ALGORITHMS FOR ANSWERING LINEAR QUERIES

Sasho Nikolov (U of Toronto)
Gerome Miklau (UMass Amherst)
Ryan McKenna (Umass Amherst)

OUTLINE

Basic Notions

Data Independent Mechanisms

Gaussian Noise + Projection

Learning the database

DATABASE MODEL & COUNTING QUERIES

ID #	Gender	Education	Age
15737	M	BSc	24
13555	F	PhD	35
63323	F	High School	20
12984	M	High School	19
16750	M	MSc	27
46188	M	BSc	40

Data Universe $\mathfrak{U}$, i.e. set of all possible database rows

- $\mathfrak{U}$ = {possible IDs} × {M, F} × {High School, BSc, MSc, PhD} × {0, …, 150}

Database $X = (x^1, \dots, x^n) \in \mathfrak{U}^n$, x^i are the *database rows*

- *each rows corresponds to the data of one person*

Predicate $q: \mathfrak{U} \to \{0,1\}$

- E.g. IsMale$(x) = 1 \iff$ Gender attribute of x is Male.
- Weighted version: $q: \mathfrak{U} \to [0,1]$

Counting query: $q(X) = \sum_{i=1}^{n} q(x^i)$, i.e. number of db rows satisfying q.

Normalized counting query: $q(X) = \frac{1}{n}\sum_{i=1}^{n} q(x^i)$

QUERY WORKLOAD

2017 US Census

Subject	New York city, New York Estimate
SEX AND AGE	
Total population	8,560,072
18 years and over	6,765,428
Male	3,164,326
Female	3,601,102
Sex ratio (males per 100 females)	87.9
65 years and over	1,168,268
Male	475,903
Female	692,365

Normalized

Query Workload: a collection $Q = \{q_1, \dots, q_k\}$ of counting queries

$$Q(X) = \begin{pmatrix} q_1(X) \\ q_2(X) \\ \vdots \\ q_k(X) \end{pmatrix}$$

HISTOGRAM

The *histogram* of $X = (x^1, \dots, x^n)$ is a vector $h \in \mathbb{N}^{\mathfrak{U}}$:

$$\forall x \in \mathfrak{U}: \ \ h_x = |\{i : x^i = x\}|$$

- i.e. h_x is the number of copies of x in X

E.g. $\mathfrak{U} = \{0,1\}^3$, $X = (001,100,101,111,001,101)$:

$h =($

000	001	010	011	100	101	110	111
0	2	0	0	1	2	0	1

$)$

$\|h\|_1 = \sum_{x \in \mathfrak{U}} |h_x| = n$

If X and X' are neighboring, then $\|h - h'\|_1 \leq 1$.

QUERY MATRIX

We can encode a query workload $\mathcal{Q}$ by a matrix $W \in [0,1]^{\mathcal{Q}\times\mathfrak{U}}$:

$$\forall q \in \mathcal{Q}, \forall x \in \mathfrak{U}: \;\; W_{q,x} = q(x)$$

E.g. $\mathfrak{U} = \{0,1\}^3$ and $\mathcal{Q}$ are 1-way marginals: $q_i(x) = i$-th bit of x

$W =$

	000	001	010	011	100	101	110	111
q_1	0	0	0	0	1	1	1	1
q_2	0	0	1	1	0	0	1	1
q_3	0	1	0	1	0	1	0	1

Then the workload answers are the product $\mathcal{Q}(X) = Wh$:

$$q(X) = \sum_{i=1}^{n} q\left(x^i\right) = \sum_{x\in\mathfrak{U}} q(x)\left|\left\{i: x^i = x\right\}\right| = (Wh)_q$$

MEASURING ERROR

Worst-Case Error of a mechanism $\mathcal{M}$:

$$\mathrm{err}^{\infty}(Q, \mathcal{M}, n) = \max_{X \in \mathfrak{U}^n} \mathbb{E} \max_{q \in Q} |q(X) - \mathcal{M}(Q, X)_q|$$

i.e. $\mathrm{err}^{\infty}(W, \mathcal{M}, n) = \max_{\|h\|_1 \leq n} \mathbb{E}\|Wh - \mathcal{M}(W, h)\|_{\infty}$

(Expectations over the randomness of $\mathcal{M}$.)

Mean Squared Error of a mechanism $\mathcal{M}$:

$$\mathrm{err}^{2}(Q, \mathcal{M}, n) = \max_{X \in \mathfrak{U}^n} \sqrt{\mathbb{E} \frac{1}{|Q|} \sum_{q \in Q} |q(X) - \mathcal{M}(Q, X)_q|^2}$$

i.e. $\mathrm{err}^{2}(W, \mathcal{M}, n) = \max_{\|h\|_1 \leq n} \left(\mathbb{E} \frac{1}{|Q|} \|Wh - \mathcal{M}(W, h)\|_2^2\right)^{1/2}.$

TYPES OF MECHANISMS

Postprocessing a data independent mechanism can introduce data dependence.

Data and Workload Independent: noise only depends on the sensitivity

- Gaussian and Laplace noise mechanisms

Data Independent, adapted to the Workload: noise optimized for the workload

- Matrix Mechanism

Adapted to the Data: mechanism learns the database

- Private Multiplicative Weights Mechanism

OUTLINE

Basic Notions

Data Independent Mechanisms

Gaussian Noise + Projection

Learning the database

GAUSSIAN NOISE MECHANISM

ℓ_2 *Sensitivity:* $\Delta_2(Q) = \max_{X \sim X'} \|Q(X) - Q(X')\|_2$

Equivalently: $\Delta_2(W) = \max\{\|Wh - Wh'\|_2 : \|h - h'\|_1 \leq 1\}$

$= \max\{\|Wv\|_2 : \|v\|_1 \leq 1\}$

This is just the largest ℓ_2-norm of a column of W.

- $\Delta_2(Q) = \Delta_2(W) \leq \sqrt{|Q|}$.

Gaussian Noise Mechanism [Dinur Nissim 03, Dwork Nissim 04, DMNS 06]

$\mathcal{M}_{\mathrm{gm}}(W, h) = Wh + G, \quad G_i \sim N(0, \sigma_{\epsilon,\delta}^2 \Delta_2(W)^2)$

$\sigma_{\epsilon,\delta} = \Theta\left(\epsilon^{-1}\sqrt{\log(1/\delta)}\right).$

$\Delta_2(W) = \sqrt{3}$

0	0	0	0	1	1	1	1
0	0	1	1	0	0	1	1
0	1	0	1	0	1	0	1

HOW WELL DOES IT DO?

Will ignore dependence on ϵ and δ from now on.

On any workload W of k queries:

$$\mathrm{err}^{\infty}\left(W, \mathcal{M}_{\mathrm{gm}}, n\right) \lesssim \Delta_2(W)\sqrt{\log(k)} \leq \sqrt{k\log(k)}.$$

[Bun, Ullman, Vadhan 14] Optimal for 1-way marginals on d-dimensional data

The same query repeated k times?

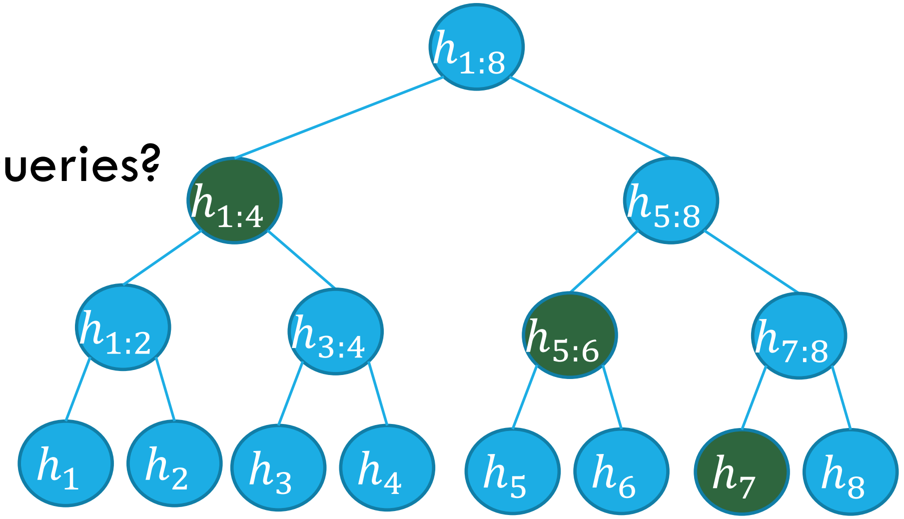
Threshold Queries: $\mathfrak{U} = \{1, \dots, N\}$, $Q = \{q_t\}$, $q_t(x) = 1$ if $x \leq t$.

- Can do $\Theta(\log(N)^{1.5})$

1	0	0	0	0
1	1	0	0	0
1	1	1	0	0
1	1	1	1	0
1	1	1	1	1

FACTORING THE QUERIES

$q_7(X) = h_{1:4} + h_{5:6} + \mathrm{h}_7$



How do you beat the Gaussian Mechanism on Threshold Queries?

Factor workload as: $W = RA$

Reconstruction Matrix

Strategy Matrix

$\mathcal{M}(h) = R(Ah + G)$, where $G_i \sim N(0, \sigma^2_{\epsilon,\delta} \Delta_2(A)^2)$

Error vector: $\mathcal{M}(h) - Wh = RG$

- $(RG)_i \sim N(0, \sigma^2_{\epsilon,\delta}\, \Delta_2(A)^2 \|r^i\|_2^2)$, where $r^i = i$-th row of R

Threshold queries: $\Delta_2(A)^2, \max_i \|r^i\|_2^2 = \Theta(\log N)$

- $\Theta(\log N)$ noise per query

ERROR BOUNDS

$\mathcal{M}(h) = R(Ah + G)$, where $W = RA$

Error vector: $\mathcal{M}(h) - Wh = RG \sim N(0, \sigma_{\epsilon,\delta}^2 \Delta_2(A)^2 \cdot RR^\intercal)$

Error for k queries:

Max row norm

Max column norm

$$\mathrm{err}^\infty(W, \mathcal{M}, n) \lesssim \rho(R)\Delta_2(A)\sqrt{\log(k)}$$

$$\rho(R) = \max_i \left\| r^i \right\|_2 = \max_i \sqrt{(RR^\intercal)_{ii}}$$

MATRIX MECHANISM [LI, MIKLAU, MCGREGOR, RASTOGI 10]

Max row norm

Max column norm

$$\mathrm{err}^{\infty}(W, \mathcal{M}, n) \lesssim \rho(R)\Delta_2(A)\sqrt{\log(k)}$$

How to choose R and A?

Just optimize the above error bound:

$$\gamma(W) = \min\{\rho(R)\Delta_2(A) : W = RA\}$$

$\mathcal{M}_{\mathrm{mm}}(h) = R(Ah + G)$, where R and A achieve $\gamma(W)$.

Then

$$\mathrm{err}^{\infty}(W, \mathcal{M}_{\mathrm{mm}}, n) \lesssim \gamma(W)\sqrt{\log(k)}.$$

Similarly for mean squared error.

GEOMETRIC INTERPRETATION

$B_1^N = \{v \colon \|v\|_1 \leq 1\}$

For neighboring databases with histograms h, h',

$$Wh - Wh' = W(h - h') \in WB_1^N = \mathrm{conv}\{\pm w_1, \pm w_2, \dots, \pm w_N\}$$

where $w_1, w_2, \dots, w_N$ are the columns of W.

$K_W = WB_1^N$ is called the *sensitivity polytope*

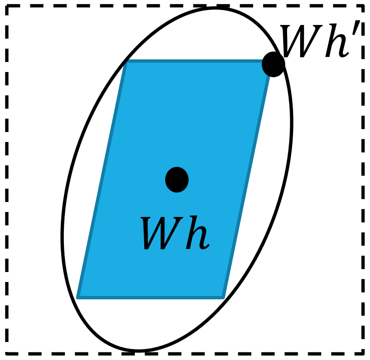
- $\Delta_2(W)$ = radius of smallest Euclidean ball containing K_W

If $W = RA$, then the ellipsoid $E = RB_2^k\big(0, \Delta_2(A)\big)$ contains K_W

The Matrix Mechanism finds the smallest ellipsoid that contains K_W

- Smallest = contained in the smallest cube.

1	0
1	1



OPTIMIZATION PROBLEM

We need to solve: $\gamma(W) = \min\{\rho(R)\Delta_2(A): W = RA\}$

Observation: we can always replace R and A by tR and A/t.

- Can assume that $\rho(R) = \Delta_2(A) = \sqrt{\gamma(W)}$.

Semidefinite Program for $\gamma(W)$:

- $r^i = i$-th row of R
- $a_j = j$-th column of A
- optimal $t = \gamma(W)$

$$\begin{array}{ll} \min t & \\ \text{subject to} & \\ r^i \cdot a_j = w_{ij} & \forall i,j \\ r^i \cdot r^i \leq t & \forall i \\ a_j \cdot a_j \leq t & \forall j \end{array}$$

$W = RA$

$\rho(R)^2 \leq t$

$\Delta_2(R)^2 \leq t$

OPTIMALITY OF THE MATRIX MECHANISM

Best achievable error: $\mathrm{opt}_{\epsilon,\delta}(W,n) = \inf\{\mathrm{err}^{\infty}(W,\mathcal{M},n) : \mathcal{M} \text{ is } (\epsilon,\delta)\text{−DP}\}$

Reconstruction Attack

Matrix Mechanism

[Nikolov, Talwar, Zhang 13]

$$\frac{\gamma(W)}{\log(k)} \lesssim \sup_{n} \mathrm{opt}_{\epsilon,\delta}(W,n) \lesssim \gamma(W)\sqrt{\log(k)}$$

Proof sketch:

- If $\gamma(W)$ is large, then W has a submatrix with large minimum singular value
- [Dinur Nissim 03] A mechanism for W with error too small relative to the smallest singular value allows a reconstruction attack.

Proof shows that

$$\frac{\gamma(W)}{\log(k)} \lesssim \mathrm{opt}_{\epsilon,\delta}(W,k/\epsilon)$$

OUTLINE

Basic Notions

Data Independent Mechanisms

Gaussian Noise + Projection

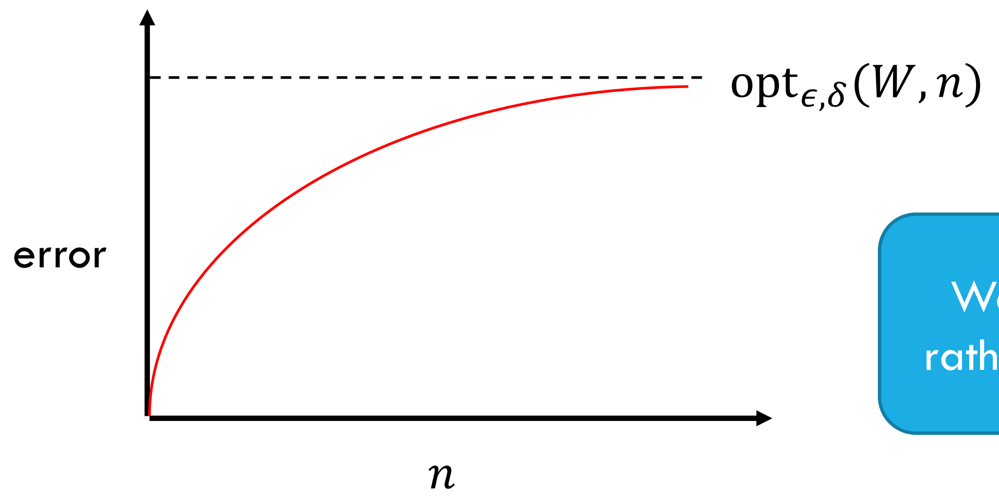
Learning the database

SMALL DATABASES

2-way marginals: $\mathfrak{U} = \{0,1\}^d$, $q_{i,j}(x) = x_i \wedge x_j$

- $\gamma(W) = \Theta(d)$ so the Matrix Mechanism has error $\mathrm{err}^\infty(W, \mathcal{M}_{\mathrm{mm}}, n) \approx d\sqrt{\log(d)}$
- Can achieve error $\sqrt{n}d^{\frac{1}{4}}\sqrt{\log(d)}$: MM is suboptimal if $n \ll d^{1.5}$.

The Matrix Mechanism can be suboptimal for small n.



Want to match the entire curve, rather than just the limit as $n \to \infty$

PROJECTION MECHANISM

Recall: if database has size n, then $\|h\|_1 \leq n$.

So, $Wh \in nWB_1^N = nK_W$

Gaussian noise mechanism: $\mathcal{M}_{\mathrm{gm}}(W,h) = \tilde{Y} = Wh + G$

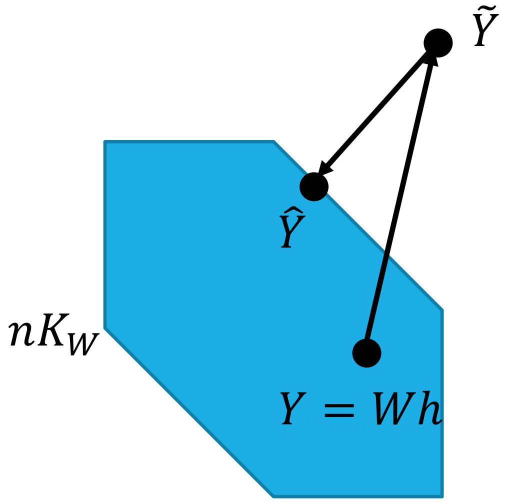
- If n is small, then very likely $\tilde{Y} \notin nK_W$
- Postprocess to "bring it back"!

<u>Projection Mechanism</u> [Nikolov Talwar Zhang 13]

$\tilde{Y} = \mathcal{M}_{\mathrm{gm}}(W,h)$

$\mathcal{M}_{\mathrm{pr}}(W,h) = \hat{Y} = \arg\min\left\{\left\|\tilde{Y} - z\right\|_2 : z \in nK_W\right\}$

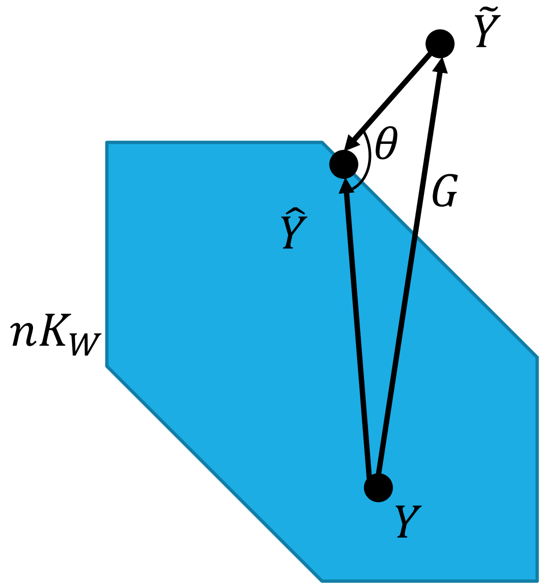
Just postprocessing: does not affect privacy.



ERROR BOUND

Using some high school geometry + probability:

$$\mathrm{err}^2(W, \mathcal{M}_{\mathrm{pr}}, n) \quad \lesssim \sqrt{n}(\log|\mathfrak{U}|)^{\frac{1}{4}}$$



[Bun, Ullman, Vadhan 14] Optimal for 2-way marginals

Remarks:

- Projection does not affect privacy, but can improve error for small n
- Projection turns a data independent mechanism into a data dependent one

IMPLICIT QUERIES

Often queries of interest are defined compactly: e.g. r-way marginals

- We want algorithms that run much faster than $O(|\mathfrak{U}|)$ time.
- [Dwork, Naor, Reingold, Rothblum, Vadhan 09] Computationally hard for artificial queries

Test case: 2-way marginals on d-dimensional data in time $\mathrm{poly}(d, n)$

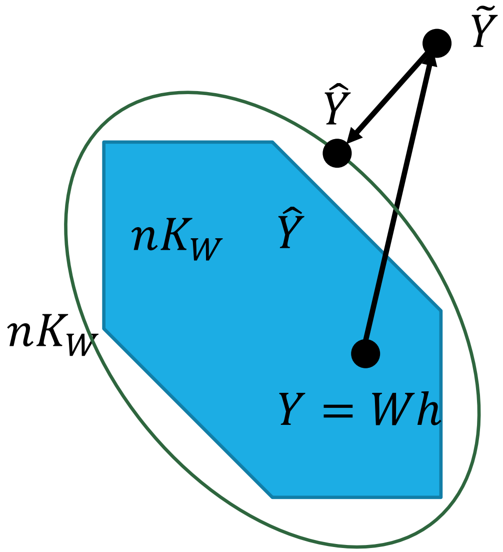
- We will use the Projection Mechanism to get mean sq error $O(\sqrt{n}d^{1/4})$

Projecting on nK_W reduces to solving $\max\{c^\intercal z : z \in K_W\}$ for arbitrary c

- NP-hard for 2-way marginals!

[Dwork, Nikolov, Talwar 14] Project on some $\mathrm{n}L$, $K_W \subseteq L \subseteq O(1) \cdot K_W$

- Projecting on L is efficient: SDP relaxation of K_W
- Error is the same as projecting on K_W, up to constants



Open: Is there an algorithm with $O(\sqrt{n}d^{1/4})$ error that computes 3-way marginals on d-dimensional data in time $\mathrm{poly}(d, n)$?

COARSE PROJECTION [BLASIOK, BUN, NIKOLOV, STEINKE 19]

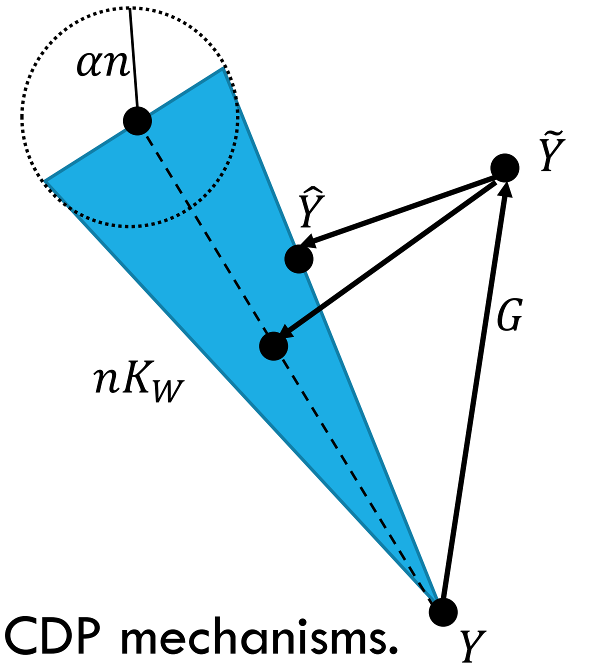
Suppose we want error at most αn.

Take an αn-net:

- Points $v_1, \dots, v_N \in K_W$ such that any vertex of K_W is within αn from some v_i

Project on $\mathrm{conv}\{v_1, \dots, v_N\}$

- Can have much smaller mean width than K_W
- Y can be outside but this cannot introduce more than αn error
- Error after projection is smaller



Achieves error $n/100$ with nearly the smallest possible n among CDP mechanisms.

- Open: similar guarantee for approximate DP.

PROJECTION AND MATRIX MECHANISM

[Nikolov Talwar Zhang 13, Nikolov 15] Combine Matrix Mechanism and Projection Mechanism:

- Optimize the upper bound on error over factorizations $W = RA$
- Add noise: $\tilde{Y} = R(Ah + G)$
- Project $\tilde{Y}$ to nK_W

In fact projection is slightly modified.

Mechanism with mean sq. error $\lesssim \left(\log(\mathrm{n})\log(|\mathfrak{U}|)\right)^{\frac{1}{4}} \cdot \mathrm{opt}_{\epsilon,\delta}(W, n)$ for every n.

- Open: An efficient mechanism whose error for every n is $\lesssim \log(nk)^{O(1)} \cdot \mathrm{opt}_{\epsilon,\delta}(W, n)$
- Open: Similar guarantee for worst-case error?

OUTLINE

Basic Notions

Data Independent Mechanisms

Gaussian Noise + Projection

Learning the database

QUERY RELEASE AS LEARNING [BLUM, LIGETT, ROTH 08]

We know how to answer few queries on large database X.

Given a large workload Q, can we generalize from few query answers to all of Q?

A change in perspective: X is a *function* from queries to query answers

- $X: Q \to \mathbb{R}$ defined by $X(q) = q(X)$ for every $q \in Q$

Learning problem: given a few examples $(q, X(q))$, learn $X: Q \to \mathbb{R}$

- Reduces answering many queries to answering few queries.

BOOSTING AVERAGE ERROR GUARANTEES

Mean squared error η: $\frac{1}{|Q|}\sum_{q\in Q}\left|q(X)-\mathcal{M}(Q,X)_q\right|^2 \leq \eta^2$

Chebyshev: for all but $|Q|/4$ queries q, $\left|q(X)-\mathcal{M}(Q,X)_q\right| \leq 2\eta$

I.e. we can approximate $\mathrm{X}(q) = q(X)$ on most of Q: we have *weakly learned* X

We want to *strongly learn* X, i.e. learn it on all of Q

[Freund Schapire 95] Boosting reduces strong learning to weak learning

[Dwork Rothblum Vadhan 10] Private Boosting reduces worst-case error to mean sq. error

- Running a base mechanism with good mean sq. error $O(\log|Q|)$ times gives a mechanism with good worst case error.

Gives an efficient algorithm for 2-way marginals with optimal worst case error

THE DATABASE AS A DISTRIBUTION

The *normalized histogram* $p = \frac{1}{n} h$ is a probability distribution on $\mathfrak{U}$

- $\forall x \in \mathfrak{U}$: $p_x = \frac{|\{i : x^i = x\}|}{n}$

$Wp = \frac{1}{n} Wh$ are the *normalized* query answers

- Approximating Wp within error $\alpha \Leftrightarrow$ Approximating Wh within error αn

Notation: $q(p) = (Wp)_q = \sum_{x \in \mathfrak{U}} q(x) p_x$

Learning $X \Leftrightarrow$ Learning p

PRIVATE MULTIPLICATIVE WEIGHTS [HARDT, ROTHBLUM 10] [HARDT, LIGETT, MCSHERRY 12]

uniform

Strategy: keep guessing distributions $p^0, p^1, \dots, p^T$

- Privately find a $q \in Q$ such that $|q(p) - q(p^t)| > \alpha$
- If not found, we are done: return current p^t
- If found, update p^t to p^{t+1}

Distinguisher between p and p^t

Sensitivity $\frac{1}{n}$

Finding a distinguishing query: exponential mechanism with score $|q(p) - q(p^t)|$

Update rule:

- $\sigma = \operatorname{sign}\big(q(p) - q(p^t)\big)$
- $\tilde{p}_x^{t+1} = p_x^t \exp\left(\frac{\alpha\sigma q(x)}{2}\right)$ and $p_x^{t+1} = \tilde{p}_x^{t+1} / \sum_{y \in \mathfrak{U}} \tilde{p}_y^{t+1}$

If $q(p^t)$ overshoots, we clamp down the probability of the contributing x.

WHY IT WORKS

$\mathrm{err}^{\infty}(\mathcal{Q}, \mathcal{M}, n) \lesssim \sqrt{n \log(k)}\, (\log |\mathfrak{U}|)^{\frac{1}{4}}$

PMW answers (unnormalized) queries with worst-case error αn if $\mathrm{n} \gtrsim \frac{\sqrt{\log |\mathfrak{U}| \log k}}{\alpha^2}$

Relative entropy $D(p||q) = \sum_{x \in \mathfrak{U}} p_x \log\left(\frac{p_x}{q_x}\right)$ as a potential

Initially $D(p||p^0) \leq \log |\mathfrak{U}|$, and $D(p||q) \geq 0$ always

Every update decreases $D(p||p^t)$ by at least $\frac{\alpha^2}{4}$: no more than $\frac{4 \log|\mathfrak{U}|}{\alpha^2}$ updates

- We need to run the exponential mechanism at most $\frac{4 \log|\mathfrak{U}|}{\alpha^2}$ times

Advanced composition + exponential mechanism analysis

If $\mathrm{n} \gtrsim \frac{\sqrt{\log |\mathfrak{U}| \log k}}{\alpha^2}$, we can identify $q \in \mathcal{Q}$ such that $|q(p) - q(p^t)| > \alpha$ for every update.

WHAT WE DID NOT COVER

Synthetic data

- Projection M and PMW can generate synthetic data
- Dual Query, GANs, etc.

Answering queries online

- PMW can be adapted to queries that arrive online
- Replace exponential mechanism with Sparse Vector

Other ways to answer implicit queries efficiently (e.g. [Thaler Ullman Vadhan 12])

- Approximate $X: Q \to \mathbb{R}$ by a low degree polynomial
- Privately compute coefficients

Information theoretic and computational lower bounds

Applied work and much of the work outside the theory community

- Stay for Gerome!