Formal verification of differentially private computations

Gilles Barthe
IMDEA Software Institute, Madrid, Spain

August 11, 2017
Data analysis

For companies:

▶ Know users
▶ Provide better services
▶ Reduce fraud

For health organizations:

▶ Establish genetic correlations
▶ Monitor epidemic
▶ Decision making

For users: your data, but not your data

▶ Accurate computations
▶ Individual privacy
Trade-off

- Individual privacy
- Overall accuracy

NB: individual privacy should entail group privacy
Data anonymization

<table>
<thead>
<tr>
<th>Name</th>
<th>Gender</th>
<th>Birth data</th>
<th>Zip code</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ann</td>
<td>F</td>
<td>12/01/1957</td>
<td>28000</td>
<td>Y</td>
</tr>
<tr>
<td>Bob</td>
<td>M</td>
<td>25/06/1970</td>
<td>28006</td>
<td>X</td>
</tr>
<tr>
<td>Eve</td>
<td>F</td>
<td>13/05/1953</td>
<td>28001</td>
<td>X</td>
</tr>
<tr>
<td>Tom</td>
<td>M</td>
<td>06/11/1966</td>
<td>28001</td>
<td>Y</td>
</tr>
</tbody>
</table>

Can infer Tom's status from knowing:
- Tom is in the database
- his gender, birth date, zip code

A real problem:
- Uniquely identify ≥ 85% of individuals
- Adding noise does not help
Data anonymization

<table>
<thead>
<tr>
<th>Name</th>
<th>Gender</th>
<th>Birth data</th>
<th>Zip code</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>bafkb</td>
<td>F</td>
<td>12/01/1957</td>
<td>28000</td>
<td>Y</td>
</tr>
<tr>
<td>kaaui</td>
<td>M</td>
<td>25/06/1970</td>
<td>28006</td>
<td>X</td>
</tr>
<tr>
<td>jmbag</td>
<td>F</td>
<td>13/05/1953</td>
<td>28001</td>
<td>X</td>
</tr>
<tr>
<td>cjhuf</td>
<td>M</td>
<td>06/11/1966</td>
<td>28001</td>
<td>Y</td>
</tr>
</tbody>
</table>

Can infer Tom's status from knowing

- Tom is in the database
- his gender, birth date, zip code
Data anonymization

<table>
<thead>
<tr>
<th>Name</th>
<th>Gender</th>
<th>Birth data</th>
<th>Zip code</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>bafkb</td>
<td>F</td>
<td>12/01/1957</td>
<td>28000</td>
<td>Y</td>
</tr>
<tr>
<td>kaaui</td>
<td>M</td>
<td>25/06/1970</td>
<td>28006</td>
<td>X</td>
</tr>
<tr>
<td>jmbag</td>
<td>F</td>
<td>13/05/1953</td>
<td>28001</td>
<td>X</td>
</tr>
<tr>
<td>cjhuf</td>
<td>M</td>
<td>06/11/1966</td>
<td>28001</td>
<td>Y</td>
</tr>
</tbody>
</table>

Can infer Tom’s status from knowing
- Tom is in the database
- his gender, birth date, zip code

A real problem
- Uniquely identify $\geq 85\%$ of individuals
- Adding noise does not help
Differential privacy
(Dwork, McSherry, Nissim, and Smith, 2006)

A randomized algorithm $K$ is $(\epsilon, \delta)$-differentially private w.r.t. $\Phi$ iff for all databases $D_1$ and $D_2$ s.t. $\Phi(D_1, D_2)$ $\forall S$. $\Pr[K(D_1) \in S] \leq \exp(\epsilon) \cdot \Pr[K(D_2) \in S] + \delta$.

Special case: If $\epsilon \approx 0$ and $\delta = 0$, then it suffices that for all databases $D_1$ and $D_2$ s.t. $\Phi(D_1, D_2)$ $\forall x$. $\Pr[K(D_1) = x] \approx (1 + \epsilon) \cdot \Pr[K(D_2) = x]$. 
Differential privacy
(Dwork, McSherry, Nissim, and Smith, 2006)

A randomized algorithm \( K \) is \((\epsilon, \delta)\)-differentially private w.r.t. \( \Phi \) iff for all databases \( D_1 \) and \( D_2 \) s.t. \( \Phi(D_1, D_2) \)

\[
\forall S. \Pr[K(D_1) \in S] \leq \exp(\epsilon) \cdot \Pr[K(D_2) \in S] + \delta.
\]

Special case
If \( \epsilon \approx 0 \) and \( \delta = 0 \), then it suffices that for all databases \( D_1 \) and \( D_2 \) s.t. \( \Phi(D_1, D_2) \)

\[
\forall x. \Pr[K(D_1) = x] \approx (1 + \epsilon) \cdot \Pr[K(D_2) = x].
\]
Differential privacy
(Dwork, McSherry, Nissim, and Smith, 2006)

A randomized algorithm $K$ is $(\epsilon, \delta)$-differentially private w.r.t. $\Phi$ iff for all databases $D_1$ and $D_2$ s.t. $\Phi(D_1, D_2)$, for all $S$, \[
\Pr[K(D_1) \in S] \leq \exp(\epsilon) \cdot \Pr[K(D_2) \in S] + \delta.
\]

Special case
If $\epsilon \approx 0$ and $\delta = 0$, then it suffices that for all databases $D_1$ and $D_2$ s.t. $\Phi(D_1, D_2)$, for all $x$, \[
\Pr[K(D_1) = x] \approx (1 + \epsilon) \cdot \Pr[K(D_2) = x].
\]
Differential privacy
(Dwork, McSherry, Nissim, and Smith, 2006)

A randomized algorithm $\mathcal{K}$ is $(\epsilon, \delta)$-differentially private w.r.t. $\Phi$ iff for all databases $D_1$ and $D_2$ s.t. $\Phi(D_1, D_2)$

$$\forall S. \ Pr[\mathcal{K}(D_1) \in S] \leq \exp(\epsilon) \cdot Pr[\mathcal{K}(D_2) \in S] + \delta$$
Differential privacy
(Dwork, McSherry, Nissim, and Smith, 2006)

A randomized algorithm $\mathcal{K}$ is $(\epsilon, \delta)$-differentially private w.r.t. $\Phi$ iff for all databases $D_1$ and $D_2$ s.t. $\Phi(D_1, D_2)$

$$\forall S. \ Pr[\mathcal{K}(D_1) \in S] \leq \exp(\epsilon) \cdot Pr[\mathcal{K}(D_2) \in S] + \delta$$

Special case
If $\epsilon \approx 0$ and $\delta = 0$, then it suffices that for all databases $D_1$ and $D_2$ s.t. $\Phi(D_1, D_2)$

$$\forall x. \ Pr[\mathcal{K}(D_1) = x] \approx (1 + \epsilon) \cdot Pr[\mathcal{K}(D_2) = x]$$
Interpretation

- $\epsilon$ can be seen as an overall privacy budget for a given collection of data
  - each access to the database consumes part of the budget (composition theorem)
  - when the budget is consumed the data cannot be accessed anymore
- $\delta$ provides a method to trade off accuracy and privacy: one can accept a risk of the private data to be revealed in exchange of a more accurate answer.

Note:

- Very small and very big values of $\epsilon$ can be problematic in practice.
- Very big values of $\delta$ ruin the privacy guarantee.
Advantages of differential privacy

- Mathematically rigorous
- Many algorithms have a private and accurate realization
- Mechanisms for achieving privacy via output perturbation
- Composition theorems for private algorithms
Randomized response

Let $q$ be a boolean-valued query and $d$ be an input:

- flip a coin $b$;
- if $b = true$, then return $q(d)$;
- else return uniformly chosen answer

Assume $q(d)$ but not $q(d')$. Then

$Pr[K(d) = true] = 3/4$ \hspace{1cm} $Pr[K(d') = true] = 1/4$

and

$Pr[K(d) = false] = 1/4$ \hspace{1cm} $Pr[K(d') = false] = 3/4$

Randomized response is $(ln 3, 0)$-differentially private.
Output perturbation

- Population is modelled as lists of values in $[0, 1]$
- How to compute its mean privately?
- For list $(x_1, \ldots, x_n)$, return true mean with noise, i.e.
  $$m + \text{Lap}_\epsilon(0)$$

Algorithm is $(\epsilon, 0)$-differentially private.
Assume $(x_1, \ldots, x_n)$ and $(x_1, \ldots, x_{i-1}, x'_i, x_{i+1}, \ldots, x_n)$ are two adjacent lists. We have:

$$\Pr[K(x_1, \ldots, x_n) = k] = \exp \left( \epsilon \cdot \left| \sum_{1 \leq i \leq n} x_i - k \right| \right)$$

Therefore

$$\frac{\Pr[K(x_1, \ldots, x_n) = k]}{\Pr[K(x_1, \ldots, x_{i-1}, x'_i, x_{i+1}, \ldots, x_n) = k]} = \frac{\exp(\epsilon \cdot |\sum_{1 \leq i \leq n} x_i - k|)}{\exp(\epsilon \cdot |\sum_{1 \leq i \leq n} x'_i - k|)}$$

$$\leq \exp(\epsilon)$$
Differential privacy via output perturbation

Let $f$ be $k$-sensitive w.r.t. $\Phi$:

$$\Phi(a, a') \implies |f(a) - f(a')| \leq k$$

Then $a \mapsto \text{Lap}_\epsilon(f(a))$ is $(k \cdot \epsilon, 0)$-differentially private w.r.t. $\Phi$.
Differential privacy by sequential composition

- If $\mathcal{K}$ is $(\epsilon, \delta)$-differentially private, and
- $\lambda a. \mathcal{K}'(a, b)$ is $(\epsilon', \delta')$-differentially private for every $b \in B$,
- then $\lambda a. \mathcal{K}'(a, \mathcal{K}(a))$ is $(\epsilon + \epsilon', \delta + \delta')$-differentially private.
Differential privacy by parallel composition

- If $\mathcal{K}$ is $(\epsilon, \delta)$-differentially private wrt $\phi$, and
- $\mathcal{K}'$ is $(\epsilon, \delta)$-differentially private with $\psi$,
- then $\lambda(a, b). (\mathcal{K}(a), \mathcal{K}(b))$ is $(\epsilon, \delta)$-differentially private wrt $\phi \cup \psi$
If $\mathcal{K}_i$ is $(\epsilon, 0)$-differentially private, then their sequential composition $\mathcal{K}$ is $(\sqrt{n} \cdot \epsilon, \delta)$-differentially private
Application: noisy sums

function \texttt{NOISYSUM}_1(a)
  \textit{s} = 0; \textit{i} = 0;
  \textbf{while} \textit{i} < \text{length}(a) \textbf{do}
    \textit{s} = \textit{s} + a[\textit{i}];
    \textit{i} = \textit{i} + 1;
  \textbf{end;}
  \textit{s} = \text{Lap}_\epsilon(\textit{s});
  \textbf{return} \textit{s}

\texttt{NOISYSUM}_1 \textit{is (}b \cdot \epsilon, 0\text{)-differentially private
Application: noisy sums

function $\text{NOISYSUM}_2(a)$

\[ s = 0; \ i = 0; \]

while $i < \text{length}(a)$ do

\[ \tilde{a} = \text{Lap}_\epsilon(a[i]); \]

\[ s = s + \tilde{a}; \]

\[ i = i + 1; \]

end;

return $s$

$\text{NOISYSUM}_2$ is $(b \cdot \epsilon, 0)$-differentially private
Iterative database construction

- Start with a uniform database
- At each iteration, pick a query privately (using Exponential Mechanism)
- Query is evaluated with noise in real database
- Apply deterministic update to synthetic database — takes as input (database, value, query) and return database
- After $t$ steps release the synthetic database.

Big gain is that we do not need to limit the number of queries
Differential privacy beyond sequential composition

There is much more to differential privacy

- Exponential mechanism
- Optimal composition
- Adaptive adversaries
- Accuracy-dependent privacy
- Also, many variants of differential privacy

Issues

- Proofs are intricate and may be wrong
- Proofs, when correct, are messy
- Hard to predict when altering an algorithm breaks privacy
The Sparse Vector Technique

\[ \text{SparseVector}_{bt}(a, b, M, N, d) := \]
\[ i = 0; l = []; \]
\[ u \leftarrow \text{Lap}_\epsilon(0); A = a - u; B = b + u; \]
while \(i < N\) do
\[ q = A(l); \]
\[ S \leftarrow \text{Lap}_\epsilon(\text{evalQ}(q, d)); \]
if \((A \leq S \leq B \land |l| < M)\) then \(l = i :: l;\)
\[ i = i + 1; \]
return \(l\)

IF all queries are 1-sensitive,
THEN algorithm achieves \((\sqrt{M}\epsilon, \delta')\)-differential privacy
EVEN IF \(M \ll N\)
3.1 Privacy Proof for Algorithm 1

We now prove the privacy of Algorithm 1. We break the proof down into two steps, to make the proof easier to understand, and, more importantly, to point out what assumptions likely caused the different non-private variants of SVT to be proposed. In the first step, we analyze the situation where the output is $z'$, a length-$\ell$ vector $(\perp, \ldots, \perp)$, indicating that all $\ell$ queries are tested to be below the threshold.

**Lemma 1.** Let $A$ be Algorithm 1. For any neighboring datasets $D$ and $D'$, and any integer $\ell$, we have

$$
\Pr[A(D) = z'] \leq \epsilon \Pr[A(D') = z'].
$$

**Proof.** We have

$$
\Pr[A(D) = z'] = \int_{-\infty}^{\infty} f_z(D, x, L) \, dx,
$$

where $f_z(D, x, L) = \Pr[p = a] \prod_{i < \ell} \Pr[q_i(D) + \nu_i < T_i + z], \quad (1)

and $L = (1, 2, \ldots, \ell)$. The probability of outputting $z'$ over $D$ is the summation (or integral) of terms $f_z(D, x, L)$, each of which is the product of $\Pr[p = a]$, the probability that the threshold noise equals $x$, and $\prod_{i < \ell} \Pr[q_i(D) + \nu_i < T_i + z]$, the conditional probability that $z'$ is the output on $D$ given that the threshold noise $\nu$ is $x$. (Note that given $D, T, \nu$, the queries, and $p$, whether one query results in $\perp$ or not depends completely on the noise $\nu_i$ and is independent from whether any other query results in $\perp$.) If we can prove

$$
f_z(D, x, L) \leq \epsilon \sum_{i < \ell} f_z(D', x, z + \Delta, L), \quad (2)
$$

then we have

$$
\Pr[A(D) = z'] \leq \int_{-\infty}^{\infty} \sum_{i < \ell} f_z(D', x, z + \Delta, L) \, dx \quad \text{from (2)}

\leq \epsilon \sum_{i < \ell} \int_{-\infty}^{\infty} f_z(D', x, z + \Delta, L) \, dx.
$$

This proves the lemma. It remains to prove Eq. (2). For any query $q_i$, because $|q_i(D) - q_i(D')| \leq \Delta$ and thus $q_i(D) = q_i(D')$, we have

$$
\Pr[q_i(D) + \nu_i < T_i + z] = \Pr[q_i(D') + \nu_i < T_i + z] 
$$

$$
\leq \Pr[q_i(D') + \nu_i < T_i + z - q_i(D')] 
$$

$$
= \Pr[q_i(D') + \nu_i < T_i + (z + \Delta)]. \quad (3)
$$

With (3), we prove (2) as follows:

$$
f_z(D, x, L) = \Pr[p = a] \prod_{i < \ell} \Pr[q_i(D) + \nu_i < T_i + z] 
$$

$$
\leq \epsilon \sum_{i < \ell} \Pr[p = a] \prod_{i < \ell} \Pr[q_i(D') + \nu_i < T_i + (z + \Delta)] 
$$

$$
= \epsilon \sum_{i < \ell} f_z(D', z + \Delta, L).
$$

That is, by using a noisy threshold, we are able to bound the probability ratio for all the negative query answers (i.e., $\perp$, $\perp$) by $\epsilon$, no matter how many negative answers there are.

We can obtain a similar result for positive query answers in the same way.

$$
f_z(D, x, L) = \Pr[p = a] \prod_{i < \ell} \Pr[q_i(D) + \nu_i \geq T_i + z].
$$

We have

$$
f_z(D, x, L) \leq \epsilon \sum_{i < \ell} f_z(D', x, z + \Delta, L)
$$

and thus

$$
\Pr[A(D) = z'] \leq \epsilon \sum_{i < \ell} \Pr[A(D') = z']
$$

This likely contributes to the misunderstandings behind Algorithms 5 and 6, which treat positive and negative answers exactly the same way. The problem is that while one is free to choose to bound positive or negative side, one cannot bound both.

We also observe that the proof of Lemma 1 will go through if no noise is added to the query answers, i.e., $\nu = 0$, because Eq. (5) holds even when $\nu = 0$. It is likely because of this observation that Algorithm 5 adds no noise to query answers. However, when considering outcomes that include both positive answers ($\top$ or $\top$) and negative answers ($\perp$ or $\perp$, one has to add noises to the query answers, as we show below.

**Theorem 2.** Algorithm 1 is $\epsilon$-DP.

**Proof.** Consider any output vector $a \in \{\perp, \top\}^\ell$. Let $a^\ell = (a_1, \ldots, a_\ell)$, $a^\top = \{i : a_i = \top\}$, and $a^\perp = \{i : a_i = \perp\}$. We have

$$
\Pr[A(D) = a] = \int_{-\infty}^{\infty} g(D, z) \, dz,
$$

where

$$
g(D, z) = \Pr[p = 1] \prod_{i \in a^\perp} \Pr[q_i(D) + \nu_i < T_i + z] \prod_{i \in a^\top} \Pr[q_i(D) + \nu_i \geq T_i + z]. \quad (6)
$$

We want to show that $g(D, z) \leq e^\epsilon g(D', z + \Delta)$. This suffices to prove that $\Pr[A(D) = a] \leq e^\epsilon \Pr[A(D') = a]$. Note that $g(D, z)$ can be written as

$$
g(D, z) = \sum_{i \in a^\perp} \Pr[q_i(D) + \nu_i < T_i + z] \prod_{i \in a^\top} \Pr[q_i(D) + \nu_i \geq T_i + z + \Delta]. \quad (4)
$$

Following the proof of Lemma 1, we can show that $f_z(D, z, \ell') \leq \epsilon \sum_{i < \ell} f_z(D', z + \Delta, \ell')$, and it remains to show

$$
\sum_{i \in a^\perp} \Pr[q_i(D) + \nu_i < T_i + z] \leq \epsilon \sum_{i < \ell'} \Pr[q_i(D') + \nu_i < T_i + z + \Delta]. \quad (5)
$$

Because $\nu_i = \text{Lap}(\Delta)$ and $|q_i(D) - q_i(D')| \leq \Delta$, we have

$$
\Pr[q_i(D) + \nu_i < T_i + z] \leq \Pr[q_i(D') + \nu_i < T_i + z - \Delta] + \Pr[q_i(D') + \nu_i < T_i + z + \Delta]
$$

$$
\leq \Pr[q_i(D') + \nu_i < T_i + z - \Delta - q_i(D')] \quad (5)
$$

$$
\leq \epsilon \sum_{i < \ell} \Pr[q_i(D') + \nu_i < T_i + z + \Delta - q_i(D') + 2\Delta] \quad (6)
$$

$$
\leq \epsilon \sum_{i < \ell} \Pr[q_i(D') + \nu_i < T_i + z + \Delta]. \quad (7)
$$

Eq (5) is because $-q_i(D') \leq -q_i(D) - \Delta$, and Eq (6) is from the Laplace distribution’s property. This proves Eq (4).

The basic idea of the proof is that when comparing $g(D, z)$ with $g(D', z + \Delta)$, we can bound the probability ratio for all outputs of $\top$ to no more than $e^\epsilon$ by using a noisy threshold, no matter how many such outputs there are. To bound the ratio for the $\perp$ outputs to no more than $e^\epsilon$, we need to add sufficient Laplacian noises, which should scale with $\epsilon$, the number of positive outputs.

Now we turn to Algorithms 3-6 to clarify what are wrong with their privacy proofs and to give their DP properties.
3.1 Privacy Proof for Algorithm 1

We now prove the privacy of Algorithm 1. We break the proof down into two steps, to make the proof easier to understand, and, more importantly, to point out what confusions likely caused the different non-private variants of SVT to be proposed. In the first step, we analyze the situation where the output is \( \hat{z}' \), a length-\( \ell \) vector \( \langle \hat{z}_1', \ldots, \hat{z}_\ell' \rangle \), indicating that all \( \ell \) queries are tested to be below the threshold.

**Lemma 1** Let \( A \) be Algorithm 1. For any neighboring datasets \( D \) and \( D' \), and any integer \( \ell \), we have:
\[
\Pr[|A(D) - A(D')| > \ell] \leq 2\epsilon \sqrt{\frac{d}{\ell}}.
\]

**Proof:** Assume \( |A(D) - A(D')| > \ell \). We have
\[
\Pr[|A(D) - A(D')| > \ell] \leq \sqrt{2\epsilon \Pr[|A(D) - A(D')| \geq \ell]}.
\]
Then we have
\[
\Pr[|A(D) - A(D')| \geq \ell] \leq 2\epsilon \sqrt{\frac{d}{\ell}}.
\]

This likely contributes to the misunderstandings behind Algorithms 5 and 6, which treat positive and negative answers exactly the same way. The problem is that while one is free to choose to bound positive or negative answers, one cannot bound both.

We also observe that the proof of Lemma 1 will go through if no noise is added to the query answers, i.e., \( \epsilon = 0 \), because Eq (3) holds even when \( \epsilon = 0 \). It is likely because of this observation that Algorithm 5 adds no noise to query answers. However, when considering outcomes that include both positive answers (\( \top \)'s) and negative answers (\( \bot \)'s), one has to add noise to the query answers, as we show below.

**Theorem 2.** Algorithm 1 is \( \epsilon \)-DP.

**Proof:** Consider any output vector \( a \in \{\top, \bot\}^\ell \). Let \( a = (a_1, \ldots, a_\ell), \top = \{1 : a_i = \top\} \), and \( \bot = \{1 : a_i = \bot\} \). We have
\[
\Pr[|A(D) - A(D')| > \ell] \leq \sqrt{2\epsilon \Pr[|A(D) - A(D')| \geq \ell]}.
\]
Then we have
\[
\Pr[|A(D) - A(D')| \geq \ell] \leq 2\epsilon \sqrt{\frac{d}{\ell}}.
\]

This proves the Lemma. It remains to prove Eq (2). For any query outcome because \( |\{D| - \{D'\}| \leq \Delta \) and \( \epsilon = 0 \) we have
\[
\Pr[(\hat{z}_i, \hat{z}_i') \notin \top \bot] \leq \epsilon \frac{1}{\ell} \sum_{i \in \ell} \Pr[(\hat{z}_i, \hat{z}_i') \notin \top \bot].
\]

We prove (2) as follows.

\[
f_i(D, z, L) = \begin{cases} 0 & \text{if } \epsilon_i(D) + \epsilon_i(L) \leq z_i \leq \epsilon_i(D) + \epsilon_i(L) + \Delta \\ \epsilon & \text{otherwise} \end{cases}
\]

That is, by using a noisy threshold, we are able to bound the probability ratio for all the negative query answers (i.e., \( \bot \)) by \( \epsilon \), no matter how many negative answers there are.

We can obtain a similar result for positive query answers in the same way.

Let \( f_i(D, z, L) = \begin{cases} 1 & \text{if } \epsilon_i(D) + \epsilon_i(L) \leq z_i \leq \epsilon_i(D) + \epsilon_i(L) + \Delta \\ 0 & \text{otherwise} \end{cases} \)

We have
\[
f_i(D, z, L) \leq \epsilon \Pr[|\hat{z}_i - \hat{z}_i'| \geq \Delta].
\]

This likely contributes to the misunderstandings behind Algorithms 5 and 6, which treat positive and negative answers exactly the same way. The problem is that while one is free to choose to bound positive or negative answers, one cannot bound both.

We also observe that the proof of Lemma 1 will go through if no noise is added to the query answers, i.e., \( \epsilon = 0 \), because Eq (3) holds even when \( \epsilon = 0 \). It is likely because of this observation that Algorithm 5 adds no noise to query answers. However, when considering outcomes that include both positive answers (\( \top \)'s) and negative answers (\( \bot \)'s), one has to add noise to the query answers, as we show below.

**Theorem 2.** Algorithm 1 is \( \epsilon \)-DP.

**Proof:** Consider any output vector \( a \in \{\top, \bot\}^\ell \). Let \( a = (a_1, \ldots, a_\ell), \top = \{1 : a_i = \top\} \), and \( \bot = \{1 : a_i = \bot\} \). We have
\[
\Pr[|A(D) - A(D')| > \ell] \leq \sqrt{2\epsilon \Pr[|A(D) - A(D')| \geq \ell]}.
\]
Then we have
\[
\Pr[|A(D) - A(D')| \geq \ell] \leq 2\epsilon \sqrt{\frac{d}{\ell}}.
\]

This proves the Lemma. It remains to prove Eq (2). For any query outcome because \( |\{D| - \{D'\}| \leq \Delta \) and \( \epsilon = 0 \) we have
\[
\Pr[(\hat{z}_i, \hat{z}_i') \notin \top \bot] \leq \epsilon \frac{1}{\ell} \sum_{i \in \ell} \Pr[(\hat{z}_i, \hat{z}_i') \notin \top \bot].
\]

We prove (2) as follows.

\[
f_i(D, z, L) = \begin{cases} 0 & \text{if } \epsilon_i(D) + \epsilon_i(L) \leq z_i \leq \epsilon_i(D) + \epsilon_i(L) + \Delta \\ \epsilon & \text{otherwise} \end{cases}
\]

That is, by using a noisy threshold, we are able to bound the probability ratio for all the negative query answers (i.e., \( \bot \)) by \( \epsilon \), no matter how many negative answers there are.

We can obtain a similar result for positive query answers in the same way.

Let \( f_i(D, z, L) = \begin{cases} 1 & \text{if } \epsilon_i(D) + \epsilon_i(L) \leq z_i \leq \epsilon_i(D) + \epsilon_i(L) + \Delta \\ 0 & \text{otherwise} \end{cases} \)

We have
\[
f_i(D, z, L) \leq \epsilon \Pr[|\hat{z}_i - \hat{z}_i'| \geq \Delta].
\]

This likely contributes to the misunderstandings behind Algorithms 5 and 6, which treat positive and negative answers exactly the same way. The problem is that while one is free to choose to bound positive or negative answers, one cannot bound both.

We also observe that the proof of Lemma 1 will go through if no noise is added to the query answers, i.e., \( \epsilon = 0 \), because Eq (3) holds even when \( \epsilon = 0 \). It is likely because of this observation that Algorithm 5 adds no noise to query answers. However, when considering outcomes that include both positive answers (\( \top \)'s) and negative answers (\( \bot \)'s), one has to add noise to the query answers, as we show below.
Approximate couplings

- Let $\mu_1$ and $\mu_2$ are distributions over $A$
- Let $\mu_L$ and $\mu_R$ be distributions over $A \times A$
- $(\mu_L, \mu_R)$ is an $(\epsilon, \delta)$-coupling of $(\mu_1, \mu_2)$ if
  - $\pi_1(\mu_L) = \mu_1$ and $\pi_2(\mu_R) = \mu_2$
  - $\Delta_\epsilon(\mu_L, \mu_R) \leq \delta$

Benefits

- (Almost) no probabilistic reasoning
- Mechanizable
- Composition theorems extend
- New proof techniques
- Extend to $f$-divergences
Approximate probabilistic Relational Hoare Logic

- Judgment

\[ \models \{ P \} \; c_1 \sim_{\epsilon, \delta} \; c_2 \; \{ Q \} \]

- Validity

\[ \forall m_1, m_2. \; (m_1, m_2) \models P \implies ([c_1] \; m_1, [c_2] \; m_2) \models Q^\#(\epsilon, \delta) \]

- \( c \) is \((\epsilon, \delta)\)-differentially private wrt \( \Phi \) iff

\[ \models \{ \Phi \} \; c \sim_{\epsilon, \delta} \; c \; \{ \equiv \} \]
Proof rules

\begin{align*}
\models \{ P \} \ c_1 \sim_{\epsilon, \delta} \ c_2 \ \{ Q \} \\
\models \{ Q \} \ c'_1 \sim_{\epsilon', \delta'} \ c'_2 \ \{ R \} \\
\models \{ P \} \ c_1 ; c'_1 \sim_{\epsilon + \epsilon', \delta + \delta'} \ c_2 ; c'_2 \ \{ R \} \\
\models \{ P \} \ c_1 \sim_{\epsilon, \delta} \ c \ \{ Q \} \\
\models \{ P \} \ c_2 \sim_{\epsilon, \delta} \ c \ \{ Q \} \\
\models \{ P \} \ if \ e \ then \ c_1 \ else \ c_2 \sim_{\epsilon, \delta} \ c \ \{ Q \} \\
\models \{ P \} \ c_1 \sim_{\epsilon, \delta} \ c'_1 \ \{ Q \} \\
\models \{ P \} \ c_2 \sim_{\epsilon, \delta} \ c'_2 \ \{ Q \} \\
\models P \rightarrow e(1) = e'(2) \\
\models \{ P \} \ if \ e \ then \ c_1 \ else \ c_2 \sim_{\epsilon, \delta} \ if \ e' \ then \ c'_1 \ else \ c'_2 \ \{ Q \}
\end{align*}
Proof principles for Laplace mechanism

Making different things look equal

\[ \Phi \triangleq |e_1\langle 1 \rangle - e_2\langle 2 \rangle| \leq k' \]
\[ \models \{ \Phi \} \ y_1 \leftarrow \text{Lap}_\epsilon(e_1) \sim_{k'\cdot \epsilon, 0} \ y_2 \leftarrow \text{Lap}_\epsilon(e_2) \ \{ y_1\langle 1 \rangle = y_2\langle 2 \rangle \} \]

Making equal things look different

\[ \Phi \triangleq e_1\langle 1 \rangle = e_2\langle 2 \rangle \]
\[ \models \{ \Phi \} \ y_1 \leftarrow \text{Lap}_\epsilon(e_1) \sim_{k\cdot \epsilon, 0} \ y_2 \leftarrow \text{Lap}_\epsilon(e_2) \ \{ y_1\langle 1 \rangle + k = y_2\langle 2 \rangle \} \]

Keeping things the same, at no cost

\[ y_1 \notin FV(e_1) \quad y_2 \notin FV(e_2) \quad \Psi \triangleq y_1\langle 1 \rangle - y_2\langle 2 \rangle = e_1\langle 1 \rangle - e_2\langle 2 \rangle \]
\[ \models \{ \top \} \ y_1 \leftarrow \text{Lap}_\epsilon(e_1) \sim_{0, 0} \ y_2 \leftarrow \text{Lap}_\epsilon(e_2) \ \{ \Psi \} \]

Pointwise equality

\[ \forall i. \models \{ \Phi \} \ c_1 \sim_{\epsilon, 0} \ c_2 \ \{ x\langle 1 \rangle = i \Rightarrow x\langle 2 \rangle = i \} \]
\[ \models \{ \Phi \} \ c_1 \sim_{\epsilon, 0} \ c_2 \ \{ x\langle 1 \rangle = x\langle 2 \rangle \} \]
Coupling proof of sparse vector

Case $b = +\infty$ and $M = 1$

- (Cost $\epsilon$): set $A^{\langle 1 \rangle} + 1 = A^{\langle 2 \rangle}$
- By pointwise equality, must prove for all $k$

\[ l^{\langle 1 \rangle} = k \Rightarrow l^{\langle 2 \rangle} = k \]

- (Cost $\epsilon$) critical iteration $i = k$: set $S^{\langle 1 \rangle} + 1 = S^{\langle 2 \rangle}$, and hence left test succeeds iff right test succeeds
- (Cost 0) iterations $i < k$: by sensitivity, $|S^{\langle 1 \rangle} - S^{\langle 2 \rangle}| \leq 1$, therefore right test succeeds implies left test succeeds
- (Cost 0) iterations $i > k$: similar

General case

- Use new optimal subset coupling for critical iterations
- Use accuracy to ensure that noisy intervals are non-empty
Accuracy via union bound logics

- Judgment $\models_{\beta} \{\Phi\} c \{\Psi\}$
- Validity: for every $m$, $m \models \Phi$ implies $\Pr_{c}(m)[\neg\Psi] \leq \beta$

Selected rules

\[
\begin{align*}
& \models_{\beta} \{\Phi\} c \{\Phi'\} \quad \models_{\beta'} \{\Phi'\} c' \{\Phi''\} \\
& \quad \models_{\beta + \beta'} \{\Phi\} c; c' \{\Phi''\}
\end{align*}
\]

\[
\begin{align*}
& \models_{\beta} \{\Phi \land e\} c \{\Psi\} \quad \models_{\beta} \{\Phi \land \neg e\} c' \{\Psi\} \\
& \quad \models_{\beta} \{\Phi\} \text{ if } e \text{ then } c \text{ else } c' \{\Psi\}
\end{align*}
\]

\[
\begin{align*}
\beta \in (0, 1) \quad \gamma &= \frac{1}{\epsilon} \log \left( \frac{1}{\beta} \right) \\
& \models_{\beta} \{\top\} x \leftarrow \text{Lap}_{\epsilon}(e) \\{|x - e| \leq \gamma\}
\end{align*}
\]
Formalization
Tools

- EasyPriv: interactive proof assistant for probabilistic programs (variant of EasyCrypt)
- Hoare2: higher-order language

Case studies

- statistics
- combinatorial optimization
- (computational) differential privacy
- mechanism design
Conclusion

- Approximate couplings naturally capture differential privacy
- Advanced concepts can be handled
- Future: variants of differential privacy