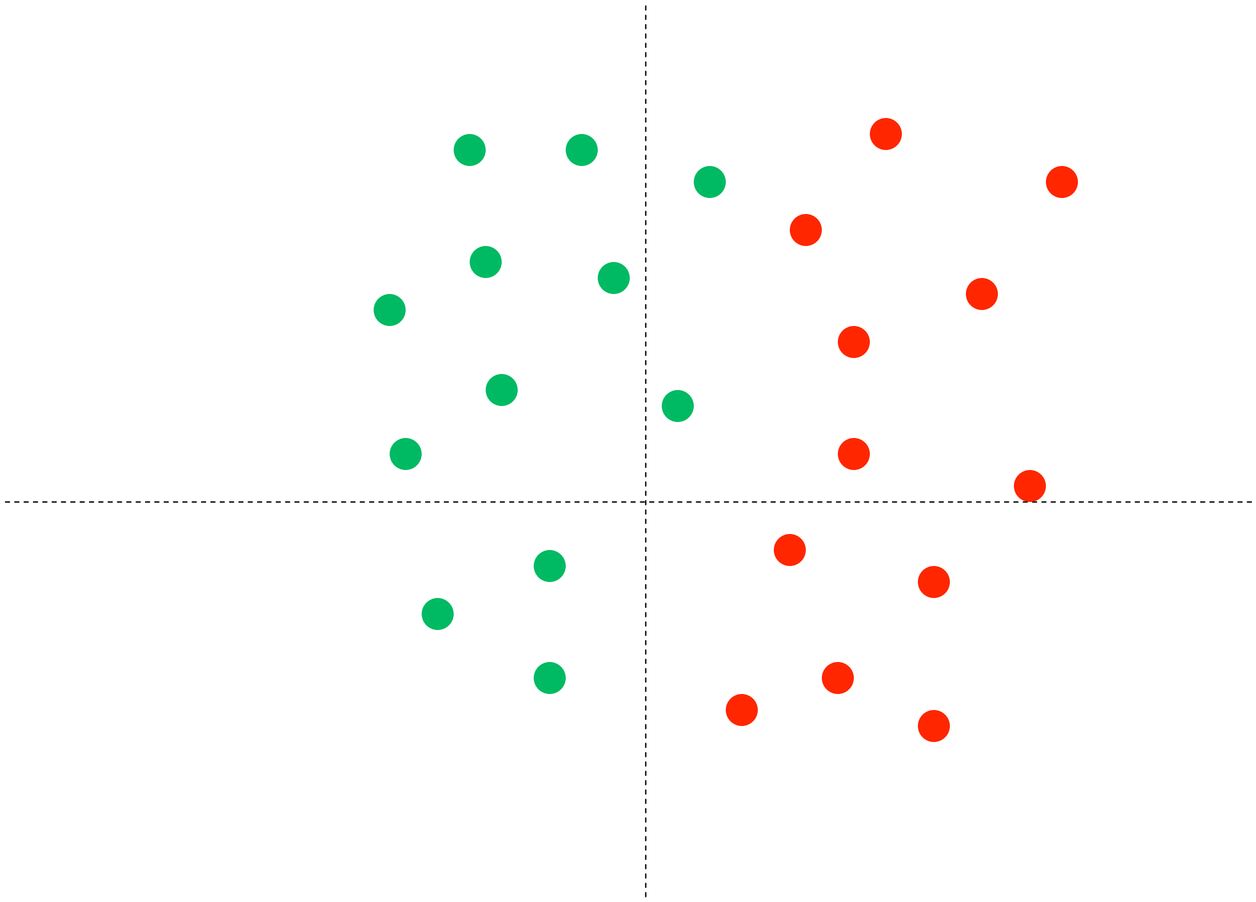
The Complexity of Statistical Algorithms

Lev Reyzin
Georgia Institute of Technology

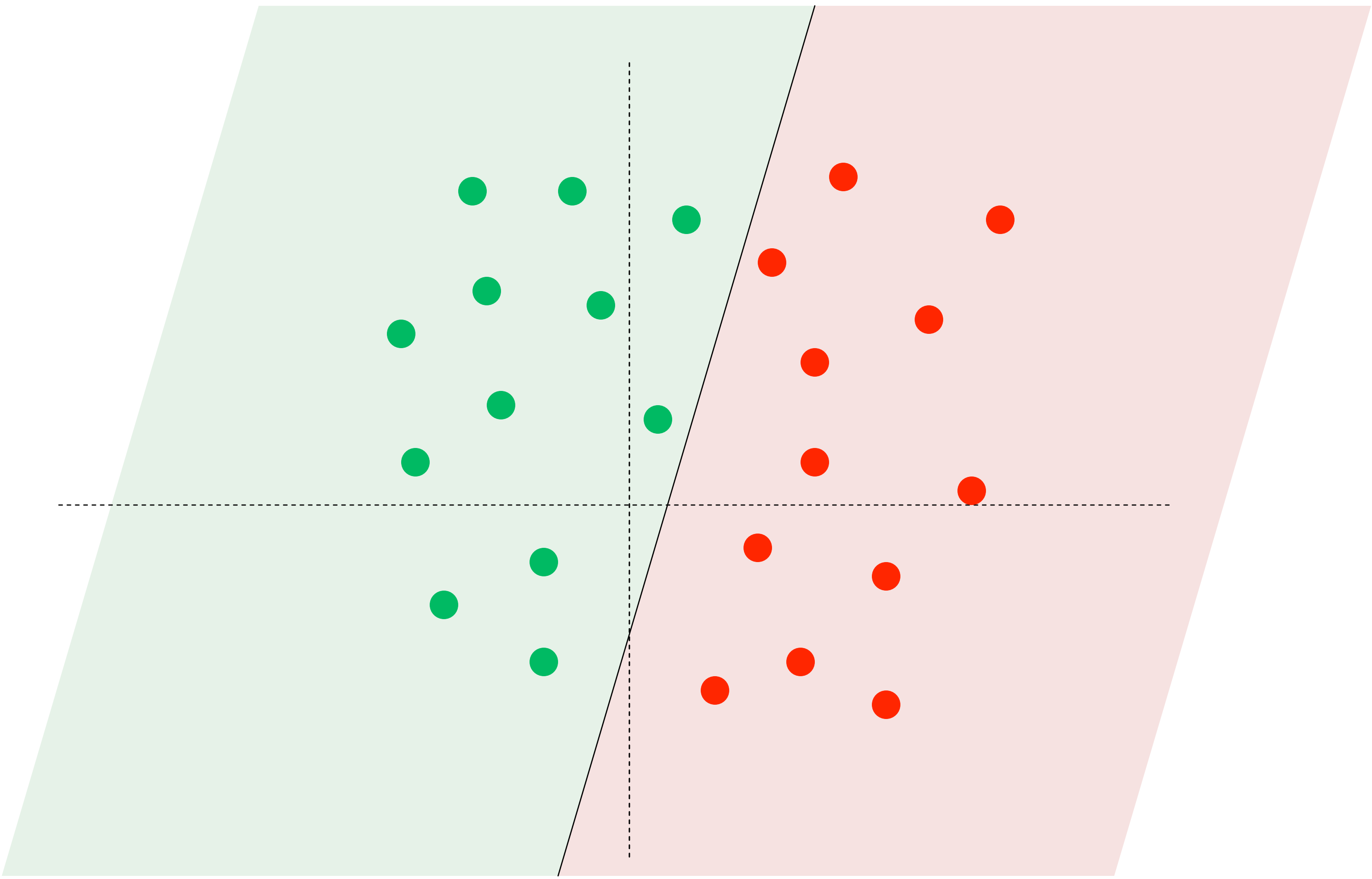
A Brief Introduction to Learning

some context and an introduction to my field

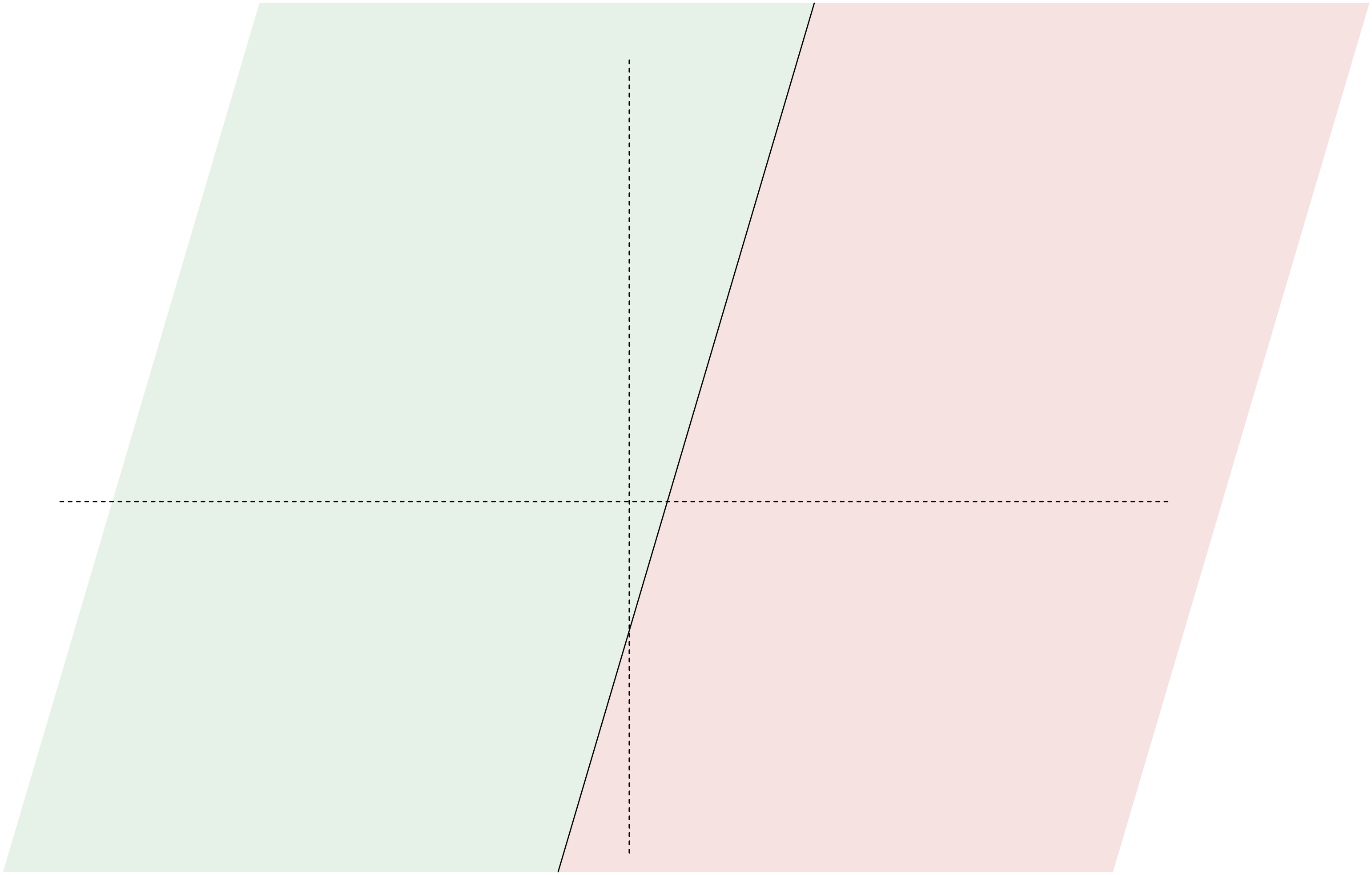
Learning Half-Planes



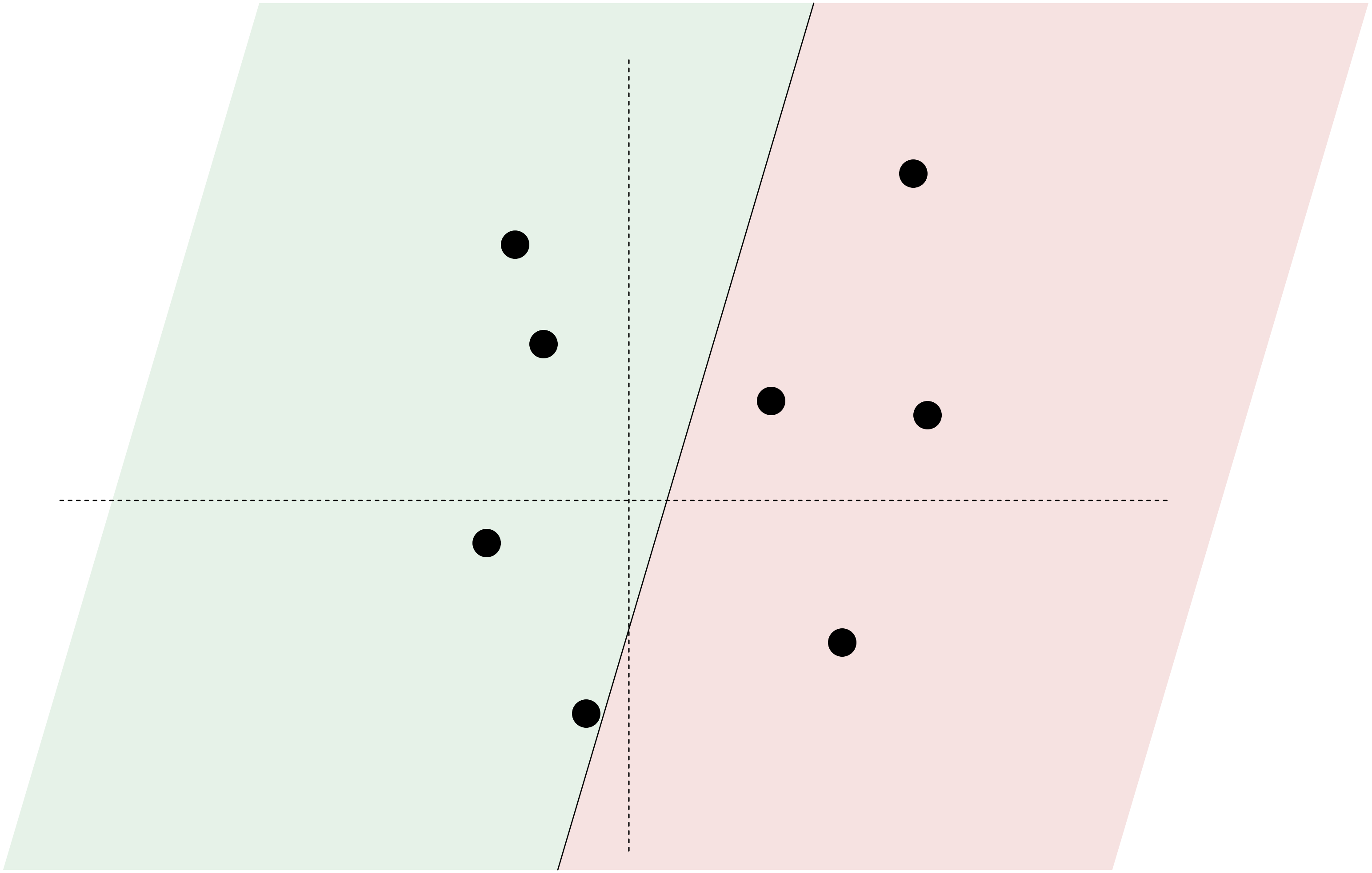
Learning Half-Planes



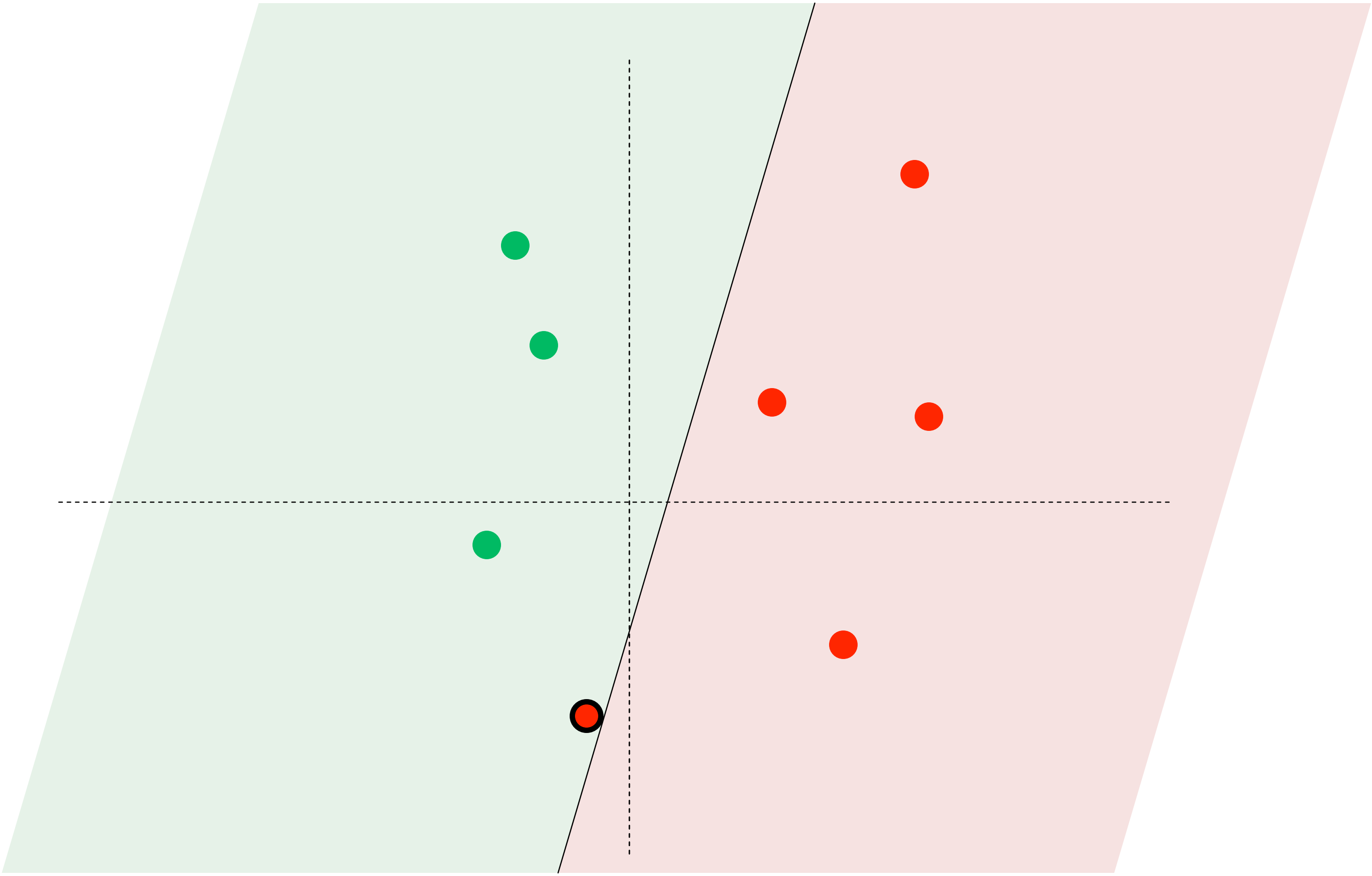
Learning Half-Planes



Learning Half-Planes



Learning Half-Planes



PAC Learning [Valiant '84]

- Let X be a domain ($\mathcal{R}^2$).
- Let D be a probability distribution over X.
- Let c: X $\rightarrow$ {-1,1} be a target "concept" (half-plane) and C be the set of possible targets c (all possible half-planes).

- Class C is learnable if $\forall$ c $\in$ C, D, ε>0, δ>0, a learner can receive a set S of m "labeled examples" from D: $\{(x_1,c(x_1)), \ldots ,(x_m,c(x_m))\}$ (colored points) and produce a hypothesis h_S: X $\rightarrow$ {-1,1} such that:

$$Pr_{S\sim D}[Pr_{x\sim D}[h_S(x) \neq c(x)] > \varepsilon] < \delta.$$

(ideally want m to be "small")

An Overview of PAC Learning

- PAC learning has a rich history, interesting results, nice theory, open problems, many applications, etc.
  - 2011 A. M. Turing Award to Valiant
- In trying to understand which PAC algorithms can handle noise, "statistical queries (SQ)" were invented [Kearns '93].
  - Algorithms that access their data via SQs are noise-tolerant.
  - It turns out that most learning algorithms fall into this category.
- Unfortunately, it is also known that SQ algorithms have serious limitations.
  - Notably, SQ algorithms cannot learn parities, among other classes of functions [Blum et al '93].

PAC Learning Parities

- [***Def.***] For $x \in \{0,1\}^n$ and $c \in \{0,1\}^n$, let $\chi_c(x)$ take the value 1 if $c \cdot x$ is odd and -1 otherwise.
  - If c has 1's only in r positions, we call c an r-parity.
- For an unknown target c, the learner sees labeled examples $(x, \chi_c(x))$ from some distribution, e.g.
  (00110101,1), (10011010,1), (00101111,-1), …
- Learner needs to determine c (or more generally predict labels of future examples).
- Learning parities turns out to be hard for SQ algorithms *even over the uniform distribution on $\{0,1\}^n$*.

Parities

- Therefore, we can prove lower bounds on statistical query learning by showing certain classes encode parities.
  - Example from my work: even random decision trees, automata, and DNF are not learnable with SQ.

    [Angluin-Eisenstat-Kontorovich-Reyzin '10]

- There has been little progress on noisy parity:
  - **For the general case**: brute-force takes $O(2^n)$ time.

    Best progress: $O(2^{n/\lg n})$ time. [Blum-Kalai-Wasserman '00]
  - **For r-parities**: brute-force takes $O(n^r)$ time.

    Best progress: $\sim O(n^{r/2})$. [Grigorescu-Reyzin-Vempala '11]

Summary

- In learning, the goal is to find a function of with low error on a distribution.
- Most learning algorithms are Statistical Query algorithms.
  - Good news: SQ algorithms are noise-tolerant.
  - Bad news: SQ algorithms have serious limitations.
- The SQ framework has elucidated many issues in learning.
- Our [Feldman-Grigorescu-Reyzin-Vempala '12] goal is to do the same thing for optimization (a generalization of learning)
  - Our results explain experimentally observed phenomena.
  - Our results generalize the SQ theory.

Optimization

the subject of this talk

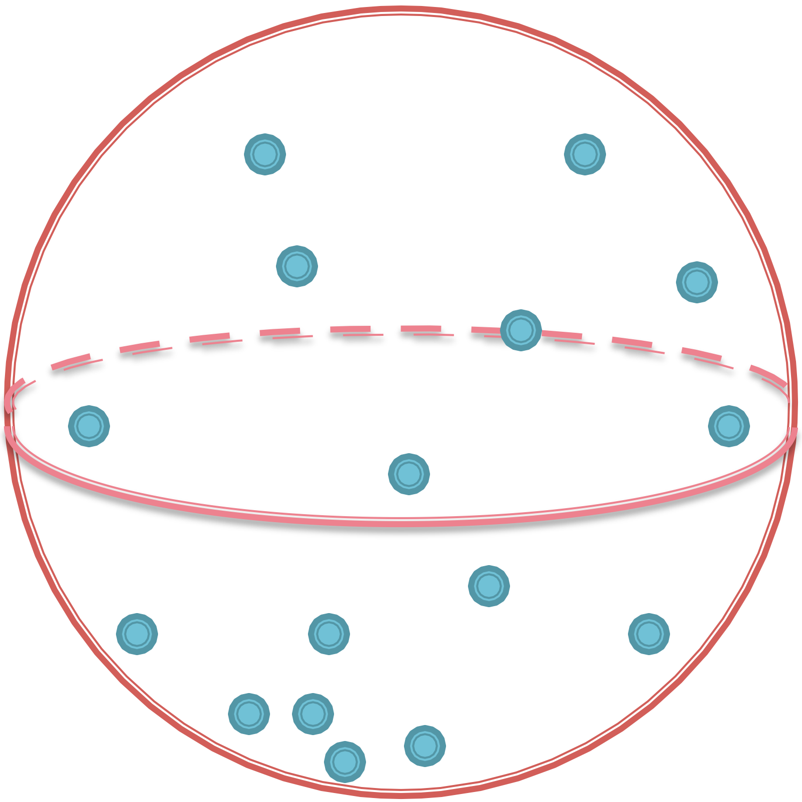
Motivating Example

problem: moment maximization

Let D be a distribution over points in $[-1,1]^n$ and let $r \in \mathcal{Z}^+$. The goal is to find a unit vector u* that **approximately maximizes the expected r'th moment** of the projection to u of a random point x chosen from D.

i.e. find

$$u^* \approx \arg\max_{u \in \Re^n:\ \|u\|=1} \mathop{\mathrm{E}}_{x \sim D}[(u \cdot x)^r].$$



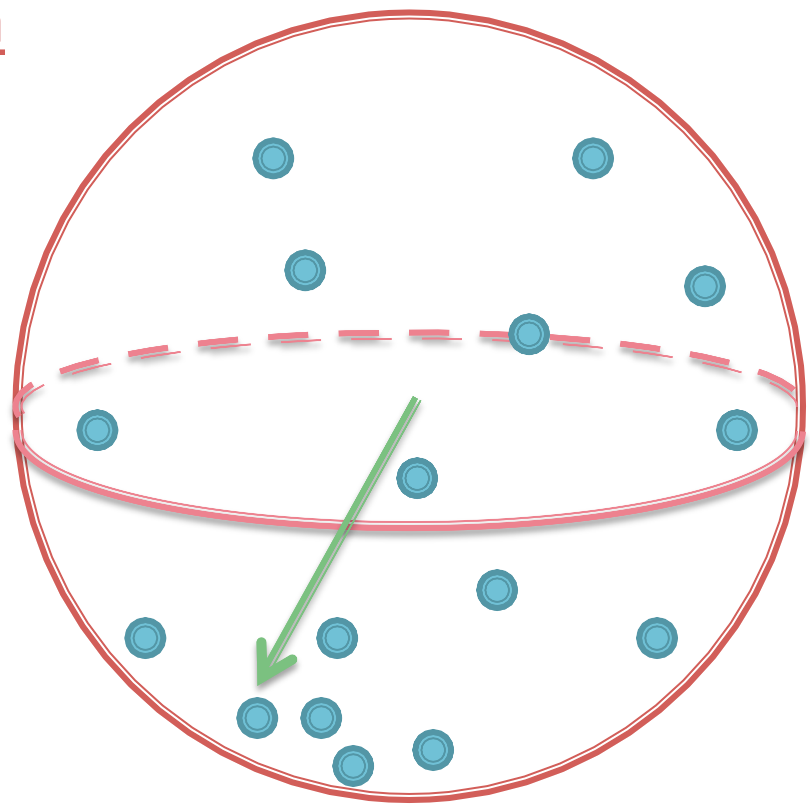
Motivating Example

problem: moment maximization

Let D be a distribution over points in $[-1,1]^n$ and let r ∈ $\mathcal{Z}^+$. The goal is to find a unit vector u* that **approximately maximizes the expected r'th moment** of the projection to u of a random point x chosen from D.

i.e. find

$$u^* \approx \arg\max_{u \in \Re^n:\, \|u\|=1} \mathop{\mathrm{E}}_{x \sim D}[(u \cdot x)^r].$$



Possible Approaches for Large r

- **Idea 1 (Gradient descent):** Start with some unit vector u. Estimate the gradient (via samples), and move in that direction. Repeat until local maximum is found.
  - Many local maxima. Can we avoid this by taking new samples with each estimate?

- **Idea 2** [Kannan] **(Markov chains)**: Consider a Markov chain that attempts to sample u with density proportional to $e^{E[(x\cdot u)^r]}$. Implement via Metropolis filter. At each step we only need to estimate $E[(x\cdot u)^r]$.
  - Does this Markov chain mix rapidly?

Possible Approaches for Large r

- **Idea 1 (Gradient descent):** Start with some unit vector u. Estimate the gradient (via samples), and move in that direction. Repeat until local maximum is found.
  - Many local maxima. Can we avoid this by taking new samples with each estimate? Our work shows that **NO!**

- **Idea 2** [Kannan] **(Markov chains)**: Consider a Markov chain that attempts to sample u with density proportional to $e^{E[(x\cdot u)^r]}$. Implement via Metropolis filter. At each step we only need to estimate $E[(x\cdot u)^r]$.
  - Does this Markov chain mix rapidly? Our work shows that **NO!**

Possible Approaches for Large r

- **Idea 1 (Gradient descent):** Start with some unit vector u. Estimate the gradient (via samples), and move in that direction. Repeat until local maximum is found.
  - Many local maxima. Can we avoid this by taking new samples with each estimate? Our work shows that **NO!**

- **Idea 2** [Kannan] **(Markov chains)**: Consider a Markov chain that attempts to sample u with density proportional to $e^{E[(x\cdot u)^r]}$. Implement via Metropolis filter. At each step we only need to estimate $E[(x\cdot u)^r]$.
  - Does this Markov chain mix rapidly? Our work shows that **NO!**

(Disclaimer: moment maximization is NP-hard for r > 3 [Brubaker '09])

Statistical Algorithms

- Both approaches fall under a class of **statistical algorithms**.
- In this talk, we will show that for many optimization problems over distributions, statistical algorithms **unconditionally** have complexity **exponential** in their input parameters.
- Our lower bounds use only a single parameter of the optimization problem we call **statistical dimension**.
  - Inspired by the statistical query model in learning theory.
- We shall use our results to give **new lower bounds** for **distribution MAX-XOR-SAT**, **k-clique**, and **moment maximization**.

General Optimization

- **Optimization problems over distributions.** Let $\mathcal{D}$ be the set of input distributions over a domain X and $\mathcal{F}$ be a set of functions $X \rightarrow \mathcal{R}$ over which we want to optimize. An optimization problem **$P(\mathcal{F},\mathcal{D},\varepsilon)$** over an input distribution $D \in \mathcal{D}$ has a solution function $f^* \in \mathcal{F}$ such that $f^* = \text{argmax}_{f \in \mathcal{F}} E_{x\sim D}[f(x)]$.

- For a function $g \in \mathcal{F}$, distribution D, and $\varepsilon > 0$, we say that g is **ε-optimal** for D if

$$E_{x\sim D}[g(x)] \geq E_{x\sim D}[f^*(x)] - \varepsilon.$$

The objective is to ε-optimize over $\mathcal{F}$ w.r.t. D, i.e. to find an ε-optimal $g \in \mathcal{F}$.

Statistical Algorithms and Statistical Dimension

the definitions

Statistical Algorithms Defined

We say an algorithm is **statistical** if it has no direct access to the target distribution D, but instead makes calls to an oracle **$STAT_D$**, which takes as inputs a **query function** $h \in H: X \rightarrow [-1,1]$ and a **tolerance parameter** $\tau > 0$. $STAT_D(h,\tau)$ returns a value

$$v \in [\mathop{\mathrm{E}}_{x\sim D}[h(x)] - \tau, \mathop{\mathrm{E}}_{x\sim D}[h(x)] + \tau].$$

Statistical Algorithms Defined

We say an algorithm is **statistical** if it has no direct access to the target distribution D, but instead makes calls to an oracle **$STAT_D$**, which takes as inputs a **query function** $h \in H: X \rightarrow [-1,1]$ and a **tolerance parameter** $\tau > 0$. $STAT_D(h,\tau)$ returns a value

$$v \in [\underset{x\sim D}{\mathrm{E}}[h(x)] - \tau, \underset{x\sim D}{\mathrm{E}}[h(x)] + \tau].$$

We say an algorithm is **"realistic"** if it interacts with the distribution via an oracle **$SAMPLE_D$**, which takes as inputs a **query function** $h \in H: X \rightarrow [-1,1]$ and a **sample size** $t > 0$. $SAMPLE_D(h,t)$ draws $x_1 \ldots x_t$ i.i.d. from D and returns $\frac{1}{t}\sum_{i=0}^{t} h(x_i)$

Statistical Algorithms Defined

We say an algorithm is **statistical** if it has no direct access to the target distribution D, but instead makes calls to an oracle **$STAT_D$**, which takes as inputs a **query function** $h \in H: X \to [-1,1]$ and a **tolerance parameter** $\tau > 0$. $STAT_D(h,\tau)$ returns a value

$$v \in [\mathop{\mathrm{E}}_{x\sim D}[h(x)] - \tau, \mathop{\mathrm{E}}_{x\sim D}[h(x)] + \tau].$$

$\approx$

We say an algorithm is **"realistic"** if it interacts with the distribution via an oracle **$SAMPLE_D$**, which takes as inputs a **query function** $h \in H: X \to [-1,1]$ and a **sample size** $t > 0$. $SAMPLE_D(h,t)$ draws $x_1 \ldots x_t$ i.i.d. from D and returns $\frac{1}{t}\sum_{i=0}^{t} h(x_i)$

Examples of Statistical Algorithms

- What optimization algorithms can be implemented via statistical estimates?
  - local search
  - k-means
  - simulated annealing
  - EM
  - MCMC
  - gradient descent
  - …
  - (almost anything practical that you can think of has a statistical variant)

A Note on Optimization vs Learning

- For **optimization** we just introduced a definition for *statistical* algorithms. It is inspired by the concept of *statistical query* algorithms [Kearns 1993] from **learning** theory.
- In **learning**, examples come from some distribution and are labeled by an unknown concept. Therefore, there can exist hard distributions. In **optimization**, for any fixed input distribution, there is a fixed answer. Hence, we can't have a "hard distribution."
- In **learning**, for many classes, the uniform distribution a hard distribution. In **optimization**, the uniform distribution is usually trivial (consider moment maximization).
- In **learning**, it is sometimes reasonable to wish to learn the target *exactly*. In **optimization**, usually we're interested in *approximating* the optimum (in our case additive).

Statistical Dimension

- **Learning** a class with statistical queries is hard if there is a distribution, under which the class contains many (nearly) **pairwise uncorrelated functions** [Blum et al. '94].
- For **optimization**, we will want something similar, but for distributions instead of labeling functions.
  - We will want there to be **many possible "uncorrelated" input distributions**, such that eliminating one distribution as the real input will not help in eliminating others.

Notation

- For two functions g,f: $X \rightarrow \mathcal{R}$ and a distribution D with probability density function D(x), define their inner product with respect to D to be

$$\langle f,g\rangle_D := E_{x\sim D}[f(x)g(x)].$$

- The norm of f over D is

$$||f||_D := \langle f,f\rangle_D^{1/2}.$$

We will often omit D when it is clear from context.

Statistical Dimension

- For ε, γ, β > 0, domain X, class of functions $\mathcal{F}$, and a class of distributions $\mathcal{D}$ over X, let m be the maximum s.t. there exists **a reference distribution D** over X s.t. for every f $\in \mathcal{F}$ there exists a **set of m distributions D_f** = $\{D_1 \ldots D_m\} \in \mathcal{D}$ satisfying:
  1. f is not ε -optimal for any D_i for i $\in$ {1…m}
  2. $$\left\langle \frac{D_i}{D} - 1, \frac{D_j}{D} - 1 \right\rangle_D \leq \begin{cases} \beta \text{ for } i = j \in [m] \\ \gamma \text{ for } i \neq j \in [m] \end{cases}$$

- We define the **statistical dimension** of ε-optimizing over F, denoted **SD($\mathcal{F}$,$\mathcal{D}$,ε,γ,β)**, to be **m**.

Lower bound for problems with high statistical dimension.

a theorem and proof

Main Theorem

Theorem: If for a class of functions $\mathcal{F}$, class of distributions $\mathcal{D}$, and ε, γ, $\beta > 0$, $SD(\mathcal{F},\mathcal{D},\varepsilon,\gamma,\beta) = m$, then **at least $m(\tau - \gamma)/\beta$ calls of tolerance τ** to the STAT oracle are **required** to ε-optimize over $\mathcal{F}$ and $\mathcal{D}$.

Proof of Theorem

Theorem: If for a class of functions $\mathcal{F}$, class of distributions $\mathcal{D}$, and $\varepsilon, \gamma, \beta > 0$, $SD(\mathcal{F},\mathcal{D},\varepsilon,\gamma,\beta) = m$, then **at least $m(\tau - \gamma)/\beta$ calls of tolerance τ** to the STAT oracle are **required** to ε-optimize over $\mathcal{F}$ and $\mathcal{D}$.

proof strategy *(inspired by argument of Szörényi [2009])*

- Let $\mathcal{A}$ be an algorithm that ε-optimizes over $\mathcal{F}$ and $\mathcal{D}$.
- We see what happens if we run $\mathcal{A}$ and always answer $E_D[h(x)]$.
  - Let f be the output of $\mathcal{A}$.
  - Let $D_1 \ldots D_m$ be the m distributions on which f is not ε-optimal.
- $\mathcal{A}$ needs to ask enough queries as to "eliminate" m D_i's.
- This will turn out to be a lot of queries!

Proof, Continued

- Let $h_1 \ldots h_q$ be the q queries (of tolerance τ) asked by $\mathcal{A}$ in the simulation.
- For every $k \leq q$ let A_k be the set of distributions D_i such that $|E_D[h_k(x)] - E_{D_i}[h_k(x)]| > \tau$.
- We will prove:
  1. $\sum_{k \leq q} |A_k| \geq m$
  2. for every k, $|A_k| \leq \beta/(\tau^2 - \gamma)$
- These together immediately imply the theorem.

Proof, Continued

- Let $h_1 \ldots h_q$ be the q queries (of tolerance τ) asked by $\mathcal{A}$ in the simulation.

- For every $k \leq q$ let A_k be the set of distributions D_i such that $|E_D[h_k(x)] - E_{D_i}[h_k(x)]| > \tau$.

- We will prove:

  1. $\sum_{k \leq q} |A_k| \geq m$ ← **not hard to argue**

  2. for every k, $|A_k| \leq \beta/(\tau^2 - \gamma)$

- These together immediately imply the theorem.

Proof, Continued

need to prove: for every k, $|A_k| \leq \beta/(\tau^2 - \gamma)$

---

note that: $\mathop{\mathbf{E}}_{D_i}[h_k(x)] - \mathop{\mathbf{E}}_{D}[h_k(x)] = \mathop{\mathbf{E}}_{D}\left[\frac{D_i}{D}\, h_k(x)\right] - \mathop{\mathbf{E}}_{D}[h_k(x)] = \left\langle h_k, \frac{D_i}{D} - 1 \right\rangle$

and define $\hat{D}_i(x) = \dfrac{D_i(x)}{D(x)} - 1$

Proof, Continued

need to prove: for every k, $|A_k| \leq \beta/(\tau^2 - \gamma)$

---

note that: $\mathop{\mathbf{E}}_{D_i}[h_k(x)] - \mathop{\mathbf{E}}_{D}[h_k(x)] = \mathop{\mathbf{E}}_{D}\left[\frac{D_i}{D}\, h_k(x)\right] - \mathop{\mathbf{E}}_{D}[h_k(x)] = \left\langle h_k, \frac{D_i}{D} - 1\right\rangle$

and define $\hat{D}_i(x) = \frac{D_i(x)}{D(x)} - 1$

---

by Cauchy-Schwartz we have

$$\begin{aligned}\left\langle h_k, \sum_{i \in A_k} \hat{D}_i \cdot \mathrm{sign}\langle h_k, \hat{D}_i\rangle \right\rangle^2 &\leq \|h_k\|^2 \cdot \left\|\sum_{i \in A_k} \hat{D}_i \cdot \mathrm{sign}\langle h_k, \hat{D}_i\rangle\right\|^2 \\ &\leq 1 \cdot \left(\sum_{i \in A_k} \|\hat{D}_i\|^2 + \sum_{i \neq j \in A_k} \langle \hat{D}_i, \hat{D}_j\rangle\right) \\ &\leq \beta|A_k| + \gamma(|A_k|^2 - |A_k|)\end{aligned}$$

Proof, Continued

need to prove: for every k, $|A_k| \le \beta/(\tau^2 - \gamma)$

note that: $\mathop{\mathbf{E}}_{D_i}[h_k(x)] - \mathop{\mathbf{E}}_{D}[h_k(x)] = \mathop{\mathbf{E}}_{D}\left[\frac{D_i}{D} h_k(x)\right] - \mathop{\mathbf{E}}_{D}[h_k(x)] = \left\langle h_k, \frac{D_i}{D} - 1 \right\rangle$

and define $\hat{D}_i(x) = \frac{D_i(x)}{D(x)} - 1$

$$\left\langle h_k, \sum_{i \in A_k} \hat{D}_i \cdot \mathrm{sign}\langle h_k, \hat{D}_i \rangle \right\rangle^2 \le \beta|A_k| + \gamma|A_k|^2$$

Proof, Continued

need to prove: for every k, $|A_k| \leq \beta/(\tau^2 - \gamma)$

---

note that: $\mathop{\mathbf{E}}_{D_i}[h_k(x)] - \mathop{\mathbf{E}}_{D}[h_k(x)] = \mathop{\mathbf{E}}_{D}\left[\frac{D_i}{D}\, h_k(x)\right] - \mathop{\mathbf{E}}_{D}[h_k(x)] = \left\langle h_k, \frac{D_i}{D} - 1 \right\rangle$

and define $\hat{D}_i(x) = \frac{D_i(x)}{D(x)} - 1$

$$\left\langle h_k, \sum_{i \in A_k} \hat{D}_i \cdot \mathsf{sign}\langle h_k, \hat{D}_i \rangle \right\rangle^2 \quad \leq \quad \beta|A_k| + \gamma|A_k|^2$$

$$\left\langle h_k, \sum_{i \in A_k} \hat{D}_i \cdot \mathsf{sign}\langle h_k, \hat{D}_i \rangle \right\rangle^2 \quad = \quad \left(\sum_{i \in A_k} \langle h_k, \hat{D}_i \rangle \cdot \mathsf{sign}\langle h_k, \hat{D}_i \rangle \right)^2$$

$$\geq \quad \tau^2 |A_k|^2$$

Proof, Continued

need to prove: for every k, $|A_k| \le \beta/(\tau^2 - \gamma)$

---

note that: $\mathop{\mathbf{E}}_{D_i}[h_k(x)] - \mathop{\mathbf{E}}_{D}[h_k(x)] = \mathop{\mathbf{E}}_{D}\left[\frac{D_i}{D}\, h_k(x)\right] - \mathop{\mathbf{E}}_{D}[h_k(x)] = \left\langle h_k, \frac{D_i}{D} - 1\right\rangle$

and define $\hat{D}_i(x) = \dfrac{D_i(x)}{D(x)} - 1$

$$\left\langle h_k, \sum_{i \in A_k} \hat{D}_i \cdot \mathrm{sign}\langle h_k, \hat{D}_i\rangle \right\rangle^2 \le \beta|A_k| + \gamma|A_k|^2$$

$$\left\langle h_k, \sum_{i \in A_k} \hat{D}_i \cdot \mathrm{sign}\langle h_k, \hat{D}_i\rangle \right\rangle^2 \ge \tau^2|A_k|^2$$

Proof, Continued

need to prove: for every k, $|A_k| \leq \beta/(\tau^2 - \gamma)$

---

note that: $\mathop{\mathbf{E}}_{D_i}[h_k(x)] - \mathop{\mathbf{E}}_{D}[h_k(x)] = \mathop{\mathbf{E}}_{D}\left[\frac{D_i}{D}\, h_k(x)\right] - \mathop{\mathbf{E}}_{D}[h_k(x)] = \left\langle h_k, \frac{D_i}{D} - 1 \right\rangle$

and define $\hat{D}_i(x) = \frac{D_i(x)}{D(x)} - 1$

$$\left\langle h_k, \sum_{i \in A_k} \hat{D}_i \cdot \mathrm{sign}\langle h_k, \hat{D}_i \rangle \right\rangle^2 \quad \leq \quad \beta|A_k| + \gamma|A_k|^2$$

$$\left\langle h_k, \sum_{i \in A_k} \hat{D}_i \cdot \mathrm{sign}\langle h_k, \hat{D}_i \rangle \right\rangle^2 \quad \geq \quad \tau^2|A_k|^2$$

$$\beta|A_k| + \gamma|A_k|^2 \quad \geq \quad \tau^2|A_k|^2$$

$\square$

What we just proved

Theorem: If for a class of functions $\mathcal{F}$, class of distributions $\mathcal{D}$, and $\varepsilon, \gamma, \beta > 0$, $SD(\mathcal{F}, \mathcal{D}, \varepsilon, \gamma, \beta) = m$, then **at least $m(\tau - \gamma)/\beta$ calls of tolerance τ** to the STAT oracle are **required** to ε-optimize over $\mathcal{F}$ and $\mathcal{D}$.

two notes

1. This also gives a lower bound for a "realistic" sampling algorithm.
2. Strictly generalizes the statistical query bounds in learning (not at all obvious).
   In fact, this gives a stronger lower bound for learning (by a factor of 2).

Applications

to MAX-XOR-SAT, k-clique, and moment maximization

Parity

- For $x \in \{0,1\}^n$ and $c \in \{0,1\}^n$, let $\chi_c(x)$ take the value 1 if $c \cdot x$ is odd and -1 otherwise.
- Let D_c be the uniform distribution over x such that $c \cdot x = 1 \pmod 2$.
- Two useful known facts about parities

  Proposition 1: $E_{x \sim D_c}[\chi_{c'}(x)] = 0$ if $c' \neq c$.

  Proposition 2: $E_{x \sim U}[\chi_c(x)\chi_{c'}(x)] = 0$ if $c \neq c'$.

MAX-XOR-SAT

- **Problem**: Let D be a distribution over XOR clauses $c \in \{0,1\}^n$ ($c_i=1$ means variable i appears in c). The problem is to find an assignment $x \in \{0,1\}^n$ that **maximizes the expected number of satisfied clauses**.
  - Clause c is **satisfied** by assignment x if $\chi_c(x)=1$.
  - Similar to the parity problem in learning, but the **distribution is over *clauses*** (e.g. parity functions).

Clause	Prob.
$c1 \oplus c3 \oplus c4$	1/2
$c1 \oplus c2$	1/8
$c4$	1/4
$c1 \oplus c2 \oplus c4$	1/16
$c2 \oplus c4$	1/16

The assignment
x1 = 1; **x2 = 0**; **x3=1**; **x4 = 1**
has probability 15/16 of satisfying a clause.

MAX-XOR-SAT

- **Problem**: Let D be a distribution over XOR clauses $c \in \{0,1\}^n$ ($c_i=1$ means variable i appears in c). The problem is to find an assignment $x \in \{0,1\}^n$ that **maximizes the expected number of satisfied clauses**.
  - Clause c is **satisfied** by assignment x if $\chi_c(x)=1$.
  - Similar to the parity problem in learning, but the **distribution is over *clauses*** (e.g. parity functions).

- **Result**: Any statistical algorithm for MAX-XOR-SAT **requires $2^{n/3}$** queries of tolerance $2^{-n/3}$ to find an assignment that approximates, **to within an additive factor of ½,** the maximum probability of satisfying a clause drawn from an unknown distribution.
  - Holds even when there exists an assignment that satisfies all clauses (with non-zero probability mass according to D).

MAX-XOR-SAT Lower Bound

proof sketch

- For **MAX-XOR-SAT** SD($\mathcal{F}$,$\mathcal{D}$,½,0,1)=2^n-1.
  - The target functions $\mathcal{F}$ are all 2^n possible assignments.
  - The reference distribution D = U.
  - For all 2^n-1 choices for x, the corresponding distributions D_i will be uniform over the clauses c s.t. c·x = 1.
  - Proposition 1 implies that any incorrect assignment satisfies only ½ of the clauses (setting ε = ½).
  - Proposition 2 can be used to show that we can set γ = 0, β =1.

The lower bound then follows from the main theorem.

MAX-XOR-SAT

- **Result**: Any statistical algorithm for MAX-XOR-SAT **requires $2^{n/3}$** queries of tolerance **$2^{-n/3}$** to find an assignment that approximates, **to within an additive factor of ½,** the maximum probability of satisfying a clause drawn from an unknown distribution.
  - Holds even when there exists an assignment that satisfies all clauses (with non-zero probability mass according to D).

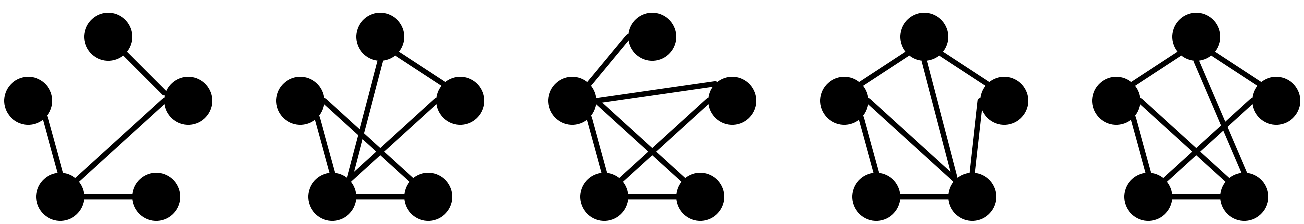
Helps explain the experimental evidence about the performance of algorithms like WalkSat [Selman et al. '95].

k-Clique

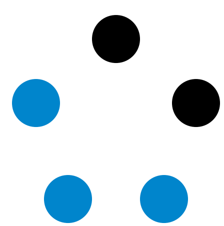
Note: C(n,k) is “n choose k”

- **Problem**: Let D be a distribution over $X = \{0,1\}^{C(n,2)}$, corresponding to graphs G on n vertices. Let **$I_S(G) = 1$ if S induces a clique on G and $I_S(G) = 0$ otherwise**. The problem is to find a subset $S \subseteq V$ that maximizes $E_{G \sim D}[I_S(G)]$.

K=3



1/5 1/5 1/5 1/5 1/5



4/5

k-Clique

- **Problem**: Let D be a distribution over $X = \{0,1\}^{C(n,2)}$, corresponding to graphs G on n vertices. Let **$I_S(G) = 1$ if S induces a clique on G and $I_S(G) = 0$ otherwise**. The problem is to find a subset $S \subseteq V$ that maximizes $E_{G\sim D}[I_s(G)]$.

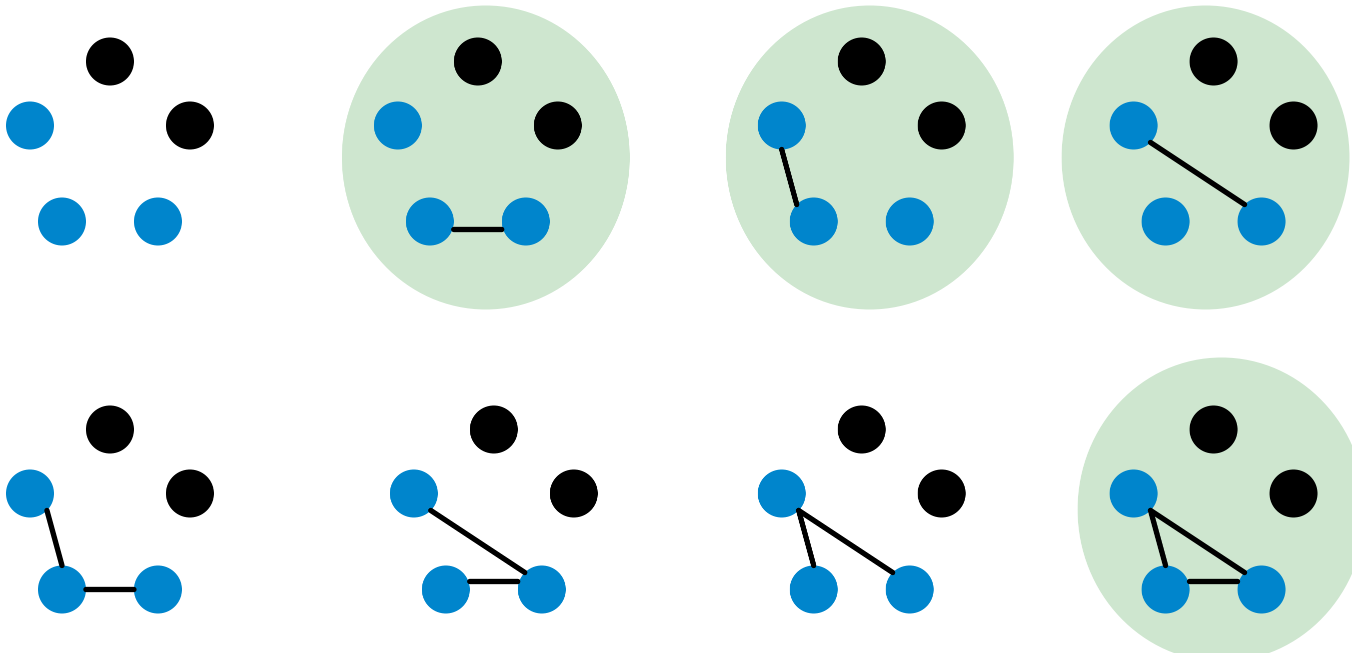
- **Result**: Any statistical algorithm **requires $C(n,k)^{1/3}$** queries of tolerance **$C(n,k)^{-1/3}$** to approximate k-clique, **to within an additive factor of $2^{-C(k,2)}$**.
  - Achieving even a constant factor approximation is hard!

k-Clique Lower Bound

proof idea

- We show that for k-Clique, SD($\mathcal{F}$, $\mathcal{D}$, $2^{-C(k,2)}$,0,1)=C(n,k).
  - For any subset of edges T∊ V × V and graph G define **parity$_T$(G,k) = 1 if |E(G) ∩ T| has same parity as C(k,2)** and **0 otherwise**.
  - $D_1 \ldots D_{C(n,k)}$ (for each subset of k vertices w/ T a clique on that subset) are uniform over all graphs with **parity$_T$(G,k) = 1.**

Cliques and Parity$_T$



Restricting to correct parity on given subset increases probability of a clique from $2^{-C(k,2)}$ to $2^{1-C(k,2)}$, so the "correct parity" corresponds to the "correct clique".

Moment Maximization

- **Problem**: Let D be a distribution over $\{-1,1\}^n$ and $r \in Z^+$. The goal is to find a vector u* that **maximizes the expected r'th moment** of the projection to u of a random point x from D. i.e.

$$u^* = \arg\max_{u \in \Re : ||u||=1} \mathop{\mathrm{E}}_{x \sim D}[(u \cdot x)^r].$$

- **Result**: For r odd, any statistical algorithm for moment maximization **requires $C(n,r)^{1/3}$** queries of tolerance **$C(n,r)^{-1/3}$** to approximate the rth moment **to within ~ $(r/e)^{r/2}$**.

Moment Maximization Lower Bound

proof idea

- Idea is to show that for the moment maximization problem, SD($\mathcal{F}$, $\mathcal{D}$, r!/(2(r+1)$^{r/2}$) ,0,1)=C(n,r)

- **Lemma**: let r ∈ $\mathcal{Z}^+$ be odd and c ∈ {0,1}^{n}. Let D_c be the distribution uniform over x ∈ {-1,1}^{n} for which χ_c(x)=-1. The for all u ∈ $\mathcal{R}$

$$\mathop{\mathrm{E}}_{x\sim D_c}[(x\cdot u)^r] = r!\prod_{i:c_i=1} u_i.$$

- By the AM/GM inequality, the moment under D_c is maximized at the vector uniform on the parity coordinates.

Discussion

- What happens if we fix and reuse a sample?
  - Our lower bounds break down…
  - But intuition and experience indicate that "statistical" algorithms still fail. How do we capture/formalize this?
- Other interesting problems?
  - Perhaps our techniques can be used to prove lower bounds for planted clique.
- Can we design *inherently* non-statistical algorithms?
  - Gaussian elimination.
  - What else?

Thank You!

Questions?