{\text{low}})$.

Overall, we obtain the desired bound on $|L|$ of $k + O(\varepsilon/w_{\text{low}})$.

B.4 PROOF OF THEOREM B.2.2

(1) PROOF OF ERROR STATEMENT We now prove that, with probability $1 - w_{\text{low}}^{O(1)}$, for the output list L of Algorithm 10,

1. there exists $\hat{\mu} \in L$ such that

$$\|\hat{\mu} - \mu^*\| \leqslant O(\psi_t(\alpha/4) + f(\alpha/4)),$$

2. if $\alpha \geqslant 1 - \varepsilon_{\text{RME}}$, then there exists $\hat{\mu} \in L$ such that

$$\|\hat{\mu} - \mu^*\| \leqslant O(g(\alpha - w_{\text{low}}^2)).$$

By Theorem B.4.1 we have $|M| \leqslant 1/w_{\text{low}}^{O(1)}$ and, with probability at least $1 - w_{\text{low}}^{O(1)}$, there exists $(\hat{\mu}, \hat{\alpha}) \in M$ such that $\hat{\alpha} \geqslant \alpha/4$ and $\|\hat{\mu} - \mu^*\| \leqslant f(\hat{\alpha})$. Then Theorem B.4.2 implies that, with probability at least $1 - |M|^2 w_{\text{low}}^{O(1)}$, $(\hat{\mu}, \hat{\alpha})$ will not be removed from M. Therefore, either $(\hat{\mu}, \hat{\alpha}) \in L$, or there exists $(\tilde{\mu}, \tilde{\alpha}) \in L$ such that (i) $\tilde{\alpha} \geqslant \hat{\alpha}$ and (ii) $\|\tilde{\mu} - \hat{\mu}\| \leqslant 4\beta(\hat{\alpha})$. The latter case implies that $\|\tilde{\mu} - \mu^*\| \leqslant 40\psi_t(\alpha/4) + 4f(\alpha/4)$.

For the second part, set first $\tilde{\mu} = \hat{\mu}$. Then, in the i^{th} iteration, $\tilde{\mu}$ moves away by at most $(3\tau/2)/2^{i-1}$. Since $\sum_{i=1}^{\infty} 1/2^i \leqslant 1$, the distance between $\tilde{\mu}$ and $\hat{\mu}$ is always bounded by 3τ. Now, assume that indeed $\alpha \geqslant 1 - \varepsilon_{\text{RME}}$ and $\|\hat{\mu} - \mu^*\| \leqslant \tau$. Whenever $\tilde{\alpha} \leqslant \alpha$, with high probability $\mathcal{A}_R$ produces some μ_{RME} such that $\|\mu_{\text{RME}} - \mu^*\| \leqslant g(\tilde{\alpha}) \leqslant \tilde{\beta}/2$. Furthermore, as long as $\tilde{\alpha} \leqslant \alpha$, at the moment of the while statement check we have $\|\tilde{\mu} - \mu^*\| \leqslant \tilde{\beta}$: in the first iteration this is by assumption, and in later iterations it follows because $\tilde{\mu}$ is the former μ_{RME}. Therefore the while statement check passes as long as $\tilde{\alpha} \leqslant \alpha$.

There exists the possibility that the algorithm returns or breaks even though $\tilde{\alpha} \leqslant \alpha$. If the algorithm returns early, then the error τ achieved by $\hat{\mu}$ is already within a factor of two of the optimal. If the algorithm breaks, either $\tilde{\alpha} + w_{\text{low}}^2 > 1$, case in which $\tilde{\mu}$ already satisfies $\|\tilde{\mu} - \mu^*\| \leqslant g(1 - w_{\text{low}}^2)$, or else $\|\tilde{\mu} - \mu^*\|$ is already within a factor of two of the optimal. Therefore these cases do not affect the error negatively.

Finally, let us consider what happens when $\tilde{\alpha} > \alpha$ and the while statement check continues to pass. The first time we reach some $\tilde{\alpha} > \alpha$, we must have $\|\tilde{\mu} - \mu^*\| \leqslant \tilde{\beta} \leqslant 2g(\alpha - w_{\text{low}}^2)$. Then, in later iterations, $\tilde{\mu}$ can move from this estimate by a distance of most $3\tilde{\beta} \leqslant 6g(\alpha - w_{\text{low}}^2)$, by the same argument as the argument that $\|\tilde{\mu} - \hat{\mu}\| \leqslant 3\tau$. Overall, at the end we have

$$\|\tilde{\mu} - \hat{\mu}\| \leqslant \max(2g(1), g(1 - w_{\text{low}}^2), 8g(\alpha)) \leqslant 8g(\alpha - w_{\text{low}}^2).$$

The number of runs is at most $1/w_{\text{low}}^2$, so with probability $1 - w_{\text{low}}^{O(1)}$ all runs of $\mathcal{A}_R$ succeed.

We showed that there exists $(\hat{\mu}, \hat{\alpha}) \in L$ such that $\hat{\alpha} \geqslant \alpha/4$ and $\|\hat{\mu} - \mu^*\| \leqslant 40\psi_t(\hat{\alpha}) + 4f(\hat{\alpha})$. This error can increase by running ImproveWithRME with $\tau = 40\psi_t(\hat{\alpha}) + 4f(\hat{\alpha})$ to at most

$$\|\hat{\mu} - \mu^*\| \leqslant 160\psi_t(\hat{\alpha}) + 16f(\hat{\alpha}) = O(\psi_t(\alpha) + f(\alpha)).$$

Furthermore, if $\alpha \geqslant 1 - \tau_{\min}$, this $(\hat{\mu}, \hat{\alpha}) \in L$ satisfies the conditions of ImproveWithRME, so with high probability the error is reduced by running ImproveWithRME with $\tau = 40\psi_t(\hat{\alpha}) + 4f(\hat{\alpha})$ to $\|\hat{\mu} - \mu^*\| \leqslant 8g(\alpha - w_{\text{low}}^2)$.

(II) PROOF OF LIST SIZE We now prove that $|L| \leqslant 1 + O((1-\alpha)/\alpha_{\text{low}})$.

First, assume that $\alpha \leqslant 9/10$. Since all $\hat{\alpha}_s \geqslant \alpha_{\text{low}}$, we have that $|L| \leqslant 10/(9\alpha_{\text{low}}) \leqslant 12(1-\alpha)/\alpha_{\text{low}}$.

For the rest of the proof we assume that $\alpha > 9/10$. We analyze sets J and T_i for $i \in J$ at the end of execution of ListFilter. In particular, recall that

$L = \{(\hat{\mu}_i, \hat{\alpha}_i),\ i \in J\}$. Also, at the end of Algorithm 8 we have the following expression for T_i:

$$T_i = \bigcap_{j \in J \setminus \{i\}} \{x \in S,\ \text{s.t. } |v_{ij}^\top (x - \hat{\mu}_i)| \leqslant \max(\beta(\hat{\alpha}_i), \beta(\hat{\alpha}_j))\},$$

where v_{ij} are unit vectors in direction $\hat{\mu}_i - \hat{\mu}_j$. Select the $s \in J$ for which $\hat{\alpha}_s$ is maximized. By (i), with probability at least $1 - w_{\text{low}}^{60}$, there exists a hypothesis in J with $\hat{\alpha} \geqslant \alpha/4 \geqslant 0.2$. Then we have that $\hat{\alpha}_s \geqslant 0.2$. In addition, for all hypotheses, $\hat{\alpha}_s \leqslant 1/3$. Let $j \in J$ be such that $j \neq s$. We will show that at least half of the points in T_j are adversarial, i.e., $\left|T_j\right| \geqslant 2\left|T_j \cap C^*\right|$. If this is indeed the case, we can treat all inlier points in all T_j as outliers, as it would at most double total number of outlier points in S.

Now, assume that for some $j \neq s$, $\left|T_j\right| < 2\left|T_j \cap C^*\right|$. Note that, because $|T_s| \geqslant 0.9 \cdot 0.2n = 0.18n$ and $|C^*| \geqslant 0.9n$, $|T_s \cap C^*| \geqslant 0.18n - 0.1n \geqslant 0.08n$. Therefore

$$\left|\left\{x \in C^*, \text{ s.t. } |v_{sj}^\top (x - \hat{\mu}_s)| \leqslant \beta(\hat{\alpha}_s)\right\}\right| \geqslant 0.08\,|C^*|\,.$$

Also note that Theorem B.8.2 for radius $10\psi_t(1/2) \leqslant 10\psi_t(\hat{\alpha}_s)$ implies that, with exponentially small failure probability,

$$\left|\left\{x \in C^*, \text{ s.t. } |v_{sj}^\top (x - \mu^*)| \leqslant \beta(\hat{\alpha}_s)\right\}\right| \geqslant 0.99\,|C^*|\,.$$

Since these two sets necessarily intersect, we can bound $|v_{sj}^\top (\hat{\mu}_s - \mu^*)| \leqslant 2\beta(\hat{\alpha}_s)$, implying that $|v_{sj}^\top (\hat{\mu}_j - \mu^*)| \geqslant 2\beta(\hat{\alpha}_j)$, since $\|\hat{\mu}_s - \hat{\mu}_j\| \geqslant 4\beta(\hat{\alpha}_j)$. Thus, if $|v_{sj}^\top (x - \hat{\mu}_j)| \leqslant \beta(\hat{\alpha}_j)$, then $|v_{sj}^\top (x - \mu^*)| > \beta(\hat{\alpha}_j)$, implying that

$$(T_j \cap C^*) \subseteq \left\{x \in C^*, \text{ s.t. } |v_{sj}^\top (x - \mu^*)| > \beta(\hat{\alpha}_j)\right\}. \tag{B.4.1}$$

However, Theorem B.8.2 tells us that with high probability only a small fraction of points in C^* satisfies $|v_{sj}^\top (x - \mu^*)| > \beta(\hat{\alpha}_j)$. In particular, applying the lemma with radius $10\psi_t(\hat{\alpha}_j)$, we get that with exponentially small failure probability,

$$\left|\left\{x \in C^*, \text{s.t. } |v_{sj}^\top (x - \mu^*)| \leqslant \beta(\hat{\alpha}_j)\right\}\right| \geqslant \left(1 - \frac{\hat{\alpha}_j}{50}\right)|C^*|\,. \tag{B.4.2}$$

From eqs. (B.4.1) and (B.4.2) it follows that $\left|T_j \cap C^*\right| \leqslant \hat{\alpha}_j\,|C^*|\,/50$. Using that $\left|T_j\right| \geqslant 9\hat{\alpha}_j n/10 \geqslant 9\hat{\alpha}_j\,|C^*|\,/10$ by the properties of ListFilter, we obtain

$$9\hat{\alpha}_j\,|C^*|\,/10 \leqslant \left|T_j\right| \leqslant 2\left|T_j \cap C^*\right| \leqslant \hat{\alpha}_j\,|C^*|\,/25,$$

which is a contradiction.

Therefore, for all $j \in J$ such that $j \neq s$, we have that $|T_j| \geqslant 2|T_j \cap C^*|$. As we said in the beginning, by treating all inlier points in those T_j as outliers we at most double total number of outlier points. Since there are at most $(1-\alpha+\alpha w_{\text{low}}^2)n$ outlier points and the sets T_j are non-intersecting, we get $\sum_{j\in J\setminus\{s\}} |T_j| \leqslant 2(1-\alpha+\alpha w_{\text{low}}^2)n$. The lower bound on the size $|T_j| \geqslant 9\alpha_{\text{low}} n/10$ implies $|J/\{s\}| \leqslant \frac{2(1-\alpha+\alpha w_{\text{low}}^2)n\cdot 10}{9\alpha_{\text{low}} n}$ and thus $|L| = |J| \leqslant 1 + 3(1-\alpha)/\alpha_{\text{low}}$.

Note that in InnerStage we set $\alpha_{\text{low}} = \min(\alpha_{\text{low}}, 1/100)$. Therefore, for the original α_{low}, the list size is bounded by $1 + 240(1-\alpha)/\alpha_{\text{low}}$.

CONCLUSION Combining the probabilities of success of all steps, we get that the algorithm succeeds with probability at least $1 - w_{\text{low}}^{O(1)}$ for some large constant in the exponent. Our algorithm, ignoring the calls to $\mathcal{A}_{\text{kLD}}$ and $\mathcal{A}_R$, has sample complexity and time complexity bounded by $\text{poly}(d, 1/w_{\text{low}})$, which gives the desired sample and time complexity when taking $\mathcal{A}_{\text{kLD}}$ and $\mathcal{A}_R$ into consideration. This completes the proof of the theorem.

B.4.1 *Auxiliary lemmas and proofs*

Lemma B.4.1 (List initialization)**.** *Let S, α_{low} and α be as in* cor-aLD *model. If* InnerStage *(Algorithm 6) is run with S and α_{low}, the size of M is at most $1/w_{\text{low}}^{O(1)}$, all $(\hat{\mu}, \hat{\alpha}) \in M$ satisfy $\hat{\alpha} \leqslant 1/3$, and with probability at least $1 - w_{\text{low}}^{O(1)}$ there exists $(\hat{\mu}, \hat{\alpha}) \in M$ such that $\alpha/4 \leqslant \hat{\alpha} \leqslant \min(\alpha, 1/3)$ and $\|\hat{\mu} - \mu^*\| \leqslant f(\hat{\alpha})$.*

Proof. There are at most $1/\alpha_{\text{low}}$ choices for $\hat{\alpha}$, and for each of them the output of $\mathcal{A}_{\text{kLD}}$ has size at most $1/w_{\text{low}}^{O(1)}$, so $|M| \leqslant 1/w_{\text{low}}^{O(1)}$. With probability $1 - w_{\text{low}}^{O(1)}$, $\mathcal{A}_{\text{kLD}}$ succeeds in all up to $1/\alpha_{\text{low}}$ runs. Then we are guaranteed to produce one $\hat{\alpha}$ with $\alpha/4 \leqslant \hat{\alpha} \leqslant \min(\alpha, 1/3)$, and then $\mathcal{A}_{\text{kLD}}$ is guaranteed to produce one corresponding $\hat{\mu}$ with $\|\hat{\mu} - \mu^*\| \leqslant f(\hat{\alpha})$. □

Lemma B.4.2 (Good hypotheses are not removed)**.** *Let S, α_{low} and α be as in* cor-aLD *model. Run* ListFilter *(Algorithm 8) on S and α_{low} with M obtained from* InnerStage *(Algorithm 6) and call a hypothesis $(\hat{\mu}, \hat{\alpha}) \in M$ good if $\hat{\alpha} \geqslant \alpha/4$ and $\|\hat{\mu} - \mu^*\| \leqslant f(\hat{\alpha})$. Then, with probability at least $|M|^2 w_{\text{low}}^{O(1)}$, no good hypothesis is removed from M (including in any of the reruns triggered by the algorithm).*

Proof. Let ℓ be an arbitrary iteration of the outer for loop. Then, at the beginning of the ℓ^{th} iteration,

1. $T_j \cap T_s = \emptyset$ for any $j < s \in J$,
2. $\left|T_j\right| \geqslant 0.9\hat{\alpha}_j n$ for any $j \in J$,
3. $|J| \leqslant 10/(9\hat{\alpha}_\ell)$.

Indeed, the second property follows directly from the algorithm.

For the first property, assume that for $j < s \in J$, there exists $x \in S$, such that $x \in T_j \cap T_s$. This would imply that $|v_{js}^\top(\hat{\mu}_j - \hat{\mu}_s)| \leqslant 2\beta(\hat{\alpha}_s)$, so $\|\hat{\mu}_j - \hat{\mu}_s\| \leqslant 2\beta(\hat{\alpha}_s)$. However, in this case the first 'if' condition would pass and we would not add s to J. Thus, $T_j \cap T_s = \emptyset$.

For the third property, note that

$$n \geqslant \sum_{j\in J} \left|T_j\right| \geqslant \sum_{j\in J} 0.9\hat{\alpha}_j n \geqslant 0.9\,|J|\,\hat{\alpha}_\ell n,$$

which implies $|J| \leqslant 10/(9\hat{\alpha}_\ell)$.

Now, let s be the index of a hypothesis with $\hat{\alpha}_s \geqslant \alpha/4$ and $\|\hat{\mu}_s - \mu^*\| \leqslant f(\hat{\alpha}_s)$. If s was skipped in the s^{th} iteration (i.e., there exists $j \in J$ with $\hat{\mu}_j$ close to $\hat{\mu}_s$), then $(\hat{\mu}_s, \hat{\alpha}_s)$ is trivially not removed from M. For the rest of the proof we assume that s is not skipped.

For the sake of the analysis, we introduce the analogue of the sets T_s, which we call $\tilde{T}_s$, defined for points in the set C^* (i.e., all inlier points before the adversarial removal), and show that (i) $\left|\tilde{T}_s\right|$ is large and (ii) $\left|\tilde{T}_s \setminus T_s\right|$ is small. In particular, let

$$\tilde{T}_s = \bigcap_{j\in J} \left\{ x \in C^*, \text{ s.t. } |v_{js}^\top(x - \hat{\mu}_s)| \leqslant \beta(\hat{\alpha}_s) \right\},$$

where we recall $\beta(\hat{\alpha}_s) = 10\psi_t(\hat{\alpha}_s) + f(\hat{\alpha}_s)$. Note that $|T_s| \geqslant \left|\tilde{T}_s\right| - |C^* \setminus S^*| \geqslant \left|\tilde{T}_s\right| - w_{\text{low}}^2 |C^*|$. Also, for any $\alpha' \leqslant \hat{\alpha}_s$, applying Theorem B.8.2 with radius $10\psi_t(\alpha')$, using that $\|\hat{\mu}_s - \mu^*\| \leqslant f(\hat{\alpha}_s) \leqslant f(\alpha')$ and $t \geqslant 2$, we get that with exponentially small failure probability,

$$\left|\left\{ x \in C^*, \text{s.t. } |v^\top(x - \hat{\mu}_s)| > \beta(\alpha') \right\}\right| \leqslant \frac{\alpha'}{50} |C^*|. \tag{B.4.3}$$

Consider the s^{th} iteration. Using a union bound over $|J| \leqslant 2/\alpha_{\text{low}}$ directions, and since all $\hat{\alpha}_s \geqslant \alpha_{\text{low}}$, we get that with exponentially small failure probability

$$\begin{aligned}\left|\tilde{T}_s\right| &\geqslant |C^*| - \sum_{i\in J}\left|\left\{x \in C^*, \text{ s.t. } |v_{is}^\top (x-\mu_s)| > \beta(\hat{\alpha}_s)\right\}\right| \\ &\geqslant \left(1 - \frac{\hat{\alpha}_s}{50}|J|\right)|C^*| \geqslant 0.95\,|C^*|,\end{aligned}$$

where we used that and $|J| \leqslant 10/(9\hat{\alpha}_s)$. This implies that

$$|T_s| \geqslant \left|\tilde{T}_s\right| - w_{\text{low}}^2 |C^*| \geqslant (0.95 - w_{\text{low}}^2)\,|C^*| \geqslant 0.92\,|C^*| \geqslant 0.9\alpha n \geqslant 0.9\hat{\alpha}_s n,$$

i.e., $(\hat{\mu}_s, \hat{\alpha}_s)$ is not removed from M during s^{th} iteration.

The pair $(\hat{\mu}_s, \hat{\alpha}_s)$ could also be removed during later iterations, when we recalculate T_s by removing points along new directions. However, following a similar argument, we show that still, with high probability, $|T_s| \geqslant 0.9\hat{\alpha}_s n$. Assume that we are now in the k^{th} iteration of the outer cycle, where $k > s$. We define again $\tilde{T}_s$ and sets A, B:

$$\begin{aligned}\tilde{T}_s &:= \bigcap_{i\in J\setminus\{s\}} \left\{x \in C^*, \text{ s.t. } |v_{is}^\top (x-\hat{\mu}_s)| \leqslant \max(\beta(\hat{\alpha}_s), \beta(\hat{\alpha}_i))\right\}, \\ A &:= \bigcap_{i<s, i\in J} A_i, \quad \text{for} \quad A_i := \left\{x \in C^*, \text{ s.t. } |v_{is}^\top (x-\hat{\mu}_s)| \leqslant \beta(\hat{\boldsymbol{\alpha}}_{\boldsymbol{s}})\right\}, \\ B &:= \bigcap_{i>s, i\in J} B_i, \quad \text{for} \quad B_i := \left\{x \in C^*, \text{ s.t. } |v_{is}^\top (x-\hat{\mu}_s)| \leqslant \beta(\hat{\boldsymbol{\alpha}}_{\boldsymbol{i}})\right\},\end{aligned}$$

so that $\tilde{T}_s = A \cap B$ and again $|T_s| \geqslant \left|\tilde{T}_s\right| - w_{\text{low}}^2 |C^*|$. It is crucial that we have different right hand sides in the definitions of A_i and B_i (we wrote them in boldface to emphasize this).

Using a union bound again, we write

$$\left|\tilde{T}_s\right| \geqslant |C^*| - \sum_{i<s, i\in J} |C^* \setminus A_i| - \sum_{i>s, i\in J} |C^* \setminus B_i|.$$

Using eq. (B.4.3), with exponentially small failure probability, for all $i \in J$,

$$\begin{aligned}|C^* \setminus A_i| &\leqslant (\hat{\alpha}_s/50)\,|C^*| \quad (\text{for } i < s) \quad \text{and} \\ |C^* \setminus B_i| &\leqslant (\hat{\alpha}_i/50)\,|C^*| \quad (\text{for } i > s).\end{aligned}$$

Next, note that before the last element was added, we had that (i) $T_i \bigcap T_j = \emptyset$ for any $i \neq j \in J$ and (ii) $|T_i| \geqslant 0.9\hat{\alpha}_i n$ for any $i \in J$. This implies that

$\sum_{i\in J}\hat{\alpha}_i < 10/9 + \hat{\alpha}_{\text{last}} < 19/9$, where $\hat{\alpha}_{\text{last}}$ corresponds to the element which was added last (it might happen that after addition of the last element, we have $|T_i| < 0.9\hat{\alpha}_i n$ for several $i \in J$). Therefore, as before, we obtain that

$$\begin{aligned}\left|\tilde{T}_s\right| &\geqslant \left(1 - \sum_{i<s, i\in J}(\hat{\alpha}_s/50) - \sum_{i>s, i\in J}(\hat{\alpha}_i/50)\right)|C^*| \\ &\geqslant (1 - 10/(9\cdot 50) - 19/(9\cdot 50))\,|C^*| \geqslant 0.93\,|C^*|\,,\end{aligned}$$

therefore $|T_s| \geqslant \left|\tilde{T}_s\right| - w_{\text{low}}^2\,|C^*| \geqslant 0.92\,|C^*| \geqslant 0.9\hat{\alpha}_s n$ and $(\hat{\mu}_s, \hat{\alpha}_s)$ will not be removed from M.

We established that in a single run of the algorithm a good hypothesis is removed with exponentially small probability. The number of good hypotheses is bounded by $|M|$. Furthermore, the number of runs of the algorithm is also bounded by $|M|$, since whenever the algorithm is rerun a hypothesis is removed from M. Then, by a union bound, we can bound the probability that any good hypothesis is removed in any run of the algorithm by $|M|^2\, w_{\text{low}}^{O(1)}$. □

B.5 PROOF OF OUTER STAGE ALGORITHM GUARANTEES IN APPENDIX B.2.3

Recall that $\gamma = 4\psi_t(w_{\text{low}}^4)$ and $\gamma' = 160\psi_t(w_{\text{low}}/4) + 16f(w_{\text{low}}/4)$.

B.5.1 *Proof of Theorem B.2.6*

In what follows we condition on the event E that the events under which the conclusions in Theorems B.8.2 and B.8.3 hold and that $\mathcal{A}_{\text{sLD}}$ succeeds. This event holds with probability $1 - w_{\text{low}}^{O(1)}$ by Theorem 5.3.1, Theorem B.2.3 and union bound (also see Appendix B.7).

PROOF OF THEOREM B.2.6 (I) The list size bound follows from the standard results on $\mathcal{A}_{\text{sLD}}$ (see [31], Proposition B.1).

PROOF OF THEOREM B.2.6 (II) Guarantees of $\mathcal{A}_{\text{sLD}}$ imply that there exists $\mu_i \in M$ such that $\|\mu_i - \mu^*\| \leqslant \gamma'$. By Theorem B.8.3, a $(1 - w_{\text{low}}^2/2)$-fraction of the samples in C^* are γ-close to μ^* along each direction v_{ij} with $i \neq j \in [|M|]$. Then, the same $(1 - w_{\text{low}}^2/2)$-fraction of samples are $(\gamma + \gamma')$-close to μ_i along each direction v_{ij}, so they are included in $S_i^{(1)}$.

PROOF OF THEOREM B.2.6 (III) Suppose $|S_i^{(1)} \cap C^*| \geqslant w_{\text{low}}^4 |C^*|$. Previous point implies that, on event E, there exists $\mu_j \in M$ be such that $\|\mu_j - \mu^*\| \leqslant \gamma'$. Then at least an w_{low}^4-fraction of the samples in C^* are $(\gamma + \gamma')$-close to μ_i in direction $\mu_i - \mu_j$. By Theorem B.8.2, μ^* is also γ-close in direction $\mu_i - \mu_j$ to more than a $(1 - w_{\text{low}}^4)$-fraction of the samples in C^*, so it is γ-close to at least one sample in any w_{low}^4-fraction of samples in C^*. Therefore μ^* is also $(2\gamma + \gamma')$-close to μ_i in direction $\mu_i - \mu_j$. Then $\|\mu_i - \mu_j\| \leqslant 2\gamma + 2\gamma'$ and $\|\mu_i - \mu^*\| \leqslant 2\gamma + 3\gamma'$. Again, using Theorem B.8.3 we obtain that there exists a $(1 - w_{\text{low}}^2/2)$-fraction of the samples in C^*, which is included in $S_i^{(2)}$.

PROOF OF THEOREM B.2.6 (IV) Similarly, if $|S_i^{(2)} \cap C^*| \geqslant w_{\text{low}}^4 |C^*|$, then there exists $\mu_j \in M$, such that at least an w_{low}^4-fraction of the samples in C^* are $(3\gamma + 3\gamma')$-close to μ_i in direction $\mu_i - \mu_j$. By the same arguments as in previous paragraph, we obtain that $\|\mu_i - \mu_j\| \leqslant 4\gamma + 4\gamma'$ and $\|\mu_i - \mu^*\| \leqslant 4\gamma + 5\gamma'$.

Then any other true cluster with mean $(\mu^*)'$ and set of samples $(C^*)'$ satisfies $\|\mu^* - (\mu^*)'\| \geqslant 16\gamma + 16\gamma'$, so $\|\mu_i - (\mu^*)'\| \geqslant 12\gamma + 11\gamma'$. From guarantees of $\mathcal{A}_{\text{sLD}}$, there exists $\mu_j' \in M$ such that $\|\mu_j' - (\mu^*)'\| \leqslant \gamma'$. Then $\|\mu_i - \mu_j'\| \geqslant 12\gamma + 10\gamma'$. By Theorem B.8.2, more than an w_{low}^4-fraction of the samples from $(C^*)'$ are γ-close to $(\mu^*)'$ in direction $\mu_i - \mu_j'$, so also $(\gamma + \gamma')$-close to μ_j' in direction $\mu_i - \mu_j'$, so also $(11\gamma + 9\gamma')$-far from μ_i in direction $\mu_i - \mu_j'$. Then $S_i^{(2)}$ selects at most a w_{low}^4-fraction of the samples from $(C^*)'$. Overall, $S_i^{(2)}$ selects from all other true clusters at most $w_{\text{low}}^4 n$ samples.

PROOF OF THEOREM B.2.6 (V) Note that by the same argument, $\|\mu_i - \mu^*\| \leqslant 4\gamma + 5\gamma'$ and $\|\mu_{i'} - (\mu^*)'\| \leqslant 4\gamma + 5\gamma'$. However, $\|\mu^* - (\mu^*)'\| \geqslant 16\gamma + 16\gamma'$, so also $\|\mu_i - \mu_{i'}\| \geqslant 8\gamma + 6\gamma'$, so $S_i^{(2)}$ and $S_{i'}^{(2)}$ are disjoint by the condition that each selects only samples that are $(3\gamma + 3\gamma')$-close along direction $\mu_i - \mu_{i'}$ to the respective means μ_i and $\mu_{i'}$.

B.5.2 *Proof of Theorem B.2.7*

In the sequel, for any $i \in G$, let m_i be the index in R after initialization that satisfies Theorem B.2.6 (ii). We condition on the event E' that event E from

the proof of Theorem B.2.6 holds and that both $\left||C_i^*| - w_i n\right| \leqslant w_{\text{low}}^{10} n$ for all $i \in [k]$ and the number of adversarial points lies in the range $\varepsilon n \pm w_{\text{low}}^{10} n$. By Hoeffding's inequality and the union bound, the probability of E' is at least $1 - w_{\text{low}}^{O(1)}$.

PROOF OF THEOREM B.2.7 (1) Let $i \in G$, and consider the beginning of the iteration when μ_{g_i} is selected. Then, using that all previous iterations could have removed at most $O(w_{\text{low}}^3)|C_i^*|$ samples from C_i^*, we have that

$$|S_{m_i}^{(1)} \cap C_i^*| \geqslant (1 - w_{\text{low}}^2/2 - O(w_{\text{low}}^3))|C_i^*|.$$

Therefore at the iteration in which μ_{g_i} is selected, we still have $m_i \in R$. We now discuss two cases: First, consider the case that $|S_{m_i}^{(2)}| \leqslant 2|S_{m_i}^{(1)}|$. Then, because we selected $\mu_{g_i} \in M$ and not $\mu_{m_i} \in M$ it means that $|S_{g_i}^{(1)}| \geqslant |S_{m_i}^{(1)}| \geqslant (1 - w_{\text{low}}^2/2 - O(w_{\text{low}}^3))|C_i^*|$. Note also by Theorem B.2.6 (iv), the number of samples from other true clusters in $S_{g_i}^{(2)}$ is at most $w_{\text{low}}^4 n$. Then the number of adversarial samples in $S_{g_i}^{(2)}$ is at least

$$\begin{aligned}|S_{g_i}^{(2)}| - |C_i^*| - w_{\text{low}}^4 n &\geqslant |S_{g_i}^{(2)} \setminus S_{g_i}^{(1)}| - O(w_{\text{low}}^2)|C_i^*| - w_{\text{low}}^4 n \\ &\geqslant |S_{g_i}^{(2)} \setminus S_{g_i}^{(1)}| - O(w_{\text{low}}^2)|C_i^*|.\end{aligned}$$

Then, either $\left|S_{g_i}^{(2)} \setminus S_{g_i}^{(1)}\right| = O(w_{\text{low}}^2)|C_i^*|$ and $|U_i| \leqslant \left|S_{g_i}^{(2)} \setminus S_{g_i}^{(1)}\right| = O(w_{\text{low}}^2)|C_i^*|$, or $\left|S_{g_i}^{(2)} \setminus S_{g_i}^{(1)}\right| \gg w_{\text{low}}^2|C_i^*|$. In the latter case, even if $S_{g_i}^{(2)} \setminus S_{g_i}^{(1)}$ consists of adversarial examples only, then, since $|S_{g_i}^{(2)}| \leqslant 2|S_{g_i}^{(1)}|$, U_i contains at most double the number of adversarial examples in $S_{g_i}^{(1)}$, i.e. $|U_i| \leqslant 2V_i$ where V_i denotes the number of adversarial examples in $S_{g_i}^{(1)}$.

Now consider the case that $|S_{m_i}^{(2)}| > 2|S_{m_i}^{(1)}|$. By Theorem B.2.6 (iv), the number of samples from true clusters in $S_{m_i}^{(2)}$ is at most $|C_i^*| + w_{\text{low}}^4 n \leqslant 1.02|S_{m_i}^{(1)}|$, so the number W_i of adversarial samples in $S_{m_i}^{(2)}$ is at least $W_i \geqslant |S_{m_i}^{(2)}| - 1.02|S_{m_i}^{(1)}| \geqslant 0.98|S_{m_i}^{(1)}| \geqslant 0.96|C_i^*|$. Then, $|U_i| = \left|(C_i^* \cap S_{g_i}^{(2)}) \setminus S_{g_i}^{(1)}\right| \leqslant |C_i^*| \leqslant 2W_i$.

Finally note that by Theorem B.2.6 (v), the sets $S_{g_i}^{(2)}$ and $S_{m_i}^{(2)}$ are disjoint from any other sets $S_{g_j}^{(2)}$ and $S_{m_j}^{(2)}$ that correspond to another component C_j^*. Therefore, the number of adversarial examples in the $S_{m_i}^{(2)}$ in the second

case and $S_{g_i}^{(2)}$ in the first case is smaller than the total number of adversarial examples, i.e.

$$\sum_{\substack{i\in G\\ |S_{m_i}^{(2)}|\leqslant 2|S_{m_i}^{(1)}|}} V_i + \sum_{\substack{i\in G\\ |S_{m_i}^{(2)}|>2|S_{m_i}^{(1)}|}} W_i \leqslant (\varepsilon + w_{\mathrm{low}}^{10})n.$$

Therefore, we directly obtain

$$\begin{aligned}
|U| &\leqslant \sum_{i\in G} \left|(C_i^* \cap S_{g_i}^{(2)}) \setminus S_{g_i}^{(1)}\right| \\
&= \sum_{\substack{i\in G\\ |S_{m_i}^{(2)}|\leqslant 2|S_{m_i}^{(1)}|}} \left|(C_i^* \cap S_{g_i}^{(2)}) \setminus S_{g_i}^{(1)}\right| + \sum_{\substack{i\in G\\ |S_{m_i}^{(2)}|>2|S_{m_i}^{(1)}|}} \left|(C_i^* \cap S_{g_i}^{(2)}) \setminus S_{g_i}^{(1)}\right| \\
&\leqslant \sum_{\substack{i\in G\\ |S_{m_i}^{(2)}|\leqslant 2|S_{m_i}^{(1)}|}} 2V_i + \sum_{\substack{i\in G\\ |S_{m_i}^{(2)}|>2|S_{m_i}^{(1)}|}} 2W_i + O(w_{\mathrm{low}}^2)n \leqslant (2\varepsilon + O(w_{\mathrm{low}}^2))n.
\end{aligned}$$

PROOF OF THEOREM B.2.7 (II) Each iteration before g_i was selected, removed at most $w_{\mathrm{low}}^4|C_i^*|$ samples from C_i^*, so all previous iterations removed at most $O(w_{\mathrm{low}}^3)|C_i^*|$ samples from C_i^*. Then, by Lemma B.2.6 (iii), $S_{g_i}^{(2)}$ contains at least $(1 - w_{\mathrm{low}}^2/2 - O(w_{\mathrm{low}}^3))|C_i^*|$ samples from C_i^*. The statement follows then since on the event E', we have $w^*n - w_{\mathrm{low}}^{10}n \leqslant |C^*| \leqslant w^*n + w_{\mathrm{low}}^{10}n$.

PROOF OF THEOREM B.2.7 (III) Here, either for all $i \in [k]$, $|S_j^{(1)} \cap C_i^*| < w_{\mathrm{low}}^4|C_i^*|$ or $i \in G$ and the algorithm had already selected in a previous iteration $\mu_{g_i} \in M$ with $|S_{g_i}^{(1)} \cap C_i^*| \geqslant w_{\mathrm{low}}^4|C_i^*|$. Consider a first case, in which $|S_j^{(1)} \cap C_i^*| < w_{\mathrm{low}}^4|C_i^*|$ for all $i \in [k]$. Then the total number of samples from true clusters in $S_j^{(1)}$ is at most $w_{\mathrm{low}}^4 n$. Using that $|S_j^{(1)}| > 100 w_{\mathrm{low}}^4 n$, it follows that more than half of the samples in $S_j^{(1)}$ are adversarial.

The second case is that $|S_j^{(1)} \cap C_i^*| \geqslant w_{\mathrm{low}}^4|C_i^*|$ for some $i \in G$ for which in a previous iteration g_i we had that $|S_{g_i}^{(1)} \cap C_i^*| \geqslant w_{\mathrm{low}}^4|C_i^*|$. Note that at most $w_{\mathrm{low}}^2|C_i^*|/2$ of the samples in $S \cap C_i^*$ are not considered adversarial at this point (the ones that were outside $S_{g_i}^{(2)}$). Also, by Theorem B.2.6 (iv), $S_j^{(1)}$

contains at most $w_{\text{low}}^4 n$ samples from other true clusters. Therefore either more than half of the samples in $S_j^{(1)}$ are considered adversarial or

$$|S_j^{(1)}| \leqslant w_{\text{low}}^2 |C_i^*| + 2w_{\text{low}}^4 n \leqslant O(w_{\text{low}}^2) n.$$

PROOF OF THEOREM B.2.7 (IV) Suppose that when the algorithm reaches the else statement we have for some $i \in [k]$ that $i \in R$ and $|S_{m_i}^{(1)} \cap C_i^*| \geqslant 20 w_{\text{low}}^2 |C_i^*|$. We have that $|S_{m_i}^{(2)} \cap C_i^*|$ is at most $|S_{m_i}^{(1)} \cap C_i^*| + w_{\text{low}}^2 |C_i^*|/2$, where we use that by Theorem B.2.6 (ii), at most $w_{\text{low}}^2 |C_i^*|/2$ samples can fail to be captured by $S_{m_i}^{(1)}$. By Theorem B.2.6 (iv), furthermore, the number of samples from other true clusters in $S_{m_i}^{(2)}$ is at most $w_{\text{low}}^4 n$. Therefore, using that $|S_{m_i}^{(2)}| > 2|S_{m_i}^{(1)}|$, the number of adversarial samples in $S_{m_i}^{(2)}$ is at least

$$|S_{m_i}^{(2)}| - |S_{m_i}^{(1)} \cap C_i^*| - w_{\text{low}}^2 |C_i^*|/2 - w_{\text{low}}^4 n \geqslant 0.45 |S_{m_i}^{(2)}| - w_{\text{low}}^4 n \geqslant 0.44 |S_{m_i}^{(2)}|,$$

where in the last inequality we used that $|S_{m_i}^{(2)}| > 100 w_{\text{low}}^4 n$. Let V be the union, over all $i \in [k]$, of all sets $S_{m_i}^{(2)}$ such that $i \in R$ and $|S_{m_i}^{(1)} \cap C_i^*| \geqslant 20 w_{\text{low}}^2 |C_i^*|$. Theorem B.2.6 (v) gives that all such sets $S_{m_i}^{(2)}$ are disjoint. Therefore at least a 0.44-fraction of the samples in V are adversarial.

Consider now for some $i \in [k]$ how many samples from $S \cap C_i^*$ can be outside V when the algorithm reaches the else statement. By Theorem B.2.6 (ii), $S_{m_i}^{(1)}$ can fail to capture at most $w_{\text{low}}^2 |C_i^*|/2$ samples from C_i^*, and we have no guarantee that these samples are in V. Consider now the samples in $S_{m_i}^{(1)} \cap C_i^*$. If $i \in R$, we may miss up to $20 w_{\text{low}}^2 |C_i^*|$ of these samples if $|S_{m_i}^{(1)} \cap C_i^*| < 20 w_{\text{low}}^2 |C_i^*|$, because in this case we do not include $S_{m_i}^{(2)}$ in V. On the other hand, if $i \notin R$, there are at most $100 w_{\text{low}}^4 n$ samples in $S_{m_i}^{(1)} \cap C_i^*$. Then the total number of samples from $S \cap C_i^*$ outside V is at most $w_{\text{low}}^2 |C_i^*|/2 + 20 w_{\text{low}}^2 |C_i^*| + 100 w_{\text{low}}^4 n$. Summed across all $i \in [k]$, this makes up at most $21 w_{\text{low}}^2 n$ samples.

Overall, the number of adversarial samples in S when the algorithm reaches the else statement is at least

$$\begin{aligned} 0.44|V| + (|S| - |V| - 21 w_{\text{low}}^2 n) &= |S| - 0.56|V| - 21 w_{\text{low}}^2 n \\ &\geqslant 0.44|S| - 21 w_{\text{low}}^2 n \geqslant 0.4|S| \end{aligned}$$

where in the last inequality we also used that $|S| \geqslant 0.1 w_{\text{low}} n$.

B.6 PROOF OF THEOREM 5.3.5

We now prove lower bounds for the case of Gaussian distributions and distributions with t-th sub-Gaussian moments.

B.6.1 *Case b): For the Gaussian inliers*

We first focus on the case when $D_i(\mu_i) = \mathcal{N}(\mu_i, I)$.

The proof goes through an efficient reduction from the problem considered by Theorem B.6.1 to the problem solved by algorithm $\mathcal{A}$.

Proposition B.6.1 ([33], Proposition 5.11)**.** *Let $\mathcal{D}$ be the class of identity covariance Gaussians on $\mathbb{R}^d$ and let $0 < \alpha \leqslant 1/2$. Then any list-decoding algorithm that learns the mean of an element of $\mathcal{D}$ with failure probability at most $1/2$, given access to $(1-\alpha)$-additively corrupted samples, must either have error bound $\beta = \Omega(\sqrt{\log 1/\alpha})$ or return $\min(2^{\Omega(d)}, (1/\alpha)^{w(1)})$ many hypotheses.*

First, we describe the means of the components in the input distribution to algorithm $\mathcal{A}$. Let $\bar{\mu}_1, \ldots, \bar{\mu}_{k-1} \in \mathbb{R}^d$ be any set of $k-1$ points with pairwise separation larger than $2C\sqrt{\log 1/w_{\text{low}}}$. Then let $\mu_k = (\bar{\mu}, 0) \in \mathbb{R}^{d+1}$ and $\mu_i = (\bar{\mu}_i, 2C\sqrt{\log 1/w_{\text{low}}} + 1) \in \mathbb{R}^{d+1}$ for all $i \in [k-1]$. Then $\mu_1, \ldots, \mu_k$ also have pairwise separation larger than $2C\sqrt{\log 1/w_{\text{low}}}$.

Then, given n points $y_1, \ldots, y_n \in \mathbb{R}^d$ as in the input to the problem in Theorem B.6.1 (i.e. $(1-\alpha)$-additively corrupted samples), we generate n points that we give as input to algorithm $\mathcal{A}$ as follows: let $S = \{1, \ldots, n\}$, and then for each of the n points, draw $i \sim \text{Unif}\{1, \ldots, k\}$ and generate the point as follows:

1. if $i \in [k-1]$, sample the point from $N(\mu_i, I_{d+1})$,
2. if $i = k$, sample $j \sim S$ uniformly at random, remove j from S, sample $g \sim N(0,1)$, and let the point be $(y_j, g) \in \mathbb{R}^{d+1}$.

We note that this construction simulates an input sampled i.i.d. according to the mixture $\frac{1}{k}N(\mu_1, I_{d+1}) + \ldots + \frac{1}{k}N(\mu_{k-1}, I_{d+1}) + \frac{\alpha}{k}N(\mu_k, I_{d+1}) + \frac{1-\alpha}{k}Q'$ for some Q'. Then with success probability at least $1/2$ running $\mathcal{A}$ on this input with $w_{\text{low}} = \frac{\alpha}{k}$ returns a list L such that there exists $\hat{\mu} \in L$ with $\|\hat{\mu} - \mu_k\| \leqslant \beta_k$. Note that this implies that $\|(\hat{\mu})_{1:d} - \bar{\mu}\| \leqslant \beta_k$. Finally, we create a pruned list L' as follows: initialize $L' = L$ and then for each $i \in [k-1]$ remove all $\hat{\mu} \in L'$ such that $\|\hat{\mu} - \mu_i\| \leqslant C\sqrt{\log 1/w_{\text{low}}}$. Then we return L' as the output for the original problem in Theorem B.6.1.

Let us analyze now this output. The separation between the means ensures that any hypothesis $\hat{\mu} \in L$ that is $C\sqrt{\log 1/w_{\text{low}}}$-close to μ_k is not removed in the pruning. Therefore L' continues to contain a hypothesis $\hat{\mu}$ such that $\|(\hat{\mu})_{1:d} - \bar{\mu}\| \leqslant \beta_k$. Then, if $\beta_k \neq \Omega(\sqrt{\log 1/\alpha})$ and $|L'| < \min\{2^{\Omega(d)}, ((w_k+\varepsilon)/w_k)^{\omega(1)}\}$, this reduction violates the lower bound of Theorem B.6.1. Therefore we must have either $\beta_k = \Omega(\sqrt{\log 1/\alpha})$ or $|L'| \geqslant \min\{2^{\Omega(d)}, (1/\tilde{w}_k)^{\omega(1)}\}$.

Finally, we show that these lower bounds on β_k and $|L'|$ imply the desired lower bound for $\mathcal{A}$. Consider first the case: $\beta_k = \Omega(\sqrt{\log 1/\alpha})$. Note that in the input to algorithm $\mathcal{A}$ we have $\tilde{w}_k = \alpha$. Therefore $\beta_k = \Omega(\sqrt{\log 1/\alpha})$ corresponds to the desired lower bound in the lemma statement. Consider second the case: $|L'| \geqslant \min\{2^{\Omega(d)}, (1/\tilde{w}_k)^{\omega(1)}\}$. We note that, for each $i \in [k-1]$, the original list L must contain some $\hat{\mu} \in L$ such that $\|\hat{\mu} - \mu_i\| \leqslant C\sqrt{\log 1/w_{\text{low}}}$. Furthermore, because the means μ_i have pairwise separation larger than $2C\sqrt{\log 1/w_{\text{low}}}$, the original list L must contain at least $k-1$ means of this kind. However, all of these means are removed in the pruning procedure, so $|L| \geqslant k-1+|L'|$, so $|L| \geqslant k-1+\min\{2^{\Omega(d)}, (1/\tilde{w}_k)^{\omega(1)}\}$. This matches the desired lower bound in the lemma statement. (The choice to make the hidden mean the k-th mean was without loss of generality, as the distribution is invariant to permutations of the components.)

B.6.2 *Case a): For distributions with t-th sub-Gaussian moments*

The proof for the case when $D_i(\mu_i)$ has sub-Gaussian t-th central moments employs the same reduction scheme, but reduces from Theorem B.6.2.

Proposition B.6.2 ([33], Proposition 5.12)**.** *Let $\mathcal{D}$ be the class of distributions on $\mathbb{R}^d$ with bounded t-th central moments for some positive even integer t, and let $0 < \alpha < 2^{-t-1}$. Then any list-decoding algorithm that learns the mean of an element of $\mathcal{D}$ with failure probability at most $1/2$, given access to $(1-\alpha)$-additively corrupted samples, must either have error bound $\beta = \Omega(\alpha^{-1/t})$ or return a list of at least d hypotheses.*

Furthermore, in [31], formal evidence of computational hardness was obtained (see their Theorem 5.7, which gives a lower bound in the statistical query model introduced by [76]) that suggests obtaining error $\Omega_t((1/\tilde{w}_s)^{1/t})$ requires running time at least $d^{\Omega(t)}$. This was proved for Gaussian inliers and the running time matches ours up to a constant in the exponent.

B.7 STABILITY OF LIST-DECODING ALGORITHMS

In this section we discuss two of the existing list-decodable mean estimation algorithms for identity-covariance Gaussian distributions and show that they also work when a w_{low}^2-fraction of the inliers is adversarially removed.

First, we consider the algorithm in Theorem 3.1 in [31]. A central object in their analysis is an "α-good multiset", which is a multiset of samples such that all are within distance $O(\sqrt{d})$ of each other and at least an α-fraction of them come from a $(1-\Omega(\alpha))$-fraction of an i.i.d. set of samples from a Gaussian distribution $N(\mu, I_d)$. Then their algorithm essentially works as long as the input contains an α-good multiset. For our case, after the removal of a w_{low}^2-fraction of inliers, the input essentially continues to contain a $(1-w_{\text{low}}^2)\alpha$-good multiset, so the algorithm continues to work in our corruption model.

Second, we consider the algorithm in Theorem 6.12 in [33]. The main distributional requirement of their algorithm is that $\mathbb{E}_{x,y\sim S^*}[p^2(x-y)] \leqslant 2\mathbb{E}_{g,h\sim N(0,I_d)}[p^2(g-h)]$ for all degree-$(t/2)$ polynomials p, where S^* is the set of inliers. Concentration arguments give with high probability that $\mathbb{E}_{x,y\sim C^*}[p^2(x-y)] \leqslant 1.5\mathbb{E}_{g,h\sim N(0,I_d)}[p^2(g-h)]$. Furthermore, the distribution over $x, y \sim S^*$ can be seen as a $(1-w_{\text{low}}^2)^2$-fraction of the distribution over $x, y \sim C^*$. Then Theorem B.7.1, which follows by standard probability calculations, also gives that any event under the former distribution can be bounded in terms of the second distribution:

Fact B.7.1. For any event A,

$$\Pr_{x,y\sim S^*}(A) \leqslant \Pr_{x,y\sim C^*}(A)/(1-w_{\text{low}}^2)^2, \tag{B.7.1}$$

where probabilities are taken over a uniform sample from S^* and C^* respectively.

Overall we obtain

$$\mathbb{E}_{x,y\sim S^*}[p^2(x-y)] \leqslant 1.5/(1-w_{\text{low}}^2)^2\mathbb{E}_{g,h\sim N(0,I_d)}[p^2(g-h)],$$

so for w_{low} small enough we have $\mathbb{E}_{x,y\sim S^*}[p^2(x-y)] \leqslant 2\mathbb{E}_{g,h\sim N(0,I_d)}[p^2(g-h)]$ and their algorithm continues to work in our corruption model.

B.8 CONCENTRATION BOUNDS

In this section we prove some concentration bounds essential to our analysis.

Lemma B.8.1. *Let D be a d-dimensional distribution with mean $\mu^* \in \mathbb{R}^d$ and sub-Gaussian t-th central moments with parameter 1. Fix a unit vector $v \in \mathbb{R}^d$. Then*

$$\Pr_{x\sim D}[|\langle x-\mu^*, v\rangle| \leqslant R] \geqslant 1 - \left(\frac{\sqrt{t}}{R}\right)^t.$$

Proof. We have that

$$\Pr_{x\sim D}[|\langle x-\mu^*, v\rangle| > R] \leqslant \frac{\mathbb{E}_{x\sim D}\langle x-\mu^*, v\rangle^t}{R^t} \leqslant \frac{(t-1)!!}{R^t} \leqslant \left(\frac{\sqrt{t}}{R}\right)^t,$$

where we used that $(t-1)!! \leqslant t^{t/2} = \sqrt{t}^t$. □

Lemma B.8.2. *Let D be a d-dimensional distribution with mean $\mu^* \in \mathbb{R}^d$ and sub-Gaussian t-th central moments with parameter 1. Let C^* be a set of i.i.d. samples drawn from D. Fix a unit vector $v \in \mathbb{R}^d$. Then with probability at least $1 - \exp\left(-2|C^*|\left(\frac{\sqrt{t}}{R}\right)^{2t}\right)$,*

$$|\{x \in C^*, s.t.\ |\langle x-\mu^*, v\rangle| \leqslant R\}| \geqslant \left(1 - 2\left(\frac{\sqrt{t}}{R}\right)^t\right)|C^*|.$$

Proof. The result follows by Theorem B.8.1 and a Binomial tail bound. □

Lemma B.8.3. *Let D be a d-dimensional distribution with mean $\mu^* \in \mathbb{R}^d$ and sub-Gaussian t-th central moments with parameter 1. Let C^* be a set of i.i.d. samples drawn from D. Fix m unit vectors $v_1, \ldots, v_m \in \mathbb{R}^d$. Then with probability at least $1 - \exp\left(-2|S^*|m^2\left(\frac{\sqrt{t}}{R}\right)^{2t}\right)$,*

$$\left|\bigcap_{i\in[m]} \{x \in S^*, s.t.\ |\langle x-\mu^*, v_i\rangle| \leqslant R\}\right| \geqslant \left(1 - 2m\left(\frac{\sqrt{t}}{R}\right)^t\right)|S^*|.$$

Proof. By Theorem B.8.1 and a union bound over the m directions, we get

$$\Pr_{x\sim D}[|\langle x-\mu^*, v_i\rangle| \leqslant R, \forall i \in [m]] \geqslant 1 - m\left(\frac{\sqrt{t}}{R}\right)^t.$$

Then the result follows by a Binomial tail bound. □

B.9 EXPERIMENTAL DETAILS

ADVERSARIAL LINE AND ADVERSARIAL CLUSTERS The following figure illustrates the adversarial distributions used in Figure 5.2 and further in this section.

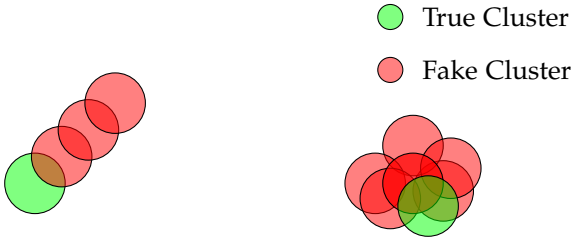

FIGURE B.1: Two variants of adversarial distribution: adversarial line (left) and adversarial clusters (right).

DATA DISTRIBUTION We consider a mixture of $k = 7$ well-separated ($\|\mu_i - \mu_j\| \geqslant 40$) $d = 100$ dimensional inlier clusters whose subgroup sizes range from 0.3 to 0.02. The experiments are conducted once using a Gaussian distribution and once using a heavy-tailed t-distribution with five degrees of freedom for both inlier and adversarial clusters. In Figure B.3 the latter suggests that our algorithm works comparatively well even for mixture distributions which do not fulfill our assumptions. We set $w_{\text{low}} = 0.02$ and $\varepsilon = 0.12$ so that it is larger than the smallest clusters but smaller than the largest ones and set the total number of data points to 10000. The Gaussian noise model simply computes the empirical mean and covariance matrix of the clean data and samples 1200 noisy samples from a Gaussian distribution with this mean and covariance. The adversarial cluster model and the adversarial model are as depicted in Figure B.1.

ATTACK DISTRIBUTIONS We consider three distinct adversarial models (see Figure B.1 for reference).

1. *Adversarial clusters*: After sampling the inlier cluster means, we choose the cluster with the smallest weight. Let μ_s denote its mean. Then, we sample a random direction v_c with $\|v_c\| = 10$. After that, we sample three directions v_1, v_2 and v_3 with $\|v_i\| = 10$. Then we put three additional (outlier) clusters with means at $\mu_s + v_c + v_i$. This roughly corresponds to the right picture in Figure B.1. The samples for each

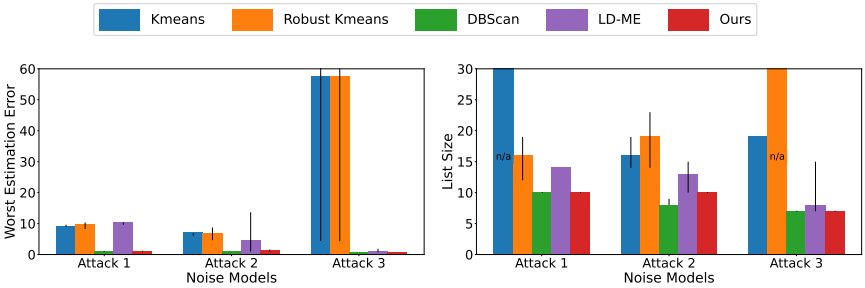

FIGURE B.2: Comparison of five algorithms with three adversarial noise models. On the left we show worst estimation error of algorithms with constrained list size and on the right the smallest list size with constrained error guarantee. We plot the median of the metrics with the error bars showing 25th and 75th percentile. We observe that our method consistently outperforms prior works in terms of list size and worst estimation error, with the exception of DBSCAN, which performs at a similiar level.

adversarial cluster are drawn from a distribution that matches the covariance of the inlier clusters, with the sample size being twice as large as of the affected inlier cluster.

2. *Adversarial line*: After sampling the inlier cluster means, we again choose the cluster with the smallest weight. Let μ_s denote its mean. Then, we sample a random direction v_c with $\|v_c\| = 10$. We put three additional (outlier) clusters with means at $\mu_s + v_c$, $\mu_s + 2v_c$ and $\mu_s + 3v_c$, which form a line as shown in Figure B.1. The samples are drawn similarly to the adversarial clusters, with the difference that the covariance is scaled by a factor of 5 in the direction of the line.

3. *Gaussian adversary*: Here we simply introduce noise matching the empirical mean and covariance of all inlier data (i.e., as if all inlier clusters are generated from the same Gaussian distribution).

Note that in the first and second attack, the adversary creates clusters that do not respect the separation assumption of the true inlier clusters: either adversarial clusters are placed around the smallest inlier cluster (Adversarial Cluster), or the adversarial clusters form a line, pointing out in some fixed direction (Adversarial Line).

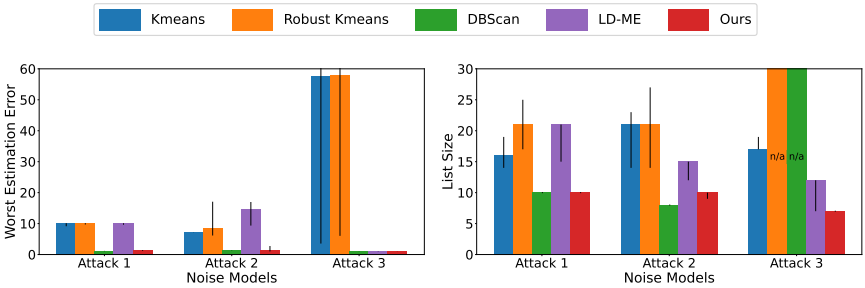

FIGURE B.3: Worst estimation error and list size comparison for the case where inlier distributions are heavy-tailed. We can observe numerical stability of our approach.

IMPLEMENTATION DETAILS We implement the list-decodable mean estimation base learner in our InnerStage algorithm (Algorithm 6) based on [36]. It leverages an iterative multi-filtering strategy and one-dimensional projections. In particular, we use the simplified gaussian version of the algorithm. It is designed for distributions sampled from a Gaussian but also shows promising results for the experiments involving a heavy-tailed t-distribution as depicted in Figure B.3. The robust mean estimator used to improve the mean hypotheses for large clusters is omitted in our implementation.

HYPER-PARAMETER SEARCH AND EXPERIMENT EXECUTION The hyper-parameters of our algorithm are tuned beforehand based on the experimental setup. For the comparative algorithms, hyper-parameter searches are conducted within each experiment after initial tuning. For our algorithm, key parameters include the pruning radius γ used in the OuterStage routine (Algorithm 10) and β used in the InnerStage (Algorithm 8). In addition, parameters for the LD-ME base learner, such as the cluster concentration threshold, also require careful selection, resulting in a total of 7 parameters. The tuning for these was performed using a grid search comparing about 250 different configurations. Similarly, we independently tune the vanilla LD-ME algorithm, which we run with w_{low} as weight parameter. For DBSCAN, we optimize the list size and error metrics by searching over a range of 100 values for ε, which controls the maximum distance between samples considered in the same neighbourhood. The minimum samples threshold, which validates the density based clusters, is pretuned beforehand and adjusted based on w_{low}. For k-means and its robust version, utilizing a median-of-means weighting scheme, we explore 21 values for k, including

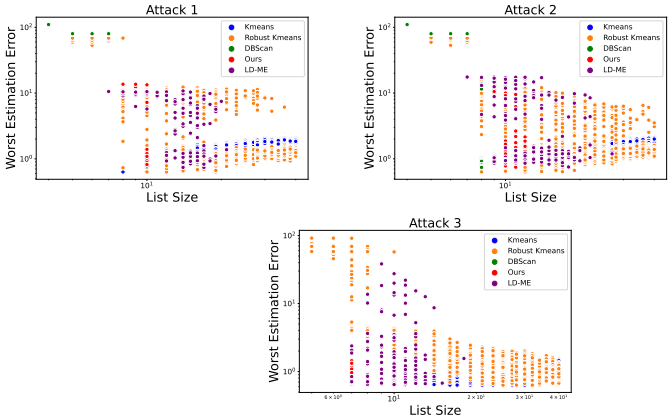

FIGURE B.4: Scatter plot of all results for one iteration of the experiment using three adversarial noise models.

the true number of clusters. Each parameter setting is executed 100 times to account for stochastic variations in the algorithmic procedures, such as k-means initialization. The list size and worst estimation error for each list of clusters obtained is visualized exemplarily for one iteration of the experiment in Figure B.4. The plot provides insight into how the different algorithms perform and vary with different list sizes.

EVALUATION DETAILS Note that we have two sources of randomness: the data is random and also the algorithms themselves are random (except DBSCAN). For a clear comparison, we sample and fix one dataset for each attack model. we plot the performance of 100 runs of each algorithm for each parameter setting, each time recording the returned list size together with the worst estimation error $\max_{i\in[k]} \min_{\hat{\mu}\in L} \|\mu_i - \hat{\mu}\|$. Then we either (i) report the worst estimation error for all runs with constrained list size (we pick the list size most frequently returned by our algorithm, specifically 7 or 10 in our experiments) (see Figure B.2, left), or (ii) report the smallest list size required to achieve the same or smaller worst estimation error (we pick the 75th quantile of errors of our algorithm for a threshold) (see Figure B.2, right). Under size constraint (i), the bar plots correspond to the median over the runs, with error bars indicating the 25th and 75th quantiles. Under error constraint (ii), the bar plots represent the minimum list size for which the median over the runs falls below the threshold, while the error bars show the minimum list size for which the 25th and 75th quantiles meet the

constraint. Note that 'n/a' indicates that, within the scope of our parameter search, no list size achieves an error below the specified constraint.

In Figure B.3 we study the numerical stability of our approach. In particular, whether the performance degrades when inlier distribution does not satisfy required assumptions. We observe that if one uses our meta-algorithm with base learner designed for Gaussian inliers, we still obtain stable results even in the case of heavy-tailed inlier distribution.

B.9.1 *Variation of w_{low}*

To study the effect of varying w_{low} input on the performance of our approach and LD-ME, we introduce a new noise model. As illustrated in Figure B.5, we consider a mixture of $k = 3$ well-separated clusters: one small cluster with a weight of 0.045 and two large clusters, each with a weight of 0.2. We place two adversarial clusters (see paragraph on attack distributions for details): one near the small cluster and another near one of the large clusters. Furthermore, uniform noise is introduced, spanning the range of the data generated by the inlier and its nearby outlier cluster and accounting for 10% of the data in this region. Overall, $\varepsilon = 0.56$ and we draw 22650 samples from this mixture distribution.

For both algorithms we run 100 seeds for each w_{low} ranging from 0.02 to 0.2, which corresponds to the weight of the largest inlier cluster. In Figure B.6, we plot the median estimation error with error bars showing the 25th and 75th quantiles for the small cluster (top left) and the large cluster near the outlier cluster (top right). As expected from our theoretical results, we observe that our algorithm performs roughly constant in estimating the mean of the large cluster, regardless of the initial w_{low}. Meanwhile, the estimation error of LD-ME increases as w_{low} decreases further below the true cluster weight. Furthermore, the plots show that our approach does consistently outperform LD-ME in terms of both worst estimation error and list size. Figure B.7 also compares the performance of the clustering algorithms in this experimental setup with results similar to the ones obtained in the previous experimental settings.

B.9.2 *Computational resources*

Our implementation of the algorithm and experiments leverages multi-threading. It utilizes CPU resources of an internal cluster with 128 cores, which results in a execution time of about 5 minutes for a single run of

the experiment for one noise model with 10000 samples. We remark that classic approaches like k-means and DBSCAN perform fast and the most time-consuming part is the execution of the LD-ME base learner. Given our experimental setup with three noise models, it takes about 15 minutes to reproduce all our results for one data distribution.

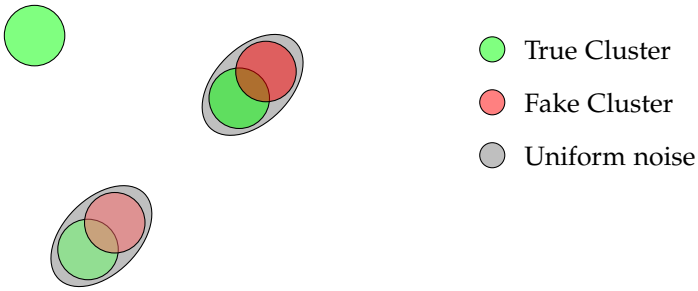

FIGURE B.5: Setup for w_{low} variation experiment with clusters contaminated by an adversarial cluster and uniform noise. Lower color intensities indicate smaller cluster weights.

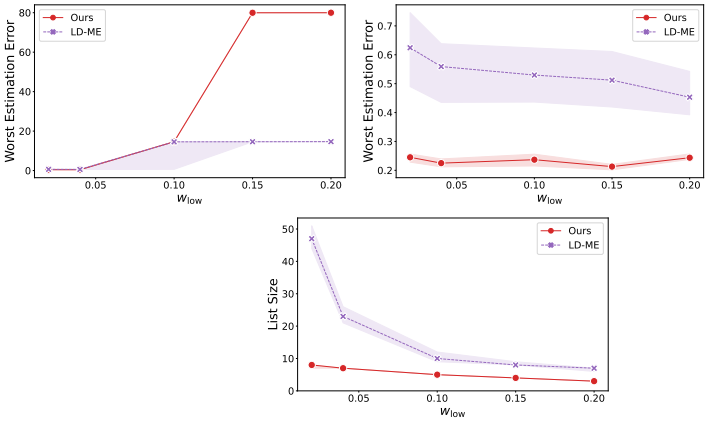

FIGURE B.6: Comparison of list size and estimation error for small and large inlier clusters for varying w_{low} inputs. The experimental setup is illustrated in Figure B.5. The plot on the top left shows the estimation error for the small cluster and the plot on the top right shows the error for the large cluster. We plot the median values with error bars indicating 25th and 75th quantiles. As w_{low} decreases, our algorithm maintains a roughly constant estimation error for the large cluster, while the error for LD-ME increases.

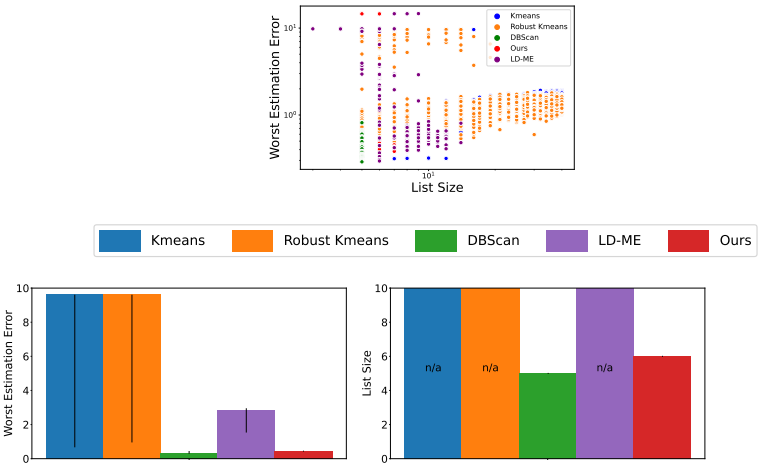

FIGURE B.7: Worst estimation error and list size comparison for the setup used in the w_{low} variation experiment.

BIBLIOGRAPHY

1. Paschos, V. T. A survey of approximately optimal solutions to some covering and packing problems. *ACM Computing Surveys (CSUR)* (1997).
2. Iliopoulos, F. & Zadik, I. *Group testing and local search: is there a computational-statistical gap?* in *Conference on Learning Theory* (2021).
3. Lovász, L. On the ratio of optimal integral and fractional covers. *Discrete mathematics* (1975).
4. Slavík, P. *A tight analysis of the greedy algorithm for set cover* in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing* (1996).
5. Lovász, L. On the Shannon capacity of a graph. *IEEE Transactions on Information theory* **25**, 1 (1979).
6. Juhász, F. The asymptotic behaviour of Lovász'theta function for random graphs. *Combinatorica* **2**, 153 (1982).
7. Arora, S. & Bhaskara, A. A note on the Lovász theta number of random graphs.
8. Coja-Oghlan, A. The Lovász number of random graphs. *Combinatorics, Probability and Computing* **14**, 439 (2005).
9. Green*, B. Counting sets with small sumset, and the clique number of random Cayley graphs. *Combinatorica* **25**, 307 (2005).
10. Green, B. & Morris, R. Counting sets with small sumset and applications. *Combinatorica* **36**, 129 (2016).
11. Magsino, M., Mixon, D. G. & Parshall, H. *Linear programming bounds for cliques in Paley graphs* in *Wavelets and Sparsity XVIII* **11138** (2019), 440.
12. Bandeira, A. S., Lewis, M. E. & Mixon, D. G. Discrete uncertainty principles and sparse signal processing. *Journal of Fourier Analysis and Applications* **24**, 935 (2018).
13. Demanet, L. & Hand, P. Scaling law for recovering the sparsest element in a subspace. *Information and Inference: A Journal of the IMA* **3**, 295 (2014).

14. Candes, E. J. & Tao, T. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE transactions on information theory* **52**, 5406 (2006).

15. Haviv, I. & Regev, O. *The restricted isometry property of subsampled Fourier matrices* in *Geometric Aspects of Functional Analysis: Israel Seminar (GAFA) 2014–2016* (2017), 163.

16. Dwork, C., McSherry, F., Nissim, K. & Smith, A. *Calibrating noise to sensitivity in private data analysis* in *Theory of Cryptography (TCC)* (2006).

17. Valiant, L. G. A theory of the learnable. *Communications of the ACM* (1984).

18. Bassily, R., Smith, A. & Thakurta, A. *Private empirical risk minimization: Efficient algorithms and tight error bounds* in *Symposium on foundations of computer science (FOCS)* (2014).

19. Chaudhuri, K., Monteleoni, C. & Sarwate, A. D. Differentially private empirical risk minimization. *Journal of Machine Learning Research (JMLR)* (2011).

20. Kasiviswanathan, S. P., Lee, H. K., Nissim, K., Raskhodnikova, S. & Smith, A. What can we learn privately? *SIAM Journal on Computing* (2011).

21. Beimel, A., Nissim, K. & Stemmer, U. *Characterizing the sample complexity of private learners* in *Conference on Innovations in Theoretical Computer Science (ITCS)* (2013).

22. Feldman, V. & Xiao, D. *Sample complexity bounds on differentially private learning via communication complexity* in *Conference on Learning Theory (COLT)* (2014).

23. Beimel, A., Brenner, H., Kasiviswanathan, S. P. & Nissim, K. Bounds on the sample complexity for private learning and private data release. *Machine learning* (2014).

24. Alon, N., Bun, M., Livni, R., Malliaris, M. & Moran, S. Private and online learnability are equivalent. *ACM Journal of the ACM (JACM)* (2022).

25. Littlestone, N. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine learning* (1988).

26. Alon, N., Livni, R., Malliaris, M. & Moran, S. *Private PAC learning implies finite Littlestone dimension* in *Symposium on Theory of Computing (STOC)* (2019).

27. Golowich, N. & Livni, R. Littlestone classes are privately online learnable. *Conference on Neural Information Processing Systems (NeurIPS)* (2021).

28. Sanyal, A. & Ramponi, G. *Open Problem: Do you pay for Privacy in Online learning?* in *Conference on Learning Theory (COLT)* (2022).

29. Dwork, C., Rothblum, G. N. & Vadhan, S. *Boosting and differential privacy* in *Symposium on Foundations of Computer Science (FOCS)* (2010).

30. Chan, T.-H. H., Shi, E. & Song, D. Private and continual release of statistics. *Transactions on Information and System Security (TISSEC)* (2011).

31. Diakonikolas, I., Kane, D. M. & Stewart, A. *List-decodable robust mean estimation and learning mixtures of spherical gaussians* in *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing* (2018), 1047.

32. Bakshi, A., Diakonikolas, I., Jia, H., Kane, D. M., Kothari, P. K. & Vempala, S. S. *Robustly learning mixtures of k arbitrary Gaussians* in *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing* (2022), 1234.

33. Diakonikolas, I. & Kane, D. M. *Algorithmic high-dimensional robust statistics* (Cambridge University Press, 2023).

34. Regev, O. & Vijayaraghavan, A. *On learning mixtures of well-separated Gaussians* in *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)* (2017), 85.

35. Ivkov, M. & Kothari, P. K. *List-decodable covariance estimation* in *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing* (2022), 1276.

36. Diakonikolas, I., Kane, D. M., Kongsgaard, D., Li, J. & Tian, K. *Clustering mixture models in almost-linear time via list-decodable mean estimation* in *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing* (2022), 1262.

37. Johnson, D. S. *Approximation algorithms for combinatorial problems* in *Proceedings of the fifth annual ACM symposium on Theory of computing* (1973).

38. Feige, U. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)* **45**, 634 (1998).

39. Fernández Anta, A., Mosteiro, M. A. & Ramón Muñoz, J. Unbounded contention resolution in multiple-access channels. *Algorithmica* (2013).

40. Erlich, Y., Gilbert, A., Ngo, H., Rudra, A., Thierry-Mieg, N., Wootters, M., Zielinski, D. & Zuk, O. Biological screens from linear codes: theory and tools. *BioRxiv* (2015).

41. Du, D., Hwang, F. K. & Hwang, F. *Combinatorial group testing and its applications* (World Scientific, 2000).

42. Mézard, M. & Tarzia, M. Statistical mechanics of the hitting set problem. *Phys. Rev. E* (2007).

43. Telelis, O. A. & Zissimopoulos, V. Absolute $O(\log m)$ error in approximating random set covering: an average case analysis. *Information Processing Letters* (2005).

44. Borst, S., Dadush, D., Huiberts, S. & Tiwari, S. On the integrality gap of binary integer programs with gaussian data. *Mathematical Programming* (2022).

45. Borst, S., Dadush, D. & Mikulincer, D. *Integrality Gaps for Random Integer Programs via Discrepancy* in *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA* (2023).

46. Bazaraa, M. S., Jarvis, J. J. & Sherali, H. D. *Linear programming and network flows* (John Wiley & Sons, 2011).

47. Blasiok, J., Lopatto, P., Luh, K., Marcinek, J. & Rao, S. *An improved lower bound for sparse reconstruction from subsampled hadamard matrices* in *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)* (2019), 1564.

48. Cohen, S. D. Clique numbers of Paley graphs. *Quaestiones Mathematicae* **11**, 225 (1988).

49. Baker, R., Ebert, G., Hemmeter, J. & Woldar, A. Maximal cliques in the Paley graph of square order. *Journal of statistical planning and inference* **56**, 33 (1996).

50. Hanson, B. & Petridis, G. Refined estimates concerning sumsets contained in the roots of unity. *Proceedings of the London Mathematical Society* **122**, 353 (2021).

51. Kunisky, D. Spectral pseudorandomness and the road to improved clique number bounds for Paley graphs. *Experimental Mathematics*, 1 (2024).

52. Kobzar, V. A. & Mody, K. *Revisiting block-diagonal sdp relaxations for the clique number of the paley graphs* in *2023 International Conference on Sampling Theory and Applications (SampTA)* (2023), 1.

53. Wang, Y., Shen, Y. & Kobzar, V. A. Lower Bounds on Block-Diagonal SDP Relaxations for the Clique Number of the Paley Graphs and Their Localizations.

54. Cahill, J. & Mixon, D. G. Robust width: A characterization of uniformly stable and robust compressed sensing. *Excursions in Harmonic Analysis, Volume 6: In Honor of John Benedetto's 80th Birthday*, 343 (2021).

55. Hanneke, S., Livni, R. & Moran, S. *Online learning with simple predictors and a combinatorial characterization of minimax in 0/1 games* in *Conference on Learning Theory (COLT)* (2021).

56. Beimel, A., Nissim, K. & Stemmer, U. *Private learning and sanitization: Pure vs. approximate differential privacy* in *International Workshop on Approximation Algorithms for Combinatorial Optimization* (2013).

57. Dwork, C. & Feldman, V. *Privacy-preserving prediction* in *Conference On Learning Theory (COLT)* (2018).

58. Naor, M., Nissim, K., Stemmer, U. & Yan, C. *Private Everlasting Prediction* in *Conference on Neural Information Processing Systems (NeurIPS)* (2023).

59. Kearns, M., Pai, M. M., Rogers, R., Roth, A. & Ullman, J. Robust mediators in large games. *arXiv:1512.02698* (2015).

60. Kaplan, H., Mansour, Y., Moran, S., Nissim, K. & Stemmer, U. *Black-Box Differential Privacy for Interactive ML* in *Conference on Neural Information Processing Systems (NeurIPS)* (2023).

61. Cesa-Bianchi, N. & Lugosi, G. *Prediction, learning, and games* (Cambridge university press, 2006).

62. Blumer, A., Ehrenfeucht, A., Haussler, D. & Warmuth, M. K. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM (JACM)* (1989).

63. Ben-David, S., Pál, D. & Shalev-Shwartz, S. *Agnostic Online Learning.* in *Conference On Learning Theory (COLT)* (2009).

64. Ghazi, B., Golowich, N., Kumar, R. & Manurangsi, P. *Sample-efficient proper PAC learning with approximate differential privacy* in *Symposium on Theory of Computing (STOC)* (2021).

65. Diakonikolas, I., Gouleakis, T. & Tzamos, C. Distribution-independent pac learning of halfspaces with massart noise. *Conference on Neural Information Processing Systems (NeurIPS)* (2019).

66. Cohen, E., Lyu, X., Nelson, J., Sarlós, T. & Stemmer, U. Lower Bounds for Differential Privacy Under Continual Observation and Online Threshold Queries. *arXiv preprint arXiv:2403.00028* (2024).

67. Jain, P., Raskhodnikova, S., Sivakumar, S. & Smith, A. *The price of differential privacy under continual observation* in *International Conference on Machine Learning (ICML)* (2023).

68. Dwork, C., Naor, M., Pitassi, T. & Rothblum, G. N. *Differential privacy under continual observation* in *Symposium on Theory of computing (STOC)* (2010).

69. Dwork, C., Roth, A., *et al.* The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science* **9**, 211 (2014).

70. Bickel, D. R. Robust cluster analysis of microarray gene expression data with the number of clusters determined biologically. *Bioinformatics* **19**, 818 (2003).

71. Feigelson, E. D. & Babu, G. J. Statistical methods for Astronomy. *arXiv preprint arXiv:1205.2064* (2012).

72. Kothari, P. K., Steinhardt, J. & Steurer, D. *Robust moment estimation and improved clustering via sum of squares* in *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing* (2018), 1035.

73. Diakonikolas, I., Kamath, G., Kane, D., Li, J., Moitra, A. & Stewart, A. Robust estimators in high-dimensions without the computational intractability. *SIAM Journal on Computing* **48**, 742 (2019).

74. Hopkins, S. B. & Li, J. *Mixture models, robustness, and sum of squares proofs* in *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing* (2018), 1021.

75. Liu, A. & Li, J. *Clustering mixtures with almost optimal separation in polynomial time* in *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing* (2022), 1248.

76. Kearns, M. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM (JACM)* **45**, 983 (1998).

77. Elias, P. *List decoding for noisy channels* tech. rep. (Research Laboratory of Electronics, Massachusetts Institute of Technology, 1957).

78. Charikar, M., Steinhardt, J. & Valiant, G. *Learning from untrusted data* in *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing* (2017), 47.

79. Diakonikolas, I., Kane, D. & Kongsgaard, D. List-decodable mean estimation via iterative multi-filtering. *Advances in Neural Information Processing Systems* **33**, 9312 (2020).

80. Diakonikolas, I., Kane, D., Kongsgaard, D., Li, J. & Tian, K. List-decodable mean estimation in nearly-PCA time. *Advances in Neural Information Processing Systems* **34**, 10195 (2021).

81. Cherapanamjeri, Y., Mohanty, S. & Yau, M. *List decodable mean estimation in nearly linear time* in *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)* (2020), 141.

82. Diakonikolas, I., Kane, D., Karmalkar, S., Pensia, A. & Pittas, T. List-decodable sparse mean estimation via difference-of-pairs filtering. *Advances in Neural Information Processing Systems* **35**, 13947 (2022).

83. Zeng, S. & Shen, J. List-decodable sparse mean estimation. *Advances in Neural Information Processing Systems* **35**, 24031 (2022).

84. Karmalkar, S., Klivans, A. & Kothari, P. List-decodable linear regression. *Advances in neural information processing systems* **32** (2019).

85. Raghavendra, P. & Yau, M. *List decodable learning via sum of squares* in *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms* (2020), 161.

86. Diakonikolas, I., Kane, D., Pensia, A., Pittas, T. & Stewart, A. Statistical query lower bounds for list-decodable linear regression. *Advances in Neural Information Processing Systems* **34**, 3191 (2021).

87. Bakshi, A. & Kothari, P. K. *List-decodable subspace recovery: Dimension independent error in polynomial time* in *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)* (2021), 1279.

88. Raghavendra, P. & Yau, M. *List decodable subspace recovery* in *Conference on Learning Theory* (2020), 3206.

89. Balcan, M.-F., Blum, A. & Vempala, S. *A discriminative framework for clustering via similarity functions* in *Proceedings of the fortieth annual ACM symposium on Theory of computing* (2008), 671.

90. Meister, M. & Valiant, G. *A data prism: Semi-verified learning in the small-alpha regime* in *Conference On Learning Theory* (2018), 1530.

91. Lai, K. A., Rao, A. B. & Vempala, S. *Agnostic estimation of mean and covariance* in *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)* (2016), 665.

92. Diakonikolas, I., Kamath, G., Kane, D. M., Li, J., Moitra, A. & Stewart, A. *Being robust (in high dimensions) can be practical* in *International Conference on Machine Learning* (2017), 999.

93. Bakshi, A., Diakonikolas, I., Hopkins, S. B., Kane, D., Karmalkar, S. & Kothari, P. K. *Outlier-robust clustering of Gaussians and other non-spherical mixtures* in *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)* (2020), 149.

94. Liu, A. & Moitra, A. *Settling the robust learnability of mixtures of Gaussians* in *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing* (2021), 518.

95. García-Escudero, L. A., Gordaliza, A., Matrán, C. & Mayo-Iscar, A. A review of robust clustering methods. *Advances in Data Analysis and Classification* **4**, 89 (2010).

96. Ester, M., Kriegel, H.-P., Sander, J., Xu, X., *et al. A density-based algorithm for discovering clusters in large spatial databases with noise* in *kdd* (1996), 226.

97. Lloyd, S. Least squares quantization in PCM. *IEEE transactions on information theory*, 129 (1982).

98. Brownlees, C., Joly, E. & Lugosi, G. Empirical risk minimization for heavy-tailed losses (2015).

99. Hoorfar, A. & Hassani, M. Inequalities on the Lambert W function and hyperpower function. *J. Inequal. Pure and Appl. Math* (2008).

100. Van Handel, R. *Probability in high dimension* Lecture notes. 2014.

101. Feller, W. & Morse, P. M. *An introduction to probability theory and its applications* (American Institute of Physics, 1958).

102. Diakonikolas, I., Kamath, G., Kane, D. M., Li, J., Moitra, A. & Stewart, A. *Robustly Learning a Gaussian: Getting Optimal Error, Efficiently* in *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018* (ed Czumaj, A.) (SIAM, 2018), 2683.

PUBLICATIONS

Articles in peer-reviewed journals:

1. Einstein, A. Über die von der molekularkinetischen Theorie der Wärme geforderte Bewegung von in ruhenden Flüssigkeiten suspendierten Teilchen. *Annalen der Physik* **322**, 549 (1905).
2. Einstein, A. Zur Elektrodynamik bewegter Körper. *Annalen der Physik* **322**, 891 (1905).

Conference contributions:

3. Einstein, A. *Implications of a fixed vacuum speed of light* in *Relativity05* Oct. 2–6, 1905 (1st Conference on Special Relativity, Zurich, Switzerland).