**Composition:**

**The Key to Differential Privacy’s Success**

* Auxiliary information
  + Even if adversary has partial information, protect against further privacy loss
* Composition
  + Even w multiple analyses, privacy maintained (smooth gradual degradation)
* Contrast 2 alternate approaches Cynthia touched on:
  + “just” releasing statistics
    - Say we’re looking at a sample of students. Say we release 2 statistics for this sample: the number of students that have a hearing disability (large university, this isn’t problematic is it?), and the number of students that have a hearing disability that are not a student president (another large, seemingly privacy protected). However, given these 2 answers, if you were to subtract the latter from the former, you may learn that the student president has a hearing disability.
  + De-identifying the data.

Composition:

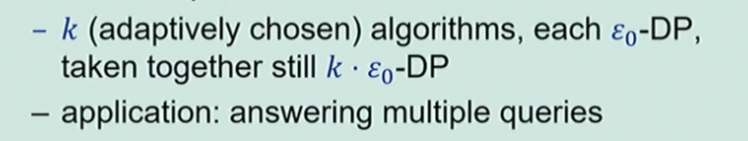
Privacy maintained even under multiple analyses. This is a core issue as it is:

1. Unavoidable: in reality, there are multiple analyses. No one collects a database of sensitive valuable information and then only runs one analyse on it. Even if someone were to only run one analysis, the data is in different data sets, so even if each data set is subject to only one analysis, the data is still in multiple analyses.
2. Programmable: private subroutines make for private algorithms. Composition is what makes DP programmable. As computer scientists, we can take algorithms or subroutines, put them together in innovative ways and get interesting differentially private algorithms.

Composition is important and we want to understand it, so the first question is “how exactly do we define composition?”. We want it to handle composition even when the adversary can be adaptive (so the adversary can choose what differentially private algorithms we run, and even when the adversary is in control of the data sets).

**Content:**

Basic composition:

* 
* The epsilons add up

Advanced Composition

**Basic Composition**

[need to expand on this equation more. Find a simplified example that explains it well]

Basic composition tells us that if we take *k* algorithms, each of which is epsilon-0 differentially private on its own, running all of them together, the epsilons just add up, we still have *k*\*epsilon-0 differential privacy.

Throughout this, think of epsilon-0 of epsilons as a small constant (you could even think of it as 1/*k* if we know in advance what *k* is), so even though the *k*\*epsilon-0 can be a very small quantity, we still have strong differential privacy.

**How do we show that basic composition holds?**

From about 8 minutes, there are a number of complicated equations; understand these before adding them to the latex document.

It’s a proof of the above statement, “Throughout this, think of epsilon-0 of epsilons as a small constant (you could even think of it as 1/*k* if we know in advance what *k* is), so even though the *k*\*epsilon-0 can be a very small quantity, we still have strong differential privacy.”.

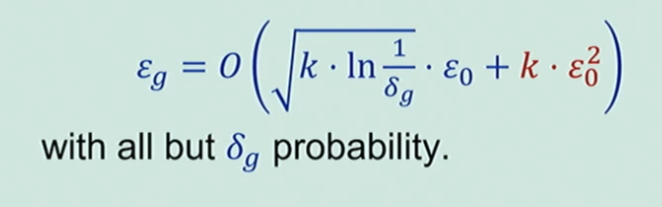
The application of this is answering queries. We can use the above to answer multiple adaptively chosen queries on a data set. We know how to answer a single statistical query on a dataset from the work of Dwork. The way we do it is, given a query, we just add some noise to it. The noise here is drawn from a Laplace distribution scaled to our privacy parameter: the smaller the privacy guarantee we want, the larger the noise we add. This gives us differential privacy, so long as the query has sensitivity 1; Laplace guarantees e0-differential privacy so long as its value can’t be affected substantially by any individual. Once we’ve collected a database of sensitive information, we want to do much more than answer a single statistical query, and composition already tells us that we can answer multiple queries. If we want to answer k queries, even k adaptively chosen queries, each of which has sensitivity 1, each of which can’t be affected too much by any 1 individual, all you have to do is add independent Laplace noise to each of the queries. The more queries we want to answer, the more noise we need to add; the noise scales as k/e [slide at 11:36]. For each of the queries, as we’re adding noise k/e, we have e0 = e/k-privacy per query. And since we’re answering k all together, the epsilons sum up and we have e-privacy for all k of the answers. Since we’re adding noise on the magnitude of k/e, the error we get per query is around k/e; . The relationship is: the more queries we answer, the more noise we have. The error depends linearly on the number of queries that are answered. For example, can answer sqrt(n) queries with an error of [0 symbol](sqrt(n)).

From this part, expand upon:

* Privacy guarantee
* Sensitivity 1
* Laplace

**Advanced Composition**

Composing k algorithms, each of which is e0-DP:



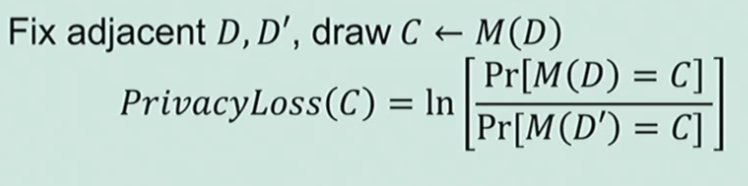
Compared with: eg = k \* e0 (basic composition).

When we look at the global privacy loss under composition, the way it scales is something like root k multiplied by epsilon-0 (compared with the basic composition where we had eg = k \* e0, we are no doing a lot better) plus k times epsilon-0 squared. For most choices of epsilon, this will be the dominant term and as epsilon-0 is much smaller than 1 we are doing much better. This is almost-always true; there is some probability of a privacy error (or of something going wrong), so we have the delta probability of some things going wrong. However, we could have a very small probability here, yet the cumulative privacy loss that we will pay is the square root of the log of that error probability. In spite of this probability of error, we will examine the intuition behind why this better theorem holds.

**What is privacy loss?**

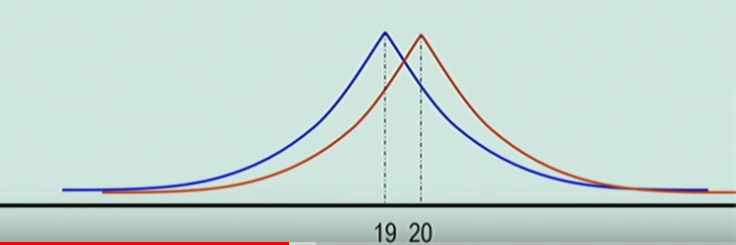
In order to understand the error probability (the error probability is only over the coins that the differential privacy algorithm is tossing). This privacy loss is actually a random variable. Differential privacy lets us quantify and reason over privacy loss, and we reason about it as follows:

* We fix our adjacent data sets D (the real data set with our data in it) and D’ (the data has been completely erased and taken out)
* Run the mechanism on the data set D, and publishes the outcome that tells us the correlation between, let’s say, smoking and cancer.



The privacy loss for this specific outcome that was generated is the log ratio of the probabilities of that outcome, when the algorithm was run on the real data set, and when it was run without the data (D and D’). This is a random variable because the outcome of this algorithm is randomised. Privacy loss lets us attach a number to this outcome; how privacy disclosive is this particular outcome that we got. This loss can be positive (in the event that we’ve lost some privacy), negative, or infinite (e.g. the alg decided to toss some coins and decided to release the private information, this is what is known as infinite privacy loss).

To understand why privacy loss can be negative, lets look at an example: the number of people that are over 6 feet tall. In the data set (D) the number is 20 and outside of it (D’), the number is 19. We’re also adding Laplace noise (observe the probability distribution function on these 2 data sets below, and we see that we’re adding Laplace noise to the number 19 and the number 20).



We run the algorithm for the real data set (20), and we receive an answer for the number of people above 6 feet. This answer (with a probability of at least a half) is going to be to the RHS of the ‘20’, and this is going to indicate that we are probably in the data set. This is because the outcomes to the right are more likely when we’re in the data set versus when we’re not in the data set, so the differential privacy [18:26] ratio is bigger than one and the log is positive.

However, suppose we run the algorithm and the noise turns out to be some really large negative number. Now we’re getting an outcome that would be more likely if we weren’t in the data set. Here the probability of being in D’ is larger than the probability of being in D, the log is less than one, the Privacy Loss is negative, so we have gained some privacy.

Because privacy loss can be negative (in this scenario, it can be negative with probability that’s close to 50%), so as long as the outcome is to the LHS of the intersection point of the curves, the loss will be negative and we will have gained some privacy. This is the advantage of advanced composition.

Really what differential privacy is actually telling us is that always, for any outcome we draw, the magnitude of the privacy loss is bounded by epsilon. The ratio is bounded by e^epsilon, its log is bounded by epsilon.

**Why is this important?**

We said composition was a core issue in the study of DP. When we have better composition guarantees, we can build better differentially private algorithms. WE CAN ALSO use more subroutines to get better accuracy for our analysis, or we can use the same number of subroutines and get a better privacy guarantee for the algorithm.

When we go back to the question of answering statistical queries using Laplace noise, this improved composition theorem lets us answer n queries (where n is the data set size) with error that roughly scales to sqrt(n); before we were only able to answer sqrt(n) queries, whereas now we are able to answer as many queries as the size of the data set by using independent Laplace noise.

**How do we prove advanced composition?**

When we think of privacy loss as a random variable we can study it; it has a mean, it has variance, etc. An interesting note on it is its expectation; if we look at the privacy loss of any epsilon-DP algorithm we know that the privacy loss is bounded by epsilon (no less than epsilon), but the expectation will always be much better than this worst-case bound because we have negative privacy losses. We can show that the expected privacy loss for any epsilon-DP is at most epsilon^2. So now we have this random variable privacy loss, it has a small expectation (epsilon^2), and we know that the absolute magnitude is always bounded by epsilon. The way composition is shown is by modelling cumulative loss from multiple analyses/algorithms as a martingale.

We have random variables that are bounded expectation and total magnitude. We can use Azuma’s inequality to show that the probability of a large loss is tightly bounded, and this powers the advanced composition theorem.

**Recap**

So far we have:

Basic composition:

*k* analyses, each epsilon-0 differentially private 🡪 an error of about epsilon-0 \* *k*-DP

Advanced composition:

*k* analyses, each epsilon-0 differentially private 🡪 an error of about epsilon-0^2 \* *k*-DP

To reach a more refined understanding we need to consider:

* Tight bounds
* The complexity of composition
* An alternative: Concentrated Differential Privacy

**Optimal DP Composition**

For given mechanisms (M1, M2, …, Mk), what are the best (epsilong, deltag) we can claim? (best over all events).

We had basic composition that had one bound and was far from optimal. Then we had advanced composition that provided a better bound.

For given bounds ((e1, d1), … , (ek, dk)), what are the best (eg, dg) we can claim? (best over all mechanisms, events).

We want to know the best over all possible mechanisms in terms of the privacy loss that we can claim. This would innately let us put together any building blocks. All that would need to be specified about the algorithm is its epsilon and its delta, then this algorithm is combined with other algorithms with their own respective epsilon and delta that are specified. This would let us know what their privacy guarantees are, and therefore what the combined privacy guarantee is.

Homogeneous case:

Tight bounds occur when for all *i,* epsiloni = epsilon, di = d i.e. all the epsilons are the same for all of the k algorithms, and all of the deltas are the same for all of the *k* algorithms. This found a 30-50% improvement over the advanced composition theorem. The advanced composition theorem is asymptotically tight, but when we’re composing differentially private algorithms we want to achieve tight bounds, we don’t want to pay large constant factors.

Heterogeneous case:

For general epsilons and deltas (ei, di), not all mechanisms need to have the same bounds. This is a complicated expression, not just visually, it’s also computationally intensive to compute what the bound is. Given a collection of mechanisms, it would take exponential time to compute what the actual privacy loss is (it is unlikely to take less than exp(k) time). Computing the total global epsilon is #P-complete. This is even harder than an NP-complete problem. As we don’t expect to compute this in less than exp(k) time, we have an existential result; we know that we can characterise the cumulative loss, but we have no reasonable method to compute exactly what it is. There is an additive approximation algorithm that is given the bounds of the individual algorithms for computing the total privacy loss.

**Composition Goals**

* Efficient composition. Polynomial time to compute privacy losses would be great.
* Better bounds and smaller errors.
  + We don’t want to think about the worst case for all the algorithms that may satisfy a certain (epsilon, delta).
* One thing that we should really focus on (where there are stark differences between given algorithms) is in the delta guarantee. We can distinguish between two deltas that we may have:
  + The delta of death and destruction.
    - These are algorithms where it is true that with probability delta all privacy is lost.
    - E.g. an algorithm that looks at the data set and, with probability delta, releases all personal information, medical history, tax reports, etc.
  + The deltas that we often see, the deltas that come due to concentration bounds.
    - Plays it safe.
    - Loss is usually smaller than epsilon.
    - Gaussian noise, advanced composition.

**Concentrated Differential Privacy**

A relaxation of differential privacy that makes the privacy loss random variable have a small mean and be tightly concentrated around its mean. So now the privacy guarantee is quantified by two parameters: (μ, τ2)

Where μ is the expectation and τ2 is the variance

We have the privacy loss has expectation bounded by mue, and its tightly concentrated around this expectation. Intuitively, the concentration should be no worse than that of a regular normal variable.

“Concentration no worse than N(μ, τ2)

Formally, the privacy loss is Subgaussian random variable.

This is a relaxation of differential privacy as we’re giving smaller probabilities of larger privacy losses while maintaining the important advantages of differential privacy particularly for composition. It composes automatically, optimally and efficiently; this is a definition geared towards composition. It also handles auxiliary data and is therefore future proof, similarly to standard differential privacy.

**Other texts**

<https://scholar.google.com/scholar?hl=en&as_sdt=0%2C5&q=research+data+privacy&btnG=>

<https://towardsdatascience.com/understanding-differential-privacy-85ce191e198a>

<https://scholar.google.com.au/scholar?q=definition+of+differential+privacy+simplified&hl=en&as_sdt=0&as_vis=1&oi=scholart>

<https://www.google.com/search?client=firefox-b-d&q=does+REDCap+use+differential+privacy%3F>

Proposed test release network