11.12.2024 ***Adaptive data analysis***

Lecture 6: Generalization via DP-Stability

Source: Lecture notes by Aaron Roth and Adam Smith

Lecturer: Uri Stemmer

The following conclusion follows from the composition theorem we proved in the last lecture:

Conclusion 1: Let $0 < \varepsilon, \delta < 1$ and let $k \in \mathbb{N}$. Denote $\gamma = \frac{\varepsilon}{\sqrt{8k \cdot \ln(1/\delta)}}$. Then the composition of k applications of $M_{\text{Lap}}^{1/(\gamma n)}$, denoted as $\vec{M}_{\text{Lap},k}^{1/(\gamma n)}$, is (ε, δ)-DP-Stable for k queries.

Note that the stability guarantees degrade only as $\sqrt{k}$ and not linearly with k. This means that the Laplace mechanism remains stable even after k queries. What about accuracy? Empirical accuracy is straightforward, as the next claim shows (we still need to show that stability + empirical accuracy guarantees statistical validity...)

Claim 2: Let $0 < \beta, \varepsilon, \delta < 1$, let $k \in \mathbb{N}$, and denote $\gamma = \frac{\varepsilon}{\sqrt{8k \cdot \ln(1/\delta)}}$. The mechanism $\vec{M}_{\text{Lap},k}^{1/(\gamma n)}$ is (α, β)-empirically accurate for k queries using a sample of size n, where

$$\alpha = \frac{1}{\gamma n} \cdot \ln\left(\frac{k}{\beta}\right) = O\left(\frac{\sqrt{k}}{\varepsilon n} \cdot \sqrt{\ln\left(\frac{1}{\delta}\right)} \cdot \ln\left(\frac{k}{\beta}\right)\right)$$

Proof:

Recall that for every query q_i, the mechanism $\vec{M}_{\text{Lap},k}^{1/(\gamma n)}$ returns an answer $q_i(S) + Y_i$ where $Y_i \sim \text{Lap}\left(\frac{1}{\gamma n}\right)$. Therefore, to bound the empirical error it suffices to bound $\max_i |Y_i|$.

Fix an index i. As we mentioned in previous lectures,

$$\Pr[|Y_i| > \alpha] = \exp(-\gamma n \alpha) \leq \frac{\beta}{k},$$

where the last inequality holds for our choice of $\alpha \geq \frac{1}{\gamma n} \cdot \ln\left(\frac{k}{\beta}\right)$.

Hence, by a union bound over $i \in [k]$,

$$\Pr\left[\max_i |Y_i| > \alpha\right] \leq \beta$$

q.e.d. (Claim 2)

To complete the picture, we need to show that if a mechanism is both DP-stable and empirically accurate then it is also statistically accurate. To this end, we need to introduce another DP-stable building block.

The NoisyArgmax Mechanism

Let $q_1, q_2, \dots, q_T$ a collection of T queries (fixed in advance) and let S be a database. Consider the task of identifying, in a DP-stable manner, an index t such that $q_t(S)$ is approximately maximized. We could do this using the Laplace mechanism by estimating $q_i(S)$ for every i and then choosing an index t that maximizes our estimations. But this is quite wasteful, as this will degrade our stability guarantees by a $\approx \sqrt{T}$ factor. Can we do better?

NoisyAgrmax

Input: Database S and T queries $q_1, q_2, \dots, q_T$

1. For every $1 \le t \le T$ let $\hat{a}_t = q_t(S) + Y_t$ where $Y_t \sim \text{Lap}\left(\frac{2}{\varepsilon n}\right)$
2. Return t such that $\hat{a}_t \ge \hat{a}_j$ for every j

We assume that the queries $q_1, \dots, q_T$ are of sensitivity $1/n$ (recall that a query q has sensitivity $1/n$ if changing one data point changes the value of q by at most $1/n$)

Theorem 3: NoisyAgrmax is $(\varepsilon, 0)$-DP-stable (w.r.t. changing one point in S).

Proof:

Fix the queries $q_1, \dots, q_T$, fix two neighboring datasets S, S', and fix an index $b \in \{1,2, \dots, T\}$.
We need to show that

$$\Pr[\text{NoisyAgrmax}(S) = b] \le e^{\varepsilon} \cdot \Pr[\text{NoisyAgrmax}(S') = b]$$

Recall that for each query q_j we add noise $Y_t \sim \text{Lap}\left(\frac{2}{\varepsilon n}\right)$.
Let us write $\vec{Y}_{-b}$ to denote the vector of all noises except for Y_b.
Then, it suffices to show that for every fixture $\vec{y}_{-b}$ of $\vec{Y}_{-b}$ it holds that

$$\Pr[\text{NoisyAgrmax}(S) = b | \vec{y}_{-b}] \le e^{\varepsilon} \cdot \Pr[\text{NoisyAgrmax}(S') = b | \vec{y}_{-b}]$$

Why is this sufficient? Because then

$$\Pr[\text{NoisyAgrmax}(S) = b] = \sum_{\vec{y}_{-b}} \Pr[\vec{y}_{-b}] \cdot \Pr[\text{NoisyAgrmax}(S) = b | \vec{y}_{-b}]$$

$$\le \sum_{\vec{y}_{-b}} \Pr[\vec{y}_{-b}] \cdot \Pr[\text{NoisyAgrmax}(S') = b | \vec{y}_{-b}] = \Pr[\text{NoisyAgrmax}(S') = b]$$

So let us fix such a noise vector $\vec{y}_{-b}$. We have that

$$\Pr[\text{NoisyAgrmax}(S) = b | \vec{y}_{-b}] = \Pr\left[q_b(S) + \text{Lap}\left(\frac{2}{\varepsilon n}\right) > \max_{j \ne b}\{q_j(S) + y_j\}\right]$$

$$= \Pr\left[q_b(S) + \frac{1}{n} + \text{Lap}\left(\frac{2}{\varepsilon n}\right) > \max_{j \ne b}\left\{q_j(S) + \frac{1}{n} + y_j\right\}\right]$$

$$\leq \Pr\left[q_b(S) + \frac{1}{n} + \mathrm{Lap}\left(\frac{2}{\varepsilon n}\right) > \max_{j \neq b}\{q_j(S') + y_j\}\right]$$

$$\leq \Pr\left[q_b(S') + \frac{2}{n} + \mathrm{Lap}\left(\frac{2}{\varepsilon n}\right) > \max_{j \neq b}\{q_j(S') + y_j\}\right]$$

$$= \Pr\left[\mathrm{Lap}\left(\frac{2}{n}, \frac{2}{\varepsilon n}\right) > \max_{j \neq b}\{q_j(S') + y_j\} - q_b(S')\right] = ((1))$$

Recall that for the density functions $h_{0,\lambda}$ and $h_{\mu,\lambda}$ corresponding to $\mathrm{Lap}(0,\lambda)$ and $\mathrm{Lap}(\mu,\lambda)$, respectively, it holds that

$$e^{-\mu/\lambda} \leq \frac{h_{0,\lambda}(x)}{h_{\mu,\lambda}(x)} \leq e^{\mu/\lambda}$$

Hence,

$$((1)) \leq e^{\varepsilon} \cdot \Pr\left[\mathrm{Lap}\left(\frac{2}{\varepsilon n}\right) > \max_{j \neq b}\{q_j(S') + y_j\} - q_b(S')\right]$$

$$= e^{\varepsilon} \cdot \Pr\left[q_b(S') + \mathrm{Lap}\left(\frac{2}{\varepsilon n}\right) > \max_{j \neq b}\{q_j(S') + y_j\}\right] = e^{\varepsilon} \cdot \Pr[\mathrm{NoisyAgrmax}(S') = b | \vec{y}_{-b}]$$

q.e.d. (Theorem 3)

Claim 4: *$NoisyAgrmax$ returns an index t s.t.* with probability at least $1 - \beta$,

$$q_t(S) \geq \max_j\{q_j(S)\} - O\left(\frac{1}{\varepsilon n} \cdot \ln\frac{T}{\beta}\right)$$

(The proof of Claim 4 is similar to the proof of Claim 2)

DP-stability + empirical accuracy implies statistical accuracy

Theorem 5: Let M be a mechanism that answers k adaptively chosen statistical queries given a sample of size $n = \Omega\left(\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right)$, such that

(a) M is (ε, δ)-DP-stable
(b) M is (α, β)-empirically accurate

Then M is $\left(\alpha + 10\varepsilon,\ \beta + \frac{k\delta}{\varepsilon}\right)$-statistically accurate.

Conclusion 6 (Theorem 5 + the Laplace mechanism):
There exists a computationally efficient mechanism that answers k adaptively chosen statistical queries to within accuracy α using a sample of size $n \gtrsim \frac{\sqrt{k}}{\alpha^2}$.

Towards proving Theorem 5:

Before proving Theorem 5, we simplify it in Theorems 7 and 8 below and explain why proving these simplified theorems suffices.

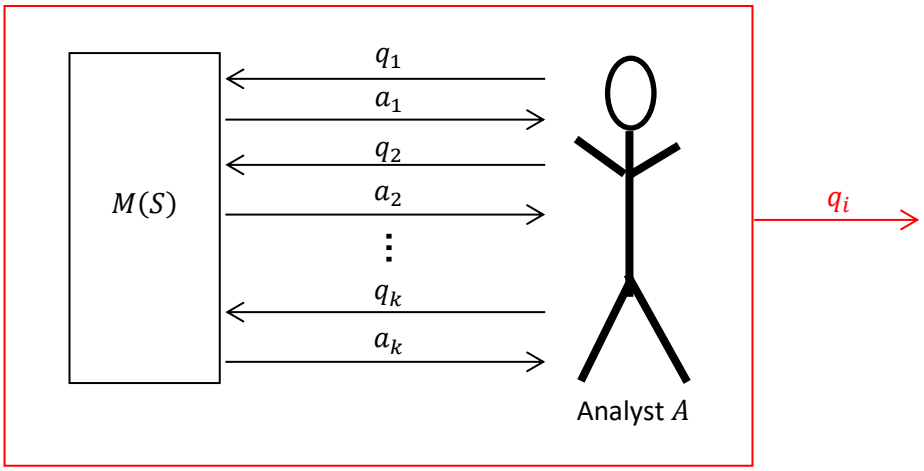
Theorem 7: Let M be an (ε, δ)-DP-stable mechanism for k statistical queries that operates on a sample of size $n = \Omega\left(\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right)$. Then, for every analyst A, every distribution $\mathcal{D}$, and every $1 \leq i \leq k$ it holds that

$$\Pr_{\substack{S\sim\mathcal{D}^n \\ \mathrm{AG}_{n,k}(A,S,M)}}\left[|q_i(S) - q_i(\mathcal{D})| > 10\varepsilon\right] < \frac{\delta}{\varepsilon}$$

Theorem 5 follows from Theorem 7 by using the union bound and the triangle inequality.

Fix an analyst A, fix a distribution $\mathcal{D}$, and fix an index $1 \leq i \leq k$. Let MA_i denote the mechanism that given S simulates the interaction between A and M and then outputs the i-th query q_i chosen by the analyst during the interaction.

In a diagram:



$MA_i(S)$

This is a mechanism whose input is a database S and its output is a single statistical query q_i. As M is DP-stable, we know that the (distribution of the) transcript between A and M is "insensitive" to changing one point in S. In particular, the distribution of q_i is also insensitive, meaning that MA_i is itself DP-stable. It will be easier for us to argue about MA_i (rather than M) as there is no analyst that interacts with it. It's a "one-shot" algorithm that takes a sample S and outputs a query q_i.

Now, instead of proving Theorem 7, we could prove the following theorem:

Theorem 8: Let W be an (ε, δ)-DP-stable mechanism that takes a database S of size $n = \Omega\left(\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right)$ and outputs a statistical query q. Then, for every distribution $\mathcal{D}$ it holds that

$$\Pr_{\substack{S\sim\mathcal{D}^n \\ q\leftarrow W(S)}}\left[|q(S) - q(\mathcal{D})| > 10\varepsilon\right] < \frac{\delta}{\varepsilon}$$

Note that Theorem 8 is simpler than Theorem 7: there is no one asking queries and no rounds. It is just a mechanism W that takes a sample and returns a query. Additionally, note that Theorem 7 follows from Theorem 8, because, as we mentioned, MA_i satisfies the requirements of Theorem 8, and therefore the result holds for the i-th query.

"Proof" of Theorem 8 with missing details:

Let S be a database containing n iid samples from $\mathcal{D}$, and denote $q \leftarrow W(S)$.

Our goal is to show that with high probability it holds that $q(S) \approx q(\mathcal{D})$.

We will first show that this holds in expectation, and afterwards strengthen the result to show that it also holds with high probability.

Claim A:

$$\mathop{\mathbb{E}}_{\substack{S\sim\mathcal{D}^n \\ q\leftarrow W(S)}}[q(S)] \approx \mathop{\mathbb{E}}_{\substack{S\sim\mathcal{D}^n \\ q\leftarrow W(S)}}[q(\mathcal{D})]$$

Explanation: Let's consider the following two experiments:

$S = (x_1, \dots, x_n) \sim \mathcal{D}^n$ $z \sim \mathcal{D}$ $i \in_R \{1,2,\dots,n\}$ $q \leftarrow W(S)$ Return $q(x_i)$	$\approx$ (DP-stability)	$S = (x_1, \dots, x_n) \sim \mathcal{D}^n$ $z \sim \mathcal{D}$ $i \in_R \{1,2,\dots,n\}$ $q \leftarrow W(S \setminus \{x_i\} \cup \{z\})$ Return $q(x_i)$
The outcome of this experiment of the application of q to a random element from S. Thus, in expectation over i, the outcome of this experiment is the empirical average, i.e., $q(S)$.		The outcome of this experiment is the application of q to a random element from $\mathcal{D}$ (independent of q). Thus, in expectation over x_i, the outcome of this experiment is $q(\mathcal{D})$.

Claim B: Let B be an algorithm that takes T databases as input $\vec{S} = (S_1, \dots, S_T) \in (X^n)^T$ and returns a pair (q, t) where $q: X \to \{0,1\}$ is a query and $1 \le t \le T$ is an integer. If B is (ε, δ)-DP-stable then

$$\mathop{\mathbb{E}}_{\substack{\vec{S}\sim\mathcal{D}^n \\ (q,t)\leftarrow B(S)}}[q(S_t)] \approx \mathop{\mathbb{E}}_{\substack{\vec{S}\sim\mathcal{D}^n \\ (q,t)\leftarrow B(S)}}[q(\mathcal{D})]$$

Proof of Claim B: (similar to the proof of Claim A, but requires tracking all T samples)

Notation: The input of algorithm B are T *sub-databases* $\vec{S} = (S_1, S_2, \dots, S_T)$ sampled iid from $\mathcal{D}$. The output is a predicate h and index $1 \le t \le T$. We denote $S_t = (x_{t,1}, \dots, x_{t,n})$.

$$\mathop{\mathbb{E}}_{\substack{\vec{S}\sim\mathfrak{D}\\(h,t)\leftarrow\mathcal{B}(\vec{S})}}[h(S_t)] = \sum_{m=1}^{T} \mathop{\mathbb{E}}_{\substack{\vec{S}\sim\mathfrak{D}\\(h,t)\leftarrow\mathcal{B}(\vec{S})}}\left[\mathbf{1}_{\{t=m\}}\cdot h(S_m)\right]$$

$$= \frac{1}{n}\sum_{i=1}^{n}\sum_{m=1}^{T} \mathop{\mathbb{E}}_{\substack{\vec{S}\sim\mathfrak{D}\\(h,t)\leftarrow\mathcal{B}(\vec{S})}}\left[\mathbf{1}_{\{t=m\}}\cdot h(x_{m,i})\right] = \frac{1}{n}\sum_{i=1}^{n}\sum_{m=1}^{T} \mathop{\Pr}_{\substack{\vec{S}\sim\mathfrak{D}\\(h,t)\leftarrow\mathcal{B}(\vec{S})}}\left[\mathbf{1}_{\{t=m\}}\cdot h(x_{m,i})=1\right]$$

Given $\vec{S}, (m,i), z$ define $\vec{S}^{(x_{m,i}:z)}$ to be as $\vec{S}$ after replacing $x_{m,i}$ with z

$$\leq \frac{1}{n}\sum_{i=1}^{n}\sum_{m=1}^{T}\left(e^{\epsilon} \mathop{\Pr}_{\substack{z,\vec{S}\sim\mathfrak{D}\\(h,t)\leftarrow\mathcal{B}\left(\vec{S}^{(x_{m,i}:z)}\right)}}\left[\mathbf{1}_{\{t=m\}}\cdot h(x_{m,i})=1\right]+\delta\right)$$

$$= \frac{1}{n}\sum_{i=1}^{n}\sum_{m=1}^{T}\left(e^{\epsilon} \mathop{\mathbb{E}}_{\substack{z,\vec{S}\sim\mathfrak{D}\\(h,t)\leftarrow\mathcal{B}\left(\vec{S}^{(x_{m,i}:z)}\right)}}\left[\mathbf{1}_{\{t=m\}}\cdot h(x_{m,i})\right]+\delta\right)$$

Every $\vec{S}^{(x_{m,i}:z)}$ above contains iid samples from $\mathfrak{D}$, and $x_{m,i}$ is independent of $\vec{S}^{(x_{m,i}:z)}$. Thus,

$$= \frac{1}{n}\sum_{i=1}^{n}\sum_{m=1}^{T}\left(e^{\epsilon} \mathop{\mathbb{E}}_{\substack{z,\vec{S}\sim\mathfrak{D}\\(h,t)\leftarrow\mathcal{B}(\vec{S})}}\left[\mathbf{1}_{\{t=m\}}\cdot h(z)\right]+\delta\right) = e^{\epsilon} \mathop{\mathbb{E}}_{\substack{z,\vec{S}\sim\mathfrak{D}\\(h,t)\leftarrow\mathcal{B}(\vec{S})}}[h(z)] + T\delta$$

$$= e^{\epsilon} \mathop{\mathbb{E}}_{\substack{\vec{S}\sim\mathfrak{D}\\(h,t)\leftarrow\mathcal{B}(\vec{S})}}[h(\mathfrak{D})] + T\delta \leq \mathop{\mathbb{E}}_{\substack{\vec{S}\sim\mathfrak{D}\\(h,t)\leftarrow\mathcal{B}(\vec{S})}}[h(\mathfrak{D})] + 2\epsilon + T\delta$$

Claim C (=Theorem 8): Let W be an (ε,δ)-DP-stable algorithm that outputs a query $q: X \to \{0,1\}$. Let $S \sim \mathcal{D}^n$ and denote $q \leftarrow W(S)$. Then with high probability it holds that

$$q(S) \approx q(\mathcal{D})$$

Explanation: Suppose towards contradiction that with some noticeable probability β, the outcome of $W(S)$ is a query q such that $|q(S) - q(\mathcal{D})|$ is large. We can use W to construct a DP-stable algorithm B that contradicts Claim B:

Algorithm B:

Input: $S_1, \ldots, S_T$ for $T \approx \frac{1}{\beta}$, where each $S_t \sim \mathcal{D}^n$

(1) For every $1 \leq t \leq T$ run $q_t \leftarrow W(S_t)$
   * According to the assumption by contradiction on W, and based on the choice of T, with high probability, there exists an index t such that $|q_t(S_t) - q_t(\mathcal{D})|$ is large.

(2) Choose such an index t using algorithm NoisyAgrmax and return (q_t, t).
   * With high probability, we will get that $|q_t(S_t) - q_t(\mathcal{D})|$ is large, which will contradict the expected bound given in Claim B.

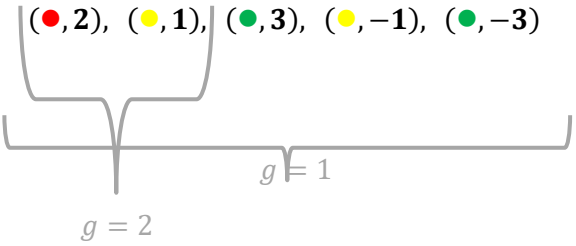
It can be shown that Algorithm B is DP-stable (with approximately the same stability parameters as W), and it can also be shown that, with high probability (and in expectation), the empirical value of the query it returns is very far from its value over the distribution. This contradicts Claim B and therefore disproves the assumption by contradiction that such a W exists.

Adaptive Streaming

Definition (Non-Adaptive Streaming Model):

- A stream of length $\boldsymbol{m}$ over domain $[\boldsymbol{n}]$ is a sequence of updates $\big((\boldsymbol{u_1}, \boldsymbol{\Delta_1}), \dots, (\boldsymbol{u_m}, \boldsymbol{\Delta_m})\big)$
- Here $\boldsymbol{u_i} \in [\boldsymbol{n}]$ is the $\boldsymbol{i}$th item and $\boldsymbol{\Delta_i} \in \mathbb{Z}$ is its weight
- Let $\boldsymbol{g}$: $([\boldsymbol{n}] \times \mathbb{Z})^* \to \mathbb{R}$ be a function
- At time $\boldsymbol{i}$ we obtain $(\boldsymbol{u_i}, \boldsymbol{\Delta_i})$ and need to output $\boldsymbol{z_i} \in (\boldsymbol{1} \pm \boldsymbol{\alpha}) \cdot \boldsymbol{g}\big((\boldsymbol{u_1}, \boldsymbol{\Delta_1}), \dots, (\boldsymbol{u_i}, \boldsymbol{\Delta_i})\big)$
- **Requirement: Sublinear space (we assume $\boldsymbol{n} \gg \boldsymbol{m}$)**

Example: Suppose the function g counts the number of distinct elements in the stream. Then:



(Here, the elements in the stream arrive from left to right.)

Another Example:

- *Every item in the stream is a pair $(\boldsymbol{u_i}, \boldsymbol{\Delta_i})$ where $\boldsymbol{u_i} \in \mathbb{R}^{\boldsymbol{n}}$ is a standard basis vector and $\boldsymbol{\Delta_i} \in \mathbb{R}$ is its weight*
- *At every time step $\boldsymbol{i}$, the goal is to estimate $\left\|\boldsymbol{f}^{(\boldsymbol{i})}\right\|_{\boldsymbol{2}}^{\boldsymbol{2}}$ for $\boldsymbol{f}^{(\boldsymbol{i})} = \boldsymbol{\Delta_1} \cdot \boldsymbol{u_1} + \cdots + \boldsymbol{\Delta_i} \cdot \boldsymbol{u_i}$*

The AMS algorithm [Alon, Matias, Szegedy 1996]:

1. *Let $\boldsymbol{A}$ be $\boldsymbol{t}$x$\boldsymbol{n}$ matrix with entries uniformly in $\{\pm\boldsymbol{1}\}$*
2. *Initiate $\boldsymbol{y} = \vec{\boldsymbol{0}} \in \mathbb{R}^{\boldsymbol{t}}$*
3. *For $\boldsymbol{i} = \boldsymbol{1}, \boldsymbol{2}, \dots, \boldsymbol{m}$ do:*
   - *Obtain the next update vector $\boldsymbol{v_i} = \boldsymbol{\Delta_i} \cdot \boldsymbol{u_i}$*
   - *Let $\boldsymbol{y} \leftarrow \boldsymbol{y} + \boldsymbol{A} \cdot \boldsymbol{v_i}$*
   - *Output estimation $\boldsymbol{z_i} = \frac{\boldsymbol{1}}{\boldsymbol{t}} \cdot \|\boldsymbol{y}\|_{\boldsymbol{2}}^{\boldsymbol{2}}$*

Analysis: Denote the ℓ-th row in the matrix A as a_ℓ. Observe that

$$z_i = \frac{1}{t} \cdot \|A \cdot v_1 + \cdots + A \cdot v_i\|_2^2 = \frac{1}{t} \cdot \left\|A \cdot f^{(i)}\right\|_2^2 = \frac{\left(a_1 \cdot f^{(i)}\right)^2 + \cdots + \left(a_t \cdot f^{(i)}\right)^2}{t}$$

Now, for every fixture of $f \in \mathbb{R}^n$ and every $\ell \in [t]$ we have

$$\mathbb{E}[(a_\ell \cdot f)^2] = \mathbb{E}\left[\left(\sum_{j\in[n]} a_{\ell,j} \cdot f_j\right)^2\right] \underset{(pairwise)}{=} \sum_{j\in[n]} \mathbb{E}\left[\left(a_{\ell,j} \cdot f_j\right)^2\right] = \sum_{j\in[n]} \mathbb{E}\left[a_{\ell,j}^2\right] \cdot f_j^2 = \sum_{j\in[n]} f_j^2 = \|f\|_2^2$$

Where the equality marked in red follows as for $j \neq j'$ we have $\mathbb{E}\left[a_{\ell,j} \cdot a_{\ell,j'}\right] = \mathbb{E}\left[a_{\ell,j}\right] \cdot \mathbb{E}\left[a_{\ell,j'}\right] = 0$ because $a_{\ell,j}$ and $a_{\ell,j'}$ are independent.

So $(a_\ell \cdot f)^2$ is an unbiased estimator for $\|f\|_2^2$. But what is the variance of this estimator? By definition,

$$\mathrm{Var}[(a_\ell \cdot f)^2] = \mathbb{E}[(a_\ell \cdot f)^4] - (\mathbb{E}[(a_\ell \cdot f)^2])^2$$

Let's examine each of these two expressions:

$$\mathbb{E}[(a_\ell \cdot f)^4] = \mathbb{E}\left[\left(\sum_{j\in[n]} a_{\ell,j} \cdot f_j\right)^4\right] \underset{(4wise)}{=} \sum_{j\in[n]} f_j^4 + 6\sum_{j\neq j'} f_j^2 \cdot f_{j'}^2$$

$$(\mathbb{E}[(a_\ell \cdot f)^2])^2 = \left(\sum_{j\in[n]} f_j^2\right)^2 = \sum_{j\in[n]} f_j^4 + 2\sum_{j\neq j'} f_j^2 \cdot f_{j'}^2$$

And so

$$\mathrm{Var}[(a_\ell \cdot f)^2] = 4\sum_{j\neq j'} f_j^2 \cdot f_{j'}^2 \leq 2\left(\sum_{j\in[n]} f_j^2\right)^2 = 2 \cdot \|f\|_2^4$$

The above calculation was for a single row. Our algorithm returns the average of this statistics over the t rows of A. This average does not change the expectation of our estimate (which is $\|f\|_2^2$), but it reduces the variance significantly. Suppose that the rows in the matrix are independent. Then,

$$\mathrm{Var}\left[\frac{\sum_{\ell=1}^t (a_\ell \cdot f)^2}{t}\right] = \frac{1}{t^2} \cdot \mathrm{Var}\left[\sum_{\ell=1}^t (a_\ell \cdot f)^2\right] \underset{\substack{independent\\ rows}}{=} \frac{1}{t^2} \cdot \sum_{\ell=1}^t \mathrm{Var}[(a_\ell \cdot f)^2] \leq \frac{2 \cdot \|f\|_2^4}{t}$$

Therefore, by Chebyshev's inequality, for every $\alpha > 0$,

$$\Pr\left[\left|\frac{\sum_{\ell=1}^t (a_\ell \cdot f)^2}{t} - \|f\|_2^2\right| > \alpha \cdot \|f\|_2^2\right] \leq \frac{\mathrm{Var}\left[\frac{\sum_{\ell=1}^t (a_\ell \cdot f)^2}{t}\right]}{\alpha^2 \cdot \|f\|_2^4} \leq \frac{2}{t \cdot \alpha^2}$$

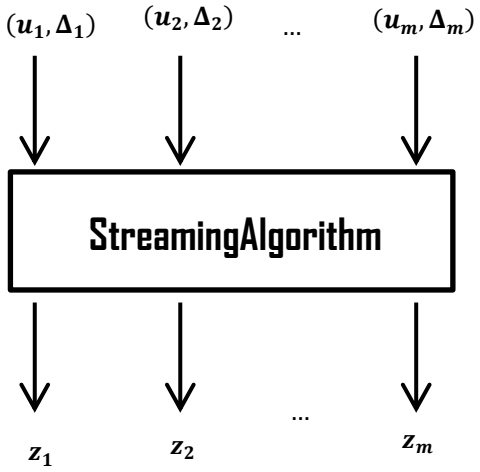
Setting $t = \frac{2}{\alpha^2 \beta}$ we have that

$$\Pr\left[\left|\frac{\sum_{\ell=1}^{t}(a_\ell \cdot f)^2}{t} - \|f\|_2^2\right| > \alpha \cdot \|f\|_2^2\right] \leq \beta$$

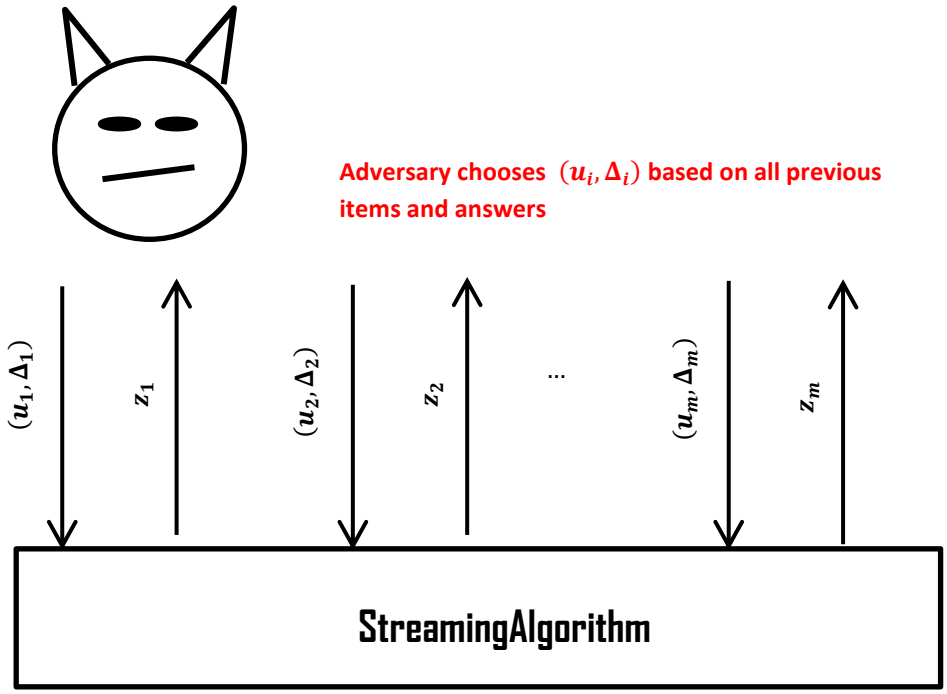
This shows that for every i, with probability at least $1-\beta$, the i-th answer z_i is accurate to within a multiplicative factor of $(1 \pm \alpha)$.

Note: For the analysis, we had to assume that the vector f is <u>fixed</u> in advance. Generally, in the non-adaptive streaming model, we assume that the entire stream is predetermined (but the algorithm processes the elements one by one). Illustration:

$\big((u_1, \Delta_1), \ldots, (u_m, \Delta_m)\big)$ = ***fixed stream (unknown to the algorithm)***



In contrast, in the adaptive model, the elements in the stream are not predetermined. Instead, they are determined during the execution by an "adversary" who observes the responses the algorithm has provided so far. Illustration:



What is the challenge now? *The elements in the stream are chosen based on the previous responses of the algorithm, meaning they can depend on the algorithm's internal randomness. These dependencies disrupt the guarantees of most existing streaming algorithms.*

Definition (Adaptive Streaming Model):

- Fix a function g: $([n] \times \mathbb{Z})^* \to \mathbb{R}$
- Two-player game between a (randomized) **StreamingAlgorithm** and an **Adversary**
- In the ith round:
  1. The **Adversary** chooses an update (u_i, Δ_i) for the stream, which can depend on all previous stream updates and outputs of **StreamingAlgorithm**
  2. The **StreamingAlgorithm** processes the new update and outputs its current response z_i
- The goal of the **Adversary** is to make the **StreamingAlgorithm** output an incorrect response z_i at some point i

Adversary for the AMS Sketch

Recall AMS sketch

- Random matrix $A \in \{\pm 1\}^{t \times n}$
- After the ith update, respond with $\frac{1}{t}\|A \cdot f^{(i)}\|_2^2 = \left\|\frac{1}{\sqrt{t}} A \cdot f^{(i)}\right\|_2^2$ where

$$f^{(i)} = \Delta_1 \cdot u_1 + \cdots + \Delta_i \cdot u_i$$

The attack

- Set $w \leftarrow C \cdot \sqrt{t} \cdot e_1$
- For $i = 2, 3, \ldots, m = O(t)$ do
  1. $\mathbf{old} \leftarrow \left\|\frac{1}{\sqrt{t}} A \cdot w\right\|_2^2$
  2. $w \leftarrow w + e_i$
  3. $\mathbf{new} \leftarrow \left\|\frac{1}{\sqrt{t}} A \cdot w\right\|_2^2$
  4. If $\mathbf{new} > \mathbf{old}$ then $w \leftarrow w - e_i$

Analysis

- At all times $\|w\|_2^2 \geq C^2 \cdot t$ by init

  $\Longrightarrow$ Suffices to show that $\left\|\frac{1}{\sqrt{t}} A \cdot w\right\|_2^2$ drops below $C^2/2 \cdot t$
- $\mathbf{new}_i = \left\|\frac{1}{\sqrt{t}} A \cdot (w + e_i)\right\|_2^2 = \left\|\frac{1}{\sqrt{t}} A \cdot w\right\|_2^2 + \left\|\frac{1}{\sqrt{t}} A \cdot e_i\right\|_2^2 + 2\left\langle \frac{1}{\sqrt{t}} Aw, \ \frac{1}{\sqrt{t}} Ae_i \right\rangle$

  $= \mathbf{old}_i + 1 + 2\left\langle \frac{1}{\sqrt{t}} Aw, \ \frac{1}{\sqrt{t}} Ae_i \right\rangle$
- So, $\mathbf{new}_i - \mathbf{old}_i \approx 2\left\langle \frac{1}{\sqrt{t}} Aw, \ \frac{1}{\sqrt{t}} Ae_i \right\rangle$

This inner product $\left\langle \frac{1}{\sqrt{t}}\boldsymbol{Aw},\ \ \frac{1}{\sqrt{t}}\boldsymbol{Ae_i} \right\rangle$ is a random variable, because $\boldsymbol{A}$ is random. First observe that this random variable is symmetric. Why? Because $\boldsymbol{Ae_i}$ is the i-th column in $\boldsymbol{A}$, and because $\boldsymbol{Aw}$ depend only on columns with indices smaller than i. So $\boldsymbol{Aw}$ and $\boldsymbol{Ae_i}$ are independent vectors. Thus, for every $\boldsymbol{y}$, the probability of that $\boldsymbol{Ae_i} = \boldsymbol{y}$ is equal to the probability that $\boldsymbol{Ae_i} = -\boldsymbol{y}$ (even given $\boldsymbol{Aw}$), showing that this inner product is symmetric.

It can be shown that the standard deviation of the inner product is constant, showing that "in the eyes of the AMS algorithm" every iteration shrinks the norm by a constant with probability $\approx \boldsymbol{1/2}$.