RECONSTRUCTION ATTACKS ON AGGRESSIVE RELAXATIONS OF DIFFERENTIAL PRIVACY

PROTTAY PROTIVASH, JOHN DURRELL, DANIEL KIFER, ZEYU DING, AND DANFENG ZHANG

The Pennsylvania State University, State College, PA
e-mail address: pxp945@psu.edu

The Pennsylvania State University, State College, PA
e-mail address: jmd6968@psu.edu

The Pennsylvania State University, State College, PA
e-mail address: dkifer@cse.psu.edu

Binghamton University, Binghamton, NY
e-mail address: dding1@binghamton.edu

The Pennsylvania State University, State College, PA
e-mail address: zhang@cse.psu.edu

Abstract. Differential privacy is a widely accepted formal privacy definition that allows aggregate information about a dataset to be released while controlling privacy leakage for individuals whose records appear in the data. Due to the unavoidable tension between privacy and utility, there have been many works trying to relax the requirements of differential privacy to achieve greater utility.

One class of relaxation, which is gaining support outside the privacy community is embodied by the definitions of *individual differential privacy* (IDP) and *bootstrap differential privacy* (BDP). Classical differential privacy defines a set of neighboring database pairs and achieves its privacy guarantees by requiring that each pair of neighbors should be nearly indistinguishable to an attacker. The privacy definitions we study, however, aggressively reduce the set of neighboring pairs that are protected.

To a non-expert, IDP and BDP can seem very appealing as they echo the same types of privacy explanations that are associated with differential privacy, and also experimentally achieve dramatically better utility. However, we show that they allow a significant portion of the dataset to be reconstructed using algorithms that have arbitrarily low privacy loss under their privacy accounting rules.

With the non-expert in mind, we demonstrate these attacks using the preferred mechanisms of these privacy definitions. In particular, we design a set of queries that, when protected by these mechanisms with high noise settings (i.e., with claims of very low privacy loss), yield more precise information about the dataset than if they were not protected at all. The specific attacks here can be defeated and we give examples of countermeasures. However, the defenses are either equivalent to using differential privacy or to ad-hoc methods tailored specifically to the attack (with no guarantee that they protect against other attacks). Thus, the defenses emphasize the deficiencies of these privacy definitions.

Key words and phrases: relaxations of differential privacy; reconstruction attacks.

JOURNAL OF PRIVACY AND CONFIDENTIALITY
www.journalprivacyconfidentiality.org
DOI:10.29012/jpc.v0.i0.000

1. Introduction

Statistical agencies face the challenge of releasing data products that are detailed and statistically useful while simultaneously meeting the legal and ethical obligations to protect the confidentiality of individuals providing the data. Similarly, companies seek to gain a competitive advantage by mining detailed information about their user base while still providing confidentiality guarantees to those users.

In some areas, differential privacy [Dwork et al., 2006b,a, Bun and Steinke, 2016, Mironov, 2017, Dong et al., 2022] is gaining acceptance as a source of viable solutions to these problems [Erlingsson et al., 2014, Bittau et al., 2017, Apple Differential Privacy Team, 2017, Ding et al., 2017, Johnson et al., 2018, Machanavajjhala et al., 2008, U. S. Census Bureau, Haney et al., 2017, Abowd, 2018]. However, the use of differential privacy to protect Census data has also drawn fierce criticism, most recently with a group of prominent economists and statisticians calling for the Census Bureau to stop using it [Hotz et al., 2022]. Such reactions are often due to frustration with the tension between utility and privacy. For example, differential privacy has many known mathematical lower bounds that clearly delineate the accuracy with which information can be released at a given level of privacy (see, for example, [Balcer and Vadhan, 2019, Vadhan, 2017, Hardt and Talwar, 2010, McGregor et al., 2010, Dinur and Nissim, 2003, Kasiviswanathan et al., 2008, Abowd et al., 2021, Steinke and Ullman, 2017]).

One the one hand, similar restrictions (hence similar criticisms) would apply to *any* method that protects confidentiality – producing "privacy-protected" data products that allow arbitrary analyses to be conducted accurately will result in reconstruction of nearly all of the underlying confidential data [Dinur and Nissim, 2003], and hence would provide no confidentiality. However, the trade-off between privacy and utility in practical applications is still very much an open question, and this has led to many relaxations of differential privacy (see the exhaustive survey by Desfontaines and Pejó [2020]).

In this paper, we study (and develop attacks for) a type of privacy definition that re-examines the concept of *neighboring databases* that is fundamental to differential privacy. Informally, differential privacy seeks to ensure that a data release mechanism M behaves "similarly" on databases D_1 and D_2 when they are "neighbors" of each other. Intuitively, this means that M masks the differences between D_1 and D_2. Thus, if neighboring datasets are defined to be all pairs of datasets that differ on the value of a record, this definition provides plausible deniability: an attacker would not be able to determine the contents of any target individual's record since the behavior of M would be almost unrelated to the actual contents of the record.

Relaxations of differential privacy that target the definition of neighbors seek to change what a mechanism M attempts to hide. The particular class of relaxations [Soria-Comas et al., 2017, O'Keefe and Charest, 2019] we are interested in, which we call *empirical neighbors*, argue that if D_1 and D_2 are unrelated to the actual dataset D_{act} that will be the input to M, why should M be designed to hide the differences between D_1 and D_2 [Soria-Comas et al., 2017, O'Keefe and Charest, 2019]? Instead, such proposed relaxations try only to hide the differences between the actual dataset and some suitable alternatives. For example, *Individual Differential Privacy* (IDP) [Soria-Comas et al., 2017] (*not* to be confused with personalized differential privacy [Jorgensen et al., 2015, Ebadi et al., 2015]) considers two databases D_1, D_2 to be neighbors if one of them is the actual dataset D_{act} owned by a statistical agency and the other can be obtained from D_{act} by modifying a single

record. Their argument is that this is precisely what statistical agencies need because it provides plausible deniability of any record in D_{act} (and hence any additional protections provided by differential privacy are unnecessary). This rationale sounds convincing to many outside the privacy community. For example Hotz et al. [2022] called for a moratorium on the use of differential privacy at the Census Bureau and mentioned that the type of mechanisms supported by IDP "may be sensible" as an alternative to differential privacy, although they worried that the relaxations might still not provide enough utility [Hotz et al., 2022, Appendix C1].

The purpose of this paper is to provide an illustration, especially to a non-expert, of why privacy definitions need to consider the behavior of a mechanism M not just on the dataset D_{act} at hand, but also on other datasets that may look quite different from D_{act}. Specifically, we show that by eliminating consideration of such datasets, IDP and BDP allow dataset reconstruction at arbitrarily low privacy parameter settings – that is, the privacy accounting frameworks for those definitions would claim that almost no information is leaked. We also provide examples of other, more targeted attacks such as verifying that a particular combination of attributes appears in the data as part of a record, verifying that the record is a sample unique, and perfectly reconstructing other variables associated with the record.

The weaknesses we exploit are known problems in the differential privacy community. For example, IDP mechanisms use a concept known as *local sensitivity* [Nissim et al., 2007] to determine how much noise to add to query answers. Nissim et al. [2007] previously argued against such uses of local sensitivity. Specifically, Nissim et al. [2007] provided an example in which the median can sometimes be released with 0 noise, thus leaking some information.

Our work goes beyond this observation. Not only do we go further and show that it opens up the possibility for full database reconstruction and targeted attacks against individuals, we also show that reconstruction can happen even when using 0 noise is prohibited.

The main part of the paper is an attack against IDP that uses its recommended noise infusion strategies [Soria-Comas et al., 2017]. Although IDP keeps track of the privacy loss budget expended on the queries, at the end of the attack, it claims that almost no privacy budget was expended even though the entire dataset can be reconstructed. We launch the attack using queries that have a special property – protecting those queries with IDP reveals more precise information about the data than if no protections were used at all for those queries. This is true even if the IDP mechanisms always add noise (i.e., even when the situations with 0 noise are avoided).

We discuss several simple ways that this attack can be foiled, but point out that these defenses only underscore the weakness of IDP's privacy accounting – either the defense amounts to using differential privacy, or the defense specifically targets features of the attack (thus using the defense implicitly acknowledges the limitations of IDP's privacy accounting without providing provable guarantees against different attacks).

After that, we briefly discuss how a similar type of result can be applied to a related privacy definition called Bootstrap Differential Privacy [O'Keefe and Charest, 2019]. However, in this case, rather than reconstructing the entire dataset, one can only reconstruct the distinct set of records – that is, one can determine which records are present but not how many times they appear.

We then analyze these styles of privacy definitions more abstractly to determine why they have different leakage properties. Overall, we conclude that this direction is unlikely to provide the right balance between privacy and utility in practice.

To summarize, our contributions are the following:

- We present a practical reconstruction attack against Individual Differential Privacy [Soria-Comas et al., 2017] (IDP) and its more challenging version named $(\epsilon_1, \ldots, \epsilon_k)$-Group Differential Privacy [Soria-Comas et al., 2017] (Group IDP). The privacy loss parameter for these definitions is ϵ and we show that for any $\epsilon > 0$, it is possible to reconstruct any dataset whose size[1] is larger than 2 (or $2k$ in the case of $(\epsilon_1, \ldots, \epsilon_k)$-group differential privacy). In particular, we construct queries such that answering the queries with the noise mechanism constructions proposed by Soria-Comas et al. [2017] provides *more* information about the data than if the queries were always answered truthfully.
- We show that the reconstruction attack can be specialized to also perform membership inference and attribute inference attacks with significantly fewer queries.
- We then briefly consider Bootstrap Differential Privacy [O'Keefe and Charest, 2019] and show that its preferred mechanism can also be used to leak the distinct set of records in the data, again for any privacy loss $\epsilon > 0$. The fact that this information can be leaked was noted by the authors [O'Keefe and Charest, 2019], but we show that it can even be leaked using the preferred mechanisms of BDP.
- In order to better understand these weaknesses, we consider various ad-hoc defenses against reconstruction and show that they do not solve the fundamental problems.
- We also study this style of privacy definition more abstractly (we call it *empirical neighbors*) and show that this privacy leakage is unavoidably built-in to the privacy definition.

The rest of this paper is structured as follows. We describe notation and present background definitions in Section 2. We present a reconstruction attack against individual differential privacy and its group-based version in Section 3, where we also explain how membership and attribute inference attacks against specific individuals can be performed. This section forms the bulk of the paper. We then review bootstrap differential privacy in Section 4 and briefly show how similar techniques can be used to launch attacks against it. Then we analyze the weaknesses of these types of definitions more generically in Section 5. We experimentally evaluate the reconstruction algorithm for IDP in Section 6 and discuss related work in Section 7. Conclusions and future work are in Section 8. All proofs can be found in the appendices.

Our code can be found at `https://github.com/cmla-psu/idpreconstruction`.

2. Background And Notation

A dataset D is a collection $r_1, \ldots, r_n$ of records, each corresponding to a distinct individual. For simplicity, we assume that the total number of records n is known. Each record has attributes $\mathbf{A}_1, \ldots, \mathbf{A}_m$ (e.g., $\mathbf{A}_1 =$"income", $\mathbf{A}_2 =$"is student?"). The value of attribute $\mathbf{A}_i$ for record r_j is denoted as $r_j[i]$. The specific dataset that has been collected by a statistical agency is denoted as D_{act}.

We say that two datasets D_1 and D_2 are *differential privacy neighbors* (or *dp-neighbors* for short) if one can be obtained from the other by modifying the record of an individual. We use the notation $D_1 \sim D_2$ to indicate that D_1 and D_2 are dp-neighbors.

[1]Note this is not $2k$ per possible record value or $2k$ per dimension of the dataset; all that is required is the dataset have at least $2k$ people total, and typically the parameter k is recommended to be 1 [Soria-Comas et al., 2017]. Large values of k such as 100 can severely damage the utility of the data.

A mechanism M is a (randomized) algorithm whose input is a confidential dataset and whose goal is to produce an output that protects the confidentiality of individuals whose records are in the input dataset.

2.1. **Differential Privacy.** Differential privacy is a set of restrictions on the behavior of *randomized* algorithms. Intuitively, it masks the effect of any record on the output of M by ensuring that the output distribution of M is relatively insensitive to changes to a record in the input, hence providing plausible deniability for the contents of a record.

Definition 2.1 (ϵ-differential privacy [Dwork et al., 2006b])**.** A randomized algorithm M satisfies ϵ-differential privacy (ϵ-DP) if for every set S $\subseteq$ Range(M) and for all pairs of dp-neighbors D_1 and D_2,

$$Pr[M(D_1) \in S] \leq e^{\epsilon} Pr[M(D_2) \in S]$$

where the probability only depends on the randomness in M.

Both IDP and BDP are variations of differential privacy, but we defer their definitions to Sections 3 and 4, respectively, to make them relatively self-contained, so that the definition, motivation, preferred privacy mechanisms, and attacks are all in one place.

2.2. **Sensitivity.** In the differential privacy literature, different notions of *sensitivity* are used to quantify the effect that a single record could have on the output of a function f and is often used to calibrate the amount of noise that a mechanism M might add to the output of f.

The first of these is *global sensitivity*, defined as follows:

Definition 2.2 (Global sensitivity [Dwork et al., 2006b])**.** The global sensitivity of a (vector-valued) function f, denoted as $\Lambda(f)$, is the largest change in f that can be achieved by modifying a record in any dataset:

$$\Lambda(f) = \sup_{D_1 \sim D_2} ||f(D_1) - f(D_2)||_1$$

where the suprememum is taken over *all* pairs D_1, D_2 that are dp-neighbors of each other.

Global sensitivity may overestimate the amount of noise that must be added to hide the effect of a record. For this reason, Nissim et al. [2007] introduced an intermediate concept called *local sensitivity*.

Definition 2.3 (Local sensitivity [Nissim et al., 2007])**.** The local sensitivity of a (vector-valued) function f with respect to a dataset D (denoted as $\Lambda^s(f, D)$ is defined as the largest change in f that can be achieved by modifying a record in D:

$$\Lambda^s(f, D) = \sup_{D' \sim D} ||f(D) - f(D')||_1$$

where the supremum is over all datasets D' that are dp-neighbors of D. Note that the global sensitivity is related to local sensitivity as follows: $\Lambda(f) = \sup_D \Lambda^s(f, D)$.

Nissim et al. [2007] noted that local sensitivity is not compatible with ϵ-differential privacy. But an upper bound of it, called *smooth sensitivity* is compatible with ϵ-differential privacy [Nissim et al., 2007]. Local sensitivity, however *is* compatible with IDP (see Section 3.1). The following generalization of local sensitivity is also needed for discussing IDP:

Definition 2.4 (k-Local Sensitivity)**.** The k-local sensitivity of a function f with respect to a dataset D (denoted by ${\Lambda^s}_k(f, D)$) is defined as the largest change in f that can be achieved by modifying up to k records in D. Let $\mathcal{N}_k(D)$ be the set of all datasets that can be obtained from D by modifying up to k records. The formula for k-local sensitivity is:

$$ {\Lambda^s}_k(f, D) = \max_{D' \in \mathcal{N}_k(D)} ||f(D) - f(D')|| $$

Note when $k = 1$, this is the same as local sensitivity.

3. Reconstruction against Individual Differential Privacy

In this section, we present reconstruction and more targeted attacks against IDP and its generalization Group IDP that is intended to provide more privacy protections [Soria-Comas et al., 2017]. Intuitively, IDP seeks only to protect the current dataset and any dataset obtainable by changing 1 record, while Group IDP tries to extend protections to datasets that differ from the current dataset by up to k records.

We first review these privacy definitions and recommended privacy mechanisms (Section 3.1). We examine the main query used for the attack in Section 3.2 that tricks the privacy mechanism into revealing private information. Using this query, we then show how to reconstruct a single column (attribute) of a table in Section 3.3. We explain how to extend these ideas to reconstruct the entire table (Section 3.4) at arbitrarily low privacy loss budget settings. Then, we explain how the attack can be specialized to membership inference and attribute inference, using many fewer queries, in Section 3.5. Finally we discuss some countermeasures and their implications for IDP in Section 3.6.

3.1. A Review of IDP and Group IDP.

The fundamental idea behind IDP and Group IDP is that the plausible deniability argument provided by differential privacy only needs to be applied to the actual dataset D_{act} collected by a data curator and does not need to apply to every possible dataset [Soria-Comas et al., 2017]. Thus IDP only seeks to mask the differences between D_{act} and any dataset that can be obtained from it by modifying a record. Meanwhile, Group IDP seeks to mask the difference between D_{act} and any dataset that differs from it by up to k records, for some prespecified k. Since Group IDP has $k + 1$ parameters named $k, \epsilon_1, \epsilon_2, \ldots, \epsilon_k$, we present a two-parameter simplification of it. Any mechanism that satisfies this simplification, also satisfies the more complex original definition, so any attack on the simplification also directly works on the original definition. Formally,

Definition 3.1 (ϵ-IDP and (ϵ, k)-Group IDP [Soria-Comas et al., 2017])**.** Given a fixed data set D_{act}, privacy loss budget $\epsilon \geq 0$, and group size $k \geq 1$, let $\mathcal{N}_k$ be the set of all datasets that can be obtained from D_{act} by modifying up to k records. A mechanism M satisfies (ϵ, k)-Group IDP with respect to D_{act} if for every $D \in \mathcal{N}_k$ and every $S \subseteq \text{range}(M)$,

$$ Pr[M(D) \in S] \leq e^{\epsilon} Pr[M(D_{\text{act}}) \in S] $$
$$ Pr[M(D_{\text{act}}) \in S] \leq e^{\epsilon} Pr[M(D) \in S] $$

When $k = 1$, we say that M satisfies ϵ-IDP with respect to D_{act}; that is, ϵ-IDP is the same as $(\epsilon, 1)$-Group IDP.

The parameter k is the group size parameter and is particularly important to reconstruction, because our attack only works on datasets of size $\geq 2k$. This is not a particularly strong restriction because a low value of k is recommended (e.g, $k = 1$) [Soria-Comas et al., 2017].

The parameter $\epsilon \geq 0$ is the privacy loss parameter. Large values of ϵ correspond to weaker privacy protections and small values of ϵ (close to 0) ostensibly correspond to stronger privacy protections.

We note that the original, more complex, definition has k privacy loss parameters $\epsilon_1, \ldots, \epsilon_k$, but a mechanism M satisfying Definition 3.1 with $\epsilon = \min_i \epsilon_i$ also satisfies that more complex definition and any reconstruction attack against Definition 3.1 is therefore also a reconstruction attack against the original definition. This privacy definition has desirable properties that are required of formal privacy definitions:

- **Postprocessing invariance:** Let M be a mechanism that satisfies (ϵ, k)-Group IDP with respect to D_{act} and let $\mathcal{A}$ be a postprocessing algorithm whose domain contains the range of M. Then the algorithm that first runs M and then runs $\mathcal{A}$ on the result satisfies (ϵ, k)-Group IDP with respect to D_{act} for the exact same privacy parameters [Soria-Comas et al., 2017].
- **Composition:** Let M_1 be a mechanism that satisfies (ϵ_1, k)-Group IDP with respect to D_{act} and let M_2 be a mechanism that satisfies (ϵ_2, k)-Group IDP with respect to D_{act}. The mechanism that releases the outputs of both M_1 and M_2 satisfies $(\epsilon_1 + \epsilon_2, k)$-Group IDP with respect to D_{act}. [Soria-Comas et al., 2017].

Mechanisms for Group IDP are based on local and k-local sensitivity (Definition 2.4). Specifically, the scale of the noise added to a query is proportional to the k-local sensitivity. Nissim et al. [2007] earlier had argued that basing the amount of noise on local sensitivity is problematic because "the noise magnitude itself reveals information about the database." They illustrated this with an example with the median function, which can have local sensitivity of 0 for some (but not all) datasets, which would result in 0 noise being added for those datasets. Their warning has often been interpreted as a caution against releasing the value of the local sensitivity [Hotz et al., 2022, Chetty and Friedman, 2019].

However, we demonstrate a more severe vulnerability. First, this is not a problem that affects just *some* datasets – it affects *all* datasets. Second, even if the noise scale is never 0 (for example, if the noise scale is proportional to k-local sensitivity +1) and even if the local sensitivity is never revealed directly, one can still infer enough information about the dataset to reconstruct it, as long as the dataset size is $\geq 2k$.

One generic mechanism for Group IDP is the *k-Laplace* mechanism, defined as follows.

Definition 3.2 (k-Laplace Mechanism [Soria-Comas et al., 2017])**.** Let g be a vector-valued function with k-local sensitivity $\Lambda^s{}_k(g, D_{\text{act}})$ with respect to the true data D_{act}. Let $\epsilon^* \in (0, \epsilon]$ be the amount of the privacy loss budget allocated to the mechanism. The k-Laplace mechanism outputs $g(D_{\text{act}}) + \text{Laplace}(\Lambda^s{}_k(g, D_{\text{act}})/\epsilon^*)$, where $\text{Laplace}(\Lambda^s{}_k(g, D_{\text{act}})/\epsilon^*)$ is a vector of independent Laplace random variables, each having density function

$$f(x) = \frac{\epsilon^*}{2\Lambda^s{}_k(g, D_{\text{act}})} \exp\left(-\frac{\epsilon^*}{2\Lambda^s{}_k(g, D_{\text{act}})}|x|\right)$$

and variance $\frac{2\Lambda^s{}_k(g, D_{\text{act}})^2}{(\epsilon^*)^2}$.

The k-Laplace mechanism satisfies (ϵ^*, k)-Group IDP [Soria-Comas et al., 2017] and our reconstruction attack will take advantage of the k-Laplace mechanism when applied to the g function corresponding to the query described in Section 3.2.

3.2. **The Attack Query.** We now identify a class of queries such that answering these queries with the k-Laplace mechanism and tiny values of ϵ^* (corresponding to very strong claims of privacy) ends up revealing more information about the data than if the queries were always answered truthfully (i.e., without any protections).

The queries we are interested in are *predicate count queries with thresholds*. That is, given a predicate ϕ (a function whose input is a record and whose output is True/False) and a threshold b, the query $q_{\phi,b}$ returns 1 if the number of records satisfying the predicate is larger than b. Formally,

$$q_{\phi,b}(D) = \begin{cases} 1 & \text{if } \left|\{r \in D : \phi(r) = \text{True}\}\right| > b \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

The k local sensitivity of $q_{\phi,b}$ is the following.

Lemma 3.3. *Let k be a positive integer (e.g., the group size parameter in Group IDP) and suppose the true dataset D_{act} has $\geq k$ records. The k-local sensitivity of $q_{\phi,b}$ with respect to D_{act} is 0 whenever $b < 0$, $b \geq n$ (number of records in D_{act}), or ϕ is always true or always false. Otherwise:*

$$\Lambda^s{}_k(q_{\phi,b}, D_{act}) = \begin{cases} 0 & \textit{when } \left|\{r \in D_{act} : \phi(r) = True\}\right| > b + k \\ 0 & \textit{when } \left|\{r \in D_{act} : \phi(r) = True\}\right| \leq b - k \\ 1 & \textit{otherwise} \end{cases}$$

□

We are particularly interested in the queries where the predicate ϕ specifies a range $[u, v)$ on an attribute $\mathbf{A}_i$. That is $\phi(r) =$True if and only if $u \leq r[i] < v$. When ϕ is such a predicate, we denote the corresponding query as $q_{\mathbf{A}_i \in [u,v),b}$. It returns 1 when the count of records having attribute $\mathbf{A}_i$ in the range $[u, v)$ is larger than b. We call this a *threshold range-count query*.

Using the k-Laplace Mechanism with a portion ϵ^* of the privacy budget to protect $q_{\mathbf{A}_i \in [u,v),b}$ results in what we shall call the Group IDP *threshold range query mechanism* for $q_{\mathbf{A}_i \in [u,v),b}$:

$$M(D) = q_{\mathbf{A}_i \in [u,v),b}(D) + \text{Laplace}\left(\frac{\Lambda^s(q_{\mathbf{A}_i \in [u,v),b}, D)}{\epsilon^*}\right) \tag{3.2}$$

Note that for some combinations of b and D, the k-local sensitivity is 0 and no noise is added. For other values of b and D, the local sensitivity is 1 and Laplace$(1/\epsilon^*)$ noise is added. Being able to distinguish between the two cases using only the output of M is the key to the attack. We explain how to do this next, but we also note that having 0 noise is not necessary for the attack to work – for example if the noise is either Laplace(a/ϵ^*) or Laplace(b/ϵ^*) for some positive numbers a and b, reconstruction is still possible (we explain how to deal with this complication in Section 5).

3.2.1. *Detecting Noiseless Answers.* When the share of the privacy budget ϵ^* is extremely small, it is possible to detect with near perfect accuracy whether M returned a value that has no noise (k-local sensitivity is 0) or is noisy (k-local sensitivity is 1). For example, suppose the share of the privacy loss budget used in the mechanism is $\epsilon^* = 10^{-10}$. When the k-local sensitivity is 1, the Laplace noise will be a non-integer – the probability that a floating point implementation of Laplace noise with scale $1/\epsilon^*$ is a non-integer is essentially 1. If no noise is added, then the output would certainly be 0 or 1. Thus the following decision rule has near perfect accuracy: if the output is not 0 or 1, it was because noise was added and so the local sensitivity is 1; if the output is 0 or 1, then with overwhelming probability no noise was added and local sensitivity is 0.

Moreover, even if one performs ad-hoc protections like rounding the output of the mechanism, it is still possible to tell whether the k-local sensitivity was 0 or 1 as follows:

- If the output is rounded to the nearest integer, then if noise is injected, the probability that the output is 0 or 1 is $\leq 1 - e^{-2\epsilon^*}$ (this is the probability that the absolute value of the noise is not greater than 2). When $\epsilon^* = 10^{-10}$, this probability is at most 2×10^{-10}. This means that the decision rule described above will fail with probability less than 2×10^{-10}.
- If the output of the mechanism is rounded to 0 or 1 (whichever is closer to the noisy value that was produced by the mechanism), one can still distinguish between the k-local sensitivity $= 0$ and k-local sensitivity $= 1$ cases. Simply ask the same query 15 times, each time with privacy loss budget share $\epsilon^* = 10^{-10}/15$. The decision rule to use is: if all the 15 answers are identical then assume no noise was added and if at least 1 answer is different from the rest, then assume noise was added. Clearly, if the k-local sensitivity is 0 then all the 15 answers are noise-free, and the rule would be correct. On the other hand, if the k-local sensitivity is 1, then the probability of getting 15 ones or 15 zeroes as the answers is approximately $2 * 2^{-15} \leq 10^{-10}$ and so the probability of the decision rule failing is virtually 0. Meanwhile, the total privacy budget spent by the 15 queries is $15 * 10^{-10}/15 = 10^{-10}$.

3.2.2. *What one learns from noisy and noiseless answers.* It turns out that the ability to detect whether an answer is noisy or not allows us to infer deterministic information about the data even if the answer was highly noisy. More surprisingly, answering $q_{\mathbf{A}_i \in [u,v),b}$ using the k-Laplace mechanism provides *more* information than one would get if no protection was used as all (i.e., if it was always answered truthfully no matter what). This finding is a consequence of the following lemma.

Lemma 3.4. *Let D_{act} be a dataset with n records (where n is publicly known). Let $\mathbf{A}_i$ be an ordered attribute and $[u, v)$ be a range that does not contain the entire domain of $\mathbf{A}_i$. Let b be an integer threshold such that $1 \leq b \leq n-1$. Let M be the k-Laplace mechanism for answering the threshold range query $q_{\mathbf{A}_i \in [u,v),b}$. If the output ω of $M(D_{act})$ is released, then the following can be learned about D_{act}:*

- *If ω is detected as a noisy output then the quantity $\left|\{r \in D_{act} : u \leq r[i] < v\}\right|$ is $\geq b-k+1$ and is also $\leq b+k$. In other words, we get an upper and lower bound on the number of people in D_{act} whose value for $\mathbf{A}_i$ is in the range $[u, v)$.*

- *If ω is detected as non-noisy and $\omega = 1$ then* $\left|\{r \in D_{act} : u \leq r[i] < v\}\right| > b + k.$
- *If ω is detected as non-noisy and $\omega = 0$ then* $\left|\{r \in D_{act} : u \leq r[i] < v\}\right| \leq b - k$

□

Since our decision rule has near-perfect accuracy and uses up at most ϵ^* of the privacy loss budget (the attack would be using $\epsilon^* \leq 10^{-10}$) then we essentially know if the answer was noisy or not, and so: (1) if the answer is noisy, we learn that something about the count of people whose attribute A_i is in the range $[u, v)$. Specifically, we learn that this count is actually somewhere between $b - k + 1$ and $b + k$ (an interval of size $2k - 1$). **Note that answering $q_{\mathbf{A}_i \in [u,v),b}$ with no protection would never result in us learning that the true answer is inside such an interval**; (2) if the answer is not noisy (i.e., suppose the answer is 1), then this non-noisy query answer directly tells us that the count of people in the range $[u, v)$ is more than b. But furthermore, since we have figured out that the k-local sensitivity is 0, we can combine this information with Lemma 3.3 to learn that the count is not just $> b$, but it is in fact $> b + k$. Again, this is more information than if the query had always been answered truthfully. The reason we get so much extra information from this k-local Laplace Mechanism compared to a mechanism that is always truthful, is the extra leakage caused by inferring what the local sensitivity is.

3.3. **Single-Attribute Reconstruction.** We next show how to reconstruct one attribute $\mathbf{A}_i$ (one column) of the table when the data size is $\geq 2k$.[2] That is, for each possible value a_j, we will determine how many records $r \in D_{\text{act}}$ have $r[i] = a_j$. We consider the case where $\mathbf{A}_i$ is a numeric attribute since this is the hardest case. The categorical case can be handled in many ways; the simplest being to assign an arbitrary ordering on the domain of a categorical attribute.[3] The amount of privacy loss budget used in this reconstruction can be made arbitrarily small. We first illustrate the attack with an example.

Example 3.5. Let us consider $(\epsilon, 1)$-Group IDP (i.e., $k = 1$). Let us set the overall privacy budget at $\epsilon = 0.01$. We will require each call to the threshold range query mechanism to use $\epsilon^* = 10^{-10}$ of the privacy loss budget and so our goal is to make sure that the total budget used by all the mechanism calls is at most $\epsilon = 0.01$. Suppose the true dataset D_{act} has an income column $\mathbf{A}_1$, and contains 6 people whose incomes are $\{5, 8, 15, 16, 17, 18\}$. An attacker can proceed as follows.

(1) The attacker first tries to find out if, say, there are more than 3 people with incomes in the range $[1, 10)$. This means $u = 1, v = 10, b = 3$ (and recall $k = 1$). Since there are actually 2 people in that range and $2 \leq b - k$, then Lemma 3.3 says that the k-local sensitivity is 0. This means that the threshold-query mechanism, even when given only 10^{-10} of the privacy loss budget, will output the true answer 0. The attacker realizes that this is almost certainly not a noisy answer. Using Lemma 3.4, the attacker determines that the count of people with income in the range $[1, 10)$ is at most $b - k \equiv 2$.

(2) The attacker can then query if there is more than 2 people with incomes in the range $[1, 10)$. Based on the previous item, the attacker knows that there are not, but by posing

[2]We assume that the data size is public because the total number of records is a query that has a k-local sensitivity of 0.

[3]This is often done in practice. For example, gender is frequently coded as 0 for female, 1 for male, etc.

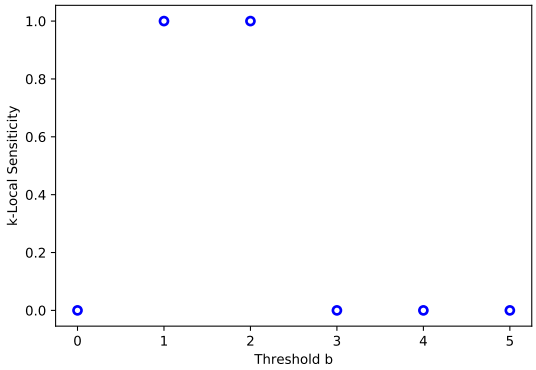

Figure 1: k-Local sensitivity of $q_{\mathbf{A}_i \in [u,v),b}$ for Example 3.5 as a function of the threshold b.

this query the attacker can extract more information out of the mechanism. So now the attacker chooses $u = 1, v = 10, b = 2$ for the query (and recall $k = 1$). Since there are 2 people in the range $[u, v)$ and $2 \not> b + k$ and $2 \not\leq b - k$, then Lemma 3.3 says that the k-local sensitivity is 1. Thus the mechanism (using 10^{-10} of the privacy loss budget) adds significant amounts of noise and produces an output like 9450462192.887615, which the attacker detects as a noisy answer. Using Lemma 3.4, the attacker determines that the number of people with income in the range $[1, 10)$ is at least $b + k - 1 \equiv 2$ and at most $b + k \equiv 3$.

(3) Putting the results of the previous two items together, the attacker concludes there are at exactly 2 people with incomes in the range $[1, 10)$, and only $10^{-10} + 10^{-10}$ privacy budget was spent on those two queries.

(4) The attacker can now perform the same kind of attack on the ranges $[1, 5)$, $[5, 10)$, and $[10, \infty)$ to determine the number of people in these ranges and could keep going, subdividing the ranges until a pre-specified precision such as 1 cent – i.e., an interval would look like $[\$9.83, \$9.84)$. Clearly, at this point the attacker would know exactly all of the incomes and as long as the attacker interacts with the mechanism less than 10^8 times, the total privacy loss will be less than the overall target of $\epsilon = 0.01$ given at the beginning of the example. Clearly, if the attacker spends even less than 10^{-10} privacy budget per query, the total privacy cost, according to Group IDP accounting, could be made arbitrarily small.

Thus the main subgoal for the attacker is to find out *exactly* how many people have values of attribute A_i in a range $[u, v)$. The attacker found a b value that is at the boundary of where the k-local sensitivity changes from 0 to 1 and used it to infer the true count. Indeed, as b varies, the k-local sensitivity looks like Figure 1 – for small b the k-local sensitivity is 0 and the mechanism produces 1 as the noise-free answer. At some point, the k-local sensitivity switches to 1, and then back to 0, after which the mechanism produces 0 as the noise-free answer.

The following Lemma shows that this is indeed the behavior, and when one identifies a b value that is on either of those two boundaries, the exact count is revealed.

Lemma 3.6. *Given a predicate ϕ, if for some integer $b^{\uparrow}$ we have (1) the k-local sensitivity of $q_{\phi,b^{\uparrow}}$ with respect to D_{act} is 0 and (2) the k-local sensitivity of $q_{\phi,(b^{\uparrow}-1)}$ is 1, then*

- *The count of people in D_{act} whose records satisfy ϕ is $b^{\uparrow} - k$.*
- *The k-Laplace mechanism $q_{\phi,b}$ will return the non-noisy answer 0 for all $b \geq b^{\uparrow}$*

Furthermore, if for some integer $b^{\downarrow}$ we have (1) the k-local sensitivity of $q_{\phi,b^{\downarrow}}$ with respect to D_{act} is 0 and (2) The k-local sensitivity of $q_{\phi,(b^{\downarrow}+1)}$ is 1, then

- *The count of people in D_{act} whose records satisfy ϕ is $b^{\downarrow} + k + 1$.*
- *The k-Laplace mechanism for $q_{\phi,b}$ will return the non-noisy answer 1 for all $b \leq b^{\downarrow}$*

□

When applied to $q_{\mathbf{A}_i \in [u,v),b}$, Lemma 3.3 tells us that the k-local sensitivity is 1 for those values of b that are between $\left|\{r \in D_{\text{act}} : u \leq r[i] < v\}\right| - k$ and $\left|\{r \in D_{\text{act}} : u \leq r[i] < v\}\right| + k - 1$. This range contains $2k$ integers, and so if the dataset size $|D_{\text{act}}|$ is $\geq 2k+1$, a boundary between k-local sensitivity of 0 and 1 will always exist for some b. Furthermore, if $|D_{\text{act}}| = 2k$, a boundary might not exist, but that can only happen if the count $\left|\{r \in D_{\text{act}} : u \leq r[i] < v\}\right|$ is k. Thus, as long as $|D_{\text{act}}| \geq 2k$ the attacker can determine the count $\left|\{r \in D_{\text{act}} : u \leq r[i] < v\}\right|$ with near perfect accuracy simply by increasing b from 0 to $|D_{\text{act}}| - 1$ until a boundary is found or b hits its upper limit. As long as the attacker splits his privacy budget across these (at most) $|D_{\text{act}}|$ queries, he can reconstruct the true count almost perfectly at arbitrarily low "privacy cost." The pseudocode is shown in Algorithm 1. For simplicity, it shows the linear search but we note that a binary search can be used instead.

Now that we have a tool for determining the counts of records within a range $[u, v)$, we can use it to reconstruct an entire attribute A_i up to a certain precision – that is, we can find all of the incomes in the dataset up to the nearest cent. Algorithm 2 shows how to do this. The algorithm starts by setting u to be the lower bound on the domain of Attribute $\mathbf{A}_i$ and v to be an upper bound.[4] For example for the Income attribute, one could set $u = 0$ and $v = 2^{40}$.

Next, the algorithm considers a decreasing sequence of values $v = v_0 > v_1 > v_2 > \ldots$. It calls Algorithm 1 to find out how many people have attribute $\mathbf{A}_i$ in the range $[u, v_j)$. If it finds that the count for $[u, v_j)$ (call this count s_0) and the count for $[u, v_{j-1})$ (call it s_1) are different, then there must be $s_0 - s_1$ people in the range $[v_{j-1}, v_j)$. If the width of the interval is ≤ 0.01, then it has reconstructed those income values up to a penny. In general, the target precision is a user-provided input called γ.

Note that Algorithm 2 uses linear search to find the next value after v_j for which the count changes, but this is shown for simplicity and can be replaced by a binary search instead for efficiency.

Algorithm 2 also upper bounds the number of calls it would need to Algorithm 1 and splits its target privacy budget ϵ equally among these calls. It ensures that the amount of privacy budget given to each Algorithm 1 call is small enough so that an answer to a k-local

[4]If bounds are not not known in advance, one could start with the interval $[-1, 1)$ and keep doubling the endpoints as long as Algorithm 1 reports that less than n records are in the interval.

Algorithm 1: Reconstruct count of records with attribute $\mathbf{A}_i$ in an interval $[u, v)$.

k, ϵ: Group IDP parameters
$n \geq 2k$: publicly known number of records in D_{act}
i: index of the target attribute
$M^{(i)}_{[u,v),b}$: mechanism that answers $q_{\mathbf{A}_i \in [u,v),b}$ using the k-Local Laplace mechanism as in Equation 3.2

def *CountReconstruct(k,* $\epsilon_{target}, u, v, i$*)***:**
  $b_0 \leftarrow 0$
  $a_0 \leftarrow$ result of $M^{(i)}_{[u,v),b_0}$ using privacy budget $\frac{\epsilon}{n}$
  `// Note linear search is shown for simplicity`
  `// Use binary search for more efficiency`
  for $j = 1, \ldots, n-1$ **do**
    $b_j \leftarrow j$
    $a_j \leftarrow$ result of $M^{(i)}_{[u,v),b_j}$ using privacy budget $\frac{\epsilon}{n}$
    if a_j *detected as noisy,* a_{j-1} *detected as non-noisy* **then**
      return $b_{j-1} + k + 1$
    else if a_j *detected as non-noisy,* a_{j-1} *detected as noisy* **then**
      return $b_j - k$
  `// After loop, either all answers were noisy`
  `// or all answers were non-noisy`
  `// But all non-noisy is impossible`
  return k

Laplace mechanism can be detected as noisy/non-noisy and so that Algorithm 2 can meet its budget goals.

3.4. **Reconstructing the Full Dataset.** Reconstructing the full dataset can be done in an iterative manner. One first reconstructs the first attribute $\mathbf{A}_1$ using Algorithm 2. This gives a set of records $r_1, \ldots, r_n$ that have just one attribute. One then needs to add attribute $\mathbf{A}_2$ to each record, then attribute $\mathbf{A}_3$, and so on. Since the algorithms used to do this are nearly identical to Algorithms 1 and 2, we do not list them here, but instead explain how the process would work with an example.

Example 3.7. Suppose Algorithm 2 has been used to reconstruct the Age column to get the values $[18, 18, 21, 21, 30]$. To add the next column, say height, we would be interested in queries of the form: "are there more than b many 18-year-olds who have *height* in $[u, v)$." This is another predicate count query with a threshold b and its k-local sensitivity is again given by Lemma 3.3. It is answerable using the k-local Laplace mechanism, similar to Equation 3.2. To identify the number of 18-year-olds who have height in $[u, v)$, again one would search for a b value on the boundary of k-local sensitivity changes, using Algorithm 1, but modified to use the k-local laplace mechanism for this new query. Then using Algorithm 2 with this modified Algorithm 1 allows us to find all of the heights associated with 18-year-olds in the data. Then we would repeat the process with 21-year-olds and 30-year-olds.

Algorithm 2: Reconstruct all elements in the column corresponding to attribute $\mathbf{A}_i$

k, ϵ: Group IDP parameters
γ: targeted decimal point precision of each reconstructed element
$n \geq 2k$: publicly known data size
def *ColumnReconstruct(k, ϵ_{target}, γ)*:
  $u \leftarrow$ Lower bound on domain of $\mathbf{A}_i$
  $v \leftarrow$ Upper bound on domain of $\mathbf{A}_i$
  `/* Target privacy parameter for each call to Algorithm 1:`
  `CountReconstruct() */`
  $\epsilon_{share} = \min(10^{-10}, \frac{\epsilon}{(v-u)/\gamma})$
  $vals \leftarrow []$ `// Will store reconstructed values`
  $s_0 \leftarrow n$ `// Number of items left to reconstruct`
  `// Note linear search is shown for simplicity`
  `// Use binary search for more efficiency`
  while $s_0 \neq 0$ **do**
    $v \leftarrow v - \gamma$ `// decrease upper bound`
    $s_1 \leftarrow$ CountReconstruct(k, ϵ_{share}, u, v, i)
    if $s_0 \neq s_1$ **then**
      `// There are` $s_0 - s_1$ `items in` $[v, v+\gamma)$
      add $s_0 - s_1$ copies of v into $vals$ array
      $s_0 \leftarrow s_1$
  return $vals$

Continuing this process with a third attribute, then a fourth, etc., would result in the entire dataset being reconstructed as long as it contains at least $2k$ people.

3.5. **Additional Attacks.** The attack algorithm can be made efficient by replacing linear search in Algorithms 1 and 2 with binary search. The algorithms could also be adapted for other kinds of attacks, not just entire data reconstruction.

Example 3.8 (Confirmation of Uniqueness)**.** Suppose the dataset schema is $\mathbf{A}_1, \ldots, \mathbf{A}_m$ and we know the values of ℓ of these attributes for a target individual's record r^* (say we know $r^*[\mathbf{A}_1] = a_1, \ldots, r^*[\mathbf{A}_\ell] = a_\ell$). We may ask if that person is unique in the data for those attributes. We can consider the following query q_b: is the number of records r with $r[\mathbf{A}_1] = a_1, \ldots, r[\mathbf{A}_\ell] = a_\ell$ larger than b? This is a predicate count query with threshold b and its k-local sensitivity is given in Lemma 3.3. Namely, when the true count of such records is $> b + k$ or $\leq b - k$, the k-local sensitivity is 0, otherwise it is 1. Let M_b be the k-Laplace mechanism for this query. Then we run M_b with $b = k$ and a tiny privacy budget and then we run it with $b = k + 1$. By Lemma 3.6, this would be an upper boundary for the k-local sensitivity change (i.e., the k-local sensitivity is detected as 1 for $b = k$ and detected as 0 for $b = k + 1$) if and only if there truly is only one such person in the data. Thus, if we observe this combination of non-noisy answer for $b = k + 1$ and a noisy answer for $b = k$, we learn the person is unique in the dataset on those attributes. On the other hand, if we

observe a different outcome, then we learn that the person is not unique. This attack only requires 2 accesses to the mechanism.

Example 3.9 (Membership Inference)**.** Suppose we know that an individual is unique in the population based on attributes $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$, and we want to know whether they are in the dataset (e.g., it could be an HIV dataset). This attack would proceed in the same way as in Example 3.8, except we run the mechanism M_b from that example with $b = k$ and $b = k - 1$ (with very small privacy budgets). If there are 0 people in the dataset with those combinations of attributes, the k-local sensitivity would be 1 when $b = k - 1$ and 0 when $b = k$. Thus again, this attack looks for the upper boundary and that is why 2 mechanism calls are needed (i.e., to identify which boundary it is).

Note that if the query was never protected, we would simply ask 1 query: if the number of people with a particular combination of uniquely identifying attributes is > 0. If the answer is True, then the person is in the dataset, if False, then they are not. This is an interesting observation because, even though the k-local laplace mechanism reveals more precise information about the dataset (as explained in Section 3.2.2), this more precise information is more complex: (1) when the query is protected, there are two boundaries and we need to determine which one we found; (2) when the query is unprotected, there is only one relevant boundary – the b at which the answer changes from True to False. If a unique person is in the data, this boundary would occur at $b = 0$.

Example 3.10 (Attribute Inference)**.** Suppose we know that an individual is in the dataset and we know the values for attributes $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ for that individual. We may be interested in learning the value of $\mathbf{A}_{\ell+1}$. This is exactly what the dataset reconstruction algorithm does (section 3.4) – it finds the multiset of values for $\mathbf{A}_{\ell+1}$ for the records for which $\mathbf{A}_1 = a_1, \ldots, \mathbf{A}_\ell = a_\ell$. The reconstruction algorithm does this for all combinations of $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell)$ values that appear in the dataset, but clearly it can be specialized to target just one particular combination as well.

3.6. **Countermeasures.** There are a variety of possible countermeasures to the attack we propose here. We list some of these countermeasures and discuss their implications.

- Instead of adding noise to the query answer directly, one could add noise to the count of the records that satisfy ϕ and then threshold the noisy count (convert it to 1 if it is $> b$ and 0 otherwise). This turns the mechanism into a differentially private mechanism and abandons IDP.
- A system could specifically disallow these queries or allow only a restricted set of possible queries. Not only would this greatly limit the ability of researchers to perform data analysis, but systems based on query restrictions may not guarantee privacy protections either [Cohen and Nissim, 2018]. Using these kinds of restrictions would also tacitly acknowledge that the framework of IDP is not trusted, whereas a formal privacy definition should guarantee protections against reconstruction.
- A system could limit the number of queries that an analyst can ask. This approach has the same limitations as the previous one. Additionally, we note that membership inference only requires 2 queries, so even query limitation is not very effective.
- A system could require queries to use a minimum privacy loss budget – in other words the system would require an analyst to use queries that leak at least a pre-specified amount of information. This can have the unintended consequence of losing the trust of people

whose data are to be collected. It also implicitly restricts the amount of queries an analyst can ask, since analysts are also constrained by the sum of the leakages associated with the queries they ask.

4. A Brief Examination of Bootstrap Differential Privacy

In this section, for the purposes of comparison with IDP, we review bootstrap differential privacy (BDP) [O'Keefe and Charest, 2019] and demonstrate how it can leak the distinct set of records in the data, using the preferred mechanism construction of O'Keefe and Charest [2019].

4.1. Bootstrap Differential Privacy.

BDP again defines its own versions of neighbors, sensitivity, and Laplace mechanism as follows.

Definition 4.1 (Bootstrap Neighbors [O'Keefe and Charest, 2019])**.** Given a true dataset D_{act}, we say that D_1 and D_2 are bootstrap neighbors conditioned on D_{act} if D_1 can be obtained from D_2 by replacing one record and both D_1 and D_2 can be obtained from D_{act} by changing the multiplicities of records in D_{act} (i.e., D_1 and D_2 cannot contain a record that D_{act} does not contain).

Definition 4.2 (ϵ-bootstrap differential privacy (BDP) [O'Keefe and Charest, 2019])**.** Given a dataset D_{act} and privacy parameter $\epsilon > 0$, a randomized algorithm M satisfies ϵ-bootstrap differential privacy if for every set $\mathrm{S} \subseteq \mathrm{Range}(M)$ and for all pairs of bootstrap neighboring data sets D_1 and D_2 (conditioned on D_{act}),

$$Pr[M(D_1) \in S] \leq e^{\epsilon} Pr[M(D_2) \in S] \tag{4.1}$$

The Bootstrap sensitivity (BS) of a function f with respect to D_{act} takes the usual definition of sensitivity from differential privacy and swaps in bootstrap neighbors conditioned on D_{act}.

Definition 4.3 (Bootstrap Sensitivity [O'Keefe and Charest, 2019])**.** The Bootstrap sensitivity (BS) of f with respect to D_{act}, denoted by ${\Lambda^s}_B(f, D)$ is

$${\Lambda^s}_B(f, D_{\text{act}}) = \max_{D_1 \sim D_2} ||f(D_1) - f(D_2)||_1 \tag{4.2}$$

where the maximum is taken over bootstrap neighbors conditioned on D_{act}.

Bootstrap sensitivity is used to calibrate noise for the bootstrap Laplace mechanism:

Definition 4.4 (Bootstrap Laplace Mechanism [O'Keefe and Charest, 2019])**.** Let f be a function whose output is a vector. Let $\epsilon^* > 0$ be a privacy parameter. The bootstrap Laplace mechanism is a mechanism M that, on input D_{act}, adds independent Laplace noise with scale ${\Lambda^s}_B(f, D_{\text{act}})/\epsilon^*$ to each component of f (i.e. $M(D) = f(D) + \mathrm{Laplace}({\Lambda^s}_B(f, D)/\epsilon^*)$.

We next show how the bootstrap Laplace mechanism can be used to verify the existence or non-existence of any record with almost no privacy expenditure. Let ϕ be an arbitrary predicate (e.g., "Income > 50k and Age = 32") and let q_ϕ be the query that returns 1 if and only if some record in D_{act} satisfies ϕ:

$$q_\phi(D) = \begin{cases} 1 & \phi(r) = \text{True for some } r \in D \\ 0 & \phi(r) = \text{False for all } r \in D \end{cases}$$

Lemma 4.5. *Given a true dataset D_{act}, the bootstrap sensitivity of q_ϕ with respect to D_{act} is:*

$$\Lambda^s{}_B(q_\phi, D_{act}) = \begin{cases} 0 & \text{if all } r \in D_{act} \text{ satisfy } \phi \\ 0 & \text{if no } r \in D_{act} \text{ satisfies } \phi \\ 1 & \text{otherwise} \end{cases}$$

□

Thus, if one uses the bootstrap Laplace mechanism with a tiny privacy loss budget (e.g., $\epsilon = 10^{-10}$) to answer q_ϕ, then Lemma 4.5 tells us that:

(1) If we receive the answer 0, then with overwhelming probability, no record in D_{act} satisfies ϕ (because it is almost impossible for an extremely noisy answer to be 0 or 1, hence this must have been a noise-free answer and the bootstrap sensitivity would have to be 0).
(2) If we receive the answer 1, then with overwhelming probability, all records in D_{act} satisfy ϕ.
(3) If we receive any other answer, it is definitely a noisy answer (bootstrap sensitivity is 1) and so there exists a record in D_{act} satisfying ϕ.

Thus the output of the bootstrap Laplace mechanism tells us whether there is or is not such a record in the database (i.e., we have figured out what the true answer to q_ϕ is) and sometimes it tells us more information (when all records satisfy ϕ or all do not). So again, protecting the query with the bootstrap Laplace mechanism reveals at least as much information compared to always answering q_ϕ accurately. It allows us to probe which records are in D_{act} (but not how many copies there are).

5. A General Study of Empirical Neighbors Definitions

We have demonstrated an attack against (ϵ, k)-Group IDP that recovers any dataset with at least $2k$ records and sketched an attack against BDP that recovers all records in the dataset but not their multiplicity. Both attacks work with arbitrarily low privacy budget parameters ϵ (which, according to those definitions should correspond to strong privacy protections). In this section, we consider whether there are simple fixes for this style of privacy definition that can prevent reconstruction or whether the problems are deeper and harder to fix.

5.1. **Ensuring all answers are noisy.** In the previous sections, we exploited the fact that we can detect whether a query answer is noisy or not using arbitrarily small amounts of privacy budget. What if one changes the mechanism so that noise is always added? For example, consider the following modification to the k-local Laplace Mechanism: add 1 to the k-local sensitivity and use that to calibrate the noise. That is, if g is a function with k-local sensitivity $\Lambda^s{}_k(g, D_{\text{act}})$ with respect to D_{act}, then this modified mechanism $M^\dagger$ returns:

$$M^\dagger(D_{\text{act}}; \epsilon) = g(D_{\text{act}}) + \text{Laplace}\left(\frac{1 + \Lambda^s{}_k(g, D_{\text{act}})}{\epsilon}\right)$$

Such a mechanism, when given privacy loss budget ϵ, would be answering the threshold range-count queries $q_{\mathbf{A}_i \in [u,v), b}$ from Section 3.2 by either adding Laplace$(2/\epsilon)$ noise (when the the k-local sensitivity is 1) or Laplace$(1/\epsilon)$ noise (when the k-local sensitivity is 0). If we can reliably detect which type of noise was added, then the same reconstruction attacks from Section 3 could be used.

m	accuracy
10	$\frac{1{,}606{,}049}{2{,}000{,}000} = 80.30245\%$
100	$\frac{1{,}976{,}433}{2{,}000{,}000} = 98.82165\%$
1000	$\frac{2{,}000{,}000}{2{,}000{,}000} = 100.00000\%$

Table 1: Empirical accuracy of the decision rule based on ϕ_m, for different values of m for 2 million simulations.

It turns out that this is also possible using statistical hypothesis testing and exploiting the composition rules for privacy definitions like IDP and BDP. For example, given a desired target ϵ and an integer m, we can run the mechanism $M^\dagger$ for m times, each time using ϵ/m privacy budget for a total cost of ϵ. This gives us m noisy numbers $z_1, \ldots, z_m$ which are either obtained by adding Laplace$(2m/\epsilon)$ noise to an unknown quantity, or Laplace(m/ϵ) noise. We can use the empirically observed variance as a test statistic ψ_m:

$$\psi_m = \frac{\epsilon^2}{m^2}\left[\frac{1}{m-1}\sum_{i=1}^{m}\left(z_i - \frac{z_1+z_m}{m}\right)^2\right]$$

If the z_i are generated with Laplace$(2m/\epsilon)$ noise, then the variance of each z_i is $8m^2/\epsilon^2$ and so the expected value of ψ_m would be $\frac{\epsilon^2}{m^2}\frac{8m^2}{\epsilon^2} = 8$. On the other hand, if the z_i are generated with Laplace(m/ϵ) noise, then the variance of each z_i is $2m^2/\epsilon^2$ and so the expected value of ψ_m would be $\frac{\epsilon^2}{m^2}\frac{2m^2}{\epsilon^2} = 2$.

Thus, there is a simple decision rule one could use. Run the mechanism $M^\dagger$ m times, with ϵ/m privacy budget each time. Compute the test statistic ψ_m and if it is < 5, decide that Laplace(m/ϵ) was used (hence k-local sensitivity is 0), otherwise decide that Laplace$(2m/\epsilon)$ (hence k-local sensitivity is 1). If this decision rule is highly accurate, then this is all that is needed to perform reconstruction using the algorithms in Section 3.

The following lemma shows that when m is large enough, ψ_m is highly concentrated around its mean (either 2 or 8), and so the decision rule is very accurate. An empirical demonstration is also shown in Table 1 which shows the empirical accuracy for different values of m. It is based on 2 million simulations, in which half of the simulations used Laplace$(2m/\epsilon)$ noise and the other half used Laplace(m/ϵ) noise. Note the total privacy budget expended is always ϵ, regardless of the value of m, and that by Lemma 5.1, the decision rule has the same accuracy for any value of $\epsilon > 0$.

Given an integer $m > 1$ and any $\epsilon > 0$ and a noise scale multiplier $\alpha \geq 0$, define the following random variables:

(1) $z_1, \ldots, z_m$, where each $z_i = \mu$+Laplace$(\alpha m/\epsilon)$ for some unknown number μ (the private value that gets noised).
(2) $z_1^*, \ldots, z_m^*$ where each z^* =Laplace(1)

Furthermore, define:

$$\psi_m = \frac{\epsilon^2}{m^2}\left[\frac{1}{m-1}\sum_{i=1}^{m}\left(z_i - \frac{z_1+z_m}{m}\right)^2\right]$$
$$\psi_m^* = \frac{1}{m-1}\sum_{i=1}^{m}\left(z_i^* - \frac{z_1^*+z_m^*}{m}\right)^2$$

Then the distribution of ψ_m is the same as the distribution of $\alpha^2\psi_m^*$ (in particular, it doesn't depend on ϵ or the private value μ). The expected value of ψ_m^* (resp., ψ_m) is 2 (resp., $2\alpha^2$) and ψ_m^* converges to 2 with probability 1 as $m \to \infty$ (hence ψ_m converges to $2\alpha^2$).

Thus, the foundational mechanisms for these privacy definitions are flawed and reconstruction-proof fixes most likely require more complex strategies like smooth sensitivity [Nissim et al., 2007] in differential privacy. We next examine flaws in the formulation of the privacy definitions themselves.

5.2. **Is leakage built in to the privacy definition?** We saw that simple modifications to the mechanisms to make sure that they always add noise is still not sufficient to protect against reconstruction (one would need to use something much more complex, such as smooth sensitivity [Nissim et al., 2007] and differential privacy). So next we study the general class of privacy definitions that IDP and BDP belong to in order to identify further flaws. We call this class of definitions *empirical neighbors*. The main components of empirical neighbors privacy definitions are:

(1) A set of pairs of neighbors to protect. This set depends on D_{act}, the data observed by the data collector. Hence we represent it as $NPairs(D_{\text{act}})$. The privacy constraints are obtained from $NPairs(D_{\text{act}})$ – for each $(D_1, D_2) \in NPairs(D_{\text{act}})$ and each possible output ω of a mechanism M, they require that $P(M(D_1) = \omega) \leq e^{\epsilon}P(M(D_2) = \omega)$. For example, in IDP, $NPairs(D_{\text{act}})$ has the form

$$\{(D_{\text{act}}, D_1), (D_{\text{act}}, D_2), \dots\} \cup \{(D_1, D_{\text{act}}), (D_2, D_{\text{act}}), \dots\}$$

where $D_1, D_2, \dots$ are the datasets that can be obtained from D_{act} by replacing one record. Similarly, in BDP, $NPairs(D_{\text{act}})$ contains all pairs (D_1, D_2) where D_1 can be obtained from D_2 by replacing one record and all records that appear in D_1 and D_2 must also appear in D_{act}.

(2) A hint function h that looks at the data. The data collector decides which mechanism to use based on the hint $h(D_{\text{act}})$. We note that the use of such hint functions is becoming increasingly common. Not only is it implicitly used in IDP [Soria-Comas et al., 2017] and BDP [O'Keefe and Charest, 2019] but it was also used for actual data releases for the Opportunity Atlas [Chetty and Friedman, 2019] and the 2020 Decennial Census (the as-enumerated population count in each state as well as the number of housing units and non-empty group quarters in each geographic area) [Abowd et al., 2022]. In fact, many papers on differential privacy implicitly use $h(D_{\text{act}}) = |D_{\text{act}}|$ because they reveal the exact size of the dataset.

(3) A mechanism selector *Chooser* whose input is $h(D_{\text{act}})$ and whose output is a mechanism that satisfies the constraints obtained by $NPairs(D_{\text{act}})$. This reflects the core principles in IDP and BDP that a mechanism is chosen after observing the data.

These pieces fit together into a privacy definition, generalizing IDP and BDP, as follows:

Definition 5.1 (Empirical Neighbors)**.** Given $NPairs$ and a hint function h, a mechanism chooser satisfies ϵ-empirical neighbors privacy if for any choice of D_{act}, then $Chooser(h(D_{\text{act}}))$ produces a mechanism M that satisfies:

$$Pr[M(D_1) = \omega] \leq e^{\epsilon} Pr[M(D_2) = \omega]$$

for all $(D_1, D_2) \in NPairs(D_{\text{act}})$ and all possible outputs ω.

Both IDP and BDP don't explicitly state the rules that must be followed when choosing a mechanism – what information about D_{act} can be used? Equivalently, what restrictions are there on the hint function h? Because these rules were not fully specified, our attacks had to use the mechanism design principles provided by those papers.

Some natural choices for h are: (1) no restrictions, (2) $h(D_{\text{act}}) = \emptyset$, (3) $h(D_{\text{act}}) = NPairs(D_{\text{act}})$ – in other words, h provides information equivalent to the set of constraints that the mechanism selected by *Chooser* should satisfy (this is most likely what was intended in IDP and BDP).

We next show the consequences of each of these choices, which is that this style of definition allows $h(D_{\text{act}})$ (the information used to decide on a mechanism) to be leaked, which can be catastrophic in the cases of IDP and BDP. Furthermore, preventing the leakage of $h(D_{\text{act}})$ results in differential privacy.

Lemma 5.2. *The empirical neighbors definitions allow the release of $h(D_{act})$ for any ϵ parameter. In particular, if $h(D_{act}) = NPairs(D_{act})$ then Group IDP allows D_{act} to be revealed whenever $|D_{act}| > 1$ and BDP allows the distinct set of records to be revealed.*

On the other hand, if $h(D_{act}) = \emptyset$ and $\bigcup_D NPairs(D)$ is the set of all pairs of datasets that differ on a record, then the empirical neighbors definition is equal to differential privacy.

6. Experiments

As a proof of concept, we empirically evaluate the attacks against IDP because of its leakage potential. We consider 3 attack scenarios:

- **Membership inference**: given a set of uniquely identifying attributes of a target individual, how many queries does an attacker need to verify that the target individual is in the dataset, using arbitrarily low privacy budget.
- **Attribute inference**: given a set of uniquely identifying attributes of a target individual, how many queries does an attacker need to reconstruct the rest of the target's record, using arbitrarily low privacy budget.
- **Full dataset reconstruction**: how many queries does an attacker need to reconstruct the entire dataset, using arbitrarily low privacy budget.

In these experiments, we optimize the attack code of Algorithms 1 and 2 to use binary search instead of sequential search. We compare (1) how many queries are needed when they are "protected" by the k-local Laplace mechanism vs. (2) how many queries are needed when no

protection is used (i.e., they are always answered without noise). When reconstructing using "protected" threshold range-count queries $q_{\mathbf{A}_i \in [u,v),b}$ the main idea is to look for a threshold b for which the threshold range query mechanism M switches from noisy to non-noisy answers. When reconstructing using unprotected queries, one looks for the threshold b for which the query answer changes from 0 to 1. The attack is for IDP (Group IDP with $k = 1$).

It is important to note that, as discussed in Example 3.9, just because the k-Laplace mechanism can leak more information about a query (such as $q_{\mathbf{A}_i \in [u,v),b}$) than if the query were not protected, this does not mean that our particular attack will benefit from it. Hence it is important to evaluate if there are inefficiencies in our attack.

6.1. **The Dataset.** As an illustration of the way the attack would be launched in practice, we use the well-known Banking dataset [Moro et al., 2014] containing records of 45211 people. There are 7 numeric (integer) and 10 categorical attributes. As discussed in Section 3.2, categorical attributes can be handled simply by encoding the values as integers (thus, for example, a yes/no attribute can be converted to an attribute whose values are 0 or 1).

Since part of the attack (Algorithm 2) uses upper and lower bounds on the domain of numeric attributes, we choose the following conservative bounds: $\left[-10^5, 10^6\right]$ for **balance**, $\left[0, 10^4\right]$ for **duration**, $[-1, 2000]$ for **pdays** (-1 is a special coding for customers who were not previously contacted), $[0, 2000]$ for **previous**, $[0, 125]$ for **age**, $[0, 31]$ for **day**, and $[0, 100]$ for **campaign**.

6.2. **Membership Inference.** In membership inference attacks, an attacker has uniquely identifying information about an individual and attempts to determine whether that individual is in the dataset (i.e., whether the number of people having the same values for those known attributes is 0 or 1). As explained in Example 3.9, when using the k-local Laplace mechanism to protect query answers, then no matter how small the privacy budget ϵ is, this attack succeeds with just two queries (no matter what the dataset is). If, on the other hand, queries are not protected at all, one simply asks whether the number of people with the known attributes is > 0. These results are summarized in Table 2.

	Protected by IDP	**No Protection**
# Queries	2	1

Table 2: Number of queries needed to launch a successful membership inference attack, no matter how small $\epsilon > 0$ is, when queries are protected using the k-local Laplace mechanism of IDP vs. no protection at all.

6.3. **Attribute Inference.** We next consider an attacker who knows a person is in the data, has uniquely identifying information about the person, and is trying to discover additional attributes (like the person's **balance** in the banking dataset).

In this experiment, an attacker knows the following attributes about a target individual: **age**, **marital status**, **level of education**, **job type**, and whether the individual has a **housing loan**. These will be treated as the identifying attributes. The attacker could try to learn just one attribute (in this case it would be **balance**) or the attacker could try to learn the complete rest of the entire record.

	Protected by IDP	**No Protection**
"Balance" # Queries	29.9	29.9
Full Record # Queries	131.2	131.2

Table 3: Average number of queries to reconstruct the **balance** for people who are unique on linking attributes and the average number of queries to reconstruct all the non-linking attributes. Comparison between threshold range-count queries with and without IDP protection.

This attack can be carried out, for any arbitrarily low privacy loss budget $\epsilon > 0$, as described in Example 3.10. In the dataset, there were 1815 people who are unique on the linking attributes. In Table 3 we show, on average, how many queries are needed to learn the balance attribute for those unique people, and how many queries are needed to learn the entire record.

To help interpret the numbers better, consider the **balance** attribute, for which we used lower and upper bounds of $-€100,000$ and $€1,000,000$, which is a range that can be represented using 21 bits. Thus, on average we need $29.9/21 \approx 1.4$ queries per bit. This number can be further reduced if an attacker is not interested in the exact amount and only needs a few significant digits, or if the attacker already has a ballpark estimate of the target's balance.

Also note that **balance** was the attribute with the largest domain. The remaining 11 non-linking attributes are reconstructed using, on average, $131.2 - 29.9 = 101.3$ additional queries. We note that recovering a binary attribute $\mathbf{A}_{binary}$ is particularly straightforward. If we know someone is in the data and we know they are unique on a set of attributes $\mathbf{A}_1 = a_1, \ldots, \mathbf{A}_m = a_m$ then, if the value of $\mathbf{A}_{binary} = 1$, there would only be one person in the data with $\mathbf{A}_1 = a_1, \ldots, \mathbf{A}_m = a_m, \mathbf{A}_{binary} = 1$. Thus we can perform a membership inference attack with this combination of attributes and if the attack returns "true," it means that $\mathbf{A}_{binary} = 1$ for the target person, and if it returns "false," then $\mathbf{A}_{binary} = 0$. The cost of this is simply 2 "protected" queries.

6.4. **Dataset Reconstruction.** Efficient membership inference and partial/full record reconstruction for a target individual is already a strong demonstration of the exploitability of IDP. We next show that there are savings in bulk when performing full dataset reconstruction. That is, the *average* number of queries *per person* needed for reconstruction is less than the number of queries needed to attack a person individually because an attribute value may be shared by multiple people, so using one binary search to find this value and its count would produce results for multiple people at once. To take advantage of this type of bulk savings, we perform reconstruction starting with the binary attributes and then adding attributes to the reconstruction in order of the size of their domain.

We consider two types of experiments: how much effort is needed to reconstruct a single attribute, and how much effort is needed to reconstruct the entire dataset.

6.4.1. *Single Attribute Reconstruction.* Here we study how many queries are needed to reconstruct each attributes in isolation. That is, for each attribute, we are just interested in determining what are the distinct values that are present, and how many people have those values (in other words, we want to get an exact 1-way marginal). The number of queries

	IDP Protection	**No Protection**
Total number of queries	5,418,936	5,200,591
Queries per person	≈ 119.9	≈ 115.0
Queries per data element	≈ 7.1	≈ 6.8
Total privacy budget spent	0.0005418936	N/A

Table 4: Full dataset reconstruction using threshold range-count queries, with and without the protection mechanisms of IDP. Each protected query access used $\epsilon = 10^{-10}$ of the privacy budget.

depends on the number of unique values that appear for that attribute and are shown in Figure 2.

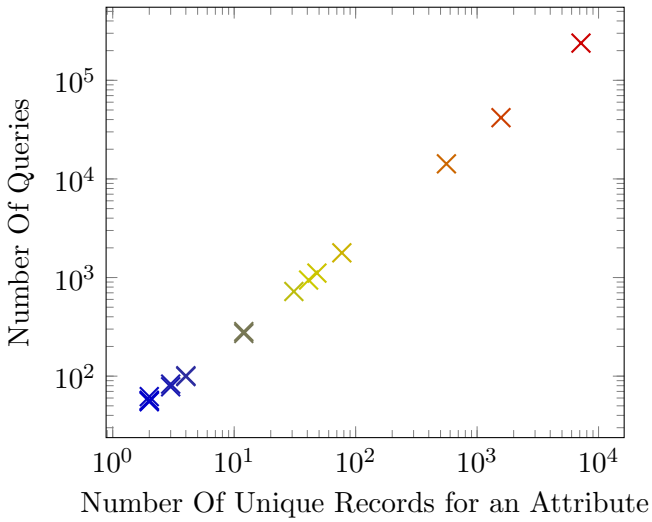

Figure 2: Number of queries needed to reconstruct an attribute in isolation. The plot indicates the number of queries vs. the number of distinct values that appear for the attribute.

Note that reconstructing a binary attributes means determining exactly how many people had 0 (resp. 1) for that attribute and so requires a binary search that takes $O(\log(n))$ queries per attribute value (0 or 1), where n is the number of people. The most difficult attribute to reconstruct as **balance**, which had 7168 distinct values in the dataset. Reconstruction required 238,462 queries, which is approximately 33 queries per distinct value, or 5 queries per person.

6.4.2. *Entire Dataset Reconstruction.* The entire dataset consists of 17 attributes and contains 45,211 people, meaning that a full reconstruction is required to produce $17 * 45211 = 768587$ total items. Thus, one would expect that the number of queries would be in the millions. We allocated an $\epsilon = 10^{-10}$ for each use of the k-local Laplace mechanism. The results are shown in Table 4.

Note that the total privacy budget spent (according to IDP privacy accounting) reconstructing the entire dataset was approximately 0.0005. It can be made arbitrarily small. For

example, if we used 10^{-11} per query, then the privacy budget would be 1/10th the size. In fact, for any target ϵ^*, it is possible to guarantee that the total spent is at most ϵ^*. For example, one could allocate $\epsilon_1 = \min(\epsilon^*/2, 10^{-10})$ for the first query, $\epsilon_2 = \epsilon_1/2$ for the next query, $\epsilon_3 = \epsilon_2/2$ for the third query, and so on. This guarantees that the total spent is at most ϵ^*.

7. Related Work

Reconstruction attacks are possible when too many queries over confidential data are answered too accurately [Dinur and Nissim, 2003], or equivalently, when one tries to create a data product that supports all possible use-cases. This is not just a theoretical possibility, but also affects commercial offerings [Cohen and Nissim, 2018].

Differential privacy, formally introduced in 2006 [Dwork et al., 2006b], has been gaining steam as a mathematically rigorous privacy definition that protects against reconstruction and other embarrassing privacy attacks against public data products. This property allows organizations to use it to collect, protect and publish data products that would otherwise not be available at all.

There has been significant research on trying to improve the accuracy of the data products by carefully weakening the original differential privacy definition, while still preventing reconstruction. This includes approximate differential privacy [Dwork et al., 2006a], concentrated differential privacy [Bun and Steinke, 2016], Renyi differential privacy [Mironov, 2017], and f-DP [Dong et al., 2022]. These privacy definitions have *group privacy* guarantees which is what prevents reconstruction attacks [Vadhan, 2017].

There have, in fact, been numerous attempts to weaken differential privacy, strengthen it, and apply it to non-tabular data – see the comprehensive comparative survey by Desfontaines and Pejó [2020].

One of the lines of research taken, which we call *empirical neighbors* is spearheaded by IDP [Soria-Comas et al., 2017]. It is noteworthy for several reasons: (1) it was proposed by several long-term experts in privacy, (2) its flaws were not observed in the authoritative comparative survey [Desfontaines and Pejó, 2020] or the literature that cites IDP (e.g., [Pratesi et al., 2018]), and (3) most notably, a group of prominent researchers, mostly from the economics field, called on the Census Bureau to stop using differential privacy and to explore alternatives [Hotz et al., 2022]. One approach, of adding noise depending on the local sensitivity (e.g., IDP) was deemed "sensible" as long as the local sensitivity itself is not explicitly revealed [Hotz et al., 2022, appendix c]. In fact, their concern with local sensitivity was that it might not provide enough *utility*.

It is known that adding noise based on local sensitivity does not satisfy differential privacy, and hence smooth sensitivity was proposed [Nissim et al., 2007] and many believed that the main weakness of local sensitivity occurs when the local sensitivity is explicitly published [Hotz et al., 2022, Chetty and Friedman, 2019]. However, the weaknesses we have demonstrated: queries that reveal more information when protected by IDP than if they had no protection at all (even when the local sensitivity is not explicitly published), and their use membership attacks, attribute inference, and full dataset reconstruction at arbitrarily low privacy costs (according to IDP privacy accounting) was not previously known, to the best of our knowledge.

Several authors investigated something similar to IDP, but as a diagnostic tool rather than a method for selecting mechanisms [Charest and Hou, 2017, Redberg and Wang, 2021].

Here, a differentially private mechanism M is chosen, it is applied to the dataset, and the privacy with respect to that dataset is studied after the fact. Charest and Hou [2017] used this methodology to compute an ϵ (this would be the ϵ that IDP would assign to M) and studied how well it correlates to the differential privacy ϵ. They concluded that it was not a good estimate. Redberg and Wang [2021] studied how to make this ex-post analysis differentially private, so that the actual privacy cost of M on the actual dataset could be revealed without breaching privacy.

8. Conclusion

In this paper, we studied a class of privacy definitions called *empirical* neighbors that condition on the observed data when choosing a mechanism. We showed that the preferred mechanisms can be exploited to reveal significant information about the true data. We also showed that the definitions themselves can be exploited to design mechanisms that directly leak private information. It is not clear whether this style of privacy definition can provide the right balance between privacy and utility in practice.

Acknowledgment

This work was supported by an NSF BAA award number 49100421C0022 and by NSF award CNS-1702760.

References

J. Abowd, R. Ashmead, R. Cumings-Menon, S. Garfinkel, M. Heineck, C. Heiss, R. Johns, D. Kifer, P. Leclerc, A. Machanavajjhala, B. Moran, W. Sexton, M. Spence, and P. Zhuravlev. The 2020 Census Disclosure Avoidance System Top-Down Algorithm. *Harvard Data Science Review*, (Special Issue 2), jun 24 2022. https://hdsr.mitpress.mit.edu/pub/7evz361i.

J. M. Abowd. The us census bureau adopts differential privacy. In *KDD*, 2018.

J. M. Abowd, R. Ashmead, R. Cumings-Menon, S. L. Garfinkel, D. Kifer, P. Leclerc, W. Sexton, A. E. Simpson, C. Task, and P. Zhuravlev. An uncertainty principle is a price of privacy-preserving microdata. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=6tGP5Z-QbMb.

Apple Differential Privacy Team. Learning with privacy at scale. *Apple Machine Learning Journal*, 1(8), 2017.

V. Balcer and S. Vadhan. Differential privacy on finite computers. *Journal of Privacy and Confidentiality*, 9(2), Sep. 2019.

A. Bittau, U. Erlingsson, P. Maniatis, I. Mironov, A. Raghunathan, D. Lie, M. Rudominer, U. Kode, J. Tinnes, and B. Seefeld. Prochlo: Strong privacy for analytics in the crowd. In *SOSP*, 2017.

M. Bun and T. Steinke. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Proceedings, Part I, of the 14th International Conference on Theory of Cryptography - Volume 9985*, 2016.

A.-S. Charest and Y. Hou. On the meaning and limits of empirical differential privacy. *Journal of Privacy and Confidentiality*, 7(3):53–66, May 2017. doi: 10.29012/jpc.v7i3.406. URL https://journalprivacyconfidentiality.org/index.php/jpc/article/view/406.

R. Chetty and J. N. Friedman. A practical method to reduce privacy loss when disclosing statistics based on small samples. *Journal of Privacy and Confidentiality*, 9(2), Oct. 2019. doi: 10.29012/jpc.716. URL https://journalprivacyconfidentiality.org/index.php/jpc/article/view/716.

A. Cohen and K. Nissim. Linear program reconstruction in practice. *CoRR*, abs/1810.05692, 2018. URL http://arxiv.org/abs/1810.05692.

D. Desfontaines and B. Pejó. Sok: Differential privacies. *Proceedings on Privacy Enhancing Technologies*, 2020(2):288–313, 2020. doi: doi:10.2478/popets-2020-0028. URL https://doi.org/10.2478/popets-2020-0028.

B. Ding, J. Kulkarni, and S. Yekhanin. Collecting telemetry data privately. In *NIPS*, 2017.

I. Dinur and K. Nissim. Revealing information while preserving privacy. In *Proceedings of the Twenty-Second ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '03, page 202–210, New York, NY, USA, 2003. Association for Computing Machinery. ISBN 1581136706. doi: 10.1145/773153.773173. URL https://doi.org/10.1145/773153.773173.

J. Dong, A. Roth, and W. J. Su. Gaussian differential privacy. *JRSS-B*, 84:3–37, 2022. URL https://doi.org/10.1111/rssb.12454.

C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In S. Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006*, pages 486–503, Berlin, Heidelberg, 2006a. Springer Berlin Heidelberg. ISBN 978-3-540-34547-3.

C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. 2006b.

H. Ebadi, D. Sands, and G. Schneider. Differential privacy: Now it's getting personal. In *Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '15, page 69–81, 2015.

Ú. Erlingsson, V. Pihur, and A. Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *CCS*, 2014.

S. Haney, A. Machanavajjhala, J. M. Abowd, M. Graham, M. Kutzbach, and L. Vilhuber. Utility cost of formal privacy for releasing national employer-employee statistics. In *SIGMOD*, 2017.

M. Hardt and K. Talwar. On the geometry of differential privacy. In *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, STOC '10, page 705–714, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781450300506. doi: 10.1145/1806689.1806786. URL https://doi.org/10.1145/1806689.1806786.

V. J. Hotz, C. R. Bollinger, T. Komarova, C. F. Manski, R. A. Moffitt, D. Nekipelov, A. Sojourner, and B. D. Spencer. Balancing data privacy and usability in the federal statistical system. *Proceedings of the National Academy of Sciences*, 119(31):e2104906119, 2022. doi: 10.1073/pnas.2104906119. URL https://www.pnas.org/doi/abs/10.1073/pnas.2104906119.

N. Johnson, J. P. Near, and D. Song. Towards practical differential privacy for sql queries. In *PVLDB*, 2018.

Z. Jorgensen, T. Yu, and G. Cormode. Conservative or liberal? personalized differential privacy. In *ICDE*, 2015.

S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. What can we learn privately? In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 531–540, 2008. doi: 10.1109/FOCS.2008.27.

A. Machanavajjhala, D. Kifer, J. Abowd, J. Gehrke, and L. Vilhuber. Privacy: From theory to practice on the map. In *ICDE*, 2008.

A. McGregor, I. Mironov, T. Pitassi, O. Reingold, K. Talwar, and S. Vadhan. The limits of two-party differential privacy. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 81–90, 2010.

I. Mironov. Rényi differential privacy. In *30th IEEE Computer Security Foundations Symposium, CSF 2017, Santa Barbara, CA, USA, August 21-25, 2017*, pages 263–275, 2017.

S. Moro, P. Cortez, and P. Rita. A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems*, 62, 06 2014. doi: 10.1016/j.dss.2014.03.001.

K. Nissim, S. Raskhodnikova, and A. D. Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing*, 2007.

C. M. O'Keefe and A.-S. Charest. Bootstrap differential privacy. *Trans. Data Priv.*, 12:1–28, 2019.

F. Pratesi, A. Monreale, R. Trasarti, F. Giannotti, D. Pedreschi, and T. Yanagihara. Prudence: A system for assessing privacy risk vs utility in data sharing ecosystems. *Transactions on Data Privacy*, 11:139–167, 08 2018.

R. Redberg and Y.-X. Wang. Privately publishable per-instance privacy. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 17335–17346. Curran

Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/9087b0efc7c7acd1ef7e153678809c77-Paper.pdf.

J. Soria-Comas, J. Domingo-Ferrer, D. Sánchez, and D. Megías. Individual differential privacy: A utility-preserving formulation of differential privacy guarantees. *IEEE Transactions on Information Forensics and Security*, 12(6), 2017.

T. Steinke and J. R. Ullman. Tight lower bounds for differentially private selection. In C. Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 552–563. IEEE Computer Society, 2017. doi: 10.1109/FOCS.2017.57. URL https://doi.org/10.1109/FOCS.2017.57.

U. S. Census Bureau. On the map: Longitudinal employer-household dynamics. https://lehd.ces.census.gov/applications/help/onthemap.html#!confidentiality_protection.

S. Vadhan. *The Complexity of Differential Privacy*, pages 347–450. Springer International Publishing, 2017.

Appendix A. Proofs from Section 3

Lemma A.1. *Let k be a positive integer (e.g., the group size parameter in Group IDP) and suppose the true dataset D_{act} has $\geq k$ records. The k-local sensitivity of $q_{\phi,b}$ with respect to D_{act} is 0 whenever $b < 0$, $b \geq n$ (number of records in D_{act}), or ϕ is always true or always false. Otherwise:*

$$\Lambda^s{}_k(q_{\phi,b}, D_{act}) = \begin{cases} 0 & when \left|\{r \in D_{act} : \phi(r) = True\}\right| > b + k \\ 0 & when \left|\{r \in D_{act} : \phi(r) = True\}\right| \leq b - k \\ 1 & otherwise \end{cases}$$

Proof of 3.3. The case when $b < 0$ or $b \geq n$ or ϕ is always true or is always false is trivial. So for the rest of the proof, we assume that none of those hold.

Let $\mathcal{N}_k$ be the set of records that can be obtained from D_{act} by modifying at most k records.

Case 1: If $\left|\{\mathbf{r} \in \mathbf{D}_{\text{act}} : \phi(\mathbf{r}) = \mathbf{True}\}\right| > \mathbf{b} + \mathbf{k}$ then changing up to k records of D_{act} can decrease the count by at most k (i.e., by taking up to k records that satisfy the predicate and changing them to some value that does not) and so for all $D^* \in \mathcal{N}_k$, $\left|\{r \in D^* : \phi(r) = True\}\right| > b$ and so $q_{\phi,b}$ would return the same answer for D_{act} and for each dataset in $\mathcal{N}_k$. Thus in this case, the k-local sensitivity is 0.

Case 2: If $\left|\{\mathbf{r} \in \mathbf{D}_{\text{act}} : \phi(\mathbf{r}) = \mathbf{True}\}\right| \leq \mathbf{b} - \mathbf{k}$ then changing up to k records of D_{act} can *increase* the count by at most k (i.e., by taking up to k records that do not satisfy ϕ and changing them to a value that does). So, for all $D^* \in \mathcal{N}_k$, $\left|\{r \in D^* : \phi(r) = True\}\right| \leq b$ and so $q_{\phi,b}$ would return the same answer for D_{act} and for each dataset in $\mathcal{N}_k$. Thus in this case, the k-local sensitivity is also 0.

Case 3: If $\mathbf{b} + \mathbf{k} \geq \left|\{\mathbf{r} \in \mathbf{D}_{\text{act}} : \phi(\mathbf{r}) = \mathbf{True}\}\right| > \mathbf{b}$. Here we have two sub-cases:

- If $\left|\{r \in D_{\text{act}} : \phi(r) = True\}\right| \geq k$ then one can change k records that satisfy ϕ to a value that does not to get a $D^* \in \mathcal{N}_k$ for which $\left|\{r \in D^* : \phi(r) = True\}\right| \leq b$ (which is a decrease from the upper bound $b + k$ that defines Case 3). Thus $q_{\phi,b}(D_{\text{act}}) = 1$ while $q_{\phi,b}(D^*) = 0$ and thus the k-local sensitivity would be 1.
- If $\left|\{r \in D_{\text{act}} : \phi(r) = True\}\right| < k$ then, one can modify all the records that satisfy ϕ to values that do not to get a $D^* \in \mathcal{N}_k$ for which $\left|\{r \in D^* : \phi(r) = True\}\right| = 0 \leq b$ (recall that the situation where b is negative has already been dealt with). Thus $q_{\phi,b}(D_{\text{act}}) = 1$ while $q_{\phi,b}(D^*) = 0$ and thus the k-local sensitivity would be 1.

Case 4: If $\mathbf{b} \geq \left|\{\mathbf{r} \in \mathbf{D}_{\text{act}} : \phi(\mathbf{r}) = \mathbf{True}\}\right| > \mathbf{b} - \mathbf{k}$. Here again we have two cases:

- If $\left|\{r \in D_{\text{act}} : \phi(r) = True\}\right| \leq n - k$ then there are at least k records not satisfying ϕ, and so k of them can be modified to values that do satisfy ϕ to get a $D^* \in \mathcal{N}_k$. This will increase the count by k and so we will have $\left|\{r \in D^* : \phi(r) = True\}\right| > b$. Thus $q_{\phi,b}(D_{\text{act}}) = 0$ while $q_{\phi,b}(D^*) = 1$ and thus the k-local sensitivity would be 1.

- If $\left|\{r \in D_{\text{act}} : \phi(r) = True\}\right| > n - k$ then there are fewer then k records that don't satisfy ϕ. If we modify all of them to have values that do satisfy ϕ then we get a $D^* \in \mathcal{N}_k$ such that $\left|\{r \in D^* : \phi(r) = True\}\right| = n > b$ (recall the situation where $b \geq n$ has already been dealt with). Thus $q_{\phi,b}(D_{\text{act}}) = 0$ while $q_{\phi,b}(D^*) = 1$ and thus the k-local sensitivity would be 1.

□

Lemma A.2. *Let D_{act} be a dataset with n records (where n is publicly known). Let $\mathbf{A}_i$ be an ordered attribute and $[u, v)$ be a range that does not contain the entire domain of $\mathbf{A}_i$. Let b be an integer threshold such that $1 \leq b \leq n-1$. Let M be the k-Laplace mechanism for answering the threshold range query $q_{\mathbf{A}_i \in [u,v),b}$. If the output ω of $M(D_{act})$ is released, then the following can be learned about D_{act}:*

- *If ω is detected as a noisy output then the quantity $\left|\{r \in D_{act} : u \leq r[i] < v\}\right|$ is $\geq b-k+1$ and is also $\leq b+k$. In other words, we get an upper and lower bound on the number of people in D_{act} whose value for $\mathbf{A}_i$ is in the range $[u, v)$.*
- *If ω is detected as non-noisy and $\omega = 1$ then $\left|\{r \in D_{act} : u \leq r[i] < v\}\right| > b + k$.*
- *If ω is detected as non-noisy and $\omega = 0$ then $\left|\{r \in D_{act} : u \leq r[i] < v\}\right| \leq b - k$*

Proof of 3.4. Since the query $q_{\mathbf{A}_i \in [u,v),b}$ has k-local sensitivity either 0 or 1, when ω is detected as noisy it means that the k-local sensitivity is 1. By Lemma 3.3, the k-local sensitivity is 1 only when the number of people in the range is $\leq b - k$ and $> b - k$. Since counts of people, b, and k are all integers, the condition $> b - k$ is the same as $\geq b - k + 1$. Hence the first item follows.

When the query answer ω is detected as non-noisy and $\omega = 1$ then we learn that the number of people in the range $[u, v)$ is $> b$ (since we know we are getting the true answer). However, this means the k-local sensitivity is 0 and we also know, by Lemma 3.3, that this only happens when the count of people in the range is $> b + k$ or $\leq b - k$. Combined with the knowledge that it is $> b$, we have that the count of people in this range is $> b + k$. This proves the second item.

Similarly, when the query answer ω is detected as non-noisy and $\omega = 0$ then we learn that the number of people in the range $[u, v)$ is $\leq b$ (since we know we are getting the true answer). However, this means the k-local sensitivity is 0 and we also know, by Lemma 3.3, that this only happens when the count of people in the range is $> b + k$ or $\leq b - k$. Combined with the knowledge that it is $\leq b$, we have that the count of people in this range is $\leq b - k$. This proves the third item.

□

Lemma A.3. *Given a predicate ϕ, if for some integer $b^{\uparrow}$ we have (1) the k-local sensitivity of $q_{\phi,b^{\uparrow}}$ with respect to D_{act} is 0 and (2) the k-local sensitivity of $q_{\phi,(b^{\uparrow}-1)}$ is 1, then*

- *The count of people in D_{act} whose records satisfy ϕ is $b^{\uparrow} - k$.*
- *The k-Laplace mechanism $q_{\phi,b}$ will return the non-noisy answer 0 for all $b \geq b^{\uparrow}$*

Furthermore, if for some integer $b^{\downarrow}$ we have (1) the k-local sensitivity of $q_{\phi,b^{\downarrow}}$ with respect to D_{act} is 0 and (2) The k-local sensitivity of $q_{\phi,(b^{\downarrow}+1)}$ is 1, then

- *The count of people in D_{act} whose records satisfy ϕ is $b^{\downarrow} + k + 1$.*
- *The k-Laplace mechanism for $q_{\phi,b}$ will return the non-noisy answer 1 for all $b \leq b^{\downarrow}$*

Proof of 3.6. First, by Lemma 3.3, the k-local sensitivity with respect to D_{act} changes from 0 to 1 when replacing $b^{\uparrow}$ with $b^{\uparrow} - 1$ only when $\left|\{r \in D_{\text{act}} : \phi(r) = True\}\right| = b^{\uparrow} - k$ (since the other condition for having 0 sensitivity remains unchanged as the threshold is decreased). Thus for any $b \geq b^{\uparrow}$, the sensitivity remains at 0 and the true answer to the query is also 0. This proves the first part.

For the second part, by Lemma 3.3, the k-local sensitivity with respect to D_{act} changes from 0 to 1 when replacing $b^{\downarrow}$ with $b^{\downarrow} + 1$ only when $\left|\{r \in D_{\text{act}} : \phi(r) = True\}\right| = b^{\downarrow} + k + 1$ (since the other condition for having 0 sensitivity remains unchanged as the threshold increases). Thus for any $b \leq b^{\downarrow}$ the sensitivity remains at 0 and the true query answer is 1. This proves the second part.

□

Appendix B. Proofs from Section 4

Lemma B.1. *Given a true dataset D_{act}, the bootstrap sensitivity of q_ϕ with respect to D_{act} is:*

$$\Lambda^s{}_B(q_\phi, D_{act}) = \begin{cases} 0 & \text{if all } r \in D_{act} \text{ satisfy } \phi \\ 0 & \text{if no } r \in D_{act} \text{ satisfies } \phi \\ 1 & \text{otherwise} \end{cases}$$

Proof of 4.5. Let D_1, D_2 be bootstrap neighbors conditioned on D_{act}. This means that any record in D_1 and D_2 also appears in D_{act}. Hence if all records in D_{act} give the same answer for ϕ (i.e., all records satisfy it or all do not) then $q_\phi(D_1) = q_\phi(D_2)$ and so the bootstrap sensitivity is 0.

If there is some record $r_1 \in D_{\text{act}}$ for which $\phi(r_1)$ =True and a $r_2 \in D_{\text{act}}$ for which $\phi(r_2)$ =False, then $D_1 = \{r_1\}$ and $D_2 = \{r_2\}$ are bootstrap neighbors conditioned on D_{act} and $q_\phi(D_1) - q_\phi(D_2) = 1$, hence the bootstrap sensitivity is 1.

□

Appendix C. Proofs from Section 5

Given an integer $m > 1$ and any $\epsilon > 0$ and a noise scale multiplier $\alpha \geq 0$, define the following random variables:

(1) $z_1, \ldots, z_m$, where each $z_i = \mu + \text{Laplace}(\alpha m/\epsilon)$ for some unknown number μ (the private value that gets noised).
(2) $z_1^*, \ldots, z_m^*$ where each z^* =Laplace(1)

Furthermore, define:

$$\psi_m = \frac{\epsilon^2}{m^2}\left[\frac{1}{m-1}\sum_{i=1}^{m}\left(z_i - \frac{z_1+z_m}{m}\right)^2\right]$$
$$\psi_m^* = \frac{1}{m-1}\sum_{i=1}^{m}\left(z_i^* - \frac{z_1^*+z_m^*}{m}\right)^2$$

Then the distribution of ψ_m is the same as the distribution of $\alpha^2\psi_m^*$ (in particular, it doesn't depend on ϵ or the private value μ). The expected value of ψ_m^* (resp., ψ_m) is 2 (resp., $2\alpha^2$) and ψ_m^* converges to 2 with probability 1 as $m \to \infty$ (hence ψ_m converges to $2\alpha^2$).

Proof of Table 5.1. We first note that $\frac{\epsilon}{m}(z_i - \mu)$ is a Laplace(α) random variable (because scaling it by ϵ/m is the same as multiplying the scale parameter by ϵ/m) so it has the same distribution as αz_i^*. Hence

$$\frac{\epsilon^2}{m^2}\left[\frac{1}{m-1}\sum_{i=1}^{m}\left(z_i - \frac{z_1+z_m}{m}\right)^2\right]$$
$$= \frac{\epsilon^2}{m^2}\left[\frac{1}{m-1}\sum_{i=1}^{m}\left(z_i - \mu + \mu - \frac{z_1+z_m}{m}\right)^2\right]$$
$$= \frac{\epsilon^2}{m^2}\left[\frac{1}{m-1}\sum_{i=1}^{m}\left((z_i - \mu) - \frac{(z_1-\mu)+(z_m-\mu)}{m}\right)^2\right]$$
$$= \left[\frac{1}{m-1}\sum_{i=1}^{m}\left(\frac{\epsilon}{m}(z_i - \mu) - \frac{\frac{\epsilon}{m}(z_1-\mu)+\frac{\epsilon}{m}(z_m-\mu)}{m}\right)^2\right]$$

and so has the same distribution as

$$= \left[\frac{1}{m-1}\sum_{i=1}^{m}\left(\alpha z_i^* - \frac{\alpha z_1^*+\alpha z_m^*}{m}\right)^2\right]$$
$$= \alpha^2\left[\frac{1}{m-1}\sum_{i=1}^{m}\left(z_i^* - \frac{z_1^*+z_m^*}{m}\right)^2\right]$$

and so ψ_m has the same distribution as $\alpha^2\psi_m^*$.

Now, the formula for ψ_m^* is known as the sample variance of a sequence of iid random variables and is known to be an unbiased estimate of their variance. Since the variance of Laplace(1) is 2, the expected value of ψ_m^* is 2 and the expected value of ψ_m is $2\alpha^2$. By the law of large numbers, the convergence happens with probability 1.

□

Lemma C.1. *The empirical neighbors definitions allow the release of $h(D_{act})$ for any ϵ parameter. In particular, if $h(D_{act}) = NPairs(D_{act})$ then Group IDP allows D_{act} to be revealed whenever $|D_{act}| > 1$ and BDP allows the distinct set of records to be revealed.*

On the other hand, if $h(D_{act}) = \emptyset$ and $\bigcup_D NPairs(D)$ is the set of all pairs of datasets that differ on a record, then the empirical neighbors definition is equal to differential privacy.

Proof of 5.2. For any set of bits b, let M_b be the mechanism that ignores its input and simply outputs b. Consider the chooser function such that $Chooser(h(D_{\text{act}}))$ that returns M_b, where $b = h(D_{\text{act}})$. Clearly this satisfies empirical neighbors privacy for any privacy parameter ϵ and always reveals $h(D_{\text{act}})$.

If $h(D_{\text{act}}) = NPairs(D_{\text{act}})$, as is the case with IDP and BDP, then we can reason as follows. For the case of IDP and Group IDP, $NPairs(D_{\text{act}})$ consists of pairs (D_1, D_2) where either $D_1 = D_{\text{act}}$ or $D_2 = D_{\text{act}}$, so D_{act} is the dataset that appears in every pair. If $|D_{\text{act}}| > 1$ then there are at least 2 pairs and only D_{act} will appear in all of them, hence D_{act} is revealed.

In the case of BDP, if we take all of the rows of all of the datasets that appear in $NPairs(D_{\text{act}})$ and then apply the database distinct operator, we get the distinct rows in D_{act}.

Finally, if $h(D_{\text{act}}) = \emptyset$, then a mechanism M must be chosen without looking at the data, and so letting $\mathcal{N} = \bigcup_D NPairs(D)$ be the set of all pairs of databases that are neighbors for some dataset, the condition of the lemma is that $\mathcal{N}$ covers all pairs of neighbors (D_1, D_2) that differ on one record and so the only way to guarantee that the empirical neighbors constraints are always satisfied is to ensure they are satisfied for all $D_1, D_2 \in \mathcal{N}$ which is equivalent to differential privacy. □