Detecting Violations of Differential Privacy

Zeyu Ding

Pennsylvania State University

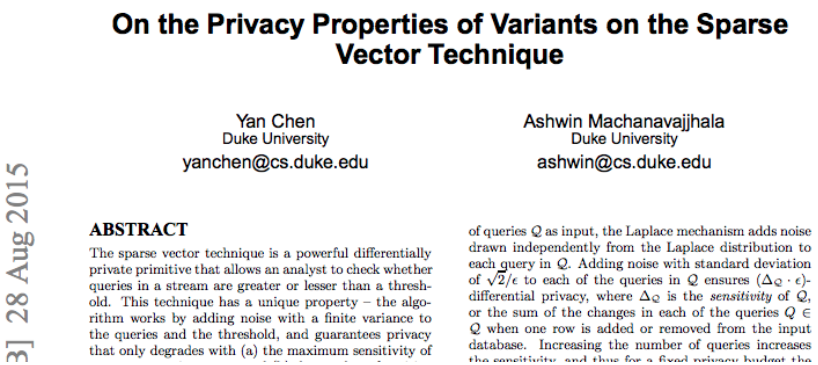
May 1, 2019

Collaborators:

- Yuxin Wang (yxwang@psu.edu)
- Danfeng Zhang (zhang@cse.psu.edu)
- Guanhong Wang (gpw5092@psu.edu)
- Daniel Kifer (dkifer@cse.psu.edu)

Motivation

- DP has seen explosive growth since 2006
  U.S. Census Bureau LEHD OnTheMap tool, Google Chrome browser, Apple's new data collection efforts, Microsoft's telemetry data collection, Uber's SQL query, etc.
- However, designing DP algorithms is no easy job.
  - Many published algorithms are incorrect.



3] 28 Aug 2015

On the Privacy Properties of Variants on the Sparse Vector Technique

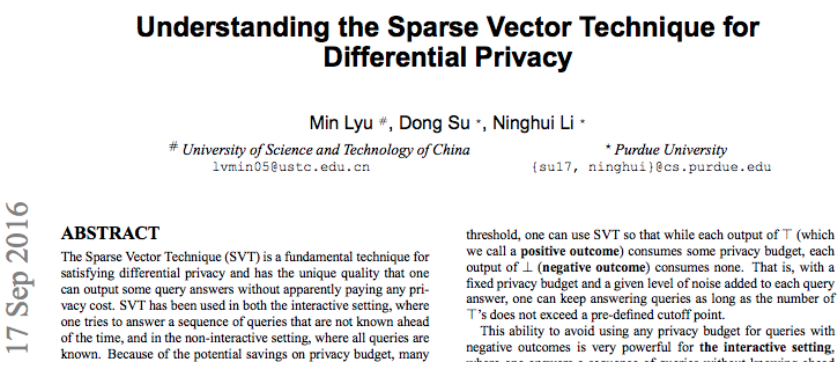
Yan Chen
Duke University
yanchen@cs.duke.edu

Ashwin Machanavajjhala
Duke University
ashwin@cs.duke.edu

ABSTRACT

The sparse vector technique is a powerful differentially private primitive that allows an analyst to check whether queries in a stream are greater or lesser than a threshold. This technique has a unique property – the algorithm works by adding noise with a finite variance to the queries and the threshold, and guarantees privacy that only degrades with (a) the maximum sensitivity of

of queries $\mathcal{Q}$ as input, the Laplace mechanism adds noise drawn independently from the Laplace distribution to each query in $\mathcal{Q}$. Adding noise with standard deviation of $\sqrt{2}/\epsilon$ to each of the queries in $\mathcal{Q}$ ensures $(\Delta_{\mathcal{Q}} \cdot \epsilon)$-differential privacy, where $\Delta_{\mathcal{Q}}$ is the *sensitivity* of $\mathcal{Q}$, or the sum of the changes in each of the queries $Q \in \mathcal{Q}$ when one row is added or removed from the input database. Increasing the number of queries increases



17 Sep 2016

Understanding the Sparse Vector Technique for Differential Privacy

Min Lyu [#], Dong Su [⋆], Ninghui Li [⋆]
[#] *University of Science and Technology of China*
lvmin05@ustc.edu.cn
[⋆] *Purdue University*
{su17, ninghui}@cs.purdue.edu

ABSTRACT

The Sparse Vector Technique (SVT) is a fundamental technique for satisfying differential privacy and has the unique quality that one can output some query answers without apparently paying any privacy cost. SVT has been used in both the interactive setting, where one tries to answer a sequence of queries that are not known ahead of the time, and in the non-interactive setting, where all queries are known. Because of the potential savings on privacy budget, many

threshold, one can use SVT so that while each output of $\top$ (which we call a **positive outcome**) consumes some privacy budget, each output of $\perp$ (**negative outcome**) consumes none. That is, with a fixed privacy budget and a given level of noise added to each query answer, one can keep answering queries as long as the number of $\top$'s does not exceed a pre-defined cutoff point.

This ability to avoid using any privacy budget for queries with negative outcomes is very powerful for **the interactive setting**,

Motivation

- DP has sparked great interests in the program verification community
  - Relational program logics [Barthe et al. 2012]
  - Coupling based proofs [Barthe et al. 2016, Albarghouthi and Hsu 2017]
  - Randomness alignment [Zhang and Kifer 2017, Wang et al. 2019]
- However, verifying DP is a challenging task
  - It's an undecidable problem, a sound and complete analysis is impossible
  - Most methods rely on composition, resulting in over-estimation of the privacy cost

Our goal

- Identify incorrect ϵ-DP algorithms in a **semi-blackbox** manner by generating **counterexamples**.
  - No restrictions on noise mechanisms used.
  - No restrictions on programming languages used.
- In a way, it is similar to identifying program bugs by finding inputs that trigger an error.

Recall the Definition of DP

Definition (Cynthia Dwork, Frank McSherry, Kobbi Nissim, Adam Smith '06)

An algorithm M is said to be ϵ-differentially private if for every pair of adjacent databases D_1, D_2, and every $E \subseteq \mathrm{Range}(M)$, we have

$$P(M(D_1) \in E) \leq e^{\epsilon} \cdot P(M(D_2) \in E).$$

What is a counterexample?

A **counterexample** of ϵ-DP consists of

- a pair of adjacent databases D_1, D_2;
- a set E of possible outputs of M;
- "strong evidence" that differential privacy is violated, i.e.,

$$\underbrace{P(M(D_1) \in E)}_{p_1} > e^{\epsilon} \cdot \underbrace{P(M(D_2) \in E)}_{p_2}.$$

Evidence? Use statistical hypothesis test

- How to show "strong evidence" that indicates

$$\underbrace{P(M(D_1) \in E)}_{p_1} > e^{\epsilon} \cdot \underbrace{P(M(D_2) \in E)}_{p_2}?$$

- We do **hypothesis testing**.
  - Formulate the **null hypothesis**: $p_1 \leq e^{\epsilon} \cdot p_2$
  - Generate data sample
  - Compute a **p-value** (significance): the probability of seeing a sample like this or more extreme if the null hypothesis is true.

How to do a hypothesis test?

First we obtain sample data.

- Run M on D_1 many (n) times.
- Count how many outputs are inside E, denote this number by c_1.
- Note: c_1 is equivalent to a sample from Binomial(n, p_1).
- Repeat this process on D_2 to get another count c_2.

How to calculate a valid p-value

Then we compute a p-value.

- The difficulty is that we don't know p_1 and p_2.
- The good news is that we don't need to know p_1 and p_2.
  We just need to know if $p_1 \leq e^{\epsilon} \cdot p_2$.
- Therefore, we
  - downsample $\tilde{c_1}$ from Binomial$(c_1, 1/e^{\epsilon})$.
    The marginal distribution of $\tilde{c_1}$ is Binomial$(n, p_1/e^{\epsilon})$.
  - apply *Fisher's Exact Test* on $\tilde{c_1}, c_2$.
  - average to reduce variance.

How our tool detects violations of DP

Find candidate pairs (D_1, D_2)
Input Generation

- Confine search space to a small but highly representative set

For every (D_1, D_2) find an E
Event Selection

- Confine search space for E
- Use hypothesis test in exploratory mode to select the best E

For every (D_1, D_2, E) try to find strong evidence that $P(M(D_1) \in E) > e^{\epsilon} \cdot P(M(D_2) \in E)$

- Use hypothesis test again to generate evidence

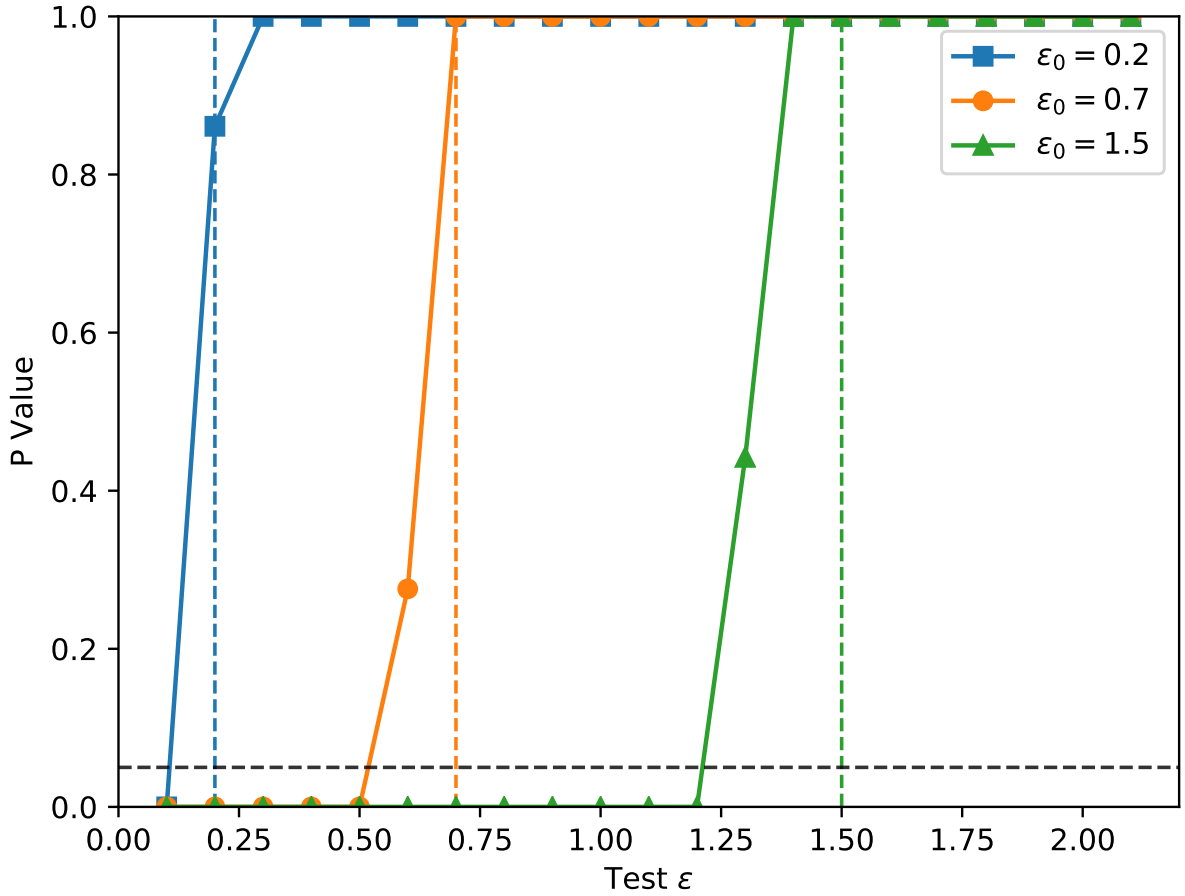
How do we evaluate our tool

- If M claims to be ϵ_0-DP, we run our counterexample detector for all ϵ's in a range containing ϵ_0.
- We plot the p-value of the counterexample found for every ϵ.
- When the p-value is close to 0 for some ϵ, it means we have strong evidence that M is not ϵ-DP.
- The largest ϵ with a p-value close to 0 can be regarded as a **lower bound** for the true ϵ_0.

Evaluation: correct SparseVector

Algorithm 1: ϵ_0-DP

function SVT($\boldsymbol{q}$, T, N, ϵ_0):
- $i \leftarrow 0$; $count \leftarrow 0$
- $\eta_1 \leftarrow \mathrm{Lap}(2/\epsilon_0)$
- $\tilde{T} \leftarrow T + \eta_1$
- **while** $(i < \mathrm{len}(\boldsymbol{q})) \wedge (count < N)$ **do**
  - $\eta_2 \leftarrow \mathrm{Lap}(4N/\epsilon_0)$
  - **if** $\boldsymbol{q}[i] + \eta_2 \geq \tilde{T}$ **then**
    - **output:** $\top$
    - $count \leftarrow count + 1$
  - **else**
    - **output:** $\perp$
  - **end**
  - $i \leftarrow i + 1$
- **end**

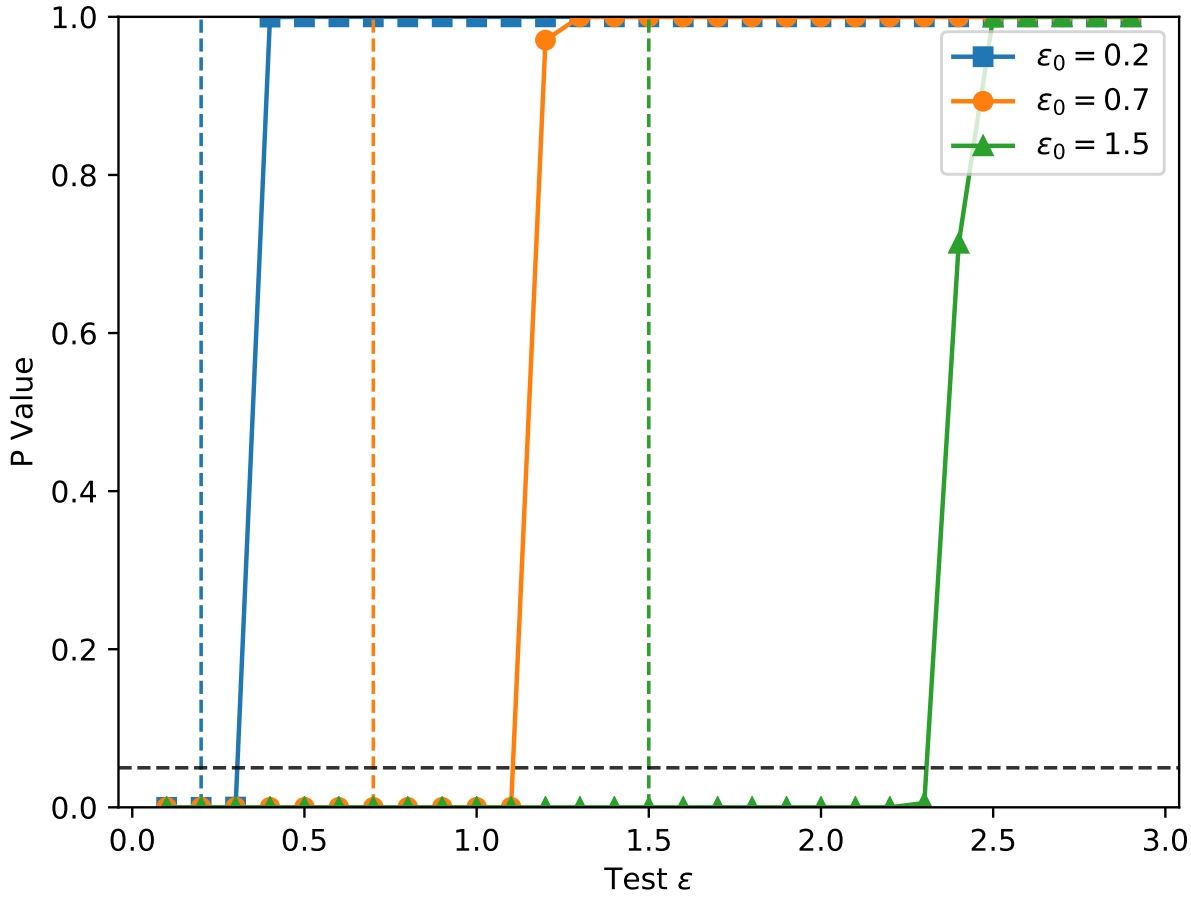


No violations found. The result is as expected.

Evaluation: incorrect SparseVector

Algorithm 2: $\frac{1+6N}{4}\epsilon_0$-DP.

function SVT($\boldsymbol{q}$, T, ϵ, N):
- $i \leftarrow 0$; $count \leftarrow 0$
- $\eta_1 \leftarrow \mathrm{Lap}(4/\epsilon_0)$
- $\tilde{T} \leftarrow T + \eta_1$
- **while** $(i < \mathrm{len}(\boldsymbol{q})) \wedge (count < N)$ **do**
  - $\eta_2 \leftarrow \mathrm{Lap}(4/3\epsilon_0)$
  - **if** $\boldsymbol{q}[i] + \eta_2 \geq \tilde{T}$ **then**
    - **output:** $\top$
    - $count \leftarrow count + 1$
  - **else**
    - **output:** $\perp$
  - **end**
  - $i \leftarrow i + 1$
- **end**



- Violations detected.
- For claimed $\epsilon_0 = 0.2, 0.7, 1.5$, the **true** $\epsilon_0 = 0.35, 1.225, 2.625$.

Summary

Pros:

- Code available at: https://github.com/cmla-psu/statdp
- Our tool performs well on (variations of) SparseVector, NoisyMax, Histogram, Laplace Mechanism, etc.
- Applicable to other algorithms (you might have to write your own input generation module)

Cons (further work):

- Currently only works with ϵ-DP
- Counterexamples are statistical in nature

Thank you!

Question?

References

Chen, R., Xiao, Q., Zhang, Y., and Xu, J. (2015).

Differentially private high-dimensional data publication via sampling-based inference.

In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 129–138. ACM.

Dwork, C., Roth, A., et al. (2014).

The algorithmic foundations of differential privacy.

Foundations and Trends® in Theoretical Computer Science, 9(3–4):211–407.

Lee, J. and Clifton, C. W. (2014).

Top-k frequent itemsets via differentially private fp-

In *Proceedings of the 20th ACM SIGKDD international on Knowledge discovery and data mining*, pages 931–940. ACM.

Lyu, M., Su, D., and Li, N. (2017).

Understanding the sparse vector technique for differential privacy.

Proceedings of the VLDB Endowment, 10(6):637–648.

Roth, A. (2011).

Sparse vector technique, lecture notes for "the algorithmic foundations of data privacy".

Stoddard, B., Chen, Y., and Machanavajjhala, A. (2014).

Differentially private algorithms for empirical machine learning.

arXiv preprint arXiv:1411.5428.