CS 860: Algorithms for Private Data Analysis Fall 2020

Lecture 2 — Reconstruction Attacks

Prof. Gautam Kamath *Scribe: Gautam Kamath*

Last time, we spoke informally about privacy violations. Today, we'll get a bit more formal. We won't yet talk about what it means for an algorithm to be private, but we'll talk about algorithms which are "blatantly non-private." Today's lecture will focus on a paper of Dinur and Nissim from 2003 [DN03], considered by many to be the spark of differential privacy. In particular, Section 4 of said paper (which the authors attribute to joint work with Cynthia Dwork) could be considered as a primitive form of differential privacy. The phrasing is somewhat unusual (with the power of hindsight, knowing differential privacy): it focuses on computationally-bounded adversaries, Binomial noise, and a restricted class of queries. In contrast, modern differential privacy focuses more on query-bounded adversaries, Laplace or Gaussian noise, and general queries of bounded-sensitivity. While we will not cover this section of the paper here, it is recommended reading for a historical perspective on where differential privacy came from.

We will focus on a class of attacks known as reconstruction attacks. First, we start by addressing how reconstruction attacks could work in settings without noise. Specifically, we will look at reconstruction attacks may be executed in settings where the only privatization is aggregation of statistics, following the census setting of Garfinkel, Abowd, and Martindale [GAM19]. We then present the Dinur-Nissim attacks for a noised setting [DN03]. Finally, we present their application in a practical setting by Cohen and Nissim [CN20].

Cracking Aggregation

We will succinctly consider how to reconstruct microdata given aggregate statistics, following the presentation of Garfinkel, Abowd, and Martindale [GAM19]. We recommend reading their article for a more thorough analysis and discussion.

A census collects data from a population, including fields such as each individual's age, sex, race, and more (though we will only be concerned with these features for the sake of discussion). Further simplifying, we will consider only two responses for race (black and white) and two responses for sex (male and female). This collection of raw data is called the *microdata*. However, this data is generally considered too personally identifying (enforced by legal regulations), so instead, aggregate statistics are released, as in Figure 1. Though this is an aggregation for a single block, census data is often reported hierarchically, with aggregates released for blocks as well as for larger regions.

Note that certain cells have been suppressed (marked with a (D)), due to small individual counts. For example, by noting that there are 4 black people, and 3 black women, we can infer that there is only 1 black man – releasing the median or mean here would equate to releasing his exact age, which we consider a privacy violation. Generalizing this observation: we note that this set of aggregate statistics gives us redundant information about the same individuals. These redundancies will allow us to form constraints on the possible microdatas which could have generated the given aggregates. Given sufficiently many constraints, the only feasible microdata will be the true one.

But we're getting ahead of ourselves. Let's consider Statistic 2B: we are told that there are three

		Age		
Statistic	Group	Count	Median	Mean
1A	Total Population	7	30	38
2A	Female	4	30	33.5
2B	Male	3	30	44
2C	Black or African American	4	51	48.5
2D	White	3	24	24
3A	Single Adults	(D)	(D)	(D)
3B	Married Adults	4	51	54
4A	Black or African American Female	3	36	36.7
4B	Black or African American Male	(D)	(D)	(D)
4C	White Male	(D)	(D)	(D)
4D	White Female	(D)	(D)	(D)
5A	Persons Under 5 Years	(D)	(D)	(D)
5B	Persons Under 18 Years	(D)	(D)	(D)
5C	Persons 64 Years or Over	(D)	(D)	(D)

Note: Married persons must be 15 or over

Figure 1: Figure from [GAM19]. Aggregate statistics for a fictional block.

males, having ages with median 30 and mean 44. Let their ages (in non-decreasing order, without loss of generality) be A, B, and C. What can we say about these values? First, we assume that only whole numbers are reported. Next, we can say that $0 \leq A, B, C \leq 125$ – the upper bound comes from the oldest verified age of any human being 122. Using only these bounds on the ages, we get that there are upwards of $300,000$ possible combinations of A, B, and C.[1] However, we know more than this: we have the mean and the median, which narrows down the possibilities significantly. Using the fact that the median is 30, we can conclude that $B = 30$. The mean gives us the constraint that $(A + B + C)/3 = 44$. It is not hard to enumerate the 31 possibilities which satisfy this constraint as well, and this is done in Figure 2.

A	B	C	A	B	C	A	B	C
1	30	101	11	30	91	21	30	81
2	30	100	12	30	90	22	30	80
3	30	99	13	30	89	23	30	79
4	30	98	14	30	88	24	30	78
5	30	97	15	30	87	25	30	77
6	30	96	16	30	86	26	30	76
7	30	95	17	30	85	27	30	75
8	30	94	18	30	84	28	30	74
9	30	93	19	30	83	29	30	73
10	30	92	20	30	82	30	30	72

Figure 2: Figure from [GAM19]. Possible sets of ages which are consistent with Statistic 2B. Note that the table excludes the possibility $A = 0$, $B = 30$, $C = 102$.

This gives us significant information already about three of the individuals in our dataset, and we have only exploited a single statistic. An attack in this setting would similarly define constraints given by all the aggregate statistics, and find any assignment that satisfies all of them simultane-

[1] [GAM19] provides a calculation of $\binom{125}{3}$ – however, this seems to exclude 0 as a valid age, and furthermore disallows repetitions. I believe the correct calculation should be $\binom{126+3-1}{3}$, but it doesn't really matter much for our purposes.

ously. While solving such a system of constraints is in general NP hard, we have highly efficient heuristics for these problems, including solvers for satisfiability problems (SAT solvers) and integer programming solvers. This specific instance has only a single feasible solution, which was obtainable in less than 0.1 seconds on a personal laptop from 2013.

While this toy example may seem harmless, the same approach was effective even for the full 2010 US Census. Abowd write a Tweetorial describing an attack performed internally [Abo19], where, given the aggregated statistics, they were able to reconstruct the microdata exactly for 46% of the population, and allowing errors in age of ± 1 year, for 71% of the population. Linking this with commercial databases allowed them to correctly re-identify over 50 million individuals by name. Such large scale reconstruction attacks were considered to be a red flag, leading to the adoption of differential privacy for disclosure avoidance in the 2020 US census.

Queries, Blatant Non-Privacy, and the Dinur-Nissim Attack

Much of this section is based off the presentation from Section 8.1 of [DR14].

Since we're going to be more formal, let's start by carefully defining the model that we're working in. We will work in a rather simplified model, a sort of bare minimum case to demonstrate interesting phenomena. Imagine that a database consists of several points, which we interchangeably refer to as "rows". For comparison, a feature or dimension may be referred to as a "column". Several of these features are "identifiers", such as name, postal code, date of birth, sex, etc. – for the sake of simplicity, we assume these all to be public data. One feature will be the individual's secret information, which (again for simplicity) will be a single bit, either 0 or 1. We let $d \in \{0,1\}^n$ be the vector of secret bits. Here's an example of such a table:

Name	Postal Code	Date of Birth	Sex	Has Disease?
Alice	K8V7R6	5/2/1984	F	1
Bob	V5K5J9	2/8/2001	M	0
Charlie	V1C7J	10/10/1954	M	1
David	R4K5T1	4/4/1944	M	0
Eve	G7N8Y3	1/1/1980	F	1

A data analyst requires access to certain types queries, while the curator must "protect the privacy" of the individuals in the dataset (to be defined shortly).

The data analyst will be able to ask queries of the form "How many of the rows satisfying (conditions on identifiers) have 'Has Disease? $= 1$'?" For instance, (conditions on identifiers) could be "Name = Alice OR Name = Charlie OR Name = David" – the true answer to this query would be 2. More abstractly, we will assume the analyst can specify queries which are a subset of $[n]$. We let $S \in \{0,1\}^n$ be a query vector, which is 1 for the indices which are included in the subset, and 0 for those which are not. We call these *subset queries*. We note that these queries might be quite complex to specify – we eschew this concern for the time being, but will revisit it when discussing the Cohen-Nissim attack. The true answer to query S is $A(S) = d \cdot S$, the dot product of d and S.

The curator will receive a query set S, and output a response $r(S)$. Observe that, if they simply output $r(S) = A(S)$, this would be easily susceptible to privacy violations: the analyst could ask the single query $S = \{i\}$, which would reveal the secret bit of individual i. As a result, the

curator will output a "noised" version of $A(S)$: specifically, they will output an $r(S)$ such that $|r(S) - A(S)| \leq E$ for some bound E. Note that we do not require that this difference $r(S) - A(S)$ is randomly distributed – the curator may output any $r(S)$ which is within distance E of $A(S)$.

We finally define what it means for an algorithm to be non-private.

Definition 1. *An algorithm is* blatantly non-private *if an adversary can construct a database $c \in \{0,1\}^n$ such that it matches the true database d in all but $o(n)$ entries.*

We note that an algorithm being blatantly non-private is indeed a rather flagrant violation of privacy. It means that the adversary can construct a database which agrees with the true database on 99% of the entries! Hence, we call these *reconstruction* attacks. We will show that fairly general schemes are blatantly non-private.

Theorem 2 ([DN03])**.** *If the analyst is allowed to ask 2^n subset queries, and the curator adds noise with some bound E, then based on the results, the adversary can reconstruct the database in all but $4E$ positions.*

In particular, if $E = n/401$, then the adversary can match the true database on 99% of the entries. Furthermore, if $E = o(n)$, then the algorithm is blatantly non-private.

Proof. In short, the analyst asks queries for all 2^n subsets of $[n]$, and the adversary will output any database which is consistent with the provided answers. Specifically, for each candidate database $c \in \{0,1\}^n$, if there exists a query set S such that $|\sum_{i \in S} c_i - r(S)| > E$, then rule out c. If a c can not be ruled out, then output c. Note that the true database d satisfies this condition and would not be ruled out, so this will output *some* datatabase.

Let I_0 be the set of indices where the true database is 0, $I_0 = \{i \mid d_i = 0\}$. We define I_1 symmetrically, $I_1 = \{i \mid d_i = 1\}$. Consider the database c which is output. By the definition of the adversary's strategy, $|\sum_{i \in I_0} c_i - r(I_0)| \leq E$. At the same time, restrictions on the the curator's noising strategy ensure that $|\sum_{i \in I_0} d_i - r(I_0)| \leq E$. By triangle inequality, c and d differ by at most $2E$ entries in the subset I_0. A symmetric argument implies the same for the subset I_1, and thus they differ by at most $4E$ entries overall. □

A succinct way to state this result: if the analyst is able to ask 2^n queries with noise bounded by $O(n)$, then the adversary can reconstruct (essentially) the entire database. While we arrive at a strong conclusion of blatant non-privacy, it required the analyst to ask *exponentially* many queries. Thus, such an attack is unlikely to be effective in practice. However, Dinur and Nissim also provide a computationally and query efficient attack.

Theorem 3 ([DN03])**.** *If the analyst is allowed to ask $O(n)$ subset queries, and the curator adds noise with some bound $E = O(\alpha\sqrt{n})$, then based on the results, a computationally efficient adversary can reconstruct the database in all but $O(\alpha^2)$ positions.*

We do not prove this here, but give some intuition as to why it is true. While the proof is more complicated, the attack algorithm is very similar to before. This time, rather than asking all 2^n queries, the analyst asks only $O(n)$ subset queries, where each S is chosen uniformly at random from all subsets of $[n]$. The adversary will once again output any database consistent with the

results. While before, this was done by enumerating over all 2^n possible databases, in this case it can be done efficiently using a linear program.

For intuition, we switch our focus to the case where c and d are $\in \{\pm 1\}^n$, rather than $\{0,1\}^n$, and similarly for the query vector S. Suppose that the true database d and some (fixed) candidate database c differ in $\Omega(n)$ coordinates. The claim is that the true answer of a random query S will differ significantly under databases c and d – in fact, so significantly that the curator would not be able to make them consistent due to the addition of noise!

Consider the inner product $(c-d) \cdot S = \sum_{i \in [n]} (c_i - d_i) S_i$, where S is chosen uniformly at random from all strings $\{\pm 1\}^n$. If $c_i = d_i$, then this coordinate will contribute 0 to the sum. Otherwise, if they differ, $(c_i - d_i)S_i$ will be ± 2 with equal probability. Thus, the overall sum will be a (rescaled and shifted) version of Binomial$(\Omega(n), 1/2)$ – crucially, $(c-d) \cdot S$ is a distribution with mean 0 and variance $\Omega(n)$, and by *anti-concentration* of the Binomial distribution, it will take a value of magnitude $\Omega(\sqrt{n})$ with high probability. But if the curator is constrained to adding error $o(\sqrt{n})$, it is impossible for them to "fool" the analyst into thinking that c could have produced this outcome for query S, and thus c can be eliminated. Note that this gives a probabilistic guarantee of eliminating only a single database c – since we ask *many* queries, this allows us to rule out any individual "far" database with very high probability. In fact, with such high probability that one can take a union bound over all databases far from d, and the only remaining ones are close. For a more complete treatment, see the proof of Theorem 8.2 in [DR14].

Let's recap what we've seen so far. The first attack required 2^n queries to break noise addition of $O(n)$. The second attack required $\Omega(n)$ queries to break noise addition of $O(\sqrt{n})$ (in fact, this attack can be strengthened – as shown by Dwork, McSherry, and Talwar, it's possible for the adversary to succeed even when the curator makes a constant fraction of their responses with *arbitrary* noise magnitudes [DMT07]). In a certain sense, the latter attack is "tight" – as we will see, differential privacy will allow us to answer $O(n)$ queries with added noise of magnitude $O(\sqrt{n})$, using either the Laplace or Gaussian mechanism under differential privacy. In fact, this privacy upper bound will also "scale down", in settings where we are asking far fewer than n queries (which might be large in certain "big data" scale applications). If the analyst only asks $m \ll n$ queries, then the curator only has to add noise of magnitude $O(\sqrt{m})$. But we will leave further discussion on this matter to when we actually introduce differential privacy.

Linear Reconstruction in Practice

All the discussion so far has been rather theoretical and academic in nature. You might be thinking, when could this attack actually be executed? Enter the Aircloak challenge.

In 2017, a company called Aircloak released a production system called Diffix. The goal of Diffix is to provide data analysts the ability to perform an *unlimited* numbers of queries on a sensitive database, while preserving user privacy yet introducing less noise than differential privacy would necessitate. In a landmark competition, Aircloak provided "the first bounty program for anonymized data re-identification." In particular, for an effective reconstruction attack (i.e., one that identifies the private data of a large fraction of the dataset), they would award cash prizes as large as $5,000.

In fact, the setting is strikingly similar to setting we have studied so far. Similar to before, the data analyst is allowed to ask count or subset queries. The main difference is that the magnitude of the

noise added to each query increases with the square root of the number of "conditions". Consider the query we mentioned much at the beginning of this lecture: "Name = Alice OR Name = Charlie OR Name = David". This would have 3 conditions. Other heuristics to protect against attacks involve suppression of small counts, modification of extreme values, and notably, disallowing the OR operator (so the naïve approach to specification above wouldn't work).

Recall that the (poly-time) Dinur-Nissim requires the analyst to ask "random queries". The naïve way of specifying a random query would involve first choosing a uniformly random subset of the individuals, and then coming up with some set of conditions to specify this subset. Even ignoring the restriction of OR queries, it would seem that a subset of size k would require something like k conditions in order to specify. Cohen and Nissim [CN20] avoided this barrier by a heuristic but effective method of turning this on its head: instead of choosing a subset and then coming up with conditions to specify it, they would come up with a query involving a low number of conditions which specifies a "random" set. With a careful choice of low-condition queries, it seems that the randomness of the sets were sufficient to mount an effective attack.

Each user in the dataset has a unique client ID, which we refer to as `client-id`. The question is whether there is a function that takes in `client-id` and includes it in the query set "sufficiently randomly"? They use functions specified by four variables: `mult, exp, d, pred`. The first three variables are numbers, the fourth is a predicate which takes in a number, and outputs true or false. A row is included in the query if it satisfies the following condition: Does the `d`-th digit of `(mult * client-id)^exp` satisfy `pred`? To give an example, suppose `mult` is 17, `client-id` is 1, `exp` 0.5, `d` is 3, and `pred` is "Is the digit even?" We would first calculate $(17 \cdot 1)^{0.5} = 4.1231\ldots$. Observe that the the third digit here is 2, which is even, so the client with ID 1 would be included in the set.

Translating this into SQL code, Cohen and Nissim asked queries of the following form:

```
SELECT count(clientId)
FROM loans
WHERE floor(100 * ((clientId * 2)^0.7) + 0.5) = floor(100 * ((clientId * 2)^0.7))
AND clientId BETWEEN 2000 and 3000
AND loanStatus = 'C'
```

Note that the last condition on `loanStatus` is the secret bit that they are attacking, and they only wished to attack the range with `clientId` between 2000 and 3000. They had just one condition as an overhead, adding only constant excess noise to each query – far less than the $O(\sqrt{n})$ which the Dinur-Nissim attack could handle! As a result, using essentially the same attack described above (with some modifications from [DMT07] to handle unbounded noise), they easily reconstructed the data perfectly, walking away with $5,000. Aircloud has recently opened a new challenge on an updated version of their system, so stay tuned to hear whether there are effective attacks.

References

[Abo19] John Abowd. Tweetorial: Reconstruction-abetted re-identification attacks and other traditional vulnerabilities. https://twitter.com/john_abowd/status/1114942180278272000, April 2019.

[CN20] Aloni Cohen and Kobbi Nissim. Linear program reconstruction in practice. *The Journal of Privacy and Confidentiality*, 10(1), 2020.

[DMT07] Cynthia Dwork, Frank McSherry, and Kunal Talwar. The price of privacy and the limits of lp decoding. In *Proceedings of the 39th Annual ACM Symposium on the Theory of Computing*, STOC '07, pages 85–94, New York, NY, USA, 2007. ACM.

[DN03] Irit Dinur and Kobbi Nissim. Revealing information while preserving privacy. In *Proceedings of the 22nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '03, pages 202–210, New York, NY, USA, 2003. ACM.

[DR14] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[GAM19] Simson Garfinkel, John M Abowd, and Christian Martindale. Understanding database reconstruction attacks on public data. *Communications of the ACM*, 62(3):46–53, 2019.