# A Supplementary Material: Proofs

## A.1 Missing Proofs from Section 3

To establish Lemmas 3.1 and 3.3, we build on the work of (VWDP+22) and (DLPES02). We first introduce the necessary notion and definitions.

We recall the following notation from the main body of the paper. We use $\mathrm{diam}_\infty$ to indicate the diameter of a set relative to the $\ell_\infty$ norm and $\overline{B}^\infty_\varepsilon(\vec{p})$ to represent the closed ball of radius $\varepsilon$ centered at $\vec{p}$ relative to the $\ell_\infty$ norm. That is, in $\mathbb{R}^d$ we have $\overline{B}^\infty_\varepsilon(\vec{p}) = \prod_{i=1}^d [p_i - \varepsilon, p_i + \varepsilon]$.

Lemma 3.1 is based on the construction of certain geometric partitions of $\mathbb{R}^d$ called *secluded partitions*. Such partitions naturally induce deterministic rounding schemes which we use in the proof.

Let $\mathcal{P}$ be a partition of $\mathbb{R}^d$. For a point $\vec{p} \in \mathbb{R}^d$, let $N_\varepsilon(\vec{p})$ denote the set of members of the partitions that have a non-empty intersection with the $\varepsilon$-ball around $\vec{p}$. That is,

$$N_\varepsilon(\vec{p}) = \{X \in \mathbb{P} \mid \overline{B}^\infty_\varepsilon(\vec{p})) \cap X \neq \emptyset\}$$

**Definition A.1** (Secluded Partition)**.** Let $\mathcal{P}$ be a partition of $\mathbb{R}^d$. We say that $\mathcal{P}$ is $(k, \varepsilon)$-*secluded*, if for every point $\vec{p} \in \mathbb{R}^d$, $|N_{\varepsilon(\vec{p})}| \leq k$.

The following theorem from (VWDP+22) gives an explicit construction of a secluded partition with desired parameters where each member of the partition is a hypercube. For such partitions, we use the following notation. For every $\vec{p} \in \mathbb{R}^d$, if $X \in \mathbb{P}$, then the *representative of* $\vec{p}$, $rep(\vec{p})$, is the center of the hypercube $X$.

**Theorem A.2.** *For each $d \in \mathbb{N}$, there exists a $(d+1, \frac{1}{2d})$-secluded partition, where each member of the partition is a unit hypercube. Moreover, the partition is efficiently computable: Given an arbitrary point $\vec{x} \in \mathbb{R}^d$, its representative can be computed in time polynomial in $d$.*

### A.1.1 Proof of Lemma 3.1

**Lemma A.3** (Lemma 3.1)**.** *Let $d \in \mathbb{N}$ and $\varepsilon \in (0, \infty)$. Let $\varepsilon_0 = \frac{\varepsilon}{2d}$. There is an efficiently computable function $f_\varepsilon : \mathbb{R}^d \to \mathbb{R}^d$ with the following two properties:*

1. *For any $x \in \mathbb{R}^d$ and any $\hat{x} \in \overline{B}^\infty_{\varepsilon_0}(x)$ it holds that $f_\varepsilon(\hat{x}) \in \overline{B}^\infty_\varepsilon(x)$.*
2. *For any $x \in \mathbb{R}^d$ the set $\left\{ f_\varepsilon(\hat{x}) \colon \hat{x} \in \overline{B}^\infty_{\varepsilon_0}(x) \right\}$ has cardinality at most $d+1$.*

As explained in the main body, intuitively, item (1) states that if $\hat{x}$ is an $\varepsilon_0$-approximation of $x$, then $f_\varepsilon(\hat{x})$ is an $\varepsilon$-approximation of $x$, and item (2) states that $f_\varepsilon$ maps every $\varepsilon_0$-approximation of $x$ to one of at most $d+1$ possible values.

*Proof.* A high-level idea behind the proof is explained in Figure 1. We scale the $(d+1, \frac{1}{2d})$-secluded unit hypercube partition by $\varepsilon$ so that each partition member is a hypercube with side length $\varepsilon$. Now, for a point $x$, the ball $\overline{B}^\infty_{\varepsilon_0}(x)$ intersects at most $d+1$ hypercubes. Consider a point $\hat{x}_1 \in \overline{B}^\infty_{\varepsilon_0}(x)$, it is rounded to $c_1$ (center of the hypercube it resides in). Note that $c_1$ lies in the ball of radius $\varepsilon$ around $x$, this is because distance from $x$ to $\hat{x}_1$ is atmost $\varepsilon_0$ and the disance from $\hat{x}_1$ to $c_1$ is at most $\varepsilon/2$. By triangle inequality $c_1$ belongs to $\overline{B}^\infty_\varepsilon(x)$. We now provide formal proof.

Let $\mathcal{P}$ be the $(d+1, \frac{1}{2d})$-secluded partition given by Theorem A.2. Thus $\mathcal{P}$ consists of unit cubes $[0,1)^d$ with the property that for any point $\vec{p} \in \mathbb{R}^d$ the closed cube of side length $1/d$ centered at $\vec{p}$ (i.e. $\overline{B}^\infty_{\frac{1}{2d}}(\vec{p})$) intersects at most $d+1$ members/cubes of $\mathcal{P}$.

We first define a *rounding* function $f : \mathbb{R}^d \to \mathbb{R}^d$ as follows: for every $x \in \mathbb{R}^d$, $f(x) = rep(x)$.

Observe that the rounding function $f$ has the following two properties. (1) For every $x \in \mathbb{R}^d$, $\|f(x) - x\|_\infty \leq \frac{1}{2}$. This is because every point $x$ is mapped via $f$ to its representative, which is the center of the unit cube in which it lies. (2) For any point $\vec{p} \in \mathbb{R}^d$, the set $\left\{ f(x) \colon x \in \overline{B}^\infty_{\frac{1}{2d}}(\vec{p}) \right\}$ has

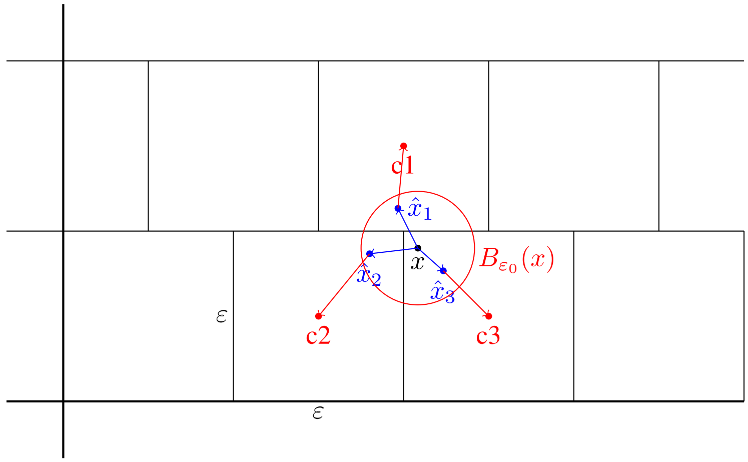


Figure 1: Illustration of proof of Lemma 3.1 for $d = 2$.

cardinality at most $d+1$. This is because $\overline{B}^{\infty}_{\frac{1}{2d}}(\vec{p})$ intersects at most $d+1$ hypercubes of $\mathcal{P}$ and for every hypercube $X$, all the points in $X$ are mapped to its center by $f$.

The function $f$ only gives an $\frac{1}{2}$-approximation guarantee. In order to get any $\varepsilon$-approximation guarantee, we scale $f$ appropriately. $f_\varepsilon$ is this *scaled* version of $f$.

Define the function $f_\varepsilon : \mathbb{R}^d \to \mathbb{R}^d$ as follows: for every $\hat{x} \in \mathbb{R}^d$, $f_\varepsilon(\hat{x}) = \varepsilon \cdot f(\frac{1}{\varepsilon}\hat{x})$. The efficient computability of $f_\varepsilon$ comes from the efficient computability of $f$.

We first establish that $f_\varepsilon$ has property (1) stated in the Lemma. Let $x \in \mathbb{R}^d$ and $\hat{x} \in \overline{B}^{\infty}_{\varepsilon_0}(x)$. Then we have the following (justifications will follow):

$$\begin{aligned}\left\|\tfrac{1}{\varepsilon} \cdot f_\varepsilon(\hat{x}) - \tfrac{1}{\varepsilon}x\right\|_\infty &= \left\|f(\tfrac{1}{\varepsilon}\hat{x}) - \tfrac{1}{\varepsilon}x\right\|_\infty \\ &\leq \left\|f(\tfrac{1}{\varepsilon}\hat{x}) - \tfrac{1}{\varepsilon}\hat{x}\right\|_\infty + \left\|\tfrac{1}{\varepsilon}\hat{x} - \tfrac{1}{\varepsilon}x\right\|_\infty \\ &\leq \left\|f(\tfrac{1}{\varepsilon}\hat{x}) - \tfrac{1}{\varepsilon}\hat{x}\right\|_\infty + \tfrac{1}{\varepsilon}\|\hat{x} - x\|_\infty \\ &\leq \tfrac{1}{2} + \tfrac{1}{\varepsilon}\varepsilon_0 \\ &= \tfrac{1}{2} + \tfrac{1}{2d} \leq 1\end{aligned}$$

The first line is by the definition of $f_\varepsilon$, the second is the triangle inequality, the third is scaling of norms, the fourth uses the property of $f$ that points are not mapped a distance more than $\frac{1}{2}$ along with the hypothesis that $\hat{x} \in \overline{B}^{\infty}_{\varepsilon_0}(x)$, the fifth uses the definition of $\varepsilon_0$, and the sixth uses the fact that $d \geq 1$.

Scaling both sides by $\varepsilon$ and using the scaling of norms, the above gives us $\|f_\varepsilon(\hat{x}) - x\|_\infty \leq \varepsilon$ which proves property (1) of the lemma.

To see that $f_\varepsilon$ has property (2), let $x \in \mathbb{R}^d$. We have the following set equalities:

$$\begin{aligned}\left\{f_\varepsilon(\hat{x}) \colon \hat{x} \in \overline{B}^{\infty}_{\varepsilon_0}(x)\right\} &= \left\{\varepsilon \cdot f(\tfrac{1}{\varepsilon}\hat{x}) \colon \hat{x} \in \overline{B}^{\infty}_{\varepsilon_0}(x)\right\} \\ &= \left\{\varepsilon \cdot f(a) \colon a \in \overline{B}^{\infty}_{\frac{1}{\varepsilon}\varepsilon_0}(x)\right\} \\ &= \left\{\varepsilon \cdot f(a) \colon a \in \overline{B}^{\infty}_{\frac{1}{2d}}(x)\right\}\end{aligned}$$

The first line is from the definition of $f_\varepsilon$, the second is from re-scaling, and the third is from the definition of $\varepsilon_0$.

Because $f$ takes on at most $d+1$ distinct values on $\overline{B}^{\infty}_{\frac{1}{2d}}(x)$, the set has cardinality at most $d+1$ which proves property (2) of the lemma. □

### A.1.2 Proof of Lemma 3.2

**Lemma A.4** (Lemma 3.2). *Let $d \in \mathbb{N}$, $\varepsilon_0 \in (0, \infty)$ and $0 < \delta < 1$. There is an efficiently computable deterministic function $f : \{0,1\}^\ell \times \mathbb{R}^d \to \mathbb{R}^d$ with the following property. For any $x \in \mathbb{R}^d$,*

$$\Pr_{r \in \{0,1\}^\ell} \left[ \exists x^* \in \overline{B}^\infty_\varepsilon(x) \ \ \forall \hat{x} \in \overline{B}^\infty_{\varepsilon_0}(x) : f(r, \hat{x}) = x^* \right] \geq 1 - \delta$$

*where $\ell = \lceil \log \frac{d}{\delta} \rceil$ and $\varepsilon = (2^\ell + 1)\varepsilon_0 \leq \frac{2\varepsilon_0 d}{\delta}$.*

*Proof.* Partition each coordinate of $\mathbb{R}^d$ into $2\varepsilon_0$-width intervals. The algorithm computing the function $f$ does the following simple randomized rounding:

*The function $f$* : Choose a random integer $r \in \{1 \ldots 2^\ell\}$. Note that $r$ can be represented using $\ell$ bits. Consider the $i^{th}$ coordinate of $\hat{x}$ denoted by $\hat{x}[i]$. Round $\hat{x}[i]$ to the nearest $k * (2\varepsilon_0)$ such that $k \mod 2^\ell \equiv r$.

Now we will prove that $f$ satisfies the required properties.

First, we prove the approximation guarantee. Let $x'$ denote the point in $\mathbb{R}^d$ obtained after rounding each coordinate of $\hat{x}$. The $k$s satisfying $k \mod 2^\ell \equiv r$ are $2^\ell \cdot 2\varepsilon_0$ apart. Therefore, $x'[i]$ is rounded by at most $2^\ell \varepsilon_0$. That is, $|x'[i] - \hat{x}[i]| \leq 2^\ell \varepsilon_0 = \frac{\varepsilon_0 d}{\delta}$ for every $i$, $1 \leq i \leq d$. Since $\hat{x}$ is an $\varepsilon_0$-approximation (i.e. each coordinate $\hat{x}[i]$ is within $\varepsilon_0$ of the true value $x[i]$), then each coordinate of $x'$ is within $(2^\ell + 1)\varepsilon_0$ of $x[i]$. Therefore $x'$ is a $(2^\ell + 1)\varepsilon_0$-approximation of $x[i]$. Thus $x' \in \overline{B}^\infty_\varepsilon(x)$ for any choice of $r$.

Now we establish that for $\geq 1 - \delta$ fraction of $r \in \{1 \ldots 2^\ell\}$, there exists $x^*$ such every $\hat{x} \in \overline{B}^\infty_{\varepsilon_0}(x)$ is rounded $x^*$. We argue this with respect to each coordinate and apply the union bound. Fix an $x$ and a coordinate $i$. For $x[i]$, consider the $\varepsilon_0$ interval around it.

Consider $r$ from $\{1 \ldots 2^\ell\}$. When this $r$ is chosen, then we round $\hat{x}[i]$ to the closest $k*(2\varepsilon_0)$ such that $k \mod 2^\ell \equiv r$. Let $p^r_1, p^r_2, \ldots p^r_j \ldots$ be the set of such points: more precisely $p_j = (j2^l + r) * 2\varepsilon_0$. Note that $\hat{x}[i]$ is rounded to an $p_j$ to some $j$. Let $m^r_j$ denote the midpoint between $p^r_j$ and $p^r_{j+1}$. I.e, $m^r = (p^r_j + p^r_{j+1})/2$ We call $r$ 'bad' for $x[i]$ if $x[i]$ is close to some $m^r_j$. That is, $r$ is 'bad' if $|x[i] - m^r_j| < \varepsilon_0$. Note that for a bad $r$ there exists $\hat{x_1}$ and $\hat{x_2}$ in $\overline{B}^\infty_{\varepsilon_0}(x)$ so that their $i^{th}$ coordinates are round to $p^r_j$ and $p^r_{j+1}$ respectively. The crucial point is that if $r$ is 'not bad' for $x[i]$, then for every $x' \in \overline{B}^\infty_{\varepsilon_0}(x)$, there exists a canonical $p^*$ such that $x'[i]$ is rounded to $p^*$. We call $r$ bad for $x$, if $r$ is bad for $x$, if there exists at least one $i$, $1 \leq i \leq d$ such that $r$ is bad for $x[i]$. With this, it follows that if $r$ is not bad for $x$, then there exists a canonical $x^*$ such that every $x' \in \overline{B}^\infty_{\varepsilon_0}(x)$ is rounded to $x^*$.

With this, the goal is to bound the probability that a randomly chosen $r$ is bad for $x$. For this, we first bound the probability that $r$ is bad for $x[i]$. We will argue that there exists almost one bad $r$ for $x[i]$. Suppose that there exist two numbers $r_1$ and $r_2$ that are both bad for $x[i]$. This means that $|x[i] - m^{r_1}_{j_1}| < \varepsilon_0$ and $|x[i] - m^{r_2}_{j_2}| < \varepsilon_0$ for some $j_1$ and $j_2$. Thus by triangle inequality $|m^{r_1}_{j_1} - m^{r_2}_{j_2}| < 2\varepsilon_0$. However, note that $|p^{r_1}_{j_1} - p^{r_2}_{j_2}|$ is $|(j_1 - j_2)2^\ell + (r_1 - r_2)|2\varepsilon_0$. Since $r_1 \neq r_2$, this value is at least $2\varepsilon_0$. This implies that the absolute value of difference between $m^{r_1}_{j_1}$ and $m^{r_2}_{j_2}$ is at least $2\varepsilon$ leading to a contradiction.

Thus the probability that $r$ is bad for $x[i]$ is at most $\frac{1}{2^\ell}$ and by the union bound the probability that $r$ is bad for $x$ is at most $\frac{d}{2^\ell} \leq \delta$. This completes the proof. □

### A.1.3 Proof of Lemma 3.3

The proof, which is based on Sperner/KKM Lemma, is present in (VWDP+22). Since our setting is slightly different, for completeness we give a proof.

We first introduce the necessary definitions and notation.

**Definition A.5** (Sperner/KKM Coloring). Let $d \in \mathbb{N}$ and $V = \{0,1\}^d$ denote a set of colors (which is exactly the set of vertices of $[0,1]^d$ so that colors and vertices are identified). Let $\chi : [0,1]^d \to V$ be a coloring function such that for any face $F$ of $[0,1]^d$, for any $x \in F$, it holds that $\chi(x) \in V(F)$

where $V(F)$ is the vertex set of $F$ (informally, the color of $x$ is one of the vertices in the face $F$). Such a function $\chi$ will be called a Sperner/KKM coloring.

**Theorem A.6** (Cubical Sperner/KKM lemma ([DLPES02])). *Let $d \in \mathbb{N}$ and $V = \{0,1\}^d$ and $\chi : [0,1]^d \to V$ be a Sperner/KKM coloring. Then there exists a subset $J \subset V$ with $|J| = d+1$ and a point $\vec{y} \in [0,1]^d$ such that for all $j \in J$, $\vec{y} \in \overline{\chi^{-1}(j)}$ (informally, $\vec{y}$ is in the closure of at least $d+1$ different colors).*

We will need to relate partitions to Sperner/KKM coloring so that we can use the Sperner/KKM Lemma.

For any co-ordinate $i$, let $\pi$ denote the standard projection map: $\pi_i : [0,1]^d \to [0,1]$ defined by $\pi_i(x) \stackrel{\text{def}}{=} x_i$ which maps $d$-dimensional points to the $i^{th}$ coordinate value. We extend this to sets: $\pi_i(X) = \{\pi_i(x) : x \in X\}$.

**Definition A.7** (Non-Spanning partition)**.** Let $d \in \mathbb{N}$ and $\mathcal{P}$ be a partition of $[0,1]^d$. We say that $\mathcal{P}$ is a non-spanning partition if it holds for all $X \in \mathcal{P}$ and for all $i \in [d]$ that either $\pi_i(X) \not\ni 0$ or $\pi_i(X) \not\ni 1$ (or both).

Next, we state a lemma that asserts that for any non-spanning partition, there is a Sperner/KKM coloring that respects the partition: that is every member gets the same color.

**Lemma A.8** (Coloring Admission)**.** *Let $d \in \mathbb{N}$, and $V = \{0,1\}^d$, and $\mathcal{P}$ a non-spanning partition of $[0,1]^d$. Then there exists a Sperner/KKM coloring $\chi : [0,1]^d \to V$ such that for every $X \in \mathcal{P}$, for every $x, y \in X$, $\chi(x) = \chi(y)$.*

Now we are ready to prove the Lemma 3.3.

**Lemma A.9.** *(Lemma 3.3) Let $\mathcal{P}$ be a partition of $[0,1]^d$ such that for each member $X \in \mathcal{P}$, it holds that $\operatorname{diam}_\infty(X) < 1$. Then there exists $\vec{p} \in [0,1]^d$ such that for all $\delta > 0$ we have that $\overline{B}_\delta^\infty(\vec{p})$ intersects at least $d+1$ members of $\mathcal{P}$.*

*Proof.* Consider an arbitrary $X \in \mathcal{P}$. For each coordinate, $i \in [d]$, the set $\{x_i : x \in X\}$ does not contain both $0$ and $1$ (if it did, this would demonstrate two points in $X$ that are $\ell_\infty$ distance at least $1$ apart and contradict that $\operatorname{diam}_\infty(X) < 1$). Thus, $\mathcal{P}$ is by definition a non-spanning partition of $[0,1]^d$. Since $\mathcal{P}$ is non-spanning, by Lemma A.8, there is a Sperner/KKM coloring where each point of $[0,1]^d$ can be assigned one of $2^d$-many colors and for any member $X \in \mathcal{P}$, all points in $X$ are assigned the same color. By Lemma A.6, there is a point $\vec{p} \in [0,1]^d$ such that $\vec{p}$ belongs to the closure of at least $d+1$ colors. Since every point of a partition has the same color, each of these $d+1$ colors corresponds to at least $d+1$ different partitions. From this, it follows that or any $\delta > 0$, $\overline{B}_\delta^\infty(\vec{p})$ intersects at least $d+1$ different members of $\mathcal{P}$. □

### A.2 Missing Proofs from Section 4

In the following we use $\mathcal{D}_{A,\vec{b},n}$ to denote the distribution of the output of an algorithm for $d$-Coin Bias Estimation Problemwhen the bias vector is $\vec{b}$ and it observes $n$ independent coin tosses (per coin).

**Lemma A.10** (Lemma 4.8)**.** *For biases $\vec{a}, \vec{b} \in [0,1]^d$ we have $d_{\mathrm{TV}}\left(\mathcal{D}_{A,\vec{a},n}, \mathcal{D}_{A,\vec{b},n}\right) \le n \cdot d \cdot \|\vec{b} - \vec{a}\|_\infty$.*

*Proof.* We use the basic fact that an algorithm (deterministic or randomized) cannot increase the total variation distance between two input distributions.

The distribution giving one sample flip of each coin in a collection with bias $\vec{b}$ is the $d$-fold product of Bernoulli distributions $\prod_{i=1}^d \operatorname{Bern}(b_i)$ (which for notational brevity we denote as $\operatorname{Bern}(\vec{b})$, so the distribution which gives $n$ independent flips of each coin is the $n$-fold product of this and is denoted as $\operatorname{Bern}(\vec{b})^{\otimes n}$). We will show that for two bias vectors $\vec{a}$ and $\vec{b}$, $d_{\mathrm{TV}}\left(\operatorname{Bern}(\vec{b})^{\otimes n}, \operatorname{Bern}(\vec{a})^{\otimes n}\right) \le n \cdot d \cdot \|\vec{b} - \vec{a}\|_\infty$. This suffices to establish the lemma.

Observe that we have for each $i \in [d]$,

$$d_{\mathrm{TV}}\left(\mathrm{Bern}(b_i), \mathrm{Bern}(a_i)\right) = |b_i - a_i|.$$

Hence we have

$$d_{\mathrm{TV}}\left(\mathrm{Bern}(\vec{b}), \mathrm{Bern}(\vec{a})\right) \leq \sum_{i=1}^{d} |b_i - a_i| \leq d \cdot \|\vec{b} - \vec{a}\|_\infty$$

and

$$d_{\mathrm{TV}}\left(\mathrm{Bern}(\vec{b})^{\otimes n}, \mathrm{Bern}(\vec{a})^{\otimes n}\right) \leq n \cdot d \cdot \|\vec{b} - \vec{a}\|_\infty.$$

□

### A.3 Missing Proofs From Section 5

#### A.3.1 Proofs of Theorem 5.4, Theorem 5.5

**Theorem A.11** (Theorem 5.4)**.** *Let $\mathcal{H}$ be a concept class that is learnable with $d$ non-adaptive statistical queries, then $\mathcal{H}$ is $(d+1)$-list reproducibly learnable. Furthermore, the sample complexity $n = n(\nu, \delta)$ of the $(d+1)$-list replicable algorithm is $O(\frac{d^2}{\nu^2} \cdot \log \frac{d}{\delta})$, where $\nu$ is the approximation error parameter of each statistical query oracle.*

*Proof.* The proof is very similar to the proof of Theorem 4.4. Our replicable algorithm $B$ works as follows. Let $\varepsilon$ and $\delta$ be input parameters and $\mathcal{D}$ be a distribution and $f \in \mathcal{H}$. Let $A$ be the statistical query learning algorithm for $\mathcal{H}$. Let $STAT(D_f, \nu)$ be the statistical query oracle for this algorithm. Let $\phi_1, \ldots, \phi_d$ be the statistical queries made by $A$.

Let $\vec{b} = \langle b[1], b[2], \ldots, b[d]\rangle$ where $b[i] = E_{\langle x,y\rangle \in \mathcal{D}_f}[\phi_i(\langle x, y\rangle]$, $1 \leq i \leq d$. Set $\varepsilon_0 = \frac{\nu}{2d}$. The algorithm $B$ first estimates the values $b[i]$ up to an approximation error of $\varepsilon_0$ with success probably $1 - \delta/d$ for each query. Note that this can be done by a simple empirical estimation algorithm, that uses a total of $n = O(\frac{d^2}{\nu^2} \cdot \log \frac{d}{\delta})$ samples. Let $\vec{v}$ be the estimated vector. It follows that $\vec{v} \in \overline{B}^{\infty}_{\varepsilon_0}(\vec{b})$ with probability at least $1 - \delta$. Note that different runs of the algorithm will output different $\vec{v}$.

Next, the algorithm $B$ evaluates the deterministic function $f_\varepsilon$ from Lemma 3.1 on input $\vec{v}$. Let $\vec{u}$ be the output vector. Finally, the algorithm $B$ simulates the statistical query algorithm $A$ with $\vec{u}[i]$ as the answer to the query $\phi_i$. By Lemma 3.1, $\vec{u} \in \overline{B}^{\infty}_{\nu}(\vec{b})$. Thus the error of the hypothesis output by the algorithm is at most $\varepsilon$. Since $A$ is a deterministic algorithm the number of possible outputs only depends on the number of outputs of the function $f_\varepsilon$, more precisely the number of possible outputs is the size of the set $\{f_\varepsilon(\vec{v}) : v \in \overline{B}^{\infty}_{\varepsilon_0}(\vec{b})\}$ which is almost $d + 1$, by Lemma 3.1. Thus the total number of possible outputs of the algorithm $B$ is at most $d + 1$ with probability at least $1 - \delta$. □

**Theorem A.12** (Theorem 5.5)**.** *Let $\mathcal{H}$ be a concept class that is learnable with $d$ non-adaptive statistical queries, then $\mathcal{H}$ is $\lceil \log \frac{d}{\delta} \rceil$-certificate reproducibly learnable. Furthermore, the sample complexity $n = n(\nu, \delta)$ of this algorithm equals $O(\frac{d^2}{\nu^2 \delta^2} \cdot \log \frac{d}{\delta})$, where $\nu$ is the approximation error parameter of each statistical query oracle.*

*Proof.* The proof is very similar to the proof of Theorem 4.6. Our replicable algorithm $B$ works as follows, let $\varepsilon$ and $\delta$ be input parameters and $\mathcal{D}$ be a distribution and $f \in \mathcal{H}$. Let $A$ be the statistical query learning algorithm for $\mathcal{H}$ that outputs a hypothesis $h$ with approximation error $e_{\mathcal{D}_f}(h) = \varepsilon$. Let $STAT(D_f, \nu)$ be the statistical query oracle for this algorithm. Let $\phi_1, \ldots, \phi_d$ be the statistical queries made by $A$.

Let $\vec{b} = \langle b[1], b[2], \cdots, b[d]\rangle$, where $b[i] = E_{\langle x,y\rangle \in \mathcal{D}_f}[\phi_i(\langle x, y\rangle)]$. Set $\varepsilon_0 = \frac{\nu\delta}{2d}$. The algorithm $B$ first estimates the values $b[i], 1 \leq i \leq d$ up to an additive approximation error of $\varepsilon_0$ with success probably $1 - \delta/d$ for each query. Note that this can be done by a simple empirical estimation algorithm that uses a total of $n = O(\frac{d^2}{\nu^2 \delta^2} \cdot \log \frac{d}{\delta})$ samples. Let $\vec{v}$ be the estimated the vector. It follows that $\vec{v} \in \overline{B}^{\infty}_{\varepsilon_0}(\vec{b})$ with probability at least $1 - \delta$. Next, the algorithm $B$ evaluates the deterministic function $f$ described in Lemma 3.2 with inputs $r \in \{0, 1\}^\ell$ where $\ell = \lceil \log \frac{d}{\delta} \rceil$ and $\vec{v}$. By Lemma 3.2 for at least $1-\delta$ fraction of the $r$'s , the function $f$ outputs a canonical $\vec{v^*} \in \overline{B}_\nu^\infty(\vec{b})$. Finally, the algorithm $B$ simulates the statistical query algorithm $A$ with $\vec{v^*}[i]$ as the answer to the query $\phi_i$. Since $A$ is a deterministic algorithm it follows that our algorithm $B$ is certificate replicable. Note that the certificate complexity is $\ell = \lceil \log \frac{d}{\delta} \rceil$. □

The following theorem states how to convert adaptive statistical query learning algorithms into certificate reproducible PAC learning algorithms. This result also appears in the work of (GKM21; ILPS22), though they did not state the certificate complexity. We explicitly state the result here.

**Theorem A.13.** *((GKM21; ILPS22))[Theorem 5.6] Let $\mathcal{H}$ be a concept class that is learnable with $d$ adaptive statistical queries, then $\mathcal{H}$ is $\lceil d \log \frac{d}{\delta} \rceil$-certificate reproducibly learnable. Furthermore, the sample complexity of this algorithm equals $O(\frac{d^3}{\nu^2\delta^2} \cdot \log \frac{d}{\delta})$, where $\nu$ is the approximation error parameter of each statistical query oracle.*

*Proof.* The proof uses similar arguments as before. The main difference is that we will evaluate each query with an approximation error of $\frac{\nu\delta}{d}$ with a probability error of $d/\delta$. This requires $O(\frac{d^2}{\nu^2\delta^2} \cdot \log \frac{d}{\delta})$ per query. We use a fresh set of certificate randomness for each such evaluation. Note that the length of the certificate for each query is $\lceil \log d/\delta \rceil$. Thus the total certificate complexity is $\lceil d \log \frac{d}{\delta} \rceil$. □

### A.3.2 Proof of Theorem 5.9

We first recall the definition of the concept class $d$-THRESHOLD.

Fix some $d \in \mathbb{N}$. Let $X = [0,1]^d$. For each value $\vec{t} \in [0,1]^d$, let $h_{\vec{t}} : X \to \{0,1\}$ be the concept defined as follows: $h_{\vec{t}}(\vec{x}) = 1$ if for every $i \in [d]$ it holds that $x_i \le t_i$ and 0 otherwise. Let $\mathcal{H}$ be the hypothesis class consisting of all such threshold concepts: $\mathcal{H} = \{h_{\vec{t}} \mid \vec{t} \in [0,1]^d\}$.

**Theorem A.14** (Theorem 5.9)**.** *In the PAC model under the uniform distribution, there is a $d+1$-list replicable algorithm for the $d$-*THRESHOLD*. Moreover, for any $k < d+1$, there does not exist a $k$-list replicable algorithm for the concept class $d$-*THRESHOLD *under the uniform distribution. Thus its list complexity is exactly $d+1$.*

It is easy to see that $d$-THRESHOLD is learnable under the uniform distribution by making $d$ non-adaptive statistical queries. Thus by Theorem 5.4, $d$-THRESHOLD under the uniform distribution admits a $(d+1)$-list replicable algorithm. So we will focus on proving the lower bound which is stated as a separate theorem below.

**Theorem A.15.** *For $k < d+1$, there does not exist a $k$-list replicable algorithm for the $d$-*THRESHOLD *in the PAC model under uniform distribution.*

The proof is similar to the proof of Theorem 4.7. The reason is that sampling $d$-many biased coins with bias vector $\vec{b}$ is similar to obtaining a point $\vec{x}$ uniformly at random from $[0,1]^d$ and evaluating the threshold function $h_{\vec{b}}$ on it—this corresponds to asking whether all of the coins were heads/1's. The two models differ though, because in the sample model for the $d$-COIN BIAS ESTIMATION PROBLEM, the algorithm sees for each coin whether it is heads or tails, but this information is not available in the PAC model for the $d$-THRESHOLD. Conversely, in the PAC model for the $d$-THRESHOLD, a random draw from $[0,1]^d$ is available to the algorithm, but in the sample model for the $d$-COIN BIAS ESTIMATION PROBLEM the algorithm does not get this information.

Furthermore, there is the following additional complexity in the impossibility result for the $d$-THRESHOLD. In the $d$-COIN BIAS ESTIMATION PROBLEM, we said by definition that a collection of $d$ coins parameterized by bias vector $\vec{a}$ was an $\varepsilon$-approximation to a collection of $d$ coins parameterized by bias vector $\vec{b}$ if and only if $\|\vec{b}-\vec{a}\|_\infty \le \varepsilon$, and we used this norm in the proofs. However, the notion of $\varepsilon$-approximation in the PAC model is quite different than this. It is possible to have a hypotheses $h_{\vec{a}}$ and $h_{\vec{b}}$ in the $d$-THRESHOLDsuch that $\|\vec{b}-\vec{a}\|_\infty > \varepsilon$ but with respect to some distribution $\mathcal{D}_X$ on the domain $X$ we have $\mathrm{e}_{\mathcal{D}_X}(h_{\vec{a}}, h_{\vec{b}}) \le \varepsilon$. For example, if $\mathcal{D}_X$ is the uniform distribution on $X = [0,1]^d$ and $\vec{a} = \vec{0}$ and $\vec{b}$ is the first standard basis vector $\vec{b} = \langle 1, 0, \ldots, 0 \rangle$, and $\varepsilon = \frac{1}{2}$, then $\|\vec{b}-\vec{a}\|_\infty = 1 > \varepsilon$, but $\mathrm{e}_{\mathcal{D}_X}(h_{\vec{a}}, h_{\vec{b}}) = 0 \le \varepsilon$ because $h_{\vec{a}}(\vec{x}) \ne h_{\vec{b}}(\vec{x})$ if and only if all of the last $d-1$ coordinates of $\vec{x}$ are 0 and the first coordinate is $> 0$, but there is probability 0 of sampling such $\vec{x}$ from the uniform distribution on $X = [0,1]^d$.

For this reason, we can't just partition $[0,1]^d$ as we did with the proof of Theorem 4.7 and must do something more clever. It turns out that it is possible to find a subset $[\alpha,1]^d$ on which hypotheses parameterized by vectors on opposite faces of this cube $[\alpha,1]^d$ have high PAC error between them. A consequence by the triangle inequality of $\mathrm{e}_{\mathcal{D}_X}$ is that two such hypotheses cannot both be approximated by a common third hypothesis. This is the following lemma states.

**Lemma A.16.** *Let $d \in \mathbb{N}$ and $\alpha = \frac{d-1}{d}$. Let $\vec{s}, \vec{t} \in [\alpha,1]^d$ such that there exists a coordinate $i_0 \in [d]$ where $s_{i_0} = \alpha$ and $t_{i_0} = 1$ (i.e. $\vec{s}$ and $\vec{t}$ are on opposite faces of this cube). Let $\varepsilon \le \frac{1}{8d}$. Then there is no point $\vec{r} \in X$ such that both $\mathrm{e}_{\mathrm{unif}}(h_{\vec{s}}, h_{\vec{r}}) \le \varepsilon$ and $\mathrm{e}_{\mathrm{unif}}(h_{\vec{t}}, h_{\vec{r}}) \le \varepsilon$ (i.e. there is no hypothesis which is an $\varepsilon$-approximation to both $h_{\vec{s}}$ and $h_{\vec{t}}$).*

*Proof.* Let $\vec{q} = \left\langle \begin{cases} s_i & i = i_0 \\ t_i & i \ne i_0 \end{cases} \right\rangle_{i=1}^{d}$ which will serve as a proxy to $\vec{s}$.

We need the following claim.

**Claim A.17.** *For each $\vec{x} \in X$, the following are equivalent:*

1. $h_{\vec{q}}(\vec{x}) \ne h_{\vec{t}}(\vec{x})$
2. $h_{\vec{q}}(\vec{x}) = 0$ *and* $h_{\vec{t}}(\vec{x}) = 1$
3. $x_{i_0} \in (q_{i_0}, t_{i_0}] = (\alpha, 1]$ *and for all* $i \in [d] \setminus \{i_0\}$, $x_i \in [0, t_i]$.

*Furthermore, the above equivalent conditions imply the following:*

4. $h_{\vec{s}}(\vec{x}) \ne h_{\vec{t}}(\vec{x})$.

*Proof of Claim A.17.*

(2) $\Longrightarrow$ (1): This is trivial.

(1) $\Longrightarrow$ (2): Note that because $q_{i_0} = s_{i_0} = \alpha < 1 = t_{i_0}$, we have for all $i \in [d]$ that $q_i \le t_i$. If $h_{\vec{t}}(\vec{x}) = 0$ then for some $i_1 \in [d]$ it must be that $x_{i_1} > t_{i_1}$, but since $t_{i_1} \ge q_{i_1}$ it would also be the case that $x_{i_1} > q_{i_1}$, so $h_{\vec{q}}(\vec{x}) = 0$ which gives the contradiction that $h_{\vec{q}}(\vec{x}) = h_{\vec{t}}(\vec{x})$. Thus $h_{\vec{t}}(\vec{x}) = 1$, and since $h_{\vec{q}}(\vec{x}) \ne h_{\vec{t}}(\vec{x})$ we have $h_{\vec{q}}(\vec{x}) = 0$.

(1) $\Longleftrightarrow$ (3): We partition $[0,1]^d$ into three sets and examine these three cases.

Case 1: $x_{i_0} \in (q_{i_0}, t_{i_0}] = (\alpha, 1]$ and for all $i \in [d] \setminus \{i_0\}$, $x_i \in [0, t_i]$. In this case, $q_{i_0} < x_{i_0}$ so $h_{\vec{q}}(\vec{x}) = 0$ and for all $i \in [d]$ $x_i \le t_i$, so $h_{\vec{t}}(\vec{x}) = 1$, so $h_{\vec{q}}(\vec{x}) \ne h_{\vec{t}}(\vec{x})$.

Case 2: $x_{i_0} \notin (q_{i_0}, t_{i_0}] = (\alpha, 1]$ and for all $i \in [d] \setminus \{i_0\}$, $x_i \in [0, t_i]$. In this case, because $x_{i_0} \in [0,1]$ and $x_{i_0} \notin (\alpha, 1]$ we have $x_{i_0} \le \alpha = q_{i_0} \le t_{i_0}$ and also for all other $i \in [d] \setminus \{i_0\}$, $x_i \le t_i = q_i$ (by definition of $\vec{q}$). Thus $h_{\vec{q}}(\vec{x}) = 1 = h_{\vec{t}}(\vec{x})$.

Case 3: For some $i_1 \in [d] \setminus \{i_0\}$, $x_{i_1} \notin [0, t_{i_1}]$. In this case, because $x_{i_1} \in [0,1]$, we have $x_{i_1} > t_{i_1} = q_{i_1}$. Thus $h_{\vec{q}}(\vec{x}) = 0 = h_{\vec{t}}(\vec{x})$.

Thus, it is the case that $h_{\vec{q}}(\vec{x}) \ne h_{\vec{t}}(\vec{x})$ if and only if $x_{i_0} \in (q_{i_0}, t_{i_0}] = (\alpha, 1]$ and for all $i \in [d] \setminus \{i_0\}$, $x_i \in [0, t_i]$.

(1, 2, 3) $\Longrightarrow$ (4): By (2), we have $x_{i_0} > q_{i_0}$, and since $q_{i_0} = s_{i_0}$ by definition of $\vec{q}$, it follows that $x_{i_0} > s_{i_0}$ which means $h_{\vec{s}}(\vec{x}) = 0$. By (3), $h_{\vec{t}}(\vec{x}) = 1$ which gives $h_{\vec{s}}(\vec{x}) \ne h_{\vec{t}}(\vec{x})$. ☐

We also need the following Lemma.

**Lemma A.18.** *Let $d \in \mathbb{N}$ and $\alpha = \frac{d-1}{d} = 1 - \frac{1}{d}$. Then $(1-\alpha) \cdot \alpha^{d-1} > \frac{1}{4d}$.*

*Proof.* If $d = 1$, then $\alpha = 0$ so $(1-\alpha) \cdot \alpha^{d-1} = 1 \ge \frac{1}{4} = \frac{1}{4d}$ (see footnote[2]).

[2] This uses the interpretation that $0^0 = 1$ which is the correct interpretation in the context in which we will use the lemma.

If $d \geq 2$, then we utilize the fact that $(1 - \frac{1}{d})^d \geq \frac{1}{4}$ in the following:

$$\begin{aligned}(1-\alpha)\cdot\alpha^{d-1} &= (\tfrac{1}{d})(1-\tfrac{1}{d})^{d-1} \\ &= (\tfrac{1}{d})\frac{(1-\frac{1}{d})^d}{1-\frac{1}{d}} \\ &= \frac{(1-\frac{1}{d})^d}{d-1} \\ &\geq \frac{1}{4(d-1)} \\ &> \frac{1}{4d}.\end{aligned}$$

This completes the proof. As an aside, $\alpha = \frac{d-1}{d}$ is the value of $\alpha$ that maximizes the expression $(1-\alpha)\cdot\alpha^{d-1}$ which is why that value was chosen. □

With the above Claim and Lemma in hand, we return to the proof of Lemma A.16. Our next step will be two prove the following two inequalities:

$$2\varepsilon < \mathrm{e}_{\mathrm{unif}}(h_{\vec{q}}, h_{\vec{t}}) \leq \mathrm{e}_{\mathrm{unif}}(h_{\vec{s}}, h_{\vec{t}}).$$

For the second of these inequalities, note that by the (1) $\implies$ (4) part of claim above, since $h_{\vec{q}}(\vec{x}) \neq h_{\vec{t}}(\vec{x})$ implies $h_{\vec{s}}(\vec{x}) \neq h_{\vec{t}}(\vec{x})$ we have

$$\begin{aligned}\mathrm{e}_{\mathrm{unif}}(h_{\vec{q}}, h_{\vec{t}}) &= \Pr_{\vec{x}\sim\mathrm{unif}(X)}[h_{\vec{q}}(\vec{x}) \neq h_{\vec{t}}(\vec{x})] \\ &\leq \Pr_{\vec{x}\sim\mathrm{unif}(X)}[h_{\vec{s}}(\vec{x}) \neq h_{\vec{t}}(\vec{x})] \\ &= \mathrm{e}_{\mathrm{unif}}(h_{\vec{s}}, h_{\vec{t}}).\end{aligned}$$

Now, for the first of the inequalities above, we will use the (1) $\iff$ (3) portion of the claim, we will use our hypothesis that $\vec{t} \in [\alpha, 1]^d$ (which implies for each $i \in [d]$ that $[0, t_i] \subseteq [0, \alpha]$), we will use the hypothesis that $\varepsilon \leq \frac{1}{8d}$, and we will use Theorem A.18. Utilizing these, we get the following:

$$\begin{aligned}&\mathrm{e}_{\mathrm{unif}}(h_{\vec{q}}, h_{\vec{t}}) \\ &= \Pr_{\vec{x}\sim\mathrm{unif}(X)}[h_{\vec{q}}(\vec{x}) \neq h_{\vec{t}}(\vec{x})] \\ &= \Pr_{\vec{x}\sim\mathrm{unif}(X)}[x_{i_0} \in (\alpha, 1] \ \wedge\ \forall i \in [d] \setminus \{i_0\},\, x_i \in [0, t_i]] \\ &= \Pr_{x_{i_0}\sim\mathrm{unif}([0,1])}[x_{i_0} \in (\alpha, 1]] \cdot \prod_{\substack{i=1 \\ i\neq i_0}}^{d} \Pr_{x\sim\mathrm{unif}([0,1])}[x \in [0, t_i]] \\ &\geq \Pr_{x_{i_0}\sim\mathrm{unif}([0,1])}[x_{i_0} \in (\alpha, 1]] \cdot \prod_{\substack{i=1 \\ i\neq i_0}}^{d} \Pr_{x\sim\mathrm{unif}([0,1])}[x \in [0, \alpha]] \\ &= (1-\alpha)\cdot\alpha^{d-1} \\ &> \frac{1}{4d} \\ &\geq 2\varepsilon.\end{aligned}$$

Thus, we get the desired two inequalities:

$$2\varepsilon < \mathrm{e}_{\mathrm{unif}}(h_{\vec{q}}, h_{\vec{t}}) \leq \mathrm{e}_{\mathrm{unif}}(h_{\vec{s}}, h_{\vec{t}}).$$

This nearly completes the proof. If there existed some point $\vec{r} \in X$ such that both $\mathrm{e}_{\mathrm{unif}}(h_{\vec{s}}, h_{\vec{r}}) \leq \varepsilon$ and $\mathrm{e}_{\mathrm{unif}}(h_{\vec{t}}, h_{\vec{r}}) \leq \varepsilon$, then it would follow from the triangle inequality of $\mathrm{e}_{\mathrm{unif}}$ that

$$\mathrm{e}_{\mathrm{unif}}(h_{\vec{s}}, h_{\vec{t}}) \leq \mathrm{e}_{\mathrm{unif}}(h_{\vec{s}}, h_{\vec{r}}) + \mathrm{e}_{\mathrm{unif}}(h_{\vec{t}}, h_{\vec{r}}) \leq 2\varepsilon$$

but this would contradict the above inequalities, so no such $\vec{r}$ exists. □

Equipped with the Lemma A.16, we are now ready to prove Theorem A.15.

*Proof of Theorem A.15.* Fix any $d \in \mathbb{N}$, and choose $\varepsilon$ and $\delta$ as $\varepsilon \leq \frac{1}{4d}$ and $\delta \leq \frac{1}{d+2}$. We will use the constant $\alpha = \frac{d-1}{d}$ and consider the cube $[\alpha, 1]^d$.

Suppose for contradiction such an algorithm $A$ does exists for some $k < d+1$. This means that for each possible threshold $\vec{t} \in [0,1]^d$, there exists some set $L_{\vec{t}} \subseteq \mathcal{H}$ of hypotheses with three properties: (1) each element of $L_{\vec{t}}$ is an $\varepsilon$-approximation to $h_{\vec{t}}$, (2) $|L_{\vec{t}}| \leq k$, and (3) with probability at least $1-\delta$, $A$ returns an element of $L_{\vec{t}}$.

By the trivial averaging argument, this means that there exists at least one element in $L_{\vec{t}}$ which is returned by $A$ with probability at least $\frac{1}{k} \cdot (1-\delta) \geq \frac{1}{k} \cdot (1 - \frac{1}{d+2}) = \frac{1}{k} \cdot \frac{d+1}{d+2} \geq \frac{1}{k} \cdot \frac{k+1}{k+2}$. Let $f \colon [\alpha, 1]^d \to [0,1]^d$ be a function which maps each threshold $\vec{t} \in [\alpha, 1]^d$ to such an element (the maximum probability element with ties broken arbitrarily) of $L_{\vec{t}}$. This is slightly different from the proof of Theorem 4.7 because we are defining the function $f$ on only a very specific subset of the possible thresholds. The reason for this was alluded to in the discussion following the statement of Theorem A.15.

The function $f$ induces a partition $\mathcal{P}$ of $[\alpha, 1]^d$ where the members of $\mathcal{P}$ are the fibers of $f$ (i.e. $\mathcal{P} = \left\{ f^{-1}(\vec{y}) \colon \vec{y} \in \text{range}(f) \right\}$). For any member $W \in \mathcal{P}$ and any coordinate $i \in [d]$, it cannot be that the set $w_i \colon \vec{w} \in W$ contains both values $\alpha$ and 1—if it did, then there would be two points $\vec{s}, \vec{t} \in W$ such that $s_i = \alpha$ and $t_i = 1$, but because they both belong to $W$, there is some $\vec{y} \in [0,1]^d$ such that $f(\vec{s}) = \vec{y} = f(\vec{t})$, but by definition of the partition, $h_{\vec{y}}$ would have to be an $\varepsilon$-approximation (in the PAC model) of both $h_{\vec{s}}$ and $h_{\vec{t}}$, but by Lemma A.16 this is not possible.

Thus, the partition $\mathcal{P}$ is a *non-spanning* partition of $[\alpha, 1]^d$ as in the proof of Lemma 3.3, so there is some point $\vec{p} \in [\alpha, 1]^d$ such that for every radius $r > 0$, it holds that $\overline{B}_r^\infty(\vec{p})$ intersects at least $d+1$ members of $\mathcal{P}$. Infact, there us some radius $r$ such that $\|\vec{t} - \vec{s}\|_\infty \leq r$, then $d_{\mathrm{TV}}(\mathcal{D}_{A,\vec{s},n}, \mathcal{D}_{A,\vec{t},n}) \leq \eta$, for $\eta$ the lies between 0 and $\frac{1}{k} \cdot \frac{k+1}{k+2} - \frac{1}{k+1}$.

Now we get the same type of contradiction as in the proof of Theorem 4.7: for the special point $\vec{p}$ we have that $\mathcal{D}_{A,\vec{p},n}$ is a distribution that has $d+1 \geq k+1$ disjoint events that each have probability greater than $\frac{1}{k+1}$. Thus, no $k$-list replicable algorithm exists. □

# B Supplementary Material: Prior and Related Work

We give a more detailed discussion on prior and related work. This section is an elaboration of the Section 2 from the main body of the paper. Since this expanded section cites more work from the literature, we include a new bibliography.

Formalizing reproducibility and replicability has gained considerable momentum in recent years. While the terms reproducibility and replicability are very close and often used interchangeably, there has been an effort to distinguish between them and accordingly, our notions fall in the replicability definition (PVLS+21).

In the context of randomized algorithms, various notions of reproducibility/replicability have been investigated. The work of Gat and Goldwasser (GG11) formalized and defined the notion of *pseudodeterministic algorithms*. A randomized algorithm $A$ is *pseudodeterministic* if, for any input $x$, there is a canonical value $v_x$ such that $\Pr[A(x) = v_x] \geq 2/3$. Gat and Goldwasser designed polynomial-time pseudodeterministic algorithms for algebraic computational problems, such as finding quadratic non-residues and finding non-roots of multivariate polynomials (GG11). Later works studied the notion of pseudodeterminism in other algorithmic settings, such as parallel computation, streaming and sub-linear algorithms, interactive proofs, and its connections to complexity theory (GG; GGH18; OS17; OS18; AV20; GGMW20; LOS21; DPVWV22).

In the algorithmic setting, mainly two generalizations of pseudodeterminism have been investigated: *multi-pseudodeterministic algorithms* (Gol19) and *influential bit algorithms* (GL19). A randomized algorithm $A$ is $k$-pseudodeterministic if, for every input $x$, there is a set $S_x$ of size at most $k$ such that the output of $A(x)$ belongs to the set $S_x$ with high probability. When $k = 1$, we get pseudodeterminism. A randomized algorithm $A$ is $\ell$-influential-bit algorithm if, for every input $x$, for most of the strings $r$ of length $\ell$, there exists a canonical value $v_{x,r}$ such that the algorithm $A$ on inputs $x$ and $r$ outputs $v_{x,r}$ with high probability. The string $r$ is called the *influential bit* string. Again, when $\ell = 0$, we get back pseudodeterminism. The main focus of these works has been to investigate reproducibility in randomized search algorithms.

Very recently, pseudodeterminism and its generalizations have been explored in the context of learning algorithms to formalize the notion of replicability. The seminal work of (BLM20) defined the notion of *global stability*. They define a learning algorithm $A$ to be $(n, \eta)$-globally stable with respect to a distribution $D$ if there is a hypothesis $h$ such that $\Pr_{S\sim D^n}(A(S) = h) \geq \eta$, here $\eta$ is called the *stability parameter*. Note that the notion of global stability is equivalent to Gat and Goldwasser's notion of pseudodeterminism when $\eta = 2/3$. Since Gat and Goldwasser's motivation is to study pseudodeterminism in the context of randomized algorithms, the success probability is taken as $2/3$. In the context of learning, studying the stability parameter $\eta$ turned out to be useful. The work of Bun, Livny and Moran (BLM20) showed that any concept class with Littlestone dimension $d$ has an $(m, \eta)$-globally stable learning algorithm with $m = \tilde{O}(2^{2^d}/\alpha)$ and $\eta = \tilde{O}(2^{-2^d})$, where the error of $h$ (with respect to the unknown hypothesis) is $\leq \alpha$. Then they established that a globally stable learner implies a differentially private learner. This, together with an earlier work of Alon, Livny, Malliaris, and Moran (ALMM19), establishes an equivalence between online learnability and differentially private PAC learnability.

The work of Ghazi, Kumar, and Manurangsi (GKM21) extended the notion of global stability to pseudo-global stability and list-global stability. The notion of pseudo-global stability is very similar to the earlier-mentioned notion of influential bit algorithms of Grossman and Liu (GL19) when translated to the context of learning. Similarly, the list-global stability is similar to Goldreich's notion of multi-pseudodeterminism (Gol19). These notions coincide with our definitions of list replicability and certificate replicability respectively. The work of (GKM21) used these concepts to design user-level differentially private algorithms.

The recent work reported in (ILPS22) introduced the notion of *$\rho$-replicability*. A learning algorithm $A$ is $\rho$-replicable if $\Pr_{S_1,S_2,r}[A(S_1, r) = A(S_2, r)] \geq 1-\rho$, where $S_1$ and $S_2$ are samples drawn from a distribution $\mathcal{D}$ and $r$ is the internal randomness of the learning algorithm $A$. They designed replicable algorithms for many learning tasks, including statistical queries, approximate heavy hitters, median, and learning half-spaces. It is known that the notions of pseudo-global stability and $\rho$-replicability are the same up to polynomial factors in the parameters (ILPS22; GKM21).

In this work, we study the notions of list and certificate complexities as a measure the *degree of (non) replicability*. Our goal is to design learning algorithms with optimal list and certificate complexities while minimizing the sample complexity. The earlier works (BLM20; GKM21; ILPS22) did not focus on minimizing these quantities. The works of (BLM20; GKM21) used replicable algorithms as an intermediate step to design differentially private algorithms. The work of (ILPS22) did not consider reducing the certificate complexity in their algorithms and also did not study list-replicability. Earlier works (GKM21; ILPS22) studied how to convert statistical query learning algorithms into certificate replicable learning algorithms, however, their focus was not on the certificate complexity. Here, we study the relationship among (nonadaptive and adaptive) statistical query learning algorithms, list replicable algorithms, and certificate replicable algorithms with a focus on list, certificate and sample complexities.

A very recent and independent work of (CMY23) investigated relations between list replicability and the stability parameter $\nu$, in the context of distribution-free PAC learning. They showed that for every concept class $\mathcal{H}$, its list complexity is exactly the inverse of the stability parameter. They also showed that the list complexity of a hypothesis class is at least its VC dimension. For establishing this they exhibited, for any $d$, a concept class whose list complexity is exactly $d$. There are some similarities between their work and the present work. We establish similar upper and lower bounds on the list complexity but for different learning tasks: $d$-THRESHOLD and $d$-COIN BIAS ESTIMATION PROBLEM. For $d$-THRESHOLD, our results are for PAC learning under *uniform* distribution and do not follow from their distribution-independent results. Thus our results, though similar in spirit, are incomparable to theirs. Moreover, their work did not focus on efficiency in sample complexity and also did not study certificate complexity which is a focus of our paper. We do not study the stability parameter.

The study of notions of reproducibility/replicability in various computational fields is an emerging topic. The article (PVLS+21) discusses the differences between replicability and reproducibility. In

(EKK+23), the authors consider replicability in the context of stochastic bandits. Their notion is similar to the notion studied in (ILPS22). In (AJJ+22), the authors investigate *reproducibility* in the context of optimization with *inexact oracles* (initialization/gradient oracles). The setup and focus of these works are different from ours.

## References


[AJJ+22] Kwangjun Ahn, Prateek Jain, Ziwei Ji, Satyen Kale, Praneeth Netrapalli, and Gil I. Shamir. Reproducibility in optimization: Theoretical framework and limits, 2022. arXiv:2202.04598.

[ALMM19] Noga Alon, Roi Livni, Maryanthe Malliaris, and Shay Moran. Private PAC learning implies finite littlestone dimension. In Moses Charikar and Edith Cohen, editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 852–860. ACM, 2019.

[AV20] Nima Anari and Vijay V. Vazirani. Matching is as easy as the decision problem, in the NC model. In Thomas Vidick, editor, *11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA*, volume 151 of *LIPIcs*, pages 54:1–54:25. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[Bak16] Monya Baker. 1,500 scientists lift the lid on reproducibility. *Nature*, 533:452–454, 2016.

[BLM20] Mark Bun, Roi Livni, and Shay Moran. An equivalence between private classification and online prediction. In Sandy Irani, editor, *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 389–402. IEEE, 2020.

[CMY23] Zachary Chase, Shay Moran, and Amir Yehudayoff. Replicability and stability in learning, 2023. arXiv:2304.03757.

[DLPES02] Jesus A. De Loera, Elisha Peterson, and Francis Edward Su. A Polytopal Generalization of Sperner's Lemma. *Journal of Combinatorial Theory, Series A*, 100(1):1–26, October 2002. URL: https://www.sciencedirect.com/science/article/pii/S0097316502932747, doi:10.1006/jcta.2002.3274.

[DPV18] Peter Dixon, Aduri Pavan, and N. V. Vinodchandran. On pseudodeterministic approximation algorithms. In Igor Potapov, Paul G. Spirakis, and James Worrell, editors, *43rd International Symposium on Mathematical Foundations of Computer Science, MFCS 2018, August 27-31, 2018, Liverpool, UK*, volume 117 of *LIPIcs*, pages 61:1–61:11. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[DPVWV22] Peter Dixon, Aduri Pavan, Jason Vander Woude, and N. V. Vinodchandran. Pseudodeterminism: promises and lowerbounds. In Stefano Leonardi and Anupam Gupta, editors, *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*, pages 1552–1565. ACM, 2022.

[eco13] How science goes wrong. *The Economist*, pages 25–30, 2013.

[EKK+23] Hossein Esfandiari, Alkis Kalavasis, Amin Karbasi, Andreas Krause, Vahab Mirrokni, and Grigoris Velegkas. Replicable bandits, 2023. arXiv:2210.01898.

[GG] Shafi Goldwasser and Ofer Grossman. Bipartite perfect matching in pseudo-deterministic NC. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, volume 80 of *LIPIcs*.

[GG11] Eran Gat and Shafi Goldwasser. Probabilistic Search Algorithms with Unique Answers and Their Cryptographic Applications. Technical Report 136, 2011. URL: https://eccc.weizmann.ac.il/report/2011/136/.

[GGH18] Shafi Goldwasser, Ofer Grossman, and Dhiraj Holden. Pseudo-deterministic proofs. In Anna R. Karlin, editor, *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, volume 94 of *LIPIcs*, pages 17:1–17:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[GGMW20] Shafi Goldwasser, Ofer Grossman, Sidhanth Mohanty, and David P. Woodruff. Pseudo-deterministic streaming. In Thomas Vidick, editor, *11th Innovations in Theoretical Computer Science Conference, ITCS*, volume 151 of *LIPIcs*, pages 79:1–79:25, 2020.

[GKM21] Badih Ghazi, Ravi Kumar, and Pasin Manurangsi. User-level differentially private learning via correlated sampling. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 20172–20184, 2021.

[GL19] Ofer Grossman and Yang P. Liu. Reproducibility and pseudo-determinism in log-space. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 606–620. SIAM, 2019. doi:10.1137/1.9781611975482.38.

[Gol19] Oded Goldreich. Multi-pseudodeterministic algorithms. *Electron. Colloquium Comput. Complex.*, TR19-012, 2019. URL: https://eccc.weizmann.ac.il/report/2019/012, arXiv:TR19-012.

[ILPS22] Russell Impagliazzo, Rex Lei, Toniann Pitassi, and Jessica Sorrell. Reproducibility in learning. In Stefano Leonardi and Anupam Gupta, editors, *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*, pages 818–831. ACM, 2022. doi:10.1145/3519935.3519973.

[JP05] Ioannidis JP. Why most published research findings are false. *PLOS Medicine*, 2(8), 2005.

[Kea98] Michael J. Kearns. Efficient noise-tolerant learning from statistical queries. *J. ACM*, 45(6):983–1006, 1998.

[LOS21] Zhenjian Lu, Igor Carboni Oliveira, and Rahul Santhanam. Pseudodeterministic algorithms and the structure of probabilistic time. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 303–316. ACM, 2021.

[MPK19] Harshal Mittal, Kartikey Pandey, and Yash Kant. Iclr reproducibility challenge report (padam : Closing the generalization gap of adaptive gradient methods in training deep neural networks), 2019. arXiv:1901.09517.

[NAS19] Reproducibility and Replicability in Science. https://doi.org/10.17226/25303, 2019. National Academies of Sciences, Engineering, and Medicine.

[OS17] I. Oliveira and R. Santhanam. Pseudodeterministic constructions in subexponential time. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 665–677, 2017.

[OS18] Igor Carboni Oliveira and Rahul Santhanam. Pseudo-derandomizing learning and approximation. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2018*, volume 116 of *LIPIcs*, pages 55:1–55:19, 2018.

[PVLS+21] Joelle Pineau, Philippe Vincent-Lamarre, Koustuv Sinha, Vincent Lariviere, Alina Beygelzimer, Florence d'Alche Buc, Emily Fox, and Hugo Larochelle. Improving reproducibility in machine learning research(a report from the neurips 2019 reproducibility program). *Journal of Machine Learning Research*, 22(164):1–20, 2021. URL: http://jmlr.org/papers/v22/20-303.html.

[SZ99] Michael E. Saks and Shiyu Zhou. $\mathrm{BP_hSPACE(S)} \subseteq \mathrm{DSPACE(S^{3/2})}$. *J. Comput. Syst. Sci.*, 58(2):376–403, 1999.

[VWDP+22] Jason Vander Woude, Peter Dixon, Aduri Pavan, Jamie Radcliffe, and N. V. Vinodchandran. Geometry of rounding. *CoRR*, abs/2211.02694, 2022. arXiv:2211.02694, doi:10.48550/arXiv.2211.02694.