Local differential privacy

Adam Smith
Penn State
Bar-Ilan Winter School
February 14, 2017
Outline

• Model
  ➢ Implementations

• Question: what computations can we carry out in this model?

• Example: randomized response (again!)
  ➢ SQ computations

• Simulating local algs via SQ
  ➢ An exponential separation

• Averaging vectors

• Heavy hitters: succinct averaging

• Lower bounds: information
  ➢ Example: selection

• Compression

• Learning and adaptivity
Local Model for Privacy

- Person $i$ randomizes their own data, say on their own device
- Requirement: Each $Q_i$ is $(\varepsilon, \delta)$-differentially private.
  - We will ignore $\delta$
  - Aggregator may talk to each person multiple times
  - For every pair of values of person $i$’s data, for all events $T$: 
    \[
    \Pr(R(x) \in T) \leq e^{\varepsilon} \cdot \Pr(R(y) \in T).
    \]
Local Model for Privacy

- Pros
  - No trusted curator
  - No single point of failure
  - Highly distributed

- Cons
  - Lower accuracy
Local differential privacy in practice


https://github.com/google/rappor
Local Model for Privacy

- Open questions
  - Efficient, network-friendly MPC protocols for simulating “exponential mechanism” in local model
  - Interaction in optimization (tomorrow)
  - Other tasks?
Local Model for Privacy

What can and can’t we do in the local model?
Example: Randomized response

• Each person has data $x_i \in \mathcal{X}$
  - Analyst wants to know average of $f: \mathcal{X} \rightarrow \{-1,1\}$ over $x$

• Randomization operator takes $y \in \{-1,1\}$:

\[
Q(y) = \begin{cases} 
+ yC_\epsilon \quad \text{w.p.} \quad \frac{e^\epsilon}{e^\epsilon + 1} \\
- yC_\epsilon \quad \text{w.p.} \quad \frac{1}{e^\epsilon + 1}
\end{cases}
\]

where $C_\epsilon = \frac{e^\epsilon + 1}{e^\epsilon - 1}$.

• Observe:
  - $E(Q(1)) = 1$ and $E(Q(-1)) = -1$.
  - $Q$ takes values in $\{-C_\epsilon, C_\epsilon\}$

• How can we estimate a proportion?
  - $A(x_1, \ldots, x_n) = \frac{1}{n} \sum_i Q(f(x_i))$

• Proposition: $\left| A(x) - \frac{1}{n} \sum_i f(x_i) \right| = O_P \left( \frac{1}{\epsilon \sqrt{n}} \right)$
**SQ algorithms**

- An “SQ algorithm” interacts with a data set by asking a series of statistical queries
  
  - Query: $f : \mathcal{X} \rightarrow [-1,1]$
  
  - Response: $\hat{a} \in \frac{1}{n} \sum_i f(x_i) \pm \alpha$ where $\alpha$ is the error

- Huge fraction of basic learning/optimization algorithms can be expressed in SQ form [Kearns 93]
SQ algorithms

• An “SQ algorithm” interacts with a data set by asking a series of statistical queries
  ➢ “Statistical Query:” $f: \mathcal{X} \rightarrow [-1,1]$
  ➢ Response: $\hat{a} \in \frac{1}{n} \sum_i f(x_i) \pm \alpha$ where $\alpha$ is the error

• Huge fraction of basic learning/optimization algorithms can be expressed in SQ form [Kearns 93]

• **Theorem:** Every sequence of $k$ SQ queries can be computed with local DP with error $\alpha = O \left( \sqrt{\frac{k \log k}{\epsilon^2 n}} \right)$.

• **Proof:**
  ➢ Randomly divide $n$ people into $k$ groups of size $\frac{n}{k}$
  ➢ Have each group answer 1 question.

- Central:
  $O \left( \frac{k}{n\epsilon} \right)$
**SQ algorithms and Local Privacy**

- Every SQ algorithm can be simulated by a LDP protocol.

- Can every centralized DP algorithm be simulated by LDP?
  - No!

- **Theorem**: Every LDP algorithm can be simulated by SQ with polynomial blow-up in $n$.

- **Theorem**: No SQ algorithm can learn parity with polynomially many samples ($n = 2^\Omega(d)$).

- **Theorem**: Centralized DP algorithms can learn parity with $n = O\left(\frac{d}{\epsilon}\right)$ samples.

- Is research on local privacy over?
  - No! Polynomial factors matter…
Outline

• Some stuff we can do
  ➢ Heavy hitters

• Some stuff we cannot do
  ➢ LDP and SQ
    • 1-bit randomizers suffice!
  ➢ Information-theoretic lower bounds
**Histograms**

- Every participant has $x_i \in \{1, 2, \ldots, d\}$.
- Histogram is $h(x) = (n_1, n_2, \ldots, n_d)$ where $n_j = \#\{i : x_i = j\}$
- Straightforward protocol: Map each $x_i$ to indicator vector $e_{x_i}$
  - So $h(x) = \sum_i e_{x_i}$
  - $Q'(x_i)$: Apply $Q(\cdot)$ to each entry of $e_{x_i}$.

**Proposition:** $Q'(\cdot)$ is $\epsilon$-LDP and

$$E \left\| \sum_i Q'(x_i) - h(x) \right\|_\infty \leq \frac{\sqrt{n \log d}}{\epsilon}$$

Succinctness

- Randomized response has optimal error $\frac{\sqrt{n \log d}}{\epsilon}$
  - Problem: Communication and server-side storage $O(d)$
  - How much is really needed?

- **Theorem** [Thakurta et al]: $\tilde{O}(\epsilon \sqrt{n \log d})$ space.

- Lower bound (for large $d$)
  - Have to store all the elements with counts at least $\epsilon \sqrt{\frac{n}{\log d}}$.
  - Each one takes $\log d$ bits.

- **Upper bound idea:**
  - [Hsu, Khanna, Roth ‘12, Bassily, S’15] Connection to “heavy hitters” algorithms from streaming
  - Adapt CountMin sketch of [Cormode Muthukrishnan]
**Succinct “Frequency Oracle”**

- Data structure that allow us to estimate \( n_j \) for any \( j \)
  - Can get whole histogram in time \( O(d) \)

- Select \( k \approx \log(d) \) hash functions \( g_m: [d] \to \left[ \frac{\varepsilon \sqrt{n}}{\log d} \right] \)
  - Divide users into \( k \) groups
  - \( m \)-th group constructs histogram for \( g_m(x_i) \)

- Aggregator stores \( k \) histograms
  - \( \text{count}(j) = \text{median}\{\text{count}_m(j) : m = 1, \ldots, k\} \)
  - Corresponds to ”CountMin” hash [Cormode Muthukrishnan]
Efficient Histograms

• When \( d \) is large, want list of large counts
  ➢ Explicit query for all items: \( O(d) \) time

• Time-efficient protocols with (near-)optimal error exist based on
  ➢ error-correcting codes [Bassily S ‘15]
  ➢ Prefix search (à la [Cormode Muthukrishnan ‘03])
    • “All unattributed heuristics are probably due to Frank McSherry”
      --A. Thakurta
    • Worse error, better space

• Open question: exactly optimal error, optimal space
Other things we can do

• Estimating averages in other norms [DJW ‘13]
  ➢ Useful special cases:
    • Histogram with small $\ell_1$ error (in small domains)
    • $\ell_2$ bounded vectors (problem set)

• Convex optimization [DJW ‘13, S Thakurta Uphadhyay ‘17]
  ➢ Via gradient descent (tomorrow)

• Selection problems [other papers]
  ➢ Find most-liked Facebook page
  ➢ Find most-liked Facebook pages with $\leq k$ likes per user
Outline

• Some stuff we can do
  ➢ Heavy hitters

• Some stuff we cannot do
  ➢ LDP and SQ
    • 1-bit randomizers suffice!
  ➢ Information-theoretic lower bounds
SQ Algorithms simulate LDP protocols

• Roughly:
  Every LDP algorithm with \( n \) data points can be simulated by an SQ algorithm with \( O(n^3) \) data points.

  ✓ Actually a distributional statement: assume that data drawn i.i.d from some distribution \( P \)

• Key piece:
  Transform the randomizer so only 1 bit is sent to aggregator by each participant.
**One-bit randomizer**


- **Theorem:** There is a $\varepsilon$-DP $R'$ such that for every $x$:
  - Conditioned on $B = 1$, output $Z$ distributed as $R(x)$
  - $\Pr(B = 1) = 1/2$

- **Replacing $R$ by $R'$…**
  - Lowers communication from participant to 1 bit;
  - Randomly drops an $1/2$ fraction of data points
  - **No need to send $z$:** Use pseudorandom generator.

---

**Diagram:**

- Participant $x \rightarrow R \rightarrow$ Aggregator
  - $z \sim R(0)$
  - $b \in \{0, 1\}$
  - Outputs $z$ iff $b = 1$
Proof

- **Algorithm** $R'(x, z)$:
  - Compute $p_{x,z} = \frac{1}{2} \cdot \frac{\Pr(R(x) = z)}{\Pr(R(0) = z)}$
  - Return $B = 1$ with probability $p_{x,z}$

- Notice that $p$ is always in $\left[\frac{e^{-\epsilon}}{2}, \frac{e^{\epsilon}}{2}\right]$, so $R'$ is $\epsilon$-DP

- $\Pr($select $z$ and $B = 1)$
  $$= \frac{1}{2} \Pr(R(0) = z) \cdot \frac{\Pr(R(x) = z)}{\Pr(R(0) = z)} = \frac{1}{2} \Pr(R(x) = z)$$

- So $\Pr(B = 1) = \frac{1}{2}$ and $Z|_{B=1} \sim R(x)$. 

Participant $x$ → $R'$, $z \sim R(0)$, $b \in \{0, 1\}$ → Aggregator Outputs $z$ iff $b = 1$
Connection to SQ

• An SQ query can evaluate the average of $p_{x_i,z}$ over a large set of data points $x_i$

• When $x_1, \ldots, x_n$ drawn i.i.d. from $P$, we can sample $Z \sim R(X)$ where $X \sim P$
  
  $$E_x(p_{x,z}) = \frac{1}{2} \cdot \frac{\Pr(R(X) = z \text{ where } X \sim P)}{\Pr(R(0) = z)}$$

• This allows us to simulate each message to the LDP algorithm.
Information-theoretic lower bounds

• As with $(\epsilon, 0)$-DP, lower bounds for $(\epsilon, \delta)$-DP are relatively easy to prove via packing arguments.

• For local algorithms, easier to use information-theoretic framework [BNO’10, DJW’13]
  ✓ Applies to $\delta > 0$ case.

• Idea: Suppose $X_1, \ldots, X_n \sim P$ i.i.d., show that protocol leaks little information about $P$. 
**Information-theoretic framework**

- **Lemma:** If $R$ is $\varepsilon$-DP, then $I(X; R(X)) \leq O(\varepsilon^2)$

  - **Proof:** For any two distributions with $p(y) \in e^{\pm \varepsilon} q(y)$, $KL(p||q) = $

- **Stronger Lemma:** If $R$ is $\varepsilon$-DP, and

  $$W(x) = \begin{cases} x & \text{w.p. } \alpha \\ 0 & \text{w.p. } 1 - \alpha \end{cases},$$

  then $I(X; R(W(X))) \leq O(\alpha^2 \varepsilon^2)$.

  - **Proof:** Show $R \circ W$ is $O(\alpha \varepsilon)$-DP.
Bounding the information about the data

- Suppose we sample $V$ from some distribution $P$ and consider $X_1 = X_2 = \cdots = X_n = V$
  - Let $Z_i = R(X_i)$ for some $\varepsilon$-DP randomizer $R$
- Then $I(V; Z_1, \ldots, Z_n) \leq \varepsilon^2 n$

**Theorem:** $I(V; A(Z_1, \ldots, Z_n)) \leq \varepsilon^2 n$
Lower bound for mode (and histograms)

• Every participant has $x_i \in \{1, 2, \ldots, d\}$.

• Consider $V$ uniform in $\{1, \ldots, d\}$
  
  $X = (V, V, \ldots, V)$

  A histogram algorithm with relative error $\alpha \leq \frac{1}{2}$ will output $V$ (with high probability)

• **Fano’s inequality:** If $A = V$ with constant probability and $V$ uniform on $\{1, \ldots, d\}$, then $I(V; A) = \Omega(\log d)$

• But $I(V; A) \leq \epsilon^2 n$, so we need $n = \Omega\left(\frac{\log d}{\epsilon^2}\right)$ to get nontrivial error.

  ➢ Upper bound $O\left(\sqrt{\frac{\log d}{\epsilon^2 n}}\right)$ is tight for constant $\alpha$
Subconstant $\alpha$

- Let $V$ be uniform in $\{1, \ldots, d\}$, and consider data set $Y_i = W(V)$ (erase with prob $1 - \alpha$)
  - Each data set has $\approx \alpha n$ copies of $V$, the rest is 0.
  - An algorithm with error $\alpha/2$ will output $V$ with high prob

- $A$ sees $Z_i = R(W(V))$
  - By “stronger lemma”, $I(V; A) \leq O(\alpha^2 \epsilon^2 n)$
  - So $\Omega(\log d) \leq O(\alpha^2 \epsilon^2 n)$, or $\alpha = \Omega \left( \sqrt{\frac{\log d}{\epsilon^2 n}} \right)$, as desired.
Outline

• Some stuff we can do
  ➢ SQ learning
  ➢ Heavy hitters

• Some stuff we cannot do
  ➢ LDP and SQ
    • 1-bit randomizers suffice!
  ➢ Information-theoretic lower bounds
Local Model for Privacy

- Apple, Google deployments use local model
- Open questions
  - Efficient, network-friendly MPC protocols for simulating “exponential mechanism” in local model
  - Interaction in optimization (tomorrow)
  - Other tasks?