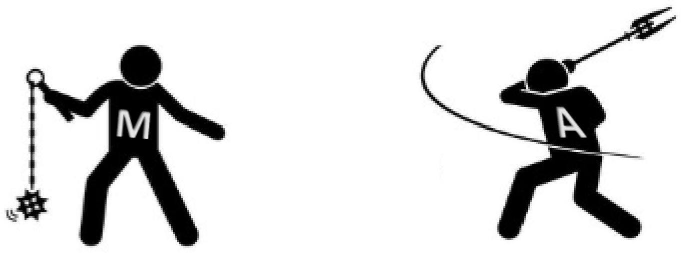
Reconstruction simulator

Project on course 67508- foundations of data privacy

Maya Cohen

24/02/2020



Intro

This work Implements a generic and modular tool, used for exploring the connection between ε –differential privacy parameter and reconstruction threats in the context of a specific dataset. The program automatically simulates both a private mechanism and an adversary, working against each other. The simulator determines a security lower bound for the ε parameter. Below this bound, the mechanism is considered vulnerable to reconstruction manipulations with high probability.

The purpose of such tool is to become an open-source platform, managing libraries of generic attacks and common mechanisms implementations. This setup allows both attackers and defenders to implement their new mechanisms according to a standard API and run them against all other known methods. Each contribution to this project helps other researchers to refine their techniques and allows data curator to select the best known solution according to all known threats.

Simulator objects and flow

Reconstruction simulator is supplied with three main objects:

A **private mechanism** (PM) - perform as the data curator, defines how queries are answered and functions as a guard-keeper between the adversary and original data

An **adversary** (A) - may generate queries, collect responses from the private mechanism and produce a prediction of the original data.

And a **data comparator** (DC) – is the data representing object. It's defined separately from the other objects, and used to holds the environment data. Data comparator also defines the distance between two datasets. It is used to evaluate reconstruction quality by comparing it to the original environment data, according to a hand-written meaningful metric.

The simulator flow consists of simulating a complete attack sessions.
An attack session contains the following steps:

- An adversary A is sequentially generating a linear amount of queries and collecting responses from the private mechanism PM.
- A creating a prediction for the private data.
- The data comparator DC evaluate the attack success rate.

This process is then repeated under a range of security parameter values and between different adversaries and privacy-preserving mechanisms. Finally, in order to estimate the reconstruction success rate from a sequence of repetitions under a certain condition, the median result is chosen as it represents a common and conceivable result.

Case study- keeping it simple:

Since the scope of this project is limited my main focus is the interface between the different objects and building the simulator as a modular environment. The implemented objects on this project are basic, but more complicated ones can be written and plugged in easily in order to enrich model finesse.

Secrete datasets- The case study consists of exploring security bounds over two sets of private data.

- The first, a binary citizenship indicator column from the 2010 census data. This datasheet contains ~250K records and was released as Massachusetts microdata sample.
- The second datasheet is the anonymized gradebook of course 67101- introduction to computer science in one of the last years. It contains ~400 records with grades in range between 0 and 100.

Data comparators-
intro_grades_DB_DC- extracts the grades vector out of the gradebook data provided datasheet. The reconstruction rate is calculated as the proportion of grades that was reconstructed successfully up to an error smaller than 2 points for both sides.
census_citizenship_DB_DC- extracts the census data out of the provided datasheet. The reconstruction rate is calculated as the proportion of bits that was reconstructed accurately.

Private mechanisms- The private mechanisms are divided into two classifications- tested mechanisms and baseline mechanisms. Baselines mechanisms are trivial algorithms, and are being used to mark the expected results on degenerate cases. Tested mechanisms are the proposed mechanisms and their results are measured in comparison to the baselines.

Implemented baselines:
No_privacy mechanism- responds on a query with the accurate true answer.
Random_answers mechanism- responds on a query with a completely random answer. Answer is sampled in two steps. First, a uniform random valid number of DB entries is being sampled. Second, that number of valid elements are randomly sampled from their valid range, the query is calculated over the random sample.

Implemented differential privacy mechanisms:
Round_to_R_multiplication mechanism- responds on a query with a noised answer. Noised answer is determined by rounding the true answer to the nearest multiple of the security parameter R.
Epsilon_gausian_noise mechanism- responds on a query with a noised answer. Noised answer is determined by adding a Gaussian noise with epsilon scale to the true answer.

Adversary- The implemented adversary, LP_reconstructor, is based on [CN18].[1] While generating queries, the adversary outputs a random a binary vector with the dataset size, representing a random subset of entries. Both generated queries and their responses are stored in the adversary memory. We mark the ith query with q_i. And the ith response with r_i. When a prediction is made, the following equations system is solved using SciPy Linalg lib [2]:

[1] Aloni Cohen, Kobbi Nissim. Linear Program Reconstruction in Practice. Technical report, arXiv, 2018. arXiv:1810.05692

[2] Converting absolute value to a linear expression was made as explained in: math.stackexchange.com/questions/432003

$$\begin{aligned}&variables: x = \{x_i\}_{i=1}^{data-size}, e = \left\{e_q\right\}_{q=1}^{|Q|}\\&minimize: \sum |e_i|\\&subject\ to:\\&\forall i:\ < x, q_i > + e_i = r_i\\&\forall j: x_j \in \{valid\ data\ values\}\end{aligned}$$

Meaning- x represents the original entries vector, e represents the errors vector on the different queries. The adversary tries to find a database x that minimize l_1 norm over the error vector.

Due to technical issues, increasing the problem scale causes the external LP solver to fail frequently even on ~30 records datasets. It seems that feasible solution cannot be easily found by the algorithm, although theoretically, such solution trivially exists ($0 \rightarrow x, r \rightarrow e$, meaning all responses are explained exclusively by the noise variables). A bug on the external lib prevents setting an explicit initialization to the solver. If the algorithm couldn't find a single feasible solution it fails and the adversary outputs a valid random guess. In such case not even a suboptimal approximation is available.

Some possible technical ways to overcome this problem- Fixing the SciPy bug, implementing an independent LP solver or migrating the project to R environment and use its commercial implementation.

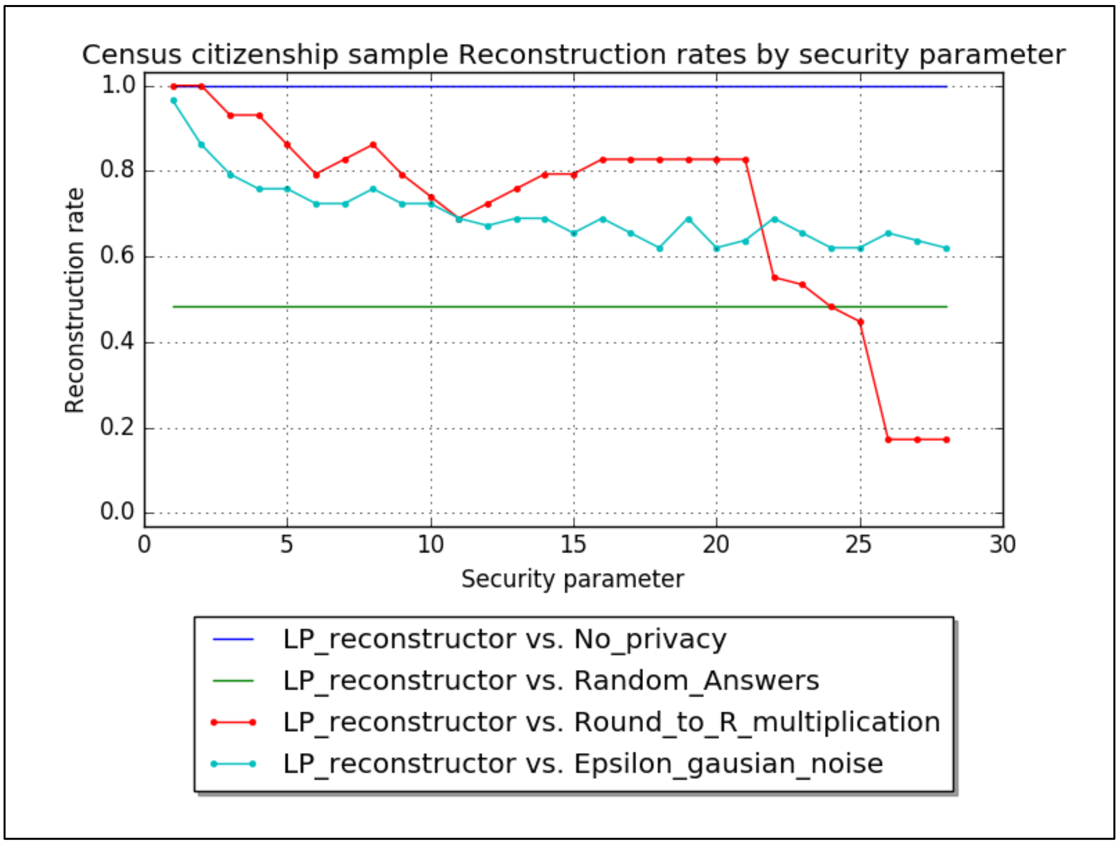
Another possible solution is to observe a relaxation on the private mechanisms, allowing only positive noise addition, which simplifies the LP objective function and changes its inner structure, allowing the external algorithm to perform better on it. This relaxation desperately reducing modularity in exchange.

Eventually, in order to shift the focus of the project from the simplex algorithm machinery to simulated bounds on privacy parameters, the datasets were reduced to small demo examples (of size 20-30 records) and the results presented are based on these tiny simulations.

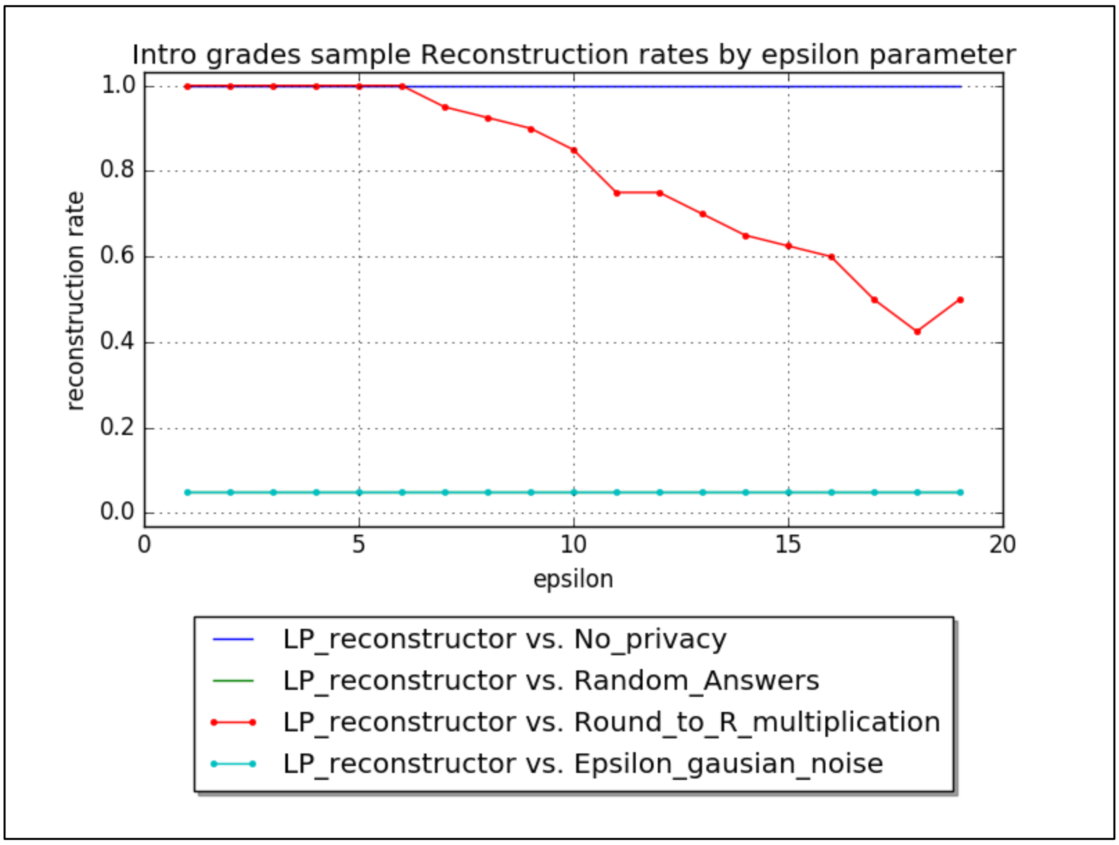
Background parameters- Were chosen arbitrarily. For a dataset with n entries, the attack session allows up to $2n$ queries and responses, each setup is sampled for 50 times, and the tested range of epsilon security parameter is $[n]$

Results

The above described classes of PM, A were plugged-in and simulated on two different environments defined by the mentioned DCs. The results are graphically presented.



Based on the result, a curator holding the Census citizenship sample, allowing a maximal reconstruction rate of 60% should conclude that using a rounding mechanism requires $R > 22$ security parameter, or using a Gaussian mechanism requires $\varepsilon > 30$.



In the second scenario, a curator protecting the course gradebook, allowing the same maximal reconstruction rate should conclude that using the rounding mechanism requires at least $R > 16$. Also it seems that for the Gaussian mechanism, the adversary was unable to reconstruct more than a trivial amount of grades. In this case the cause for the observed effect is due to the adversary limited implementation and not in favor of the quality of the Gaussian noise as privacy-preserving mechanisms.

Important reservation- These results only reflects resilience against the specific implemented LP_reconstruction attack. For a stronger version of these bounds, privacy preserving mechanisms should be tested against a richer arsenal of attacks. Likewise, tailored attacks for this datasets may yield even better lower bounds.

Appendix- code guide

All code, Datasets, simulation logs and results can be found on Github[3].

Attaks_lib.py, private_mechanisms_lib.py and data_utils.py containing implementations of Adversaries, Private Mechanisms, and Data Comparators respectively. The simulator is implemented in simulator.py. Secret data directory contains the original and demo-version data files. Simulation results directory contains the logs and data files that are automatically generated on the simulator run.

[3] https://github.com/MAyaCohenCS/PrivacyAttackSimulator