Learning and Incentives

Nika Haghtalab

UC Berkeley

Learning with Strategic Interactions

Nika Haghtalab
UC Berkeley

Learning and Learnability

One of the goals of theory of ML:

“What concepts can be learned from data, and with how many observations?”

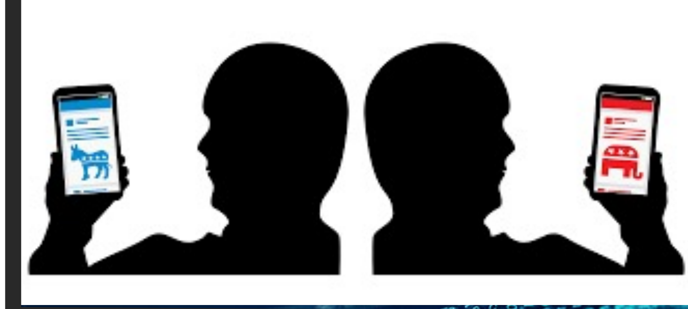
An example of concept: “Familiar object, such as a table”. [Valiant ‘84]

Most basic learning setting: Distribution over objects that remain the same.

Learnability for Today's World

Learning Algorithms

Environment



Learnability

Q1. What concepts can be learned in presence of strategic and adversarial behavior?

→ Lessons for todays world from decade of efforts for understanding.

Q2. How to design learning for strategic and adversarial environment?

→Computational overheads

→Principals on how to use/not use data in strategic environments.

Q3. How can we design collaborative environment that encourage learner participation?

→ Incentives of learning algorithms and data providers

→Deliver the optimal learning algorithms for agents and the society.

Q4. Generally, how do these learning paradigms relate to one another?

Tutorial Overview

Wednesday

1. Adversarial Interaction
   - Offline, Online adversarial learning, and Zero-sum Games
   - Beyond the worst-case adversaries
   - Computational Challenges
2. General Strategic Interactions
   - General-sum games and Stackelberg concept
   - Learning and Stackelberg equilibria
   - Learning in presence of non-myopic agents

Thursday

3. Collaborative Interactions
   - Models of data sharing for learning
   - Average vs. Per-Agent learning guarantees
   - Individual Rationality and Equilibria

Adversarial Interactions

- Offline (Stochastic) Learning
- Online (Adversarial) Learning
- Zero-Sum Games and Solution Concepts
- Nicer than worst-case adversaries
- Computational aspects

Adversarial Interactions

- Offline (Stochastic) Learning
- Online (Adversarial) Learning
- Zero-Sum Games and Solution Concepts
- Nicer than worst-case adversaries
- Computational aspects

Stochastic (Offline) Settings

Usage Example:

Learning to detect natural phenomenon or objects, e.g., trees, animals, etc.

Distributions of the images and concept remains the same over time.

No reaction from the object or environment!

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

Learner learns 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

Formal Setup: Stochastic setting

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 in H.

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

Emphasis on i.i.d

Learner observes (x_t, y_t) and makes a mistake if $f_t(x_t) \neq y_t$.

Goal: Get regret that vanishes as $T \to \infty$

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

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

Alternative Setup: (Stochastic) Offline Learning

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

Learner observes samples and picks **prediction rule** $f: X \to Y$, not necessarily in H.

Goal: How fast does regret vanish as a function of T.

Set of T i.i.d samples $(x_t, y_t) \sim D$

Emphasis on i.i.d

$$\text{Avg. REGRET} = \Pr_D[f(x) \neq y] - \min_{h \in H} \Pr_D[h(x) \neq y] \leq \epsilon$$

Sample Complexity and Regret

$$\text{Avg. Regret} = \sqrt{\frac{C}{T}} \quad \Longleftrightarrow \quad \text{Sample complexity} = \frac{C}{\epsilon^2}$$

What characterizes offline learnability?

VC dimension: largest d where there is a submatrix of d columns and 2^d unique rows.

$x \in X$

$h \in H$

$H \backslash X$	x_1	x_2	x_3	…	
h_1	-1	-1	1		-1
h_2	1	-1	-1		1
h_3	-1	1	-1		1
h_4	1	1	1		-1
⋮					

Characterization of Offline Learnability

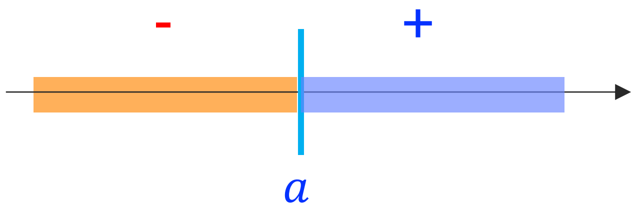
For any H, optimal sample complexity (uniformly over all D) is

Sample complexity = $\tilde{\Theta}(\mathrm{VCD}(H)/\epsilon^2)$ Avg. Regret = $\tilde{\Theta}\left(\sqrt{VCD(H)/T}\right)$

[Vapnik-Chervonenkis 71, Blumer-Ehrenfeucht-Haussler-Warmuth 1989]

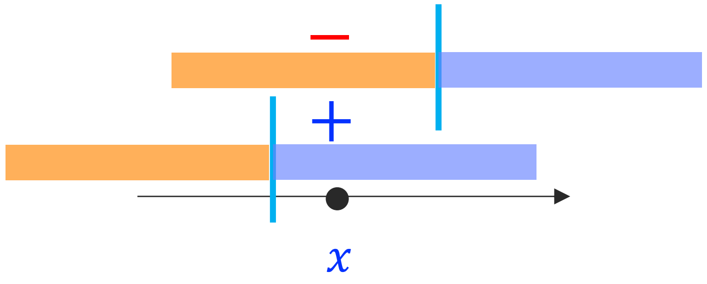
VC Dimension Example

$H = \{h_a(x) = sign(x - a) \mid a \in \mathbb{R}\}$ is the set of ***thresholds on a line***.



What is $\mathrm{VCDim}(H)$ for thresholds on a line? 1

1. Example of a set of size 1 that can be labeled in all 2^1 ways.



2. No set of size 2 can be labeled in all 2^2 ways.

→Can't label the smaller one $+$ and the larger one $-$.

Why VC Dimension?

Characterization of Offline Learnability

For any H, optimal sample complexity (uniformly over all D) is

Sample complexity = $\tilde{\Theta}(\mathrm{VCD}(H)/\epsilon^2)$ Avg. Regret = $\sqrt{VCD(H)/T}$

Why VC dimension lower bounds sample complexity?

Sample set

x_1 x_2 x_3 x_{d-1}

Can be labeled either way

$\Pr[\mathrm{err}] = 1/2$

x_d

Why VC dimension upper bounds sample complexity?

- H finite: concentration and union bound gives

$$\Pr\left[\begin{array}{c}\text{For at least one } h \in H \\ |\text{esimtated } err\ of\ h\ - \text{expected } err\ of\ h| > \epsilon\end{array}\right] \leq \overbrace{|H|}^{\text{Union bound}} \times \overbrace{2\exp(-2m\epsilon^2)}^{\text{Hoeffding}}$$

- H infinite: VC dimension determines the $\boxed{\text{effective size}}$ of the hypothesis class on m points

Adversarial Interactions

Offline (Stochastic) Learning

Online (Adversarial) Learning

Zero-Sum Games and Solution Concepts

Nicer than worst-case adversaries

Computational aspects

Stochastic (Offline) Settings

Usage Examples:

Quality control faces adversarial manipulation of future instances and policies must be updated.

Learning in games (see Costis and Eva's tutorials.)

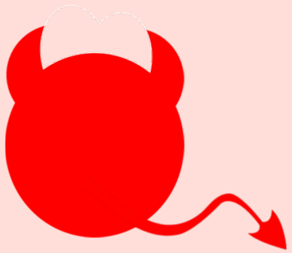
No distributions. Observations evolve in unpredictable or adversarial ways.

Adversarial reactions by the object or environment!

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.

Robust to any adversarial reactions to earlier decisions.



Adversarial Online

Formal Setup: Online vs Stochastic Setting

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 regret that is vanishing as $T \to \infty$.

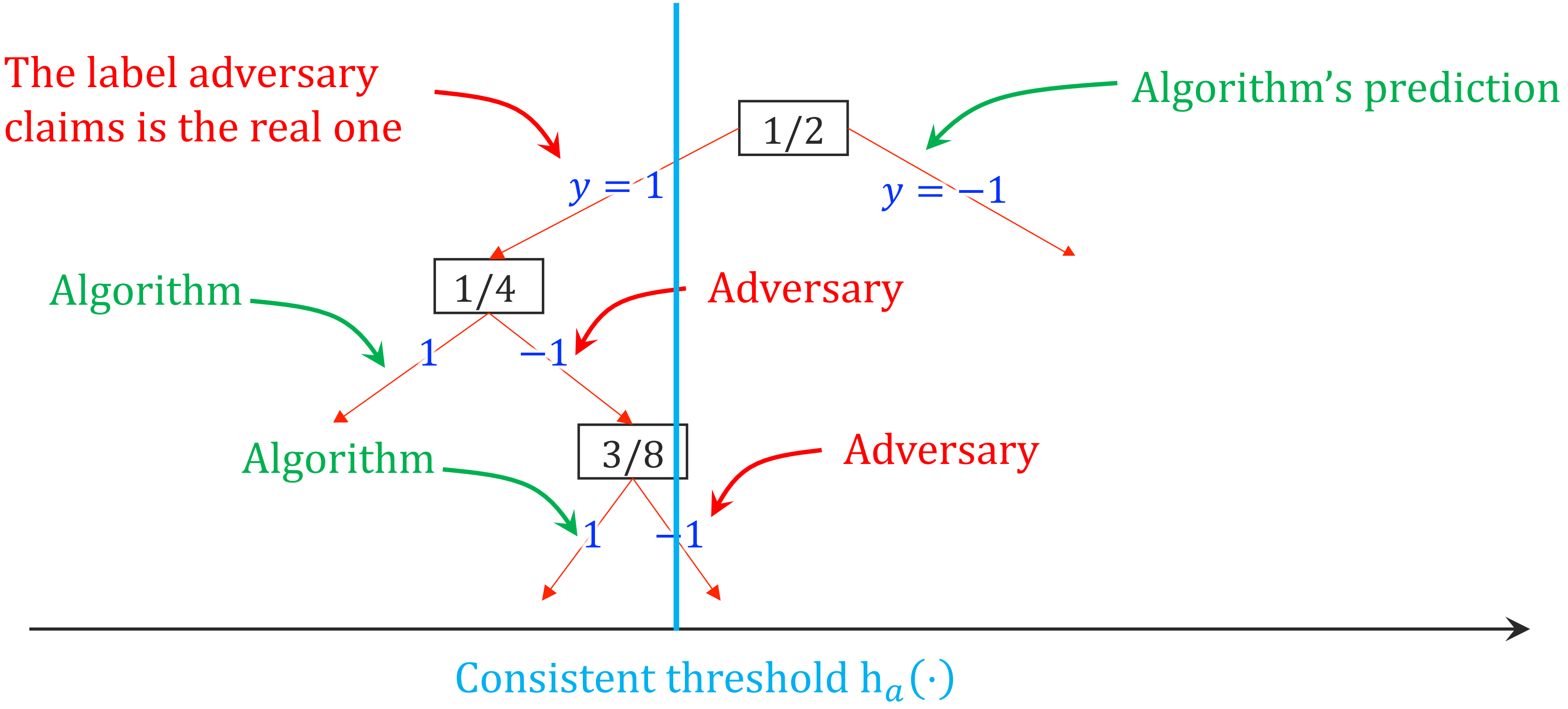
$$\text{Avg. REGRET} \;=\; \frac{1}{T}\sum_{t=1}^{T} 1(f_t(x_t) \neq y_t) \;-\; \min_{h \in H} \frac{1}{T}\sum_{t=1}^{T} 1(h(x_t) \neq y_t)$$

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

An Online Learning Example

Take $H = \{h_a(x) = sign(x - a) \mid a \in \mathbb{R}\}$ is the set of ***thresholds on a line***.

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



Algorithm is forced to err at every round → T mistakes over T instances → Avg Regret O(1).

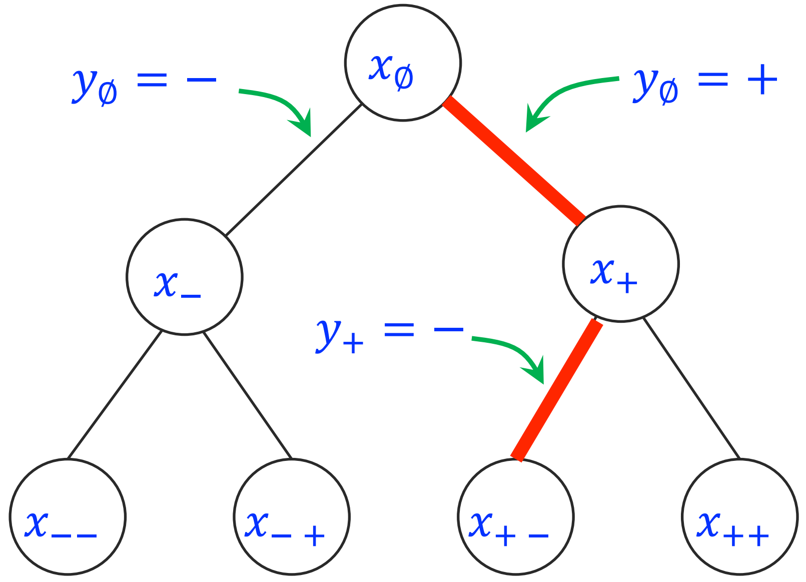
Characterizing Online Learnability

Role of VC dimension:

- Finite VC dimension is not sufficient, because of *thresholds on a line*.
- VC dimension focuses on labeling a set.
- But we need to consider labelings of sequences.

Littlestone Tree: Full decision tree with nodes in X and paths determined by $+$ and $-$ sequences. For every path, there is an $h \in H$ that's consistent with the labels.

Littlestone Dimension: Height of the largest Littlestone tree.

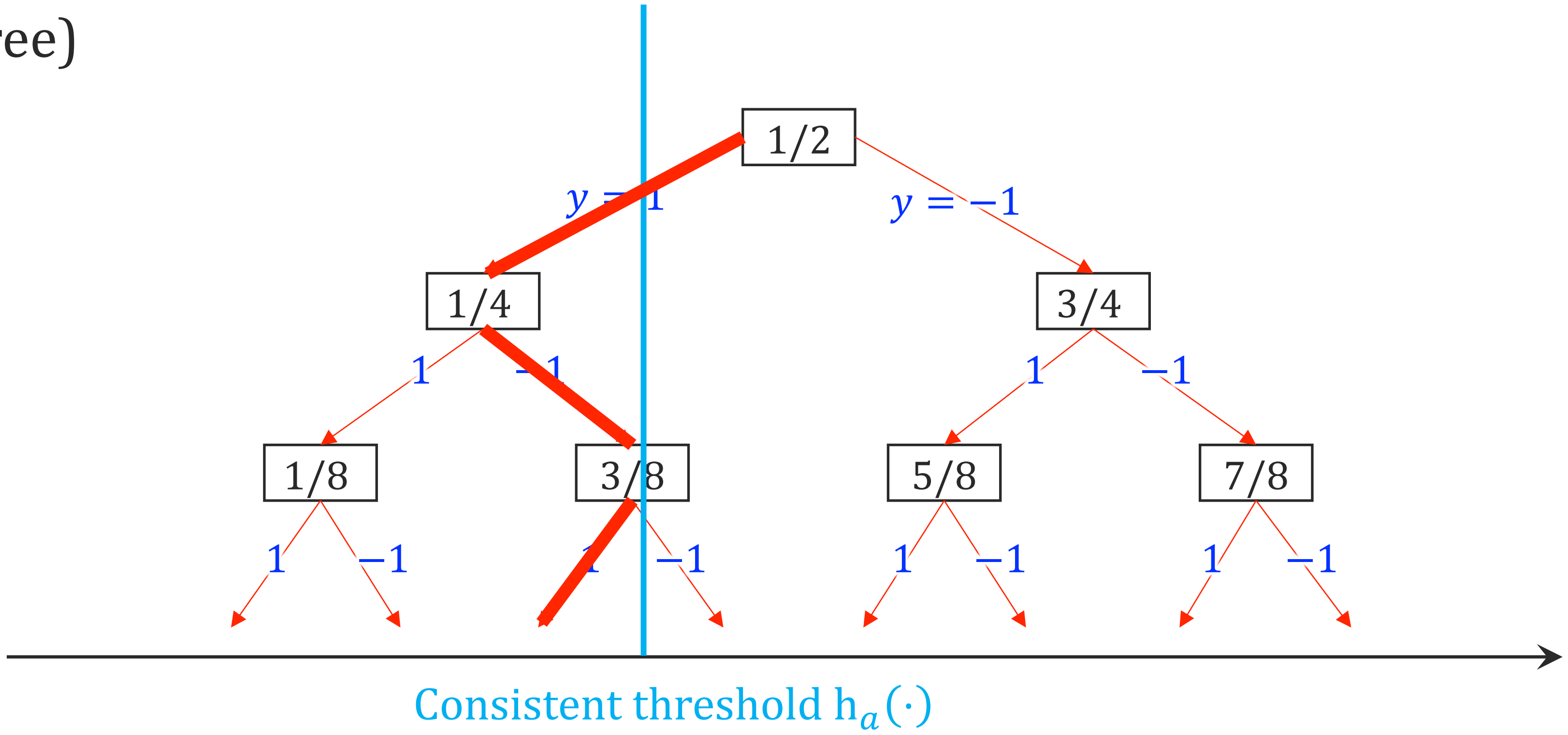


[Littlestone'87]

Recall: Example of Littlestone Dimension

The Littlestone dimension of $H = \{h_a(x) = sign(x - a) \mid a \in \mathbb{R}\}$, the set of ***thresholds on a line***, in infinite.

(Mirror this tree)



Two other Examples of Littlestone Dimension

Small LDim

- Class H where each $h \in H$ assigns +1 label to $\leq d$ points.
- Littlestone dimension is d.

→ We can branch right at most d times.

Large LDim

- Class $H = \{h_a(x) = 1(x \in [a, 2a)) \mid a \in \mathbb{N}\}$.
- Littlestone dimension is ∞.

→ For any d, the H in range of $[2^d, 2^{d+1}]$ includes the set of all ***thresholds.***

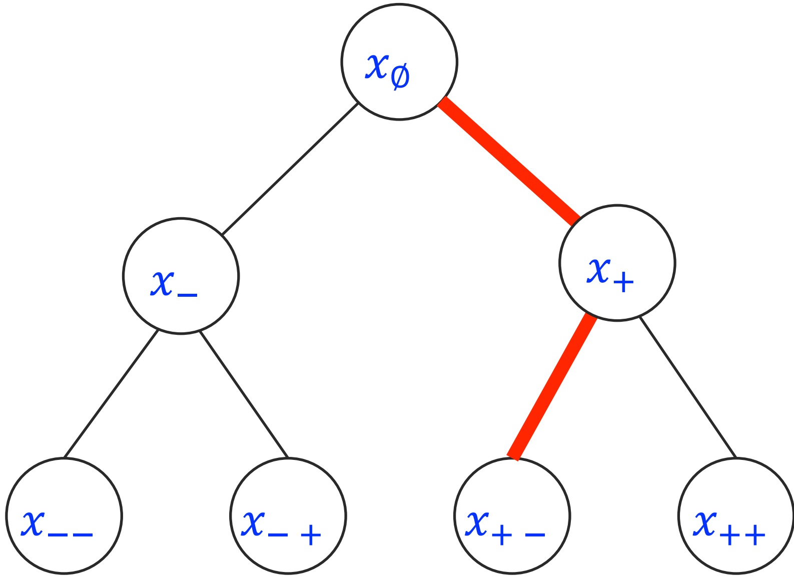
Characterization of Online Learnability

Characterization of Online Learnability

For any H, the optimal bound on average regret is $\tilde{\Theta}\left(\sqrt{\frac{Ldim(H)}{T}}\right)$

Why Littlestone dimension lower bounds regret?

- Adversary picks sequence (x, y)s for a uniformly random path.
- Learner makes a mistake with prob 0.5 per round.
- But a perfect classifier exists, so average regret is 0.5



More formally,

- Repeat each x, $\frac{T}{d}$ times with random labels.
- There is a classifier that beats the standard deviation, but alg gets 0.5.

[Ban-David, Pal, and Shalev-Shwartz'09]

Algorithms based on Littlestone Dimension

Littlestone trees result in an inductive algorithm.

Easy case: Say, the best classifier in hindsight has error 0**.**

- Idea: Keep track of hypothesis that haven't made a mistake so far.
- Make a prediction, so that if it were wrong the **prediction complexity** of the remaining set of classifiers is small.
- What is **prediction complexity**? **Littlestone dimension.**

Standard Optimal Algorithm:

- H_t is the set of classifiers that agree with $(x_1, y_1), \dots, (x_t, y_t)$.
- On x_{t+1} guess label,

$$\hat{y}_{t+1} = \max_{y} LDim\,(\text{Subset of } H_t \textit{ that agrees with } (x_{t+1}, y))$$

[Littlestone'87, Ban-David, Pal, and Shalev-Shwartz'09]

Adversarial Interactions

- Offline (Stochastic) Learning
- Online (Adversarial) Learning
- **Zero-Sum Games and Solution Concepts**
- Nicer than worst-case adversaries
- Computational aspects

(Zero-sum) Games

Usage Examples:

Most two-player board/card games.

Competition between two rival firms, splitting the market share.

Actions are played by self-interested agents in a win-lose game.

Each player takes some actions.

Equilibrium, if neither can improve their position.

Equilibria

Two player Games

Players: Player 1 and 2

Strategies: Sets of actions X, Y

Payoffs: When 1 plays x and 2 plays y.

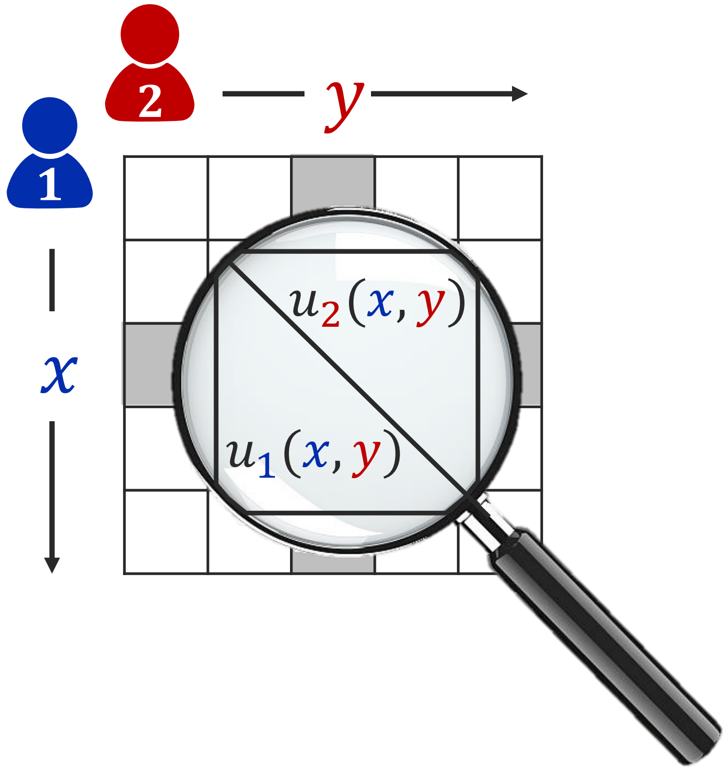
1's payoff : $u_1(x, y)$ 2's payoff : $u_2(x, y)$

Zero-sum games: focus of this section

$$-u_1(x, y) = u_2(x, y)$$

We'll call one of the loss and one gain/utility

$$\ell(x, y) = -u_1(x, y) \quad \text{(in this section)}$$



Solution Concepts

<u>Mixed Strategies:</u> picks $P \in \Delta(X)$ and picks $Q \in \Delta(Y)$. $\mathrm{L}(P, Q)$ is expected loss.

MinMax value

$$\min_{P} \max_{Q} \mathrm{L}(P, Q)$$

(player 1 goes first)

MaxMin value

$$\max_{Q} \min_{P} \mathrm{L}(P, Q)$$

(player 2 goes first)

(P, Q) is a **Nash equilibrium** if can't improve their utility by unilaterally changing P, and can't improve their utility by changing Q.

Von Neumann's MinMax Theorm

MinMax value = MaxMin value (= Mixed Nash Equilibrium payoff)

Under some conditions, e.g., $\Delta(X)$ and $\Delta(Y)$ compact,

Why does MinMax Theorem hold?

1. Easy to see: Whoever goes second does a better job (minimizing or maximizing)

$$\min_P \max_Q \mathrm{L}(P, Q) \geq \max_Q \min_P \mathrm{L}(P, Q)$$

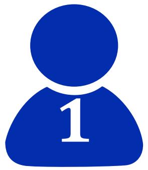
MinMax through online learning

[Freund-Schapire'96]

Online learnability and **MinMax** are about interactions with an adversary.

2. Interesting: One player plays no-regret, the other best responds

$$\min_P \max_Q \mathrm{L}(P, Q) \leq \max_Q \min_P \mathrm{L}(P, Q) + Avg.Regret$$



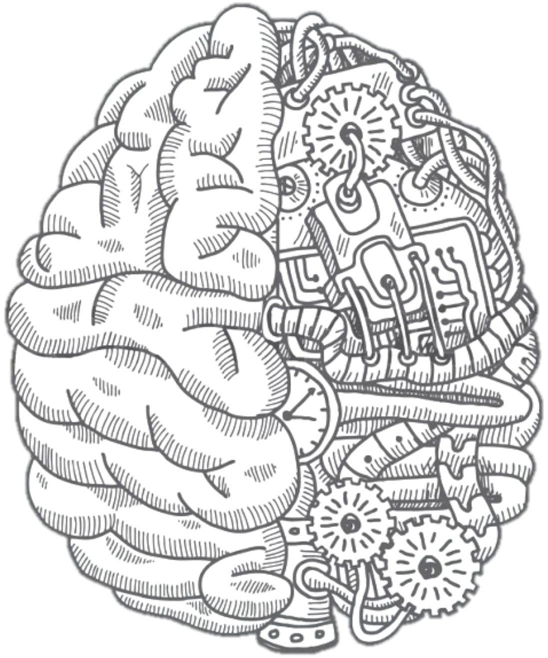
$$\bar{P} = \frac{1}{T}\sum P_t \qquad \bar{Q} = \frac{1}{T}\sum Q_t$$

$$\frac{1}{T}\sum L(P_t, Q_t) - \min_P \frac{1}{T}\sum L(P, Q_t)$$

$$Q_t = \max_Q L(P_t, Q)$$

Question

What is the role of online/offline learnability characterization on equilibrium definitions.



The Role of Littlestone Dimension

Is online learnability a sufficient condition for MinMax to hold?

Subtlety:

- Games require the mixed strategy to be supported on the predefined action set.
- Online learning doesn't necessarily (can be "improper").

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 regret that is vanishing as $T \to \infty$.

"Proper" learning algorithm if $f_t \in H$

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

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

The Role of Littlestone Dimension

Is online learnability a sufficient condition for MinMax to hold?

Subtlety:

- Games require the mixed strategy to be supported on the predefined actions.
- Online learning doesn't necessarily (can be "improper").
    - If "proper", then a randomized learner's choice of X, is equivalent to mixed strategy.
- The way Standard Optimal Algorithm (SOA) was defined, "properness" not guaranteed.

Proper Standard Optimal Algorithm

Simple Optimal Algorithm can be implemented as a "proper" learning algorithm and finite support, giving regret $\widetilde{\Theta}\left(\sqrt{\frac{Ldim}{T}}\right)$.

Finite $Ldim$ is sufficient for MinMax to hold.

[Hanneke-Livni-Moran'21]

Is finiteness of Littlestone Dimension necessary?

Surprisingly not! Recall

Infinite LDim

- Class $H = \{h_a(x) = 1(x \in [a, 2a)) \mid a \in \mathbb{N}\}$.

  → For any d, the H in range of $[2^d, 2^{d+1}]$ includes the set of all ***thresholds.***

The minmax and maxmin values are both tending to 0.

Example from Steve Hanneke

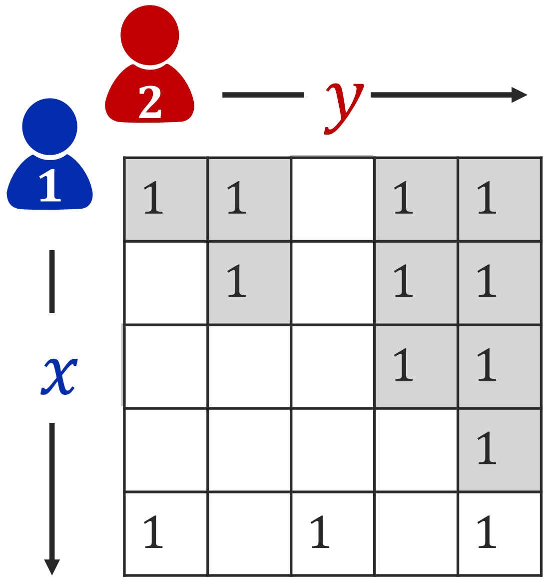
What characterizes MinMax?

Related but not the same thing as finiteness of Littlestone dimension.

Minmax characterization

For a 0/1 game matrix, minmax theorem holds if and only if the game has **no infinite subgame** that can be **rearranged to a triangular matrix**.

[Hanneke-Livni-Moran’21]



1	1		1	1
	1		1	1
			1	1
				1
1		1		1

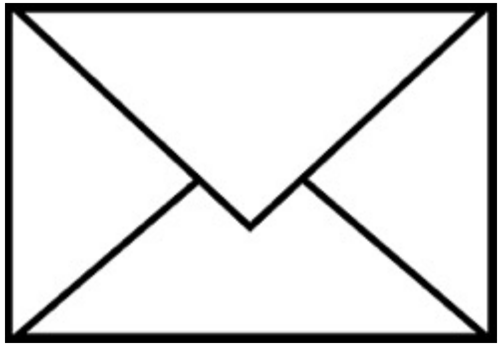
Subtlety:

- Littlestone dimension may be infinite, because for each d there is a Littlestone tree of height d. Even if no single tree could be grown infinitely.
- In that case, no single triangular subgame of infinite size might exist.

Important Message

Learnability is very sensitive to the adversarial assumptions

Offline learning	$\tilde{\Theta}(\sqrt{\mathrm{VCDim(H)}\ T})$
Online Learning	$\tilde{\Theta}(\sqrt{\mathrm{Ldim(H)}\ T})$
Zero-sum Games (Minmax theorem)	Largest triangular subgame



Real Valued Learning and Games

Real-valued learning problems and games:

Offline and online learnability characterizations are well-understood. Rademacher complexity [Bartlett and Mendelson'03], psuedo-dimension [Pollard'84], sequential Rademacher complexity [Sridharan, Rakhlin, and Tewari'15],, etc.

For Minmax, sufficient conditions via fat-shattering [Daskalakis-Golowich 21]. A characterization is open.

Adversarial Interactions

Offline (Stochastic) Learning

Online (Adversarial) Learning

Zero-Sum Games and Solution Concepts

Nicer than worst-case adversaries

Computational aspects

Statistical Guarantees

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

Learner learns a function using the data

Successful if it gets good performance over the underlying distribution.

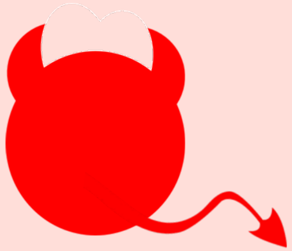
Not concerned with robustness or what happens if the world were to change.

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.

Robust to any adversarial reactions to earlier decisions.



Adversarial Online

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.

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**.

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

Smoothed Analysis: Basic Idea

Idea [Spielman & Teng 01]:

- Adversary chooses an instance, 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
- Simplex 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)

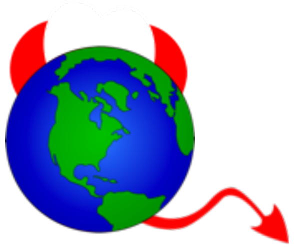
- Even what is “learnable” depends on the model of the adversary.
- Fundamental application of smoothed analysis

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.



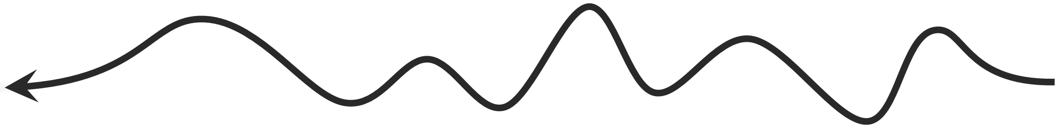
Adversary picks a σ-smooth distribution D_t knowing the history for $1, \ldots, t-1$ and the algorithm

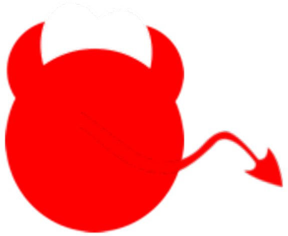
σ-smooth distribution: max density is $\leq \frac{1}{\sigma} \times$uniform density on X

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

[Sridharan-Rakhlin-Tewari'11]

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





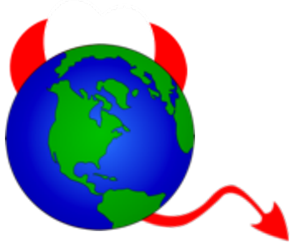
Adversary picks an instance $(\bar{x}_t, \bar{y}_t)$.

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 $1, \ldots, t-1$ and the algorithm

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

Goal: Vanishing average regret

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

Recall

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

Interpreted as an impossibility result, because VCDim $\ll$ Ldim
→ For simple classes, Ldim $= \infty$ but VCDim $= 1$.

Smoothed Analysis for online learning

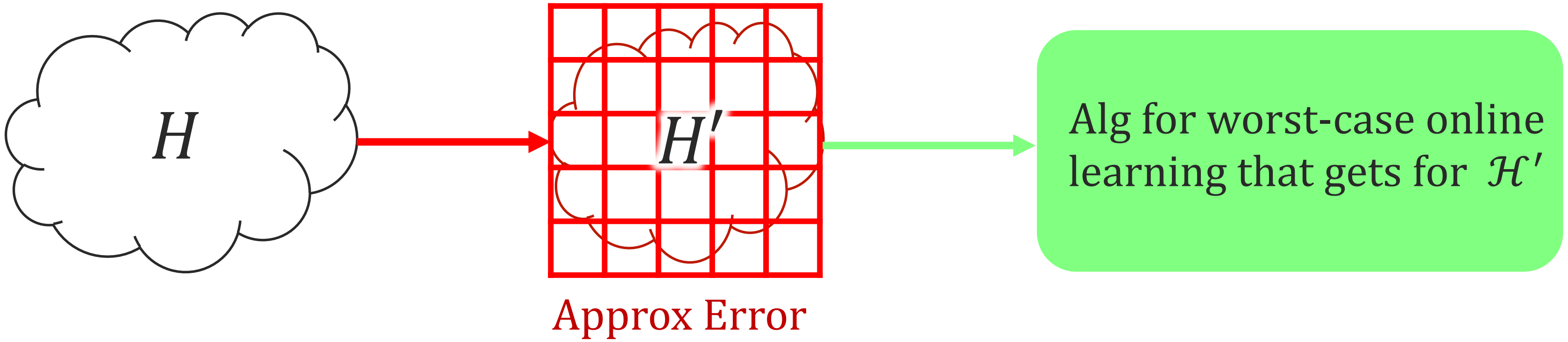
In presence of Adaptive but Smooth Adversaries the regret is $\tilde{O}\left(\sqrt{\mathrm{VCDim}(H)\ T \ln(1/\sigma)}\right)$

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

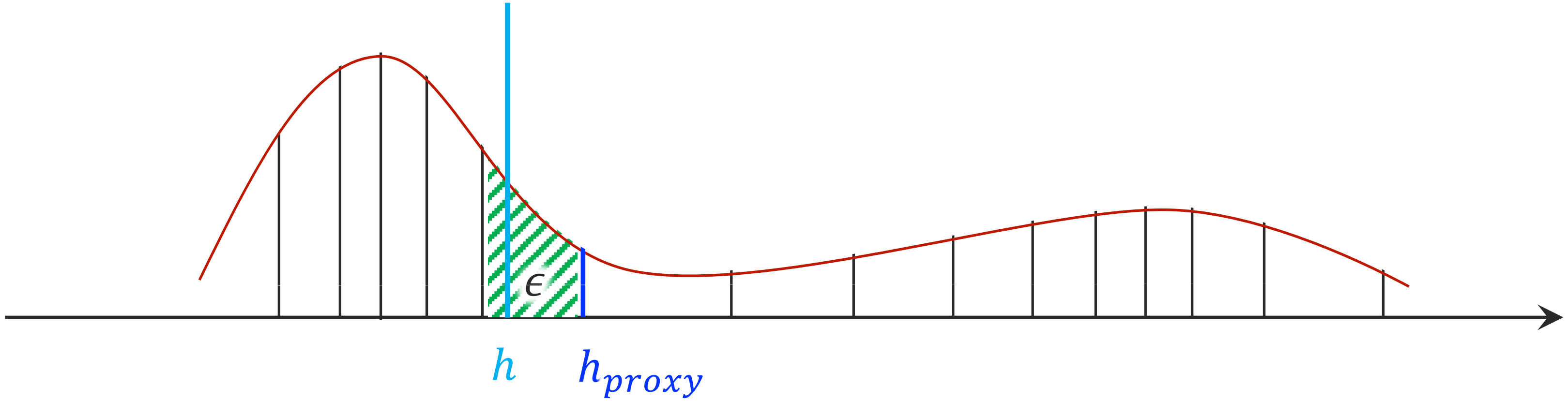
[**H.**, Roughgarden, Shetty'21]

Why did the Stochastic Case Work?

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

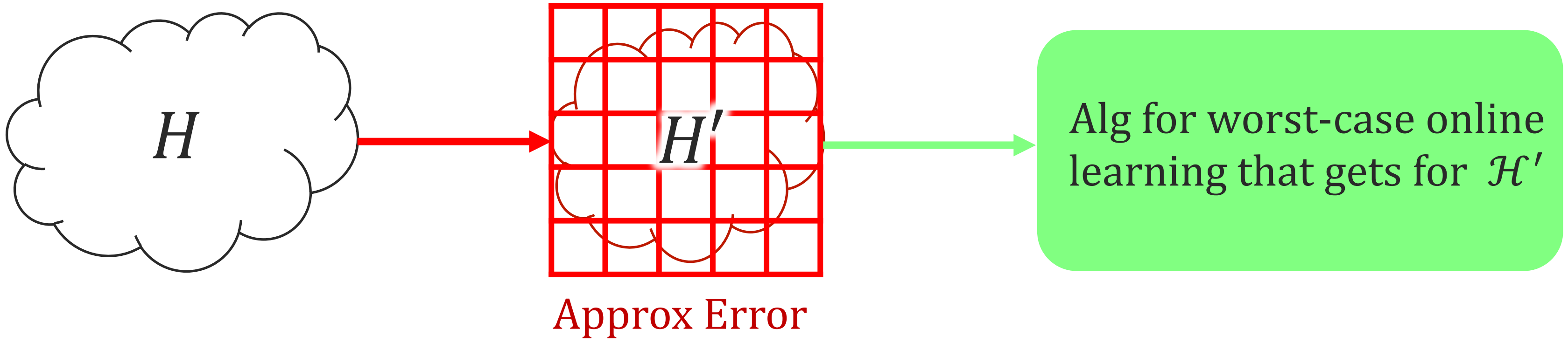


The Net: For each $h \in H$ there is $h_{proxy} \in H'$, where $\mathbb{E}[h \,\Delta h_{proxy}] \leq \epsilon$ is small.



Why did the Stochastic Case Work?

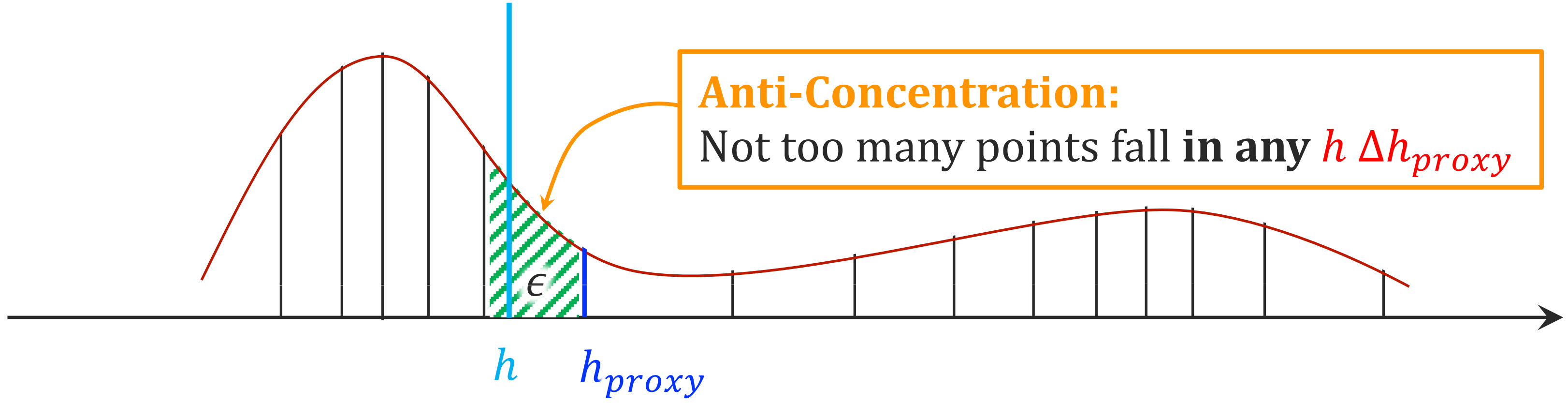
We could approximate H that's potentially infinite, with a finite H′.



The Net: For each $h \in H$ there is $h_{proxy} \in H'$, where $\mathbb{E}\left[h \,\Delta h_{proxy}\right] \le \epsilon$ is small.

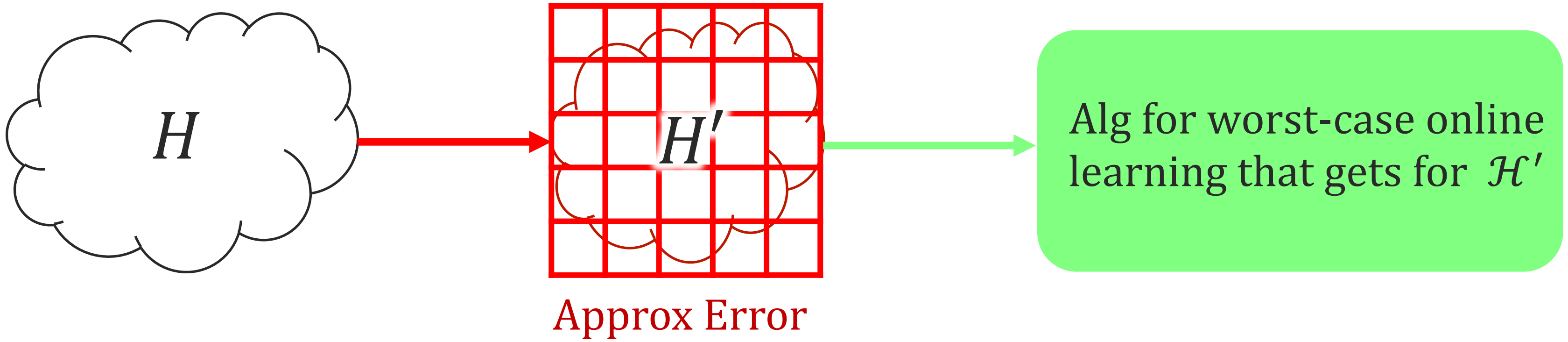
Approx Error is small: Performance of every $h \in H$ is close to the corresponding $h_{proxy} \in H'$

Infinitely many $h \,\Delta h_{proxy}$: i.i.d instances and finite VC dimension bounds this.



What went wrong for the online case?

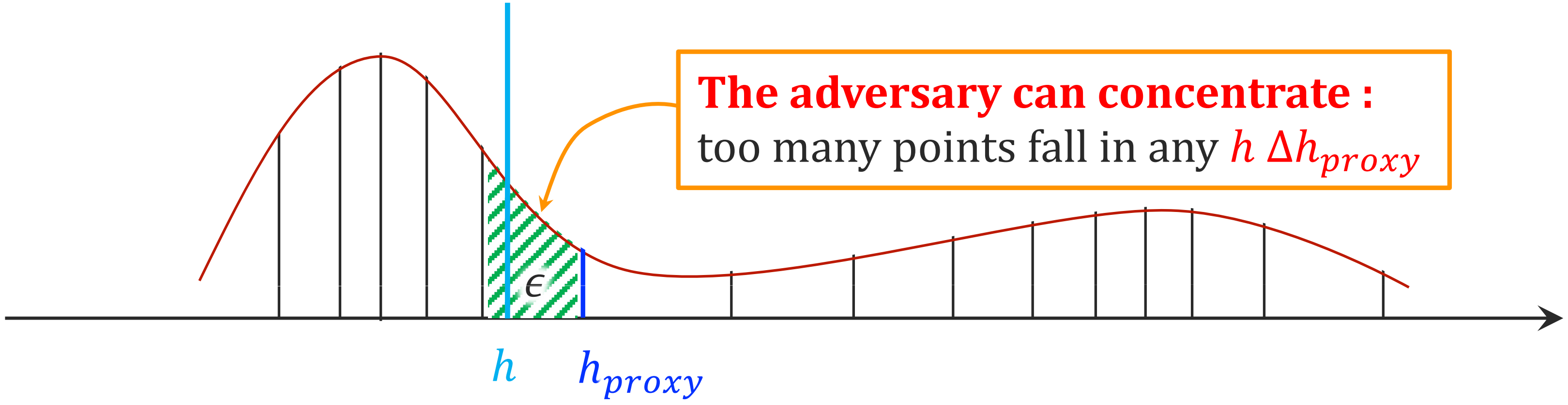
We could approximate H that's potentially infinite, with a finite H′.



The Net: For each $h \in H$ there is $h_{proxy} \in H'$, where $\mathbb{E}[h \,\Delta h_{proxy}] \leq \epsilon$ is small.

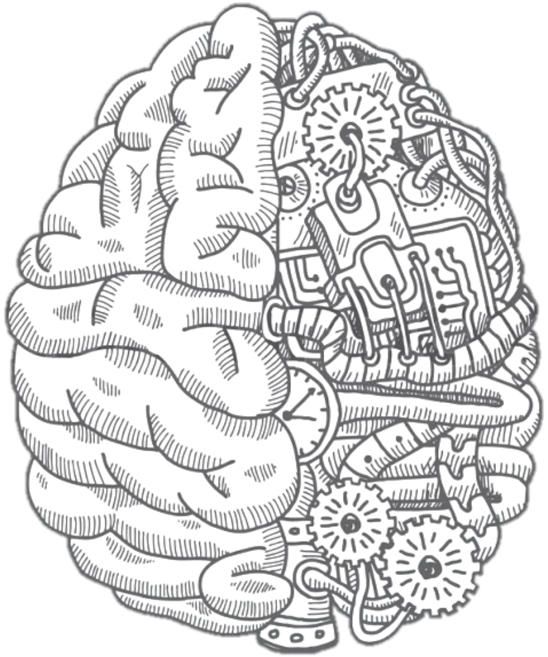
Approx Error is small: Performance of every $h \in H$ is close to the corresponding $h_{proxy} \in H'$

Infinitely many $h \,\Delta h_{proxy}$: ~~i.i.d instances~~ and finite VC dimension bounds this.



Broad Question

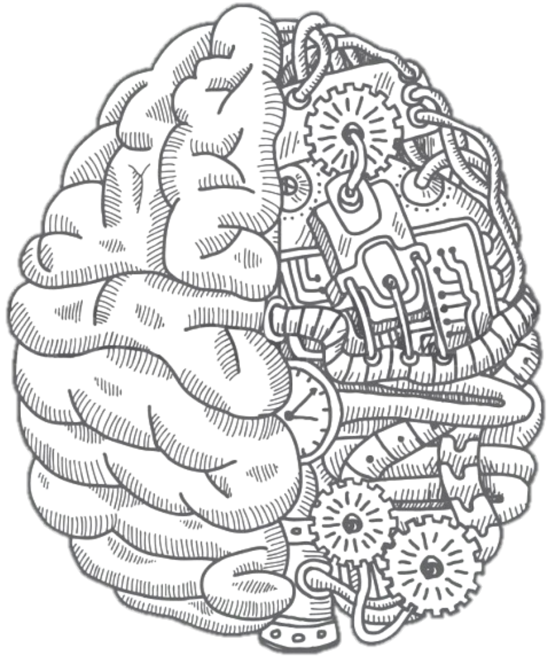
How do we preserve anti-concentration when a sequence of smooth distributions are adaptively chosen?



Challenge

Each σ-smooth distribution is anti-concentrated.

The challenge is correlations between these smooth distributions.



Couple Adaptive Smoothness with Uniformity

Probability Couplings: Given distributions X and Z.

- A joint distribution on $X \times Z$, such that there is a "nice property" between the draws (x, z).
- Couple a sequence of smooth distributions with draws from a uniform distribution.

Coupling Theorem: For any adaptive sequence of T distributions, there is a coupling between:

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

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

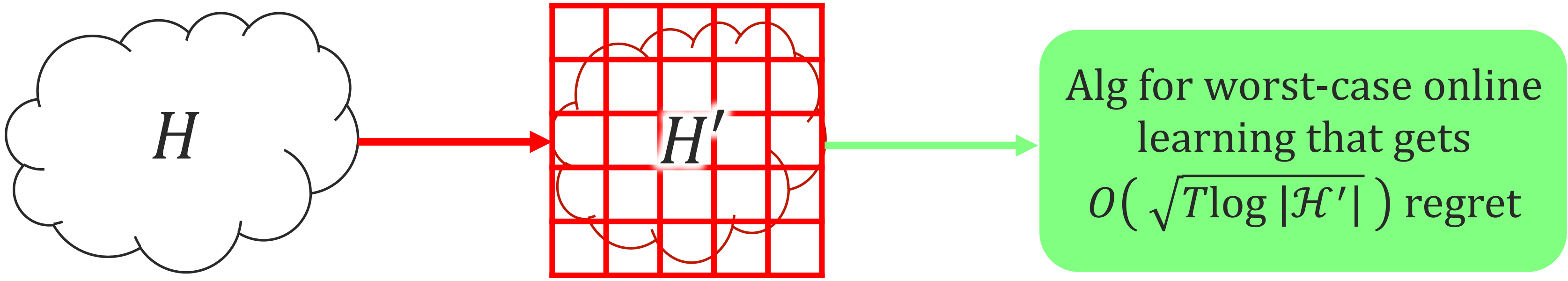
- We want to say that no $h \, \Delta h_{proxy}$ includes too many $X_1, \ldots, X_T$.
- Sufficient to say no $h \, \Delta h_{proxy}$ includes too many $Z_1 \ldots, Z_{Tk}$.
- $Z_1 \ldots, Z_{Tk}$ are i.i.d and guaranteed to be scattered.

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

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



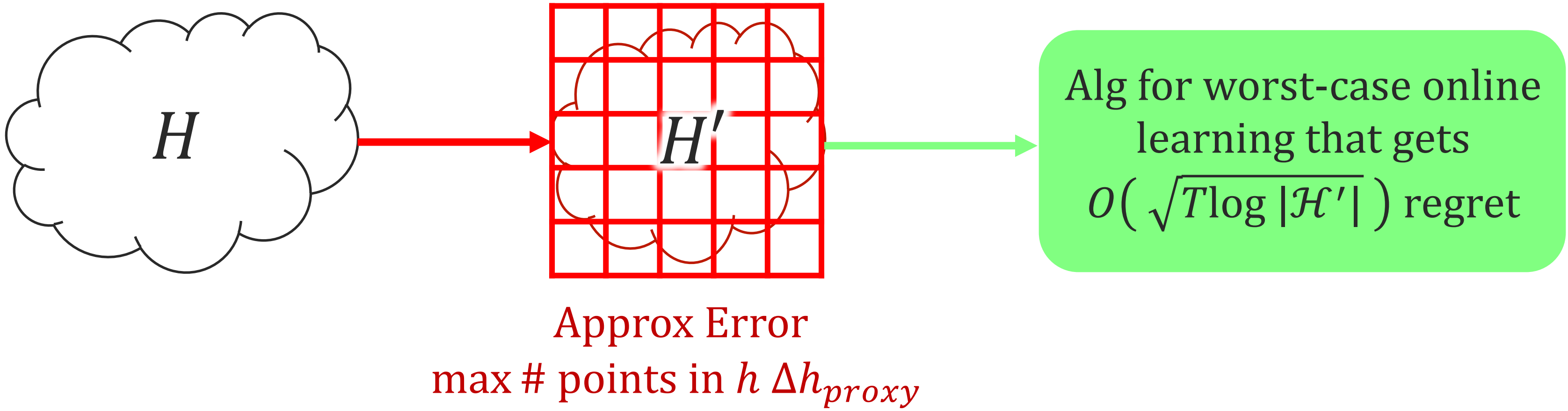
Approx Error
max # points in $h\ \Delta h_{proxy}$

How do we select H′?

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

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

Overview of the Main Results



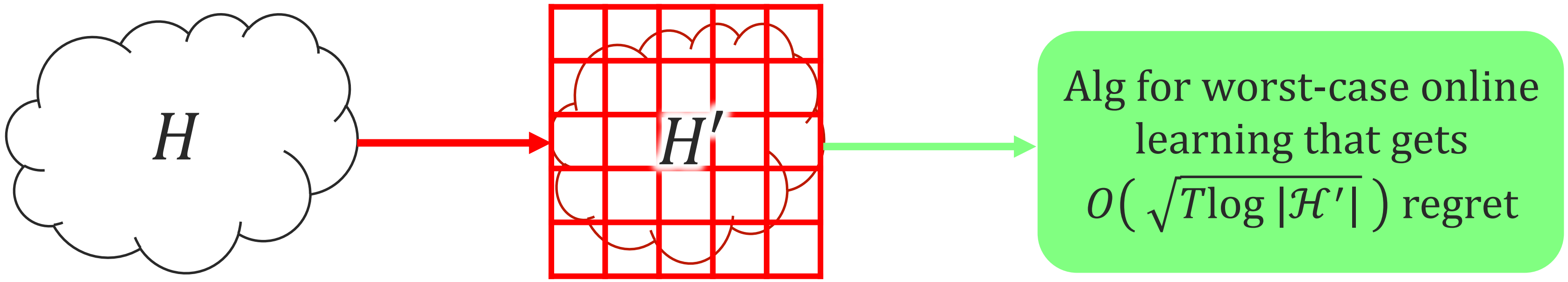
Step 1: We got that $\mathbb{E}_D\left[\#\text{points in } h\ \Delta h_{proxy}\right] \leq T\epsilon/\sigma$.

Step 2: Apply the coupling

$$\underset{h\in H}{\max} \overset{\text{Approx Error}}{\underset{\text{fall in } h\ \Delta h_{proxy}}{\#\text{ points} \sim D_1, \ldots \mathrm{D_T}}} \leq \underset{h\in H}{\max} \overset{\text{Approx Error}}{\underset{\text{fall in } h\ \Delta h_{proxy}}{\#\text{ points} \sim Unif}}$$

"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



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

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

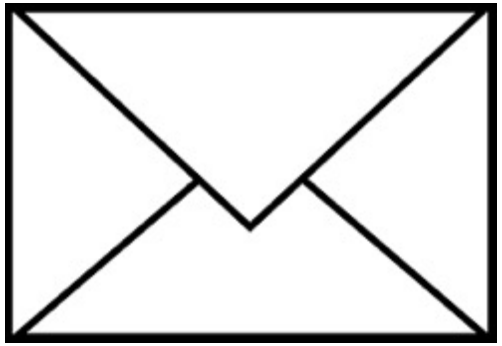
Step 3: Bound the Approx Error for the uniform distribution.

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.

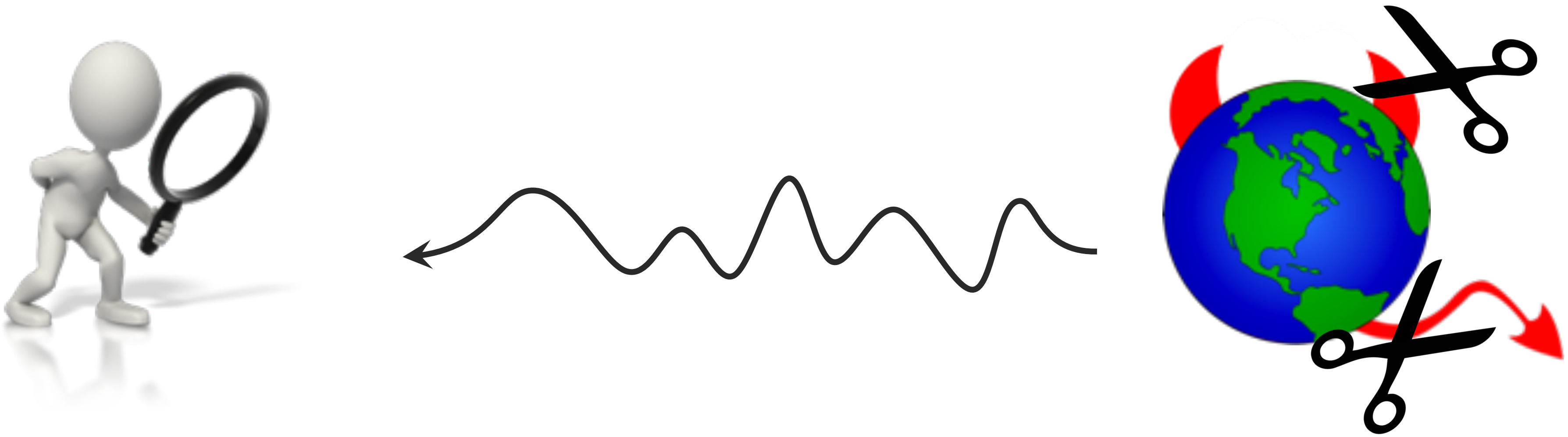
Classical algorithms and analysis from the stochastic case can be lifted and be use with smoothed adaptive adversaries



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.



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/σ 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.

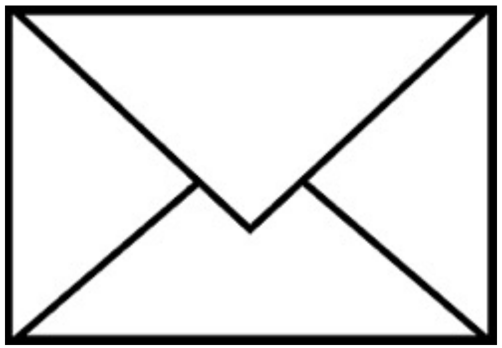
Applications

Applications to other problems where Minmax and repeated games have influenced.

- Online Learning (in the talk)
- Online Discrepancy Minimization
- Data Driven Algorithm Design
- Differential Privacy (Using slightly simpler techniques H., Roughgarden, and Shetty 20)

Important Message

Learnability's sensitive dependence on adversarial assumptions is partly "brittle" and won't be observed in the nature.



Beyond worst-case analysis need for reevaluating statistical characterization.

Adversarial Interactions

Offline (Stochastic) Learning

Online (Adversarial) Learning

Zero-Sum Games and Solution Concepts

Nicer than worst-case adversaries

Computational aspects

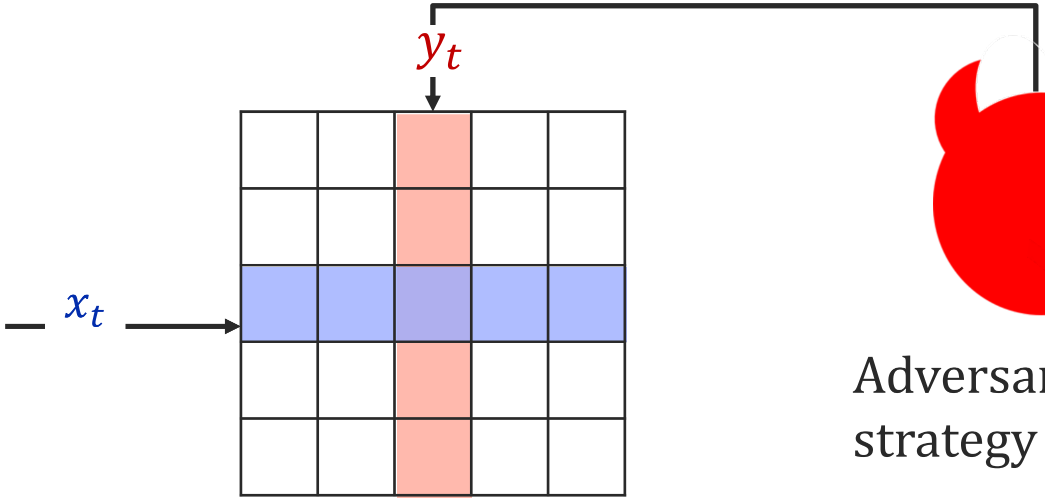
More on Computational Aspects Tomorrow

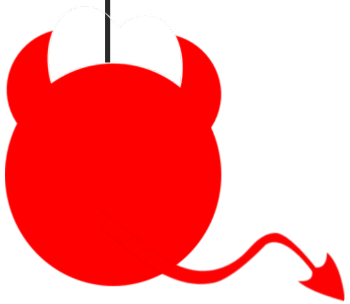
Up to now, we have established strict ordering between the statistical difficulty of learning tasks.

- Algorithmically, is computation against an adaptive adversary strictly harder than a stochastic ones? [also discussed in Costis's tutorial]
- How sensitive are the computational result on the specific adversarial assumptions, e.g., worst-case versus smoothed analysis.
- Elegant framework of game value relaxations of Sridharan, Shamir, Rakhlin'12.
  - →Direct connection between statistical aspects online computation and algorithm design.
- What are combinatorial structures that make efficient online learning possible?

Algorithms for Online Learning

Learner picks a strategy x_t from $\mathcal{X}$ at random





Adversary picks a strategy y_t from $\mathcal{Y}$.

An algorithm for online learning

There is an algorithm with average regret $\boldsymbol{O}\left(\sqrt{\frac{\log|\mathcal{X}|}{T}}\right)$ and runtime $\mathbf{O}(T|\mathcal{X}|)$.

\# time steps

\# learner's actions

Algorithm (Hedge): Start with uniform distribution over $\mathcal{X}$. At each step, adjust up/down probability of each $x \in \mathcal{X}$ based on historical performance.

[Freund & Schapire'95]

Online Computation with Offline Oracles

Part of the difficulty comes from offline computation:

- Even **minimization (or maximization)** is difficult for some problem, e.g., deep networks, non-convex objectives, etc.
- What part of the difficulty should be blamed on existence of adversaries?

Oracle-efficient Online learning

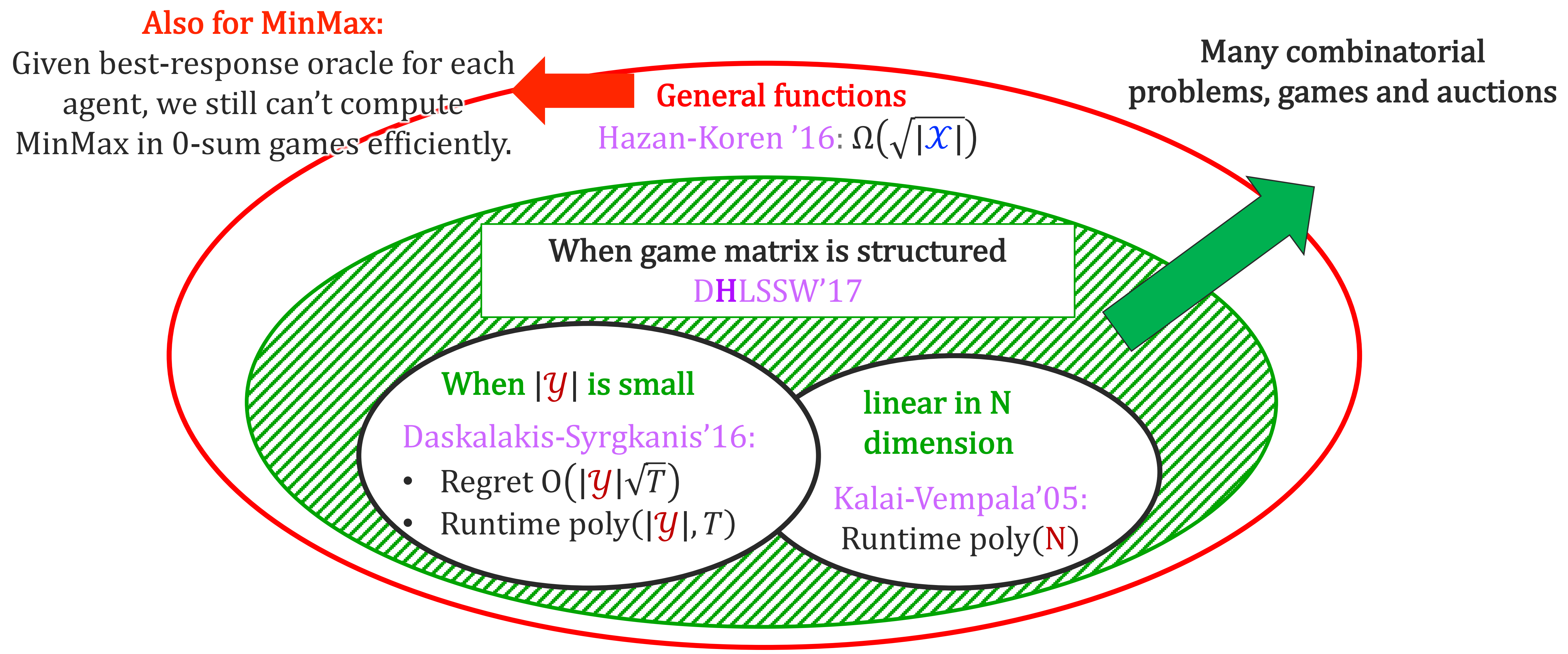
Effective tools for computing **optimal offline optimal** classifiers

$\Downarrow$

Design **online algorithms** for adversarial environments

- **Offline Oracle:** For any $y_1, y_2, \dots, y_t$, compute $\operatorname{argmax}_{x \in \mathcal{X}} \sum_{\tau=1}^{t} u(x, y_\tau)$.

Characterize Online Oracle-Efficient Learnability



No characterization! But sufficient conditions that are easy to find in practice.

Combinatorial Structure

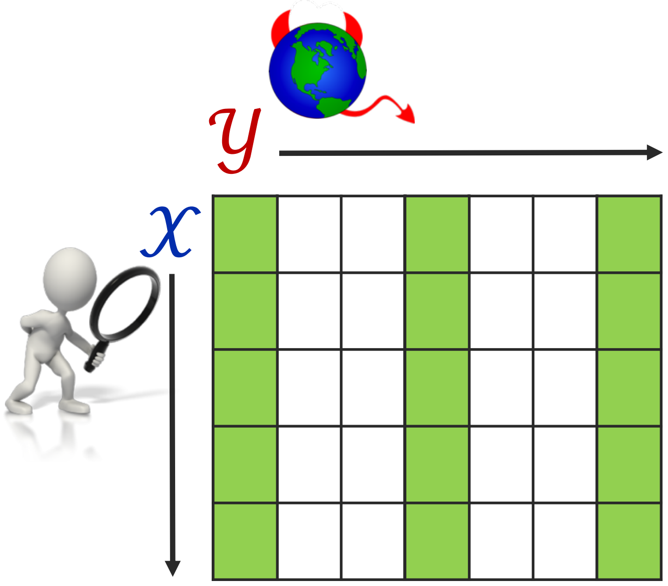
The d-Structure

There exists a set of d adversary pure strategies that sufficiently* distinguish between any two learner pure strategies.

Smallest d, for which there is $y^{(1)} \dots y^{(d)}$, s.t., for all $x \in \mathcal{X}$

$$u\left(x, y^{(i)}\right) \neq u\left(x, y^{(i)}\right) \text{ for some } i \in [d]$$

*Sufficiently = Gap of δ between distinct utilities.



All green rows are still different

Oracle-efficient Online learning

1. There is an oracle-efficient algorithm with regret $O\left(d\sqrt{T}/\delta\right)$.
2. Many problem classes have small d-structures, e.g. most auctions $\mathrm{d} = \mathrm{poly}(\#\text{ bidders})$

[Dudik, **H.** , Luo, Schapire, Syrgkanis, Wortman '17]

Structure of Auctions

For many N-bidder auction classes

There are $poly(N)$ bid profiles that **distinguish** between any two auctions.

In many auctions, outcome depends on a few parameters, e.g., winners, second place in line to winning, and their bid.

Auction 1 $\neq$ Auction 2

$$\boxed{b_1}, b_2, \boxed{b_3}, b_4, \boxed{b_5}, b_6, b_7 \ldots, \boxed{b_N}$$

$$\boxed{b_1}, 0, \boxed{b_3}, 0, \boxed{b_5}, 0, 0 \ldots, \boxed{b_N}$$

Winner in Auction1

Next in line in Auction 1

Winner in Auction 2

Next in line in Auction 2

Bid profiles with 4 non-zero bids distinguish between any two such auction.

Beyond this tutorial

Using offline oracles more broadly in adversarial settings:

- Approximate oracles and approximate regret: Kakade Kalai Liggett'07, Hazan Li Li'18, Garber'17, Niazadeh Golrezaei Wang Susan Badanidiyuru'20, etc.
- Beyond worst-case adversaries
    - For smoothed analysis?
    - Some notions of predictable sequences [Sridharan and Rakhlin'13]
        - Transductive learning (where future instances, but not labels, are known) [Kalai Kakade'06, Cesa-Bianchi Shamir'12]
        - Better regret bounds approaches in minmax [Costis's tutorial]

Tutorial Overview

Wednesday

1. Adversarial Interaction
   - Offline, Online adversarial learning, and Zero-sum Games
   - Beyond the worst-case adversaries
   - Computational Challenges

Thursday

2. General Strategic Interactions
   - General-sum games and Stackelberg concept
   - Learning and Stackelberg equilibria
   - Learning in presence of non-myopic agents
3. Collaborative Interactions
   - Models of data sharing for learning
   - Average vs. Per-Agent learning guarantees
   - Individual Rationality and Equilibria

Adversarial Interactions

General-Sum Games

Computing Stackelberg equilibria

Learning Stackelberg equilibria

Commitment and non-myopic agents

Adversarial Interactions

General-Sum Games

Computing Stackelberg equilibria

Learning Stackelberg equilibria

Commitment and non-myopic agents

General-sum Games

Usage Examples:

Strategic manipulations

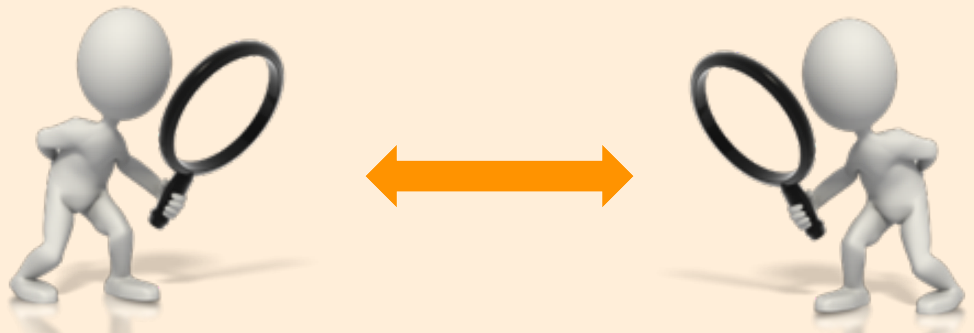
- In ride-sharing apps, drivers and riders manipulate supply and demand achieve better deals shortly after the manipulations.
- In lending, admission, hiring, search, applicants strategic manipulate content to receive favorable outcomes.

Environment responds to the decisions, but strategic manipulation are not meant to hurt others necessarily.

Actions are played by self-interested agents.

Agents may have the ability to commit to strategies, in verifiable ways.

What are the optimal or stable outcome for the agents?



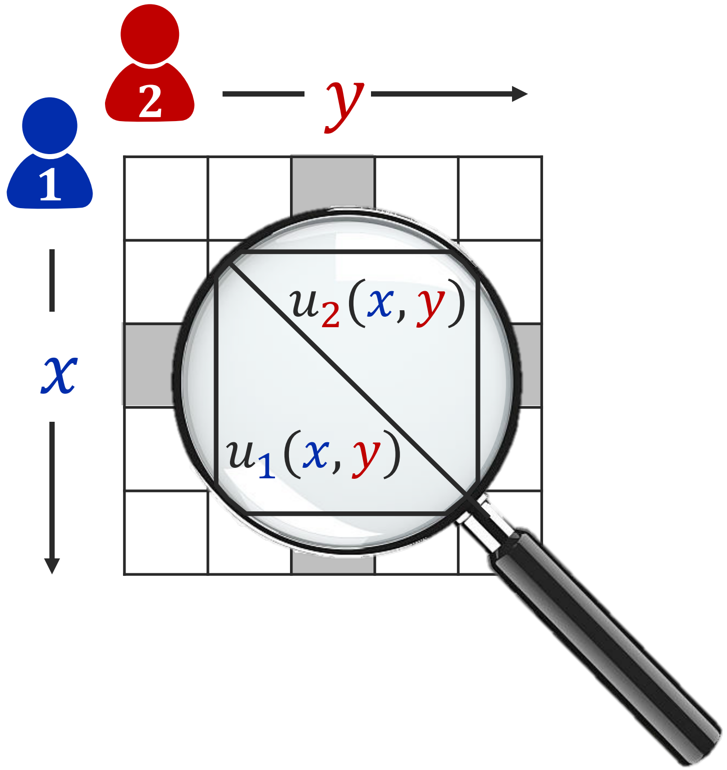
Recall: Two player Games

<u>Players</u>: Player **1** and **2**

<u>Strategies</u>: Sets of actions X, Y

<u>Payoffs</u>: When **1** plays x and **2** plays y.

1's payoff : $u_1(x, y)$ **2**'s payoff : $u_2(x, y)$



~~<u>Zero-sum games:</u> focus of this section~~

$$\cancel{-u_1(x, y) = u_2(x, y)}$$

Von Neumann's MinMax Theorm

~~MinMax value = MaxMin value (= Mixed Nash Equilibrium payoff)~~
~~Under some conditions, e.g., X and Y size or $\Delta(X)$ and $\Delta(Y)$ compact,~~

It Matters Who Goes First

Mixed Strategies: 1 picks $P \in \Delta(X)$ and 2 picks $Q \in \Delta(Y)$.

What is the Nash Equilibrium?

Player 1: Dominant strategy to play U.

Player 2: Will play L as response.

Player 1: +1

What if 1 can commit in a verifiable way? Sequential game

Player 1: Say, commits to playing D.

Player 2: Will play R as response.

Player 1: +2

	2: L	2: R
1: 0.49 U	(1,1)	(3,0)
1: 0.51 D	(0,1)	(2,1)

von Stengel and Zamir' 04

Stackelberg Solution Concept

(Mixed) Stackelberg Optimal Solution

- Player 1 **(leader)** commits to a $P \in \Delta(X)$
- Player 2 **(follower)** best-responds to P,

  → plays $\mathrm{BR(P)} = \mathrm{argmax}_y\ U_2(P, y)$

Leader commits to best $P \in \Delta(X)$ accounting for BR(P)

$$\mathrm{argmax}_{P \in \Delta(X)}\ U_1(P, BR(P))$$

Stackelberg vs Nash Equilibrium

In any general-sum game, leader's (mixed) Stackelberg optimal solution is **weakly advantageous** to player 1's payoff under any Nash equilibrium.

There are games where the inequality is strict.

Pure Stackelberg Solution Concept

(Pure) Stackelberg Optimal Solution

- Player 1 **(leader)** commits to a $x \in X$
- Player 2 **(follower)** best-responds to x,

  → plays $\mathrm{BR}(x) = \mathrm{argmax}_y \, U_2(x, y)$

Leader commits to best $x \in X$ accounting for BR(x)

$$\mathrm{argmax}_{x \in X} \, U_1(x, BR(x))$$

Stackelberg vs Nash Equilibrium

~~In any general-sum game, leader's (mixed) Stackelberg optimal solution is **weakly advantageous** to player 1's payoff under any Nash equilibrium.~~

~~There are games where the inequality is strict.~~

Pure Stackelberg Solution Concept

(Pure) Stackelberg Optimal Solution

- Player 1 **(leader)** commits to a $x \in X$
- Player 2 **(follower)** best-responds to x,

  → plays $\mathrm{BR}(x) = \mathrm{argmax}_y\, U_2(x, y)$

Leader commits to best $x \in X$ accounting for BR(x)

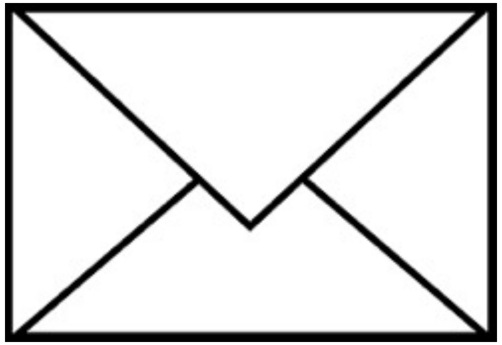
$$\mathrm{argmax}_{x \in X}\, U_1(x, BR(x))$$

In many applications

- X and U_1 and U_2 are highly structured (simplex, convex, concave)
- So pure Stackelberg optimal solution is still advantageous (and easier to compute) than a mixed Nash equilibrium.

Important Message

Commitment
(to a mixed strategy)
is good for you!



Application: Security Games

Security Games:

- Sophisticated attackers target the weakest point.
- Protect targets, so the high value targets are not attacked.

Defender (leader):

- X: set of resources, each able to protect some targets.

Attacker (follower)

- Y: set of targets

$u_1(x, y)$ and $u_2(x, y)$ utilities only depend on whether x protects y.

Mixed strategy: Random protection schedules.

Tambe 2012

Application: Strategic Classification

Strategic Classification:

- Decisions based on observable attributes of applicants.
- Applicants can attempt to change this to improve outcome.

Learner (leader):

- H: set of classifiers.

Distribution of Applicants (distribution of follower)

- x: Initial attributes. Best-response $\mathrm{BR}_x(h)$ is the manipulated attributed.

$u_1(h, \mathrm{BR}_x(h))$ captures accuracy of h on the new instance.
$u_2(h, \mathrm{BR}_x(h))$ accounting for utility of "being admitted" and the manipulation costs.

Pure strategy with a parameterized classifier class.

E.g., Hardt Megiddo, Papadimitriou, Wootters '15

Adversarial Interactions

General-Sum Games

Computing Stackelberg equilibria

Learning Stackelberg equilibria

Commitment and non-myopic agents

The Utility Function

Need to compute

(Mixed) $\quad \text{argmax}_{P\in\Delta(X)}\, U_1(P, BR(P))$, $\quad$ where $\text{BR}(P) = \text{argmax}_y\, U_2(P, y)$

(Pure) $\quad \text{argmax}_{x\in X}\, U_1(x, BR(x))$, $\quad$ where $\text{BR}(x) = \text{argmax}_y\, U_2(x, y)$

In rare cases $U_1(P, BR(P))$ or $U_1(x, BR(x))$ are concave or Lipschitz in the choice of the leader.

E.g., Dong, Roth, Schutzman, Waggoner, Wu '18

Generally, these are not even Lipschitz and at best have piecewise properties.

(mixed) Stackelberg in Finite Games

Need to compute

$$\text{argmax}_{P\in\Delta(X)}\, U_1(P, BR(P)), \qquad \text{where } \mathrm{BR}(P) = \text{argmax}_y\, U_2(P, y)$$

Multiple Linear Program Approach

For finite Stackelberg Games, there is an algorithm with $poly(|X|, |Y|)$.

For each column y, mixed strategies the lead to best-response of y forms a convex polytope.

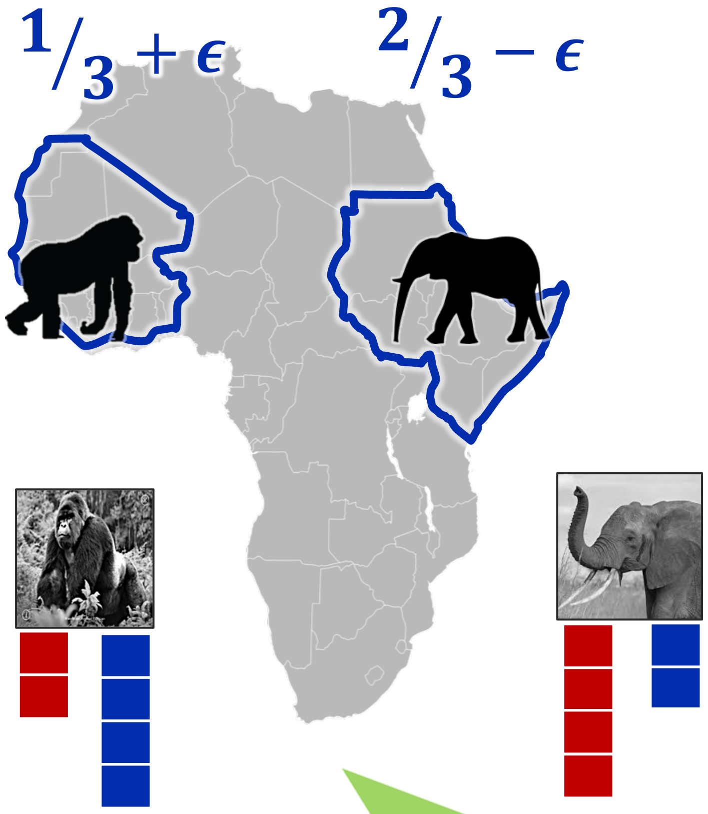
$$P_y = \{P \in \Delta(X) \mid BR(P) = y\}$$

For all y', $U_2(P, y') \le U_2(P, y)$

Compute $P_y^* = \underset{P\in P_y}{\text{argmax}}\, U_1(P, y)$ for each polytope. Take the ones in $y^* = \text{argmax}_y\, U_1\left(P_y^*, y\right)$.

von Stengel and Zamir' 09, Korzhyk Conitzer and Parr'10

Example of the Multiple LP approach



Attacking

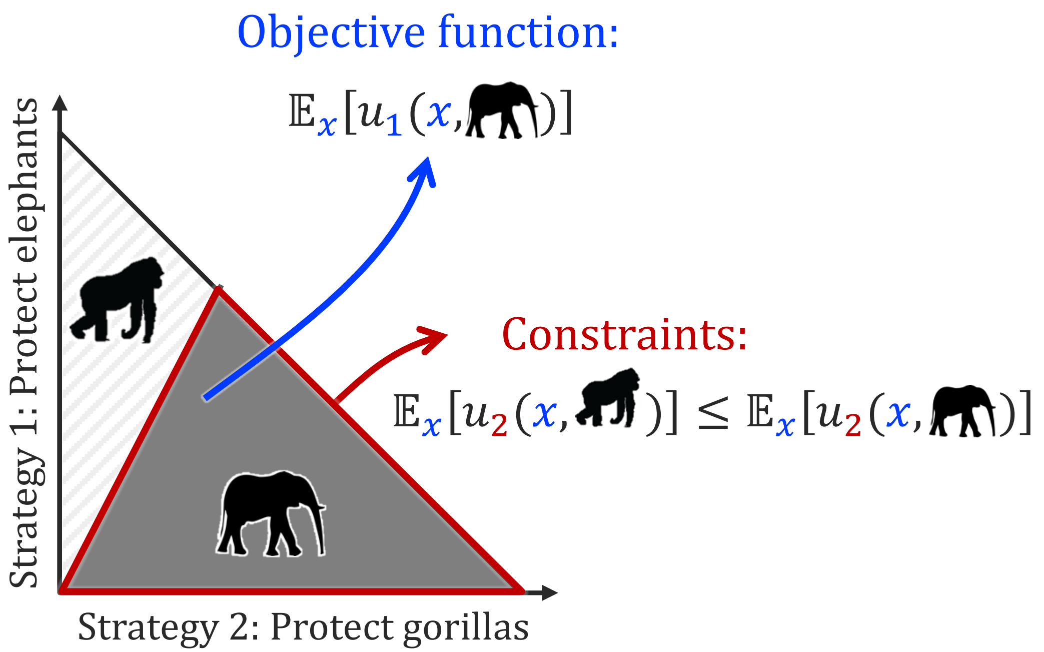
Defender	Gorilla	Elephant
Left	0 / 0	-2 / 4
Right	-4 / 2	0 / 0

Known payoffs

Solve multiple LPs. Convex polytope

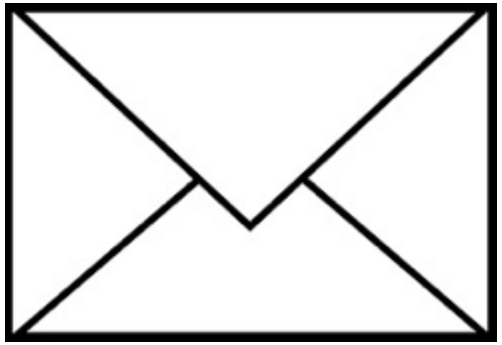
$$P_y = \{P \in \Delta(X) \mid BR(P) = y\}$$

Compute $\max\limits_{Y} \max\limits_{P \in P_Y} U_1(P, y)$



Important Message

Commitment in general-sum games also makes computation easier.



Adversarial Interactions

General-Sum Games

Computing Stackelberg equilibria

Learning Stackelberg equilibria

Commitment and non-myopic agents

Learning a Stackelberg Optimal Strategy

What do we typically **know**? And what has to be **learned**?

- For general-sum games, we know $u_1(x, y)$ but not $u_2(x, y)$.

  → We are able to observe $\underbrace{BR(P)}_{\text{Actual action}}$.

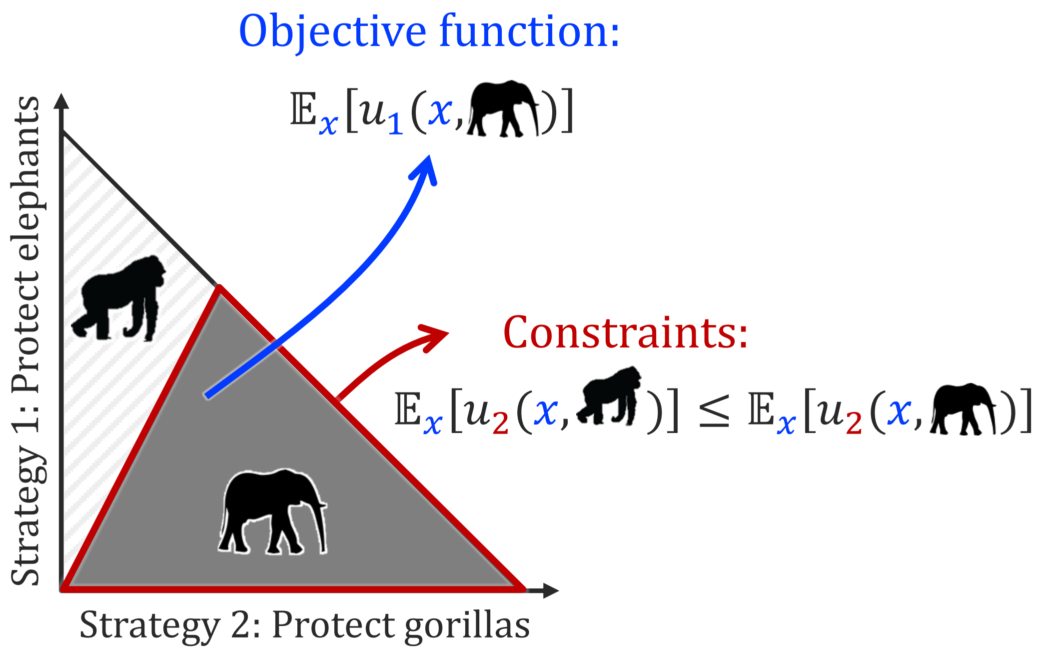
Need to compute

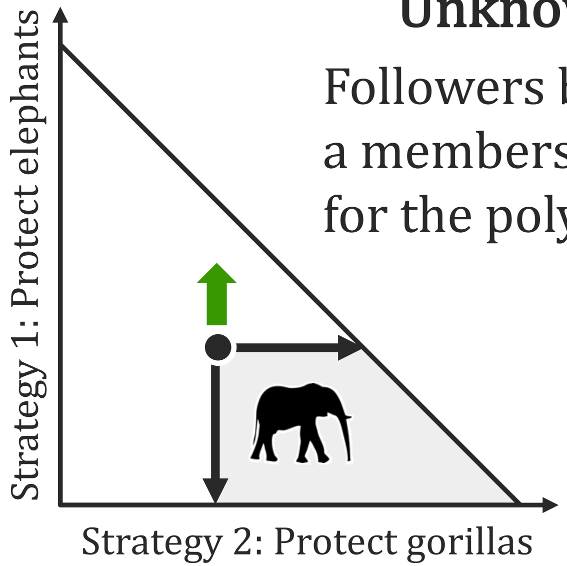
$$\operatorname{argmax}_{P \in P_y} \underbrace{U_1(P, y)}_{\text{Objective is known}}, \text{ where } P_y = \{P \in \Delta(X) \mid \text{For all } y', \underbrace{U_2(P, y') \le U_2(P, y)}_{\text{Polytope is unknown}}\}$$

How to optimize a linear program without knowing the polytope?

Optimization with Best-Response oracle

We can use access to $BR(\cdot)$ to learn a Stackelberg optimal strategy





Unknown payoffs

Followers best-response is a membership query oracle for the polytope.

Solving LPs with Membership queries

There are algorithms that optimize a linear program in R^n with accuracy ϵ, using $O(n^2 \ln(1/\epsilon))$ membership queries.

Kalai and Vempala ’05, Lee Sidford Vempala ‘18

Optimization with Best-Response oracle

We can use access to $BR(\cdot)$ to learn a Stackelberg optimal strategy

Solving LPs with Membership queries

There are algorithms that optimize a linear program in $\mathbb{R}^n$ with accuracy ϵ, using $O(n^2 \ln(1/\epsilon))$ membership queries.

Kalai and Vempala '05, Lee Sidford Vempala '18

Solving LPs with Membership queries

Using the above algorithm for each $P_y, \mathrm{y} \in Y$, gives an algorithm for learning the mixed optimal Stackelberg solution in $\mathrm{poly}(|X|, |Y|)$ membership queries.

H. Blum, Procaccia'14

Different approaches in Letchford Conitzer Muanagal '09, Peng, Shen, Tang, Zuo '19, etc.

Stackelberg Regret

Offline versus Online Learning a Stackelberg Optimal strategy.

In a repeated game:

Stackelberg Optimal Strategy
(single or a distribution of followers)

Leader Utility per round

$$\text{Stackelberg Regret} = \boxed{\max_{P^*} \frac{1}{T} \sum_{t\in[T]} U_1\big(P^*, \mathrm{BR}_t(P^*)\big)} - \frac{1}{T} \sum_{t\in[T]} \boxed{U_1\big(P_t, \mathrm{BR}_t(P_t)\big)}$$

Balcan, Blum, **H.** , Procaccia ’15
Dong, Roth, Schutzman, Waggoner, Wu ‘18

$BR_t(P^*)$ allows for having different types of followers each round.

Offline algorithms that **learn the optimal Stackelberg strategy** from **best-response queries** also lead to **No-Stackelberg-Regret** algorithms.

Stackelberg Regret vs (External) Regret

Recall the notion of regret from yesterday (aka External Regret)

Utility of best in Hindsight, on the historical observation

$$\text{(External) Regret} = \boxed{\max_{P^*} \frac{1}{T} \sum_{t \in [T]} U_1\big(P^*, \mathrm{BR}_t(P_t)\big)} - \frac{1}{T} \sum_{t \in [T]} U_1\big(P_t, \mathrm{BR}_t(P_t)\big)$$

$$\Updownarrow$$

$$\text{Stackelberg Regret} = \boxed{\max_{P^*} \frac{1}{T} \sum_{t \in [T]} U_1\big(P^*, \mathrm{BR}_t(P^*)\big)} - \frac{1}{T} \sum_{t \in [T]} U_1\big(P_t, \mathrm{BR}_t(P_t)\big)$$

Stackelberg Optimal Strategy

(single or a distribution of followers)

Stackelberg Regret vs (External) Regret

Comparison between Regret notions

Stackelberg and External Regret are worst-case Incompatible
- Any no-regret algorithm, will have O(1) Stackelberg regret in some cases.
- Any no-Stackelberg-regret algorithm, will have O(1) external regret in some cases.

Chen, Liu, Podimata'19

Utility of best in Hindsight,
on the historical observation **VS** **Stackelberg Optimal Strategy**

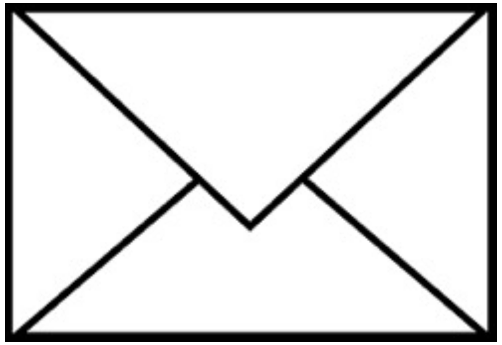
Why?
- The advantage of Stackelberg optimal solution is that **it's not a best-response** to the follower (that's Nash's job)
  → Stackelberg solution must appear to be not optimal over the past.

- External regret does not account for the fact that the follower will adapt to best respond.

Important Message

We can not have best of both world.

Need to know whether the strategically react to us or not



Bandits and Stackelberg Regret

Generally online Stackelberg games are partial-information optimization problems

$$\text{Stackelberg Regret} = \max_{P^*} \frac{1}{T} \sum_{t\in[T]} \underbrace{U_1\big(P^*, \mathrm{BR}_t(P^*)\big)}_{f_t(P^*)} - \frac{1}{T} \sum_{t\in[T]} \underbrace{U_1\big(P_t, \mathrm{BR}_t(P_t)\big)}_{f_t(P_t)}$$

Two challenges as discussed before:

- Optimization problems is usually non-convex, non-Lipschitz.
  → Structured, piecewise in particular for finite games.
- Partial information:
  - Observation in one round $f_t(P_t)$ does’t reveal $f_t(P')$.
  - More than bandit information, we see $\mathrm{BR}_t(P_t)$.
  - Exploration more tuned to the information and structure.

Adversarial Interactions

- General-Sum Games
- Computing Stackelberg equilibria
- Learning Stackelberg equilibria
- **Commitment and non-myopic agents**

Learning and Commitment Revisited

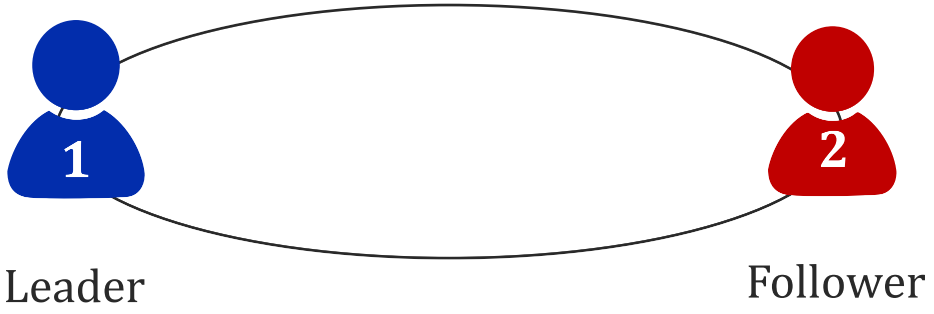
Learning is antithetical to Stackelberg games:

- Advantage of Stackelberg (over Nash) is the power to commit.
- Learning algorithms don't commit.

Non-myopic agents: Agents optimize over or infinitely repeated game.

Cheap talk

Actions that have no impact on long term utility



Inhibits learning

Discussion of ongoing work H., Lykouris, Nietert, and Wei '22. Preprint coming soon.

Infinitely Repeated Games Formality

Typical assumptions:

- One or both agents receive discounted utilities*.
- One or both agents come from a larger set (large market).

Just the follower

Follower:

- Doesn't best respond necessarily.
- Strategy account for both past and future
- Chooses a policy to select $Q_t s$ that (approx.) optimize expected discounted utility

$$\mathbb{E}\left[\sum_{t=1}^{T} \gamma^t \, U_2(P_t, Q_t) \mid \underbrace{\text{Algorithm,}}_{} \text{ follower policy}\right]$$

Leader's commitment to an algorithm
Principled approach to design

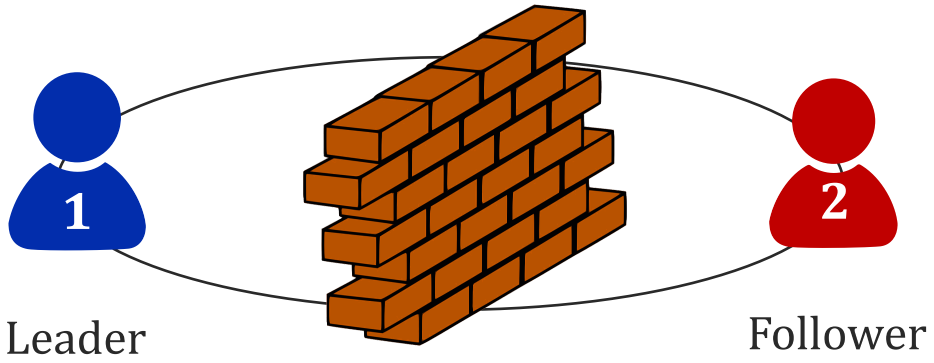
* Common in Reinforcement learning and various various Folk theorems in Economics.

Controlling the flow of information

Cheap talk not so cheap anymore

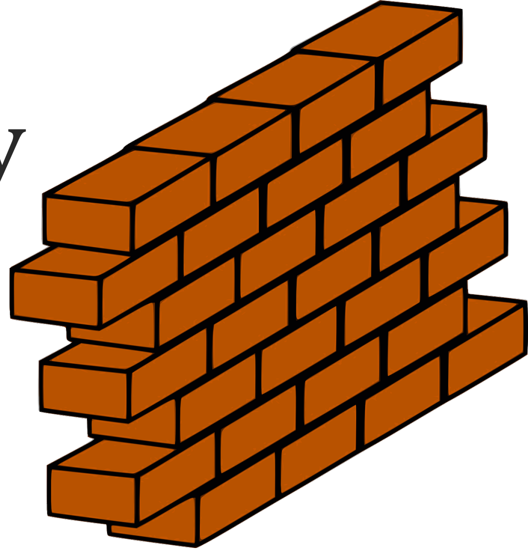
- Discounted utility: Lost opportunity, for not best-responding instantaneously.
- We can control the rate using additional barriers

Encourages approximate best responding



Can a learner learn despite the barrier?

“Barriers” to encourage Incentive-Compatibility



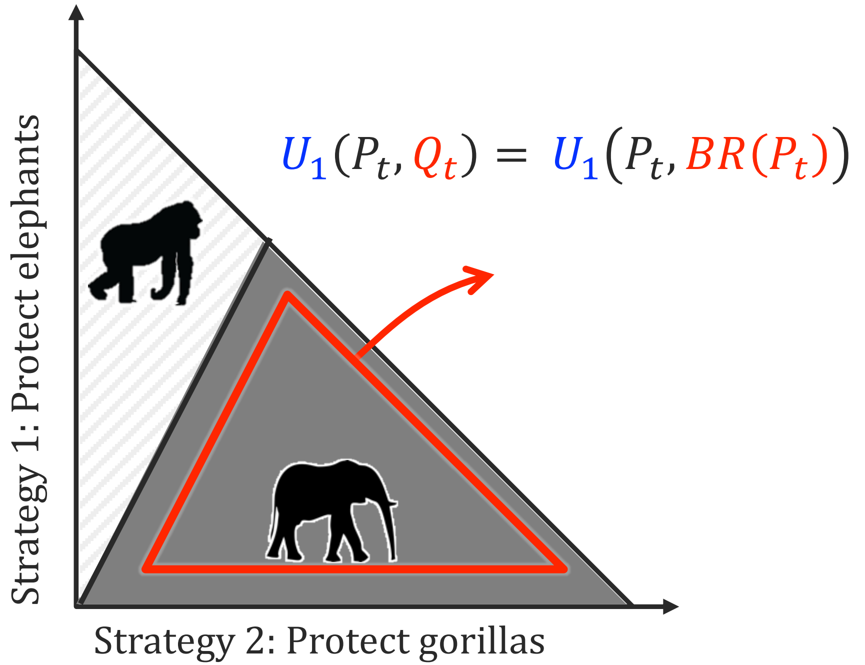
Barriers:

- Natural to delay information, by D steps.
- For large enough D, the total expected gain from future is small.
- So Q_t is an approximate best response.

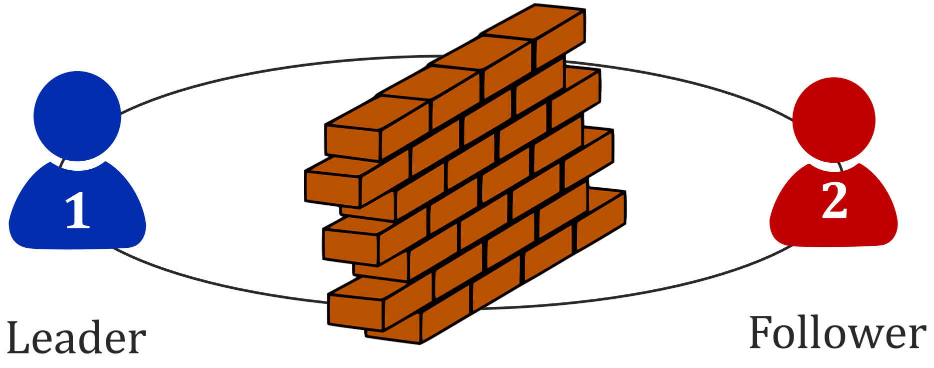
$$U_2(P_t, Q_t) \geq U_2\big(P_t, BR(P_t)\big) - \epsilon$$

Approximate best response:

- Guarantees “some” closeness between $U_1(P_t, Q_t)$ and $U_1\big(P_t, \underbrace{BR(P_t)}_{\text{Wish we could see } f_t(P_t)}\big)$
- In finite games, only problem at the boundaries



Algorithmic Desiderata



1. Optimize $f_t(P_t) = U_1\big(P_t, BR(P_t)\big)$ from bandit observations. **(Even for myopic agent)**

2. Be robust to some misspecification of $f_t(P_t)$, say $\hat{f}_t(P_t)$. **(Common robustness guarantees)**

   →Different only in some small or structured sets.

   →Pointwise close everywhere.

3. Be able to handle delays **(More on this)**

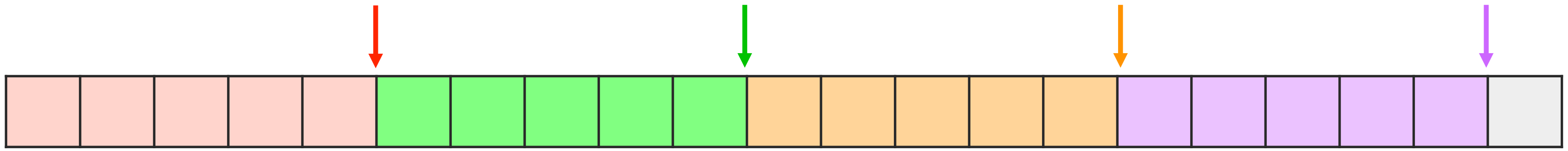
Designing algorithms for delayed feedback

A better studied setting of “Batched Bandits”.

Batched Bandits:

Algorithm submits queries in batches of D and receives responses after the batch is done.

- Advantage: More common in optimization.



Delays = Batched Bandits

Algorithm with Regret_D for delays of D steps.

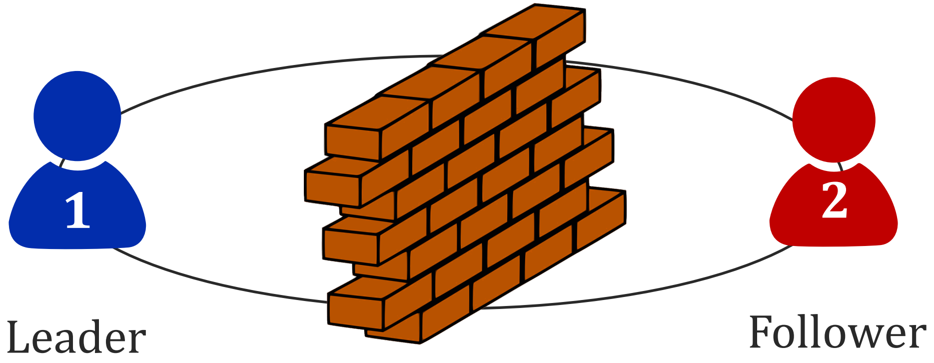
Algorithm with $Regret_D$ and batches of size D each batch delayed by 1 batch.

$O(1)\times$

Algorithm with $Regret_D$ and batches of size D.

HLNW’22

Algorithmic Desiderata



1. Optimize $f_t(P_t) = U_1\left(P_t, BR(P_t)\right)$ from bandit observations.
2. Be robust to some misspecification of $f_t(P_t)$, say $\hat{f}_t(P_t)$.
   →Different only in some small or structured sets.
   →Pointwise close everywhere.
3. Be able to handle delays

Robust Batched Bandit algorithm

Algorithmic Desiderata

Reduction for Non-Myopic Agents

Any robust batched bandit algorithm for $U_1\big(P_t, BR(P_t)\big)$ can turn into an algorithm that in presence of non-myopic agents, achieves vanishing Stackelberg regret

HLNW'22

1. Optimize $f_t(P_t) = U_1\big(P_t, BR(P_t)\big)$ from bandit observations.

2. Be robust to some misspecification of $f_t(P_t)$, say $\hat{f}_t(P_t)$.

   →Different only in some small or structured sets.

   →Pointwise close everywhere.

3. Be able to handle delays

Robust Batched Bandit algorithm

Robust Batched Bandit Algorithm

In the most common three frameworks of bandit optimization, we can design robust batched bandit algorithms.

- Multi-armed bandits: We introduce an algorithm that's both robust and is especially effective when batched.
  → Useful for all discretized algorithms, auctions, demand learning, etc.

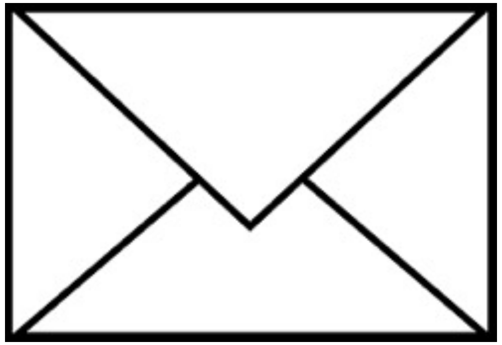
- Multiple LP approach with Membership Queries: Lee Sidford Vempala comes with robustness built in. Adjust the approach of Blum Procaccia **H.** to use this robustness.
  → Useful for all finite games. Especially nice with security games.

- Bandit Convex Lipschitz Optimization (without gradients): Some algorithms come with robustness built in.
  → Useful for many strategic classification settings.

Important Message

Handle non-myopic agents, by controlling the flow of information.

Leverage (adversarial robustness) in bandit algorithms.



Beyond this tutorial

Non-myopic agents:

- Online auctions: Amin Rostamizadeh Syed'13 and '14, Mohri Munoz'14, Huang Liu Wang'18, Abernethy Cummings Kumar Morgenstern Taggart'19, Golrezaei Javanmard Mirrokni'19, Golrezaei, Jaillet, Liang'19.
- Strategic classification and commitment through the algorithmic framework: Zrnic Mazumdar Sastry Jordan '21.

Other tools for learning and incentive-compatibility:

- Differential privacy as a tool: McSherry Talwar'07, Nissim Smorodinsky, Tennenholz'12, Kearns Pai Roth Ullman'14, Huang Liu Wang'18, Abernethy Cummings Kumar Morgenstern Taggart'19.

Tutorial Overview

Wednesday

1. Adversarial Interaction
   - Offline, Online adversarial learning, and Zero-sum Games
   - Beyond the worst-case adversaries
   - Computational Challenges

Thursday

2. General Strategic Interactions
   - General-sum games and Stackelberg concept
   - Learning and Stackelberg equilibria
   - Learning in presence of non-myopic agents
3. Collaborative Interactions
   - Models of data sharing for learning
   - Average vs. Per-Agent learning guarantees
   - Individual Rationality and Equilibria

Collaboration in Learning

Models for collaborative learning

Average vs Per-agent guarantees

Rationality and Equilibria

Collaboration in Learning

Models for collaborative learning

Average vs Per-agent guarantees

Rationality and Equilibria

Collaboration

Decisions to Act

Information Collection

More Data … More Stakeholders

1. Data is spread across several sources

2. Individualized and heterogenous learning objectives

3. Individual data sources have external objectives as a whole

Data Sharing and Federated Learning

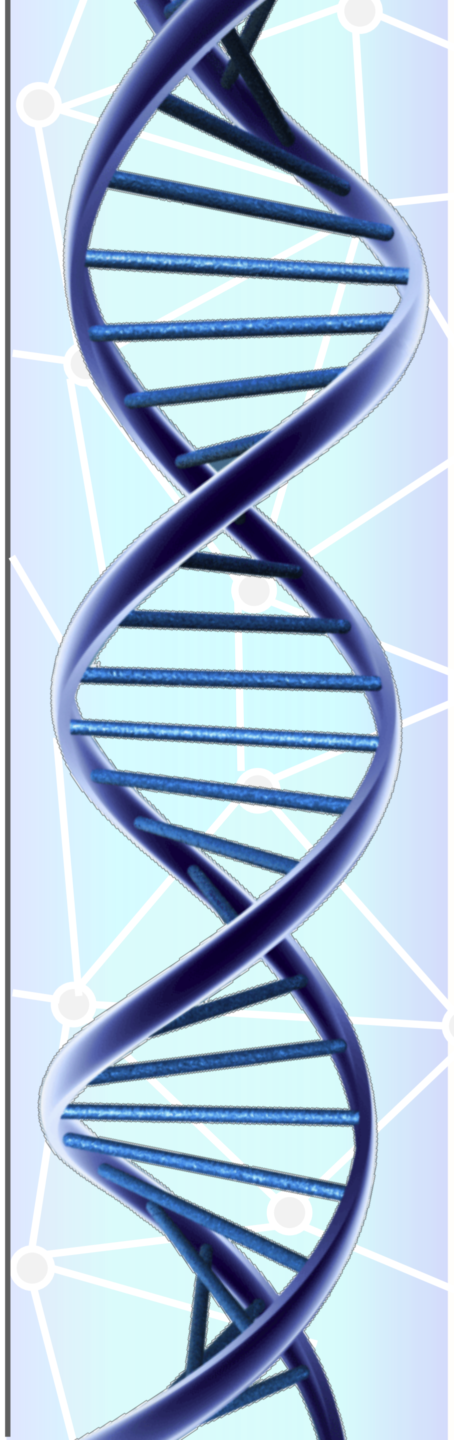
Enabling large numbers of learning agents

to **collaboratively accomplish** their goals

using **collectively fewer resources**.

Starting to be used across network of devices, hospitals, etc.

Behind major scientific breakthroughs: Mapping the biological mechanisms underlying schizophrenia in a large scale collaboration of data from than 100 institutions.



Large Scale Impact from
Mass Participation

Recruit and Retain

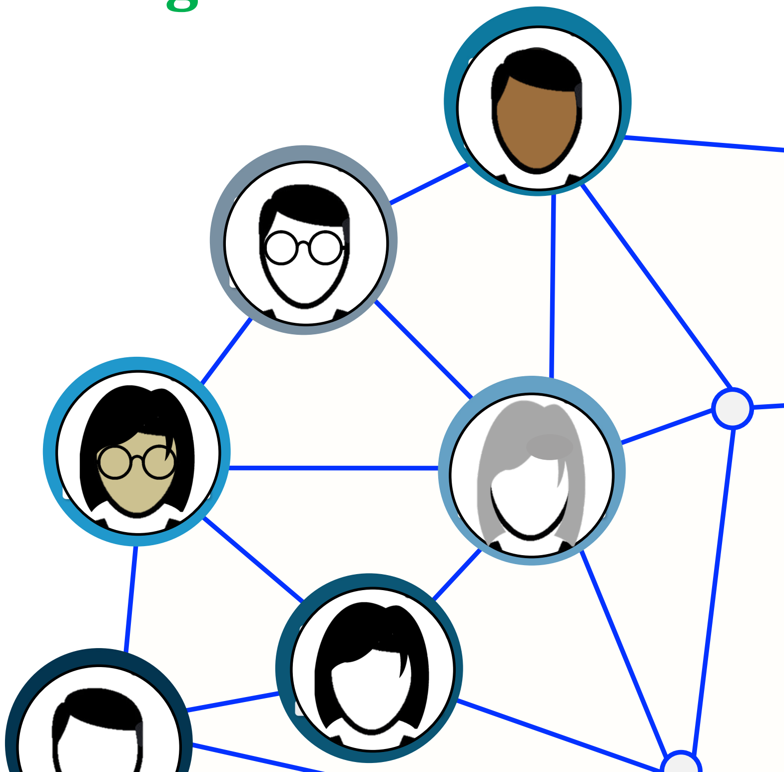
Meeting Individual Learning Guarantees

Federated algorithms work well **on average** over the data sources

- Good for learning across data centers.
- Good for when the data is homogenous across sources.

Human and organization data:

- For non-homogenous tasks, a model that has **5% error on average** can have **50% error for $^{1}/_{10}$ of the agents**.



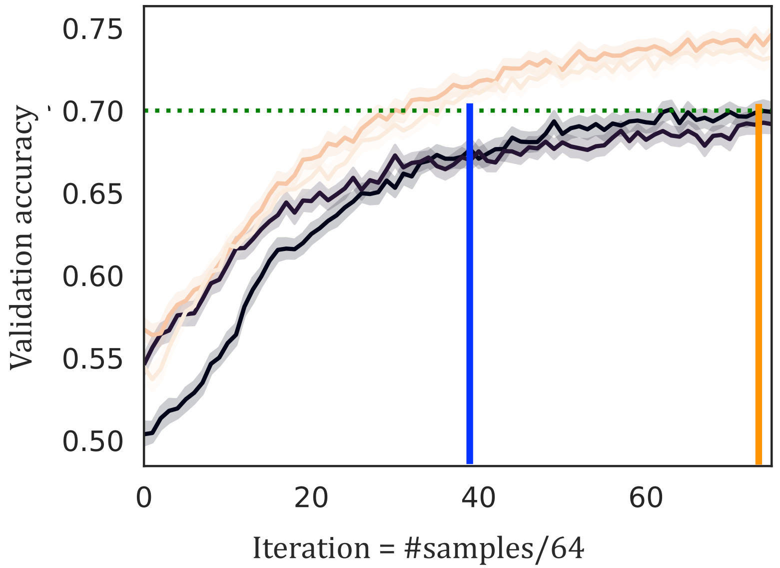
Meeting Individual Learning Guarantees

Federated algorithms work well **on average** over the data sources
- Good for learning across data centers.
- Good for when the data is homogenous across sources.

Human and organization data:
- For non-homogenous tasks, a model that has **5% error on average** can have **50% error for $^1/_{10}$ of the agents**.

Avg accuracy >0.7 **min accuracy >0.7**

Every agent uses 40 iterations.

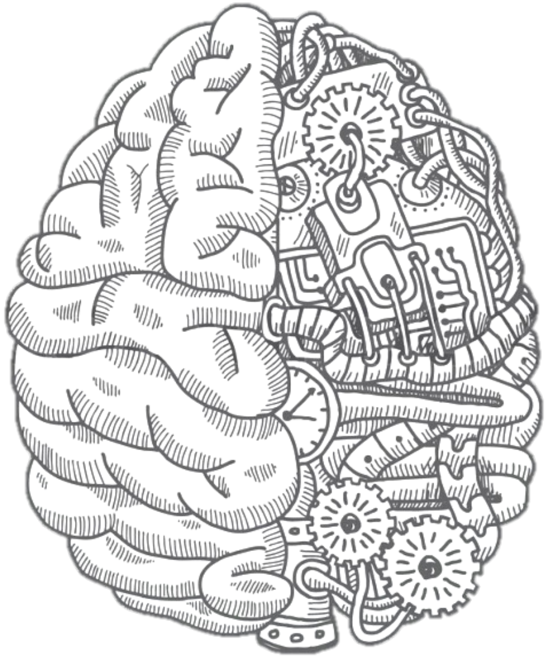


Every agent has to use 75 iterations.

Blum, **H**, Phillips, Shao '21

Can we ensure that every learning agent has high accuracy …

… from reasonably small amount of data?



Collaboration in Learning

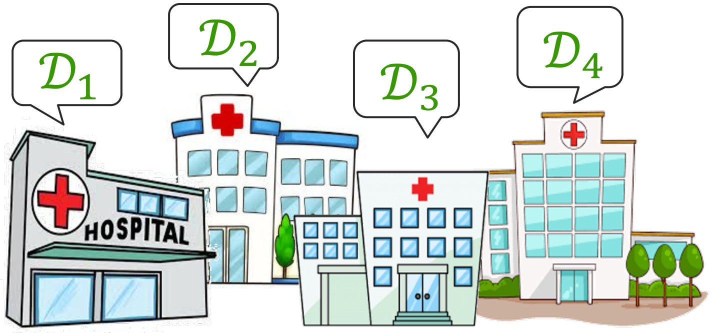
Models for collaborative learning

Average vs Per-agent guarantees

Rationality and Equilibria

Collaborative Learning

There are k populations/distributions $\mathcal{D}_1, \mathcal{D}_2, \dots, \mathcal{D}_k$.



We want the to learn a model f that is good **for every population.**

$$\max_{i\in[k]} \mathrm{err}_{\mathcal{D}_i}(f) \le \epsilon$$

How much data suffices for every learner to have high accuracy?

Blum, **H**, Procaccia, Qiao ’17

Collaboration Needs Interactions

The trouble with standard algorithms:

Lack of interactions (except to perform distributed computation)
→ # of samples, learning rates, and update frequencies for an agent is decided non-interactively.

Sample complexity of existing algorithms, for k agents $= \Theta(k) \times$ Learning for 1 agent separately
1 agent # samples

Without an "interactive" protocol, collaboratively learning is (almost) as ineffective as not collaborating at all.

Collaboration Needs Interactions

The trouble with standard algorithms:

Lack of interactions (except to perform distributed computation)
→ # of samples, learning rates, and update frequencies for an agent is decided non-interactively.

$$\frac{\text{Sample complexity of existing algorithms, for } k \text{ agents}}{} = \Theta(k) \times \frac{\text{Learning for 1 agent separately}}{\text{1 agent \# samples}}$$

Sample complexity of existing algorithms, for k agents $= \Theta(k)\times$ Learning for 1 agent separately 1 agent # samples

Interactivity

Adjusting sample collection based on past performance

$$\frac{\text{There is an algorithm}}{\text{Overall \# samples}} = \Theta(\log k) \times \frac{\text{Learning for 1 agent separately}}{\text{1 agent \# samples}}$$

A MinMax Optimization

Between the algorithm and a hypothetical adversary that chooses the worst-off agent

Player 1

$$\min_{h \in H} \max_{i \in [k]} \ err_{D_i}(H)$$

Player 2

Solve with a **no-regret algorithm** against a **best-responding agent**.

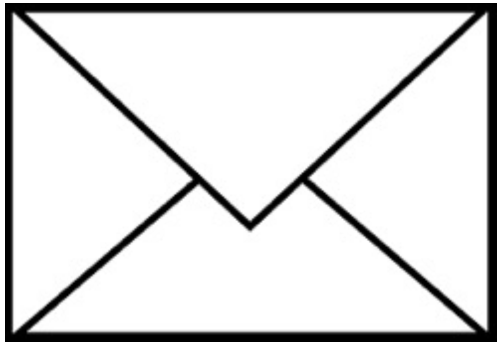
Player 1: The best-responding agent. For any distribution over [k], $\alpha_1^t, \ldots, \alpha_k^t$, it uses an Empirical Risk Minimizer to learn $h^t \in H$ on the distribution $P^t = \sum \alpha_i^t D_i$

Player 2: The no-regret learning agent. Maintains a distribution over $[k]$, say weights $\alpha_1^t, \ldots, \alpha_k^t$ over the agents. Proxy of how poorly they've been doing so far.

$$\epsilon' \geq \frac{1}{T} \sum_t err_{P^t}(h^t) \geq \max_{i \in [k]} \frac{1}{T} \sum err_{D_i}(h^t) \ - Regret$$

Important Message

Online learning as a medium for collaboration



Collaboration in Learning

Models for collaborative learning

Average vs Per-agent guarantees

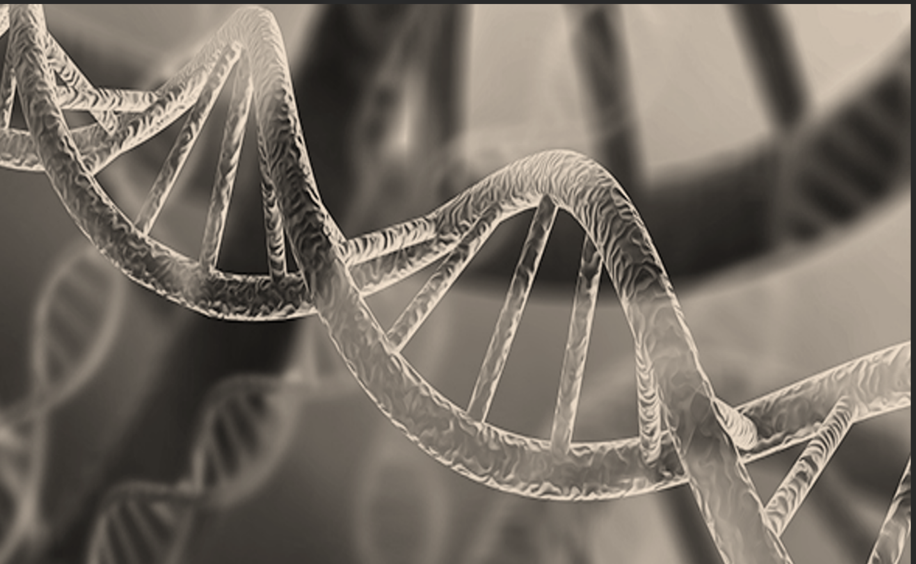
Rationality and Equilibria

Beyond Accuracy Guarantees

Agents also incur cost for collecting information:

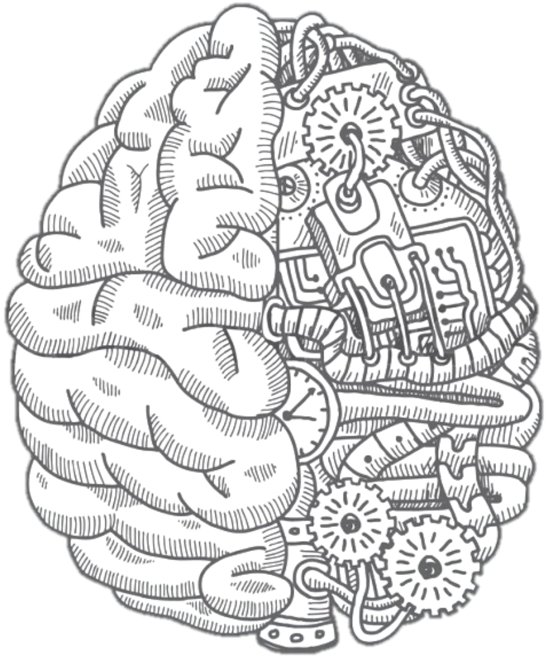
- E.g., cost for data set curation, privacy cost, etc.
- The protocol shouldn't ask for "unreasonable" amount of data.

→Collaboration should be beneficial to all of its users.



Achieve desirable
per-agent tradeoff between
accuracy and sample complexity

A theory for multi-agent sample
complexity!



Reasonable Share of Data

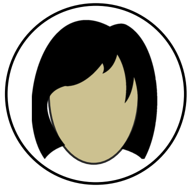
What we ask of agent i is unreasonable if:

- Ask i for more data than necessary, if he were to learn by himself.
- Part of i's contribution is exclusively used to meet the accuracy constraint of other agents and did not affect agent i.

[Blum, **H**, Phillips, Shao '21]

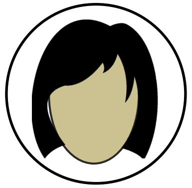
Individually Rational

1. Every agent's accuracy constraint is met, and
2. No agent collects more data than he needs, by himself.



$$\underbrace{(m_1, m_2, \dots m_i \;\; \dots m_k)}_{\#\text{ of collected samples}}$$

If  's accuracy constraint is met $m_i \leq m'_i$



$$(0,0, \dots, m'_i \,, \dots 0)$$

Reasonable Share of Data

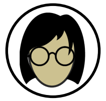
What we ask of agent i is unreasonable if:

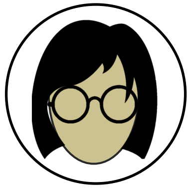
- Ask i for more data than necessary, if he were to learn by himself.
- Part of i's contribution is exclusively used to meet the accuracy constraint of other agents and did not affect agent i.

[Blum, **H**, Phillips, Shao '21]

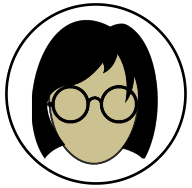
Stable Equilibrium

1. Every agent's accuracy constraint is met, and
2. No agent can reduce her contribution and still meet her accuracy constraint.

's accuracy constraint won't be met



$(m_1, m_2, \ldots m_i \;\; \ldots m_k)$



$(m_1, m_2, \ldots m'_i \;\; \ldots m_k)$

$m'_i < m_i$

Rationality and Equilibria Matter

Welfare of the agents:

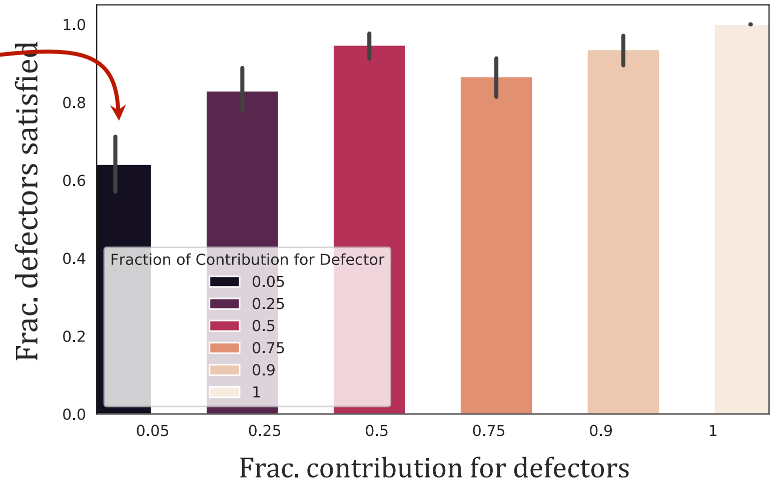
- Receiving a reasonable return in what resources you put in.

Usability and stability of systems over time:

- Even a small reduction in contribution across the agents impacts algorithmic performance.

State of the art learning algorithms are VERY far from equilibrium

60% of agents can unilaterally reduce their contributions to 5% of current levels.



Do Equilibria exist?

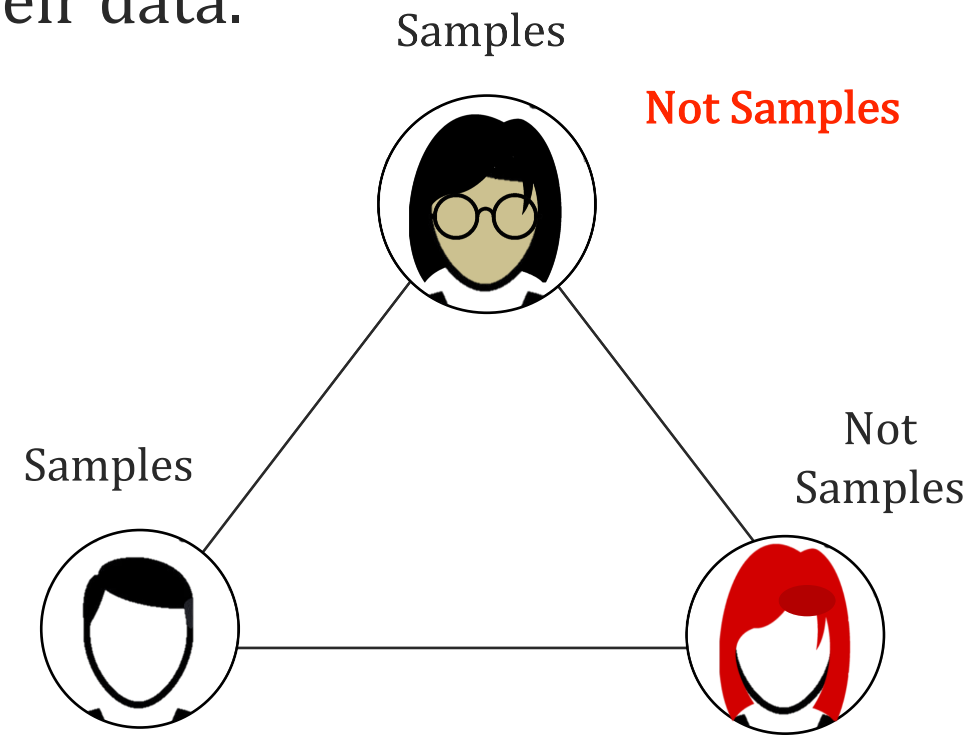
Unfortunately, some learning problems have no stable equilibrium!

But they do generally exist under mild assumptions that are met in most applications.

Existence of Equilibria

Each agent is much much better at completing the next agents task, then their own.

Let the labels an instance in  ‘s distribution, encode the target function for the next agent, as well as revealing the target on their data.



Cycling behavior:

- Non-continuous functions and actions
- More of a pure strategy equilibrium.

Are Equilibria Efficient?

They may require more collective resources than the optimal collaboration!

In some cases,

Best equilibrium = Some agents don't contribute, others optimally collaborate.

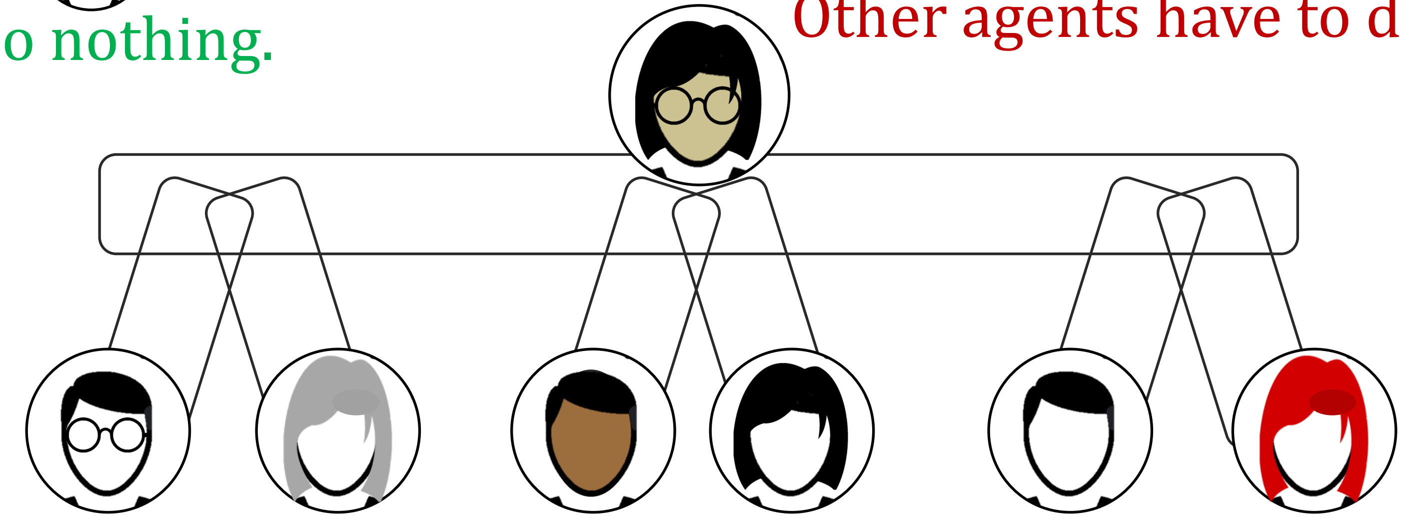
Judiciously introduce small inefficiencies, so everyone can continue benefitting from the system.

Difficulty of Rationality and Stability

Individually rational or stable equilibria, require more collective resources than the optimal collaboration.

Optimal: does all the work, others do nothing.

Equilibrium: does no work. Other agents have to do the work.



Equilibrium/Individual Rationality: Total work required to be done by other agents is large.

$$\text{Overall \# samples in the best equilibrium} = \Omega\left(\sqrt{\text{\# agents}}\right) \times \text{Overall \# samples in the optimal collaboration}$$

Optimality, Equilibria, and Free Riding

In some cases, equilibria are highly structured.

If the utility/loss of agents are linear functions of the contribution:

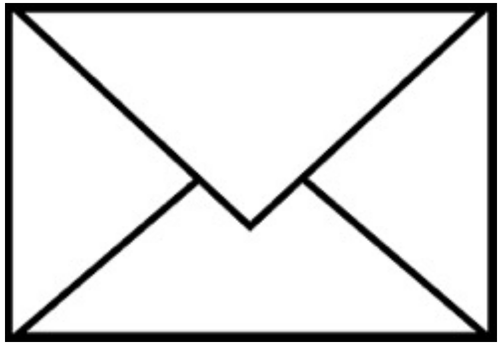
Difference between optimal:

- Any **equilibrium** is an optimal collaboration among **a subset of agents**.
- Free riding is part of equilibria.

→ But it doesn't impact **optimality** of the contributions of **participating agents.**

Important Message

New mathematical foundation needed to
design learning algorithms that **act globally**,
and consider **per-agent incentives and objectives.**



Tutorial Overview

Wednesday

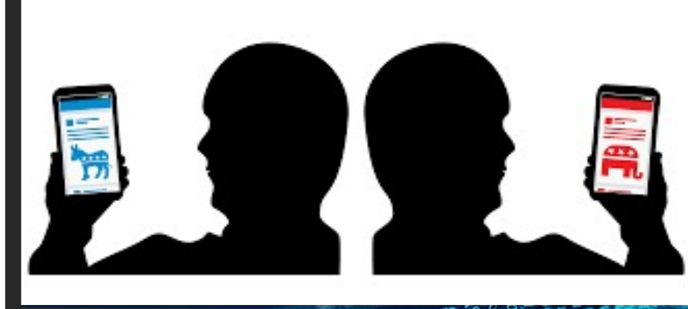
1. Adversarial Interaction
   - Offline, Online adversarial learning, and Zero-sum Games
   - Beyond the worst-case adversaries
   - Computational Challenges
2. General Strategic Interactions
   - General-sum games and Stackelberg concept
   - Learning and Stackelberg equilibria
   - Learning in presence of non-myopic agents

Thursday

3. Collaborative Interactions
   - Models of data sharing for learning
   - Average vs. Per-Agent learning guarantees
   - Individual Rationality and Equilibria

Learnability for Today's World

Learning Algorithms ⟷ Environment



Learnability

Q1. What concepts can be learned in presence of strategic and adversarial behavior?
→ Lessons for todays world from decade of efforts for understanding.

Q2. How to design learning for strategic and adversarial environment?
→Computational overheads
→Principals on how to use/not use data in strategic environments.

Q3. How can we design collaborative environment that encourage learner participation?
→ Incentives of learning algorithms and data providers
→Deliver the optimal learning algorithms for agents and the society.

Q4. Generally, how do these learning paradigms relate to one another?