Revealing Information while Preserving Privacy

Irit Dinur Kobbi Nissim

December 21, 2004

Abstract

We examine the tradeoff between privacy and usability of statistical databases. We model a statistical database by an n-bit string $d_1, .., d_n$, with a query being a subset $q \subseteq [n]$ to be answered by $\sum_{i \in q} d_i$. Our main result is a polynomial reconstruction algorithm of data from noisy (perturbed) subset sums. Applying this reconstruction algorithm to statistical databases we show that in order to achieve privacy one has to add perturbation of magnitude $\Omega(\sqrt{n})$. That is, smaller perturbation always results in a strong violation of privacy. We show that this result is tight by exemplifying access algorithms for statistical databases that preserve privacy while adding perturbation of magnitude $\tilde{O}(\sqrt{n})$.

For time-$\mathcal{T}$ bounded adversaries we demonstrate a privacy-preserving access algorithm whose perturbation magnitude is $\approx \sqrt{\mathcal{T}}$.

1 Introduction

Let us begin with a short story. Envision a database of a hospital containing the medical history of some population. On one hand, the hospital would like to advance medical research which is based (among other things) on statistics of the information in the database. On the other hand, the hospital is obliged to keep the privacy of its patients, i.e. leak no medical information that could be related to a specific patient. The hospital needs an access mechanism to the database that allows certain 'statistical' queries to be answered, as long as they do not violate the privacy of any single patient.

One simple tempting solution is to remove from the database all 'identifying' attributes such as the patients' names and social security numbers. However, this solution is not enough to protect patient privacy since there usually exist other means of identifying patients, via *indirectly identifying* attributes stored in the database. Usually, identification may still be achieved by coming across just a few 'innocuous' looking attributes[1].

The topic of this work is to explore the conditions under which such a privacy preserving database access mechanism can exist.

A Threshold for Noisy Reconstruction. Viewing query-answer pairs as an 'encoding' of the bits $d_1, .., d_n$, the goal of the privacy-breaking adversary is to efficiently 'decode' this encoding i.e. to obtain values of some d_is. In our setting, the 'decoding' algorithm is given access to subset sums of the d_is perturbed by adding some random noise of magnitude $\leq \mathcal{E}$. We show an interesting

[1] A patient's gender, approximate age, approximate weight, ethnicity, and marital status – may already suffice for a complete identification of most patients in a database of a thousand patients. The situation is much worse if a relatively 'rare' attribute of some patient is known. For example, a patient having Cystic Fibrosis (frequency $\approx 1/3000$) may be uniquely identified within about a million patients.

threshold phenomenon where either almost all of the d_is can be reconstructed, in case $\mathcal{E} \ll \sqrt{n}$, or none of them, when $\mathcal{E} \gg \sqrt{n}$.

1.1 A Brief Background

The problem of protecting sensitive information in a database while allowing *statistical* queries (i.e. queries about sums of entries, and the like) has been studied extensively since the late 70's, (see surveys in [2, 18]).

In their comparative survey of privacy methods for statistical databases, Adam and Wortmann [2] classified the approaches taken into three main categories: (i) query restriction, (ii) data perturbation, and (iii) output perturbation. We give a brief review of these approaches below, and refer the reader to [2] for a detailed survey of the methods and their weaknesses.

Query Restriction. In the query restriction approach, queries are required to obey a special structure, supposedly to prevent the querying adversary from gaining too much information about specific database entries. The limit of this approach is that it allows for a relatively small number of queries.

A related idea is of query auditing [7], i.e. a log of the queries is kept, and every new query is checked for possible compromise, allowing/disallowing the query accordingly. (See also [14] where it is shown that the auditor's computational task is NP-hard.) The problem arising here is that the auditor's 'refusals', in conjunction with the answers to 'valid' queries, may be used adversarially to achieve a partial (in some cases total) compromise of the database. We include some further discussion in Appendix B.

Data/Output Perturbation. In the data perturbation approach queries are answered according to a perturbed database. In the output perturbation approach, the database first computes an 'exact' answer, but returns a 'noisy' version of it. Methods of data perturbation include *swapping* [17, 15] where portions of the data are replaced with data taken from the same distribution, and *fixed perturbations* [20] where a random perturbation is added to every data entry (see also [4, 3]). Methods of output perturbation include *varying output perturbations* [6], where a random perturbation is added to the query answer, with increasing variance as the query is repeated, and *rounding* - either deterministic [1] or probabilistic [11].

1.2 Database Privacy

Intuitively, to achieve database privacy one has to play a game of balancing two sets of functions: (i) the "private" functions that we wish to hide and (ii) the "information" functions whose values we wish to reveal. This general view allows for a great variety of privacy definitions. However, in most works the privacy functions are taken to be the single entries of the database, i.e. $\{\pi_i\}_{i\in[n]}$ where $\pi_i(d_1, .., d_n) = d_i$. This choice captures the intuition that privacy is violated if an adversary is capable of computing a confidential attribute d_i from its identity i. In the context of statistical database privacy, the information functions are usually taken to be sums of subsets of the database entries i.e. $\{f_q\}_{q\subseteq[n]}$ where $f_q(d_1, .., d_n) = \sum_{i\in q} d_i$. I.e. f_q measures the number of occurrences of an attribute within the population q.

We present a *computational* definition of privacy that asserts that it is *computationally infeasible* to retrieve private information from the database. We prefer that to other 'natural' measures that were in use in previous works – such as the variance of query answers, and the estimator variance.

There are two potential drawbacks to these definitions. Firstly, it is not clear that large variance necessarily prevents private information from being leaked[2]. Secondly, this kind of definition does not allow us to capitalize on the limits of an adversary.

One difficulty in estimating (partial) compromise stems from the unknown extent of the adversary's a-priori knowledge. A way to model prior knowledge is by having the database drawn from some distribution $\mathcal{DB}$ over binary strings $\{0,1\}^n$. Having no prior knowledge is conceptually equivalent to having all possible database configurations (n-bit strings) equally likely, a situation that is modelled by letting $\mathcal{DB}$ be the uniform distribution over $\{0,1\}^n$.

Privacy and Cryptography. Privacy is treated in various aspects of cryptography, usually in a manner that is complementary to our discussion. For example, in secure function evaluation [21, 12] several parties compute a function F of their private inputs $d_1,..,d_n$. Privacy is perceived here as protecting each party's private input so that other parties can not deduce information that is not already deducible from the function outcome $F(d_1,..,d_n)$. In other words, the function F dictates which information is to be revealed, and the goal is to leak no additional information. Note that privacy is defined here *implicitly* – according to the computed function F, this may lead to leaking no information about the private inputs on one end of the spectrum, and leaking complete information on the other end.

In this work we reverse the order. We first specify *explicitly* which information should not be leaked, and then look for functions revealing the maximum information still possible. Our privacy vs. information game can be viewed as an interplay between the "private" functions and the "information" functions whose values we wish to compute (or approximate) while maintaining privacy.

Our Definitions. A succinct catch-all definition of privacy is very elusive. To the best of our knowledge none of the current definitions of privacy in statistical databases serves as one that is good for all possible situations. For the first part of this work, we avoid giving a direct definition of privacy, and instead say what privacy is not. We define a notion of *non-privacy* – a situation we believe should not be allowed in any reasonable private database setting. We say a database is non-private, if a computationally-bounded adversary can expose a $1-\varepsilon$ fraction of the database entries for all constant $\varepsilon > 0$. In other words, non-privacy excludes even very weak notions of privacy.

For the second part of this work, we give a definition of privacy with respect to a bounded adversary with no prior knowledge. Our definition of privacy tries to capture the fact that such an adversary should not be able to predict the ith bit, regardless of the content of the rest of the database. The database-adversary is modelled as a game that consists of two phases. In the first phase, the adversary queries the database (adaptively). At the end of this phase, the adversary outputs an index i, of the private function $\pi_i(d_1,..,d_n) = d_i$ it intends to guess. In the second phase, the adversary is given the query-response transcript of the first phase plus all but the ith database entries, and outputs a guess. Privacy is preserved if the adversary fails to guess d_i with high probability.

[2] As is seen by the following example: Let $d_i \in \{0,1\}$. Consider an estimator $\tilde{d_i} = d_i + E \cdot e$ where $e \in_R \{-1,1\}$ and E is a large even number. We have that although $\mathbf{Var}[\tilde{d_i}]$ is very large, d_i may be exactly computed given just a single a sample of $\tilde{d_i}$, just by checking if the outcome is even. See related discussion in [3].

1.3 This Work

We use a general model for statistical databases as follows. Denoting the database content by $(d_1, .., d_n) \in \{0,1\}^n$, a query $q \subseteq [n]$ is answered by $\mathcal{A}(q)$. We allow the database to perturb its answers in such a way that hides values of specific bits, but (hopefully) still yields meaningful information about sums of bits. I.e. $\mathcal{A}(q)$ may be a perturbed answer with additive perturbation $|\mathcal{A}(q) - \sum_{i \in q} d_i|$ (this notion captures *any* perturbation method, including the output perturbation and the data perturbation methods). In this model we present tight impossibility results. In fact we show that unless the perturbation is as large as $\sqrt{n}$ (possibly totally ruining the database usability) almost the whole database can be recovered by a polynomial adversary. Thus, any reasonable definition of privacy cannot be maintained.

We proceed to define a bounded adversary model, where privacy may be achieved with lower perturbation magnitude. We demonstrate the feasibility of this approach on random databases.

Impossibility Results as a Reconstruction Problem. Our main result is a polynomial reconstruction algorithm from noisy subset-sums, that – in case the answer to queries are within additive perturbation error $\mathcal{E} = o(\sqrt{n})$ – succeeds in reconstructing a 'candidate' database c whose Hamming distance from d is at most $o(n)$. In particular, c agrees with d on over 99% of their bits.

The reconstruction problem we solve may be viewed as a variant of list-decoding of error correcting codes, where the task is to come up with all preimages whose codewords differ from a given (noisy) word on at most r locations [19, 13]. In our case, the error correcting code is in the form of subset-sums $\mathcal{C}(d) = \{a_q\}_{q \subseteq [n]} = \{\sum_{i \in q} d_i\}_{q \subseteq [n]}$, and has exponential length. Given a *noisy* encoding $\{\tilde{a}_q\}_{q \subseteq [n]}$ of the bits $d_1, .., d_n$, we wish to efficiently decode this encoding. Here, the list of feasible candidates is itself *exponential* in size, so our goal is not to compute the entire list, but to come up with *one* such candidate.

1.4 Organization of this paper

We begin in Section 2 by presenting a model of statistical databases and perturbation. In Section 3 we show our lower bounds on the perturbation needed for privacy, depending on the adversary power. In Section 4 we discuss privacy in the case of a (time/query) bounded adversary. The appendices include a proof of a technical lemma and a short discussion of a variant of the auditor idea.

2 Preliminaries

Notation. We denote by $\mathsf{neg}(n)$ (read: negligible) a function that is asymptotically smaller than any inverse polynomial. I.e. for all $c > 0$, for all sufficiently large n it holds that $\mathsf{neg}(n) < 1/n^c$. We use $\mathbf{dist}(c, d)$ for the Hamming distance of two binary strings $c, d \in \{0,1\}^n$. I.e. $\mathbf{dist}(c, d) = |\{i : c_i \neq d_i\}|$. We write $\tilde{O}(T(n))$ for $T(n) \cdot \mathbf{polylog}(n)$.

2.1 The Model - Statistical Databases and Statistical Queries

A statistical database is a query-response mechanism that enables users to access its content via statistical queries. We focus on binary databases, where the content is of n binary (0-1) entries, and give its appropriate definition below. A statistical query specifies a subset of entries; the answer to the statistical query is the number of entries having value 1 among those specified in it. Users issue statistical queries to the database; the response to these queries is computed by the database algorithm, that accesses the database content. This algorithm may keep additional information (in form of an *internal state*), and may update its state whenever it is invoked. We are not concerned with the specifics of the database language, and its indexing mechanisms. Instead, we assume that users are able to specify any subset of database entries. Our model is consistent with previous models in the literature, see e.g. [2].

Definition 1. *Let $d = (d_1, \ldots, d_n) \in \{0,1\}^n$. A* **statistical query** *(*query *for short) is a subset $q \subseteq [n]$. The* **(exact) answer to a query** *q is the sum of all database entries specified by q i.e. $a_q = \sum_{i \in q} d_i$.*

A **statistical database** *(*database *for short) $\mathcal{D} = (d, \mathcal{A})$ is a query-response mechanism. The response to a query q is $\mathcal{A}(q, d, \sigma)$ where σ is the internal state of the algorithm $\mathcal{A}$ (the internal state σ may be affected by the computation). When it is clear from the context, we usually omit d, σ and write $\mathcal{A}(q)$ for $\mathcal{A}(q, d, \sigma)$.*

We now define the quality of the database algorithm $\mathcal{A}$ in terms of its perturbation magnitude:

Definition 2. *An answer $\mathcal{A}(q)$ is* **within $\mathcal{E}$ perturbation** *if $|a_q - \mathcal{A}(q)| \leq \mathcal{E}$.*

We say that algorithm $\mathcal{A}$ is a **within $\mathcal{E}$ perturbation** *if for all queries $q \subseteq [n]$ the answer $\mathcal{A}(q)$ is within $\mathcal{E}$ perturbation*[3].

We write $\mathcal{M}^{\mathcal{A}}$ to denote a Turing Machine $\mathcal{M}$ with access to database algorithm $\mathcal{A}$. The time complexity of $\mathcal{M}^{\mathcal{A}}$ is defined as usual, where each call to $\mathcal{A}$ costs a unit time.

2.2 A Probability Tool

Theorem 1 (Azuma's Inequality, [5, Chapter 7]). *Let $0 = s_0, s_1, .., s_n$ be random variables so that $E[s_i | s_{i-1}] = s_{i-1}$, and so that $|s_i - s_{i-1}| \leq 1$, then*

$$\Pr[|s_n| > \lambda\sqrt{n}] < 2e^{-\lambda^2/2} .$$

3 Impossibility results

The main result in this section is a lower bound on the perturbation needed to maintain any reasonable notion of privacy. We show that any database algorithm that is within $o(\sqrt{n})$ perturbation, is non private with respect to polynomial time adversaries. More accurately, we show that whenever the perturbation is smaller than $\sqrt{n}$, a polynomial number of queries can be used to efficiently reconstruct a 'good' approximation of the entire database.

We first formalize what we mean by a *non-private database*. Intuitively, for a database to be non-private, an adversary should be able to efficiently reconstruct the entire database very accurately.

Definition 3 (Non-Privacy). *A database $\mathcal{D} = (d, \mathcal{A})$ is $t(n)$-*non-private *if for every constant $\varepsilon > 0$ there exists a probabilistic Turing Machine $\mathcal{M}$ with time complexity $t(n)$ so that*

$$\Pr[\mathcal{M}^{\mathcal{A}}(1^n) \textit{ outputs } c \textit{ s.t. } \mathbf{dist}(c, d) < \varepsilon n] \geq 1 - \mathsf{neg}(n) .$$

The probability is taken over the coin tosses of $\mathcal{A}$ and $\mathcal{M}$.

Note that Definition 3 implicitly bounds the number of queries the adversary issues by $t(n)$.

[3] For most of our results it suffices that $\mathcal{A}(q)$ is within $\mathcal{E}$ perturbation for all but a negligible fraction of the queries q.

Remark 1. In our analysis below we further restrict the adversary by making its queries non-adaptive. I.e. the adversary first specifies its entire set of queries, and then is answered by the database access algorithm.

3.1 Exponential Adversary

We begin with a result pertaining to very small databases, where the adversary may issue all possible queries. We show that such an adversary can violate privacy even with a nearly linear perturbation magnitude.

Theorem 2. *Let $\mathcal{D} = (d, \mathcal{A})$ be a database where $\mathcal{A}$ is within $o(n)$ perturbation. Then $\mathcal{D}$ is* $\mathbf{exp}(n)$*-non-private.*

Proof. Let $\mathcal{A}$ be within $\mathcal{E}$ perturbation for $\mathcal{E} = o(n)$. Let $\mathcal{M}$ be the following algorithm:

```
[Query Phase]
For all $q \subseteq [n]$: let $\tilde{a}_q \leftarrow \mathcal{A}(q)$.

[Weeding Phase]
For all $c \in \{0,1\}^n$: if $|\sum_{i \in q} c_i - \tilde{a}_q| \leq \mathcal{E}$ for all $q \subseteq [n]$ then output $c$ and halt.
```

It is easy to verify that $\mathcal{M}$ runs in exponential time. Note that $\mathcal{M}$ always halts and outputs some 'candidate' c – because the real database d is never weeded out. We now show that the output candidate c satisfies $\mathbf{dist}(d, c) \leq 4\mathcal{E} = o(n)$. Assume this is not the case, i.e. $\mathbf{dist}(d, c) > 4\mathcal{E}$. Let $q_0 = \{i \mid d_i = 1, c_i = 0\}$ and $q_1 = \{i \mid d_i = 0, c_i = 1\}$. Since $|q_1| + |q_0| = \mathbf{dist}(d, c) > 4\mathcal{E}$, at least one of the disjoint sets q_1, q_2 has size $2\mathcal{E} + 1$ or more. W.l.o.g. assume $|q_1| > 2\mathcal{E}$. We have that $\sum_{i \in q_1} d_i = 0$ and hence it must be that $\tilde{a}_{q_1} \leq \mathcal{E}$. On the other hand $\sum_{i \in q_1} c_i = |q_1| > 2\mathcal{E}$. We get that $|\sum_{i \in q_1} c_i - \tilde{a}_{q_1}| > \mathcal{E}$, contradicting the fact that c survives the weeding phase. □

3.2 Polynomially Bounded Adversary – $\Omega(\sqrt{n})$ Perturbation Needed for Privacy

We next turn to the more interesting (and perhaps realistic) case where the adversary is polynomially bounded. We show that a minimal perturbation level of $\Omega(\sqrt{n})$ is necessary for achieving even weak privacy in our model. More concretely, we show that any database algorithm that is within $\mathcal{E} = o(\sqrt{n})$ perturbation is non-private. We prove this by presenting for every $\varepsilon > 0$ a linear-programming algorithm, with which we reconstruct, in polynomial-time, a candidate database c and prove that $\mathbf{dist}(c, d) < \varepsilon n$.

Theorem 3. *Let $\mathcal{D} = (d, \mathcal{A})$ be a database where $\mathcal{A}$ is within $o(\sqrt{n})$ perturbation then $\mathcal{D}$ is* $\mathbf{poly}(n)$*-non-private.*

Proof. Let $\mathcal{A}$ be within $\mathcal{E}$ perturbation for $\mathcal{E} = o(\sqrt{n})$. We fix $\varepsilon > 0$ arbitrarily, and show how to find some candidate database c for which $\mathbf{dist}(c, d) < \varepsilon n$ with high probability. Let $\mathcal{M}^{\mathcal{A}}$ be the following algorithm:

[Query Phase]
Let $t = n(\log n)^2$. For $1 \leq j \leq t$ choose uniformly at random $q_j \subseteq_R [n]$, and set $\tilde{a}_{q_j} \leftarrow \mathcal{A}(q_j)$.

[Weeding Phase]
Solve the following linear program with unknowns $c_1, \ldots, c_n$:

$$\begin{array}{cl} \tilde{a}_{q_j} - \mathcal{E} \leq \sum_{i \in q_j} c_i \leq \tilde{a}_{q_j} + \mathcal{E} & \text{for } 1 \leq j \leq t \\ 0 \leq c_i \leq 1 & \text{for } 1 \leq i \leq n \end{array} \tag{1}$$

[Rounding phase]
Let $c'_i = 1$ if $c_i > 1/2$ and $c'_i = 0$ otherwise. Output c'.

It is easy to see that the LP in our algorithm always has a solution (in particular $c = d$ is a feasible solution), and hence the algorithm always has an output c'. We next prove that $\mathbf{dist}(c', d) < \varepsilon n$. We use the probabilistic method to show that a random choice of $q_1, .., q_t$ will weed out *all* possible candidate databases c that are far from the original one.

Fix a precision parameter $k = n$ and define $K = \{0, \frac{1}{k}, \frac{2}{k}, .., \frac{k-1}{k}, 1\}$. For any $x \in [0,1]^n$ denote by $\bar{x} \in K^n$ the vector obtained by rounding each coordinate of x to the nearest integer multiple of $\frac{1}{k}$. Applying the triangle inequality with Eq. (1) we get that for $1 \leq j \leq t$,

$$\begin{aligned} &|\sum_{i \in q_j} (\bar{c}_i - d_i)| \leq \\ &\quad |\sum_{i \in q_j} (\bar{c}_i - c_i)| + |\sum_{i \in q_j} c_i - \tilde{a}_{q_j}| + |\tilde{a}_{q_j} - \sum_{i \in q_j} d_i| \leq \\ &\quad \frac{|q_j|}{k} + \mathcal{E} + \mathcal{E} \leq 1 + 2\mathcal{E}\,. \end{aligned}$$

For any $x \in [0,1]^n$, we say that a query $q \subseteq [n]$ *disqualifies* $\bar{x}$ if $|\sum_{i \in q} (\bar{x}_i - d_i)| > 2\mathcal{E} + 1$. If q happens to be one of the queries in our algorithm, this means that x is an invalid solution to the LP system. We now show that if x is far from d on many coordinates, then (with high probability) $\bar{x}$ is disqualified by at least one of the queries $q_1, \ldots, q_t$. We use the following lemma (see proof in Appendix A):

Lemma 4 (Disqualifying Lemma). *Let $x, d \in [0,1]^n$ and $\mathcal{E} = o(\sqrt{n})$. If $\Pr_i[|x_i - d_i| \geq \frac{1}{3}] > \varepsilon$ then there exists a constant $\delta > 0$ such that*

$$\Pr_{q \subseteq_R [n]} \left[|\sum_{i \in q} (x_i - d_i)| > 2\mathcal{E} + 1 \right] > \delta\,.$$

Define the set of 'discrete' x's that are 'far' from d:

$$X = \left\{ x \in K^n \mid \Pr_i[|x_i - d_i| \geq \frac{1}{3}] > \varepsilon n \right\}\,.$$

Consider $x \in X$. By Lemma 4 there exists a constant $\delta > 0$ such that $\Pr_{q \subseteq_R [n]}[q \text{ disqualifies } x] \geq \delta$. By choosing t independent random queries $q_1, .., q_t$, of $[n]$ we get that at least one of them disqualifies x with probability $1 - (1-\delta)^t$. I.e. for each $x \in X$ there is just a tiny $(1-\delta)^t$ fraction of the selections $q_1, \ldots, q_t$ do not disqualify it. Taking the union bound over X, noting $|X| \leq |K|^n = (k+1)^n$, we have that

$$\Pr_{q_1,..,q_t \subseteq_R [n]} [\forall x \in X \; \exists i, \; q_i \text{ disqualifies } x] \geq$$
$$1 - (k+1)^n (1-\delta)^t > 1 - \mathsf{neg}(n)\,.$$

The last inequality holds assuming we take t to be large enough ($t = n(\log n)^2$ will work).

To complete the proof, observe that $\bar{c}$ is *not* disqualified by any of the random subsets $q_1, .., q_t \subseteq [n]$ chosen by our algorithm (recall that c is the solution to the LP system, and $\bar{c}$ is the vector obtained from rounding it to the nearest integer multiple of $\frac{1}{k}$). Thus, if these $q_1, .., q_t$ disqualify all $x \in X$, it must be that $\bar{c} \notin X$ so $\mathbf{dist}(c', d) \leq \varepsilon n$. □

Note that the proof relies on the fact that a random subset of linear size deviates from the expectation by roughly $\sqrt{n}$. We are bounding from *below* the deviation from expectation, converse to the standard use of tail-inequalities in which the upper bound is needed.

We stress that the reconstruction algorithm presented in the proof above is oblivious of the perturbation method used by the database and of the database distribution.

3.3 Tightness of the Impossibility Results

Theorem 3 shows that for any database distribution a perturbation of magnitude $\Omega(\sqrt{n})$ is necessary for having (even a very weak notion of) privacy. We next show that this bound is tight by exemplifying a database algorithm that is within $\tilde{O}(\sqrt{n})$ perturbation and is private against polynomial adversaries in the strongest possible sense. That is, if the database is queried by a polynomial-time machine then with extremely high probability it does not reveal *any* information about the data. Note that for showing tightness we assume a specific distribution on the database, namely the uniform distribution over all strings of n bits[4].

Let $d \in_R \{0,1\}^n$. Set the perturbation magnitude to $\mathcal{E} = \sqrt{n} \cdot (\log n)^{1+\varepsilon} = \tilde{O}(\sqrt{n})$. Consider the database $\mathcal{D} = (d, \mathcal{A})$ with algorithm $\mathcal{A}$ as follows: (i) On input a query $q \subseteq [n]$ algorithm $\mathcal{A}$ computes $a_q = \sum_{i \in q} d_i$. (ii) If $|a_q - \frac{|q|}{2}| < \mathcal{E}$ then $\mathcal{A}$ returns $\frac{|q|}{2}$ (iii) Otherwise $\mathcal{A}$ returns a_q. It is easy to see that $\mathcal{A}$ is within perturbation $\mathcal{E}$. Moreover, for any probabilistic polynomial time machine $\mathcal{M}$, the probability (over d and the coin tosses of $\mathcal{M}$) that $\mathcal{A}$ acts according to rule (iii) is negligible.

Note that, although guaranteeing perturbation magnitude of $\tilde{O}(\sqrt{n})$, the above algorithm renders the database effectively useless – users are extremely unlikely to get any non-trivial information by querying the database, and hence they are unlikely to compute any non-trivial functionality of it.

3.4 A Closer Look on the Threshold

The private database of the previous subsection imposes a disturbing question – must a private database be *useless*? We show that is not necessarily the case by demonstrating a database algorithm that guarantees some privacy and some usability. For this section we slightly relax the requirements of Definition 2, and require that $\mathcal{A}(q)$ is within perturbation $\mathcal{E}$ for most q:

$$\Pr_{q \subseteq [n]} [\mathcal{A}(q) \text{ is within perturbation } \mathcal{E}] = 1 - \mathsf{neg}(n)\;.$$

[4] Such an assumption was not needed for the proof of Theorem 3, the adversary there is oblivious both of the database distribution and the perturbation method.

Note that Theorem 3 holds also with this relaxed definition.

Let $\mathcal{DB}$ be the uniform distribution over $\{0,1\}^n$, and $d \in_R \mathcal{DB}$. Recall that the database algorithm may (see Definition 1) use an internal *state* storage σ to keep information between its invocations. Algorithm $\mathcal{A}$ maintains such a state, that it initializes upon the first call (and never updates – unless new entries are added to the database). The internal state of $\mathcal{A}$ consists of an n bits $d' = (d'_1, \ldots, d'_n)$, where d'_i is chosen to be d_i with probability $1/2 + \delta$ and $1 - d_i$ otherwise. On input a query $q \subseteq [n]$ algorithm $\mathcal{A}$ answers $\tilde{a} = \sum_{i \in q} d'_i$. Note that while $\mathcal{A}$ is within $\tilde{O}(\sqrt{n})$ perturbation, the above database has some usability. For instance it is possible to compute a subset S of $[n]$ so that significantly more than half of the entries specified by S are set to 1 in the original database.

The CD Model. The database algorithm above essentially creates a 'private' version of the database d', and then answers queries using d'. Note that a user may retrieve the entire content of d' by querying $q_i = \{i\}$ for $1 \leq i \leq n$, after which she may answer all her other queries by herself. This result indicates that it is in some cases possible to achieve privacy in a *CD model*, where users get a 'private' version of the database (written on a CD), which they may manipulate (say, without being restricted to statistical queries).

4 Feasibility Results – Preserving Privacy in a Bounded Model

In Section 3 we saw that if the querying adversary has exponential computational power, a linear perturbation magnitude is needed for preserving privacy; and that a $\sqrt{n}$ perturbation magnitude is needed for preserving privacy with respect to polynomially-bounded adversaries. A natural question is whether further restricting the adversary complexity enables achieving privacy using a smaller perturbation[5]. In this section we answer this question positively. We present a database access algorithm that preserves privacy with respect to an adversary whose running time is no more than $\mathcal{T}(n)$ for an arbitrary $\mathcal{T}$. Our database algorithm uses a perturbation error of roughly $\sqrt{\mathcal{T}(n)}$.

To show such a 'feasibility' result, it seems that one has no choice but to make an assumption regarding the adversary's a-priori knowledge about the database. Otherwise, considering a database $(d, \mathcal{A})$ where it is known that d is either 1^n or 0^n, unless the perturbation magnitude is at least $n/2$ a single query obviously reveals the entire database. We model prior knowledge as having the database drawn from some arbitrary distribution $\mathcal{DB}$.

Our definition of privacy is very strong in that it requires that even if the adversary happens to learn all the database content except the ith bit she still cannot predict the ith bit with good probability. This scenario is modelled by a two-phase adversary. In the first phase, the adversary is allowed to adaptively query the database. At the end of this phase the adversary commits to a challenge – an index i of the bit it intends to guess. In the second phase, all the database entries except the ith bit are revealed to the adversary. The adversary succeeds if it outputs a correct d_i. The definition models the two phases by two Turing Machines $\mathcal{M}_1, \mathcal{M}_2$, of which only $\mathcal{M}_1$ has access to the database algorithm. More formally:

Definition 4 (Privacy). *Let $\mathcal{DB}$ be a distribution over $\{0,1\}^n$ and let d be drawn according to $\mathcal{DB}$. A database $\mathcal{D} = (d, \mathcal{A})$ is $(\mathcal{T}(n), \delta)$-*private *if for every pair of probabilistic Turing Machines*

[5] One way to interpret these restrictions is by considering an adversary of fixed power acting on a bigger and bigger databases.

$\mathcal{M}_1^{\mathcal{A}}, \mathcal{M}_2$ with time complexity $\mathcal{T}(n)$, it holds that

$$\Pr\left[\begin{array}{l}\mathcal{M}_1^{\mathcal{A}}(1^n) \text{ outputs } (i, \mathit{view}) \ ; \\ \mathcal{M}_2(\mathit{view}, d^{-i}) \text{ outputs } d_i\end{array}\right] < \frac{1}{2} + \delta$$

where $d^{-i} = (d_1, \ldots, d_{i-1}, d_{i+1}, \ldots, d_n)$. The probability is taken over the choice of d from $\mathcal{DB}$ and the coin tosses of all machines involved.

In the following section we focus on the case where the adversary has no prior information. This is modelled by assuming that the database is drawn from the uniform distribution over n-bit strings. we write $\log^k n = (\log n)^k$.

Theorem 5. *Let $\mathcal{T}(n) > polylog(n)$, and let $\delta > 0$. Let $\mathcal{DB}$ be the uniform distribution over $\{0,1\}^n$, and $d \in_R \mathcal{DB}$. There exists a $\tilde{O}(\sqrt{\mathcal{T}(n)})$-perturbation algorithm $\mathcal{A}$ such that $\mathcal{D} = (d, \mathcal{A})$ is $(\mathcal{T}(n), \delta)$-private.*

To prove the theorem, we demonstrate a database algorithm that answers each query q by adding a random perturbation to the exact value $a_q = \sum_{i \in q} d_i$. The perturbation $\mathcal{A}$ adds is independent of previous queries, hence $\mathcal{A}$ does not maintain a *state* between invocations.

In the main part of the proof, we claim that the "confidence" a bounded adversary gains with respect to a value it tries to predict remains low. To show that, we define a random walk on a line, corresponding to how the adversary's confidence evolves as the adversary queries the database. Analyzing the random walk, we prove that, with extremely high probability, a sub-linear number of steps is not sufficient for reaching the confidence required for violating privacy.

of Theorem 5. Let $\mathcal{T}(n)$ be some running-time bound, e.g. $\mathcal{T}(n) = n$, and define $R = \mathcal{T}(n) \cdot \log^{\mu} n$ for some $\mu > 0$ (taking $\mu = 9$ will work). Let $\mathcal{A}$ be the following algorithm with input a query q:

1. Let $a_q = \sum_{i \in q} d_i$.

2. Generate a perturbation value: Let $(e_1, \ldots, e_R) \in_R \{0,1\}^R$ and $\mathcal{E} \leftarrow \sum_{i=1}^{R} e_i - R/2$.

3. Return $a_q + \mathcal{E}$.

Note that $\mathcal{E}$ is a binomial random variable with $\mathbf{E}[\mathcal{E}] = 0$ and variance $\sqrt{R}$ so that $\Pr[|\mathcal{E}| > \log^2 n \cdot \sqrt{R}] < \mathsf{neg}(n)$, hence $\mathcal{A}$ is a $\tilde{O}(\sqrt{\mathcal{T}(n)})$-perturbation algorithm.

We now turn to prove that $(d, \mathcal{A})$ is private. Let $\mathcal{M}_1^{\mathcal{A}}, \mathcal{M}_2$ be as in Definition 4. W.l.o.g, assume that $\mathcal{M}_1^{\mathcal{A}}$ records all its queries and their answers in the *view* variable it transmits to $\mathcal{M}_2$. Note that $\mathcal{M}_1^{\mathcal{A}}$ does not get any information issuing small queries, hence we assume it issues only large queries – at most $t = \mathcal{T}(n)$ queries which we denote $q_1, \ldots, q_t$. Let $a_1 = \mathcal{A}(q_1), \ldots, a_t = \mathcal{A}(q_t)$ be the answers to these queries.

In the following we analyze the (a-posteriori) probability p that $d_i = 1$ given the query-answer pairs (q_ℓ, a_ℓ) and $d_1, \ldots, d_{i-1}, d_{i+1}, \ldots, d_n$. Note that p bounds the correct prediction probability of $\mathcal{M}_2$. We show that $1/2 - \delta \leq p \leq 1/2 + \delta$, hence $\mathcal{D} = (d, \mathcal{A})$ is private. For the analysis we run a mental experiment in which we first reveal $d_1, \ldots, d_{i-1}, d_{i+1}, \ldots, d_n$, and then the query-answer pairs (q_ℓ, a_ℓ) one by one (in the same order $\mathcal{M}^A$ issued these queries).

For $0 \leq \ell \leq t$ let p_ℓ be the probability that $d_i = 1$ given the answers to the first ℓ queries $q_1, \ldots, q_\ell$:

$$p_\ell = \Pr[d_i = 1 | a_1, \ldots, a_\ell].$$

We will now use the following fact. Let A_1, A_2, D be events where $\Pr[A_1, A_2|D] = \Pr[A_1|d]\cdot\Pr[A_2|d]$. Then Applying Bayes Rule[6], and using the independence of A_1, A_2 when conditioned on D we get:

$$\begin{aligned}\Pr[D|A_1, A_2] &= \frac{\Pr[A_1, A_2|D]\cdot\Pr[D]}{\Pr[A_1, A_2]}\\ &= \frac{\Pr[A_1|D]\cdot\Pr[A_2|D]\cdot\Pr[D]}{\Pr[A_1, A_2]}\\ &= \frac{\Pr[A_1|D]\cdot\Pr[D|A_2]\cdot\Pr[A_2]}{\Pr[A_1, A_2]}\,.\end{aligned}$$

Since a_ℓ is answered (by $\mathcal{A}$) independently of the previous answers $a_1, \ldots, a_{\ell-1}$ we plug in the events $A_1 = \{a_1 = \mathcal{A}(q_1), \ldots, a_{\ell-1} = \mathcal{A}(q_{\ell-1})\}$, $A_2 = \{a_\ell = \mathcal{A}(q_\ell)\}$ and $D = \{d_i = 1\}$ to the above proposition to get

$$p_\ell = \Pr[D|A_1, A_2] = p_{\ell-1}\cdot\frac{\Pr[a_\ell|d_i = 1]\cdot\Pr[a_1, \ldots, a_{\ell-1}]}{\Pr[a_1, \ldots, a_\ell]} \tag{2}$$

and similarly (this time with $D = \{d_i = 0\}$),

$$1 - p_\ell = (1 - p_{\ell-1})\cdot\frac{\Pr[a_\ell|d_i = 0]\cdot\Pr[a_1, \ldots, a_{\ell-1}]}{\Pr[a_1, \ldots, a_\ell]}\,. \tag{3}$$

We define the adversary's *confidence in $d_i = 1$ after ℓ queries* as the log-ratio of the a-posteriori probabilities that $d_i = 1$ and $d_i = 0$ i.e.

$$\mathrm{conf}_\ell \overset{def}{=} \log\left(p_\ell/(1 - p_\ell)\right).$$

Note that $\mathrm{conf}_0 = \log\left(p_0/(1-p_0)\right) = \log\left(\frac{1}{2}/(1-\frac{1}{2})\right) = 0$ and that $\mathrm{conf}_\ell = \mathrm{conf}_{\ell-1}$ whenever $i \notin q_\ell$ (we assume that w.l.o.g. no such 'useless' queries occur). $\mathcal{D}$ is private if with high probability $|\mathrm{conf}|$ never reaches a high value. I.e. $|\mathrm{conf}_\ell| < \delta' = \log\left(\frac{1/2+\delta}{1/2-\delta}\right)$ for all $0 < \ell \leq t$. Substituting Eq. (2) for p_ℓ and Eq. (3) for $1 - p_\ell$ we get the following equation that describes how the random variable conf evolves:

$$step_\ell \overset{def}{=} \mathrm{conf}_\ell - \mathrm{conf}_{\ell-1} = \log\left(\frac{\Pr[a_\ell|d_i = 1]}{\Pr[a_\ell|d_i = 0]}\right)\,.$$

Note that $\mathrm{conf}_l = \sum_{j=0}^{l} step_l$ and that $step_i$ are mutually independent.

The sequence $0 = \mathrm{conf}_0, \ldots, \mathrm{conf}_t$ defines a random walk on the line, starting from zero, and advancing according to the random variable $step_\ell$. To complete the proof we show that, with extremely high probability, more than t steps are needed to reach confidence δ'.

Assume $d_i = 1$ (a similar analysis holds for the case $d_i = 0$). Let $k = a_\ell - \sum_{j\neq i} d_j - 1$. Since for $\sigma \in \{0, 1\}$

$$\begin{aligned}\Pr[A(q_\ell) = a_\ell \,|\, d_i = \sigma] &= \Pr[\mathcal{E} = a_\ell - \sum_{j\neq i} d_j - \sigma]\\ &= \Pr[\mathcal{E} = k + 1 - \sigma]\end{aligned}$$

we get that the random variable $step_\ell$ takes the value

$$\log\left(\binom{R}{k}\Big/\binom{R}{k+1}\right) = \log\left((k+1)/(R-k)\right)$$

[6] I.e. $\Pr[E_1|E_2]\cdot\Pr[E_2] = \Pr[E_2|E_1]\cdot\Pr[E_1]$.

with probability $\binom{R}{k}/2^R$.

Although k may take values in $[0, \ldots, R]$, the probability that it largely deviates from $R/2$ is small. Hence, in the rest of the proof we neglect this case, and assume that k is always in the interval $K = [R/2 - \sqrt{R} \cdot \log^2 n, R/2 + \sqrt{R} \cdot \log^2 n]$. Conditioned on this we show that the expectation of step_ℓ is very small, and that $|\mathit{step}_\ell|$ is small. We use Azuma's inequality to conclude the proof. Details follow.

To bound the expectation of step_ℓ compare it with a random variable B that takes the value $\log(k/(R-k))$ with probability $\binom{R}{k}/2^R$. Clearly $\mathbf{E}[B] = 0$ because $\Pr[B = \log(k/(R-k))] = \Pr[B = -\log(k/(R-k))]$ and hence

$$\begin{aligned}\mathbf{E}[\mathit{step}_\ell] &= \mathbf{E}[\mathit{step}_\ell - B] = \sum \frac{\binom{R}{k}}{2^R} \cdot \log((k+1)/k) \\ &= \sum \frac{\binom{R}{k}}{2^R} \cdot \log(1 + 1/k)\end{aligned}$$

Since for small x $\log(1+x) \approx x$ we have $\log((k+1)/k) = O(1/R)$ for $k \in K$, and

$$\mathbf{E}[\mathit{step}_\ell] = O(1/R) \sum \frac{\binom{R}{k}}{2^R} = O(1/R) \ .$$

Thus,

$$\mathbf{E}[\sum_{l=1}^{t} \mathit{step}_\ell] \leq t \cdot O(1/R) = O(1/(\log^\mu n)) \ .$$

To bound $|\mathit{step}_\ell|$ note that it reaches its maximum on the ends of the interval K, hence we get $|\mathit{step}_\ell| < \log((R/2 + \sqrt{R} \cdot \log^2 n)/(R/2 - \sqrt{R} \cdot \log^2 n)) = \log(1 + \log^2 n/\sqrt{R}) = O(\log^2 n/\sqrt{R})$.

Define the random variable $s_\ell = \sum_{j=1}^{\ell} \mathit{step}_j - \mathbf{E} \sum_{j=1}^{\ell} \mathit{step}_j$. This is a martingale with expectation zero, satisfying $|s_\ell - s_{\ell-1}| = O(\log^2 n)^2/\sqrt{R})$. Using Azuma's inequality (Theorem 1) we get that $\Pr[|s_\ell| > \lambda \cdot O(\log^2 n/\sqrt{R}) \cdot \sqrt{\ell}] < 2e^{-\lambda^2/2}$.

Setting $\lambda = \log^2 n$ we get that $\Pr[|s_\ell| > \frac{1}{\log^{\mu/2-4} n}] < \mathsf{neg}(n)$, so choosing $\mu = 9$ makes $|s_\ell| < O(1/\log n)$ almost always. Hence,

$$\begin{aligned}\Pr[\mathrm{conf}_l > O(1/\log n)] &= \Pr\left[\sum_{j=0}^{l} \mathit{step}_l > O(1/\log n)\right] \\ &\leq \Pr\left[|s_\ell| > O(1/\log n) - \mathbf{E}[\sum_{j=0}^{l} \mathit{step}_l]\right] \leq \mathsf{neg}(n)\end{aligned}$$

Taking the union bound (for $0 < \ell \leq t$) completes the proof. □

5 Acknowledgments

We thank Ronitt Rubinfeld for suggesting the problem of database privacy and for pointing to us some of the work done in the area. We thank Cynthia Dwork for in-depth discussions. The work in Section 4 is joint with Cynthia Dwork.

References

[1] J. O. Achugbue and F. Y. Chin, *The effectiveness of output modification by rounding for protection of statistical databases*, INFOR 17, 3: 209-218, 1979.

[2] N. R. Adam and J. C. Wortmann, *Security-Control Methods for Statistical Databases: A Comparative Study*, ACM Computing Surveys 21(4): 515-556 (1989).

[3] D. Agrawal and C. C. Aggarwal, *On the design and quantification of privacy preserving data mining algorithms*, Symposium on Principles of Database Systems, 2001.

[4] R. Agrawal and R. Srikant, *Privacy-preserving data mining*, Proc. of the ACM SIGMOD Conference on Management of Data, pages 439–450, 2000.

[5] N. Alon and J. H. Spencer, **The probabilistic method**, Wiley-Interscience [John Wiley & Sons], New York, second edition, 2000.

[6] L. L. Beck, *A security mechanism for statistical databases*, ACM TODS, 5(3):316–338, September 1980.

[7] F. Y. Chin and G. Ozsoyoglu, *Auditing and infrence control in statistical databases*, IEEE Trans. Softw. Eng., SE-8(6):113–139, April 1982.

[8] D. Denning, P. Denning, and M. Schwartz, *The tracker: A threat to statistical database security*, ACM Trans. on Database Systems, 4(1):76–96, March 1979.

[9] D. E. Denning, *Secure statistical databases with random sample queries*, ACM Transactions on Database Systems, 5(3):291–315, September 1980.

[10] D. E. Denning, **Cryptography and data security**, Addison-Wesley, Reading MA, 1982.

[11] I. Fellegi, *On the question of statistical confidentiality*, Journal of the American Statistical Association, 1972, pp. 7-18.

[12] O. Goldreich, S. Micali and A. Wigderson, *How to Play any Mental Game or A Completeness Theorem for Protocols with Honest Majority*, STOC 1987: 218-229.

[13] V. Guruswami and M. Sudan, *Improved Decoding of Reed-Solomon and Algebraic-Geometric Codes*, IEEE Symposium on Foundations of Computer Science, 1998, 28–39.

[14] J. M. Kleinberg, C. H. Papadimitriou and P. Raghavan, *Auditing Boolean Attributes*, PODS 2000: 86-91.

[15] C. K. Liew, U. J. Choi, and C. J. Liew, *A data distortion by probability distribution*, ACM TODS, 10(3):395–411, 1985.

[16] E. Lefons, A. Silvestri, and F. Tangorra, *An analytic approach to statistical databases*, In 9th Int. Conf. Very Large Data Bases, pages 260– 274. Morgan Kaufmann, Oct-Nov 1983.

[17] S. Reiss, *Practical Data Swapping: The First Steps*, ACM TODS, 9, 1, pp. 20–37, 1984.

[18] A. Shoshani, *Statistical databases: Characteristics, problems and some solutions*, Proceedings of the 8th International Conference on Very Large Data Bases (VLDB'82), pages 208–222, 1982.

[19] M. Sudan, *Decoding of Reed Solomon Codes beyond the Error-Correction Bound*, Journal of Complexity, 13 (1), 180–193, 1997.

[20] J. Traub, Y. Yemini, H. Wozniakowksi, *The Statistical Security of a Statistical Database*, ACM TODS, 9, 4 pp. 672–679, 1984. 11

[21] A. Yao, *Protocols for Secure Computations (Extended Abstract)*, FOCS 1982: 160-164.

A The Disqualifying Lemma

Lemma 4. *Let $x, d \in [0,1]^n$ and $\mathcal{E} = o(\sqrt{n})$. If $\Pr_i[|x_i - d_i| \geq \frac{1}{3}] > \varepsilon$ then there exists a constant $\delta > 0$ such that*

$$\Pr_{q \subseteq_R [n]} \left[|\sum_{i \in q} (x_i - d_i)| > 2\mathcal{E} + 1 \right] > \delta\,.$$

Proof. Let $X_1, .., X_n$ be independent random variables such that X_i takes value $x_i - d_i$ with probability $\frac{1}{2}$ and 0 with probability $\frac{1}{2}$, and define $X \overset{def}{=} \sum_{i=1}^n X_i$. Observe that all we need to prove is that $\Pr[|X| > 2\mathcal{E} + 1] > \Omega(1)$.

We divide the proof to two cases according to the size of $\mathbf{E}[X]$. Let $T \geq \sqrt{\frac{\varepsilon}{500}}$ be some constant to be specified later.

First assume $\mathbf{E}[X] \geq T\sqrt{n}$, in which case the proof follows simply from Azuma's inequality (see Theorem 1). We apply this inequality by setting $q_i = \sum_{j=1}^i (X_j - \mathbf{E}[X_j])$, noting that $\mathbf{E}[X_j - \mathbf{E}[X_j]] = 0$ so $0 = q_0, q_1, .., q_n$ is a martingale (i.e. $E[q_i|q_{i-1}] = q_{i-1}$), and also $|q_i - q_{i-1}| = |X_i - \mathbf{E}[X_i]| \leq 1$. Since $X - \mathbf{E}[X] = q_n$ and $\mathcal{E} \ll \sqrt{n}$, we deduce

$$\Pr[|X| > 2\mathcal{E} + 1] \geq \Pr[|X| > \frac{T}{2}\sqrt{n}] \geq$$
$$\Pr[|X - \mathbf{E}[X]| < \frac{T}{2}\sqrt{n}] \geq 1 - 2e^{-T^2/8} = \Omega(1)$$

The second and more interesting case is when $\mathbf{E}[X] < T\sqrt{n}$. We will prove that $\Pr[|X - \mathbf{E}[X]| > 2T\sqrt{n}] > \Omega(1)$ and obtain the desired result since $\Pr[|X| > 2\mathcal{E} + 1] \geq \Pr[|X - \mathbf{E}[X]| > 2T\sqrt{n}]$ (recall $\mathcal{E} \ll \sqrt{n}$).

This inequality can fail to hold only if X is very concentrated around $\mathbf{E}[X]$, so let us examine X's variance. We set $Y = (X - \mathbf{E}[X])^2$ and observe that since the X_i are independent, for some $\alpha \geq \frac{\varepsilon}{36}$

$$\mathbf{E}[Y] = VarX = \sum_{i=1}^n Var(X_i) = \sum_{i=1}^n \frac{(x_i - d_i)^2}{4} = \alpha \cdot n\,.$$

Now partitioning into three regions, with β to be chosen later, $A = \left\{Y < \frac{1}{3} \cdot \alpha n\right\}$, $B = \left\{\frac{1}{3} \cdot \alpha n \leq Y \leq \beta n\right\}$ and $C = \{Y > \beta n\}$ we can write:

$$\begin{aligned} \alpha n &= \mathbf{E}[Y] \\ &= \Pr[A] \cdot \mathbf{E}[Y|A] + \Pr[B] \cdot \mathbf{E}[Y|B] + \Pr[C] \cdot \mathbf{E}[Y|C] \\ &\leq 1 \cdot \frac{1}{3} \cdot \alpha n + \Pr[B] \cdot \beta n + \Pr[C] \cdot \mathbf{E}[Y|C] \end{aligned}$$

To complete our proof, we have the following claim,

Claim 6. *There exists a constant $\beta = \beta(\alpha)$, s.t.*

$$\Pr[C] \cdot \mathbf{E}[Y|C] = \Pr[Y > \beta n] \cdot \mathbf{E}[Y|Y > \beta n] < \frac{1}{3} \cdot \alpha n$$

Proof. Whenever X_i are independent random variables, and $|X_i - \mathbf{E}[X_i]| \leq 1$ and setting $q_i = X_1 + ... + X_i$, then $q_i - \mathbf{E}[q_i]$ is a martingale, and by Azuma's inequality (see Theorem 1),

$$Pr[q_n > \mathbf{E}[q_n] + t\sqrt{n}] < 2e^{-t^2/2}$$

Now,

$$\begin{aligned} C = \{Y > \beta n\} &= \{|X - \mathbf{E}[X]| > \sqrt{\beta n}\} \\ &= \{|q_n - \mathbf{E}[q_n]| > \sqrt{\beta n}\} \end{aligned}$$

so $\Pr[Y > \beta n] < 2e^{-\beta/2}$ and setting $I_k = [k \cdot \beta n\, ,\, (k+1) \cdot \beta n]$ for $k \geq 0$:

$$\begin{aligned} \Pr[C] \cdot \mathbf{E}[Y|C] &= \sum_{t > \beta n} t \cdot \Pr[Y = t] \\ &\leq \sum_{k=1}^{n^2} \Pr[Y \in I_k] \cdot (k+1) \cdot \beta n \\ &\leq \sum_{k=1}^{n^2} \Pr[Y > k \cdot \beta n] \cdot (k+1) \cdot \beta n \\ &\leq n \cdot \sum_{k=1}^{\infty} (k+1) \beta e^{-k \cdot \beta/2} \end{aligned}$$

Since $\sum_{k=1}^{\infty} \beta \cdot e^{-k \cdot \beta/2}(k+1)$ converges to a function of β that is decreasing as β increases, we can choose β large enough so that $\sum_{k=1}^{\infty} \beta \cdot e^{-k \cdot \beta/2}(k+1) < \frac{\alpha}{3}$. □

Going back to (1), and choosing $T = \sqrt{\frac{\alpha}{12}} \geq \sqrt{\frac{\varepsilon}{500}}$, we can now deduce that

$$\Pr[|X - \mathbf{E}[X]| > 2T \cdot \sqrt{n}] = \Pr[Y > \frac{\alpha n}{3}] \geq$$
$$\Pr[B] \geq \frac{\alpha n - \frac{1}{3}\alpha n - \frac{1}{3}\alpha n}{\beta n} = \frac{\alpha}{3\beta} = \Omega(1)$$

□

B Self Auditing

Auditing is a query restriction method in which a log of the queries is saved, and every query is checked for possible compromise allowing or disallowing the query accordingly [7]. One problem with this approach is of efficiency – the problem of deciding whether a sequence of queries violates privacy was shown to be computationally hard ([14] show that it is NP-hard to decide, given a database d and a set of queries, whether an exact answer to these queries lead to fully determining the value of at least one database entry).

Another, perhaps more acute problem, is that the auditor's approvals/refusals leak information about the database. It is conceivable that this information helps an adversary to violate the database privacy. We give a simple example to illustrate this problem. Consider a statistical database with n binary entries $d = d_1, \ldots, d_n$. The access to d is monitored by an auditor, with the notion of privacy as in [14] – i.e. privacy is violated when one or more of the database entries is fully determined by the queries.

Our attacker queries the sets $\{i, i+1\}$ for $1 \leq i < n$, and records for each query whether it was allowed or not. Note that the auditor refuses to answer queries exactly when both entries are zero or both are one (since in these cases the answer reveals the exact value of both entries). Hence, if a query $\{i, i+1\}$ is approved, the attacker learns that $d_i \neq d_{i+1}$ otherwise, she learns that $d_i = d_{i+1}$. As a result the attacker remains with only two (complementary) candidates for the database, namely $d_1, \ldots, d_n$ and $1 - d_1, \ldots, 1 - d_n$. The attacker can learn the entire database if it manages to distinguish between these two candidates. Only a little more information is needed, such as whether there are more zeros than ones in the database.

The reason the above auditing paradigm fails to provide privacy is that the auditor function takes into account not only the data that is already known to the user, but also the *answer to the query in question.* To avoid leakage from the auditor we suggest to construct it such that its decision whether to disqualify a query is based solely on information that is already known to the potential attacker. It follows that, in principle, this function may be computed by every user, hence the term *self auditing.*

Our analysis in the proof of Theorem 5 suggests a possible self auditing function, assuming that the database distribution $\mathcal{DB}$ is known. The self-auditor algorithm is based on the analysis of the confidence a potential attacker may have in the database sensitive entries. The auditor maintains the confidence the attacker has in each of these entries. The auditor then checks whether a query should be allowed by estimating the random variable that measures how an answer to the query may contribute to the confidence. The distribution of this random variable (or its estimate) may be computed, in an analogous manner to the computation in Theorem 5, from the history of query-answer pairs, the current query and the properties of the distribution. A query should be disallowed if it may, with too high probability, increase the confidence in any of the database entries so as to violate its privacy.