# Lecture 8: Instance Optimality

Daniel Kane
Scribe:Zihao Xu

April 19, 2017

## 1 Review

Previously, we've been talking about testing and learning algorithms for unstructured distributions. The three basic optimal results we've seen up to now are that learning takes $\Theta(n/\varepsilon^2)$ samples to get $L_1$ errors in $\varepsilon$, identity testing takes $\Theta(\sqrt{n}/\varepsilon^2)$ and closeness testing takes $\Theta(\sqrt{n}/\varepsilon^2 + n^{\frac{2}{3}}/\varepsilon^{\frac{4}{3}})$.
Based on these worst case bounds, a good question here to ask is when we can do better bounds on what instances.

## 2 Examples of Instance Optimality in Learning

The learning algorithm takes N samples and returns an empirical distribution. The expected is $\mathbb{E}[L_1 \text{ error}] \leq \sum_i \frac{\sqrt{p_i}}{\sqrt{N}} \leq \sqrt{\frac{n}{N}}$. This is bound is tight if $p$ is the uniform distribution or close to the uniform distribution. What we want to do here is to get away from this worst case bound and have a better bound for some circumstances that are not in the worst case.
The goal here is to get an error $\varepsilon$ such that $\sum_i \sqrt{\frac{p_i}{N}} \ll \varepsilon$. In other words, we want to have the number of samples required $N \gg \frac{(\sum_i \sqrt{p_i})^2}{\varepsilon^2} = \frac{|p|_{\frac{1}{2}}}{\varepsilon^2}$. But if we know "Bound",B, s.t. $B \geq |p|_{\frac{1}{2}}$, then we can learn the unstructured distribution only in $\mathcal{O}(B/\varepsilon^2)$ samples.
Remark: This is saying that in some cases, if you have some extra information about the distribution and if that extra information that is helpful to prove a better bound on the $\frac{1}{2}$-norm, then it's possible to learn by taking fewer samples.

## 3 Basic Ideas in Instance Optimality Testing

As what we've seen in the previous example, the question that we want to address in this lecture to know when can these three algorithms be improved? For instance, for non-advsersarial distribution, how much better can we expect to do and where is the

limit?
Let's consider a special case, the identity testing, $p = q$ v.s. $|p - q|_1 > \varepsilon$, where p is where we take samples from and q is some distribution that is explicitly given to the algorithm.
Here we are looking for an instance optimality.
**Goal** : Since $q$ is explicitly given, this enables us to manipulate it in order to extract more information for the algorithm. For each distribution $q$, we want the best possible algorithm for that $q$.

**Remark.** *The reason why we can expect to do better than the worst case bound for almost all q's is that the worst case q that we saw is when q is the uniform distribution. In order to get enough information about bins, we need to keep taking samples until the collision happens among those bins. If q is far from uniform, then it might be possible that the effective support of q is much smaller than the collection of all bins. If this happens, it should be much easier to do the tests, in that only a few relevant bins involved and the sample complexity is reduced to the number of relevant bins instead of the total number of bins. Therefore, we need to divide up these bins and use the weighted way to evaluate the values of all bins.*

## 3.1 Main Idea

**Idea** :Split bins into categories based on approximated size of $q_i$.
For instance, on the j-th category, $q_i \in [2^{-j}, 2^{1-j}]$. From this splitting scheme, we obtain only $O(\log(n/\varepsilon))$ categories. Let's consider one category at a time. Assume there are $m_j$ bins in j-th category. The idea here is to test whether $p = q$ or $|p - q|_1 > \varepsilon$ in the j-th category.
Let's define $p^{[j]}$ to be $p$ restrict to j-th category. As you can see, it's a pseudodistribution, in the sense that it doesn't contain all the mass.
Then the formal testing is $p^{[j]} = q^{[j]}$ v.s. $|p^{[j]} - q^{[j]}|_1 > \varepsilon$.

## 3.2 Algorithm

The procedures of the algorithm are as following:

1. Test $|p^{[j]}|_1 = |q^{[j]}|_1 + \mathcal{O}(\varepsilon)$
2. Normalize these pseudodistributions. Namely, $\tilde{p}^{[j]} = \frac{p^{[j]}}{|p^{[j]}|_1}$ and $\tilde{q}^{[j]} = \frac{q^{[j]}}{|q^{[j]}|_1}$
3. test $\tilde{p}^{[j]} = \tilde{q}^{[j]}$ v.s. $|\tilde{p}^{[j]} - \tilde{q}^{[j]}|_1 > \varepsilon/m_j 2^{-j}$

**Remark.** *If $|p^{[j]} - q^{[j]}|_1 > \varepsilon$, then we don't need to test it any furthur and the algorithm will simply return no. Otherwise, we should test for all j's. In the step 1., the algorithm takes $\mathcal{O}(1/\varepsilon^2)$ samples; and in the step 3., the algorithm takes samples from $\tilde{p}^{[j]}$ by sampling from p, rejecting samples in the wrong category(not in j-th category). In this*

*case, the number of samples from $\tilde{p}$ is $\mathcal{O}(\sqrt{m_j}m_j^2 4^{-j}/\varepsilon^2)$ and the number of samples from $p$ is $\mathcal{O}(m_j^{\frac{3}{2}}2^{-j}/\varepsilon^2)$.*

For the testing $p = q$ v.s. $|p-q|_1 > \varepsilon$, the algorithm breaks it down to the testing of the distribution of each category j's. That is, test either $p^{[j]} = q^{[j]}$ v.s. $|p^{[j]} - q^{[j]}|_1 > \varepsilon/\log(...)$ with low probability of failure $< 1/\log(...)$. The logarithmic factor comes from the fact that there are only logarithmically many relevant bins. Note that the tester will only return yes for $p = q$ if $p^{[j]} = q^{[j]}$ holds for all categories, j's and return no if $p^{[j]} = q^{[j]}$ fails for at least one j. The number of samples required for this algorithm is therefore $polylog(n/\varepsilon)\max_j(m_j^{\frac{3}{2}}2^{-j}/\varepsilon^2)$.
Now, let's take a look at the "max" factor.
$\max_j(m_j^{\frac{3}{2}}2^{-j}) = \max_j(m_j(2^{-j})^{\frac{2}{3}})^{\frac{3}{2}} \approx \max_j(\sum_{\text{bin in j-th category}} q_i^{\frac{2}{3}})^{\frac{3}{2}} \approx (\sum q_i^{\frac{2}{3}})^{\frac{3}{2}} = |q|_{\frac{2}{3}} \leq \sqrt{n}$.
Then we have an interesting bound $polylog(n/\varepsilon)\mathcal{O}(|q|_{\frac{2}{3}}/\varepsilon^2)$ for the identity testing.

# 4 A Better Way

In the previous discussion, we notice that the step to take biggest differences on individual categories is wasteful. A better idea is that if there is a moderate big difference on many categories, then we should combine these information together.
The better approach is to combine all errors in a weighted fashion and do the L2 tester on it. For this purpose, we need a clever way to do this by taking a special statistics. Take $Z = \sum_i \frac{|X_i - Y_i|^2 - X_i - Y_i}{q_i^{\frac{2}{3}}}$. Then, we need to compute $\mathbb{E}[Z]$, $Var(Z)$ and do the analysis that is similar to what we did in previous lectures. If the L1 norm is zero, then the variance isn't too big. If the L1 norm is large, then the expectation value is substantially larger than the square root of the variance. Then Take Z to compare with some threshold. If p = q with high probability, then Z is below the threshold. If p,q are $\varepsilon$ far from each other with high probability, then Z is above the threshold. Detailed discussions about the proof techniques are in the reference.[1]
This choice gets rid of the polylog factor and reaches the optimal bound $\mathcal{O}(|q|_{\frac{2}{3}}/\varepsilon^2)$. In addition to this, there are several improvements to compute the 2/3-norm.1. Ignore single heaviest bins. For instance, if some bin has 99% mass, then it's easy to see that the variance is small and close to $q_i(1-q_i)$.;2.Ignore any number of bins with total mass $< \varepsilon/16$. This is saying that we can throw out these light bins and take them as the error terms in the end of the analysis. Now, these two improvements cost $\mathcal{O}(1/\varepsilon)$.

# 5 Lower Bound

Assume no bin is super massive in q.
Let's use the adversary method. If $X = 0$,then $p_i = q_i$;if $X = 1$, then $p_i = q_i(1+\varepsilon_i)$.

We can see that $|p|_i = \mathcal{O}(1)$ and $|p/|p|_1 - q| \gg \varepsilon$ when $X = 1$, where $\varepsilon = \sum q_i \varepsilon_i$.
Let's look at the shared information between X and the count vectors $A_1, ..., A_n$. Assume these $A_i$'s are conditional independent, then we are able to upper-bound the shared information.
$I(X : A_1, ..., A_n) \leq \sum_i I(X : A_i) \leq \sum_i \mathcal{O}(m^2 q_i^2 \varepsilon_i^4) = \mathcal{O}(m^2 \sum q_i^2 \varepsilon_i^4)$.
Now, we would like to do some optimization in order to let the upper bound be as small as possible subject to the constraints above. By Lagrange multiplier, we obtain that $q_i \propto q_i^2 \varepsilon_i^3$. This is $\varepsilon_i = c q_i^{-\frac{1}{3}}$ and $c \sum q_i^{\frac{2}{3}} = \varepsilon$, which imply that $\varepsilon_i = \varepsilon / |q|_{\frac{2}{3}}^{\frac{2}{3}} q_i^{\frac{1}{3}}$.
Then we have the upper bound $I(X : A_1, ..., A_n) \leq \mathcal{O}(m^2) \sum q_i^{\frac{2}{3}} \varepsilon^4 / |q|_{\frac{2}{3}}^{\frac{8}{3}}$.
The upshot of this is if the shared information is substantially larger than 1, then we have $m \ll |q|_{\frac{2}{3}} / \varepsilon^2$.

# 6 Unknown Q

Let's think about the problem where q is unknown. In this case, we can't divide into categories as what we did previously, since we don't know what the categories should be. Now we need a slightly different strategy.
The idea is as follows:

1. Take $\tilde{\mathcal{O}}(m)$ samples.
2. Sort bins heavier than $\frac{1}{m}$ into categories.
3. Run the instance optimal tester on heavy bins.
4. Run the $L^2$ tests on light bins.

This will give us the bound $\tilde{\mathcal{O}}(m + |q|_{\frac{2}{3}} / \varepsilon^2 + |q_{<\frac{1}{m}}|_2 |q_{<\frac{1}{m}}|_0 / \varepsilon^2)$, where $q_\delta$ means the pseudodistribution of bins with mass less than $\delta$.

**Remark.** *1. Try* $m = 1, 2, 4, 8, ...$ *and get the run time with the* $\min_m(\tilde{\mathcal{O}}(m + |q|_{\frac{2}{3}} / \varepsilon^2 + |q_{<\frac{1}{m}}|_2 |q_{<\frac{1}{m}}|_0 / \varepsilon^2))$

*2. If* $m = n^{2/3} / \varepsilon^{4/3}$*, then the runtime is actually* $\tilde{\mathcal{O}}(n^{2/3)/\varepsilon^{4/3}}$*. This is the closeness tester.*

*3. When* $\varepsilon$ *is small or* $|q_{<\frac{1}{m}}|_2$ *is small, then the runtime is dominated by* $|q|_{2/3} / \varepsilon^2$*.*

Unfortunately, the algorithm is not instance optimal. And it's not clear what the meaning of instance optimal is. In particular, you can prove that there is no algorithm that for any q does as well as other closeness tester on that q. If there is an algorithm that guesses q, then it could split i-th bins into $\lceil nq \rceil$ parts and do $L2$ testers. This will give us $\mathcal{O}(\sqrt{n} / \varepsilon^2)$.

# References

[1] Gregory Valiant and Paul Valiant *An Automatic Inequality Prover and Instance Optimal Identity Testing*. Foundations Of Computer Science, (FOCS) 2016.