Algorithmic Approaches to Preventing Overfitting in Adaptive Data Analysis

Part 1

Aaron Roth

Penn
UNIVERSITY of PENNSYLVANIA

The 2015 ImageNet competition

- An image classification competition during a heated war for deep learning talent amongst big tech companies.
- Training set of 1.5 million images, to be classified into 1,000 different categories.
    - E.g. “frilled lizard”, “banded gecko”, “reflex camera”, “osciliscope”
- Held out validation set of 100,000 images.
    - Competitors could submit models twice per week to check performance on validation set.

The 2015 ImageNet competition

- In March, Baidu announced it had achieved record accuracy, beating Google.
  - Posted a paper: “Deep Image: Scaling up Image Recognition”
  - Team lead: *“our company is now leading the race in computer intelligence…We have great power in our hands—much greater than our competitors.”*
  - 4.82% error -> 4.58% error
- But they had cheated!
  - Registered 30 fake accounts to circumvent the 2 validations per week rule.
  - Upon discovery, they were banned from the competition, the paper was withdrawn, and team lead was fired.
  - Why did this help and how can we prevent it?

The Multiple Comparisons Problem (and Uniform Convergence)

- Suppose we have a classifier $f: X \to A$, a dataset $S \sim P^n$, and a loss function $\ell(\hat{y}, y) = \delta(y \neq \hat{y})$.
- We want to know the true loss of our classifier:

$$L(f) = \mathbb{E}_{(x,y)\sim P}[\ell(f(x), y)]$$

but can only estimate the empirical loss:

$$\hat{L}(f) = \mathbb{E}_{(x,y)\sim S}[\ell(f(x), y)]$$

- Hoeffding's inequality tells us that with probability $1 - \delta$:

$$\left|L(f) - \hat{L}(f)\right| \leq \sqrt{\frac{\ln 2/\delta}{2n}}$$

The Multiple Comparisons Problem (and Uniform Convergence)

- Now what if we have a collection of classifiers $C = (f_1, \dots f_k)$.
- Can no longer use bound from last slide if we select amongst f_i as a function of $\hat{L}(f_i)$: $\max_i |L(f_i) - \hat{L}(f_i)|$ will be larger.
- To be conservative, we can ask for *uniform convergence*.
- Just take a union bound (aka Bonferroni correction): w.p. $1 - \delta$

$$\max_i \left| L(f_i) - \hat{L}(f_i) \right| \leq \sqrt{\frac{\ln 2k/\delta}{2n}}$$

The Multiple Comparisons Problem (and Uniform Convergence)

- For large data sets, this is still very good: multiple comparisons problem is mild.
- Baidu only submitted $k \approx 200$ models, for $n = 100{,}000$. So we have simultaneous 95% confidence intervals of width ≈ 0.0067
- Seemingly enough to confirm their improvement over Google!

But this assumes the functions f_i are chosen independently of the data.

What can go wrong

- A simple model:
  - Binary data: $X \in \{-1,1\}^d$, $y \in \{-1,1\}$.
- Consider the following learning procedure that operates only through a model validation interface:
  1. For each feature $i \in [d]$, validate the classifier $f_i(x) = x_i$.
  2. If $\hat{L}(f_i) < 50\%$, set $c_i = 1$. Else set $c_i = -1$
  3. Construct the final classifier f^* by majority vote:

$$f(x) = \delta(\langle c, x \rangle \geq 0)$$

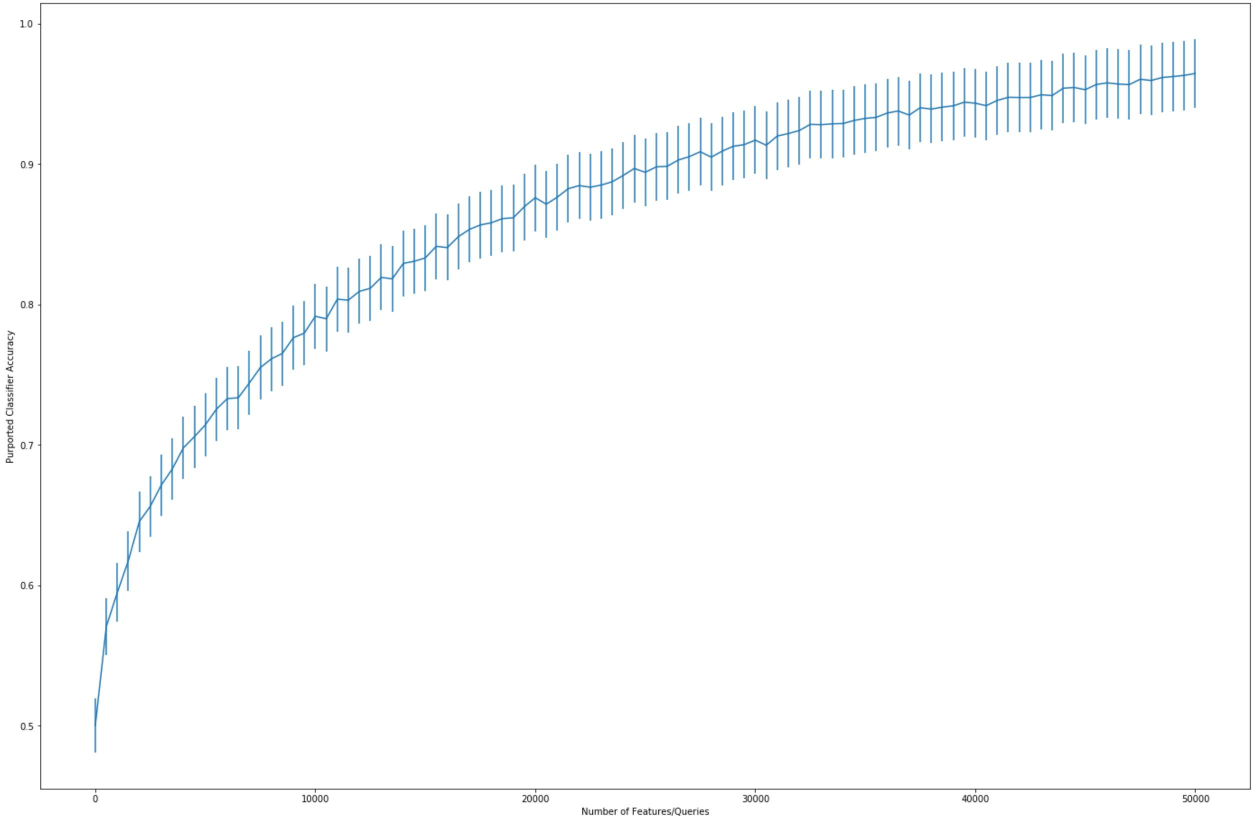
Validates d+1 models in total.

Lets see how it does!

What can go wrong

$$n = 10{,}000 \quad d \in [1, \ldots, 50{,}000]$$

Plot: Accuracy + Bonferroni Corrected Confidence Intervals vs. d



What can go wrong

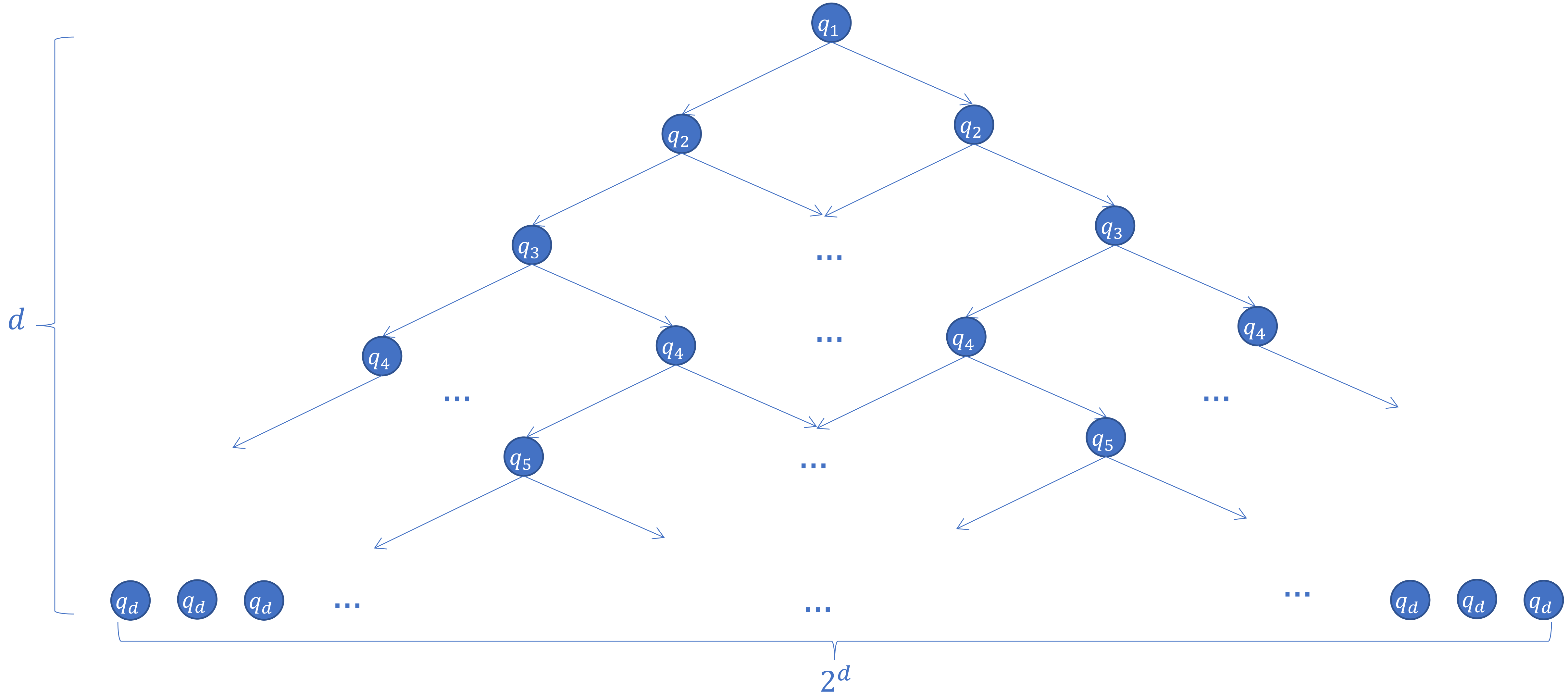
The data: X, y uniformly distributed and uncorrelated.

All classifiers have error = 50%.

Bonferroni correction disastrously failed.

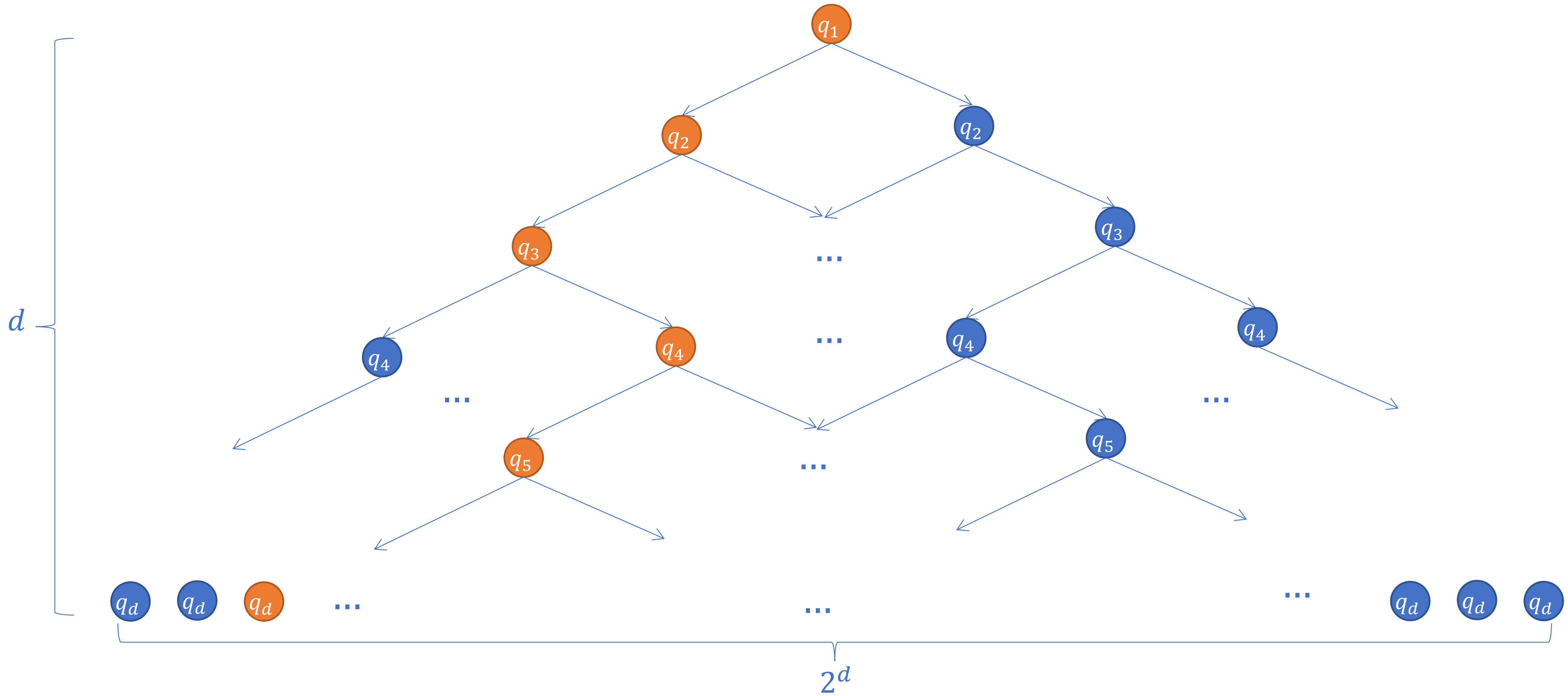
The Garden of the Forking Paths [GL14]

We can map out what our algorithm would have done in every eventuality:



The Garden of the Forking Paths [GL14]

We can map out what our algorithm would have done in every eventuality:



The Garden of the Forking Paths [GL14]

- We only asked $d+1$ queries, but there were 2^d models that we could have tested (all equally likely) depending on what answers we got.
  - Bonferroni correction on the queries asked is *not enough*.
  - A much larger *implicit* multiple comparisons problem: (conservatively) must correct for all models that could have been validated.
    - In this case, really do have to.

The Garden of the Forking Paths [GL14]

- Issues:
  - These corrections are giant: adaptivity leads to exponential blowup in multiple comparisons problem.
  - Generally, we won't have a map of the garden.
    - e.g. whenever human decision making is involved, or algorithms are complicated.
- Solution: Pre-registration?
  - Gates off the garden. Forces analysis to walk a straight line.
  - Safe but overly conservative. Incompatible with data re-use.

How can we make it safe to wander the garden?

A Formalization of the Problem: Statistical Queries

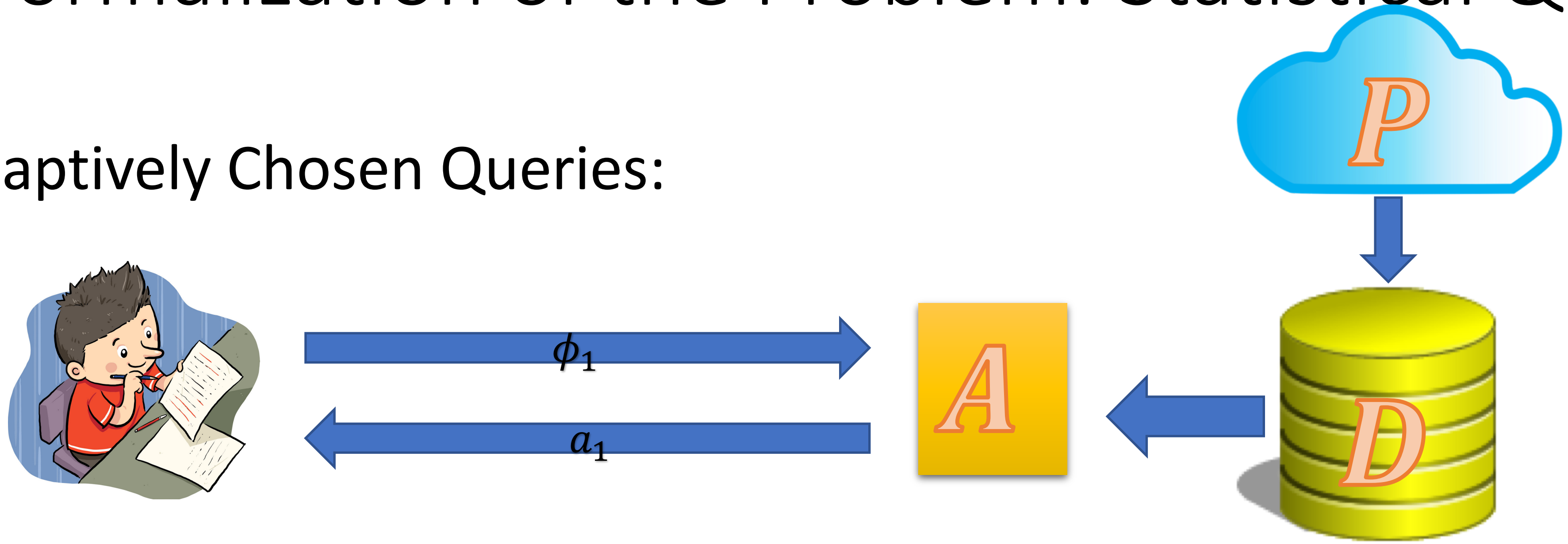
- A data universe X (e.g. $X = \{0,1\}^d$)
- A distribution $P \in \Delta X$
- A dataset $S \sim P^n$ consisting of n points $x \in X$ sampled i.i.d. from P.

A Formalization of the Problem: Statistical Queries

- A *statistical query* is defined by a predicate
$$\phi: X \to [0,1].$$
- The answer to a statistical query is
$$\phi(P) = E_{x\sim P}[\phi(x)]$$
- A statistical query oracle is an algorithm for answering statistical queries: $A: SQ \to [0,1]$
  - Parameterized by a dataset: A_S

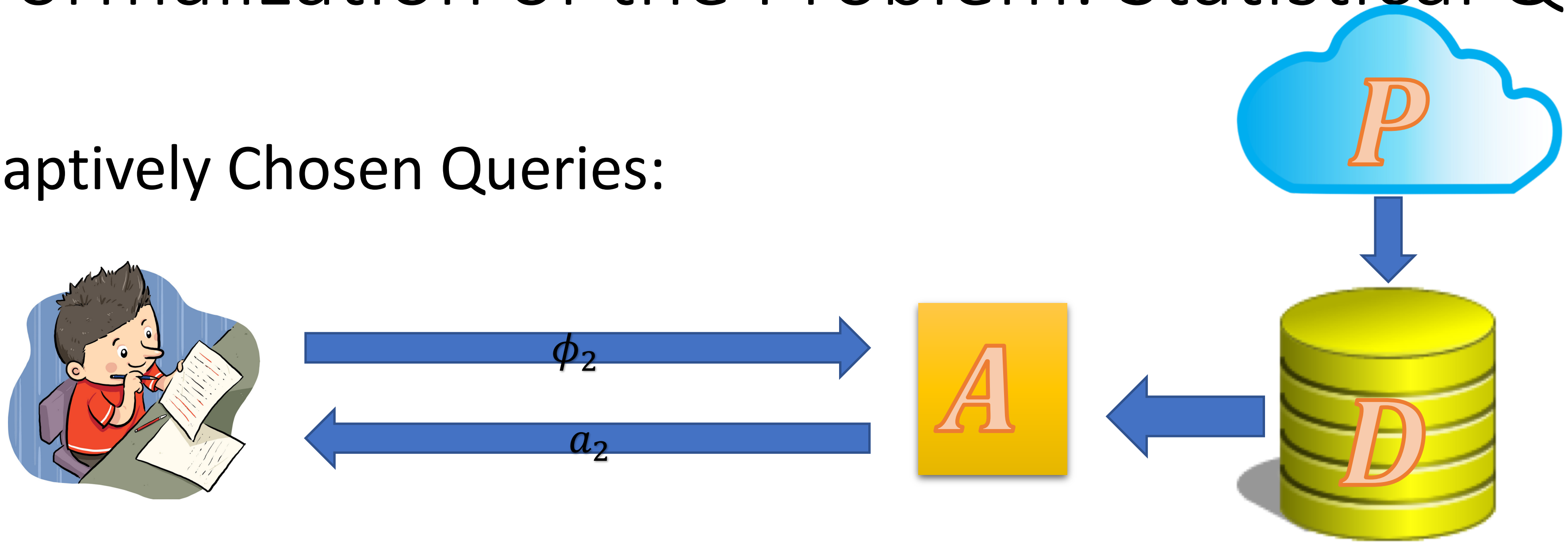
A Formalization of the Problem: Statistical Queries

- Adaptively Chosen Queries:



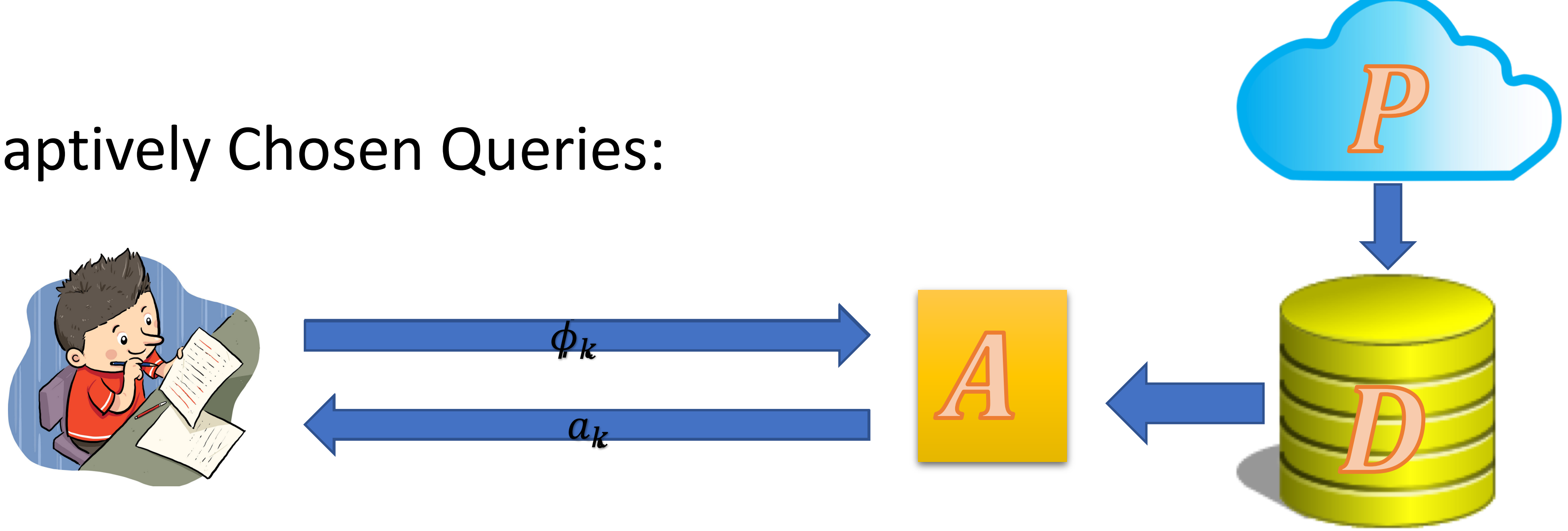
A Formalization of the Problem: Statistical Queries

- Adaptively Chosen Queries:



A Formalization of the Problem: Statistical Queries

- Adaptively Chosen Queries:



- A statistical estimator A is (ϵ, δ)-accurate for sequences of k adaptively chosen queries $\phi_1, \ldots, \phi_k$ if for all  and 

, with probability $1 - \delta$:

$$\max_i |A_S(\phi_i) - \phi_i(P)| \leq \epsilon.$$

A Formalization of the Problem: Statistical Queries

- Main quantity of interest: How must ϵ scale with n, k?

Recall: non-adaptive case: $\epsilon = O\left(\sqrt{\frac{\log k}{n}}\right)$

Our adaptive example had $\epsilon \geq \Omega\left(\sqrt{\frac{k}{n}}\right)$

By carefully designing a statistical estimator A, can we do better?

Warmup: An Easy Theorem (If Pigs Could Fly)

Theorem (informal): Let A be a statistical estimator such that for any sequence of k adaptively chosen queries $\phi_1, \ldots, \phi_k$ we have:

1. **Empirical accuracy**: $\max_i |A_S(\phi_i) - \phi_i(S)| \leq \tau$ and
2. **Compressibility**: the transcript produced by A can be compressed to $\leq t$ bits.

then A is (ϵ, δ)-accurate for $\epsilon = \tau + \sqrt{\frac{t + \log 2k/\delta}{2n}}$

Warmup: An Easy Theorem (If Pigs Could Fly)

Proof:

Fix any data analyst (mapping from query answers to queries). Each sequence of k queries asked corresponds to a transcript of answers generated by A.

By compressibility, there are at most 2^t such transcripts, and so at most $k \cdot 2^t$ queries that can ever be asked.

Apply a Bonferroni correction to these $k \cdot 2^t$ queries:

$$\max_i |\phi_i(S) - \phi_i(P)| \leq \sqrt{\frac{t + \log 2k/\delta}{2n}}$$

By empirical accuracy:

$$\max_i |A(\phi_i) - \phi_i(S)| \leq \tau$$

Theorem follows from triangle inequality.

Strengths of this style of theorem

- Don’t need a map of the garden: can apply Bonferroni correction to a small set of queries even if we don’t know what they are.
  - So don’t need to understand data analyst – can be a human being e.g.
- Don’t need to constrain data analyst at all (e.g. as in pre-registration) except that they should access data only via our interface.

Are there non-trivial estimators that satisfy the conditions of our theorem?

Towards Compressible Estimators

- Suppose queries ϕ_i were paired with guesses $g_i \in [0,1]$.
- Given a query (ϕ_i, g_i), A can either answer:
  - "Yup": Guess was correct ($|g_i - \phi_i(S)| \leq \tau$)
  - "Nope, the answer is $a_i \in [0,1]$"
- How well can we compress the transcript of answers if only w of the guesses are wrong?

Towards Compressible Estimators

- One way to encode the transcript: list tuples corresponding to the *indices* of the queries whose guesses were wrong, together with their empirical answers (to $\log 1/\tau$ bits of precision).
- Encoding length: $t \leq w \cdot (\log k + \log 1/\tau)$
  - $\leq$ w entries in the list
  - Each contains an index ($\log k$ bits) and a value ($\log 1/\tau$ bits)

$$\text{Error: } \epsilon = O\left(\sqrt{\frac{w(\log k + \log n) + \log\left(\frac{k}{\delta}\right)}{n}}\right) = \tilde{O}\left(\sqrt{\frac{w \cdot \log \frac{k}{\delta}}{n}}\right)$$

To come up with compressible estimators, it suffices to come up with good guesses.

Coming up with good guesses

A Heuristic: The Reusable Holdout [DFHPRR15].

1. Split the data set S into a "dirty" set S_D and "clean" set S_C
2. For each query ϕ_i, compute a guess $g_i = \phi_i(S_D)$
3. Submit the pair (ϕ_i, g_i) to A_{S_C}.
4. Halt after more than w guesses erred by $\widetilde{\Omega}\left(\sqrt{\frac{w \cdot \log\frac{k}{\delta}}{n}}\right)$.

Coming up with good guesses

Guarantees error $\tilde{O}\left(\sqrt{\frac{w \cdot \log\frac{k}{\delta}}{n}}\right)$ *for any set of* k *queries.*

(But could halt early.)

- Prevents simple "majority" algorithm from overfitting.
- More generally, allows a data analyst to ask queries for a long time so long as he is not getting lost in the garden. Catches/corrects up to w instances of overfitting.

Coming up with good guesses

A Leaderboard: The Ladder Mechanism [BH15]

Goal: Keep track of most accurate classifier so far.

1. Set $bestError_0 = 1.0$
2. For each candidate classifier f_t:
    1. Construct query $\phi_t(S) = \min_{i \le t} \hat{L}(f_i)$
    2. Construct guess $g_t = bestError_{t-1}$
    3. Compute $a_t = A_S(\phi_t, g_t)$
    4. If guess was in error by more than τ, set $bestError_t = a_t$.
    5. Otherwise set $bestError_t = bestError_{t-1}$

Coming up with good guesses

- Each time guess is in error, bestError improves by $\geq \tau$
  - So guess is in error at most $w = 1/\tau$ times.

$$\text{Total error is } \tau + \sqrt{\frac{\frac{1}{\tau} \cdot \log \frac{k}{\delta}}{n}}$$

$$\text{Optimizing: Error is } \epsilon = \tilde{O}\left(\left(\frac{\log \frac{k}{\delta}}{n}\right)^{\frac{1}{3}}\right)$$

Coming up with good guesses

Guarantees for General Statistical Queries: Median Mechanism [RR10]

- Let $X = \{0,1\}^d$.
- Important fact: For any set of k statistical queries, there is a dataset of size $O\left(\frac{\log k}{\tau^2}\right)$ that encodes all queries with τ-accuracy.
  - And the set of *all* such datasets is of size $\approx 2^{d \cdot \frac{\log k}{\tau^2}}$

Coming up with good guesses

1. Let $C_1 = \left\{S' \subset X : |S'| \leq \frac{\log k}{\tau^2}\right\}$
2. For each query ϕ_t:
   1. Construct guess $g_t = median(\phi_t(S') \colon S' \in C_t)$
   2. Compute $a_t = A_S(\phi_t, g_t)$
   3. If the guess was in error by more than τ:
   $$C_{t+1} = \{S' \in C_i \colon |\phi_i(S') - g_t| \leq \tau\}$$
   4. Otherwise:
   $$C_{t+1} = C_t$$

Coming up with good guesses

- We know that $|C_1| = 2^{d \cdot \log k / \tau^2}$, and $|C_t| \geq 1$ for all t.
- Each incorrect guess halves C_t.
- The number of mistaken guesses is $w \leq \frac{d \cdot \log k}{\tau^2}$.

Total error is $\tau + \sqrt{\frac{d \cdot \log k / \delta}{\tau^2 \cdot n}}$

Optimizing, error is $\epsilon = \left(\frac{d \cdot \log \frac{k}{\delta}}{n}\right)^{\frac{1}{4}}$

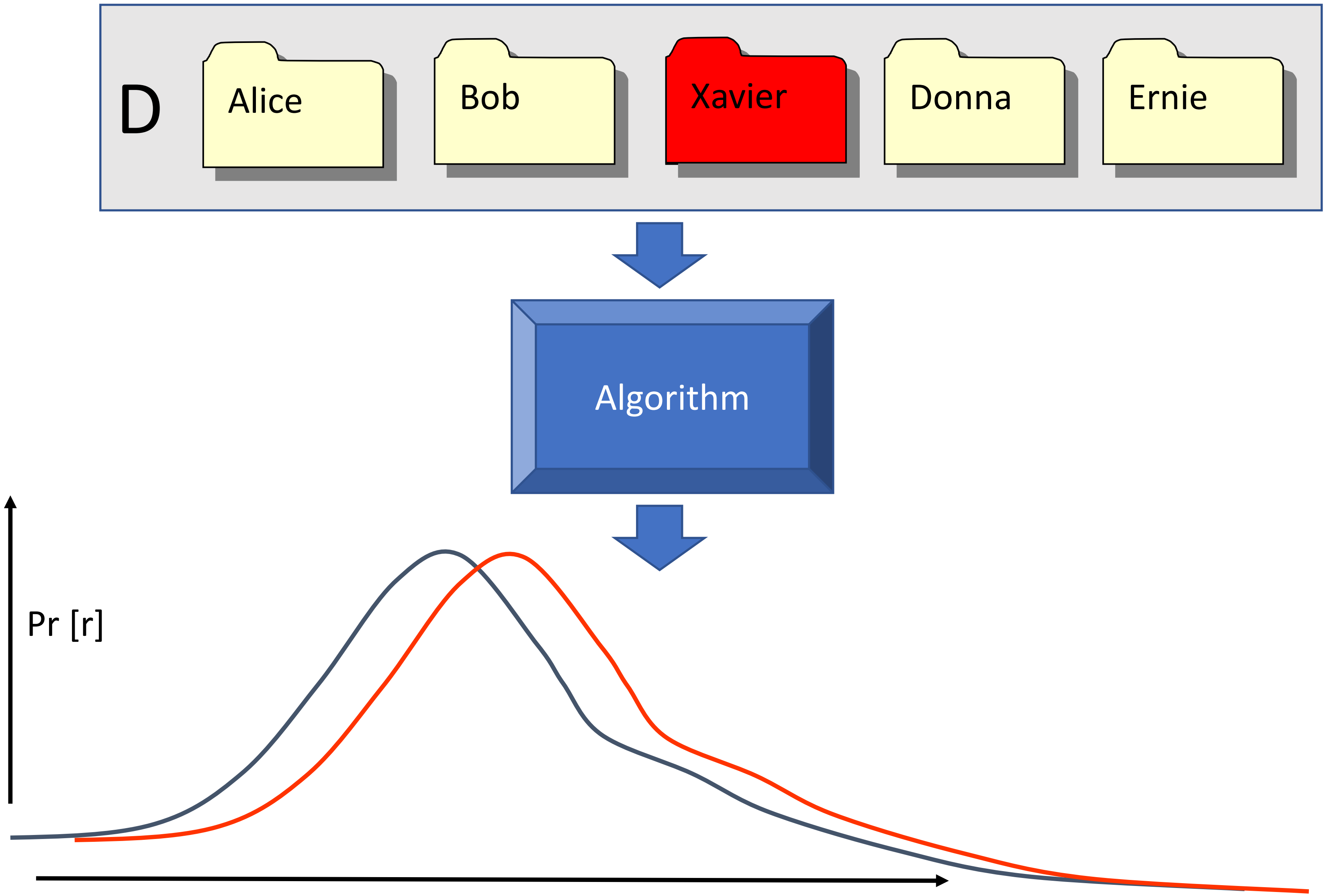
Takeaway

- We can obtain error scaling only polylogarithmically with k!
  - Comparable to the non-adaptive case. ☺
- But…
  - Our dependence on $n, \log k$ could be better, and…
  - Our statistical estimator is not efficient. ☹
- We can become really good at guessing the answers to SQs as soon as k is larger than the (effective) dimension of the data.
  - So big improvements when $n \gg d$ ☺
  - But no guaranteed improvement when $\mathrm{n} \ll d$ ☹

Takeaway

- We don’t yet fully understand how to mitigate all of these caveats.
- But we can get part way there.
- Need to move beyond description length.
  - Some information theoretic measure?
  - Needs to be robust to “post-processing” and should compose well.

Differential Privacy [Dwork, McSherry, Nissim, Smith]



A stability condition on the output *distribution:*

$A: X^n \to \mathcal{O}$ is (α, β)-differentially private if for every pair of neighboring datasets S, S', and outcome E:

$$\Pr[A(S) \in E] \leq e^{\alpha} \Pr[A(S') \in E] + \beta$$

Crucial: Stability on the distribution.
No metric on $\mathcal{O}$.

Distributional Stability Yields Robustness to Postprocessing

Theorem: If $A: X^n \to \mathcal{O}$ is (α, β)-differentially private, and $f: \mathcal{O} \to \mathcal{O}'$ is an *arbitrary* algorithm, then $f \circ A : X^n \to \mathcal{O}'$ is (α, β)-differentially private.

Important:

Don't need to understand *anything* about f.

$f =$  $f =$ 

Distributional Stability Degrades Gracefully Under Composition

Compose( **;D)**
For i = 1 to k:
1. Let  choose an α-DP A_i based on $o_1, \ldots, o_{i-1}$.
2. Let $o_i = A_i(D)$

Output $(o_1, \ldots, o_n)$.

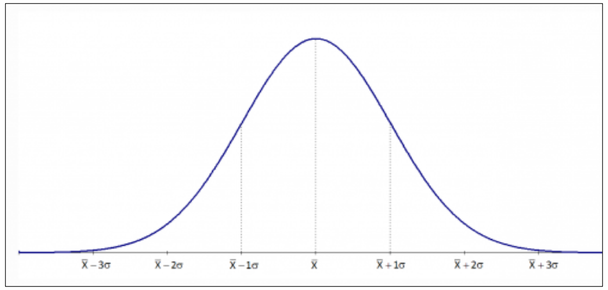
Theorem* [DRV]: For every , and β', **Compose(**  **;D)** is (α', β')-differentially private for:

$$\alpha' = O\left(\alpha \cdot \sqrt{k \cdot \ln\left(\frac{1}{\beta'}\right)}\right)$$

Composition and Post-processing: Modular Algorithm Design

- Differential Privacy is a powerful *language* for stable algorithm design.
- Can combine a collection of differentially private primitives *modularly* in arbitrary ways.
- Simplest primitive: independent, Gaussian noise addition.
  - e.g. Output $\phi(S) + N(0, \sigma^2)$

    where $\sigma = O\left(\frac{\sqrt{\ln(\frac{1}{\beta})}}{\alpha n}\right)$



Another Transfer Theorem

Theorem: [DFHPRR'15,BNSSSU'16]: Let A be a statistical estimator that satisfies:

1. **Differential Privacy:** A is $(\epsilon, \epsilon \cdot \delta)$-differentially private, and
2. **Empirical Accuracy**: For any sequence of k adaptively chosen queries $\phi_1, \ldots, \phi_k$, with probability $1 - \epsilon \cdot \delta$:
   $\max_{\mathrm{i}} |A_S(\phi_i) - \phi_i(S)| \leq \epsilon$

Then A is $(\mathrm{O}(\epsilon), O(\delta))$-accurate.

References

- [RR10] "Interactive privacy via the median mechanism." Roth, Roughgarden. STOC 2010.
- [GL14] "The Statistical Crisis in Science." Gelman, Loken. *American Scientist*, 2014.
- [DFHPRR15] Dwork, Feldman, Hardt, Pitassi, Reingold, Roth. 2015.
  - "Preserving statistical validity in adaptive data analysis" *STOC*
  - "Generalization in adaptive data analysis and holdout reuse" *NIPS*
  - "The reusable holdout: Preserving validity in adaptive data analysis" *Science*
- [BH15] "The ladder: A reliable leaderboard for machine learning competitions." Blum, Hardt. ICML 2015.
- [BNSSSU16] "Algorithmic stability for adaptive data analysis." Bassily, Nissim, Smith, Steinke, Stemmer, Ullman. STOC 2016.

See http://www.adaptivedataanalysis.com for lecture notes.