Beyond Worst-Case Adversaries in Machine Learning

Nika Haghtalab, UC Berkeley

Based on joint work with Tim Roughgarden and Abhishek Shetty

ML Algorithmic Guarantees

Data is generated **stochastically** from a fixed distribution

Learner learn a function using the data

Successful if it gets good performance over the underlying distribution.

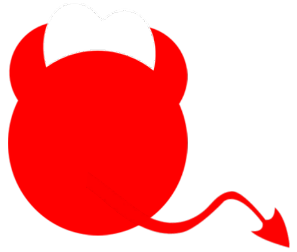
Not concerned with robustness or what happens if the world were to change.

Stochastic or Offline

Data is generated by an all-powerful **adaptive adversary,** who knows the algorithm and history.

Successful if it gets good performance over adversarially generated data.

If successful, then it's going to be robust to any misspecifications or attacks.



Adversarial, Robust, or Online

Importance of Robustness

Increased interactions between learning algorithms and people requires robust learning guarantees.

Need new models to guide the design of learning algorithms that are robust to day-to-day adversaries.

Get essentially the same performance guarantees for the learner against these adversaries, as you could in the stochastic world.

Algorithm Design and Analysis

Instance is generated **stochastically** from a fixed distribution

Algorithm computes a **solution.**

Successful if it is a **good solution** in expectation over the distribution.

Average-Case Analysis

Instance is generated by an all-powerful **adaptive adversary,** who knows the algorithm and history.

Successful if it can find a **good solution** even for the **worst-case instance**.

Worst-Case Analysis

Smoothed Analysis: Basic Idea

Idea [Spielman & Teng 01]:

- Adversary chooses and instances, then nature slightly perturbs it, e.g., Gaussian.
- Goal: For any instance, perform well in expectation/w.h.p over the perturbations.

Modern perspective:

- Adversary chooses a distribution over instances. The distribution has to be "sufficiently anti-concentrated".
- Goal: For any "anti-concentrated" distribution, perform well in expectation/w.h.p.

When is it useful? When the worst-case instances are "brittle"

Ideally:

- We can get essentially same performance guarantees as in the average-case for the smoothed adversaries.

Average-Case Analysis **Smoothed Analysis** **Worst-Case Analysis**

Smoothed Analysis: Past, Present, Future

Running time of simplex method [**Spielman & Teng 01, Deshpande & Spielman 05, …**]

- Simplex can take exponential time for worst-case instances
- Simples takes polynomial time in expectation when the Gaussian variance is $\geq 1/poly(n)$

Running time of local search methods:

- Lloyd algorithm for k-means, 2-OPT heuristic for TSP, take exponential number of iteration in worst case, but polynomial in the smoothed case.

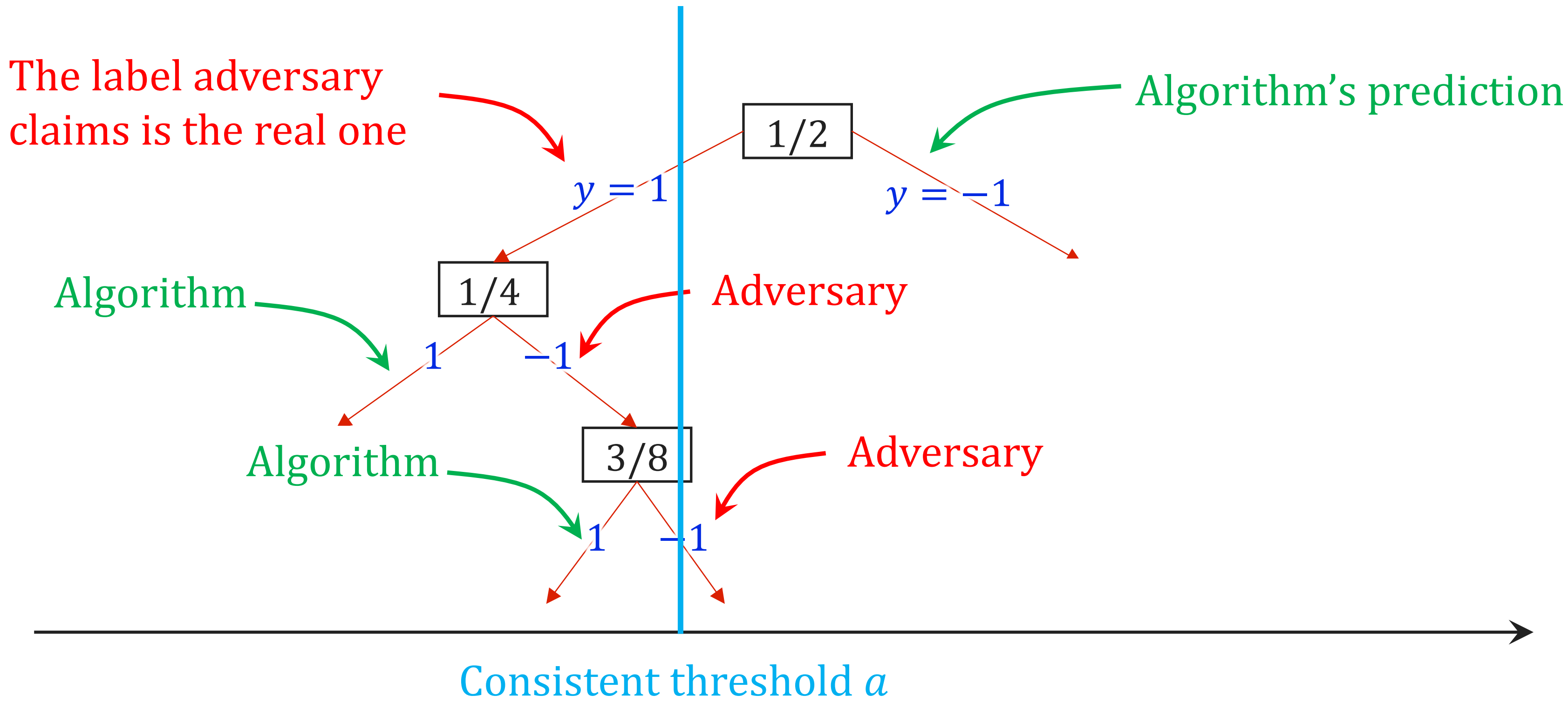
Machine learning (Information + Computation)

- Even what is “learnable” depends on the model of the adversary.
- Fundamental application of smoothed analysis

Worst-Case Adversaries in Learning (Online)

Unknown threshold function $a : x \geq a$ is labeled $+$, and $x < a$ is labeled $-$.

- Algorithm has to predict labels of **adaptively and adversarially** selected points.



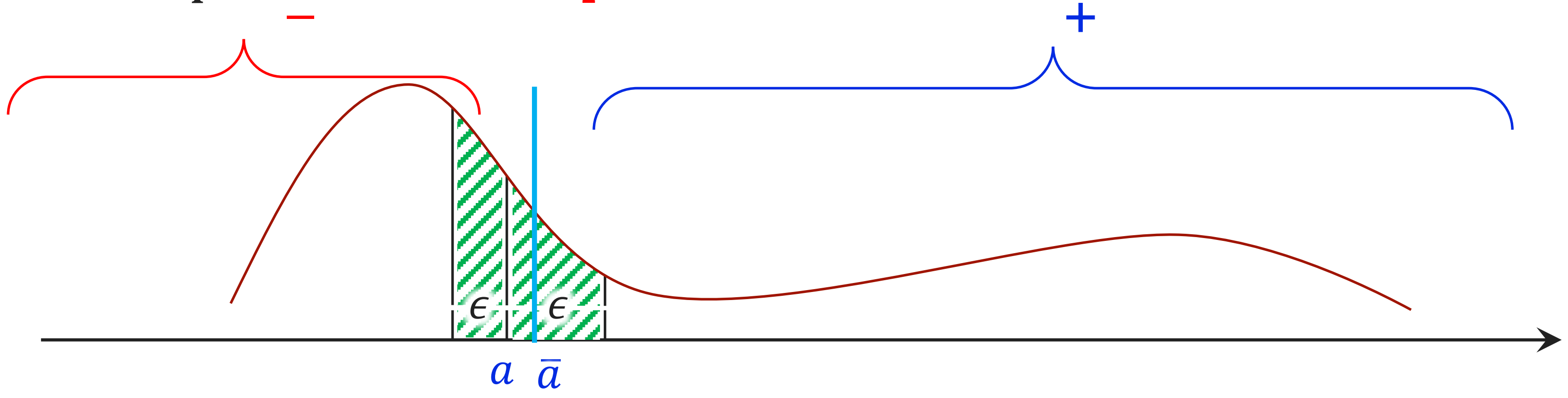
Algorithm is forced to make a mistake at every round.

- T mistakes over T instances.

Stochastic "Adversaries" in Learning (Offline)

Unknown threshold a :

- Larger point are positive, and smaller point are negative.
- Algorithm has to predict labels of **points drawn from a distribution.**



Algorithm: Let $\epsilon = \sqrt{T}$.

First $\frac{1}{\epsilon}$ rounds, predict however you like. ($\leq \frac{1}{\epsilon} = \sqrt{T}$ mistakes up to here.)

Then take any $\bar{a}$ that is consistent with the data, predict $+$ if $\mathrm{x} \geq \bar{a}$ and $-$ if $\mathrm{x} < \bar{a}$.

Claim: $\leq \epsilon$ total prob in $|a - \bar{a}|$

- After $\frac{1}{\epsilon}$ rounds, for any a' that's more than ϵ in total probability away from a, we would have seen a point with opposite label.

Claim: $\leq \sqrt{T}$ additional mistakes, because $\epsilon T = \sqrt{T}$.

Formal Setup: Offline and Online Learning

Offline Learning: Unknown distribution D over $X \times Y$ and function class H.

At round t

Learner picks prediction rule $f_t : X \to Y$, not necessarily deterministic.

The world picks $(x_t, y_t) \sim D$

Algorithm makes a mistake if $f_t(x_t) \neq y_t$.

Goal: Get $o(T)$ regret.

REGRET $=$ $\boxed{\text{Alg's \# of mistakes}}$ $-$ $\boxed{\text{\# of Mistakes the best } h \in H \text{ makes, in hindsight}}$

As $T \to \infty$, avg number of mistakes Alg makes is no worst than the best predictor.

Formal Setup: Offline and Online Learning

Online Learning

~~Offline Learning: Unknown distribution D over $X \times Y$~~ and function class H.

At round t

Learner picks prediction rule $f_t : X \to Y$, not necessarily deterministic.

Adversary picks (x_t, y_t), knowing the history for $1, \ldots, t-1$ and the algorithm

Algorithm makes a mistake if $f_t(x_t) \neq y_t$.

Goal: Get $o(T)$ regret.

$$\text{REGRET} = \sum_{t=1}^{T} 1(f_t(x_t) \neq y_t) - \min_{h \in H} \sum_{t=1}^{T} 1(h(x_t) \neq y_t)$$

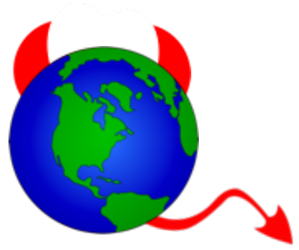
As $T \to \infty$, avg number of mistakes Alg makes is no worst than the best predictor.

Formal Setup: Smoothed Analysis of Online Learning

There is a function class H and domain X ($X \subseteq R^n$ has finite Lebesgue measure)

At round t

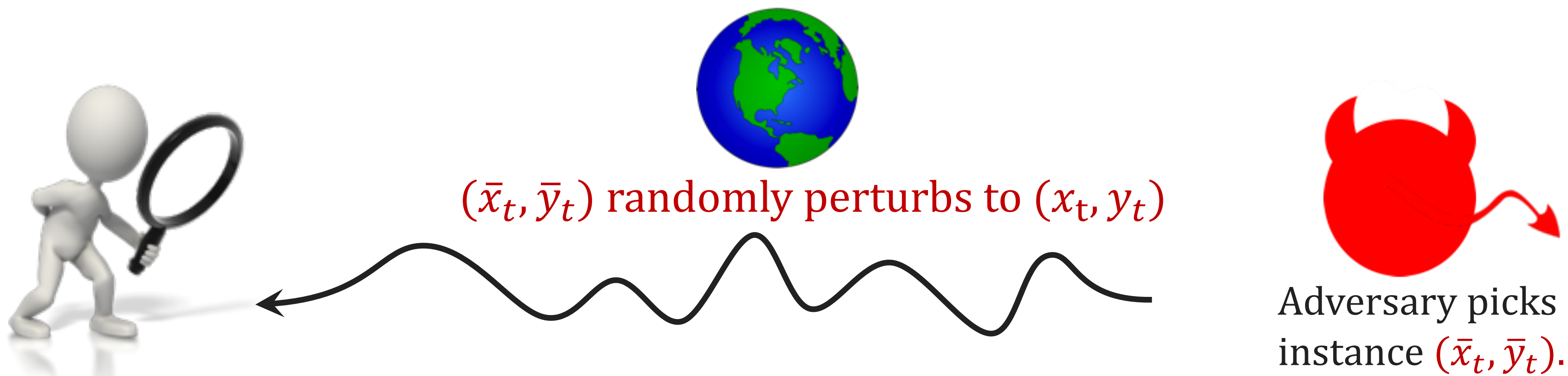
Learner picks prediction rule $f_t : X \to Y$,
not necessarily deterministic.



Adversary picks a σ-smooth distribution D_t knowing the history for $1, \dots, t-1$ and the algorithm

σ-smooth distribution: max density is $\leq \frac{1}{\sigma} \times$uniform density on X

Modern perspective on smoothness (more general for finite lebesgue measure X)



$(\bar{x}_t, \bar{y}_t)$ randomly perturbs to (x_t, y_t)

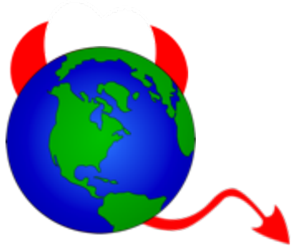
Adversary picks an instance $(\bar{x}_t, \bar{y}_t)$.

Formal Setup: Smoothed Analysis of Online Learning

There is a function class H and domain X ($X \subseteq R^n$ has finite Lebesgue measure)

At round t

Learner picks prediction rule $f_t : X \to Y$, not necessarily deterministic.



Adversary picks a σ-smooth distribution D_t knowing the history for $1, \dots, t-1$ and the algorithm

σ-smooth distribution: max density is $\leq \frac{1}{\sigma} \times$uniform density on X

Algorithm makes a mistake if $f_t(x_t) \neq y_t$.

Goal: Get $o(T)$ regret.

$$\text{REGRET} = \sum_{t=1}^{T} 1(f_t(x_t) \neq y_t) - \min_{h \in H} \sum_{t=1}^{T} 1(h(x_t) \neq y_t)$$

As $T \to \infty$, avg number of mistakes Alg makes is no worst than the best predictor.

The Landscape

	Online Learning Regret	Perturbation
Online Learning (Worst-Case)	$\widetilde{\Theta}\left(\sqrt{\mathrm{Ldim(H)}\,T}\right)$ $\widetilde{\Theta}\left(\sqrt{\log(\lvert\mathrm{H}\rvert)\,T}\right)$	No perturbation $\boldsymbol{\sigma = 0}$
Offline learning or Uniform Case	$\widetilde{\Theta}\left(\sqrt{\mathrm{VCDim(H)}\,T}\right)$	Maximum perturbation $\boldsymbol{\sigma = 1}$

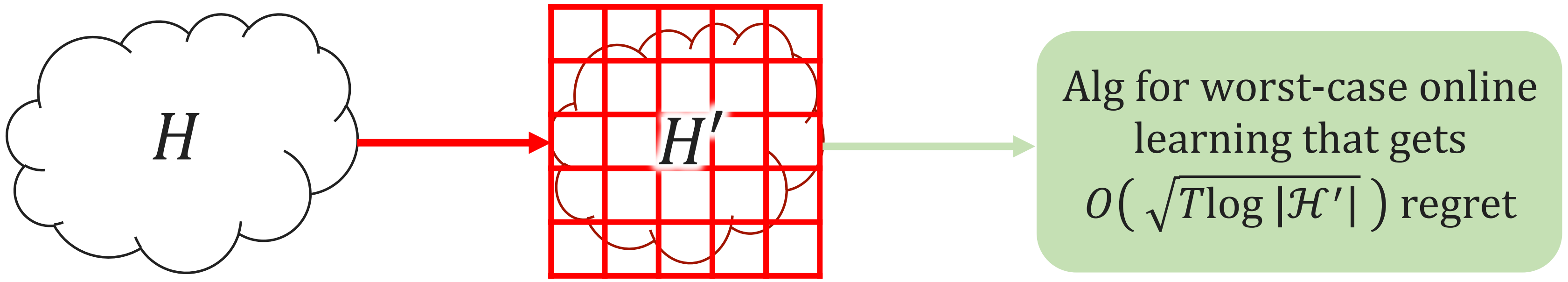
Interpreted as an impossibility result, because VCDim $\ll$ Ldim
→ For simple classes, Ldim $= \infty$ (and $\log(\lvert\mathrm{H}\rvert) = \infty$) but VCDim $= 1$.

Theorem [H., Roughgarden, Shetty '21]
In presence of Adaptive but Smooth Adversaries the regret is $\widetilde{\Theta}\left(\sqrt{\mathrm{VCDim(H)}\,T\ln(1/\sigma)}\right)$

H is learnable with under smoothed analysis if and only if H is learnable on a uniform distribution.

Why did the Stochastic Case Work?

We could approximate H that's potentially infinite, with a finite H′.



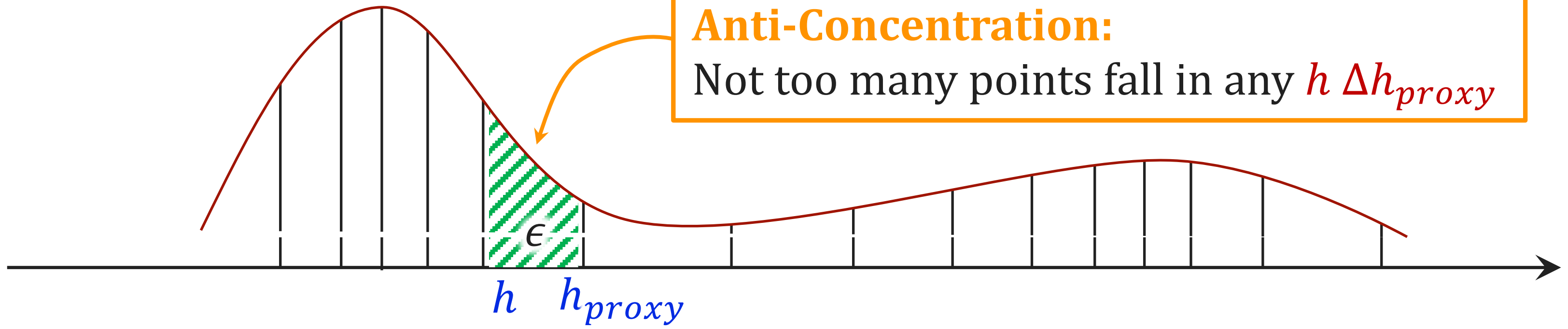
Alg for worst-case online learning that gets $O(\sqrt{T\log|\mathcal{H}'|})$ regret

Approx Error

max # points in $h\ \Delta h_{proxy}$

The Net: For each $h \in H$ there is $h_{proxy} \in H'$, where the total density in $h\ \Delta h_{proxy}$ is small.

Approx Error is small: Performance of any $h \in H$ is close to the performance of corresponding $h_{proxy} \in H'$

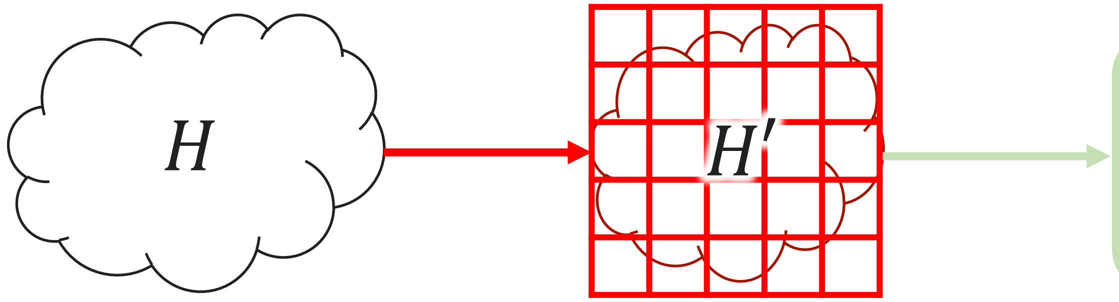


Anti-Concentration:
Not too many points fall in any $h\ \Delta h_{proxy}$

Infinitely many $h\ \Delta h_{proxy}$, but because instances are i.i.d we can take union bound over them.

What went wrong for the online case?

We could approximate H that's potentially infinite, with a finite H′.



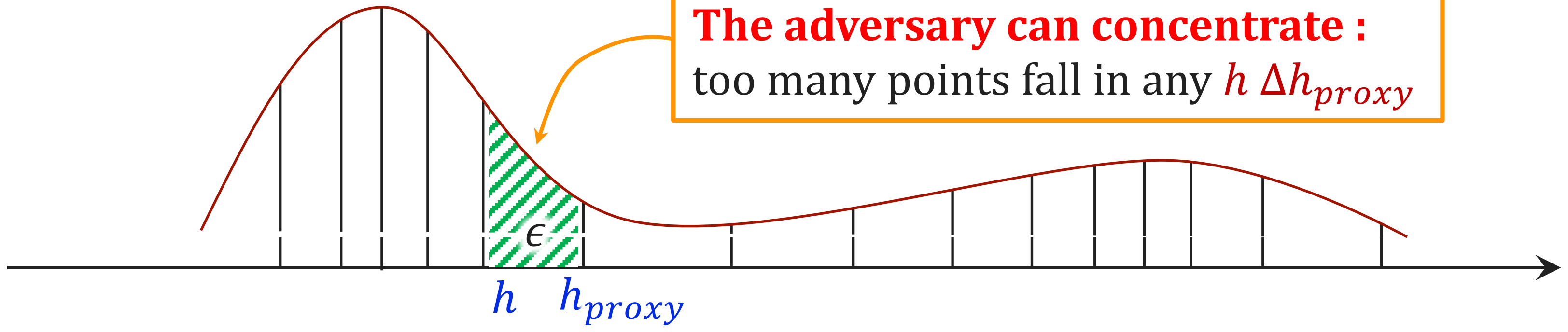
Alg for worst-case online learning that gets $O(\sqrt{T \log |\mathcal{H}'|})$ regret

Approx Error

max # points in $h\ \Delta h_{proxy}$

The Net: For each $h \in H$ there is $h_{proxy} \in H'$, where the total density in $h\ \Delta h_{proxy}$ is small.

Approx Error is small: Performance of any $h \in H$ is close to the performance of corresponding $h_{proxy} \in H'$

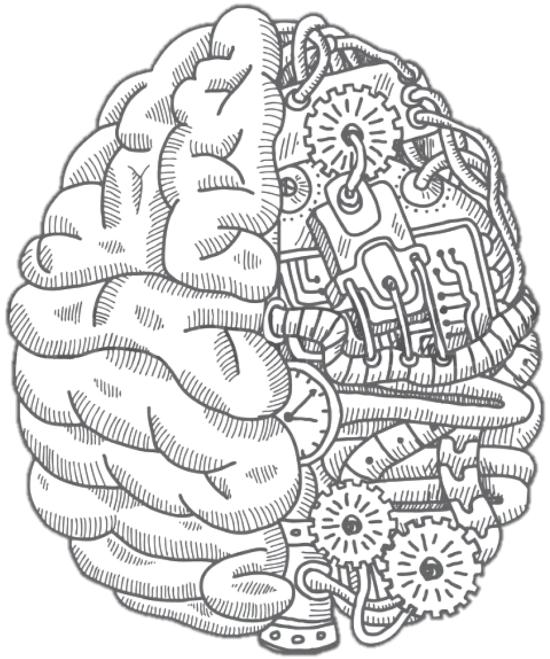


The adversary can concentrate : too many points fall in any $h\ \Delta h_{proxy}$

Infinitely many $h\ \Delta h_{proxy}$, but because ~~instances are i.i.d we can take union bound over them~~.

Broad Question

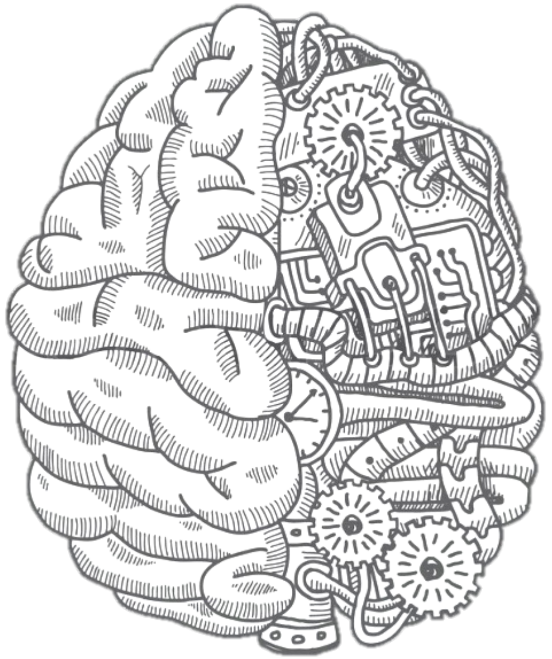
How do we preserve anti-concentration when a sequence of smooth distributions are adaptively chosen?



Challenge

Each σ-smooth distribution is anti-concentrated.

The challenge is correlations between these smooth distributions.



Couple Adaptive Smoothness with Uniformity

Probability Couplings:

- We have two or more distributions X and Z.
- A coupling is a joint distribution on $X \times Z$ that connects them, so that there is a "nice property" between the draws (x, z).

Ideally, we want to couple random variables from a sequence of smooth distributions $(D_1, D_2, \dots, D_T)$ with draws from a uniform distribution.

"Nice Property":

- $X_1, \dots, X_T$ drawn from $D_1, D_2, \dots, D_T$ are a subset of $Z_1, \dots, D_{kT}$ drawn from uniform distribution.

A Good Coupling

Theorem [H., Roughgarden, Shetty '21]
For any adaptive sequence of T distributions, there is a coupling between:

1. $(X_1, \dots, X_T) \sim (D_1, D_2, \dots, D_T)$
2. $Z_1 \dots, Z_{Tk} \sim Unif$ and independent and $k \approx 1/\sigma$.
3. With high prob. $X_1, \dots, X_T \subseteq Z_1 \dots, Z_{Tk}$

Uniform distribution is not "concentrated". So, $X_1, \dots, X_T \subseteq Z_1 \dots, Z_{Tk}$ aren't either.

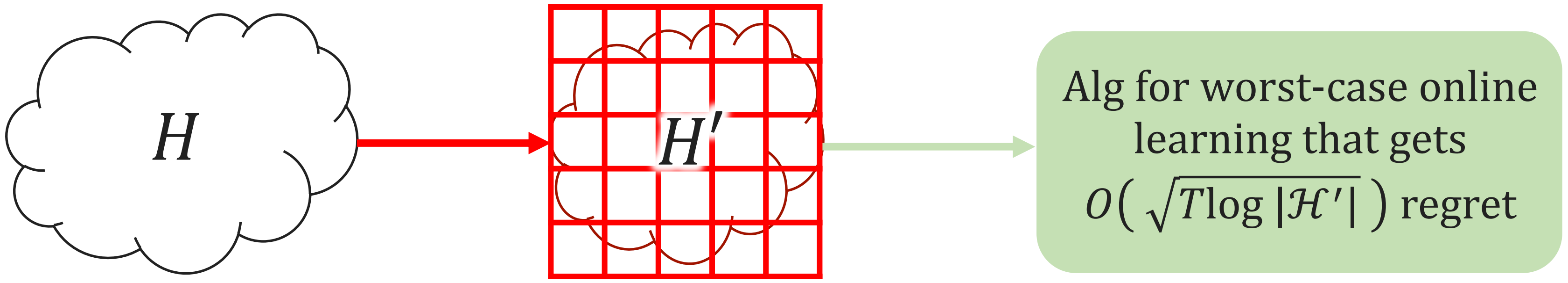
- I want to say that no $h \, \Delta h_{proxy}$ includes too many $X_1, \dots, X_T$.
- Sufficient to say no $h \, \Delta h_{proxy}$ includes too many $Z_1 \dots, Z_{Tk}$.
- $Z_1 \dots, Z_{Tk}$ are i.i.d and guaranteed to be scattered.

Adaptive smoothed adversaries can't be much worst than stochastic adversaries (on a slightly longer time scale).

Overview of the Main Results

Theorem [H., Roughgarden, Shetty '20]
In presence of Adaptive but Smooth Adversaries the regret is $\widetilde{\Theta}\left(\sqrt{\mathrm{VCDim(H)}\, T \ln(1/\sigma)}\right)$

Step 1: Choose H′ that is a finite approximation of H



H
H'
Alg for worst-case online learning that gets $O\left(\sqrt{T\log|\mathcal{H}'|}\right)$ regret

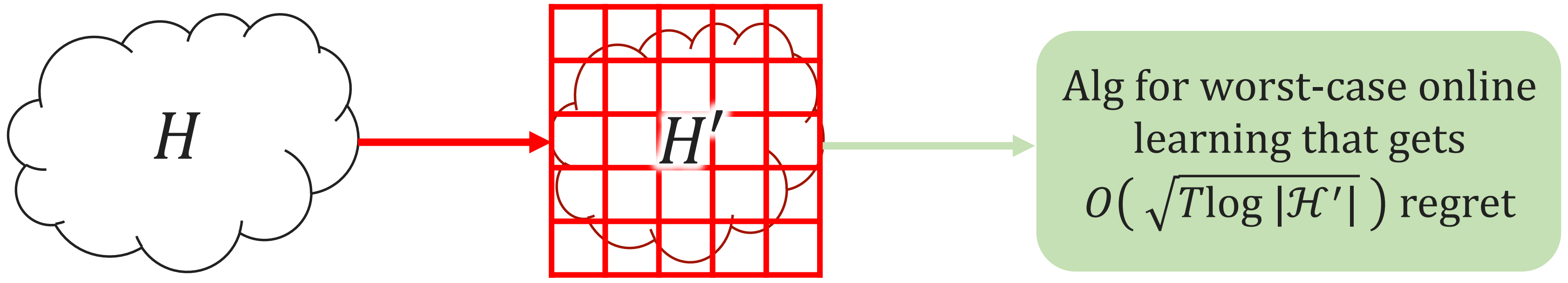
Approx Error
max # points in $h\ \Delta h_{proxy}$

How do we select H′?

- Take H′ that such that $x \sim$ Unif, i.e., $\Pr_U[\text{a point falls in } h\ \Delta h_{proxy}] \le \epsilon$.
- Works nicely for σ-smooth distributions too:

$$\mathbb{E}_D[\#\text{points in } h\ \Delta h_{proxy}] \le T\epsilon/\sigma.$$

Overview of the Main Results


H
H'
Alg for worst-case online learning that gets $O(\sqrt{T \log |\mathcal{H}'|})$ regret

Approx Error
max # points in $h \, \Delta h_{proxy}$

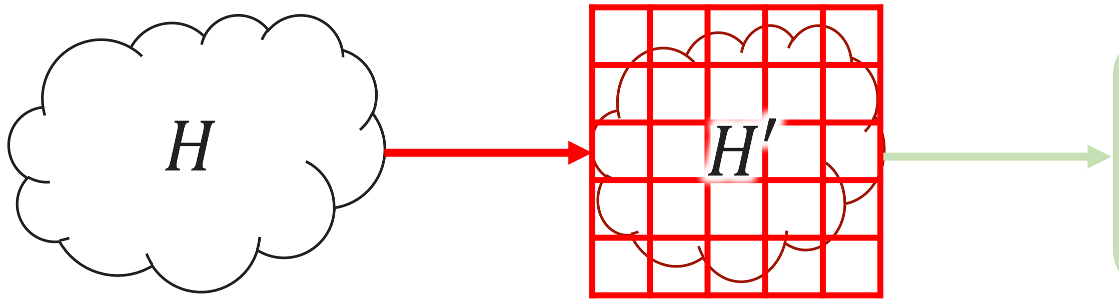
Step 1: We got that $\mathbb{E}_D\left[\#\text{points in } h \, \Delta h_{proxy}\right] \le T\epsilon/\sigma$.

Step 2: Apply the coupling

$$\underbrace{\max_{h \in H} \begin{matrix}\#\text{ points} \sim D_1, \ldots \mathrm{D_T} \\ \text{fall in } h \, \Delta h_{proxy}\end{matrix}}_{\text{Approx Error}} \le \underbrace{\max_{h \in H} \begin{matrix}\#\text{ points} \sim Unif \\ \text{fall in } h \, \Delta h_{proxy}\end{matrix}}_{\text{Approx Error}}$$

"Nice Property": $X_1, \ldots, X_T$ drawn from $D_1, D_2, \ldots, D_T$ are a subset of $Z_1, \ldots, D_{kT}$ drawn from uniform distribution.

Overview of the Main Results



Alg for worst-case online learning that gets $O(\sqrt{T \log |\mathcal{H}'|})$ regret

Approx Error

max # points in $h\ \Delta h_{proxy}$

Step 1: We got that $\mathbb{E}_D\left[\#\text{points in } h\ \Delta h_{proxy}\right] \leq T\epsilon/\sigma$.

Step 2: Apply the coupling

$$\underbrace{\max_{h \in H}\ \begin{matrix}\text{Approx Error} \\ \#\text{ points} \sim D_1, \ldots D_T \\ \text{fall in } h\ \Delta h_{proxy}\end{matrix}}\ \leq\ \max_{h \in H}\ \begin{matrix}\text{Approx Error} \\ \#\text{ points} \sim Unif \\ \text{fall in } h\ \Delta h_{proxy}\end{matrix}$$

Step 3: Bound the Approx Error for the uniform distribution.

No concerns about the adversary and robustness. Just the classical stuff!
VC dimension uses i.i.d uniform r.v. to show that approx. error is small.

Main Message

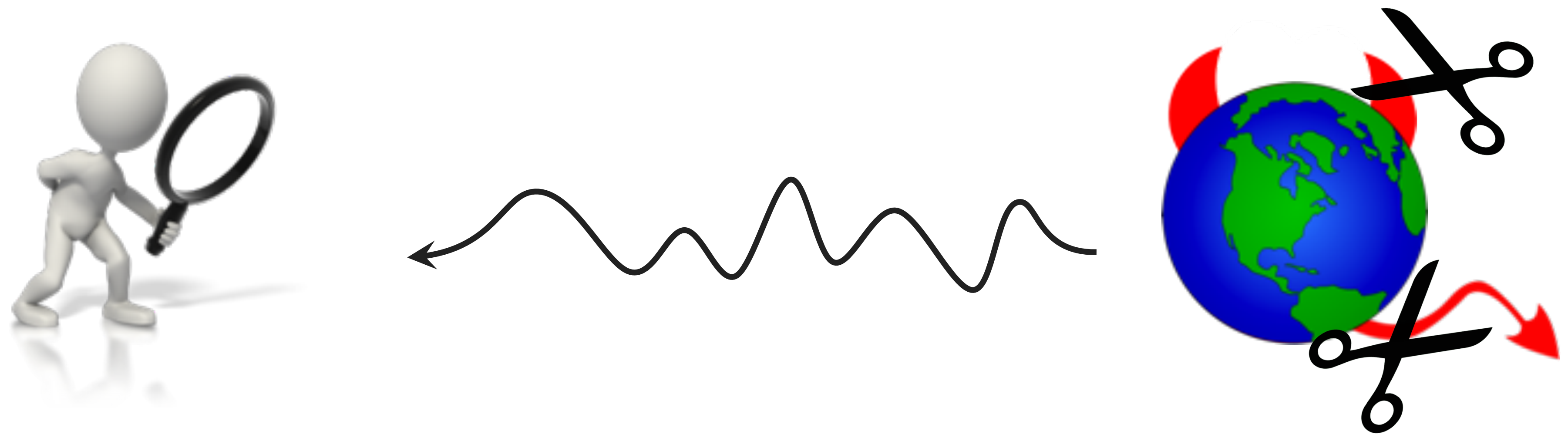
We want to be robust over T interactions with an adaptive smooth adversary.

Classical algorithms and analysis from the stochastic case can be lifted and be use with smoothed adaptive adversaries

Smoothed Analysis of Adaptive Adversaries

Ideal Results

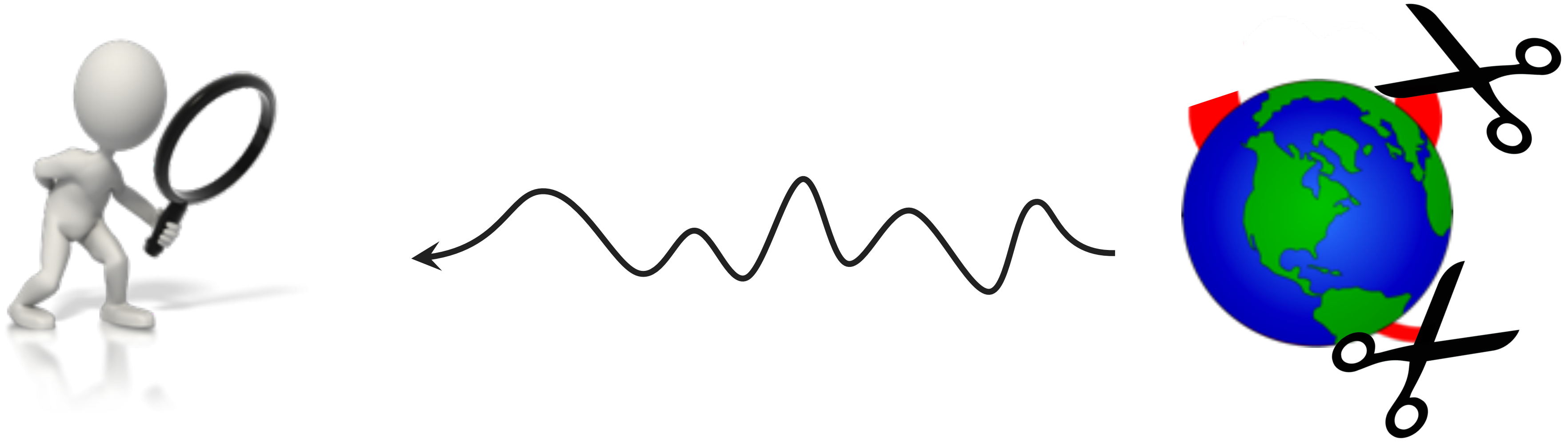
Get essentially the same performance guarantees for the algorithm against an adversary, as you could in the stochastic world.



Smoothed Analysis of Adaptive Adversaries

Ideal Results

Get essentially the same performance guarantees for the algorithm against an adversary, as you could in the stochastic world.



Reducing interactions with smooth adaptive adversary to the stochastic world.

Getting rid of the worst aspect of being adversarial.

Recipe: Smoothed Analysis with Adaptive Adversaries

1. Solve the problem for the uniform case.
   1. Isolate and identify the the steps that rely on anti-concentration.

      → Look at where randomness comes in and identify concentration property, potential functions, or other monotone set functions that implicitly measure concentration of some measure.
2. Apply the coupling lemma
   1. Replace T round of an adaptive smoothed adversary with T/σ uniform R.Vs.
   2. Update the dependence of step 1.1. for T/sigma uniform R.Vs.

      →The property $X_1, \dots, X_T \subseteq Z_1 \dots, Z_{T/\sigma}$ can only increases concentration, potential functions, or other monotone set functions.

      → $Z_1 \dots, Z_{T/\sigma}$ are uniform, so only moderate increase in concentration, etc.
3. Put it all back together, use the original algorithm and analysis technique.

Use Case 2: Online Discrepency Minimiztion

A sequence of vectors $v_1, v_2, \ldots, v_T$ from L_2 ball arrive online.

At every round, the algorithm has to assign coefficients $\alpha_i = +1 \; or \; -1$.

Goal: Keep the discrepancy low at every time step,

$$\left\| \sum_{i=1}^{t} \alpha_i v_i \right\|_\infty$$

Worst-case adaptive adversary [Spencer 77]:

- Lower bound of $\Omega(\sqrt{T})$

Average Case (Uniform distribution, fixed distribution, oblivious adversary) [Bansal-Spencer 19, Bansal-Jiang-Singla-Sinha 20, Alweiss-Liu-Sawhney 20, …]: $polylog(nT)$

Theorem [H., Roughgarden, Shetty '21]: $polylog(nT/\sigma)$ for smoothed adaptive + isotropic
Apply the recipe and exploit the anti-concentration properties of BJSS20 (that's monotone in $X_1, \ldots, X_T \subseteq Z_1 \ldots, Z_{T/\sigma}$).

Conclusion

There are more use cases for this recipe:

- HRS21: Online learning, online discrepancy, data-driven algorithm design
- HRS20: Differential privacy using a different technique.

Smoothed analysis beyond runtime analysis, understanding the fundamental information theoretic limits of learnability.

Open problems:

- Most reduction go through computationally efficient reduction
- Except for online learning and differential privacy.

→ Is there a computational and statistical gap in smoothed analysis? As there is in the worst-case setting.