CSC2419:Algorithms and Complexity in Private Data Analysis Winter 2017

Lecture 6 — February 28, 2017

Aleksandar Nikolov *Scribe: Ian Mertz*

0 Recap

Definition 1. *The query relase problem is defined by a set of m linear queries $q_1 \dots q_m$ over the database $s \in \mathcal{X}^n$, where $q_i(x) = \frac{1}{n}\sum_{j\in[n]} q_i(x_j)$. The goal is to minimize n while still being able to publish $\mathcal{M}(x)$ for some f such that $|\mathcal{M}(x)_i - q_i(x)| \leq \alpha$ for all i, with a guarantee that $\mathcal{M}$ is ϵ-differentially private, or (ϵ, δ)-differentially private.*

While we are primarily interested in minimizing n (as a function of the size of the universe $|\mathcal{X}|$), we also make note of the runtime of our ϵ–DP algorithm. Oftentimes to guarantee a better n we take huge tradeoffs in our algorithm, to the point where on a moderately large universe the problem is intractable. So far we have two examples of this tradeoff:

- Laplace mechanism: $\mathcal{M}(x)_i = q_i(x) + \text{Lap}(\frac{1}{\epsilon n})$ for all i, where $\text{Lap}(\frac{1}{\epsilon n}) \propto \frac{\epsilon n}{2} e^{-|y|\epsilon n}$. In other words, we just add independent Laplace noise to every query. This is ϵ–DP with error α for $n = \Omega\left(\frac{m}{\alpha\epsilon}\right)$, and the algorithm takes time $O(nm)$ in total.
- SmallDB: use the exponential mechanism to generate synthetic databases of size roughly $\log|X|$, and then sample a database from among them that gives very similar answers to all queries q_i. This is ϵ–DP with error α for $n\Omega\left(\frac{\log m \log|\mathcal{X}|}{\alpha^3\epsilon}\right)$, and the algorithm takes time exponential in $\frac{\log m \log|\mathcal{X}|}{\alpha^2}$.

Is it possible to get n to be on the order of $\frac{\log m \log|\mathcal{X}|}{\alpha^{O(1)}\epsilon}$ without running in time which is superpolynomial in $|\mathcal{X}|$? It turns out in many cases we can't hope for better than $|\mathcal{X}|^{O(1)}$, but we can remove the dependance on m in the exponent, which provides a huge runtime improvement over SmallDB while getting a similar lower bound on n. The idea will be to "learn" the database from (noisy) answers to as few queries as possible, and use this new database to answer the remaining queries instead. We save the details of the learning algorithm for the end, as it uses fairly standard techniques that are less relevant to the differential privacy side of the overall algorithm for generating $\mathcal{M}(x)$.

1 Using the learning algorithm

For the purposes of this algorithm it helps to think of the database $x \in \mathcal{X}^n$ as a distribution over the universe $\mathcal{X}$ as follows: for all $r \in \mathcal{X}$ let $p_r = \frac{|\{i:r=x_i\}|}{n}$. From here we can see that $q_i(x) = \sum_{r\in\mathcal{X}} q_i(r)p_r$, which we henceforth denote as the inner product $\langle q_i, p\rangle$. At a high level our algorithm is going to iteratively generate new learned vectors p^i whose rth entry is the current approximation of p_r, and then seek out a query on which $q(x)$ and $\langle q, p_i\rangle$ greatly differ, using that query for the next learning step.

Algorithm 1. *We define $\mathcal{U} : [0,1]^n \times \{q_i\} \times [0,1] \to [0,1]^n$ to be our learning algorithm, and $L = L(\alpha, |\mathcal{X}|)$ to be determined later.*

1. *Let p^0 be defined by $p^0_r = \frac{1}{|\mathcal{X}|}$ for any $r \in \mathcal{X}$, i.e. p^0 is the uniform distribution on $\mathcal{X}$.*
2. *For $t = 1 \ldots L$:*
    - *(a) sample $i_t \propto \exp(\frac{|q_{i_t}(x) - \langle q_{i_t}, p^{t-1}\rangle}{2\epsilon_0/n})$ and let $y_t = q_{i_t}(x) + \mathrm{Lap}(\frac{1}{\epsilon n})$.*
    - *(b) If $|y_t - \langle q_{i_t}, p^{t-1}\rangle| > 2\alpha$, set $p^t = \mathcal{U}(p^{t-1}, q_{i_t}, y_t)$.*
    - *(c) Otherwise output $\mathcal{M}(x) = (\langle q_1, p^{t-1}\rangle, \ldots, \langle q_m, p^{t-1}\rangle)$ and terminate.*
3. *Output $\mathcal{M}(x) = (\langle q_1, p^L\rangle, \ldots, \langle q_m, p^L\rangle)$*

We now proceed to verify both the privacy guarantee and accuracy of our algorithm, as well as the follwing properties necessary for $\mathcal{U}$ to function properly: $\forall t, |q_{i_t}(x) - y_t| \leq \alpha$, and $\forall t, |q_{i_t}(x) - \langle q_{i_t}, p^{t-1}\rangle| > \alpha$. (The first property says that each y_t is an accurate approximation of q_{i_t}; the second says that q_{i_t} distinguishes the true database from p^{t-1}.)

1.1 Privacy

Because we used the exponential mechanism in sampling i_t and the Laplace mechanism in generating y_t, we have an algorithm that is exactly $2\epsilon_0$–DP for each iteration, meaning that overall we have $2L\epsilon_0$–DP, which by a more careful argument based on last week's methods implies it is $(O(\sqrt{L/\delta} + L\epsilon_0^2), \delta)$–DP. Thus by setting $\epsilon_0 = \frac{\epsilon}{c\sqrt{L \log 1/\delta}}$ for large enough c our algorithm is (ϵ, δ)–DP.

1.2 Accuracy

The learning algorithm $\mathcal{U}$, which we describe in the next section, has the property that if $p^0, \ldots, p^k$, $q_{i_1}, \ldots, q_{i_k}$, and $y_1, \ldots y_k$ are generated so that the following hold:

1. p^0 is uniform on $\mathcal{X}$, and for each $t > 1$, $p^t = \mathcal{U}(p^{t-1}, q_{i_t}, y_t)$;
2. $\forall t, |q_{i_t}(x) - y_t| \leq \alpha$;
3. $\forall t, |q_{i_t}(x) - \langle q_{i_t}, p^{t-1}\rangle| > \alpha$;

then $k \leq L(\alpha, |\mathcal{X}|)$. In fact, we will show that $L(\alpha, |\mathcal{X}|)$ can be taken to be $\frac{4 \log |\mathcal{X}|}{\alpha^2}$. In other words, this guarantee says that $\mathcal{U}$ will learn x after being given at most L distinguishing queries.

By our accuracy analysis of the exponential mechanism (used for sampling i_t) and the Laplace mechanism (used in generating y_t), with large constant probability it holds that for all t,

$$|q_{i_t}(x) - \langle q_{i_t}, p^{t-1}\rangle| \geq \max_i |q_i(x) - \langle q_i, p^{t-1}\rangle| - O\left(\frac{\log mL}{n\epsilon_0}\right) \qquad (1)$$

and

$$|q_{i_t}(x) - y_t| \leq O\left(\frac{\log L}{n\epsilon_0}\right). \qquad (2)$$

Setting $n \geq \frac{C \log mL}{\alpha \epsilon_0}$ for a large enough constant C makes both big-O terms above at most α. Let us condition on these inequalities holding.

First we consider the error in the event that the algorithm does not terminate early. If algorithm $\mathcal{U}$ is called on the t-th iteration, it must be the case that $|q_{i_t}(x) - y_t| \leq \alpha$ and

$$|q_{i_t}(x) - \langle q_{i_t}, p^{t-1} \rangle| \geq |y_t - \langle q_{i_t}, p^{t-1} \rangle| - |q_{i_t}(x) - y_t| > \alpha.$$

Therefore, the conditions for calling $\mathcal{U}$ specified above are both satisfied, and, if the algorithm does not terminate early, then it must be the case that upon termination there exists no query $q_{i_{L+1}}$ such that $|q_{i_{L+1}}(x) - \langle q_{i_t}, p^L \rangle| > \alpha$. I.e. p^L answers all queries accurately up to error α.

Now we bound the error when the algorithm does terminate early, say on step t. By the termination condition, the traingle inequality, and (2), it must be the case that $|q_{i_t}(x) - \langle q_{i_t}, p^{t-1} \rangle| \leq 3\alpha$. Together with (1), this implies that for all queries, p^{t-1} gives error at most 4α.

Overall, this analysis shows that the algorithm has error at most 4α total error of our algorithm. Setting $\epsilon_0 = \frac{\epsilon}{c\sqrt{L \log 1/\delta}}$ and $L = \frac{4 \log |\mathcal{X}|}{\alpha^2}$ gives an algorithm which is (ϵ, δ)-DP and accurate on all queries up to an error of 4α as long as $n = \Omega\left(\frac{\log m \sqrt{\log |\mathcal{X}|} \sqrt{\log 1/\delta}}{\alpha^2 \epsilon}\right)$. Rescaling α appropriately gives an α-accurate algorithm while increasing the lower bound on n only by a factor of four. This bound is in general information-theoretically the best possible. The running time is dominated by that of the learning algorithm, which we describe in the next section.

2 The learning algorithm

We finally arrive at the algorithm $\mathcal{U}$ itself. The main property we need to satisfy in designing $\mathcal{U}$ is that it should converge to a good approximation of the true p (which we denote p^*) as fast as possible, to let us minimize L. By the design of our algorithm we are guaranteed that for all t, $|q_{i_t}(x) - \langle q_{i_t}, p^{t-1} \rangle| > \alpha$ and $|q_{i_t}(x) - y_t| \leq \alpha$, which we will use in our analysis.

Algorithm 2 **($\mathcal{U}(p, q, y)$).** *Observe that $\langle q, p \rangle - y$ has the same sign as $\langle q, p \rangle - q(x)$ by the guarantees above along with the triangle inequality. Intuitively we will change q to guarantee that p "overshoots" the correct value of $q(x)$ when taking the inner product with the new q. We then adjust for this by decreasing the entries of p proportional to how large the corresponding q entries are.*

1. *If $y < \langle q, p \rangle$, $\hat{q} = q$, else $\hat{q} := \mathbb{1}^{\mathcal{X}} - q$.*
2. *For all $r \in \mathcal{X}$, $\hat{p_r} = p_r e^{-\hat{q}(r)\alpha/2}$.*
3. *Output $\frac{\hat{p}}{\sum_r \hat{p}_r}$ (where $\hat{p}$ is the vector with entries $\hat{p}_r$).*

This is called the multiplicative weights update method. Notice that this algorithm runs in time linear in $|\mathcal{X}|$, so the running time of our private mechanism will be $O(mnL|\mathcal{X}|)$. While $|\mathcal{X}|$ can be huge (e.g. exponential in the dimensionality of the data), as we mentioned above, running time polynomial in $|\mathcal{X}|$ cannot be avoided in general, under standard cryptographic assumptions. Nevertheless, we will see in the coming weeks that for special natural classes of queries we can design even more efficient algorithms.

We now state the main technical theorem necessary to prove bounds on L, which gives us our value of n as well as the runtime for our algorithm.

Theorem 2. *Let $p_r^* = \frac{|\{i : x_i = r\}|}{n}$, and let $\Psi_t = D_{KL}(p^*||p^t) = \sum_{r\in\mathcal{X}} p_r^* \log\frac{p_r^*}{p_r^t}$. Then*

$$\Psi_{t-1} - \Psi_t \geq \frac{\alpha^2}{4}$$

Proof. Note that we can derive the following bounds from standard Taylor approximations for all x s.t. $|x| < 1$:

- $e^x \leq 1 + x + x^2$
- $\log(1 + x) \leq x$.

Let $p = p^{t-1}$, $p' = p^t$ (to avoid confusion with exponents in the proof), and likewise let $q = q_{i_t}$.

$$\begin{aligned}
\Psi_{t-1} - \Psi_t &= \sum_{r\in\mathcal{X}} p_r^*(\log\frac{p_r^*}{p_r} - \log\frac{p_r^*}{p_r'}) \\
&= \sum_{r\in\mathcal{X}} p_r^* \log\frac{p_r'}{p_r} \\
&= \sum_{r\in\mathcal{X}} p_r^* \log\frac{\hat{p}_r/\sum_{s\in\mathcal{X}}\hat{p}_s}{p_r} \\
&= \sum_{r\in\mathcal{X}} p_r^* \log e^{-\hat{q}(r)\alpha/2}\frac{1}{\sum_{s\in\mathcal{X}}\hat{p}_s} \\
&= \sum_{r\in\mathcal{X}} p_r^*(-\hat{q}(r)\frac{\alpha}{2} - \log\sum_{s\in\mathcal{X}}\hat{p}_s) \\
&= -\frac{\alpha}{2}\hat{q}(x) - \log\sum_{s\in\mathcal{X}} p_s e^{-\hat{q}(s)\alpha/2} \\
&\geq -\frac{\alpha}{2}\hat{q}(x) - \log\sum_{s\in\mathcal{X}} p_s(1 - \frac{\alpha}{2}\hat{q}(s) + \frac{\alpha^2}{4}\hat{q}^2(s)) \\
&\geq -\frac{\alpha}{2}\hat{q}(x) - \log(1 - \frac{\alpha}{2}\langle\hat{q}, p\rangle + \frac{\alpha^2}{4}) \\
&\geq -\frac{\alpha}{2}\hat{q}(x) + \frac{\alpha}{2}\langle\hat{q}, p\rangle - \frac{\alpha^2}{4} \\
&= \frac{\alpha}{2}(\langle\hat{q}, p\rangle - \hat{q}(x)) - \frac{\alpha^2}{4} \\
&\geq \frac{\alpha}{2}\alpha - \frac{\alpha^2}{4} = \frac{\alpha^2}{4}
\end{aligned}$$

□

Corollary 3. *$L \leq \frac{4}{\alpha^2}\log|\mathcal{X}|$, which implies that $n \geq \frac{c\log mL}{\alpha\epsilon_0} \geq \frac{c'\sqrt{\log|\mathcal{X}|}\log m\sqrt{\log 1/\delta}}{\alpha^2\epsilon}$.*

Proof. This follows directly from the fact that because p^0 was generated uniformly,

$$\Psi_0 = \sum_{r\in\mathcal{X}} p_r^*\log(|\mathcal{X}|p_r^*) = \log|X| - \sum_{r\in\mathcal{X}} p_r^*\log\frac{1}{p_r^*} \leq \log|\mathcal{X}|,$$

along with the fact that $\Psi_t \geq 0$ for all t by the nonnegativity of KL divergence (proved in last week's lecture). □