Privacy Enhancing Technologies FS2025
Lecture 23-24-25 – Differential Privacy

Florian Tramèr

AGENDA

1. Randomized Response
2. Differential Privacy and its properties
3. The Laplace Mechanism

Recap

What have we seen so far? If we have a database D, and we reveal too many accurate statistics over D, then we can reconstruct essentially all of D.

The attacks we saw relied on one crucial principle: only a small number of possible databases were *consistent* with the revealed noisy statistics. That is, most databases can be ruled out with 100% confidence because they could *never* have produced some of the noisy statistics.

To get around this issue, we will introduce new privacy algorithms that are *probabilistic* in nature, and that output statistics that are consistent (with some probability) with *any* underlying database. This will lead us to the formalization of differential privacy. The remarkable thing we'll see is that such algorithms can still be *useful* and produce answers that are sufficiently accurate.

1 Randomized Response

We consider a similar but simplified setup compared to the last lecture, where a curator collects some sensitive bit x_i from a population of n individuals to form a database $D = \{x_1, \dots, x_n\}$.

For now, let's forget about the adversary that queries the database, and suppose that the individuals don't even trust the curator. E.g., the curator could be a teacher that wants to (privately) learn what fraction of students cheat on their exams. Students would likely be wary to answer honestly.

Let's consider the following strategy: instead of reporting their true bit x_i, each individual instead reports

$$y_i = \begin{cases} x_i & \text{with probability } 1/2 + \gamma \\ 1 - x_i & \text{with probability } 1/2 - \gamma\,, \end{cases}$$

for some fixed $\gamma \in [0, 1/2]$.

Notice that this "randomized response" strategy [War65] provides a form of *plausible deniability* for each individual. If you report $y_i = 1$, you might have cheated, but you might also have just flipped your response with probability $1/2 - \gamma$. Intuitively, there seems to be some privacy in this protocol. We'll see how to formally quantify this privacy later on.

For now, let's show that this protocol is still useful for the curator! Recall that we want to estimate the fraction of students that cheat on their exams. That is, we want to compute

$$p = \frac{1}{n} \sum_{i=1}^{n} x_i \, .$$

Since the students are using the randomized response strategy, the curator will observe a biased value:

$$\begin{aligned}
\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} y_i\right] &= \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}[y_i] \\
&= \frac{1}{n}\sum_{i=1}^{n} \left((1/2+\gamma)x_i + (1/2-\gamma)(1-x_i)\right) \\
&= \frac{1}{n}\sum_{i=1}^{n} \left(2\gamma x_i + (1/2-\gamma)\right) \\
&= 2\gamma p + 1/2 - \gamma \, .
\end{aligned}$$

So, the curator can get an unbiased estimate by measuring

$$\hat{p} = \frac{1}{2\gamma} \cdot \left(\frac{1}{n}\sum_{i=1}^{n} y_i - 1/2 + \gamma\right) ,$$

such that $\mathbb{E}[\hat{p}] = p$.

Since $\hat{p}$ is a sum of n independent random variables (each rescaled from a simple Bernoulli random variable), we can apply standard concentration inequalities (e.g., Chernoff) to show that, with high probability,[1]

$$|p - \hat{p}| \leq \tilde{O}\left(\frac{1}{\gamma\sqrt{n}}\right) .$$[2]

So, not only do we get plausible deniability for each student, but also an accurate estimate with an error that tends to 0 as the number of participants n grows. Specifically, if we want to estimate p with an additive error at most α, then we need $n = \tilde{O}\left(\frac{1}{\gamma^2\alpha^2}\right)$ participants.

2 Differential Privacy

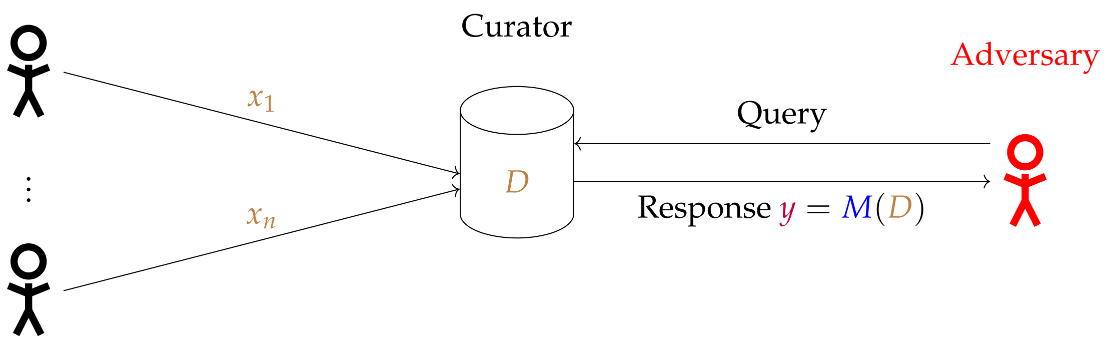
Let's now see how to extend this notion of plausible deniability to more general settings and formalize it. We'll go back to our curator setting (also called the *central model of differential privacy*), where a trusted curator collects a database of sensitive data and gives (noisy) query access to the adversary.

Instead of singe bits, we'll now let each individual's data be elements from some arbitrary set, $x \in \mathcal{X}$, which form the n rows of the database $D \in \mathcal{X}^n$. The adversary (also called the *analyst*) can make specific queries to this database (e.g., to compute the average across some

[1] See Section 4 for a formal treatment of this type of argument.

[2] The $\tilde{O}$ notation hides logarithmic factors.

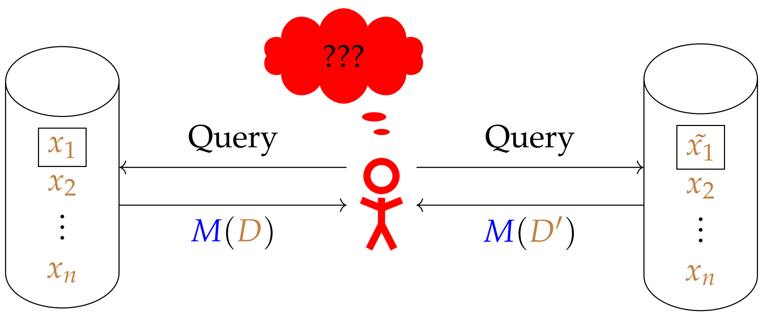
column). In response, the curator runs an algorithm[3] M over the database D to produce a response $y = M(D)$.



The "plausible deniability" property that we want the curator's algorithm M to achieve is informally the following:

> *"The adversary cannot distinguish the curator's response if one individual's data is changed"*

That is, from the perspective of each individual, the adversary learns roughly the same information whether the individual reports their true data x_i or a different vector $\tilde{x}_i \in \mathcal{X}$.



A nice way to think of this (still informal) notion of privacy is as a *promise* to each data holder [DR14]: giving your data to the curator will not adversely affect you if this data is used for a later study or analysis. Note that this does not mean the adversary gets to learn *nothing at all*. Rather, the promise is that anything the adversary can learn from the population of individuals your data is part of it could also have learned if you had decided not to reveal your data. So the adversary might learn information about the population as a whole (e.g., "smoking causes cancer"), but it won't be able to learn anything about any specific individual in the population (e.g., do *you* have cancer?).

Definition 1 (Neighboring databases)**.** Two databases D and D' are *neighboring*, denoted $D \sim D'$, if $|D| = |D'|$ and they differ in at most one row.[a]

[a] Neighboring databases are sometimes defined by *addition* or *removal* of a single row. The two definitions result in similar privacy definitions and algorithms, up to a constant factor.

[3] This algorithm is often called a *mechanism* in the differential privacy literature.

Definition 2 (Differential Privacy (DP) [DMNS06]). A randomized algorithm $M : \mathcal{X}^n \to \mathcal{Y}$ is ε-differentially private if, for all neighboring databases $D \sim D'$ and for all events $S \subseteq \mathcal{Y}$,

$$\Pr[M(D) \in S] \le e^{\varepsilon} \cdot \Pr[M(D') \in S] \,.$$

2.1 Interpreting the DP Definition

This is a somewhat technical and unintuitive definition. So let's discuss some points:

The parameter ε. The parameter ε quantifies the "privacy loss". So we want ε to be as small as possible. This is the opposite of a *security parameter* λ in cryptography, where a larger λ typically means a more secure system.

So how small should ε be? This is actually a tricky question. Coming from cryptography, one might be tempted to think of ε as $\mathsf{negl}(n)$. In this case, the definition would simply become:

$$M(D) \approx M(D') \,,$$

i.e., the distributions $M(D)$ and $M(D')$ are statistically indistinguishable. However, this is not possible (as you will show in next week's homework), so we need to settle for something weaker. We'll typically think of ε as a small constant, e.g., $e^{\varepsilon} = 2$.

Why does the definition use a multiplicative factor and not an additive one? The reason is primarily that we need to prevent the adversary from easily distinguishing D and D' for *rare* events. E.g., suppose there is some output that has probability 0 under D. Then, observing this output would give the adversary undeniable evidence that the database is *not* D. And so we'd need an additive factor to be very small to still get a good privacy guarantee (we'll talk about this more in the next lecture).

The guarantees of DP. DP is a property of the algorithm, not the output. We cannot say $y = M(D)$ is differentially private, but rather that the algorithm M that produced this output is differentially private. Some informal discussions of DP often make this mistake, e.g., by saying that some machine learning model is differentially private, when it is really the training algorithm that is private (or not).

Unless M is a trivial algorithm (e.g., one that outputs a constant independent of the input), we need randomness to achieve differential privacy.

Differential privacy (with appropriate ε) provably prevents the type of reconstruction attacks we saw in the last lecture (you'll show this formally in the homework). So differential privacy is a *stronger* notion of privacy than just asking for M to not be blatantly non-private.

DP is a *definition*. You might sometimes read things like "differential privacy hurts utility" or "differential privacy doesn't work in practice", which are technically type-errors. DP as a definition does not "work" (or "not work)", and it does not say anything about the "utility" of the algorithms that satisfy it. Existing algorithms that satisfy DP (e.g., Randomized Response, as we'll show) may result in more or less utility than other algorithms, and may therefore be more or less useful for a given application. This distinction is important: when deploying a privacy mechanism, first think about the *type of privacy* you want (e.g., is it cryptographic, differential privacy, or something else?). Then, find an algorithm that satisfies the definition of privacy you want, and hopefully it will also be efficient and useful enough for your application.

A note about subsets. Reasoning about probabilities over subsets $S \subseteq \mathcal{Y}$ is a bit awkward. If $\mathcal{Y}$ is discrete, the definition is equivalent to asking that $\Pr[M(D) = y] \leq e^{\varepsilon} \cdot \Pr[M(D') = y]$ for all $y \in \mathcal{Y}$. If $\mathcal{Y}$ is continuous, the definition is equivalent to asking that $f_D(y) \leq e^{\varepsilon} \cdot f_{D'}(y)$ for all $y \in \mathcal{Y}$, where $f_D, f_{D'}$ are the probability density functions of $M(D)$ and $M(D')$ respectively.

2.2 Neat Properties of Differential Privacy

Differential privacy has some neat properties that make it a very useful (and, in many ways, the "right") privacy notion. We discuss some of the most important ones here.

The first property is that differential privacy is *closed under post-processing*. This means that if we have a differentially private algorithm M, then any function of the output of M is also differentially private (as long as this function does not involve querying the database). In particular, this means that you don't need to worry about what the adversary might do with the output of a differentially private algorithm, or what background knowledge the adversary might have.

Theorem 1 (Post-Processing)**.** Let $M : \mathcal{X}^n \to \mathcal{Y}$ be ε-differentially private and let $f : \mathcal{Y} \to \mathcal{Z}$ be any (possibly randomized) function. Then $f \circ M$ is ε-differentially private.

Proof. We'll prove the statement when f is deterministic. Fix any neighboring databases D, D' and event $S \subseteq \mathcal{Z}$. Let $T = \{y \in \mathcal{Y} : f(y) \in S\}$. Then we have:

$$\Pr[f(M(D)) \in S] = \Pr[M(D) \in T] \leq e^{\varepsilon} \cdot \Pr[M(D') \in T] = \Pr[f(M(D')) \in S]$$

When f is randomized, we can view it as a distribution over deterministic functions, and apply a similar argument. $\square$

The second important property of differential privacy is that it *composes* nicely. Many old definitions of privacy, such as k-anonymity, have the annoying property that if you have two algorithms that are individually private, releasing them together might completely break privacy. This is annoying, because it means that you can never be sure that some data release is private without also knowing all the other ways in which your data is collected and "privatized". Differential privacy does not have this problem: the level of privacy degrades linearly with the number of times that the private data is used to answer queries.

Theorem 2 (Basic Composition)**.** Let $M = (M_1, M_2, \ldots, M_k)$ be a sequence of algorithms where M_i is ε_i-differentially private, and the algorithms can be chosen sequentially and adaptively (i.e., M_i can depend on the output of $M_1, \ldots, M_{i-1}$). Then M is $(\sum_{i=1}^{k} \varepsilon_i)$-differentially private.

Proof. We prove the result in the case where the mechanisms M_i are independent of one another. The proof for the general case where the mechanisms can be adaptive is a bit more tricky, but the same result holds.

Fix two neighboring databases $D \sim D'$ and an output $y = (y_1, \dots, y_k)$. Then we have:

$$\begin{aligned}
\frac{\Pr[M(D) = y]}{\Pr[M(D') = y]} &= \prod_{i=1}^{k} \frac{\Pr[M_i(D) = y_i \mid (M_1(D), \dots, M_{i-1}(D)) = (y_1, \dots, y_{i-1})]}{\Pr[M_i(D') = y_i \mid (M_1(D'), \dots, M_{i-1}(D')) = (y_1, \dots, y_{i-1})]} \\
&\leq \prod_{i=1}^{k} e^{\varepsilon_i} \\
&= \exp\left(\sum_{i=1}^{k} \varepsilon_i\right) .
\end{aligned}$$

□

Finally, this last nice property is a generalization of the notion of "neighboring databases" to more than just one row difference. Here also, privacy degrades in a simple way with the number of changes in a database we want the adversary to be oblivious to.

Theorem 3 (Group Privacy)**.** Let $M : \mathcal{X}^n \to \mathcal{Y}$ be ε-differentially private, and let D and D' be databases that differ in up to k rows. Then, for all events $S \subseteq \mathcal{Y}$,

$$\Pr[M(D) \in S] \leq e^{k\varepsilon} \cdot \Pr[M(D') \in S] .$$

Proof. The proof follows from a basic *hybrid argument*. Given D and D' that differ in k elements, we can construct a sequence of $k+1$ databases:

$$D = D_0, \quad D_1, \quad D_2, \quad \dots, \quad D_k = D'$$

such that each D_i and D_{i+1} differ in exactly one row. Then, we can apply the definition of differential privacy k times:

$$\begin{aligned}
\Pr[M(D) \in S] &= \Pr[M(D_0) \in S] \\
&\leq e^{\varepsilon} \cdot \Pr[M(D_1) \in S] \\
&\leq e^{2\varepsilon} \cdot \Pr[M(D_2) \in S] \\
&\;\;\vdots \\
&\leq e^{k\varepsilon} \cdot \Pr[M(D') \in S] .
\end{aligned}$$

□

2.3 Randomized Response Is (Locally) Differentially Private

We now have the mathematical toolkit to quantify the amount of privacy provided by the randomized response algorithm (this only took about 40 years...).

We will show that the mechanism that outputs all the randomized responses $M(x_1, \dots, x_n) = (y_1, \dots, y_n)$ is differentially private. Any statistic computed on this database (such as our estimate $\hat{p}$) is then also automatically differentially private, thanks to the neat post-processing property of differential privacy. Consider some set of randomized responses $\vec{b} \in \{0,1\}^n$. Let D and D' be neighboring databases that differ only in the value of x_j. Then, we have that

$$\frac{\Pr[M(D) = \vec{b}]}{\Pr[M(D') = \vec{b}]} = \frac{\prod_{i=1}^{n} \Pr[y_i = b_i]}{\prod_{i=1}^{n} \Pr[y'_i = b_i]} = \frac{\Pr[y_j = b_j]}{\Pr[y'_j = b_j]} \leq \frac{1/2 + \gamma}{1/2 - \gamma}$$

For example, if $\gamma = 1/4$, the randomized response algorithm is ε-differentially private for $e^\varepsilon = \frac{3/4}{1/4}$, or $\varepsilon = \ln 3 \approx 1.1$.

More generally, if γ is small (say $\gamma \leq 1/4$), randomized response is ε-differentially private for $\varepsilon = O(\gamma)$.

The local model of DP. In fact, the randomized response algorithm provides a notion of privacy that is *stronger* than the notion of privacy we considered in our curator model. Indeed, here it is not any specific query over the data that is differentially private, but the entire database itself. Thus, even if the curator is untrusted, each person that provides their data is individually protected with 1.1-differential privacy! This is also called the *local model* of differential privacy (as opposed to the *central model* we considered before).

Note that if an algorithm is private in the local model, it is also private in the central model, but the converse need not hold. In fact, we will now see that in the central model we can get much stronger privacy guarantees (or, equivalently, much higher utility for a same level of privacy) than in the local model.

3 The Laplace Mechanism

Recall that to achieve DP, we need the output distributions of our algorithm on two neighboring databases to be close to each other. The general idea of many DP algorithms is to first compute the true answer of the function in question, and then add noise to *hide* which database was used. For this, we need a sense of how much the output of the function can change when a single input changes, which we call the *sensitivity* of a function.

Definition 3 (ℓ_1-Sensitivity)**.** The ℓ_1-sensitivity of a function $f\colon \mathcal{X}^n \to \mathbb{R}^k$ is defined as

$$\Delta_1 = \max_{D \sim D'} \|f(D) - f(D')\|_1 = \max_{D,D'} \sum_{i=1}^{k} |f(D)_i - f(D')_i| \,,$$

where D and D' are neighboring databases.

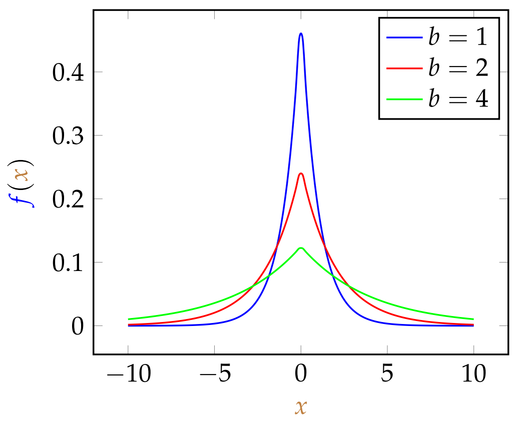
When f outputs a scalar, this is simply the maximum difference in the outputs of f on two neighboring databases. For example, if f is the sum of n bits, the sensitivity is 1.

Now, to achieve DP, intuitively we want to add noise to the output of f taken from a probability distribution whose density changes by at most e^ε when the input changes by Δ_1. There is a distribution that exactly satisfies this property: the Laplace distribution.

Definition 4 (Laplace Distribution)**.** The Laplace distribution $\texttt{Laplace}(\mu, b)$ with location μ and scale $b > 0$ has density

$$f(x \mid \mu, b) = \frac{1}{2b} e^{-\frac{|x-\mu|}{b}}, \quad x \in \mathbb{R}\,.$$

This distribution has mean μ and variance $2b^2$. The distribution is plotted below for a few values of b and $\mu = 0$.



As a point of comparison, the Gaussian distribution (which we'll discuss in the next lecture) has density $\propto e^{-x^2}$ instead of $e^{-|x|}$, and thus decays much faster than the Laplace distribution.

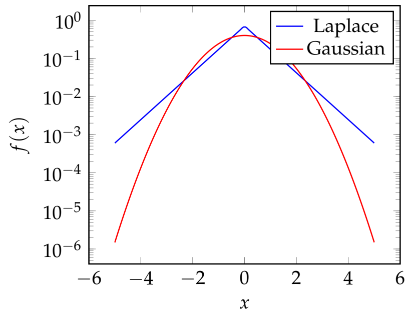


Figure 1: Laplace and Gaussian of variance 1. The Gaussian decays much faster.

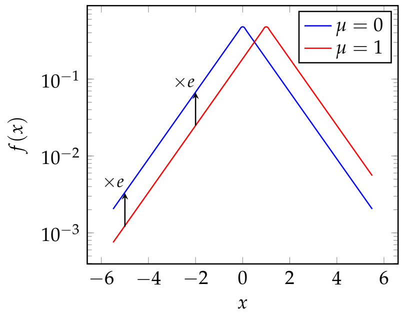


Figure 2: Laplace distributions shifted by 1 stay within a factor of e ($b = 1$).

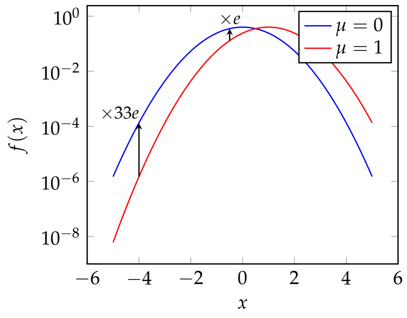


Figure 3: Gaussians shifted by 1 deviate by an increasing factor ($\sigma = 1$).

We are now ready to introduce the Laplace mechanism, which adds noise from a Laplace distribution scaled to the sensitivity of the function:

Definition 5 (Laplace Mechanism [DMNS06]). Let $f\colon \mathcal{X}^n \to \mathbb{R}^k$ be a function with ℓ_1-sensitivity Δ_1, and let $\varepsilon > 0$. The Laplace mechanism is defined as

$$M(D) = f(D) + (Y_1, \ldots, Y_k)$$

where the Y_i are i.i.d. samples from the $\texttt{Laplace}(0, \Delta_1/\varepsilon)$ distribution.

We can now prove that the Laplace mechanism is differentially private.

Theorem 4 ([DMNS06]). The Laplace mechanism is ε-DP.

Proof. Let D and D' be neighboring databases. Let $p_D(y)$ and $p_{D'}(y)$ be the probability density functions of the mechanism outputs $M(D)$ and $M(D')$ respectively, evaluated at an arbitrary point $y \in \mathbb{R}^k$. Let $b = \Delta_1/\varepsilon$. Then,

$$\begin{aligned}
\frac{p_D(y)}{p_{D'}(y)} &= \frac{\prod_{i=1}^{k} \frac{1}{2b} \exp\left(-\frac{|y_i - f(D)_i|}{b}\right)}{\prod_{i=1}^{k} \frac{1}{2b} \exp\left(-\frac{|y_i - f(D')_i|}{b}\right)} \\
&= \prod_{i=1}^{k} \exp\left(\frac{|y_i - f(D')_i| - |y_i - f(D)_i|}{b}\right) \\
&\leq \prod_{i=1}^{k} \exp\left(\frac{|f(D')_i - f(D)_i|}{b}\right) && \text{by reverse triangle inequality} \\
&= \exp\left(\frac{\sum_{i=1}^{k} |f(D')_i - f(D)_i|}{b}\right) \\
&= \exp\left(\frac{\|f(D') - f(D)\|_1}{b}\right) \\
&\leq \exp\left(\frac{\Delta_1}{b}\right) = \exp(\varepsilon)\ . && \text{by def. of sensitivity}
\end{aligned}$$

□

3.1 Answering Counting Queries With the Laplace Mechanism

So we now have two differentially private algorithms: Randomized Response and the Laplace mechanism. Recall that for computing a mean of n private bits, the randomized response algorithm we analyzed achieves error $\tilde{O}(1/\varepsilon\sqrt{n})$.

To achieve a similar privacy level with the Laplace mechanism, we need to add Laplace noise with scale parameter $b = \Delta_1/\varepsilon = O(1/\varepsilon n)$ (the sensitivity of the mean is $1/n$). The variance of the noise is $2b^2 = O(1/\varepsilon^2 n^2)$, and so the error in our estimate is $\tilde{O}(1/\varepsilon n)$ with high probability (see Section 4). This is *quadratically smaller* than the error of the randomized response algorithm! This is the power of the central model of DP over the local model.

Multiple queries. For now, we asked a single query of our database. What if we wanted to answer many queries? Our multi-dimensional definition of the Laplace mechanism makes this easy to reason about. Suppose we had k counting functions $f = (f_1, \ldots, f_k)$ executed on the same dataset, where each function f_i is of the form $f_i(D) = \frac{1}{n}\sum_{i=1}^{n} \phi_i(x_i)$ for some predicate $\phi_i : \mathcal{X} \to \{0,1\}$.[4] We would then output the vector $f(D) + Y$, where the Y_i's are i.i.d. Laplace random variables.

What's the sensitivity of the function f? Each individual counting query f_i has sensitivity $1/n$, but they are all computed on the same dataset. So swapping a single individual might affect the result of all queries. More formally, for any two neighboring databases D and D' we have $\|f(D) - f(D')\|_1 = \sum_{i=1}^{k} |f_i(D) - f_i(D')| \leq k/n$. So we can add noise $Y_i \sim \texttt{Laplace}(0, k/\varepsilon n)$ to each coordinate. A standard concentration bound for the Laplace distribution shows that, with high probability, the maximum noise across all coordinates is $\tilde{O}(\frac{k \log k}{\varepsilon n}) = \tilde{O}(k/\varepsilon n)$. See Section 4 for the proof.

[4]The notation here is somewhat different compared to the Dinur-Nissim setting: we don't distinguish between identifiers and private bits here, and just assume the adversary wants to count the number of individuals that satisfy some predicate.

Note that we would get the exact same result if we answer each query one after the other and compute the total privacy budget using the basic composition theorem. The advantage of this latter approach is that it holds even when the queries are chosen *adaptively*, i.e., the output of one query is used to inform the next queries.

4 Bounding the Error of DP Algorithms

Our analysis of the error bounds of various mechanisms glossed over some details of how to apply concentration bounds. Let's formalize these things here.

Randomized Response. In Section 1, we argued that the error of the randomized response algorithm is $\tilde{O}(1/\sqrt{n})$ with high probability, and hinted at a Chernoff/Hoeffding bound argument (see the background sheet for a description on these bounds).

Without loss of generality, assume that all responses are zero, i.e., $x_i = 0, \forall i$ (it's easy to see that this is the worst case, as individual errors cannot "cancel out"). The error of the randomized response algorithm is then simply $\hat{p} = \frac{2}{n}\sum_{i=1}^{n} y_i - \frac{1}{2} = \sum_{i=1}^{n} \frac{2}{n}(y_i - \frac{1}{4})$. This is the sum of n i.i.d. Bernoulli random variables taking values in $[-1/2n, 3/2n]$. Also, recall that we showed that $\mathbb{E}[\hat{p}] = p = 0$. So we have that:

$$\begin{aligned}\Pr[|\hat{p}| \geq t] &\leq 2\exp\left(-\frac{2t^2}{n \cdot (2/n)^2}\right) \\ &= 2\exp\left(-nt^2/2\right) .\end{aligned}$$

We wanted to show that errors larger than $\tilde{O}(1/\sqrt{n})$ are unlikely. And indeed, we have that:

$$\Pr[|\hat{p}| \geq \sqrt{\log n/n}] \leq 2e^{-\log n/2} = 2/\sqrt{n} \, ,$$

which drops off exponentially fast in n.

Laplace noise for one query. In Section 3.1, we said that if we add Laplace noise of scale $b = O(1/\varepsilon n)$ to a counting query, then the error is $\tilde{O}(1/\varepsilon n)$ with high probability. This follows directly from the following tail bound for the Laplace distribution:

If $Z \sim \texttt{Laplace}(0, b)$, then for every $t > 0$: $\Pr(|Z| \geq t \cdot b) \leq \exp(-t)$.

So the error exceeds $\log n \cdot b = \tilde{O}(1/\varepsilon n)$ with probability at most $e^{-\log n} = 1/n = o(1)$.

Laplace noise for multiple queries. In Section 3.1, we said that if we answer k counting queries with the Laplace mechanism, the maximum error is only a factor $O(\log k)$ larger.

We can derive this from a union bound. Let $Z_1, \ldots, Z_k$ be the Laplace noise with scale $b = O(k/\varepsilon n)$ added for the k queries. Combining the above tail bound with a union bound, we have that:

$$\Pr[\max_i |Z_i| \geq t \cdot b] \leq k \exp(-t) \, .$$

Setting $t = \log(n\,k)$, we get that:

$$\Pr[\max_i |Z_i| \geq \tilde{O}(k/\varepsilon n)] \leq k \exp(-\log nk) = 1/n = o(1) \, .$$

References

[DMNS06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3*, pages 265–284. Springer, 2006.

[DR14] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[War65] Stanley L Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American statistical association*, 60(309):63–69, 1965.