

差分隐私机器学习:理论、算法与应用

<http://t.cn/RQWfiWo>

【差分隐私机器学习:理论、算法与应用】《Differentially Private Machine Learning: Theory, Algorithms and Applications(NIPS 2017 Tutorial)》 by Kamalika Chaudhuri, Anand D Sarwate <http://t.cn/RQWfiWo>

<https://www.bilibili.com/video/av18545545?from=search&seid=12596259961554168901>

Logistics and Goals

- **Tutorial Time:** 2 hr (15 min break after first hour)
- **What this tutorial will do:**
 - Motivate and define differential privacy
 - Provide overview of common methods and tools to design differentially private ML algorithms
- **What this tutorial will not do:**
 - Provide detailed results on the state of the art in differentially private versions of specific problems

Learning Outcomes

At the end of the tutorial, you should be able to:

- Explain the definition of differential privacy,
- Design basic differentially private machine learning algorithms using standard tools,
- Try different approaches for introducing differential privacy into optimization methods,
- Understand the basics of privacy risk accounting,
- Understand how these ideas blend together in more complex systems.

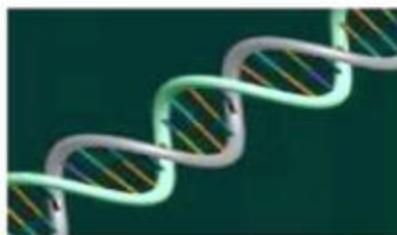
Motivating Differential Privacy

Sensitive Data

Medical Records



Genetic Data



Search Logs



AOL Violates Privacy

A Face Is Exposed for AOL Searcher No. 4417749

By MICHAEL BARBARO and TOM ZELLER Jr.

Published: August 9, 2006

Buried in a list of 20 million Web search queries collected by AOL and recently released on the Internet is user No. 4417749. The number was assigned by the company to protect the searcher's anonymity, but it was not much of a shield.



No. 4417749 conducted hundreds of searches over a three-month period on topics ranging from "numb fingers" to "60 single men" to "dog that urinates on

Netflix Violates Privacy [NS08]



	Movies				
User 1					
User 2					
User 3					

2-8 movie-ratings and dates for Alice reveals:

- Whether Alice is in the dataset or not
- Alice's other movie ratings

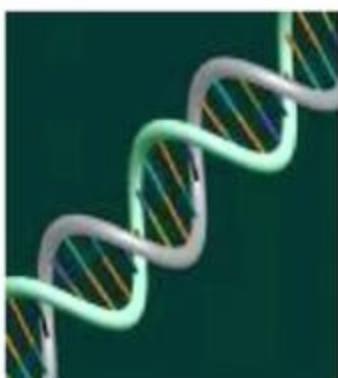
High-dimensional Data is Unique

Example: UCSD Employee Salary Table

Position	Gender	Department	Ethnicity	Salary
Faculty	Female	CSE	SE Asian	-

One employee (Kamalika) fits description!

Disease Association Studies [WLWTZ09]



Cancer

100
190 1.00
210 251 1.00
180 117 .047 1.00
154 011 .170 .083 1.00
190 140 .122 .095 .039 1.00
270 215 254 .248 .140 .141 1.00
101 .065 .170 .056 .034 .099 .175 1.00
230 071 .153 .111 .061 .093 .159 .157 1.00
471 .117 .243 .294 .144 .123 .253 .216 .274 1.00
173 .202 .132 .294 .087 .159 .257 .108 .362 .294 1.00

Healthy

Correlations

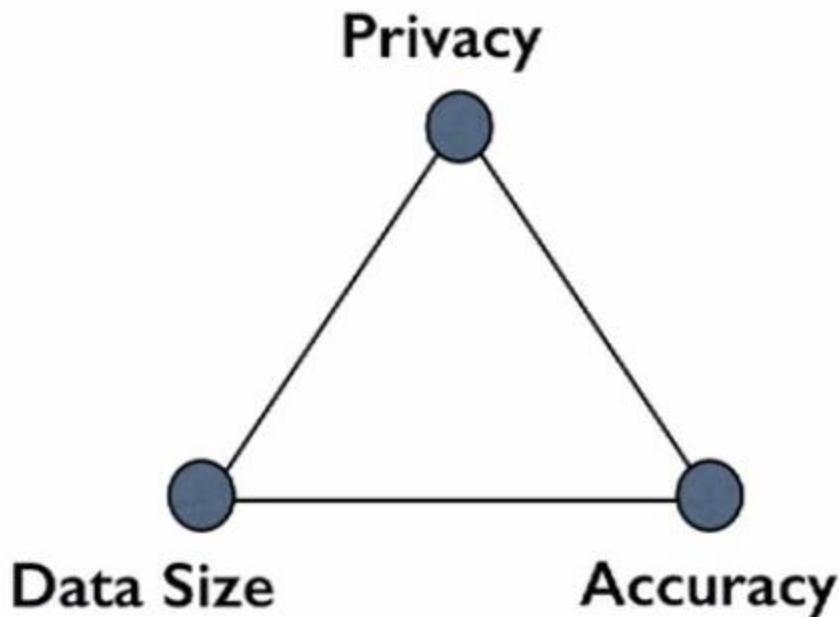
Correlations

Correlation (R^2 values), Alice's DNA reveals:

If Alice is in the **Cancer** set or **Healthy** set

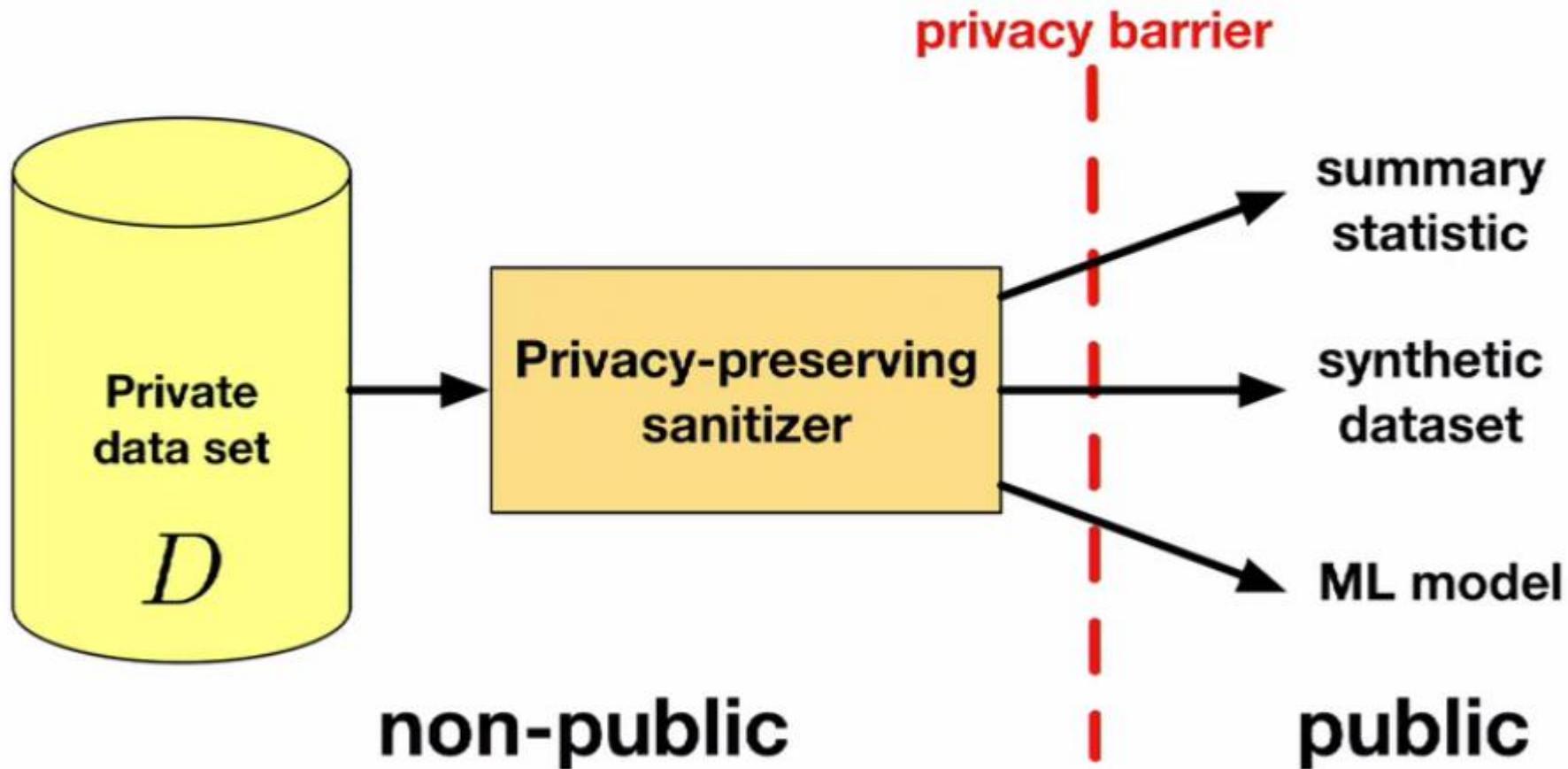
Simply anonymizing data is unsafe!

Statistics on small data sets is unsafe!

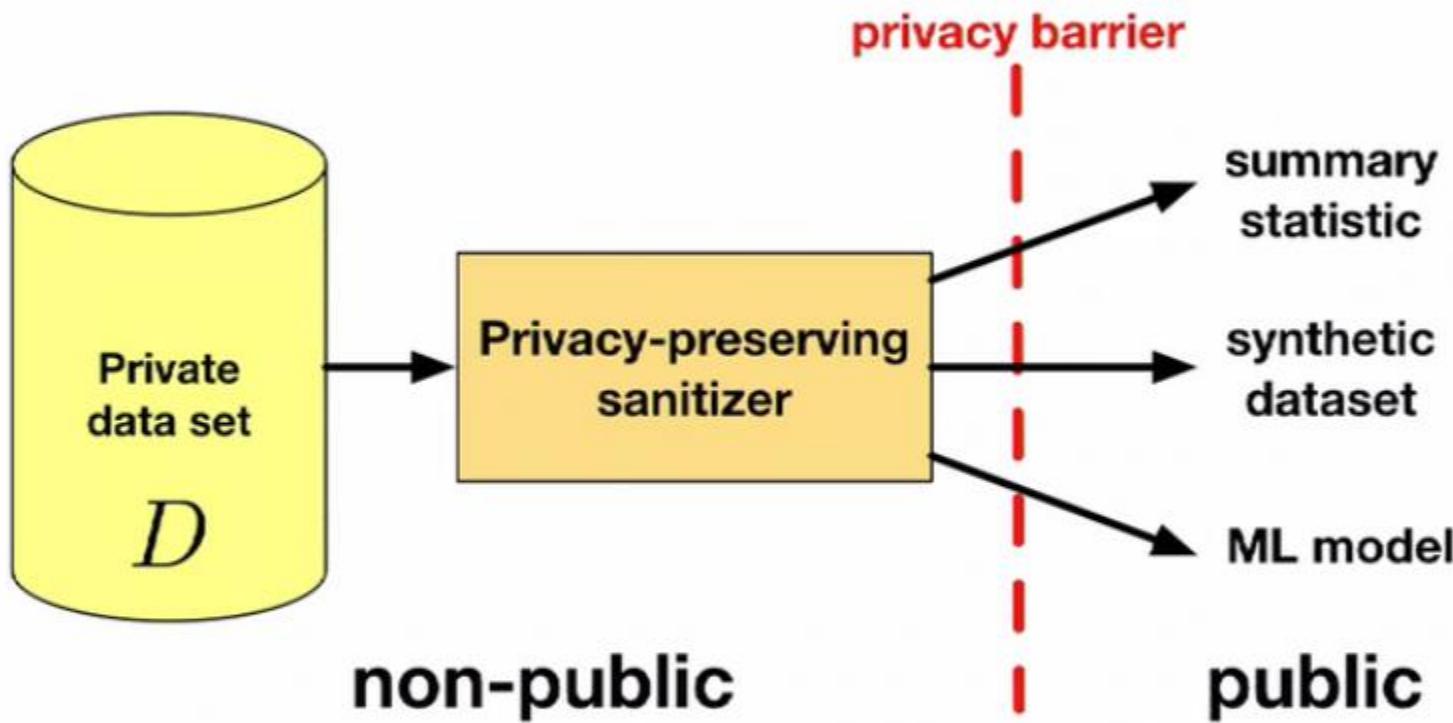


Privacy Definition

The Setting



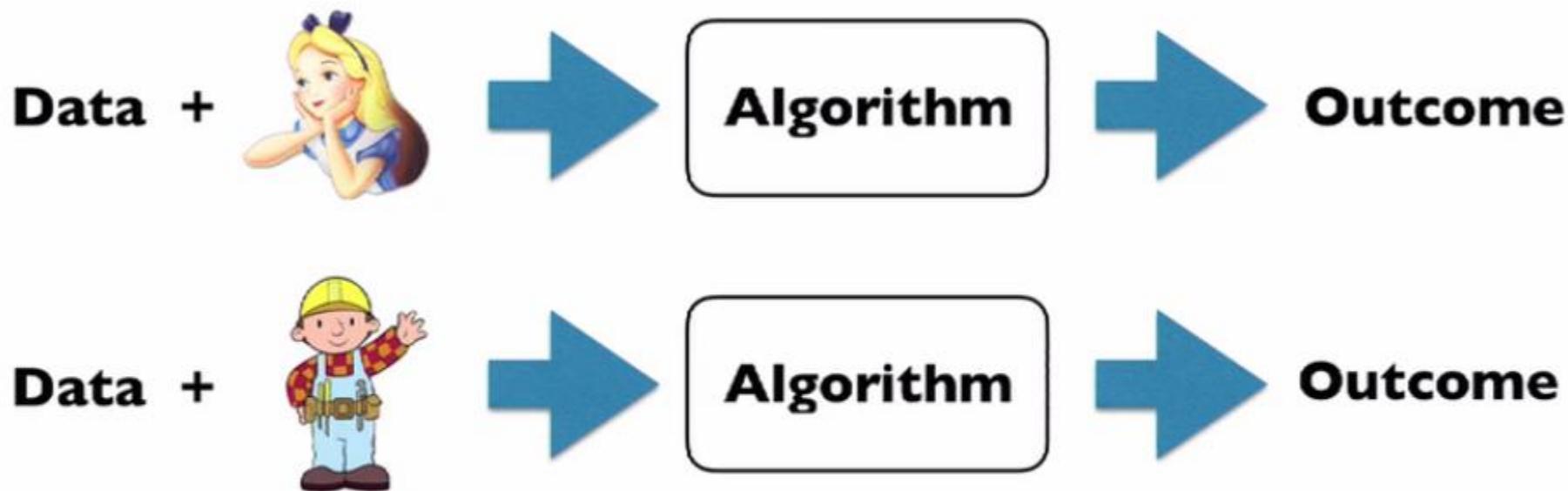
Property of Sanitizer



Aggregate information computable

Individual information protected
(robust to side-information)

Differential Privacy



Participation of a person does not change outcome

Since a person has agency, they can decide
to participate in a dataset or not

Adversary



Prior Knowledge:
A's Genetic profile
A smokes

Case I: Study

Cancer

[Study violates A's privacy]

Case 2: Study



Smoking causes cancer

A has
cancer

A probably
has cancer

[Study does not violate privacy]

How to ensure this?

...through randomness

$$A(Data +)$$



Random
variables

have close
distributions

$$A(Data +)$$



How to ensure this?

Random variables

$A(\mathbf{Data} + \dots)$



have close distributions

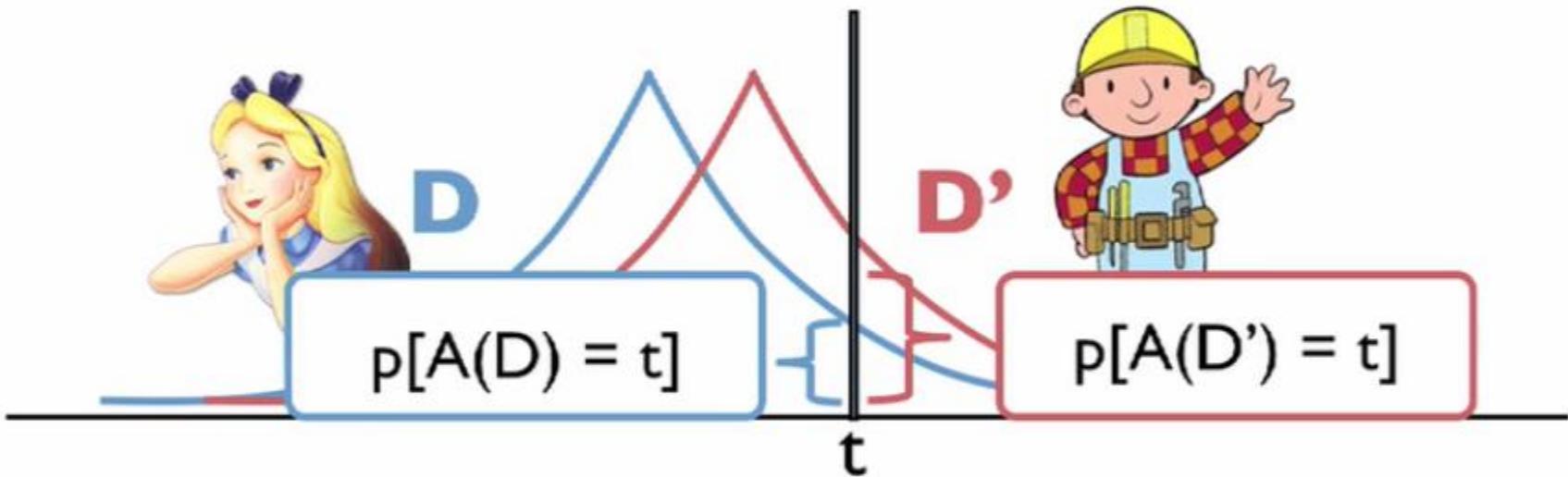
$A(\mathbf{Data} + \dots)$



Randomness: Added by randomized algorithm A

Closeness: Likelihood ratio at every point bounded

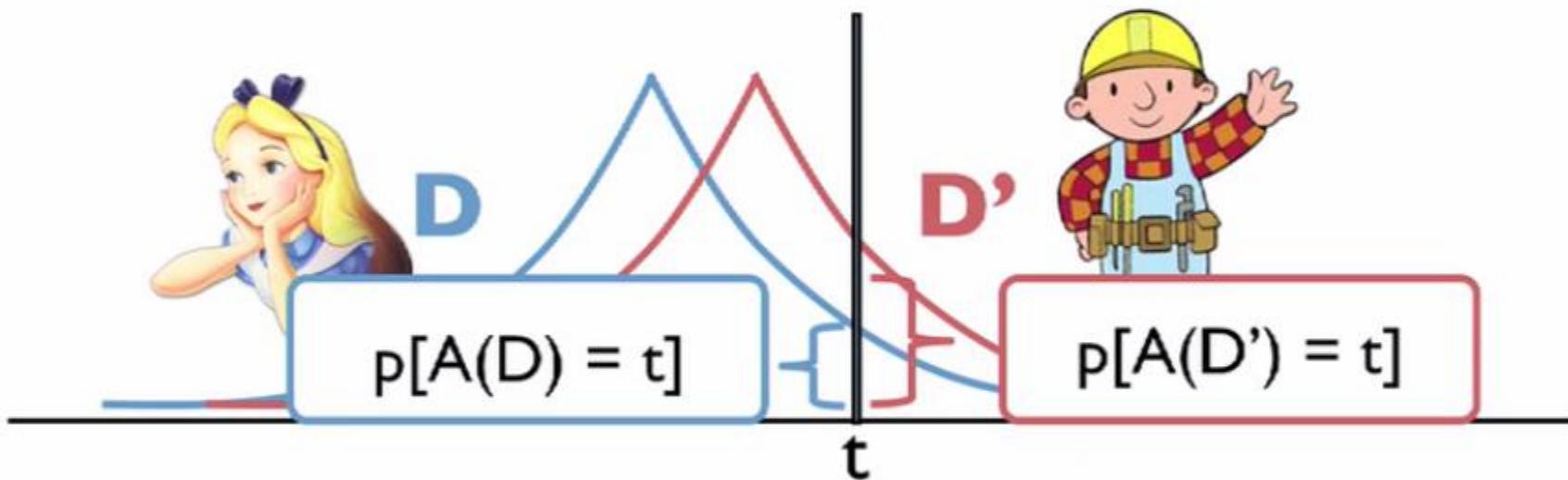
Differential Privacy [DMNS06]



For all D, D' that differ in one person's value,
If A = ϵ -differentially private randomized algorithm, then:

$$\sup_t \left| \log \frac{p(A(D) = t)}{p(A(D') = t)} \right| \leq \epsilon$$

Approx. Differential Privacy [DKM+06]



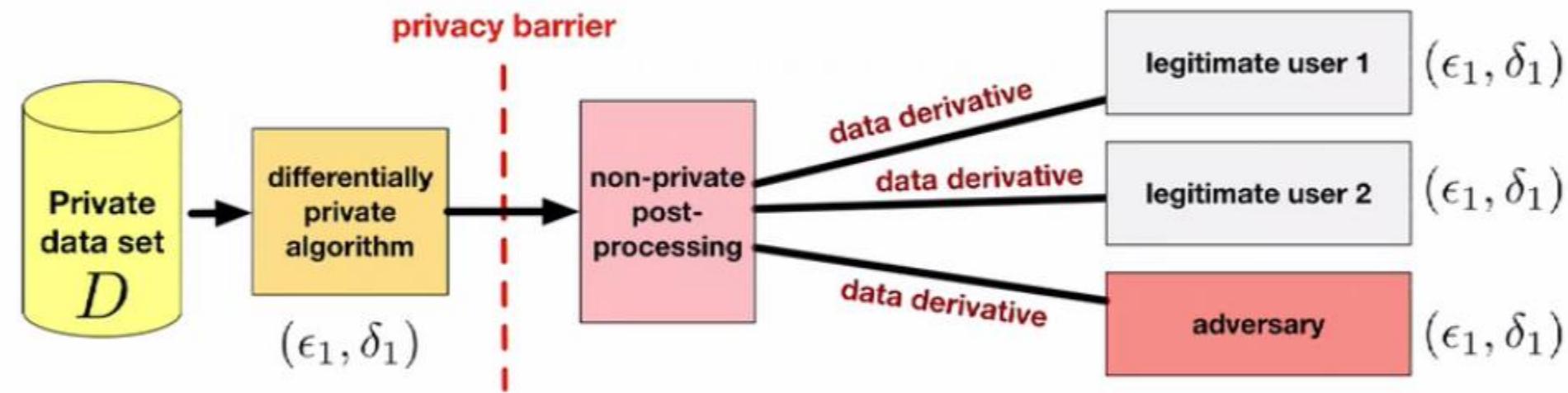
For all D, D' that differ in one person's value,

If $A = (\epsilon, \delta)$ -differentially private randomized algorithm, then:

$$\max_{S, \Pr(A(D) \in S) > \delta} \left[\log \frac{\Pr(A(D) \in S) - \delta}{\Pr(A(D') \in S)} \right] \leq \epsilon$$

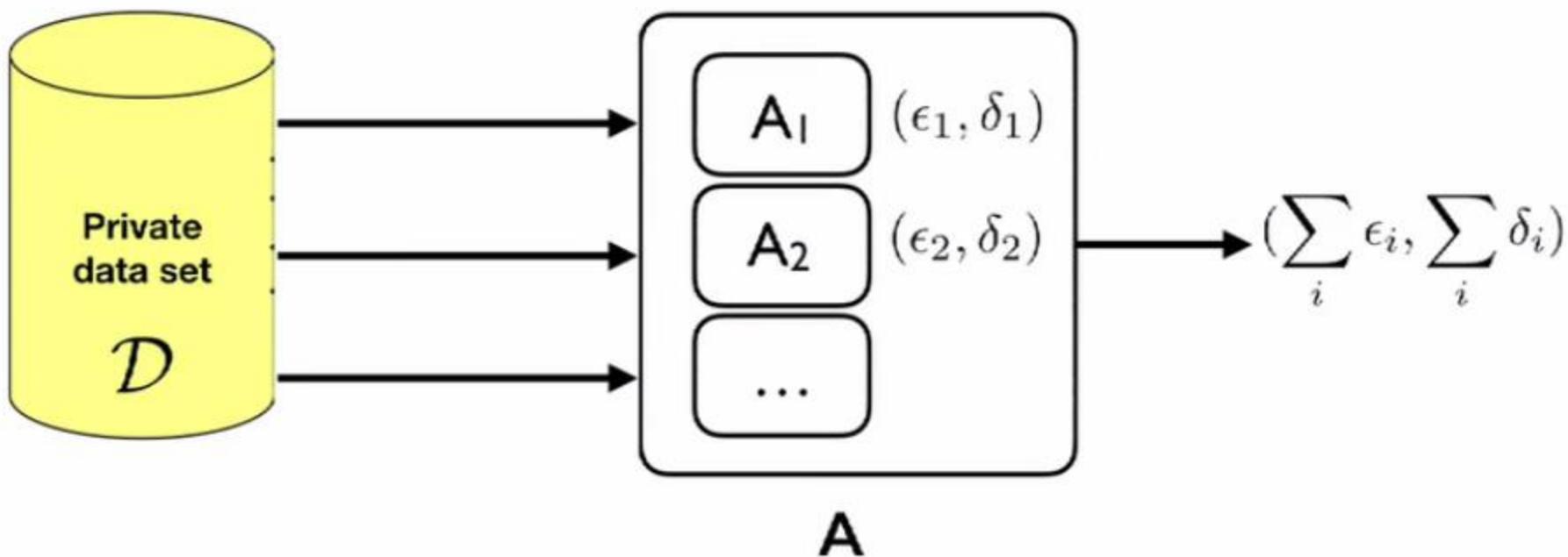
Properties of Differential Privacy

Property I: Post-processing Invariance



Risk doesn't increase if you don't touch the data again

Property 2: Graceful Composition



Total privacy loss is the sum of privacy losses

(Better composition possible — coming up later)

How to achieve Differential Privacy?

Tools for Differentially Private Algorithm Design

- Global Sensitivity Method [DMNS06]
- Exponential Mechanism [MT07]

Many others we will not cover [DL09, NRS07, ...]

Global Sensitivity Method [DMNS06]

Problem:

Given function f , sensitive dataset D

Find a differentially private approximation to $f(D)$

Example: $f(D) = \text{mean of data points in } D$

The Global Sensitivity Method [DMNS06]

Given: A function f , sensitive dataset D

Define: $\text{dist}(D, D') = \#\text{records that } D, D' \text{ differ by}$

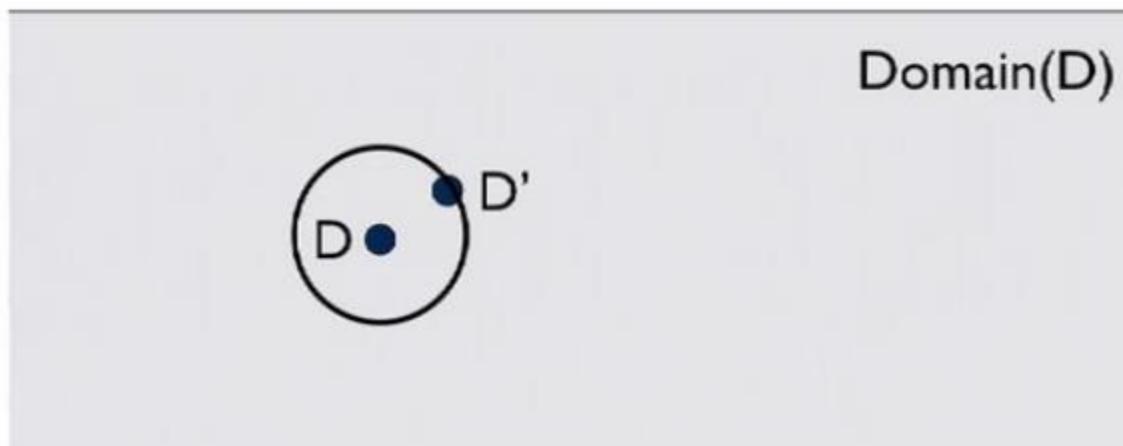
The Global Sensitivity Method [DMNS06]

Given: A function f , sensitive dataset D

Define: $\text{dist}(D, D') = \#\text{records that } D, D' \text{ differ by}$

Global Sensitivity of f :

$$S(f) = |f(D) - f(D')|$$



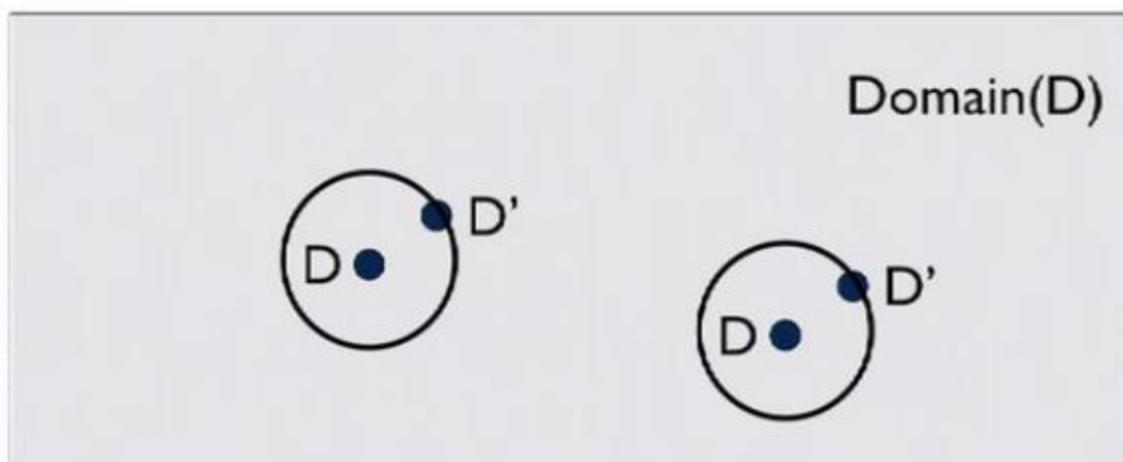
The Global Sensitivity Method [DMNS06]

Given: A function f , sensitive dataset D

Define: $\text{dist}(D, D') = \#\text{records that } D, D' \text{ differ by}$

Global Sensitivity of f :

$$S(f) = \max_{\text{dist}(D, D') = 1} |f(D) - f(D')|$$



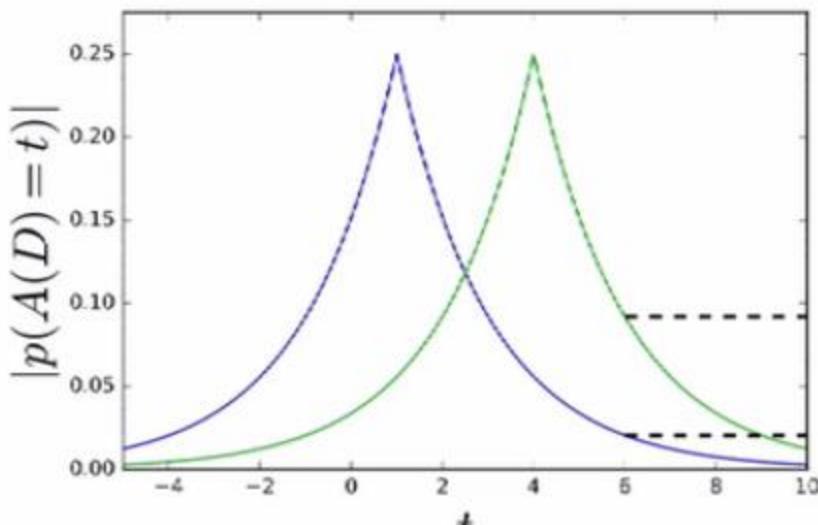
Laplace Mechanism

Global Sensitivity of f is $S(f) = \max_{\text{dist}(D, D') = 1} |f(D) - f(D')|$

Output $f(D) + Z$, where

$$Z \sim \frac{S(f)}{\epsilon} \text{Lap}(0, 1)$$

ϵ -differentially
private



Laplace distribution:

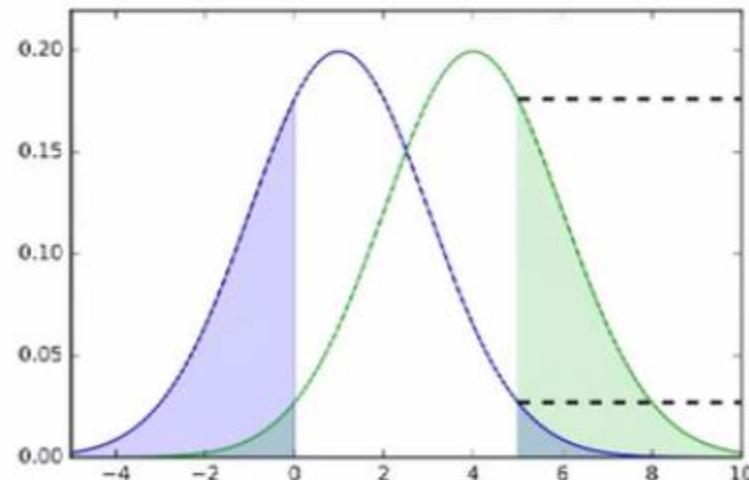
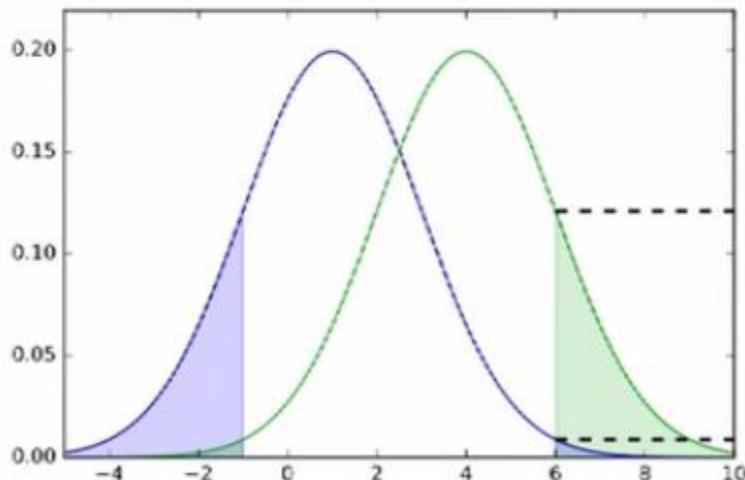
$$p(z|\mu, b) = \frac{1}{2b} \exp\left(-\frac{|z - \mu|}{b}\right)$$

Gaussian Mechanism

Global Sensitivity of f is $S(f) = \max_{\text{dist}(D, D') = 1} |f(D) - f(D')|$

Output $f(D) + Z$, where

$$Z \sim \frac{S(f)}{\epsilon} \mathcal{N}(0, 2 \ln(1.25/\delta)) \quad (\epsilon, \delta)\text{-differentially private}$$



Example I: Mean

$f(D) = \text{Mean}(D)$, where each record is a scalar in $[0, 1]$

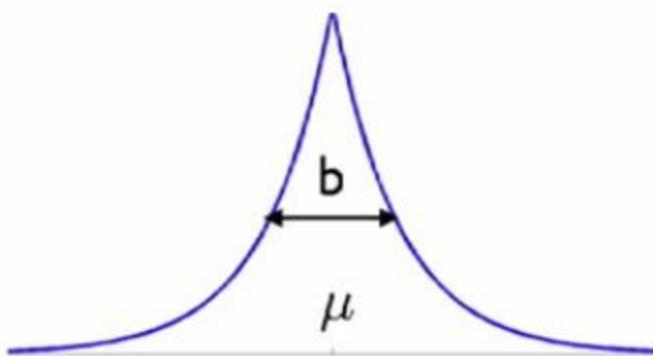
Example I: Mean

$f(D) = \text{Mean}(D)$, where each record is a scalar in $[0, 1]$

Global Sensitivity of $f = 1/n$

Laplace Mechanism:

Output $f(D) + Z$, where $Z \sim \frac{1}{n\epsilon} \text{Lap}(0, 1)$



How to get Differential Privacy?

- Global Sensitivity Method [DMNS06]
 - Two variants: Laplace and Gaussian
- Exponential Mechanism [MT07]

Many others we will not cover [DL09, NRS07, ...]

Exponential Mechanism [MT07]

Problem:

Given function $f(w, D)$, Sensitive Data D

Find differentially private approximation to

$$w^* = \operatorname{argmax}_w f(w, D)$$

Example: $f(w, D)$ = accuracy of classifier w on dataset D

The Exponential Mechanism [MT07]

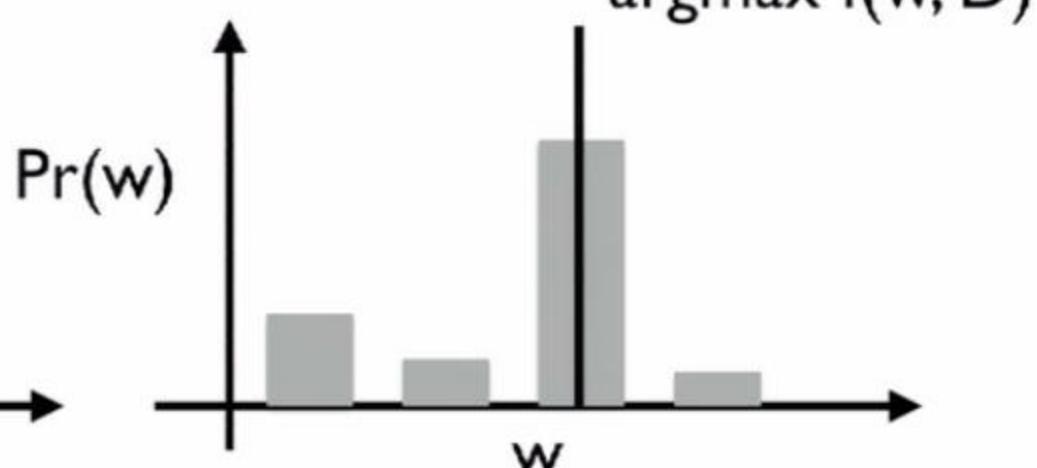
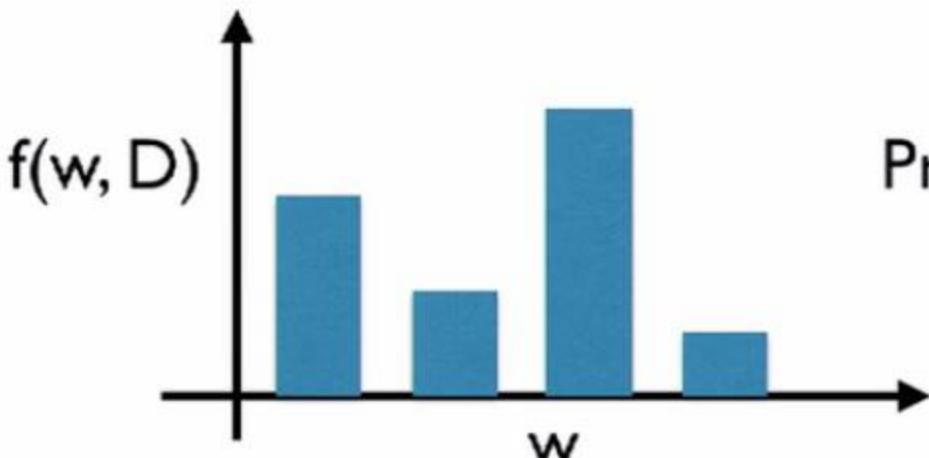
Suppose for any w ,

$$|f(w, D) - f(w, D')| \leq S$$

when D and D' differ in 1 record. Sample w from:

$$p(w) \propto e^{\epsilon f(w, D)/2S}$$

for ϵ -differential privacy.



Example: Parameter Tuning

Given validation data D , k classifiers w_1, \dots, w_k
(privately) find the classifier with highest accuracy on D

Here, $f(w, D)$ = classification accuracy of w on D

For any w , any D and D' that differ by one record,

$$|f(w, D) - f(w, D')| \leq \frac{1}{|D|}$$

So, the exponential mechanism outputs w_i with prob:

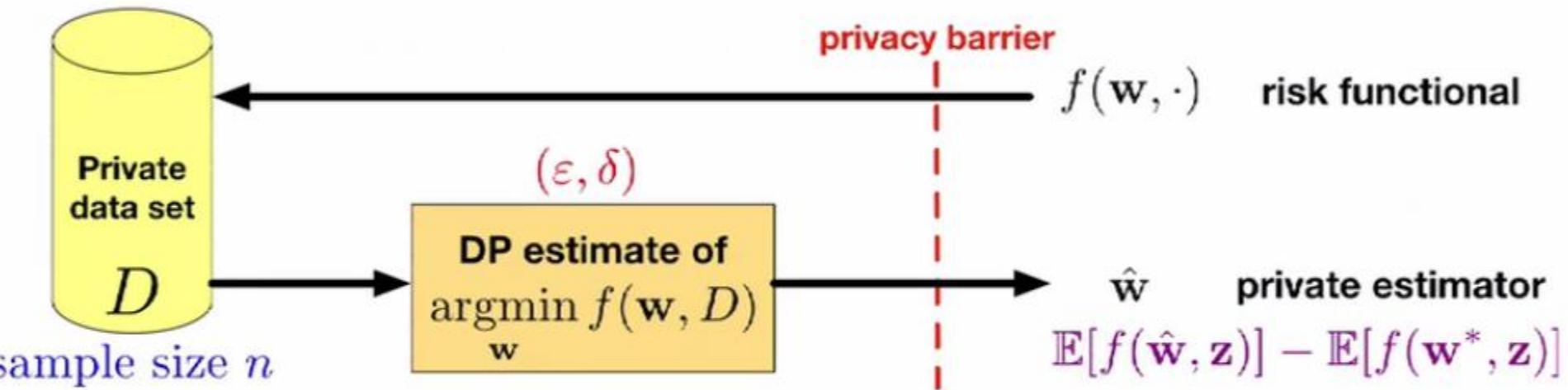
$$\Pr(w_i) \propto e^{\epsilon |D| f(w_i, D) / 2}$$

Summary

- Motivation
- What is differential privacy?
- Basic differential privacy algorithm design tools:
 - The Global Sensitivity Method
 - Laplace Mechanism
 - Gaussian Mechanism
 - Exponential Mechanism

Differential privacy in estimation and prediction

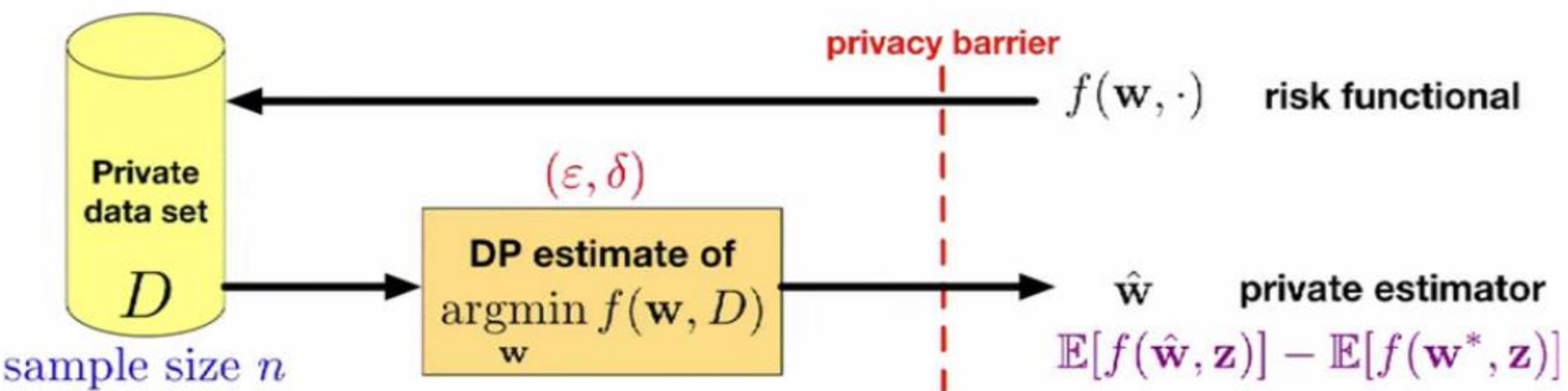
Estimation and prediction problems



Statistical estimation: estimate a parameter or predictor using private data that has good expected performance on future data.

Goal: Good **privacy-accuracy-sample size tradeoff**

Privacy and accuracy make different assumptions about the data



Privacy – differential privacy makes *no assumptions on the data distribution*: privacy holds unconditionally.

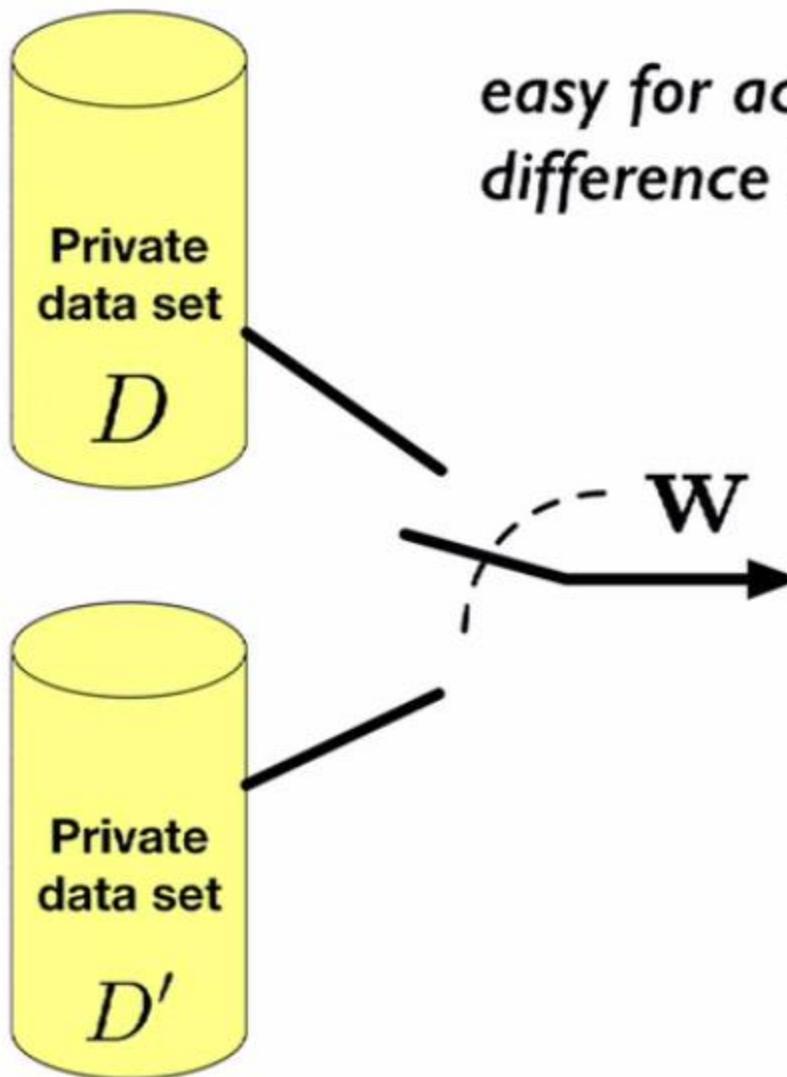
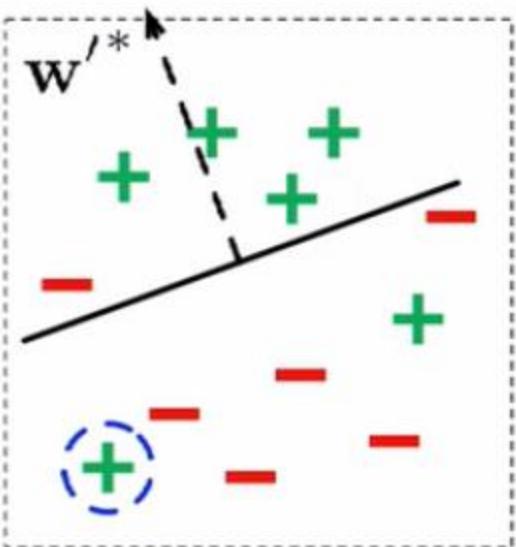
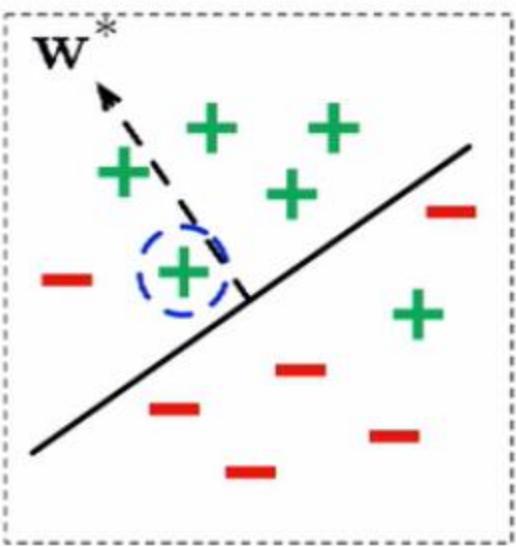
Accuracy – accuracy measured w.r.t a “true population distribution”: expected excess statistical risk.

Statistical Learning as Risk Minimization

$$\mathbf{w}^* = \operatorname{argmin}_{\mathbf{w}} \frac{1}{n} \sum_{i=1}^n \ell(\mathbf{w}, (\mathbf{x}_i, y_i)) + \lambda R(\mathbf{w})$$

- *Empirical Risk Minimization (ERM)* is a common paradigm for prediction problems.
- Produces a predictor \mathbf{w} for a label/response y given a vector of features/covariates \mathbf{x} .
- Typically use a convex loss function and regularizer to “prevent overfitting.”

Why is ERM not private?



easy for adversary to tell the
difference between D and D'

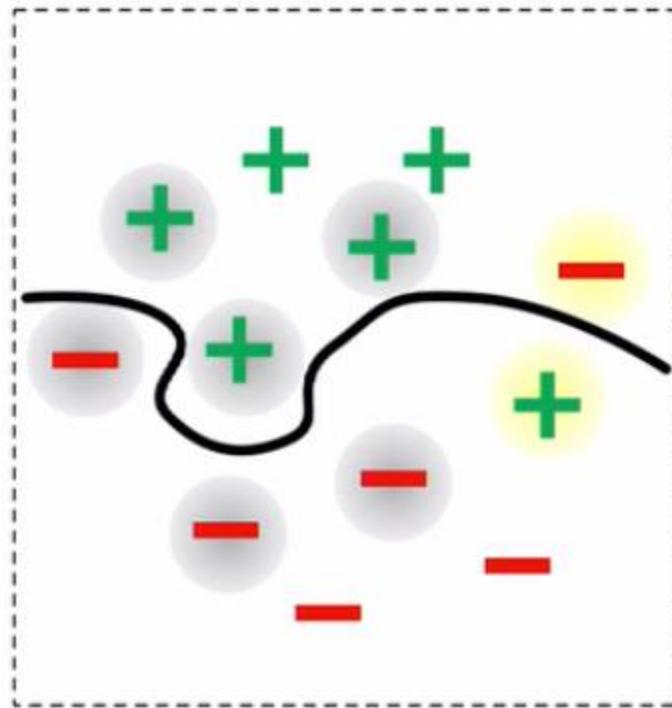


[CMSI I, RBHT12]

Kernel learning: even worse

- Kernel-based methods produce a classifier that is a function of the data points.
- Even adversary with black-box access to w could potentially learn those points.

$$w(x) = \sum_{i=1}^n \alpha_i k(x, x_i)$$



[CMSII]

Privacy is compatible with learning

- Good learning algorithms *generalize* to the population distribution, not individuals.
- *Stable learning algorithms* generalize [BE02].
- Differential privacy can be interpreted as a form of stability that also implies generalization [BNS+15].
- Two parts of the same story:
Privacy implies *generalization* asymptotically.
Tradeoffs between *privacy-accuracy-sample size* for finite n .

Revisiting ERM

$$\mathbf{w}^* = \operatorname*{argmin}_{\mathbf{w}} \frac{1}{n} \sum_{i=1}^n \ell(\mathbf{w}, (\mathbf{x}_i, y_i)) + \lambda R(\mathbf{w})$$

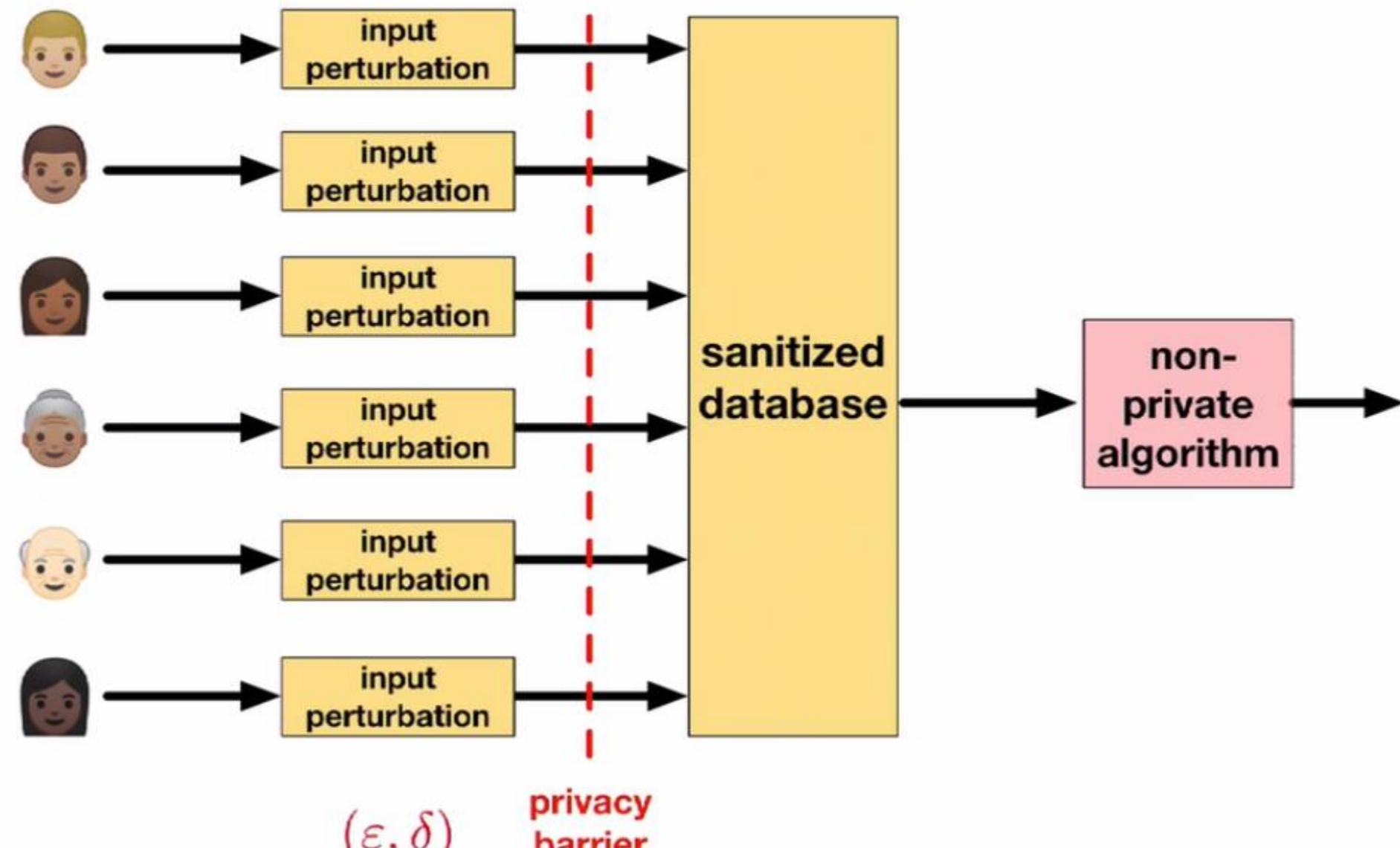
- Learning using (convex) optimization uses three steps:
 1. read in the data
 2. form the objective function
 3. perform the minimization
- We can try to introduce privacy in each step!

Revisiting ERM

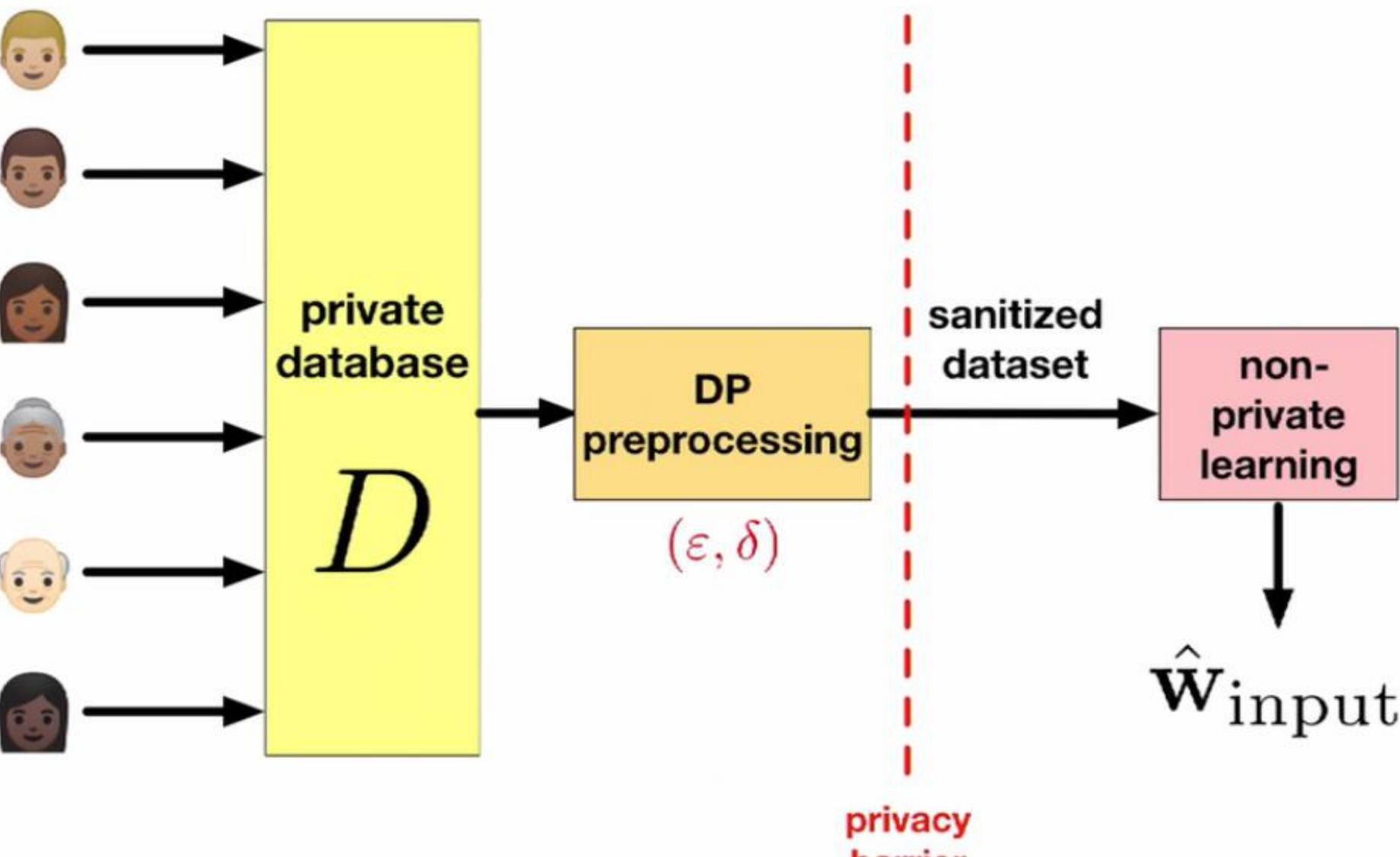
$$\mathbf{w}^* = \operatorname{argmin}_{\mathbf{w}} \frac{1}{n} \sum_{i=1}^n \ell(\mathbf{w}, (\mathbf{x}_i, y_i)) + \lambda R(\mathbf{w})$$

- Learning using (convex) optimization uses three steps:
 1. read in the data **input perturbation**
 2. form the objective function **objective perturbation**
 3. perform the minimization **output perturbation**
- We can try to introduce privacy in each step!

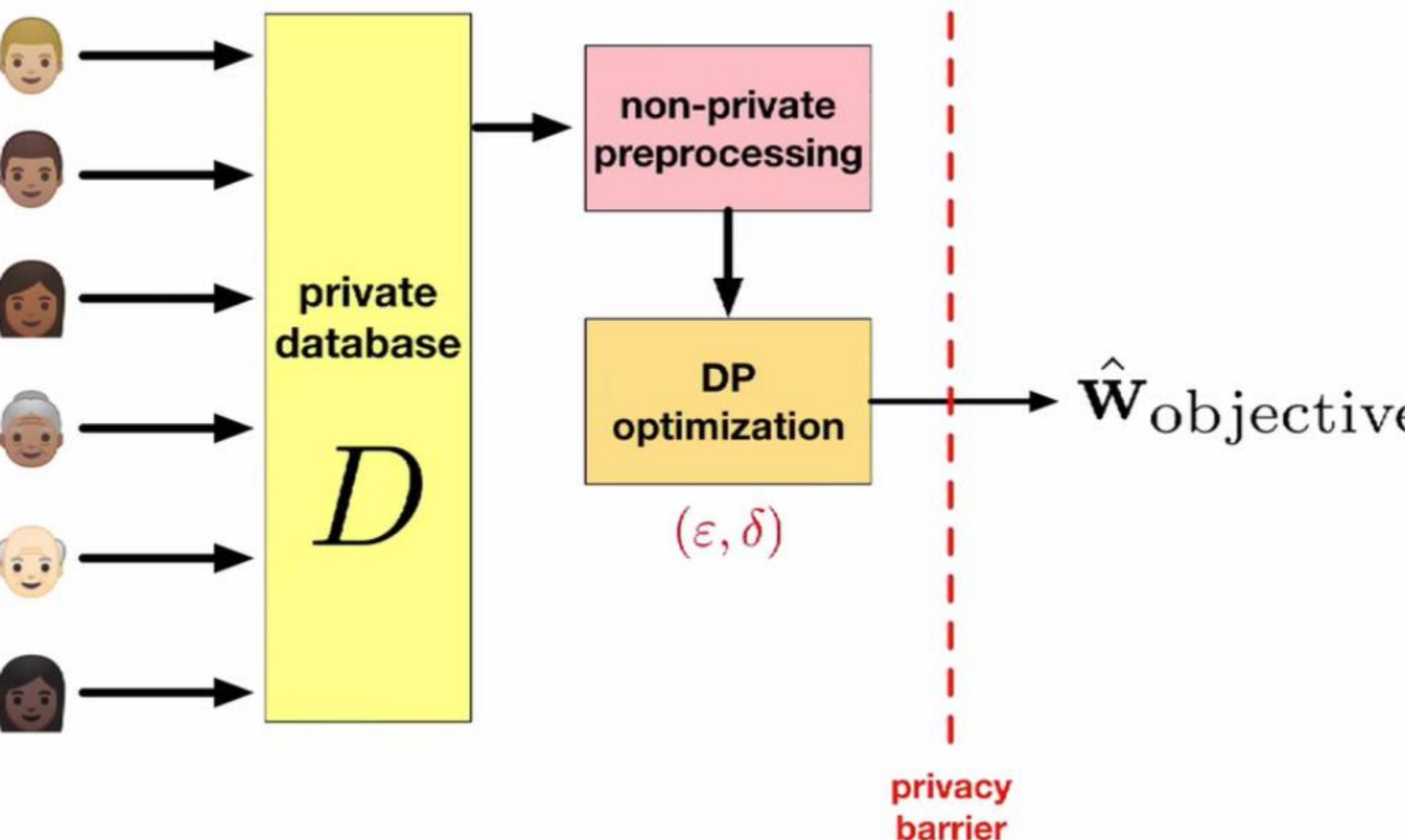
Privacy in ERM: options



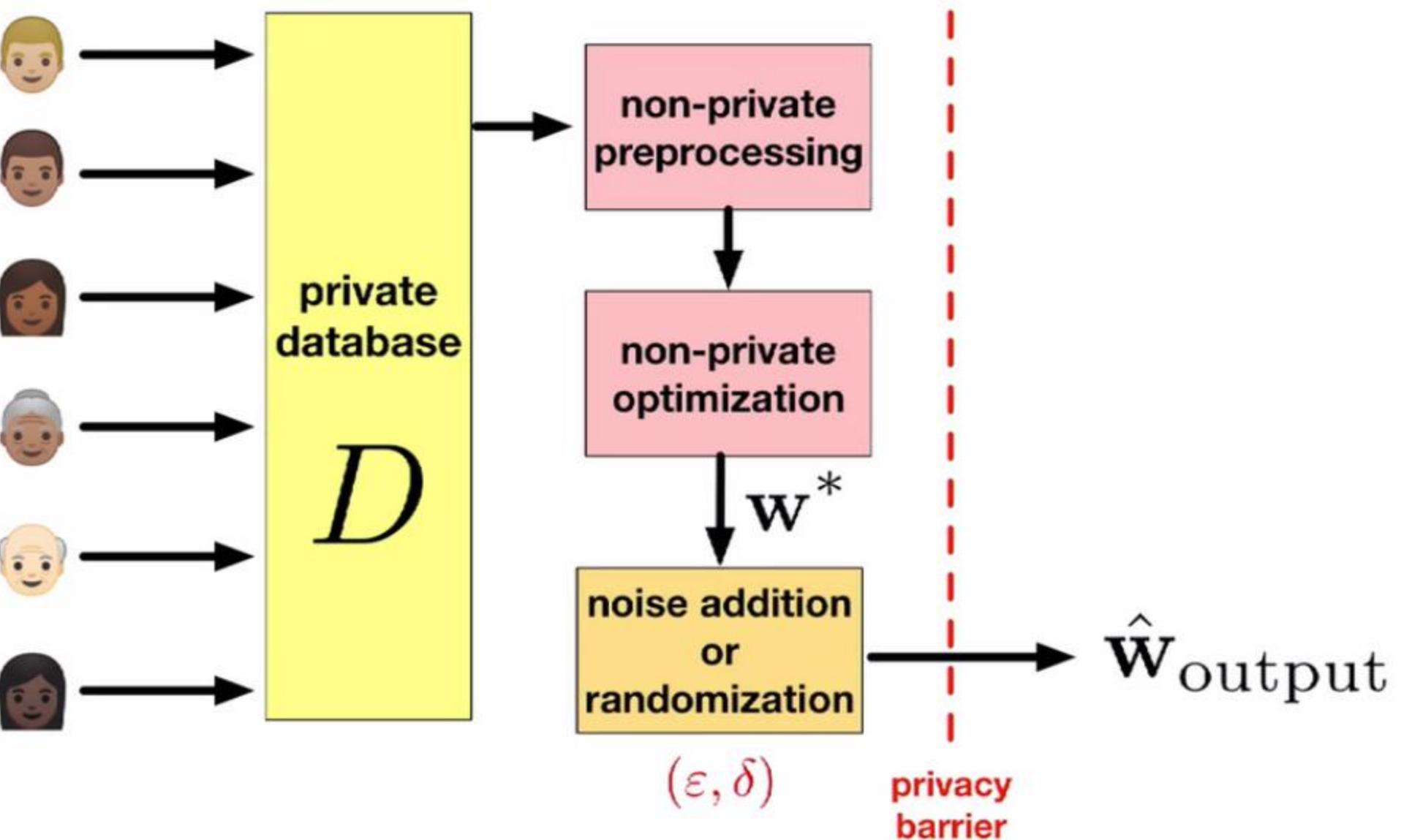
Privacy in ERM: options



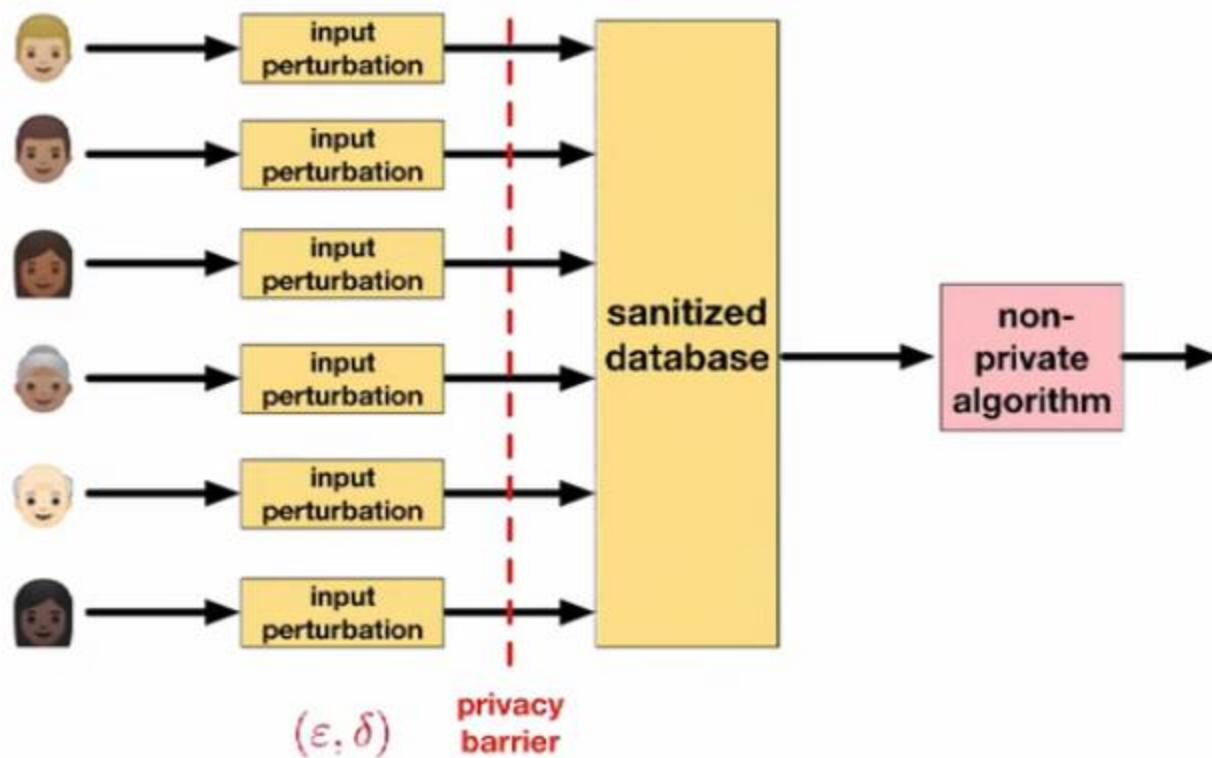
Privacy in ERM: options



Privacy in ERM: options

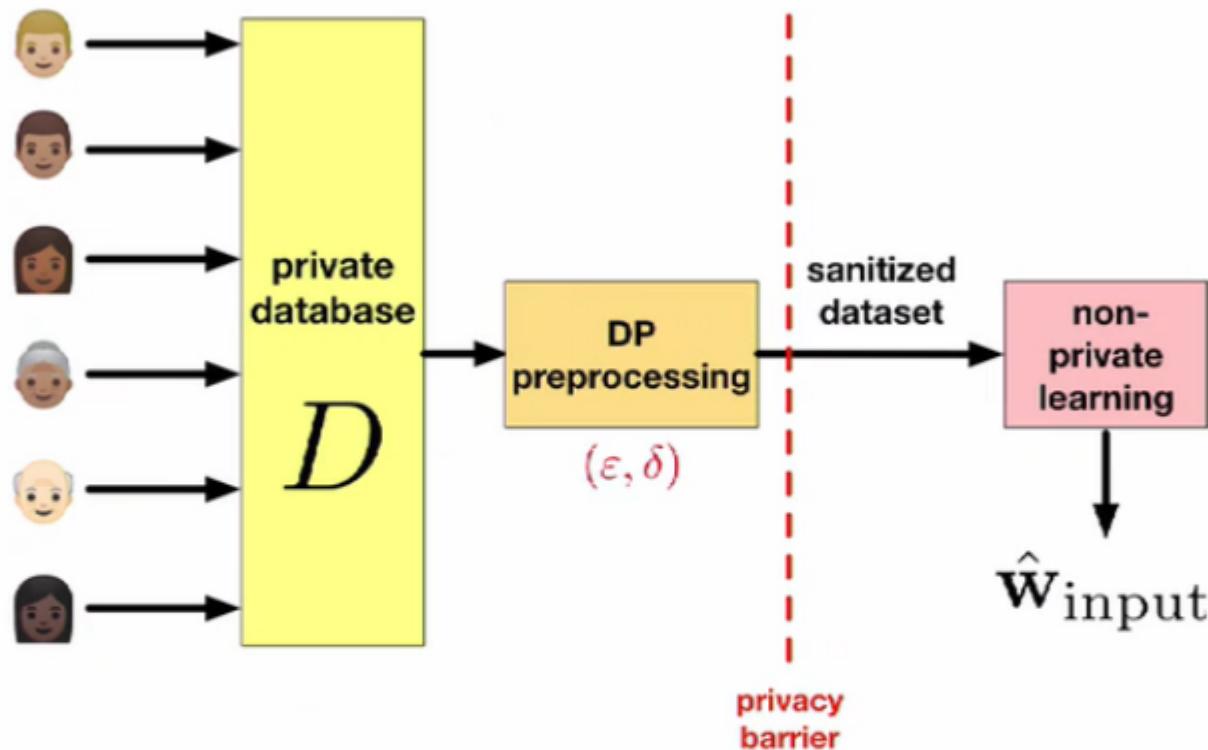


Local Privacy



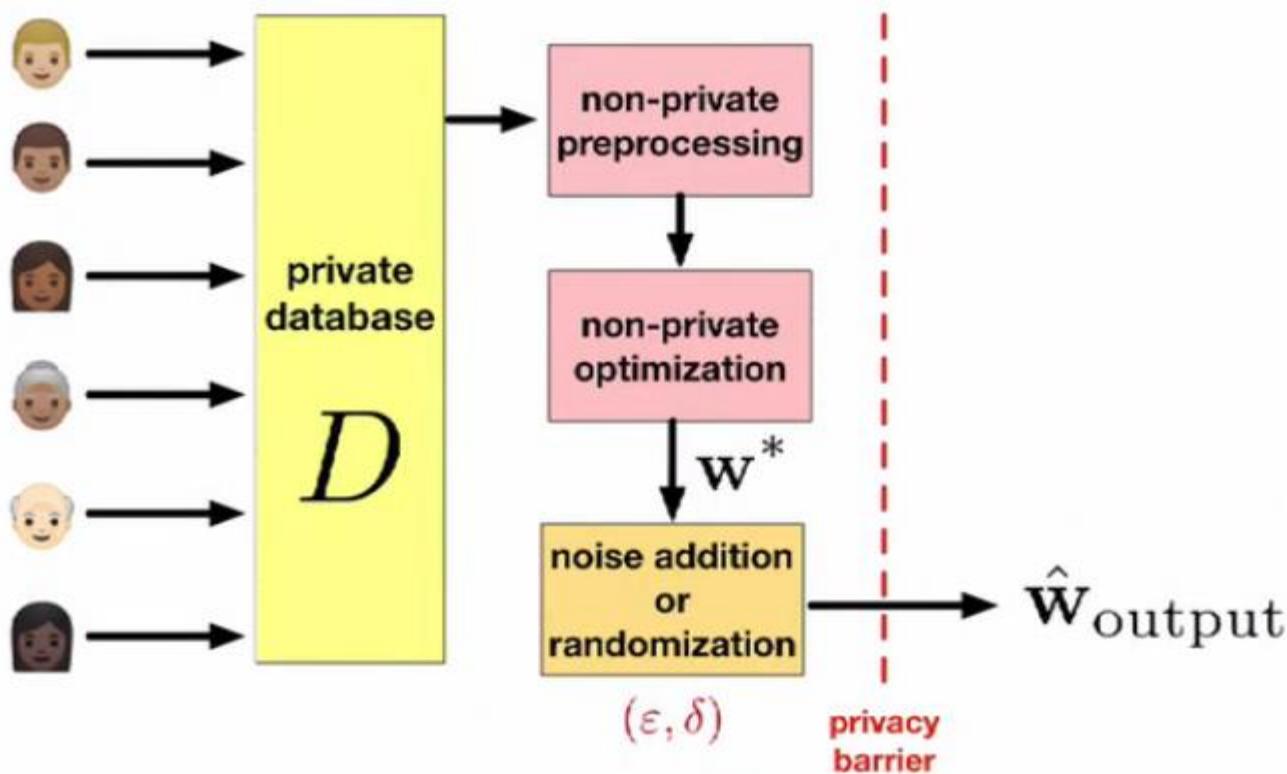
- Local privacy: data contributors sanitize data before collection.
- Classical technique: *randomized response* [W65].
- Interactive variant can be minimax optimal [DJW13].

Input Perturbation



- Input perturbation: add noise to the input data.
- Advantages: easy to implement, results in reusable sanitized data set.

Output Perturbation

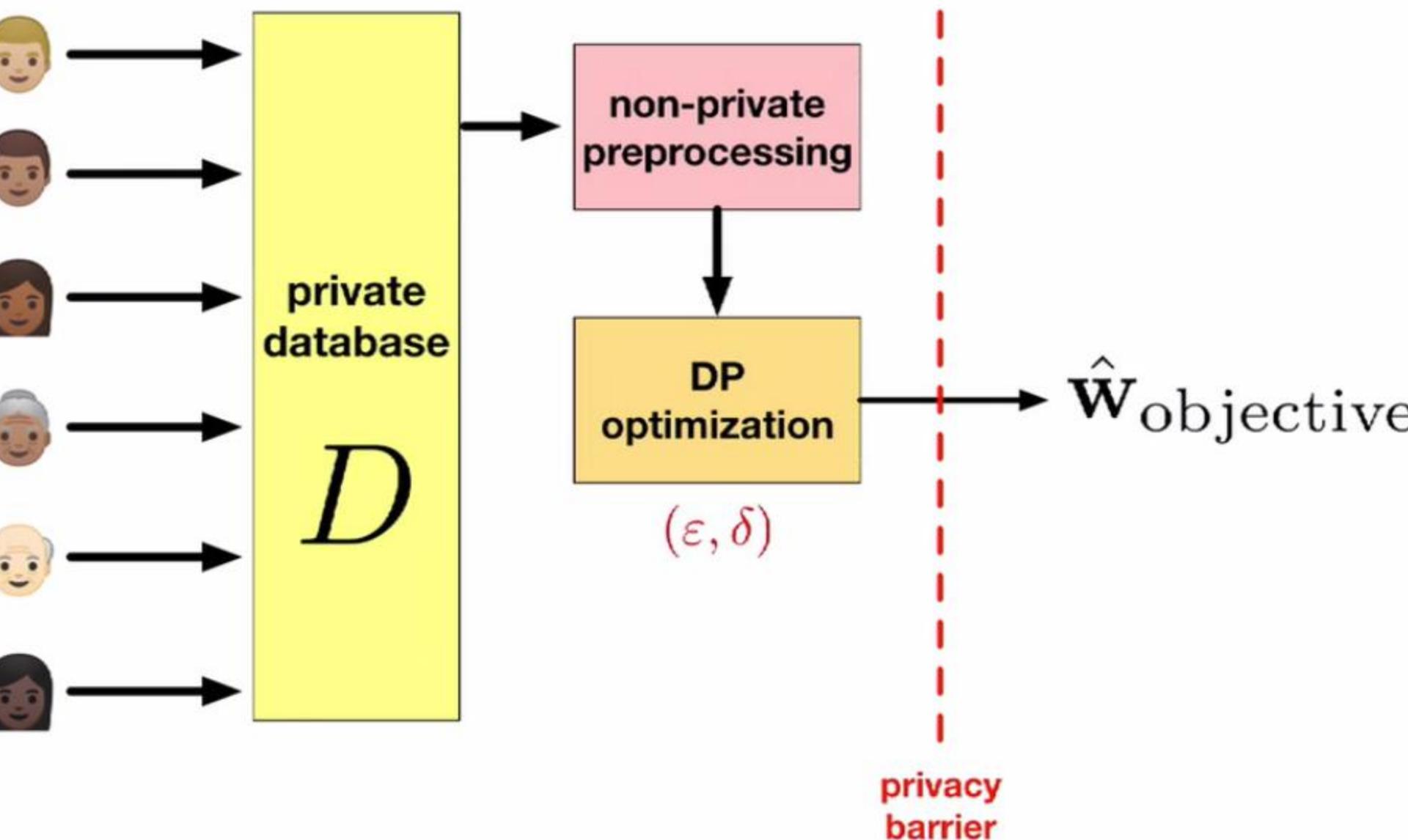


- Compute the minimizer and add noise.
- Does not require re-engineering baseline algorithms

Noise depends on the sensitivity of the argmin.

[CMS11, RBHT12]

Objective Perturbation



Objective Perturbation

$$J(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^n \ell(\mathbf{w}, (\mathbf{x}_i, y_i)) + \lambda R(\mathbf{w})$$

A. Add a random term to the objective:

$$\hat{\mathbf{w}}_{\text{priv}} = \operatorname*{argmin}_{\mathbf{w}} (J(\mathbf{w}) + \mathbf{w}^\top \mathbf{b})$$

B. Do a randomized approximation of the objective:

$$\hat{\mathbf{w}}_{\text{priv}} = \operatorname*{argmin}_{\mathbf{w}} \hat{J}(\mathbf{w})$$

Randomness depends on the sensitivity properties of $J(\mathbf{w})$.

Sensitivity of the argmin

$$\hat{\mathbf{w}}_{\text{priv}} = \operatorname*{argmin}_{\mathbf{w}} (J(\mathbf{w}) + \mathbf{w}^\top \mathbf{b})$$

- Non-private optimization solves $\nabla J(\mathbf{w}) = 0 \implies \mathbf{w}^*$
- Generate vector analogue of Laplace: $\mathbf{b} \sim p(\mathbf{z}) \propto e^{-\varepsilon/2\|\mathbf{z}\|}$
- Private optimization solves $\nabla J(\mathbf{w}) = -\mathbf{b} \implies \mathbf{w}_{\text{priv}}$
- Have to bound the sensitivity of the gradient.

Theoretical bounds on excess risk

Same important parameters:

- privacy parameters (ε, δ)
- data dimension d
- data bounds $\|\mathbf{x}_i\| \leq B$
- analytical properties of the loss
(Lipschitz, smoothness)
- regularization parameter λ

Theoretical bounds on excess risk

input perturbation

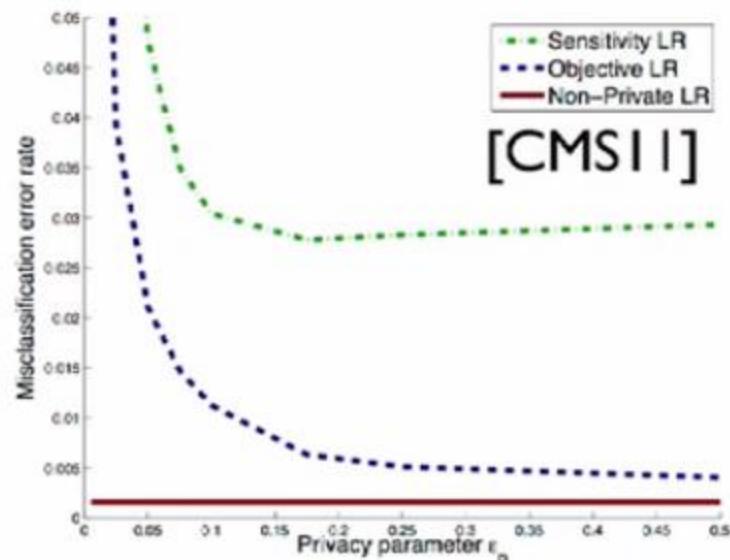
$$\tilde{O} \left(\frac{\sqrt{d} \log(1/\delta)}{n\epsilon} \right)$$

(quadratic loss)
[FTS17]

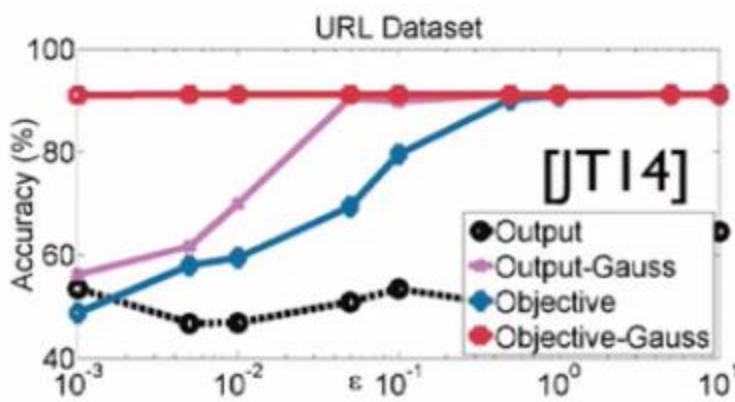
Same important parameters:

- privacy parameters (ϵ, δ)
- data dimension d
- data bounds $\|\mathbf{x}_i\| \leq B$
- analytical properties of the loss
(Lipschitz, smoothness)
- regularization parameter λ

Typical empirical results



(a) Regularized logistic regression, KDDCup99



In general:

- **Objective** perturbation empirically outperforms output perturbation.
- **Gaussian** mechanism with (ϵ, δ) guarantees outperform Laplace - like mechanisms with ϵ -guarantees.
- **Loss vs. non-private methods** is very dataset-dependent.

Gaps between theory and practice

- Theoretical analysis is for fixed privacy parameters – how should we choose them in practice?
- Given a data set, can I tell what the privacy-utility-sample-size tradeoff is?
- What about more general optimization problems/algorithms?
- What about scaling (computationally) to large data sets?

Summary

- Training does not on its own guarantee privacy.
- There are many ways to incorporate DP into prediction and learning using ERM with different **privacy-accuracy-sample size** tradeoffs.
- Good DP algorithms should **generalize** since they learn about populations, not individuals.
- Theory and experiment show that (ϵ, δ) -DP algorithms have better **accuracy** than ϵ -DP algorithms at the cost of a weaker **privacy** guarantee.

Summary

- Training does not on its own guarantee privacy.
- There are many ways to incorporate DP into prediction and learning using ERM with different **privacy-accuracy-sample size** tradeoffs.
- Good DP algorithms should **generalize** since they learn about populations, not individuals.
- Theory and experiment show that (ϵ, δ) -DP algorithms have better **accuracy** than ϵ -DP algorithms at the cost of a weaker **privacy** guarantee.

Differential privacy and optimization algorithms

Scaling up private optimization

- Large data sets are challenging for optimization:
⇒ batch methods not feasible
- Using more data can help our tradeoffs look better:
⇒ better privacy and accuracy
- Online learning involves multiple releases:
⇒ potential for more privacy loss

Scaling up private optimization

- Large data sets are challenging for optimization:
⇒ batch methods not feasible
- Using more data can help our tradeoffs look better:
⇒ better privacy and accuracy
- Online learning involves multiple releases:
⇒ potential for more privacy loss

Goal: guarantee privacy using the optimization *algorithm*.

Stochastic Gradient Descent

- Stochastic gradient descent (SGD) is a moderately popular method for optimization
- Stochastic gradients are random
⇒ already noisy ⇒ already private?
- Optimization is iterative
⇒ intermediate results leak information

Non-private SGD

$$J(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^n \ell(\mathbf{w}, (\mathbf{x}_i, y_i)) + \lambda R(\mathbf{w})$$

$$\mathbf{w}_0 = \mathbf{0}$$

- select a random data point

For $t = 1, 2, \dots, T$

$$i_t \sim \text{Unif}\{1, 2, \dots, n\}$$

- take a gradient step

$$\mathbf{g}_t = \nabla \ell(\mathbf{w}_{t-1}, (\mathbf{x}_{i_t}, y_{i_t})) + \lambda \nabla R(\mathbf{w}_{t-1})$$

$$\mathbf{w}_t = \Pi_{\mathcal{W}}(\mathbf{w}_{t-1} - \eta_t \mathbf{g}_t)$$

$$\hat{\mathbf{w}} = \mathbf{w}_T$$

Private SGD with noise

$$J(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^n \ell(\mathbf{w}, (\mathbf{x}_i, y_i)) + \lambda R(\mathbf{w})$$

$$\mathbf{w}_0 = \mathbf{0}$$

For $t = 1, 2, \dots, T$

$$i_t \sim \text{Unif}\{1, 2, \dots, n\}$$

$$\mathbf{z}_t \sim p_{(\varepsilon, \delta)}(\mathbf{z})$$

$$\hat{\mathbf{g}}_t = \mathbf{z}_t + \nabla \ell(\mathbf{w}_{t-1}, (\mathbf{x}_{i_t}, y_{i_t})) + \lambda \nabla R(\mathbf{w}_{t-1})$$

$$\mathbf{w}_t = \Pi_{\mathcal{W}}(\mathbf{w}_{t-1} - \eta_t \hat{\mathbf{g}}_t)$$

$$\hat{\mathbf{w}} = \mathbf{w}_T$$

- select random data point
- add noise to gradient

[SSCI5]

Choosing a noise distribution

“Laplace” mechanism

$$p(\mathbf{z}) \propto e^{(\varepsilon/2)\|\mathbf{z}\|}$$

$$\varepsilon - \text{DP}$$

Gaussian mechanism

$$p(\mathbf{z}) \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}\left(0, \tilde{O}\left(\frac{\log(1/\delta)}{\varepsilon^2}\right)\right)$$

$$(\varepsilon, \delta) - \text{DP}$$

- Have to choose noise according to the sensitivity of the gradient:

$$\max_{D, D'} \max_{\mathbf{w}} \|\nabla J(\mathbf{w}; D) - \nabla J(\mathbf{w}; D')\|$$

- Sensitivity depends on the data distribution, Lipschitz parameter of the loss, etc.

Private SGD with randomized selection

$$J(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^n \ell(\mathbf{w}, (\mathbf{x}_i, y_i)) + \lambda R(\mathbf{w})$$

$$\mathbf{w}_0 = \mathbf{0}$$

For $t = 1, 2, \dots, T$

$$i_t \sim \text{Unif}\{1, 2, \dots, n\}$$

$$\mathbf{g}_t = \nabla \ell(\mathbf{w}_{t-1}, (\mathbf{x}_{i_t}, y_{i_t})) + \lambda \nabla R(\mathbf{w}_{t-1})$$

$$\hat{\mathbf{g}}_t \sim p_{(\varepsilon, \delta), \mathbf{g}}(\mathbf{z})$$

$$\mathbf{w}_t = \Pi_{\mathcal{W}}(\mathbf{w}_{t-1} - \eta_t \hat{\mathbf{g}}_t)$$

$$\hat{\mathbf{w}} = \mathbf{w}_T$$

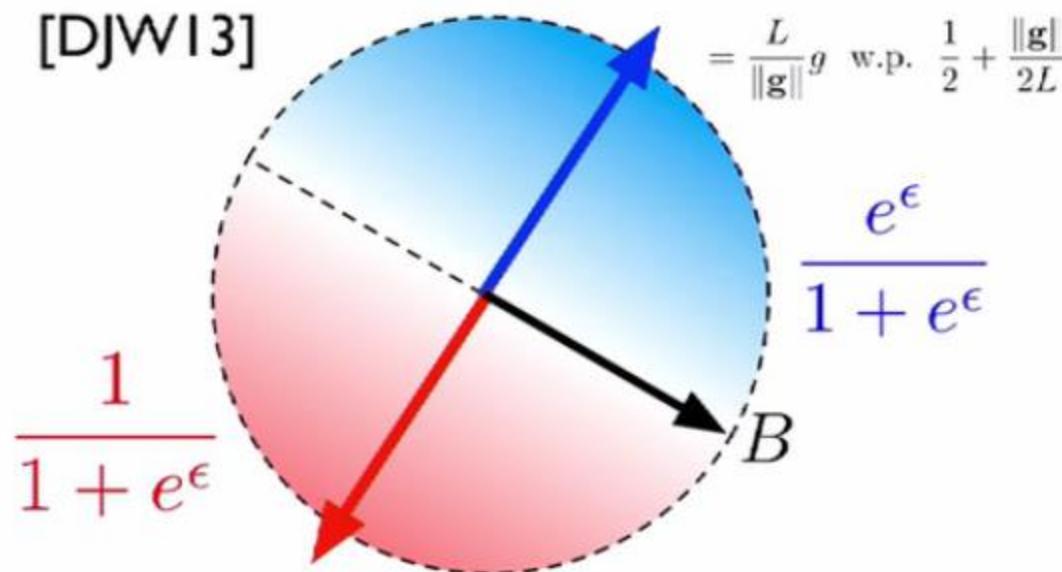
- select random data point

- randomly select unbiased gradient estimate

[DJW13]

Randomized directions

[DJW13]

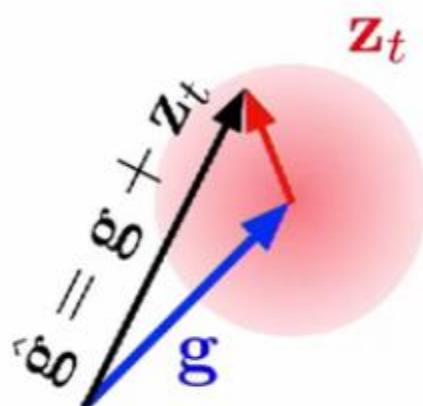


Select hemisphere in direction of gradient or opposite.

Pick uniformly from the hemisphere and take a step

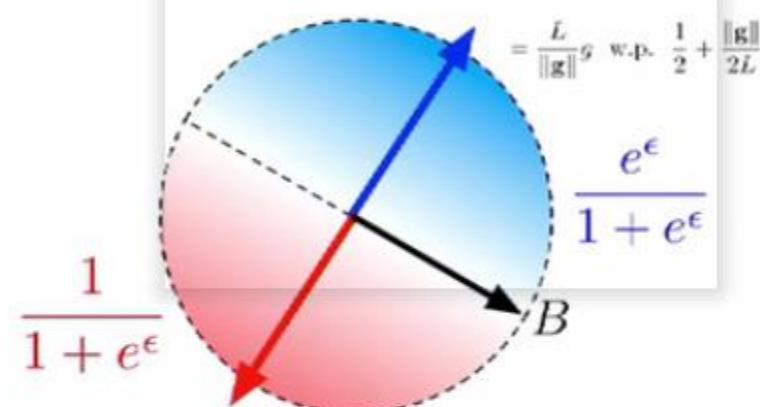
- Need to have control of gradient norms: $\|g\| \leq L$
- Keep some probability of going in the wrong direction.

Why does DP-SGD work?



Noisy Gradient

Choose noise distribution using the sensitivity of the gradient.



Random Gradient

Randomly select direction biased towards the true gradient.

Both methods

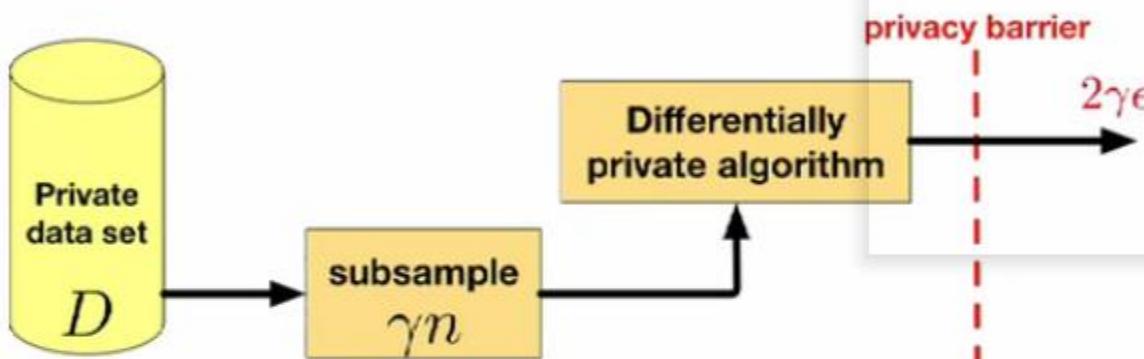
- Guarantee DP at each iteration.
- Ensure unbiased estimate of \mathbf{g} to guarantee convergence

Making DP-SGD more practical

“SGD is robust to noise”

- True up to a point — for small epsilon (more privacy), the gradients can become too noisy.
- **Solution 1:** more iterations ([BST14]: need $O(n^2)$)
- **Solution 2:** use standard tricks: mini-batching, etc. [SSCI3]
- **Solution 3:** use better analysis to show the privacy loss is not so bad [BST14][ACG+16]

Randomly sampling data can amplify privacy



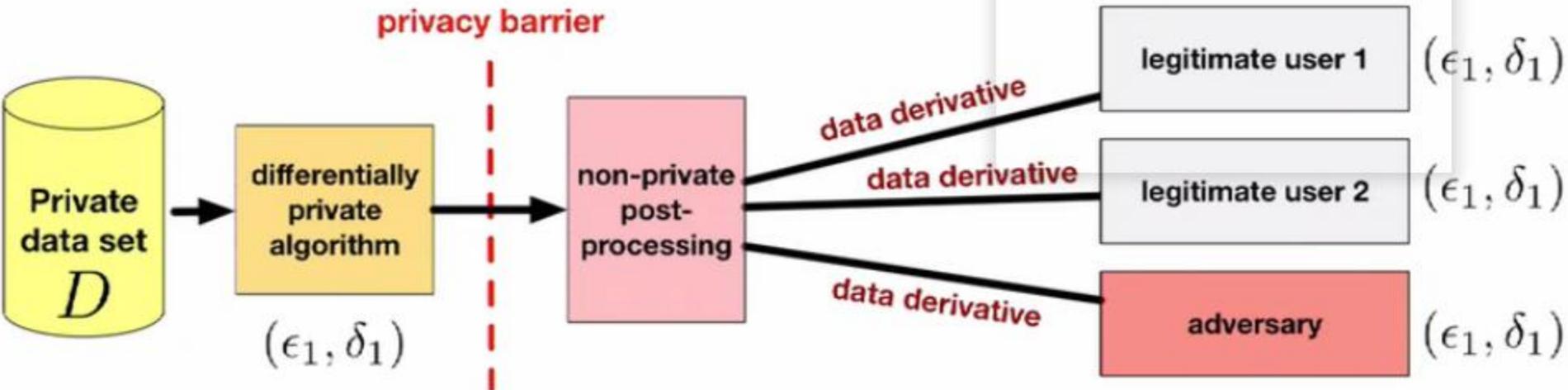
- Suppose we have an algorithm A which is ϵ -differentially private for $\epsilon \leq 1$.
- Sample γn entries of D uniformly at random and run A on those.
- Randomized method guarantees $2\gamma\epsilon$ -differential privacy.

[BBKN14,BST14]

Summary

- Stochastic gradient descent can be made differentially private in several ways by randomizing the gradient.
- Keeping gradient estimates unbiased will help ensure convergence.
- Standard approach for variance reduction/stability (such as minibatching) can help with performance.
- Random subsampling of the data can amplify privacy guarantees.

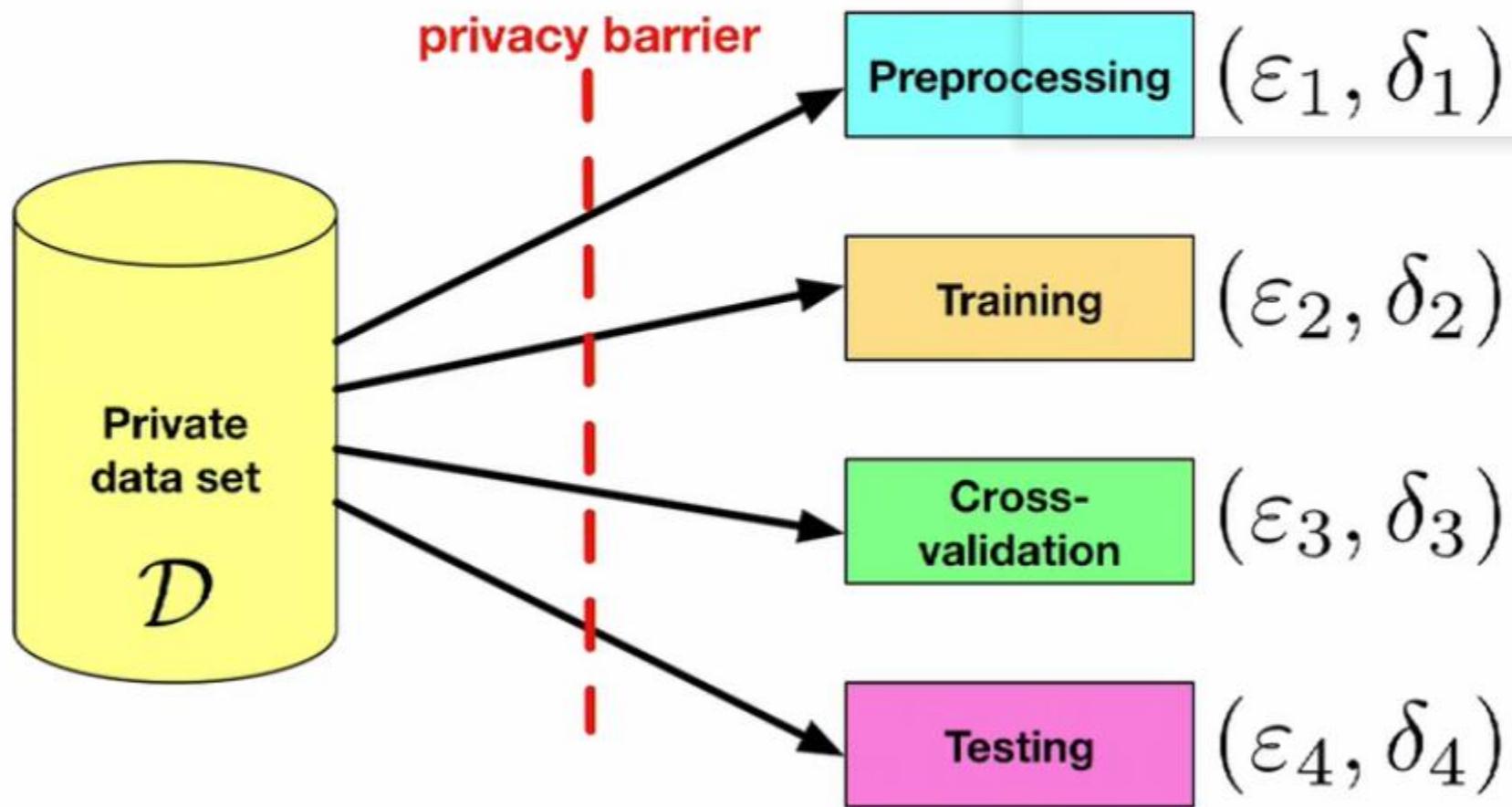
Measuring total privacy loss



Post processing invariance: risk doesn't increase if you don't touch the data again

- more complex algorithms have multiple stages
⇒ all stages have to guarantee DP
- need a way to do *privacy accounting*: what is lost over time/
multiple queries?

A simple example

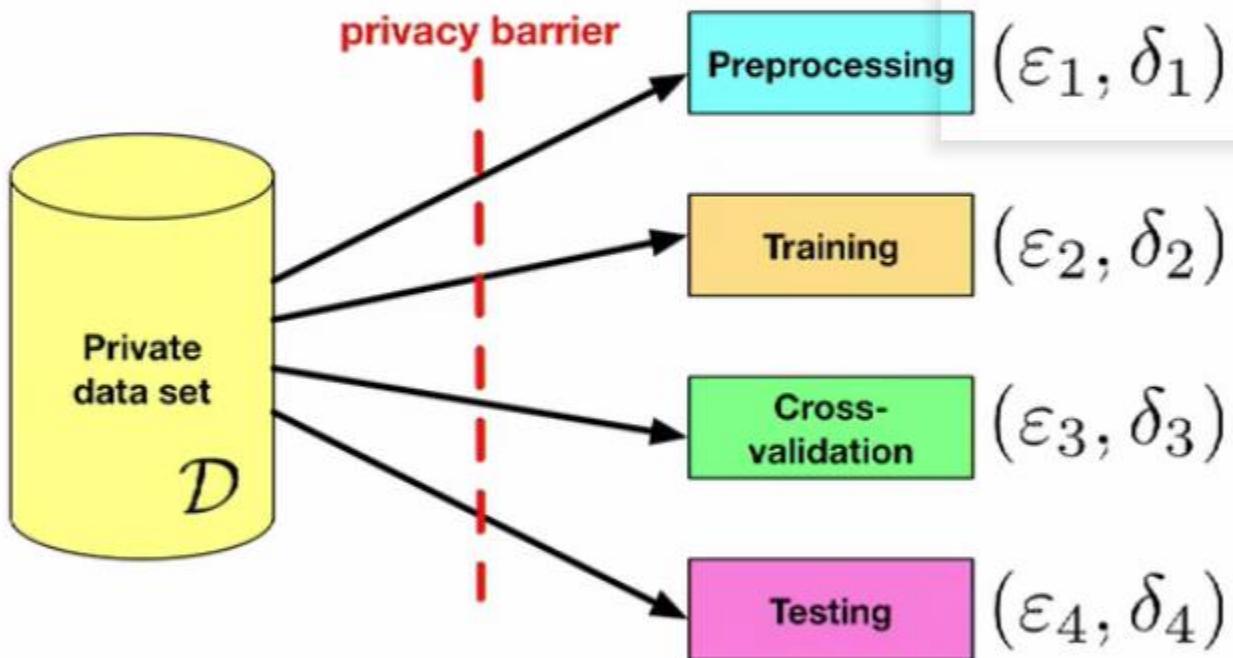


Composition property of differential privacy

Basic composition: privacy loss is additive:

- Apply R algorithms with $(\epsilon_i, \delta_i) : i = 1, 2, \dots, R$
- Total privacy loss:
$$\left(\sum_{i=1}^R \epsilon_i, \sum_{i=1}^R \delta_i \right)$$
- Worst-case analysis: each result exposes the worst privacy risk.

What composition says about multi-stage methods



Total privacy loss is the sum of the privacy losses...

An open question: privacy allocation across stages

Compositions means we have a *privacy budget*.

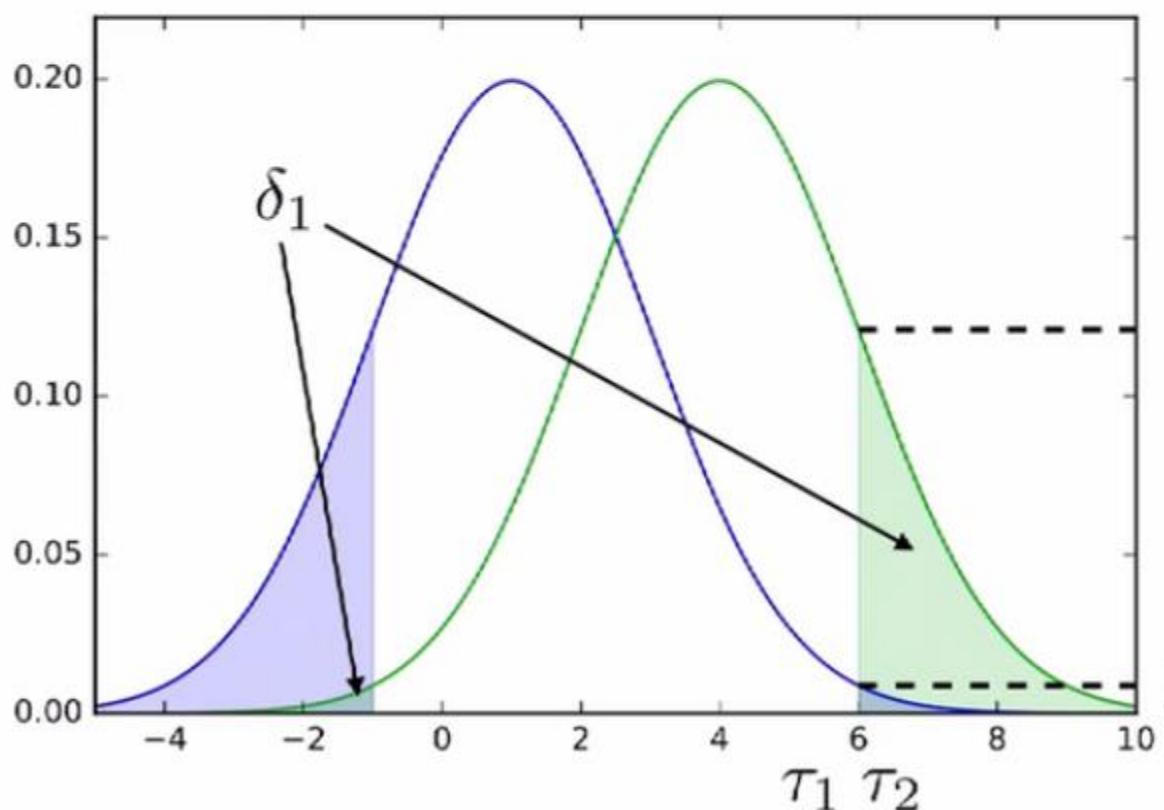
How should we allocate privacy risk across different stages of a pipeline?

- Noisy features + accurate training?
- Clean features + sloppy training?

It's application dependent! Still an open question...

A closer look at ε and δ

Gaussian noise of a given variance produces a spectrum of (ε, δ) guarantees:



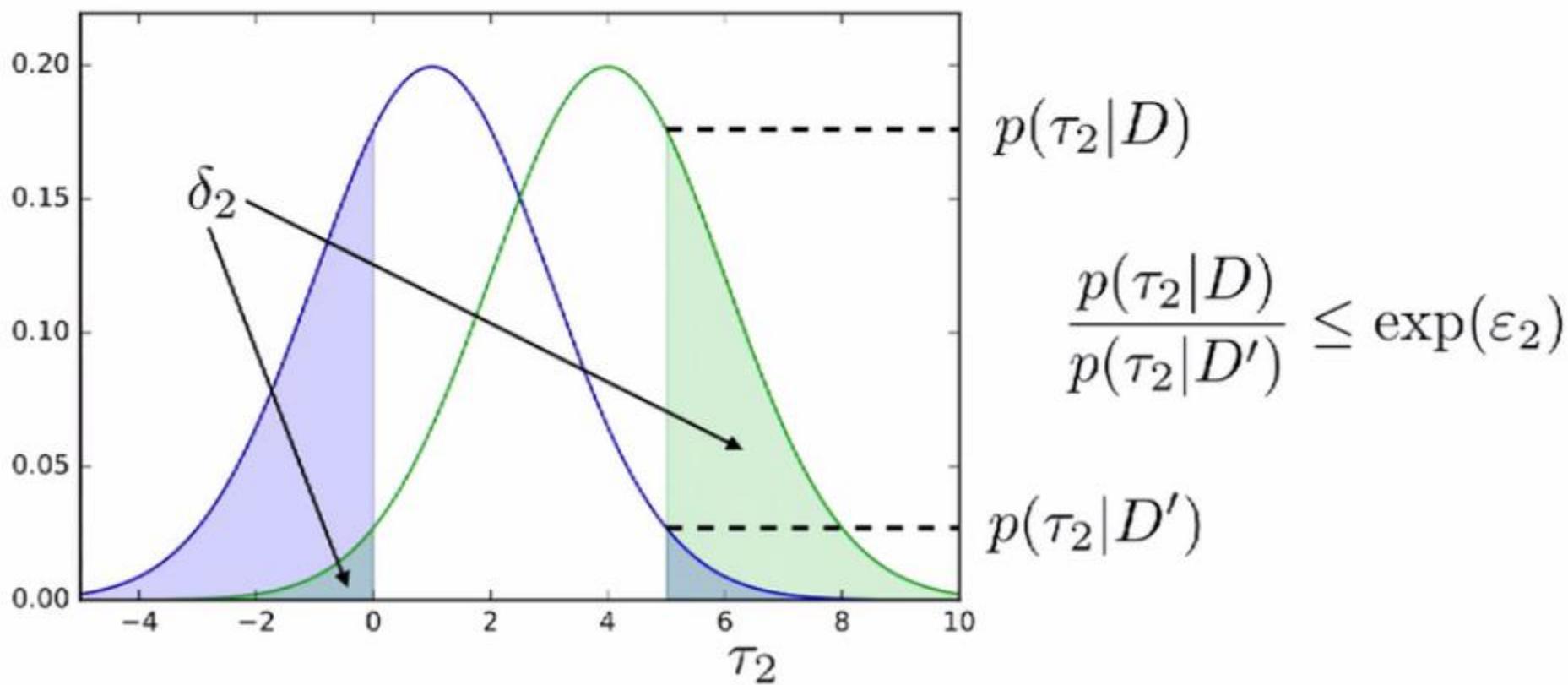
$$p(\tau_1 | D)$$

$$\frac{p(\tau_1 | D)}{p(\tau_1 | D')} \leq \exp(\varepsilon_1)$$

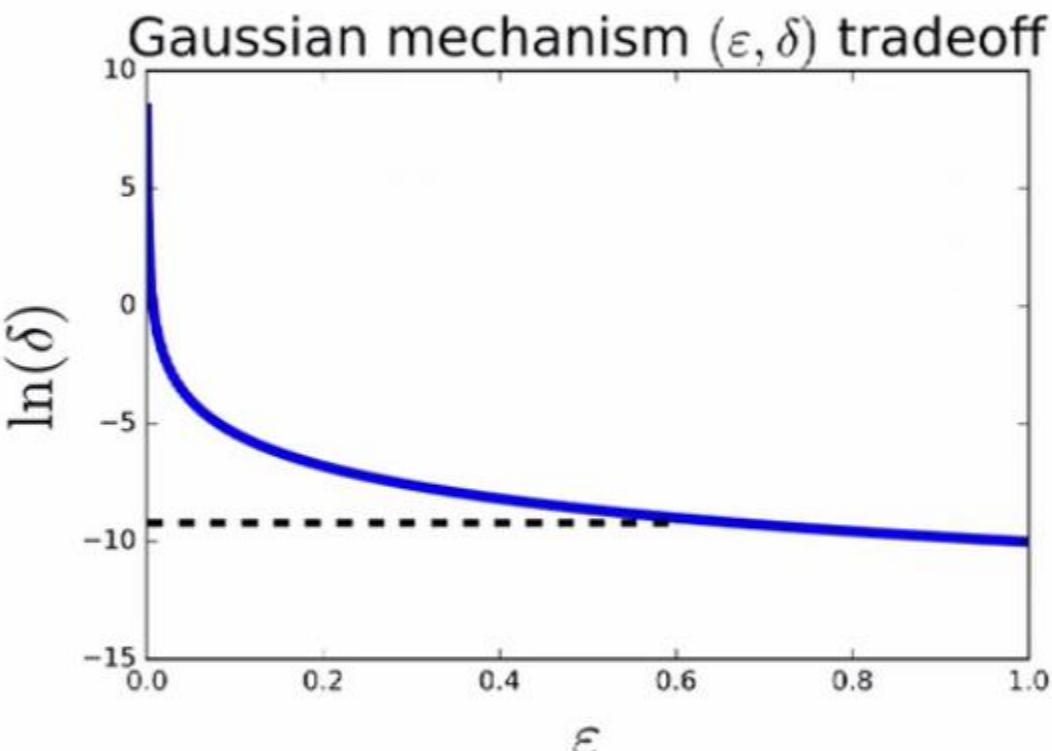
$$p(\tau_1 | D')$$

A closer look at ε and δ

Gaussian noise of a given variance produces a spectrum of (ε, δ) guarantees:



Privacy loss as a random variable



Spectrum of (ε, δ) guarantees means we can trade off ε and δ when *analyzing* a particular mechanism.

Actual privacy loss is a random variable that depends on D:

$$Z_{D,D'} = \log \frac{p(A(D) = t)}{p(A(D') = t)} \text{ w.p. } p(A(D) = t)$$

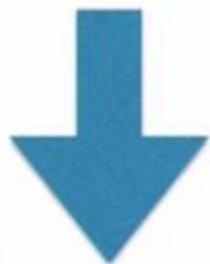
Random privacy loss

$$Z_{D,D'} = \log \frac{p(A(D) = t)}{p(A(D') = t)} \text{ w.p. } p(A(D) = t)$$

- Bounding the max loss over (D, D') is still a random variable.
- Sequentially *computing* functions on private data is like sequentially *sampling* independent privacy losses.
- Concentration of measure shows that the loss is much closer to its expectation.

Strong composition bounds

$\overbrace{(\varepsilon, \delta), (\varepsilon, \delta), \dots, (\varepsilon, \delta)}$
k times

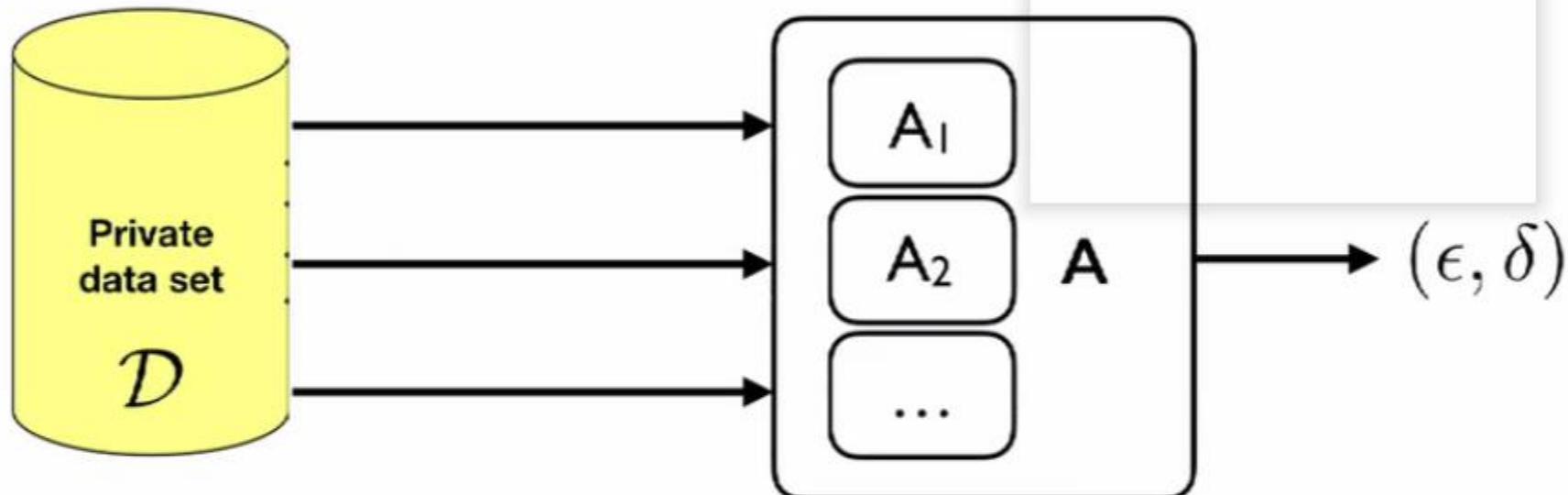


$$((k - 2i)\varepsilon, 1 - (1 - \delta)^k(1 - \delta_i))$$

$$\delta_i = \frac{\sum_{\ell=0}^{i-1} \binom{k}{\ell} (e^{(k-\ell)\varepsilon} - e^{(k-2i+\ell)\varepsilon})}{(1 + e^\varepsilon)^k}$$

- Given only the (ε, δ) guarantees for k algorithms operating on the data.
- Composition again gives a family of (ε, δ) tradeoffs: can quantify privacy loss by choosing any valid (ε, δ) pair.

Moments accountant



Basic Idea: Directly calculate parameters (ϵ, δ)

from composing a sequence of mechanisms

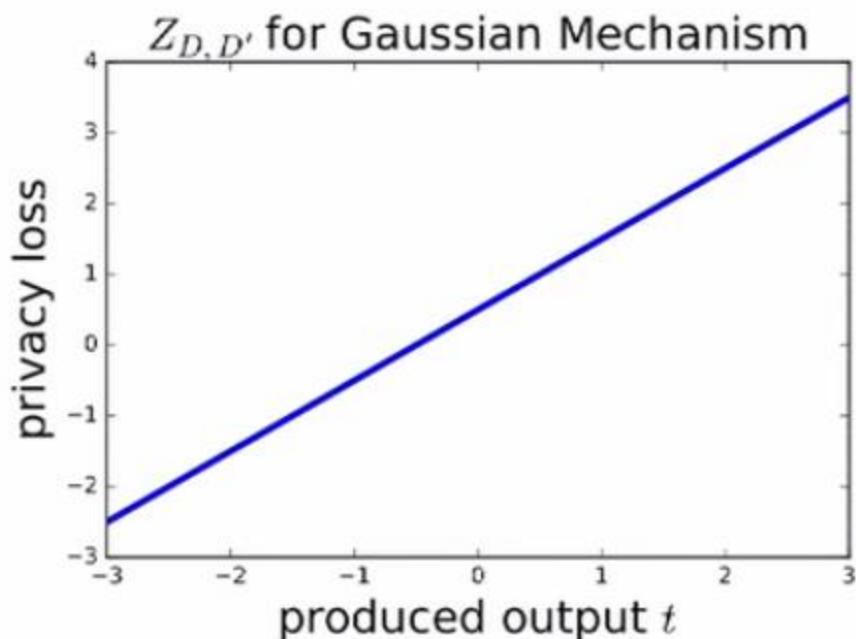
More efficient than composition theorems

[ACG+16]

How to Compose Directly?

Given datasets D and D' with one different record, mechanism A , define privacy loss random variable as:

$$Z_{D,D'} = \log \frac{p(A(D) = t)}{p(A(D') = t)}, \quad \text{w.p. } p(A(D) = t)$$

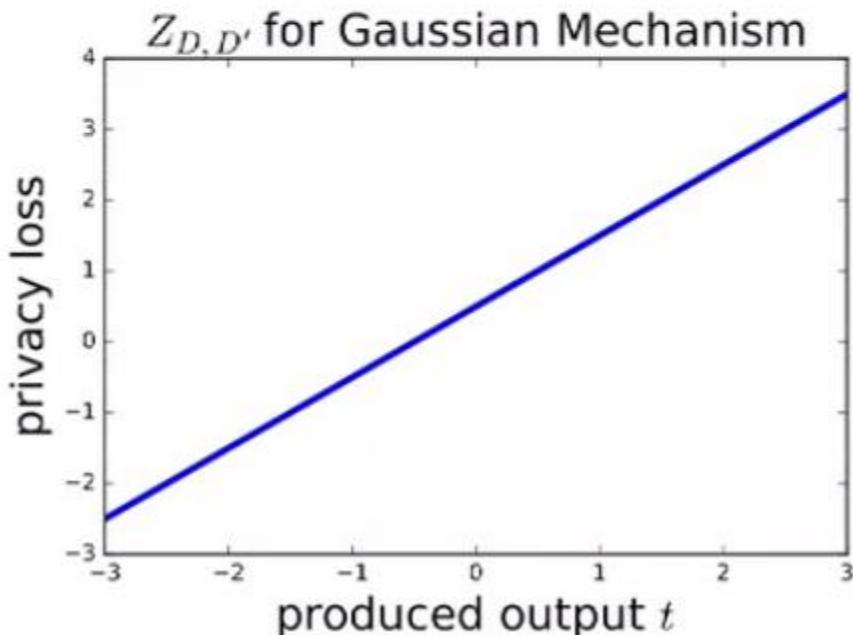


- Properties of $Z_{D,D'}$ related to privacy loss of A
- If max absolute value of $Z_{D,D'}$ over all D, D' is ϵ , then A is $(\epsilon, 0)$ -differentially private

How to Compose Directly?

Given datasets D and D' with one different record, mechanism A , define privacy loss random variable as:

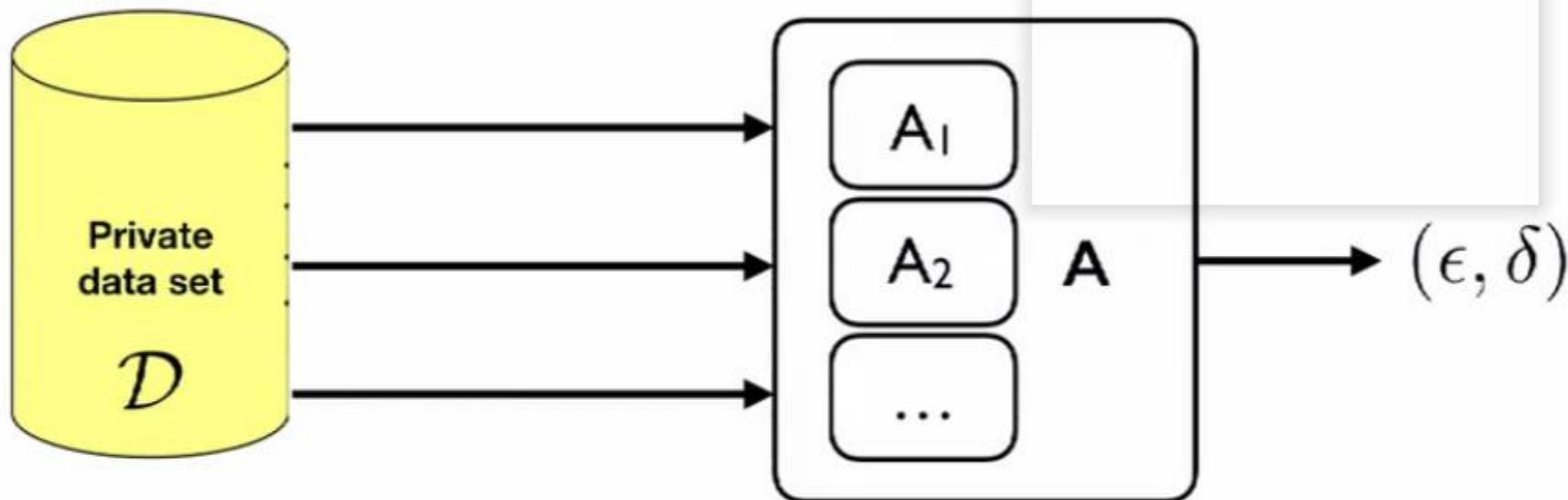
$$Z_{D,D'} = \log \frac{p(A(D) = t)}{p(A(D') = t)}, \quad \text{w.p. } p(A(D) = t)$$



Challenge: To reason about the worst case over all D, D'

Key idea in [ACG+16]: Use moment generating functions

Accounting for Moments...

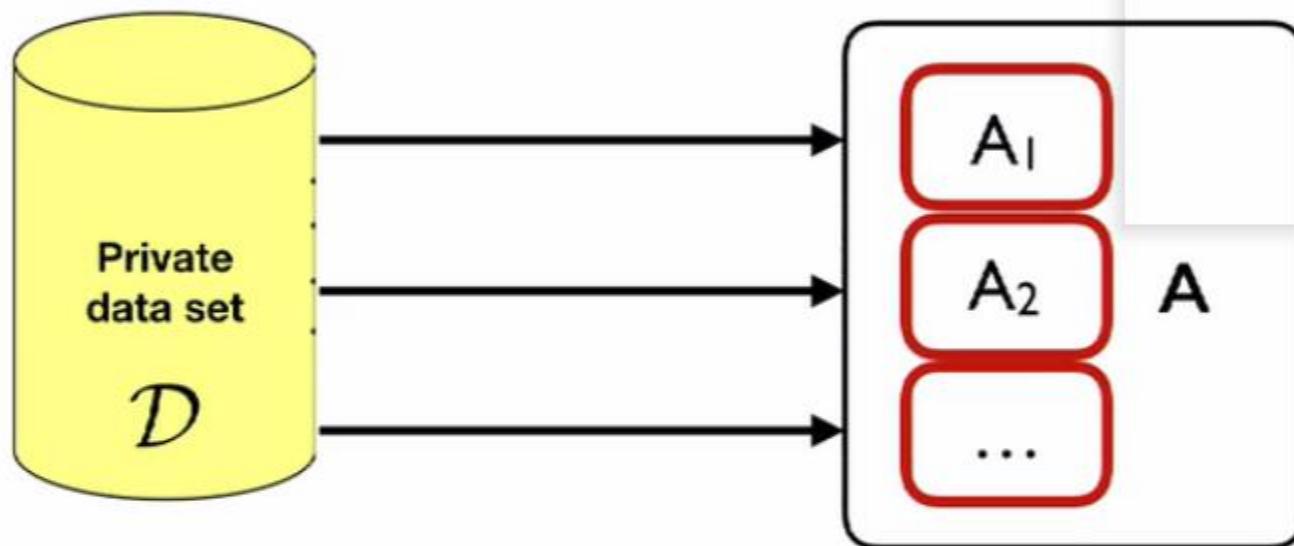


Three Steps:

1. Calculate moment generating functions for A_1, A_2, \dots
2. Compose
3. Calculate final privacy parameters

[ACG+16]

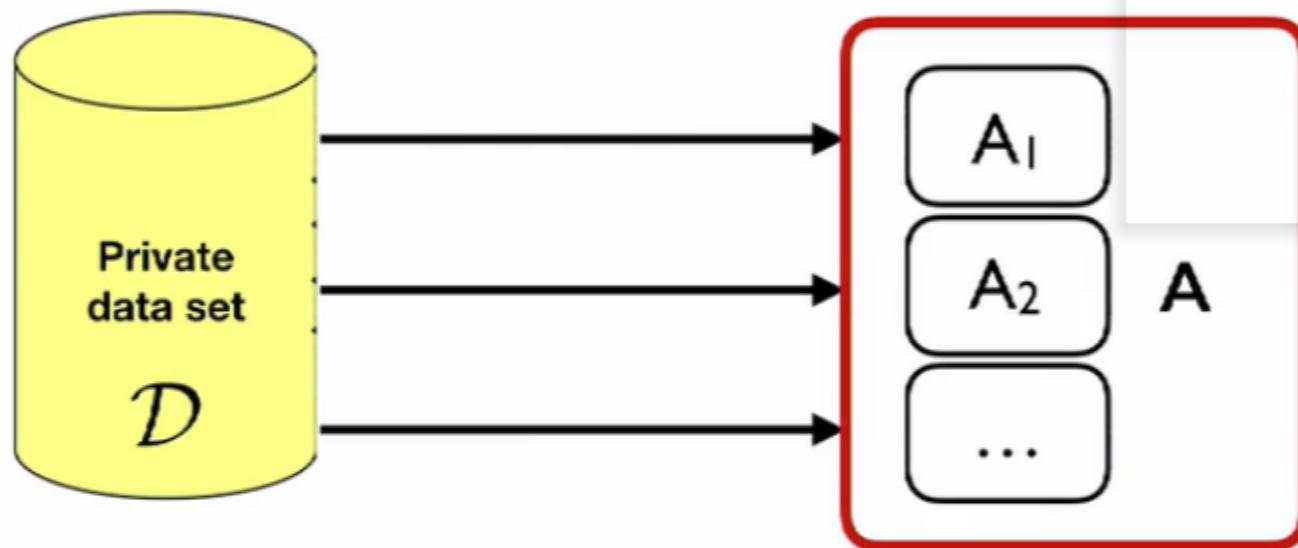
I. Stepwise Moments



Define: Stepwise Moment at time t of A_t at any s :

$$\alpha_{A_t}(s) = \sup_{D, D'} \log \mathbb{E}[e^{sZ_{D, D'}}] \quad (\text{D and D' differ by one record})$$

2. Compose

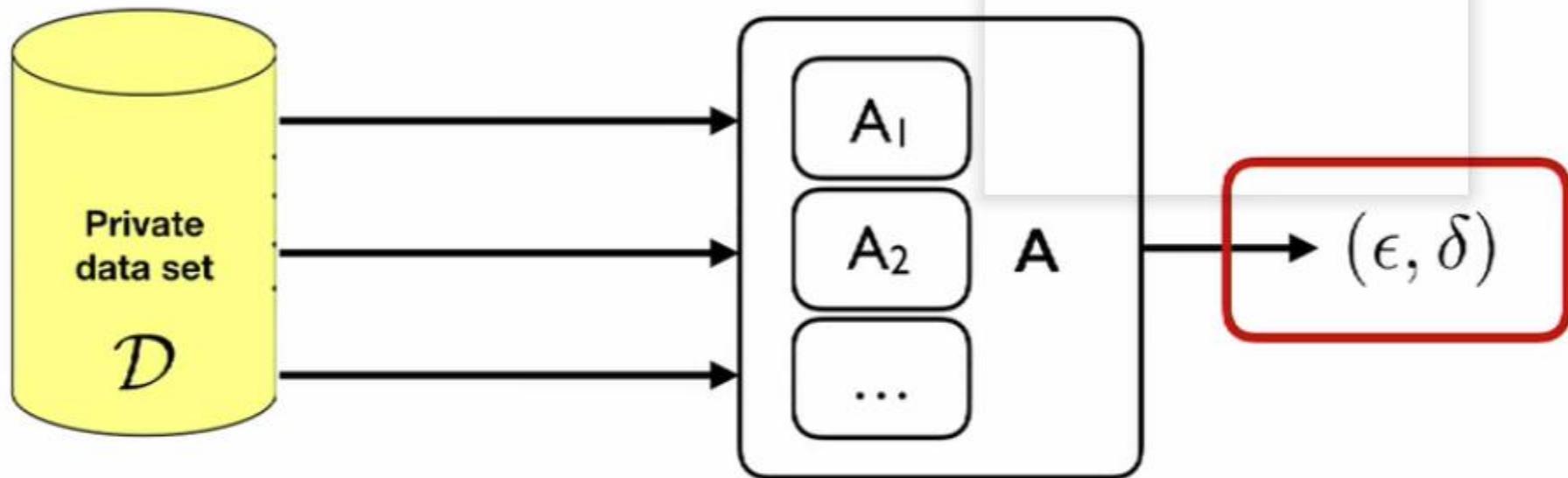


Theorem: Suppose $A = (A_1, \dots, A_T)$. For any s :

$$\alpha_A(s) \leq \sum_{t=1}^T \alpha_{A_t}(s)$$

[ACG+16]

3. Final Calculation

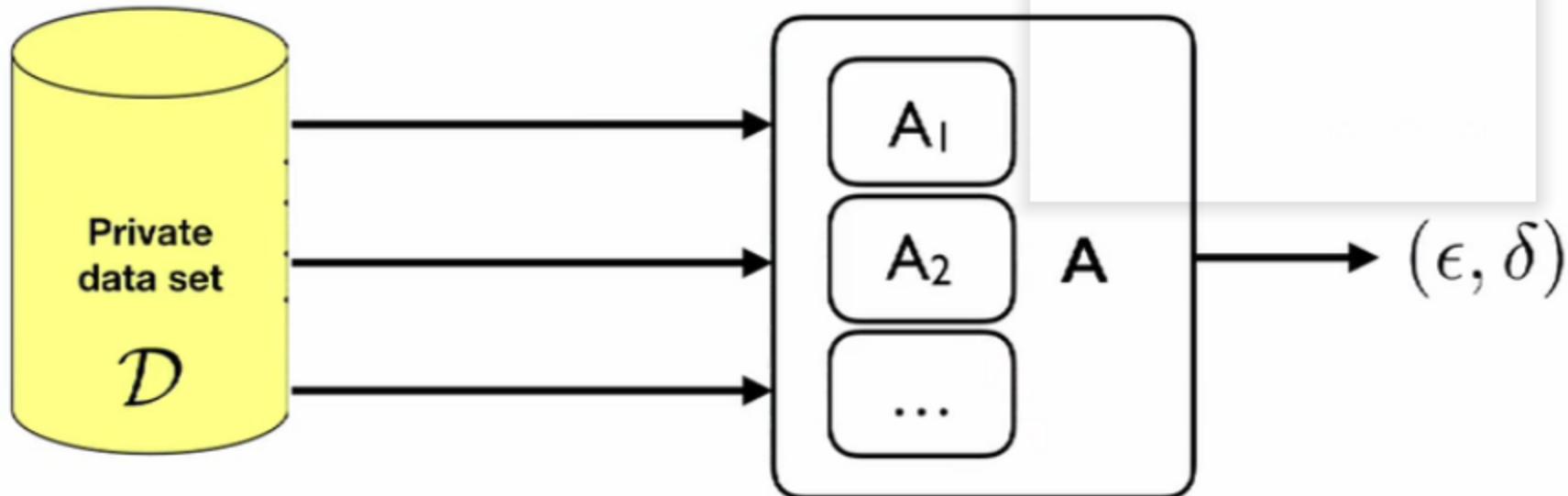


Theorem: For any ϵ , mechanism A is (ϵ, δ) -DP for

$$\delta = \min_s \exp(\alpha_A(s) - s\epsilon)$$

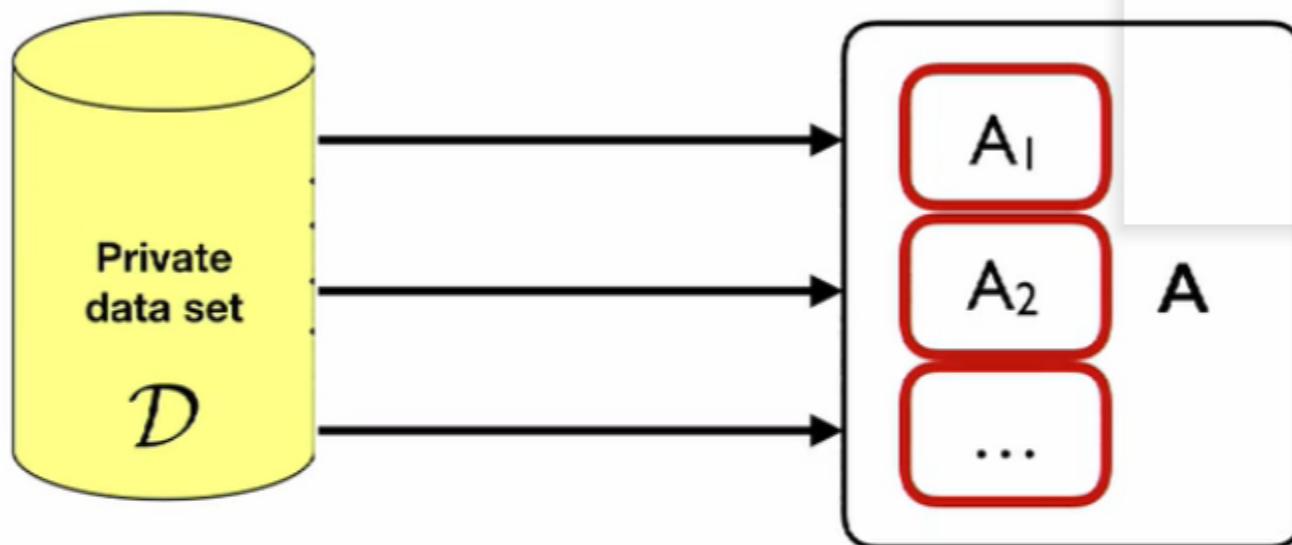
Use theorem to find best ϵ for a given δ from closed form
or by searching over s_1, s_2, \dots, s_k [ACG+16]

Example: composing Gaussian mechanism



Suppose A_t answers a query with global sensitivity l by adding $N(0, l)$ noise

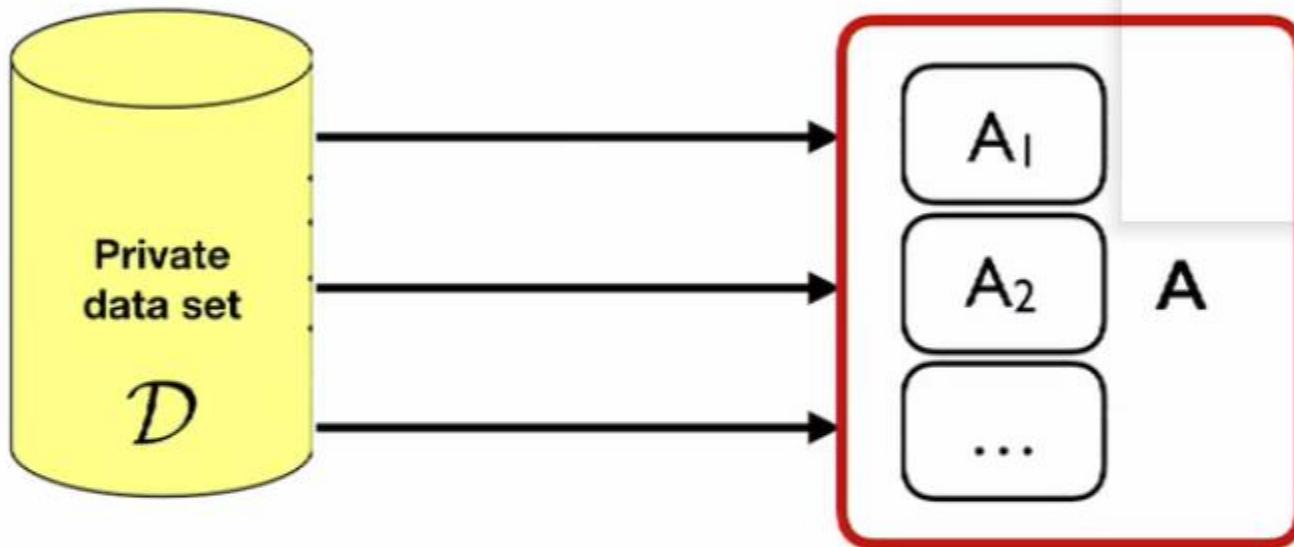
I. Stepwise Moments



Suppose A_t answers a query with global sensitivity l by adding $N(0, l)$ noise

Simple algebra gives for any s : $\alpha_{A_t}(s) = \frac{s(s + 1)}{2}$

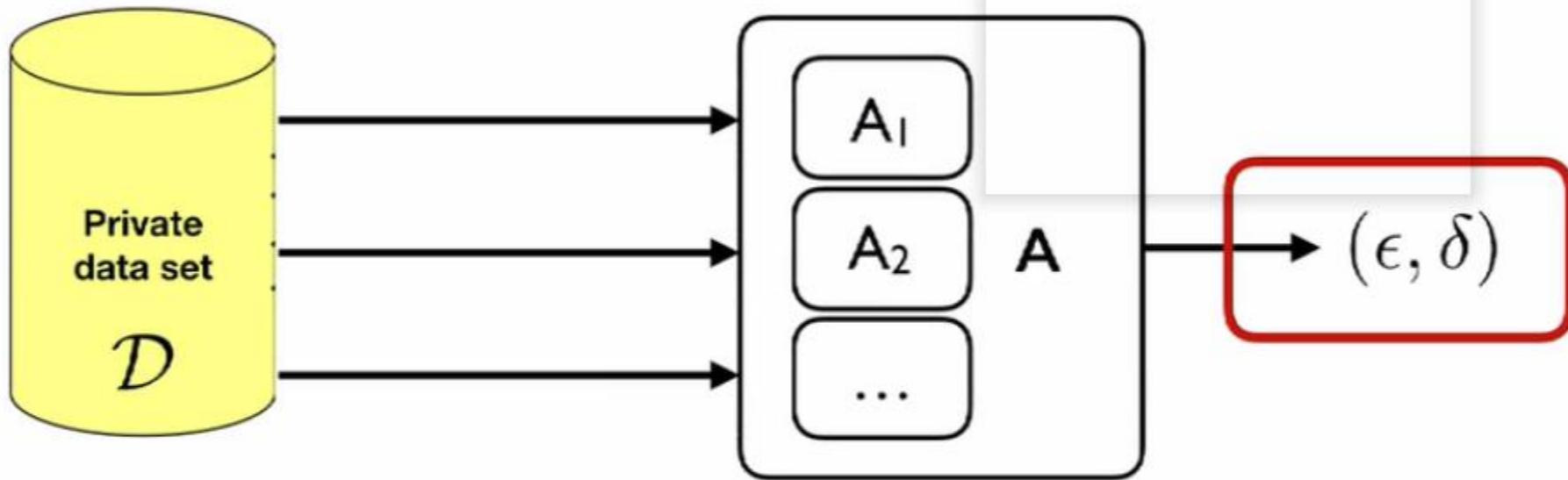
2. Compose



Suppose A_t answers a query with global sensitivity l by adding $N(0, l)$ noise

$$\alpha_A(s) \leq \sum_{t=1}^T \alpha_{A_t}(s) = \frac{Ts(s+1)}{2}$$

3. Final Calculation

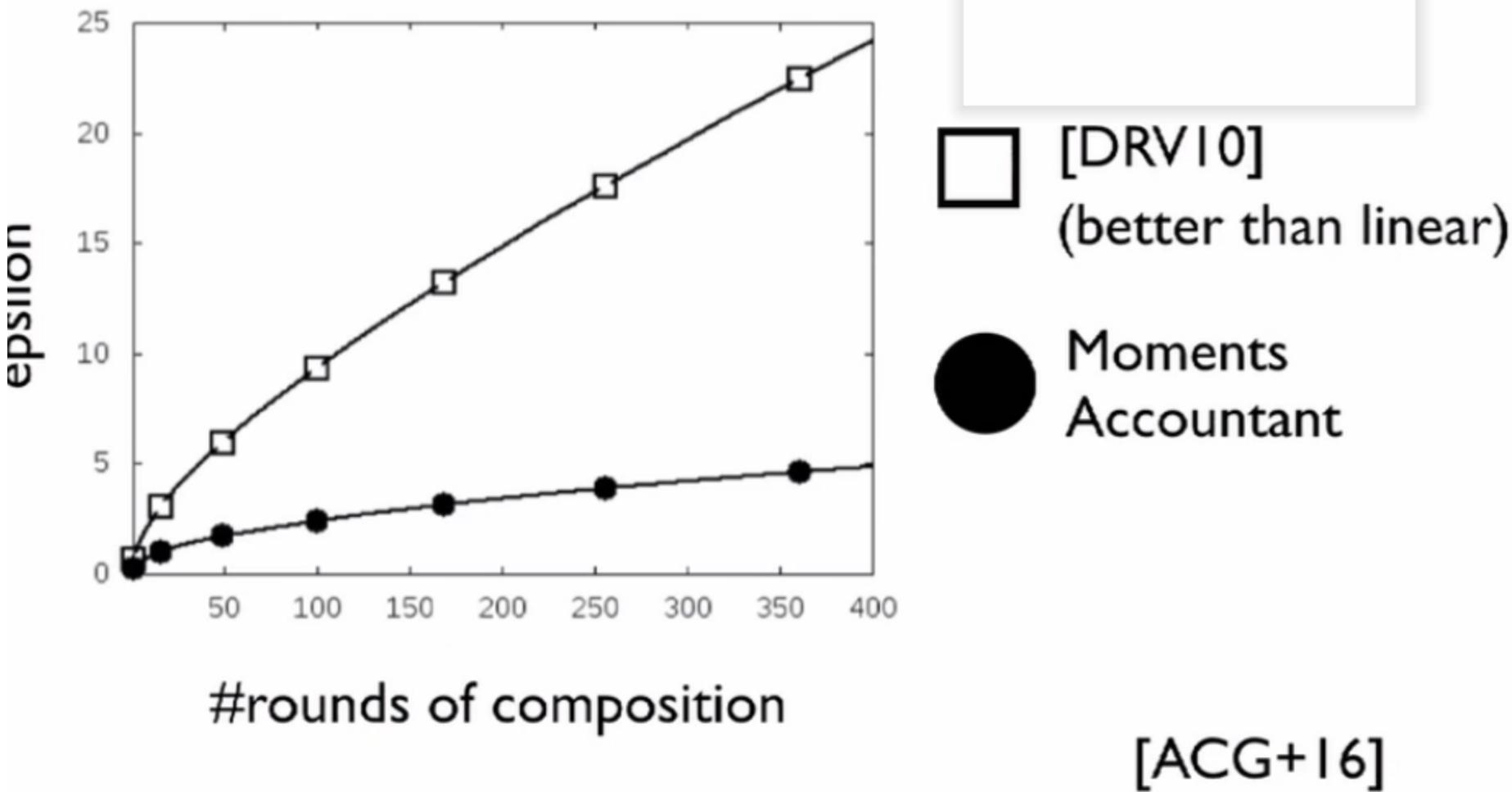


Find lowest δ for a given ϵ (or vice versa) by solving:

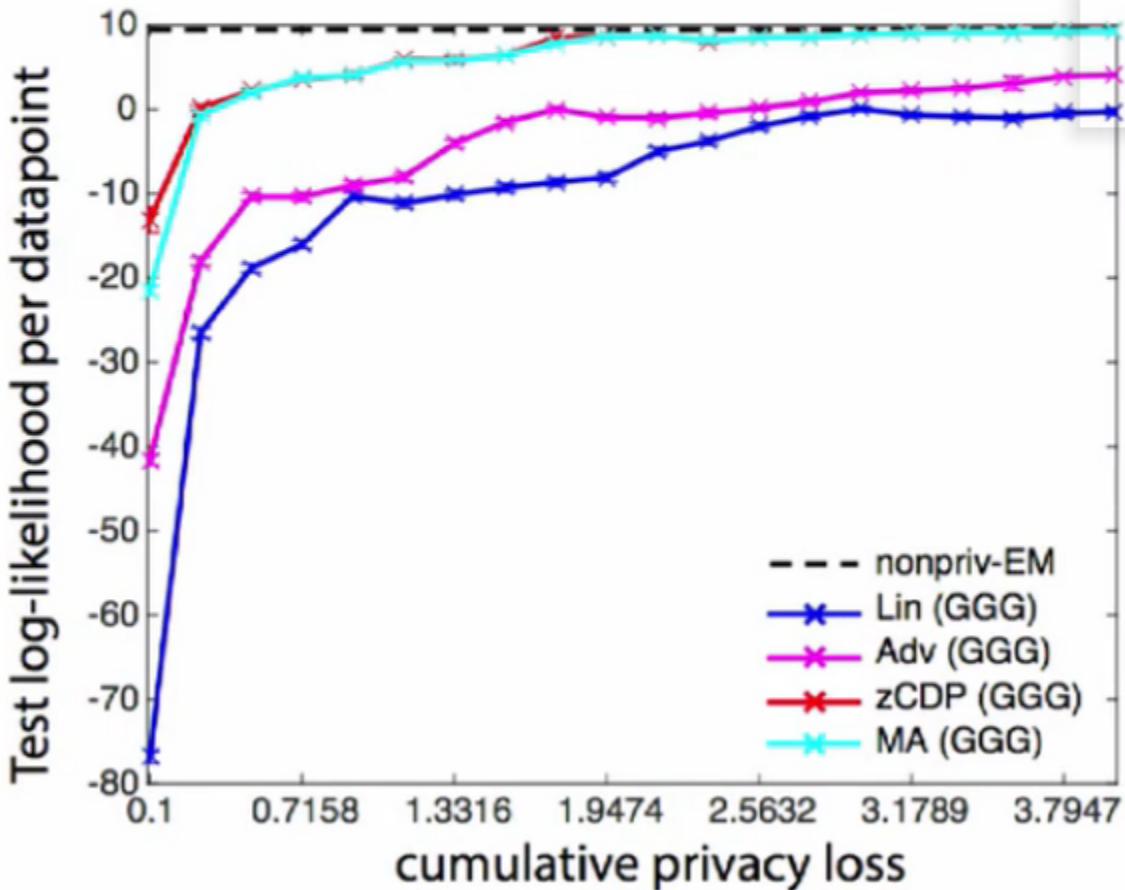
$$\delta = \min_s \exp(Ts(s+1)/2 - s\epsilon)$$

In this case, solution can be found in closed form.

How does it compare?



How does it compare on real data?



EM for MOG
with Gaussian
Mechanism
 $\delta = 10^{-4}$

Summary

- Practical machine learning looks at the data many times.
- Post-processing invariance means we just have to track the cumulative privacy loss.
- Good composition methods use the fact that the *actual privacy loss* may behave much better than the worst-case bound.
- The Moments Accountant method tracks the actual privacy loss more accurately: better analysis for better privacy guarantees.

When is differential privacy practical?

Differential privacy is best suited for understanding population-level statistics and structure:

- Inferences about the population should not depend strongly on individuals.
- Large sample sizes usually mean lower sensitivity and less noise.

To build and analyze systems we have to leverage *post-processing invariance* and *composition* properties.

Differential privacy in practice



Google: RAPPOR for tracking statistics in Chrome.

Apple: various iPhone usage statistics.

Census: 2020 US Census will use differential privacy.

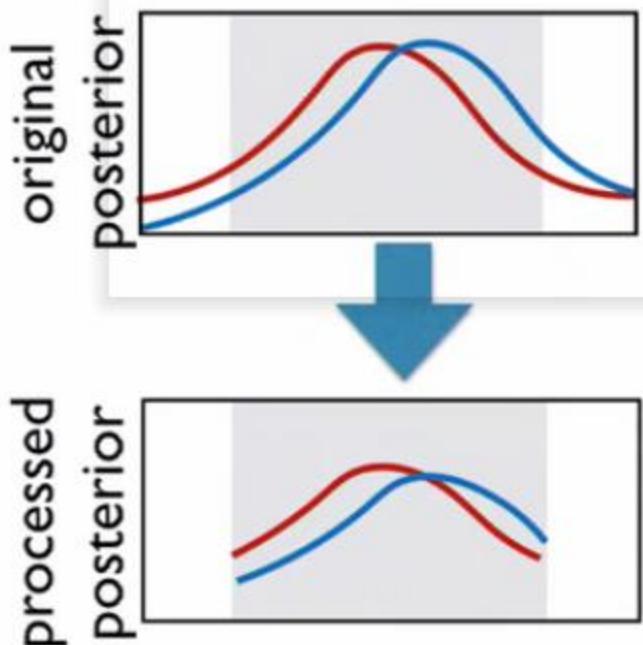
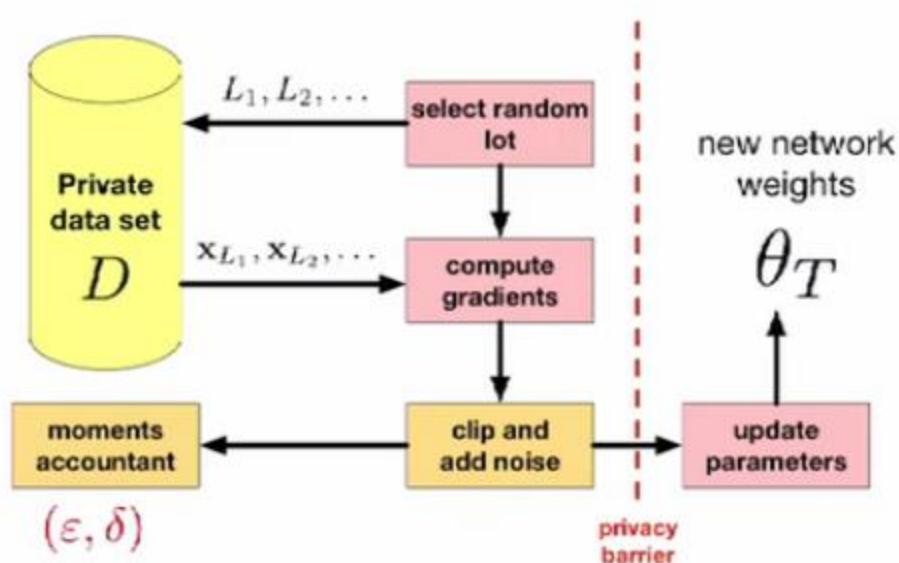
mostly focused on count and average statistics

Challenges for machine learning applications

Differentially private ML is complicated because real ML algorithms are complicated:

- Multi-stage pipelines, parameter tuning, etc.
- Need to “play around” with the data before committing to a particular pipeline/algorithm.
- “Modern” ML approaches (= deep learning) have many parameters and less theoretical guidance.

Some selected examples



For today, we will describe some recent examples:

1. Differentially private deep learning [ACG+16]
2. Differential privacy and Bayesian inference

Differential privacy and deep learning

```
class DPSGD_Optimizer():
    def __init__(self, accountant, sanitizer):
        self._accountant = accountant
        self._sanitizer = sanitizer

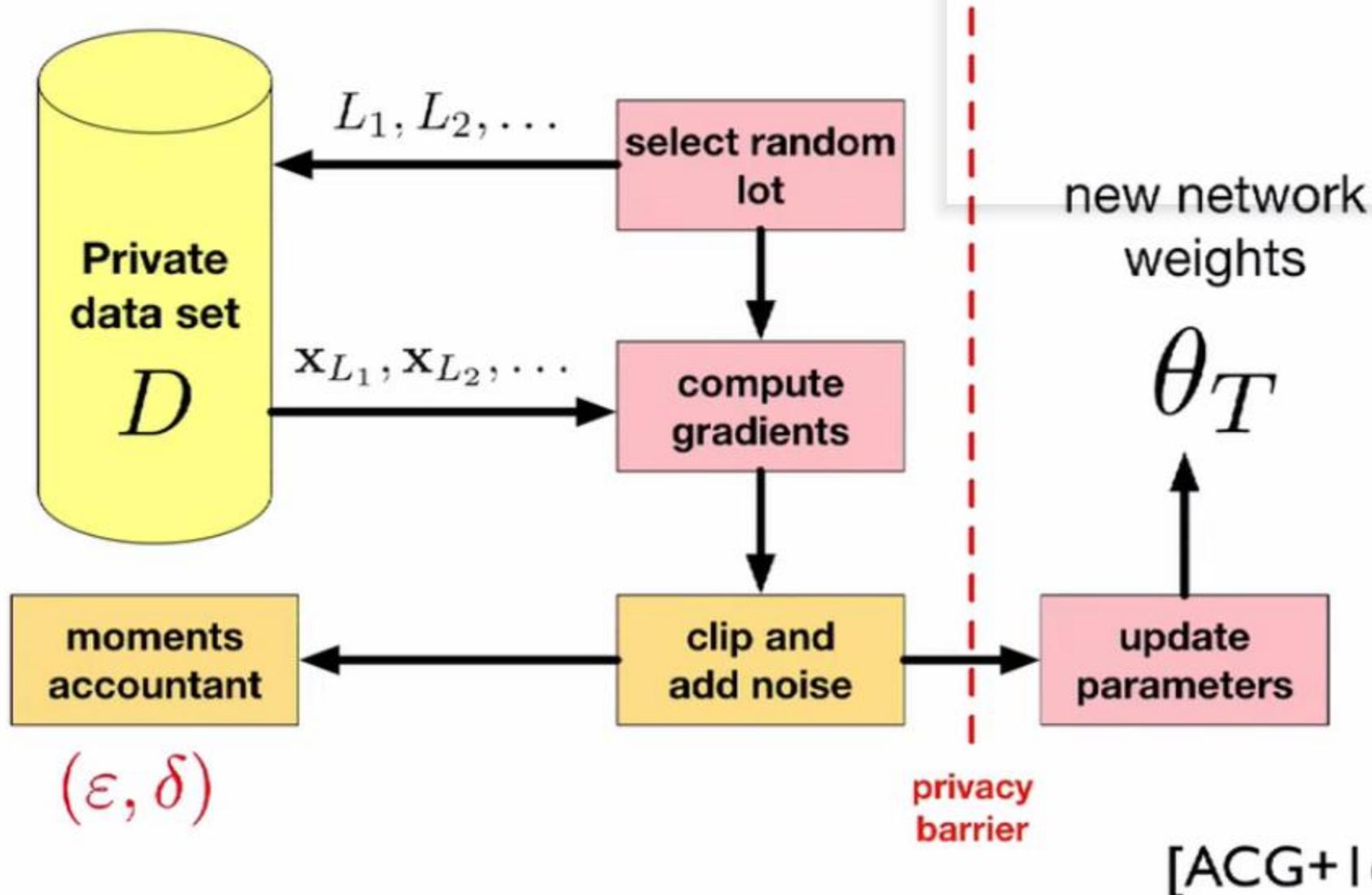
    def Minimize(self, loss, params,
                batch_size, noise_options):
        # Accumulate privacy spending before computing
        # and using the gradients.
        priv_accum_op =
            self._accountant.AccumulatePrivacySpending(
                batch_size, noise_options)
        with tf.control_dependencies(priv_accum_op):
            # Compute per example gradients
            px_grads = per_example_gradients(loss, params)
            # Sanitize gradients
            sanitized_grads = self._sanitizer.Sanitize(
                px_grads, noise_options)
            # Take a gradient descent step
            return apply_gradients(params, sanitized_grads)

def DPTrain(loss, params, batch_size, noise_options):
    accountant = PrivacyAccountant()
    sanitizer = Sanitizer()
    dp_opt = DPSGD_Optimizer(accountant, sanitizer)
    sgd_op = dp_opt.Minimize(
        loss, params, batch_size, noise_options)
    eps, delta = (0, 0)
    # Carry out the training as long as the privacy
    # is within the pre-set limit.
    while within_limit(eps, delta):
        sgd_op.run()
        eps, delta = accountant.GetSpentPrivacy()
```

Main idea: train a deep network using differentially private SGD and use moments accountant to track privacy loss.

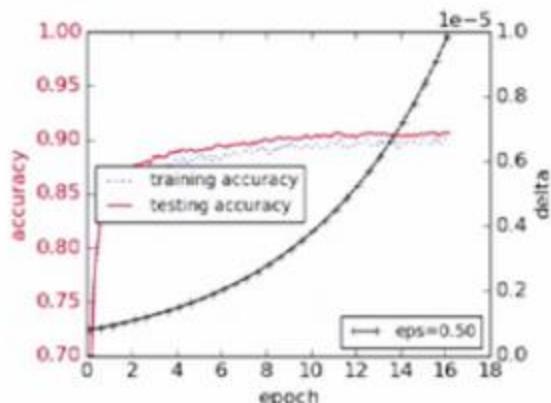
Additional components: gradient clipping, minibatching, data augmentation, etc.

Overview of the algorithm

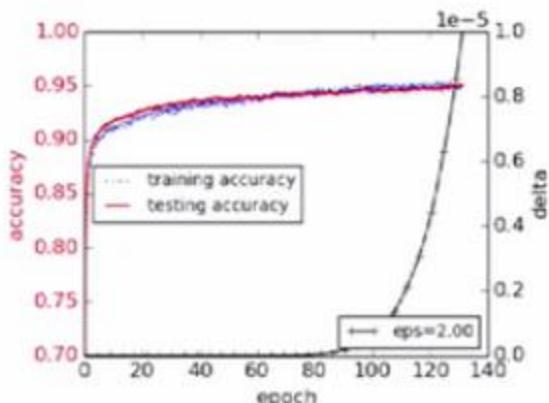


[ACG+16]

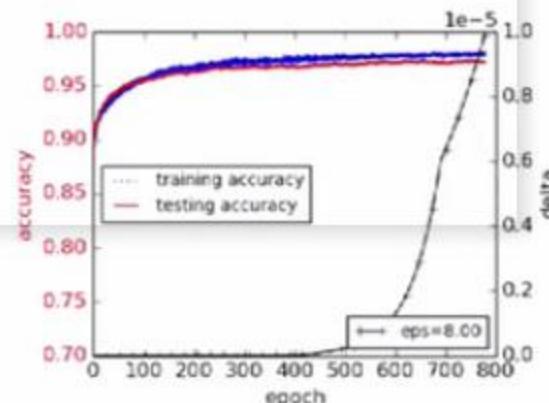
Effectiveness of DP deep learning



(1) Large noise



(2) Medium noise



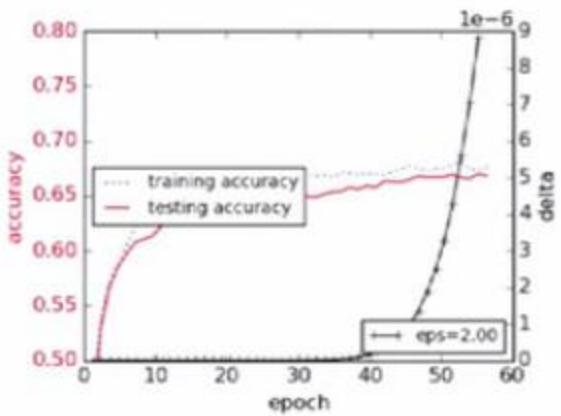
(3) Small noise

Empirical results on MNIST and CIFAR:

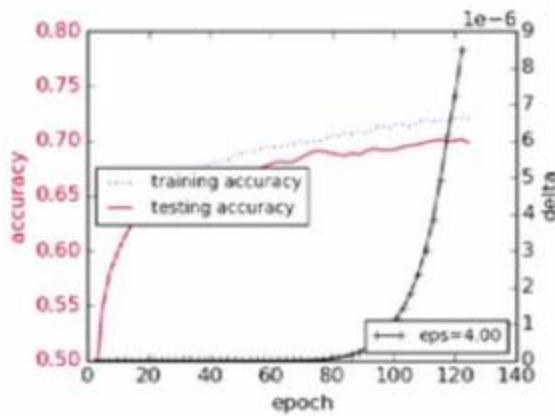
- Training and test error come close to baseline non-private deep learning methods.
- To get moderate loss in performance, epsilon and delta are not “negligible”

[ACG+16]

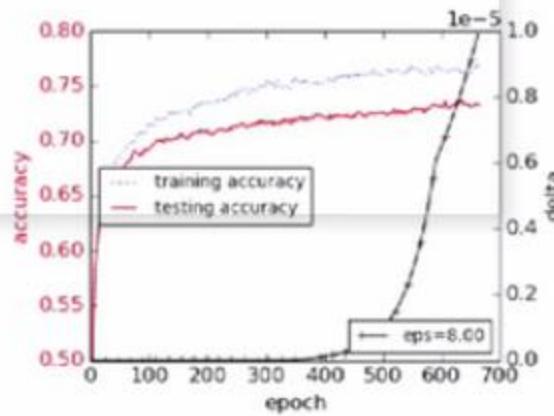
Effectiveness of DP deep learning



(1) $\epsilon = 2$



(2) $\epsilon = 4$



(3) $\epsilon = 8$

Empirical results on MNIST and CIFAR:

- Training and test error come close to baseline non-private deep learning methods.
- To get moderate loss in performance, epsilon and delta are not “negligible”

[ACG+16]

Moving forward in deep learning

This is a good *proof of concept* for differential privacy for deep neural networks. There are lots of interesting ways to expand this.

- Just used one NN model: what about other architectures? RNNs? GANs?
- Can regularization methods for deep learning (e.g. dropout) help with privacy?
- What are good rules of thumb for lot/batch size, learning rate, # of hidden units, etc?

Differentially Private Bayesian Inference

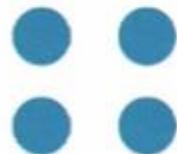
Data $X = \{x_1, x_2, \dots\}$
Model Class Θ

}

Related through
likelihood $p(x|\theta)$

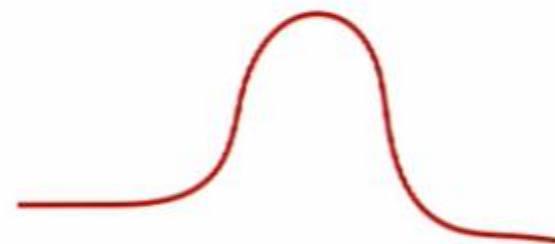


+



Prior $\pi(\theta)$

=



Data X

Posterior $p(\theta|X)$

Find differentially private approx to posterior

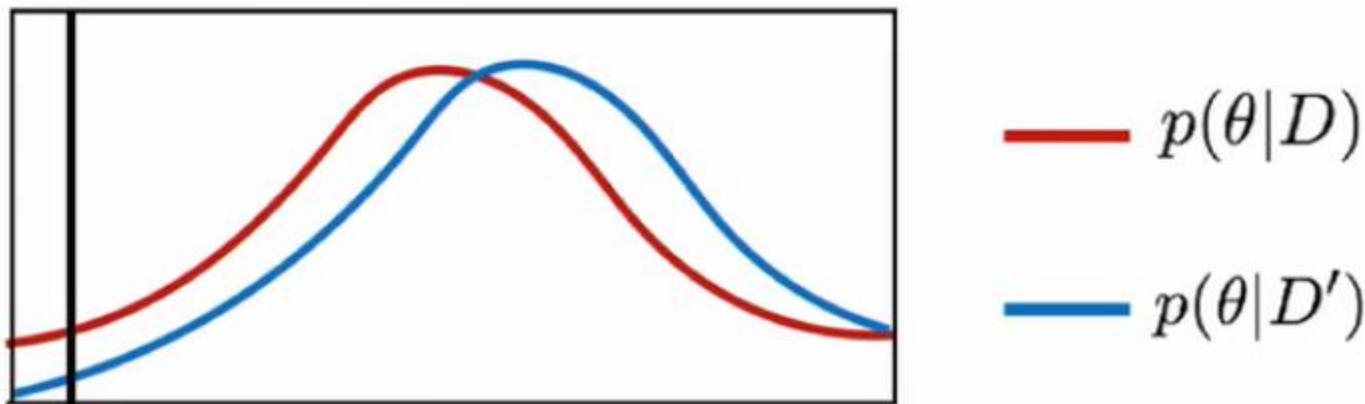
Differentially Private Bayesian Inference

- General methods for private posterior approximation
- A Special Case: Exponential Families
- Variational Inference

How to make posterior private?

Option I: Direct posterior sampling [DMNR14]

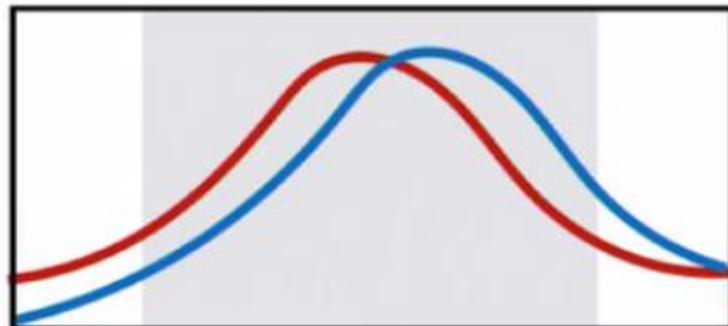
Not differentially private except under restrictive conditions - likelihood ratios may be unbounded!



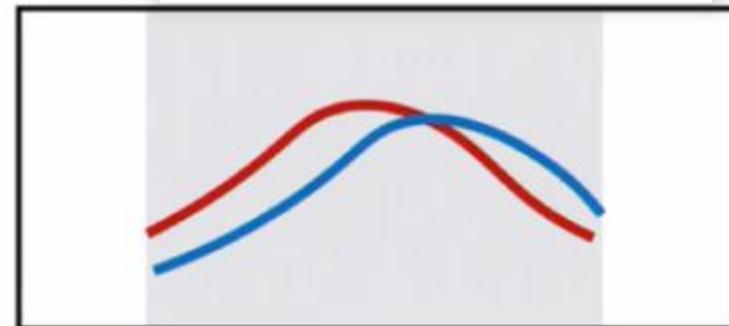
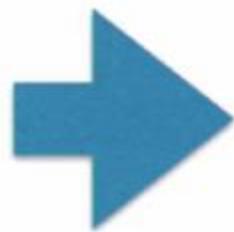
[GSC17] Answer changes under a new relaxation
Rényi differential privacy [M17]

How to make posterior private?

Option 2: One Posterior Sample (OPS) Method [WFS15]



original posteriors

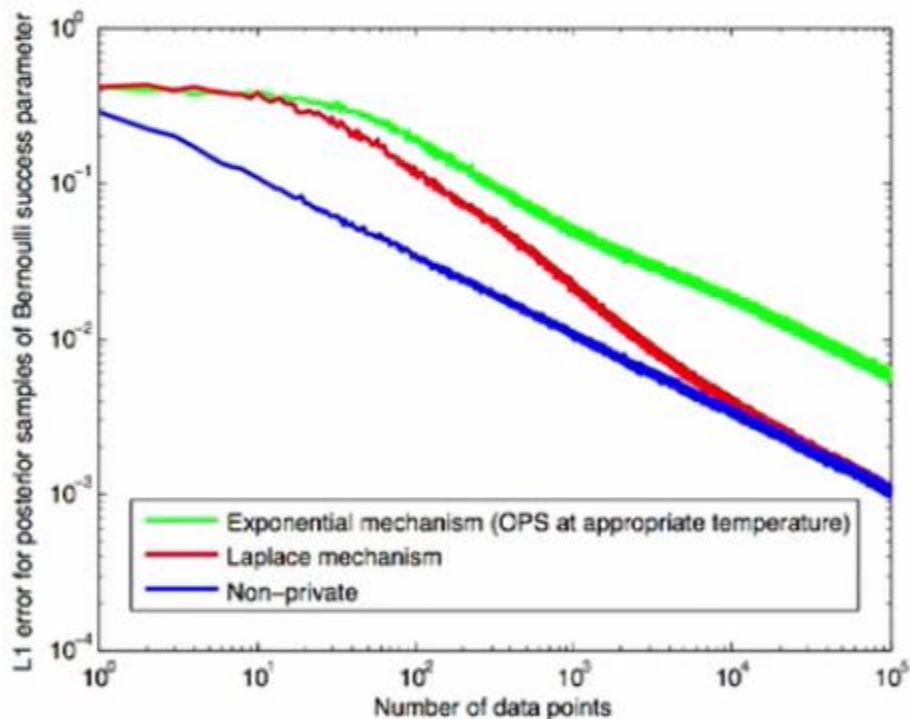


processed posteriors

1. Truncate posterior so that likelihood ratio is bounded in the truncated region.
2. Raise truncated posterior to a higher temperature

How to make posterior private?

Option 2: One Posterior Sample (OPS) Method:



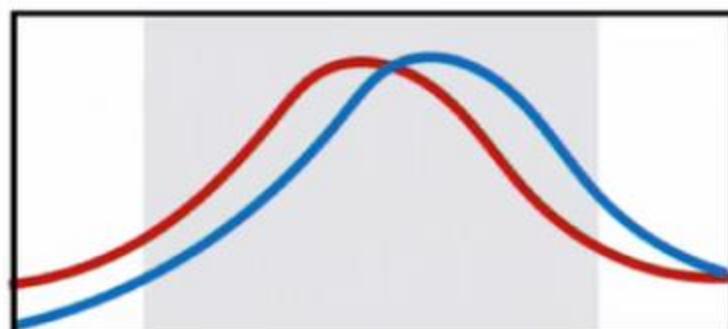
Advantage: General

Pitfalls:

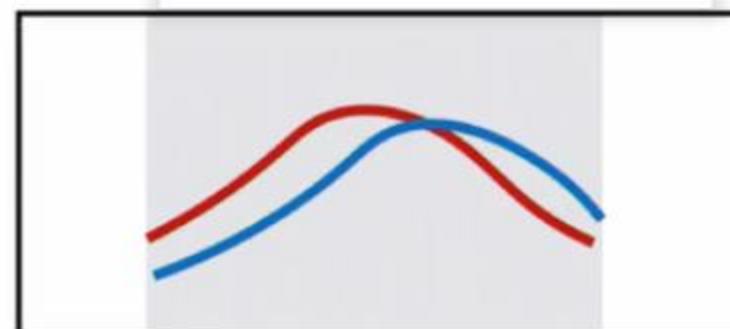
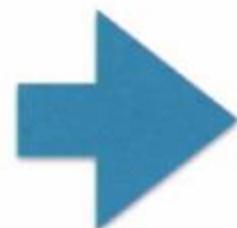
- Intractable - only exact distribution private
- Low statistical efficiency even for large n

How to make posterior private?

Option 3: Approximate the OPS distribution via Stochastic Gradient MCMC [WFS15]



original posteriors



processed posteriors

Advantage: Noise added during stochastic gradient MCMC contributes to privacy

Disadvantage: Statistical efficiency lower than exact OPS

Exponential Family Posteriors

(Non-private) posterior comes from exp. family:

$$p(\theta|x) \propto e^{\eta(\theta)^\top (\sum_i T(x_i)) - B(\theta)}$$

given data x_1, x_2, \dots

Posterior depends on data through sufficient statistic T

Exponential Family Posteriors

(Non-private) posterior comes from exp. family:

$$p(\theta|x) \propto e^{\eta(\theta)^\top (\sum_i T(x_i)) - B(\theta)}$$

given data x_1, x_2, \dots , sufficient statistic T

Private Sampling:

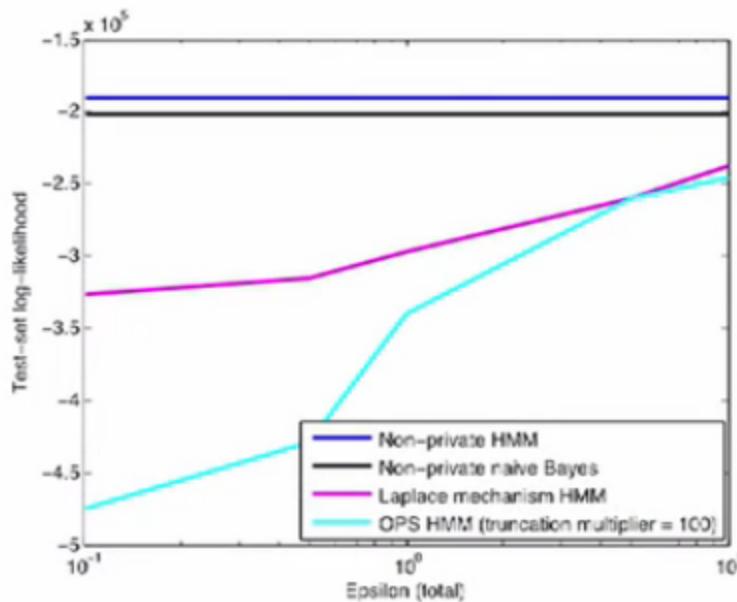
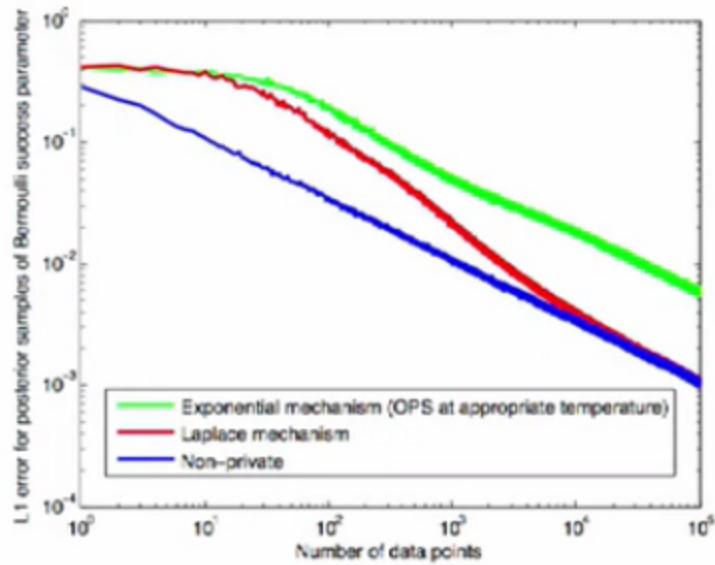
1. If T is bounded, add noise to $\sum_i T(x_i)$ to get private version T'

2. Sample from the perturbed posterior:

$$p(\theta|x) \propto e^{\eta(\theta)^\top T' - B(\theta)}$$

[ZRD16, FGWC16]

How well does it work?



Statistically efficient

Performance worse than non-private, better than OPS

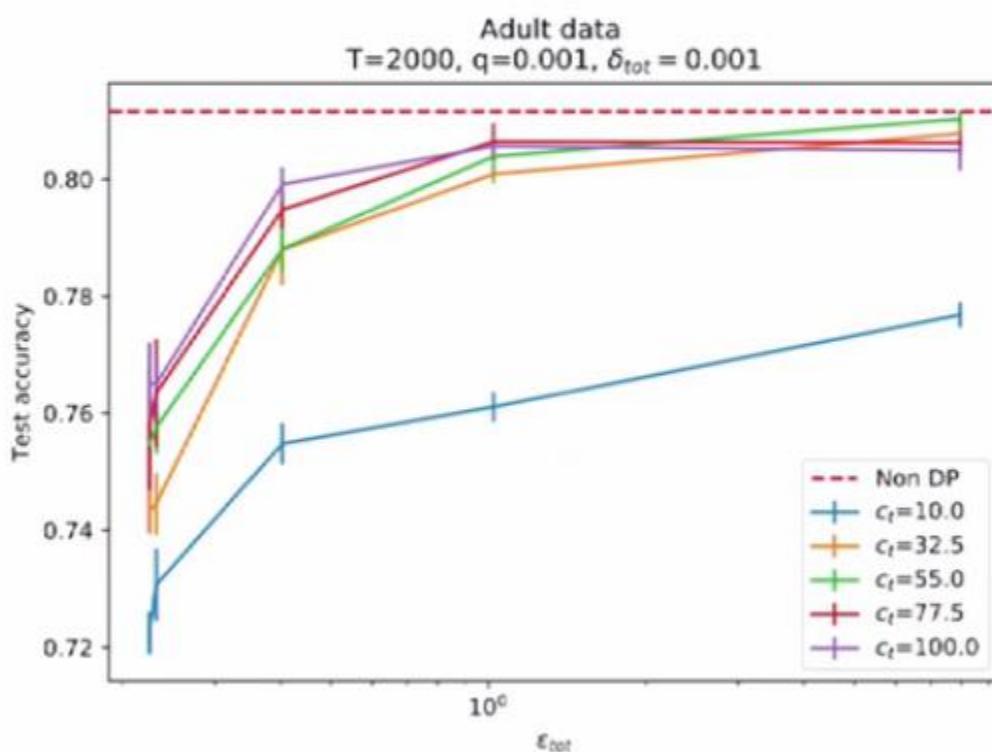
Can do inference in relatively complex systems by building up on this method — eg, time series clustering in HMMs

Differentially Private Bayesian Inference

- General methods for private posterior approximation
- A Special Case: Exponential Families
- Variational Inference

Variational Inference

Key Idea: Start with a stochastic variational inference method, and make each step private by adding Laplace noise. Use moments accountant and subsampling to track privacy loss.



[JDH16, PFCW16]

Summary

- Two examples of differentially private complex machine learning algorithms
 - Deep learning
 - Bayesian inference

Summary

1. Differential privacy: basic definitions and mechanisms
2. Differential privacy and statistical learning: ERM and SGD.
3. Composition and tracking privacy loss
4. Applications of differential privacy in ML: deep learning and Bayesian methods

Things we didn't cover...

- Synthetic data generation.
- Interactive data analysis.
- Statistical/estimation theory and fundamental limits
- Feature learning and dimensionality reduction
- Systems questions for large-scale deployment
- ... and many others...

Where to learn more

Several video lectures and other more technical introductions available from the Simons Institute for the Theory of Computing:

<https://simons.berkeley.edu/workshops/bigdata2013-4>

Monograph by Dwork and Roth:

<http://www.nowpublishers.com/article/Details/TCS-042>

Final Takeaways

- Differential privacy measures the risk incurred by algorithms operating on private data.
- Commonly-used tools in machine learning can be made differentially private.
- Accounting for total privacy loss can enable more complex private algorithms.
- Still lots of work to be done in both theory and practice.

Thanks!