Reducing ML Bias using Truncated Statistics

Constantinos Daskalakis
EECS and CSAIL, MIT

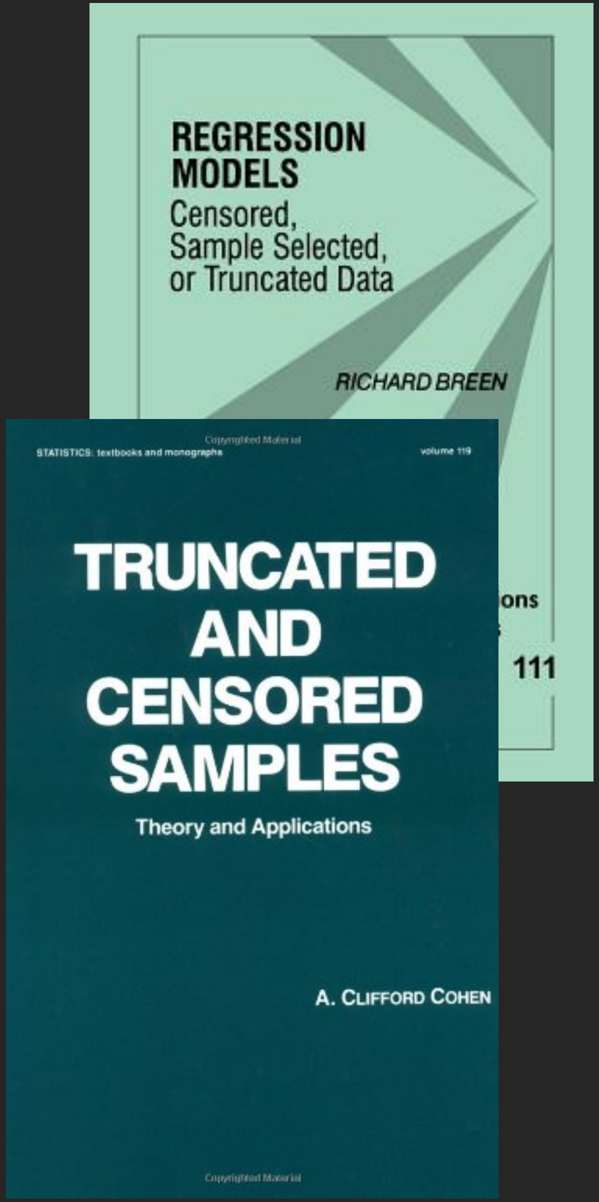
High-Level Goals

- ❑ **Selection bias in data collection**
  - ⇒ **train distribution ≠ test distribution**
  - ⇒ **prediction bias (a.k.a. "ML bias")**
- ❑ **Our Work:** decrease bias, by developing machine learning methods more robust to **censored and truncated samples**

Truncated Statistics: samples falling outside of observation "window" are hidden and their count is also hidden

Censored Statistics: ditto, but count of hidden data is provided

- limitations of measurement devices
- limitations of data collection
  - experimental design, ethical or privacy considerations,…



Motivating Example: IQ vs Income

Goal: Regress (IQ, Training, Education) vs Earnings [Wolfle&Smith'56, Hause'71,…]

Data Collection: survey families whose income is smaller than 1.5 times the poverty line; collect data $(x_i, y_i)_i$ where

- x_i: (IQ, Training, Education,…) of individual i
- y_i: earnings of individual i

Regression: fit some model $y = h_\theta(x) + \varepsilon$, e.g. $y = \theta^T x + \varepsilon$

Obvious Issue: **thresholding incomes may introduce bias**

It does, as pointed out by [Hausman-Wise, Econometrica'76] debunking prior results "showing" effects of education are strong, while of IQ and training are not

Motivating Example 2: Height vs Basketball

Goal: Regress **Height vs Basketball Performance**

Data Collection: use **NBA data** $(x_i, y_i)_i$ where

- x_i: height of individual i
- y_i: average number of points per game scored by individual i

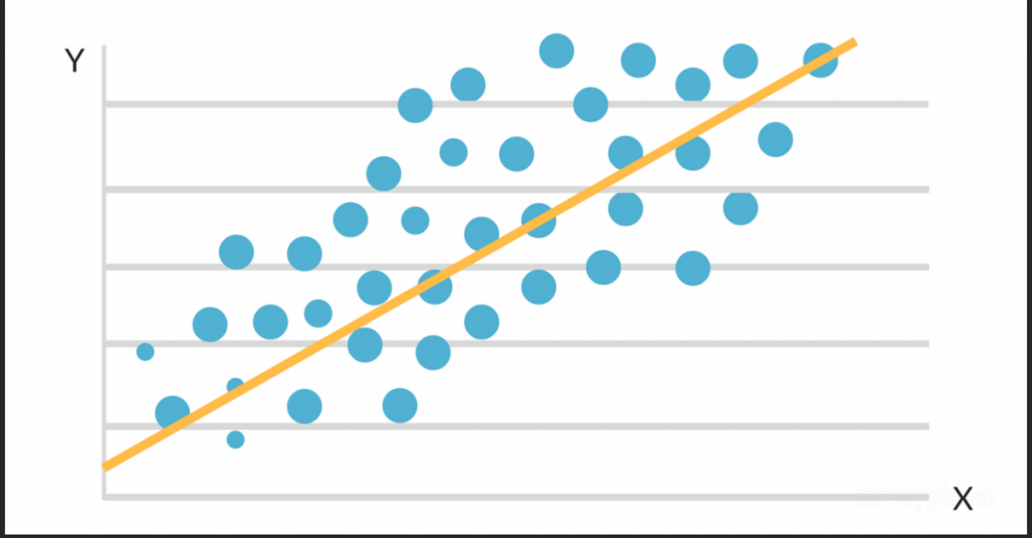
Regression: fit some model $y = h_\theta(x) + \varepsilon$

Obvious Issue: by using NBA data, we might infer that height is neutral or even negatively correlated with performance

What Happened?

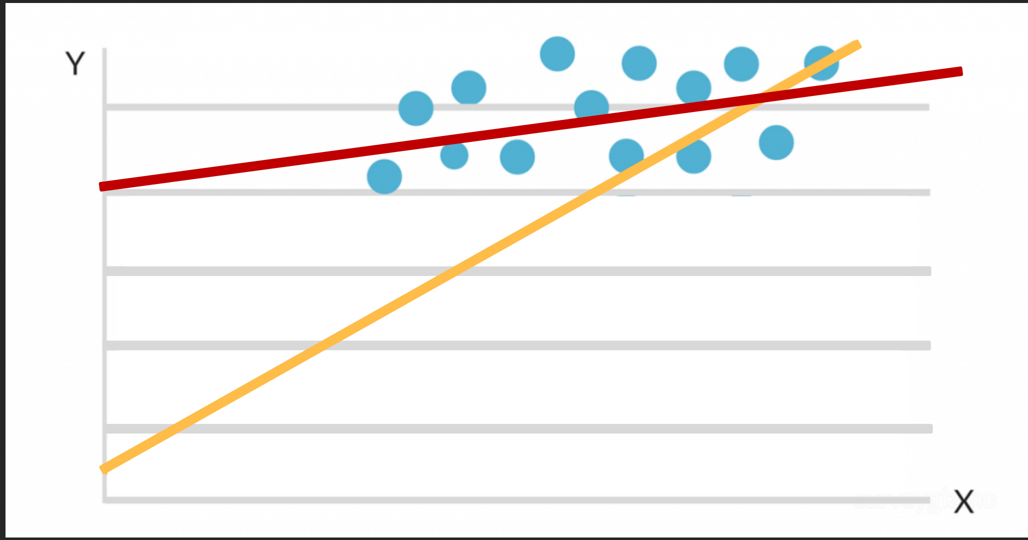
Mental Picture:

Vanilla Linear Regression



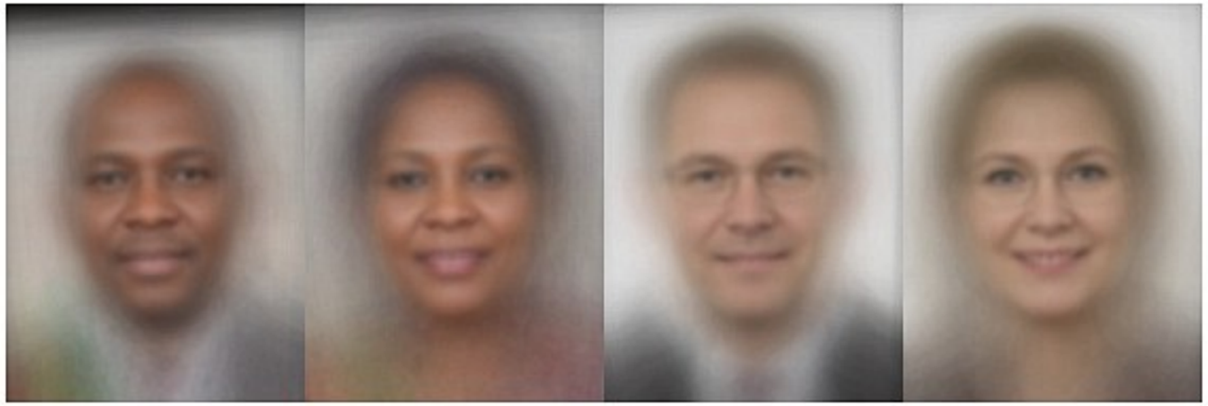
Truth: $y_i = \theta \cdot x_i + \varepsilon_i$, for all i

Data truncated on the Y-axis



Motivating Eg 3: Truncation on the X-axis

Gender Classifier	Darker Male	Darker Female	Lighter Male	Lighter Female	Largest Gap
Microsoft	94.0%	79.2%	100%	98.3%	20.8%
FACE++	99.3%	65.5%	99.2%	94.0%	33.8%
IBM	88.0%	65.3%	99.7%	92.9%	34.4%



Buolamwini, Gebru, FAT 2018

Explanation: Training data contains more faces that are of lighter skin tone, male gender, Caucasian

⇒ Training loss of gender classifier pays less attention to faces that are of darker skin tone, female gender, non-Caucasian

⇒ Test loss on faces that are of darker skin tone, female gender, non-Caucasian is worse

Classical example of bias in ML systems

Menu

- **Motivation**
- Flavor of Models, Techniques, Results

Menu

- **Motivation**
- **Flavor of Models, Techniques, Results**

Menu

- **Motivation**
- **Flavor of Models, Techniques, Results**
  - Supervised learning: known truncation on the y-axis
  - Supervised learning: unknown truncation on the x-axis
  - Unsupervised learning: learning truncated densities

Menu

- **Motivation**
- **Flavor of Models, Techniques, Results**
  - Supervised learning: known truncation on the y-axis
    - small dive into techniques
  - Supervised learning: unknown truncation on the x-axis
  - Unsupervised learning: learning truncated densities
    - bigger dive into techniques

Menu

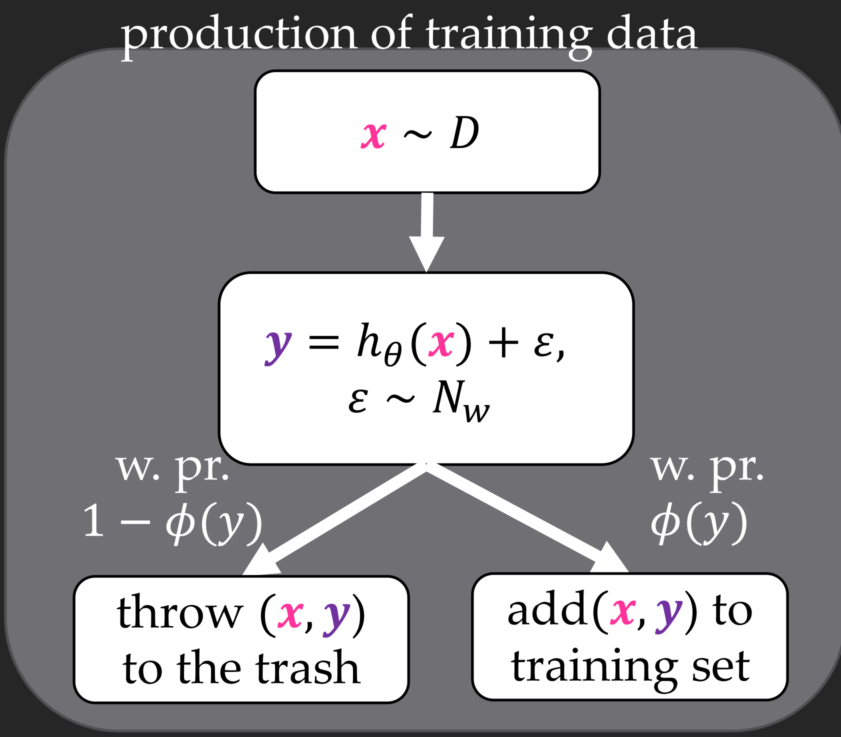
- **Motivation**
- **Flavor of Models, Techniques, Results**
  - **Supervised learning: known truncation on the y-axis**
    - **small dive into techniques**
  - Supervised learning: unknown truncation on the x-axis
  - Unsupervised learning: learning truncated densities
    - bigger dive into techniques

Problem 1: Truncation on the Y-Axis

(recall: IQ vs Earnings, Height vs Basketball)

Truncated Regression Model:

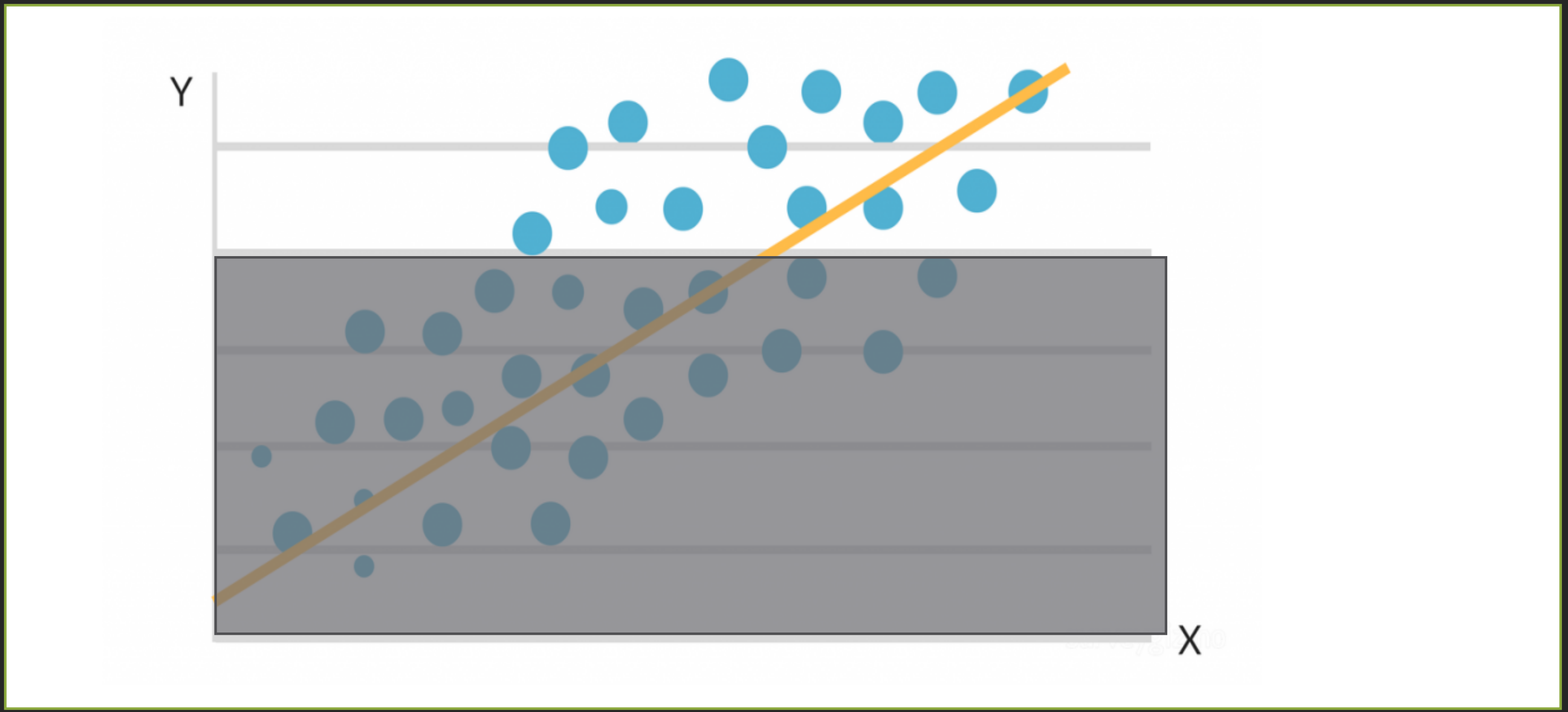
- (*unknown*) distribution D over covariates x
- (*unknown*) response mechanism $h_\theta: x \mapsto y,\ \theta \in \Theta$
- (*unknown*) noise distribution $N_w,\ w \in \Omega$
- (*known*) filtering mechanism $\phi: y \mapsto p \in [0,1]$



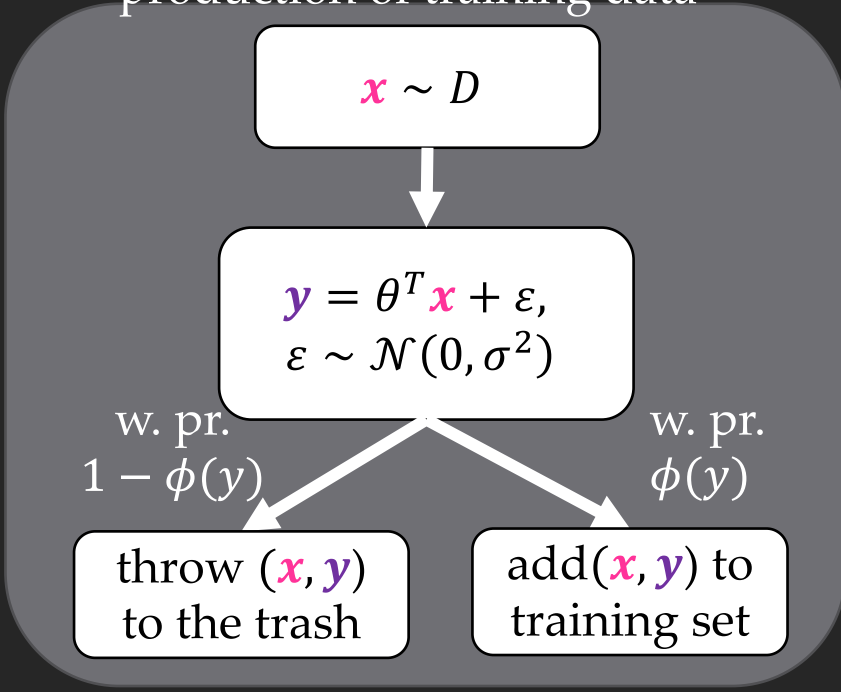
Problem 1: Truncation on the Y-Axis

(recall: IQ vs Earnings, Height vs Basketball)

Truncated Regression Model:



production of training data



Problem 1: Truncation on the Y-Axis

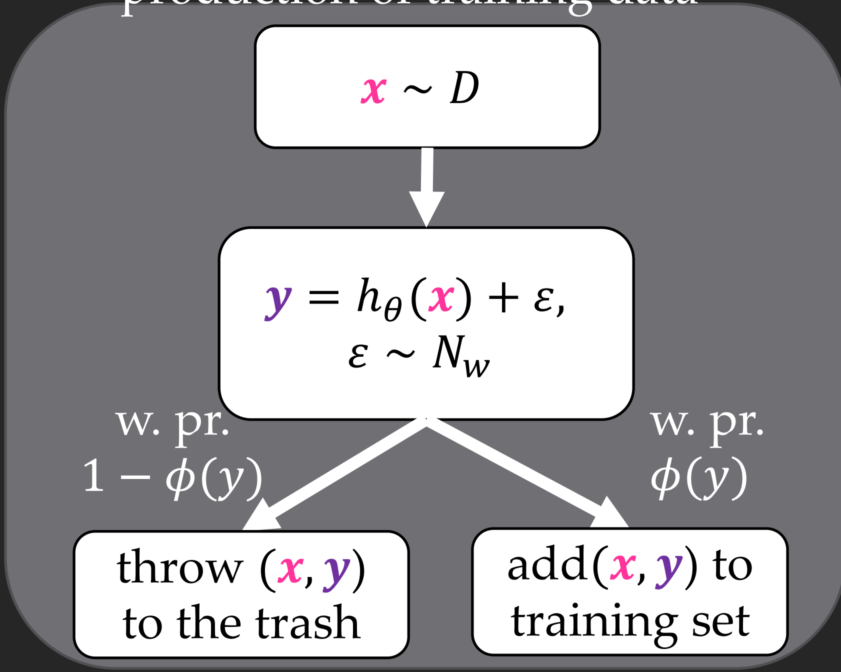
(recall: IQ vs Earnings, Height vs Basketball)

Truncated Regression Model:

- (*unknown*) distribution D over covariates x
- (*unknown*) response mechanism $h_\theta: x \mapsto y$, $\theta \in \Theta$
- (*unknown*) noise distribution N_w
- (*known*) filtering mechanism $\phi: y \mapsto p \in [0,1]$

Goal: given filtered data $(x_i, y_i)_i$ recover θ

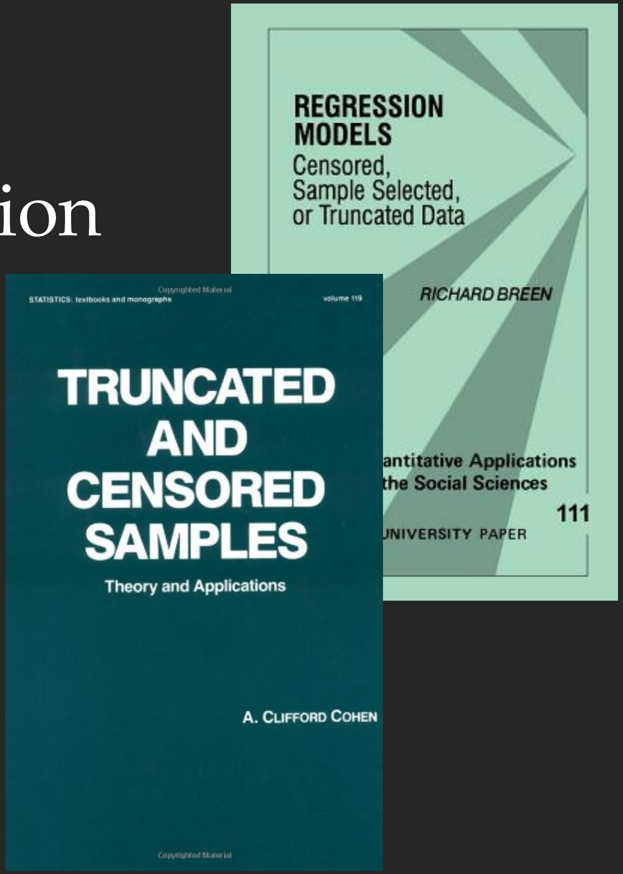
production of training data



Results [w/ Gouleakis, Tzamos, Zampetakis COLT'19, w/ Ilyas, Rao, Zampetakis'19] :

- Practical, SGD-based likelihood optimization framework
- Computationally and statistically efficient recovery of true parameters for truncated linear/probit*/logistic regression*
  - **prior work:** inefficient algorithms, and error rates exponential in dimension

Comparison to Prior Work On Truncated Regression



Asymptotic Analysis of Truncated/Censored Regression

[Tobin 1958], [Amemiya 1973], [Hausman, Wise 1976], [Breen 1996], [Hajivassiliou-McFadden'97], [Balakrishnan, Cramer 2014], Limited Dependent Variables models, Method of Simulated Scores, GHK Algorithm

Technical Bottlenecks:

- Convergence rates: $O_d\left(\frac{1}{\sqrt{n}}\right)$
- Computationally inefficient algorithms

Our work: optimal rates $O\left(\sqrt{\frac{d}{n}}\right)$, efficient algorithms, arbitrary truncation sets

Assumptions: whatever needed for standard regression + roughly that the average x_i in the training set has $\geq c$ probability of resulting in some y_i that won't be pruned

Technical Vignette: Truncated Linear Regression

Data distribution: $p_\theta(x,y) = \frac{1}{Z_\theta} \cdot D(x) \cdot e^{-\frac{\left(y-\theta^T x\right)^2}{2}} \cdot \phi(y)$

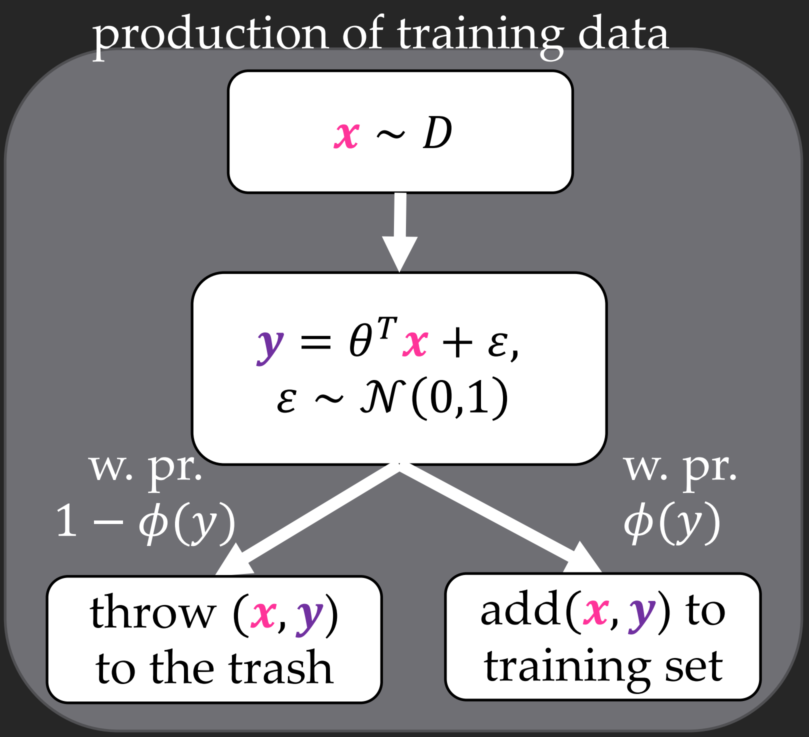
Population Log-Likelihood:

$$\mathrm{LL}(\theta) = \mathbb{E}_{(x,y)\sim \mathrm{p}_{\theta^*}}\left[\log D(x) - \frac{(y-\theta^T x)^2}{2} + \log \phi(y) - \log Z_\theta\right]$$

Issue: $\mathrm{LL}(\theta)$ involves stuff we don't know (D), and even if we did it involves stuff that is difficult to calculate (Z_θ)

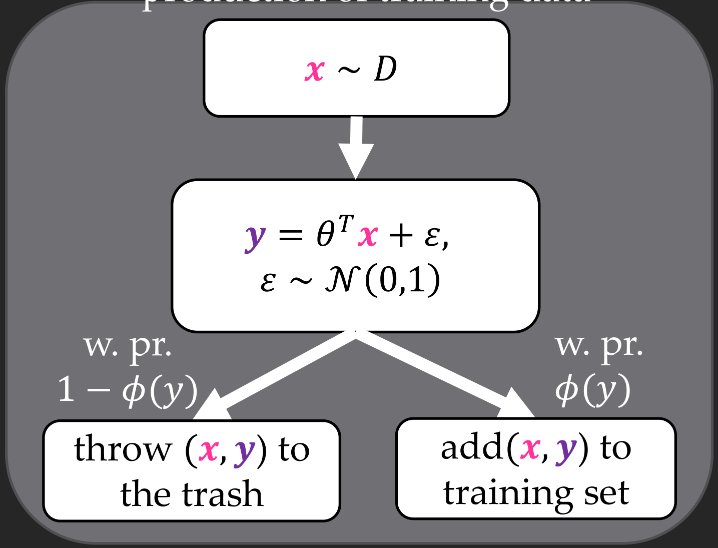
Yet, Stochastic Gradient Descent (SGD) *can* be performed on negative log-likelihood!

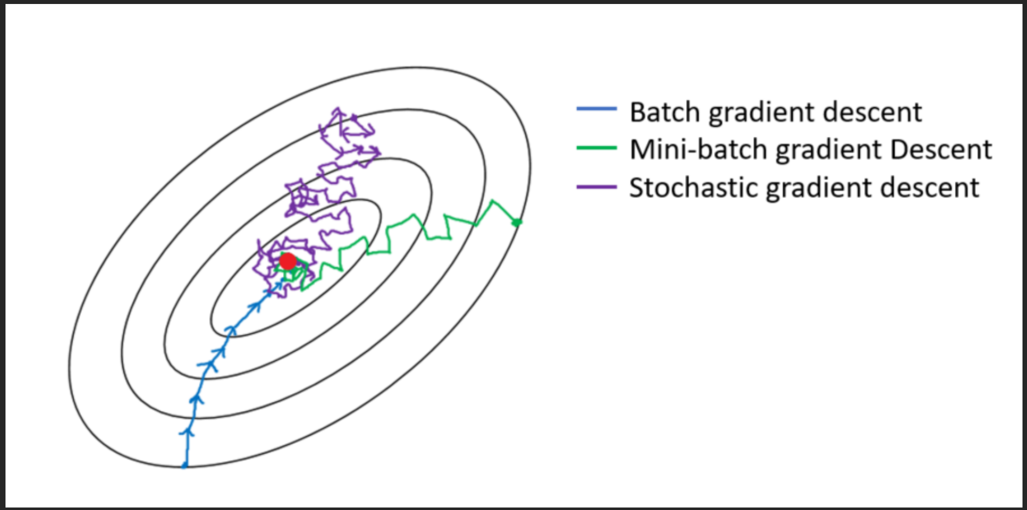
In particular, easy to define random variable whose expectation is the gradient at a given θ, without knowledge of D and no need to compute Z_θ



Technical Vignette: Truncated Linear Regression

production of training data





$$\mathrm{LL}(\theta) = \mathbb{E}_{(x,y)\sim p_{\theta^*}}\left[\log D(x) - \frac{(y-\theta^T x)^2}{2} + \log \phi(y)\right] - \log Z_\theta$$

$$\nabla_\theta \mathrm{LL}(\theta) = \mathbb{E}_{(x,y)\sim p_{\theta^*}}\left[-(y-\theta^T x)x\right] - \mathbb{E}_{(x,y)\sim p_{\theta}}\left[-(y-\theta^T x)x\right]$$

Easy* to sample r.v. whose expectation is the gradient at a given θ, with no need to compute Z_θ

Summary: We cannot run **blue** or **green**, but we *can* run **purple**

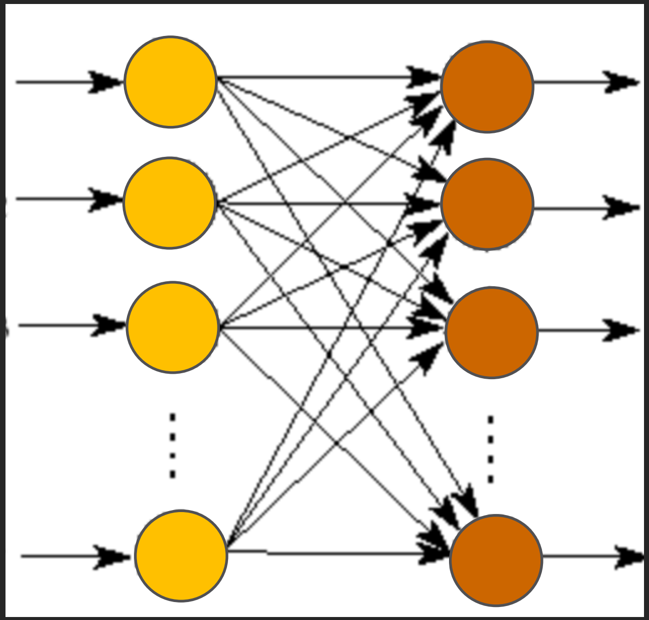
Issue 2: this random variable better be efficiently samplable, have small variance

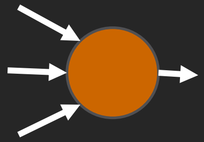
Requires restricting optimization in appropriately defined space

Issue 3: for parameter estimation need neg. log-likelihood to be strongly convex

Requires anti-concentration of measure

E.g. Application: Learning Single-layer Relu Nets



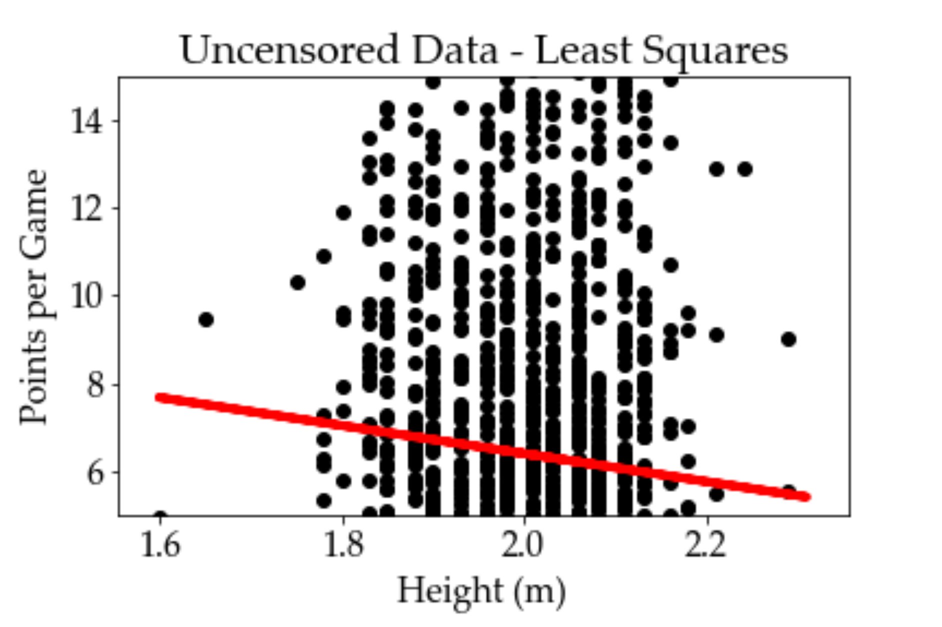


$= \text{Noisy-Relu} = \max\{0, w^T \cdot x + \varepsilon\}$,
where $\varepsilon \sim \mathcal{N}(0,1)$

Direct corollary: In the realizable setting, given input-output pairs, obtain $O\left(\sqrt{\frac{input-dimension}{n}}\right)$ error rate

E.g. Application 2: NBA data

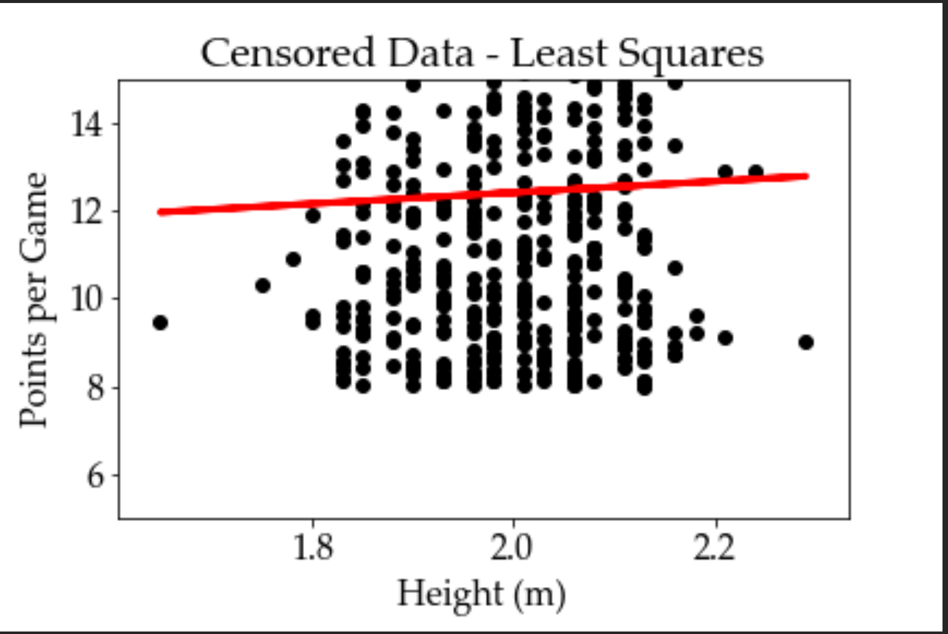
NBA player data after year 2000:
x_i: height of player i
y_i: number of points per game of player i



Points per Game are **negatively correlated** with height!

E.g. Application 2: NBA data

NBA player data after year 2000:
x_i: height of player i
y_i: number of points per game of player i
filtering : look at players with at least 8 points per game



Points per Game seem **positively correlated** with height!

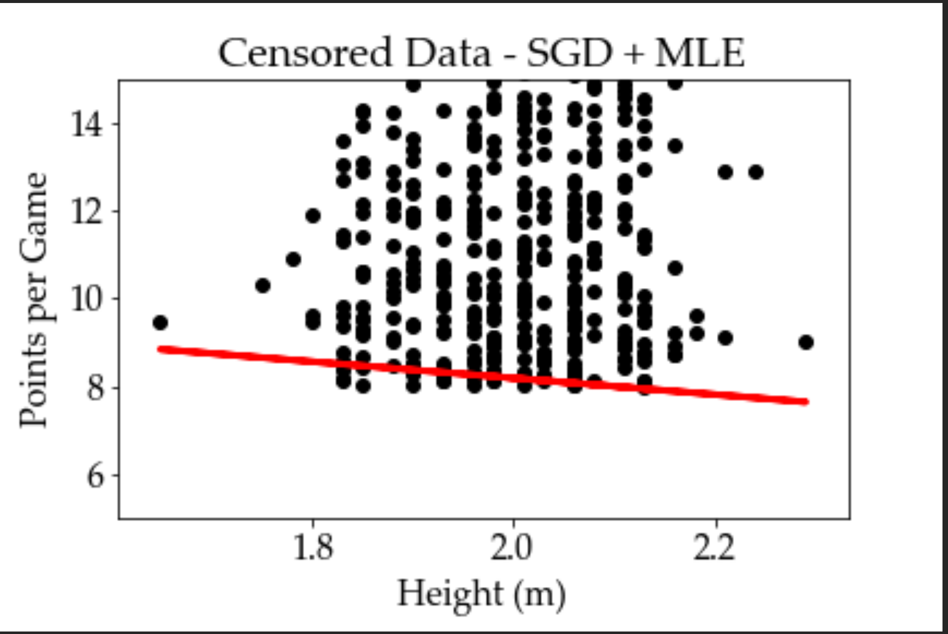
E.g. Application 2: NBA data

NBA player data after year 2000:

x_i: height of player i

y_i: number of points per game of player i

filtering : look at players with at least 8 points per game



Points per Game are **negatively correlated** with height!

Menu

- **Motivation**
- **Flavor of Models, Techniques, Results**
  - **Supervised learning: known truncation on the y-axis**
    - **small dive into techniques**
  - Supervised learning: unknown truncation on the x-axis
  - Unsupervised learning: learning truncated densities
    - bigger dive into techniques

Menu

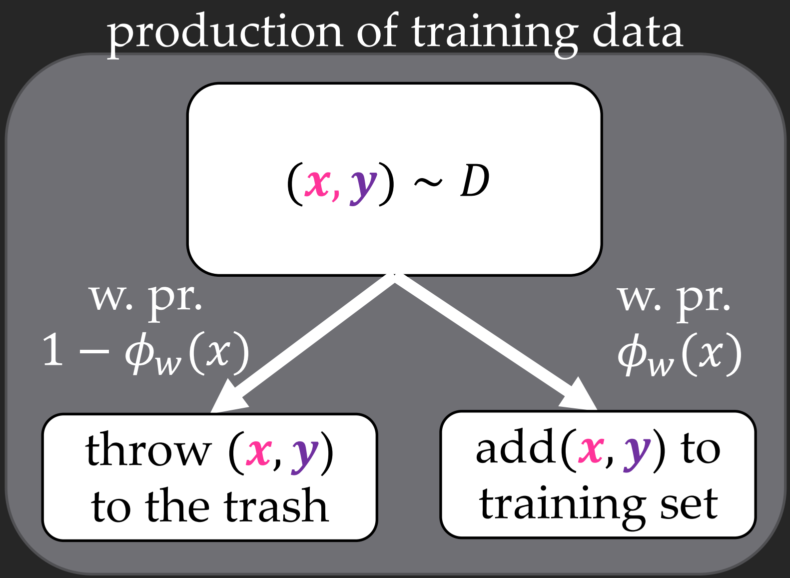
- **Motivation**
- **Flavor of Models, Techniques, Results**
  - **Supervised learning: known truncation on the y-axis**
    - **small dive into techniques**
  - **Supervised learning: unknown truncation on the x-axis**
  - Unsupervised learning: learning truncated densities
    - bigger dive into techniques

Problem 2: (Unknown) Truncation on the X-Axis

(recall: gender classification viz-a-viz skin tone)

Truncated Classification Model:

- (unknown) distribution D over uncensored image-label pairs $(x, y) \sim D$
- (unknown) filtering mechanism ϕ_w, $w \in \Omega$, s.t. (x, y) is included in train set with probability $\phi_w(x)$
- (sample access) unlabeled image distribution D_x i.e. big enough set of test images



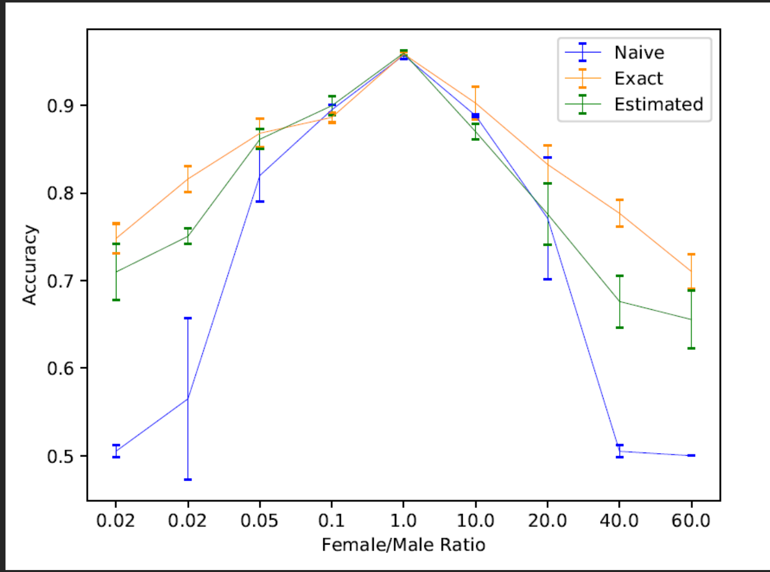
- **Goals:** given labeled but biased data $(x_i, y_i)_i$ and unbiased but unlabeled data,
  - find image-to-label classifier minimizing *classification loss on uncensored data*
- **Results:** practical, SGD-based likelihood optimization framework [w/ Kontonis,Tzamos,Zampetakis]
  - alternative to other domain adaptation approaches

Example Application: Gender Classification

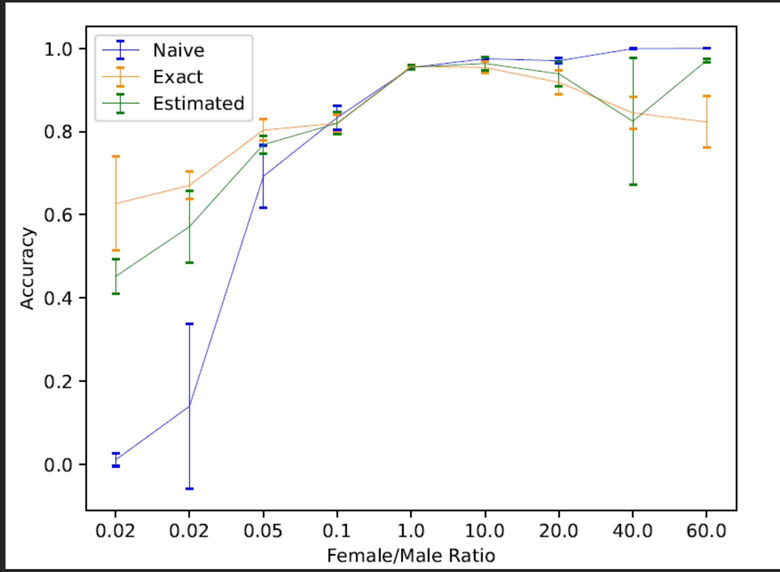
Train set: wildly gender-unbalanced subset of CelebA
Test set: gender-balanced subset of CelebA
No knowledge that train set was censored according to label variable

Compare: (i) Naïve classifier; (ii) our classifier; (iii) omniscient classifier (does accurate propensity scoring of train set – c.f. David Sontag's class)

Overall accuracy



Female accuracy



Example Application: Gender Classification 2

Train gender classifier on an adversarially constructed *gender-balanced* subset of CelebA, whi … d images” (images that a 95% accu … misclassifies)

(a) Males that an 95% accuracy AlexNet *incorrectly* classifies .

(b) Males that an 95% accuracy AlexNet *correctly* classifies .

(c) Females that an 95% accuracy AlexNet *incorrectly* classifies .

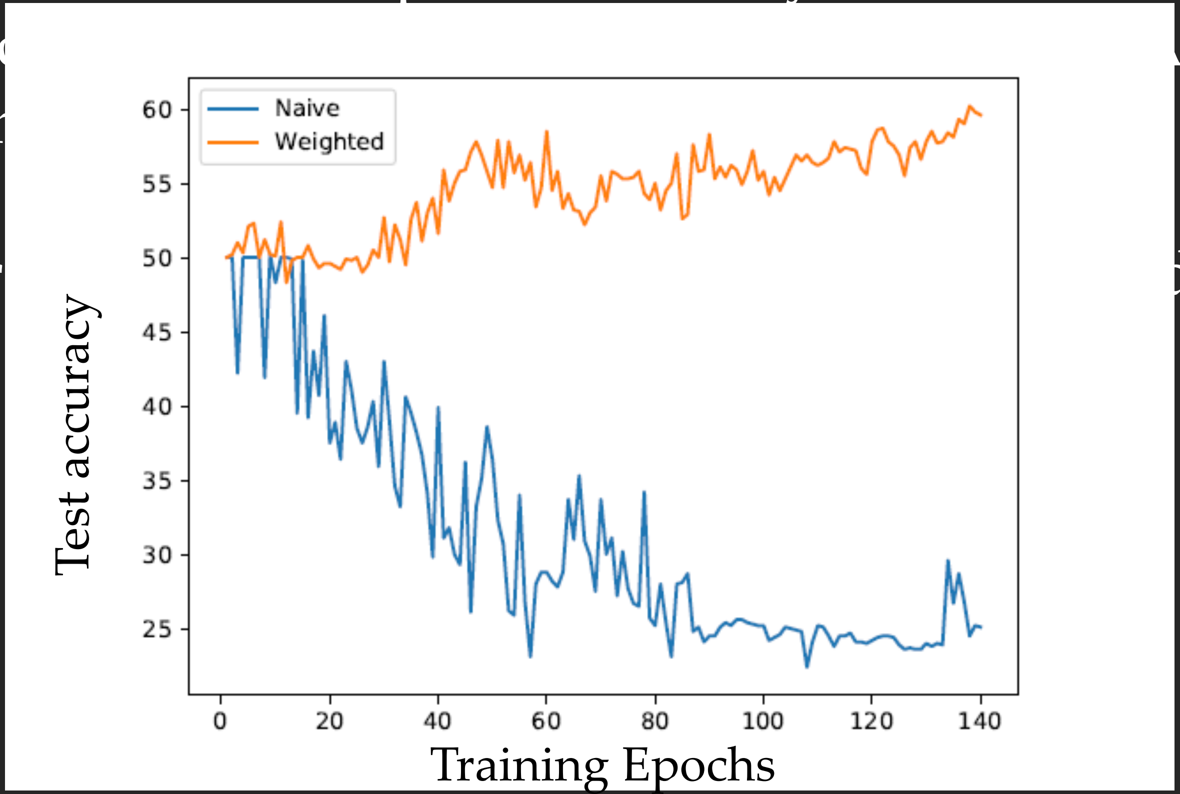
(d) Females that an 95% accuracy AlexNet *correctly* classifies .

Example Application: Gender Classification 2

Train gender classifier on an adversarially constructed *gender-balanced* subset of CelebA, which however predominantly contains “hard images” (images that a 95% ac misclassifies)

- use 1000 “h

Test classifier dataset



Menu

- **Motivation**
- **Flavor of Models, Techniques, Results**
    - **Supervised learning: known truncation on the y-axis**
        - **small dive into techniques**
    - **Supervised learning: unknown truncation on the x-axis**
    - Unsupervised learning: learning truncated densities
        - bigger dive into techniques

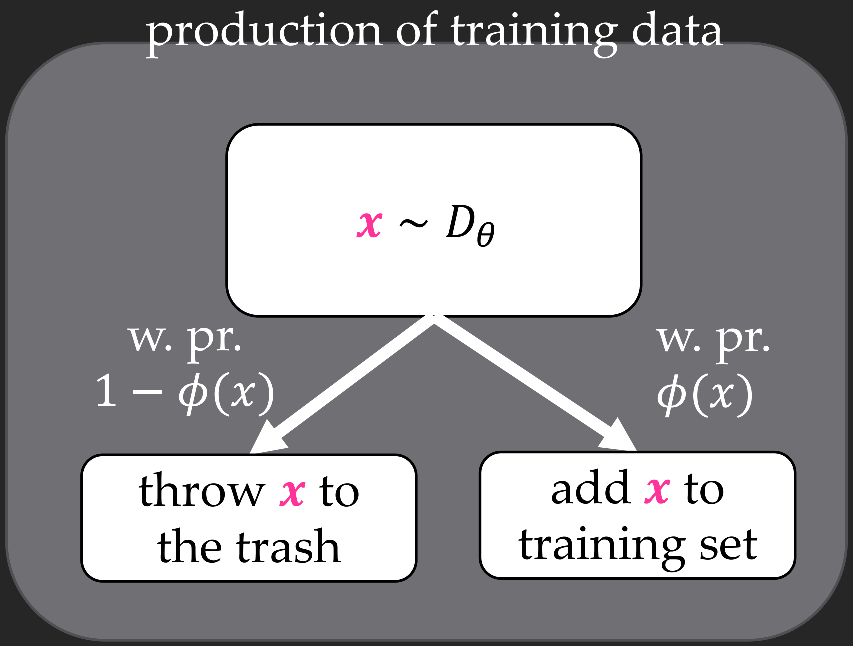
Menu

- **Motivation**
- **Flavor of Models, Techniques, Results**
  - **Supervised learning: known truncation on the y-axis**
    - **small dive into techniques**
  - **Supervised learning: unknown truncation on the x-axis**
  - **Unsupervised learning: learning truncated densities**
    - bigger dive into techniques

Problem 3: Truncated Density Estimation

Model:

- (*unknown*) parametric distribution D_θ over $\mathbb{R}^d$
  - uncensored data-points are vectors $x \sim D_\theta$
- (*known*) filtering mechanism $\phi: \mathbb{R}^d \to [0,1]$
  - x included in train set with probability $\phi(x)$



Goal: given filtered data $(x_i)_i$ recover θ

Results: practical SGD & MLE based framework **[w/ Ilyas, Zampetakis]**

- Fast rates + rigorous recovery of true parameters for Gaussians and other exponential families **[w/ Gouleakis, Tzamos, Zampetakis FOCS'18]**
- Unknown 0/1 filtering: **[Kontonis, Tzamos, Zampetakis FOCS'19]**

Comparison to Prior Work On Truncated Density Estimation

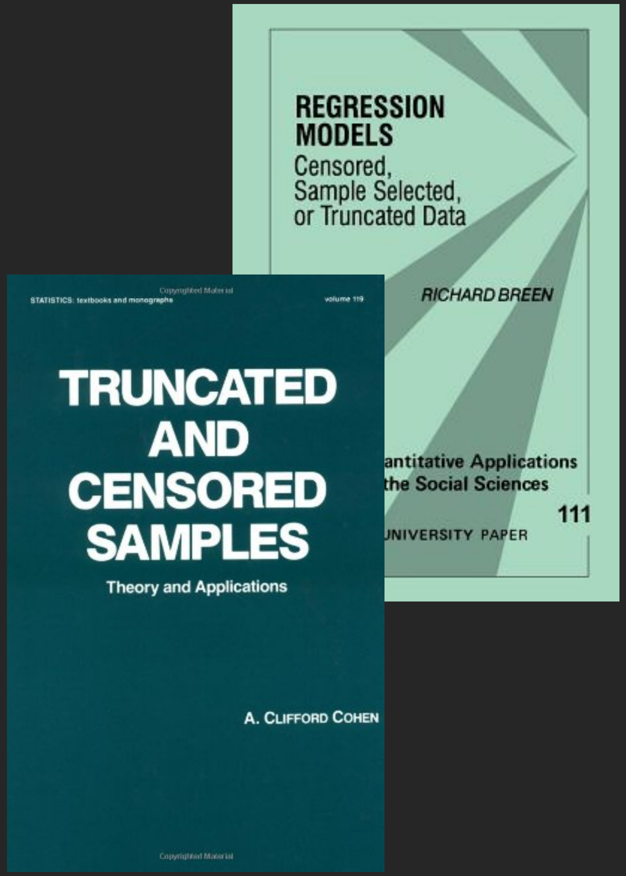
Learning Truncated/Censored Distributions

[Galton 1897], [Pearson 1902], [Pearson, Lee 1908], [Lee 1914], [Fisher 1931], [Hotelling 1948, [Tukey 1949],…,[Cohen'16]

Technical Bottlenecks:

- Convergence rates: $O_d\left(\frac{1}{\sqrt{n}}\right)$
- Computationally inefficient algorithms

Our work: optimal rates $O\left(\sqrt{\frac{\#\text{params}}{n}}\right)$, efficient algorithms, arbitrary truncation sets



Example Application: Query Optimization

Query Optimization:

- Given list of predicates/queries, $\{Q_1, Q_2, Q_3, \dots, Q_n\}$, want to efficient return all elements satisfying all predicates, i.e. $Q_1 \wedge Q_2 \wedge \cdots \wedge Q_n$

- More efficient to schedule the queries in order of selectivity
  - more selective = fewer elements satisfy it

- Need <u>cardinality estimates</u> (how selective is each query?)

Example Application: Query Optimization

- ***Cardinality Estimation:*** Given a query Q to a table, estimate the proportion of elements in the table satisfying the query

Height (cm)	Age	Weight (lbs)
150	16	120
155	18	140
153	15	137
160	19	150
165	18	155
163	17	200
…	…	…

e.g. Q: $\textit{height} < 162$, $\textit{age} > 17$, $\textit{weight} < 153$

Example Application: Query Optimization

- ***Cardinality Estimation:*** Given a query Q to a table, estimate the proportion of elements in the table satisfying the query

Height (cm)	Age	Weight (lbs)
150	16	120
155	18	140
153	15	137
160	19	150
165	18	155
163	17	200
…	…	…

e.g. Q: height < 162, *age* > 17, *weight* < 153

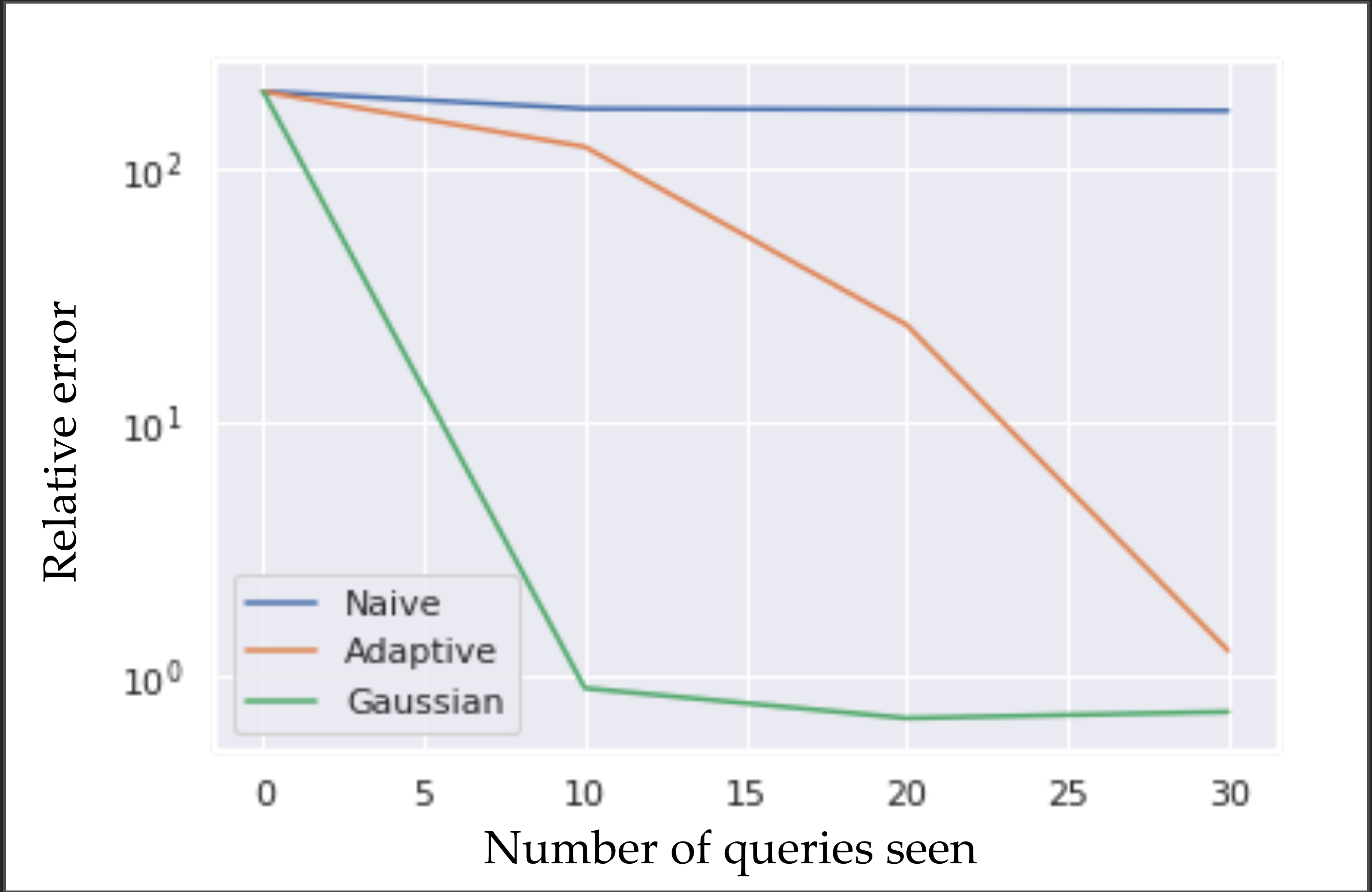
Selectivity: 33.33%

Example Application: Query Optimization

- ***Cardinality Estimation:*** Given a query Q to a table, estimate the proportion of elements in the table satisfying the query

- ***Our Approach:***
  - model database elements as sampled from a parametric distribution D_θ
  - use results of previously seen queries and truncated density estimation to learn θ, and use that to estimate cardinality of future queries

Results: Cardinality Estimation



Naïve: assume each element in DB samples its features (zip, model, year, etc.) independently from uniform distributions over corresponding ranges (no learning)

Adaptive: use query feedback to build a histogram over the possible elements

Gaussian: assume elements are sampled from a Gaussian mixture, learned using truncated density estimation

DMV dataset: estimating cardinality of random queries

Menu

- **Motivation**
- **Flavor of Models, Techniques, Results**
  - **Supervised learning: known truncation on the y-axis**
    - **small dive into techniques**
  - **Supervised learning: unknown truncation on the x-axis**
  - **Unsupervised learning: learning truncated densities**
    - **bigger dive into techniques**

Estimating a truncated Normal

[w/ Gouleakis, Tzamos, Zampetakis FOCS'18]

Truncated Normal

Let $C \subseteq \mathbb{R}^d$, we define the *truncated normal distribution* $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}, C)$ as

$$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}, C; \boldsymbol{x}) = \begin{cases} \frac{1}{\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}; C)} \cdot \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}; \boldsymbol{x}) & \boldsymbol{x} \in C \\ 0 & \boldsymbol{x} \notin C \end{cases}.$$

where:

$$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}; \boldsymbol{x}) = \frac{1}{\sqrt{2\pi \det(\boldsymbol{\Sigma})}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu})\right),$$

$$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}; C) = \int_C \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}; \boldsymbol{y}) d\boldsymbol{y}.$$

Estimation from Truncated Samples

Let $C \subseteq \mathbb{R}^d$, we define the *truncated normal distribution* $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}, C)$ as

$$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}, C; \boldsymbol{x}) = \begin{cases} \dfrac{1}{\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}; C)} \cdot \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}; \boldsymbol{x}) & \boldsymbol{x} \in C \\ 0 & \boldsymbol{x} \notin C \end{cases}.$$

Goal: Given samples from $\mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*, C)$ compute estimates $\hat{\boldsymbol{\mu}}$, $\hat{\boldsymbol{\Sigma}}$ such that

$$d_{TV}(\mathcal{N}(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}}), \mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*)) \leq \varepsilon.$$

Prior work and Comparison to ours

1. Only axes-aligned truncation
2. Known truncation
3. No sample complexity analysis
4. No efficient algorithm

Prior work and Comparison to ours

1. Only axes-aligned truncation
2. Known truncation
3. No sample complexity analysis
4. No efficient algorithm

The truncation set can be arbitrarily complex, as far as it has non-trivial Gaussian volume!

Prior work and Comparison to ours

1. Only axes-aligned truncation
2. Known truncation
3. No sample complexity analysis
4. No efficient algorithm

The truncation set can be arbitrarily complex, as far as it has non-trivial Gaussian volume!

Prior work and Comparison to ours

1. Only axes-aligned truncation
2. Known truncation
3. No sample complexity analysis
4. No efficient algorithm

We don't need full knowledge of the set but we only need **oracle access to it**!

Prior work and Comparison to ours

1. Only axes-aligned truncation
2. Known truncation
3. No sample complexity analysis
4. No efficient algorithm

We prove that the problem admits a **convex** programming formulation which can be solved efficiently!

Prior work and Comparison to ours

1. Only axes-aligned truncation
2. Known truncation
3. No sample complexity analysis
4. No efficient algorithm

We prove that the problem admits a **convex** programming formulation which can be solved efficiently!

1. Applies to general exponential families,

Prior work and Comparison to ours

1. Only axes-aligned truncation
2. Known truncation
3. No sample complexity analysis
4. No efficient algorithm

We prove that the problem admits a **convex** programming formulation which can be solved efficiently!

1. Applies to general exponential families,
2. yields very simple algorithms! (compared to moment methods)

Prior work and Comparison to ours

1. Only for axes aligned box!
2. Known set.
3. No sample complexity analysis.
4. No efficient algorithm.

We get # of samples that are **optimal** up to polylogarithmic factors.

Estimation from Truncated Samples

Theorem. *Let* $C \subseteq \mathbb{R}^d$ *that satisfies Assumptions 1 and 2. Given samples* $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$ *from* $\mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*, C)$ *we can efficiently find estimates* $\hat{\boldsymbol{\mu}}$ *and* $\hat{\boldsymbol{\Sigma}}$ *that satisfy with probability at least 99%:*

$$d_{TV}(\mathcal{N}(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}}), \mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*)) \leq \varepsilon$$

for $n = \tilde{\Theta}(d^2/\varepsilon^2)$.

Truncation Set

Assumption 1 (Constant Mass)

$$\mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*; C) \geq \alpha$$

Truncation Set

Assumption 1 (Constant Mass)

$$\mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*; C) \geq \alpha$$

As $\alpha \to 0$, required # of samples $\to \infty$.

Truncation Set

Assumption 1 (Constant Mass)

$$\mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*; C) \geq \alpha$$

Assumption 2 (Oracle Access)

Given $\boldsymbol{x} \in \mathbb{R}^d$ we can answer if $\boldsymbol{x} \in C$ or not.

Truncation Set

Assumption 1 (Constant Mass)

$$\mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*; C) \geq \alpha$$

Assumption 2 (Oracle Access)

Given $\boldsymbol{x} \in \mathbb{R}^d$ we can answer if $\boldsymbol{x} \in C$ or not.

Impossible for unknown C!

Truncation Set

Assumption 1 (Constant Mass)

$$\mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*; C) \geq \alpha$$

Assumption 2 (Oracle Access)

Given $\boldsymbol{x} \in \mathbb{R}^d$ we can answer if $\boldsymbol{x} \in C$ or not.

Estimation from Truncated Samples

Theorem. *Let* $C \subseteq \mathbb{R}^d$ *that satisfies Assumptions 1 and 2. Given samples* $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$ *from* $\mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*, C)$ *we can efficiently find estimates* $\hat{\boldsymbol{\mu}}$ *and* $\hat{\boldsymbol{\Sigma}}$ *that satisfy with probability at least 99%:*

$$d_{TV}(\mathcal{N}(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}}), \mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*)) \leq \varepsilon$$

for $n = \tilde{\Theta}(d^2/\varepsilon^2)$.

In the remaining of this talk: focus $\Sigma = I$

Theorem. *Let $C \subseteq \mathbb{R}^d$ that satisfies Assumptions 1 and 2. Given samples $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$ from $\mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{I}, C)$ we can find an estimate $\hat{\boldsymbol{\mu}}$ that satisfies with probability at least 99%:*

$$d_{TV}(\mathcal{N}(\hat{\boldsymbol{\mu}}, \boldsymbol{I}), \mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{I})) \leq \varepsilon$$

for $n = \tilde{\Theta}(d/\varepsilon^2)$.

The Estimation Algorithm

The Estimation Algorithm (untruncated)

Maximize Sample Likelihood

⬇

For $C = \mathbb{R}^d$ maximum likelihood estimator is

$$\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

⬇

Prove Consistency

The Estimation Algorithm (truncated)

Maximize Population Likelihood

The Estimation Algorithm

Pretend had access to
Infinitely many samples

(fantasy)

Maximize Population Likelihood

The Estimation Algorithm

Pretend had access to
Infinitely many samples

(fantasy) Maximize Population Likelihood ➡ Consistency

[Tukey 1949]

The Estimation Algorithm

(fantasy) Maximize Population Likelihood ➡ Consistency

Reality check: we don’t have access to infinitely many samples.

The Estimation Algorithm

(fantasy) Maximize Population Likelihood ➡ Consistency

⬇

(can still pretend) Stochastic Gradient Descent

The Estimation Algorithm

(fantasy) Maximize Population Likelihood $\Rightarrow$ Consistency

$\Downarrow$

(can still pretend) Stochastic Gradient Descent

$\Downarrow$

Proof of Convergence

The Estimation Algorithm

(fantasy) Maximize Population Likelihood → Consistency

↓

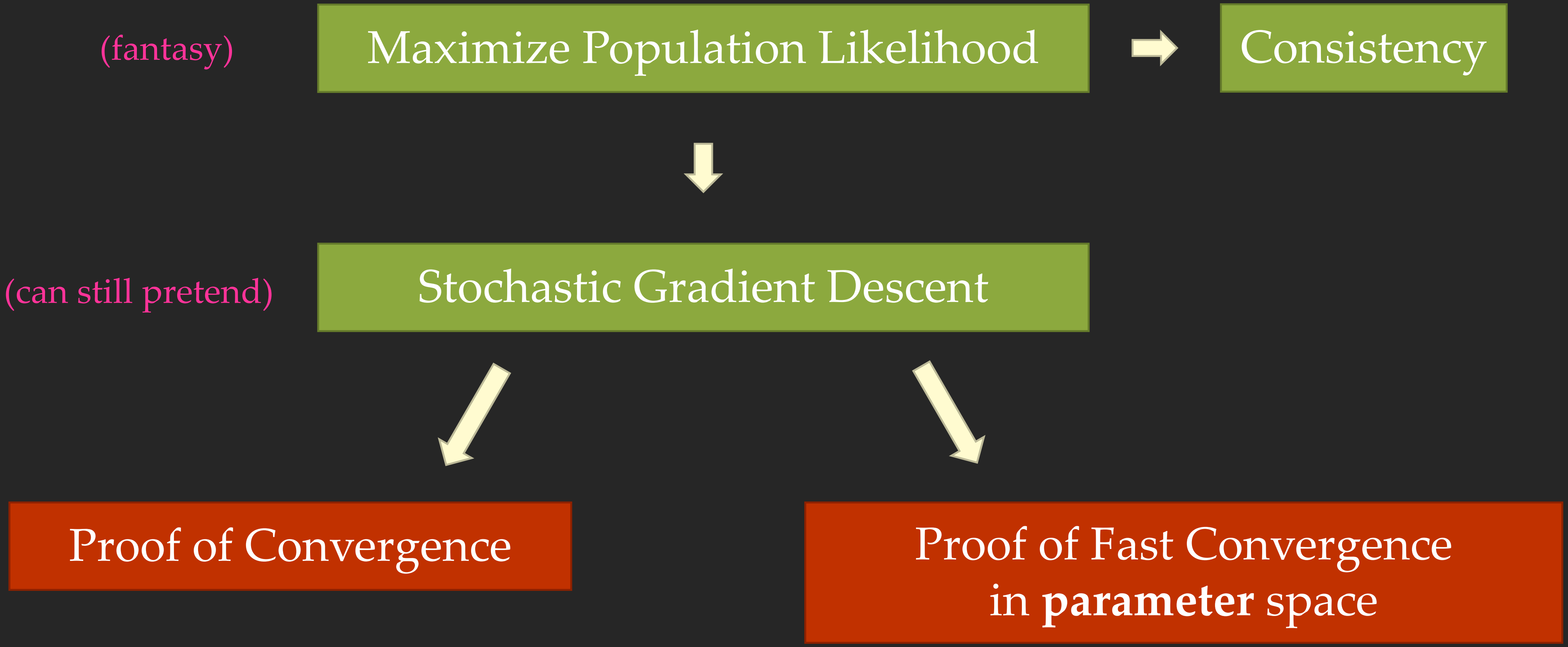
(can still pretend) Stochastic Gradient Descent

↙

Proof of Convergence

population likelihood is **convex**
(standard, holds for exponential family)

The Estimation Algorithm



The Estimation Algorithm

(fantasy) Maximize Population Likelihood → Consistency

(can still pretend) Stochastic Gradient Descent

Proof of Convergence

Proof of Fast Convergence in **parameter** space

population likelihood is **strongly convex**!

The Estimation Algorithm

1. estimate $\boldsymbol{\mu}^{(0)} = \frac{1}{n}\sum_{i=1}^{n} x_i$,
2. **for** $t = 1 \ldots T$ **do**
3. $\quad \boldsymbol{r} \leftarrow$ Sample $\mathcal{N}(\boldsymbol{\mu}^*, I, C)$,
4. $\quad \hat{\boldsymbol{r}} \leftarrow$ Sample $\mathcal{N}(\boldsymbol{\mu}^{(t-1)}, I, C)$
5. $\quad \boldsymbol{v}^{(t)} \leftarrow \boldsymbol{\mu}^{(t-1)} - \eta_t(\hat{\boldsymbol{r}} - \boldsymbol{r})$
6. $\quad \boldsymbol{\mu}^{(t)} \leftarrow$ project $\boldsymbol{v}^{(t)}$ to the ball $B = \left\{x \mid \left\|x - \boldsymbol{\mu}^{(0)}\right\| \leq R\right\}$
7. **output** $\boldsymbol{\mu}^{(T)}$

The Estimation Algorithm

1. estimate $\boldsymbol{\mu}^{(0)} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i$,
2. **for** $t = 1 \dots T$ **do**
3. $\quad \boldsymbol{r} \leftarrow$ Sample $\mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{I}, C)$,
4. $\quad \hat{\boldsymbol{r}} \leftarrow$ Sample $\mathcal{N}(\boldsymbol{\mu}^{(t-1)}, \boldsymbol{I}, C)$
5. $\quad \boldsymbol{v}^{(t)} \leftarrow \boldsymbol{\mu}^{(t-1)} - \eta_t(\hat{\boldsymbol{r}} - \boldsymbol{r})$
6. $\quad \boldsymbol{\mu}^{(t)} \leftarrow$ project $\boldsymbol{v}^{(t)}$ to the ball $B = \left\{x \mid \left\|x - \boldsymbol{\mu}^{(0)}\right\| \leq R\right\}$
7. **output** $\boldsymbol{\mu}^{(T)}$

The Estimation Algorithm

1. estimate $\mu^{(0)} = \frac{1}{n}\sum_{i=1}^{n} x_i$,
2. **for** $t = 1 \ldots T$ **do**
3. $\quad r \leftarrow$ Sample $\mathcal{N}(\mu^*, I, C)$,
4. $\quad \hat{r} \leftarrow$ Sample $\mathcal{N}(\mu^{(t-1)}, I, C)$
5. $\quad v^{(t)} \leftarrow \mu^{(t-1)} - \eta_t(\hat{r} - r)$
6. $\quad \mu^{(t)} \leftarrow$ project $v^{(t)}$ to the ball $B = \left\{x \mid \left\| x - \mu^{(0)} \right\| \leq R\right\}$
7. **output** $\mu^{(T)}$

The Estimation Algorithm

1. estimate $\mu^{(0)} = \frac{1}{n}\sum_{i=1}^{n} x_i$,
2. **for** $t = 1 \ldots T$ **do**
3. $\quad \boldsymbol{r} \leftarrow$ Sample $\mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{I}, C)$,
4. $\quad \hat{\boldsymbol{r}} \leftarrow$ Sample $\mathcal{N}(\boldsymbol{\mu}^{(t-1)}, \boldsymbol{I}, C)$
5. $\quad \boldsymbol{v}^{(t)} \leftarrow \boldsymbol{\mu}^{(t-1)} - \eta_t(\hat{\boldsymbol{r}} - \boldsymbol{r})$
6. $\quad \boldsymbol{\mu}^{(t)} \leftarrow$ project $\boldsymbol{v}^{(t)}$ to the ball $B = \left\{x \mid \left\|x - \mu^{(0)}\right\| \leq R\right\}$
7. **output** $\mu^{(T)}$

The Estimation Algorithm

1. estimate $\mu^{(0)} = \frac{1}{n}\sum_{i=1}^{n} x_i$,
2. **for** $t = 1 \ldots T$ **do**
3. $\quad \boldsymbol{r} \leftarrow$ Sample $\mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{I}, C)$,
4. $\quad \hat{\boldsymbol{r}} \leftarrow$ Sample $\mathcal{N}(\boldsymbol{\mu}^{(t-1)}, \boldsymbol{I}, C)$
5. $\quad \boldsymbol{v}^{(t)} \leftarrow \boldsymbol{\mu}^{(t-1)} - \eta_t(\hat{\boldsymbol{r}} - \boldsymbol{r})$
6. $\quad \mu^{(t)} \leftarrow$ project $v^{(t)}$ to the ball $B = \left\{x \mid \left\|x - \mu^{(0)}\right\| \leq R\right\}$
7. **output** $\mu^{(T)}$

The Estimation Algorithm

1. estimate $\boldsymbol{\mu}^{(0)} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i$,
2. **for** $t = 1 \dots T$ **do**
3. $\quad \boldsymbol{r} \leftarrow$ Sample $\mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{I}, C)$,
4. $\quad \hat{\boldsymbol{r}} \leftarrow$ Sample $\mathcal{N}(\boldsymbol{\mu}^{(t-1)}, \boldsymbol{I}, C)$
5. $\quad \boldsymbol{v}^{(t)} \leftarrow \boldsymbol{\mu}^{(t-1)} - \eta_t(\hat{\boldsymbol{r}} - \boldsymbol{r})$
6. $\quad \boldsymbol{\mu}^{(t)} \leftarrow$ project $\boldsymbol{v}^{(t)}$ to the ball $B = \left\{\boldsymbol{x} \mid \left\|\boldsymbol{x} - \boldsymbol{\mu}^{(0)}\right\| \leq R\right\}$
7. **output** $\boldsymbol{\mu}^{(T)}$

The Estimation Algorithm

1. estimate $\boldsymbol{\mu}^{(0)} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i$,
2. **for** $t = 1 \ldots T$ **do**
3. $\quad \boldsymbol{r} \leftarrow$ Sample $\mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{I}, C)$,
4. $\quad \hat{\boldsymbol{r}} \leftarrow$ Sample $\mathcal{N}(\boldsymbol{\mu}^{(t-1)}, \boldsymbol{I}, C)$
5. $\quad \boldsymbol{v}^{(t)} \leftarrow \boldsymbol{\mu}^{(t-1)} - \eta_t(\hat{\boldsymbol{r}} - \boldsymbol{r})$
6. $\quad \boldsymbol{\mu}^{(t)} \leftarrow$ project $\boldsymbol{v}^{(t)}$ to the ball $B = \left\{ \boldsymbol{x} \mid \left\| \boldsymbol{x} - \boldsymbol{\mu}^{(0)} \right\| \leq R \right\}$
7. **output** $\boldsymbol{\mu}^{(T)}$

The Estimation Algorithm

1. estimate $\boldsymbol{\mu}^{(0)} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i$,
2. **for** $t = 1 \ldots T$ **do**
3. $\quad \boldsymbol{r} \leftarrow$ Sample $\mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{I}, C)$,
4. $\quad \hat{\boldsymbol{r}} \leftarrow$ Sample $\mathcal{N}(\boldsymbol{\mu}^{(t-1)}, \boldsymbol{I}, C)$
5. $\quad \boldsymbol{v}^{(t)} \leftarrow \boldsymbol{\mu}^{(t-1)} - \eta_t(\hat{\boldsymbol{r}} - \boldsymbol{r})$
6. $\quad \boldsymbol{\mu}^{(t)} \leftarrow$ project $\boldsymbol{v}^{(t)}$ to the ball $B = \left\{\boldsymbol{x} \mid \left\|\boldsymbol{x} - \boldsymbol{\mu}^{(0)}\right\| \leq R\right\}$
7. **output** $\frac{1}{T+1}\sum_{t=0}^{T} \boldsymbol{\mu}^{(t)}$

Convergence Analysis

Convergence Analysis

Algorithm: Stochastic Gradient Descent in the population negative log-likelihood function.

Population Negative Log-Likelihood

$$\bar{\ell}(\boldsymbol{\mu}) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{N}(\boldsymbol{\mu}^*,I,C)}\left[\frac{1}{2}\boldsymbol{x}^T\boldsymbol{x} - \boldsymbol{x}^T\boldsymbol{\mu}\right] + \log\left(\int_C \exp\left(-\frac{1}{2}\boldsymbol{z}^T\boldsymbol{z} + \boldsymbol{z}^T\boldsymbol{\mu}\right) d\boldsymbol{z}\right)$$

Gradient of Population Negative Log-Likelihood

$$\nabla\bar{\ell}(\boldsymbol{\mu}) = \mathbb{E}_{\hat{r}\sim\mathcal{N}(\boldsymbol{\mu},I,C),\;\; r\sim\mathcal{N}(\boldsymbol{\mu}^*,I,C)}\left[\hat{r} - r\right]$$

Hessian of Population Negative Log-Likelihood

$$\mathcal{H}_{\bar{\ell}}(\boldsymbol{\mu}) = \mathrm{cov}_{\boldsymbol{x}\sim\mathcal{N}(\boldsymbol{\mu},I,C)}\left(\boldsymbol{x},\boldsymbol{x}\right)$$

Convergence Analysis

Conditions for Fast Convergence of SGD

1. $\bar{\ell}(\boldsymbol{\mu})$ is *strongly* convex,
2. $\mathbb{E}\left[\|\hat{\boldsymbol{r}} - \boldsymbol{r}\|_2^2\right]$ is bounded, where recall $\mathbb{E}\left[\hat{\boldsymbol{r}} - \boldsymbol{r}\right] = \nabla\bar{\ell}(\boldsymbol{\mu})$

Convergence Analysis

Conditions for Fast Convergence of SGD

1. $\bar{\ell}(\boldsymbol{\mu})$ is *strongly* convex,
2. $\mathbb{E}\left[\|\hat{\boldsymbol{r}} - \boldsymbol{r}\|_2^2\right]$ is bounded, where recall $\mathbb{E}\left[\hat{\boldsymbol{r}} - \boldsymbol{r}\right] = \nabla \bar{\ell}(\boldsymbol{\mu})$

Convergence Analysis

Is $\bar{\ell}(\boldsymbol{\mu})$ strongly convex?

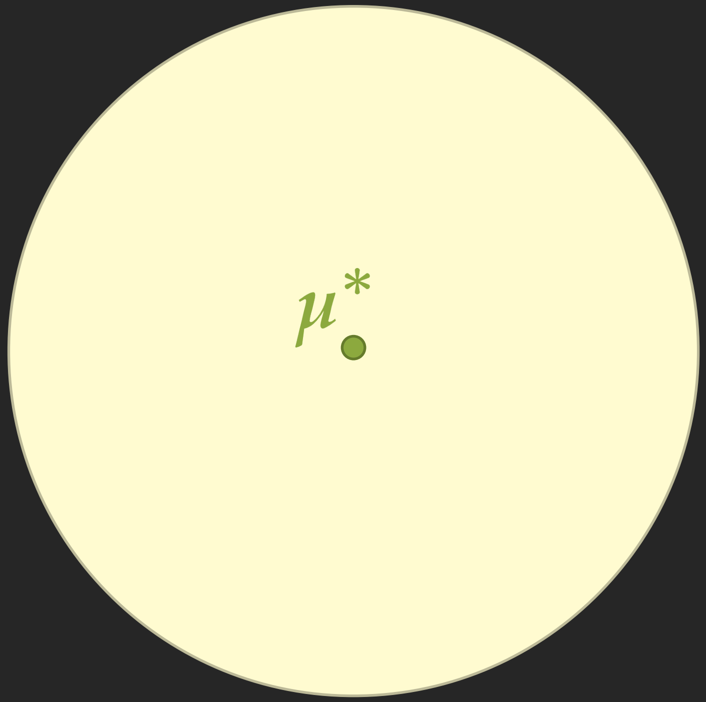
Not for all $\boldsymbol{\mu} \in \mathbb{R}^d$.

μ^*

Convergence Analysis

Is $\bar{\ell}(\boldsymbol{\mu})$ strongly convex?

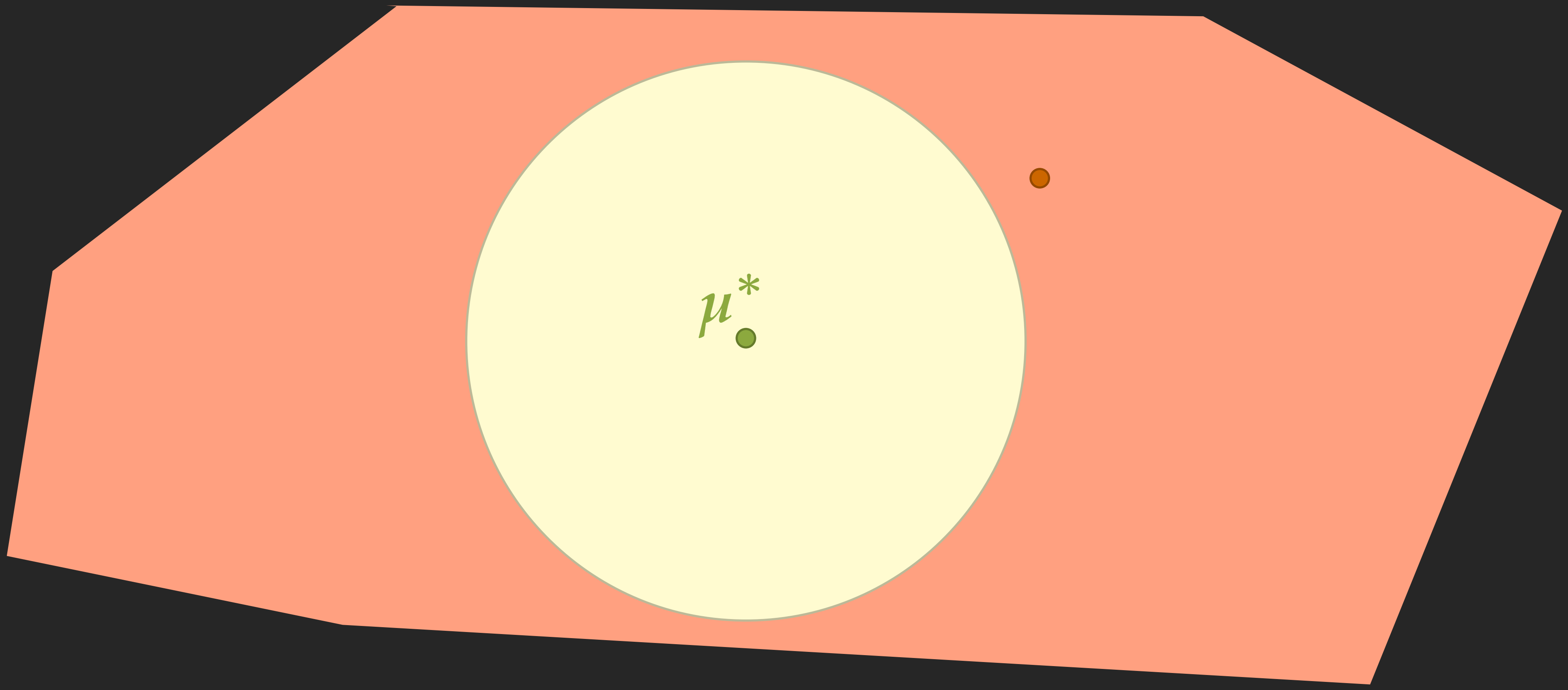
Not for all $\boldsymbol{\mu} \in \mathbb{R}^d$.



Convergence Analysis

Is $\bar{\ell}(\boldsymbol{\mu})$ strongly convex?

Not for all $\boldsymbol{\mu} \in \mathbb{R}^d$.

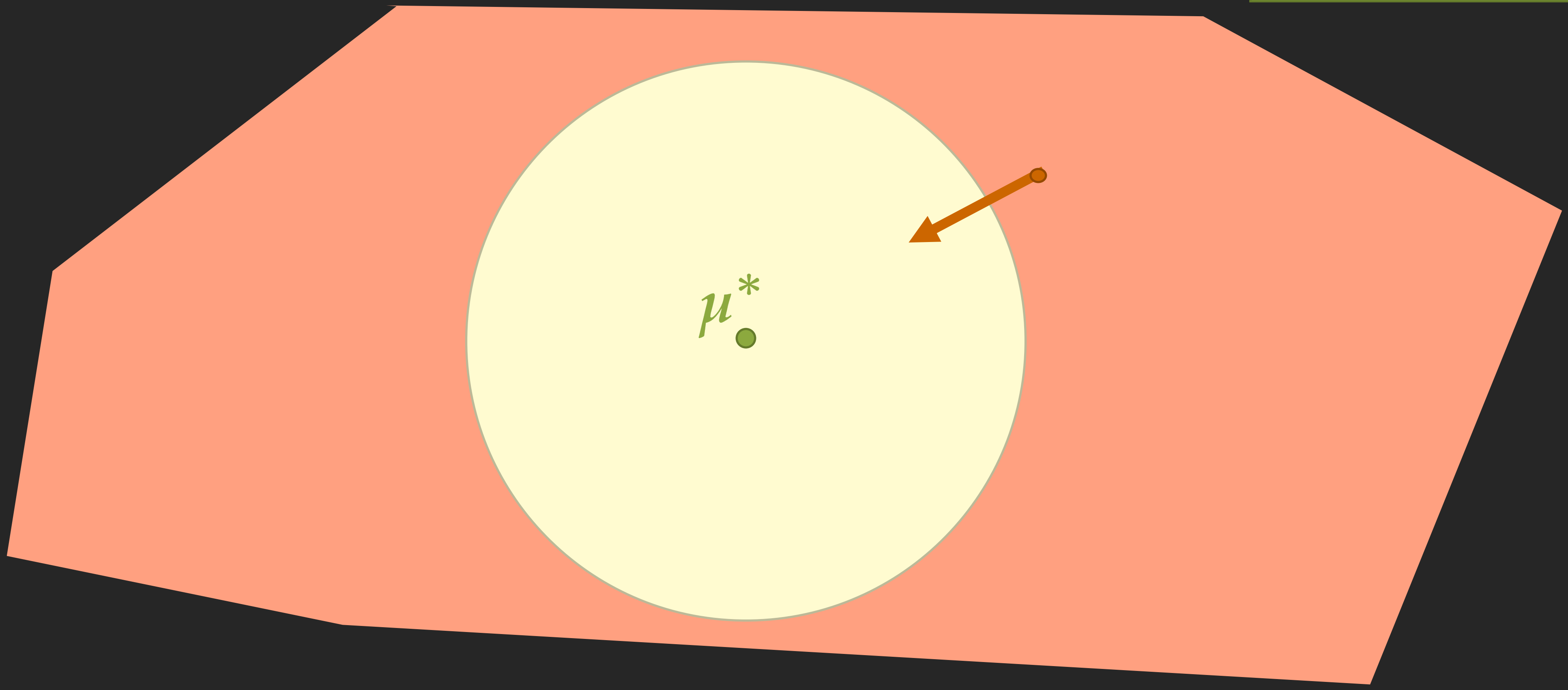


Convergence Analysis

Is $\bar{\ell}(\boldsymbol{\mu})$ strongly convex?

Not for all $\boldsymbol{\mu} \in \mathbb{R}^d$.

Project to a convex region where it is



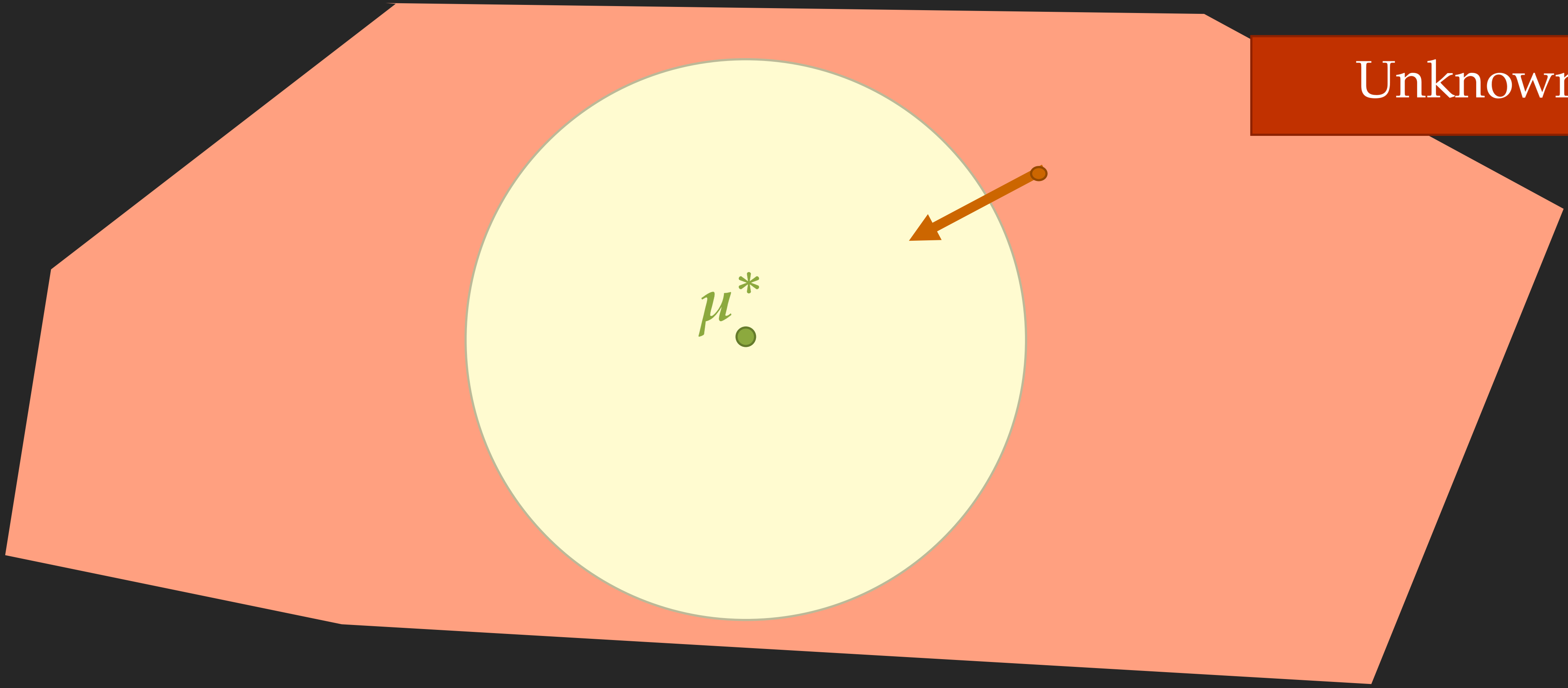
Convergence Analysis

Is $\bar{\ell}(\boldsymbol{\mu})$ strongly convex?

Not for all $\boldsymbol{\mu} \in \mathbb{R}^d$.

Project to a convex region where it is

Unknown $\boldsymbol{\mu}^*$!



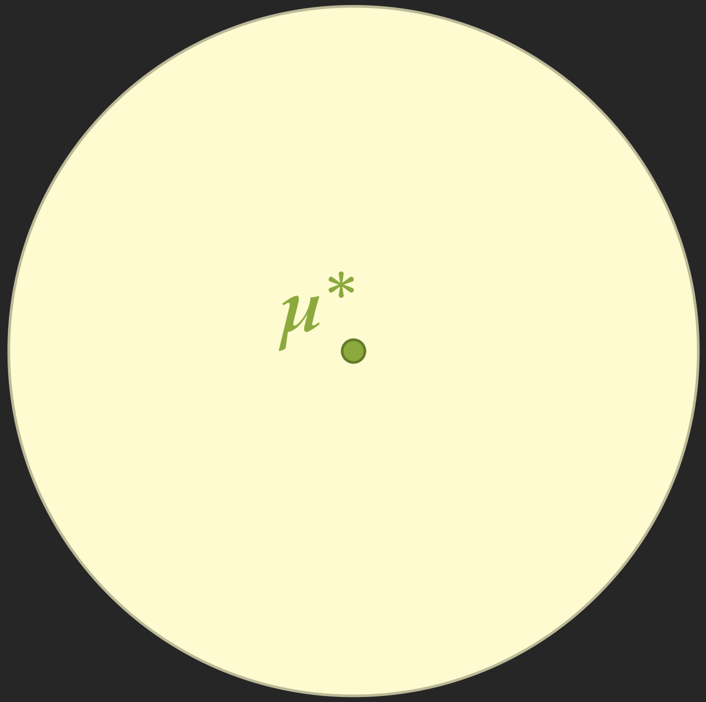
Convergence Analysis

Is $\bar{\ell}(\boldsymbol{\mu})$ strongly convex?

Not for all $\boldsymbol{\mu} \in \mathbb{R}^d$.

Project to a convex region where it is

Unknown $\boldsymbol{\mu}^*$!



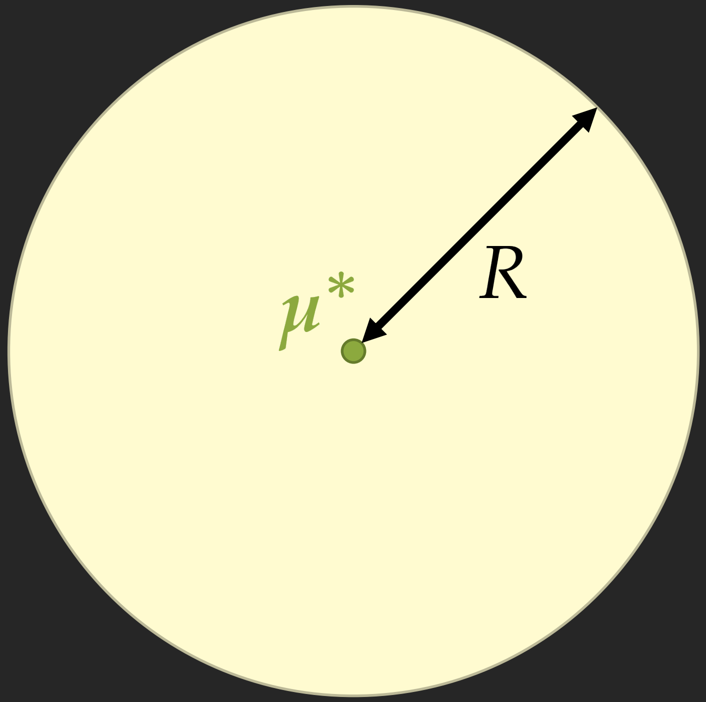
Convergence Analysis

Is $\bar{\ell}(\boldsymbol{\mu})$ strongly convex?

Not for all $\boldsymbol{\mu} \in \mathbb{R}^d$.

Project to a convex region where it is

Unknown $\boldsymbol{\mu}^*$!



Convergence Analysis

Is $\bar{\ell}(\boldsymbol{\mu})$ strongly convex?

Not for all $\boldsymbol{\mu} \in \mathbb{R}^d$.

Lemma.

$$\boldsymbol{\mu}^{(0)} = \frac{1}{T}\sum_{i=1}^{T} \boldsymbol{x}_i,$$

$$\left\| \boldsymbol{\mu}^{(0)} - \boldsymbol{\mu}^* \right\| \leq \frac{R}{4}.$$

Project to a convex region where it is

Unknown $\boldsymbol{\mu}^*$!

Project around $\boldsymbol{\mu}^{(0)}$!

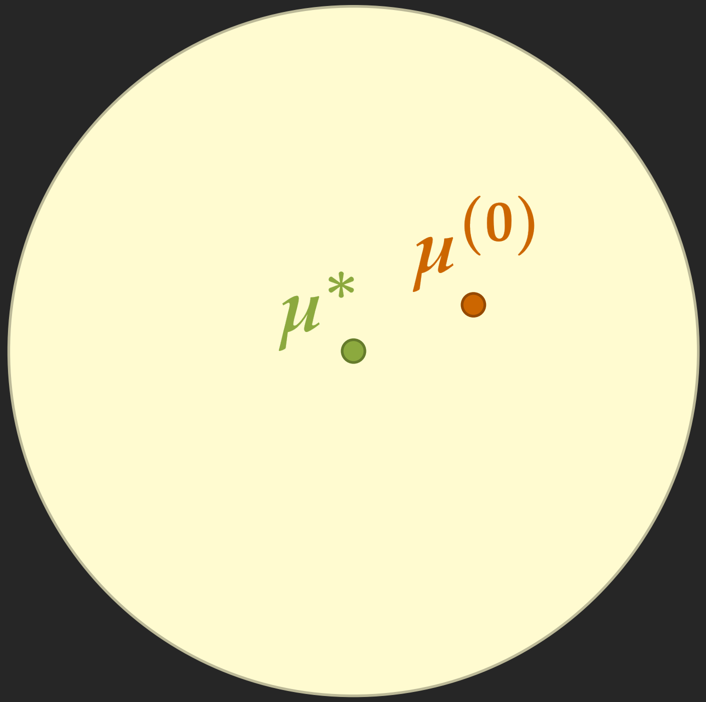
Convergence Analysis

Is $\bar{\ell}(\boldsymbol{\mu})$ strongly convex?

Not for all $\boldsymbol{\mu} \in \mathbb{R}^d$.

Project to a convex region where it is

Unknown $\boldsymbol{\mu}^*$!



Convergence Analysis

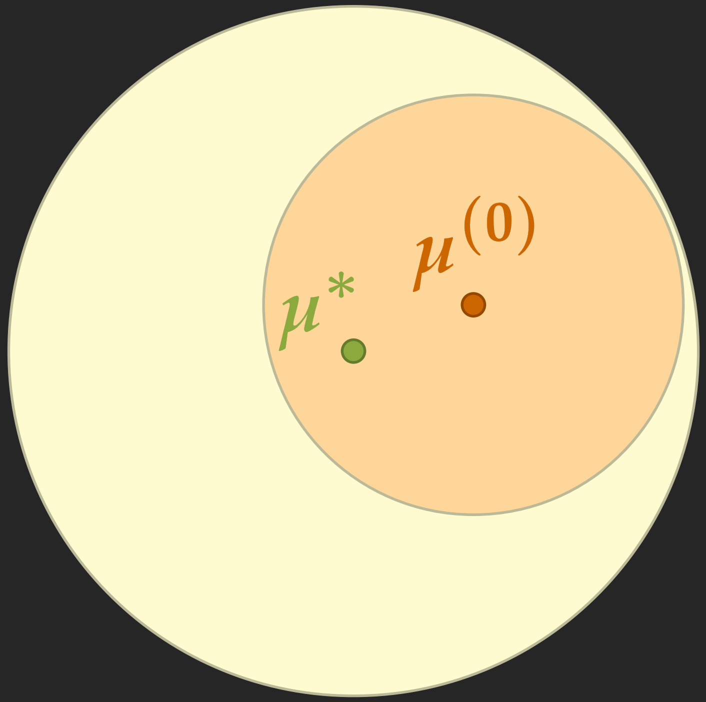
Is $\bar{\ell}(\boldsymbol{\mu})$ strongly convex?

Not for all $\boldsymbol{\mu} \in \mathbb{R}^d$.

Project to a convex region where it is

Unknown $\boldsymbol{\mu}^*$!

Project around $\boldsymbol{\mu}^{(0)}$!



Convergence Analysis

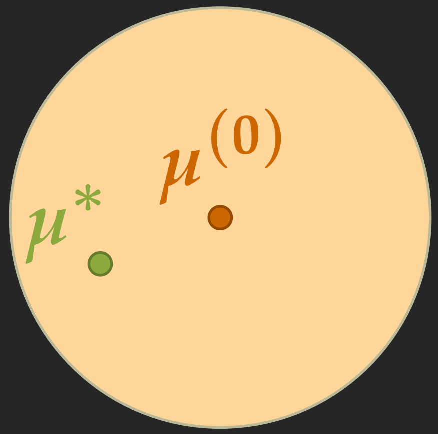
Is $\bar{\ell}(\boldsymbol{\mu})$ strongly convex?

Not for all $\boldsymbol{\mu} \in \mathbb{R}^d$.

Project to a convex region where it is

Unknown $\boldsymbol{\mu}^*$!

Project around $\boldsymbol{\mu}^{(0)}$!



The Estimation Algorithm

1. estimate $\boldsymbol{\mu}^{(0)} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_i$,
2. **for** $t = 1 \ldots T$ **do**
3. $\quad \boldsymbol{r} \leftarrow$ Sample $\mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{I}, C)$,
4. $\quad \hat{\boldsymbol{r}} \leftarrow$ Sample $\mathcal{N}(\boldsymbol{\mu}^{(t-1)}, \boldsymbol{I}, C)$
5. $\quad \boldsymbol{v}^{(t)} \leftarrow \boldsymbol{\mu}^{(t-1)} - \eta_t(\hat{\boldsymbol{r}} - \boldsymbol{r})$
6. $\quad \boxed{\boldsymbol{\mu}^{(t)} \leftarrow \text{ project } \boldsymbol{v}^{(t)} \text{ to the ball } B = \left\{\boldsymbol{x} \mid \left\|\boldsymbol{x} - \boldsymbol{\mu}^{(0)}\right\| \leq R\right\}}$
7. **output** $\frac{1}{T+1} \sum_{t=0}^{T} \boldsymbol{\mu}^{(t)}$

Convergence Analysis

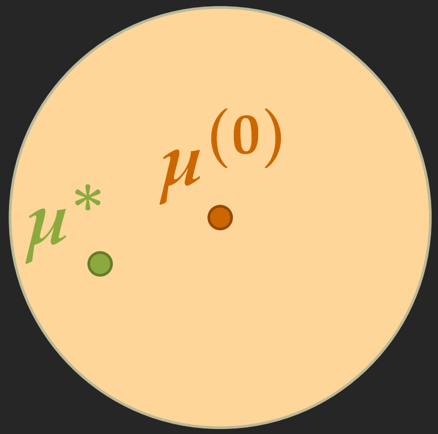
Is $\bar{\ell}(\boldsymbol{\mu})$ strongly convex?

Not for all $\boldsymbol{\mu} \in \mathbb{R}^d$.

Project to a convex region where it is

Unknown $\boldsymbol{\mu}^*$!

Project around $\boldsymbol{\mu}^{(0)}$!



How to show strong convexity around μ^*?

Convergence Analysis

How to show strong convexity around μ^*?

Convergence Analysis

How to show strong convexity around μ^*?

Hessian of Population Log-Likelihood

$$\mathcal{H}_{\bar{\ell}}(\boldsymbol{\mu}) = \operatorname{cov}_{\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{I}, C)} (\boldsymbol{x}, \boldsymbol{x})$$

Convergence Analysis

How to show strong convexity around μ^*?

Hessian of Population Log-Likelihood

$$\mathcal{H}_{\bar{\ell}}(\boldsymbol{\mu}) = \operatorname{cov}_{\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{I}, C)} (\boldsymbol{x}, \boldsymbol{x}) \succeq \lambda \boldsymbol{I}$$

Convergence Analysis

How to show strong convexity around μ^*?

Hessian of Population Log-Likelihood

$$\mathcal{H}_{\bar{\ell}}(\boldsymbol{\mu}) = \operatorname{cov}_{\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{I}, C)} (\boldsymbol{x}, \boldsymbol{x}) \succeq \lambda \boldsymbol{I}$$

High variance in every direction.

Convergence Analysis

How to show strong convexity around μ^*?

Hessian of Population Log-Likelihood

$$\mathcal{H}_{\bar{\ell}}(\boldsymbol{\mu}) = \mathrm{cov}_{\boldsymbol{x}\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{I},C)}(\boldsymbol{x},\boldsymbol{x}) \succeq \lambda \boldsymbol{I}$$

High variance in every direction.

Proof idea:
we use anti-concentration of polynomials under the Gaussian measure.

Menu

- Motivation
- Flavor of Models, Techniques, Results
  - Supervised learning: known truncation on the y-axis
    - small dive into techniques
  - Supervised learning: unknown truncation on the x-axis
  - Unsupervised learning: learning truncated densities
    - bigger dive into techniques
- Conclusions

Summary

- **Missing Observations**

  ⇒ **train set dist'n ≠ test set distribution**

  ⇒ **prediction bias (a.k.a. "AI bias")**

- **Our Work:** decrease bias, by developing machine learning methods more robust to **censored and truncated samples**
- **General Framework:** SGD on Population Log-Likelihood
- **End-to-end guarantees:** optimal rates and efficient algorithms for truncated Gaussian estimation, and truncated linear/logistic/probit regression
- **Many Open Problems:**
  - make fewer parametric assumptions; identifiability?
  - Statistical inference problem X – (untruncated samples) + (truncated samples)
  - Bias in ML applications

The End

Unknown Covariance Matrix

Additional difficulties

- Log-likelihood is not convex.

Unknown Covariance Matrix

Additional difficulties

- Log-likelihood is not convex.

We have to reparametrize.

Unknown Covariance Matrix

Additional difficulties

- Log-likelihood is not convex.
- The strong convexity set is not a ball.

Unknown Covariance Matrix

Additional difficulties

- Log-likelihood is not convex.
- The strong convexity set is not a ball.
  - Difficult to prove that the initial estimators lie in the set.

Unknown Covariance Matrix

Additional difficulties

- Log-likelihood is not convex.
- The strong convexity set is not a ball.
  - Difficult to prove that the initial estimators lie in the set.
  - Difficult to project to this set.

Unknown Covariance Matrix

Additional difficulties

- Log-likelihood is not convex.
- The strong convexity set is not a ball.
  - Difficult to prove that the initial estimators lie in the set.
  - Difficult to project to this set. We have to find a convex subset.

Unknown Covariance Matrix

Additional difficulties

- Log-likelihood is not convex.
- The strong convexity set is not a ball.
    - Difficult to prove that the initial estimators lie in the set.
    - Difficult to project to this set.
- Anti-concentration of degree 4 polynomials is needed.