19

1 Admistrivia

• Wednesday, Lyle will talk on Eigenwords

• Next monday (Sept 24) we will move to R2D2 441. (Right hand glass conference room in statistics.)

2 Draw some pictures

30 degrees

• Draw $d/2$ pairs of axis

• Draw $d/2$ vectors with angle 30 degrees

• Draw exotic axis: $a$ and $b - \frac{\langle a, b \rangle}{\langle a, a \rangle} a$

• Question: what angle do I draw?
A few others pairs:

- Draw 120 degree pair
- Draw 90 degree pair
- Draw random pair: say it looks like 90 degrees on exotic axis

3 Some necessary math

- Random projects look like normals
- Proof:
  - Let $X$ be an arbitrary vector, and $R$ be a random direction (chosen from the surface of the unit ball)
  - Rotate axis so that $X$ points in the direction of the first coordinate
  - Now compute inner product: which is nothing but a random projection of a sphere onto the first coordinate
  - This looks like a normal

4 Trival theorem: Inner product

What is the angle of the exotic axis in general?

- Easier to compute cosine of angle
cos(θ) = \langle a, b \rangle / \sqrt{\langle a, a \rangle \langle b, b \rangle}

WOLG assume \langle a, a \rangle = \langle b, b \rangle = 1.

So we only need \langle a, b \rangle

But, \langle a, b \rangle = \sum_{i=1}^{d/2} a_x b_x + a_y b_y = \sum_{i=1}^{d/2} \langle a, b \rangle_i

NOTE: Each 2-D picture is of two correlated bivariate normals. So we can draw them exactly as a, Z ∼ N(0, I), and b = \rho a + \sqrt{1 - \rho^2} Z, where \rho is \langle a, b \rangle / \sqrt{\langle a, a \rangle \langle b, b \rangle}.

5 Theorem: E(JL)

If we project two vectors down to a random pair of axes which are at angle θ with respect to each other, the “expected” angle is θ.

6 JL itself

Now JL is pretty obvious. We need to sample enough random directions such that we can estimate the expected value of the cos of the angle. ow apply Bonferoni, and we are done.
7 Example: Balls / normals

Consider IID normal vectors in $d$ space. We actually know the distance between any two of these vectors. If each coordinate is a $N(0, 1)$, then the distance between two of them is $\sqrt{2d}$. This will be pretty accurate until we have something like $2^d$ different vectors. So we don’t really need $\log(d)$ dimensions to answer the distance question--but we DO need them to represent the answer!

8 Will it work for regression?

Suppose we $X_i$ which are $d$ different vectors in $d$ dim space, each an independent normal. Suppose $Y \sum B_i X_i/\sqrt{d}$ where $B_i$ are IID bernulli trials. Then $Y$ is also a standard normal. If we knew all the distances between these, we woud see that $\text{dist}(Y, X_i) \approx \sqrt{2d}$ for all $X_i$–whether or not it is part of the regression or not.

So, using JL we would need to get a very accurate estimate (within a fraction of about $1 + 1/\sqrt{d}$ accuracy.) This will take about all of our $d$ diminsions.

9 Can JL determine a cluster?

Suppose we have two clusters, in one coordinate they are about $k$ sd apart.

Can JL determine this direction? Not really since the distance between a point in the same group are $\sqrt{2d}$ away from each other
but in different groups they are $\sqrt{2d + k^2}$ away from each other. So again—we need extreme accuracy. I.e. $k$ needs to be about $\sqrt{d}$ to be seen.

10 Will it work for PCA?

Suppose the first coordinate is a $N(0, k)$ and all others are $N(0, 1)$. Can JL determine this direction? Not really since the distance between a point on the tip of this direction is $\sqrt{2d + k^2}$ to other points. So again—we need extreme accuracy. I.e. $k$ needs to be about $\sqrt{d}$ to be seen.

11 Fast matrix factorization

What is a random projection anyway?

• Just a matrix multiply
• Put data in a matrix $X$ which is $n \times d$
• multiply by a random $d \times k$ matrix
• Now we are in a $k$ dimensional space

Random projections do not find PCA solutions. But, if we looked at $(X'X)^{\log(d)}$ then singular values which were size 2 are now size $d$. So a random project will solve it.

New algorithm:

• Pick random matrix $M$
• Compute $X'XX'X X'XM$

• Now compute SVD of resulting object and it will find the first principle component of $X$

This is basically what I call ”Tropp”’s algorithm. It is a fast way of discovering a low rank version of a matrix.

12 Tropp

Describe his method and why we orthogonalize each step.