

# Streaming Algorithms for High-Dimensional Robust Statistics

Ilias Diakonikolas<sup>†</sup>  
University of Wisconsin-Madison  
ilias@cs.wisc.edu

Daniel M. Kane<sup>‡</sup>  
University of California, San Diego  
dakane@cs.ucsd.edu

Ankit Pensia<sup>§</sup>  
University of Wisconsin-Madison  
ankitp@cs.wisc.edu

Thanasis Pittas<sup>¶</sup>  
University of Wisconsin-Madison  
pittas@wisc.edu

April 27, 2022

## Abstract

We study high-dimensional robust statistics tasks in the streaming model. A recent line of work obtained computationally efficient algorithms for a range of high-dimensional robust estimation tasks. Unfortunately, all previous algorithms require storing the entire dataset, incurring memory at least quadratic in the dimension. In this work, we develop the first efficient streaming algorithms for high-dimensional robust statistics with near-optimal memory requirements (up to logarithmic factors). Our main result is for the task of high-dimensional robust mean estimation in (a strengthening of) Huber’s contamination model. We give an efficient single-pass streaming algorithm for this task with near-optimal error guarantees and space complexity nearly-linear in the dimension. As a corollary, we obtain streaming algorithms with near-optimal space complexity for several more complex tasks, including robust covariance estimation, robust regression, and more generally robust stochastic optimization.

---

\*Authors are in alphabetical order. Part of this work was done while a subset of the authors were visiting the Simons Institute for the Theory of Computing.

<sup>†</sup>Supported by NSF Medium Award CCF-2107079, NSF Award CCF-1652862 (CAREER), a Sloan Research Fellowship, and a DARPA Learning with Less Labels (LwLL) grant.

<sup>‡</sup>Supported by NSF Medium Award CCF-2107547, NSF Award CCF-1553288 (CAREER), and a Sloan Research Fellowship.

<sup>§</sup>Supported by NSF grants NSF Award CCF-1652862 (CAREER), DMS-1749857, and CCF-1841190.

<sup>¶</sup>Supported by NSF Award CCF-1652862 (CAREER) and NSF Award DMS-2023239 (TRIPODS).

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Our Results . . . . .	1
1.2	Overview of Techniques . . . . .	3
1.3	Prior and Related Work . . . . .	4
1.4	Organization . . . . .	5
<b>2</b>	<b>Preliminaries</b>	<b>5</b>
2.1	Notation and Basic Facts . . . . .	5
2.2	Stability Condition and Its Properties . . . . .	7
<b>3</b>	<b>Filtering Algorithm with Small Number of Iterations</b>	<b>8</b>
3.1	Setup and Algorithm Description . . . . .	8
3.2	Establishing the Deterministic Conditions . . . . .	10
3.3	Downweighting Filter . . . . .	11
3.4	Correctness of Algorithm 1: Proof of Theorem 3.2 . . . . .	12
3.4.1	Invariants of Algorithm 1 . . . . .	13
3.4.2	Reducing the Potential Function . . . . .	13
<b>4</b>	<b>Efficient Streaming Algorithm for Robust Mean Estimation</b>	<b>16</b>
4.1	Setup and Algorithm Description . . . . .	17
4.2	Correctness of Algorithm 3 . . . . .	20
4.2.1	Proof of Lemma 4.7 via a Cover Argument . . . . .	21
4.3	Establishing Condition 4.5 . . . . .	22
4.3.1	Item 1 . . . . .	23
4.3.2	Items 2a to 2c . . . . .	23
4.3.3	Item 3 . . . . .	27
<b>5</b>	<b>Applications: Beyond Robust Mean Estimation</b>	<b>28</b>
5.1	Robust Covariance Estimation . . . . .	28
5.2	Stochastic Convex Optimization . . . . .	29
5.2.1	Linear Regression . . . . .	30
5.2.2	Logistic Regression . . . . .	31
5.3	Byzantine Adversary and Second-order Optimal Point . . . . .	32
<b>6</b>	<b>Discussion</b>	<b>33</b>
<b>A</b>	<b>Omitted Proofs from Section 2: Technical Details Regarding Stability</b>	<b>38</b>
<b>B</b>	<b>Omitted Proofs from Section 3</b>	<b>40</b>
B.1	Johnson-Lindenstrauss Sketch . . . . .	40
B.2	Proof of Lemma 3.6 . . . . .	41
B.3	Proof of Lemma 3.7 . . . . .	42
B.4	Proof of Claim 3.11 . . . . .	44
B.5	Omitted Proofs from Section 3.4 . . . . .	44

<b>C</b>	<b>Omitted Proofs from Section 4</b>	<b>46</b>
C.1	Omitted Proofs from Section 4.2 . . . . .	46
C.2	Omitted Proofs from Section 4.2.1 . . . . .	46
C.3	Omitted Proofs from Section 4.3 . . . . .	51
C.4	Omitted Proofs from Section 4.3.1 . . . . .	51
<b>D</b>	<b>Adaptive Choice of Upper Bound on Covariance</b>	<b>52</b>
<b>E</b>	<b>Omitted Details from Section 5</b>	<b>54</b>
E.1	Proof Sketch of Theorem 5.3 . . . . .	54
E.2	Omitted Proofs from Section 5.2 . . . . .	55
<b>F</b>	<b>Bit Complexity of Algorithm 3</b>	<b>56</b>

# 1 Introduction

This work studies high-dimensional learning in the presence of a constant fraction of arbitrary outliers. Outlier-robust learning in high dimensions is motivated by pressing machine learning (ML) applications, including ML security [BNJT10, BNL12, SKL17, TLM18, DKK<sup>+</sup>19a] and exploratory analysis of datasets with natural outliers [RPW<sup>+</sup>02, PLJD10, LAT<sup>+</sup>08]. This field has its roots in robust statistics, a branch of statistics initiated in the 60s with the pioneering works of Tukey and Huber [Tuk60, Hub64]. Early work developed minimax optimal estimators for various robust estimation tasks, albeit with runtimes exponential in the dimension. A recent line of work in computer science, starting with [DKK<sup>+</sup>16, LRV16], developed polynomial time robust estimators for a range of high-dimensional statistical tasks. Algorithmic high-dimensional robust statistics is by now a relatively mature field, see, e.g., [DK19, DKK<sup>+</sup>21a] for surveys.

This recent progress notwithstanding, even for the basic task of mean estimation, previous robust estimators require the entire dataset in main memory. This space requirement can be a major bottleneck in large-scale applications, where an algorithm has access to a very large stream of data. Indeed, practical machine learning methods are typically simple iterative algorithms that make a single pass over the data and require a small amount of storage — with stochastic gradient descent being the prototypical example [Bot10, BCN18]. Concretely, in prior applications of robust statistics in data analysis [DKK<sup>+</sup>17] and data poisoning defenses [DKK<sup>+</sup>19a], the storage requirements of the underlying algorithms were observed to significantly hinder scalability. This discussion motivates the following natural question:

*Can we develop efficient robust estimators in the streaming model  
with (near-) optimal space complexity?*

We emphasize that this broad question is meaningful and interesting even ignoring computational considerations. While any method requires space complexity  $\Omega(d)$ , where  $d$  is the dimension of the problem (to store a single sample), it is not obvious that a matching upper bound exists. We note that it is relatively simple to design  $O(d)$ -memory streaming algorithms with sample complexity exponential in  $d$ . But it is by no means clear whether there exists an estimator with near-linear space requirements and  $\text{poly}(d)$  sample complexity (independent of its runtime).

## 1.1 Our Results

In this work, we initiate a systematic investigation of high-dimensional robust statistics in the streaming model. We start by focusing on the most basic task — that of robust mean estimation. Our main result is the first space-efficient streaming algorithm for robust mean estimation under natural distributional assumptions. Our computationally efficient algorithm makes a single pass over the data, uses near-optimal space, and matches the error guarantees of previous polynomial-time algorithms for the problem.

Given this result, we leverage the fact that several robust statistics tasks can be reduced to robust mean estimation to obtain near-optimal space, single-pass streaming algorithms for more complex statistical tasks.

To formally state our contributions, we require some basic definitions. We start with the standard streaming model.

**Definition 1.1** (Single-Pass Streaming Model). *Let  $S$  be a fixed set. In the one-pass streaming model, the elements of  $S$  are revealed one at a time to the algorithm, and the algorithm is allowed a single pass over these points.*

Our robust estimators work in the following contamination model, where the adversary can corrupt the true distribution in total variation distance (for distributions  $P$  and  $Q$ , we use  $d_{\text{TV}}(P, Q)$  to denote their total variation distance).

**Definition 1.2** (TV-contamination). *Given a parameter  $\epsilon < 1/2$  and a distribution class  $\mathcal{D}$ , the adversary specifies a distribution  $D'$  such that there exists  $D \in \mathcal{D}$  with  $d_{\text{TV}}(D, D') \leq \epsilon$ . Then the algorithm draws i.i.d. samples from  $D'$ . We say that the distribution  $D'$  is an  $\epsilon$ -corrupted version of the distribution  $D$  in total variation distance.*

The distribution  $D'$  in [Definition 1.2](#) can be adversarially selected (and can even depend on our learning algorithm). Since Huber's contamination model [[Hub64](#)] only allows additive errors, TV-contamination is a stronger model.

## Streaming Algorithm for Robust Mean Estimation

The main result of this paper is the following (see [Theorem 4.2](#) for a more general statement):

**Theorem 1.3** (Streaming Robust Mean Estimation). *Let  $\mathcal{D}$  be a distribution family on  $\mathbb{R}^d$  and  $0 < \epsilon < \epsilon_0$  for a sufficiently small constant  $\epsilon_0 > 0$ . Let  $P$  be an  $\epsilon$ -corrupted version of  $D$  in total variation distance for some  $D \in \mathcal{D}$  with unknown mean  $\mu_D$ . There is a single-pass streaming algorithm that, given  $\epsilon$  and  $\mathcal{D}$ , reads a stream of  $n$  i.i.d. samples from  $P$ , runs in sample near-linear time, uses memory  $d \text{polylog}(d/\epsilon)$ , and outputs an estimate  $\hat{\mu}$  that, with probability at least  $9/10$ , satisfies the following:*

1. *If  $\mathcal{D}$  is the family of distributions with identity-bounded covariance, then  $n = \tilde{O}(d^2/\epsilon)$  and  $\|\hat{\mu} - \mu_D\|_2 = O(\sqrt{\epsilon})$ .*
2. *If  $\mathcal{D}$  is the family of identity-covariance subgaussian distributions, then  $n = \tilde{O}(d^2/\epsilon^2)$  and  $\|\hat{\mu} - \mu_D\|_2 = O(\epsilon\sqrt{\log(1/\epsilon)})$ .*

We note that the above error guarantees are information-theoretically optimal, even in absence of resource constraints. While prior work had obtained efficient robust mean estimators matching these error guarantees [[DKK<sup>+</sup>16](#), [DKK<sup>+</sup>17](#), [SCV18](#)], all previous algorithms with dimension-independent error incurred space complexity  $\Omega(d^2)$ .

## Beyond Robust Mean Estimation

Using the algorithm of [Theorem 1.3](#) as a black-box, we obtain the first efficient single-pass streaming algorithms with near-optimal space complexity for a range of more complex statistical tasks. These contributions are presented in detail in [Section 5](#). Here we highlight some of these results.

Our first application is a streaming algorithm for robust covariance estimation.

**Theorem 1.4** (Robust Gaussian Covariance Estimation). *Let  $Q$  be a distribution on  $\mathbb{R}^d$  with  $d_{\text{TV}}(Q, \mathcal{N}(0, \Sigma)) \leq \epsilon$  and assume  $\frac{1}{\kappa}\mathbf{I}_d \preceq \Sigma \preceq \mathbf{I}_d$ . There is a single-pass streaming algorithm that uses  $n = (d^4/\epsilon^2)\text{polylog}(d, \kappa, 1/\epsilon)$  samples from  $Q$ , runs in time  $nd^2\text{polylog}(d, \kappa, 1/\epsilon)$ , uses memory  $d^2\text{polylog}(d, \kappa, 1/\epsilon)$ , and outputs a matrix  $\hat{\Sigma}$  such that  $\|\Sigma^{-1/2}\hat{\Sigma}\Sigma^{-1/2} - \mathbf{I}_d\|_F = O(\epsilon \log(1/\epsilon))$  with probability at least  $9/10$ .*

See [Theorem 5.3](#) for a more detailed statement.

Our second application is for the general problem of robust stochastic optimization. Here we state two concrete results for robust linear and logistic regression (see [Theorem 5.9](#) and [Theorem 5.12](#) for more detailed statements). Both of these statements are special cases of a streaming algorithm for robust stochastic convex optimization (see [Corollary 5.6](#)).

**Theorem 1.5** (Streaming Robust Linear Regression). *Let  $D$  be the distribution of  $(X, Y)$  defined by  $Y = X^T \theta^* + Z$ , where  $X \sim \mathcal{N}(0, \mathbf{I}_d)$ ,  $Z \sim \mathcal{N}(0, 1)$  independent of  $X$ , and  $\|\theta^*\|_2 \leq r$ . Let  $P$  be an  $\epsilon$ -corruption of  $D$  in total variation distance. There is a single-pass streaming algorithm that uses  $n = (d^2/\epsilon) \text{polylog}(d(1+r)/\epsilon)$  samples from  $P$ , runs in time  $nd \text{polylog}(d(1+r)/\epsilon)$ , uses memory  $d \text{polylog}(dr/\epsilon)$ , and outputs an estimate  $\hat{\theta} \in \mathbb{R}^d$  such  $\|\hat{\theta} - \theta^*\|_2 = O(\sqrt{\epsilon})$  with high probability.*

**Theorem 1.6** (Streaming Robust Logistic Regression). *Consider the following model: Let  $(X, Y) \sim D$ , where  $X \sim \mathcal{N}(0, \mathbf{I}_d)$ ,  $Y | X \sim \text{Bern}(p)$ , for  $p = 1/(1 + e^{-X^T \theta^*})$ , and  $\|\theta^*\|_2 = O(1)$ . Let  $P$  be an  $\epsilon$ -corruption of  $D$  in total variation distance. There is a single-pass streaming algorithm that uses  $n = (d^2/\epsilon) \text{polylog}(d/\epsilon)$  samples from  $P$ , runs in time  $nd \text{polylog}(d/\epsilon)$ , uses memory  $d \text{polylog}(d/\epsilon)$ , and outputs an estimate  $\hat{\theta} \in \mathbb{R}^d$  such  $\|\hat{\theta} - \theta^*\|_2 = O(\sqrt{\epsilon})$  with high probability.*

Finally, in [Section 5](#), we include an additional application to distributed non-convex optimization in the streaming setting.

**Remark 1.7** (Bit complexity). For simplicity of presentation, in the main body of the paper, we consider the model of computation where the algorithms can store and manipulate real numbers exactly. We show in [Appendix F](#) that our algorithms can tolerate errors due to finite precision. In particular, all our algorithms (including [Algorithm 3](#)) can be implemented in the word RAM model with  $d \text{polylog}(d/\epsilon)$  bits.

## 1.2 Overview of Techniques

In this section, we provide a brief overview of our approach to establish [Theorem 1.3](#). We start by recalling how robust mean estimation algorithms typically work without space constraints. A standard tool in the literature is the filtering technique of [[DKK<sup>+</sup>16](#), [DKK<sup>+</sup>17](#), [DK19](#)]. The idea of the filtering method is the following: Given a set  $S$  of corrupted samples, by analyzing spectral properties of the covariance of  $S$ , we can either certify that the sample mean of  $S$  is close to the true mean of the distribution, or can construct a filter. The filter is a method for selecting some elements of  $S$  to remove, with the guarantee that it will remove more outliers than inliers. If we can efficiently construct a filter, our algorithm can then remove the selected samples from  $S$ , obtaining a cleaner dataset and repeat the process. Eventually, this procedure must terminate, giving an accurate estimate of the true mean.

We proceed to explain how to implement the filtering method in a streaming model. We start with the easier case where the dataset is stored in read-only-memory, or more generally in a *multi-pass* streaming setting. At each round of the algorithm, one has a subset  $S'$  of the original dataset  $S$  that needs to be maintained (in particular, the set of samples that has survived the filters applied thus far). To do this naïvely would require  $n = |S|$  many bits of memory, which is too much for us. A more inventive strategy would be the following: instead of storing these subsets  $S'$  explicitly, store them implicitly by instead storing enough information to reconstruct the filters used to obtain  $S'$ . This seems like a productive idea, as most filters are relatively simple. For example, a commonly used filter is to remove all points  $x \in S$  for which  $v^T x > t$ , for some vector  $v$  and scalar  $t$ . One could store enough information to apply this filter by merely storing  $(v, t)$ , which would take  $O(d)$  bits of information. Unfortunately, most filtering algorithms may require  $\Omega(d)$  many iterations before attaining their final answer. Consequently, the sets  $S'$  one needs to store are not just the result of applying a single filter, but instead the result of iteratively applying  $\Omega(d)$  of them. In order to store all of these extra filters, one would need  $\Omega(d^2)$  bits. (For the sake of this intuitive description, we focused on “hard-thresholding” filters. Our algorithm will actually use a soft-thresholding filter, assigning weights to each point.)

To circumvent this first obstacle, one requires as a starting point a filtering algorithm that is guaranteed to terminate after a small (namely, at most poly-logarithmic) number of iterations. Recent work [DHL19, DKK<sup>+</sup>21b] has obtained such algorithms. Here we generalize and simplify the filtering method of [DKK<sup>+</sup>21b]. This allows us to obtain an algorithm with space complexity  $d \text{polylog}(d/\epsilon)$  that works in the *multi-pass streaming model*, where  $\text{polylog}(d/\epsilon)$  passes over the same dataset are allowed.

To obtain a *single-pass* streaming algorithm, new ideas are required. In the single-pass setting, we cannot implicitly store a subset of the full dataset  $S$ ; once we access some points from  $S$ , we will never be able to see them again. To deal with this issue, we will need to slightly alter our way of thinking about the algorithm. Instead of being given a set  $S$  of samples, an  $\epsilon$ -fraction of which have been corrupted, we instead adopt the view of having sample access to a distribution  $P$ , which is  $\epsilon$ -close in total variation distance to the inlier distribution  $G$ . Given this point of view, instead of a filter defining a procedure for removing samples from  $S$  and outputting a subset  $S'$ , we think of it as a rejection sampling procedure that replaces  $P$  with a cleaner distribution  $P'$ .

This shift in perspective comes with new technical challenges. In particular, when constructing the next round of filters, we will need to compute quantities pertaining to the current distribution  $P$  of the data points. In the setting of the multiple-pass model, this imposed no problem; these quantities could be calculated exactly. This is no longer possible when we merely have sample access to  $P$ . The best one can hope for is to approximate these quantities to sufficient precision for the rest of our analysis to carry over. However, the natural estimators for some required quantities (e.g., powers of the covariance matrix) would need to access the data multiple times. Circumventing this issue requires non-trivial technical work. Roughly speaking, instead of iterating over the same dataset to approximate the desired quantities, we show that it suffices to iterate over statistically identical datasets.

### 1.3 Prior and Related Work

Since the dissemination of [DKK<sup>+</sup>16, LRV16], there has been an explosion of research in algorithmic aspects of robust statistics. We now have efficient robust estimators for a range of more complex problems, including covariance estimation [DKK<sup>+</sup>16, CDGW19a], sparse estimation tasks [BDLS17, DKK<sup>+</sup>19b, CDK<sup>+</sup>21], learning graphical models [CDKS18, DKSS21], linear regression [KKM18, DKS19, PJJ20], stochastic optimization [PSBR20, DKK<sup>+</sup>19a], and robust clustering/learning various mixture models [HL18, KSS18, DKS18, DKK<sup>+</sup>20, DKK<sup>+</sup>21b, BDH<sup>+</sup>20, LM20, BDJ<sup>+</sup>20]. The reader is referred to [DK19] for a detailed overview. We reiterate that all previously developed algorithms work in the batch setting, i.e., require the entire dataset in memory.

For the problem of robust mean estimation, [DHL19, DKK<sup>+</sup>21b] gave filtering-based algorithms with a poly-logarithmic number of iterations. The former algorithm relies on the matrix multiplicative weights framework, while the latter is based on first principles. Our starting point in Section 3 can be viewed as a generalization and further simplification of the ideas in [DKK<sup>+</sup>21b]. Specifically, our algorithm works under the stability condition (Definition 2.8), which broadly generalizes the bounded covariance assumption used in [DKK<sup>+</sup>21b].

In the context of robust supervised learning (including, e.g., our robust linear regression application), low-space streaming algorithms are known in weaker contamination models that only allow *label* corruptions, see, e.g., [PF20, SWS20, DKTZ20]. We emphasize that the contamination model of Definition 1.2 is significantly more challenging, and no low-space streaming algorithms were previously known in this model.

Finally, we note that recent work [TPBR21] studies streaming algorithms for heavy-tailed stochastic optimization. While the goal of developing low-space streaming algorithms is qualitatively

similar to the goal of our work, the algorithmic results in [TPBR21] have no implications in the corrupted setting studied in this work.

## 1.4 Organization

The structure of this paper is as follows: In Section 2, we record the notation and technical background that will be used throughout the paper. In Section 3, we design a filter-based algorithm for robust mean estimation under the stability condition with a poly-logarithmic number of iterations. In Section 4, we build on the algorithm from Section 3, to obtain our single-pass streaming algorithm for robust mean estimation under the stability condition. Finally, in Section 5, we obtain our streaming algorithms for more complex robust estimation tasks. To facilitate the flow of the presentation, some proofs of intermediate lemmas are deferred to the Appendix.

## 2 Preliminaries

### 2.1 Notation and Basic Facts

**Basic Notation** We use  $\mathbb{Z}_+$  to denote the set of positive integers. For  $n \in \mathbb{Z}_+$ , we denote  $[n] := \{1, \dots, n\}$  and use  $\mathcal{S}^{d-1}$  for the  $d$ -dimensional unit sphere. For a vector  $v$ , we let  $\|v\|_2$  denote its  $\ell_2$ -norm. We use boldface letters for matrices. We use  $\mathbf{I}_d$  to denote the  $d \times d$  identity matrix. For a matrix  $\mathbf{A}$ , we use  $\|\mathbf{A}\|_F$  and  $\|\mathbf{A}\|_2$  to denote the Frobenius and spectral norms respectively. For  $\mathbf{A} \in \mathbb{R}^{m \times n}$ , we use  $\mathbf{A}^\flat$  to denote the  $nm$ -dimensional vector obtained by concatenating the rows of  $\mathbf{A}$ . We say that a symmetric  $d \times d$  matrix  $\mathbf{A}$  is PSD (positive semidefinite), and write  $\mathbf{A} \succeq 0$ , if for all vectors  $x \in \mathbb{R}^d$  we have that  $x^T \mathbf{A} x \geq 0$ . We denote  $\lambda_{\max}(\mathbf{A}) := \max_{u \in \mathcal{S}^{d-1}} x^T \mathbf{A} x$ . We write  $\mathbf{A} \preceq \mathbf{B}$  when  $\mathbf{B} - \mathbf{A}$  is PSD. For a matrix  $\mathbf{A} \in \mathbb{R}^{d \times d}$ ,  $\text{tr}(\mathbf{A})$  denotes the trace of the matrix  $\mathbf{A}$ . We use  $\otimes$  to denote the Kronecker product. For the sake of conciseness, we sometimes use  $x = a \pm b$  as a shorthand for  $a - b \leq x \leq a + b$ . We use  $a \lesssim b$ , to denote that there exists an absolute universal constant  $C > 0$  (independent of the variables or parameters on which  $a$  and  $b$  depend) such that  $a \leq Cb$ . Similarly, we use the notation  $a \gtrsim b$  to denote that  $b \lesssim a$ . We use  $c, c', C, C'$  to denote absolute constants that may change from line to line, whereas we use constants  $C_1, C_2, C_3, \dots$  to denote fixed absolute constants that are important for our algorithms. We use  $\tilde{O}(\cdot)$  to ignore poly-logarithmic factors in all variables appearing inside the parentheses. For the sake of simplicity, we sometimes omit rounding non-integer quantities to integer ones. For example, we treat logarithmic factors as integers when they appear in the sample complexity or number of iterations of an algorithm. We use  $\text{poly}(\cdot)$  to indicate a quantity that is polynomial in its arguments. Similarly,  $\text{polylog}(\cdot)$  denotes a quantity that is polynomial in the logarithm of its arguments.

**Probability Notation** For a random variable  $X$ , we use  $\mathbf{E}[X]$  for its expectation. For a set  $S$ , we use  $\mathcal{U}(S)$  to denote the uniform distribution on  $S$ . We use  $\mathcal{N}(\mu, \Sigma)$  to denote the Gaussian distribution with mean  $\mu$  and covariance matrix  $\Sigma$ . For a distribution  $D$  on  $\mathbb{R}^d$ , we denote  $\mu_D = \mathbf{E}_{X \sim D}[X]$  and  $\Sigma_D = \mathbf{E}_{X \sim D}[(X - \mu_D)(X - \mu_D)^T]$ . Moreover, given a weight function  $w : \mathbb{R}^d \rightarrow [0, 1]$ , we define the re-weighted distribution  $D_w$  to be  $D_w(x) := D(x)w(x) / \int_{\mathbb{R}^d} w(x)D(x)dx$ . We use  $\mu_{w,D} = \mathbf{E}_{X \sim D_w}[X]$  for its mean and  $\bar{\Sigma}_{w,D}^\mu = \mathbf{E}_{X \sim D_w}[(X - \mu)(X - \mu)^T]$  for the second moment that is centered with respect to  $\mu$  (we will often drop  $\mu$  from the notation when it is clear from the context). We use  $\mathbb{1}\{x \in E\}$  to denote the indicator function of the set  $E$ .

**Basic Facts** We will use the following two basic facts.



**Fact 2.1.** Let  $x \in \mathbb{R}^d$  and  $p \geq 1$ . Then  $\|x\|_{p+1} \leq \|x\|_p \leq \|x\|_{p+1} d^{\frac{1}{p(p+1)}}$ .

**Fact 2.2.** If  $\mathbf{A}, \mathbf{B}, \mathbf{C}$  are symmetric  $d \times d$  matrices satisfying  $\mathbf{A} \succeq 0$  and  $\mathbf{B} \preceq \mathbf{C}$ , we have that  $\text{tr}(\mathbf{AB}) \leq \text{tr}(\mathbf{AC})$ .

*Proof.* Since  $\mathbf{A}$  is PSD, we can consider its spectral decomposition  $\mathbf{A} = \sum_{i=1}^d \lambda_i v_i v_i^T$ , where  $\lambda_i \geq 0$ . Using the linearity of trace operator, we have that

$$\text{tr}(\mathbf{AB}) = \sum_{i=1}^d \lambda_i \text{tr}(v_i v_i^T \mathbf{B}) = \sum_{i=1}^d \lambda_i \text{tr}(v_i^T \mathbf{B} v_i) \leq \sum_{i=1}^d \lambda_i \text{tr}(v_i^T \mathbf{C} v_i) = \sum_{i=1}^d \lambda_i \text{tr}(v_i v_i^T \mathbf{C}) = \text{tr}(\mathbf{AC}),$$

where the inequality uses that  $\mathbf{B} \preceq \mathbf{C}$  and  $\lambda_i \geq 0$ .  $\square$

We will use the notion of total variation distance, defined below.

**Definition 2.3.** Let  $P, Q$  be two probability distributions on  $\mathbb{R}^d$ . The total variation distance between  $P$  and  $Q$ , denoted by  $d_{\text{TV}}(P, Q)$ , is defined as  $d_{\text{TV}}(P, Q) = \sup_{A \subseteq \mathbb{R}^d} |P(A) - Q(A)|$ . For continuous distributions  $P, Q$  with densities  $p, q$ , we have that  $d_{\text{TV}}(P, Q) = \frac{1}{2} \int_{\mathbb{R}^d} |p(x) - q(x)| dx$ .

Whenever  $d_{\text{TV}}(P, Q) = \epsilon$ , it is sometimes helpful to consider the decomposition below.

**Fact 2.4.** Let a domain  $\mathcal{X}$ . For any  $\epsilon \in [0, 1]$  and for any two distributions  $D_1, D_2$  on  $\mathcal{X}$  with  $d_{\text{TV}}(D_1, D_2) = \epsilon$ , there exist distributions  $D, Q_1, Q_2$  such that  $D_1 = (1 - \epsilon)D + \epsilon Q_1$  and  $D_2 = (1 - \epsilon)D + \epsilon Q_2$ .

This decomposition can be achieved by the following choice of  $Q_1, Q_2$ , and  $D$ :

$$Q_1(x) = \begin{cases} \frac{D_1(x) - D_2(x)}{\epsilon}, & \text{if } D_1(x) > D_2(x) \\ 0, & \text{otherwise} \end{cases}, \quad Q_2(x) = \begin{cases} \frac{D_2(x) - D_1(x)}{\epsilon}, & \text{if } D_2(x) > D_1(x) \\ 0, & \text{otherwise} \end{cases},$$

and  $D(x) = \min\{D_1(x), D_2(x)\} / (1 - \epsilon)$ . In light of [Fact 2.4](#), the adversary that performs corruption in total variation distance can be thought of as ‘‘both additive and subtractive’’ adversary.

**Concentration Inequalities** We will also require following standard results regarding concentration of random variables:

**Fact 2.5** ([\[Ver10\]](#)). Consider a distribution  $D$  on  $\mathbb{R}^d$  that has zero mean and is supported in an  $\ell_2$ -ball of radius  $R$  from the origin. Denote by  $\Sigma$  its covariance matrix and denote by  $\Sigma_N = (1/n) \sum_{i=1}^N X_i X_i^T$  the empirical covariance matrix using  $N$  samples  $X_i \sim D$ . There is a constant  $C$  such that for any  $0 < \epsilon' < 1$  and  $0 < \tau < 1$ , if  $N > C \epsilon'^{-2} \|\Sigma\|_2^{-1} R^2 \log(d/\tau)$ , we have that  $\|\Sigma - \Sigma_N\|_2 \leq \epsilon' \|\Sigma\|_2$ , with probability at least  $1 - \tau$ .

**Fact 2.6** (Gaussian Norm Concentration). For every  $\beta > 0$ ,  $\Pr_{X \sim \mathcal{N}(0, \mathbf{I}_d)} [|\|X\|^2 - d| > \beta] \leq 2e^{-c\beta^2/d}$ , where  $c > 0$  is a universal constant.

**Fact 2.7** ([\[Ach03\]](#)). Let  $0 < \gamma < 1$  and  $u_1, \dots, u_N \in \mathbb{R}^d$ . Let  $z_j$  for  $j \in [L]$  drawn from the uniform distribution on  $\{\pm 1\}^d$ . There exists a constant  $C > 0$  such that, if  $L > C \log(N/\gamma)$ , then, with probability at least  $1 - \gamma$ , we have that  $0.8 \|u_i\|^2 \leq \frac{1}{L} \sum_{j=1}^L (z_j^T u_i)^2 \leq 1.2 \|u_i\|^2$  for all  $i \in [N]$ .

## 2.2 Stability Condition and Its Properties

Our results will hold for every distribution satisfying the following key property [DK19].

**Definition 2.8** ( $(\epsilon, \delta)$ -stable distribution). *Fix  $0 < \epsilon < 1/2$  and  $\delta \geq \epsilon$ . A distribution  $G$  on  $\mathbb{R}^d$  is  $(\epsilon, \delta)$ -stable with respect to  $\mu \in \mathbb{R}^d$  if for any weight function  $w : \mathbb{R}^d \rightarrow [0, 1]$  with  $\mathbf{E}_{X \sim G}[w(X)] \geq 1 - \epsilon$  we have that*

$$\|\mu_{w,G} - \mu\|_2 \leq \delta \quad \text{and} \quad \|\bar{\Sigma}_{w,G} - \mathbf{I}_d\|_2 \leq \delta^2/\epsilon.$$

We call a set of points  $S$   $(\epsilon, \delta)$ -stable when the uniform distribution on  $S$  is stable :

**Definition 2.9** ( $(\epsilon, \delta)$ -stable set). *Fix  $0 < \epsilon < 1/2$  and  $\delta \geq \epsilon$ . A finite set  $S_0 \subset \mathbb{R}^d$  is  $(\epsilon, \delta)$ -stable with respect to  $\mu \in \mathbb{R}^d$  if the empirical distribution  $\mathcal{U}(S_0)$  is  $(\epsilon, \delta)$ -stable with respect to  $\mu$ .*

We begin by stating some examples of stable distributions (see [DK19] for more details). If  $G$  is a subgaussian distribution with identity covariance, then  $G$  is  $(\epsilon, \delta)$ -stable with  $\delta = O(\epsilon\sqrt{\log(1/\epsilon)})$ . If  $G$  is a distribution with covariance at most identity, i.e.,  $\Sigma_G \preceq \mathbf{I}_d$ , then  $G$  is  $(\epsilon, \delta)$ -stable with  $\delta = O(\sqrt{\epsilon})$ . Interpolating these two results, we have that if  $G$  is a distribution with identity covariance and bounded  $k$ -th moment for  $k \geq 4$ , i.e.,  $(\mathbf{E}_{X \sim G}[|v^T(X - \mu)|^k])^{1/k} = O(1)$ , then  $G$  is  $(\epsilon, \delta)$ -stable with  $\delta = O(\epsilon^{1-1/k})$ . Furthermore, it is known that  $\text{poly}(d/\epsilon)$  i.i.d. samples from these distributions also yields a set that contains a large stable subset (see, for example, [DK19, DKP20, DHL19, DKK<sup>+</sup>16]):

**Fact 2.10** ([DK19]). *A set of  $O(d/(\epsilon^2 \log(1/\epsilon)))$  i.i.d. samples from an identity covariance subgaussian distribution is  $(\epsilon, O(\epsilon\sqrt{\log(1/\epsilon)}))$ -stable with respect to  $\mu$  with high probability. Similarly, a set of  $\tilde{O}(d/\epsilon)$  i.i.d. samples from a distribution  $X$  with  $\text{Cov}[X] \preceq \mathbf{I}_d$  contains a large subset  $S$ , which is  $O(\epsilon, O(\sqrt{\epsilon}))$ -stable with respect to its mean  $\mathbf{E}[X]$  with high probability.*

The basic fact regarding stability, which is the starting point of many robust estimation algorithms, is that any slight modification of a stable distribution can not perturb the mean by a large amount, unless it significantly changes its covariance (see, for example, [DKK<sup>+</sup>16, LRV16, DK19]). Here we require a slightly different statement than that of [DK19], and hence provide a proof in [Appendix A](#) for completeness.

**Lemma 2.11** (Certificate Lemma). *Let  $G$  be an  $(\epsilon, \delta)$ -stable distribution with respect to  $\mu \in \mathbb{R}^d$ , for some  $0 < \epsilon < 1/3$  and  $\delta \geq \epsilon$ . Let  $P$  be a distribution with  $d_{\text{TV}}(P, G) \leq \epsilon$ . Denoting by  $\mu_P, \Sigma_P$  the mean and covariance of  $P$ , if  $\lambda_{\max}(\Sigma_P) \leq 1 + \lambda$ , for some  $\lambda \geq 0$ , then  $\|\mu_P - \mu\|_2 = O(\delta + \sqrt{\epsilon\lambda})$ .*

Given [Fact 2.4](#), we can essentially think of an  $\epsilon$ -corrupted version of a stable distribution as a mixture of a stable distribution with a noise distribution, as shown below (see [Appendix A](#) for a proof).

**Lemma 2.12.** *For any  $0 < \epsilon < 1/2$  and  $\delta \geq \epsilon$ , if a distribution  $G$  is  $(2\epsilon, \delta)$ -stable with respect to  $\mu \in \mathbb{R}^d$ , and  $P$  is an  $\epsilon$ -corrupted version of  $G$  in total variation distance, there exist distributions  $G_0$  and  $B$  such that  $P = (1 - \epsilon)G_0 + \epsilon B$  and  $G_0$  is  $(\epsilon, \delta)$ -stable with respect to  $\mu$ .*

We continue with some technical claims related to stability that we prove in [Appendix A](#). Let  $G$  be an  $(\epsilon, \delta)$ -stable distribution with respect to  $\mu$  and  $w$  a weight function with  $\mathbf{E}_{X \sim G}[w(X)] \geq 1 - \epsilon$ . Denoting by  $G_w$  the re-weighted distribution  $G_w(x) := G(x)w(x) / \int_{\mathbb{R}^d} w(x)G(x)dx$ , the stability of  $G$  directly implies that  $1 - \delta^2/\epsilon \leq \mathbf{E}_{X \sim G_w}[(v^T(X - \mu))^2] \leq 1 + \delta^2/\epsilon$ . We require a generalization of this fact for a matrix  $\mathbf{U}$  in place of  $v$  and an arbitrary vector  $b$  in place of  $\mu$ :

**Lemma 2.13.** Fix  $0 < \epsilon < 1/2$  and  $\delta \geq \epsilon$ . Let  $w : \mathbb{R}^d \rightarrow [0, 1]$  such that  $\mathbf{E}_{X \sim G}[w(X)] \geq 1 - \epsilon$  and let  $G$  be an  $(\epsilon, \delta)$ -stable distribution with respect to  $\mu \in \mathbb{R}^d$ . For any matrix  $\mathbf{U} \in \mathbb{R}^{d \times d}$  and any vector  $b \in \mathbb{R}^d$ , we have that

$$\mathbf{E}_{X \sim G_w} [\|\mathbf{U}(X - b)\|_2^2] = \|\mathbf{U}\|_F^2 (1 \pm \delta^2/\epsilon) + \|\mathbf{U}(\mu - b)\|_2^2 \pm 2\delta \|\mathbf{U}\|_F^2 \|\mu - b\|_2.$$

We use this to show [Corollary 2.14](#), which will be required when proving correctness of our algorithm. Although its exact role will become clearer later on, the corollary will be relevant to our analysis because we will filter out outliers using scores of the form  $\|\mathbf{U}(x - b)\|_2^2$  for each point  $x$ .

**Corollary 2.14.** Fix  $0 < \epsilon < 1/2$  and  $\delta \geq \epsilon$ . Let  $G$  be an  $(\epsilon, \delta)$ -stable distribution with respect to  $\mu \in \mathbb{R}^d$ . Let a matrix  $\mathbf{U} \in \mathbb{R}^{d \times d}$  and a function  $w : \mathbb{R}^d \rightarrow [0, 1]$  with  $\mathbf{E}_{X \sim G}[w(X)] \geq 1 - \epsilon$ . For the function  $\tilde{g}(x) = \|\mathbf{U}(x - b)\|_2^2$ , we have that

$$(1 - \epsilon) \|\mathbf{U}\|_F^2 (1 - \delta^2/\epsilon - 2\delta\|b - \mu\|_2) \leq \mathbf{E}_{X \sim G} [w(X)\tilde{g}(X)] \leq \|\mathbf{U}\|_F^2 (1 + \delta^2/\epsilon + \|b - \mu\|_2^2 + 2\delta\|b - \mu\|_2).$$

### 3 Filtering Algorithm with Small Number of Iterations

In this section, we develop a filtering algorithm (in the batch setting) that terminates in  $\text{polylog}(d/\epsilon)$  iterations for any stable set. This leads to an algorithm that runs in near-linear time, i.e.,  $nd \text{polylog}(nd/\epsilon)$ , generalizing the results of [\[DHL19, DKK<sup>+</sup>21b\]](#). Crucially, this algorithm will form the building block of our streaming algorithm in [Section 4](#). We remark that the algorithm of this section works even against the *strong-contamination model* ([Definition 3.1](#) below), where the outliers may not be i.i.d. samples from any distribution, but are allowed to be completely arbitrary.

**Definition 3.1** (Strong Contamination Model). Given a parameter  $0 < \epsilon < 1/2$  and a class of distributions  $\mathcal{D}$ , the strong adversary operates as follows: The algorithm specifies a number of samples  $n$ , then the adversary draws a set of  $n$  i.i.d. samples from some  $D \in \mathcal{D}$  and after inspecting them, removes up to  $\epsilon n$  of them and replaces them with arbitrary points. The resulting set is given as input to the learning algorithm. We call a set  $\epsilon$ -corrupted if it has been generated by the above process.

The main result of this section is the following.

**Theorem 3.2.** Let  $d \in \mathbb{Z}_+$ ,  $0 < \tau < 1$ ,  $0 < \epsilon < \epsilon_0$  for a sufficiently small constant  $\epsilon_0$ , and  $\delta \geq \epsilon$ . Let  $S_0$  be a set of  $n$  points that is  $(C\epsilon, \delta)$ -stable with respect to the (unknown) vector  $\mu \in \mathbb{R}^d$ , for a sufficiently large constant  $C > 0$ . Let  $S$  be an  $\epsilon$ -corrupted version of  $S_0$  in the strong contamination model. There exists an algorithm that given  $\epsilon, \delta, \tau$ , and  $S$ , runs in time  $nd \text{polylog}(d, n, 1/\epsilon, 1/\tau)$ , and outputs a vector  $\hat{\mu}$  such that, with probability at least  $1 - \tau$ , it holds  $\|\mu - \hat{\mu}\|_2 = O(\delta)$ .

We note that [Theorem 3.2](#) applies to any stable set. By [Fact 2.10](#), we directly obtain (i) an  $O(\epsilon\sqrt{\log(1/\epsilon)})$ -accurate estimator given  $O(d/(\epsilon^2/\log(1/\epsilon)))$  many  $\epsilon$ -corrupted samples from an identity covariance subgaussian distribution; and (ii) an  $O(\sqrt{\epsilon})$ -accurate estimator for any distribution  $X \sim D$  with  $\text{Cov}[X] \preceq \mathbf{I}_d$ , given  $\tilde{O}(d/\epsilon)$  many  $\epsilon$ -corrupted samples.

#### 3.1 Setup and Algorithm Description

The pseudocode of the algorithm establishing [Theorem 3.2](#) is presented in [Algorithm 1](#). We will define the necessary notation as needed (see the pseudocode for details). First, we assume that the distribution over the input samples is of the form  $P = (1 - \epsilon)G + \epsilon B$ , where  $G$  is the uniform distribution over the stable set of inliers and  $B$  is the uniform distribution on the outliers. Although

this mixture may seem to suggest that the adversary only adds points, it is without loss of generality. Indeed, in the case that the adversary also removes points, we can think of  $G$  as the distribution of the remaining inliers (which continues to be stable with slightly worse parameters; see [Lemma 2.12](#)).

We begin with a high-level explanation of [Algorithm 1](#). At each iteration  $t$ , we assign a weight  $w_t(x) \in [0, 1]$  to each point  $x$ . Let  $P_t$  be the distribution on  $S$ , weighted according to  $w_t$ . Let  $\mu_t$  and  $\Sigma_t$  be the mean and covariance of  $P_t$ , respectively. We want to assign scores to each point, using spectral properties of  $\Sigma_t$  and the stability of inliers, so that the scores over outliers are more than those of inliers. Essentially, if a direction  $v$  has variance larger than  $1 + \Omega(\delta^2/\epsilon)$ , then the stability of inliers implies that this must be due to outliers. Thus, we can assign scores based on the values  $(v^T(x - \mu_t))^2$  that have provably more mass on outliers than inliers. The filters proposed in [[DKK<sup>+</sup>16](#), [DKK<sup>+</sup>17](#)] assigned scores based on a single direction, the leading eigenvector of  $\Sigma_t$ , and can take as many as  $\Omega(d)$  iterations (see [Section 1.2](#)).

To reduce the number of iterations, we need to filter in all directions of large variance simultaneously. Letting  $\mathbf{B}_t \approx \Sigma_t - (1 - C_1\delta^2/\epsilon)\mathbf{I}_d$ , we would like to filter along all directions where the eigenvalue of  $\mathbf{B}_t$  is within a constant factor from  $\lambda_t := \|\mathbf{B}_t\|_2$ , not necessarily the leading eigenvector of  $\mathbf{B}_t$ . As we show in [Section 3.4](#), this can be approximately achieved by assigning scores for each point  $x$  based on  $g_t(x) := \|\mathbf{M}_t(x - \mu_t)\|_2^2$ , where  $\mathbf{M}_t = \mathbf{B}_t^{\log d}$ . At a high level, this happens because the spectrum of  $\mathbf{M}_t$  is distributed across along *all* large eigenvectors of  $\mathbf{B}_t$ .

---

**Algorithm 1** Robust Mean Estimation in polylog iterations

---

- 1: **Input:**  $S = \{x_i\}_{i \in [n]}$ ,  $\delta, \epsilon$
  - 2: Let  $C_1 \geq 22$ ,  $C$  be a sufficiently large constant,  $C_2 = 100C$  and  $C_3 = 0.1$ .
  - 3: Let  $R = \sqrt{(d/\epsilon)(1 + \delta^2/\epsilon)}$ .
  - 4: Let  $P = (1 - \epsilon)G + \epsilon B$  be the empirical distribution on the points from  $S$ . <sup>1</sup>
  - 5: Let  $K = C \log d \log(dR/\epsilon)$ ,  $L = C \log((n + d)K/\tau)$ .
  - 6: Obtain a naïve estimate  $\hat{\mu}$  of  $\mu$  with  $\|\hat{\mu} - \mu\|_2 \leq 4R$ .
  - 7: Initialize  $w_1(x) \leftarrow \mathbf{1}\{\|x - \hat{\mu}\|_2 \leq 5R\}$  for all  $x \in S$ .
  - 8: **for**  $t \in [K]$  **do**
  - 9: Let  $P_t$  be the distribution of  $P$  weighted by  $w_t$ .
  - 10: Let  $\mu_t, \Sigma_t$  be the mean and covariance of  $P_t$ .
  - 11: Let  $\mathbf{B}_t = (\mathbf{E}_{X \sim P}[w_t(X)])^2 \Sigma_t - \left(1 - C_1 \frac{\delta^2}{\epsilon}\right) \mathbf{I}_d$
  - 12: Let  $\mathbf{M}_t = \mathbf{B}_t^{\log d}$ . ▷  $\mathbf{M}_t$  does not need to be explicitly calculated.
  - 13: Let  $\lambda_t = \|\mathbf{B}_t\|_2$
  - 14: Find  $\hat{\lambda}_t \in [0.8\lambda_t, 1.2\lambda_t]$  by power iteration. ▷ See [Remark 3.3](#) for efficient implementation.
  - 15: **if**  $\hat{\lambda}_t > C_2\delta^2/\epsilon$  **then**
  - 16: **for**  $j \in [L]$  **do**
  - 17:  $z_{t,j} \sim \mathcal{U}(\{\pm 1\}^d)$ ,
  - 18:  $v_{t,j} \leftarrow \mathbf{M}_t z_{t,j}$ . ▷ See [Remark 3.3](#) for efficient implementation.
  - 19: **end for**
  - 20: Denote by  $\mathbf{U}_t$  the matrix having the vectors  $\frac{1}{\sqrt{L}}v_{t,j}$  for  $j \in [L]$  as rows.
  - 21: Let  $\tilde{g}_t(x) = \|\mathbf{U}_t(x - \mu_t)\|_2^2$  and  $\tilde{\tau}_t(x) = \tilde{g}_t(x) \mathbf{1}\{\tilde{g}_t(x) > C_3 \|\mathbf{U}_t\|_F^2 \hat{\lambda}_t/\epsilon\}$ ,
  - 22:  $\ell_{\max} \leftarrow (dR/\epsilon)^{C \log d}$ ,  $T \leftarrow 0.01 \hat{\lambda}_t \|\mathbf{U}_t\|_F^2$ .
  - 23:  $w_{t+1} \leftarrow \text{DownweightingFilter}(P, w_t, \tilde{\tau}_t, R, T, \ell_{\max})$  ▷ [Algorithm 2](#)
  - 24: **end if**
  - 25: **end for**
  - 26: **return**  $\mu_t$ .
-

---

**Algorithm 2** Downweighting Filter

---

- 1: **Input:**  $P, w, \tilde{\tau}, R, T, \ell_{\max}$
  - 2:  $r \leftarrow CdR^{2+4\log d}$ .
  - 3: Let  $w_\ell(x) = w(x)(1 - \tilde{\tau}(x)/r)^\ell$ .
  - 4: Find the smallest  $\ell \in \{1, \dots, \ell_{\max}\}$  satisfying  $\mathbf{E}_{X \sim P} [w_\ell(X)\tilde{\tau}(X)] \leq 2T$  using binary search.
  - 5: **return**  $w_\ell$ .
- 

Even though assigning scores based on  $\mathbf{M}_t$ , i.e.,  $g_t(x)$ , reduces the number of iterations, computing  $g_t(x)$  for all  $x \in S$  is slow. We thus use a Johnson-Lindenstrauss (JL) sketch of  $\mathbf{M}_t$ , denoted by  $\mathbf{U}_t$ . We denote by  $\tilde{g}_t(x) := \|\mathbf{U}_t(x - \mu_t)\|_2$  the resulting scores. We claim that the set  $\{\tilde{g}_t(x)\}_{x \in S}$  can be calculated in time  $\tilde{O}(nd)$  such that for each  $x \in S$ ,  $\tilde{g}_t(x) \approx g_t(x)$ . First, we will show (see [Lemma 3.5](#)) that  $\mathbf{U}_t$  can be as small as  $L \times d$ , where  $L$  is some polylog( $nd/(\epsilon\tau)$ ), which follows from the classical JL lemma (stating that  $n$  points can be linearly embedded into a  $\log n$ -dimensional space). Also, each row of  $\mathbf{U}_t$  can be computed by repeatedly multiplying a vector  $\log d$  times by  $\mathbf{B}_t$  ([Line 18](#)). By the following remark, all rows of  $\mathbf{U}_t$  can be computed in time  $\tilde{O}(nd)$  and thus, each iteration of [Algorithm 1](#) runs in near-linear time.

**Remark 3.3.** (Efficient Implementation) Note that for any  $v \in \mathbb{R}^d$ , the vector  $\mathbf{B}_t v$  can be calculated in  $O(nd)$  time. This is because  $\Sigma_t v = \sum_{x \in S} w_t(x)(v^T(x - \mu_t))(x - \mu_t) / (\sum_{x \in S} w_t(x))$  which means that the result can be computed in  $O(nd)$  time by calculating  $\mu_t$  and  $v^T(x - \mu_t)$  first. Regarding [Line 14](#), an approximate large eigenvector can be computed via power iteration, i.e., starting from a random Gaussian vector and multiplying by  $\mathbf{B}_t$  iteratively  $\log d$  many times (see, for example, [\[BHK20\]](#)). As mentioned above, each of these multiplications can be done in  $O(nd)$  time.

For the proof of correctness, we require that the JL and spectral approximations used by the algorithm are sufficiently accurate. We prove that the following event occurs with high probability.

**Condition 3.4** (Deterministic Conditions For [Algorithm 1](#)). For all  $t \in [K]$ , the following hold:

1. *Spectral norm of  $\mathbf{B}_t$ :*  $\hat{\lambda}_t \in [0.8\lambda_t, 1.2\lambda_t]$ .
2. *Frobenius norm:*  $\|\mathbf{U}_t\|_F^2 \in [0.8\|\mathbf{M}_t\|_F^2, 1.2\|\mathbf{M}_t\|_F^2]$ .
3. *Scores:* For all  $x \in S$ ,  $\tilde{g}_t(x) \in [0.8g_t(x), 1.2g_t(x)]$ .

### 3.2 Establishing the Deterministic Conditions

In this section, we establish that [Condition 3.4](#) holds with high probability. Regarding [Item 1](#) of this condition, an approximate large eigenvector can be computed via power iteration as described in [Remark 3.3](#). This gives us an algorithm that runs in time  $O(nd \log d \log(K/\tau))$  and satisfies [Item 1](#) with probability  $1 - \tau$ .

We now move to the other two conditions. The claim is that instead of using the matrix  $\mathbf{M}_t$  to calculate the scores, it suffices to store and use only a small set of random projections  $\{\mathbf{M}_t^{z_{t,j}}\}_{j \in [L]}$ . This is exactly the Johnson-Lindenstrauss sketch that is computed in [Line 16](#) of [Algorithm 1](#). Using [Fact 2.7](#), we get the following guarantees (see [Appendix B.1](#) for the proof).

**Lemma 3.5.** *Fix a set of  $n$  points  $x_1, \dots, x_n \in \mathbb{R}^d$ . For  $t \in [K]$ , define  $g_t(x) := \|\mathbf{M}_t(x - \mu_t)\|_2^2$  and let  $\tilde{g}_t(x), v_{t,j}$  as in [Algorithm 1](#). If  $C$  is a sufficiently large constant and  $L = C \log((n + d)K/\tau)$ , with probability at least  $1 - \tau$ , for every  $t \in [K]$  we have the following:*

---

<sup>1</sup>Without loss of generality, outliers are within  $O(R)$  from  $\mu$  in  $\ell_2$ -norm. This is ensured in [Line 7](#), which removes only  $\epsilon$ -fraction of inliers ([Claim 3.12](#)).

1.  $0.8g_t(x_i) \leq \tilde{g}_t(x_i) \leq 1.2g_t(x_i)$  for every  $i \in [n]$ ,
2.  $0.8\|\mathbf{M}_t\|_F^2 \leq \left(\frac{1}{L} \sum_{j=1}^L \|v_{t,j}\|_2^2\right) \leq 1.2\|\mathbf{M}_t\|_F^2$ .

This concludes the proof that [Condition 3.4](#) is satisfied with high probability.

### 3.3 Downweighting Filter

We use the following re-weighting procedure also used in [\[DHL19\]](#). Recall that  $P$  denotes the empirical distribution on the samples, which we write as  $P = (1 - \epsilon)G + \epsilon B$ , where  $G$  and  $B$  are the contributions from the good and bad samples respectively. Roughly speaking, our filter guarantees two things when going from the weights  $w(x)$  to  $w'(x)$ :

1. *The weight removed from the outliers is greater than the weight removed from the inliers.*
2.  $\mathbf{E}_{X \sim P}[w'(X)\tilde{\tau}(X)] \leq 2\mathbf{E}_{X \sim G}[w(X)\tilde{\tau}(X)]$ , i.e., *the weighted mean of scores after filtering over both inliers and outliers is at most twice the weighted mean of scores of inliers before filtering.*

Regarding the first guarantee, since the fraction of outliers is at most  $\epsilon$ , this ensures that the filtered distribution  $P_t$  will never be more than  $O(\epsilon)$ -far in total variation distance from the initial (corrupted) distribution, and thus the condition of the certificate lemma that  $d_{\text{TV}}(P, P_t) \leq O(\epsilon)$  will always be satisfied. The second guarantee ensures that the filtering step reduces the average score significantly. We prove the following in [Appendix B.2](#).

**Lemma 3.6.** *Let  $P = (1 - \epsilon)G + \epsilon B$  be the empirical distribution on  $n$  samples, as in [Algorithm 1](#). If  $(1 - \epsilon)\mathbf{E}_{X \sim G}[w(X)\tilde{\tau}(X)] \leq T$ ,  $\|\tilde{\tau}\|_\infty \leq r$ , and  $\ell_{\max} > r/T$ , then [Algorithm 2](#) modifies the weight function  $w$  to  $w'$  such that*

1.  $(1 - \epsilon)\mathbf{E}_{X \sim G}[w(X) - w'(X)] < \epsilon\mathbf{E}_{X \sim B}[w(X) - w'(X)]$ ,
2.  $\mathbf{E}_{X \sim P}[w'(X)\tilde{\tau}(X)] \leq 2T$ ,

and the algorithm terminates after  $O(\log(\ell_{\max}))$  iterations, each of which takes  $O(n)$  time.

We note that the two conditions  $\|\tilde{\tau}\|_\infty \leq r$ ,  $\ell_{\max} > r/T$  of [Lemma 3.6](#) hold by our choice of  $\ell_{\max}$  and  $r$  inside [Algorithm 1](#) and [Algorithm 2](#) as follows. For  $\|\tilde{\tau}\|_\infty$ , we have the following upper bound

$$\tilde{\tau}_t(x) \leq \tilde{g}_t(x) \leq \|\mathbf{U}_t(x - \mu_t)\|_2^2 \lesssim R^2\|\mathbf{U}_t\|_2^2 \lesssim R^2\|\mathbf{M}_t\|_F^2 \lesssim R^2\|\Sigma_t\|_2^{2\log d} = O(dR^{2+4\log d}), \quad (1)$$

where we used the guarantee of our JL sketch that  $\|\mathbf{U}_t\|_2^2 \leq 1.2\|\mathbf{M}_t\|_F^2$  ([Lemma 3.5](#)). A crude upper bound on  $r/T$  follows from the following inequalities:

$$\frac{r}{T} \lesssim \frac{dR^{2+4\log d}}{\widehat{\lambda}_t\|\mathbf{M}_t\|_F^2} \lesssim \frac{dR^{2+4\log d}}{\lambda_t\|\mathbf{M}_t\|_F^2} \lesssim \left(\frac{dR}{\delta^2/\epsilon}\right)^{O(\log d)},$$

where the first inequality uses the values of  $r$  and  $T$  as set in the algorithm, the second inequality uses [Items 1](#) and [2](#) of the deterministic conditions of [Condition 3.4](#), and the last inequality uses the fact that  $\|\mathbf{M}_t\|_F^2 \geq \|\mathbf{M}_t\|_2^2$  and  $\|\mathbf{M}_t\|_2^2$  cannot be smaller than  $(C_2\delta^2/\epsilon)^{O(\log d)}$  (otherwise [Line 22](#) terminates the algorithm).

We now use the guarantees of [Algorithm 2](#) as follows. We first show that the weighted mean of the inliers' scores is small.



**Lemma 3.7.** *Under the setting of [Algorithm 1](#) and the deterministic [Condition 3.4](#), we have that  $\mathbf{E}_{X \sim G}[w_t(X)\tau_t(X)]$  and  $\mathbf{E}_{X \sim G}[w_t(X)\tilde{\tau}_t(X)]$  are bounded from above by  $c\lambda_t\|\mathbf{M}_t\|_F^2$  for some constant  $c$  of the form  $c = C/C_2$ , where  $C_2$  is the constant used in [Line 15](#) and  $C$  is some absolute constant.*

The proof is based on stability arguments from [Section 2.2](#) and can be found in [Appendix B.3](#).

**Remark 3.8.** In our analysis, it will be important that the constant  $c$  in [Lemma 3.7](#) can be made sufficiently smaller than 1, for example,  $c < 0.01$ . This can be achieved by choosing  $C_2$  to be a large enough constant.

Using [Algorithm 2](#), we get that the weighted sum of scores after filtering is also small.

**Lemma 3.9.** *Under the setting of [Algorithm 1](#) and the deterministic [Condition 3.4](#), we have that  $\epsilon \mathbf{E}_{X \sim B}[w_{t+1}(X)\tilde{\tau}_t(X)] < c\lambda_t\|\mathbf{M}_t\|_F^2$ , with  $c$  being of the form  $c = C/C_2$ , where  $C_2$  is the constant used in [Line 15](#) and  $C$  is some absolute constant. Furthermore,  $\epsilon \mathbf{E}_{X \sim B}[w_{t+1}(X)\tau_t(X)] < c\lambda_t\|\mathbf{M}_t\|_F^2$ .*

*Proof.* The first claim follows by the stopping condition of the algorithm, [Lemma 3.7](#), and the fact that  $w_{t+1} \leq w_t$ . We now prove the second conclusion by relating  $\tau$  to  $\tilde{\tau}$ : Recall that we denote by  $S$  the  $\epsilon$ -corrupted version of the original set of samples  $S_0$ . Since  $\tilde{g}_t(x)$  is within a constant factor of  $g_t(x)$  ([Condition 3.4](#)) for all  $x \in S$ , the scores  $\tilde{\tau}_t(x)$  and  $\tau_t(x)$  are comparable (up to an additive term) as shown below.

**Claim 3.10.** *In the setting of [Algorithm 1](#) and under the [Condition 3.4](#), if  $x \in S$ , we have that  $\tau_t(x) \leq 1.25\tilde{\tau}_t(x) + 3C_3(\lambda_t/\epsilon)\text{tr}(\mathbf{M}_t^2)$ , where  $C_3$  is the constant used in [Algorithm 1](#).*

We prove [Claim 3.10](#) in [Appendix B.5](#). Using [Claim 3.10](#), we have the following set of inequalities:

$$\begin{aligned} \epsilon \cdot \mathbf{E}_{X \sim B}[w_{t+1}(X)\tau_t(X)] &= \frac{1}{n} \sum_{x_i \in S \setminus S_0} w_{t+1}(x_i)\tau_t(x_i) \\ &\leq 3C_3\lambda_t \|\mathbf{M}_t\|_F^2 + \frac{1}{n} 1.25 \sum_{i \in S \setminus S_0} w_{t+1}(x_i)\tilde{\tau}_t(x_i) \quad (\text{using [Claim 3.10](#)}) \\ &= 3C_3\lambda_t\|\mathbf{M}_t\|_F^2 + 1.25\epsilon \mathbf{E}_{X \sim B}[w_{t+1}(X)\tilde{\tau}_t(X)] \\ &\leq 3C_3\lambda_t\|\mathbf{M}_t\|_F^2 + 1.25c\lambda_t\|\mathbf{M}_t\|_F^2 \quad (\text{using the first part of [Lemma 3.9](#)}) \\ &< 10(C/C_2)\lambda_t\|\mathbf{M}_t\|_F^2. \quad (\text{using the value of } C_3) \end{aligned}$$

The last inequality above uses the fact that the constant  $C_3$  is chosen to be  $C_3 = C/C_2$  in [Algorithm 1](#), where  $C$  is a sufficiently large constant.  $\square$

### 3.4 Correctness of [Algorithm 1](#): Proof of [Theorem 3.2](#)

The rest of this section is dedicated to proving [Theorem 3.2](#). We first state the correctness of the naive approximation step of [Line 6](#), then record the invariants of the algorithm in [Section 3.4.1](#), and finally show in [Section 3.4.2](#) that it suffices for the number of iterations  $K$  to be bounded by some  $\text{polylog}(d, R, 1/\epsilon)$ .

The naive approximation step of [Line 6](#) is based on the following folklore fact (see [Appendix B.4](#) for more details).

**Claim 3.11.** *Let the fraction of outliers be  $\epsilon < 1/10$  and a parameter  $0 < \tau < 1$ . Let the distribution  $P = (1 - \epsilon)G + \epsilon B$ . Let  $R > 0, \mu \in \mathbb{R}^d$  be such that  $\Pr_{X \sim G}[\|X - \mu\|_2 > R] \leq \epsilon$ . There is an estimator  $\hat{\mu}$  on  $k = O(\log(1/\tau))$  samples from  $P$  such that  $\|\hat{\mu} - \mu\|_2 \leq 4R$  with probability at least  $1 - \tau$ . Furthermore,  $\hat{\mu}$  can be computed in time  $O(k^2d)$  and memory  $O(kd)$ .*

[Claim 3.12](#) below gives a valid upper bound on  $R$  using the  $(\epsilon, \delta)$ -stability of the good distribution.

**Claim 3.12.** *If  $R = \sqrt{\frac{d}{\epsilon}(1 + \delta^2/\epsilon)}$ , then  $\Pr_{X \sim G}[\|X - \mu\|_2 > R] \leq \epsilon$ .*

*Proof.* By Markov's inequality, we have that

$$\Pr_{X \sim G} \left[ \|X - \mu\|_2^2 \geq \frac{d}{\epsilon} (1 + \delta^2/\epsilon) \right] \leq \epsilon \frac{\mathbf{E}_{X \sim G}[\|X - \mu\|_2^2]}{d(1 + \delta^2/\epsilon)} \leq \epsilon.$$

□

### 3.4.1 Invariants of [Algorithm 1](#)

Recall that the end goal is to obtain a filtered version,  $P_t$ , of  $P$  that is not too far from  $P$  in total variation distance  $d_{\text{TV}}(P, P_t) = O(\epsilon)$ , and satisfies that  $\|\mathbf{B}_t\|_2 = O(\delta^2/\epsilon)$ . For the first condition to be satisfied, we ensure that the Downweighting filter removes more weight from  $G$  than  $B$  ([Lemma 3.6](#)). Using this, we show that  $\mathbf{E}_{X \sim G}[w_t(X)] \geq 1 - O(\epsilon)$ , which implies the bound on the total variation distance ([Claim 3.13](#)). The proofs are deferred to [Appendix B.5](#).

**Claim 3.13.** *Under [Condition 3.4](#), [Algorithm 1](#) maintains the following invariant:  $\mathbf{E}_{X \sim G}[w_t(X)] \geq 1 - 3\epsilon$ . In particular, if  $\epsilon \leq 1/8$ , then  $d_{\text{TV}}(P_t, P) \leq 9\epsilon$ .*

The following properties of  $\mathbf{B}_t$  as PSD operator will also be useful later on.

**Claim 3.14.** *Under [Condition 3.4](#), if  $C_1 \geq 22$ ,  $\mathbf{B}_t \succeq (0.5C_1\delta^2/\epsilon)\mathbf{I}_d$  for every  $t \in [K]$ .*

The proof of [Claim 3.14](#) is provided in [Appendix B.5](#). Although just showing that  $\mathbf{B}_t \succeq 0$  would suffice for this section, the slightly stronger bound of the above claim will be useful in [Section 4](#). [Claim 3.14](#) follows from [Claim 3.13](#) and the stability of  $G$ . In particular, the stability of  $G$  implies that  $\bar{\Sigma}_G \succeq (1 - \delta^2/\epsilon)\mathbf{I}_d$ . We now prove the following claim, which is the reason for having the multiplicative factor of  $\mathbf{E}_{X \sim P}[w_t(X)]^2$  in the definition of  $\mathbf{B}_t$ .

**Claim 3.15.** *We have that  $\mathbf{B}_{t+1} \preceq \mathbf{B}_t$  for every  $t \in [K]$ .*

*Proof.* We use the alternative definition of the covariance matrix: Let  $X, Y$  be i.i.d. from  $P$ , then

$$\Sigma_t = \frac{1}{2(\mathbf{E}_{X \sim P}[w_t(X)]^2)} \mathbf{E}_{X, Y \sim P} [w_t(X)w_t(Y)(X - Y)(X - Y)^T].$$

Since  $w_{t+1}(x) \leq w_t(x)$  for all  $x$ , this completes the proof. □

### 3.4.2 Reducing the Potential Function

Recall that each iteration of [Algorithm 1](#) can be implemented in near-linear time. Thus, it remains to show that the choice  $K = C \log d \log(dR/\epsilon)$  suffices to guarantee correctness of our algorithm. We now sketch the proof using a potential function argument. Let  $\Lambda_t$  be the vector in  $\mathbb{R}^d$  containing the eigenvalues of  $\mathbf{B}_t$ . Recall that our goal is to show that  $\|\mathbf{B}_t\|_2 = \|\Lambda_t\|_\infty = O(\delta^2/\epsilon)$  in polylog many iterations. Let  $p := 2 \log d$ . Since  $\|x\|_p = \Theta(\|x\|_\infty)$  for any  $x \in \mathbb{R}^d$ , we are motivated to use the potential function  $\phi_t := \|\Lambda_t\|_p^p$ . We now focus on showing that  $\phi_t$  decreases rapidly. Observe that for any  $i \in \mathbb{Z}_+$ ,  $\text{tr}(\mathbf{B}_t^i) = \|\Lambda_t\|_i^i$ . We start with the following inequalities (and explain them directly below):

$$\phi_{t+1} = \|\Lambda_{t+1}\|_p^p \leq \left( d^{\frac{1}{p(p+1)}} \|\Lambda_{t+1}\|_{p+1} \right)^p$$



$$\begin{aligned}
&= d^{\frac{1}{p+1}} \left( \|\Lambda_{t+1}\|_{p+1}^{p+1} \right)^{\frac{p}{p+1}} \\
&= d^{\frac{1}{p+1}} (\text{tr}(\mathbf{B}_{t+1}^{p+1}))^{\frac{p}{p+1}} \\
&= d^{\frac{1}{p+1}} (\text{tr}(\mathbf{M}_{t+1}\mathbf{B}_{t+1}\mathbf{M}_{t+1}))^{\frac{p}{p+1}} \\
&\leq d^{\frac{1}{p}} (\text{tr}(\mathbf{M}_t\mathbf{B}_{t+1}\mathbf{M}_t))^{\frac{p}{p+1}}, \tag{2}
\end{aligned}$$

where the first line uses [Fact 2.1](#), the third one uses  $\text{tr}(\mathbf{B}_{t+1}^i) = \|\Lambda_{t+1}\|_i^i$ , the fourth line uses that  $\mathbf{M}_{t+1}\mathbf{B}_{t+1}\mathbf{M}_{t+1} = \mathbf{B}_{t+1}^{p+1}$ , and the last one uses the fact  $\mathbf{B}_{t+1} \preceq \mathbf{B}_t$ , which holds because removing points can only make their covariance smaller; see [Section 3.4](#) for more details.

Then the goal becomes to bound from above the term  $\text{tr}(\mathbf{M}_t\mathbf{B}_{t+1}\mathbf{M}_t)$ . The claim is that  $\text{tr}(\mathbf{M}_t\mathbf{B}_{t+1}\mathbf{M}_t)$  is related to  $\mathbf{E}_{X \sim P}[w_{t+1}(X)\tau_t(X)]$ , and thus can be bounded by  $c\lambda_t\|\mathbf{M}_t\|_F^2$ . Using the guarantees of the Downweighting filter ([Lemma 3.9](#)), we prove the following result:

**Lemma 3.16.** *Consider the setting of [Algorithm 1](#) and assume that [Condition 3.4](#) holds. Then  $\text{tr}(\mathbf{M}_t\mathbf{B}_{t+1}\mathbf{M}_t) \leq c\lambda_t\|\mathbf{M}_t\|_F^2$  for some  $c$  of the form  $C/C_2$ , where  $C_2$  is the constant used in [Line 15](#) and  $C$  is some absolute constant.*

Before giving the details regarding [Lemma 3.16](#), we first show that it suffices to prove that the potential function decreases by a multiplicative factor. In the rest of the proof, we will assume that  $c < 0.1$ , which can be guaranteed by taking  $C_2$  to be a sufficiently large constant (cf. [Remark 3.8](#)). We continue with [Equation \(2\)](#) as follows:

$$\begin{aligned}
\phi_{t+1} &\leq d^{\frac{1}{p}} (\text{tr}(\mathbf{M}_t\mathbf{B}_{t+1}\mathbf{M}_t))^{\frac{p}{p+1}} \\
&\leq d^{\frac{1}{p}} (c\|\Lambda_t\|_\infty\|\Lambda_t\|_p^p)^{\frac{p}{p+1}} && \text{(using [Lemma 3.16](#))} \\
&\leq d^{\frac{1}{p}} c^{\frac{p}{p+1}} (\|\Lambda_t\|_p\|\Lambda_t\|_p^p)^{\frac{p}{p+1}} && \text{(using } \|\Lambda_t\|_\infty \leq \|\Lambda_t\|_i \text{ for } i \geq 1) \\
&= d^{\frac{1}{p}} c^{\frac{p}{p+1}} \|\Lambda_t\|_p^p \\
&\leq 3\sqrt{c}\|\Lambda_t\|_p^p \leq 0.9999\phi_t,
\end{aligned}$$

where the last line uses that  $d^{1/p} = \exp(\frac{\log d}{2\log d}) \leq 3$ ,  $p/(p+1) \geq 0.5$ , and  $c < 1$ . We thus get the desired convergence.

The final step is to bound the number of iterations needed for [Lemma 2.11](#) to ensure that  $\|\mu_t - \mu\|_2 = O(\delta)$ . Concretely, due to our naïve pruning, at the beginning of the algorithm we have the upper bound  $\phi_1 \leq dR^{O(\log d)}$ . After  $K$  iterations, we have that  $\phi_K \leq 0.99^K dR^{O(\log d)}$ . Setting  $K$  as

$$K = C \log d \log \left( \frac{dR}{\delta^2/\epsilon} \right) \tag{3}$$

suffices to have that  $\|\mathbf{B}_K\|_2 \leq (dO(\delta^2/\epsilon)^{2\log d})^{\frac{1}{2\log d}} = O(\delta^2/\epsilon)$ . This implies that

$$\|\Sigma_K\|_2 \leq \frac{1}{\mathbf{E}_{X \sim P}[w_K(X)]^2} (\|\mathbf{B}_K\|_2 + 1) \leq 1 + \left( \frac{1}{(1-3\epsilon)^2} - 1 \right) + O\left(\frac{\delta^2}{\epsilon}\right) \leq 1 + O\left(\frac{\delta^2}{\epsilon}\right), \tag{4}$$

where we used that  $\mathbf{E}_{X \sim P}[w_K(X)]^2 \geq 1 - 3\epsilon$ ,  $\delta \geq \epsilon$ , and  $\epsilon \leq \epsilon_0$ . An application of [Lemma 2.11](#) shows that the estimate has error at most  $\|\mu_t - \mu\|_2 = O(\delta)$ . This completes the proof of [Theorem 3.2](#). The rest of the section focuses on proving [Lemma 3.16](#).

**Proof Sketch of Lemma 3.16** Before giving the full proof of Lemma 3.16, we provide a brief proof sketch. By the definition of  $\mathbf{B}_{t+1}$ , we have the following (the full proof is deferred to the end of this section)

$$\text{tr}(\mathbf{M}_t \mathbf{B}_{t+1} \mathbf{M}_t) \leq \mathbf{E}_{X \sim P} [w_{t+1}(X)] \mathbf{E}_{X \sim P} [w_{t+1}(X) g_t(X)] - \left(1 - C_1 \frac{\delta^2}{\epsilon}\right) \|\mathbf{M}_t\|_F^2.$$

In order to bound from above  $\mathbf{E}_{X \sim P} [w_{t+1}(X) g_t(X)]$ , one can consider the contribution due to inliers (distribution  $G$ ) and contribution due to outliers (distribution  $B$ ). Using the stability of inliers and Corollary 2.14, we have that  $\mathbf{E}_{X \sim G} [w(X) g_t(X)] \leq (1 + c\lambda_t) \|\mathbf{M}_t\|_F^2$ , for any weight function  $w$  satisfying the conditions of Corollary 2.14. We know that  $w_{t+1}$  satisfies them because of our invariant in Claim 3.13. Turning to the contribution of outliers, we want to bound  $\epsilon \cdot \mathbf{E}_{X \sim B} [w_{t+1}(X) g_t(X)]$ . By definition, we have that  $g_t(x) \leq \tau_t(x) + C_3 \lambda_t \|\mathbf{M}_t\|_F^2 / \epsilon$ , and thus we get that the desired expression is bounded from above by  $\epsilon \mathbf{E}_{X \sim B} [w_{t+1}(X) \tau_t(X)] + C_3 \lambda_t \|\mathbf{M}_t\|_F^2$ . The first expression was bounded from above in Lemma 3.9 by using the downweighting filter, and the second is small because of how  $C_3$  is set in our algorithm.

This completes the proof sketch of Lemma 3.16. We now provide the complete proof.

*Proof of Lemma 3.16.* We will bound the contribution of inliers and outliers to the quantity  $\mathbf{E}_{X \sim P} [w_{t+1}(X) g_t(X)]$  from above. Recall from our notation that the decomposition into inliers and outliers is  $P = (1 - \epsilon)G + \epsilon B$ . For the inliers, we use Corollary 2.14 with  $\mathbf{U} = \mathbf{M}_t$  and  $b = \mu_t$  to obtain the following:

$$\mathbf{E}_{X \sim G} [w_{t+1}(X) g_t(X)] \leq \|\mathbf{M}_t\|_F^2 \left(1 + \frac{\delta^2}{\epsilon} + \|\mu_t - \mu\|_2^2 + 2\delta \|\mu_t - \mu\|_2\right) \leq \|\mathbf{M}_t\|_F^2 (1 + c\lambda_t), \quad (5)$$

where the last inequality uses that, by the certificate lemma (Lemma 2.11), every term except the first in the previous expression is less than a sufficiently small fraction of  $\lambda_t$ .

Regarding the outliers, we decompose their contribution to  $\mathbf{E}_{X \sim P} [w_{t+1}(X) g_t(X)]$  into two sets: (i) the set of points with projection greater than the threshold  $C_3 \|\mathbf{M}_t\|_F^2 \lambda_t / \epsilon$  used in Line 21 of the algorithm, and (ii) the set of points with smaller projection. Concretely, letting  $L_t := \{x \in \mathbb{R}^d : g_t(x) > C_3 \|\mathbf{M}_t\|_F^2 \lambda_t / \epsilon\}$ , we have that

$$\begin{aligned} \epsilon \mathbf{E}_{X \sim B} [w_{t+1}(X) g_t(X)] &= \epsilon \mathbf{E}_{X \sim B} [w_{t+1}(X) \tau_t(X)] + \epsilon \mathbf{E}_{X \sim B} [w_{t+1}(X) g_t(X) \mathbf{1}\{x \notin L_t\}] \\ &\leq c\lambda_t \|\mathbf{M}_t\|_F^2 + \epsilon C_3 \|\mathbf{M}_t\|_F^2 \lambda_t / \epsilon \leq c' \|\mathbf{M}_t\|_F^2 \lambda_t, \end{aligned} \quad (6)$$

where the first inequality follows from Lemma 3.9 and the second inequality follows from the choice of  $C_3$  in Algorithm 1.

We also use the following relation on  $\Sigma_{t+1}$ :

$$\begin{aligned} \Sigma_{t+1} &= \mathbf{E}_{X \sim P_{t+1}} [(X - \mu_{t+1})(X - \mu_{t+1})^T] \\ &\preceq \mathbf{E}_{X \sim P_{t+1}} [(X - \mu_t)(X - \mu_t)^T] \\ &= \frac{1}{\mathbf{E}_{X \sim P} [w_{t+1}(X)]} \mathbf{E}_{X \sim P} [w_{t+1}(X) (X - \mu_t)(X - \mu_t)^T]. \end{aligned} \quad (7)$$

Recalling the definition  $\mathbf{B}_{t+1} = (\mathbf{E}_{X \sim P} [w_{t+1}(X)])^2 \Sigma_{t+1} - \left(1 - C_1 \frac{\delta^2}{\epsilon}\right) \mathbf{I}_d$ , Equation (7) implies that  $\mathbf{B}_{t+1} \preceq \mathbf{F}_{t+1}$ , where  $\mathbf{F}_{t+1} := (\mathbf{E}_{X \sim P} [w_{t+1}(X)]) \mathbf{E}_{X \sim P} [w_{t+1}(X) (X - \mu_t)(X - \mu_t)^T] - \left(1 - C_1 \frac{\delta^2}{\epsilon}\right) \mathbf{I}_d$ . Using Fact 2.2 along with the fact that  $\mathbf{B}_{t+1} \succeq 0$  (Claim 3.14), we get the following:

$$\text{tr}(\mathbf{M}_t \mathbf{B}_{t+1} \mathbf{M}_t) = \text{tr}(\mathbf{M}_t^2 \mathbf{B}_{t+1}) \leq \text{tr}(\mathbf{M}_t^2 \mathbf{F}_{t+1}) \quad (\text{using } \text{tr}(\mathbf{ABC}) = \text{tr}(\mathbf{CAB}))$$

$$\begin{aligned}
&= \text{tr} \left( \mathbf{M}_t \left( \mathbf{E}_{X \sim P} [w_{t+1}(X)] \mathbf{E}_{X \sim P} [w_{t+1}(X)(X - \mu_t)(X - \mu_t)^T] - \left(1 - C_1 \frac{\delta^2}{\epsilon}\right) \mathbf{I}_d \right) \mathbf{M}_t \right) \\
&= \mathbf{E}_{X \sim P} [w_{t+1}(X)] \mathbf{E}_{X \sim P} [w_{t+1}(X) \text{tr}((X - \mu_t)^T \mathbf{M}_t^2 (X - \mu_t))] - \left(1 - C_1 \frac{\delta^2}{\epsilon}\right) \|\mathbf{M}_t\|_F^2 \\
&= \mathbf{E}_{X \sim P} [w_{t+1}(X)] \mathbf{E}_{X \sim P} [w_{t+1}(X) g_t(X)] - \left(1 - C_1 \frac{\delta^2}{\epsilon}\right) \|\mathbf{M}_t\|_F^2 \\
&\leq (1 + c\lambda_t + c'\lambda_t - (1 - C_1\delta^2/\epsilon)) \|\mathbf{M}_t\|_F^2 \quad (\text{using Equations (5) and (6)}) \\
&\leq c''\lambda_t \|\mathbf{M}_t\|_F^2.
\end{aligned}$$

This concludes the proof.  $\square$

## 4 Efficient Streaming Algorithm for Robust Mean Estimation

We now turn to the main focus of this paper and present a low-memory algorithm for robust mean estimation. Our algorithm works in two setups: (i) the single-pass streaming setting, where a set of i.i.d. samples from an  $\epsilon$ -corrupted distribution in total variation distance (Definition 1.2) arrive *one at a time* (Definition 1.1), and (ii) the strong-contamination model (Definition 3.1), where the algorithm is allowed poly-logarithmically many passes over the input stream (defined below).

**Definition 4.1** (Streaming Model in  $k$  Passes). *For a fixed set  $S$ , the elements of  $S$  are revealed to the algorithm one at a time. This process is repeated  $k$  times. The sequence of elements in  $S$  within each pass can be arbitrary.*

Our main result is the following theorem for the single-pass streaming model, which is a generalized version of Theorem 1.3:

**Theorem 4.2** (Robust Mean Estimation in Single-Pass Streaming Model). *Let  $d \in \mathbb{Z}_+$ ,  $0 < \tau < 1$ ,  $0 < \epsilon < \epsilon_0$  for a sufficiently small constant  $\epsilon_0$ , and  $\delta \geq \epsilon$ . Let  $D$  be a distribution which is  $(C\epsilon, \delta)$ -stable with respect to the (unknown) vector  $\mu \in \mathbb{R}^d$ , for a sufficiently large constant  $C > 0$ . Let  $R$  be any radius such that  $\Pr_{X \sim D}[\|X - \mu\|_2 > R] \leq \epsilon$ . Let  $P$  be a distribution with  $d_{\text{TV}}(P, D) \leq \epsilon$ . There exists an algorithm that given  $\epsilon, \delta, \tau$  and*

$$n = O \left( R^2 \max \left( d, \frac{\epsilon}{\delta^2}, \frac{(1 + \delta^2/\epsilon)d}{\delta^2 R^2}, \frac{\epsilon^2 d}{\delta^4}, \frac{R^2 \epsilon^2}{\delta^2}, \frac{R^2 \epsilon^4}{\delta^6} \right) \text{polylog} \left( d, \frac{1}{\epsilon}, \frac{1}{\tau}, R \right) \right) \quad (8)$$

*i.i.d. samples from  $P$  in a stream according to the model of Definition 1.1, runs in time  $nd \text{polylog}(d, 1/\epsilon, 1/\tau, R)$ , uses memory  $d \text{polylog}(d, 1/\epsilon, 1/\tau, R)$ , and returns a vector  $\hat{\mu}$  such that, with probability at least  $1 - \tau$ , it holds that  $\|\mu - \hat{\mu}\|_2 = O(\delta)$ .*

Note that Theorem 1.3 in Section 1.1 is a special case of Theorem 4.2 for the two important families of distributions: (i) subgaussian distributions with identity covariance, and (ii) distributions with bounded covariance.

1. For subgaussian distributions with identity covariance, we have that  $R = \Theta(\sqrt{d \log(1/\epsilon)})$ ,  $\delta = O(\epsilon \sqrt{\log(1/\epsilon)})$ , and thus  $n = \tilde{O}(d^2/\epsilon^2)$ .
2. For distributions with covariance at most identity, we have that  $R = \Theta(\sqrt{d/\epsilon})$ ,  $\delta = O(\sqrt{\epsilon})$ , and thus  $n = \tilde{O}(d^2/\epsilon)$ .

In order to obtain a low-memory algorithm for the robust mean estimation problem, we start with an obstacle that one faces when trying to modify the existing [Algorithm 1](#) to that setting. The issue is that, since  $n$  can be much larger than  $d$ , we cannot even store the weight function  $w_t$ . Fortunately, this can be handled by freshly computing the scores  $w_t(x)$  for any given  $x$ , whenever we need them. This requires us to store only  $\{(\mathbf{U}_t, \ell_t) : t \in [K]\}$ , where  $\mathbf{U}_t$  is the Johnson-Lindenstrauss sketch at the iteration  $t$ , and  $\ell_t$  is the corresponding count from the downweighting filter. This can be achieved with additional poly-logarithmic memory. Thus, [Algorithm 1](#) can be readily extended to setting (ii), giving us [Corollary 4.3](#).

**Corollary 4.3** (Robust Mean Estimation in Multiple Passes Streaming Model). *Let  $d \in \mathbb{Z}_+$ ,  $0 < \tau < 1$  and  $0 < \epsilon < \epsilon_0$  for a sufficiently small constant  $\epsilon_0$ , and  $\delta \geq \epsilon$ . Let  $S$  be an  $\epsilon$ -corrupted version of a set that is  $(C\epsilon, \delta)$ -stable with respect to the (unknown) vector  $\mu \in \mathbb{R}^d$ , for a sufficiently large constant  $C$ . Denote by  $n$  the cardinality of  $S$ . There exists an algorithm that operates in the streaming model of [Definition 4.1](#) with  $k = \text{polylog}(d, 1/\epsilon, 1/\tau)$  and, given  $\epsilon, \delta, \tau$  and  $T$ , runs in time  $nd \text{polylog}(d, 1/\epsilon, 1/\tau)$ , uses additional memory  $d \text{polylog}(d, 1/\epsilon, 1/\tau)$ , and finds a vector  $\hat{\mu}$  such that, with probability at least  $1 - \tau$ , it holds  $\|\mu - \hat{\mu}\|_2 = O(\delta)$ .*

In the main body of this section, we prove [Theorem 4.2](#).

## 4.1 Setup and Algorithm Description

Moving to the single-pass streaming model and [Theorem 4.2](#) requires a change in perspective: instead of having a corrupted dataset, we now have sample access to a distribution  $P$  such that  $d_{\text{TV}}(P, D) \leq \epsilon$ , where  $D$  is a stable distribution. We will reweight this distribution using weights,  $w_t(\cdot)$ , that are now *functions on the whole*  $\mathbb{R}^d$  instead of a fixed dataset. Thus,  $P_t$  now denotes the reweighting of the (corrupted) distribution  $P$  with the weights  $w_t$ . Similarly  $\mu_t, \Sigma_t, \mathbf{B}_t, \mathbf{M}_t$  denote the quantities that pertain to the distribution  $P_t$ . The goal of our algorithm remains essentially the same: obtain  $P_t$  such that  $d_{\text{TV}}(P_t, P) = O(\epsilon)$  and  $\|\Sigma_t\|_2 \leq 1 + O(\delta^2/\epsilon)$ ; [Lemma 2.11](#) would then imply that  $\|\mu_t - \mu\|_2 = O(\delta)$ . Before presenting the pseudocode of [Algorithm 3](#), we identify two problems that arise in generalizing our results from [Section 3](#) and provide an overview of their solutions:

**Calculating Scores** Recall that the only place where  $\mathbf{M}_t$  is used in [Algorithm 1](#) is [Line 18](#), where  $\mathbf{M}_t$  is multiplied with the vectors  $z_{t,j}$ . Let  $z$  be an arbitrary vector. Since  $\mathbf{M}_t = \mathbf{B}^{\log d}$ , in the previous section we were able to compute  $\mathbf{M}_t z$  by iteratively multiplying  $z$  by  $\mathbf{B}_t$ . Since we now do not have access to  $\mathbf{B}_t$ , but only sample access to  $P_t$ , we need a sufficiently fine approximation  $\hat{\mathbf{B}}_t$  of  $\mathbf{B}_t$  (obtained using i.i.d. samples). The natural approach would then be to multiply  $\hat{\mathbf{B}}_t$  with  $z$  iteratively  $\log d$  many times. Even though  $\hat{\mathbf{B}}_t z$  can be computed in a streaming fashion (as outlined in the previous section), it is not possible to compute  $(\hat{\mathbf{B}}_t)^{\log d} z$  without accessing the data  $\log d$  times. To circumvent this issue, we use a fresh sample approximation of  $\mathbf{B}_t$  in every multiplication step. That is, we approximate  $\mathbf{M}_t z$  by  $\widehat{\mathbf{M}}_t z$ , where  $\widehat{\mathbf{M}}_t := \prod_{j=1}^{\log d} \hat{\mathbf{B}}_{t,j}$  and each  $\hat{\mathbf{B}}_{t,j}$  is computed on a different set of samples. This approach crucially leverages the fact that in the contamination model of [Definition 1.2](#), outliers are added in a way that is oblivious to the inliers, and therefore these datasets are statistically identical and independent of each other. We show in [Section 4.3](#) that the resulting  $\widehat{\mathbf{M}}_t$  is a sufficiently accurate approximation of  $\mathbf{M}_t$ . Similarly, we need to modify the Downweighting filter, since its implementation using binary search requires performing checks of the form  $\mathbf{E}_{X \sim P}[w(X)\tilde{\tau}(x)] > 2T$  and calculating the weighted mean exactly is no longer possible. We propose a sample-efficient estimator to approximate that expectation (see [Lemma 4.16](#) in [Section 4.3.3](#)) and run an “approximate” variant of binary search (see [Section 4.2](#)).

**Cover Argument** We now turn to the more technical issue of controlling the size of the JL-sketch, i.e., the number of rows,  $L$ , of the matrix  $\mathbf{U}_t \in \mathbb{R}^{L \times d}$ . For simplicity, assume  $\widehat{\mathbf{M}}_t = \mathbf{M}_t$ , and recall that  $\tilde{\tau}(x)$  is the thresholded version of  $\|\mathbf{U}_t(x - \mu_t)\|_2^2$ , as defined in [Line 21](#) and  $\tau(x)$  is the same score but using  $\mathbf{M}_t$ . The potential-based analysis in [Section 3](#) requires that  $\mathbf{E}_{X \sim P}[w_{t+1}(X)\tau_t(X)]$  is small. However, the stopping condition of the Downweighting filter implies only that  $\mathbf{E}_{X \sim P}[w_{t+1}(X)\tilde{\tau}_t(X)]$  is small. In [Section 3](#), the bound on the former was obtained from the bound on the latter by using that  $\|\mathbf{U}_t(x - \mu_t)\|_2 \approx \|\mathbf{M}_t(x - \mu_t)\|_2$  pointwise in the support of  $P$  ([Claim 3.10](#)).

By the classical JL lemma, the size of the JL sketch,  $L$ , needs to be at most logarithmic in the size of the set  $S$  where we require the pointwise approximation to hold. Thus, in the previous section,  $L$  scaled as  $\log |S| = \log n$ . However, in the streaming model where there is no such dataset, it is far from obvious how the analysis should proceed. A naïve approach would be to require the approximation to hold on a cover  $\tilde{S}$  of the support of  $P_t$ . Since  $|\tilde{S}|$  scales exponentially with  $d$ , the required bound on  $L$  would be  $\log |\tilde{S}| = \Omega(d)$ , which is too large for our purposes. Luckily, we can still find a fixed set  $S_{\text{cover}}$  such that the following holds: (i)  $\log |S_{\text{cover}}| = \text{polylog}(d/\epsilon)$ , and (ii) the expectation of scores over  $\mathcal{U}(S_{\text{cover}})$  approximates the expectation of scores over  $P$ . That is, as far as the expectations of the scores are concerned,  $P$  can be approximated by the uniform distribution over  $S_{\text{cover}}$ . Arguing as before, if  $\|\mathbf{U}_t(x - \mu_t)\| \approx \|\mathbf{M}_t(x - \mu_t)\|$  for each  $x \in S_{\text{cover}}$ , then the downweighting filter also ensures that  $\mathbf{E}_{X \sim P}[w_{t+1}(X)\tilde{\tau}_t(X)]$  is small. Thus,  $S_{\text{cover}}$  can serve as a proxy dataset (used only in the analysis) to ensure that the size of the JL sketch is sufficiently bounded, i.e., that  $\log |S_{\text{cover}}| \leq C \text{polylog}(d/\epsilon)$ .

Establishing the desired upper bound on the cardinality of  $S_{\text{cover}}$  requires a somewhat more sophisticated argument that relies on the VC-dimension of a family of functions corresponding to the weight update rule. This result is stated in [Section 4.2.1](#).

We now present the algorithm more formally. We start by clarifying the notation used.

**Notation regarding [Algorithm 3](#):** The quantities involved in the algorithm and its analysis now are based on the underlying data distribution  $P$  as well as its approximations. We note that  $P_t, \mu_t, \boldsymbol{\Sigma}_t, \mathbf{B}_t, \mathbf{M}_t, \lambda_t$  are functionals of the distribution  $P$  and are primarily used in the analysis. The parameters  $\lambda_t, \widehat{\mathbf{M}}_t$  are approximations for  $\|\mathbf{B}_t\|_2$  and  $\mathbf{M}_t$  respectively that the algorithm forms using samples from  $P_t$ . Regarding score functions,  $g_t(x) = \|\mathbf{M}_t(x - \mu_t)\|_2^2$  is as before. The computations however use only the Johnson-Lindenstrauss versions  $\tilde{g}_t(x) := \frac{1}{L} \sum_{i=1}^L (v_{t,i}^T(x - \hat{\mu}_t))^2$ , which can also be written as  $\|\mathbf{U}_t(x - \hat{\mu}_t)\|_2^2$  in matrix form, by defining  $\mathbf{U}_t$  to have the vectors  $\frac{1}{\sqrt{L}}v_{t,i}$  as its rows. Note that  $\tilde{g}_t(x)$  is defined using  $\hat{\mu}_t$  instead of  $\mu_t$ . Finally, we denote by  $\tau_t(x) = g_t(x)\mathbb{1}\{g_t(x) > C_3\|\mathbf{M}_t\|_F^2\lambda_t/\epsilon\}$  and  $\tilde{\tau}_t(x) = \tilde{g}_t(x)\mathbb{1}\{\tilde{g}_t(x) > C_3\|\mathbf{U}_t\|_F^2\hat{\lambda}_t/\epsilon\}$ .

**Remark 4.4.** Recalling [Lemma 2.12](#), we may again treat the input distribution as a mixture  $P = (1 - \epsilon)G + \epsilon B$ , where  $G$  is a distribution that is  $(C'\epsilon, \delta)$ -stable with respect to  $\mu$ .

As already mentioned, [Algorithm 3](#) uses two levels of approximation: the first level is approximating the true distributional quantities by taking samples, and the second is preserving the latter quantities using the JL sketch. If both of these approximations are sufficiently accurate, the correctness of [Algorithm 3](#) would follow similarly to [Algorithm 1](#). Of course, the challenge is to ensure that these approximations hold over the entire distribution, while controlling the sample and memory complexity of the algorithm. As we show in [Section 4.2.1](#), this can be achieved by restricting our attention to a finite set (cover) of sufficiently large cardinality. Thus, the deterministic conditions that we require now also involve the cover set, which we denote by  $S_{\text{cover}}$ . The reader may think of  $S_{\text{cover}}$  as a fixed set, which will be specified later on ([Lemma 4.9](#)).

---

**Algorithm 3** Robust Mean Estimation In Single-Pass Streaming Model
 

---

```

1: function ROBUSTMEANSTREAMING( $\delta, \epsilon, \tau$  and sample access to  $P$ )
2:   Let  $R$  be such that  $\Pr_{X \sim G}[\|X - \mu\|_2 > R] \leq \epsilon$ .
3:   Let  $P = (1 - \epsilon)G + \epsilon B$ . Without loss of generality, we assume that the points added by the
   adversary are within  $O(R)$  from  $\mu$  in Euclidean norm (see Section 3.1).
4:   Let  $C$  be a sufficiently large constant.
5:   Let  $K = C \log d \log(dR/\epsilon)$ .
6:   Let  $L = C \log^3(dR/\epsilon) \log^2(1/(\tau\epsilon))$ .
7:   Let  $r = CdR^{2+4\log d}$ .
8:   Obtain a naïve estimation  $\hat{\mu}$  of  $\mu$  such that  $\|\hat{\mu} - \mu\|_2 \leq 4R$ .
9:   Let  $w : \mathbb{R}^d \rightarrow [0, 1]$  be the weight function.
10:  Initialize  $w_0(x) \leftarrow \mathbb{1}\{\|x - \hat{\mu}\|_2 \leq 5R\}$  for all  $x \in T$  and  $\ell_1 \leftarrow 0$ .
11:  for  $t \in [K]$  do
12:    Define  $w_t(x) = w_{t-1}(x)(1 - \tilde{\tau}_t(x)/r)^{\ell_t}$ .
13:    Let  $P_t$  be the distribution of  $P$  weighted by  $w_t$ , i.e.,  $P_t(x) = P(x)w_t(x)/\mathbf{E}_{X \sim P}[w_t(X)]$ .
14:    Let  $\mu_t$  be the mean of  $P_t$ .
15:    Let  $\Sigma_t$  be the covariance matrix of  $P_t$ .
16:    Let  $\mathbf{B}_t = (\mathbf{E}_{X \sim P}[w_t(X)])^2 \Sigma_t - \left(1 - C_1 \frac{\delta^2}{\epsilon}\right) \mathbf{I}_d$  and  $\mathbf{M}_t = \mathbf{B}_t^{\log d}$ .
17:    Compute an  $O(\delta)$ -accurate estimator  $\hat{\mu}_t$  of  $\mu_t$  (see Lemma 4.11).
18:    Let  $\tilde{n} = C'' R^2 (\log d)^2 \max\left(d, \frac{\epsilon^2 d}{\delta^4}, \frac{R^2 \epsilon^2}{\delta^2}, \frac{R^2 \epsilon^4}{\delta^6}\right) \log\left(\frac{dK \log d}{\tau}\right)$ .
19:    For  $k \in [\log d]$ , denote by  $\hat{\mathbf{B}}_{t,k}$  the empirical version of  $\mathbf{B}_t$  over  $\tilde{n}$  fresh i.i.d. samples (see
    Section 4.3 for more details).
20:    Define  $\hat{\mathbf{M}}_t := \prod_{k=1}^{\log d} \hat{\mathbf{B}}_{t,k}$  ▷  $\hat{\mathbf{M}}_t$  is not stored in memory.
21:    Let  $\lambda_t = \|\mathbf{B}_t\|_2$  and an approximation  $\hat{\lambda}_t$  such that  $\hat{\lambda}_t/\lambda_t \in [0.8, 1.2]$ .
22:    if  $\hat{\lambda}_t > C_2 \delta^2/\epsilon$  then
23:      for  $j \in [L]$  do
24:         $z_{t,j} \sim \mathcal{U}(\{\pm 1\}^d)$ .
25:         $v_{t,j} \leftarrow \hat{\mathbf{M}}_t z_{t,j}$ . ▷ See Remark 4.12 for efficient implementation.
26:        Store  $v_{t,j}$  in memory.
27:      end for
28:      Denote by  $\mathbf{U}_t$  the matrix with rows  $\frac{1}{\sqrt{L}} v_{t,j}$  for  $j \in [L]$ .
29:      Let  $\tilde{g}_t(x) = \|\mathbf{U}_t(x - \hat{\mu}_t)\|_2^2$  and  $\tilde{\tau}_t(x) = \tilde{g}_t(x) \mathbb{1}\{\tilde{g}_t(x) > C_3 \|\mathbf{U}_t\|_F^2 \hat{\lambda}_t/\epsilon\}$ .
30:       $\ell_{\max} \leftarrow \left(\frac{dR}{\delta^2/\epsilon}\right)^{C \log d}$ .
31:       $\ell_t \leftarrow \text{DownweightingFilter}(P, w_t, \tilde{\tau}_t, R, c\hat{\lambda}_t \|\mathbf{U}_t\|_F^2, \ell_{\max})$ . ▷ Algorithm 4
32:      Store  $\ell_t$  in memory.
33:    end if
34:  end for
35:  return an  $O(\delta)$  approximation  $\hat{\mu}_t$  of the mean  $\mu_t$  of the distribution  $P_t$  (see Lemma 4.11).
36: end function

```

---



**Condition 4.5** (Deterministic Conditions for [Algorithm 3](#)). Let  $S_{\text{cover}}$  denote the cover of [Lemma 4.9](#) for  $\epsilon' = \text{poly}(d, R, 1/\epsilon)^{\log d}$ . Our condition consists of the following event:

1. *Estimator  $\hat{\mu}_t$* : For all  $t \in [K]$ , we have that  $\|\hat{\mu}_t - \mu_t\|_2 \leq \delta/100$ .
2. For every  $t \in [K]$ , if  $\|\mathbf{B}_t\|_2 \geq (C_1/2)\delta^2/\epsilon$  and  $\mathbf{E}_{X \sim P}[w_t(X)] \geq 1 - O(\epsilon)$ , we have that:
  - (a) *Spectral norm of  $\mathbf{B}_t$* :  $\hat{\lambda}_t \in [0.8\lambda_t, 3\lambda_t]$ .
  - (b) *Frobenius norm*:  $\|\mathbf{U}_t\|_F^2 \in [0.8\|\mathbf{M}_t\|_F^2, 1.2\|\mathbf{M}_t\|_F^2]$ .
  - (c) *Scores*:  $\tilde{g}_t(x) \geq 0.2g_t(x) - 0.8(\delta^2/\epsilon^2)\|\mathbf{M}_t\|_F^2$ , for all  $x \in S_{\text{cover}}$ .
3. *Stopping condition*: Let  $T_t := c\hat{\lambda}_t\|\mathbf{U}_t\|_F^2$ . For every  $w : \mathbb{R}^d \rightarrow [0, 1]$ , the algorithm has access to an estimator  $f(w)$  for the quantity  $\mathbf{E}_{X \sim P}[w(X)\tilde{\tau}_t(X)]$ , such that  $\hat{F}(P) > T_t/2$  whenever  $\mathbf{E}_{X \sim P}[w(X)\tilde{\tau}_t(X)] > T_t$ . This estimator is accurate when called  $O(\log(d) \log(dR/\epsilon))$  times in every iteration  $t \in [K]$ .

We note that the [Item 3](#) above is needed to evaluate the stopping condition in the downweighting filter. For every  $t \in [K]$ , the stopping condition is evaluated at most  $O(\log(\ell_{\max}))$  times, with  $\ell_{\max} = O(dR^{2+\log d}/(\hat{\lambda}_t\|\mathbf{U}_t\|_F^2))$  (using [Lemma 3.6](#) with  $r = C_4dR^{2+4\log d}$  and  $T := O(\hat{\lambda}_t\|\mathbf{U}_t\|_F^2)$ ). This means that we require the estimator in [Item 3](#) to be accurate on  $O(K \log(d) \log(dR/\epsilon))$  calls.

## 4.2 Correctness of [Algorithm 3](#)

The analysis in this section is along the same lines as that of [Section 3.4](#). The naïve estimation step of [Line 8](#) is the same as that used in [Algorithm 1](#) (see [Appendix B.4](#)).

Given [Condition 4.5](#), we first show the correctness of [Algorithm 3](#) and leave the task of establishing [Condition 4.5](#) for [Section 4.3](#). The proof of correctness would largely follow by our work done in [Section 3.4](#). There are two adjustments needed in these arguments.

The first concerns [Lemma 3.6](#), since the algorithm cannot perform exact binary search. Instead, it can use the approximate oracle of [Item 3](#) of [Condition 4.5](#), resulting in a multiplicative constant in the final guarantee. For completeness, we prove correctness for this case in [Appendix C.1](#).

**Lemma 4.6.** *In the context of [Algorithm 3](#), if  $(1 - \epsilon)\mathbf{E}_{X \sim G}[w(X)\tilde{\tau}(X)] \leq T$ ,  $\|\tilde{\tau}\|_\infty \leq r$ , and  $\ell_{\max} > r/T$ , then [Algorithm 4](#) modifies the weight function  $w$  to  $w'$  such that (i)  $(1 - \epsilon)\mathbf{E}_{X \sim G}[w(X) - w'(X)] < \epsilon\mathbf{E}_{X \sim B}[w(X) - w'(X)]$ , and (ii) upon termination we have  $\mathbf{E}_{X \sim P}[w'(X)\tilde{\tau}(X)] \leq 54T$ . Furthermore, if the estimator of [Line 3](#) is set to be that of [Lemma 4.16](#), the algorithm terminates after  $O(\log(\ell_{\max}))$  iterations, each of which uses  $O((R^2\epsilon/\delta^2) \log(1/\tau))$  samples, takes  $O(nd)$  time and memory  $O(\log(1/\tau))$ .*

The second adjustment is regarding the analog of [Lemma 3.9](#), i.e.,  $\epsilon\mathbf{E}_{X \sim B}[w_{t+1}\tau_t(X)]$  is small (the bound on  $\epsilon\mathbf{E}_{X \sim B}[w_{t+1}\tilde{\tau}_t(X)]$  follows from the stopping condition as before). Since the support is unbounded, we use an argument based on a fixed cover to show that the downweighting filter succeeds with the JL-sketch of size  $L$ . The statement is given below.

**Lemma 4.7.** *Under the deterministic [Condition 4.5](#) and the context of [Algorithm 3](#), we have that  $\mathbf{E}_{X \sim B}[w_{t+1}(X)\tau_t(X)] \leq 5\mathbf{E}_{X \sim B}[w_{t+1}(X)\tilde{\tau}_t(X)] + c(\lambda_t/\epsilon)\|\mathbf{M}_t\|_F^2$ , where  $c$  is of the form  $C/C_2$  with  $C$  being a sufficiently large constant and  $C_2$  being the constant used in [Algorithm 3](#).*

The next section is dedicated to proving [Lemma 4.7](#). Here we just show that [Lemma 4.7](#) suffices to prove the analog of [Lemma 3.9](#) below.

---

**Algorithm 4** Downweighting Filter using Approximate Oracle

---

```
1: function DOWNWEIGHTINGFILTER( $P, w, \tilde{\tau}, R, T, \ell_{\max}$ )
2:    $r \leftarrow CdR^{2+4\log d}$ .
3:   Denote by  $f(\ell)$  an estimator close to  $\mathbf{E}_{X \sim P}[w(X)(1 - \tilde{\tau}(X)/r)^\ell \tilde{\tau}(X)]$  (see Lemma 4.16 for
   details).
4:    $L \leftarrow \{1, 2, \dots, \ell_{\max}\}$ 
5:   while  $|L| > 2$  do
6:     Let  $\ell$  be the element in the middle of  $L$ .
7:     if  $f(\ell) > 9T$  then
8:       Discard all elements smaller than  $\ell$  from  $L$ .
9:     else
10:      Discard all elements greater than  $\ell$  from  $L$ .
11:    end if
12:  end while
13:  return any  $\ell$  of  $L$  satisfying  $4T \leq f(\ell) \leq 36T$ .
14: end function
```

---

**Lemma 4.8.** *Consider the context of Algorithm 3 and assume that Condition 4.5 holds. Then  $\epsilon \cdot \mathbf{E}_{X \sim B}[w_{t+1}(X)\tau_t(X)] \leq c'\lambda_t \|\mathbf{M}_t\|_F^2$ , for some constant  $c'$  of the form  $C''/C_2$ , where  $C_2$  is the constant used in Line 22 and  $C''$  is some absolute constant.*

*Proof.* Denoting by  $c, c', c'' > 0$  constants that are all multiples of  $1/C_2$ , we have the following:

$$\epsilon \mathbf{E}_{X \sim B}[w_{t+1}(X)\tau_t(x)] \leq 5\epsilon \mathbf{E}_{X \sim B}[w_{t+1}(X)\tilde{\tau}_t(x)] + c\lambda_t \|\mathbf{M}_t\|_F^2 \leq 5c'\lambda_t \|\mathbf{M}_t\|_F^2 + c\lambda_t \|\mathbf{M}_t\|_F^2 \leq c''\lambda_t \|\mathbf{M}_t\|_F^2,$$

where the first inequality uses Lemma 4.7 and the second inequality uses Lemma 4.6.  $\square$

Letting  $\phi_t := \text{tr}(\mathbf{M}_t^2)$  denote the potential function, the above result allows us to follow the same steps as in Section 3.4.2 to prove that  $\phi_{t+1} \leq 0.9999\phi_t$  exactly as in Section 3.4.2. Thus, we get that after  $K$  iterations, we have that  $\|\Sigma_t\|_2 \lesssim \delta^2/\epsilon$ . Under Item 1 of Condition 4.5, we have that the final estimate  $\hat{\mu}_t$  satisfies that  $\|\hat{\mu}_t - \mu\|_2 = O(\delta)$ . This completes the proof of correctness of Algorithm 3.

#### 4.2.1 Proof of Lemma 4.7 via a Cover Argument

To outline the idea of proving Lemma 4.7, recall the proof in the setting of Section 3. There, we just required that  $\tilde{g}_t(x)/g_t(x) \in [0.8, 1.2]$  for all samples  $x$  in our dataset, which can be translated to some relation between  $\tilde{\tau}(x)$  and  $\tau(x)$ . Then, since  $B$  was the empirical distribution on  $\epsilon n$  of these points, the desired condition followed. However, in our case we cannot use pointwise relationships, since the distribution  $B$  may be continuous and the Johnson-Lindenstrauss argument might not work for the entire  $\mathbb{R}^d$  with  $\text{polylog}(d)$  vectors. The idea is first to relate  $\mathbf{E}_{X \sim B}[w_{t+1}(X)\tau_t(X)]$  to a discrete expectation over  $N$  (not too many) points from a fixed set, then use the relationship between  $\tilde{\tau}$  and  $\tau$  for these points, and finally relate that discrete expectation back to  $\mathbf{E}_{X \sim B}[w_{t+1}(X)\tilde{\tau}_t(X)]$ . The existence of a cover of a small size is stated in the following.

**Lemma 4.9.** *Consider the setting of Algorithm 3, where  $B$  is the distribution of outliers supported in a ball of radius  $R$  around  $\mu$ . Let  $r' := (CdR^2 + 1 + C_1\delta^2/\epsilon)^{C \log d}$  for sufficiently large constant  $C$ . Denote by  $\epsilon$  the contamination rate and let an arbitrary  $\epsilon' \in (0, 1)$ . There exists a set  $S_{\text{cover}}$  of*



$N = \frac{1}{\epsilon^3} d^4 K^2 L^2 (dR\epsilon/\delta^2)^{O(\log d)}$  points  $x_1, \dots, x_N$  lying in the ball of radius  $R$  around  $\mu$ , such that for all  $t \in [K]$ , for all choices of the vectors  $z_{t,j}$  of [Line 24](#) of [Algorithm 3](#) it holds

$$\left| \mathbf{E}_{X \sim B} \left[ \frac{1}{r'} w_{t+1}(X) \tilde{\tau}_t(X) \right] - \frac{1}{N} \sum_{i=1}^N \frac{1}{r'} w_{t+1}(x_i) \tilde{\tau}_t(x_i) \right| \leq \epsilon'$$

and

$$\left| \mathbf{E}_{X \sim B} \left[ \frac{1}{r'} w_{t+1}(X) \tau_t(X) \right] - \frac{1}{N} \sum_{i=1}^N \frac{1}{r'} w_{t+1}(x_i) \tau_t(x_i) \right| \leq \epsilon'.$$

We prove this result in [Appendix C.2](#). Here we show how it implies the desired condition, following the proof sketch from the start of this section.

*Proof of [Lemma 4.7](#).* Let  $r' := (CdR^2 + 1 + C_1\delta^2/\epsilon)^{C \log d}$  and  $\epsilon' \in (0, 1)$ . Applying [Lemma 4.9](#), let  $S_{\text{cover}}$  be the corresponding cover of cardinality  $N$ . From the guarantee of approximation of  $\tilde{g}_t$  for every  $x \in S_{\text{cover}}$ , we get the following approximation for  $\tilde{\tau}_t(x)$  for  $x \in S_{\text{cover}}$  (proved in [Appendix C.2](#)).

**Claim 4.10.** *Let  $S$  be the cover of [Lemma 4.9](#) with  $r'$  and  $\epsilon'$  as defined above. Suppose that the deterministic condition [Condition 4.5](#) holds. If  $x \in S_{\text{cover}}$ , then  $\tau_t(x) \leq 5\tilde{\tau}_t(x) + (18C_3 + 12/C_2)(\lambda_t/\epsilon)\|\mathbf{M}_t\|_F^2$ , where  $C_3$  and  $C_2$  are the constants used in [Algorithm 3](#).*

Using [Claim 4.10](#) and [Lemma 4.9](#), we obtain the following series of inequalities:

$$\begin{aligned} \mathbf{E}_{X \sim B} [w_{t+1}(X) \tau_t(X)] &= r' \mathbf{E}_{X \sim B} \left[ \frac{1}{r'} w_{t+1}(X) \tau_t(X) \right] \\ &\leq \epsilon' r' + r' \frac{1}{N} \sum_{i=1}^N \frac{1}{r'} w_{t+1}(x_i) \tau_t(x_i) && \text{(using [Lemma 4.9](#) for } \tau_t) \\ &= \epsilon' r' + \frac{1}{N} \sum_{i=1}^N w_{t+1}(x_i) \tau_t(x_i) \\ &\leq \epsilon' r' + (18C_3 + 12/C_2)(\lambda_t/\epsilon)\|\mathbf{M}_t\|_F^2 + 5 \frac{1}{N} \sum_{i=1}^N w_{t+1}(x_i) \tilde{\tau}_t(x_i) && \text{(using [Claim 4.10](#) and } w_t \leq 1) \\ &\leq 6\epsilon' r' + (18C_3 + 12/C_2)(\lambda_t/\epsilon)\|\mathbf{M}_t\|_F^2 + 5 \mathbf{E}_{X \sim B} [w_{t+1}(X) \tilde{\tau}_t(X)] && \text{(using [Lemma 4.9](#) for } \tilde{\tau}_t) \\ &= 5 \mathbf{E}_{X \sim B} [w_{t+1}(X) \tilde{\tau}_t(X)] + (19C_3 + 12/C_2)(\lambda_t/\epsilon)\|\mathbf{M}_t\|_F^2. && \text{(using the definition of } \epsilon') \end{aligned}$$

For the last line above, we want to choose  $\epsilon'$  such that  $\epsilon' \leq \frac{C_3 \lambda_t}{\epsilon r'} \|\mathbf{M}_t\|_F^2$ . Since  $\|\mathbf{M}_t\|_F^2 \geq (C_2 \delta^2 / \epsilon)^{2 \log d}$  (otherwise the algorithm has already terminated), it suffices to choose an  $\epsilon'$  that satisfies  $\epsilon' \gtrsim \frac{(C_2 \delta^2 / \epsilon)^{2 \log d}}{\epsilon (CdR^2 + 1 + C_1 \delta^2 / \epsilon)^{C \log d}}$ . This gives an upper bound on the cardinality of the set  $S_{\text{cover}}$ , which gives the upper bound on the size of the JL-sketch, i.e.,  $L$ . We provide explicit calculations in [Remark C.3](#).  $\square$

### 4.3 Establishing [Condition 4.5](#)

Throughout this section, we assume sample access to the distribution  $P_t$ . As mentioned earlier, [Algorithm 2](#) can simulate this by drawing a sample  $x$  from  $P$ , calculating  $w_t(x)$  (with poly-logarithmic cost in terms of running time and memory), and rejecting the sample with probability  $1 - w_t(x)$ . With high probability, rejection sampling can increase the sample complexity by at most a constant factor because  $\mathbf{E}[w_t(X)] \geq 1 - O(\epsilon)$  (cf. [Claim 3.13](#)).

### 4.3.1 Item 1

We establish [Item 1](#) in the following, which is proved in [Appendix C.4](#).

**Lemma 4.11.** *In the setting of [Algorithm 3](#), there exist estimators  $\widehat{\mu}_t$  such that, with probability at least  $1 - \tau$ , for all  $t \in [K]$  we have that  $\|\widehat{\mu}_t - \mu_t\|_2 \leq \delta/100$ . Furthermore, each  $\widehat{\mu}_t$  can be computed on a stream of  $n = O\left(\frac{R^2}{\delta^2/\epsilon} \log(K/\tau) + \frac{d(1+\delta^2/\epsilon)}{\delta^2} \log(K/\tau)\right)$  independent samples from  $P_t$ , in time  $O(nd \log(K/\tau))$  and using memory  $O(d \log(K/\tau))$ .*

### 4.3.2 Items 2a to 2c

Given that [Item 1](#) holds, in this section, we show that [Items 2a to 2c](#) of [Condition 4.5](#) hold with high probability if sufficiently many samples from the underlying corrupted distribution  $P$  are drawn. Let the scores  $\widehat{g}_t(x) := \|\widehat{\mathbf{M}}_t(x - \mu_t)\|_2^2$ , where  $\widehat{\mathbf{M}}_t$  is the following sample-based estimator of  $\mathbf{M}_t$ :

1. Draw a batch  $S_0$  of  $\tilde{n}$  samples from  $P$  and let the estimate  $\widehat{W}_t = \mathbf{E}_{X \sim \mathcal{U}(S_0)}[w_t(X)]$ .
2. Let  $P'_t$  be the distribution of the differences  $(X - X')/\sqrt{2}$  for two independent  $X, X' \sim P_t$ .
3. Draw  $\log d$  batches  $S_1, \dots, S_{\log d}$  of  $\tilde{n}$  samples, each from  $P'_t$ .
4. For  $k \in [\log d]$ ,
  - (a) Let  $\widehat{\Sigma}_{t,k} = \frac{1}{\tilde{n}} \sum_{x \in S_k} x x^T$ .
  - (b) Let  $\widehat{\mathbf{B}}_{t,k} = \widehat{W}_t^2 \widehat{\Sigma}_{t,k} - (1 - C_1 \delta^2/\epsilon) \mathbf{I}_d$ .
5. Return  $\widehat{\mathbf{M}}_t = \prod_{k=1}^{\log d} \widehat{\mathbf{B}}_{t,k}$ .

**Remark 4.12.** [Algorithm 3](#) does not need to calculate or store  $\widehat{\mathbf{M}}_t$  because it requires only that we can calculate products of  $\widehat{\mathbf{M}}_t$  with vectors  $z$  as in [Line 25](#). This operation can be implemented in linear runtime and memory. Given the description of the estimator above, it suffices to show how to multiply  $\widehat{\Sigma}_{t,k}$  by a vector  $z$  in linear time and memory. To this end, we observe that  $\widehat{\Sigma}_{t,k} z = \frac{1}{\tilde{n}} \sum_{x \in S_k} x(x^T z)$ , thus by calculating the inner product  $(x^T z)$  first, the result can be found in  $O(nd)$  time in a streaming fashion.

We will show that [Items 2a to 2c](#) of [Condition 4.5](#) follow if  $\|\widehat{\mathbf{M}}_t - \mathbf{M}_t\|_2 \leq 0.01 \min\left(\frac{\delta/\epsilon}{R}, \frac{1}{\sqrt{d}}\right) \|\mathbf{M}_t\|_F$  (cf. [Lemma 4.15](#)) and  $\|\widehat{\mu}_t - \mu_t\|_2 = O(\delta)$  (cf. [Lemma 4.11](#)).

**Lemma 4.13.** *Suppose that the estimators  $\widehat{\mathbf{M}}_t$  in [Algorithm 3](#) are defined by the procedure as in [Lines 1 to 5](#) above. Let  $C$  be a sufficiently large constant and assume that the dimension is  $d = \Omega(1)^2$ . Further assume that  $\|\widehat{\mu}_t - \mu\| \leq 0.01\delta$ . If  $\tilde{n} \geq CR^2(\log d)^2 \max\left(d, \frac{\epsilon^2 d}{\delta^4}, \frac{R^2 \epsilon^2}{\delta^2}, \frac{R^2 \epsilon^4}{\delta^6}\right) \log\left(\frac{Kd \log d}{\tau}\right)$ , then [Items 2a to 2c](#) hold with probability at least  $1 - \tau$ .*

*Proof.* For now, we will assume that for all  $t \in [K]$ ,  $\|\widehat{\mathbf{M}}_t - \mathbf{M}_t\|_2 \leq 0.01 \min\left(\frac{\delta/\epsilon}{R}, \frac{1}{\sqrt{d}}\right) \|\mathbf{M}_t\|_F$  with the claimed sample complexity. This follows from [Lemma 4.15](#) and a union bound. We will prove each of these conditions separately.

<sup>2</sup>This is without loss of generality as we could avoid the JL-sketch when  $d = O(1)$ .

**Proof of Item 2c:** Denote  $T := 0.1\delta/\epsilon$ . Fix an iteration  $t \in [K]$ . We will prove that the conditions hold in the  $t$ -th iteration with probability at least  $1 - \tau/K$  and then a union bound will conclude the proof.

Define  $\widehat{g}_t(x) := \|\mathbf{M}_t(x - \widehat{\mu}_t)\|_2^2$ . Since  $\|\widehat{\mu}_t - \mu_t\|_2 \leq 0.01\delta$ , we have the following relation between  $\widehat{g}_t$  and  $g_t$ :  $\|\mathbf{M}_t(x - \widehat{\mu}_t)\|_2^2 \geq 0.5\|\mathbf{M}_t(x - \mu_t)\|_2^2 - T^2\|\mathbf{M}_t\|_F^2$ , i.e.,  $\widehat{g}_t(x) \geq 0.5g_t(x) - T^2\|\mathbf{M}_t\|_F^2$ . We have that  $\|\widehat{\mathbf{M}}_t - \mathbf{M}_t\|_2 \leq 0.1(T/R)\|\mathbf{M}_t\|_F$  with probability at least  $1 - \tau/K$  (see Lemma 4.15). This means that for any point  $x$  with  $\|x - \widehat{\mu}_t\|_2 \leq 2R$ , we have  $\|\mathbf{M}_t(x - \widehat{\mu}_t)\|_2 \leq \|\widehat{\mathbf{M}}_t(x - \widehat{\mu}_t)\|_2 + 0.2T\|\mathbf{M}_t\|_F$ , which implies that  $\|\mathbf{M}_t(x - \widehat{\mu}_t)\|_2^2 \leq 2\|\widehat{\mathbf{M}}_t(x - \widehat{\mu}_t)\|_2^2 + 0.08T^2\|\mathbf{M}_t\|_F^2$ , or equivalently

$$\|\widehat{\mathbf{M}}_t(x - \widehat{\mu}_t)\|_2^2 \geq 0.5\widehat{g}_t(x) - 0.04T^2\text{tr}(\mathbf{M}_t^2). \quad (9)$$

The final step is taking the Johnson-Lindenstrauss sketch of  $\widehat{\mathbf{M}}_t$ , which gives the matrix  $\mathbf{U}_t$  used in the definition of  $\tilde{g}_t$ . By repeating the proof of Lemma 3.5 with  $\widehat{\mathbf{M}}_t$  in place of  $\mathbf{M}_t$ , we get that if  $L = C \log((|S_{\text{cover}}| + d)K/\tau)$ , then  $\tilde{g}_t(x) \geq 0.8\|\widehat{\mathbf{M}}_t(x - \widehat{\mu}_t)\|_2$  for all the points in the set  $S_{\text{cover}}$  (the cover from Lemma 4.9). The value used for  $L$  in Algorithm 3 satisfies this condition (c.f. Remark C.3). Combining this with Equation (9) and the relation between  $\widehat{g}_t$  and  $g_t$ , we get that  $\tilde{g}_t(x) \geq 0.4\widehat{g}_t(x) - 0.04T^2\|\mathbf{M}_t\|_F^2 \geq 0.2g_t(x) - 0.44T^2\|\mathbf{M}_t\|_F^2$ .

**Proof of Item 2b:** Again, fix a  $t \in [K]$ . We have that  $\|\widehat{\mathbf{M}}_t - \mathbf{M}_t\|_2 \leq \frac{0.01}{\sqrt{d}}\|\mathbf{M}_t\|_F$  with probability  $1 - \tau/K$  using Lemma 4.15. We thus have that  $\|\mathbf{M}_t - \widehat{\mathbf{M}}_t\|_F \leq \sqrt{d}\|\mathbf{M}_t - \widehat{\mathbf{M}}_t\|_2 \leq 0.01\|\mathbf{M}_t\|_F$ , which implies

$$\left| \|\mathbf{M}_t\|_F - \|\widehat{\mathbf{M}}_t\|_F \right| \leq 0.01\|\mathbf{M}_t\|_F. \quad (10)$$

It is easy to check that this is stronger than what we initially wanted. Indeed, squaring Equation (10) and using  $\|\widehat{\mathbf{M}}_t\|_F \leq 1.01\|\mathbf{M}_t\|_F$  gives

$$\|\widehat{\mathbf{M}}_t\|_F^2 \leq 2\|\mathbf{M}_t\|_F\|\widehat{\mathbf{M}}_t\|_F - \|\mathbf{M}_t\|_F^2 + (0.01)^2\|\mathbf{M}_t\|_F^2 < 1.1\|\mathbf{M}_t\|_F^2,$$

which means that  $\|\widehat{\mathbf{M}}_t\|_F^2 - \|\mathbf{M}_t\|_F^2 \leq 0.1\|\mathbf{M}_t\|_F^2$ . For the other bound, Equation (10) implies

$$\begin{aligned} \|\mathbf{M}_t\|_F^2 &\leq 2\|\mathbf{M}_t\|_F\|\widehat{\mathbf{M}}_t\|_F - \|\widehat{\mathbf{M}}_t\|_F^2 + (0.01)^2\|\mathbf{M}_t\|_F^2 \\ &\leq 2\|\widehat{\mathbf{M}}_t\|_F^2 + 2(0.01)\|\mathbf{M}_t\|_F\|\widehat{\mathbf{M}}_t\|_F - \|\widehat{\mathbf{M}}_t\|_F^2 + (0.01)^2\|\mathbf{M}_t\|_F^2 \\ &< 1.1\|\mathbf{M}_t\|_F^2, \end{aligned}$$

which means that  $\|\mathbf{M}_t\|_F^2 - \|\widehat{\mathbf{M}}_t\|_F^2 \leq 0.1\|\mathbf{M}_t\|_F^2$ . Therefore we obtain the following

$$\left| \|\mathbf{M}_t\|_F^2 - \|\widehat{\mathbf{M}}_t\|_F^2 \right| \leq 0.1\|\mathbf{M}_t\|_F^2. \quad (11)$$

Finally, the Johnson-Lindenstrauss step is exactly as described in the proof of Item 2c.

**Proof of Item 2a:** Since we cannot access the same samples twice, the power-iteration algorithm now uses a different dataset in every step. Let the matrix  $\widehat{\mathbf{M}}_t$  as in the beginning of Section 4.3. We have already shown in the previous paragraph that, with probability  $1 - \tau$ , for all  $t \in [K]$ ,  $\widehat{\mathbf{M}}_t$  has Frobenius norm close to that of  $\mathbf{M}_t$  (Equation (11)). For the rest of the proof, we condition on this event. Consider the algorithm that calculates  $v = \widehat{\mathbf{M}}_t w$ , where  $w \sim \mathcal{N}(0, \mathbf{I}_d)$  (this can be done in the streaming model by multiplying with  $\widehat{\mathbf{B}}_{t,k}$  iteratively; moreover this multiplication can be

implemented in time  $O(\tilde{n}d)$ ). We claim that the value  $\widehat{\lambda}_t = \|v\|_2^{1/\log d}$  satisfies the desired relation. First, we note that with probability 0.9 we have that

$$0.8\text{tr}(\widehat{\mathbf{M}}_t^2) \leq \|v\|_2^2 \leq 1.2\text{tr}(\widehat{\mathbf{M}}_t^2). \quad (12)$$

This is because  $\|v\|_2^2 = \text{tr}(w^T \widehat{\mathbf{M}}_t^T \widehat{\mathbf{M}}_t w) = \text{tr}(w w^T \widehat{\mathbf{M}}_t^T \widehat{\mathbf{M}}_t)$  and  $\Pr[0.8d \leq \|w\|_2^2 \leq 1.2d] \leq e^{-cd} \leq 0.1$  (by Gaussian norm concentration, assuming that the dimension is sufficiently large). Equations (11) and (12) imply that  $0.7\text{tr}(\mathbf{M}_t^2) \leq \|v\|_2^2 \leq 1.4\text{tr}(\mathbf{M}_t^2)$ . Furthermore, we have that

$$\|v\|_2^{\frac{1}{\log d}} \leq (1.4\text{tr}(\mathbf{M}_t^2))^{\frac{1}{2\log d}} \leq \left(1.4d\|\mathbf{B}_t\|_2^{2\log d}\right)^{\frac{1}{2\log d}} \leq 2\|\mathbf{B}_t\|_2. \quad (13)$$

Similarly, for the lower bound, we have that

$$\|v\|_2^{\frac{1}{\log d}} \geq (0.7\text{tr}(\mathbf{M}_t^2))^{\frac{1}{2\log d}} \geq (0.7)^{\frac{1}{2\log d}} \|\mathbf{B}_t\|_2 \geq (0.7)\|\mathbf{B}_t\|_2, \quad (14)$$

where in the last inequality we assumed that the dimension is sufficiently large. Putting Equations (11) to (13) together, with probability at least 0.9, we have that  $0.7\|\mathbf{B}_t\|_2 \leq \|v\|_2^{1/\log d} \leq 3\|\mathbf{B}_t\|_2$ . By repeating the procedure  $O(\log(1/\tau))$  times and taking the median, we boost the probability of failure to  $\tau$ . By union bound, choosing  $\tau = \tau/K$  makes the event hold for all iterations  $t \in [K]$ .  $\square$

The remainder of this section is dedicated to showing that  $\|\widehat{\mathbf{M}}_t - \mathbf{M}_t\|_2 \leq \min\left(\frac{\delta/\epsilon}{R}, \frac{0.01}{\sqrt{d}}\right) \|\mathbf{M}_t\|_F$ . We require the following lemma, which we prove in Appendix C.3. We use  $\prod_{i=1}^p \mathbf{B}_i$  to denote the matrix product  $\mathbf{B}_1 \mathbf{B}_2 \cdots \mathbf{B}_p$ .

**Lemma 4.14.** *Let  $\mathbf{A}, \mathbf{B}, \mathbf{B}_1, \dots, \mathbf{B}_p$  be symmetric  $d \times d$  matrices and define  $\mathbf{M} = \mathbf{B}^p$ ,  $\mathbf{M}_S = \prod_{i=1}^p \mathbf{B}_i$ . If  $\|\mathbf{B}_i - \mathbf{B}\|_2 \leq \delta\|\mathbf{B}\|_2$ , then  $\|\mathbf{M}_S - \mathbf{B}^p\|_2 \leq p\delta(1 + \delta)^p \|\mathbf{B}\|_2^p$ .*

We are now ready to prove our main technical result.

**Lemma 4.15.** *Assume that  $\|\mathbf{B}_t\|_2 \geq (C_1/2)\delta^2/\epsilon$  and  $\mathbf{E}_{X \sim P}[w_t(X)] \geq 1 - O(\epsilon)$  hold in the  $t$ -th iteration of Algorithm 3. If  $\widehat{W}$  and every  $\widehat{\mathbf{B}}_{t,k}$  in the product  $\widehat{\mathbf{M}}_t = \prod_{k=1}^{\log d} \widehat{\mathbf{B}}_{t,k}$  is calculated using*

$$\tilde{n} \geq CR^2(\log d)^2 \max\left(d, \frac{\epsilon^2 d}{\delta^4}, \frac{R^2 \epsilon^2}{\delta^2}, \frac{R^2 \epsilon^4}{\delta^6}\right) \log\left(\frac{d \log d}{\tau}\right)$$

*samples, where  $C$  is a sufficiently large constant, we have that*

$$\|\widehat{\mathbf{M}}_t - \mathbf{M}_t\|_2 \leq 0.01 \min\left(\frac{\delta/\epsilon}{R}, \frac{1}{\sqrt{d}}\right) \|\mathbf{M}_t\|_F,$$

*with probability at least  $1 - \tau$ .*

*Proof.* Let  $T := \delta/\epsilon$  and  $p := \log d$  for brevity. Using Lemma 4.14 we have that  $\|\widehat{\mathbf{M}}_t - \mathbf{M}_t\|_2 \leq p\gamma e^{\gamma p} \|\mathbf{M}_t\|_2$ , where  $\gamma > 0$  is such that

$$\|\widehat{\mathbf{B}}_{t,k} - \mathbf{B}_t\|_2 \leq \gamma \|\mathbf{B}_t\|_2 \quad (15)$$

for all  $k \in [p]$ . Therefore, for the lemma to hold, it suffices that  $p\gamma e^{\gamma p} \leq 0.01 \min\left(\frac{T\|\mathbf{M}_t\|_F}{R\|\mathbf{M}_t\|_2}, \frac{1}{\sqrt{d}}\right)$ .

For that, it suffices to choose  $\gamma = \frac{0.01}{3p} \min\left(\frac{1}{\sqrt{d}}, \frac{T\|\mathbf{M}_t\|_F}{R\|\mathbf{M}_t\|_2}\right)$ . At this point, we also assume two things:

First, that the estimate  $\widehat{\Sigma}_{t,k}$  (defined in step 4a) is such that  $\|\widehat{\Sigma}_{t,k} - \Sigma_t\|_2 \leq \epsilon' \|\Sigma_t\|_2$  for some  $\epsilon'$  to be specified later on. Second, that we have an estimate  $\widehat{W}_t$  for  $\mathbf{E}_{X \sim P_t}[w_t(X)]$  such that

$$\widehat{W}_t = \mathbf{E}_{X \sim P_t}[w_t(X)] + \eta, \quad (16)$$

with  $|\eta| \leq \xi$  for some  $\xi \leq 1$  to be decided later. By Hoeffding's inequality, if we compute  $\widehat{W}$  as shown in Step 1, then  $\log(2/\tau)/\xi^2$  samples suffice to guarantee that Equation (16) holds with probability  $1 - \tau/2$ . We now focus on Equation (15). We note that

$$\begin{aligned} \|\widehat{\mathbf{B}}_{t,k} - \mathbf{B}_t\|_2 &= \left\| \left( \mathbf{E}_{X \sim P_t}[w_t(X)] + \eta \right)^2 \widehat{\Sigma}_{t,k} - \mathbf{E}_{X \sim P_t}[w_t(X)]^2 \Sigma_t \right\|_2 \\ &\leq \mathbf{E}_{X \sim P_t}[w_t(X)]^2 \|\widehat{\Sigma}_{t,k} - \Sigma_t\|_2 + (\eta^2 + 2\eta \mathbf{E}_{X \sim P_t}[w_t(X)]) \|\widehat{\Sigma}_{t,k}\|_2 \\ &\leq \|\widehat{\Sigma}_{t,k} - \Sigma_t\|_2 + 3\xi (\|\widehat{\Sigma}_{t,k} - \Sigma_t\|_2 + \|\Sigma_t\|_2) \\ &= (1 + 3\xi) \|\widehat{\Sigma}_{t,k} - \Sigma_t\|_2 + 3\xi \|\Sigma_t\|_2. \end{aligned}$$

By choosing  $\xi = \min(1, \epsilon'/3)$  and  $\epsilon' = \frac{1}{5}\gamma \|\mathbf{B}_t\|_2 / \|\Sigma_t\|_2$ , the above implies that Equation (15) holds. Thus, it suffices to show that  $\|\widehat{\Sigma}_{t,k} - \Sigma_t\|_2 \leq \epsilon' \|\Sigma_t\|_2$  for our choice of  $\epsilon'$ . Note that Fact 2.5 is not directly applicable to the distribution  $P_t$  since it does not have zero mean. This is why we are working with samples of the form  $(X - X')/\sqrt{2}$ . By Fact 2.5 with  $\epsilon'$  set as above and  $\tau = \tau/(2p)$ , we have the following upper bound on the sufficient number of samples:

$$\begin{aligned} \tilde{n} &= C \frac{R^2}{\epsilon'^2 \|\Sigma_t\|_2} \log \left( \frac{2pd}{\tau} \right) \\ &\lesssim \frac{R^2 \|\Sigma_t\|_2}{\gamma^2 \|\mathbf{B}_t\|_2^2} \log \left( \frac{pd}{\tau} \right) \\ &\lesssim \frac{R^2 \epsilon \|\Sigma_t\|_2}{\delta^2 \gamma^2 \|\mathbf{B}_t\|_2} \log \left( \frac{pd}{\tau} \right) \quad (\text{using } \|\mathbf{B}_t\|_2 \geq (C_1/2)\delta^2/\epsilon) \\ &\lesssim \frac{R^2 \epsilon}{\delta^2 \gamma^2} \max \left( 1, \frac{\epsilon}{\delta^2} \right) \log \left( \frac{pd}{\tau} \right) \quad (17) \end{aligned}$$

$$\begin{aligned} &\lesssim \frac{R^2 p^2 \epsilon}{\delta^2} \max \left( d, \frac{R^2 \epsilon^2}{\delta^2} \frac{\|\mathbf{M}_t\|_2^2}{\|\mathbf{M}_t\|_F^2} \right) \max \left( 1, \frac{\epsilon}{\delta^2} \right) \log \left( \frac{pd}{\tau} \right) \\ &\leq \frac{R^2 p^2 \epsilon}{\delta^2} \max \left( d, \frac{R^2 \epsilon^2}{\delta^2} \right) \max \left( 1, \frac{\epsilon}{\delta^2} \right) \log \left( \frac{pd}{\tau} \right) \quad (\text{using } \|\mathbf{M}_t\|_2 \leq \|\mathbf{M}_t\|_F) \\ &\leq \frac{R^2 p^2 \epsilon}{\delta^2} \max \left( d, \frac{\epsilon d}{\delta^2}, \frac{R^2 \epsilon^2}{\delta^2}, \frac{R^2 \epsilon^3}{\delta^4} \right) \log \left( \frac{pd}{\tau} \right), \quad (18) \end{aligned}$$

Equation (17) is derived as follows: First we note that  $\|\Sigma_t\|_2 \leq \frac{\|\mathbf{B}_t\|_2 + 1}{\mathbf{E}_{X \sim P_t}[w_t(X)]^2} \lesssim \|\mathbf{B}_t\|_2 + 1$ , where the last inequality uses our assumption that  $\mathbf{E}_{X \sim P_t}[w_t(X)] \geq 1 - O(\epsilon)$ . We combine this with  $\|\mathbf{B}_t\|_2 \gtrsim \delta^2/\epsilon$  as follows:

$$\frac{\|\Sigma_t\|_2^2}{\|\mathbf{B}_t\|_2^2} \lesssim \frac{(\|\mathbf{B}_t\|_2 + 1)^2}{\|\mathbf{B}_t\|_2^2} \leq 2 + \frac{2}{\|\mathbf{B}_t\|_2^2} \lesssim 1 + \frac{1}{(\delta^2/\epsilon)^2}.$$

Regarding the samples required to achieve Equation (16), a sufficient number is

$$\tilde{n} = C \log \left( \frac{2}{\tau} \right) \frac{1}{\xi^2}$$

$$\begin{aligned}
&\lesssim \log\left(\frac{1}{\tau}\right) \max\left(1, \frac{1}{\epsilon'^2}\right) \\
&\lesssim \log\left(\frac{1}{\tau}\right) \max\left(1, \frac{1}{\gamma^2} \frac{\|\boldsymbol{\Sigma}_t\|_2^2}{\|\mathbf{B}_t\|_2^2}\right) \\
&\lesssim \log\left(\frac{1}{\tau}\right) \max\left(1, \frac{1}{\gamma^2} \max\left(1, \frac{\epsilon^2}{\delta^4}\right)\right) \\
&\lesssim \log\left(\frac{1}{\tau}\right) \max\left(1, p^2 \max\left(d, \frac{R^2 \epsilon^2}{\delta^2}\right) \max\left(1, \frac{\epsilon^2}{\delta^4}\right)\right) \\
&\lesssim \log\left(\frac{1}{\tau}\right) \max\left(p^2 d, \frac{p^2 R^2 \epsilon^2}{\delta^2}, \frac{p^2 d \epsilon^2}{\delta^4}, \frac{p^2 R^2 \epsilon^4}{\delta^6}\right),
\end{aligned}$$

which is smaller compared to the right-hand side of Equation (18).  $\square$

### 4.3.3 Item 3

The following lemma establishes that the estimator of Item 3 is accurate when called once. By using a union bound on the maximum number of times that it can be called, we get the sample complexity requirement of  $n = O((R^2/(\delta^2/\epsilon))\text{polylog}(d, R, \frac{1}{\epsilon}, \frac{1}{\tau}))$ .

**Lemma 4.16.** *Consider the context of Algorithm 3 and denote  $T_t := c\hat{\lambda}_t\|\mathbf{U}_t\|_F^2$ . Given a weight function  $w : \mathbb{R}^d \rightarrow [0, 1]$ , there exists an estimator  $f(w)$  on  $n = O(\frac{R^2}{\delta^2/\epsilon} \log(1/\tau))$  samples such that, if  $\mathbf{E}_{X \sim P}[w(X)\tilde{\tau}_t(X)] > T_t$ , then with probability at least  $1 - \tau$ ,  $f > T_t/2$ . Similarly,  $\mathbf{E}_{X \sim P}[w(X)\tilde{\tau}_t(X)] < T_t$  implies  $f < (3/2)T_t$ . Moreover, the estimator uses  $O(\log(1/\tau))$  memory and runs in  $O(nd)$  time.*

*Proof.* We show the first direction; the other one has a symmetric proof. Suppose  $\mathbf{E}_{X \sim P}[w(X)\tilde{\tau}_t(X)] > c\hat{\lambda}_t\|\mathbf{U}_t\|_F^2$ . It suffices to show that with probability at least 0.9 we have that

$$\frac{1}{n} \sum_{i=1}^N w(X_i)\tilde{\tau}_t(X_i) > \frac{3}{4} \mathbf{E}_{X \sim P}[w(X)\tilde{\tau}_t(X)] - \frac{1}{4} c\hat{\lambda}_t\|\mathbf{U}_t\|_F^2, \quad (19)$$

as we can repeat the procedure  $O(\log(1/\tau))$  times and take the majority vote to boost the probability to  $1 - \tau$ . By Chebyshev's inequality, we have that with probability 0.9 it holds that

$$\frac{1}{n} \sum_{i=1}^n w(X_i)\tilde{\tau}_t(X_i) > \mathbf{E}_{X \sim P}[w(X)\tilde{\tau}_t(X)] - \sqrt{\frac{10 \text{Var}_{X \sim P}(w(X)\tilde{\tau}_t(X))}{n}}.$$

Therefore, it suffices to have  $\sqrt{\frac{10 \text{Var}_{X \sim P}(w(X)\tilde{\tau}_t(X))}{n}} \leq \frac{1}{4} \mathbf{E}_{X \sim P}[w(X)\tilde{\tau}_t(X)] + \frac{1}{4} c\hat{\lambda}_t\|\mathbf{U}_t\|_F^2$ , and thus we need  $n$  to be a sufficiently large multiple of  $\text{Var}_{X \sim P}(w(X)\tilde{\tau}_t(X))/(\mathbf{E}_{X \sim P}[w(X)\tilde{\tau}_t(X)] + c\hat{\lambda}_t\|\mathbf{U}_t\|_F^2)^2$ . For that, it suffices to choose

$$n = \Theta\left(\frac{\text{Var}_{X \sim P}(w(X)\tilde{\tau}_t(X))}{\mathbf{E}_{X \sim P}[w(X)\tilde{\tau}_t(X)]c\hat{\lambda}_t\|\mathbf{U}_t\|_F^2}\right).$$

We now focus on bounding by above the right-hand side. Let  $T'_t := C_3\hat{\lambda}_t\|\mathbf{U}_t\|_F^2$  be the threshold used in the definition of  $\tilde{\tau}_t(x) = \tilde{g}_t(x)\mathbb{1}\{\tilde{g}_t(x) \geq T'_t\}$ . For the variance we have that

$$\text{Var}(w(X)\tilde{\tau}_t(X)) \leq \mathbf{E}_{X \sim P}[(w(X)\tilde{\tau}_t(X))^2]$$

$$\begin{aligned}
&\leq \mathbf{E}_{X \sim P} [w(X) \tilde{\tau}_t^2(X)] \\
&\lesssim R^2 \|\mathbf{U}_t\|_F^2 \mathbf{E}_{X \sim P} [w(X) \tilde{\tau}_t(X)], \tag{20}
\end{aligned}$$

where the last inequality uses that  $\mathbf{E}_{X \sim P_t} [\tilde{\tau}_t^2(X)] = \mathbf{E}_{X \sim P_t} [\tilde{g}_t^2(X) \mathbf{1}\{\tilde{g}_t(X) \geq T'_t\}]$  and bounds from above the one of the two factors of  $\tilde{g}_t$  as follows:

$$\tilde{g}_t(x) = \|\mathbf{U}_t(x - \mu_t)\|_2^2 \leq \|\mathbf{U}_t\|_F^2 R^2, \tag{21}$$

where  $\mathbf{U}_t$  is the matrix used in [Line 29](#) of the algorithm. Using [Equation \(20\)](#), the number of samples that suffice can now be bounded as follows:

$$\frac{\text{Var}_{X \sim P}(\tilde{\tau}_t(X))}{\widehat{\lambda}_t \mathbf{E}_{X \sim P} [w(X) \tilde{\tau}_t(X)] \|\mathbf{U}_t\|_F^2} \lesssim \frac{R^2 \|\mathbf{U}_t\|_F^2}{\widehat{\lambda}_t \|\mathbf{U}_t\|_F^2} \lesssim \frac{R^2}{\delta^2/\epsilon},$$

where we used that  $\widehat{\lambda}_t > C_2 \delta^2/\epsilon$  from [Line 22](#) of our algorithm. □

## 5 Applications: Beyond Robust Mean Estimation

In this section, we develop robust streaming algorithms with near-optimal space complexity for more complex statistical tasks, specifically for robust covariance estimation and robust stochastic optimization. The main idea enabling these applications is that these tasks can be effectively reduced to robust mean estimation.

### 5.1 Robust Covariance Estimation

In this subsection, we study the problem of estimating the covariance matrix  $\Sigma$  of a distribution  $D$ , having access to  $\epsilon$ -corrupted samples from  $D$  in the sense of [Definition 1.2](#). Let  $X \sim D$  and the Kronecker product  $Y = X \otimes X$ . Note that  $\mathbf{E}[Y] = \Sigma^{\flat}$ , where  $\flat$  denotes the flattening operation. Then, using any robust mean estimation algorithm on this  $d^2$ -dimensional distribution, one efficiently compute a vector close to  $\Sigma^{\flat}$  in  $\ell_2$ -norm, which translates to a Frobenius-norm guarantee for  $\Sigma$ . Of course, our mean estimator works as long as the distribution of  $Y$  is stable. If  $\text{Cov}[Y]$  is bounded from above by a multiple of the identity matrix, then  $Y$  is  $(\epsilon, O(\sqrt{\epsilon}))$ -stable with respect to  $\Sigma^{\flat}$ , and thus we get the following as a corollary of [Theorem 4.2](#):

**Theorem 5.1** (Robust Covariance Estimation for Distributions with Bounded Moments). *Let a distribution  $D$  with  $\text{Cov}_{X \sim D}[X \otimes X] \preceq \mathbf{I}_{d^2}$  and denote by  $\Sigma$  its covariance matrix. Let  $d \in \mathbb{Z}_+$ ,  $0 < \tau < 1$  and  $0 < \epsilon < \epsilon_0$  for a sufficiently small constant  $\epsilon_0$ . There exists an algorithm that given  $\epsilon, \tau$  and a set of  $n = (d^4/\epsilon) \text{polylog}(d, 1/\epsilon, 1/\tau)$  samples in the single-pass streaming model of [Definition 1.1](#) from a distribution  $Q$  with  $d_{\text{TV}}(D, Q) \leq \epsilon$ , runs in time  $nd^2 \text{polylog}(d, 1/\epsilon, 1/\tau)$ , uses memory  $d^2 \text{polylog}(d, 1/\epsilon, 1/\tau)$ , and outputs a matrix  $\widehat{\Sigma}$  such that  $\|\widehat{\Sigma} - \Sigma\|_F = O(\sqrt{\epsilon})$ , with probability at least  $1 - \tau$ .*

For the special case when  $D$  is Gaussian we have that the fourth moment tensor of  $D$  is bounded:

**Fact 5.2** (see, e.g., [\[CDGW19b\]](#)). *Let  $X \sim \mathcal{N}(0, \Sigma)$  with  $\Sigma \preceq \mathbf{I}_d$  and  $Y = X \otimes X$ . Then,  $\text{Cov}[Y] \preceq 2\mathbf{I}_{d^2}$ .*



Using the above fact, we have that the guarantees of [Theorem 5.1](#) hold in the Gaussian case, giving an algorithm for  $O(\sqrt{\epsilon})$ -approximation in Frobenius norm. However, the information-theoretic lower bound for covariance estimation of the Gaussian distribution is of the order of  $\epsilon$ . We can plug-in our streaming robust mean estimation algorithm to the covariance estimator given in [\[CDGW19b\]](#), and achieve the nearly-optimal error of  $O(\epsilon \log(1/\epsilon))$ . This algorithm creates a series of estimates  $\widehat{\Sigma}_i$ . At the  $(i + 1)$ -th step, all samples are multiplied by  $\widehat{\Sigma}_i^{-1/2}$  thus, given that  $\widehat{\Sigma}_i$  is a good approximation for  $\Sigma$ , this makes the distribution of the transformed samples closer to  $\mathcal{N}(0, \mathbf{I}_d)$ , which in turn allows us to produce a better approximation  $\widehat{\Sigma}_{i+1}$  of  $\Sigma$ . The resulting guarantees are summarized in the following theorem.

**Theorem 5.3** (Robust Gaussian Covariance Estimation). *Let  $Q$  be a distribution on  $\mathbb{R}^d$  with  $d_{\text{TV}}(Q, \mathcal{N}(0, \Sigma)) \leq \epsilon$  and assume that  $\frac{1}{\kappa} \mathbf{I}_d \preceq \Sigma \preceq \mathbf{I}_d$ , for some  $\kappa > 0$ . There is a single-pass streaming algorithm that uses  $n = (d^4/\epsilon^2) \text{polylog}(d, \kappa, 1/\epsilon, 1/\tau)$  samples from  $Q$ , runs in time  $nd^2 \text{polylog}(d, \kappa, 1/\epsilon, 1/\tau)$ , uses memory  $d^2 \text{polylog}(d, \kappa, 1/\epsilon, 1/\tau)$ , and outputs a matrix  $\widehat{\Sigma}$  such that  $\|\Sigma^{-1/2} \widehat{\Sigma} \Sigma^{-1/2} - \mathbf{I}_d\|_F = O(\epsilon \log(1/\epsilon))$ , with probability at least  $1 - \tau$ .*

The reader is referred to [Appendix E](#) for more details on using [Algorithm 3](#) to obtain [Theorem 5.3](#).

## 5.2 Stochastic Convex Optimization

Here we explore the implications of [Algorithm 3](#) in outlier-robust stochastic convex optimization. This subsection crucially leverages the prior works [\[PSBR20, DKK<sup>+</sup>19a\]](#), which apply robust mean estimation algorithms to perform robust stochastic optimization. In particular, we follow the framework of [\[PSBR20\]](#).

Concretely, we study the following generic optimization problem: Let a parameter space  $\Theta$ , sample space  $\mathcal{Z}$ , and a loss function  $f(\theta; z) : \Theta \times \mathcal{Z} \rightarrow \mathbb{R}^+$ . For an unknown distribution  $D$  over  $\mathcal{Z}$ , the goal is to minimize the associated *risk*  $\bar{f}(\theta) = \mathbf{E}_{z \sim D}[f(\theta; z)]$ , given sample access to the distribution  $D$ . We will occasionally just write  $f(\theta)$  instead of  $f(\theta; z)$  when no confusion arises. This setup is central in machine learning, since it captures a plethora of learning tasks. For example,  $f$  can be a negative log-likelihood function for the learning problem of interest, e.g., square loss for linear regression and logistic loss for logistic regression. In the robust version of the problem, the algorithm has access only to an  $\epsilon$ -corrupted version of  $D$  in the sense of [Definition 1.2](#).

We start by recalling a generic optimization algorithm that works whenever  $\bar{f}$  is  $\tau_\ell$ -strongly convex and  $\tau_u$ -smooth, i.e., for all  $\theta_1, \theta_2 \in \Theta$ , we have that

$$\frac{\tau_\ell}{2} \|\theta_1 - \theta_2\|_2^2 \leq \bar{f}(\theta_1) - \bar{f}(\theta_2) - (\nabla \bar{f}(\theta_2))^T (\theta_1 - \theta_2) \leq \frac{\tau_u}{2} \|\theta_1 - \theta_2\|_2^2.$$

We then give specific applications for robust linear regression and logistic regression.

The work of [\[PSBR20\]](#) provides an analysis of projected gradient descent assuming oracle access to approximations of the gradient:

**Definition 5.4** ( $(\alpha, \beta)$ -gradient estimator). *A function  $g(\theta)$  is an  $(\alpha, \beta)$ -gradient estimator for  $\bar{f}$  if  $\|g(\theta) - \nabla \bar{f}(\theta)\|_2 \leq \alpha \|\theta - \theta^*\|_2 + \beta$ , for every  $\theta \in \Theta$ .*

Denoting by  $\eta$  the step size of gradient descent, define the following parameter:

$$\kappa := \sqrt{1 - \frac{2\eta\tau_\ell\tau_u}{\tau_\ell + \tau_u}} + \eta\alpha. \quad (22)$$



---

**Algorithm 5** Robust Gradient Descent
 

---

- 1: **Input:**  $g(\cdot), \tau$
  - 2: **for**  $t = 0$  **to**  $T - 1$  **do**
  - 3:      $\theta^{t+1} = \arg \min_{\theta \in \Theta} \|\theta^t - \eta g(\theta)\|_2^2$
  - 4: **end for**
- 

**Theorem 5.5** ([PSBR20]). *Let the domains  $\Theta, \mathcal{Z} \subset \mathbb{R}^d$ , a distribution  $D$  over  $\mathcal{Z}$ , and a loss function  $f : \Theta \times \mathcal{Z} \rightarrow \mathbb{R}^+$  such that  $\bar{f}(\theta) := \mathbf{E}_{z \sim D}[f(\theta; z)]$  is  $\tau_\ell$ -strongly convex and  $\tau_u$ -smooth. Let  $\underline{g}$  be an  $(\alpha, \beta)$ -gradient estimator with  $\alpha < \tau_\ell$ . Let  $\kappa$  from Equation (22) and  $\theta^*$  be the minimizer of  $\bar{f}$ . Then Algorithm 5, initialized at  $\theta^0$  with step size  $\eta = 2/(\tau_\ell + \tau_u)$ , after*

$$T = \log_{\frac{1}{\kappa}} \left( \frac{(1 - \kappa) \|\theta^0 - \theta^*\|_2}{\beta} \right) \quad (23)$$

iterations, returns a vector  $\hat{\theta}$  such that

$$\|\hat{\theta} - \theta^*\|_2 \leq \frac{2}{1 - \kappa} \beta. \quad (24)$$

If the distribution of the gradients has bounded covariance, then one can use the low-memory estimator of the previous sections in place of  $g(\cdot)$ . This bound on the covariance will not necessarily be known to the algorithm, thus we first need to strengthen the robust mean estimator so that it is adaptive to that unknown scale. This can be done using Lepski’s method [Lep91, Bir01] (the details are deferred to Appendix D). Having that version of the estimator at hand, we then obtain the following statement (see Appendix E for the proof):

**Corollary 5.6.** *In the setting of Theorem 5.5, suppose that the distribution of gradients satisfies  $\text{Cov}[\nabla f(\theta)] \preceq \sigma^2 \mathbf{I}_d$  with  $\sigma^2 = \alpha^2 \|\theta - \theta^*\|_2^2 + \beta^2$  for all  $\theta \in \Theta$ , where  $\alpha\sqrt{\epsilon} < \tau_\ell$ . Assume that the radius of the domain  $\Theta$ ,  $r := \max_{\theta \in \Theta} \|\theta\|_2$  is finite. There exists a single-pass streaming algorithm that given  $O(T(d^2/\epsilon) \log(1 + \alpha r/\beta) \text{polylog}(d, 1/\epsilon, T/\tau, 1 + \alpha r/\beta))$  samples, runs in time  $Tnd \text{polylog}(d, 1/\epsilon, T/\tau, 1 + \alpha r/\beta)$ , uses memory  $d \text{polylog}(d, 1/\epsilon, T/\tau, 1 + \alpha r/\beta)$ , and returns a vector  $\hat{\theta} \in \mathbb{R}^d$  such that  $\|\hat{\theta} - \theta^*\|_2 = O(\sqrt{\epsilon}\beta/(1 - \kappa))$  with probability at least  $1 - \tau$ .*

We now proceed to more specific applications, where we work out the parameters  $\alpha, \beta$  for some distributions of interest.

### 5.2.1 Linear Regression

For linear regression, we assume the following generative model:

$$Y = X^T \theta^* + Z, \quad (25)$$

where  $\theta^* \in \mathbb{R}^d$  belongs in the ball  $\|\theta^*\|_2 \leq r$ ,  $X \sim D_x$ ,  $Z \sim D_Z$  independently, and  $D_Z$  has zero mean. The loss function that we use in this case is  $f(\theta) = \frac{1}{2}(Y - \theta^T X)^2$ , and the risk function is

$$\bar{f}(\theta) = \mathbf{E}_{(X,Y)} [f(\theta)] = \frac{1}{2}(\theta - \theta^*)^T \mathbf{E}_{X \sim D_x} [X X^T](\theta - \theta^*) + \frac{1}{2} \text{Var}(Z).$$

Letting  $\lambda_{\max}(\mathbf{E}[X X^T])$  and  $\lambda_{\min}(\mathbf{E}[X X^T])$  denote the largest and smallest eigenvalue of  $\mathbf{E}[X X^T]$  respectively, it can be checked that for any  $\tau_\ell \leq \lambda_{\min}(\mathbf{E}[X X^T])$  and  $\tau_u \geq \lambda_{\max}(\mathbf{E}[X X^T])$ ,  $\bar{f}$  is  $\tau_\ell$ -strongly convex and  $\tau_\ell$ -smooth.

Since we want the distribution of gradients to be stable, we impose the following sufficient conditions on the distributions  $D_x$  and  $D_Z$ .

**Assumption 5.7.** The random variables  $X, Z$  are independent and satisfy the following conditions:

1.  $\mathbf{E}_{Z \sim D_Z}[Z] = 0$
2.  $\text{Var}_{Z \sim D_Z}[Z] \preceq \xi^2$
3.  $\gamma \mathbf{I}_d \preceq \mathbf{E}_{X \sim D_x}[XX^T] \preceq \sigma^2 \mathbf{I}_d$ .
4. For some constant  $C > 0$ , for every  $v \in \mathcal{S}^{d-1}$ ,  $\mathbf{E}_{X \sim D_x}[(X^T v)^4] \leq C\sigma^4$ .

As shown below, these assumptions imply that the resulting distribution of the gradients has bounded covariance (and thus is stable with respect to its mean).

**Lemma 5.8** (see, e.g., [DKK<sup>+</sup>19a]). *For  $D_x, D_Z$  satisfying Assumption 5.7, for every  $\theta \in \Theta$ , we have that  $\text{Cov}[\nabla f(\theta)] \preceq (4\sigma^2\xi^2 + 4C\sigma^4\|\theta - \theta^*\|_2^2)\mathbf{I}_d$ .*

Having Lemma 5.8 in hand, Corollary 5.6 gives the following.

**Theorem 5.9** (Robust Linear Regression; full version of Theorem 1.5). *Consider the linear regression model of Equation (25) and suppose that Assumption 5.7 holds. Let  $0 < \epsilon < \epsilon_0$  for a sufficiently small constant  $\epsilon_0$ . Assume that  $C\sigma^2\sqrt{\epsilon} < \gamma/2$ . Let  $\kappa, T$  as in Equations (22) and (23) with  $\tau_\ell = \gamma$ ,  $\tau_u = \sigma^2$ . There is an algorithm that uses  $n = T \cdot (d^2/\epsilon) \log(1 + r\sigma/\xi) \text{polylog}(d, 1/\epsilon, T/\tau, 1 + r\sigma/\xi)$  samples, runs in time  $Tnd \text{polylog}(d, 1/\epsilon, T/\tau, 1 + r\sigma/\xi)$ , uses memory  $d \text{polylog}(d, 1/\epsilon, T/\tau, 1 + r\sigma/\xi)$ , and returns a vector  $\hat{\theta} \in \mathbb{R}^d$  such that  $\|\hat{\theta} - \theta^*\|_2 = O(\sigma\xi\sqrt{\epsilon}/(1 - \kappa))$  with probability at least  $1 - \tau$ .*

*Proof.* In our case, we have that  $\tau_\ell = \gamma$  and  $\tau_u = \sigma^2$ . Given the bound of Lemma 5.8, we use Corollary 5.6 with  $\alpha = 2C\sigma^2$  and  $\beta = 2\sigma\xi$ . The requirement from that corollary that  $\alpha\sqrt{\epsilon} \leq \tau_\ell$  becomes  $C\sigma^2\sqrt{\epsilon} < \gamma/2$ . Moreover,  $\alpha r/\beta = O(r\sigma/\xi)$ .  $\square$

### 5.2.2 Logistic Regression

We consider the joint distribution of  $X \in \mathbb{R}^d, Y \in \{0, 1\}$ , where  $X \sim D_x$  and  $Y$  given  $X$  is Bernoulli random variable:

$$Y|X \sim \text{Bernoulli}(p), \quad \text{with } p = \frac{1}{1 + e^{-x^T \theta^*}}. \quad (26)$$

The loss function we are minimizing in this case is the negative log-likelihood, which eventually can be written as  $f(\theta) = -(\theta^T x)y + \Phi(\theta^T x)$ , where  $\Phi(t) := \log(1 + e^t)$ . Regarding the strong convexity parameters, the Hessian of  $\bar{f}$  can be shown to be

$$\nabla^2 \bar{f}(\theta) = \mathbf{E}_{X \sim D_x} \left[ \frac{e^{\theta^T X}}{(1 + e^{\theta^T X})^2} XX^T \right]. \quad (27)$$

The parameter space  $\Theta$  needs to be bounded in order for the eigenvalues of the Hessian to remain away from zero; we thus use  $\Theta = \{\theta \in \mathbb{R}^d : \|\theta\|_2^2 \leq r\}$  with  $r > 0$  being a universal constant. We also impose the following assumptions on the covariates.

**Assumption 5.10.** We assume the following for the distribution of  $X$ :

1.  $\mathbf{E}[X] = 0$ .
2. (*concentration*) For some constant  $C > 0$ ,  $\mathbf{E}[XX^T] \preceq C^2 \mathbf{I}_d$ .

3. (*anti-concentration*) There exists constant  $c_1 > 0$  and  $c_2 \in (0, 1/2)$  such that for every unit vector  $v$ ,  $\Pr_{X \sim D_x}[(v^T X)^2 > c_1 \|v\|_2^2] \geq c_2$ .

Under these assumptions, we have the following:

**Lemma 5.11** (Lemma 4 in [PSBR20]). *Supposing that Assumption 5.10 holds, for every  $\theta \in \Theta$ , we have that  $\text{Cov}[\nabla f(\theta)] \preceq O(1)\mathbf{I}_d$ .*

The above lemma shows that the distribution of  $\nabla f(\theta)$  is  $(\epsilon, O(\sqrt{\epsilon}))$ -stable, and thus using our robust mean estimation algorithm one can get an  $(\alpha, \beta)$ -gradient estimator with  $\alpha = 0$  and  $\beta = O(\sqrt{\epsilon})$ . This proves the following (see Appendix E for a detailed proof):

**Theorem 5.12** (Robust Logistic Regression; full version of Theorem 1.6). *Consider the logistic regression model of Equation (26) with the domain  $\Theta$  of the unknown regressor being the ball of radius  $r$ , for some universal constant  $r > 0$ , and suppose that Assumption 5.10 holds. Assume that  $0 < \epsilon < \epsilon_0$  for a sufficiently small constant  $\epsilon_0$ . There is a single-pass streaming algorithm that uses  $n = (d^2/\epsilon)$  polylog( $d, 1/\epsilon, 1/\tau$ ) samples, runs in time  $nd$  polylog( $d, 1/\epsilon, 1/\tau$ ), uses memory  $d$  polylog( $d, 1/\epsilon, 1/\tau$ ), and returns a vector  $\hat{\theta} \in \mathbb{R}^d$  such that  $\|\hat{\theta} - \theta^*\|_2 = O(\sqrt{\epsilon})$  with probability at least  $1 - \tau$ .*

### 5.3 Byzantine Adversary and Second-order Optimal Point

We now describe the application of our algorithm to the setting of robust distributed non-convex optimization. As before, for a parameter space  $\Theta \subset \mathbb{R}^d$ , a loss function  $f : \Theta \times \mathcal{Z} \rightarrow \mathbb{R}^+$ , and a distribution  $D$  over  $\mathcal{Z}$ , the goal is to approximately minimize  $\bar{f}(\theta) = \mathbf{E}_{z \sim D}[f(\theta; z)]$ . In this section, we consider the case when  $D$  is a uniform distribution over  $mn$  points  $\{z_{i,j} : i \in [m], j \in [n]\}$  that are distributed over  $m$  machines (workers), with each machine having access to  $n$  samples. Furthermore, we do not impose convexity constraints on  $f$ , and thus would restrict ourselves to finding a second-order stationary point, i.e., a stationary point  $\hat{\theta}$  such that the Hessian on  $\hat{\theta}$  is not too negative in any direction.

We now explain the distributed setup in more detail. There are  $m$  workers who have their own private samples, and a single master machine which is responsible for collecting gradient estimates from the workers and updating the candidate vector iteratively. Concretely, the  $i$ -th worker has  $n$  samples  $\{z_{ij}\}_{j=1}^n$ . The master machine queries all workers with a parameter  $\theta \in \Theta$ , and each  $i$ -th worker responds with  $g_i(\theta)$ , where  $g_i : \mathbb{R}^d \rightarrow \mathbb{R}^d$  is defined as follows: (i) if the  $i$ -th worker is honest, then  $g_i(\theta)$  is the average of the gradients of  $f$  at  $\theta$  of their samples, i.e.,  $g_i(\theta) := (1/n) \sum_{j=1}^n \nabla f(\theta; z_{ij})$ , and (ii) if the  $i$ -th worker is dishonest, then  $g_i(\cdot)$  is an arbitrary function. In our results, we require only that  $(1 - \epsilon)$ -fraction of workers are honest. Recent work of [YCRB19] provided an algorithm that uses a robust mean estimation algorithm on the gradients as a black-box procedure. In particular, the algorithm of [YCRB19] requires only an access to the following oracle:

**Definition 5.13** ( $\Delta$ -inexact gradient). *We call the vector  $v(\theta)$  a  $\Delta$ -inexact gradient of  $\bar{f}$  at the point  $\theta$  if  $\|v(\theta) - \nabla \bar{f}(\theta)\|_2 \leq \Delta$ .*

We assume that each worker machine has access to its own samples throughout the optimization process, and our goal is to reduce the memory requirement of the master machine. Thus, we will use the algorithm from Corollary 4.3 to calculate  $\Delta$ -inexact gradient for the master machine, which requires only an oracle access to the gradient estimates  $\{g_i(\theta) : i \in [m]\}$ .

**Assumption 5.14.** Let  $\mathcal{I} \subseteq [m]$  be the set of honest workers with  $|\mathcal{I}| \geq (1 - \epsilon)m$ .

1. There exists  $\delta$  with  $0 \leq \epsilon \leq \delta \leq \delta_0$ , for some sufficiently small  $\delta_0$ , such that for every  $\theta \in \Theta$ , the set  $\{g_i(\theta) \mid i \in \mathcal{I}\}$  is  $(C\epsilon, \delta)$ -stable with respect to  $\nabla \bar{f}(\theta)$  for a large enough constant  $C$ .
2. We assume that  $\bar{f}$  is  $L$ -smooth and  $\rho$ -Hessian Lipschitz on  $\Theta$ , i.e., for every  $\theta_1, \theta_2 \in \Theta$  we have that  $\|\nabla \bar{f}(\theta_1) - \nabla \bar{f}(\theta_2)\|_2 \leq L\|\theta_1 - \theta_2\|_2$  and  $\|\nabla^2 \bar{f}(\theta_1) - \nabla^2 \bar{f}(\theta_2)\|_2 \leq \rho\|\theta_1 - \theta_2\|_2$ .

We note that if the samples of honest workers are sampled i.i.d. from a distribution  $P$ , then the set  $\{g_i(\theta) : i \in \mathcal{I}\}$  for a fixed  $\theta \in \Theta$  will be stable with respect to  $\nabla \bar{f}(\theta)$  with high probability, provided that the distribution of  $\nabla f(\theta; Z)$  satisfies mild concentration under  $Z \sim P$  and  $m$  is sufficiently large. Using a standard cover argument with the smoothness properties of  $f$ , this can be extended to all  $\theta \in \Theta$ . We thus obtain the following theorem, under [Assumption 5.14](#).

**Theorem 5.15.** *Suppose that [Assumption 5.14](#) holds. Let  $m$  denote the number of workers. Assume  $0 < \tau < 1$ ,  $\Delta := C'\delta < 1$ , for  $C'$  a sufficiently large constant and define*

$$Q := 2 \log \left( \frac{\rho(\bar{f}(\theta_0) - \inf_{\theta \in \mathbb{R}^d} \bar{f}(\theta))}{48L\tau(\Delta^{6/5}d^{3/5} + \Delta^{7/5}d^{7/10})} \right), \quad T_{th} := \frac{L}{384(\rho^{1/2} + L(\Delta^{2/5}d^{1/5} + \Delta^{3/5}d^{3/10}))}.$$

*There is an algorithm where the master, if initialized at  $\theta_0$ , does  $T = \frac{2(\bar{f}(\theta_0) - \inf_{\theta \in \mathbb{R}^d} \bar{f}(\theta))}{3\Delta^2} Q T_{th}$  iterations, each running in  $md \text{polylog}(d, 1/\epsilon, T/\tau)$  time, uses  $d \text{polylog}(d, 1/\epsilon, T/\tau)$  memory, and outputs a vector  $\hat{\theta}$  such that, with probability  $1 - \tau$ ,  $\|\nabla \bar{f}(\hat{\theta})\|_2 \leq 4\Delta$  and  $\lambda_{\min}(\nabla^2 \bar{f}(\hat{\theta})) \geq -\Delta^{2/5}d^{1/5}$ .*

## 6 Discussion

In this work, we gave the first efficient streaming algorithm with near-optimal space complexity for outlier-robust high-dimensional mean estimation. As an application, we also obtained low-space streaming algorithms for a range of other robust estimation tasks. Our work is a first step towards understanding the space complexity of high-dimensional robust statistics in the streaming setting.

Our work suggests a number of open problems. First, the sample complexity of our mean estimation algorithm is  $\tilde{O}(d^2/\epsilon^2)$ , while the information-theoretic optimum (without space constraints!) is  $\tilde{O}(d/\epsilon^2)$ . What is the *optimal* sample-space tradeoff? A similar question can be asked for the broader tasks of covariance estimation and stochastic optimization. A more general goal is to characterize the tradeoff between space complexity, number of passes, and sample size/runtime for other robust high-dimensional statistics tasks, e.g., clustering and learning of mixture models.

Finally, another research direction concerns the considered contamination model. Throughout this paper, we focused on the TV-contamination model. One can consider an even stronger contamination model with an adaptive adversary, where the outliers can be completely arbitrary (i.e., not follow any distribution), and the adversary can additionally control the order in which the points are presented in the stream. Is it possible to obtain  $\tilde{O}_\epsilon(d)$ -space single-pass streaming algorithms for robust mean estimation in the presence of such an adversary? While our algorithms can be shown to work in this model with a poly-logarithmic number of passes, it is not clear whether a single-pass algorithm with sub-quadratic space complexity exists in this setting.

## References

- [AB99] M. Anthony and P. L. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, first edition, November 1999.
- [Ach03] D. Achlioptas. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *Journal of computer and System Sciences*, 66(4):671–687, 2003.
- [BCN18] L. Bottou, F. E. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018.
- [BDH<sup>+</sup>20] A. Bakshi, I. Diakonikolas, S. B. Hopkins, D. Kane, S. Karmalkar, and P. K. Kothari. Outlier-robust clustering of gaussians and other non-spherical mixtures. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020*, pages 149–159. IEEE, 2020.
- [BDJ<sup>+</sup>20] A. Bakshi, I. Diakonikolas, H. Jia, D. M. Kane, P. K. Kothari, and S. S. Vempala. Robustly learning mixtures of k arbitrary gaussians. *CoRR*, abs/2012.02119, 2020.
- [BDLS17] S. Balakrishnan, S. S. Du, J. Li, and A. Singh. Computationally efficient robust sparse estimation in high dimensions. In *Proceedings of the 30th Conference on Learning Theory, COLT 2017*, pages 169–212, 2017.
- [BGVKS20] J. Banks, J. Garza-Vargas, A. Kulkarni, and N. Srivastava. Pseudospectral shattering, the sign function, and diagonalization in nearly matrix multiplication time. In *FOCS 2020*, pages 529–540, 2020.
- [BHK20] A. Blum, J. Hopcroft, and R. Kannan. *Foundations of Data Science*. Cambridge University Press, first edition, January 2020.
- [Bir01] L. Birgé. An alternative point of view on lepski’s method. *Lecture Notes-Monograph Series*, pages 113–133, 2001.
- [BNJT10] M. Barreno, B. Nelson, A. D. Joseph, and J. D. Tygar. The security of machine learning. *Machine Learning*, 81(2):121–148, 2010.
- [BNL12] B. Biggio, B. Nelson, and P. Laskov. Poisoning attacks against support vector machines. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, 2012.
- [Bot10] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *COMPSTAT*, 2010.
- [CDGW19a] Y. Cheng, I. Diakonikolas, R. Ge, and D. P. Woodruff. Faster algorithms for high-dimensional robust covariance estimation. In *Conference on Learning Theory, COLT 2019*, pages 727–757, 2019.
- [CDGW19b] Y. Cheng, I. Diakonikolas, R. Ge, and D. P. Woodruff. Faster algorithms for high-dimensional robust covariance estimation. In *COLT*, pages 727–757, 2019.
- [CDK<sup>+</sup>21] Y. Cheng, I. Diakonikolas, D. M. Kane, R. Ge, S. Gupta, and M. Soltanolkotabi. Outlier-robust sparse estimation via non-convex optimization. *CoRR*, abs/2109.11515, 2021.

- [CDKS18] Y. Cheng, I. Diakonikolas, D. Kane, and A. Stewart. Robust learning of fixed-structure bayesian networks. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018*, pages 10304–10316, 2018. Full version available at <https://arxiv.org/abs/1606.07384>.
- [DHL19] Y. Dong, S. B. Hopkins, and J. Li. Quantum entropy scoring for fast robust mean estimation and improved outlier detection. *NeurIPS*, 32:6067–6077, 2019.
- [DK19] I. Diakonikolas and D. M. Kane. Recent advances in algorithmic high-dimensional robust statistics. *CoRR*, abs/1911.05911, 2019.
- [DKK<sup>+</sup>16] I. Diakonikolas, G. Kamath, D. M. Kane, J. Li, A. Moitra, and A. Stewart. Robust estimators in high dimensions without the computational intractability. In *FOCS*, pages 655–664, 2016.
- [DKK<sup>+</sup>17] I. Diakonikolas, G. Kamath, D. M. Kane, J. Li, A. Moitra, and A. Stewart. Being robust (in high dimensions) can be practical. In *ICML 2017*, pages 999–1008, 2017.
- [DKK<sup>+</sup>19a] I. Diakonikolas, G. Kamath, D. Kane, J. Li, J. Steinhardt, and A. Stewart. Sever: A robust meta-algorithm for stochastic optimization. In *ICML 2019*, pages 1596–1606, 2019.
- [DKK<sup>+</sup>19b] I. Diakonikolas, S. Karmalkar, D. Kane, E. Price, and A. Stewart. Outlier-robust high-dimensional sparse estimation via iterative filtering. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2019*, 2019.
- [DKK<sup>+</sup>20] I. Diakonikolas, D. M. Kane, D. Kongsgaard, J., and K. Tian. List-decodable mean estimation in nearly-pca time. *CoRR*, abs/2011.09973, 2020.
- [DKK<sup>+</sup>21a] I. Diakonikolas, G. Kamath, D. M. Kane, J. Li, A. Moitra, and A. Stewart. Robustness meets algorithms. *Commun. ACM*, 64(5):107–115, 2021.
- [DKK<sup>+</sup>21b] I. Diakonikolas, D. M. Kane, D. Kongsgaard, J. Li, and K. Tian. Clustering mixture models in almost-linear time via list-decodable mean estimation. *CoRR*, abs/2106.08537, 2021.
- [DKP20] I. Diakonikolas, D. M. Kane, and A. Pensia. Outlier robust mean estimation with subgaussian rates via stability. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, 2020.
- [DKS18] I. Diakonikolas, D. M. Kane, and A. Stewart. List-decodable robust mean estimation and learning mixtures of spherical gaussians. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018*, pages 1047–1060, 2018. Full version available at <https://arxiv.org/abs/1711.07211>.
- [DKS19] I. Diakonikolas, W. Kong, and A. Stewart. Efficient algorithms and lower bounds for robust linear regression. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019*, pages 2745–2754, 2019.

- [DKSS21] I. Diakonikolas, D. M. Kane, A. Stewart, and Y. Sun. Outlier-robust learning of ising models under dobrushin’s condition. In *Conference on Learning Theory, COLT 2021*, volume 134 of *Proceedings of Machine Learning Research*, pages 1645–1682. PMLR, 2021.
- [DKTZ20] I. Diakonikolas, V. Kontonis, C. Tzamos, and N. Zarifis. Non-convex SGD learns halfspaces with adversarial label noise. In *NeurIPS 2020*, 2020.
- [GJ95] P. W. Goldberg and M. R. Jerrum. Bounding the vapnik-chervonenkis dimension of concept classes parameterized by real numbers. *Machine Learning*, 18(2):131–148, 1995.
- [HL18] S. B. Hopkins and J. Li. Mixture models, robustness, and sum of squares proofs. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018*, pages 1021–1034, 2018.
- [Hub64] P. J. Huber. Robust estimation of a location parameter. *Ann. Math. Statist.*, 35(1):73–101, 03 1964.
- [KKM18] A. R. Klivans, P. K. Kothari, and R. Meka. Efficient algorithms for outlier-robust regression. In *Conference On Learning Theory, COLT 2018*, pages 1420–1430, 2018.
- [KSS18] P. K. Kothari, J. Steinhardt, and D. Steurer. Robust moment estimation and improved clustering via sum of squares. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018*, pages 1035–1046, 2018.
- [LAT<sup>+</sup>08] J.Z. Li, D.M. Absher, H. Tang, A.M. Southwick, A.M. Casto, S. Ramachandran, H.M. Cann, G.S. Barsh, M. Feldman, L.L. Cavalli-Sforza, and R.M. Myers. Worldwide human relationships inferred from genome-wide patterns of variation. *Science*, 319:1100–1104, 2008.
- [Lep91] O. V. Lepskii. On a problem of adaptive estimation in gaussian white noise. *Theory of Probability & Its Applications*, 35(3):454–466, 1991.
- [LM20] A. Liu and A. Moitra. Settling the robust learnability of mixtures of gaussians. *CoRR*, abs/2011.03622, 2020.
- [LRV16] K. A. Lai, A. B. Rao, and S. Vempala. Agnostic estimation of mean and covariance. In *FOCS*, 2016.
- [PF20] S. Pesme and N. Flammarion. Online robust regression via SGD on the l1 loss. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, 2020.
- [PJL20] A. Pensia, V. Jog, and P. Loh. Robust regression with covariate filtering: Heavy tails and adversarial contamination. *arXiv preprint arXiv:2009.12976*, 2020.
- [PLJD10] P. Paschou, J. Lewis, A. Javed, and P. Drineas. Ancestry informative markers for fine-scale individual assignment to worldwide populations. *Journal of Medical Genetics*, 47:835–847, 2010.
- [PSBR20] A. Prasad, A. S. Suggala, S. Balakrishnan, and P. Ravikumar. Robust estimation via robust gradient estimation. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 82(3):601–627, 2020.

- [RPW<sup>+</sup>02] N. Rosenberg, J. Pritchard, J. Weber, H. Cann, K. Kidd, L.A. Zhivotovsky, and M.W. Feldman. Genetic structure of human populations. *Science*, 298:2381–2385, 2002.
- [SCV18] J. Steinhardt, M. Charikar, and G. Valiant. Resilience: A criterion for learning in the presence of arbitrary outliers. In *ITCS 2018*, pages 45:1–45:21, 2018.
- [SKL17] J. Steinhardt, P. W. Koh, and P. S. Liang. Certified defenses for data poisoning attacks. In *NeurIPS*, pages 3520–3532, 2017.
- [SWS20] V. Shah, X. Wu, and S. Sanghavi. Choosing the sample with lowest loss makes SGD robust. In *AISTATS 2020*, volume 108 of *Proceedings of Machine Learning Research*, pages 2120–2130, 2020.
- [TLM18] B. Tran, J. Li, and A. Madry. Spectral signatures in backdoor attacks. In *NeurIPS 2018*, pages 8011–8021, 2018.
- [TPBR21] C. Tsai, A. Prasad, S. Balakrishnan, and P. Ravikumar. Heavy-tailed streaming statistical estimation. *CoRR*, abs/2108.11483, 2021.
- [Tuk60] J. W. Tukey. A survey of sampling from contaminated distributions. *Contributions to probability and statistics*, 2:448–485, 1960.
- [Ver10] R. Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.
- [YCRB19] D. Yin, Y. Chen, K. Ramchandran, and P. Bartlett. Defending against saddle point attack in byzantine-robust distributed learning. In *ICML*, pages 7074–7084, 2019.



## A Omitted Proofs from Section 2: Technical Details Regarding Stability

**Lemma 2.11** (Certificate Lemma). *Let  $G$  be an  $(\epsilon, \delta)$ -stable distribution with respect to  $\mu \in \mathbb{R}^d$ , for some  $0 < \epsilon < 1/3$  and  $\delta \geq \epsilon$ . Let  $P$  be a distribution with  $d_{\text{TV}}(P, G) \leq \epsilon$ . Denoting by  $\mu_P, \Sigma_P$  the mean and covariance of  $P$ , if  $\lambda_{\max}(\Sigma_P) \leq 1 + \lambda$ , for some  $\lambda \geq 0$ , then  $\|\mu_P - \mu\|_2 = O(\delta + \sqrt{\epsilon\lambda})$ .*

*Proof.* Let  $d_{\text{TV}}(P, G) = \alpha$ . By [Fact 2.4](#) we can write  $P = (1 - \alpha)G_0 + \alpha B$ . We may assume without loss of generality  $\alpha = \epsilon$ , since we can always treat a part of the inliers as outliers. Denoting by  $\mu_P, \Sigma_P$  the mean and covariance of  $P$ , and using  $\mu_{G_0}, \mu_B, \Sigma_{G_0}, \Sigma_B$  for the corresponding quantities of the other two distributions, we have that

$$\Sigma_P = (1 - \epsilon)\Sigma_{G_0} + \epsilon\Sigma_B + \epsilon(1 - \epsilon)(\mu_{G_0} - \mu_B)(\mu_{G_0} - \mu_B)^T.$$

Letting  $v$  be the unit vector in the direction of  $\mu_{G_0} - \mu_B$ , we have that

$$1 + \lambda \geq v^T \Sigma_P v = (1 - \epsilon)v^T \Sigma_{G_0} v + \epsilon v^T \Sigma_B v + \epsilon(1 - \epsilon)(v^T(\mu_{G_0} - \mu_B))^2. \quad (28)$$

The second term of the left-hand side is nonzero and the third one is just  $\epsilon(1 - \epsilon) \|\mu_{G_0} - \mu_B\|_2^2$ . We now focus on the first term, which by adding and subtracting  $\mu$  (the vector realizing the definition of stability for  $G$ ) can be written as

$$(1 - \epsilon) \mathbf{E}_{X \sim G_0} [(v^T(X - \mu_{G_0}))^2] = (1 - \epsilon) \left( \mathbf{E}_{X \sim G_0} [(v^T(X - \mu))^2] - (v^T(\mu - \mu_{G_0}))^2 \right). \quad (29)$$

We note that in the decomposition of [Fact 2.4](#), we can write  $G_0(x) = w_0(x)G(x)$  with

$$w_0(x) = \frac{1}{1 - \epsilon} \begin{cases} P(x)/G(x), & \text{if } G(x) > P(x) \\ 1, & \text{otherwise.} \end{cases}$$

Letting  $h(x) := (1 - \epsilon)w_0(x)$  we have that  $h(x) \leq 1$  for all  $x$  and  $\mathbf{E}_{X \sim G}[h(X)] = 1 - \epsilon$ , thus  $G_0(x) = h(x)G(x)/(\int h(x)G(x)dx) =: G_h(x)$ . Returning to [Equation \(29\)](#), this means that

$$\mathbf{E}_{X \sim G_0} [(v^T(X - \mu))^2] = \mathbf{E}_{X \sim G_h} [(v^T(X - \mu))^2] = v^T \bar{\Sigma}_{h,G} v \geq 1 - \frac{\delta^2}{\epsilon}, \quad (30)$$

by applying stability. Similarly, the other term in [Equation \(29\)](#) is  $(v^T(\mu - \mu_{G_0}))^2 \leq \delta^2$ . Putting everything together, [Equation \(28\)](#) becomes

$$\begin{aligned} 1 + \lambda &\geq (1 - \epsilon)(1 - \delta^2/\epsilon - \delta^2) + \epsilon(1 - \epsilon)\|\mu_{G_0} - \mu_B\|_2^2 \\ &\geq 1 - 3\delta^2/\epsilon + (\epsilon/2)\|\mu_{G_0} - \mu_B\|_2^2, \end{aligned}$$

which yields  $\|\mu_{G_0} - \mu_B\|_2 \lesssim \sqrt{\lambda/\epsilon} + \delta/\epsilon$ . Then, writing  $\mu_P = (1 - \epsilon)\mu_{G_0} + \epsilon\mu_B$  and using stability follows that  $\|\mu_P - \mu\|_2 \lesssim \delta + \sqrt{\lambda\epsilon}$ .  $\square$

**Lemma 2.12.** *For any  $0 < \epsilon < 1/2$  and  $\delta \geq \epsilon$ , if a distribution  $G$  is  $(2\epsilon, \delta)$ -stable with respect to  $\mu \in \mathbb{R}^d$ , and  $P$  is an  $\epsilon$ -corrupted version of  $G$  in total variation distance, there exist distributions  $G_0$  and  $B$  such that  $P = (1 - \epsilon)G_0 + \epsilon B$  and  $G_0$  is  $(\epsilon, \delta)$ -stable with respect to  $\mu$ .*

*Proof.* By [Fact 2.4](#), we have the decomposition  $P = (1-\epsilon)G_0 + \epsilon B$ , where  $G_0(x) = \min\{G(x), P(x)\}/(1-\epsilon)$ . We can write  $G_0(x) = w_0(x)G(x)$ , where

$$w_0(x) = \frac{1}{1-\epsilon} \begin{cases} P(x)/G(x), & \text{if } G(x) > P(x) \\ 1, & \text{otherwise.} \end{cases}$$

To see why the final claim is true, we consider a weight function  $w : \mathbb{R}^d \rightarrow [0, 1]$  such that  $\mathbf{E}_{X \sim G_0}[w(X)] \geq 1 - \epsilon$  and examine the adjusted distribution  $G_{0w}$ . We have that

$$G_{0w}(x) = \frac{w(x)G_0(x)}{\int_{\mathbb{R}^d} w(x)G_0(x)dx} = \frac{(1-\epsilon)w(x)w_0(x)G(x)}{\int_{\mathbb{R}^d} (1-\epsilon)w(x)w_0(x)G(x)dx} = \frac{h(x)G(x)}{\mathbf{E}_{X \sim G}[h(X)]} = G_h(x),$$

where we let  $h(x) := (1-\epsilon)w(x)w_0(x)$ . We have that  $h(x) \leq 1$  point-wise and  $\int_{\mathbb{R}^d} h(x)G(x)dx = (1-\epsilon)\mathbf{E}_{X \sim G_0}[w(X)] \geq (1-\epsilon)^2 \geq 1-2\epsilon$ . Recalling that  $G$  is  $(2\epsilon, \delta)$ -stable, the conclusion follows.  $\square$

**Lemma 2.13.** *Fix  $0 < \epsilon < 1/2$  and  $\delta \geq \epsilon$ . Let  $w : \mathbb{R}^d \rightarrow [0, 1]$  such that  $\mathbf{E}_{X \sim G}[w(X)] \geq 1 - \epsilon$  and let  $G$  be an  $(\epsilon, \delta)$ -stable distribution with respect to  $\mu \in \mathbb{R}^d$ . For any matrix  $\mathbf{U} \in \mathbb{R}^{d \times d}$  and any vector  $b \in \mathbb{R}^d$ , we have that*

$$\mathbf{E}_{X \sim G_w} [\|\mathbf{U}(X - b)\|_2^2] = \|\mathbf{U}\|_F^2(1 \pm \delta^2/\epsilon) + \|\mathbf{U}(\mu - b)\|_2^2 \pm 2\delta \|\mathbf{U}\|_F^2 \|\mu - b\|_2.$$

*Proof.* We can write

$$\begin{aligned} \mathbf{E}_{X \sim G_w} [\|\mathbf{U}(X - b)\|_2^2] &= \mathbf{E}_{X \sim G_w} [\|\mathbf{U}(X - \mu)\|_2^2 + \|\mathbf{U}(\mu - b)\|_2^2 + 2(X - \mu)^T \mathbf{U}^T \mathbf{U}(\mu - b)] \\ &= \mathbf{E}_{X \sim G_w} [\|\mathbf{U}(X - \mu)\|_2^2] + \|\mathbf{U}(\mu - b)\|_2^2 + 2(\mu_{w,G} - \mu)^T \mathbf{U}^T \mathbf{U}(\mu - b). \end{aligned} \quad (31)$$

We now focus on the first term. Let the spectral decomposition  $\mathbf{U}^T \mathbf{U} = \text{tr}(\mathbf{U}^T \mathbf{U}) \sum_{i=1}^d \alpha_i v_i v_i^T$ , where  $\sum_{i=1}^d \alpha_i = 1$  and  $\alpha_i \geq 0$ . We have that

$$\begin{aligned} \mathbf{E}_{X \sim G_w} [\|\mathbf{U}(X - \mu)\|_2^2] &= \text{tr} \left( \mathbf{U}^T \mathbf{U} \mathbf{E}_{X \sim G_w} [(X - \mu)(X - \mu)^T] \right) = \text{tr}(\mathbf{U}^T \mathbf{U}) \sum_{i=1}^d \alpha_i \text{tr}(v_i v_i^T \bar{\Sigma}_{w,G}) \\ &= \text{tr}(\mathbf{U}^T \mathbf{U}) \sum_{i=1}^d \alpha_i v_i^T \bar{\Sigma}_{w,G} v_i = \text{tr}(\mathbf{U}^T \mathbf{U})(1 \pm \delta^2/\epsilon) = \|\mathbf{U}\|_F^2(1 \pm \delta^2/\epsilon), \end{aligned}$$

where the second from the end relation is due to stability. Regarding the last term of [Equation \(31\)](#), we have that

$$\begin{aligned} |(\mu_{w,G} - \mu)^T \mathbf{U}^T \mathbf{U}(\mu - b)| &= |\text{tr}(\mathbf{U}^T \mathbf{U}(\mu - b)(\mu_{w,G} - \mu)^T)| \leq \text{tr}(\mathbf{U}^T \mathbf{U}) \|(\mu - b)(\mu_{w,G} - \mu)^T\|_2 \\ &= \|\mathbf{U}\|_F^2 \|\mu - b\|_2 \|\mu_{w,G} - \mu\|_2 \leq \|\mathbf{U}\|_F^2 \delta \|\mu - b\|_2, \end{aligned}$$

where the last inequality uses stability condition for the mean.  $\square$

**Corollary 2.14.** *Fix  $0 < \epsilon < 1/2$  and  $\delta \geq \epsilon$ . Let  $G$  be an  $(\epsilon, \delta)$ -stable distribution with respect to  $\mu \in \mathbb{R}^d$ . Let a matrix  $\mathbf{U} \in \mathbb{R}^{d \times d}$  and a function  $w : \mathbb{R}^d \rightarrow [0, 1]$  with  $\mathbf{E}_{X \sim G}[w(X)] \geq 1 - \epsilon$ . For the function  $\tilde{g}(x) = \|\mathbf{U}(x - b)\|_2^2$ , we have that*

$$(1 - \epsilon) \|\mathbf{U}\|_F^2 (1 - \delta^2/\epsilon - 2\delta \|b - \mu\|_2) \leq \mathbf{E}_{X \sim G} [w(X) \tilde{g}(X)] \leq \|\mathbf{U}\|_F^2 (1 + \delta^2/\epsilon + \|b - \mu\|_2^2 + 2\delta \|b - \mu\|_2).$$

*Proof.* Beginning with the upper bound, we have the following inequalities:

$$\begin{aligned}
\mathbf{E}_{X \sim G} [w(X)\tilde{g}(X)] &= \mathbf{E}_{X \sim G} [w(X)] \mathbf{E}_{X \sim G_w} [\tilde{g}(X)] \\
&\leq \mathbf{E}_{X \sim G_w} [\tilde{g}(X)] && \text{(Using } \tilde{g}(x) \geq 0 \text{ and } w(x) \leq 1) \\
&\leq \|\mathbf{U}\|_F^2(1 + \delta^2/\epsilon) + \|\mathbf{U}(\mu - b)\|_2^2 + 2\delta\|\mathbf{U}\|_F^2\|b - \mu\|_2 \\
&\leq \|\mathbf{U}\|_F^2(1 + \delta^2/\epsilon + \|b - \mu\|_2^2 + 2\delta\|b - \mu\|_2) ,
\end{aligned}$$

where the second inequality from the end uses [Lemma 2.13](#). The lower bound is derived similarly:

$$\begin{aligned}
\mathbf{E}_{X \sim G} [w(X)\tilde{g}(X)] &= \mathbf{E}_{X \sim G} [w(X)] \mathbf{E}_{X \sim G_w} [\tilde{g}(X)] \\
&\geq (1 - \epsilon) \mathbf{E}_{X \sim G_w} [\tilde{g}(X)] \\
&\geq (1 - \epsilon)\|\mathbf{U}\|_F^2(1 - \delta^2/\epsilon - 2\delta\|b - \mu\|_2) ,
\end{aligned}$$

where we applied [Lemma 2.13](#) in the last step.  $\square$

## B Omitted Proofs from [Section 3](#)

### B.1 Johnson-Lindenstrauss Sketch

**Lemma 3.5.** *Fix a set of  $n$  points  $x_1, \dots, x_n \in \mathbb{R}^d$ . For  $t \in [K]$ , define  $g_t(x) := \|\mathbf{M}_t(x - \mu_t)\|_2^2$  and let  $\tilde{g}_t(x), v_{t,j}$  as in [Algorithm 1](#). If  $C$  is a sufficiently large constant and  $L = C \log((n + d)K/\tau)$ , with probability at least  $1 - \tau$ , for every  $t \in [K]$  we have the following:*

1.  $0.8g_t(x_i) \leq \tilde{g}_t(x_i) \leq 1.2g_t(x_i)$  for every  $i \in [n]$ ,
2.  $0.8\|\mathbf{M}_t\|_F^2 \leq \left(\frac{1}{L} \sum_{j=1}^L \|v_{t,j}\|_2^2\right) \leq 1.2\|\mathbf{M}_t\|_F^2$ .

*Proof.* We show that the claim holds for a fixed iteration  $t$  with probability  $\tau/K$ . Recall that  $\tilde{g}_t(x)$  from [Algorithm 1](#), can be written as

$$\tilde{g}_t(x) = \|\mathbf{U}_t(x - \mu_t)\|_2^2 = \frac{1}{L} \sum_{j \in [L]} (v_{t,j}^T(x - \mu_t))^2 = \frac{1}{L} \sum_{j \in [L]} (z_{t,j}^T \mathbf{M}_t(x_i - \mu_t))^2 .$$

Applying [Fact 2.7](#) with  $\gamma = \tau/K$  and  $u_i = \mathbf{M}_t(x_i - \mu_t)$ , gives that choosing  $L = C \log(nK/\tau)$  suffices to guarantee that  $\tilde{g}_t(x_i)/g_t(x_i) \in [0.8, 1.2]$  for every  $x_i$  with probability  $1 - \tau$ .

We now show the second claim. Again, fix a  $t \in [K]$ . Consider the orthonormal base  $\{e_i\}_{i=1}^d$  of  $\mathbb{R}^d$ . We apply [Fact 2.7](#) with  $\gamma = \tau/K$  and  $u_i = \mathbf{M}_t e_i$ ,  $i \in [d]$ . This yields that choosing  $L = C \log(dK/\tau)$ , we get that for all  $i \in [d]$ :

$$\frac{1}{L} \sum_{j=1}^L \text{tr}(z_{t,j} z_{t,j}^T \mathbf{M}_t e_i e_i^T \mathbf{M}_t^T) = \frac{1}{L} \sum_{j=1}^L (z_{t,j}^T u_i)^2 = [0.8, 1.2] \frac{1}{L} \sum_{j=1}^L \|u_i\|^2 = [0.8, 1.2] \text{tr}(\mathbf{M}_t^T \mathbf{M}_t e_i e_i^T) ,$$

with probability  $1 - \tau/K$ . Summing all these inequalities for  $i = 1, \dots, d$  and noting that  $\sum_{i=1}^d e_i e_i^T = \mathbf{I}_d$  gives that

$$\frac{1}{L} \sum_{j=1}^L \text{tr}(z_{t,j} z_{t,j}^T \mathbf{M}_t^T \mathbf{M}_t) = [0.8, 1.2] \text{tr}(\mathbf{M}_t^T \mathbf{M}_t) ,$$

which precisely means that  $\frac{1}{L} \sum_{j=1}^L \|v_{t,j}\|_2^2 = [0.8, 1.2] \|\mathbf{M}_t\|_F^2$ . To have both claims hold simultaneously, we can just apply [Fact 2.7](#) for all  $n + d$  points, giving the result.  $\square$

## B.2 Proof of Lemma 3.6

It is more useful to think of the algorithm in the following equivalent form.

---

### Algorithm 6 Downweighting Filter

---

```

1: function DOWNWEIGHTINGFILTER( $P, w, \tilde{\tau}, R, T, \ell_{\max}$ )
2:    $r \leftarrow CdR^{2+4\log d}$ .
3:    $w' \leftarrow w, \ell \leftarrow 1$ .
4:   while  $\mathbf{E}_{X \sim P} [w'(X)\tilde{\tau}(X)] > 2T$  and  $\ell \leq \ell_{\max}$  do
5:      $\ell \leftarrow \ell + 1$ 
6:      $w'(x) \leftarrow w(x)(1 - \tilde{\tau}(x)/r)$ 
7:   end while
8:   return  $w'$ .
9: end function

```

---

**Lemma 3.6.** *Let  $P = (1 - \epsilon)G + \epsilon B$  be the empirical distribution on  $n$  samples, as in Algorithm 1. If  $(1 - \epsilon) \mathbf{E}_{X \sim G} [w(X)\tilde{\tau}(X)] \leq T$ ,  $\|\tilde{\tau}\|_{\infty} \leq r$ , and  $\ell_{\max} > r/T$ , then Algorithm 2 modifies the weight function  $w$  to  $w'$  such that*

1.  $(1 - \epsilon) \mathbf{E}_{X \sim G} [w(X) - w'(X)] < \epsilon \mathbf{E}_{X \sim B} [w(X) - w'(X)]$ ,
2.  $\mathbf{E}_{X \sim P} [w'(X)\tilde{\tau}(X)] \leq 2T$ ,

and the algorithm terminates after  $O(\log(\ell_{\max}))$  iterations, each of which takes  $O(n)$  time.

*Proof.* We show correctness of Algorithm 6. We denote by  $w_{\ell}$  the weight function at the  $\ell$ -th iteration of the filter, which is of the form  $w_{\ell}(x) = w(x)(1 - \tilde{\tau}(x)/r)^{\ell}$  for every  $x \in \mathbb{R}^d$ . To show the first claim, we fix an iteration  $\ell$  for which the algorithm has not stopped yet and examine the loss in weight between that iteration and the  $(\ell + 1)$ -th iteration. From the update rule  $w_{\ell+1}(x) = w_{\ell}(x)(1 - \tilde{\tau}(x)/r)$  we get that  $w_{\ell}(x) - w_{\ell+1}(x) = w_{\ell}(x)\tilde{\tau}(x)/r$ . Thus, the weight removed in that iteration from the good distribution is

$$(1 - \epsilon) \mathbf{E}_{X \sim G} [w_{\ell}(X) - w_{\ell+1}(X)] = \frac{1 - \epsilon}{r} \mathbf{E}_{X \sim G} [w_{\ell}(X)\tilde{\tau}(X)] \leq \frac{1}{r} T,$$

while the weight removed from the bad distribution is

$$\begin{aligned} \epsilon \mathbf{E}_{X \sim B} [w_{\ell}(X) - w_{\ell+1}(X)] &= \frac{1}{r} \epsilon \mathbf{E}_{X \sim B} [w_{\ell}(X)\tilde{\tau}(X)] \\ &= \frac{1}{r} \left( \mathbf{E}_{X \sim P} [w_{\ell}(X)\tilde{\tau}(X)] - (1 - \epsilon) \mathbf{E}_{X \sim G} [w_{\ell}(X)\tilde{\tau}(X)] \right) \\ &> \frac{1}{r} T, \end{aligned}$$

where the last inequality uses that  $(1 - \epsilon) \mathbf{E}_{X \sim G} [w_{\ell}(X)\tilde{\tau}(X)] \leq (1 - \epsilon) \mathbf{E}_{X \sim G} [w(X)\tilde{\tau}(X)] \leq T$  and the fact that since the algorithm has not terminated in the  $\ell$ -th iteration it must be true that  $\mathbf{E}_{X \sim P} [w_{\ell}(X)\tilde{\tau}(X)] > 2T$ . This completes the proof of the first claim.

Regarding runtime, it suffices to show that for any  $\ell > \frac{r}{eT}$ ,  $\mathbf{E}_{X \sim P} [w_{\ell}(X)\tilde{\tau}(X)] \leq 2T$ . This follows from the inequalities

$$\mathbf{E}_{X \sim P} [w_{\ell}(X)\tilde{\tau}(X)] = \mathbf{E}_{X \sim P} \left[ w(X)(1 - \tilde{\tau}(X)/r)^{\ell} \tilde{\tau}(X) \right]$$

$$\begin{aligned} &\leq \mathbf{E}_{X \sim P} [w(X) \exp(-\ell \tilde{\tau}(X)/r) \tau(X)] \\ &\leq \frac{r}{e \cdot \ell} \mathbf{E}_{X \sim P} [w(X)] \leq \frac{r}{e \cdot \ell} \leq T, \end{aligned}$$

where we used the fact that  $x e^{-\alpha x} \leq 1/(e \cdot \alpha)$  for all  $x \in \mathbb{R}$ . By noting that  $w(x)(1 - \tilde{\tau}(x)/r)^\ell$  is monotonically decreasing as  $\ell$  grows, we can improve the running time by using a binary search implementation. This gives the logarithmic guarantee of our statement.  $\square$

### B.3 Proof of Lemma 3.7

We state and prove a more general version of Lemma 3.7 so that it can be also used in Section 4. The difference is that we allow the scores to center points using a vector different from the true mean  $\mu_t$  of  $P_t$ , so long as this vector is  $O(\delta)$ -close to  $\mu_t$  in Euclidean norm. Lemma 3.7 is obtained by using Lemma B.1 below with  $\hat{\mu}_t = \mu_t$ .

**Lemma B.1.** *Consider the setting of Algorithm 1 and the deterministic Condition 3.4. Moreover, let  $\hat{\mu}_t$  be any vector in  $\mathbb{R}^d$  with  $\|\hat{\mu}_t - \mu_t\| = O(\delta)$  and define the functions*

$$\begin{aligned} f_t(x) &:= \|\mathbf{M}_t(x - \hat{\mu}_t)\|_2^2, & \tilde{f}_t(x) &:= \|\mathbf{U}_t(x - \hat{\mu}_t)\|_2^2 \\ h_t(x) &:= f_t(x) \mathbb{1}\{f_t(x) > C_3 \|\mathbf{M}_t\|_F^2 \lambda_t / \epsilon\}, & \tilde{h}_t(x) &:= \tilde{f}_t(x) \mathbb{1}\{\tilde{f}_t(x) > C_3 \|\mathbf{U}_t\|_F^2 \hat{\lambda}_t / \epsilon\}. \end{aligned} \quad (32)$$

We have that  $\mathbf{E}_{X \sim G}[w_t(X)h_t(X)]$  and  $\mathbf{E}_{X \sim G}[w_t(X)\tilde{h}_t(X)]$  are bounded from above by  $c\lambda_t \|\mathbf{M}_t\|_F^2$  for some constant  $c$  of the form  $c = C/C_2$ , where  $C_2$  is the constant used in Line 15 and  $C$  is some absolute constant.

We prove the result by using the facts from Section 2.2. For brevity, we will prove the results for  $\tilde{h}_t$  by using the functions  $\tilde{f}_t$  and the matrix  $\mathbf{U}_t$ ; the results for  $h_t$  would follow by replacing  $\tilde{f}_t$  and  $\mathbf{U}_t$  with  $f_t$  and  $\mathbf{M}_t$  respectively and using that  $\|\mathbf{U}_t\|_F$  is close to  $\|\mathbf{M}_t\|_F$  (Item 2 of the deterministic condition). We begin with Lemma B.2, which is a generalization of the following implication of stability: The  $(\epsilon, \delta)$ -stability of a distribution  $G$  implies that  $\mathbf{E}_{X \sim G}[(v^T(X - \mu))^2 \mathbb{1}\{X \in L\}] \leq 3\delta^2/\epsilon$  for any set  $L$  with mass  $\Pr_{X \sim G}[X \in L] \leq \epsilon$  (see, for example, Proposition C.3 of [P JL20]). The following lemma generalizes this to having a matrix in place of  $v$ .

**Lemma B.2.** *Under the setting of Algorithm 1, the deterministic Condition 3.4, and using the notation of Equation (32), if  $L_t \subseteq \mathbb{R}^d$  is a set with  $\mathbf{E}_{X \sim G}[w_t(X) \mathbb{1}\{X \in L_t\}] \leq \epsilon$ , then we have that*

$$\mathbf{E}_{X \sim G}[w_t(X) \tilde{f}_t(X) \mathbb{1}\{X \in L_t\}] \leq c\lambda_t \|\mathbf{U}_t\|_F^2,$$

for some constant  $c$  of the form  $c = C'/C_2$ , where  $C'$  is a sufficiently large constant.

*Proof.* Define the new weight function  $w'_t(x) = w_t(x) \mathbb{1}\{x \notin L_t\}$ . We have assumed that the distribution  $G$  is  $(C''\epsilon, \delta)$ -stable. Let  $\epsilon' := C''\epsilon$  for brevity. We have the following inequalities, which we explain below.

$$\begin{aligned} \mathbf{E}_{X \sim G}[w_t(X) \tilde{f}_t(X) \mathbb{1}\{X \in L_t\}] &= \mathbf{E}_{X \sim G}[w_t(X) \tilde{f}_t(X)] - \mathbf{E}_{X \sim G}[w_t(X) \tilde{f}_t(X) \mathbb{1}\{X \notin L_t\}] \\ &= \mathbf{E}_{X \sim G}[w_t(X) \tilde{f}_t(X)] - \mathbf{E}_{X \sim G}[w'_t(X) \tilde{f}_t(X)] \\ &\leq \|\mathbf{U}_t\|_F^2 \left( 1 + \frac{\delta^2}{\epsilon'} + \|\hat{\mu}_t - \mu\|_2^2 + 2\delta \|\hat{\mu}_t - \mu\|_2 \right) \end{aligned}$$

$$\begin{aligned}
& - (1 - 2\epsilon) \|\mathbf{U}_t\|_F^2 \left( 1 - \frac{\delta^2}{\epsilon'} - 2\delta \|\widehat{\mu}_t - \mu\|_2 \right) \\
& \leq \|\mathbf{U}_t\|_F^2 \left( 3 \frac{\delta^2}{\epsilon'} + 4\delta \|\widehat{\mu}_t - \mu\|_2 + \|\widehat{\mu}_t - \mu\|_2^2 \right) \\
& \leq \|\mathbf{U}_t\|_F^2 \left( 3 \frac{\delta^2}{\epsilon'} + 4\delta (\|\widehat{\mu}_t - \mu_t\|_2 + \|\mu_t - \mu\|_2) + 4\|\widehat{\mu}_t - \mu_t\|_2^2 + 4\|\mu_t - \mu\|_2^2 \right) \\
& \leq \|\mathbf{U}_t\|_F^2 \left( 3 \frac{\delta^2}{\epsilon'} + \delta O(\delta + \sqrt{\epsilon' \lambda_t}) + O(\delta^2 + \epsilon' \lambda_t) \right) \\
& \leq c \|\mathbf{U}_t\|_F^2 \lambda_t .
\end{aligned}$$

We note that the third line above follows by applying [Corollary 2.14](#) with  $\mathbf{U} = \mathbf{U}_t$  and  $b = \mu_t$  on both terms of the previous line. The fifth line uses the triangle inequality. The sixth line uses the assumption that  $\|\widehat{\mu}_t - \mu_t\|_2 = O(\delta)$  as well as the certificate lemma ([Lemma 2.11](#)). Note that the required assumption  $d_{\text{TV}}(P_t, P) = O(\epsilon)$  from that lemma is satisfied because  $\mathbf{E}_{X \sim G}[w'_t(X)] \geq \mathbf{E}_{X \sim G}[w_t(X)] - \epsilon \geq 1 - O(\epsilon)$  (see [Claim 3.13](#)).

Regarding that last line, we recall [Line 15](#) of [Algorithm 1](#), which implies that  $\lambda_t \gtrsim C_2 \delta^2 / \epsilon$ . Thus the terms  $\delta^2 / \epsilon'$  can be bounded as  $\delta^2 / \epsilon' \lesssim 1 / C_2$ . Using that, it can be seen that we can choose  $c = C' / C_2$  for some constant  $C' > 0$ .  $\square$

We are now ready to prove our result.

*Proof of [Lemma B.1](#).* We first show the following.

**Claim B.3.** *Consider the setting of [Algorithm 1](#), the notation of [Equation \(32\)](#), and assume that the deterministic [Condition 3.4](#) holds. Let  $L_t = \{x : \tilde{f}_t(x) > C_3 \|\mathbf{U}_t\|_F^2 \hat{\lambda}_t / \epsilon\}$ . We have that  $\mathbf{E}_{X \sim G}[w_t(X) \mathbf{1}\{X \in L_t\}] \leq \epsilon$ .*

*Proof.* Let  $u^* := \arg \max\{u : \mathbf{E}_{X \sim G}[w_t(X) \mathbf{1}\{\tilde{f}_t(X) > u\}] \geq \epsilon\}$  and the set  $L_t^* = \{x : \tilde{f}_t(x) > u^*\}$ . It suffices to show that  $u^* \leq C_3 \|\mathbf{U}_t\|_F^2 \hat{\lambda}_t / \epsilon$  (because this would mean that  $L_t \subseteq L_t^*$ ).

By [Lemma B.2](#), we have that  $\mathbf{E}_{X \sim G}[w_t(X) \tilde{f}_t(X) \mathbf{1}\{X \in L_t^*\}] \leq (C' / C_2) \lambda_t \|\mathbf{U}_t\|_F^2$ . If we define the new weights  $w'_t(x) = w_t(x) \mathbf{1}\{x \in L_t^*\}$  and use them to normalize the distribution, we get that

$$\mathbf{E}_{X \sim G_{w'_t}}[\tilde{f}_t(X)] = \frac{1}{\mathbf{E}_{X \sim G}[w_t(X) \mathbf{1}\{X \in L_t^*\}]} \mathbf{E}_{X \sim G}[w_t(X) \tilde{f}_t(X) \mathbf{1}\{X \in L_t^*\}] \leq (C' / C_2) \|\mathbf{U}_t\|_F^2 \frac{\hat{\lambda}_t}{\epsilon} ,$$

where we used that the denominator is  $\epsilon$ . The fact that  $\mathbf{E}_{X \sim G_{w'_t}}[\tilde{f}_t(X)] \leq (C' / C_2) \|\mathbf{U}_t\|_F^2 \hat{\lambda}_t / \epsilon$  means that at least one point in  $L_t^*$  has  $\tilde{f}_t(X) \leq (C' / C_2) \|\mathbf{U}_t\|_F^2 \hat{\lambda}_t / \epsilon$ , which shows that  $u^* \leq (C' / C_2) \|\mathbf{U}_t\|_F^2 \hat{\lambda}_t / \epsilon$ . Since the algorithm uses the value  $C_3 = (C' / C_2)$ , the proof is completed.  $\square$

[Lemma B.1](#) now follows by combining [Claim B.3](#) with [Lemma B.2](#):

$$\begin{aligned}
\mathbf{E}_{X \sim G}[w_t(X) \tilde{h}_t(X)] &= \mathbf{E}_{X \sim G} \left[ w_t(X) \tilde{f}_t(X) \mathbf{1} \left\{ \tilde{f}_t(X) > C_3 \|\mathbf{U}_t\|_F^2 \frac{\hat{\lambda}_t}{\epsilon} \right\} \right] \\
&\leq (C' / C_2) \lambda_t \|\mathbf{U}_t\|_F^2 \leq 2(C' / C_2) \lambda_t \|\mathbf{M}_t\|_F^2 ,
\end{aligned}$$

where the last inequality is due to [Item 2](#) of [Condition 3.4](#). Letting  $C = 2C'$  we have that  $\mathbf{E}_{X \sim G}[w_t(X) \tilde{h}_t(X)] \leq C / C_2$  as claimed. As mentioned earlier, the same techniques lead to a similar bound on  $\mathbf{E}_{X \sim G}[w_t(X) h_t(X)]$ .  $\square$

## B.4 Proof of Claim 3.11

**Claim 3.11.** *Let the fraction of outliers be  $\epsilon < 1/10$  and a parameter  $0 < \tau < 1$ . Let the distribution  $P = (1 - \epsilon)G + \epsilon B$ . Let  $R > 0, \mu \in \mathbb{R}^d$  be such that  $\Pr_{X \sim G}[\|X - \mu\|_2 > R] \leq \epsilon$ . There is an estimator  $\hat{\mu}$  on  $k = O(\log(1/\tau))$  samples from  $P$  such that  $\|\hat{\mu} - \mu\|_2 \leq 4R$  with probability at least  $1 - \tau$ . Furthermore,  $\hat{\mu}$  can be computed in time  $O(k^2 d)$  and memory  $O(kd)$ .*

We can use the following well-known fact (see, e.g., [DHL19], for a proof):

**Fact B.4.** *There is an algorithm NaivePrune with the following guarantees. Let  $\epsilon' < 1/2$ , and  $\tau > 0$ . Let  $S \subset \mathbb{R}^d$  be a set of  $n$  points so that there exists a  $\mu \in \mathbb{R}^d$  and a subset  $S' \subseteq S$  so that  $|S'| \geq (1 - \epsilon')n$ , and  $\|x - \mu\|_2 \leq R$  for all  $x \in S'$ . Then  $\text{NaivePrune}(S, R, \tau)$  runs in time  $O(nd \log(1/\tau))$ , uses memory  $O(nd)$ , and with probability  $1 - \tau$  outputs a set of points  $T \subset S$  so that  $S' \subseteq T$ , and  $\|x - \mu\|_2 \leq 4R$  for all  $x \in T$ .*

*Proof of Claim 3.11.* The estimator draws a set  $S = \{X_1, \dots, X_k\}$  of  $k$  samples from the distribution  $P$ . Letting  $Y_i := \mathbb{1}\{\|X_i - \mu\|_2 > R\}$  we have that  $\mathbf{E}[Y_i] \leq 2\epsilon \leq 1/5$ . Thus, using Hoeffding bound,

$$\Pr \left[ \frac{1}{k} \sum_{i=1}^k Y_i > 1/4 \right] \leq \Pr \left[ \frac{1}{k} \sum_{i=1}^k Y_i < \mathbf{E} \left[ \frac{1}{k} \sum_{i=1}^k Y_i \right] + 0.05 \right] \leq 2e^{-2k(0.05)^2}.$$

Choosing  $k = 200 \log(2/\tau)$  makes this probability at most  $\tau$ . Conditioning on that event, the fraction of points outside the ball is at most  $\epsilon' := 1/4$ , thus running  $\text{NaivePruning}(S, R, \tau)$  algorithm of Fact B.4 and outputting any point from the returned set yields the desired estimator.  $\square$

## B.5 Omitted Proofs from Section 3.4

**Claim 3.13.** *Under Condition 3.4, Algorithm 1 maintains the following invariant:  $\mathbf{E}_{X \sim G}[w_t(X)] \geq 1 - 3\epsilon$ . In particular, if  $\epsilon \leq 1/8$ , then  $d_{\text{TV}}(P_t, P) \leq 9\epsilon$ .*

*Proof.* For every iteration, we denote by  $\Delta w_t = w_t - w_{t+1}$ , that is for every point  $x \in \mathbb{R}^d$ ,  $\Delta w_t(x) = w_t(x) - w_{t+1}(x)$  is the difference between the weights for the two consecutive iterations  $t$  and  $t + 1$ .

$$\begin{aligned} \mathbf{E}_{X \sim G}[w_t(X)] &= \mathbf{E}_{X \sim G}[w_1(X)] - \sum_{i=1}^{t-1} \mathbf{E}_{X \sim G}[\Delta w_i(X)] \\ &\geq 1 - \epsilon - \sum_{i=1}^{t-1} \mathbf{E}_{X \sim G}[\Delta w_i(X)] && \text{(Claim 3.12)} \\ &\geq 1 - \epsilon - \frac{\epsilon}{1 - \epsilon} \sum_{i=1}^{t-1} \mathbf{E}_{X \sim B}[\Delta w_i(X)] && \text{(Lemma 3.6)} \\ &\geq 1 - \epsilon - \frac{\epsilon}{1 - \epsilon} \left( \mathbf{E}_{X \sim B}[w_1(X)] - \mathbf{E}_{X \sim B}[w_t(X)] \right) \\ &\geq 1 - \epsilon - \frac{\epsilon}{1 - \epsilon} \\ &\geq 1 - 3\epsilon, \end{aligned}$$

where the last line uses that  $\epsilon \leq 1/2$ . The proof of the second conclusion follows from Claim B.5 (stated below).  $\square$



**Claim B.5.** Let  $\epsilon \leq 1/8$ . If  $\mathbf{E}_{X \sim G}[w_t(X)] \geq 1 - 3\epsilon$ , then  $d_{\text{TV}}(P_t, P) \leq 9\epsilon$ .

Although we work with discrete distributions (the empirical distributions on the samples) in [Section 3](#), we prove the claim for continuous distributions because it will be useful in [Section 4](#).

*Proof.* By definition  $P_t(x) = w_t(x)P(x) / \mathbf{E}_{X \sim P}[w_t(X)]$ . Letting  $L := \int_{\mathbb{R}^d} w_t(x)P(x)dx = \mathbf{E}_{X \sim P}[w_t(x)]$ , we have that

$$\int_{\mathbb{R}^d} |P_t(x) - P(x)|dx = (1 - \epsilon) \int_{\mathbb{R}^d} G(x) \left| \frac{w_t(x) - L}{L} \right| dx + \epsilon \int_{\mathbb{R}^d} B(x) \left| \frac{w_t(x) - L}{L} \right| dx.$$

We note that  $1 \geq L \geq (1 - \epsilon) \mathbf{E}_{X \sim G}[w_t(x)] \geq 1 - 4\epsilon$  using [Claim 3.13](#). The second term can be bounded as

$$\epsilon \int_{\mathbb{R}^d} B(x) \left| \frac{w_t(x) - L}{L} \right| dx \leq \frac{\epsilon}{L} \int_{\mathbb{R}^d} B(x)(w_t(x) + L) \leq \frac{2\epsilon}{1 - 4\epsilon} \leq 4\epsilon.$$

For the first term, we have that

$$\begin{aligned} (1 - \epsilon) \int_{\mathbb{R}^d} G(x) \left| \frac{w_t(x) - L}{L} \right| dx &\leq \frac{1 - \epsilon}{L} \int_{\mathbb{R}^d} G(x)(|1 - w_t(x)| + |1 - L|)dx \\ &\leq \frac{1}{1 - 4\epsilon} \left( 1 - \mathbf{E}_{X \sim G}[w_t(X)] + 4\epsilon \right) \leq 14\epsilon. \end{aligned}$$

□

**Claim 3.14.** Under [Condition 3.4](#), if  $C_1 \geq 22$ ,  $\mathbf{B}_t \succeq (0.5C_1\delta^2/\epsilon)\mathbf{I}_d$  for every  $t \in [K]$ .

*Proof.* Using the simple fact that for random variables  $X, Y$  it holds  $\text{Var}(Y) \geq \mathbf{E}_X[\text{Var}(Y|X)]$ , we get that

$$\begin{aligned} \Sigma_t &\succeq (1 - \epsilon) \mathbf{E}_{X \sim G_{w_t}} [(X - \mu_{G_{w_t}})(X - \mu_{G_{w_t}})^T] \\ &= (1 - \epsilon) \left( \mathbf{E}_{X \sim G_{w_t}} [(X - \mu)(X - \mu)^T] - (\mu_{G_{w_t}} - \mu)(\mu_{G_{w_t}} - \mu)^T \right) \\ &\succeq (1 - \epsilon) ((1 - \delta^2/\epsilon)\mathbf{I}_d - \delta^2\mathbf{I}_d) && \text{(by stability and Claim 3.13)} \\ &\succeq (1 - \epsilon)(1 - 2\delta^2/\epsilon)\mathbf{I}_d \\ &\succeq (1 - 3\delta^2/\epsilon)\mathbf{I}_d, \end{aligned}$$

where we used that  $\epsilon \leq \delta$ . We recall the definition  $\mathbf{B}_t = (\mathbf{E}_{X \sim P_t}[w_t(X)])^2 \Sigma_t - (1 - C_1\delta^2/\epsilon)\mathbf{I}_d$  and bound the first term as follows

$$\begin{aligned} \left( \mathbf{E}_{X \sim P_t}[w_t(X)] \right)^2 \Sigma_t &\succeq (1 - 3\epsilon)^2 (1 - 3\delta^2/\epsilon)\mathbf{I}_d && \text{(Claim 3.13)} \\ &\succeq (1 - 4\delta^2/\epsilon - 6\epsilon - 27\delta^2\epsilon)\mathbf{I}_d \\ &\succeq (1 - 11\delta^2/\epsilon)\mathbf{I}_d, \end{aligned}$$

where the last line uses  $\epsilon < 1/6$  and  $\epsilon \leq \delta$ . Therefore, if we choose  $C_1 \geq 22$ , we get that  $\mathbf{B}_t \succeq (0.5C_1\delta^2/\epsilon)\mathbf{I}_d$ . □

**Claim 3.10.** In the setting of [Algorithm 1](#) and under the [Condition 3.4](#), if  $x \in S$ , we have that  $\tau_t(x) \leq 1.25\tilde{\tau}_t(x) + 3C_3(\lambda_t/\epsilon)\text{tr}(\mathbf{M}_t^2)$ , where  $C_3$  is the constant used in [Algorithm 1](#).

*Proof.* By [Condition 3.4](#) we have that for all the  $n$  samples,  $\tilde{g}_t(x) \geq 0.8g_t(x)$ . Recall the definitions  $\tilde{\tau}_t(x) = \tilde{g}(x)\mathbf{1}\{\tilde{g}(x) > C_3\|\mathbf{U}_t\|_F^2\hat{\lambda}_t/\epsilon\}$  and  $\tau_t(x) = g(x)\mathbf{1}\{g(x) > C_3\|\mathbf{M}_t\|_F^2\lambda_t/\epsilon\}$ . We split into cases based on whether each of  $g_t, \tilde{g}_t$  has been zeroed by its thresholding operation:

- If  $\tau_t(x)$  has been zeroed, (i.e.,  $g_t(x) < C_3\|\mathbf{M}_t\|_F^2\lambda_t/\epsilon$ ), the claim trivially holds since the left-hand side is zero.
- If none of  $\tilde{\tau}_t(x), \tau_t(x)$  has been zeroed, then  $\tilde{\tau}_t(x) = \tilde{g}_t(x)$  and  $\tau_t(x) = g_t(x)$ , thus the claim holds by the aforementioned fact that  $\tilde{g}_t(x) \geq 0.8g_t(x)$ .
- If  $\tilde{\tau}_t(x)$  has been zeroed but  $\tau_t(x)$  has not, then the worst case is  $g_t(x) = (1/0.8)\tilde{g}_t(x)$ . This means that in this case:

$$\tau_t(x) \leq \frac{1}{0.8}C_3\frac{\hat{\lambda}_t}{\epsilon}\|\mathbf{U}_t\|_F^2 < 3C_3\frac{\lambda_t}{\epsilon}\|\mathbf{M}_t\|_F^2,$$

where we used that  $\hat{\lambda} \leq 1.2\lambda_t$  and  $\|\mathbf{U}_t\|_F^2 \leq 1.2\|\mathbf{M}_t\|_F^2$ , due to [Condition 3.4](#). □

## C Omitted Proofs from [Section 4](#)

### C.1 Omitted Proofs from [Section 4.2](#)

**Lemma 4.6.** *In the context of [Algorithm 3](#), if  $(1 - \epsilon)\mathbf{E}_{X \sim G}[w(X)\tilde{\tau}(X)] \leq T$ ,  $\|\tilde{\tau}\|_\infty \leq r$ , and  $\ell_{\max} > r/T$ , then [Algorithm 4](#) modifies the weight function  $w$  to  $w'$  such that (i)  $(1 - \epsilon)\mathbf{E}_{X \sim G}[w(X) - w'(X)] < \epsilon\mathbf{E}_{X \sim B}[w(X) - w'(X)]$ , and (ii) upon termination we have  $\mathbf{E}_{X \sim P}[w'(X)\tilde{\tau}(X)] \leq 54T$ . Furthermore, if the estimator of [Line 3](#) is set to be that of [Lemma 4.16](#), the algorithm terminates after  $O(\log(\ell_{\max}))$  iterations, each of which uses  $O((R^2\epsilon/\delta^2)\log(1/\tau))$  samples, takes  $O(nd)$  time and memory  $O(\log(1/\tau))$ .*

*Proof.* Let the set  $L^* = \{\ell \in [\ell_{\max}] : 6T \leq \mathbf{E}_{X \sim P}[w(X)(1 - \tilde{\tau}(X)/r)^\ell \tilde{\tau}(X)] \leq 18T\}$ . The invariant is that throughout [Algorithm 4](#), the set  $L$  maintained has non-zero intersection with  $L^*$ . This can be seen by examining cases about  $\ell$  in [Line 6](#). If  $\ell \in L^*$ , then  $\ell$  is kept in  $L$ . If  $\ell \notin L^*$ , then all elements discarded are not members of  $L^*$  (for example if  $\mathbf{E}_{X \sim P}[w(X)(1 - \tilde{\tau}(X)/r)^\ell \tilde{\tau}(X)] > 18T$ , then by [Lemma 4.16](#)  $f(\ell) > 9T$  and we discard the lower half of  $L$ ). Thus, at the end,  $L$  has at most two elements with at least one of them belonging in  $L^*$ . This element would satisfy  $3T \leq f(\ell) \leq 27T$ . Thus the algorithm will definitely return some element. On the other hand, any element returned will satisfy  $2T < \mathbf{E}_{X \sim P}[w(X)(1 - \tilde{\tau}(X)/r)^\ell \tilde{\tau}(X)] < 54T$ . This has already shown part (ii) of the lemma. For part (i), it is more convenient to imagine let  $\ell$  increased by one at each step until it reaches the value finally returned by the algorithm and consider the loss in weight between that and the next iteration, exactly as in the proof of [Lemma 3.6](#). That proof was using only the facts that for all  $\ell' \leq \ell$ ,  $2T < \mathbf{E}_{X \sim P}[w(X)(1 - \tilde{\tau}(X)/r)^{\ell'} \tilde{\tau}(X)]$  (which we just showed above) and  $\mathbf{E}_{X \sim G}[w(X)(1 - \tilde{\tau}(X)/r)^{\ell'} \tilde{\tau}(X)] < T$  (which is true by assumption). The reason why  $\ell_{\max} = r/T$  suffices is also shown identically to the proof of [Lemma 3.6](#). □

### C.2 Omitted Proofs from [Section 4.2.1](#)

We now focus on showing [Lemma 4.9](#). In order to avoid confusion with the fraction of outliers  $\epsilon$ , we use  $\epsilon'$  for our accuracy parameter. We will use a uniform convergence result from [\[AB99\]](#) combined with a powerful VC-dimension bound from [\[GJ95\]](#) for the class of functions that are computable by

a small number of arithmetic operations. [GJ95] considers the class of concepts parameterized by  $k$  real numbers,  $\mathcal{F} = \{h_a\}_{a \in \mathbb{R}^k}$ , for which there exists an algorithm  $\mathcal{A}$  for calculating  $h_a(x)$  that takes as input  $x, a$  and each line of  $\mathcal{A}$  is one of the following:

- an arithmetic operation  $+, -, \times,$  and  $/$  on two inputs or previously computed values,
- a jump to a different line of the algorithm conditioned on whether an input or a previously calculated value is greater than or equal to zero,
- output zero or one.

The parameters  $a$  and the inputs  $x$  consist of real numbers, and the model of computation assumed allows for arithmetic operations and the comparisons between reals to be done in constant time. We refer the reader to [GJ95, Section 2] for more details and relation with algebraic decision trees with bounded depths. The result from [GJ95] is that  $\text{VCdim}(\mathcal{F}) = O(mk)$  where  $k$  is the size of the parameterization and  $m$  is the runtime of the algorithm  $\mathcal{A}$ . Using the bound on the VC dimension, we have the following result for the uniform convergence:

**Proposition C.1** ([GJ95, AB99]). *Let  $\mathcal{F}$  be a class of functions of the form  $\mathcal{F} = \{h_a : \mathbb{R}^d \rightarrow [0, 1] \mid a \in \mathbb{R}^k\}$ , where for any  $(a, x) \in \mathbb{R}^k \times \mathbb{R}^d$ ,  $h_a(x)$  can be computed by an algorithm  $\mathcal{A}$  with runtime  $m$  that takes as input  $a, x$  and is allowed to perform conditional jumps (conditioned on equality and inequality of real values) and execute the standard arithmetic operations on real numbers  $(+, -, \times, /)$  in constant time. For any distribution  $D$  on  $\mathbb{R}^d$  and any  $\epsilon' \in (0, 1)$ , there exist  $N = O\left(\frac{1}{(\epsilon')^2}(\log(km) + km \log(1/\epsilon'))\right)$  points  $x_1, \dots, x_N$  in  $\mathbb{R}^d$  such that*

$$\sup_{h \in \mathcal{F}} \left| \mathbf{E}_{X \sim D} [h(X)] - \frac{1}{N} \sum_{i=1}^N h(x_i) \right| \leq \epsilon'.$$

For completeness, we show at the end of this section how this is derived from the statements of [AB99] and [GJ95]. We now apply this result to our case. We need to specify a family  $\mathcal{F}$  of functions broad enough to capture every  $w_{t+1}\tilde{\tau}_t$  and  $w_{t+1}\tau_t$  that could be encountered during the execution of Algorithm 3. The factor  $r$  used in the statement below is a normalization factor to make sure that the functions are in  $[0, 1]$ .

**Lemma C.2.** *Consider the setting of Algorithm 3. Let  $r' := (CdR^2 + 1 + C_1\delta^2/\epsilon)^{C \log d}$ . There exists a family  $\mathcal{F}$  of functions from  $\mathbb{R}^d$  to  $[0, 1]$  such that:*

1. *For every iteration  $t$  of Algorithm 3, we have that  $\frac{1}{r'}w_{t+1}\tau_t \in \mathcal{F}$  and  $\frac{1}{r'}w_{t+1}\tilde{\tau}_t \in \mathcal{F}$ .*
2. *Functions of  $\mathcal{F}$  are parameterized by at most  $k = O(dK \max(L, d))$  real numbers, that is,  $\mathcal{F}$  has the form  $\mathcal{F} = \{h_a : \mathbb{R}^d \rightarrow [0, 1] \mid a \in \mathbb{R}^k\}$ .*
3. *For every  $h_a \in \mathcal{F}$  and  $x \in \mathbb{R}^d$ ,  $h_a(x)$  can be in  $m = dK \max(L, d)(dR\epsilon/\delta^2)^{O(\log d)}$  steps in the model that allows conditional jumps and standard arithmetic operations on real numbers.*

*Proof.* Let  $L'_2 := \max(L, d)$ . Every function in our family will be parameterized by  $2K + 1$  scalars,  $\{u_t \in \mathbb{R} : t \in [K + 1]\} \cup \{\ell_t \in \mathbb{R} : t \in [K]\}$ , and  $(K + 1)(L'_2 + 1)$  vectors in  $\mathbb{R}^d$ ,  $\{a_t : t \in [K + 1]\} \cup \{v_{t,j} : t \in [K + 1], j \in [L'_2]\}$ . For brevity, we denote by  $\mathbf{V}$  the tensor in  $\mathbb{R}^{(K+1) \times L'_2 \times d}$  having all the vectors  $\mathbf{V}_{t,j} = v_{t,j}$  and by  $\mathbf{A}$  the tensor in  $\mathbb{R}^{(K+1) \times d}$  with  $\mathbf{A}_t = a_t$ ,

$t \in [K+1]$ . Similarly, denote by  $u$  the vector  $(u_1, \dots, u_{K+1})$  and let  $\ell = (\ell_1, \dots, \ell_K)$ . We define our class to be

$$\mathcal{F} = \left\{ h_{\ell, u, \mathbf{V}, \mathbf{A}} : \mathbb{R}^d \rightarrow [0, 1] : u \in \mathbb{R}^{K+1}, \ell \in \mathbb{R}^K, \mathbf{V} \in \mathbb{R}^{(K+1) \times L'_2 \times d}, \mathbf{A} \in \mathbb{R}^{(K+1) \times d} \right\},$$

which includes all functions of the form  $h_{\ell, u, \mathbf{V}, \mathbf{A}}(x) = \tilde{h}_{\ell, u, \mathbf{V}, \mathbf{A}} \mathbb{1}\{\tilde{h}_{\ell, u, \mathbf{V}, \mathbf{A}}(x) \in (0, 1)\}$ , where

$$\begin{aligned} \tilde{h}_{\ell, u, \mathbf{V}, \mathbf{A}}(x) = & \mathbb{1}\{\|x - \hat{\mu}\|_2 \leq 5R\} \frac{1}{r} \cdot \frac{\sum_{j=1}^{L'_2} (v_{K+1, j}^T(x - a_{K+1}))^2}{L'_2} \mathbb{1}\left\{ \frac{\sum_{j=1}^{L'_2} (v_{K+1, j}^T(x - a_{K+1}))^2}{L'_2} > u_{K+1} \right\} \\ & \cdot \prod_{t=1}^K \left( 1 - \frac{1}{r} \frac{\sum_{j=1}^{L'_2} (v_{t, j}^T(x - a_t))^2}{L'_2} \mathbb{1}\left\{ \frac{\sum_{j=1}^{L'_2} (v_{t, j}^T(x - a_t))^2}{L'_2} > u_t \right\} \right)^{\ell_t}. \end{aligned} \quad (33)$$

We note that the radius  $r' := (CdR^2 + 1 + C_1\delta^2/\epsilon)^{C \log d}$  is an upper bound on the value that the functions  $w_{t+1}\tau_t$  and  $w_{t+1}\tilde{\tau}_t$  in [Algorithm 3](#) can take: For  $\tau_t(x)$  we have

$$\tau_t(x) \leq \|\mathbf{M}_t(x - \mu_t)\|_2^2 \leq \|\mathbf{M}_t\|_2^2 \|x - \mu_t\|_2^2 \leq \|\Sigma_t\|_2^{2 \log d} R^2 = O(dR^{2+4 \log d}),$$

while for  $\tilde{\tau}(x)$  we have the bounds

$$\begin{aligned} \tilde{\tau}_t(x) & \leq \tilde{g}_t(x) \leq \|\mathbf{U}_t(x - \mu_t)\|_2^2 \lesssim R^2 \|\mathbf{U}_t\|_2^2 \lesssim R^2 \|\mathbf{U}_t\|_F^2 \\ & \leq R^2 \frac{1}{L} \sum_{j=1}^L \|\mathbf{M}_t z_{t, j}\|_2^2 \leq dR^2 \|\mathbf{M}_t\|_2^2 \leq dR^2 \|\mathbf{B}_t\|_2^{2 \log d} \\ & \leq dR^2 (\|\Sigma_t\|_2 + 1 + C_1\delta^2/\epsilon)^{2 \log d} \leq dR^2 (CR^2 + 1 + C_1\delta^2/\epsilon)^{2 \log d} \\ & \leq (CdR^2 + 1 + C_1\delta^2/\epsilon)^{O(\log d)}. \end{aligned}$$

We check that  $\mathcal{F}$  can indeed implement the functions  $w_{t+1}\tilde{\tau}_t$  used in [Algorithm 3](#) for any  $t \in [K]$ : Note that the scores  $\tilde{g}_t$  used in the algorithm are means of the form  $\frac{1}{L} \sum_{j=1}^L (v_{t, j}^T(x - a_t))^2$ . Thus, the first line of [Equation \(33\)](#) implements  $\mathbb{1}\{\|x - \hat{\mu}\|_2 \leq 5R\} \frac{1}{r} \tilde{\tau}_t$ . The purpose of the second line in [Equation \(33\)](#) is to match the operation of the Downweighting filter, which, in the  $t$ -th round multiplies  $w_t$  with  $(1 - \tilde{\tau}_t(x)/r)^{\ell_t}$  for some power  $\ell_t$ . Finally, we note that  $w_{t+1}\tau_t$  are implemented in  $\mathcal{F}$  by taking  $v_{t, j}$  to be the rows of the matrix  $\mathbf{M}_t$  (this is why we need the sums to be on  $L' = \max(L, d)$  terms in [Equation \(33\)](#)).

We need to specify the arithmetic complexity  $m$  and the dimension of the parameterization  $k$  of our family  $\mathcal{F}$ . For the first, we have that for any  $h \in \mathcal{F}$  and  $x \in \mathbb{R}^d$ , the value  $h(x)$  can be computed using  $O(KdL'\ell_{\max})$  standard arithmetic operations and jumps, where  $\ell_{\max}$  is the maximum exponent that  $\ell_t$  can have and is set to be  $\ell_{\max} := \left(\frac{dR}{\delta^2/\epsilon}\right)^{C \log d}$  in [Line 22](#) of [Algorithm 3](#). The  $dL'$  comes from the computation of the means  $(1/L') \sum_{j=1}^{L'} (v_{t, j}^T(x - a_t))^2$  and the  $K$  comes from the fact that we have  $K$  factors in the expression of  $h$ .

Regarding the other parameter  $k$ , we have that every  $h \in \mathcal{F}$  is parameterized by  $O(K)$  scalars and  $O(KL')$   $d$ -dimensional vectors. Thus,  $k = O(KL'd)$ .  $\square$

We are now ready to prove [Lemma 4.9](#).

**Lemma 4.9.** Consider the setting of [Algorithm 3](#), where  $B$  is the distribution of outliers supported in a ball of radius  $R$  around  $\mu$ . Let  $r' := (CdR^2 + 1 + C_1\delta^2/\epsilon)^{C \log d}$  for sufficiently large constant  $C$ . Denote by  $\epsilon$  the contamination rate and let an arbitrary  $\epsilon' \in (0, 1)$ . There exists a set  $S_{\text{cover}}$  of  $N = \frac{1}{\epsilon^3} d^4 K^2 L^2 (dR\epsilon/\delta^2)^{O(\log d)}$  points  $x_1, \dots, x_N$  lying in the ball of radius  $R$  around  $\mu$ , such that for all  $t \in [K]$ , for all choices of the vectors  $z_{t,j}$  of [Line 24](#) of [Algorithm 3](#) it holds

$$\left| \mathbf{E}_{X \sim B} \left[ \frac{1}{r'} w_{t+1}(X) \tilde{\tau}_t(X) \right] - \frac{1}{N} \sum_{i=1}^N \frac{1}{r'} w_{t+1}(x_i) \tilde{\tau}_t(x_i) \right| \leq \epsilon'$$

and

$$\left| \mathbf{E}_{X \sim B} \left[ \frac{1}{r'} w_{t+1}(X) \tau_t(X) \right] - \frac{1}{N} \sum_{i=1}^N \frac{1}{r'} w_{t+1}(x_i) \tau_t(x_i) \right| \leq \epsilon' .$$

*Proof of [Lemma 4.9](#).* We use [Proposition C.1](#) for the family  $\mathcal{F}$  of [Lemma C.2](#) and plug the upper bounds for the arithmetic complexity  $m$  and the dimension of the parameters  $k$ . [Proposition C.1](#) states that  $N$  can be chosen to be a multiple of

$$\frac{1}{\epsilon'^2} (\log(km) + km \log(1/\epsilon')) .$$

Taking the much looser bound  $N = \Theta(\frac{km}{\epsilon'^3})$  suffices for our purposes. Plugging in  $k = O(dK \max(L, d))$ ,  $m = dK \max(L, d) (dR\epsilon/\delta^2)^{O(\log d)}$  from [Lemma C.2](#), we get  $km = d^2 K^2 \max(d^2, L^2) (dR\epsilon/\delta^2)^{O(\log d)} \lesssim d^4 K^2 L^2 (dR\epsilon/\delta^2)^{O(\log d)}$ .  $\square$

For completeness, we provide the proof of [Proposition C.1](#).

*Proof of [Proposition C.1](#).* We derive the result from the statements of [\[AB99\]](#) without explaining all of the definitions of the notions involved. Please see [\[AB99\]](#) for more details. Applying [Theorem 17.7 \[AB99\]](#) with the loss function  $\ell_h(x, y) = h(x)$  we obtain

$$\Pr \left[ \sup_{h \in \mathcal{F}} \left| \mathbf{E}_{X \sim D} [h(X)] - \frac{1}{N} \sum_{i=1}^N h(X_i) \right| > \epsilon' \right] \leq 4\mathcal{N}_1(\epsilon'/8, \mathcal{F}, 2N) \exp(-\epsilon'^2 N/32) , \quad (34)$$

where the probability is taken over a set of  $N$  i.i.d. points  $X_1, \dots, X_N$  drawn from  $D$ .

To bound from above the covering number  $\mathcal{N}_1(\epsilon'/8, \mathcal{F}, 2N)$ , we use [Theorem 18.4](#) from [\[AB99\]](#) which gives that  $\mathcal{N}_1(\epsilon'/8, \mathcal{F}, 2N) \leq e(d' + 1)(16e/\epsilon')^{d'}$  where  $d' = \text{Pdim}(\mathcal{F})$  is the pseudo-dimension of  $\mathcal{F}$ . From that, we conclude that choosing any

$$N > \frac{32}{\epsilon'^2} \log(4e(d' + 1)) + \frac{32d'}{\epsilon'^2} \log(16e/\epsilon')$$

makes the probability in [Equation \(34\)](#) less than 1.

It remains to bound  $d'$  from above, which can be done as follows. Define the *subgraph class* associated to the family  $\mathcal{F}$

$$\mathcal{B}_{\mathcal{F}} := \{B_h \mid h \in \mathcal{F}\} ,$$

where for any  $h \in \mathcal{F}$ ,  $B_h : \mathbb{R}^{d+1} \rightarrow \{0, 1\}$  is defined as  $B_h(x, y) = \mathbb{1}\{h(x) \geq y\}$ . The pseudo-dimension is defined to be  $\text{Pdim}(\mathcal{F}) = \text{VCdim}(\mathcal{B}_{\mathcal{F}})$  (see [Section 11.2](#) in [\[AB99\]](#)). By [Theorem 2.3](#) in [\[GJ95\]](#), we have that  $\text{VCdim}(\mathcal{B}_{\mathcal{F}}) = O(km)$  since it  $\mathcal{B}_{\mathcal{F}}$  functions that are parametrized by vectors of  $\mathbb{R}^k$  (same as for family  $\mathcal{F}$ ) and the functions of  $B_h(x, y)$  can be computed using at most  $m + 2$

operations ( $m$  to compute  $h$  and two to do the comparison with  $y$  and threshold). Putting everything together, it suffices to choose

$$N = C \frac{1}{\epsilon'^2} (\log(km) + km \log(1/\epsilon'))$$

in order to make the probability in Equation (34) less than 1. In that case, by probabilistic argument, there exists at least one set of  $N$  points satisfying the desired event.  $\square$

**Claim 4.10.** *Let  $S$  be the cover of Lemma 4.9 with  $r'$  and  $\epsilon'$  as defined above. Suppose that the deterministic condition Condition 4.5 holds. If  $x \in S_{\text{cover}}$ , then  $\tau_t(x) \leq 5\tilde{\tau}_t(x) + (18C_3 + 12/C_2)(\lambda_t/\epsilon)\|\mathbf{M}_t\|_F^2$ , where  $C_3$  and  $C_2$  are the constants used in Algorithm 3.*

*Proof.* By Condition 4.5 we have that for all the  $N$  samples of the cover,  $\tilde{g}_t(x) \geq 0.2g_t(x) - 0.8(\delta^2/\epsilon^2)\|\mathbf{M}_t\|_F^2$ . Recall the definitions  $\tilde{\tau}_t(x) = \tilde{g}_t(x)\mathbb{1}\{\tilde{g}_t(x) > C_3\|\mathbf{U}_t\|_F^2\hat{\lambda}_t/\epsilon\}$ ,  $\tau_t(x) = g_t(x)\mathbb{1}\{g_t(x) > C_3\|\mathbf{M}_t\|_F^2\lambda_t/\epsilon\}$ . We split into cases based on whether each of  $g_t, \tilde{g}_t$  has been zeroed by their thresholding operation:

- If  $\tau_t(x)$  has been zeroed, (i.e.,  $g_t(x) < C_3\|\mathbf{U}_t\|_F^2\lambda_t/\epsilon$ ), the claim trivially holds since the left-hand side is zero.
- If none of  $\tilde{\tau}_t(x), \tau_t(x)$  has been zeroed, then  $\tilde{\tau}_t(x) = \tilde{g}_t(x)$  and  $\tau_t(x) = g_t(x)$ , thus the claim holds by the aforementioned fact that  $\tilde{g}_t(x) \geq 0.2g_t(x) - 0.8(\delta^2/\epsilon^2)\|\mathbf{M}_t\|_F^2$ .
- If  $\tilde{\tau}_t(x)$  has been zeroed but  $\tau_t(x)$  has not, then the worst case is  $\tilde{g}_t(x) = 0.2g_t(x) - 0.8(\delta^2/\epsilon^2)\|\mathbf{M}_t\|_F^2$ . This means that in this case:

$$\begin{aligned} \tau_t(x) &\leq \frac{1}{0.2}C_3\frac{\hat{\lambda}_t}{\epsilon}\|\mathbf{U}_t\|_F^2 + 4\frac{\delta^2}{\epsilon^2}\|\mathbf{M}_t\|_F^2 \\ &< 18C_3\frac{\lambda_t}{\epsilon}\|\mathbf{M}_t\|_F^2 + 4\frac{\delta^2}{\epsilon^2}\|\mathbf{M}_t\|_F^2 \\ &\leq 18C_3\frac{\lambda_t}{\epsilon}\|\mathbf{M}_t\|_F^2 + \frac{12}{C_2}\frac{\lambda_t}{\epsilon}\|\mathbf{M}_t\|_F^2, \end{aligned}$$

where in the second inequality we used that  $\hat{\lambda}_t \leq 3\lambda_t$  and  $\|\mathbf{U}_t\|_F^2 \leq 1.2\|\mathbf{M}_t\|_F^2$  due to Condition 4.5, and in the last inequality we used that  $\delta^2/\epsilon < \hat{\lambda}_t/C_2$  and  $\hat{\lambda}_t \leq 3\lambda_t$  (Condition 4.5 again).  $\square$

**Remark C.3** (On the choice of  $K$  and  $L$ ). We comment on how the values for the number of iterations  $K$  and  $L$  that are used in Algorithm 3 are derived. First, the derivation of  $K = C \log d \log(dR/(\delta^2/\epsilon))$  for large enough constant  $C$  is identical to that of Section 3.4 (see Equation (3)). We will thus focus on  $L$ . We note that in the proof of Lemma 4.7 we use Lemma 4.9 with  $\epsilon' \gtrsim \frac{(\delta^2/\epsilon)^{2 \log d}}{\epsilon(CdR^2+1+C_1\delta^2/\epsilon)^{C \log d}}$ . This means that the cover  $S_{\text{cover}}$  of that lemma has size bounded by

$$|S_{\text{cover}}| \leq \frac{1}{\epsilon^3} d^4 K^2 L^2 \left( \frac{dR}{\delta^2/\epsilon} \right)^{O(\log d)} \lesssim \frac{(CdR^2 + 1 + C_1\delta^2/\epsilon)^{O(\log d)}}{(\delta^2/\epsilon)^{O(\log d)}} L^2.$$

The analog of Lemma 3.5 thus requires that  $L$  is multiple of  $\log\left(\frac{|S_{\text{cover}}|+d}{\tau}\right)$ , where  $\tau$  is the desired probability of failure. Note that we have the following (rough) bounds

$$\log\left(\frac{|S_{\text{cover}}|+d}{\tau}\right) \lesssim \log\left(\frac{(L(CdR^2 + 1 + C_1\delta^2/\epsilon)^{O(\log d)})}{\tau(\delta^2/\epsilon)^{O(\log d)}}\right)$$

$$\lesssim \log^2(d) \log(CdR^2 + 1 + C_1\delta^2/\epsilon) \log\left(\frac{1}{\tau\epsilon}\right) \log(L).$$

Thus, we want to choose  $L$  such that it holds  $L \geq C \log^2(d) \log(CdR^2 + 1 + C_1\delta^2/\epsilon) \log\left(\frac{1}{\tau\epsilon}\right) \log L$ . Using the basic fact that for any  $a > 0$ ,  $x \geq 2a \log a \Rightarrow x \geq a \log x$  with  $a = C \log^2(d) \log(CdR^2 + 1 + C_1\delta^2/\epsilon) \log\left(\frac{1}{\tau\epsilon}\right)$ , it suffices to choose any  $L$  satisfying the following

$$L \geq C \log^2(d) \log\left(CdR^2 + 1 + C_1\frac{\delta^2}{\epsilon}\right) \log\left(\frac{1}{\tau\epsilon}\right) \log\left(\log^2(d) \log\left(CdR^2 + 1 + C_1\frac{\delta^2}{\epsilon}\right) \log\left(\frac{1}{\tau\epsilon}\right)\right).$$

We see that the choice in [Algorithm 3](#) satisfies this condition.

### C.3 Omitted Proofs from [Section 4.3](#)

**Lemma 4.14.** *Let  $\mathbf{A}, \mathbf{B}, \mathbf{B}_1, \dots, \mathbf{B}_p$  be symmetric  $d \times d$  matrices and define  $\mathbf{M} = \mathbf{B}^p$ ,  $\mathbf{M}_S = \prod_{i=1}^p \mathbf{B}_i$ . If  $\|\mathbf{B}_i - \mathbf{B}\|_2 \leq \delta \|\mathbf{B}\|_2$ , then  $\|\mathbf{M}_S - \mathbf{B}^p\|_2 \leq p\delta(1 + \delta)^p \|\mathbf{B}\|_2^p$ .*

*Proof.* We have the following:

$$\mathbf{B}^p - \prod_{i=1}^p \mathbf{B}_i = \sum_{i=0}^{p-1} \left( \left( \prod_{j=1}^i \mathbf{B}_j \right) \mathbf{B}^{p-i} - \left( \prod_{j=1}^{i+1} \mathbf{B}_j \right) \mathbf{B}^{p-i-1} \right) = \sum_{i=0}^{p-1} \left( \left( \prod_{j=1}^i \mathbf{B}_j \right) (\mathbf{B} - \mathbf{B}_{i+1}) \mathbf{B}^{p-i-1} \right).$$

Using that  $\|\mathbf{B}_j\|_2 \leq (1 + \delta)\|\mathbf{B}\|_2$ , we obtain the following bound:

$$\begin{aligned} \left\| \mathbf{B}^p - \prod_{i=1}^p \mathbf{B}_i \right\|_2 &\leq \sum_{i=0}^{p-1} \left\| \left( \prod_{j=1}^i \mathbf{B}_j \right) (\mathbf{B} - \mathbf{B}_{i+1}) \mathbf{B}^{p-i-1} \right\|_2 \\ &\leq \sum_{i=0}^{p-1} \left( \left( \prod_{j=1}^i \|\mathbf{B}_j\| \right) \|\mathbf{B} - \mathbf{B}_{i+1}\|_2 \|\mathbf{B}\|_2^{p-i-1} \right) \\ &\leq \sum_{i=0}^{p-1} \|\mathbf{B}\|_2^p (1 + \delta)^i \delta \leq p\delta(1 + \delta)^p \|\mathbf{B}\|_2^p. \end{aligned}$$

□

### C.4 Omitted Proofs from [Section 4.3.1](#)

**Lemma C.4.** *For any  $\delta, \tau \in (0, 1)$  and any distribution  $D$  on  $\mathbb{R}^d$  with mean  $\mu$  and covariance matrix  $\Sigma$ , there exists an estimator  $\hat{\mu}$  on  $n = O((\text{tr}(\Sigma)/\delta^2) \log(1/\tau))$  i.i.d. samples from  $D$ , such that  $\|\hat{\mu} - \mu\|_2 = O(\delta)$ . Moreover, this  $\hat{\mu}$  can be computed in time  $O(nd \log(1/\tau))$  and using memory  $O(d \log(1/\tau))$ .*

*Proof.* Let  $X_1, \dots, X_m$  be independent samples from  $D$ . We first show that the empirical mean  $Y := (1/m) \sum_{i=1}^m X_i$ , is  $\delta$ -accurate with constant probability.

$$\mathbf{E}[\|Y - \mu\|_2^2] = \sum_{j=1}^d \mathbf{E}[(Y_j - \mu_j)^2] = \frac{1}{m} \sum_{j=1}^d \Sigma_{jj} = \frac{\text{tr}(\Sigma)}{m}.$$



By Markov's inequality, we get that

$$\Pr[\|Y - \mu\|_2^2 > \delta^2] \leq \frac{\text{tr}(\Sigma)}{m\delta^2} \leq \frac{1}{20}, \quad (35)$$

where the last inequality is true if we choose  $m = 20\text{tr}(\Sigma)/\delta^2$ . Having Equation (35) at hand, the probability of success of the above estimator can be boosted to  $1 - \tau$  by using Claim 3.11. We use that claim with  $G$  being the distribution of  $Y$ ,  $B = G$ ,  $\epsilon = 1/20$  and  $R = \delta$ . This completes the proof.  $\square$

As a corollary of the above, we obtain the estimators  $\hat{\mu}_t$  of Algorithm 3.

**Lemma 4.11.** *In the setting of Algorithm 3, there exist estimators  $\hat{\mu}_t$  such that, with probability at least  $1 - \tau$ , for all  $t \in [K]$  we have that  $\|\hat{\mu}_t - \mu_t\|_2 \leq \delta/100$ . Furthermore, each  $\hat{\mu}_t$  can be computed on a stream of  $n = O\left(\frac{R^2}{\delta^2/\epsilon} \log(K/\tau) + \frac{d(1+\delta^2/\epsilon)}{\delta^2} \log(K/\tau)\right)$  independent samples from  $P_t$ , in time  $O(nd \log(K/\tau))$  and using memory  $O(d \log(K/\tau))$ .*

*Proof.* We use the estimator of Lemma C.4 with  $\tau/K$  in place of  $\tau$ . It remains to bound  $\text{tr}(\Sigma_t)$ . We have that  $d_{\text{TV}}(P_t, G) = 1 - O(\epsilon)$ , thus, by Fact 2.4 we can write  $P_t = (1 - \alpha)G_0 + \alpha B$ , with  $\alpha = O(\epsilon)$  and  $G_0(x) = h(x)G(x)/(\int h(x)G(x)dx)$  some weighted version of the inlier's distribution with  $\mathbf{E}_{X \sim G}[h(X)] = 1 - \alpha$  (same argument that we have used before in the proof of Lemma 2.11). We have that

$$\Sigma_t = (1 - \alpha)\Sigma_{G_0} + \alpha\Sigma_B + \alpha(1 - \alpha)(\mu_{G_0} - \mu_B)(\mu_{G_0} - \mu_B)^T.$$

Due to stability, the first term has  $\Sigma_{G_0} \preceq (1 + \delta^2/\epsilon)\mathbf{I}_d$ . For the second term we use that

$$\text{tr}(\Sigma_B) = \mathbf{E}_{X \sim B}[\text{tr}((X - \mu_B)(X - \mu_B)^T)] = \mathbf{E}_{X \sim B}[\|X - \mu_B\|_2^2] \leq O(R^2).$$

We also bound the trace of the last term by  $O(\epsilon R^2)$ . Therefore, we obtain that  $\text{tr}(\Sigma_t) \lesssim d(1 + \delta^2/\epsilon) + \epsilon R^2$ .  $\square$

## D Adaptive Choice of Upper Bound on Covariance

In this section, we show that a simple procedure can be used to make the algorithm adaptive to the scale of covariance (such a procedure is useful for some of our applications in Section 5).

As noted earlier, the definition of stability that we have used so far (Definitions 2.8 and 2.9) was designed for distributions with covariance matrix comparable to  $\mathbf{I}_d$ . In particular, if inliers satisfy  $\text{Cov}[X] \preceq \mathbf{I}_d$ , then our algorithms result in error  $O(\sqrt{\epsilon})$ . In many practical cases, some of which are encountered in Section 5, the inliers are much better concentrated, satisfying  $\text{Cov}[X] \preceq \sigma\mathbf{I}_d$ , with  $\sigma$  much smaller than 1. In that case, the optimal asymptotic error is  $\Theta(\sigma\sqrt{\epsilon})$ . If  $\sigma$  is known beforehand, then a simple preprocessing step allows our algorithms to obtain the error  $O(\sigma\sqrt{\epsilon})$ . We now describe a procedure using Lepski's method [Lep91, Bir01] that can adapt to the setting when  $\sigma$  is unknown. Concretely, we consider the task of robustly estimating the mean  $\mu$  of a distribution where inliers have bounded covariance,  $\text{Cov}[X] \preceq \sigma^2\mathbf{I}_d$ , but  $\sigma$  is unknown to the algorithm.

Let  $\text{RobustMean}(\tilde{\sigma}, \gamma)$  be any black-box robust mean estimation algorithm, where  $\tilde{\sigma}$  is a guess for an upper bound on the covariance of inliers (ideally, we would like to use  $\tilde{\sigma} = \sigma$ ) and  $\gamma$  is the probability of failure. The procedure below tries different values for  $\tilde{\sigma}$  in order to find a vector that

is as good as the output of RobustMean when run with the best choice of  $\tilde{\sigma} = \sigma$ . The assumption made here is that  $\sigma$  belongs in some known interval  $[A, B]$ .

As a small note, a more explicit notation would be  $\text{RobustMean}(S, \tilde{\sigma}, \gamma)$ , where  $S$  is the dataset used, but we omit  $S$  because this depends on the data-access model: If a streaming model is assumed, then  $S$  necessarily has to be different in each call of the algorithm, otherwise there is no need for using different datasets.

---

**Algorithm 7** Adaptive search for  $\sigma$

---

```

1: input:  $A, B, \gamma, r(\cdot)$ 
2: Denote  $\tilde{\sigma}_j := B/2^j$  for  $j = 0, 1, \dots, \log(B/A)$  and set  $\gamma' := \gamma/\log(B/A)$ .
3:  $J \leftarrow 0$ 
4:  $\hat{\mu}^{(0)} \leftarrow \text{RobustMean}(\tilde{\sigma}_0, \gamma')$ 
5: while  $\tilde{\sigma}_j \geq A$  and  $\|\hat{\mu}^{(j)} - \hat{\mu}^{(j-1)}\|_2 \leq r(\tilde{\sigma}_j) + r(\tilde{\sigma}_{j-1})$  for all  $j = 0, 1, \dots, J-1$  do
6:    $J \leftarrow J + 1$ .
7:    $\hat{\mu}^{(J)} \leftarrow \text{RobustMean}(\tilde{\sigma}_J, \gamma')$ 
8: end while
9:  $\hat{J} \leftarrow J - 1$ 
10: return  $\hat{\mu}^{(\hat{J})}$ 

```

---

**Theorem D.1.** *Let  $\mu \in \mathbb{R}^d$ ,  $A, B > 0$ ,  $\sigma \in [A, B]$ , and a non-decreasing function  $r : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ . Suppose that  $\text{RobustMean}(\tilde{\sigma}, \gamma)$  is a black-box algorithm which is guaranteed to return a vector  $\hat{\mu}$  such that  $\|\hat{\mu} - \mu\|_2 \leq r(\tilde{\sigma})$  with probability  $1 - \gamma$ , whenever  $\tilde{\sigma} \geq \sigma$ . Then, [Algorithm 7](#) returns  $\hat{\mu}^{(\hat{J})}$  such that, with probability at least  $1 - \gamma$ , we have that  $\|\hat{\mu}^{(\hat{J})} - \mu\|_2 \leq 3r(2\sigma)$ . Moreover, [Algorithm 7](#) calls  $\text{RobustMean}$   $O(\log(B/A))$  times with desired failure probability set to  $\gamma/\log(B/A)$  and using at most  $O(d \log(B/A))$  additional memory.*

*Proof.* For  $j = 0, 1, \dots, \log(B/A)$ , denote by  $\mathcal{E}_j$  the event that  $\|\hat{\mu}^{(j)} - \mu\|_2 \leq r(\tilde{\sigma}_j)$ . Let  $J$  be the index corresponding to the value of the unknown parameter  $\sigma$ , i.e.,  $\tilde{\sigma}_{J+1} \leq \sigma \leq \tilde{\sigma}_J$ . Conditioned on the event  $\bigcap_{j=0}^J \mathcal{E}_j$ , we have that  $\|\hat{\mu}^{(j)} - \mu\|_2 \leq r(\tilde{\sigma}_j)$  for all  $j = 0, 1, \dots, J$ . Using the triangle inequality, this gives that  $\|\hat{\mu}^{(J)} - \hat{\mu}^{(j)}\|_2 \leq r(\tilde{\sigma}_J) + r(\tilde{\sigma}_j)$ . This means that the stopping condition of [Line 5](#) is satisfied on round  $J$  and thus, if  $\hat{\mu}^{(\hat{J})}$  denotes the vector returned by the algorithm, we have that

$$\|\hat{\mu}^{(\hat{J})} - \hat{\mu}^{(J)}\|_2 \leq r(\tilde{\sigma}_{\hat{J}}) + r(\tilde{\sigma}_J) \leq 2r(\tilde{\sigma}_J) \leq 2r(2\sigma),$$

where the first inequality uses that the condition of [Line 5](#), the second uses that  $r$  is non-decreasing and  $\tilde{\sigma}_{\hat{J}} \leq \tilde{\sigma}_J$ , and the last one uses that  $J$  was defined to be such that  $\tilde{\sigma}_{J+1} \leq \sigma \leq \tilde{\sigma}_J$  so multiplying  $\sigma$  by 2 makes it greater than  $\tilde{\sigma}_J$ . Using the triangle inequality once more, we get  $\|\hat{\mu}^{(\hat{J})} - \mu\|_2 \leq 3r(2\sigma)$ . Finally, by union bound on the events  $\mathcal{E}_j$ , the probability of error is upper bounded by  $\sum_{j=0}^J \gamma' \leq \gamma$ . The additional memory requirement of this algorithm is to store  $\{\hat{\mu}_j : j \in \{0, \dots, \log(B/A)\}\}$ .  $\square$

We now state the implications that [Theorem D.1](#) has for [Algorithms 1](#) and [3](#), given in [Sections 3](#) and [4](#):

**Corollary D.2.** *Let  $A, B > 0$ . In the setting of [Corollary 4.3](#), let  $\sigma > 0$  be such that the scaled version  $S' = \{x/\sigma : x \in S\}$  of the dataset  $S$  is  $(C\epsilon, \delta)$ -stable with respect to  $\mu/\sigma$ . Assuming that  $\sigma \in [A, B]$ , there exists an algorithm that given  $S, \epsilon, \delta, \tau, A, B$  (but not  $\sigma$ ), accesses each point of  $S$  at most  $\text{polylog}(d, 1/\epsilon, 1/\tau, B/A)$  times, runs in time  $nd \text{polylog}(d, 1/\epsilon, 1/\tau, B/A)$ , uses additional*

memory  $d \text{polylog}(d, 1/\epsilon, 1/\tau, B/A)$ , and outputs a vector  $\hat{\mu}$  such that, with probability at least  $1 - \tau$ , it holds  $\|\mu - \hat{\mu}\|_2 = O(\sigma\delta)$ .

*Proof.* In order to use the search method of [Algorithm 7](#), we define the procedure  $\text{RobustMean}(\tilde{\sigma}, \gamma)$  to be the following:

- Let  $\tilde{S} = \{x/\tilde{\sigma} : x \in S\}$ .
- Let  $\tilde{\mu}$  be the vector found by the estimator of [Corollary 4.3](#) on  $\tilde{S}$  using  $\gamma$  for the desired probability of failure.
- Return  $\tilde{\sigma}\tilde{\mu}$ .

[Theorem D.1](#) with  $r(\tilde{\sigma}) = C'\sigma\delta$ , for a sufficiently large  $C' > 0$ , implies the correctness. In terms of resources used, [Algorithm 7](#) calls the robust mean estimation algorithm at most  $\log(B/A)$  times, and thus the running time gets multiplied by  $\log(B/A)$ . We also need to store one vector for each call, thus  $d \log(B/A)$  additional memory suffices.  $\square$

**Corollary D.3.** *Let  $A, B > 0$ . In the setting of [Theorem 4.2](#), let  $\sigma > 0$  be such that the distribution  $D'$  of the points  $X/\sigma$ ,  $X \sim D$  is  $(C\epsilon, \delta)$ -stable with respect to  $\mu$ . Assuming that  $\sigma \in [A, B]$ , there exists an algorithm that given*

$$n = O\left(R^2 \max\left(d, \frac{\epsilon}{\delta^2}, \frac{(1 + \delta^2/\epsilon)d}{\delta^2 R^2}, \frac{\epsilon^2 d}{\delta^4}, \frac{R^2 \epsilon^2}{\delta^2}, \frac{R^2 \epsilon^4}{\delta^6}\right) \text{polylog}\left(d, \frac{1}{\epsilon}, \frac{1}{\tau}, R, \frac{B}{A}\right)\right) \quad (36)$$

*samples in a stream according to the model of [Definition 1.1](#), and given the parameters  $\epsilon, \delta, \tau, A, B$  (but not  $\sigma$ ), runs in time  $nd \text{polylog}(d, 1/\epsilon, 1/\tau, R, B/A)$ , uses additional memory  $d \text{polylog}(d, 1/\epsilon, 1/\tau, R, B/A)$ , and returns a vector  $\hat{\mu}$  such that, with probability at least  $1 - \tau$ , it holds  $\|\mu - \hat{\mu}\|_2 = O(\sigma\delta)$ .*

Finally, we note that a similar search procedure can be used for designing algorithms that are adaptive to the parameter  $\delta$  when  $\sigma$  is known. However, we will not need this generalization for our applications.

## E Omitted Details from [Section 5](#)

### E.1 Proof Sketch of [Theorem 5.3](#)

We describe how [Algorithm 3](#) can be plugged in the algorithm of [\[CDGW19b\]](#). We outline the analysis and describe in more detail only the parts from [\[CDGW19b\]](#) that need to be changed. The algorithm is [Algorithm 1](#) from [\[CDGW19b\]](#), which remains unchanged. This uses [Algorithm 2](#) as a subroutine, which we replace by our estimator of [Algorithm 3](#).

Regarding the analysis, the proof in [\[CDGW19b\]](#) uses two claims that state correctness of the black-box robust mean estimator: [Lemma 3.4](#) and [Lemma 3.5](#). For our case, [Lemma 3.4](#) is replaced by our [Theorem 4.2](#) specialized to bounded covariance distributions (also see part 2 of [Theorem 1.3](#) which says that the sample complexity of [Algorithm 3](#) for that case is  $\tilde{O}(d^2/\epsilon)$ ).

[Lemma 3.5](#) in [\[CDGW19b\]](#) also holds when using our estimator. We restate this as a claim below and provide a proof:

**Claim E.1.** *Let  $D$  be a distribution supported on  $\mathbb{R}^d$  with unknown mean  $\mu^*$  and covariance  $\Sigma$ . Let  $0 < \gamma < 1$ ,  $0 < \epsilon < \epsilon_0$  for some universal constant  $\epsilon_0$  and  $\delta = O(\sqrt{\tau\epsilon} + \epsilon \log(1/\epsilon))$  for some*

$\tau = O(\sqrt{\epsilon})$ . Suppose that  $D$  has exponentially decaying tails and  $\Sigma$  is close to the identity matrix  $\|\Sigma - \mathbf{I}_d\|_2 \leq \tau$ . Denote  $R := \sqrt{(d/\epsilon)(1 + \delta^2/\epsilon)}$ . [Algorithm 3](#) uses

$$n = \tilde{O} \left( R^2 \max \left( d, \frac{\epsilon}{\delta^2}, \frac{(1 + \delta^2/\epsilon)d}{\delta^2 R^2}, \frac{\epsilon^2 d}{\delta^4}, \frac{R^2 \epsilon^2}{\delta^2}, \frac{R^2 \epsilon^4}{\delta^6} \right) \right) \quad (37)$$

samples drawn from  $D$  and outputs a hypothesis vector  $\hat{\mu}$  such that  $\|\hat{\mu} - \mu^*\|_2 = O(\delta)$ , with probability  $1 - \gamma$ . Moreover, this is done in  $nd$  polylog( $d, 1/\epsilon, 1/\gamma$ ) time and  $d$  polylog( $d, 1/\epsilon, 1/\gamma$ ) space.

*Proof.* Since  $D$  has exponentially decaying tails, we know that  $D$  is stable with respect to its mean  $\mu^*$  and covariance  $\Sigma \preceq O(1)\mathbf{I}_d$  with parameter  $\delta = O(\epsilon \log(1/\epsilon))$  (this follows from the tails of the distribution and [Definition 2.8](#)). That is, for any weight function  $w : \mathbb{R}^d \rightarrow [0, 1]$  with  $\mathbf{E}_{X \sim D}[w(X)] \geq 1 - \epsilon$  we have that

$$\|\mu_{w,D} - \mu\|_2 \leq \delta \quad \text{and} \quad \|\bar{\Sigma}_{w,D} - \Sigma\|_2 \leq \frac{\delta^2}{\epsilon}.$$

We claim that  $D$  is  $(\epsilon, O(\sqrt{\tau\epsilon} + \epsilon \log(1/\epsilon)))$ -stable in the sense of [Definition 2.8](#) (the difference from what written above is that [Definition 2.8](#) uses identity matrix in place of  $\Sigma$ ). This can be seen by using triangle inequality:

$$\|\bar{\Sigma}_{w,D} - \mathbf{I}_d\|_2 \leq \|\bar{\Sigma}_{w,D} - \Sigma\|_2 + \|\Sigma - \mathbf{I}_d\|_2 \leq \frac{1}{\epsilon} (\delta + \sqrt{\epsilon\tau})^2. \quad (38)$$

The proof is concluded by recalling the guarantee of [Algorithm 3](#) for  $(\epsilon, O(\sqrt{\tau\epsilon} + \epsilon \log(1/\epsilon)))$ -stable distributions and using [Claim 3.12](#) for the value of  $R$ .  $\square$

We also note that [\[CDGW19b\]](#) uses a fast matrix inversion and multiplication procedure for calculating the rotated versions  $Y = \hat{\Sigma}_i^{-1/2} X$  of the samples  $X$ . In our case, the run-time of our robust mean-estimation procedure exceeds that of these methods, thus we do not need to use them. We can instead use any numerically stable method that has running time up to  $\tilde{O}(d^6)$  and approximates the result within error poly( $\epsilon\kappa/d$ ) (see, e.g., [\[BGVKS20\]](#)). Finally, since [Claim E.1](#) is essentially used for the  $d^2$ -dimensional distributions of the points  $Y \otimes Y$ , we get the  $d^4$  factor in the final sample complexity, as well as the  $d^2$  factors in the time and space complexity.

## E.2 Omitted Proofs from Section 5.2

**Corollary 5.6.** *In the setting of [Theorem 5.5](#), suppose that the distribution of gradients satisfies  $\text{Cov}[\nabla f(\theta)] \preceq \sigma^2 \mathbf{I}_d$  with  $\sigma^2 = \alpha^2 \|\theta - \theta^*\|_2^2 + \beta^2$  for all  $\theta \in \Theta$ , where  $\alpha\sqrt{\epsilon} < \tau_\ell$ . Assume that the radius of the domain  $\Theta$ ,  $r := \max_{\theta \in \Theta} \|\theta\|_2$  is finite. There exists a single-pass streaming algorithm that given  $O(T(d^2/\epsilon) \log(1 + \alpha r/\beta) \text{polylog}(d, 1/\epsilon, T/\tau, 1 + \alpha r/\beta))$  samples, runs in time  $Tnd \text{polylog}(d, 1/\epsilon, T/\tau, 1 + \alpha r/\beta)$ , uses memory  $d \text{polylog}(d, 1/\epsilon, T/\tau, 1 + \alpha r/\beta)$ , and returns a vector  $\hat{\theta} \in \mathbb{R}^d$  such that  $\|\hat{\theta} - \theta^*\|_2 = O(\sqrt{\epsilon}\beta/(1 - \kappa))$  with probability at least  $1 - \tau$ .*

*Proof.* This follows by using the estimator of [Corollary D.3](#) in place of  $g(\cdot)$  in [Algorithm 5](#). The known bounds for  $\sigma$ ,  $A \leq \sigma \leq B$  are  $A = \beta$  and  $B = 2\alpha r + \beta$ , thus  $B/A \leq 1 + 2\alpha r/\beta$ . The distribution of the scaled gradients  $\frac{1}{\sigma} \nabla f(\theta)$  is  $(C\epsilon, O(\sqrt{\epsilon}))$ -stable. For these parameters,  $n$  from [Equation \(36\)](#) gives  $n = (d^2/\epsilon) \text{polylog}(d, 1/\epsilon, \tau', 1 + \alpha r/\beta)$ , where  $\tau'$  is the desired probability of failure for each call of the estimator. Setting  $\tau' = \tau/T$  ensures that the estimates of all rounds are successful with probability  $1 - \tau$ . Successful estimates of the gradients are within  $O(\sigma\delta) = O((\alpha\|\theta - \theta^*\|_2 + \beta)\sqrt{\epsilon})$  from the true one in Euclidean norm, thus in every round we have an  $(\sqrt{\epsilon}\alpha, \sqrt{\epsilon}\beta)$ -gradient estimation (in the sense of [Definition 5.4](#)). Finally, [Theorem 5.5](#) requires the condition  $\alpha\sqrt{\epsilon} < \tau_\ell$ . Assuming that this is true, that theorem concludes the proof.  $\square$

**Theorem 5.12** (Robust Logistic Regression; full version of [Theorem 1.6](#)). *Consider the logistic regression model of [Equation \(26\)](#) with the domain  $\Theta$  of the unknown regressor being the ball of radius  $r$ , for some universal constant  $r > 0$ , and suppose that [Assumption 5.10](#) holds. Assume that  $0 < \epsilon < \epsilon_0$  for a sufficiently small constant  $\epsilon_0$ . There is a single-pass streaming algorithm that uses  $n = (d^2/\epsilon)$  polylog  $(d, 1/\epsilon, 1/\tau)$  samples, runs in time  $nd$  polylog  $(d, 1/\epsilon, 1/\tau)$ , uses memory  $d$  polylog  $(d, 1/\epsilon, 1/\tau)$ , and returns a vector  $\hat{\theta} \in \mathbb{R}^d$  such that  $\|\hat{\theta} - \theta^*\|_2 = O(\sqrt{\epsilon})$  with probability at least  $1 - \tau$ .*

*Proof.* The algorithm is that of [Theorem 5.5](#) using the estimator of [Algorithm 3](#) in place of  $g(\cdot)$  in [Algorithm 5](#). The distribution of the gradients is  $(C\epsilon, O(\sqrt{\epsilon}))$ -stable because of [Lemma 5.11](#). For these stability parameters, a sufficient number of samples is  $(d^2/\epsilon)$  polylog  $(d, 1/\epsilon, T/\tau)$  (see [Equation \(8\)](#) with  $\delta = O(\sqrt{\epsilon})$  and  $R = O(\sqrt{d})$ ), where  $T$  is the number of iterations over which take a union bound. It thus remains to specify the parameters  $\tau_\ell, \tau_u, k, T$ .

Using [Assumption 5.10](#), we can calculate bounds on the parameters  $\tau_\ell, \tau_u$ . For  $\tau_\ell$ , let  $v$  be a unit vector from  $\mathbb{R}^d$ . Let the event  $\mathcal{E}_{v,\theta} := \{(v^T X)^2 \geq c_1 \text{ and } |\theta^T X| \leq 2rC^2/c_2\}$ , where  $c_1, c_2, C$  are the constants from [Assumption 5.10](#). The probability of the complement of this event is

$$\Pr[\mathcal{E}_{v,\theta}^c] \leq \Pr[(v^T X)^2 < c_1] + \Pr[|\theta^T X| > 2rC^2/c_2] \leq 1 - c_2 + c_2/2 \leq 1 - c_2/2,$$

where the first term is bounded using the anti-concentration property and the second is bounded using the concentration property along with Markov's inequality. Thus, using the formula of [Equation \(27\)](#) for the Hessian, we have that

$$v^T \nabla^2 \bar{f}(\theta) v \geq \Pr[\mathcal{E}_{v,\theta}] \mathbf{E}_{X \sim D_x} \left[ \frac{e^{\theta^T X}}{(1 + e^{\theta^T X})^2} (v^T X)^2 \mid \mathcal{E}_{v,\theta} \right] \geq 0.5c_2 \frac{e^{2rC^2/c_2}}{(1 + e^{2rC^2/c_2})^2} c_1.$$

Regarding the upper bound  $\tau_u$ , using the bounded covariance property we get that  $v^T \nabla^2 \bar{f}(\theta) v \leq C^2 \sup_{a \in \mathbb{R}} e^a / (1 + e^a)^2 = C^2/4$ . Therefore, we can choose the values

$$\tau_\ell = 0.5c_2 \frac{e^{2rC^2/c_2}}{(1 + e^{2rC^2/c_2})^2} c_1 \quad \text{and} \quad \tau_u = C^2/4,$$

for [Algorithm 5](#). The guarantees of our mean estimator imply that we have an  $(0, O(\sqrt{\epsilon}))$ -gradient estimator (in the sense of [Definition 5.4](#)). Regarding the value of  $\kappa$ , we use [Equation \(22\)](#) with  $a = 0$ . Since that  $\tau_\ell, \tau_u$  are positive constants, this means that  $\kappa$  is bounded away from 1. Therefore, we have that the factor  $1/(1 - \kappa)$  appearing in the final error ([Equation \(24\)](#)) is  $O(1)$  and the number of iterations from [Equation \(23\)](#) are upper bounded by  $T \lesssim \log_2(\|\theta_0 - \theta^*\|_2/\sqrt{\epsilon}) \lesssim \log(1/\epsilon)$ , where we used that the radius of the domain  $\Theta$  is  $r = O(1)$ .  $\square$

## F Bit Complexity of [Algorithm 3](#)

Until this point, we have assumed that our algorithms could save real numbers exactly in a single memory cell and perform calculations involving reals in  $O(1)$  time. Thus, by saying that [Algorithm 3](#) uses extra memory at most  $d$  polylog  $(d, R, 1/\epsilon, 1/\tau)$ , we meant that it needs to store only that many real numbers. We now describe how the algorithm would work in the most realistic word RAM model, where finite precision numbers can be stored in registers of predetermined word size and operations like addition, subtraction and multiplication are performed in  $O(1)$  time. We now show that the previous bound of  $d$  polylog  $(d, R, 1/\epsilon, 1/\tau)$ , worsened only by another poly-logarithmic factor, holds for the total number of bits that need to be stored. We begin by clarifying how the input is given to the algorithm.

**Definition F.1** (Single-Pass Streaming Model with Oracle Access for Real Inputs). *Let  $S$  be a fixed set of points in  $\mathbb{R}^d$ . The elements of  $S$  are revealed one at a time to the algorithm as follows: For each point of  $S$  that is about to be revealed, the algorithm is allowed to query as many bits as it wants from that point with whatever order it wants. The process then continues with the next point in the stream. Each point of  $S$  is presented only once to the algorithm in the aforementioned way.*

In the reminder of this section, we use the same notation as in [Theorem 4.2](#). We assume  $R \leq M$  and that  $\|\mu\|_2 \leq M$ , for some  $M = (d/\epsilon)^{\text{polylog}(d/\epsilon)}$  (otherwise, the estimation of the mean with extra memory of the order  $d \text{polylog}(d/\epsilon)$  becomes impossible). The modified algorithm for this model is the following: Every input point  $X$  is ignored if found to have norm greater than  $2M$ . Otherwise, it is deterministically rounded to an  $X'$  so that their difference  $X - X' := \eta(X)$  has norm at most  $\eta$ , for some  $\eta \leq O(\min\{\delta, \frac{\delta^2}{\epsilon R}, R\})$  (see below for more on this choice of  $\eta$ ). The exact same algorithm as [Algorithm 3](#) is run on these rounded points.

**Correctness** First, we note the rejection step removes less than an  $\epsilon$ -fraction of the input, thus the resulting distribution has not changed more than  $\epsilon$  in total variation distance from the original one. Moreover, the distribution of the rounded points has essentially the same stability property required by our theorem. Concretely, if we choose the rounding error  $\eta$  to be  $\eta = O(\min\{\delta, \frac{\delta^2}{\epsilon M}, M\})$ , then it can be seen ([Lemma F.2](#) below) that the distribution of the rounded points is  $(\epsilon, O(\delta))$ -stable and  $\Pr_{X'}[\|X' - \mu\|_2 = O(R)] \geq 1 - \epsilon$ , which are the only assumptions needed for [Algorithm 3](#) to provide an accurate estimate up to  $O(\delta)$  error.

**Lemma F.2.** *Fix  $0 < \epsilon < 1/2$  and  $\delta \geq \epsilon$ . Let  $G$  be an  $(\epsilon, \delta)$ -stable distribution with respect to some vector  $\mu \in \mathbb{R}^d$  and assume  $G$  is a distribution such that  $\|X - \mu\|_2 \leq M$  almost surely for some  $M > 0$ . For any deterministic function  $\eta : \mathbb{R}^d \rightarrow \mathbb{R}^d$  with  $\|\eta(x)\|_2 \leq \eta$  for all  $x$  in the support of  $G$ , if  $G'$  denotes the distribution of the points  $X' = X + \eta(X)$ , where  $X \sim G$ , then  $G'$  is  $(\epsilon, O(\delta + \eta + \sqrt{\epsilon\eta M}))$ -stable with respect to  $\mu$ .*

*Proof.* We check the two conditions for stability. Let a weight function  $w : \mathbb{R}^d \rightarrow [0, 1]$  with  $\mathbf{E}_{X \sim G}[w(X)] \geq 1 - \epsilon$  and let  $\delta' = O(\delta + \eta + \sqrt{\epsilon\eta M})$ . We have that

$$\begin{aligned} \|\mu_{w, G'} - \mu\|_2 &\leq \|\mu_{w, G'} - \mu_{w, G}\|_2 + \|\mu_{w, G} - \mu\|_2 \\ &\leq \left\| \int_{\mathbb{R}^d} (x + \eta(x)) \frac{w(x)G(x)}{\mathbf{E}_{X \sim G}[w(X)]} dx - \mu_{w, G} \right\|_2 + \delta \\ &\leq \left\| \int_{\mathbb{R}^d} \eta(x) \frac{w(x)G(x)}{\mathbf{E}_{X \sim G}[w(X)]} dx \right\|_2 + \delta \\ &\leq \eta + \delta \leq \delta', \end{aligned}$$

where first inequality uses the triangle inequality and the second one uses the stability of  $G$ . Regarding the second stability condition, we have the following:

$$\begin{aligned} \|\bar{\Sigma}_{w, G'} - \mathbf{I}_d\|_2 &\leq \|\bar{\Sigma}_{w, G'} - \bar{\Sigma}_{w, G}\|_2 + \|\bar{\Sigma}_{w, G} - \mathbf{I}_d\|_2 \\ &\leq \left\| \int_{\mathbb{R}^d} (x - \mu + \eta(x))(x - \mu + \eta(x))^T \frac{w(x)G(x)}{\mathbf{E}_{X \sim G}[w(X)]} dx \right\|_2 + \frac{\delta^2}{\epsilon} \\ &\leq \left\| \int_{\mathbb{R}^d} (x - \mu)\eta(x)^T \frac{w(x)G(x)}{\mathbf{E}_{X \sim G}[w(X)]} dx \right\|_2 + \left\| \int_{\mathbb{R}^d} (x - \mu)^T \eta(x) \frac{w(x)G(x)}{\mathbf{E}_{X \sim G}[w(X)]} dx \right\|_2 \\ &\quad + \left\| \int_{\mathbb{R}^d} \eta(x)\eta(x)^T \frac{w(x)G(x)}{\mathbf{E}_{X \sim G}[w(X)]} dx \right\|_2 + \frac{\delta^2}{\epsilon} \end{aligned}$$



$$\leq 2M\eta + \eta^2 + \frac{\delta^2}{\epsilon} \leq \frac{\delta'^2}{\epsilon},$$

where we used stability of  $G$ , triangle inequality and the bounds  $\|x - \mu\|_2 \leq M$ ,  $\|\eta(x)\|_2 \leq \eta$ .  $\square$

**Total Bits of Memory Used** In order for the differences  $X - X' := \eta(X)$  to have  $\|\eta(X)\|_2 \leq \eta$  for all  $X$ , it is sufficient to round every coordinate to absolute error  $O(\eta/\sqrt{d})$ . Recall that by our assumption on the a priori bound on the norm of the true mean and the way that we reject input points of large norm, we know that all points surviving will have norm at most  $2M$ . Thus, each coordinate of such points can be stored in a word of  $O(\log(Md/\eta))$  bits after being rounded to accuracy  $\eta$ . Therefore, each  $d$ -dimensional point that the algorithm will need to manipulate can be stored using  $d$  registers of size  $O(\log(Md/\eta))$ . However, we need to show that the results of all intermediate calculations can be calculated in low memory. We show the following result to this end:

**Claim F.3.** *In the context of [Theorem 4.2](#), given a stream of  $d$ -dimensional points, where each coordinate has bit complexity  $B$ , [Algorithm 3](#) can be implemented in a word RAM machine using  $d$  polylog( $d, 1/\epsilon, 1/\tau, R$ ) many of registers of size  $B$  polylog( $d, 1/\epsilon, 1/\tau, R$ ).*

*Proof Sketch.* Multiplying two numbers of bit complexity  $B_1$  and  $B_2$  may result in bit complexity  $B_1 + B_2$ . Adding  $k$  numbers of bit complexity  $B$ , may make the resulting bit complexity  $B + \log(k)$ . We need to check that every step of the algorithm performs calculations that cannot cause the bit complexity to grow by more than poly-log factors.

[Line 8](#) performs only comparisons and counting. Regarding [Line 26](#): As pointed out at the beginning of [Section 4](#), the vector  $v_{t,j} \leftarrow \widehat{\mathbf{M}}_t z_{t,j}$  is calculated by multiplying  $z_{t,j}$  by  $\widehat{\mathbf{B}}_{t,k}$  for  $k = 1, \dots, \log d$  iteratively. Consider a single iteration, say the first one. Performing  $\widehat{\mathbf{B}}_{t,k} z_{t,j}$  involves calculating  $\frac{1}{n} (\sum_x x x^T) z_{t,j}$  (see [Section 4.3.2](#)), which can be done as  $\frac{1}{n} \sum_x x (x^T z_{t,j})$ , i.e., calculating the inner products  $x^T z_{t,j}$  first). A single inner product of that form is just a sum of  $d$  numbers of bit complexity  $B$  with appropriate signs, thus the bit complexity increases only by  $O(\log d)$ . Finally, multiplying by  $x$  and taking the mean over for all of the  $x$ 's can add only another  $O(B + \log(n))$ . Since the number of iterations of such calculations is  $\log d$ , the final result has the claimed bit complexity.

Regarding the Downweighting filter ([Algorithm 4](#)), there are a couple of places where the weights  $w_t$  are involved in calculations. We note that [Algorithm 2](#) stores only the counts  $\ell_t$ , which fit in registers of size  $\log(\ell_{\max}) = \text{polylog}(d, 1/\epsilon, R)$ . These counts are used to calculate  $w_t(x)$  as  $w_t(x) = \prod_{t' \leq t} (1 - \tilde{\tau}_{t'}(x)/r)^{\ell_{t'}}$  whenever there is such a need. An exact calculation would require operations of the form  $x^y$ , for some  $x \in [0, 1]$  and  $y \in [\ell_{\max}]$ , i.e., exponentiation of a real number. In fact, as we will show later on, instead of calculating  $w_t(x)$  with perfect accuracy, it suffices to use an approximate value of  $w_t(x) \pm \eta$  for some error  $|\eta| < \text{poly}(1/d, 1/R, \epsilon, \tau)^{\log d}$ . This will allow us to calculate a good enough approximation in  $\text{polylog}(d, R, 1/\epsilon, 1/\tau)$  bits as follows: we can use exponentiation by squaring algorithm for calculating  $w_t(x)$ 's and round the result in each step to make it fit into our registers. We first explain this in more detail below.

**Claim F.4.** *Let  $x \in [0, 1]$ ,  $y \in \mathbb{Z}_+$ , and assume that both  $x, y$  have bit complexity  $B$ . The power  $x^y$  can be calculated up to a rounding error of  $2^{-\Omega(B)}$  in the word RAM model that uses registers of size  $2B$ . Furthermore, this can be done in  $O(B)$  standard arithmetic operations.*

*Proof.* We can use exponentiation by squaring: This consists of writing  $x$  in binary as  $b_k \cdots b_0$  for  $k = \log y$  and calculating the sequence  $r_{k+1}, \dots, r_0$  as  $r_{k+1} = 1$ ,  $r_i = r_{k+1}^2 x^{b_i}$  for  $i = k, \dots, 0$ . We assume every  $r_i$  gets rounded to  $2B$  bits. Because of the rounding, we incur error  $2^{-2B}$  in each



round. However, the error of the previous rounds gets amplified, since the result of that round (true value plus error) gets squared. We consider one such iteration to see how that sequence of errors grows: In the  $t$ -th iteration, let  $\text{res}_{t-1}$  denote the true result (before rounding) from the previous round and  $\eta_t$  the rounding error of that round (i.e.,  $r_t = \text{res}_t + \eta_t$ ). Then, we have that

$$\text{res}_t + \eta_t := (\text{res}_{t-1} + \eta_{t-1})^2 + 2^{-2B} \leq \text{res}_{t-1}^2 + \eta_{t-1}^2 + 2\eta_{t-1} + 2^{-2B},$$

where w.l.o.g. we assume that  $\text{res}_t \leq 1$  always. Thus, the rounding error grows as  $\eta_t \leq \eta_{t-1}^2 + 2\eta_{t-1} + 2^{-2B} \leq 3\eta_{t-1} + 2^{-2B}$ . In the first round, we start with rounding error of  $2^{-2B}$ . Thus, after  $k = \log(y) = B$  rounds, the final error is  $\eta_k \leq 2^{-\Omega(B)}$ .  $\square$

We continue with examining how fine approximations for  $w_t$  are needed. First, in [Section 4.3.2](#), we use the estimator  $\widehat{W}_t = \mathbf{E}_{X \sim \mathcal{U}(S_0)}[w_t(X)]$ , which we require to be  $\eta$ -close to  $\mathbf{E}_{X \sim P}[w_t(X)]$  ([Equation \(16\)](#)) for some  $\eta > \text{poly}(1/d, 1/R, \epsilon, \tau)$ . Therefore, when calculating  $w_t$ , it suffices to round the intermediate results to error  $\eta$ . This would mean using [Claim F.4](#) with  $B = O(\log((1/d, 1/R, \epsilon, \tau)))$ .

Second, the weights  $w_t$  are also used in evaluating the stopping condition of the Downweighting filter. [Line 3](#) of that filter is implemented using the estimator of [Lemma 4.16](#). As it can be seen in [Equation \(19\)](#), it suffices to use rounded versions of  $w_t$  in  $(1/n) \sum_{i=1}^N w_t(X_i) \tilde{\tau}_t(X_i)$ , as long as it does not change the result by an additive factor of  $c\widehat{\lambda}_t \|\mathbf{U}_t\|_F^2$ , for a small constant  $c$ . Since  $\tilde{\tau}_t(X_i) = O(dR^{2+4\log d})$  ([Equation \(1\)](#)) and  $\widehat{\lambda}_t \|\mathbf{U}_t\|_F^2 > (\delta^2/\epsilon)^{\Theta(\log d)}$  (otherwise, the algorithm has terminated), we can again round  $w_t$  up to error  $\text{poly}(1/d, 1/R, \epsilon)^{\log(d)}$ . This means that the results of these calculations fit into  $\text{polylog}(d, R, 1/\epsilon, R)$  bits.

Finally, there are two places in [Algorithm 3](#) where we need to simulate samples from the weighted distribution  $P_{w_t}$ : (i) [Line 19](#), whose implementation is outlined in [Section 4.3](#) and (ii) [Line 35](#). We focus on the first one since the argument for the other case is identical. To simulate  $P_{w_t}$ , we use rejection sampling, as described at the beginning of [Section 4.3](#), with the only difference that we use the rounded versions of the weights  $w_t$ . We are thus essentially simulating samples from a slightly different distribution  $P_{\widehat{w}_t}$ . However, this is close to  $P_{w_t}$  in total variation distance, as shown below.

**Claim F.5.** *Let  $P$  be a distribution on  $\mathbb{R}^d$  and let  $P_w$  denote the weighted version of  $P$  according to the function  $w : \mathbb{R}^d \rightarrow [0, 1]$ , i.e.,  $P_w(x) = w(x)P(x) / \int_{\mathbb{R}^d} w(x)P(x)dx$ . For any  $w, \widehat{w} : \mathbb{R}^d \rightarrow [0, 1]$  such that  $\int_{\mathbb{R}^d} w(x)P(x)dx \geq 1/2$  and  $\sup_{x \in \mathbb{R}^d} |\widehat{w}(x) - w(x)| \leq \xi$  with  $\xi \leq 1/8$ , it holds that  $d_{\text{TV}}(P_{\widehat{w}}, P_w) \leq 8\xi$ .*

*Proof.* First, letting the normalization factors  $\widehat{C} := \int_{\mathbb{R}^d} \widehat{w}(x)P(x)dx$  and  $C := \int_{\mathbb{R}^d} w(x)P(x)dx$ , we note that  $|C - \widehat{C}| \leq \xi$ . Letting  $\Delta w(x) := \widehat{w}(x) - w(x)$  and  $\Delta C := \widehat{C} - C$ , we have that

$$\begin{aligned} d_{\text{TV}}(P_{\widehat{w}}, P_w) &= \frac{1}{2} \int_{\mathbb{R}^d} |P_{\widehat{w}}(x) - P_w(x)| dx = \frac{1}{2} \int_{\mathbb{R}^d} \left| \frac{w(x) + \Delta w(x)}{C + \Delta C} - \frac{w(x)}{C} \right| P(x) dx \\ &\leq \frac{1}{2} \int_{\mathbb{R}^d} \left| \frac{Cw(x) + C\Delta w(x) - Cw(x) - \Delta Cw(x)}{C^2 + C\Delta C} \right| P(x) dx \\ &\leq 4 \int_{\mathbb{R}^d} |C\Delta w(x) - \Delta Cw(x)| P(x) dx \leq 8\xi, \end{aligned}$$

where in the last line, we first use that  $C^2 + C\Delta C \geq 1/4 - \xi \geq 1/8$  (since  $1/2 \leq C \leq 1$  and  $0 \leq \xi \leq 1/8$ ) and then we use that  $|C\Delta w(x) - \Delta Cw(x)| \leq |\Delta w(x)| + |\Delta C| \leq 2\xi$ .  $\square$

As  $N$  (more than one) samples are drawn from  $P_t$  in the  $t$ -th iteration (see [Section 4.3](#)), we require the joint distribution of these  $N$  samples from  $P_{w_t}$  and  $P_{\widehat{w}_t}$  to be within total variation  $\tau$  (the probability under which the conclusion of [Lemma 4.13](#) holds true). This bound on the total

variation distance implies that [Lemma 4.13](#) continues to hold for  $P_{\hat{w}_i}$  with an additional failure probability of  $\tau$ . To do this, we use [Claim F.5](#) with  $\xi = \Theta(\tau/N)$ , which means that these rounded weights have bit complexity  $\Theta(\log \xi) = \text{polylog}(d, R, 1/\epsilon, 1/\tau)$ .  $\square$