LECTURE 21

Last time

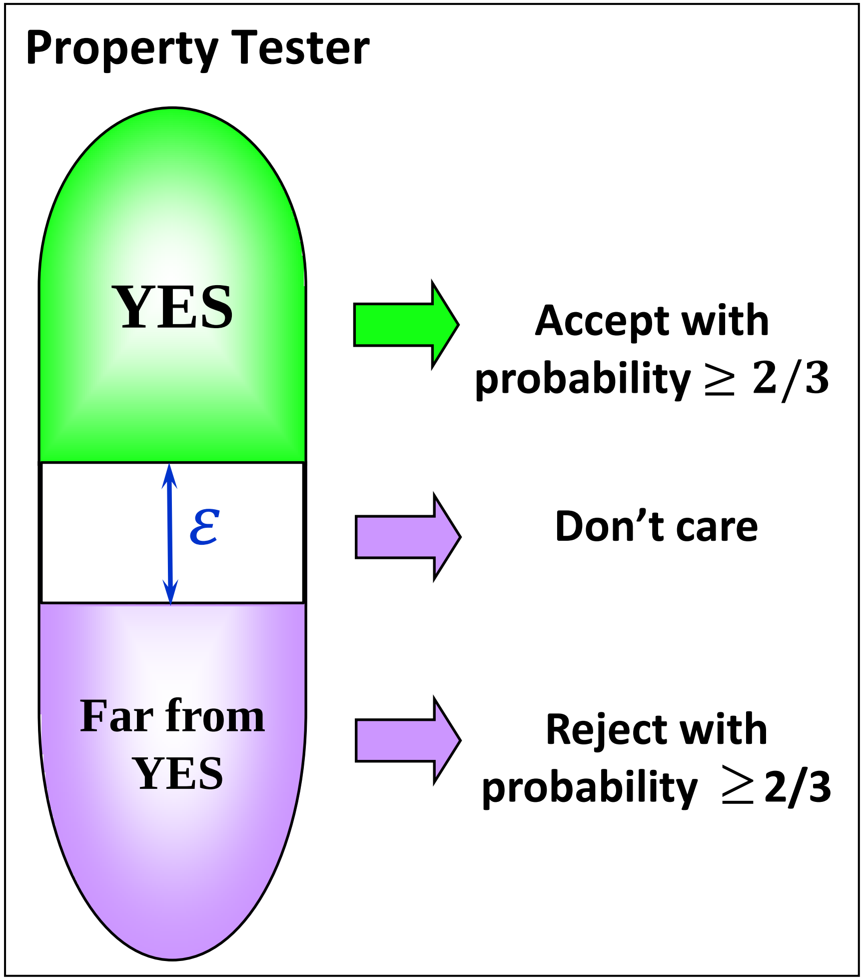
- L_p-testing

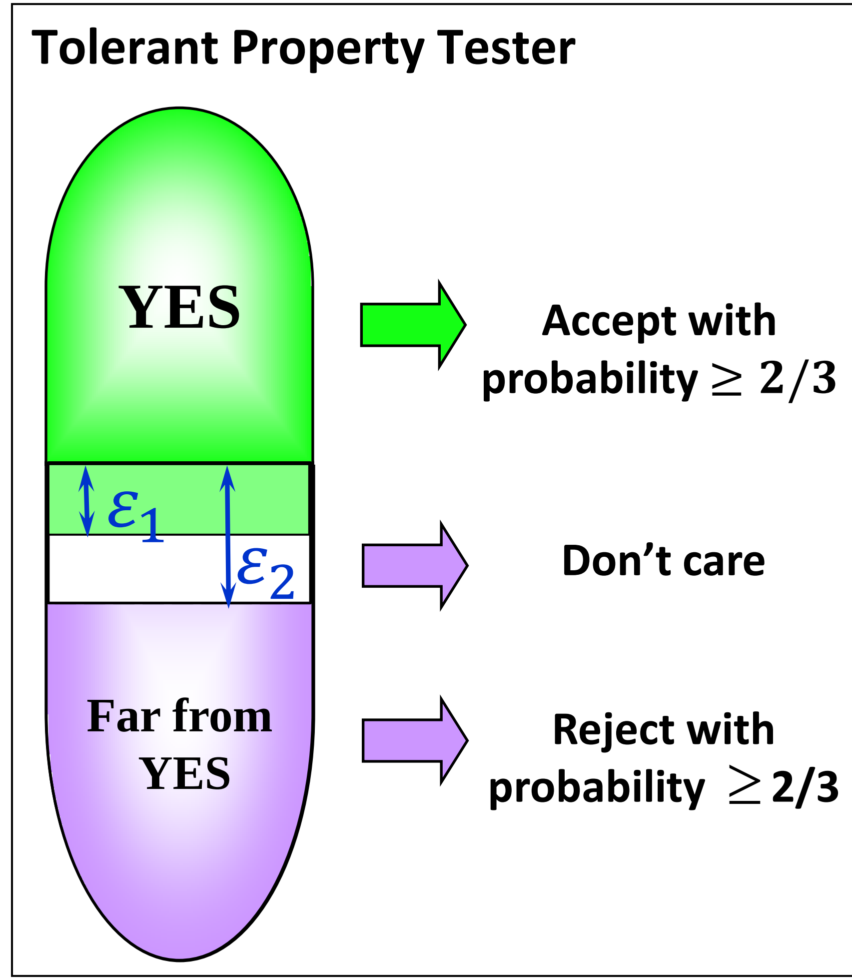
Today

- L_p-testing of monotonicity
- Work investment strategy
- Testing via learning

Project Reports are due April 24

L_p-Testing and Tolerant L_p-Testing



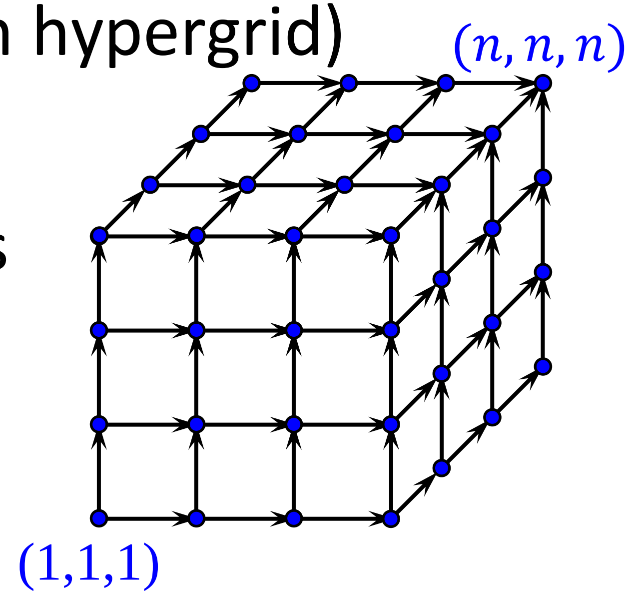


Functions $f, g\colon D \to [0,1]$ are at distance ε if $d_p = \frac{\|f-g\|_p}{\|\mathbf{1}\|_p} = \varepsilon$.

Monotonicity

- Domain D=$[n]^d$ (vertices of d-dim hypergrid)
- A function $f: D \to \mathbb{R}$ is monotone

  if increasing a coordinate of x does not decrease $f(x)$.



- Special case $d = 1$

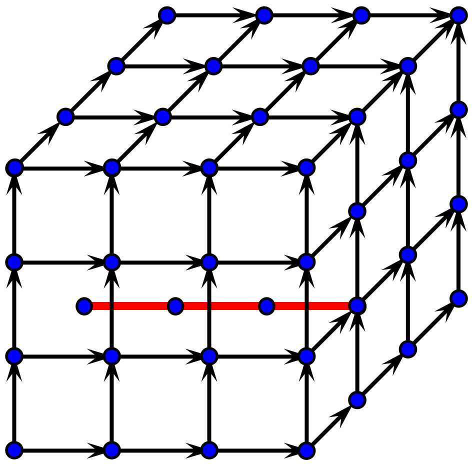
  $f: [n] \to \mathbb{R}$ is monotone $\Leftrightarrow f(1), \ldots f(n)$ is sorted.

Monotonicity Testers: Running Time

f	L_0	L_p
$[n] \to [0,1]$	$\Theta\left(\frac{\log n}{\varepsilon}\right)$ [Ergün Kannan Kumar Rubinfeld Viswanathan 00, Fischer 04, Belovs, Chakrabarty Seshadhri]	$\Theta\left(\frac{1}{\varepsilon^p}\right)$
$[n]^d \to [0,1]$	$\Theta\left(\frac{d \cdot \log n}{\varepsilon}\right)$ [Chakrabarty Seshadhri 13]	$O\left(\min\left\{\frac{d}{\varepsilon^p}\log\frac{d}{\varepsilon^p}, \frac{d^{1/2+o(1)}}{\varepsilon^{2p}}\right\}\right)$ $\Omega\left(\frac{1}{\varepsilon^p}\log\frac{1}{\varepsilon^p}\right)$ for $d = 2$ nonadaptive 1-sided error [Berman Raskhodnikova Yaroslavtsev 14, Black Chakrabarty Seshadhri 23]

* Hiding some $\log 1/\varepsilon$ dependence

L_0-Testing Monotonicity of $f: [n]^d \to \{0,1\}$



Idea: 1. Pick axis-parallel lines ℓ.

2. Sample points from each ℓ,

and check for violations of $f_{|\ell}$.

[DGLRRS 99]

- Testing sortedness: If $f: [n] \to \{0,1\}$ is ε-far from sorted then $O\left(\frac{1}{\varepsilon}\right)$ samples are sufficient to find a violation w/ const. prob.
- Dimension reduction: For $f: [n]^d \to \{0,1\}$

$$\mathbb{E}\left[d_0\left(f_{|\ell}, M\right)\right] \geq \frac{d_0(f,M)}{2d}.$$

How many lines should we sample?

How many points form each line?

General Work Investment Problem [Goldreich 13]

- Algorithm needs to find ``evidence'' (e.g., **a violation**).
- It can select an element from distr. Π (e.g., **a uniform line**).
- Elements e have different quality $q(e) \in [0,1]$

  (e.g., $\boldsymbol{d_0(f_{|\ell}, M)}$).
- Algorithm must invest more work into e with lower $q(e)$ to extract evidence from e (e.g., **need $\boldsymbol{\Theta\left(\frac{1}{q(e)}\right)}$ samples**).
- $\mathbb{E}_{\boldsymbol{e \leftarrow \Pi}}[q(e)] \geq \mu$.

What's a good work investment strategy?

Used in [Levin 85, Goldreich Levin 89], testing connectedness of a graph [Goldreich Ron 97], testing properties of images [R 03], multi-input testing problems [G13]

Work Investment Strategies

- ``Reverse'' Markov Inequality

For a random variable $X \in [0,1]$ with expectation $\mathbb{E}[X] \geq \mu$,

$$\Pr\left[X \geq \frac{\mu}{2}\right] \geq \frac{\mu}{2}.$$

Proof: $\mu \leq \mathbb{E}[X] \leq \Pr\left[X \geq \frac{\mu}{2}\right] \cdot 1 + \Pr\left[X < \frac{\mu}{2}\right] \cdot \frac{\mu}{2}.$

``Reverse'' Markov Strategy:

1. Sample $\Theta\left(\frac{1}{\mu}\right)$ lines.
2. Sample $\Theta\left(\frac{1}{\mu}\right)$ points from each line.

Cost: $\Theta\left(\frac{1}{\mu^2}\right)$ queries.

Work Investment Strategies

Bucketing idea [Levin, Goldreich 13]:

Invest in elements of quality $q(e) \geq \frac{1}{2^i}$ separately.

Bucketing Inequality [Berman R Yaroslavtsev 14]

For a random variable $X \in [0,1]$ with $\mathbb{E}[X] \geq \mu$, let

$$p_i = \Pr\left[X \geq \frac{1}{2^i}\right] \text{ and } k_i = \Theta\left(\frac{1}{2^i \mu}\right) \text{ for all } i \in \left[\log \frac{4}{\mu}\right].$$

Then $\prod_{i=1}^{\log 4/\mu} (1 - p_i)^{k_i} \leq 1/3$.

Bucketing Strategy: For each bucket $i \in \left[\log \frac{4}{\mu}\right]$

1. Sample $k_i = \Theta\left(\frac{1}{2^i \mu}\right)$ lines.
2. Sample $\Theta(2^i)$ points from each line.

Cost: $\Theta\left(\frac{1}{\mu} \log \frac{1}{\mu}\right)$ queries (**for monotonicity, $\mu = \frac{\varepsilon}{2d}$**)

Proof of Bucketing Inequality

Bucketing Inequality [Berman R Yaroslavtsev 14]

For a random variable $X \in [0,1]$ with $\mathbb{E}[X] \geq \mu$, let

$$t = \log\frac{4}{\mu}, \quad p_i = \Pr\left[X \geq \frac{1}{2^i}\right], \quad \text{and } k_i = \frac{4 \ln 1/\delta}{2^i \mu}.$$

Then $\prod_{i=1}^{t}(1-p_i)^{k_i} \leq \delta$.

Proof: It suffices to prove $\sum_{i\in[t]} \frac{p_i}{2^i} \geq \frac{\mu}{4}$ because then

$$\prod_{i\in[t]}(1-p_i)^{k_i} \leq \prod_{i\in[t]} e^{-p_i k_i} = \exp\left(-\sum_{i\in[t]} p_i k_i\right)$$

$$= \exp\left(-\sum_{i\in[t]} p_i \cdot \frac{4\ln 1/\delta}{2^i \mu}\right) = \exp\left(-\frac{4 \ln 1/\delta}{\mu}\sum_{i\in[t]} \frac{p_i}{2^i}\cdot\right)$$

$$\leq \exp\left(-\frac{4\ln 1/\delta}{\mu}\cdot\frac{\mu}{4}\right)$$

Proof of Bucketing Inequality (Continued)

Bucketing Inequality [Berman R Yaroslavtsev 14]

For a random variable $X \in [0,1]$ with $\mathbb{E}[X] \geq \mu$, let

$$t = \log \frac{4}{\mu}, \quad p_i = \Pr\left[X \geq \frac{1}{2^i}\right], \quad \text{and } k_i = \frac{4 \ln 1/\delta}{2^i \mu}.$$

Then $\prod_{i=1}^{t} (1 - p_i)^{k_i} \leq \delta$.

Proof: It suffices to prove $\sum_{i \in [t]} \frac{p_i}{2^i} \geq \frac{\mu}{4}$.

$$\sum_{i=1}^{\infty} \frac{p_i}{2^i} = \frac{1}{2} \sum_{i=1}^{\infty} \frac{1}{2^{i-1}} \Pr\left[X \geq \frac{1}{2^i}\right] \geq \frac{1}{2} \sum_{i=1}^{\infty} \frac{1}{2^{i-1}} \Pr\left[X \in \left(\frac{1}{2^i}, \frac{1}{2^{i-1}}\right]\right]$$

$$\geq \frac{1}{2} \mathbb{E}[X] \geq \frac{\mu}{2}$$

$$\sum_{i=t+1}^{\infty} \frac{p_i}{2^i} \leq \sum_{i=t+1}^{\infty} \frac{1}{2^i} \leq \frac{1}{2^t} \leq \frac{\mu}{4}$$

$$\sum_{i \in [t]} \frac{p_i}{2^i} = \sum_{i=1}^{\infty} \frac{p_i}{2^i} - \sum_{i=t+1}^{\infty} \frac{p_i}{2^i} \geq \frac{\mu}{2} - \frac{\mu}{4} \geq \frac{\mu}{4}$$

Monotonicity Testers: Running Time

f	L_0	L_p
$[n] \to \{0,1\}$	$\Theta\left(\frac{1}{\boldsymbol{\varepsilon}}\right)$	$\Theta\left(\frac{1}{\boldsymbol{\varepsilon}^p}\right)$
$[n]^d \to \{0,1\}$	$O\left(\frac{d}{\boldsymbol{\varepsilon}} \cdot \log \frac{d}{\boldsymbol{\varepsilon}}\right)$ [Berman Raskhodnikova Yaroslavtsev 14] $O\left(\frac{d^{1/2+o(1)}}{\varepsilon^2}\right)$ [Black Chakrabarty Seshadhri 23]	$O\left(\min\left\{\frac{d}{\boldsymbol{\varepsilon}^p} \log \frac{d}{\boldsymbol{\varepsilon}^p}, \frac{d^{1/2+o(1)}}{\varepsilon^{2p}}\right\}\right)$ $\Omega\left(\frac{1}{\boldsymbol{\varepsilon}^p} \log \frac{1}{\boldsymbol{\varepsilon}^p}\right)$ for $d = 2$ nonadaptive 1-sided error $\Theta\left(\frac{1}{\boldsymbol{\varepsilon}^p}\right)$ for constant d adaptive 1-sided error

Testing Monotonicity of $f: [n]^2 \to \{0, 1\}$

- For nonadaptive, 1-sided error testers, $\Omega\left(\frac{1}{\varepsilon}\log\frac{1}{\varepsilon}\right)$ queries are needed.
- There is an adaptive, 1-sided error tester with $O\left(\frac{1}{\varepsilon}\right)$ queries. Method: testing via learning.

Partial Learning

- An ε-partial function g with domain D and range R is a function $g : D \rightarrow R \cup \{?\}$ that satisfies $\Pr_{x \in D}[g(x) = ?] \leq \varepsilon$.
- An ε-partial function g agrees with a function f if $g(x) = f(x)$ for all x on which $g(x) \neq ?$.
- Given a function class $\mathcal{C}$, let $\mathcal{C}_\varepsilon$ denote the class of ε-partial functions, each of which agrees with some function in $\mathcal{C}$.
- An ε-partial learner for a function class $\mathcal{C}$ is an algorithm that, given a parameter ε and oracle access to a function f, outputs a hypothesis $g \in \mathcal{C}_\varepsilon$ or fails. Moreover, if $f \in \mathcal{C}$ then it outputs g that agrees with f.

Lemma (Conversion from Learner to Tester)

If there is an ε-partial learner for a function class $\mathcal{C}$ that makes $q(\varepsilon)$ queries then $\mathcal{C}$ can be ε-tested with 1-sided error with $q(\varepsilon/2) + O(1/\varepsilon)$ queries.

Proof of the Conversion Lemma

Lemma (Conversion from Learner to Tester)

If there is an ε-partial learner for a function class $\boldsymbol{C}$ that makes $q(\varepsilon)$ queries then $\boldsymbol{C}$ can be ε-tested with 1-sided error with $q(\varepsilon/2) + O(1/\varepsilon)$ queries.

Proof:

Tester (**Input:** ε, D; query $access\ to\ function\ f\ on\ domain\ D$)

1. Run the learner with parameter $\frac{\varepsilon}{2}$ to get an $\frac{\varepsilon}{2}$-partial function g.
2. If the learner fails, **reject**.
3. Repeat $\frac{2\ln 3}{\varepsilon}$ times:
4. Query f at a uniformly random point $x \in D$.
5. **Reject** if $g(x) \neq ?$, but $g(x) \neq f(x)$.
6. **Accept.**

Correctness:

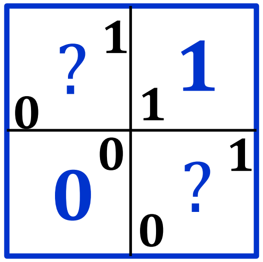
1. If $f \in \boldsymbol{C}$, then the learner outputs a hypothesis g that agrees with f on all non-question-marks. So, the tester accepts.
2. If f is ε-far from $\boldsymbol{C}$, then the learner either fails or outputs $g \in \boldsymbol{C}_{\boldsymbol{\varepsilon/2}}$.

In the latter case, g differs from f on $\geq \varepsilon$ fraction of positions, at most $\varepsilon/2$ of which can be ?'s. The Witness lemma implies probability of rejection $\geq 2/3$

Partial Learner of Monotone functions $f: [n]^2 \to \{0,1\}$

Lemma

There is an ε-partial learner for the class of monotone Boolean functions over $[n]^2$ that makes $O(1/\varepsilon)$ queries.



Idea:

- Divide the grid into quarters.
- Query the bottom left and the top right corner for each quarter.
- If the value of the function is NOT determined by the corners, recurse.

Details: Keep a quad tree and stop at $\log\frac{1}{\varepsilon} + 1$ levels.

- If $\geq 2^{j+1}$ nodes at level j are ?, fail.

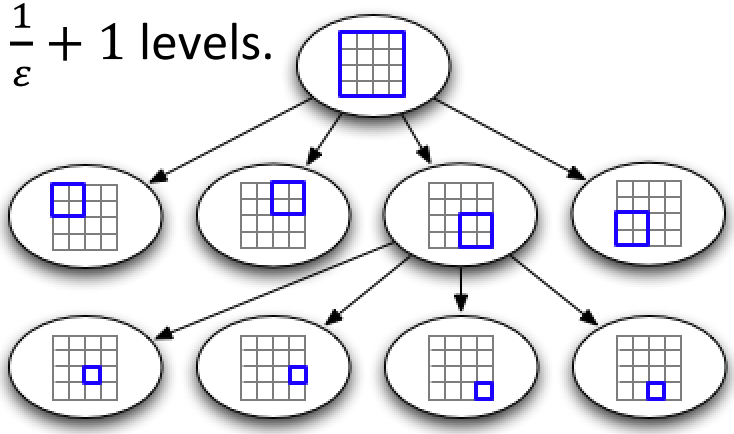


Image credit: Pradeep Pujari

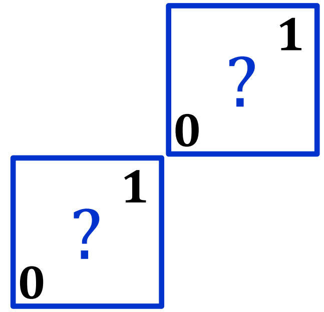
Correctness of the Learner

Claim

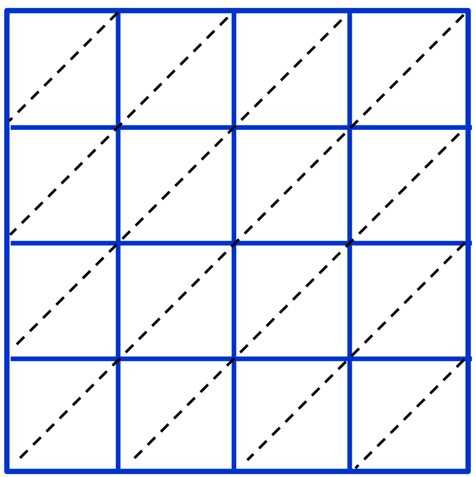
If the input function is monotone, level j will have fewer than 2^{j+1} nodes ?.

Proof: Suppose f is monotone.

- Fix level j. It partitions the domain into $2^j \times 2^j$ squares.
- Two comparable squares cannot both have ?s
- At most one square from each diagonal can have a ?



2^j



2^j

Cor. 1. Learner does not fail on monotone functions.

Cor. 2. Learner outputs an ε-partial function.

Cor. 3. Learner's run time is $O(1/\varepsilon)$.

Monotonicity Testers: Running Time

f	L_0	L_p
$[n] \to \{0,1\}$	$\Theta\left(\frac{1}{\boldsymbol{\varepsilon}}\right)$	$\Theta\left(\frac{1}{\boldsymbol{\varepsilon}^p}\right)$
$[n]^d \to \{0,1\}$	$O\left(\frac{d}{\boldsymbol{\varepsilon}} \cdot \log \frac{d}{\boldsymbol{\varepsilon}}\right)$ [Berman Raskhodnikova Yaroslavtsev 14] $O\left(\frac{d^{1/2+o(1)}}{\varepsilon^2}\right)$ [Black Chakrabarty Seshadhri 23]	$O\left(\min\left\{\frac{d}{\boldsymbol{\varepsilon}^p}\log\frac{d}{\boldsymbol{\varepsilon}^p}, \frac{d^{1/2+o(1)}}{\varepsilon^{2p}}\right\}\right)$ $\Omega\left(\frac{1}{\boldsymbol{\varepsilon}^p}\log\frac{1}{\boldsymbol{\varepsilon}^p}\right)$ for $d = 2$ nonadaptive 1-sided error $\Theta\left(\frac{1}{\boldsymbol{\varepsilon}^p}\right)$ for constant d adaptive 1-sided error

Monotonicity Testers: Running Time

f	L_0	L_p
$[n] \to [0,1]$	$\Theta\left(\frac{\log n}{\varepsilon}\right)$ [Ergün Kannan Kumar Rubinfeld Viswanathan 00, Fischer 04, Belovs, Chakrabarty Seshadhri]	$\Theta\left(\frac{1}{\varepsilon^p}\right)$
$[n]^d \to [0,1]$	$\Theta\left(\frac{d \cdot \log n}{\varepsilon}\right)$ [Chakrabarty Seshadhri 13]	$O\left(\min\left\{\frac{d}{\varepsilon^p}\log\frac{d}{\varepsilon^p}, \frac{d^{1/2+o(1)}}{\varepsilon^{2p}}\right\}\right)$ $\Omega\left(\frac{1}{\varepsilon^p}\log\frac{1}{\varepsilon^p}\right)$ for $d = 2$ nonadaptive 1-sided error [Berman Raskhodnikova Yaroslavtsev 14, Black Chakrabarty Seshadhri 23]

* Hiding some $\log 1/\varepsilon$ dependence

Distance Approximation and Tolerant Testing

Approximating L_1-distance to monotonicity $\pm\varepsilon$ $w.p. \geq 2/3$

f	L_0	L_1
$[n] \rightarrow [0,1]$	polylog $n \cdot \left(\frac{1}{\varepsilon}\right)^{O(1/\varepsilon)}$ [Saks Seshadhri 10]	$\Theta\left(\frac{1}{\varepsilon^2}\right)$

- Time complexity of tolerant L_1-testing for monotonicity is

$$O\left(\frac{\varepsilon_2}{(\varepsilon_2 - \varepsilon_1)^2}\right).$$

Open Problems

- Our L_1-tester for monotonicity is nonadaptive, but adaptivity helps for Boolean range.

Is there a better adaptive tester?

- All our algorithms for L_p-testing for $p \geq 1$ were obtained directly from L_1-testers.

Can one design better algorithms by working directly with L_p-distances?

- Distance to monotoniciy of $f: \{0,1\}^d \to \{0,1\}$ can be approximated with $\tilde{O}\left(\sqrt{d}\right)$ additive error with poly(d, ε) assuming f is ε-far from monotone [Pallavoor R Waingarten 21]

Take advantage of adaptivity?

Other properties?