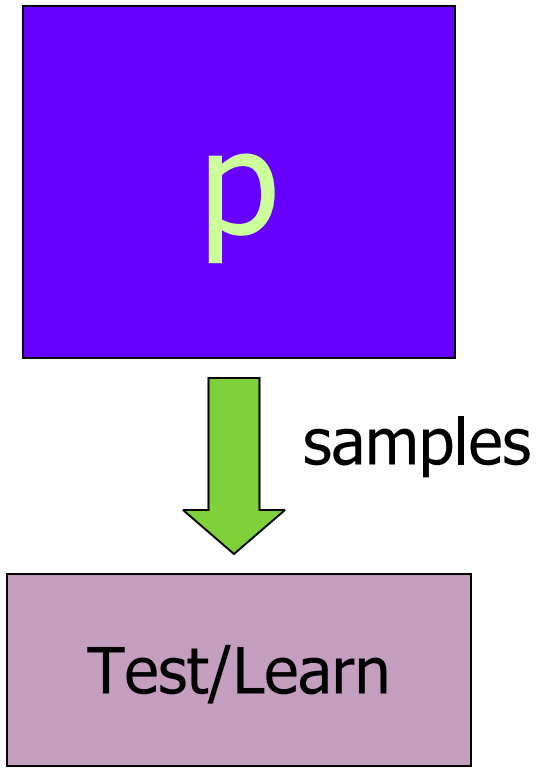
“Classy” sample correctors[1]

Ronitt Rubinfeld
MIT and Tel Aviv University

joint work with Clement Canonne (Columbia) and Themis Gouleakis (MIT)

[1]thanks to Clement and G for inspiring this classy title

Our usual model:



What if your samples aren't *quite* right?

What are the traffic patterns?

Some sensors lost power, others went crazy!

Astronomical data

A meteor shower confused some of the measurements

Teen drug addiction recovery rates

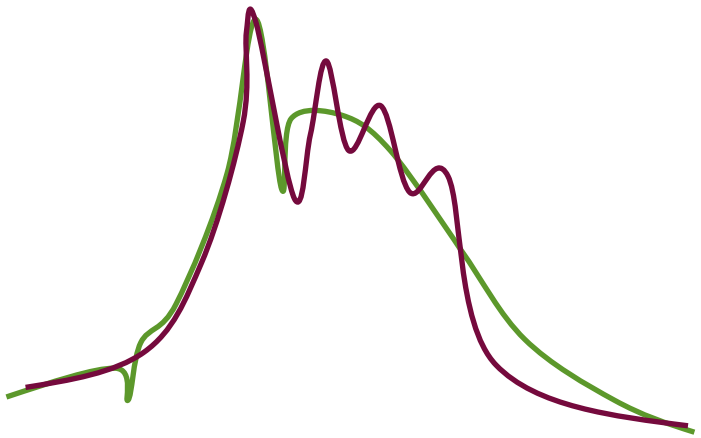
Never received data from three of the community centers!

Whooping cranes

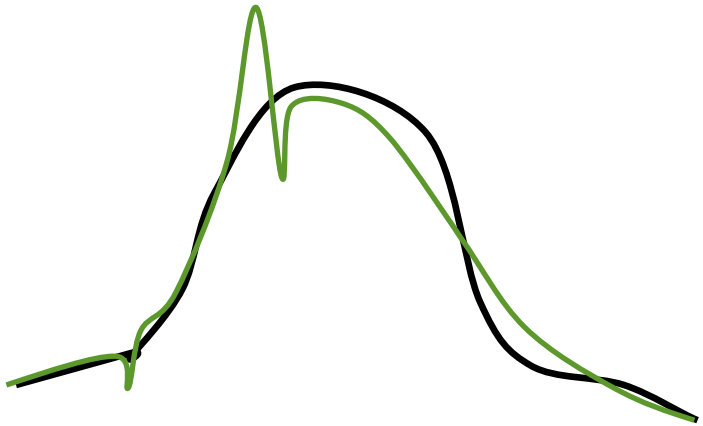
Correction of location errors for presence-only species distribution models

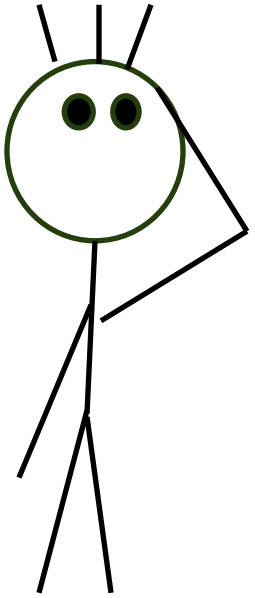
[Hefley, Baasch, Tyre, Blankenship 2013]

What is correct?



What is correct?





What to do?

- Outlier detection/removal
- Imputation
- Missingness
- Robust statistics
- …

What if don't know that the distribution (and even noise) is normal, Gaussian, …?

Weaker assumption?

A suggestion for a methodology

What is correct?

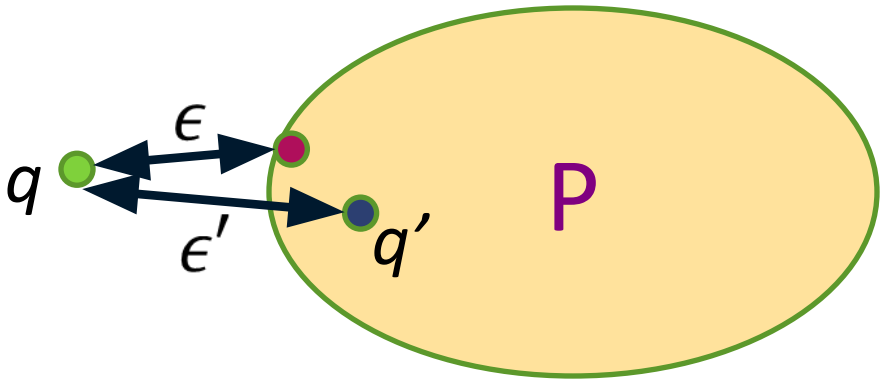
Sample corrector assumes that original distribution in *class P*

(e.g., *P* is class of Lipshitz, monotone, *k*-modal, or *k*-histogram distributions)

Classy Sample Correctors

- Given: Samples of distribution q assumed to be ϵ-close to class P

- Output: Samples of some q' such that
  - q' is ϵ'-close to original distribution q
  - q' in P



Classy Sample Correctors

- Given: Samples of distribution q assumed to be ϵ-close to class P

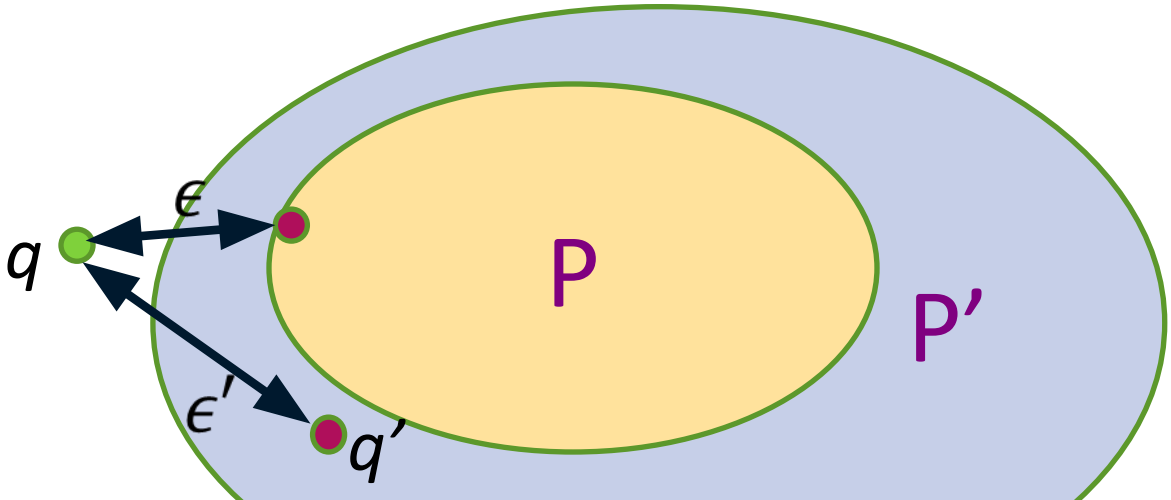
- Output: Samples of some q' such that
  - q' is ϵ'-close to original distribution q
  - q' in P

1. *Sample* complexity per output sample of q'?
2. *Randomness* complexity per output sample of q'?

Classy "non-Proper" Sample Correctors

- Given: Samples of distribution q assumed to be ϵ-close to class P
- Output: Samples of some q' such that
  - q' is ϵ'-close to distribution q
  - q' in P'

In our example $P \subseteq P'$ and P' not too much bigger than P



A very simple (nonproper) example

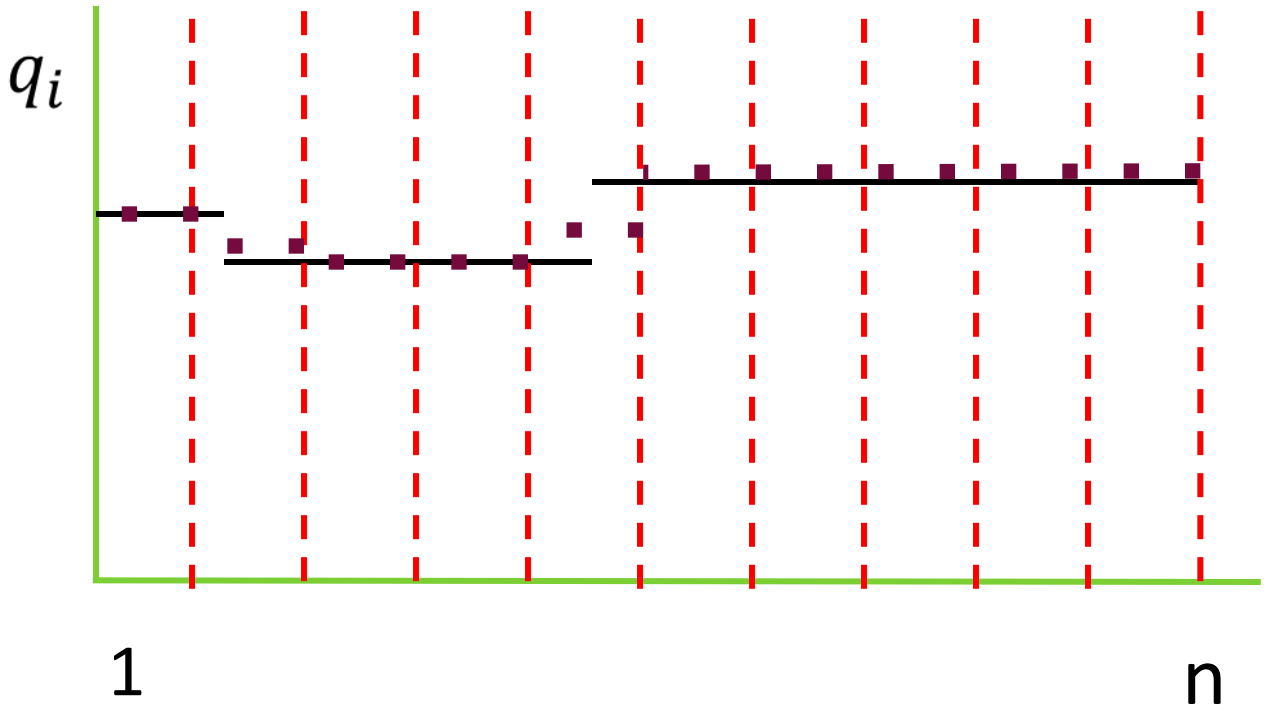
$P_{k,c}$ distributions: k-histograms, $||q||_\infty < c/n$

- (non-proper) Sample-corrector:
  - Input: samples of q , ϵ-close to $P_{k,c}$
  - Output: samples of q' in $P_{k/\epsilon,c}$

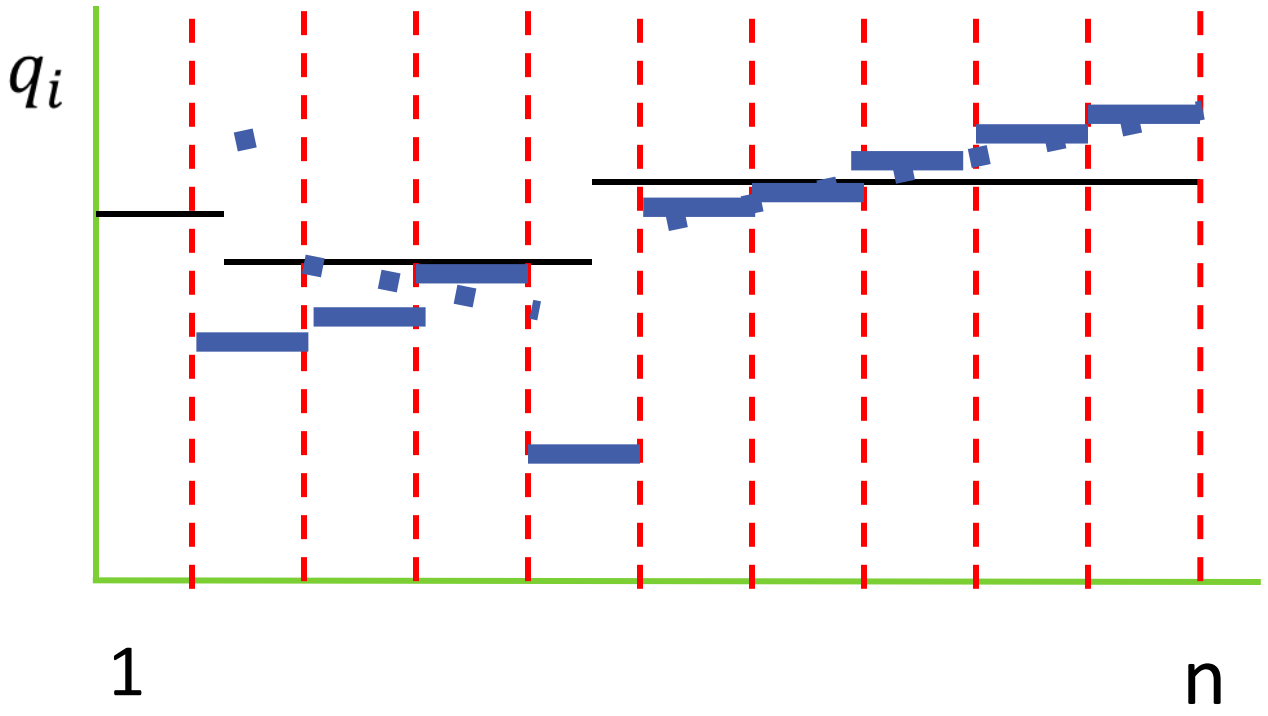
- Algorithm: (O(1) sample, O(log n) randomness)
  - Partition domain into k/ϵ equal sized partitions
  - Given sample x from q, output uniform element of x's partition

Why is q'in $P_{k/\epsilon,c}$? close to q?

k-histogram distribution



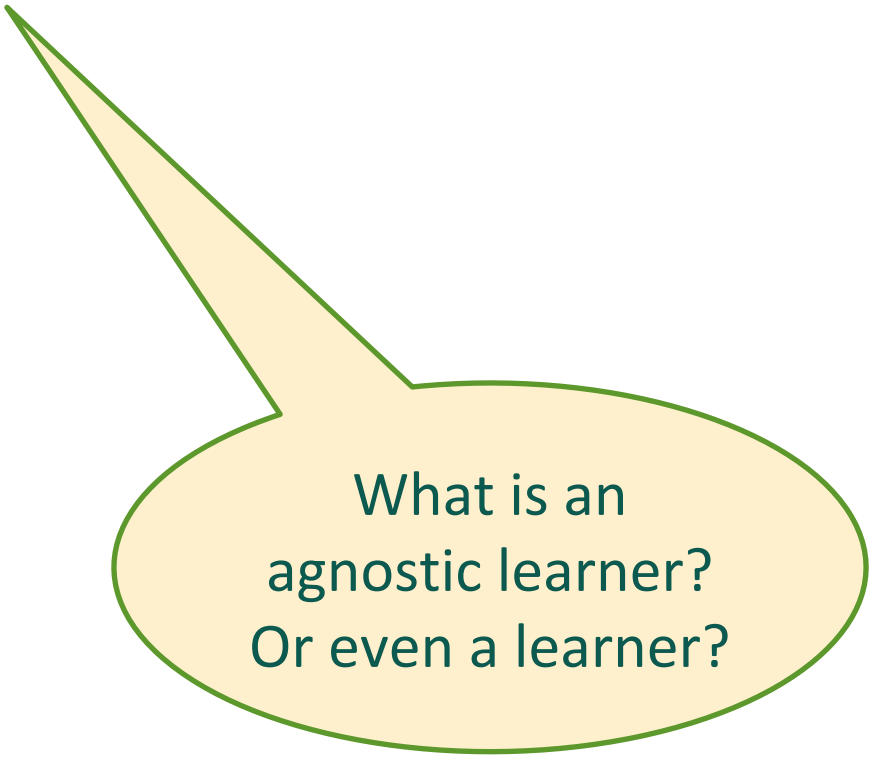
Close to k-histogram distribution



A generic way to get a sample corrector:

An observation

Agnostic learner → Sample corrector



What is a ``classy'' learner?

- Learning distributions for class *P* *(lots of definitions, see [Kearns Mansour Ron R. Schapire Sellie], [Dasgupta],…)*:
  - Get samples of D (promised to belong to class of distributions *P)*
  - Output representation of hypothesis $\widehat{D}$ such that $\left|\left|D - \widehat{D}\right|\right|_1 \leq \epsilon$

What is sample complexity in terms of ϵ, n?

What is a ``classy'' agnostic learner?

- Learning distributions via class P :
  - Get samples of D (NOT promised to belong to class of distributions P)
  - Output hypothesis $\widehat{D}$ such that

$$\left|\left|D - \widehat{D}\right|\right|_1 \leq f(\epsilon_{opt})$$

What is sample complexity in terms of ϵ, n?

Can be HARDER than regular learning

An observation

Agnostic learner  Sample corrector

Corollaries: Sample correctors for

- monotone distributions
- histogram distributions
- histogram distributions under promises (e.g., distribution is MHR or monotone)

Learning monotone distributions

- Learning monotone distributions requires $\theta(\frac{1}{poly(\epsilon)} \log n)$ samples

[Birge][Daskalakis Diakonikolas Servedio]

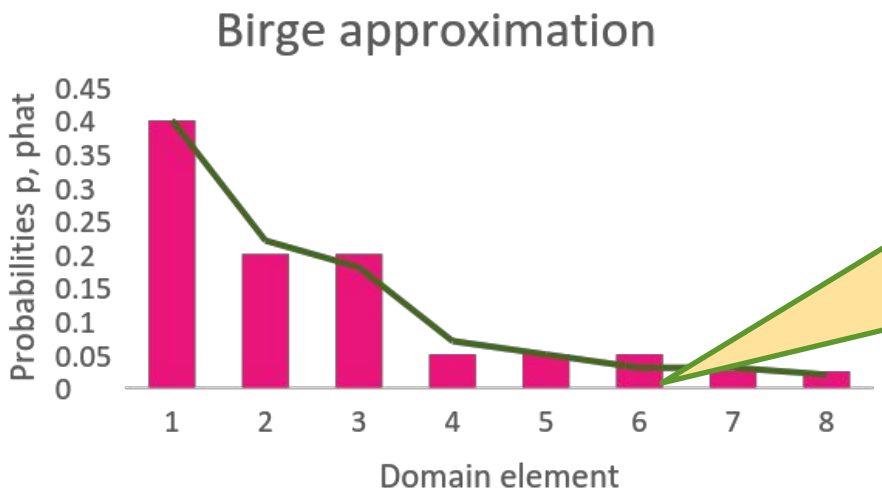
Birge Buckets

- Partition of domain into buckets (segments) of size $(1+\epsilon)^i$

$(O(\frac{1}{\epsilon}\log n)$ buckets total)

For distribution p, let $\hat{p}$ be uniform on each bucket, same marginal in each bucket

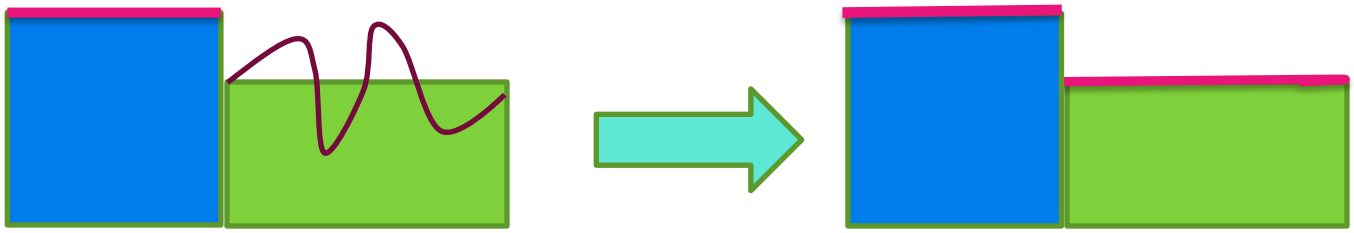
Thm: If p monotone, then $\left|\left|p-\hat{p}\right|\right| \leq \epsilon$



You know the boundaries! Enough to learn the marginals of each bucket

A very special kind of error

- Suppose ALL error located internally to Birge Buckets



Then, easy to correct to $\hat{p}$:

1. Pick sample *x* from *p*
2. Output y chosen UNIFORMLY from *x*'s Birge Bucket

"Birge Bucket Correction"

The big open question:

When can sample correctors be *more* efficient than agnostic learners?

Some answers for monotone distributions:

- Error is REALLY small
- Have access to powerful queries
- Missing consecutive data errors
- Unfortunately, not likely in general case (constant arbitrary error, no extra queries) [P. Valiant]

Learning monotone distributions

- Thm: Exists Sample Corrector which given p which is $\left(\frac{1}{\log^2 n}\right)$ −close to monotone, uses $O(1)$ samples of p per output sample.

OBLIVIOUS CORRECTION!!

Proof Idea:
Mix Birge Bucket correction with slightly decreasing distribution (flat on buckets with some space between buckets)

A lower bound [P. Valiant]

- Sample correctors for $\Omega\left(\frac{1}{\log(n)}\right)$-close to monotone distributions require $\Omega(\log n)$ samples

Open: Can we handle error $o\left(\frac{1}{\log(n)}\right)$?

What about stronger queries?

What if have lots and lots of *sorted samples*?

Easy to implement both samples, and queries to cumulative distribution function (cdf)!

Thm: Exists Sample Corrector using $O((\log(n))^{1/2})$ *cdf+sample* queries per output sample.

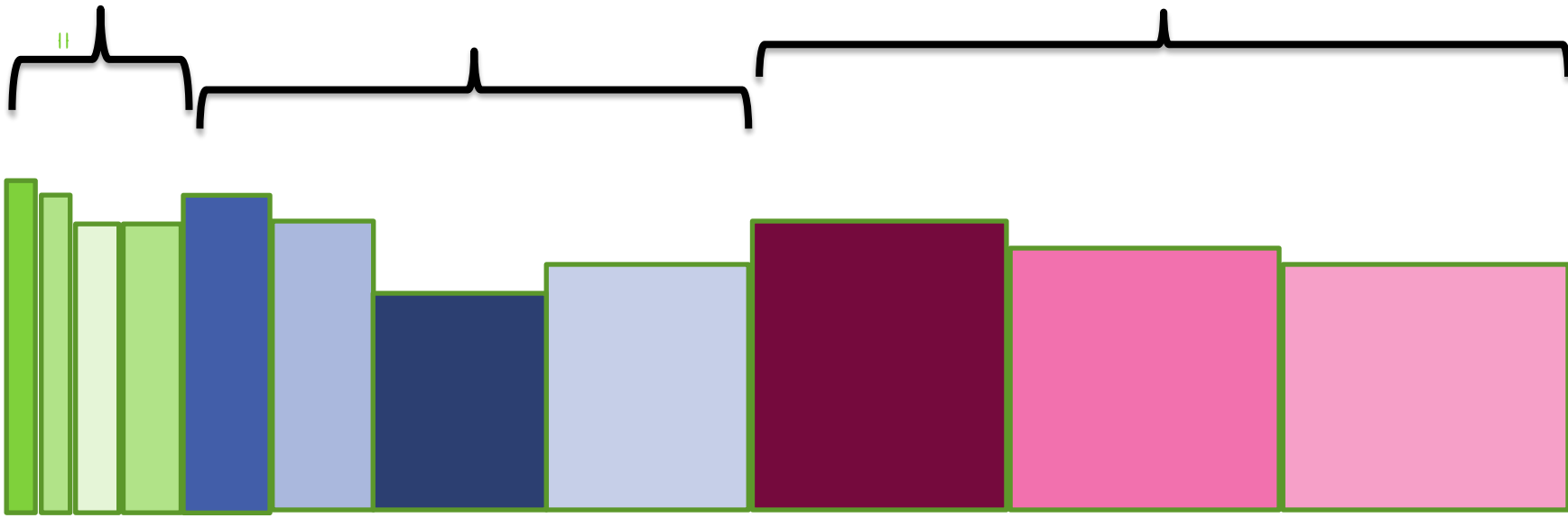
First step

Use Birge bucketing to reduce p to an *O(log n)*-histogram distribution

Fixing with CDF queries

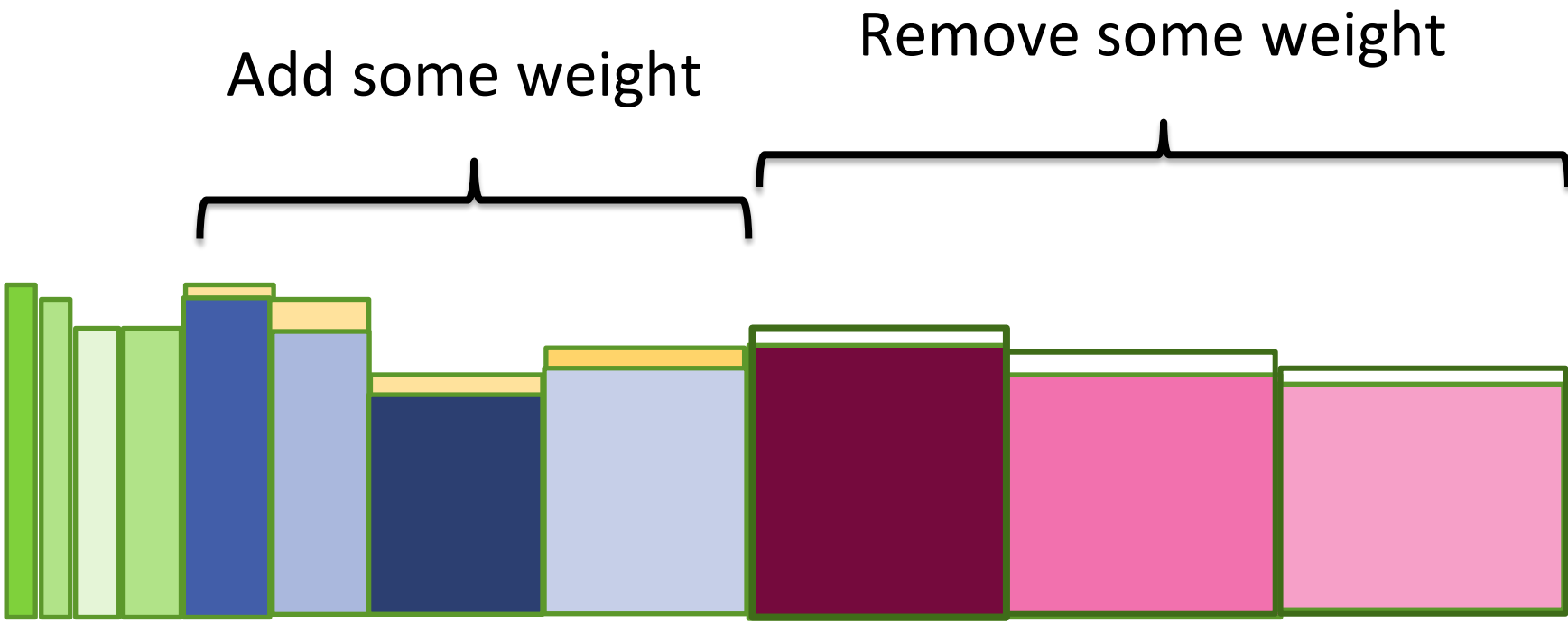
- Each *super bucket* is $\sqrt{\log n}$ consecutive Birge buckets
- Query conditional distribution of superbuckets and reweight if needed

superbuckets



Fixing with CDF queries

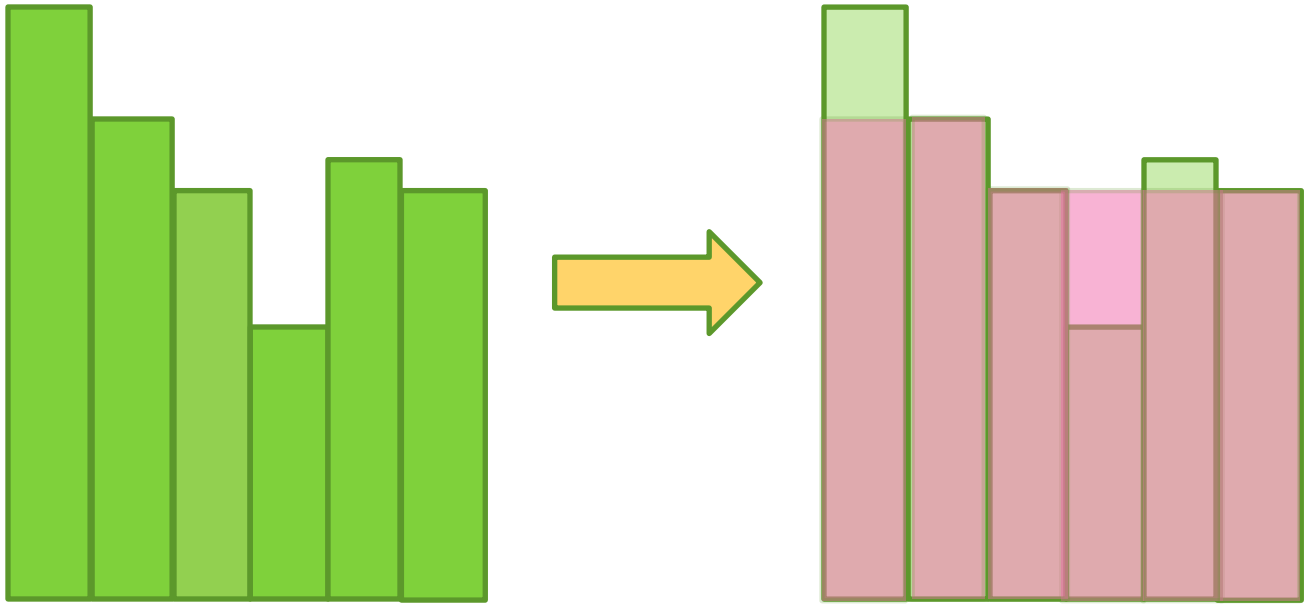
- Each *super bucket* is $\sqrt{\log n}$ consecutive Birge buckets
- Query conditional distribution of superbuckets and reweight if needed (decide how using LP)



Fixing with CDF queries

- Each *super bucket* is $\sqrt{\log n}$ consecutive Birge buckets
- Query conditional distribution of superbuckets and reweight if needed
- Within super buckets, use O($\sqrt{\log n}$) queries to all buckets in current, previous and next super buckets in order to "fix" inside
  - Fix must be done *quickly* and *on the fly*…
    - Monotone within superbucket
    - Don't violate monotonicity with neighbor superbuckets

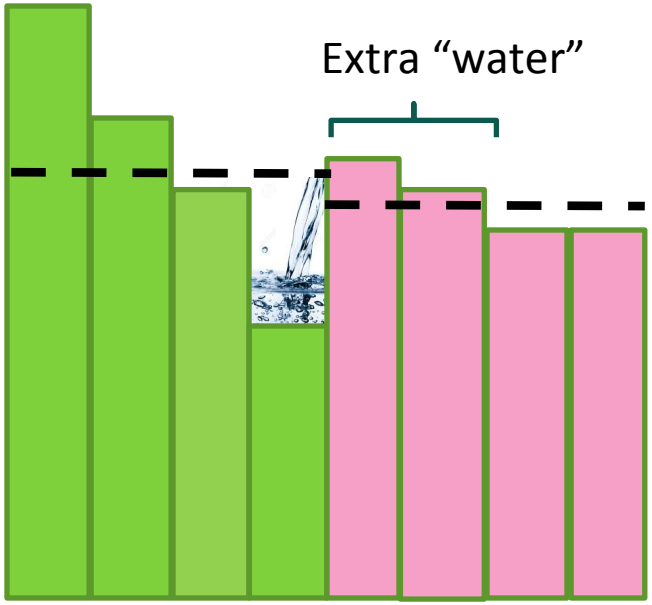
Reweighting within a superbucket



$\sqrt{\log n}$ CDF queries

Minimize L1 distance via LP

“Water pouring” to fix superbucket boundaries



Could it cascade arbitrarily far?

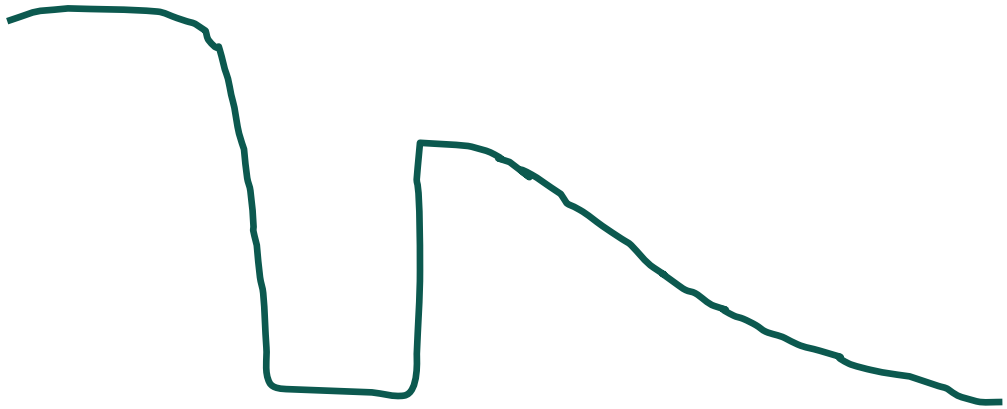
What if there is not enough pink water?

What if there is too much pink water?

Special error classes

- Missing data segment errors – *p* is a member of *P* with a segment of the domain removed
  - E.g. power failure for a whole block in traffic data

More efficient sample correctors via "learning" missing part



Sample correctors provide power!

Sample correctors provide more powerful learners:

-
- Sample Corrector + regular learner → agnostic learner (for low error distributions)
- Why? To agnostically learn q
  - Corrector: q close to P → q' in P and close to q
  - Learner: learns q'' close to q' (and so, close to q)

Sample correctors provide more powerful property testers:

- Tester for class P: Given ϵ and samples of q
  - If q in P, tester PASSES
  - If q is ϵ −far from any distribution in P, tester FAILS

- Tolerant tester for P: Given $\epsilon < \epsilon'$, and samples of q
  - If q ϵ −close from some distribution in P, tester PASSES
  - If q is ϵ' −far from any distribution in P, tester FAILS

Often much harder

Sample correctors provide more powerful testers:

-

- Sample Corrector + distance approximator + tester → tolerant tester

Sample correctors provide more powerful testers:

- Estimates distance between two distributions

- Sample Corrector + distance approximator + tester → tolerant tester

  Gives weakly tolerant monotonicity tester

Proof: Modifying Brakerski's idea to get tolerant tester

- Use sample corrector on p to output p'
- Test that p' in D
- Ensure that p' close to p using distance approximator

If p close to D, then p' close to p and in D

If p not close to D, we know *nothing* about p':
(1) may not be in D
(2) may not be close to p

Randomness Scarcity

- Can we correct using little randomness of our own?
  - Note that agnostic learning method relies on using our own random source
  - Compare to extractors (not the same)

Randomness Scarcity

- Can we correct using little randomness of our own?
  - Generalization of Von Neumann corrector of biased coin
  - For monotone distributions, YES!

Randomness scarcity: a simple case

- Correcting to uniform distribution
  - Output convolution of a few samples

In conclusion…

Yet another new model!

What next for correction?

What classes can we correct?

What next for correction?

When is correction easier than agnostic learning?

When is correction easier than (non-agnostic) learning?

How good is the corrected data?

- Estimating averages of survey/experimental data
- Learning

Thank you