Outline

- L0 estimation
- Projection onto Complicated Objects and Gaussian Mean Width
- Compressed Sensing

Estimating the Number of Non-Zero Entries

- $|x|_0 = |\{i \text{ such that } x_i \neq 0\}|$

- How can we output a number Z with $(1-\epsilon)Z \leq |x|_0 \leq (1+\epsilon)Z$ with prob. 9/10?
  - Want $O((\log n)/\epsilon^2)$ bits of space

- Suppose $|x|_0 = O(\frac{1}{\epsilon^2})$. What can we do in this case?

- Use our algorithm for recovering a k-sparse vector from last time, $k = O\left(\frac{1}{\epsilon^2}\right)$
  - What is another way?

- But what if $|x|_0 \gg \frac{1}{\epsilon^2}$?

Estimating the Number of Non-Zero Entries

- Suppose we somehow had an estimate Z with $Z \leq |x|_0 \leq 2Z$, what could we do?
- Independently sample each coordinate i with probability $p = 100/(Z\,\epsilon^2)$
- Let Y_i be an indicator random variable if coordinate i is sampled
- Let y be the vector restricted to coordinates i for which $Y_i = 1$
- $E[|y|_0] = \sum_{i \text{ such that } x_i \neq 0} E[Y_i] = p|x|_0 \geq \frac{100}{\epsilon^2}$
- $Var[|y|_0] = \sum_{i \text{ such that } x_i \neq 0} Var[Y_i] \leq \frac{200}{\epsilon^2}$
- $\Pr\left[\left||y|_0 - E[|y|_0]\right| > \frac{100}{\epsilon}\right] \leq \frac{Var[|y|_0]\epsilon^2}{100^2} \leq \frac{1}{50}$
- Use sparse recovery or CountSketch to compute $|y|_0$ exactly
- Output $\frac{|y|_0}{p}$

But we don't know Z…

Estimating the Number of Non-Zero Entries

- Guess Z in powers of 2
- Since $0 \leq |x|_0 \leq n$, there are $O(\log n)$ guesses
- The i-th guess Z = 2^i corresponds to sampling each coordinate with probability $p = \min(1, \frac{100}{2^i \epsilon^2})$
- Sample the coordinates as nested subsets $[n] = S_0 \supseteq S_1 \supseteq S_2 \supseteq \cdots S_{\log n}$
- Run previous algorithm for each guess
- One of our guesses Z satisfies $Z \leq |x|_0 \leq 2Z$ and we should use that guess

- *But how do we know which one?*

Estimating the Number of Non-Zero Entries

- Use the largest guess $Z = 2^i$ for which $\frac{400}{\epsilon^2} \le |y|_0 \le \frac{3200}{\epsilon^2}$
- If $\frac{800}{\epsilon^2} \le E[|y|_0] \le \frac{1600}{\epsilon^2}$, then $\frac{400}{\epsilon^2} \le |y|_0 \le \frac{3200}{\epsilon^2}$ with probability at least 49/50
- If $\frac{100}{\epsilon^2} \le E[|y|_0] \le \frac{200}{\epsilon^2}$, then $|y|_0 < \frac{400}{\epsilon^2}$ with probability at least 49/50
- So with probability 48/50, we choose an i for which $\frac{200}{\epsilon^2} \le E[|y|_0] \le \frac{1600}{\epsilon^2}$
- There are only 4 such indices i, and all 4 of them satisfy $|y|_0 = (1 \pm \epsilon)E[|y|_0]$ simultaneously with probability 1-4/50. So doesn't matter which i we choose

- Overall, our success probability is 1-2/50-4/50 > 4/5

What is Our Overall Space Complexity?

- If we use our k-sparse recovery algorithm for $k = O\left(\frac{1}{\epsilon^2}\right)$, then it takes $O(\frac{\log n}{\epsilon^2})$ bits of space in each of log n levels, so $O(\frac{\log^2 n}{\epsilon^2})$ total bits of space ignoring random bits
  - How much randomness do we need?
  - Pairwise independence is enough for Chebyshev's inequality
  - Implement nested sampling by choosing a hash function $h: [n] \rightarrow [n]$, checking if first i bits of h(j) = 0
  - O(log n) bits of space for the randomness

- Can improve to $O\left(\frac{\log \quad (\log\left(\frac{1}{\epsilon}\right)+\log\log n)}{\epsilon^2}\right)$ bits. How?

- Just need to know number of non-zero counters, so reduce counters from log n bits to $O(\log\left(\frac{1}{\epsilon}\right) + \log\log n)$ to bits

Reducing Counter Size

- In sampling levels that we care about, we have $O\left(\frac{1}{\epsilon^2}\right)$ counters, each of O(log n) bits

- At most $O\left(\frac{\log n}{\epsilon^2}\right)$ prime numbers dividing any of these counters

- Choose a random prime q = $O\left(\frac{\log n \log\log n}{\epsilon^2}\right)$. Unlikely that q divides any counters

- Just maintain our sparse recovery structure mod q, so $O\left(\frac{(\log\log n + \log\left(\frac{1}{\epsilon}\right)}{\epsilon^2}\right)$ bits per each of O(log n) sparse recovery instances

Outline

- L0 estimation
- Projection onto Complicated Objects and Gaussian Mean Width
- Compressed Sensing

Projection onto other Objects

- Least squares regression finds the closest point y in a subspace K to a given point b

- Given a (possibly infinite) set of points K, and a point b, compute $\min_{y \in K} |y - b|$
  - All norms are Euclidean norms

- Let S be a sketching matrix, we want that if y' = $\operatorname{argmin}_{y \in K} |Sy - Sb|$, then

$$|y' - b| \leq (1 + \epsilon) \min_{y \in K} |y - b|$$

- More generally, want to preserve distances of all vectors in a set K, that is, $|S(y-y')| = (1 \pm \epsilon)|y - y'|$ for all $y, y' \in K$

What properties of K determine the dimension and sparsity of S?

Example: Preserving Distances in a Set

- More generally, want to preserve distances of all vectors in a set K, that is, |S(y-y')| = $(1 \pm \epsilon)|y - y'|$ for all y, y' ∈ K

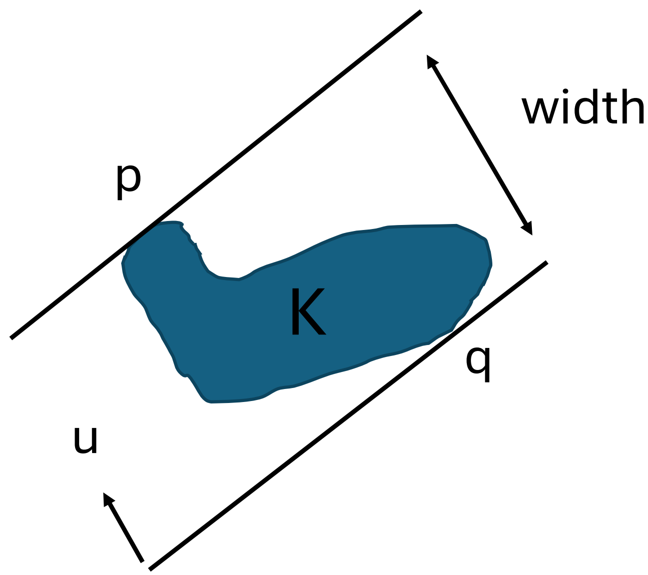
- What is the dimension of S needed if K is:

  - n arbitrary points in R^d?

  - n arbitrary points on a line in R^d?

Spherical Mean Width

- Let K be a bounded subset in R^{n}
- Consider the width in direction u for a unit vector u:



- Width in direction u = $\sup_{\mathrm{p,q\ in\ K}} < \mathrm{u}, \mathrm{p} - \mathrm{q} >$
- Spherical mean width = $\mathrm{E_u}[\sup_{\mathrm{p,q\ in\ K}} < \mathrm{u}, \mathrm{p} - \mathrm{q} >]$

Gaussian Mean Width

- Let $g \sim N(0, I_n)$ be an i.i.d. Gaussian vector
- Gaussian mean width g(K) = $E_g[\sup_{p,q \text{ in } K} < g, p - q >]$

  $= \Theta(n^{.5}) \cdot$ spherical mean width
- Examples
  - K = S^{n-1}
    - $\Theta(n^{.5})$
  - K = set of unit vectors in a d-dimensional subspace of R^n
    - $\Theta(d^{.5})$
  - K = t arbitrary unit vectors in R^n
    - $\Theta(\log^{.5} t)$

Gaussian Mean Width of t Arbitrary Unit Vectors

- Let $u^1, \dots, u^t$ be t arbitrary unit vectors in R^n
- Let g in R^n have i.i.d. N(0,1) entries
- Define random variables $Z_j = < u^j, g >$ which are N(0,1) random variables
- Want to bound $E_g[\max_j Z_j]$
- Fact: for an N(0,1) random variable W, $E\left[e^{\lambda W}\right] = e^{\lambda^2/2}$
- For any $\lambda > 0$, $E\left[e^{\lambda \max_j Z_j}\right] \leq \sum_j E\left[e^{\lambda Z_j}\right] \leq t\, e^{\lambda^2/2}$
- For all $\lambda > 0$, $E_g[\max_j Z_j] \leq \left(\frac{1}{\lambda}\right) \log E[e^{\lambda \max_j Z_j}] \leq (\frac{\log t}{\lambda} + \frac{\lambda}{2}) \leq 2\sqrt{\log t}$

Sketching Bounds

- [Gordon] Let K be a subset of S^{n-1}. A random Gaussian matrix S with $g(K)^2/\epsilon^2$ rows satisfies

  $$|S(y-y')|^2 = (1 \pm \epsilon)|y-y'|^2 \text{ for all y, y' in K}$$

- What about sparse sketching matrices S?

- [Bourgain, Dirksen, Nelson] S can have m = $g(K)^2 \text{poly}(\log n)/\epsilon^2$ rows and $s = \text{poly}(\log n)/\epsilon^2$ non- zeros per column if m and s satisfy a condition related to higher moments of $\sup_{p,q} < g, p-q >$
  - Applied to finite and infinite unions of subspaces

Outline

- L0 estimation
- Projection onto Complicated Objects and Gaussian Mean Width
- Compressed Sensing

Compressed Sensing

- We take random “linear measurements” of an n-dimensional vector x
- In our language, we choose a random r x n sketching matrix S and observe $S \cdot x$
- Output a vector x’ with $|x - x'|_p = D \cdot \min_{k-\text{sparse } z} |x - z|_q$, where D is the distortion (the ℓ_p/ℓ_q-guarantee)
- Let x_k be the best k-sparse approximation to x, i.e., the largest k coordinates in magnitude
- Randomized (“for-each”) scheme versus deterministic (“for-all”) scheme
- CountSketch is a randomized scheme achieving ℓ_2/ℓ_2 w.h.p.

$$|x - x'|_2 = O(1) \cdot |x - x_k|_2$$

CountSketch for Compressed Sensing

- CountSketch had O(log n) repetitions of hashing into O(k) buckets
- S is a random linear map S with O(k log n) rows
- For an n-dimensional vector x, estimate every x_i up to additive error $\frac{|x-x_k|_2}{\sqrt{k}}$
- Output a 2k-sparse x' consisting of the top 2k estimates given by CountSketch
- Say coordinate i is heavy if $|x_i| \geq |x - x_k|_2/\sqrt{k}$
  - How many heavy coordinates can there be?
- Say a coordinate i is super-heavy if $|x_i| \geq 3|x - x_k|_2/\sqrt{k}$
  - Claim: the set T of super-heavy coordinates is in the support of x'
- $|x - x'|_2 \leq |(x - x')_T|_2 + \left|(x - x')_{[n]\setminus T}\right|_2$

$$\leq \sqrt{|T|} \cdot \frac{|x-x_k|_2}{\sqrt{k}} + |(x - x_k)_{[n]\setminus T}|_2 + \left|(x_k - x')_{[n]\setminus T}\right|_2 = O(|x - x_k|_2)$$

No Deterministic Algorithm Achieves ℓ_2/ℓ_2

- Recall ℓ_2/ℓ_2: output x' with $|x - x'|_2 = O(1) \cdot |x - x_k|_2$
- Consider k = 1
- Suppose S is a deterministic sketching matrix with r = o(n) rows
- Suffices to show there is a vector x in kernel(S) with $|x|_\infty \geq C|x - x_1|_2$ for any constant C > 0
- W.l.o.g., can assume S has orthonormal rows
- $\sum_i |Se_i|_2^2 = r$, so there exists an i with $|Se_i|_2^2 \leq \frac{r}{n}$
- Let $x = e_i - S^T Se_i$, so x is in kernel(S)
- But $|x|_\infty^2 \geq |x_i|^2 = \left(e_i^T e_i - e_i^T S^T Se_i\right)^2 \geq \left(1 - \frac{r}{n}\right)^2$, while
- $|x - x_1|_2 \leq |x - e_i|_2 = \left|S^T Se_i\right|_2 = |Se_i|_2 \leq \sqrt{\frac{r}{n}} = o(1)$

Deterministic Algorithms Achieve ℓ_2/ℓ_1

- ℓ_2/ℓ_1: output x' with $|x - x'|_2 = O(1/k^{.5}) \cdot |x - x_k|_1$
- S has the (ϵ, k)-restricted isometry property (RIP) if for all k-sparse vectors x,

$$(1 - \epsilon)|x|_2^2 \le |Sx|_2^2 \le (1 + \epsilon)|x|_2^2$$

- What are some matrices S with O(k log(n/k)) rows that have the (ϵ, k)-RIP property for constant ϵ?
- Deterministic, but not explicit!
- Major open question: explicit matrix with (ϵ, k)-RIP with $o(k^2)$ rows
- Bourgain et al.: can get $k^{2-\gamma}$ rows for a constant $\gamma > 0$ and $k \approx n^{.5}$

Deterministic Algorithms Achieve ℓ_2/ℓ_1

- If S has the (ϵ, k)-RIP then one can efficiently output an x' for which
$$|x - x'|_2 = O\left(1/k^{.5}\right) \cdot |x - x_k|_1$$

- In fact, can just solve a linear program!

$$\min_{z \in R^n} |z|_1$$
$$\text{s.t. } Sz = Sx$$

- If x' is the solution, then $|x - x'|_2 \leq O\left(\frac{1}{k^{.5}}\right) |x - x_k|_1$

- Proof uses (ϵ, k)-RIP and elementary norm manipulations