Lecture 3 Sparse vector technique and linear query release

Yu-Xiang Wang

COMPUTER SCIENCE
UC SANTA BARBARA
Computing. ReInvented.

Recap: last lecture

- Definition of differential privacy
  - Bayesian interpretation
  - Effect of conditioning on side-information
- Notations and setup
  - Representation of a dataset (let's go over it again!)
- DP mechanisms and their privacy analysis
  - Randomized response
  - Laplace mechanism

Recap: Mathematical notations to represent a dataset / dataset space

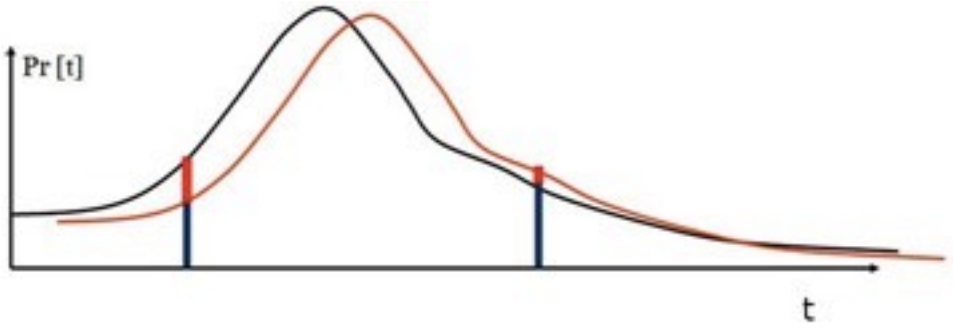
- Histogram representation:
  - A dataset is a histogram supported on the space of data point $\mathcal{X}$
- Indicator representation:
  - A bit vector on the size of the population of individuals, each has their associated features.
- Standard data-table representation:
  - Union of datasets of size d by n for all integer n.

Recap: DP definition

Definition 2.4 (Differential Privacy)**.** A randomized algorithm $\mathcal{M}$ with domain $\mathbb{N}^{|\mathcal{X}|}$ is (ε, δ)-differentially private if for all $\mathcal{S} \subseteq \text{Range}(\mathcal{M})$ and for all $x, y \in \mathbb{N}^{|\mathcal{X}|}$ such that $\|x - y\|_1 \leq 1$:

$$\Pr[\mathcal{M}(x) \in \mathcal{S}] \leq \exp(\varepsilon) \Pr[\mathcal{M}(y) \in \mathcal{S}] + \delta,$$

where the probability space is over the coin flips of the mechanism $\mathcal{M}$. If $\delta = 0$, we say that $\mathcal{M}$ is ε-differentially private.



Recap: Randomized Response

- Space of the answer: {0,1}

1. Each individual tosses an independent coin with probability p > 0.5
2. If “head”, keep your answer.
3. Otherwise, flip your answer.

- The following choice of the parameter satisfies ε-DP.

1. Each individual tosses an independent coin with probability $e^\epsilon/(1+e^\epsilon)$
2. If “head”, keep your answer.
3. Otherwise, flip your answer.

- Unbiased estimator of the data

$$Y \sim \mathrm{RR}_p(x) \qquad \hat{x} = 0.5 + \frac{Y - 0.5}{2p - 1}$$

Recap: Laplace mechanism

- Consider the query aims at releasing real value(s)

$$f : \mathbb{N}^{|\mathcal{X}|} \rightarrow \mathbb{R}^k$$

- L1 Sensitivity of the query:

$$\Delta f = \max_{\substack{x,y \in \mathbb{N}^{|\mathcal{X}|} \\ \|x-y\|_1=1}} \|f(x) - f(y)\|_1$$

- Laplace mechanism returns

$$f(x) + Z \text{ where } Z_i \sim \text{Lap}(\Delta f/\epsilon) \text{ i.i.d. for } i \in [k]$$

This lecture

1. Finish the utility analysis of the Laplace mechanism

2. Apply Laplace mechanism and Randomized Response to the problem of linear query release

3. Sparse Vector Technique

4. Using SVT to obtain stronger bounds for releasing linear queries

Readings

- Dwork and Roth textbook.
  - Chapter 3.6, Chapter 4.2.

- Supplementary reading:
  - Lyu, M., Su, D., & Li, N. (2017). Understanding the Sparse Vector Technique for Differential Privacy. *Proceedings of the VLDB Endowment*, *10*(6).
  - Zhu, Y., & W. (2020). Improving Sparse Vector Technique with Renyi Differential Privacy. *Advances in Neural Information Processing Systems*, *33*.

Utility of the Laplace Mechanism

- CDF of the Laplace distribution:

$$\begin{cases} \frac{1}{2}\exp\left(\frac{x-\mu}{b}\right) & \text{if } x \leq \mu \\ 1 - \frac{1}{2}\exp\left(-\frac{x-\mu}{b}\right) & \text{if } x \geq \mu \end{cases}$$

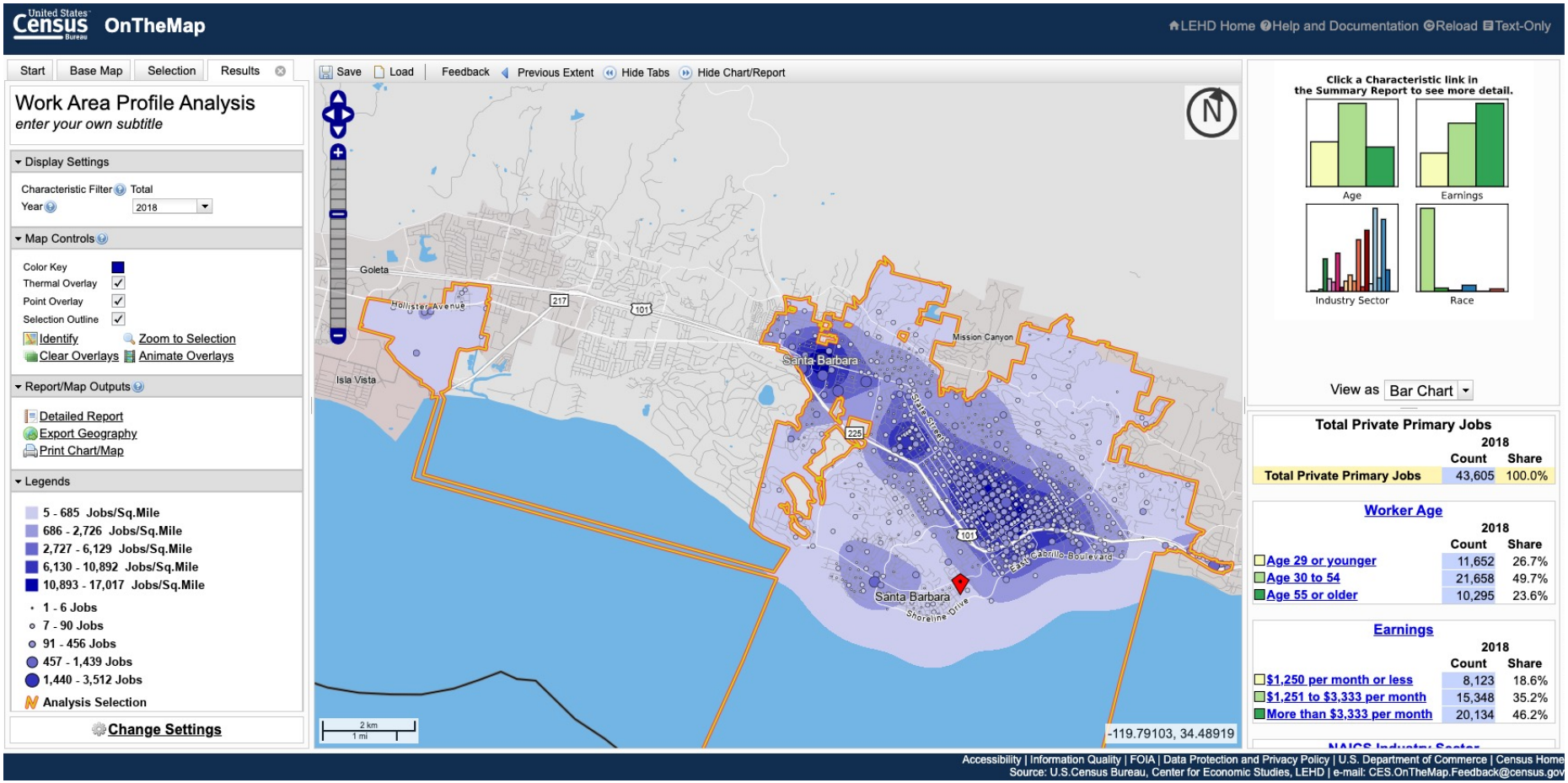
Theorem 3.8. Let $f : \mathbb{N}^{|\mathcal{X}|} \to \mathbb{R}^k$, and let $y = \mathcal{M}_L(x, f(\cdot), \varepsilon)$. Then $\forall \delta \in (0, 1]$:

$$\Pr\left[\|f(x) - y\|_\infty \geq \ln\left(\frac{k}{\delta}\right) \cdot \left(\frac{\Delta f}{\varepsilon}\right)\right] \leq \delta$$

Example applications of Laplace mechanism. What is the L1 sensitivity?

- Linear query (from the last lecture)
- Histograms: distribution of grades in a class
- Demographics statistics over map:
  - Number of people living in different zip code by race and gender
- COVID'19 Hospitalization Data:
  - Number of active patients in the ICU of each hospital

Example: “OnTheMap” by US Census --- the first large-scale deployment of DP from 2008.



Check it out yourself: https://onthemap.ces.census.gov/

The paper: Machanavajjahala et al. “Privacy: Theory meets Practice on the Map”

Apply Laplace mechanism to answer many linear queries

1. Set privacy budget, and number of queries
2. Decide how much noise to add
3. Work out the error bound
4. Error bound => sample complexity

Apply randomized response to answer linear queries

- Answering a single linear query $\hat{x} = 0.5 + \frac{Y - 0.5}{2p - 1}$

Hoeffding's inequality: Suppose that $X_1, \ldots, X_n$ are independent and that, $a_i \leq X_i \leq b_i$, and $\mathbb{E}[X_i] = \mu$. Then for any $t > 0$,

$$\mathbb{P}\left(\left|\overline{X} - \mu\right| \geq t\right) \leq 2\exp\left(-\frac{2n^2t^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\right) \quad \text{where } \overline{X}_n = n^{-1}\sum_i X_i.$$

Apply randomized response to answer linear queries

- Answering many linear query

- Question: does it cost any additional privacy?

Comparing randomized response and Laplace mechanism in answering linear queries.

Back to the table for releasing linear queries

Target accuracy	k = O(2^n) linear queries	k = O(n) linear queries	k << n linear queries
α = O(1) (any non-trivial error)	Blatantly non-private	?	?
α = O(1/sqrt(n)) (statistical error)	Blatantly non-private	Blatantly non-private	**DP / Laplace mech**
α = o(1/sqrt(n)) (<< statistical error)	Blatantly non-private	Blatantly non-private	**DP / Laplace mech**

Multiply error with with n if unnormalized.

Let's generalizing the definition of linear query a little bit.

- Statistical query

- In the "histogram" representation
  - This is a linear function with

$$q \in [0,1]^{|\mathcal{X}|}$$

Alternative way of applying Laplace mechanism to release the entire dataset

- For each $x \in \mathcal{X}$, using Laplace mechanism to release

$$\sum_{i=1}^{n} \mathbf{1}(\phi_i = x) = \langle \mathbf{e}_x, \mathbf{Hist}([\phi_1, ..., \phi_n]) \rangle$$

- What is the L1-sensitivity?

Checkpoint

- Laplace mechanism to release queries directly
  - Each query is $O(|Q| \log |Q| / \varepsilon)$ accurate.

- Randomized response to release data (under a more restrictive model)
  - Error is $O(\log N/ \varepsilon)$ per coordinate.
  - For answering $|Q|$ linear queries, the error is $O(\sqrt{N}\log N \log |Q| / \varepsilon)$

- Laplace mechanism to release contingency table
  - Error is $O(\log |X| / \varepsilon)$ per coordinate.
  - For answering $|Q|$ linear queries, the error is $O(\sqrt{|X|}\log |X| \log |Q| / \varepsilon)$

- Is there an algorithm that can achieve $O(\text{polylog}(|X|,|Q|))$ error in answering linear queries for a constant ε?

When there is **a large number of** queries, but **only a few** that are interesting

- Example 1: Anomaly detection in a time series.
  - Most queries have small values, we do not wish to know the exact numbers.

- Example 2: Exploiting additional information
  - When we have access to a data-independent oracle, e.g., a ML model, that is often right.
  - We query the private dataset for answers only if the answer differs substantially with the oracle.

Can we get away with paying only for those interesting ones?

AboveThreshold mechanism

Algorithm 1 Input is a private database D, an adaptively chosen stream of sensitivity 1 queries $f_1, \ldots$, and a threshold T. Output is a stream of responses $a_1, \ldots$

AboveThreshold$(D, \{f_i\}, T, \epsilon)$

Let $\hat{T} = T + \text{Lap}\left(\frac{2}{\epsilon}\right)$.
for Each query i **do**
  Let $\nu_i = \text{Lap}(\frac{4}{\epsilon})$
  if $f_i(D) + \nu_i \geq \hat{T}$ **then**
    Output $a_i = \top$.
    Halt.
  else
    Output $a_i = \perp$.
  end if
end for

SparseVector mechanism

1. Start with a budget of ε, and a maximum number of "discoveries" c
2. Split ε into c equal parts and run **AboveThreshold** for up to c times, each with a privacy budget of ε/c.
3. Stop when either the stream of queries are exhausted or if all c discoveries are made.

SVT is notoriously tricky to analyze. Many authors got it wrong.

Algorithm 1 An instantiation of the SVT proposed in this paper.

Input: $D, Q, \Delta, \mathbf{T} = T_1, T_2, \cdots, c.$
1: $\epsilon_1 = \epsilon/2, \quad \rho = \mathsf{Lap}(\Delta/\epsilon_1)$
2: $\epsilon_2 = \epsilon - \epsilon_1, \quad \text{count} = 0$
3: **for** each query $q_i \in Q$ **do**
4: $\quad \nu_i = \mathsf{Lap}(2c\Delta/\epsilon_2)$
5: $\quad$ **if** $q_i(D) + \nu_i \geq T_i + \rho$ **then**
6: $\quad\quad$ Output $a_i = \top$
7: $\quad\quad$ count = count + 1, **Abort** if count $\geq c$.
8: $\quad$ **else**
9: $\quad\quad$ Output $a_i = \bot$

Algorithm 2 SVT in Dwork and Roth 2014 [8].

Input: $D, Q, \Delta, T, c.$
1: $\epsilon_1 = \epsilon/2, \quad \rho = \mathsf{Lap}(c\Delta/\epsilon_1)$
2: $\epsilon_2 = \epsilon - \epsilon_1, \quad \text{count} = 0$
3: **for** each query $q_i \in Q$ **do**
4: $\quad \nu_i = \mathsf{Lap}(2c\Delta/\epsilon_1)$
5: $\quad$ **if** $q_i(D) + \nu_i \geq T + \rho$ **then**
6: $\quad\quad$ Output $a_i = \top, \rho = \mathsf{Lap}(c\Delta/\epsilon_2)$
7: $\quad\quad$ count = count + 1, **Abort** if count $\geq c$.
8: $\quad$ **else**
9: $\quad\quad$ Output $a_i = \bot$

Algorithm 3 SVT in Roth's 2011 Lecture Notes [15].

Input: $D, Q, \Delta, T, c.$
1: $\epsilon_1 = \epsilon/2, \quad \rho = \mathsf{Lap}(\Delta/\epsilon_1),$
2: $\epsilon_2 = \epsilon - \epsilon_1, \quad \text{count} = 0$
3: **for** each query $q_i \in Q$ **do**
4: $\quad \nu_i = \mathsf{Lap}(c\Delta/\epsilon_2)$
5: $\quad$ **if** $q_i(D) + \nu_i \geq T + \rho$ **then**
6: $\quad\quad$ Output $a_i = q_i(D) + \nu_i$
7: $\quad\quad$ count = count + 1, **Abort** if count $\geq c$.
8: $\quad$ **else**
9: $\quad\quad$ Output $a_i = \bot$

Algorithm 4 SVT in Lee and Clifton 2014 [13].

Input: $D, Q, \Delta, T, c.$
1: $\epsilon_1 = \epsilon/4, \quad \rho = \mathsf{Lap}(\Delta/\epsilon_1)$
2: $\epsilon_2 = \epsilon - \epsilon_1, \quad \text{count} = 0$
3: **for** each query $q_i \in Q$ **do**
4: $\quad \nu_i = \mathsf{Lap}(\Delta/\epsilon_2)$
5: $\quad$ **if** $q_i(D) + \nu_i \geq T + \rho$ **then**
6: $\quad\quad$ Output $a_i = \top$
7: $\quad\quad$ count = count + 1, **Abort** if count $\geq c$.
8: $\quad$ **else**
9: $\quad\quad$ Output $a_i = \bot$

Algorithm 5 SVT in Stoddard et al. 2014 [18].

Input: $D, Q, \Delta, T.$
1: $\epsilon_1 = \epsilon/2, \quad \rho = \mathsf{Lap}(\Delta/\epsilon_1)$
2: $\epsilon_2 = \epsilon - \epsilon_1$
3: **for** each query $q_i \in Q$ **do**
4: $\quad \nu_i = 0$
5: $\quad$ **if** $q_i(D) + \nu_i \geq T + \rho$ **then**
6: $\quad\quad$ Output $a_i = \top$
7:
8: $\quad$ **else**
9: $\quad\quad$ Output $a_i = \bot$

Algorithm 6 SVT in Chen et al. 2015 [1].

Input: $D, Q, \Delta, \mathbf{T} = T_1, T_2, \cdots.$
1: $\epsilon_1 = \epsilon/2, \quad \rho = \mathsf{Lap}(\Delta/\epsilon_1)$
2: $\epsilon_2 = \epsilon - \epsilon_1$
3: **for** each query $q_i \in Q$ **do**
4: $\quad \nu_i = \mathsf{Lap}(\Delta/\epsilon_2)$
5: $\quad$ **if** $q_i(D) + \nu_i \geq T_i + \rho$ **then**
6: $\quad\quad$ Output $a_i = \top$
7:
8: $\quad$ **else**
9: $\quad\quad$ Output $a_i = \bot$

Lyu, Min, Dong Su, and Ninghui Li. "Understanding the Sparse Vector Technique for Differential Privacy." *Proceedings of the VLDB Endowment* 10.6 (2017).

Slight generalization of AboveThreshold mechanism

Algorithm 1 Input is a private database D, an adaptively chosen stream of sensitivity 1 queries $f_1, \ldots$, and a threshold T. Output is a stream of responses $a_1, \ldots$

AboveThreshold$(D, \{f_i\}, T, \epsilon)$

Let $\hat{T} = T + \text{Lap}\left(\frac{2}{\epsilon}\right)$.
for Each query i **do**
  Let $\nu_i = \text{Lap}(\frac{4}{\epsilon})$
  if $f_i(D) + \nu_i \geq \hat{T}$ **then**
    Output $a_i = \top$.
    Halt.
  else
    Output $a_i = \perp$.
  end if
end for

The privacy guarantee of AboveThreshold

- We will prove something slightly stronger.
  - Any noise-adding mechanism M1 satisfying ε/2-DP for queries with sensitivity 1.
  - Any noise-adding mechanism M2 satisfying ε/2-DP for queries with sensitivity 2.
- The above generalized AboveThreshold satisfies ε-DP
  - Thus by the composition theorem, SVT satisfies cε-DP
- A few notations:
  - The threshold noise R.V., denoted by z
  - The query noise R.V. denoted by $\nu_1, \nu_2, \ldots, \nu_k, \nu_{k+1}, \cdots$
  - Denote the queries by $q_1, q_2, \ldots, q_k, q_{k+1}, \cdots$

Let's parse the elements of the proof, and make a few observations

1. The output space of the algorithm

$$\{\perp^k \top | k = 0, 1, ..., \infty\}$$

   - The output can be completely described by a random integer k
2. w.l.o.g., we can assume T = 0 (why?)
3. The probability of outputting k is

$$\int_{-\infty}^{+\infty} p_\rho(z) \left(\prod_{i \leq k} \int_{-\infty}^{z - q_i(D')} p(\nu_i) d\nu_i \right) \cdot \int_{z + q_{k+1}(D')}^{\infty} p(\nu_{k+1}) d\nu_{k+1} dz.$$

The analysis of AboveThreshold

$$
\begin{aligned}
&\Pr[\mathcal{M}(D) = k] = \mathbb{E}_{z\sim p_\rho}[\Pr[\mathcal{M}(D) = k|z]] \\
=&\mathbb{E}_{z\sim p_\rho}[\prod_{i\le k} \Pr[q_i(D) + \nu_i < z|z]\Pr[q_{k+1}(D) + \nu_i \ge z|z]] \\
=&\int_{-\infty}^{+\infty} p_\rho(z)\left(\prod_{i\le k}\int_{-\infty}^{z-q_i(D)} p(\nu_i)d\nu_i\right)\cdot\int_{z-q_{k+1}(D)}^{\infty} p(\nu_{k+1})d\nu_{k+1}dz
\end{aligned}
$$

The analysis of AboveThreshold

$$\Pr[\mathcal{M}(D)=k]=\mathbb{E}_{z\sim p_\rho}[\Pr[\mathcal{M}(D)=k|z]]$$

$$=\mathbb{E}_{z\sim p_\rho}[\prod_{i\le k}\Pr[q_i(D)+\nu_i<z|z]\Pr[q_{k+1}(D)+\nu_i\ge z|z]]$$

$$=\int_{-\infty}^{+\infty}p_\rho(z)\Big(\prod_{i\le k}\int_{-\infty}^{z-q_i(D)}p(\nu_i)d\nu_i\Big)\cdot\int_{z-q_{k+1}(D)}^{\infty}p(\nu_{k+1})d\nu_{k+1}dz$$

$$\overset{u:=z+\triangle}{=}\int_{-\infty}^{+\infty}p_\rho(u-\triangle)\Big(\prod_{i\le k}\int_{-\infty}^{u-\triangle-q_i(D)}p(\nu_i)d\nu_i\Big)\cdot\int_{u-\triangle-q_{k+1}(D)}^{\infty}p(\nu_{k+1})d\nu_{k+1}du$$

$$=\int_{-\infty}^{+\infty}p_\rho(u)\left(\frac{p_\rho(u-\triangle)}{p_\rho(u)}\right)\Big(\prod_{i\le k}\int_{-\infty}^{u-\triangle-q_i(D)}p(\nu_i)d\nu_i\Big)\cdot\int_{u-\triangle-q_{k+1}(D)}^{\infty}p(\nu_{k+1})d\nu_{k+1}du$$

$$=\mathbb{E}_{z\sim p_\rho}\left[\left(\frac{p_\rho(z-\triangle)}{p_\rho(z)}\right)\Big(\prod_{i\le k}\int_{-\infty}^{z-\triangle-q_i(D)}p(\nu_i)d\nu_i\Big)\cdot\int_{z-\triangle-q_{k+1}(D)}^{\infty}p(\nu_{k+1})d\nu_{k+1}\right]$$

Bounding the third term via a fictitious query

Utility of SparseVector

- Idea is to simply bound the magnitude of the noise
  - (All true discoveries.) With high probability, we do not wrongly reject interesting queries.
  - (No-false discovery). With high probability, we also do not wrongly identify queries that are not interesting as interesting.
  - Union bound over all of them.

We are outputting only the selections, but not numerical values?

- This is trivial to fix with twice the privacy budget.
- Compose the following:
  - AboveThresh1, LapMech1, AboveThresh2, LapMech2,…
- Each one of the mechanism is adaptively chosen based on realized previous outcomes.
  - How does LapMech_j depends on the output of AboveThresh_j?
  - How does the AboveThresh_j depends on all previous outputs?

Let's apply the above SVT method for online query release.

- Problem setup:
  - A adaptive online sequence of linear queries.
  - The curator has to answer them as they arrive.

- Baseline:
  - Laplace mechanism for releasing queries $O(|Q|/ \varepsilon)$
  - Laplace mechanism for releasing contingency table $O(\sqrt{|X|} \log |Q| / \varepsilon)$.

- Question: Is it possible to get $O(\text{polylog}(|Q|,|X|))$ error?

Idea: Use correlated noise by learning a synthetic dataset

- We will be using sparse vector technique!
- For an online sequence of queries
  - Continue to run ”AboveThreshold”, if error below a noise threshold
    - Return what the synthetic data set returns
  - else: Release the query using Laplace Mechanism
    - Update the synthetic data
    - Restart “AboveThreshold”

Detour: No-regret online learning from expert advice

- N experts, each give advices on stock choices
- After each day, their losses are revealed
- Can I come up with a strategy that does as well as the best expert with (asymptotically) no regret?
- Define **"Regret":**

Multiplicative Weights Algorithm (i.e., the Hedge algorithm)

Algorithm 1 Hedge

1: Initialize: $\forall i \in [N],\ W_1(i) = 1$
2: **for** $t = 1$ to T **do**
3: Pick $i_t \sim_R W_t$, i.e., $i_t = i$ with probability $\mathbf{x}_t(i) = \frac{W_t(i)}{\sum_j W_t(j)}$
4: Incur loss $\ell_t(i_t)$.
5: Update weights $W_{t+1}(i) = W_t(i)e^{-\varepsilon \ell_t(i)}$
6: **end for**

Theorem 1.5. Let ℓ_t^2 denote the N-dimensional vector of square losses, i.e., $\ell_t^2(i) = \ell_t(i)^2$, let $\varepsilon > 0$, and assume all losses to be non-negative. The Hedge algorithm satisfies for any expert $i^\star \in [N]$:

$$\sum_{t=1}^{T} \mathbf{x}_t^\top \ell_t \leq \sum_{t=1}^{T} \ell_t(i^\star) + \varepsilon \sum_{t=1}^{T} \mathbf{x}_t^\top \ell_t^2 + \frac{\log N}{\varepsilon}$$

(From the Online Convex Optimization book by Elad Hazan)

Corollary: we can also compete with the best probability distribution!

Theorem 1.5. Let ℓ_t^2 denote the N-dimensional vector of square losses, i.e., $\ell_t^2(i) = \ell_t(i)^2$, let $\varepsilon > 0$, and assume all losses to be non-negative. The Hedge algorithm satisfies for any expert $i^\star \in [N]$:

$$\sum_{t=1}^{T} \mathbf{x}_t^\top \ell_t \leq \sum_{t=1}^{T} \ell_t(i^\star) + \varepsilon \sum_{t=1}^{T} \mathbf{x}_t^\top \ell_t^2 + \frac{\log N}{\varepsilon}$$

- Why?

How does MW applies to the problem of linear query release?

- Let's consider the problem without privacy.
- True data $p = x/n$, initial synthetic data $\tilde{p}_1 = 1/|\mathcal{X}|$
- Adversary selects an online sequence of queries
  - If $|q^T \tilde{p}_t - q^T p| \geq \frac{\alpha}{n}$
    - Set the loss vector to be $\ell_t := \mathrm{sign}(q^T \tilde{p}_t - q^T p) \cdot q$
    - Update $\tilde{p}_{t+1} = \mathrm{Normalize}_n\left(\tilde{p}_t \cdot \exp(-\eta \ell_t)\right)$
    - Increment t, i.e., t = t + 1
    - Output $q^T x$
  - Else: output $q^T \tilde{x}_t$

The regret bound of MW implies that the number of iterations of the MW algorithm is small!

Next lecture

- Private Multiplicative Weight:
  - How to combine SVT and Hedge to obtain an error of O(polylog (|Q|,|X|)) while satisfying pure-DP
- The problem of private selection
  - Exponential mechanism
  - Report noisy (arg)max