**CS 395T: Sublinear Algorithms, Fall 2020** October 22nd, 2020

# Lecture 17: Adaptive k-sparse compressed sensing

*Prof. Eric Price* *Scribe: Nikos Mouzakis*

**NOTE:** THESE NOTES HAVE NOT BEEN EDITED OR CHECKED FOR CORRECTNESS

## 1 Overview

In the last lecture we showed lower bounds for non-adaptive compressed sensing and we proved that we need $klog(\frac{n}{k})$ linear measurements.

In this lecture we are going to discuss how to beat that lower bound with adaptive measurements.

## 2 Setting

Assume that $x$ is k-sparse and that $w$ denotes the (gaussian) noise. In adaptive sensing, we pick the measurement vector $v^{(i)}$ and we learn $< v^{(i)}, x + w >$. Then, after learning $< v^{(i)}, x + w >$ we can pick $v^{(i+1)}$ and repeat the process for every $i \in [m]$. The goal is to eventually output an $\hat{x}$ such that $||\hat{x} - x||_2 \leq O(1)||w||_2$

## 3 k=1

### 3.1 Lower bound

Lets consider the case $k = 1$.

In the non-adaptive case we showed that the problem is hard even if $x_i = e_i, w \sim N(0, \frac{1}{100n} I_n)$. Specifically we showed about the mutual information that $I(< v, x + w >; < v, x >) \leq \frac{1}{2} log(1 + \frac{E[<v,x>^2]}{E[<v,w>^2]}) = \frac{1}{2} log(1 + \frac{E[||v_i||^2]}{E[<v,w>^2]}) = \frac{1}{2} log(1 + \frac{||v||^2/n}{||v||^2/100n}) = O(1)$, which means that the information we can learn is only a constant. However, the mutual information required for any valid recovery is $I(\hat{x}, x) = \Omega(logn)$, so we needed $m = \Omega(logn)$ measurements.

**Question:** Why would it be possible to overcome this bound? Why does this bound not apply for the adaptive case?

The problem lies with the equation $E[||v_i||||^2] = ||v||^2/n$. Recall that to prove this we used that $v$ and $x$ are uncorrelated, i.e. $E[||v_i||||^2] = \sum_{j=1}^{n} E[||v_j||^2 * \mathbb{I}_{x=e_j}] = ||v||^2/n$. But the whole point of having adaptivity is that with each step we learn more about $x$ and can make better queries.

For example, for the first query we might have that $E[||v_i^{(1)}||^2] = ||v^{(1)}||^2/n$, but at the end we can even have that $E[||v_i^{(m)}||^2] = ||v^{(m)}||^2$. So the mutual information bound would become $I(<$

$v, x + w >; < v, x >) \leq \frac{1}{2} log(1 + n)$. This naive bound essentially says that once we know the location of the answer, we can learn it in constant measurements.

Let's get a better bound. The information we gain in round $r$ is $I(< v, x + w >; < v, x >) \leq \frac{1}{2} log(1 + \frac{E[||v_i^{(r)}||^2]}{||v||^2/100n})$.

Suppose at round $r$ we know $b$ bits of information about $i$. Let's assume that these are the first $b$ bits of the answer's index $i$, i.e. $i$ is in an interval of width $\frac{n}{2^b}$. Then we could set $v$ to be 1 over that interval, which means $E[||v_i^{(r)}||^2] = 1$ and $||v||^2 = n/2^b$. This would make the mutual information for the $r$-th round $\frac{1}{2} log(1 + 100 * 2^b) \approx b/2$. So if we know $b$ bits of information about $i$, in two measurements we can learn $b$ more bits. Effectively, we could hope to learn 1 bit for the first two measurements, 2 bits for the next two measurements, 4 for the next two, etc. We would then need $loglogn$ measurements to learn the full answer. This analysis can be made tight (even without the assumption of $i$ being concentrated in an interval, and that our knowledge of $i$ being its first few bits) to show a lower bound of $O(loglogn)$.

### 3.2 Algorithm

We need to know $\Omega(log(SNR))$ bits per measurement (since the lower bound says that's optimal).

Consider first the case with SNR $=\infty$. (i.e. $x = ae_i$, $w = 0$). In this noiseless case, we can set $v^{(1)} = (1, 1, 1.., 1)$ and $v^{(2)} = (1, 2, 3.., n)$ with $i = \frac{<v^{(2)}, x>}{<v^{(1)}, x>} = \frac{a*i}{a}$.

Next, lets assume SNR is bounded, i.e. $||w||_1 \leq \frac{a}{R}$, the same $v$'s give $\frac{<v^{(2)}, x+w>}{<v^{(1)}, x+w>} = \frac{ai \pm an/R}{a \pm a/R} \approx i \pm O(n/R)$. So we can learn $i$ exactly if $R \geq O(n)$.

However, even if R is less than that, we get that after a round with SNR=R, we can restrict $i$ to $O(n/R)$ possibilities. This gives us the following algorithm idea.

---

$R \leftarrow O(1)$
$k \leftarrow n/2, \Delta \leftarrow n/2$ $\quad i \in [k \pm \Delta]$
**repeat**
  $S \leftarrow \{k + j || j| \leq \Delta\}$
  $y_1 \leftarrow \sum_{j \in S} x_j$
  $y_2 \leftarrow \sum_{j \in S} j * x_j$
  $k \leftarrow \frac{y_2}{y_1}$
  $\Delta \leftarrow \Delta / R$
**until** $\Delta \leq 1$

---

However, we can notice that our SNR increases with every step. This is because the region that $i$ lies in is becoming smaller. Recall that $R \geq \frac{a}{||w||_1}$, but $R' \geq \frac{a}{||w_s||_1}$, and $||w_s||_1 \approx \frac{|S|}{n} ||w||_1 = R||w||$. Hence $R' \geq \frac{a}{||w_s||_1} \approx R^2$, if we assume that the noise $w$ is spread out evenly.

To make this work even if $w$ isn't sufficiently spread out we can virtually permute $x$ (probe the permuted indices of $x$), so the expected value of $w$ is the same.

However this has some probability of failure $\delta_i = \delta / i^2$ for each round. We can counteract this by setting $R' = R^2 \delta_i$ without slowing the algorithm down significantly (it still scales doubly exponen-

tially).

Overall, this solves the $k = 1$ case.

## 4 k=2

An idea to extend this for higher $k$ would be to use multiple equations (instead of just $(1, 1, 1..., 1)$ and $(1, 2, 3, 4..., n)$).

A simpler idea is to sample at rate $\frac{1}{k}$ and run the algorithm on this sample, which gives a $k * \frac{1}{k}(1 - \frac{1}{k})^n \approx \frac{1}{e}$ chance of sampling exactly one of the k heavy hitters. Then the noise is $E[||w_s||_1] = \frac{||w||_1}{k}$. This means that with O(log log n) measurements we have $\frac{1}{e}$ chance of finding 1 heavy hitter.

Extending this to find all heavy hitters, we can sample $x$ as above into $k$ samples and run the algorithm in each, so the probability to not find some heavy hitter is for it to never have been sampled alone. So Pr[any given HH not found] $\leq (1 - \frac{1}{k}(1 - \frac{1}{k})^{k-1})^{O(k)} \approx 2^{-O(1)}$, which is a small constant. So every heavy hitter will be found with constant probability, and we will get a constant fraction of the heavy hitters (but still not all).

One way is to take $O(k)log(k)$ samples which gives $2^{-O(1)logk}$ and a $O(klog(log(n))log(k))$ bound, which is good only for small k.

Another option is to repeat the process. Since with $O(kloglogn)$ measurements we find 90% of the heavy hitters we can repeat the process for $k' = k/10$. This gives a $O(kloglogn) + O(\frac{k}{10}loglogn) + O(\frac{k}{100}loglogn) = O(kloglogn)$ lower bound.