CS208: Applied Privacy for Data Science
Machine Learning under DP: Lecture

James Honaker & Salil Vadhan

School of Engineering & Applied Sciences
Harvard University

April 8, 2019

CRCS Center for Research on Computation and Society

Some Survey Feedback

- Practicums a bit fast
- Difficulties with R
- Helpful to work through details on board during lecture
- Difficulty on high side, ps1 20hrs median, ps2 15hrs median
- Section extremely valuable
- Top topic choices:
  - Differentially private machine learning (this week)
  - Industry lecture (trying to arrange)
  - Statistical inference under DP (we'll see…)

  Also going to do a week on law and policy.

Supervised ML Inputs

- Data $(x_1, y_1), \ldots, (x_n, y_n) \sim \mathcal{P}$
  - Examples $x_i \in \mathcal{X}$ d-dimensional, discrete or continuous
  - Labels $y_i \in \mathcal{Y}$ 1-dimensional, discrete or continuous
  - $\mathcal{P}$ typically unknown
- A family $\mathcal{M}$ of models $m_\theta\colon \mathcal{X} \to \mathcal{Y}$
  - Parameters $\theta \in \Theta$ are k-dimensional, discrete or continuous
  - Linear regression: $m_{\beta,\alpha}(x) = \langle \beta, x \rangle + \alpha$
    or $m_{\beta,\alpha,\sigma}(x) = \langle \beta, x \rangle + \alpha + \mathcal{N}(0, \sigma^2)$.
  - Deep neural nets: $\theta =$ vector of weights at all nodes
- A loss function $\ell : \Theta \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$
  - Classification error: $\ell(\theta | x, y) = I(m_\theta(x) \neq y)$.
  - Squared loss: $\ell(\theta | x, y) = |m_\theta(x) - y|^2$.
  - Negative Log-likelihood: $\ell(\theta | x, y) = -\log \Pr_m[m_\theta(y) = x]$

Supervised ML Output

Primary Goal (risk minimization):

- Find $\theta \in \Theta$ minimizing $L(\theta) = \mathrm{E}_{(x,y)\sim\mathcal{P}}[\ell(\theta|x,y)]$.
- Difficulty: $\mathcal{P}$ unknown.

Subgoal 1 (empirical risk minimization (ERM)):

- Find $\theta \in \Theta$ minimizing $L(\theta|\vec{x},\vec{y}) = \frac{1}{n}\sum_{i=1}^{n} \ell(\theta|x_i,y_i)$.
- Turns learning into optimization.
- Difficulty: overfitting*

Subgoal 2 (regularized ERM):

- Find $\theta \in \Theta$ minimizing $L(\theta|\vec{x},\vec{y}) = \frac{1}{n}\sum_{i=1}^{n} \ell(\theta|x_i,y_i) + R(\theta)$.
- $R(\theta)$ typically penalizes “large” θ, can capture Bayesian prior.

*Fact: DP automatically helps prevent overfitting! [Dwork et al. `15]

APPROACHES TO ML WITH DP

Output Perturbation

[Chaudhuri-Monteleoni-Sarwate `11]

$$M(\vec{x},\vec{y}) = \mathrm{argmin}_\theta \left(\frac{1}{n} \sum_{i=1}^{n} \ell(\theta|x_i, y_i) + R(\theta) \right) + \mathrm{Noise}$$

Challenge: bounding sensitivity of $\theta_{opt} = \mathrm{argmin}_\theta(\cdot)$

- Global sensitivity can be infinite (e.g. OLS regression)
- Global sensitivity can be bounded when ℓ is strictly convex, has bounded gradient (as a function of θ), and R is strongly convex. Even analyzing local sensitivity seems to require unique global optimum and using an optimizer that is guaranteed to succeed.

Objective Perturbation

[Chaudhuri-Monteleoni-Sarwate `11]

$$M(\vec{x}, \vec{y}) = \mathrm{argmin}_\theta \left(\frac{1}{n} \sum_{i=1}^{n} \ell(\theta | x_i, y_i) + R(\theta) + R_{\mathrm{priv}}(\theta, \mathrm{noise}) \right)$$

Challenge: how to put noise in the objective function?

- [CMS11] use $R_{\mathrm{priv}}(\theta, v) = \langle \theta, v \rangle + c\|\theta\|^2$ where v is sampled with probability density $\propto \exp(-c' \varepsilon \|v\|)$.
- Privacy proven under similar assumptions on ℓ and R as before, plus ℓ having bounded Jacobian.
- Has better performance than output perturbation [CMS11].

Exponential Mechanism for ML

[Kasiwiswanathan-Lee-Nissim-Raskhodnikova-Smith `11]

Use utility function

$$u\big((\vec{x},\vec{y}),\theta\big) = -L(\theta|\vec{x},\vec{y}) = -\frac{1}{n}\sum_{i=1}^{n}\ell(\theta|x_i,y_i) - R(\theta).$$

That is,

$$\Pr[M(\vec{x},\vec{y}) = \theta] \propto e^{-\frac{\varepsilon}{2}\sum_{i=1}^{n}\ell(\theta|x_i,y_i) - \frac{\varepsilon n}{2}R(\theta)}.$$

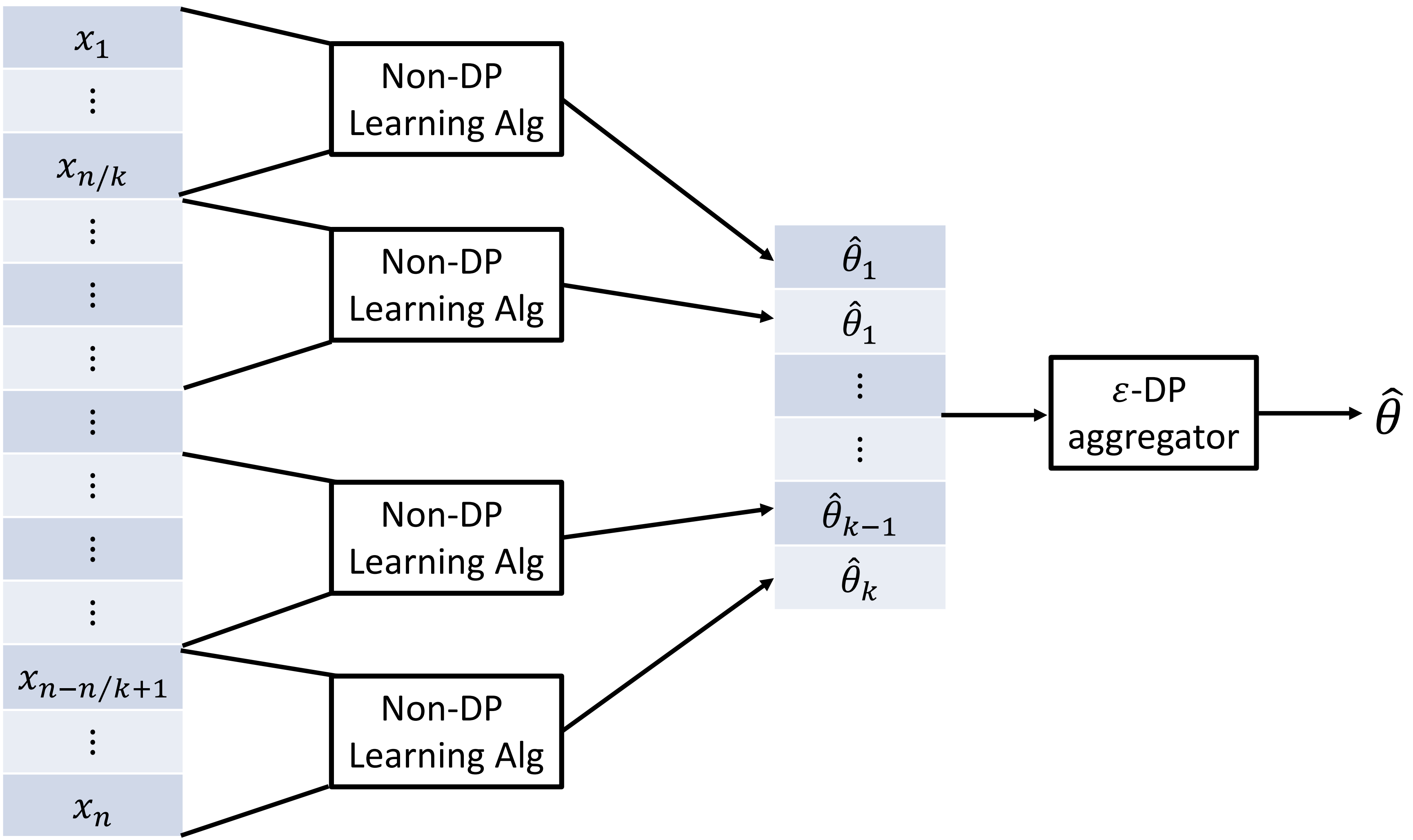
Is ε-DP if the loss functions are clipped to $[0,1]$. (why?)

Thm [KLNRS `11, informally stated]: anything learnable non-privately on a finite data universe is also learnable with DP (with larger n).

Problem: runtime often exponential in dimensionality of θ.

Subsample & Aggregate

[Nissim-Rakhodnikova-Smith `07, Smith `11]



Q: Why is this ε-DP?

Subsample & Aggregate

[Nissim-Rakhodnikova-Smith `07, Smith `11]

- Typical aggregators: DP (clipped) mean, DP median
- Benefits:
  - Use any non-private estimator as a black box
  - Can give optimal asymptotic convergence rates: for many statistical estimators, variance is asymptotically $c_\theta/(\text{sample size})$, so variance of DP mean $\hat{\theta}$ is

$$(1/k)\cdot(c_\theta\cdot k/n)+O(1/\varepsilon k)^2=\big(1+o(1)\big)\cdot c_\theta/n$$

if $k=\omega\big(\sqrt{n}\big)$.

- Drawbacks:
  - Dependence on dimension, model parameters, distribution can be bad.
  - Often takes very large sample size to kick in.

Modifying ML Algorithms

- Another approach: decompose existing ML/inference algorithms into steps that can be made DP, like Statistical Queries (estimating means of bounded functions)

- Example: linear regression
  - $S_{xx}/n\,, S_{xy}/n\,, \bar{x}, \bar{y}$ are all statistical queries

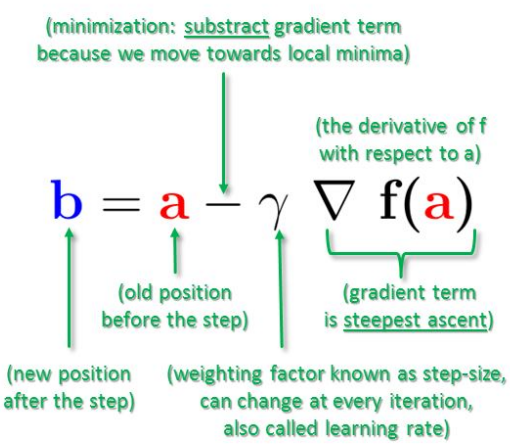
- Today: gradient descent and variants
  - main method used in practice for training deep neural nets

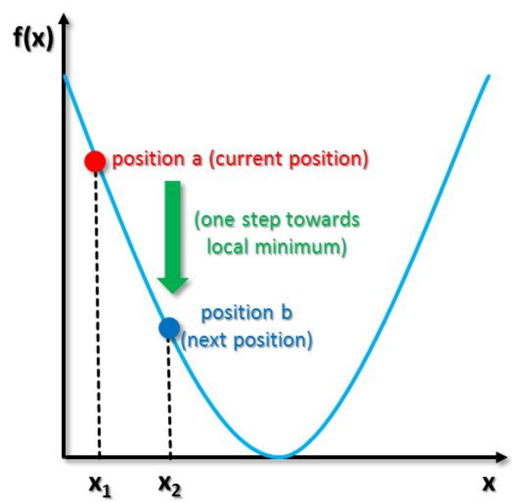
(slides modified from Adam Smith, BU CS 591 Fall 2018)

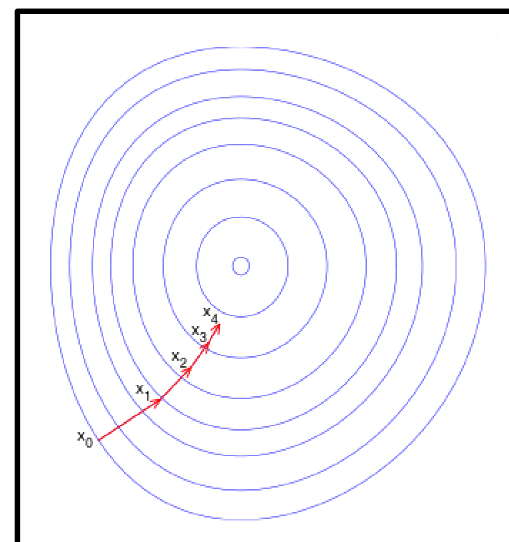
GRADIENT DESCENT WITH DP

Gradient Descent

- Proceed in steps
- Start from (carefully chosen) initial parameters $\hat{\theta}_0$
- At each step, move in direction opposite to the gradient of the loss $\nabla L(\hat{\theta} \mid \vec{x}, \vec{y})$.
- Gradient is the vector of partial derivatives





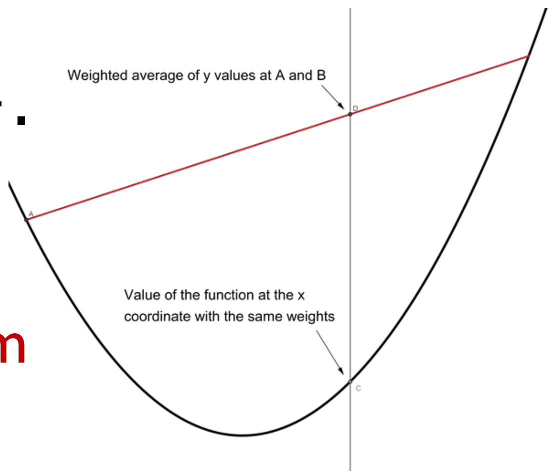


[Wikipedia]

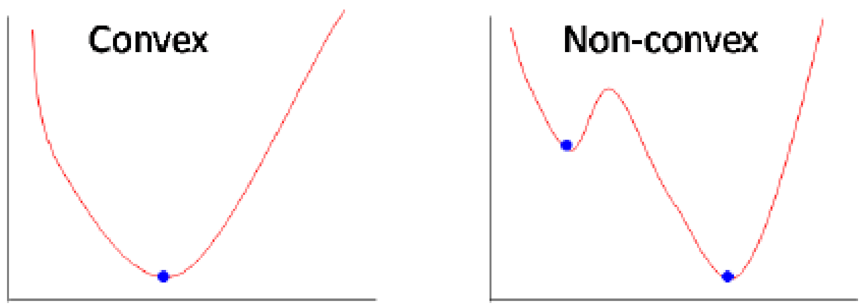
Gradient Descent for Convex Loss

- Def: L is convex if for all points $\vec{a}, \vec{b}$, we have

$$L\left(\frac{\vec{a}+\vec{b}}{2}\right) \leq \frac{L(\vec{a}) + L\left(\vec{b}\right)}{2}.$$

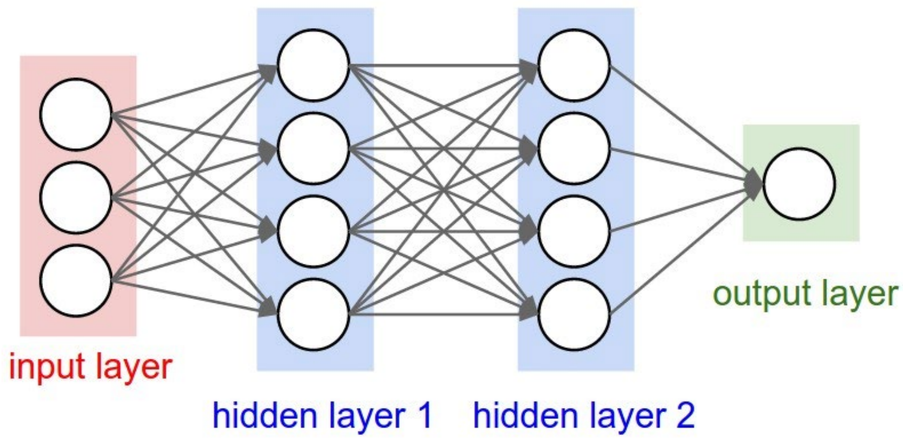


- Convex functions have no local minima
  - and gradient descent finds a global minimum



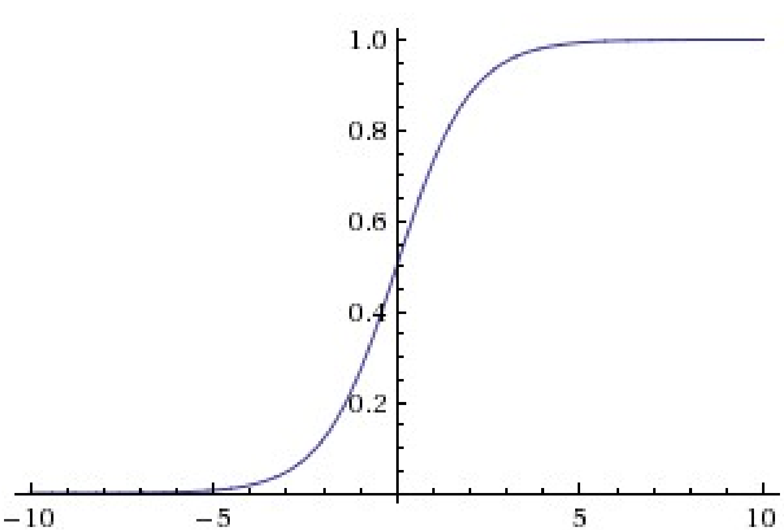
- Loss function for logistic regression is convex
  - No closed form solution for minimum, but it is easy to find
  - Gradient easy to compute

Gradient Descent for Neural Networks

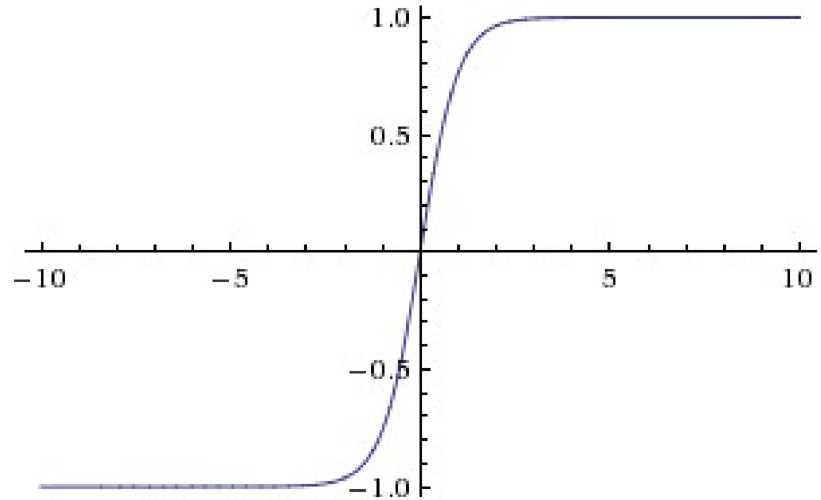


- Each node is a linear function of inputs (specified by θ) composed with a nonlinear "activation" function
- Gradient of Loss function can be computed quickly
  - Using chain rule (technique called "backpropagation")
- But no longer convex, has many local minima
  - Can get stuck in a bad place
  - But works well in practice!

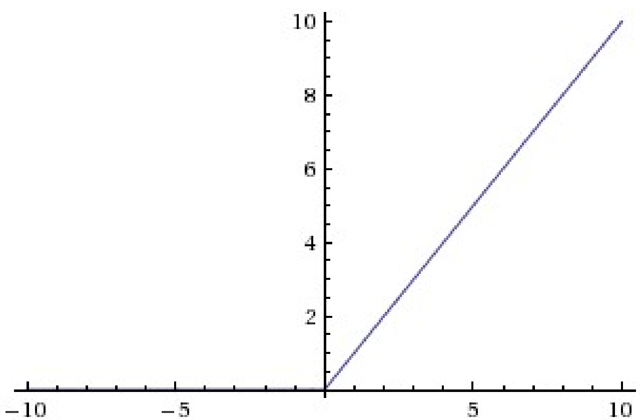
Common activation functions



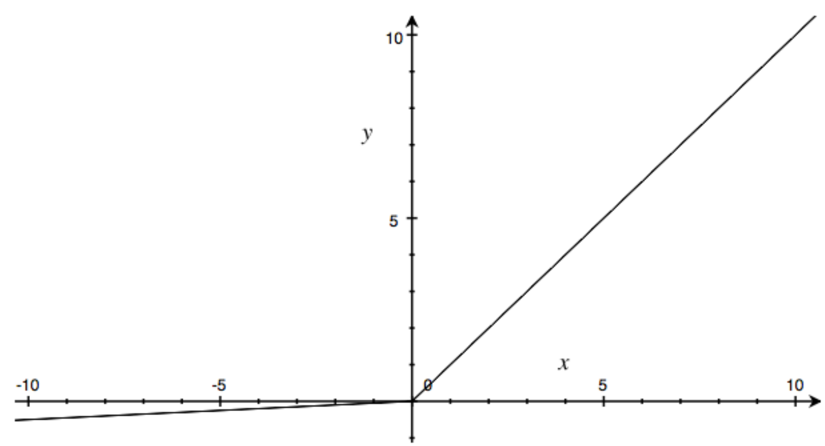
Sigmoid $\sigma(x) = \frac{1}{1+e^{-x}}$



$\tanh(x) = 2\sigma(2x) - 1$



$\text{ReLU}(x) = \max(0, x)$



$\text{Leaky ReLU}(x) = \max(0.05x, x)$

Neural Networks & Privacy

- Best known models for many problems are DNNs
  - Sentence prediction (smart completion)
  - Image/video/text recognition
  - Author recognition
  - Textual analysis
  - …
- Problem: The best DNNs end up memorizing their inputs!
  - Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. International Conference on Learning Representations (ICLR), 2017.
  - Nicholas Carlini, Chang Liu, Jernej Kos, Úlfar Erlingsson, Dawn Song. The Secret Sharer: Measuring Unintended Neural Network Memorization & Extracting Secrets. arXiv:1802.08232
- Potential solution
  - Differentially private training of DNNs
  - Input: training data
  - Output: architecture & weights

Gradient Descent: Formal Description

- Specify
  - Number of steps T
  - Learning rate η

- Pick initial point $\hat{\theta}_0$
- For $t = 1$ to T
  - Compute gradient

$$g_t = \frac{1}{n}\sum_i \nabla \ell\left(\hat{\theta}_{t-1} | x_i, y_i\right) + \nabla R\left(\hat{\theta}_{t-1}\right)$$

  - $\hat{\theta}_t = \hat{\theta}_{t-1} - \eta \cdot g_t$

DP for Vector-Valued Functions

- Let $f : \mathcal{X}^n \to \mathbb{R}^k$, and $M(x) = f(x) + Z$ for noise $Z \in \mathbb{R}^k$.
- Def: for a norm $\|\cdot\|$ on $\mathbb{R}^k$. the global $\|\cdot\|$-sensitivity of f is

$$\mathrm{GS}_{f,\|\cdot\|} \stackrel{\text{def}}{=} \max_{x \sim x'} \|f(x) - f(x')\|.$$

- $\|\cdot\|$-Norm Mechanism: density of Z at z is $\propto \; e^{-\varepsilon \|z\| / \mathrm{GS}_{f,\|\cdot\|}}$
  - ε-DP for any norm $\|\cdot\|$. [Hardt-Talwar 2010]
  - If $\|z\| = \sum_j |z_j|$ [ℓ_1-norm]: independent Laplace noise per coordinate.
- Gaussian Mechanism: $Z \sim \mathcal{N}\left(\vec{0}, \left(\frac{\mathrm{GS}_{f,\|\cdot\|}}{\varepsilon}\right)^2 \cdot \ln \frac{1.25}{\delta} \cdot I_k\right)$
  - When $\|z\| = \left(\sum_j |z_j|^2\right)^{1/2}$ [ℓ_2-norm], (ε, δ)-DP and independent Gaussian noise per coordinate.

DP Gradient Descent

[Williams-McSherry`10, …]

- Specify
  - ➢ Number of steps T
  - ➢ Learning rate η
  - ➢ Privacy parameters ε, δ
  - ➢ Clipping parameter Δ. Write $[\vec{z}]_\Delta = \vec{z} \cdot \max\left(1, \frac{\Delta}{\|\vec{z}\|_2}\right)$.
  - ➢ Noise variance $\sigma^2 = \mathrm{TBD}(T, \varepsilon, \delta, \Delta)$.
- Pick initial point $\hat{\theta}_0$
- For $t = 1$ to T
  - ➢ Estimate gradient as noisy average of clipped gradients
    $\hat{g}_t = \frac{1}{n}\sum_i \left[\nabla \ell\left(\hat{\theta}_{t-1} | x_i, y_i\right)\right]_\Delta + \nabla R\left(\hat{\theta}_{t-1}\right) + \mathcal{N}(0, \sigma^2 I)$
  - ➢ $\hat{\theta}_t = \hat{\theta}_{t-1} - \eta \cdot \hat{g}_t$

DP Gradient Descent: Privacy Analysis

- By Gaussian Mechanism, each iteration is $(\varepsilon_0, \delta_0)$-DP if

$$\sigma^2 = \left(\frac{2\Delta}{\varepsilon_0 n}\right)^2 \cdot \ln \frac{1.25}{\delta_0}$$

- By Advanced Composition for adaptive queries, overall algorithm is (ε, δ)-DP for:

$$\varepsilon = O\left(\varepsilon_0 \cdot \sqrt{T \ln(2/\delta)}\right)$$
$$\delta = 2T \cdot \delta_0$$

- Solving, suffices to use noise variance

$$\sigma^2 = O\left(\left(\frac{\Delta}{\varepsilon n}\right)^2 \cdot T \cdot \ln\frac{T}{\delta} \cdot \ln\frac{1}{\delta}\right)$$

Improved Analysis with "Concentrated DP"

[Dwork-Rothblum `16, Bun-Steinke `16]

- By Gaussian Mechanism, each iteration is ε_0^2 -zCDP if

$$\sigma^2 = 2\left(\frac{\Delta}{\varepsilon_0 n}\right)^2 \cdot \cancel{\ln \frac{1.25}{\delta_0}}$$

- By composition of zCDP, overall alg. is $T \cdot \varepsilon_0^2$-zCDP.

- By properties of zCDP, overall alg is (ε, δ)-DP for:

$$\varepsilon = T \cdot \varepsilon_0^2 + 2\sqrt{T \cdot \varepsilon_0^2 \cdot \ln(1/\delta)}$$

- Solving, suffices to use noise variance

$$\sigma^2 = O\left(\left(\frac{\Delta}{\varepsilon n}\right)^2 \cdot T \cdot \ln\frac{1}{\delta} \cdot \cancel{\ln\frac{T}{\delta}}\right)$$

DP Stochastic Gradient Descent (SGD)

[Jain-Kothari-Thakurta `12, Song-Chaudhuri-Sarwate `13, Bassily-Smith-Thakurta `14]

- Specify
  - Number of steps T, learning rate η, privacy parameters ε, δ, clipping parameter Δ.
  - Batch size $B \ll n$ (for efficiency)
  - Noise variance $\sigma^2 = \mathrm{TBD}(T, \varepsilon, \delta, \Delta, B)$.
- Pick initial point $\hat{\theta}_0$
- For $t = 1$ to T
  - Select a random batch $S_t \subseteq \{1, \ldots, n\}$ of size B.
  - Estimate gradient as noisy average of clipped gradients $\hat{g}_t = \frac{1}{B} \sum_{i \in S_t} \left[\nabla \ell\left(\hat{\theta}_{t-1} | x_i, y_i\right)\right]_\Delta + \nabla R\left(\hat{\theta}_{t-1}\right) + \mathcal{N}(0, \sigma^2 I)$
  - $\hat{\theta}_t = \hat{\theta}_{t-1} - \eta \cdot \hat{g}_t$

DP SGD: Privacy Analysis

- By Gaussian Mechanism, each iteration is ε_0^2 -zCDP if

$$\sigma^2 = 2\left(\frac{\Delta}{\varepsilon_0 B}\right)^2$$

- By composition of zCDP, overall alg. is $T \cdot \varepsilon_0^2$-zCDP.

- By properties of zCDP, overall alg is (ε, δ)-DP for:

$$\varepsilon = T \cdot \varepsilon_0^2 + 2\sqrt{T \cdot \varepsilon_0^2 \cdot \ln(1/\delta)}$$

- Solving, suffices to use noise variance

$$\sigma^2 = O\left(\left(\frac{\Delta}{\varepsilon B}\right)^2 \cdot T \cdot \ln\frac{1}{\delta}\right)$$

- Worse than ordinary gradient descent!

DP SGD: Improved Privacy Analysis

[Bassily-Smith-Thakurta `14, Abadi-Chu-Goodfellow-McMahan-Mironov-Talwar-Zhang `17]

- **Privacy amplification by subsampling:**
  If $S : \mathcal{X}^n \to \mathcal{X}^B$ outputs a random subset of pn out of n rows and $M : \mathcal{X}^B \to \mathcal{Y}$ is (ε, δ)-DP, then $M'(x) = M(S(x))$ is $((e^p - 1) \cdot \varepsilon, p \cdot \delta)$-DP.

- We can take $p = B/n$.
  - Unfortunately does privacy amplification by subsampling does not hold for zCDP.
  - But similar analysis can be recovered using the "moments accountant" [Abadi et al. `17] or "truncated zCDP" [Bun et al. `18].

Local DP SGD *[Duchi-Jordan-Wainwright `14]*

- Specify
  - Number of steps T, learning rate η, privacy parameters ε, δ, clipping parameter Δ.
  - Batch size $B \ll n$ (for efficiency)
  - Noise variance $\tau^2 = \mathrm{TBD}(T, \varepsilon, \delta, \Delta, B)$.
- Pick initial point $\hat{\theta}_0$
- For $t = 1$ to T
  - Server selects a batch $S_t \subseteq \{1, \dots, n\}$ of size B, and sends $\hat{\theta}_{t-1}$ to subjects in S_t.
  - Every subject $i \in S_t$ sends noisy clipped gradient
    $$\hat{g}_{t,i} = \left[\nabla \ell\left(\hat{\theta}_{t-1} | x_i, y_i\right)\right]_\Delta + \mathcal{N}(0, \tau^2 I)$$
  - Server averages to estimate gradient
    $$\hat{g}_t = \frac{1}{B} \sum_{i \in S_t} \hat{g}_{t,i} + \nabla R\left(\hat{\theta}_{t-1}\right)$$
  - $\hat{\theta}_t = \hat{\theta}_{t-1} - \eta \cdot \hat{g}_t$

Local DP SGD: Privacy Analysis

- There is no privacy amplification by subsampling for local DP. But we can ensure each subject participates in only $T \cdot (B/n)$ batches.
- Using zCDP analysis, suffices to use local noise variance

$$\tau^2 = O\left(\left(\frac{\Delta}{\varepsilon}\right)^2 \cdot \left(T \cdot \frac{B}{n}\right) \cdot \ln\frac{1}{\delta}\right)$$

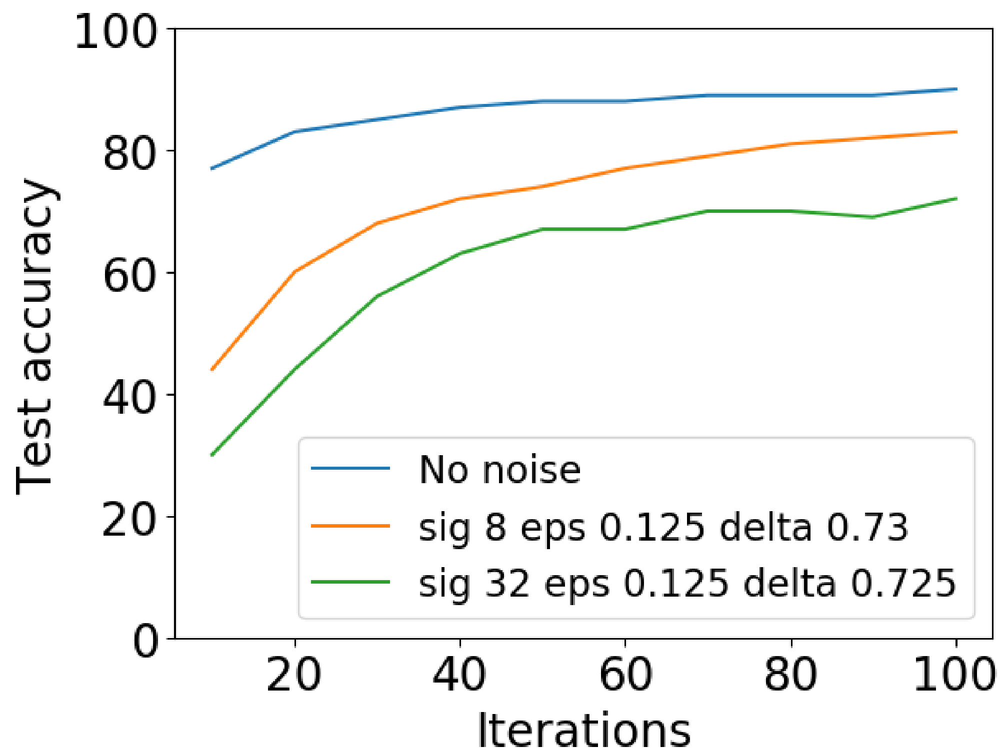
- Since server averages B reports, this is equivalent to centralized DP SGD with noise variance:

$$\sigma^2 = \frac{\tau^2}{B} = O\left(\left(\frac{\Delta}{\varepsilon}\right)^2 \cdot \frac{T}{n} \cdot \ln\frac{1}{\delta}\right)$$

- Can eliminate δ using a pure DP local randomizer (ps4)

Publicly available tools for DP Learning

- Basic Mechanisms as building blocks (R): link
- Techniques for DP Convex Optimization (Python): link
- DP SGD (DNNs) using Tensorflow (Python): link
  - DP MNIST example
  - DP Optimizer link



More on Non-DP Deep Learning

- Stanford CS231n lecture notes
  http://cs231n.github.io/neural-networks-1/
- Deep learning tutorial
  http://www.deeplearning.net/tutorial/mlp.html
- TensorFlow visual demo
  https://playground.tensorflow.org
- Tensorflow and PyTorch tutorials