**The Algorithmic Foundations of Adaptive Data Analysis** September, 2017

# Lecture 6

*Lecturer: Aaron Roth* *Scribe: Aaron Roth*

## Description Length Bounds III

In this lecture, we will continue to develop transcript-compressible statistical estimators. We will focus on more general purpose estimators for statistical queries. First, we recall the AboveThreshold algorithm, which will continue to be a useful building block for us.

---

**AboveThreshold**$(T, q_1, q_2, \ldots)$:
  **AllDone** $\leftarrow$ **FALSE**
  **while** not **AllDone do**
    Accept the next query $q_i$
    Compute $a_i \leftarrow q_i(S)$
    **if** $a_i < T$ **then**
      Return $\perp$
    **else**
      Return $\top$
      **AllDone** $\leftarrow$ **TRUE**.
    **end if**
  **end while**

---

**Theorem 1** *For any threshold $T$,* **AboveThreshold**$(T)$ *is transcript compressible to $b(n,k)$ bits where $b(n,k) = \log(k+1)$.*

One thing we can use the primitive **AboveThreshold** for is to check whether a *guess* we have for the answer to a query $q_i$ is approximately correct or not. Given a guess $g_i$ for the answer to a query $q_i$, denote the compound query that asks "Is $|q_i(\mathcal{D}) - g_i| \leq \eta$?" as $(q_i, g_i, \eta)$. Given a fixed cutoff $\eta$ and a sequence of such queries $(q_1, g_1, \eta), \ldots, (q_k, g_k, \eta)$ we can initialize an instance of **AboveThreshold** with the threshold $\eta$, and start asking the sequence of queries $\hat{q}_i = |q_i(\mathcal{D}) - g_i|$. An answer of $\perp$ corresponds to "yes", and an answer of $\top$ corresponds to "no". We can keep asking these questions until we get our first "no". By itself, this does not give us the *answers* to all of our queries: but note, that for any query for which our guess was approximately correct ($\hat{q}_i(S) \leq \eta$), we know that the guess $g_i$ itself is sample-accurate to accuracy $\eta$. What about the final query $\hat{q}_i$, to which the answer was "yes"? We do not know the answer to this query – but we can obtain the answer by running the truncated estimator $\mathcal{O}_b^T(q_i)$. If we set $b = \log(1/\eta)$, this will also be sample accurate to accuracy $\eta$. Putting this together, we get the following sample estimator.

---

**OneWrongGuess**$(\eta, (q_1, g_1), (q_2, g_2), \ldots)$
  Start an instance of **AboveThreshold** with threshold $\eta$.
  **while AboveThreshold** has not halted **do**
    Accept the next query $(q_i, g_i)$.
    Feed **AboveThreshold** the query $\hat{q}_i(S) = |q_i(S) - g_i|$.
    **if AboveThreshold** returns $\perp$ **then**
      Return the answer $a_i = g_i$
    **end if**
  **end while**
  Return the answer $a_i = \mathcal{O}_b^T(q_i)$ for $b = \log(1/\eta)$.

---

**Theorem 2** *For any threshold* $0 < \eta \leq 1$, **OneWrongGuess** *is* $(\eta, 0)$*-sample accurate and transcript compressible to* $b(n,k)$ *bits where* $b(n,k) = \log(k+1) + \log(1/\eta)$.

**Proof** Let $f$ be a post-processing function which replaces query $(q_i, g_i)$ with query $\hat{q}_i(S) = |q_i(S) - g_i|$, and answer $a_i = \perp$ with answer $a_i = g_i$. Then **OneWrongGuess** can be viewed as a composition of $f$(**AboveThreshold**) and $\mathcal{O}_b^T$. We have proven that **AboveThreshold** is $\log(k+1)$-transcript compressible, and by the postprocessing theorem, so is $f$(**AboveThreshold**). Similarly, when $b = \log(1/\eta)$, $\mathcal{O}_b^T$ is transcript compressible to $\log(1/\eta)$ bits. By the composition theorem then, **OneWrongGuess** is transcript compressible to $b(n,k) = \log(k+1) + \log(1/\eta)$ many bits.

Sample accuracy follows trivially. We already know that $\mathcal{O}_b^T$ is $(1/2^b, 0)$-accurate, which is $(\eta, 0)$ accurate for our choice of $b$. Similarly, for every other answered query, we provide answer $g_i$, but by definition of **AboveThreshold**, we know that $|g_i - q_i(S)| \leq \eta$. ■

This procedure lets us answer many queries while being highly compressible so long as our guesses are always accurate, but it stops answering queries as soon as we have one incorrect guess. What if we want to continue until we have $m$ incorrect guesses? We can just compose **OneWrongGuess** with itself $m$ times. Consider the following procedure:

---

**GuessAndCheck**$(\eta, m, (q_1, g_1), (q_2, g_2), \ldots)$
  TimesWrong $\leftarrow 0$
  **while** TimesWrong $< m$ **do**
    Start an instance of **AboveThreshold** with threshold $\eta$.
    **while AboveThreshold** has not halted **do**
      Accept the next query $(q_i, g_i)$.
      Feed **AboveThreshold** the query $\hat{q}_i(S) = |q_i(S) - g_i|$.
      **if AboveThreshold** returns $\perp$ **then**
        Return the answer $a_i = g_i$
      **end if**
    **end while**
    Return the answer $a_i = \mathcal{O}_b^T(q_i)$ for $b = \log(1/\eta)$.
    TimesWrong $\leftarrow$ TimesWrong $+ 1$
  **end while**

---

**Theorem 3** *For any* $\eta, m$, **GuessAndCheck** *is* $(\eta, 0)$ *sample-accurate and transcript compressible to* $b(n,k)$ *bits where* $b(n,k) = m(\log(k+1) + \log(1/\eta))$.

**Proof** **GuessAndCheck** is just a composition of **OneWrongGuess** with itself, $m$ times. The result follows from our composition theorem. ■

**Theorem 4** *Fix a value of* $m$ *and a value of* $\delta > 0$. *Setting* $\eta = \sqrt{\frac{m}{n}}$, **GuessAndCheck**$(\eta, m)$ *is* $(\epsilon, \delta)$*-accurate for any sequence of compound queries* $(q_i, g_i)$ *until it halts, where* $q_i$ *can be any* $1/n$*-sensitive query, for:*

$$\epsilon = O\left(\sqrt{\frac{m\left(\log(k) + \log(n/m)\right) + \log(k/\delta)}{n}}\right)$$

**Proof** We again use the transfer theorem for transcript compressibility. We have shown that **GuessAndCheck** is $b(n,k) = m(\log(k+1) + \log(1/\eta))$ transcript compressible, and $(\eta, 0)$-sample accurate. Therefore, we know that it is $(\epsilon, \delta)$-accurate for:

$$\epsilon = \eta + \sqrt{\frac{(m(\log(k+1) + \log(1/\eta)) + 1)\ln(2) + \ln(k/\delta)}{2n}}$$

Plugging in our choice of $\eta$ yields the bound. As always, small improvements in asymptotics and constants can be obtained by choosing $\eta$ to optimize the above expression exactly. ■

So we can answer *lots* of queries accurately (with error scaling only logarithmically with $k$, as in the non-adaptive case) so long as we can correctly guess the answer up to our error tolerance for all but some number $m$ of our queries. But how can we in general come up with good guesses for query answers? We will see several ways.

**Lemma 5** *For any $\epsilon > 0$, any $k$ statistical queries $\phi_1, \ldots, \phi_k$ and for any dataset $S \in \mathcal{X}^n$, there is an $S' \in \mathcal{X}^{n'}$ such that:*

1. $\max_i \left|\mathbb{E}_S[\phi_i] - \mathbb{E}'_S[\phi_i]\right| \leq \epsilon$
2. $n' = \frac{\ln(4k)}{2\epsilon^2}$

**Proof** Consider generating $S'$ by subsampling $m$ points from $S$ with replacement. Under this sampling distribution, $\mathbb{E}[\phi_i] = \mathbb{E}_S[\phi_i]$ for each $i$. So we can apply a Chernoff bound to deduce that with probability $1/2$:

$$\max_i |\mathbb{E}_S[\phi_i] - \mathbb{E}_{S'}[\phi_i]| \leq \sqrt{\frac{\ln(4k)}{2m}} \leq \epsilon$$

by our choice of $m$. Since this occurs with probability $1/2$ over our selection of $S'$, in particular, there must exist some $S'$ of size $m$ satisfying this bound. ■

With this fact in hand, we can give a simple but computationally inefficient statistical estimator that can answer arbitrary sequences of statistical queries with error bounds scaling only logarithmically with $k$:

---

**MedianOracle**$(q_1, \ldots, q_k)$

Initialize an instance of **GuessAndCheck**$(\eta, m)$ with $m = \sqrt{\frac{n \log |\mathcal{X}| \ln(4k)}{2}}$ and $\eta = \sqrt{\frac{m}{n}}$.
Initialize a version space $\mathcal{S}_0 = \mathcal{X}^{n'}$ where $n' = \frac{\ln(4k)}{2\eta^2}$
**for** $i = 1$ to $k$ **do**
  Given query $q_i$, construct a guess $g_i = \text{median}\left(\{q_i(S') : S' \in \mathcal{S}_{i-1}\}\right)$
  Feed the query $(q_i, g_i)$ to **GuessAndCheck** and receive answer $a_i$.
  **if** $\hat{a}_i = g_i$ **then**
    $\mathcal{S}_i \leftarrow \mathcal{S}_{i-1}$
  **else**
    $\mathcal{S}_i \leftarrow \mathcal{S}_{i-1} \setminus \{S' \in \mathcal{S}_{i-1} : |q_i(S') - a_i| > \eta\}$
  **end if**
  Return answer $a_i$.
**end for**

---

**Theorem 6** *For any $\delta > 0$,* MedianOracle *is $(\epsilon, \delta)$-accurate for any sequence of $k$ statistical queries where:*

$$\epsilon = O\left(\frac{\log(|\mathcal{X}| \log(k))^{1/4} \sqrt{\log k + \log n}}{n^{1/4}}\right)$$

**Proof** The median oracle is simply an instantiation of (a postprocessing of) GuessAndCheck. Thus, with our choice of $\eta$, the accuracy bound *for the queries asked before the algorithm halts* follows from the accuracy guarantee of GuessAndCheck, together with our choice of $m$. It remains to show that MedianOracle will answer all $k$ queries asked, and never halt. By the definition of GuessAndCheck, this is equivalent to showing that $|q_i(S) - g_i| \leq \eta$ for all but $m$ rounds $i$. Call such rounds "Mistaken Guesses". Note that we have chosen $m$ and $\eta$ such that: $m = \frac{\ln(4k) \log |\mathcal{X}|}{2\eta^2} = n' \log |\mathcal{X}|$, and recall that GuessAndCheck is $\eta$-sample accurate.

We prove this by tracking $|\mathcal{S}_i|$. Note that by construction, $|\mathcal{S}_0| = |\mathcal{X}|^{n'}$. Next note that at every round $i$ such that a mistaken guess is made, $|\mathcal{S}_i| \leq |\mathcal{S}_{i-1}|/2$. This is because on those rounds (by definition of GuessAndCheck, $|g_i - q_i(S)| > \eta$, and all of the sets $S'$ such that $|q_i(S') - a_i| > \eta$ are

removed from $\mathcal{S}_i$. But by definition, $g_i = \text{median}(\{q_i(S') : S' \in \mathcal{S}_{i-1}\})$, and so at least half of the sets $S'$ in $\mathcal{S}_i$ are removed by this update. Finally, by Lemma 5, we know that for every set of $k$ statistical queries $\phi_1, \ldots, \phi_k$, there is *some* $S' \in \mathcal{S}_0$ such that $|q_i(S') - q_i(S)| \leq \eta$. This $S'$ is never removed, so we know that $S' \in \mathcal{S}_i$ for every $i$, and hence $|\mathcal{S}_i| \geq 1$ for every $i$. Thus, the number of mistaken guesses can be at most $\log |\mathcal{S}_0| = n' \log |\mathcal{X}| = m$, which completes the proof. ■

What are we to make of these bounds? On the one hand, we obtain the polylogarithmic dependence on $k$ in our error bounds that is close to the best achievable even in the non-adaptive setting for answering arbitrary statistical queries! This is an exponential improvement on what we can achieve with the simpler general-purpose statistical estimators that we have seen before. On the other hand, the bound has a number of drawbacks. Fist, the dependence on $n$ is suboptimal — we get error tending to zero at a rate of $1/n^{1/4}$ instead of $1/\sqrt{n}$. We will see how to improve this dependence later in the course. Next, the error scales with $\log |\mathcal{X}|$, which should be taken as a measure of the dimension of the data domain. This was something that was not necessary in the non-adaptive case. Finally, the estimator is computationally intractable — it needs to maintain an enormous version space of sets $S'$. We will see that these properties are unavoidable for any statistical estimator in the adaptive setting that obtains error rates scaling only logarithmically in $k$ (compared to the non-adaptive setting, where this efficient scaling with $k$ is achievable via a trivially tractable mechanism: simply computing the empirical average.)

We will end our discussion of description length bounds with a computationally efficient heuristic, leveraging the **GuessAndCheck** sub-routine. The heuristic can be viewed as an extension of standard test/train methodology, in which the data set is divided into a *training* and *holdout* set. The training set can be used in arbitrary ways — in this case, it is used to produce the *guesses* for the query answers. The holdout set in this case is only accessed via transcript compressible mechanisms,

---

**ReusableHoldout**$(m, q_1, \ldots, q_k)$

Randomly split the dataset $S$ into two equal parts: a training set $S_T$ and a holdout set $S_H$, each of size $n/2$.

Initialize an instance of **GuessAndCheck**$(\eta, m)$ on $S_H$ with $\eta = \sqrt{\frac{2m}{n}}$.

**for** $i = 1$ to $k$ **do**

  Given query $q_i$, construct a guess $g_i = q_i(S_T)$

  Feed the query $(q_i, g_i)$ to **GuessAndCheck** and receive answer $a_i$.

  Return answer $a_i$.

**end for**

---

Since the re-usable holdout directly calls **GuessAndCheck** on half of the dataset, it directly inherits its accuracy bounds:

**Theorem 7** *Fix a value of $m$ and a value of $\delta > 0$.* **ReusableHoldout**$(m)$ *is $(\epsilon, \delta)$-accurate for any sequence of $1/n$ sensitive queries $q_i$ until it halts, for:*

$$\epsilon = O\left(\sqrt{\frac{m\left(\log(k) + \log(n/m)\right) + \log(k/\delta)}{n}}\right)$$

Unlike the median mechanism, we do not have a guarantee about when the mechanism will halt. Informally, it will halt after the analyst asks $m$ queries that overfit the training set by more than $\sqrt{\frac{2m}{n}}$. But note a couple of things:

1. If the analyst is actually non-adaptive, then (if $m \geq \log k$) a Chernoff bound will tell us that the mechanism will be able to answer exponentially many queries before it halts, since overfitting by more than $\eta$ is extremely unlikely on any given query.

2. The reusable holdout with $m \geq \log d$ also defeats the linear classification "attack" we saw on the empirical average mechanism: that mechanism only asks a single "overfitting" query – the last one.

3. More generally, the reusable holdout will work well whenever the analyst does not overfit very frequently. So the "less adversarial" the analyst, the better the guarantees.

**Bibliographic Information** The "AboveThreshold" algorithm is adapted from a similar algorithm from the differential privacy literature [DR14] that we will see again in the next section of our course. Similarly, the median mechanism [RR10] was initially designed as a differentially private algorithm. The description-length-bounded version presented here appeared in the appendix of [DFH$^+$15a]. The reusable holdout was also first presented as a differentially private algorithm, in [DFH$^+$15b]. Later in the class we will re-visit these mechanisms, and derive improved bounds using differential privacy.

# References


[DFH$^+$15a] Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Roth. Generalization in adaptive data analysis and holdout reuse. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2350–2358, 2015.

[DFH$^+$15b] Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Roth. The reusable holdout: Preserving validity in adaptive data analysis. *Science*, 349(6248):636–638, 2015.

[DR14] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[RR10] Aaron Roth and Tim Roughgarden. Interactive privacy via the median mechanism. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 765–774. ACM, 2010.