CS 224: Advanced Algorithms Spring 2017

Lecture 17 — March 21, 2017

Prof. Jelani Nelson *Scribe: Brian Hentschel*

1 Overview

In this lecture we finish streaming algorithms (more can be seen in CS 226, Algorithms for Big Data next semester) and go over the "power of +-1 random variables". We also cover least squares regression.

2 Turnstile Streaming

Goal is to answer queries about $x \in \mathbb{R}^n$ approximately. x starts as the 0 vector. In turnstile streaming, we are going to pick some (possibly random) vector $\Pi \in \mathbb{R}^{m \times n}$ with $m << n$ and maintain $y = \Pi x$. In addition to having Πx approximate x, we want Π to have a very small memory footprint.

2.1 ℓ_2-estimation in Turnstile Streaming:

Let $\Pi_{ij} = \frac{\sigma_{ij}}{\sqrt{m}}$ where $\sigma_{ij} : [m] \times [n] \to \{-1, +1\}$ is a hash function chosen from a 4-wise independent family.

Claim: (due to Alon, Matias, and Szegedy in [1]) For $m = \Omega(\frac{1}{\epsilon^2}, \mathbb{P}(\|y\|_2^2 - \|x\|_2^2 > \epsilon\|x\|_2^2) < \frac{1}{3}$

Note: If a lower probability is needed, it is enough to take the median of many runs.

Proof: We will prove $\mathbb{E}\|y\|_2^2 = \mathbb{E}\|x\|_2^2$ and $\text{Var}[\|y\|_2^2] \leq \frac{c}{m}\|x\|_2^4$. If both are true, then using Chebyshev's inequality we have

$$P(\|y\|_2^2 - E\|y\|_2^2) > \epsilon\,\mathbb{E}\|y\|_2^2 < \frac{1}{\epsilon^2(\mathbb{E}\|y\|_2^2)^2} \cdot \text{Var}(\|y\|_2^2) < \frac{1}{3}$$

for m as chosen.

Claim 1: $\mathbb{E}\|y\|_2^2 = \mathbb{E}\|x\|_2^2$

$$\mathbb{E}\|y\|_2^2 = E\sum_i y_i^2 = \sum_i \mathbb{E}\, y_i^2 = m\,\mathbb{E}\, y_1^2$$

$$\begin{aligned}\mathbb{E}\, y_1^2 &= \frac{1}{m} E < \sigma, x >^2 \\ &= \frac{1}{m}\, \mathbb{E}(\sum_j (\sigma_{j^2} x_j^2 + \sum_{j' \neq j} \sigma_j \sigma_{j'} x_j x_{j'})) \\ &= \frac{1}{m}(\sum_j x_j^2 + \sum_{j' \neq j} \mathbb{E}(\sigma_j \sigma_{j'}) x_j x_{j'}) \\ &= \frac{1}{m}(\sum_j x_j^2) = \frac{1}{m}\|x\|_2^2\end{aligned}$$

where the third line follows from the linearity of expectation and the last line follows from 4-wise independence. Altogether, it follows that $\mathbb{E}\|y\|_2^2 = \|x\|_2^2$.

Claim 2: $\mathrm{Var}[\|y\|_2^2] \leq \frac{c}{m}\|x\|_2^4$
$\mathrm{Var}[\|y\|_2^2] = \mathbb{E}\|y\|_2^4 - (\mathbb{E}\|y\|_2^2)^2$. From claim 1, the second term is $\|x\|_2^4$. The first term expands as

$$\mathbb{E}\|y\|_2^4 = \mathbb{E}(\sum_i y_i^2)^2 = \mathbb{E}(\sum_i y_i^4 + \sum_{i \neq i'} y_i^2 y_{i'}^2)$$

We calculate the first only in the notes. The second calculation is similar.

$$\mathbb{E}\sum_i y_i^4 = m \cdot \mathbb{E}\, y_1^4 = \frac{1}{m^2}\, \mathbb{E} < \sigma, x >= \frac{1}{m^2}\, \mathbb{E}(\sum_j \sigma_j x_j)^4$$

This is a sum of monomials of the type: $\sigma_{j1}\sigma_{j2}\sigma_{j3}\sigma_{j4} x_{j1} x_{j2} x_{j3} x_{j4}$, where the j_i are not necessarily distinct. If there exists a σ_{ji} to an odd power, then by 4-wise indendence of the σ the expectation of that term is 0. Thus we are left with a monomial in our expectation consisting only of $\sigma_{j1}^2 \sigma_{j2}^2 x_{j1}^2 x_{j2}^2$ and $\sigma_{j1}^4 x_{j1}^4$ terms. The expectation then becomes

$$\mathbb{E}[\sum_j x_j^4 + \sum_{j \neq j'} 3 x_j^2 x_{j'}^2] \leq 3 \cdot \|x\|_2^4$$

Then $\mathrm{Var}[\|y\|_2^2] = \frac{2}{m}\|x\|_2^4$.

Algorithm: When updating, for $x_i \leftarrow x_i + \Delta$, we update $y_i \leftarrow y_i + \Delta \cdot \Pi^i$, where Π^i is the ith column of Π. In this case, this requires getting each entry of Π^i by computing the 4-wise hash. This is a constant time operation and we do it for m values in Π so the total update time is $\Theta(m) = \Theta(\frac{1}{\epsilon^2})$. In 2004, Thorup and Zhong showed that letting Π have a single $+-1$ value in each column gives the same guarantees [6], where the row in each column is chosen from a two-wise indpendent $h : [n] \to [m]$ and $\sigma : [n] \to \{-1, +1\}$ is chosen still from a 4-wise independent family.

2.2 Heavy-Hitters

This section covers the approximation of heavy hitters using the same matrix as was used by Thorup and Zhang. The matrix was in fact introduced in this paper by Charikar, Chen and Farach-Colton [7], and is known as the CountSketch.

Goal: Find all i such that $|x_i|$ is "large". Formally: A word i is an ϵ-ℓ_2 heavy hitter if $|x_i| > \epsilon\|x\|_2$. There are at most $\frac{1}{\epsilon^2}$ heavy hitters. The following algorithm reports a list L of size $O(\frac{1}{\epsilon^2})$ containing all ϵ heavy hitters.

Count Sketch: Maintain $y = \Pi x$ for the same Π as seen in the previous section, but stacks $R = \Theta(\lg n)$ of them on top of each other. You can imagine we maintain a grid of counters (the entries of y), which we will call $C_{a,b}$ for $a \in [R]$ and $b \in [k]$ where $R = \Theta(\log n)$ and $k = \Theta(1/\epsilon^2)$. The counters are initialized to zero. The algorithm has R hash functions h_i from $[n]$ to $[k]$ and another R sigma functions from $[n]$ to $\{-1, +1\}$. Both the h_r and σ_r functions are chosen from 2-wise independent families.

To process an update "$x_i \leftarrow x_i + \Delta$": for each $r \in [R]$, perform the update $C_{r,h_r(i)} \leftarrow C_{r,h_r(i)} + \Delta \cdot \sigma_r(i)$.

Now notice that, any any index $i \in [n]$ and any $r \in [R]$, we have that $\sigma_r(i) \cdot C_{r,h_r(i)}$ is equal to x_i plus some "noise". The noise comes from other coordinates that collide with i under h_r (and their mass is added to the same counter, after taking a dot product with random signs). To get a good estimate of x_i, we then output the median over all r of $\sigma_r(i) \cdot C_{r,h_r(i)}$. Why does this work? For any $r \in [R]$, let us look at the expected error of the estimate. We will use Cauchy-Schwarz, which states that for any random variable Z, $\mathbb{E}\,|Z| \le (\mathbb{E}\,Z^2)^{1/2}$.

Fix $r \in [R]$. Let $I_{\mathcal{E}}$ denote the indicator random variable for event $\mathcal{E}$.

$$\begin{aligned}
\mathbb{E}\,|x_i - \sigma_r(i)C_{r,h_r(i)}| &= \mathbb{E}\,|\sum_{\substack{j\neq i\\ h(i)=h(j)}} \sigma_r(i)\sigma_r(j)x_j| \\
&= \mathbb{E}\,|\sum_{\substack{j\neq i\\ h_r(i)=h_r(j)}} \sigma_r(j)x_j| \\
&\le (\mathbb{E}(\sum_{\substack{j\neq i\\ h_r(i)=h_r(j)}} \sigma_r(j)x_j)^2)^{1/2} \\
&= (\sum_{j\neq j'}(\mathbb{E}\,I_{h_r(j)=h_r(i)})(\mathbb{E}\,\sigma_r(j)\sigma_r(j'))x_jx_{j'} + \sum_{j\neq i}(\mathbb{E}\,I_{h_r(j)=h_r(i)})x_j^2)^{1/2} \\
&= (\sum_{j\neq j'}(\mathbb{E}\,I_{h_r(j)=h_r(i)})\cdot 0\cdot x_jx_{j'} + \sum_{j\neq i}(\mathbb{E}\,I_{h_r(j)=h_r(i)})x_j^2)^{1/2} \text{ (using 2-wise independence)} \\
&\le \frac{1}{\sqrt{k}}\|x\|_2 \\
&< (\epsilon/12)\|x\|_2 \text{ (picking } k = 144/\epsilon^2)
\end{aligned}$$

We have thus shown that the expected error from a single $C_{r,h_r(i)}$ in estimating x_i is at most $(\epsilon/12)\|x\|_2$. Thus by Markov, the probability of error more than $(\epsilon/4)\|x\|_2$ is at most $1/3$. Thus we expect at most $R/3$ of the $C_{r,h_r(i)}$ counters across $r \in [R]$ to give error more than $(\epsilon/4)\|x\|_2$ error. Thus by Chernoff, the probability that at least $R/2$ of these counters give more than this error is $\exp(-\Omega(R)) \le 1/n^{c+1}$ for $R = C\log n$, where c can be made an arbitrarily large constant by increasing C. Thus by a union bound over all $i \in [n]$, with probability $1 - 1/n^c$ we can obtain estimates $\tilde{x}_i$ such that $|\tilde{x}_i - x_i| < (\epsilon/4)\|x\|_2$ for all $i \in [n]$ simultaneously. Now note that every heavy hitter i will be be estimated with $|\tilde{x}_i| > (\epsilon - \epsilon/4)\|x\|_2 = (3\epsilon/4)\|x\|_2$. Meanwhile, every index

i which is not even $(\epsilon/2)$-heavy will be estimated with $|\tilde{x}_i| < (\epsilon/2+\epsilon/4) = (3\epsilon/4)\|x\|_2$. The number of $(\epsilon/2)$-heavy hitters is at most $4/\epsilon^2$. Thus if we let L be the set of the indices with the top $4/\epsilon^2$ $|\tilde{x}_i|$ values, L will contain the actual ϵ-heavy hitters with probability at least $1 - 1/n^c$.

Querying the data structure is a for loop over all $i \in [n]$, taking $O(n \log n)$ time. The space is $O(kR)$, which is $O(\epsilon^{-2} \log n)$. The update time is $O(\log n)$, and the failure probability is $1/n^c$. It is possible to come up with a data structure with the same complexities and failure probability, but where the query time is exponentially better, taking only $O(\epsilon^{-2}\, poly(\log n))$; see [3].

3 Least Squares Regression

The previous matrix can also be used to estimate the parameters β_j in least squares regression.

Problem Description: We are given a matrix X in $\mathbb{R}^{n\times d}, n >> d$ and $y \in \mathbb{R}^n$. For each $y_i \in y$, we have $y = \sum_{i=1}^{d} \beta_j x_{ij} + \epsilon$, where ϵ is distributed normally with variance σ^2 and mean 0. Our goal is to estimate the parameters B_j. This will be done by solving

$$B^{LS} = \arg\min_{\beta\in\mathbb{R}^d} \|X\beta - y\|_2^2 = (X^\top X)^+ X^\top y$$

Computing $X^\top X$ takes $O(nd^2)$ if we do simple matrix multiplication and calculating the pseudo-inverse is $O(d^3)$. Since $n > d$, our goal will be to lower the cost of the matrix multiplication $X^\top X$ via approximation. The following technique was introduced by Sarlós in the paper *Improved Approximation Algorithms for Large Matrices via Random Projections* [8].

Definition: For a linear subpsace $V \subset \mathbb{R}^n, \Pi$ is an ϵ subspace embedding if

$$\forall x \in V \quad (1-\epsilon)\|x\|_2^2 \le \|\Pi x\|_2^2 \le (1+\epsilon)\|x\|_2^2$$

Let V have dimension r. Then V can be written as $V = \{Uz : z \in \mathbb{R}^r\}$ where $UU^\top = I, U \in \mathbb{R}^{n\times r}$. The definition of Π could also be written as:

- $\forall z, \quad (1-\epsilon)\|z\|_2^2 \le \|\Pi U z\|_2^2 \le (1+\epsilon)\|z\|_2^2$
- $\forall z, \quad \|z^\top[(\Pi U)^\top \Pi U - I]z\|_2 < \epsilon\|z\|_2^2$

The last bullet point is equivalent to claiming that $(\Pi U)^\top \Pi U - I$ has no eigenvalue larger than ϵ.

Claim 1: If Π is an ϵ-subspace embedding for span {y, cols(x)} then for $\tilde{\beta}^{LS} = \arg\min\|\Pi x\beta - \Pi y\|_2^2$, we have

$$\|X\tilde{\beta}^{LS} - y\| \le \frac{1+\epsilon}{1-\epsilon}\|X\beta^{LS} - y\|_2^2$$

Proof: First we note that $\Pi x\beta^{LS} - \Pi y = \Pi(x\beta^{LS} - y)$, and $(x\beta^{LS} - y)$ is in the subspace of

$\{y, cols(x)\}$. Then we have

$$\begin{aligned}(1-\epsilon)\|X\tilde{\beta}^{LS} - y\|_2^2 &\leq \|\Pi(X\tilde{\beta}^{LS} - y)\|_2^2 \\ &\leq \|\Pi(X\beta^{LS} - y)\|_2^2 \\ &\leq (1+\epsilon)\|X\beta^{LS} - y\|_2^2 \\ \|X\tilde{\beta}^{LS} - y\|_2^2 &\leq \frac{1+\epsilon}{1-\epsilon}\|X\beta^{LS} - y\|_2^2\end{aligned}$$

We can now replace β_{LS} with $\tilde{\beta}^{LS}$, with $\tilde{\beta}^{LS}$ an $m \times d$ matrix. Performing the matrix multiplication $(\Pi x)^\top(\Pi x)$ then takes $O(md^2)$ time. We also need to compute Πx, which since Π is extremely sparse ends up proportional to the number of nonzero entries in x. This is $O(nd)$.

Claim 2: If $m = \Theta(\frac{d^2}{\epsilon^2})$ and U is an orthonormal basis of dimension $r \times d$, then

$$P(\|(\Pi U)^\top(\Pi U) - I\| > \epsilon) < \frac{1}{\epsilon}$$

Proof: By Markov's inequality, $P(\|(\Pi U)^\top(\Pi U) - I\|_2 > \epsilon) < \frac{1}{\epsilon^2}\,\mathbb{E}\|M\|_2^2$ where $M = (\Pi U)^\top(\Pi U) - I$. Now, M is a $d \times d$ real symmetric matrix and thus has d real eigenvalues $|\lambda_1| \geq |\lambda_2| \geq \ldots \geq |\lambda_d|$. We have $\|M\|_2^2 = \lambda_1^2$. Meanwhile $trace(M^2) = \sum_{i=1}^d \lambda_i^2$. Thus $\mathbb{P}(\|\|_2 > \varepsilon) < \frac{1}{\epsilon^2}\,\mathbb{E}\,trace(M^2)$. A simple calcuation shows that $\mathbb{E}\,trace(M^2) = O(d^2/m)$ (see for example [4, 5]; also see an earlier, but suboptimal analysis in [2]). Thus we can set $m = \Theta(d^2/(\varepsilon^2\delta))$ to achieve failure probability δ.

References

[1] Noga Alon, Yossi Matias, Mario Szegedy. The Space Complexity of Approximating the Frequency Moments. *J. Comput. Syst. Sci.*, 58(1):137–147, 1999.

[2] Kenneth L. Clarkson, David P. Woodruff. Low rank approximation and regression in input sparsity time. *STOC*, 81–19, 2013.

[3] Kasper Green Larsen, Jelani Nelson, Huy L. Nguyen, Mikkel Thorup. Heavy Hitters via Cluster-Preserving Clustering. *FOCS*, 61–70, 2016.

[4] , Xiangrui Meng, Michael W. Mahoney. Low-distortion subspace embeddings in input-sparsity time and applications to robust linear regression. *STOC*, 91–100, 2013.

[5] Jelani Nelson, Huy L. Nguyen. OSNAP: Faster Numerical Linear Algebra Algorithms via Sparser Subspace Embeddings. *FOCS*, 117–126, 2013.

[6] Thorup Mikkel, Zhang Yin. Tabulation based 4-universal hashing with applications to second moment estimation. *SODA*, 615–624, 2004.

[7] Charikar Moses, Chen Kevin C., Farach-Colton Martin. Finding Frequent Items in Data Streams. *Theor. Comput. Sci.* 3–15, 2004.

[8] Sarlós Tamás. Improved Approximation Algorithms for Large Matrices via Random Projections. *IEEE Symposium on Foundations of Computer Science (FOCS).* 143–152, 2006.