CS 6966 : High-Dimensional Data Analysis (Fall 2024) : Jeff M. Phillips

L1: Curse of Dimensionality: Basic Geometry

Vectors $x = (x_1, x_2, ..., x_d) \in \mathbb{R}^d$

For $a, b \in \mathbb{R}^d$

- ℓ_2 or Euclidean distance: $\|a - b\| = \|a - b\|_2 = \sqrt{\sum j = 1^d (a_j - b_j)^2}$
- ℓ_∞ or max distance: $\|a - b\|_\infty = \max_{j=1}^d |a_j - b_j|$

1. Volume of cube vs. inscribed sphere with dimension d.

$[-1, 1]^d$ square has volume 2^d for all d
$2 \times 2 \times ... \times 2 = 2^d$

volume of ball inscribed
radius = 1

$$\frac{r^d \pi^{d/2}}{\Gamma(d/2+1)} \approx \frac{1 \cdot \pi^{d/2}}{(d/2)!}$$

d	ball-vol	box-vol
2	3.14	4
4	4.93	16
6	5.16	64
8	4.05	256
10	2.55	1024
12	1.33	4096
14	0.60	16384
16	0.24	65536
18	0.08	262144

what happens for 1x1x1 box? Is it inscribed?

2. Approx orthogonality of high-d Gaussians

$a, b \sim G_d(0, 1)$ a d-dimensional Gaussian

$E[(a_i - b_i)^2] = E[a_i^2] + E[b_i^2] - 2E[a_i b_i] = Var[a_i] + Var[b_i] - 2E[a_i]E[b_i] = 2$
so $\|a - b\|^2 = 2d$

Need also: $E[||a||^2] = d$

Pythagorean if a orthogonal to b (w.r.t 0) then $\|a\| = \sqrt{d}$, $\|b\| = \sqrt{d}$ and hence $\|a - b\|^2 = \|a\|^2 + \|b\|^2 = 2d$

3. Big Annulus

For any object $A \subset \mathbb{R}^d$ let

$$(1 - \varepsilon)A = \{(1 - \varepsilon)x | x \in A\}$$

(imagine shrinking into origin .. but works more generally)

Thm: $Vol((1 - \varepsilon)A = (1 - \varepsilon)^d Vol(A)$

proof: decompose A into a set of d-dim cubes $C_1, C_2, ...$
so $Vol(A) = \sum_j Vol(C_j)$
Each C_j has side length l_j, and $Vol(C_j) = l_j^d$
We can replace $(1-\varepsilon)A$ by same series $(1-\varepsilon)C_1, (1-\varepsilon)C_2, ...$

$$Vol((1-\varepsilon)C_j) = ((1-\varepsilon)l_j)^d = (1-\varepsilon)^d Vol(C_j)$$

QED

Now notice that $(1-\varepsilon)^d \leq e^{-\varepsilon d}$
thus for fixed ε, as d grows larger than $1/\varepsilon$ and then
$e^{-\varepsilon d}$ exponentially decreases after that.

Consequence:

- For unit ball B subset $\mathbb{R}^d$:
  $1 - e^{-\varepsilon d}$ fraction of volume in $B \;\; (1-\varepsilon)B$.

- For eps = 1/10, and d=100 then
  $1 - e^{-\varepsilon d} = 1 - e^{-10} = \; 0.99995$
  within the last 10% of radius
- For eps = 1/20, and d=100 then
  $1 - e^{-\varepsilon d} = 1 - e^{-5} = \; 0.993$
  within the last 5% of radius
- For eps = 1/25, and d=100 then
  $1 - e^{-\varepsilon d} = 1 - e^{-4} = \; 0.98$
  within the last 4% of radius
- For eps = 1/50, and d=100 then
  $1 - e^{-\varepsilon d} = 1 - e^{-2} = \; 0.86$
  within the last 2% of radius

4. Volume near Equator

Consider unit ball B subset $\mathbb{R}^d$
Let v = (1,0,0, …, 0) – think of this as pointing "up" or "North"

Consider the "tropical zone" as being near the equator if the first coordinate has magnitude at most $c/\sqrt{d}$ for some $c \geq 1$ (think of c=10) and d=100. We show that

$$Vol_d(B_r) = \frac{r^d \pi^{d/2}}{\Gamma(d/2)} = r^d V_d$$

The "disk" at $1/\sqrt{d}$ above equator is a $(d-1)$-dimensional Ball with radius
$x = \sqrt{1-1/d}$ since $1^2 = x^2 + 1/d$
So $Vol_{d-1}(B_{\sqrt{1-1/d}}) = (1-1/d)^{d/2} V_{d-1}$

Also "disk" at $2/\sqrt{d}$ above equator is a $(d-1)$-dimensional Ball with radius
$\sqrt{1-4/d}$
So $Vol_{d-1)}(B_{\sqrt{1-4/d}}) = (1-4/d)^{d/2} V_{d-1}$

$$\frac{Vol_{d-1}(B_{\sqrt{1-1/d}})}{Vol_{d-1}(B_{\sqrt{1-4/d}})} = \frac{(1-1/d)^{d/2} V_{d-1}}{(1-4/d)^{d/2} V_{d-1}} \approx \frac{e^{-1/2}}{e^{-2}}$$

Each are "layers of a cake" with same height $1/\sqrt{d}$.
Volume decreases, geometrically so layers $j = 3...\sqrt{d} <$ layer 2

$$\sum_{j=2}^{\sqrt{d}} Vol_{d-1}(B_{\sqrt{1-j^2/d}}) < 2Vol_{d-1}(B_{\sqrt{1-4/d}})$$

So $e^{-1/2} > 2/e^2$ … the lowest layer is larger than all above layers combined (for large d)

If we make the first layer $\kappa/\sqrt{d}$ for $\kappa > 1$, then the gap is even larger.

5. Spiky Boxes

Now consider a ball B with radius 1 in $\mathbb{R}^d$.
And a box $C = [-1/2, 1/2]^d$ with volume $Vol(C) = 1$

Note $(1 - 1/2)B \subset C$, where $(1 - 1/2)B$ is ball or radius 1/2.

For $d = 2$, we have $C \subset B$.

For $d = 4$, still $C \subset B$
but $\|0 - (1/2, 1/2, 1/2, 1/2)\| = \sqrt{4(1/2)^2} = 1$
so the corner of C touches now touches boundary of B.

How about $d = 5$?
Corner of C outside of B.

How about $d = 8$?
Corner $c \in C$ has distance $\sqrt{8(1/2)^2} = 2$, far outside of B
center of face $(1/2, 0, ..., 0)$, well inside of B.

In general, corner a distance $\sqrt{d}/2$ from 0

Most of volume of the C is outside of B, since $Vol(B) \to 0$ as d grows

Question:

How do you sample a random point in B in $\mathbb{R}^d$ for large d?

L2: Curse of Dimensionality: Nearest Neighbors

Consider $X \subset \mathbb{R}^d$.
Query point $q \in \mathbb{R}^d$.
Nearest neighbor is point $NN_X(q) = \arg\min_{x \in X} \|x - q\|$.

We can solve $NN_X(q)$ by calculating $\|x - q\|$ for all $x \in X$.

The challenge is to:

1. **Pre-process:** Build data structure S in time that is not much longer than reading data X
2. **Query:** Solve $NN_X(q)$ using S, in time that is much faster than $O(nd)$.

Rumor:

One can construct examples $X \sim Unif(B_d(r = 1))$ so for each $x \in X$ the distance to all other data points are about the same. So the notion of nearest neighbor is not that meaningful.

These are mathematically true, but not reflected in data analysis. Real data rarely exhibits this behavior. If it does, it is (typically) not interesting for analysis – at least not the analysis where this matters.

1-dimensional Nearest Neighbor Search

First consider data $X \in \mathbb{R}^1$. *How do we preprocess X for efficient queries?*

Preprocess:

- Sort X.
- Build binary tree.
- For each internal node: store smallest value, at the root of subtree.
- $O(n \log n)$ time, $O(n)$ space.

NN Query:

- From root: check right-child if value smaller –> recurse left
  otherwise, recurse right
- At leaf: compare value to right-adjacent, return closest.
- $O(\log n)$ time.

Range query: interval I = [a,b]. Returns all $X \cap [a, b]$.

- $O(\log n + k)$ to return k items.
- $O(\log n)$ to return number of items.

d-dimensional Range Search

How do generalize range search to 2 dimensions?

Range Tree:

- Build binary tree on first coordinate
- Each internal node builds binary tree on second coordinate (over subtree data).
- Space $O(n \log n)$
- Query in $O(\log^2 n)$ time
- Save factor $\log n$ with "fractional cascading"

How does Range Tree work in d dimensions?

- recursively build binary tree on each coordinate.
- Space $O(n \log^d n) \to O(n \log^{d-1} n)$.
- Query $O(\log^d n) \to O(\log^{d-1} n)$.

For what value of d is this useful?
For $d = \log n / \log \log n$ then $\log^d n = n$

$\log^d n = n$
$\log(\log^d n) = \log n$
$d \log \log n = \log n$
$d = \log n / \log \log n$

Balls and Halfspaces

if $x = NN_X(q)$, the need to verify that the interior of ball $B_r(q)$ centered at q with radius $r = \|q - x\|$ is empty.
So *Ball Range Emptiness Query.*

In low dimensions, balls are harder than rectangles, since cannot use orthogonal decomposition.

In high-dimensions, balls are essentially halfspaces.
Assuming we are in regime where d vs. $d + 1$ does not matter much.

Let $X \subset \mathbb{R}^d$, and consider ball B (can be any radius, center).
Map to $X' \subset \mathbb{R}^{d+1}$ so $x = (x_1, x_2, ..., x_d) \in X$ then $x' = (x_1, x_2, ..., x_d, x_1^2 + x_2^2 + ... + x_d^2) \in \mathbb{R}^{d+1}$.
Now there exists a halfspace $H_B \subset \mathbb{R}^{d+1}$ that contains x' if and only if B contains x.

Moreover, consider $X \subset \mathbb{R}^d$, and consider halfspace $H \subset \mathbb{R}^d$.
For subset $S = X \cap H$, there exists a ball $B \subset \mathbb{R}^d$ so that $S = X \cap B$
–> start with a small ball b with boundary tangent to boundary of H. Keep the point boundaries touch fixed, and move center of b away from H.

Halfspace Range Queries

For any binary split, a halfspace query may need to recurse on both sides!
How can we get efficient $o(n)$ time queries?

Willard [1982]: Partition Trees for d=2

- Split with 2 lines, into 4 regions (each approximately $n/4$ points)
- each query H intersects at most 3 regions (other 1 region ia either *all in*, or *all out*).
- recurse on 3/4, so on roughly $(3/4)n$ points.
- query time $O(n^{\log_4 3}) \approx O(n^{0.792})$.
- and slight improvements with more cuts

Matousek, Chazelle [1989 - 1992]: optimal partition trees for general d

- query time $\tilde{O}(n^{1-1/\lfloor d/2 \rfloor})$
- space $O(n)$

So for balls in $\mathbb{R}^d$, invoke above for halfspaces in $\mathbb{R}^{d+1}$

- query time $\tilde{O}(n^{1-1/\lfloor d+1/2 \rfloor})$
- space $O(n)$

kd Trees

Hierarchical range and NN searching, useful in practice.
Each node splits subtree's data in half along one coordinate at median. Cycles through coordinates.
Height $\log_2 n$; space $O(n)$; but queries more complicated.

NN queries

- root to leaf in tree on query q, based on node-splits. Finds potential p = NN(q)
- backtracks, pruning subtrees which cannot improve upon p
- worst case $O(n)$ time, but fast in practice.
- allow (1+eps)-approximation (search to stop *much!* earlier)

Data-adaptive splits

Instead of splitting on coordinates (which is fast), adapt to data. e.g.:

- 2-means clustering
- run PCA, split along top PC axis
- split in random direction (among pairs of data points)
- Ball tree: pick any data point x, find median distance m to x, left tree is all points within m of x, and right tree those further.

With these ideas, often extends to $d \approx 100$ on "real" data

Space Filing Curves

Create 1-dimensional ordering along points in d=2,3,… dimensions.

To find nearest neighbor, use 1-dimensional techniques on this ordering $O(\log n)$ time.
Not always works, so repeat 3-7 times on randomly shifted curve structures…

L3: Curse of Dimensionality: Convex Hulls

Given a set of points $X \subset \mathbb{R}^d$, the *convex hull* $CH(X)$ is

- the smallest convex set which contains X
- if you put nails in a board, and surround with rubber band – it is inside the rubber band
- line segment between any pair of points is in $CH(X)$. Recursively, line segments between points on these line segments in $CH(X)$. In d dimensions, recurse up to d times…
- point $p \in CH(X)$ if there exists $\alpha \in [0,1]^d$ so $\alpha_j > 0$ and $\sum_j \alpha_j = 1$; then $p = \sum_j \alpha_j x_j$ (where $X = \{x_j\}_j$)
- point $p \notin CH(X)$ if and only if there exists a hyperplane H which can separate p from all points in X.
- the intersection of all balls (and hence all halfspaces) which contain X.

The convex hull $CH(X)$ is the most spatially compact way to represent X
… and intimately tied to *linear classification*.

Convex Hull for $d = 2, 3$

For $d = 2$, store sequence of edges connecting pairs of vertices as one "walks" around boundary.
Takes $O(n \log n)$ time, $O(n)$ space.

For $d = 3$, it is more complicated. Need to store edges, and also *faces*:

- 2-dimensional objects with edges on boundary.
- in general position, these are filled triangles.

How hard, big is it to construct?

- Storing all points, edges, faces is $O(n)$ space.
  Planar graph (wrapped on sphere)
- Can build in $O(n \log n)$ time

Convex Hull for High Dimensions

How do we store/represet $CH(X)$ in high-dimensions?

- just points?
- just faces?

- all j-dimensional (for $j \in [0, 1, \ldots, d-1]$) facets?

Moment curve: $x(t) = (t^1, t^2, \ldots, t^d) \in \mathbb{R}^d$ for $t \in \mathbb{R}$.
For n (lets say $n > 2d$) points X on moment curve, they form a *cyclic polytope*, for which all subsets of size $j+1$ for $j < \lfloor d/2 \rfloor$ define a j-dimensional face on the convex hull.
For $j = \lfloor d/2 \rfloor - 1$, there are $\binom{n}{j+1} = \Theta(n^{\lfloor d/2 \rfloor})$ such j-dimensional facets.
Can be shown the number of j-dimensional facets for $j > \lfloor d/2 \rfloor$ is also $\Theta(n^{\lfloor d/2 \rfloor})$.
Including $\Theta(n^{\lfloor d/2 \rfloor})$ of the faces (the $(d-1)$-dimensional facets).
Upper Bound Theorem: This is the most possible.

Approximate Convex Hulls

Can we break exponential dependence on d is we approximate?

Most common definition: *ε-kernel coreset.*
A subset $S \subset X$ approximates all directions within $(1+\varepsilon)$.

Unit direction $u \in \mathbb{R}^d$ such that $\|u\| = 1$. or say $u \in \mathbb{S}^{d-1}$
Width in direction u is $wid_u(X) = \max_{x \in X} \langle x, u \rangle - \min_{x \in X} \langle x, u \rangle$.
Goal for S is for **all** directions u that

$$wid_u(X) - wid_u(S) \leq \varepsilon \cdot wid_u(X)$$

Hardest case is ball B.
In $d = 2$ it requires $\Omega(1/\sqrt{\varepsilon})$ points
$\varepsilon = 0.01 = 1/100$ (so 1% error) needs only about 10 points.

In high dimensions it requires $\Omega(1/\varepsilon^{\lfloor d/2 \rfloor})$ points.
Exist algorithm to compute ε-kernel of size $O(1/\varepsilon^{\lfloor d/2 \rfloor})$ points.

$(1+\varepsilon)$-width approximation by faces also requires $\Omega(1/\varepsilon^{\lfloor d/2 \rfloor})$ faces.

John Ellipsoid Approximation

An *ellipsoid* is a ball in $\mathbb{R}^d$ after any affine transformation.
Let $A \in \mathbb{R}^{d \times d}$ full-rank matrix.
An affine transformation of $x \in \mathbb{R}^d$ by A is simply $x' = Ax + t$ for $t \in \mathbb{R}^d$.
Thus an ellipse $E_{A,t}$ is the set of points $x' \in \mathbb{R}^d$ such that $\{x' = Ax + t \mid x \in B\}$ where B is a unit ball.

An ellipsoid is not any "round" shape.
It is controlled by an orthonormal basis $U = [u_1, u_2, \ldots, u_d]$ where each $\|u_j\| = 1$ and each pair $u_j, u_{j'}$ are orthogonal so $\langle u_j, u_{j'} \rangle = 0$.
Then each j is associated with a scaling λ_j.
(indeed, these are the eigenvectors u_j and values λ_j of A)
For an ellipsoid E with those basis and scaling, and center t, then a point $x \in \mathbb{R}^d$ is in E if

$$\sum_{j=1}^{d} \langle u_j, x - t \rangle^2 / \lambda_j^2 \leq 1.$$

How well can we approximate a convex set K (e.g., $K = CH(X)$) with an ellipsoid?

Lowner-John's Ellipsoid theorem says for any convex set $K \subset \mathbb{R}^d$ there exists an ellipsoid E so

$$(1/d)E \subset K \subset E$$

This E is the minimum volume ellipsoid which circumscribes K. This is the best possible dilation ratio.

Making $CH(X)$ Fat

Compute Lowner-John Ellipsoid $E_{A,t}$ for $CH(X)$.
Invert data X by A^{-1}; that is

$$X' = \{x' = A^{-1}x \mid x \in X\}$$

Now X' is *d-fat*, this means that

$$\frac{\max_{u \in \mathbb{S}^{d-1}} wid_u(X')}{\min_{u \in \mathbb{S}^{d-1}} wid_u(X')} \leq d$$

L4: Hardness of Estimation: Learning Theory

Data X, y of size n so each $(x_i, y_i) \in \mathbf{R}^d \times \{-1, +1\}$

$X \sim P$ for probability distribution on $\mathbb{R}^d$ and $y \sim \sigma$.
$X, y \sim P, \sigma$ a *joint* distribution, y_i depends on x_i (may be functional relation).

Goal is to learn $f : \mathbb{R}^d \to \mathbb{R}$ so if $f(x) > 0$, predict $+1$, and otherwise predict -1.
Want $y \approx \mathsf{sign}(f(X))$

Function class F (e.g., family of all halfspaces, or neural net with fixed architecture).
$h_{u,b} \in H$ (halfspaces), we can define $h_{u,b}(x) =< x, u > -b$

Goal 1: Find $f \in F$ on (X, y) so that

$$err(f, X, y) = \frac{1}{n} \sum_i (\mathsf{sign}(f(x_i)) \neq y_i)$$

is as small as possible.

But this only works with existing data (X, y).
The real goal is to understand P, σ, and potential new data drawn again from there.

$$err(f, P, \sigma) = E_{(x,y)\sim(P,\sigma)}(\mathsf{sign}(f(x_i)) \neq y_i)$$

Sample Complexity for Learning Bounds

Separable Data:
Assume first there exists some $h \in H$ so that $err(h, P, \sigma) = 0$.
Let $h \in H$ satisfies $err(h, X, y) = 0$.

Let $n = \Omega((\nu/\varepsilon)\log(\nu/\varepsilon\delta))$ for $\varepsilon, \delta \in (0, 1)$; we will explain ν later.
Then with probability at least $1 - \delta$, $err(h, P, \sigma) \leq \varepsilon$.

Non-Separable Data:
Let $h \in H$ satisfies $err(h, X, y) = \gamma$.

Let $n = \Omega((1/\varepsilon^2)(\nu + \log(1/\delta))$ for $\varepsilon, \delta \in (0, 1)$
Then with probability at least $1 - \delta$, $err(h, P, \sigma) \leq \gamma + \varepsilon$.

VC (Vapnik-Chervonenkis) Dimension

Let (X, F) be a *range space*, where (in this class) $X \subset \mathbb{R}^d$, and F provides a family of subsets of X (e.g., H, those defined by inclusion in a halfspace).

We say a range space (Y, F) for $Y \subset X$, can be *shattered* if all subsets of Y exist.
That is, for each $Z \subset Y$, there exists some "shape" $f \in F$ so the $f \cap Y = Z$.

Any subset of size 3 points in the $\mathbb{R}^2$ can be shattered by halfspaces (unless they are co-linear). But no set of 4 points can be shattered by halfspaces.

The **VC-dimension** of a range space (X, F) is the size of the largest subset $Y \subset X$ which can be shattered. For *halfspaces* in $\mathbb{R}^d$, the VC-dimension is $d + 1$.

More generally, let (X, F) for $X \subset \mathbb{R}^d$ and F be a family of functions (e.g., a neural net) which can be evaluated with t *simple operations* with the follow structure:

- +, -, x, /
- jumps using <, <=, >, >=, =, != on real numbers
- return 0, 1

Then the VC-dimension of (X, F) is at most $4d(t + 2)$.

If you also allow $q > 1$ exponential $\exp(\cdot)$ operations in functions in F
then the VC-dimension of (X, F) is $O(d(q^2 + q(t + \log(dq)))$

Take-away: the number of samples needed to generalize grows linearly (if not quadratically) with dimension d.

Which Function Class?

So are simpler (lower VC-dim) function classes better?

If we only use halfspaces, on the first 3 coordinates, then we get better generalization with same samples, right?
Then only need $n = O((1/\varepsilon^2)(4 + \log(1/\delta))$.
–> But $\gamma = err(h, X, y)$ is larger!

Let the model error

$$\gamma_F = \min_{f \in F} err(f, P, \sigma)$$

be the minimal amount of error from a function class F. Simple classifiers tend to have larger γ_F.
Complicated (high-dimensional) classifiers have smaller γ_F.

- If $d = n$, then for halfspaces $\gamma_H = 0$. Since we can shatter X.
- In general for (X, F) with VC-dimension ν if $n = \nu$, we might be able to shatter X, in which case $\gamma_F = 0$.
- For H_p described as polynomials of sufficiently degree p, it can approximate any function f. But VC dimension $O(d^p)$.
- Even 2-layer neural networks with sufficiently wide second layer, can also approximate any function.

But then if $n \approx \nu$, it does not satisfy $n = \Omega(\nu/\varepsilon^2)$, so do not get sample complexity bound, and $|err(f, X, y) - err(f, P, \sigma)|$ can be large – it is not controlled.

L5: Hardness of Estimation: Mean Estimation

Review Learning Theory

VC-Dimension: $\nu \approx$ number of parameters in model class F
"dimension for model F"

Labeled data (X, y) size n

Sample Error (training error)

- separable: $n \approx (\nu/\varepsilon)\log(\nu/\varepsilon)$
  $error(X) \leq \varepsilon \approx (\nu/n)\log(\nu/n)$
- non-separable: $n \approx \nu/\varepsilon^2$
  $error(X) \leq \varepsilon \approx \nu/\sqrt{n}$

increases with ν increasing

Model Error

- $\gamma_F(X) = \min_{f \in F} error(f, X, y)$

decreases with ν increasing

Total Error (test error) = Sample Error + Model Error

Parameter Estimation

(Bayesian-y view)

Assume data $X \sim g(\alpha)$ for some model g with parameters $\alpha \in \mathbb{R}^d$.

Distributional so g provides probability distribution
e.g., each $x \in X$ from "perfect" $h(\alpha) + Noise$, where $Noise$ is random, independent of h.

The simplest case is

- $h(\alpha) = \alpha$
- $Noise = \mathcal{G}_d(0, I)$ (d-dimensional Gaussian/Normal noise)
- Goal: from $X \sim g$, recover α

Alternatively, if

- $h(\alpha) = 0$
- $Noise = \mathcal{G}_d(\alpha, I)$
- Goal: from $X \sim g$, recover α
  the same problem, but now clear we are aiming to recover the **mean**
  which is $E_{X \sim Noise}[X] = \alpha$

Chebyshev Inequality (Law of Large Numbers)
For n iid RVs $X_1, X_2, \ldots, X_n$ with $Var[X_i] = \sigma^2$

$$Pr[|\bar{x} - E[X_j]| \geq \eta] \leq \frac{\sigma^2}{n\eta^2}$$

Note that simple Chebyshev we have
$Pr[|X_j - E[X_j]| \geq \eta] \leq \frac{\sigma^2}{\eta^2}$,
but for $Var[\bar{x}] = Var[X_j]/n = \sigma^2/n$ so
$Pr[|X - E[X]| \geq \eta] \leq \frac{Var[\bar{x}]}{\eta^2} = \frac{\sigma^2}{n\eta^2}$

Chernoff-Hoeffding Inequality (simplified as in Azuma)
For n iid RVs $X_1, X_2, \ldots X_n$ with $X_i \in [0, \Delta]$

$$Pr[|X - E[X]| \geq \eta] \leq 2\exp(-\frac{2\eta^2 n}{\Delta^2})$$

Thus as n increases our bound on the error η from an expected value decreases with $1/\sqrt{n}$.
Fix either $Pr[...] = \delta$, and solve for η as a function of n.

Trouble with High-Dimensional Mean Estimation

For each $x \in X \sim G_d(\alpha, I)$, then

$$E[\|x - \alpha\|^2] = \sum_{j=1}^{d} E[(x_j - \alpha_j)^2] = \sum_{j=1}^{d} E[(x_j - E[x_j])^2] = \sum_{j=1}^{d} Var[x_j] = d$$

Then for $\bar{x} = \frac{1}{n}\sum_{j=1}^{d} x_i$

$$E[\|\bar{x} - \alpha\|^2] = \sum_{j=1}^{d} Var[\bar{x}_j] = d/n$$

We can also analyze the convergence

$$Pr[\|\bar{x} - \alpha\| > \eta] \leq d/(n\eta^2)$$

To show this we will use the **Union Bound** that if there are k events $E_1, \ldots, E_k$, then the probability all events are true $Pr[E_1 \& \ldots \& E_k] \leq 1 - \sum_{j=1}^{k} Pr[E_j = FALSE]$.

$Pr[(\bar{x}_j - \alpha_j)^2 > (\eta')^2] \leq \frac{Var[X_j]}{n(\eta')^2} = \frac{1}{n(\eta')^2} = \delta'$
So applying the union bound on d coordinates, with $\delta = \delta' \cdot d$,
Setting $\eta = \eta' \cdot \sqrt{d}$ so $\eta^2 = (\eta')^2 d$, we have
$Pr[\|\bar{x} - \alpha\|^2 > \eta^2] \leq \delta$
Solving for $\eta = \eta' \cdot \sqrt{d}$ and $\eta' = \frac{1}{\sqrt{n\delta}}$, so $\eta = \frac{\sqrt{d}}{\sqrt{n\delta}}$.
Or $n = d/(\eta^2\delta)$

Two Mean Example:
Consider two mean estimations in $\mathbb{R}^d$
$X_1 \sim \mathcal{G}_d(\alpha, I)$ and $X_2 \sim \mathcal{G}_d(\alpha', I)$
where we are promised that $|\alpha_1 - \alpha_1'| = 2$ and $\alpha_j = \alpha_j'$ for $j > 1$.

As n increases, we can get estimates of α_1 and α_1' to concentrate to values η much less than 2.
But each $x \in X_1$ has $E[\|x - \alpha\|^2] = d$. and $E[\|\bar{x} - \alpha\|^2] = d/n$
Moreover $Pr[\|\bar{x} - \alpha\| \geq \sqrt{d/n\delta}] \leq \delta$.

Let $\bar{x}_1 = \frac{1}{|X_1|}\sum_{x \in X_1} x$ and similar for $\bar{x}_2$.
$E[\|\bar{x}_1 - \alpha\|^2] = d/|X_1|$.
To get $\|\bar{x}_i - \alpha\| \leq \eta$ we need about d/η^2 samples.

Outliers:

One outlier can significantly affect sample mean $\bar{x}$
In $d = 1$, the median is a good estimate for α and resistant to outliers.
What is analog in high dimensions?
- coordinate-wise median: $v = (v_1, ..., v_d)$ has v_j as median of j th coordinates.
- L1 median (geometric median): v minimize sum of distances to $x \in X$ - centerpoint: v so no halfspace containing v contains more than $|X|/(d+1)$ points - Turkey median: $v = \arg\max_{v \in \mathbb{R}^d} \min_{h \in H, v \in h} \frac{|X \cap h|}{|X|}$

Turkey median works well (uses d/η^2 samples, even with outliers), but best algorithms take about $|X|^{d-1}$ time to compute.

Robust High-Dimensional Mean Estimation

With $n = d/\eta^2$ samples, one can use new approaches - that allow for η-fraction of outliers, rest from $\mathcal{G}_d(0, I)$ - in $\mathsf{poly}(nd/\eta)$ time - find a point $\hat{v} \in \mathbb{R}^d$ so $\|v - \hat{v}\| \leq \tilde{O}(\eta)$

Careful Pruning

Start with Sample Mean $\bar{x}$, and center data.
Compute top principal vector u, project data along u : $X_u = \{x_u = \langle x, u \rangle \mid x \in X\}$.
Compare CDF to that of 1-d normal. Prune extreme points that are too far off. [*]
Repeat until no sign of outliers.

[*] Vershynin (2011): $\Sigma \in \mathbb{R}^{n \times d}$ so each entry iid $\Sigma_{i,j} \sim \mathcal{N}(0,1)$.
With probability at least $1 - 2\exp(-t^2/2)$

$$\sqrt{n} - \sqrt{d} - t \leq s_{\min}(\Sigma) \leq \|\Sigma\|_2 \leq \sqrt{n} + \sqrt{d} + t$$

Median of Means

Decompose X into k components randomly (e.g., for $k = 3, 5$, or 7)
Compute mean of each $\bar{x}_1, \ldots, \bar{x}_k$.
Return coordinate-wise median of set $\{\bar{x}_1, \ldots, \bar{x}_k\}$.

Works quite well if $|X|$ is large enough to split.

2. Build High-Dimensional Data

- Lifting Approach
  - (low) -> (high)
  - nonlinear
- Metric Embeddings
  - l_2 distance, l_1, l_∞
- Features
- MasRod Word
- Constostive Learning

3. Dimensionality Reduction

- PCA, MDS, t-SNE
- Random Projections

4. Similarity Search

- Data Query
- Graph-Based NN Query

5. Optimization

- Classification

6. Coresets

7. Sampling in Polytopes