CSCI 1951-W Sublinear Algorithms for Big Data **Fall 2020**

Lecture 15: Other Models in Distributional Testing and Learning

Lecturer: Jasper Lee *Scribe: Danny Li*

1 Instance Optimality

In property testing, and theoretical computer science in general, typical analyses of algorithms give worst-case optimality. For example, consider the problem of identity testing against a distribution $\mathbf{q}$ over $[n]$ versus being ϵ-far from $\mathbf{q}$. We formulate the upper bound and lower bound of the algorithm as follows:

Upper bound $\forall$ distribution $\mathbf{q}$ over $[n]$, identity testing takes only $c_1 \cdot f(n, \epsilon)$ samples, where $f(n, \epsilon) = \frac{\sqrt{n}}{\epsilon^2}$.

Lower bound $\exists$ distribution $\mathbf{q}$ over $[n]$ such that identity testing takes at least $c_2 \cdot f(n, \epsilon)$ samples, where $f(n, \epsilon) = \frac{\sqrt{n}}{\epsilon^2}$ as well.

As we can see, the analysis is implicitly identifying tester performance with its "worst-case" performance over distributions $\mathbf{q}$ (i.e. identifying tester performance with a $\forall$ quantifier). However, it is reasonable to ask for an algorithm that performs much better on "easier" distributions than in the "worst case". This is the notion of instance optimality, which replaces the complexity measure $f(n, \epsilon)$ with $f'(\mathbf{q}, \epsilon)$, which is a much stronger notion of optimality since it is defined on the specific instance being tested. We next give an example of one such result.

Definition 15.1 Given a distribution $\mathbf{q}$ over $[n]$, define $\mathbf{q}_{-\epsilon}^{-\max}$ as a vector that removes 1 maximum element from $\mathbf{q}$ and also the elements with the smallest mass, stopping before more than ϵ mass is removed.

With Definition 15.1 comes the instance optimal version of identity testing:

Theorem 15.2 (Valiant and Valiant, 2017) *There exists a tester $\mathcal{A}$, universal constants c_1, c_2 such that for any $\epsilon > 0$, known distribution $\mathbf{q}$ over $[n]$, $\mathcal{A}$ tests identity to $\mathbf{q}$ (versus ϵ-far from $\mathbf{q}$) with probability $\geq \frac{2}{3}$ when run on $c_1 \cdot \max\left(\frac{1}{\epsilon}, \frac{\|\mathbf{q}_{-\epsilon/16}^{-\max}\|_{2/3}}{\epsilon^2}\right)$ samples.*

No tester can test identity to $\mathbf{q}$ (versus ϵ-far from $\mathbf{q}$) with probability $\geq \frac{2}{3}$ when run on fewer than $c_2 \cdot \max\left(\frac{1}{\epsilon}, \frac{\|\mathbf{q}_{-\epsilon}^{-\max}\|_{2/3}}{\epsilon^2}\right)$ samples.

Note that the corresponding tester, though unstated here, is also an adaptation of the χ^2-test and is easy to implement (which is a feature, not a bug!).

2 Tolerant Testing

In standard property testing, we only consider the gap to be on one side: we look for a tester such that given an object O (graph, or distribution, or otherwise), it tests

- $O \in \mathcal{P}$, versus
- $d(O, \mathcal{P}) \geq \epsilon$ with an ϵ gap

However, in the framework of tolerant testing, we allow the gap to be 2-sided: we look for a tester such that, given an object O, it tests

- $d(O, \mathcal{P}) \leq \epsilon_1$, versus
- $d(O, \mathcal{P}) \geq \epsilon_2$ with a gap of $\epsilon_2 - \epsilon_1$.

It is immediate that tolerant testing strictly generalises standard testing, since standard testing is just tolerant testing with $\epsilon_1 = 0$. Are they really the same problem though, in that is tolerant testing no harder than standard testing? The answer is (unfortunately) no. It turns out that tolerant testing can be much harder than standard testing.

Recall that standard uniformity testing (requires $\Theta(\frac{\sqrt{n}}{\epsilon^2})$ samples) is no harder than identity testing, which is in turn no harder than closeness testing. The next theorem suggests that even tolerant uniformity testing is hard, which implies hardness for the tolerant version of the other two problems.

Theorem 15.3 (Tolerant uniformity testing is hard) *There exists a universal constant* $\epsilon_0 > 0$ *such that any tester with sample access to* $\mathbf{p}$ *distinguishing: 1)* $d_{\mathrm{TV}}(\mathbf{p}, \mathrm{Unif}[n]) \leq \epsilon_0$ *versus 2)* $d_{\mathrm{TV}}(\mathbf{p}, \mathrm{Unif}[n]) \geq \frac{1}{2} - \epsilon_0$ *with probability* $\geq \frac{2}{3}$ *requires at least* $\Omega(\frac{n}{\log n})$ *samples.*

It turns out that $\frac{n}{\log n}$ is also the upper bound of tolerant closeness testing, so it is the upper bound of tolerant identity testing and tolerant uniformity testing as well.

Theorem 15.4 (Tolerant closeness testing in $O(\frac{n}{\log n})$ samples) *There exists a tester* $\mathcal{A}$ *such that with sample access to* $\mathbf{p}$, $\mathbf{q}$ *over* $[n]$ *and on input* $\epsilon_1 < \epsilon_2 \in (0, 1]$, $\mathcal{A}$ *distinguishes* $d_{\mathrm{TV}}(\mathbf{p}, \mathbf{q}) \leq \epsilon_1$ *versus* $d_{\mathrm{TV}}(\mathbf{p}, \mathbf{q}) \geq \epsilon_2$ *with probability* $\geq \frac{2}{3}$ *using* $O(\frac{1}{(\epsilon_2 - \epsilon_1)^2} \frac{n}{\log n})$ *samples from both* $\mathbf{p}$, $\mathbf{q}$.

Note that the $\frac{1}{(\epsilon_2 - \epsilon_1)^2}$ dependence is not known to be optimal. It remains an open question what the optimal dependence is.

Now that we have established tolerant testing as a much harder problem than standard property testing, we want to understand why it is a much harder problem. One potential indication is that we can reduce another problem to tolerant testing, namely, to use a tolerant tester to solve a different problem.

It turns out that, up to polylogarithmic factors, tolerant testing is essentially equivalent to the *distance approximation* problem, that is, to approximate the distance of an object from a property. Given that in tolerant testing we can specify the gap we want to test on both sides, it is reasonable to consider using a tolerant tester to estimate the distance from a property The following two propositions establish the relationship between tolerant testing and distance approximation.

Proposition 15.5 (Tolerant testing $\Rightarrow$ Distance approximation) *Suppose* $\mathcal{A}$ *is a tolerant tester for property* $\mathcal{P}$, *with sample/query complexity* $Q(\epsilon_2 - \epsilon_1, n)$, *then there is an algorithm* $\mathcal{A}'$ *that, using* $(\log \frac{1}{\epsilon})(\log \log \frac{1}{\epsilon}) Q(2\epsilon, n)$ *samples/queries, outputs a distance estimate* $\hat{d} \in [d(O, \mathcal{P}) - \epsilon, d(O, \mathcal{P}) + \epsilon]$ *with probability* $\geq \frac{2}{3}$ *when* $\mathcal{A}'$ *is given sample/query access to* O.

The idea is relatively straightforward: just conduct a binary/ternary search for the desired distance. This is where the log factor comes from. The $\log\log$ factor comes from additional technicalities regarding failure probabilities (needing to take a union bound at some point).

On the other hand, we can also use a distance approximation algorithm for tolerant testing.

Proposition 15.6 (Distance approximation $\Rightarrow$ Tolerant testing) *Suppose $\mathcal{A}$ is an algorithm approximating the distance to $\mathcal{P}$ to within additive error $\pm\epsilon$ using sample/query complexity $Q(\epsilon, n)$. Then there is a tolerant tester $\mathcal{A}'$ for $\mathcal{P}$, with complexity $Q(\frac{\epsilon_2-\epsilon_1}{100}, n)$.*

This is even more straightforward than the previous direction, via a direct application of the given algorithm $\mathcal{A}$.

3 Robust Statistics

In standard statistics, we draw i.i.d. samples from some (unknown) distribution D. Typically, we make assumptions about D, such as it is over $[n]$, or assuming D is Gaussian. However, what if the above model is not quite true:

- Assumption on D only hold "approximately"?
  e.g. We assume D is Gaussian, but D is only close to Gaussian.
- Data is corrupted adversarially?
  e.g. The possibility of data poisoning attacks.

The area of robust statistics seeks to answer this question: How can we robustly perform statistical tasks despite (small) deviations from the assumptions? To address this problem, the first step is to come up with a suitable model. We do so with a general data contamination model:

Consider a class $\mathcal{D}$ of distributions, encoding assumptions on D (e.g. $\mathcal{D}$ = the set of Gaussian distributions).

1. Generate m samples from the unknown $D \in \mathcal{D}$.
2. There is an adversary that changes an ϵ-fraction of samples arbitrarily/adversarially.

The second step is a little bit vague, and there are many variations in formalising this model. For example:

- Can the adversary see samples before editing? (Adaptive versus Data oblivious)
- In terms of the edits made by the adversary, is it add only? Remove only? Can it do both?

To gain some intuition, let us consider a very simple example. Consider $\mathcal{D}$ = Set of 1D Gaussian Distributions. How can we robustly estimate its mean?

Answer: Take the median.

Fact 15.7 (Folklore) *Given m samples from $\mathcal{N}(\mu, \sigma^2)$ that are "ϵ-corrupted" (as defined above), with probability $\geq 99\%$, for $\hat{\mu}$ = sample median, we have*

$$|\hat{\mu} - \mu| \leq \sigma \cdot O\left(\epsilon + \sqrt{\frac{1}{m}}\right)$$

Note that the ϵ term comes from the ϵ-corruption and the $\sqrt{\frac{1}{m}}$ comes from the standard (vanilla) mean estimation (for example, derivable using Chebyshev's).

So what about learning both the mean and the variance?
Answer: Take the median, as well as some width (say the interquartile range) and scale it.

Fact 15.8 (Folklore) *Given m "ϵ-corrupted" samples from $\mathcal{N}(\mu, \sigma^2)$. Let $\hat{\mu}$ = sample median, $\hat{\sigma}$ = appropriately scaled interquartile range, then with probability $\geq 99\%$,*

$$d_{\mathrm{TV}}(\mathcal{N}(\mu, \sigma^2), \mathcal{N}(\hat{\mu}, \hat{\sigma}^2)) \leq O\left(\epsilon + \sqrt{\frac{1}{m}}\right)$$

In higher dimensions, appropriate generalisations of median are still pretty robust, but they turn out to be computationally expensive (e.g. the Tukey median is NP-hard to compute). However, in recent years, significant progress has been made in finding efficient algorithms for robust statistics in high-dimensions, including mean and covariance estimation.

4 Sampling Corrector

The setting is similar to that of robust statistics: we start with the assumption that D is close to having property $\mathcal{P}$. The new task is to "correct" D into $D' \in \mathcal{P}$ while keeping $d_{\mathrm{TV}}(D, D')$ small.

Note that this is a relatively new notion, and not too much is known about it. A more formal definition is as follows:

Definition 15.9 (Sampling Corrector) Given a property $\mathcal{P}$ and parameters $\epsilon_0 < \epsilon_1 \in (0, 1]$ and a natural number m, an m-sample (ϵ_0, ϵ_1)-sampling corrector $\mathcal{A}$ is such that, when given sample access to a distribution D with $d_{\mathrm{TV}}(D, \mathcal{P}) \leq \epsilon_0$, with probability $\geq 1 - \delta$ over the samples it draws from D and $\mathcal{A}$'s internal randomness, it outputs m i.i.d. samples from a distribution $\tilde{D}$ where

- $\tilde{D} \in \mathcal{P}$
- $d_{\mathrm{TV}}(D, \tilde{D}) \leq \epsilon_1$

The average sample complexity of $\mathcal{A}$ is $\frac{1}{m}$ times the number of samples it draws from D.

One trivial approach to this problem is to use $O(\frac{n}{\epsilon^2})$ samples to learn D to within total variation distance of $O(\epsilon)$ for $\epsilon = O(\epsilon_1 - \epsilon_0)$. Find the closest $\tilde{D} \in \mathcal{P}$ to the estimate $\hat{D}$ (the learned distribution) and then output samples from $\tilde{D}$ using internal randomness. Note that in spite of the one-time cost of using $O(\frac{n}{\epsilon^2})$ samples, this algorithm needs no further samples from D.

However, can we do better than this? We present one simple example below.

A simple example: The property we want is the independence of a pair of random variables over $[k] \times [n]$, i.e. $\mathcal{P} = \{(P_1, P_2) : P_1 \text{ over } [k] \text{ and } P_2 \text{ over } [n]\}$.

One easy way to get an independent distribution from an arbitrary one is to take "the product of its marginals", and the following structural property comes in handy.

Fact 15.10 *Suppose D (over $[k] \times [n]$) is ϵ-close to independent, then "the product of its marginals" $(\pi_1 D, \pi_2 D)$ is 3ϵ-close to D.*

Given this fact, the following proposition follows:

Proposition 15.11 *There is an m-sample $(\epsilon, 3\epsilon)$-sampling corrector for independence that succeeds with probability 1 and has average sample complexity 2.*

Proof. From Fact 15.10, it suffices to generate samples from $(\pi_1 D, \pi_2 D)$. To generate each sample, we take 2 samples $(x_1, y_1), (x_2, y_2)$ from D and combine them into (x_1, y_2). $\square$

It remains an open question whether we can do better than the average sample complexity of 2.