CS 325: Private Data Analysis **Fall 2022**

Lecture 11: Releasing Multiple Linear Queries Privately I

Lecturer: Di Wang *Scribes: Di Wang*

Note: *LaTeX template courtesy of UC Berkeley EECS dept.*

Disclaimer: *These notes have not been subjected to the usual scrutiny reserved for formal publications. They may be distributed outside this class only with the permission of the Instructor.*

So far weve talked about linear query release as a natural and important problem that captures many of the things a data collector would want to do with a sensitive dataset. However, we havent seen interesting algorithms for this problem except for adding Laplace or Gaussian noise calibrated to sensitivity. In the next couple of lectures were going to do more of a deep dive on this problem and see a variety of exciting algorithms for query release. In this lecture well focus on a natural framework for query-release algorithms based on two ideas for reducing the error factorization and projection. These two tools are often combined in a particular way, leading to what is called the matrix mechanism, which itself is the starting point for the US Census Bureaus algorithm for the 2020 Decennial Census. We refer the methods provided by [4, 1, 2, 3, 5]

11.1 Query Release Recap and Histograms

Were going to start by recapping the problem of linear query release and then introduce a different viewpoint on the problem that will be useful for developing the framework. Recall that we start with a data universe $\mathcal{D} = \{s_1, s_2, \cdots, s_{|\mathcal{D}|}\}$ and a dataset $D \in \mathcal{D}^n$. **A linear query** is a function $f : \mathcal{D} \mapsto [0, 1]$ and we overload the notation to also refer to the function applied to a dataset: letting $D \in \mathcal{D}^n$, $f(D) = \frac{1}{n}\sum_{i=1}^n f(x_i)$. The core question of answering linear queries is the following: suppose we have a set k of linear queries $f_1, \cdots, f_k$. How large does a dataset have to be in order to answer all of them up to accuracy α under differential privacy?

Recall that our baseline algorithm for query release is the Gaussian mechanism, where we add noise from a Gaussian distribution with noise scaled to the global ℓ_2-norm sensitivity. We define the function

$$F(D) = (f_1(D), \cdots, f_k(D))$$

and define the Gaussian mechanism

$$M(D) = F(D) + \mathcal{N}(0, \sigma^2 I_k),$$

where the noise variance $\sigma = \sigma_{\epsilon,\delta}\Delta_2$ with

$$\Delta = \max_{D\sim D'} \|F(D) - F(D')\|_2 \leq \frac{2\sqrt{k}}{n}, \sigma_{\epsilon,\delta} = O(\frac{\sqrt{\log 1/\delta}}{\epsilon}).$$

We will suppress $\sigma_{\epsilon,\delta}$ because the dependence on the privacy parameters themselves is not particularly interesting for this lecture. For this lecture, were also going to make a small change in how we measure the error, which will again make things a little easier going forward. Instead of asking for the error to be small for every query, well ask for it to be small for the average query. Specifically, given answers $a = (a_1, \cdots, a_k)$ we want to have

$$\frac{1}{\sqrt{k}}\|F(D) - a\| = (\frac{1}{k}\sum_{i=1}(f_i(D) - a_i)^2)^{\frac{1}{2}} \leq \alpha.$$

for as small an error α as possible. Note that dividing by $\frac{1}{\sqrt{k}}$ allows us to put the error in the same units as the maximum error over all queries. In other words, if the error for each query is at most α then the error is also at most α.

For the Gaussian mechanism, we can say that the error will be

$$\mathbb{E}(\frac{1}{\sqrt{k}}\|M(D) - F(D)\|_2) = O(\sigma_{\epsilon,\delta}\frac{\sqrt{k}}{n}). \tag{11.1}$$

This error rate of $O(\frac{\sqrt{k}}{n})$ is what we think of as a baseline for any linear query release problem, and is what were going to try to improve in this and the next several lectures.

Note that we are focusing on the Gaussian mechanism and (ϵ, δ)-differential privacy because it gives better accuracy, lets us work with the nicer Gaussian distribution, and avoid having two parallel explanations of the same mechanism. Everything we are saying can be done with the Laplace mechanism and $(\epsilon, 0)$-differential privacy without any significant conceptual differences.

OK, now that we did all that somewhat painful recapping, lets do it all again in different notation! The reason we want to switch notation is that it will allow us to express linear queries in the language ofsurprise, surpriselinear algebra. To do so, well start by defining the histogram representation of a dataset. Given a dataset $D = \{x_1, \cdots, x_n\} \in \mathcal{D}^n$, we can represent the dataset as a normalized histogram that tells us how many times each element $x \in \mathcal{D}$ appears in the dataset. Specifically, $h_D \in \mathbb{R}^{|\mathcal{D}|}$ where

$$(h_D)_x = \frac{\{\text{numbers of } j \text{ such that } x_j = x\}}{n}. \tag{11.2}$$

For example, if $\mathcal{D} = \{1, 2, 3\}$ and $D = (1, 3, 3, 2, 3)$ then $h_D = (\frac{1}{5}, \frac{1}{5}, \frac{3}{5})$. Note that we can convert back and forth between the dataset and the histogram, so they contain exactly the same information.

If we are going to design mechanisms in terms of the histogram, we need to understand what happens to the histogram when we change one element of the original dataset. In particular, this can change at most two buckets of the dataset by at most $1/n$ each. Thus we have that for every pair of neighboring datasets of size n

$$\|h_D - h_{D'}\|_1 \leq \frac{2}{n}. \tag{11.3}$$

The reason we like to use the histogram representation is that we can finally make sense of the term linear queriesthe reason they are called linear is because they are a linear function of the datasets histogram. Specifically, given a single query f we can rewrite

$$f(D) = v \cdot h_D, v = (f(s_1), \cdots, f(s_{|\mathcal{D}|})).$$

Then, given a set of queries $f_1, f_2, \cdots, f_k$ we can write

$$F(D) = Ah_D, \tag{11.4}$$

where $A \in [0, 1]^{k\times|\mathcal{D}|}$ and $A_{i,j} = f_i(s_j)$. Thus, the answers are a k column vector whose i-th row is the answer to the i-th query.

11.1.1 The Gaussian Mechanism in the Histogram Representation

In this notation we can write the Gaussian mechanism in a completely equivalent way as

$$M(D) = Ah_D + \mathcal{N}(0, \sigma^2 I_k). \tag{11.5}$$

Now we consider the ℓ_2-norm sensitivity of Ah_D:

$$\Delta = \max_{D\sim D'} \|Ah_D - Ah_{D'}\|_2 \leq \max_{h,h'\in\mathbb{R}^{|\mathcal{D}|}, \|h-h'\|_1\leq \frac{2}{n}} \|A(h-h')\|_2 = \frac{2}{n} \max_{v\in\mathbb{R}^{|\mathcal{D}|}, \|v\|_1\leq 1} \|Av\|_2 = \frac{2\|A\|_{1,2}}{n},$$

where $\|A\|_{1,2}$ is the largest ℓ_2-norm of any column of A. Given the above, we can set the variance of the noise to be $\sigma^2 = \sigma^2_{\epsilon,\delta}\frac{4}{n^2}\|A\|^2_{1,2}$. Thus for the error we have

$$\mathbb{E}(\frac{1}{\sqrt{k}}\|M(D) - F(D)\|_2) = O(\sigma_{\epsilon,\delta}\frac{\|A\|_{1,2}}{n}). \tag{11.6}$$

11.2 A Framework for Improving the Gaussian Mechanism

We're finally ready to see some ideas for improving the basic Gaussian mechanism. We're going to see three conceptually distinct ways of reducing the error, which can be mixed and matched with each other. The first two can both be viewed as taking advantage of different types of structure in the queries, and the latter two is a form of post-processing of the Gaussian mechanism that can actually improve the error of the Gaussian mechanism dramatically in some cases.

11.2.1 Factorization

The first idea is called factorization, because we factor the mechanism into a differentially private measurement phase followed by a reconstruction phase that is just a postprocessing of what we measured. To get a feel for the idea, lets consider two special cases. First, suppose we are asked to answer the exact same query k times, so suppose $f_1 = f_2 = \cdots = f_k$. Then, in our histogram representation, the query matrix will be something like:

$$A = \begin{bmatrix} 1,1,0,0,1 \\ 1,1,0,0,1 \\ 1,1,0,0,1 \\ 1,1,0,0,1 \end{bmatrix} \tag{11.7}$$

where each row is a copy of the same query. Notice that, as long as the query is not the all-zero function, then the largest ℓ_2 norm of any column will be exactly $\sqrt{k}$, so the Gaussian mechanism will add noise proportional to $\sqrt{k}/n$. Clearly this is wasteful, because were really only answering a single query with sensitivity $1/n$. In this example its clear what to douse the Gaussian mechanism to answer just the query f_1, obtain an answer a, and then output the vector of k answers (a, a, a, a). Since were just answering a single query, the error is proportional to $1/n$, which is what it should be.

Although this idea is simple enough, the general principle is that we are measuring a single query f_1 and then performing a linear reconstruction of the answers to the remaining queries. The set of queries we measure is described by a matrix $N \in \mathbb{R}^{l\times|\mathcal{D}|}$ and the reconstruction is described by another matrix $R \in \mathbb{R}^{k\times l}$. All we are actually using here to make this a viable approach is the fact that $RN = A$. As long as we have this, then our mechanism will have the form

$$M_{R,N} = R(Nh_D + Z) = Ah_D + RZ, Z \sim \mathcal{N}(0, \sigma^2 I_l). \tag{11.8}$$

In other words, we are just answering the queries A but with some alternative noise random variable RZ rather than Z. Note that the Gaussian mechanism itself corresponds to the trivial factorization where $N = A$ and $R = I$, so in that sense factorization can only help and never hurt.

For a given factorization $RN = A$, what is the error of the mechanism? To answer this question, we need to understand the quantity.

$$\mathbb{E}[\frac{1}{\sqrt{k}}\|M_{R,N} - Ah_D\|_2] = \mathbb{E}[\frac{1}{\sqrt{k}}\|RZ\|_2]. \tag{11.9}$$

Recall that Z is from a Gaussian distribution where each coordinate is independent and has variance $\sigma^2 = O(\sigma^2_{\epsilon,\delta}\frac{\|N\|^2_{1,2}}{n^2}))$. Since $Z \sim \mathcal{N}(0, \sigma^2 I_l)$, each row of $(RZ)_i \sim \mathcal{N}(0, \|r_i\|_2^2\sigma^2)$ where r_i is the i-th row of R. Thus we have

$$\mathbb{E}\|RZ\|_2 \leq (\mathbb{E}\|RZ\|_2^2)^{\frac{1}{2}} = (\mathbb{E}\sum_{i=1}^{k}\mathcal{N}(0, \|r_i\|_2^2\sigma^2))^{\frac{1}{2}} = \|R\|_F^2\sigma^2 = O(\sigma^2_{\epsilon,\delta}\frac{\|R\|_F^2\|N\|^2_{1,2}}{n^2})).$$

Lastly, notice that there can be many different pairs R, N such that $RN = A$, and of course we should chose the one that gives us the best error! Since we can describe the best error as being proportional to $\|R\|_F^2\|N\|_{1,2}^2$, we can just choose the factorization that minimizes this error! We can actually give a name to the quantity that represents the best error we can achieve via this frameworkthe factorization norm of A.

Definition 11.1 *Given a matrix $A \in \mathbb{R}^{k\times m}$, the factorization norm is defined as*

$$\gamma(A) = \min\{\frac{\|R\|_F^2\|N\|^2_{1,2}}{\sqrt{k}} : RN = A\}. \tag{11.10}$$

Theorem 11.2 *For every set of linear queries given by a matrix $A \in \mathbb{R}^{k\times|\mathcal{D}|}$, there is an (ϵ,δ)-differentially private mechanism M that answers these queries with error $O(\sigma_{\epsilon,\delta}\frac{\gamma(A)}{n})$.*

11.2.2 Approximation

The next avenue for improvement is to give up on the constraint that $RN = A$. Suppose we run the factorization mechanism with an approximate factorization where $RN = A + E$ for some small matrix E? In this more general setting we have

$$M_{R,N}(D) = R(Nh_D + Z) = Fh_D + Eh_D + RZ.$$

Thus the error will be

$$\mathbb{E}\frac{\|Eh_D + RZ\|_2}{\sqrt{k}} \leq \frac{\|E\|_{1,2}}{\sqrt{k}} + \frac{\sigma_{\epsilon,\delta}\|R\|_F^2\|N\|^2_{1,2}}{\sqrt{k}} \tag{11.11}$$

One (somewhat boring) example where this kind of approximation is useful is when the queries are approximately the same but not identical. For example, if the query matrix is

$$A = \begin{bmatrix} 1+\alpha, 1+\alpha, 0, 0, 1+\alpha \\ 1+\alpha, 1+\alpha, 0, 0, 1+\alpha \\ 1+\alpha, 1+\alpha, 0, 0, 1+\alpha \\ 1+\alpha, 1+\alpha, 0, 0, 1+\alpha \end{bmatrix} \tag{11.12}$$

As above, we can choose the factorization R, N to optimize this error term, trading of the two sources of error.

Definition 11.3 *Given a matrix $A \in \mathbb{R}^{k\times m}$, the α-approximate factorization norm is defined as*

$$\gamma_\alpha(A) = \min\{\frac{\|R\|_F^2\|N\|^2_{1,2}}{\sqrt{k}} : \|RN - A\|_{1,2} \leq \alpha\sqrt{k}\}. \tag{11.13}$$

For every set of linear queries given by a matrix $A \in \mathbb{R}^{k \times |\mathcal{D}|}$ and for every α, there is an (ϵ, δ)-differentially private mechanism M that answers these queries with error $O(\alpha + \sigma_{\epsilon,\delta} \frac{\gamma_\alpha(A)}{n})$.

11.2.3 Projection Mechanism

Now were going to look at a very different approach to improve the error of the mechanism, albeit one that can be combined seamlessly with the factorization approaches we just described. The approach can be described most easily with an example: For a simple, but again somewhat boring example, for any count query the true answer should lie in the interval [0, 1]. If we let a^* be the true answer and a be our noisy answer, then the error is $|a - a^*|$. However, what if $a < 0$, meaning a cannot possibly by the correct answer? Well, clearly we can only give a more accurate answer if we replace a with $\tilde{a} = \max\{a, 0\}$, and you can check that $|a^* - \tilde{a}|$ is never larger than $|a - a^*|$ and can be smaller. Similarly, we can also let $\bar{a} = \max\{\min\{a, 1\}, 0\}$ to capture the constraint that $0 \leq a^* \leq 1$.

Generally speaking, that is, by adding Gaussian or Laplacian noise, the our answer maybe inconsistent to some original constraints. In other words, we might obtain inconsistent answers that are not the exact answers we would obtain from any real dataset. We might not, in principle, care about having consistent answers, enforcing that the answers be consistent with the exact answers on some dataset can actually improve the error-in some cases quite dramatically. In the multiple linear queries, the consistent set should be all possible correct answers. Specifically

$$\mathcal{C} = \{a \in \mathbb{R}^k : \exists h \in \mathbb{R}^{|\mathcal{D}|} \text{ s.t. } \|h\|_1 = 1, Ah = a\} \tag{11.14}$$

Given that we know that the true answer $a^* = Ah_D$ is in the set $\mathcal{C}$, we can improve the error by projection into $\mathcal{C}$. That is, we define the projection operator

$$\Pi_{\mathcal{C}}(a) = \arg\min_{a' \in \mathcal{C}} \|a - a'\|_2. \tag{11.15}$$

and replace the answers a with $\tilde{a} = \Pi_{\mathcal{C}}(a)$. There are two really important facts about the projected answers. First: since $\mathcal{C}$ is convex and $a^* \in \mathcal{C}$, we will always have $\|\tilde{a} - a^*\|_2 \leq \|a - a^*\|_2$. Secondly, $\tilde{a}$ is (ϵ, δ)-DP due to the post-processing property of DP. In other words, if what we care about is minimizing ℓ_2 error, then projection is a cost free approach for improving the answers. It can never make the error larger and does not increase the privacy parameters in any way.

While the (approximate) factorization mechanism is a useful tool for exploiting structure in queries, some queries simply dont have structure to exploit, and there are many families of queries where the best factorization mechanism is to simply add Gaussian noise. In this section, well see for the first time that its possible to do much better than Gaussian noise for arbitrary linear queriesat when the dataset and data domain are not too largevia projection. More specifically, well see that projecting the answers to be consistent with some dataset can not only never increase the error, it can sometimes dramatically decrease the error! We will formalize the projection mechanism:

1. First calculate
$$a = Ah_D + \mathcal{N}(0, \sigma^2 I_k),$$
where the noise variance $\sigma = \sigma_{\epsilon,\delta}\Delta_2$ with
$$\Delta = \max_{D \sim D'} \|F(D) - F(D')\|_2 \leq \frac{2\sqrt{k}}{n}, \sigma_{\epsilon,\delta} = O(\frac{\sqrt{\log 1/\delta}}{\epsilon}).$$
2. Let $\tilde{a} = \Pi_{\mathcal{C}}(a)$.

Next we will focus on the error bound of $\tilde{a}$ to the true answer $a^* = Ah_D$. Let ℓ be the line through a^* and $\tilde{a}$, and let p be the projection of a onto ℓ. The key observation is that p lies on the ray from $\tilde{a}$ to infinity because otherwise p would be a point in $\mathcal{C}$ that is closer to a than $\tilde{a}$, which violates the definition of the projection.

Using this simple observation fact we can calculate

$$\|\tilde{a} - a^*\|_2^2 \leq \langle \tilde{a} - a^*, \tilde{a} - a^* \rangle \leq \langle \tilde{a} - a^*, p - a^* \rangle = \langle \tilde{a} - a^*, a - a^* \rangle \leq 2 \max_{v \in \mathcal{C}} |\langle v, Z \rangle|.$$

Thus, the expected error:

$$\mathbb{E} \frac{\|\tilde{a} - a^*\|_2}{\sqrt{k}} \leq \sqrt{\frac{2\mathbb{E} \max_{v \in \mathcal{C}} |\langle v, Z \rangle|}{k}}. \tag{11.16}$$

So the key quantity we need to analyze is now strange quantity $\mathbb{E} \max_{v \in \mathcal{C}} |\langle v, Z \rangle|$. A key observation is that is that $\mathcal{C}$ is the convex hull of the $|\mathcal{D}|$ columns of the matrix $A \in \mathbb{R}^{k \times |\mathcal{D}|}$, so these columns $c_1, \cdots, c_{|\mathcal{D}|} \in \mathbb{R}^k$ are the vertices of the set $\mathcal{C}$. Thus, we have

$$\mathbb{E} \max_{v \in \mathcal{C}} |\langle v, Z \rangle| \leq \mathbb{E} \max_{j \in [|\mathcal{D}|]} |\langle c_j, Z \rangle| = O(\sigma \sqrt{k} \log^{1/2} |\mathcal{D}|). \tag{11.17}$$

Thus in total we have

$$\mathbb{E} \frac{\|\tilde{a} - a^*\|_2}{\sqrt{k}} \leq O\left(\left(\frac{\sigma_{\epsilon,\delta} \log^{1/2} |\mathcal{D}|}{n}\right)^{\frac{1}{2}}\right).$$

Interpreting the Bound. Lets compare this bound to what we got for the standard Gaussian mechanism without projection, which gave error $O(\sigma_{\epsilon,\delta} \frac{\sqrt{k}}{n})$. On the downside, the standard Gaussian mechanism has error going to 0 at a rate of about $1/n$ whereas our analysis of the projection mechanism only has error going to 0 at a rate of about $1/\sqrt{n}$. So the new analysis only helps when n is relatively small.

However, suppose $n \ll k^{1/2}$. Then the Gaussian mechanism will give error that is $O(1)$, which means the noise in the answers is dramatically larger than the range of the answers, which is between -1 and 1 so we seemingly get nothing useful out of the Gaussian mechanism. However, after we project, as long as $n \gg \log^{1/2} |\mathcal{D}|$, we actually get answers with error going to 0! But wait, doesnt that mean that, as long as $n \gg \log^{1/2} |\mathcal{D}|$, we can answer an arbitrarily large number of queries with error independent of the number of queries, and going to 0 as n goes to ∞? Yes it does!

References

[1] Aditya Bhaskara, Daniel Dadush, Ravishankar Krishnaswamy, and Kunal Talwar. Unconditional differentially private mechanisms for linear queries. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 1269–1284, 2012.

[2] Alexander Edmonds, Aleksandar Nikolov, and Jonathan Ullman. The power of factorization meisms in local and central differential privacy. In *ACM Symposium on the Theory of Computing, STOC*, volume 20, pages 425–438, 2020.

[3] Moritz Hardt and Kunal Talwar. On the geometry of differential privacy. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 705–714, 2010.

[4] Chao Li, Michael Hay, Vibhor Rastogi, Gerome Miklau, and Andrew McGregor. Optimizing linear counting queries under differential privacy. In *Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 123–134, 2010.

[5] Aleksandar Nikolov, Kunal Talwar, and Li Zhang. The geometry of differential privacy: the small database and approximate cases. *SIAM Journal on Computing*, 45(2):575–616, 2016.