princeton univ. F'16 cos 521: Advanced Algorithm Design

Lecture 9: High Dimensional Geometry, Curse of Dimensionality, Dimension Reduction

Lecturer: *Pravesh Kothari* Scribe:*Sanjeev Arora*

High-dimensional vectors are ubiquitous in applications (gene expression data, set of movies watched by Netflix customer, etc.) and this lecture seeks to introduce high dimensional geometry. We encounter the so-called *curse of dimensionality* which refers to the fact that algorithms are simply harder to design in high dimensions and often have a running time exponential in the dimension. We also encounter the *blessings of dimensionality*, which allows us to reason about higher dimensional geometry using tools like Chernoff bounds. We also show the possibility of *dimension reduction* — it is sometimes possible to reduce the dimension of a dataset, for some purposes.

Notation: For a vector $x \in \Re^d$ its *ℓ_2-norm* is $|x|_2 = (\sum_i x_i^2)^{1/2}$ and the ℓ_1-norm is $|x|_1 = \sum_i |x_i|$. For any two vectors x, y their Euclidean distance refers to $|x-y|_2$ and Manhattan distance refers to $|x-y|_1$.

High dimensional geometry is inherently different from low-dimensional geometry.

Example 1 Consider how many *almost orthogonal* unit vectors we can have in space, such that all pairwise angles lie between 88 degrees and 92 degrees.

In $\Re^2$ the answer is 2. In $\Re^3$ it is 3. (Prove these to yourself.)

In $\Re^d$ the answer is $\exp(cd)$ where $c > 0$ is some constant.

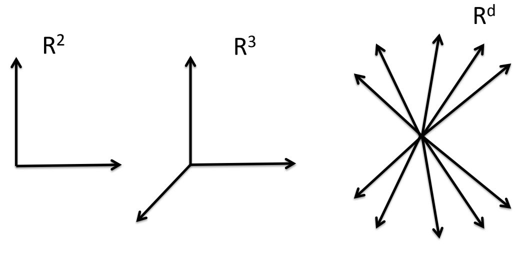

Figure 1: Number of almost-orthogonal vectors in $\Re^2, \Re^3, \Re^d$

Example 2 Another example is the ratio of the the volume of the unit sphere to its circumscribing cube (i.e. cube of side 2). In $\Re^2$ it is $\pi/4$ or about 0.78. In $\Re^3$ it is $\pi/6$ or about 0.52. In d dimensions it is d^{-cd} for some constant $c > 0$.

Let's start with useful generalizations of some geometric objects to higher dimensional geometry:

- The *n-cube* in $\Re^n$: $\{(x_1...x_n) : 0 \leq x_i \leq 1\}$. To visualize this in $\Re^4$, think of yourself as looking at one of the faces, say $x_1 = 1$. This is a cube in $\Re^3$ and if you were able to look in the fourth dimension you would see a parallel cube at $x_1 = 0$. The visualization in $\Re^n$ is similar.

  The volume of the n-cube is 1.

- The unit *n-ball* in $\Re^d$: $B_d := \{(x_1...x_d) : \sum x_i^2 \leq 1\}$. Again, to visualize the ball in $\Re^4$, imagine you have sliced through it with a hyperplane, say $x_1 = 1/2$. This slice is a ball in $\Re^3$ of radius $\sqrt{1 - 1/2^2} = \sqrt{3}/2$. Every parallel slice also gives a ball.

  The volume of B_d is $\frac{\pi^{d/2}}{(d/2)!}$ (assume d is even if the previous expression bothers you), which is $\frac{1}{d^{\Theta(d)}}$.

- In $\Re^2$, if we slice the unit ball (i.e., disk) with a line at distance $1/2$ from the center then a significant fraction of the ball's volume lies on each side. In $\Re^d$ if we do the same with a hyperplane, then the radius of the $d-1$ dimensional ball is $\sqrt{3}/2$, and so the volume on the other side is negligible. In fact a constant fraction of the volume lies within a slice at distance $1/\sqrt{d}$ from the center, and for any $c > 1$, a $(1 - 1/c)$-fraction of the volume of the d-ball lies in a strip of width $O(\sqrt{\frac{\log c}{d}})$ around the center. This fact is closely related to Chernoff bounds, and a related phenomenon called *concentration of measure.* (*Measure* is the mathematical name for volume.)

- A good approximation to picking a random point on the surface of B_n is by choosing random $x_i \in \{-1, 1\}$ independently for $i = 1..n$ and normalizing to get $\frac{1}{\sqrt{n}}(x_1, ..., x_n)$.

  An exact way to pick a random point on the surface of B^n is to choose x_i from the standard normal distribution for $i = 1..n$, and to normalize: $\frac{1}{l}(x_1, ..., x_n)$, where $l = (\sum_i x_i^2)^{1/2}$.

0.1 Number of almost-orthogonal vectors

Now we show there are $\exp(d)$ vectors in $\Re^d$ that are almost-orthogonal. Recall that the angle between two vectors x, y is given by $\cos(\theta) = \langle x, y \rangle / |x|_2 \, |y|_2$.

LEMMA 1
Suppose a is a unit vector in $\Re^n$. Let $x = (x_1, ..., x_n) \in R^n$ be chosen from the surface of B_n by choosing each coordinate at random from $\{1, -1\}$ and normalizing by factor $\frac{1}{\sqrt{n}}$. Denote by X the random variable $a \cdot x = \sum a_i x_i$. Then:

$$Pr(|X| > t) < e^{-nt^2}$$

PROOF: We have:

$$\mu = E(X) = E(\sum a_i x_i) = 0$$

$$\sigma^2 = E[(\sum a_i x_i)^2] = E[\sum a_i a_j x_i x_j] = \sum a_i a_j E[x_i x_j] = \sum \frac{a_i^2}{n} = \frac{1}{n}.$$

Using the Chernoff bound, we see that,

$$Pr(|X| > t) < e^{-(\frac{t}{\sigma})^2} = e^{-nt^2}.$$

□

Corollary 2
If x, y are chosen at random from $\{-1,1\}^n$, and the angle between them is $\theta_{x,y}$ then

$$Pr\left[|cos(\theta_{x,y})| > \sqrt{\frac{\log c}{n}}\right] < \frac{1}{c}.$$

Hence by if we pick say $\sqrt{c}/2$ random vectors in $\{-1,1\}^n$, the union bound says that the chance that they all make a pairwise angle with cosine less than $\sqrt{\frac{\log c}{n}}$ is less than 1/2. Hence we can make $c = \exp(0.01n)$ and still have the vectors be almost-orthogonal (i.e. cosine is a very small constant).

0.2 Curse of dimensionality

Curse of dimensionality —a catchy term due to Richard Bellman, who also invented the term dynamic programming—refers to the fact that problems can get a lot harder to solve on high-dimensional instances. This term can mean many things.

The simplest is NP-hardness. Some problems can be easy to solve for dimension 2 and become NP-hard as d is allowed to grow.

Another is *sample complexity.* Many simple machine learning algorithms are based upon *nearest neighbor* ideas: maintain a database of points S you know how to solve, and when presented with a new point y, use the solution/answer for the point in S that is closest to y. This runs into problems in $\Re^d$ because, as we saw, you need more than $\exp(d)$ points in S before a random y is guaranteed to have a reasonably close point in S. Thus for any reasonable d, the data is spread too thinly. In practice people try to reduce dimension in some way before applying a nearest-neighbor method. A different variant of this curse is when d is so large —e.g. a million—that even d^2 samples are too many to hope for. Then one has to look for really sample-efficient algorithms.

Another interpretation for curse of dimensionality is that algorithms for simple problems — —nearest neighbor, minimum spanning tree, point location etc.— become more inefficient, though they stay polynomial-time. For example, minimum spanning tree for n points in d dimensions can be solved in time nearly linear in n for $d = 2$, but for larger d it has to be n^2 or depend upon $\exp(d)$.

I hereby coin a new term: *Blessing of dimensionality.* This refers to the fact that many phenomena become much clearer and easier to think about in high dimensions because one can use simple rules of thumb (e.g., Chernoff bounds, measure concentration) which don't hold in low dimensions.

0.3 Dimension Reduction

Now we describe a central result of high-dimensional geometry (at least when distances are measured in the ℓ_2 norm). Problem: Given n points $z^1, z^2, ..., z^n$ in $\Re^n$, we would like to find n points $u^1, u^2, ..., u^n$ in $\Re^m$ where m is of low dimension (compared to n) and the metric restricted to the points is almost preserved, namely:

$$\|z^i - z^j\|_2 \leq \|u^i - u^j\|_2 \leq (1+\varepsilon)\|z^j - z^j\|_2 \ \forall i, j. \tag{1}$$

The following main result is by Johnson & Lindenstrauss :

Theorem 3
In order to ensure (1), $m = O(\frac{\log n}{\varepsilon^2})$ suffices, and in fact the mapping can be a linear mapping.

The following ideas do not work to prove this theorem (as we discussed in class): (a) take a random sample of m coordinates out of n. (b) Partition the n coordinates into m subsets of size about n/m and *add* up the values in each subset to get a new coordinate.

Proof: Choose m vectors $x^1, ..., x^m \in \Re^n$ at random by choosing each coordinate randomly from $\{\sqrt{\frac{1+\varepsilon}{m}}, -\sqrt{\frac{1+\varepsilon}{m}}\}$. Then consider the mapping from $\Re^n$ to $\Re^m$ given by

$$z \longrightarrow (z \cdot x^1, z \cdot x^2, \ldots, z \cdot x^m).$$

In other words $u^i = (z^i \cdot x^1, z^i \cdot x^2, ..., z^i \cdot x^m)$ for $i = 1, \ldots, k$. (Alternatively, we can think of the mapping as a random linear transformation $u = A \cdot z$ where A is a matrix with random entries in $\{\sqrt{\frac{1+\varepsilon}{m}}, -\sqrt{\frac{1+\varepsilon}{m}}\}$.) We want to show that with positive probability, $u^1, ..., u^k$ has the desired properties. This would mean that there exists at least one choice of $u^1, ..., u^k$ satisfying inequality (1). To show this, first we write the expression $\|u^i - u^j\|$ explicitly:

$$\|u^i - u^j\|^2 = \sum_{k=1}^{m} \left(\sum_{l=1}^{n} (z_l^i - z_l^j) x_l^k \right)^2 .$$

Denote by z the vector $z^i - z^j$, and by u the vector $u^i - u^j$. So we get:

$$\|u\|^2 = \|u^i - u^j\|^2 = \sum_{k=1}^{m} \left(\sum_{l=1}^{n} z_l x_l^k \right)^2 .$$

Let X_k be the random variable $(\sum_{l=1}^n z_l x_l^k)^2$. Its expectation is $\mu = \frac{1+\varepsilon}{m}\|z\|^2$ (can be seen similarly to the proof of Lemma 1). Therefore, the expectation of $\|u\|^2$ is $(1+\varepsilon)\|z\|^2$. If we show that $\|u\|^2$ is concentrated enough around its mean, then it would prove the theorem. More formally, this is done in the following Chernoff bound lemma. □

Lemma 4
There exist constants $c_1 > 0$ and $c_2 > 0$ such that:

1. *$Pr[\|u\|^2 > (1+\beta)\mu] < e^{-c_1 \beta^2 m}$*

2. $Pr[\|u\|^2 < (1-\beta)\mu] < e^{-c_2\beta^2 m}$

Therefore there is a constant c such that the probability of a "bad" case is bounded by:

$$Pr[(\|u\|^2 > (1+\beta)\mu) \vee (\|u\|^2 < (1-\beta)\mu)] < e^{-c\beta^2 m}$$

Now, we have $\binom{n}{2}$ random variables of the type $\|u_i - u_j\|^2$. Choose $\beta = \frac{\varepsilon}{2}$. Using the union bound, we get that the probability that any of these random variables is not within $(1 \pm \frac{\varepsilon}{2})$ of their expected value is bounded by

$$\binom{n}{2} e^{-c\frac{\varepsilon^2}{4}m}.$$

So if we choose $m > \frac{8(\log n + \log c)}{\varepsilon^2}$, we get that with positive probability, all the variables are close to their expectation within factor $(1 \pm \frac{\varepsilon}{2})$. This means that for all i,j:

$$(1-\frac{\varepsilon}{2})(1+\varepsilon)\|z^i - z^j\|^2 \leq \|u^i - u^j\|^2 \leq (1+\frac{\varepsilon}{2})(1+\varepsilon)\|z^i - z^j\|^2$$

Therefore,

$$\|z_i - z_j\|^2 \leq \|u^i - u^j\|^2 \leq (1+\varepsilon)^2\|z^i - z^j\|^2,$$

and taking square root:

$$\|z^i - z^j\| \leq \|u^i - u^j\| \leq (1+\varepsilon)\|z^i - z^j\|,$$

as required.

More Questions about Dimension Reduction. The JL lemma fits within a long history of study of metric spaces in mathematics. Here are some other questions that have been studied.

Question 1: The above dimension reduction preserves (approximately) ℓ_2-distances. Can we do dimension reduction that preserves ℓ_1 distance? This was an open problem for many years until Brinkman and Charikar (of Princeton) showed in 2003 that no such dimension reduction is possible. They exhibit a cleverly chosen set of n points in $\Re^n$ such that their interpoint distances measured in ℓ_1 norm *cannot* be captured using n points in $\Re^m$ when m is much smaller than n. This rules out a very general class of mappings, not just linear mappings used in the JL lemma.

Question: Is the JL theorem tight, or can we reduce the dimension even further below $O(\log n/\varepsilon^2)$? Alon has shown that this is essentially tight.

Finally, we note that there is a now-extensive literature on more efficient techniques for JL-style dimension reduction, with a major role played by a 2006 paper of Ailon and Chazelle (also of Princeton). Do a google search for "Fast Johnson Lindenstrauss Transforms."These are most effective when the points are *sparse* vectors (i.e., have many zero coordinates) in which case the running time can be much lower.

0.3.1 Locality preserving hashing

Suppose we wish to hash high-dimensional vectors so that nearby vectors tend to hash into the same bucket. To do this we can do a random projection into say the cube in 5 dimensions. We discretise the cube into smaller cubes of size ε. Then there are $1/\varepsilon^5$ smaller cubes; these can be the buckets.

This is simplistic; more complicated schemes have been constructed. Things get even more interesting when we are interested in ℓ_1-distance.

0.3.2 Dimension reduction for efficiently learning a linear classifier

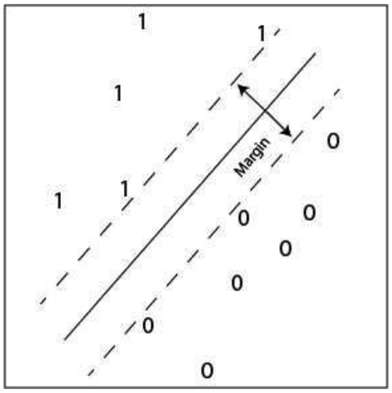

Figure 2: Margin of a linear classifier with respect to some labeled points

Suppose we are given a set of m data points in $\Re^d$, each labeled with 0 or 1. For example the data points may represent emails (represented by the vector giving frequencies of various words in them) and the label indicates whether or not the user labeled them as spam. We are trying to learn the rule (or "classifier") that separates the 1's from 0's.

The simplest classifier is a halfspace. Finding whether there exists a halfspace $\sum_i a_i x_i \geq b$ that separates the 0's from 1's is solvable via Linear Programming. This LP has $n+1$ variables and m constraints.

However, there is no guarantee in general that the halfspace that separates the training data will generalize to new examples? ML theory suggests conditions under which the classifier does generalize, and the simplest is *margin*. Suppose the data points are unit vectors. We say the halfspace has *margin* ε if every datapoint has distance at least ε to the halfspace.

In the next homework you will show that if such a margin exists then dimension reduction to $O(\log n/\varepsilon^2)$ dimensions at most halves the margin. Hence the LP to find it only has $O(\log n/\varepsilon^2)$ variables instead of $n+1$.

Bibliography:

1. Noga Alon. Problems and results in extremal combinatorics I. *Discrete Mathematics*, 273(1- 3):3153, 2003.

2. W. Brinkman and M. Charikar. On the impossibility of Dimension Reduction in ℓ_1. IEEE FOCS 2003.

3. N. Ailon and B. Chazelle. Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform. In *Proceedings of the 38th ACM Symposium on Theory of Computing (STOC)*, pages 557563, 2006.