CS 493: Algorithms for Massive Data Sets	Random Sampling
April 16, 2002	Scribe: Yitzhak Mandelbaum

1 Review

The last class concluded in the middle of Dasgupta and Gupta's proof of the Johnson-Lindenstrauss lemma. To review:

Lemma 1.1 *We can project n points from $\mathbb{R}^d \rightarrow \mathbb{R}^k$, while preserving distances properly. That is,*

$$\textit{distance } r \rightarrow r(1 \pm \epsilon)$$

with

$$k = O\left(\frac{\lg n}{\epsilon^2}\right)$$

In particular, we choose k to be a positive integer such that

$$k \geq \frac{4 \ln n}{\frac{\epsilon^2}{2} \cdot \frac{\epsilon^3}{3}}$$

Take $\mathbb{R}^d$, choose a random $\mathbb{R}^k$ (choose k random unit vectors), project all points into this new space, and then scale the distances between the points.

2 Finishing the Johnson-Lindenstrauss Proof

Lemma 2.1

$$\Pr[d(x_1^2 + \cdots + x_k^2) \leq k\beta(x_1^2 + \cdots + x_d^2)] \leq \beta^{\frac{k}{2}} \left(1 + \frac{k(1-\beta)}{d-k}\right)^{\frac{d-k}{2}}$$

Proof. All of the x_i are normal variables, that is, they take on values according to the normal distribution. Therefore, from the previous lecture we know that:

$$E[e^{tx^2}] = \frac{1}{\sqrt{1-2t}} \qquad -\infty < t < \frac{1}{2} \quad \text{(for } x \text{ normal)}$$

Now,

$$
\begin{aligned}
&\Pr[d(x_1^2 + \cdots + x_k^2) \le k\beta(x_1^2 + \cdots + x_d^2)] \\
&\quad = \Pr[k\beta(x_1^2 + \cdots + x_d^2) - d(x_1^2 + \cdots + x_k^2) \ge 0] \\
&\quad = \Pr\left[e^{t(k\beta(x_1^2+\cdots+x_d^2)-d(x_1^2+\cdots+x_k^2))} \ge 1\right] \qquad \text{(with } t \text{ a parameter)} \\
&\quad \le E\left[e^{tk\beta x^2}\right]^{d-k} \cdot E\left[e^{t(k\beta-d)x^2}\right]^{k} \\
&\quad = (1-2tk\beta)^{-\frac{d-k}{2}} \cdot (1-2t(k\beta-d))^{-\frac{k}{2}} \qquad \text{(from claim)} \\
&\quad = g(t)
\end{aligned}
$$

Now, choose t so as to minimize g. Alternatively, consider $f(t)$ where

$$f(t) = \left(\frac{1}{g(t)}\right)^2 = (1-2tk\beta)^{d-k}$$

and maximize $f(t)$ instead.

Our constraints are $tk\beta < 1/2$ and $t(k\beta - d) < 1/2$. Using calculus we get:

$$t_{\text{opt}} = \frac{1-\beta}{2\beta(d-k\beta)}$$

■

3 Variants of Johnson-Lindenstrauss

There are a number of other algorithms which are simpler than the original J-L algorithm while providing similar guarantees.

Frankl-Maehara use orthogonal unit-vectors.

Indyk-Motwani use any unit vectors.

Achlioptas use simple projections.

4 Random Sampling

There exists a class of problems for which solving the problem on the entirety of a large data set can be prohibitively expensive. For a number of these problems, techniques have been developed to solve the problem on a randomly chosen sample of the data set, and then to translate the solution (or sometimes simply apply it) to the larger data set. The goal of these techniques is often that the solution for the sample be within an ϵ fraction of the solution for the larger data set with some probability $1 - \delta$.

Before now, we have seen examples of streaming algorithms for calculating a number of interesting properties of large data sets. These algorithms make only one pass over the data and use a relatively small amount of memory to perform there calculations.

We now consider a new class of algorithms, known as sampling algorithms, which do not look at all of the data. As with streaming, memory use is a function of sample size, which tends to be small relative to the size of the data set. Sampling algorithms are more restrictive than streaming algorithms, and form a strict subset of the streaming algorithms. To see this, we can convert any sampling algorithm into a streaming algorithm by using a hypothetical one-pass "sampler" to produce a sample set and then passing this set to the sampling algorithm.

4.1 Sampling a Data Stream

If the size of the stream that we would like to sample is known a priori, then sampling is actually quite easy.

Algorithm 1 (Naive Sampling) *Given a data set of size N, we can create a random sample of size k with one pass over the data as follows: consider each element in turn. With probability k/N add the element to the sample set. If k sample elements are chosen before the end of the stream is reached, then for every successful sampling (i.e. for every element chosen to be added to the sample set), choose an evictee from the set uniformly at random. Replace the evictee with the new element.*

However, it is often the case with large data sets that we do not know the size of the set a priori. It is therefore necessary to devise a sampling algorithm that does not depend on such knowledge.

Algorithm 2 (Reservoir Sampling) *1. Add the first k items to the sample.*

2. Having seen $n-1$ items, choose item n with probability k/n. If chosen, choose an evictee from the sample set uniformly at random, to be replaced by item n.

Note that we are sampling without replacement.

The probability, then, that the n^{th} item belongs to the sample set is k/n. Intuitively, it should be

$$\frac{\binom{n-1}{k-1}}{\binom{n}{k}} \qquad \frac{\text{number of } k\text{-element subsets that include this element}}{\text{number of } k\text{-element subsets}}$$

$$= \frac{k}{n}$$

This sampling technique is known as **Reservoir Sampling**. A more efficient scheme exists whereby we guess when the next successful choice will be and do not examine items in between.

With reservoir sampling, any sampling algorithm can be converted into a streaming algorithm, and streaming is, therefore, strictly more powerful than sampling.

5 Examples

5.1 Medians and Quantiles

Definition 5.1 *The ϕ-quantile of a set of N elements is the element of rank $\lceil \phi N \rceil$. The median is the* 50%*-quantile.*

Definition 5.2 *The ϵ-approximate ϕ-quantile is the element whose rank is within the range $\lceil (\phi \pm \epsilon) N \rceil$.*

A brief history of median and quantile results:
Blum, Floyd, Pratt, Rivest, Tarjan:

k-th largest element can be found in $5.43N$ comparisons. Improved to $2.9423N$ comparisons. Median $\geq 1.5N$ comparisons. Improved to $(2+\alpha)N$ with $\alpha \approx 2^{-40}$.
Frances Yao:

Proved a lower-bound of $\Omega(N)$ comparisons for a deterministic approximate median algorithm.
Manku, Rajagopatan, Lindsay: "Approximate Medians and Other Quantities in One Pass and with Limited Memory"

With randomization, an ϵ-approximation of the median can be achieved with probability $1-\delta$ using only

$$O\left(\frac{1}{\epsilon^2}\log\left(\frac{1}{\delta}\right)\right)$$

elements. If one pass is allowed, then the deterministic algorithm uses $O(1/\epsilon \log^2(\epsilon N))$ space.

We will now prove the result obtained by Manku, et al. with regard to randomized algorithms for ϵ-approximations of ϕ-quantiles.
Proof. We obtain an approximation of the ϕ-quantile of a set of elements by taking a sample set, S, and finding its ϕ-quantile. In order for the sample's ϕ-quantile to be an accurate approximation of the set's ϕ- quantile, it must fall within the desired ϵ range in the data set. In other words, for it to be a bad approximation, at least $\phi|S|$ sample elements must fall below the $(\phi-\epsilon)N$th element of the set or at least $(1-\phi)|S|$ sample elements must fall above the $(\phi+\epsilon)N$th element of the set.

Now,

$$\begin{aligned} &Pr[\text{the number of elements in upper range } > (1-\phi)|S|\,] \\ &= Pr[\text{the number of elements in rest of range } < \phi|S|\,] \end{aligned}$$

As the expected number of elements drawn from the range $0\ldots(\phi+\epsilon)N$ is $(\phi+\epsilon)|S|$, the probability that there be more elements is $< e^{-2\epsilon^2|S|^2}$. By a symetry argument, the probability that at least $\phi|S|$ sample elements fall below the $(\phi-\epsilon)N$th element of the set is also $< e^{-2\epsilon^2|S|^2}$, and so, the probability that either of these events occur, δ, is $< 2e^{-2\epsilon^2|S|^2}$. Therefore, with probability $1-\delta$ the approximation succeeds and

$$|S| \geq \frac{1}{2\epsilon^2}\log\frac{2}{\delta} = O\left(\frac{1}{\epsilon^2}\log\frac{1}{\delta}\right)$$

■

One further result about computing medians in one pass:
Munro-Patteson '80: exact median in P passes takes space $= O(N^{\frac{1}{P}})$

5.2 Clustering

k-center objective: minimize the maximum distance of points to the cluster center.

Now, take a sample, cluster it, and then guarantee that the solution to the sample applies to the original data set by excluding at most an ϵ-fraction of the points, with high probability (i.e. $1-\delta$). Such a guarantee can be made for a sample size $= O(k \log n/\epsilon)$.

To rephrase: Given any clustering solution on the original data set which excludes greater than an ϵ-fraction of points, the same solution excludes no points in the sample in the sample with low probability, $\delta/(\binom{n}{k}\binom{n}{2})$. That is, the "sample is representative" for all solutions.

To see this, note that the total number of possible solutions is the total number of centers times the total number of radiuses:

$$\binom{n}{k}\binom{n}{2}$$

Now, in order for no points in the sample solution to be excluded, none of the $> \epsilon n$ excluded points can be in the sample set. The probability that any given sample point chosen from the data set not be in the excluded fraction of the points is $< 1-\epsilon$, so the probability that all the sample points not be from the excluded fraction is:

$$\begin{aligned}
&\Pr[\text{all sample points are from included fraction of data}] < (1-\epsilon)^{|S|} \\
&= (1-\epsilon)^{O(\frac{k \log n + \log \frac{1}{\delta}}{\epsilon})} \\
&\leq (1-\epsilon)^{c(\frac{k \log n + \log \frac{1}{\delta}}{\epsilon})} \\
&= (1-\epsilon)^{\frac{c}{\epsilon}(k \log n + \log \frac{1}{\delta})} \\
&= e^{\ln(1-\epsilon)\frac{c}{\epsilon}(k \log n + \log \frac{1}{\delta})} \\
&\leq e^{-(\epsilon+\epsilon^2/2)\frac{c}{\epsilon}(k \log n + \log \frac{1}{\delta})} \\
&< e^{-c(k \log n + \log \frac{1}{\delta})} \\
&< \frac{\delta}{\binom{n}{k}\binom{n}{2}} \qquad \text{because } \lg \binom{n}{k} < k \lg n
\end{aligned}$$