Algorithms for Big Data Notes

Instructor: David P. Woodruff
Notes compiled by: Lichen Zhang
Carnegie Mellon University
15-859 Algorithms for Big Data

Fall 2020

Contents

1 Least Squares Regression and Subspace Embeddings **4**
1.1 Regression analysis . . . 4
1.2 The sketch-and-solve paradigm . . . 7
1.3 Subspace embeddings . . . 7
1.4 Back to regression . . . 12

2 The Subsampled Randomized Hadamard Transform and Approximate Matrix Product **14**
2.1 The Hadamard matrix . . . 14
2.2 The Subsampled Randomized Hadamard Transform . . . 15
2.3 The flattening lemma . . . 16
2.4 The matrix Chernoff bound . . . 17
2.5 Wrapping up the SRHT . . . 20
2.6 CountSketch . . . 20

3 CountSketch, Affine Embeddings, and Low Rank Approximation **23**
3.1 From vectors to matrices . . . 23
3.2 CountSketch satisfies the JL moment property . . . 25
3.3 Affine embeddings . . . 28
3.4 Low rank approximation . . . 32

4 Fast Low Rank Approximation, High Precision Regression, and Leverage Score Sampling **35**
4.1 Recap: low rank approximation via sketching . . . 35
4.2 Approximate projection via affine embeddings . . . 35
4.3 High precision regression . . . 37
4.4 Leverage score sampling . . . 41

5 Distributed Low Rank Approximation and Coresets **45**
5.1 The distributed setting . . . 45
5.2 Constructing a coreset [FSS20] . . . 47
5.3 Unions of coresets . . . 49

5.4 The [FSS20] row partition protocol . . . 50
5.5 The [KVW14] arbitrary partition protocol . . . 50
5.6 The [BWZ16] protocol . . . 51

6 Distributed Low Rank Approximation Protocols and ℓ_1 Regression 53
6.1 The [BWZ16] protocol . . . 53
6.2 ℓ_1 regression . . . 55
6.3 Well-conditioned bases . . . 56
6.4 A net for the ℓ_1 ball . . . 57
6.5 A rough algorithm . . . 58
6.6 The sketching theorem and its applications . . . 59

7 Sketching for ℓ_1 Regression and the Data Stream Model 60
7.1 Recap: sketching for ℓ_1 regression . . . 60
7.2 Proof of the sketching theorem . . . 61
7.3 Faster sketches for ℓ_1 regression . . . 64
7.4 A fun fact about Cauchy random variables . . . 65
7.5 The data stream model . . . 65

8 Estimating Norms in a Data Stream 69
8.1 The streaming model and norm estimation . . . 69
8.2 Estimating the Euclidean norm . . . 69
8.3 Estimating the ℓ_1 norm . . . 70
8.4 Estimating ℓ_p norms for $0 < p < 2$ . . . 72
8.5 Estimating ℓ_p norms for $p > 2$ . . . 72
8.6 Analysis of the estimator for $p > 2$ . . . 74

9 Heavy Hitters 79
9.1 Recap: norm estimation for $p > 2$ . . . 79
9.2 Heavy hitters and their guarantees . . . 79
9.3 Intuition: finding a single heavy item . . . 80
9.4 CountSketch achieves the ℓ_2-guarantee . . . 80
9.5 Deterministic ℓ_1 heavy hitters . . . 82
9.6 Finding the heavy hitters quickly . . . 83

10 Estimating the Number of Non-Zeros, and Information Theory 85
10.1 Estimating the number of non-zero entries . . . 85
10.2 Information theory . . . 87

11 Streaming Lower Bounds, Gaussian Width, and Compressed Sensing 94
11.1 The one-way communication complexity of Index . . . 94
11.2 Lower bounds for streaming algorithms by reduction . . . 95
11.3 An $\Omega(\log n)$ bit lower bound for estimating norms . . . 96
11.4 An $\Omega(1/\varepsilon^2)$ bit lower bound for estimating norms . . . 97
11.5 Index with public randomness, and on average . . . 97
11.6 Distributional communication complexity . . . 98
11.7 Index with low error . . . 99
11.8 Projection onto complicated objects . . . 100
11.9 Compressed sensing . . . 102

References **105**

1 Least Squares Regression and Subspace Embeddings

Lecture 1, September 10, 2020, by David P. Woodruff. Compiled from the scribe notes of Bailey Miller (both halves), and from the lecture slides.

Massive data sets are becoming ubiquitous; they show up everywhere from internet traffic logs to financial data. Classical algorithms are no longer a feasible tool for working with these data sets, as we often need running times that are nearly linear or better. Randomized approximations provide an alternative that has the efficiency we seek, at the cost of approximate results whose error is bounded with high probability.

1.1 Regression analysis

Regression is a statistical method that studies the dependencies between variables in the presence of noise. In these notes we focus on *linear regression*, which models linear dependencies between variables.

For example, we can estimate the resistance R of a circuit element using linear regression. By Ohm's law, $V = R \cdot I$, so if we measure the voltage for several given currents, we can use linear regression to find the linear function that best fits the data (Figure 1).

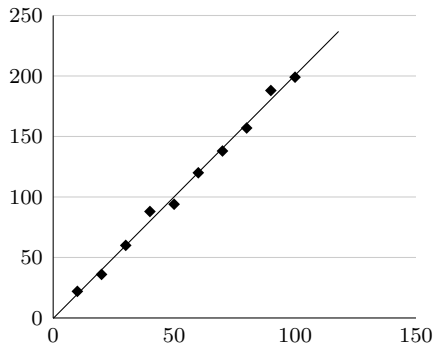


Figure 1: An example of regression studying Ohm's law: measurements and the least squares line through them.

1.1.1 The standard setting

In linear regression we assume that a *measured variable* b is the value of a linear function

$$b = x_0 + a_1 x_1 + \cdots + a_d x_d + \eta,$$

where $a_1, \ldots, a_d$ are *predictor variables*, $x_0, \ldots, x_d$ are *model parameters* that we want to learn, and η is a term that represents *noise*. We can assume $x_0 = 0$, since an intercept can be absorbed by adding a predictor that is always equal to 1.

With n observations of the measured and predictor variables, we use a matrix representation: an $n \times d$ matrix A of predictor variables (one row per observation) and a vector $b = (b_1, \ldots, b_n) \in \mathbb{R}^n$ of measured variables. Ideally we would find x with $Ax = b$. We focus on the *over-constrained* case, with many more observations than predictor variables, $n \gg d$. In this case there may be no exact solution of $Ax = b$, and instead we look for x such that Ax and b are close.

1.1.2 The least squares method

Least squares is the most common way of choosing x in the over-constrained case. It finds the x^* that minimizes the Euclidean distance between Ax and b, that is, x^* minimizes

$$\|Ax - b\|_2^2 = \sum_{i=1}^{n} (b_i - A_{i,*}x)^2,$$

where $A_{i,*}$ is the i-th row of A. Least squares has desirable statistical properties: for instance, it is the maximum likelihood estimator of the model parameters if the noise η is Gaussian and independent across observations.

Least squares also has a clean geometry. The product Ax can be written as a combination of the columns of A,

$$Ax = A_{*,1}x_1 + A_{*,2}x_2 + \cdots + A_{*,d}x_d,$$

where $A_{*,j}$ is the j-th column of A. As x ranges over $\mathbb{R}^d$, these vectors form the column span of A, a linear subspace of dimension $\operatorname{rank}(A) \le d$. So $\min_x \|Ax - b\|_2$ asks for the point of the column span of A nearest to b in the ℓ_2 norm, which is the projection of b onto the column span of A.

1.1.3 Solving least squares via the normal equations

When A has linearly independent columns, the least squares problem $\min_x \|Ax - b\|_2^2$ has the closed-form solution

$$x = (A^\top A)^{-1} A^\top b. \tag{1.1}$$

To see this, write

$$b = Ax' + b',$$

where b' is orthogonal to the column span of A. Then, by the Pythagorean theorem,

$$\|Ax - b\|_2^2 = \|A(x - x') - b'\|_2^2 = \|A(x - x')\|_2^2 + \|b'\|_2^2,$$

so the cost is minimized by $x = x'$, and x is optimal if and only if $A(x - x') = 0$. Since $A^\top b' = 0$, this is equivalent to

$$A^\top(Ax - b) = A^\top(Ax - Ax' - b') = A^\top A(x - x') = 0.$$

Note that $A^\top Az = 0$ holds exactly when $Az = 0$: if $A^\top Az = 0$, then $\|Az\|_2^2 = z^\top A^\top Az = 0$. So every optimal x satisfies the *normal equations*

$$A^\top Ax = A^\top b,$$

and when $A^\top A$ is invertible this gives (1.1).

If the columns of A are not linearly independent, then $(A^\top A)^{-1}$ does not exist, and we use the Moore–Penrose pseudoinverse instead. For this we need the singular value decomposition in the following form, in which Σ may have zero diagonal entries (unlike the thin SVD used elsewhere in these notes, where all diagonal entries of Σ are positive).

Definition 1.1 (Singular value decomposition)**.** Any matrix can be written as $A = U\Sigma V^\top$, where

- U has orthonormal columns,
- Σ is diagonal with non-increasing non-negative entries down the diagonal, and

- $V^\top$ has orthonormal rows (so V has orthonormal columns).

Definition 1.2 (Moore–Penrose pseudoinverse)**.** The pseudoinverse of $A = U\Sigma V^\top$ is $A^\dagger = V\Sigma^\dagger U^\top$, where $\Sigma^\dagger$ is the diagonal matrix whose i-th diagonal entry is $1/\Sigma_{ii}$ if $\Sigma_{ii} > 0$ and 0 otherwise. (If no diagonal entry of Σ is zero, $\Sigma^\dagger = \Sigma^{-1}$.)

The pseudoinverse gives an optimal solution $x = A^\dagger b$ to the least squares problem, whether or not the columns of A are linearly independent.

Claim 1.3. *$x = A^\dagger b$ is an optimal least squares solution.*

Proof. We show that $x = A^\dagger b$ satisfies the normal equations $A^\top Ax = A^\top b$. Since U and V have orthonormal columns, $U^\top U = I$ and $V^\top V = I$, so

$$A^\top AA^\dagger b = (V\Sigma U^\top)(U\Sigma V^\top)(V\Sigma^\dagger U^\top)b = V\Sigma(U^\top U)\Sigma(V^\top V)\Sigma^\dagger U^\top b = V\Sigma\Sigma\Sigma^\dagger U^\top b.$$

The matrix $\Sigma\Sigma^\dagger$ is diagonal, with (i,i) entry 1 if $\Sigma_{ii} > 0$ and 0 otherwise. Therefore $\Sigma(\Sigma\Sigma^\dagger) = \Sigma$, and

$$A^\top AA^\dagger b = V\Sigma U^\top b = A^\top b. \qquad \square$$

If the columns of A are not linearly independent, then $x = A^\dagger b$ is not the unique optimal solution, but it has the minimum norm among them.

Claim 1.4. *$x = A^\dagger b$ has the minimum ℓ_2 norm among all optimal solutions.*

Proof. Let $r = \operatorname{rank}(A)$. Let $(V')^\top$ consist of the rows i of $V^\top$ for which $\Sigma_{ii} > 0$, and $(V'')^\top$ of the rows i of $V^\top$ for which $\Sigma_{ii} = 0$, so that

$$A = U \begin{bmatrix} \Sigma_{11} & & & & & \\ & \ddots & & & & \\ & & \Sigma_{rr} & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{bmatrix} \begin{bmatrix} (V')^\top \\ (V'')^\top \end{bmatrix}.$$

The rows of $(V')^\top$ span the row space of A, so $I - V'(V')^\top$ is the projection onto the kernel of A, and $A(I - V'(V')^\top)z = 0$ for every z. Therefore $A^\dagger b + (I - V'(V')^\top)z$ also solves the normal equations, and so it is optimal too; these solutions form a $(d - \operatorname{rank}(A))$-dimensional affine space. In fact, every optimal x has this form. Both x and $A^\dagger b$ solve the normal equations, so $A^\top A(x - A^\dagger b) = 0$, hence $A(x - A^\dagger b) = 0$. So $x = A^\dagger b + (I - V'(V')^\top)z$ with $z = x - A^\dagger b$. Finally, $A^\dagger b = V\Sigma^\dagger U^\top b$ lies in the column span of V', which is orthogonal to the range of $I - V'(V')^\top$, so by the Pythagorean theorem

$$\|A^\dagger b + (I - V'(V')^\top)z\|_2^2 = \|A^\dagger b\|_2^2 + \|(I - V'(V')^\top)z\|_2^2 \geq \|A^\dagger b\|_2^2. \qquad \square$$

1.1.4 Running time

The most expensive part of solving least squares this way is computing $x = A^\dagger b$, which takes $O(nd^2)$ time naively. With fast matrix multiplication it can be done in $O(nd^{1.376})$ time. This is still too slow for massive data sets with $n \gg d$; we want a much better running time.

1.2 The sketch-and-solve paradigm

To solve least squares faster, we settle for an approximate solution. Our goal is to output x' such that, with high probability,

$$\|Ax' - b\|_2 \leq (1+\varepsilon)\min_x \|Ax - b\|_2. \tag{1.2}$$

The *sketch-and-solve* method works as follows.

1. Draw S from a random family of $m \times n$ matrices, for a value $m \ll n$.
2. Compute SA and Sb.
3. Output the solution x' to $\min_x \|(SA)x - (Sb)\|_2$, that is, $x' = (SA)^\dagger Sb$.

Given SA, computing $(SA)^\dagger$ takes $O(md^2)$ time, a major improvement over computing $A^\dagger$ when $m \ll n$. We still have to compute SA, which naively takes $O(mnd)$ time; later we will choose random families of very structured matrices, for which SA is much faster to compute (Lecture 2).

Many random families of matrices work. In this lecture, S is a Gaussian sketch: an $m \times n$ matrix, with $m = O(d/\varepsilon^2)$, whose entries are i.i.d. $\mathcal{N}(0, 1/m)$ random variables. To see why this works, we introduce the notion of a subspace embedding.

1.3 Subspace embeddings

Definition 1.5 (Subspace embedding)**.** Let $A \in \mathbb{R}^{n\times d}$. A matrix $S \in \mathbb{R}^{m\times n}$ is an *ε-subspace embedding* for (the column span of) A if

$$\|SAx\|_2 = (1 \pm \varepsilon)\|Ax\|_2 \quad \text{for all } x \in \mathbb{R}^d. \tag{1.3}$$

A subspace embedding is valid simultaneously for *all* vectors in a fixed subspace of dimension at most d: the lengths of all vectors Ax in the column span of A are preserved. We will work with the squared form $\|SAx\|_2^2 = (1 \pm \varepsilon)\|Ax\|_2^2$, which is equivalent up to a constant factor in ε: for $\varepsilon \leq 1$, $\sqrt{1+\varepsilon} \leq 1 + \varepsilon$ and $\sqrt{1-\varepsilon} \geq 1 - \varepsilon$, and $(1 \pm \varepsilon)^2 \subseteq 1 \pm 3\varepsilon$.

Theorem 1.6 (Gaussian matrices are subspace embeddings)**.** *Let $m = O(d/\varepsilon^2)$ (with a large enough constant) and let S be an $m \times n$ matrix of i.i.d. $\mathcal{N}(0, 1/m)$ random variables. For any fixed $A \in \mathbb{R}^{n\times d}$, with probability at least $1 - 2^{-\Theta(d)}$, S is an ε-subspace embedding for A.*

The rest of this section proves Theorem 1.6.

1.3.1 Simplifications

We can assume that the columns of A are orthonormal, since we prove the claim for all x at once. The set $\{Ax : x \in \mathbb{R}^d\}$ is the column span of A, which equals $\{Uz : z \in \mathbb{R}^r\}$ for a matrix U with $r = \operatorname{rank}(A) \leq d$ orthonormal columns. So (1.3) for U and all z is the same statement as for A and all x. We can also assume that x is a unit vector, since both sides of (1.3) scale linearly with x. If A has orthonormal columns and $\|x\|_2 = 1$, then $\|Ax\|_2 = \|x\|_2 = 1$, so we want to show that $\|SAx\|_2 = 1 \pm \varepsilon$, or in squared form $\|SAx\|_2^2 = (1 \pm \varepsilon)^2$, simultaneously for all unit x.

1.3.2 Properties of Gaussians

Lemma 1.7. *If $X \sim \mathcal{N}(0, a^2)$ and $Y \sim \mathcal{N}(0, b^2)$ are independent, then $Z = X + Y \sim \mathcal{N}(0, a^2 + b^2)$.*

Proof. The probability density function f_Z of Z is the convolution of the densities f_X and f_Y of X and Y:

$$f_Z(z) = \int f_X(z-y) f_Y(y)\, dy = \int \frac{1}{a\sqrt{2\pi}} \exp\Big(-\frac{(z-y)^2}{2a^2}\Big) \frac{1}{b\sqrt{2\pi}} \exp\Big(-\frac{y^2}{2b^2}\Big) dy$$
$$= \int \frac{1}{2\pi ab} \exp\Big(-\frac{(z-y)^2}{2a^2} - \frac{y^2}{2b^2}\Big) dy = \frac{1}{\sqrt{a^2+b^2}\sqrt{2\pi}} \int \frac{\sqrt{a^2+b^2}}{ab\sqrt{2\pi}} \exp\Big(-\frac{(z-y)^2}{2a^2} - \frac{y^2}{2b^2}\Big) dy.$$

A calculation (completing the square in y) shows that

$$-\frac{(z-y)^2}{2a^2} - \frac{y^2}{2b^2} = -\frac{z^2}{2(a^2+b^2)} - \frac{\big(y - \frac{b^2 z}{a^2+b^2}\big)^2}{2\frac{(ab)^2}{a^2+b^2}}.$$

This splits the exponential into two Gaussian densities:

$$f_Z(z) = \underbrace{\frac{1}{\sqrt{a^2+b^2}\sqrt{2\pi}} \exp\Big(-\frac{z^2}{2(a^2+b^2)}\Big)}_{\text{density of } \mathcal{N}(0, a^2+b^2)} \int \underbrace{\frac{\sqrt{a^2+b^2}}{ab\sqrt{2\pi}} \exp\left(-\frac{\big(y - \frac{b^2 z}{a^2+b^2}\big)^2}{2\frac{(ab)^2}{a^2+b^2}}\right)}_{\text{density of } \mathcal{N}\left(\frac{b^2 z}{a^2+b^2}, \frac{(ab)^2}{a^2+b^2}\right)} dy.$$

A probability density integrates to 1 over $(-\infty, \infty)$, so

$$f_Z(z) = \frac{1}{\sqrt{a^2+b^2}\sqrt{2\pi}} \exp\Big(-\frac{z^2}{2(a^2+b^2)}\Big) \cdot 1. \qquad \square$$

By induction, and since $cX \sim \mathcal{N}(0, c^2 a^2)$ for a constant c, a linear combination $\sum_j c_j X_j$ of independent $X_j \sim \mathcal{N}(0, a^2)$ is distributed as $\mathcal{N}(0, a^2 \sum_j c_j^2)$.

Fact 1.8. *If $g \sim \mathcal{N}(0, I_n)$ is an n-dimensional vector of i.i.d. $\mathcal{N}(0,1)$ random variables and R is a fixed invertible $n \times n$ matrix, then the probability density function of Rg is*

$$f(x) = \frac{1}{\det(RR^\top)^{1/2}(2\pi)^{n/2}} \exp\Big(-\frac{x^\top (RR^\top)^{-1} x}{2}\Big),$$

where $RR^\top$ is the covariance matrix.

Remark 1.9 (Rotational invariance). If R is a rotation (an orthogonal matrix), then $RR^\top = I$, so the density of Rg is $(2\pi)^{-n/2} \exp(-x^\top x/2)$: Rg and g have the same distribution. The same holds for any rescaling cg of g.

Lemma 1.10 (Orthogonal implies independent)**.** *If $u, v \in \mathbb{R}^n$ satisfy $\langle u, v\rangle = 0$, then $\langle g, u\rangle$ and $\langle g, v\rangle$ are independent, where $g \sim \mathcal{N}(0, I_n)$. The same holds for any rescaling cg; in particular, for each row of S, which is distributed as $g^\top/\sqrt{m}$.*

Proof. Since $\langle u, v\rangle = 0$, we can choose a rotation R such that

$$Ru = \alpha e_1 = \alpha(1, 0, \ldots, 0) \quad \text{and} \quad Rv = \beta e_2 = \beta(0, 1, 0, \ldots, 0),$$

where $\alpha = \|u\|_2$ and $\beta = \|v\|_2$. Rotations preserve inner products, so with $h = Rg$,

$$\langle g, u\rangle = \langle Rg, Ru\rangle = \langle h, \alpha e_1\rangle = \alpha h_1 \quad \text{and} \quad \langle g, v\rangle = \langle Rg, Rv\rangle = \langle h, \beta e_2\rangle = \beta h_2.$$

By rotational invariance, h is again a vector of i.i.d. $\mathcal{N}(0,1)$ random variables, so h_1 and h_2 are independent by definition. $\square$

Lemma 1.11. *Let S be an $m \times n$ matrix of i.i.d. $\mathcal{N}(0, 1/m)$ random variables, and let $A \in \mathbb{R}^{n \times d}$ have orthonormal columns. Then SA is an $m \times d$ matrix of i.i.d. $\mathcal{N}(0, 1/m)$ random variables.*

Proof. The rows of S are independent, so the rows of SA are independent. Write a row of S as $g^\top/\sqrt{m}$ with $g \sim \mathcal{N}(0, I_n)$; the corresponding row of SA is

$$\frac{1}{\sqrt{m}}(\langle g, A_{*,1}\rangle, \ldots, \langle g, A_{*,d}\rangle).$$

Each entry $\langle g, A_{*,i}\rangle/\sqrt{m} = \sum_j (g_j/\sqrt{m})A_{ji}$ is a sum of independent $\mathcal{N}(0, 1/m)$ random variables scaled by the entries A_{ji}, so by Lemma 1.7 it is distributed as $\mathcal{N}(0, \|A_{*,i}\|_2^2/m) = \mathcal{N}(0, 1/m)$, since $\|A_{*,i}\|_2 = 1$. Finally, since the columns of A are orthonormal, Lemma 1.10 implies that any two entries of a row are independent. Note that pairwise independence alone does not give independence of all d entries, but the proof of Lemma 1.10 does. We extend the columns of A to an orthonormal basis of $\mathbb{R}^n$ and let R be the rotation mapping $A_{*,i}$ to e_i for all i at once. Then $\langle g, A_{*,i}\rangle = \langle Rg, e_i\rangle = h_i$ for $i \in [d]$, where $h = Rg$ has i.i.d. $\mathcal{N}(0, 1)$ entries, so the entries of a row are mutually independent. Therefore all the entries of SA are independent. □

Lemma 1.12. *Let S be an $m \times n$ matrix of i.i.d. $\mathcal{N}(0, 1/m)$ random variables, and let $A \in \mathbb{R}^{n \times d}$ have orthonormal columns. Then for any fixed unit vector $x \in \mathbb{R}^d$,*

$$\mathbb{E}[\|SAx\|_2^2] = 1.$$

Proof. Write $w_i^\top$ for the i-th row of SA, so that

$$\|SAx\|_2^2 = \sum_{i \in [m]} \langle w_i, x\rangle^2.$$

By Lemma 1.11, w_i has i.i.d. $\mathcal{N}(0, 1/m)$ entries, so $\langle w_i, x\rangle \sim \mathcal{N}(0, \|x\|_2^2/m) = \mathcal{N}(0, 1/m)$, and $\mathbb{E}[\langle w_i, x\rangle^2] = 1/m$. By linearity of expectation,

$$\mathbb{E}\Big[\sum_{i \in [m]} \langle w_i, x\rangle^2\Big] = \sum_{i \in [m]} \mathbb{E}[\langle w_i, x\rangle^2] = \sum_{i \in [m]} \frac{1}{m} = 1.$$ □

So $\|SAx\|_2^2$ has the right expectation. Next we show that it is concentrated around its expectation.

1.3.3 Concentration for a fixed vector: the JL lemma

Fact 1.13 (Tail bounds for χ^2 random variables)**.** *Let $h_1, \ldots, h_m$ be i.i.d. $\mathcal{N}(0, 1)$ random variables. Then $G = \sum_{i=1}^m h_i^2$ is a χ^2 random variable (with m degrees of freedom), and for every $t > 0$,*

$$\Pr[G \geq m + 2\sqrt{mt} + 2t] \leq e^{-t},$$
$$\Pr[G \leq m - 2\sqrt{mt}] \leq e^{-t}.$$

Theorem 1.14 (Johnson–Lindenstrauss)**.** *Let S be an $m \times n$ matrix of i.i.d. $\mathcal{N}(0, 1/m)$ random variables, let $A \in \mathbb{R}^{n \times d}$ have orthonormal columns, and let $x \in \mathbb{R}^d$ be a fixed unit vector. For $0 < \varepsilon \leq 1$,*

$$\Pr[\|SAx\|_2^2 \in 1 \pm \varepsilon] \geq 1 - 2e^{-\varepsilon^2 m/16}.$$

In particular, if $m = \Theta(\varepsilon^{-2} \log(1/\delta))$ this probability is at least $1 - \delta$, and if $m = \Theta(d/\varepsilon^2)$ it is at least $1 - 2^{-\Theta(d)}$.

Proof. Take $t = \varepsilon^2 m/16$ in Fact 1.13. Then $2\sqrt{mt} = \varepsilon m/2$ and $2t = \varepsilon^2 m/8$, so

$$\Pr\Big[G \geq m + \frac{\varepsilon m}{2} + \frac{\varepsilon^2 m}{8}\Big] \leq e^{-\varepsilon^2 m/16},$$
$$\Pr\Big[G \leq m - \frac{\varepsilon m}{2}\Big] \leq e^{-\varepsilon^2 m/16}.$$

Since $\varepsilon m/2 + \varepsilon^2 m/8 \leq \varepsilon m$, these two bounds give

$$\Pr[G \in m(1 \pm \varepsilon)] \geq 1 - 2e^{-\varepsilon^2 m/16}.$$

Now $\|SAx\|_2^2 = \sum_{i\in[m]} \langle w_i, x\rangle^2$, where the $\langle w_i, x\rangle$ are i.i.d. $\mathcal{N}(0, 1/m)$ (proof of Lemma 1.12). So $\|SAx\|_2^2$ has the same distribution as G/m, and

$$\Pr[\|SAx\|_2^2 \in 1 \pm \varepsilon] = \Pr\Big[\frac{G}{m} \in 1 \pm \varepsilon\Big] \geq 1 - 2e^{-\varepsilon^2 m/16}.$$

With $m = \Theta(\varepsilon^{-2}\log(1/\delta))$ the failure probability is at most δ; with $m = \Theta(d/\varepsilon^2)$ it is $2^{-\Theta(d)}$. □

Theorem 1.14 holds for a single fixed x. We cannot simply take a union bound over all $x \in \mathbb{R}^d$: there are infinitely many of them. Instead we use a *net argument*, which lets us reason about a finite set of vectors.

1.3.4 Nets

Definition 1.15 (γ-net)**.** Consider the unit sphere $\mathbb{S}^{d-1} = \{x \in \mathbb{R}^d : \|x\|_2 = 1\}$. A subset $N \subseteq \mathbb{S}^{d-1}$ is a *γ-net* if for every $x \in \mathbb{S}^{d-1}$ there is a $y \in N$ with $\|x - y\|_2 \leq \gamma$.

A net can be constructed greedily: start with $N = \emptyset$, and while there is a point $x \in \mathbb{S}^{d-1}$ with $\|x - y\|_2 > \gamma$ for all $y \in N$, add x to N.

Lemma 1.16. *For $\gamma > 0$, the greedy construction terminates, and it outputs a γ-net N with*

$$|N| \leq \frac{(1 + \gamma/2)^d}{(\gamma/2)^d}.$$

In particular, for $\gamma = 1/2$ we get a $\frac{1}{2}$-net with $|N| \leq 5^d$.

Proof. Every point added to N is at distance more than γ from all points added before it, so the points of N are pairwise at distance more than γ. Place a ball $B(x, \gamma/2)$ of radius $\gamma/2$ around each $x \in N$. These balls are pairwise disjoint: if $B(x, \gamma/2)$ and $B(y, \gamma/2)$ intersected, then $\|x - y\|_2 \leq \gamma$, contradicting the rule for adding the later of the two points. Every net point is at distance 1 from the origin 0^d, so by the triangle inequality every ball $B(x, \gamma/2)$ is contained in the ball $B(0^d, 1 + \gamma/2)$.

The ratio of the volume of a d-dimensional ball of radius $1 + \gamma/2$ to that of a d-dimensional ball of radius $\gamma/2$ is the ratio of the radii raised to the d-th power, $(1 + \gamma/2)^d/(\gamma/2)^d$. Since the disjoint balls fit inside $B(0^d, 1 + \gamma/2)$, at most this many points are ever added to N, so the construction terminates, and when it does every point of $\mathbb{S}^{d-1}$ is within distance γ of N. For $\gamma = 1/2$ the bound is $(5/4)^d/(1/4)^d = 5^d$. □

A net for the sphere induces a net for the unit vectors of the column span of A.

Claim 1.17. *Let A have orthonormal columns, let N be a γ-net for $\mathbb{S}^{d-1}$, and let $M = \{Ax : x \in N\} \subseteq \mathbb{R}^n$, so that $|M| \leq |N|$. For every $x \in \mathbb{S}^{d-1}$ there is a $y \in M$ with $\|Ax - y\|_2 \leq \gamma$.*

Proof. Let $x' \in N$ be such that $\|x - x'\|_2 \leq \gamma$. Since the columns of A are orthonormal,

$$\|Ax - Ax'\|_2 = \|x - x'\|_2 \leq \gamma,$$

so we can take $y = Ax'$. □

1.3.5 From a fixed vector to all vectors

By Theorem 1.14 with $m = \Theta(d/\varepsilon^2)$, for a fixed unit vector x, $\Pr[\|SAx\|_2^2 \in 1 \pm \varepsilon] \geq 1 - 2^{-\Theta(d)}$. For a fixed pair of unit vectors x, x', a union bound shows that $\|SAx\|_2^2$, $\|SAx'\|_2^2$ and $\|SA(x - x')\|_2^2$ are all preserved up to a factor $1 \pm \varepsilon$, with probability $1 - 2^{-\Theta(d)}$. Although $x - x'$ is not a unit vector, we can apply Theorem 1.14 to the fixed unit vector $(x - x')/\|x - x'\|_2$, which gives

$$\Pr\Big[\Big\|\frac{SA(x - x')}{\|x - x'\|_2}\Big\|_2^2 \in 1 \pm \varepsilon\Big] \geq 1 - 2^{-\Theta(d)},$$

that is,

$$\Pr\big[\|SA(x - x')\|_2^2 \in (1 \pm \varepsilon)\|x - x'\|_2^2\big] \geq 1 - 2^{-\Theta(d)},$$

and $\|x - x'\|_2 \leq 2$ since x and x' are unit vectors.

Claim 1.18. *If $x, x' \in \mathbb{R}^d$ are fixed unit vectors, then*

$$\Pr\big[\langle SAx, SAx'\rangle = \langle Ax, Ax'\rangle \pm O(\varepsilon)\big] \geq 1 - 2^{-\Theta(d)}.$$

Proof. Expanding $\|u - v\|_2^2$,

$$\begin{aligned} 2\langle SAx, SAx'\rangle &= \|SAx\|_2^2 + \|SAx'\|_2^2 - \|SA(x - x')\|_2^2, \\ 2\langle Ax, Ax'\rangle &= \|Ax\|_2^2 + \|Ax'\|_2^2 - \|A(x - x')\|_2^2. \end{aligned}$$

Subtracting the second line from the first,

$$\begin{aligned} 2(\langle SAx, SAx'\rangle - \langle Ax, Ax'\rangle) = (\|SAx\|_2^2 - \|Ax\|_2^2) + (\|SAx'\|_2^2 - \|Ax'\|_2^2) \\ + (\|A(x - x')\|_2^2 - \|SA(x - x')\|_2^2). \end{aligned}$$

With probability $1 - 2^{-\Theta(d)}$, all three norms are preserved as above, and since $\|Ax\|_2 = \|Ax'\|_2 = 1$ and $\|A(x - x')\|_2 = \|x - x'\|_2$,

$$2(\langle SAx, SAx'\rangle - \langle Ax, Ax'\rangle) = \pm\varepsilon \pm \varepsilon \pm \varepsilon\|x - x'\|_2^2.$$

Since $\|x - x'\|_2 \leq 2$, this gives $\langle SAx, SAx'\rangle - \langle Ax, Ax'\rangle = \pm O(\varepsilon)$. □

Now take a union bound over the points of a $\frac{1}{2}$-net $M = \{Ax : x \in N\}$, built from a $\frac{1}{2}$-net N of size at most 5^d (Lemma 1.16). There are at most $(5^d)^2 = 25^d$ pairs of net points (including pairs $y = y'$), and choosing the constant in $m = \Theta(d/\varepsilon^2)$ large enough makes the failure probability for each pair at most 50^{-d}. Therefore

$$\Pr\big[\forall y, y' \in M :\ \langle Sy, Sy'\rangle = \langle y, y'\rangle \pm O(\varepsilon)\big] \geq 1 - 25^d \cdot 50^{-d} = 1 - 2^{-d}. \tag{1.4}$$

For the rest of the proof we condition on this event. By linearity, if it holds for $y, y' \in M$, then for any scalars α, β,

$$\langle S\alpha y, S\beta y'\rangle = \alpha\beta\langle Sy, Sy'\rangle = \alpha\beta\langle y, y'\rangle \pm O(\varepsilon|\alpha\beta|) = \langle \alpha y, \beta y'\rangle \pm O(\varepsilon|\alpha\beta|). \tag{1.5}$$

Now let $y = Ax$ for an arbitrary $x \in \mathbb{S}^{d-1}$. Then y is a unit vector in the column span of A, but it is not necessarily a net point. We write y as a combination of scaled net points, and then evaluate $\|Sy\|_2^2$ through this combination. Let $\gamma = 1/2$.

- Choose $y_1 \in M$ with $\|y - y_1\|_2 \leq \gamma$ (Claim 1.17).

- Choose α with $\|\alpha(y - y_1)\|_2 = 1$; then $\alpha \geq 1/\gamma$ (if $y = y_1$, then α is infinite and we stop, with $y = y_1$).
- Choose $y_2' \in M$ with $\|\alpha(y - y_1) - y_2'\|_2 \leq \gamma$. Then

$$\left\| y - y_1 - \frac{y_2'}{\alpha} \right\|_2 \leq \frac{\gamma}{\alpha} \leq \gamma^2.$$

- Set $y_2 = y_2'/\alpha$, and repeat with $y - y_1 - y_2$ in place of $y - y_1$.

By induction, this produces $y_1, y_2, y_3, \ldots$, each a scalar multiple of a point of M, such that for every integer $i \geq 1$,

$$\|y - y_1 - y_2 - \cdots - y_i\|_2 \leq \gamma^i. \tag{1.6}$$

By the triangle inequality and (1.6) (with the empty sum for $i = 1$, where $\|y\|_2 = 1 = \gamma^0$),

$$\begin{aligned} \|y_i\|_2 &= \|(y - y_1 - \cdots - y_{i-1}) - (y - y_1 - \cdots - y_i)\|_2 \\ &\leq \|y - y_1 - \cdots - y_{i-1}\|_2 + \|y - y_1 - \cdots - y_i\|_2 \leq \gamma^{i-1} + \gamma^i \leq 2\gamma^{i-1}. \end{aligned}$$

Therefore $y = \sum_{i=1}^{\infty} y_i$, and the series converges.

Proof of Theorem 1.6. Condition on the event in (1.4), which has probability at least $1 - 2^{-d}$. Let $x \in \mathbb{S}^{d-1}$ and $y = Ax$, so that $SAx = Sy$, and write $y = \sum_i y_i$ as above. Then

$$\begin{aligned} \|Sy\|_2^2 &= \left\| S \sum_i y_i \right\|_2^2 = \sum_i \|Sy_i\|_2^2 + 2 \sum_{i<j} \langle Sy_i, Sy_j \rangle \\ &= \sum_i \|y_i\|_2^2 + 2 \sum_{i<j} \langle y_i, y_j \rangle \pm O(\varepsilon) \sum_{i,j} \|y_i\|_2 \|y_j\|_2 \\ &= \sum_i \|y_i\|_2^2 + 2 \sum_{i<j} \langle y_i, y_j \rangle \pm O(\varepsilon) = \left\| \sum_i y_i \right\|_2^2 \pm O(\varepsilon) = \|y\|_2^2 \pm O(\varepsilon) = 1 \pm O(\varepsilon), \end{aligned}$$

where the second line applies (1.5) to every pair y_i, y_j (each y_i is a scalar multiple of a point of M, and the absolute value of the scalar is $\|y_i\|_2$). The third line uses that $\sum_{i,j} \|y_i\|_2 \|y_j\|_2 = O(1)$, by a geometric series:

$$\sum_{i,j} \|y_i\|_2 \|y_j\|_2 = \Big(\sum_i \|y_i\|_2 \Big) \Big(\sum_j \|y_j\|_2 \Big) \leq \Big(\sum_{i \geq 1} 2\gamma^{i-1} \Big)^2 = \Big(\frac{2}{1 - \gamma} \Big)^2 = 16.$$

So $\|SAx\|_2^2 = 1 \pm O(\varepsilon)$ for every unit vector x, and replacing ε by ε/c for a suitable constant c (which changes m only by a constant factor) gives $\|SAx\|_2^2 = 1 \pm \varepsilon$. Since both sides are homogeneous in x, $\|SAx\|_2^2 = (1 \pm \varepsilon)\|Ax\|_2^2$ for all $x \in \mathbb{R}^d$, and taking square roots, $\|SAx\|_2 = (1 \pm \varepsilon)\|Ax\|_2$ for all x. □

1.4 Back to regression

We showed that S is a subspace embedding: simultaneously for all x, $\|SAx\|_2 = (1 \pm \varepsilon)\|Ax\|_2$. Now we return to the goal (1.2) of the sketch-and-solve method.

Theorem 1.19. *Let S be an $m \times n$ matrix of i.i.d. $\mathcal{N}(0, 1/m)$ random variables with $m = O(d/\varepsilon^2)$, and let*

$$\tilde{x} = \arg\min_x \|S(Ax - b)\|_2.$$

Then, with probability at least $1 - 2^{-\Theta(d)}$,

$$\|A\tilde{x} - b\|_2 \leq (1 + O(\varepsilon)) \min_x \|Ax - b\|_2.$$

Proof. Consider the subspace L spanned by the columns of A together with b, of dimension at most $d + 1$. By Theorem 1.6 (applied to the $n \times (d + 1)$ matrix $[A, b]$), with probability $1 - 2^{-\Theta(d)}$, $\|Sy\|_2 = (1 \pm \varepsilon)\|y\|_2$ for every $y \in L$; in particular $\|S(Ax - b)\|_2 = (1 \pm \varepsilon)\|Ax - b\|_2$ for all x. Let $x^* = \arg\min_x \|Ax - b\|_2$ and $\tilde{x} = \arg\min_x \|(SA)x - (Sb)\|_2$. Then

$$\begin{aligned} \|A\tilde{x} - b\|_2 &\leq \frac{1}{1-\varepsilon}\|SA\tilde{x} - Sb\|_2 \leq \frac{1}{1-\varepsilon}\|SAx^* - Sb\|_2 \\ &\leq \frac{1+\varepsilon}{1-\varepsilon}\|Ax^* - b\|_2 \leq (1 + O(\varepsilon))\|Ax^* - b\|_2. \end{aligned}$$

□

As a result, the sketch-and-solve method computes a good approximation to least squares with high probability: given SA and Sb, the problem $\arg\min_x \|(SA)x - (Sb)\|_2$ can be solved in $\mathrm{poly}(d/\varepsilon)$ time, where $d \ll n$. The only remaining problem is that computing SA for a Gaussian S is slow: it takes $O(nd^2)$ time for constant ε. In Lecture 2 we will replace S by a structured matrix for which SA can be computed much faster.

2 The Subsampled Randomized Hadamard Transform and Approximate Matrix Product

Lecture 2, September 17, 2020, by David P. Woodruff. Compiled from the scribe notes of Eric Chen (first half) and Tommy Jiang (second half), and from the lecture slides.

In Lecture 1 we saw that an $m \times n$ matrix S of i.i.d. $\mathcal{N}(0, 1/m)$ random variables, with $m = O(d/\varepsilon^2)$, is a subspace embedding: with high probability, $\|SAx\|_2 = (1 \pm \varepsilon)\|Ax\|_2$ for all $x \in \mathbb{R}^d$. This reduces least squares regression on the $n \times d$ matrix A to a much smaller regression problem on SA. One issue with this approach is computing the product SA, which naively takes $O(mnd)$ time, that is, $O(nd^2)$ for constant ε (with fast matrix multiplication, $O(nd^{1.373})$ time). In this lecture we see a sketching matrix, the Subsampled Randomized Hadamard Transform (SRHT), for which SA can be computed in $O(nd \log n)$ time, and then CountSketch, for which SA can be computed in $O(\text{nnz}(A))$ time.

2.1 The Hadamard matrix

Throughout, $n = 2^q$ is a power of 2. We index the rows and columns of an $n \times n$ matrix by $\{0, 1, \ldots, n-1\}$, and identify an index i with its binary representation $(i_0, i_1, \ldots, i_{q-1}) \in \{0,1\}^q$, where i_0 is the most significant bit. For two indices i, j, let $\langle i, j\rangle = \sum_{p=0}^{q-1} i_p j_p$ be the inner product of their binary strings, viewed as vectors.

Definition 2.1 (Hadamard matrix)**.** The (normalized) Hadamard matrix $H = H_n$ is the $n \times n$ matrix with entries

$$H_{ij} = \frac{1}{\sqrt{n}}(-1)^{\langle i,j\rangle}.$$

So H_{ij} is $\pm 1/\sqrt{n}$, and only the parity of $\langle i, j\rangle$ matters: we can work mod 2.

Claim 2.2. *H is orthogonal: $H^\top H = HH^\top = I_n$.*

Proof. First, the i-th row $H_{i,*}$ has norm 1:

$$\|H_{i,*}\|_2^2 = \sum_{j=0}^{n-1} H_{ij}^2 = \sum_{j=0}^{n-1}\Big(\frac{1}{\sqrt{n}}(-1)^{\langle i,j\rangle}\Big)^2 = \sum_{j=0}^{n-1}\frac{1}{n} = 1.$$

It remains to show that $\langle H_{i,*}, H_{i',*}\rangle = 0$ for rows $i \neq i'$. We have

$$\langle H_{i,*}, H_{i',*}\rangle = \sum_{j=0}^{n-1} H_{ij}H_{i'j} = \sum_{j=0}^{n-1}\frac{1}{\sqrt{n}}(-1)^{\langle i,j\rangle}\frac{1}{\sqrt{n}}(-1)^{\langle i',j\rangle} = \frac{1}{n}\sum_{j=0}^{n-1}(-1)^{\langle i+i',j\rangle},$$

where $i + i'$ is the sum of the binary strings. Because $i \neq i'$, the string $i + i'$ has an odd entry, say at position p. We construct a bijection between the set of j with $\langle i + i', j\rangle$ even and the set of j with $\langle i + i', j\rangle$ odd: given j, flip its p-th bit. This flips the parity of $\langle i + i', j\rangle$, because $i + i'$ is odd in the p-th entry. So the number of j with $(-1)^{\langle i+i',j\rangle} = 1$ equals the number of j with $(-1)^{\langle i+i',j\rangle} = -1$, and $\sum_{j=0}^{n-1}(-1)^{\langle i+i',j\rangle} = 0$. □

Claim 2.3. *For $x \in \mathbb{R}^n$, the product $H_n x$ can be computed in $O(n \log n)$ time.*

Proof. Let $H'_n = \sqrt{n} H_n$, the matrix of signs $(-1)^{\langle i,j \rangle}$. The key fact is the recursive structure

$$H'_n = \begin{bmatrix} H'_{n/2} & H'_{n/2} \\ H'_{n/2} & -H'_{n/2} \end{bmatrix}.$$

To see it, split according to the most significant bits i_0 and j_0, and note that $\langle i, j \rangle = i_0 j_0 + \sum_{p=1}^{q-1} i_p j_p$.

- If $i, j < n/2$, then i and j both have most significant bit 0, so $\langle i, j \rangle = 0 \cdot 0 + \sum_{p=1}^{q-1} i_p j_p$, which is the inner product of i and j as strings of length $q-1$. Thus $(H'_n)_{ij} = (-1)^{\langle i,j \rangle} = (H'_{n/2})_{ij}$.
- If $i \geq n/2$ and $j < n/2$, then $\langle i, j \rangle = 1 \cdot 0 + \sum_{p=1}^{q-1} i_p j_p = \langle i - n/2, j \rangle$, since removing the most significant bit subtracts $2^{q-1} = n/2$. Thus $(H'_n)_{ij} = (-1)^{\langle i-n/2, j \rangle} = (H'_{n/2})_{i-n/2,j}$.
- If $j \geq n/2$ and $i < n/2$, the case is symmetric to the previous one.
- If $i, j \geq n/2$, then $\langle i, j \rangle = 1 \cdot 1 + \sum_{p=1}^{q-1} i_p j_p = \langle i - n/2, j - n/2 \rangle + 1$. Thus $(H'_n)_{ij} = (-1)^{\langle i-n/2, j-n/2 \rangle + 1} = -(H'_{n/2})_{i-n/2,j-n/2}$.

To compute $H_n x$, we compute $H'_n x$ and then scale by $1/\sqrt{n}$. To compute $H'_n x$, split x into its top $n/2$ entries x' and its bottom $n/2$ entries x'':

$$H'_n x = \begin{bmatrix} H'_{n/2} & H'_{n/2} \\ H'_{n/2} & -H'_{n/2} \end{bmatrix} \begin{bmatrix} x' \\ x'' \end{bmatrix} = \begin{bmatrix} H'_{n/2} x' + H'_{n/2} x'' \\ H'_{n/2} x' - H'_{n/2} x'' \end{bmatrix}.$$

So we only need to compute $H'_{n/2} x'$ and $H'_{n/2} x''$ and combine the results in $O(n)$ time. The running time satisfies $T(n) = 2T(n/2) + O(n)$, so $T(n) = O(n \log n)$. □

We needed the labels $0, \ldots, n-1$ only for the binary representations in this section. From now on we use only that H is orthogonal, that $H_{ij} = \pm 1/\sqrt{n}$, and Claim 2.3, and we index coordinates by $[n]$ as usual.

2.2 The Subsampled Randomized Hadamard Transform

Definition 2.4 (Subsampled Randomized Hadamard Transform)**.** The SRHT is the random matrix $S = PHD \in \mathbb{R}^{m \times n}$, where

- D is an $n \times n$ diagonal matrix whose diagonal entries are i.i.d. Unif$\{-1, 1\}$, that is, each is $+1$ or -1 with probability $1/2$, independently;
- H is the $n \times n$ normalized Hadamard matrix; and
- P is an $m \times n$ matrix that samples m rows of HD uniformly at random, with replacement, and rescales them: if the i-th row of HD is the j-th sample, then $P_{ji} = \sqrt{n/m}$ is the only nonzero entry of row j of P.

Claim 2.5. *SA can be computed in $O(nd \log n)$ time, and Sb in $O(n \log n)$ time.*

Proof. For a column $A_{*,j}$ of A, we can compute $DA_{*,j}$ in $O(n)$ time, because D is diagonal, and then $HDA_{*,j}$ in $O(n \log n)$ time by Claim 2.3. Since A has d columns, HDA takes $O(nd \log n)$ time. Applying P just selects and rescales m rows of HDA, in $O(md)$ time. So SA takes $O(nd \log n)$ time, and similarly Sb takes $O(n \log n)$ time. □

So the SRHT saves time over the Gaussian sketch. It remains to show that it is a subspace embedding.

2.3 The flattening lemma

As in Lecture 1, we can assume that A has orthonormal columns, and it suffices to show that with high probability,

$$\|SAx\|_2^2 = \|PHDAx\|_2^2 = 1 \pm \varepsilon \quad \text{for all } x \in \mathbb{R}^d \text{ with } \|x\|_2 = 1. \tag{2.1}$$

Since H is orthogonal and D is diagonal with entries ± 1, HD is also orthogonal (a rotation). Therefore

$$\|HDAx\|_2^2 = \|Ax\|_2^2 = 1$$

for every unit vector x. From now on we write $y = Ax$.

The next lemma says that, with good probability, the vector HDy is "flat": none of its entries is much larger than the others.

Lemma 2.6 (Flattening lemma)**.** *There is a constant $C > 0$ such that for every fixed unit vector $y \in \mathbb{R}^n$,*

$$\Pr\Big[\|HDy\|_\infty > C\sqrt{\frac{\log(nd/\delta)}{n}}\Big] \le \frac{\delta}{2d}.$$

This is close to the best we can hope for: since $\|HDy\|_2 = 1$, the smallest possible value of $\|HDy\|_\infty$ is $1/\sqrt{n}$, and the lemma matches it up to the factor $C\sqrt{\log(nd/\delta)}$. Intuitively, the lemma is useful because the norm of a flat vector can be estimated from a few sampled entries, which is exactly what P does. So HD flattens the vector, which makes the subsampling by P accurate.

The proof uses the following concentration inequality.

Theorem 2.7 (Azuma–Hoeffding)**.** *Let $Z_1, Z_2, \dots$ be independent zero-mean random variables with $|Z_j| \le \beta_j$ with probability 1. Then for every $t > 0$,*

$$\Pr\Big[\Big|\sum_j Z_j\Big| > t\Big] \le 2\exp\Big(-\frac{t^2}{2\sum_j \beta_j^2}\Big).$$

Proof of Lemma 2.6. We show that for each fixed coordinate i,

$$\Pr\Big[|(HDy)_i| > C\sqrt{\frac{\log(nd/\delta)}{n}}\Big] \le \frac{\delta}{2nd}, \tag{2.2}$$

and then apply a union bound over the n coordinates. We can write

$$(HDy)_i = \sum_{j=1}^n H_{ij} D_{jj} y_j.$$

Each term $Z_j = H_{ij} D_{jj} y_j$ is a random variable with mean 0: the only randomness is in D_{jj}, which is 1 with probability $1/2$ and -1 with probability $1/2$. The terms are independent, because the diagonal entries of D are independent. Finally, $|Z_j| \le |y_j|/\sqrt{n}$, because $D_{jj} = \pm 1$ and $H_{ij} = \pm 1/\sqrt{n}$.

So we can apply Theorem 2.7 with $\beta_j = |y_j|/\sqrt{n}$ and $t = C\sqrt{\log(nd/\delta)/n}$. Here $\sum_j \beta_j^2 = \frac{1}{n}\sum_{j=1}^n y_j^2 = \frac{1}{n}$, because y is a unit vector. Thus

$$\Pr\Big[\Big|\sum_{j=1}^n H_{ij} D_{jj} y_j\Big| > C\sqrt{\frac{\log(nd/\delta)}{n}}\Big] \le 2\exp\Big(-\frac{t^2}{2\sum_j \beta_j^2}\Big) = 2\exp\Big(-\frac{C^2}{2}\log\frac{nd}{\delta}\Big) \le 2\Big(\frac{\delta}{nd}\Big)^{C^2/2},$$

using $\log z \ge \ln z$ for $z \ge 1$. Choosing C to be a large enough constant makes this at most $\delta/(2nd)$ ($C = 2$ suffices when $nd \ge 4$, since then $2(\delta/(nd))^2 \le \delta/(2nd)$), which proves (2.2). By a union bound, the probability that some coordinate of HDy has absolute value more than $C\sqrt{\log(nd/\delta)/n}$ is at most $n \cdot \frac{\delta}{2nd} = \frac{\delta}{2d}$. □

We use the flattening lemma to bound the norms of the rows of HDA. Recall that A has orthonormal columns. Since HD is orthogonal, and orthogonal matrices preserve inner products, the columns of HDA are orthonormal as well.

Corollary 2.8. *With probability at least $1 - \delta/2$, for all $j \in [n]$,*

$$\|(HDA)_{j,*}\|_2 \le C\sqrt{\frac{d\log(nd/\delta)}{n}}.$$

Proof. Fix $i \in [d]$ and apply Lemma 2.6 to the unit vector $y = Ae_i$: with probability at least $1 - \frac{\delta}{2d}$,

$$\|HDAe_i\|_\infty \le C\sqrt{\frac{\log(nd/\delta)}{n}},$$

that is, all entries in the i-th column of HDA have absolute value at most $C\sqrt{\log(nd/\delta)/n}$. By a union bound over the d columns, with probability at least $1 - \frac{\delta}{2}$ all entries of HDA have absolute value at most $C\sqrt{\log(nd/\delta)/n}$. In that case, for every $j \in [n]$,

$$\|(HDA)_{j,*}\|_2 = \sqrt{\sum_{i=1}^{d}(HDA)_{ji}^2} \le C\sqrt{\frac{\log(nd/\delta)}{n}} \cdot \sqrt{d}.$$ □

This upper bound on the norm of every row of HDA is what we need to prove (2.1). (The bound can be optimized further.) From now on we condition on the event of Corollary 2.8, which has probability at least $1 - \delta/2$.

2.4 The matrix Chernoff bound

Theorem 2.9 (Matrix Chernoff bound)**.** *Let $X_1, \dots, X_m$ be independent copies of a symmetric random matrix $X \in \mathbb{R}^{d\times d}$ with $\mathbb{E}[X] = 0$, $\|X\|_2 \le \gamma$ with probability 1, and $\|\mathbb{E}[X^\top X]\|_2 \le \sigma^2$. Let $W = \frac{1}{m}\sum_{i\in[m]} X_i$. For any $\varepsilon > 0$,*

$$\Pr[\|W\|_2 > \varepsilon] \le 2d \cdot \exp\Big(-\frac{m\varepsilon^2}{2\sigma^2 + 2\gamma\varepsilon/3}\Big).$$

Here $\|W\|_2 = \sup_{x\neq 0}\|Wx\|_2/\|x\|_2$ is the spectral norm.

Note that the denominator of the exponent is $2\sigma^2 + 2\gamma\varepsilon/3$, not $\sigma^2 + \gamma\varepsilon/3$. The factor 2 is needed, and it changes the number of samples below only by a constant factor. We use the same form in Lecture 4.

To apply the matrix Chernoff bound, we define the random matrix X. Let $V = HDA$, and recall that V has orthonormal columns. For $j \in [n]$ let $v_j = (V_{j,*})^\top \in \mathbb{R}^d$ be the j-th row of V, written as a column vector. The sampling matrix P of the SRHT samples m rows uniformly and with replacement, scaling each sampled row by $\sqrt{n/m}$: if row j is the i-th sample, then $P_{ij} = \sqrt{n/m}$. Let $Y_i \in \mathbb{R}^d$ be the i-th sampled row of V (as a column vector), so $Y_i = v_j$ with probability $1/n$ for each $j \in [n]$, and let

$$X_i = I_d - nY_iY_i^\top.$$

Each X_i is symmetric, since I_d and the outer product $Y_iY_i^\top$ are symmetric. The rows Y_i are sampled uniformly with replacement, so they are independent, and hence the X_i are independent copies of a random matrix X. The sampling in P is independent of D, so this remains true after we condition on the event of Corollary 2.8, which depends only on D.

Claim 2.10. $\mathbb{E}[X_i] = 0$.

Proof. Since Y_i is a uniformly random row of V,

$$\mathbb{E}[Y_i Y_i^\top] = \sum_{j=1}^{n} \frac{1}{n} v_j v_j^\top = \frac{1}{n} V^\top V = \frac{1}{n} I_d,$$

because V has orthonormal columns. Therefore

$$\mathbb{E}[X_i] = I_d - n\,\mathbb{E}[Y_i Y_i^\top] = I_d - n \cdot \frac{1}{n} I_d = 0. \qquad \square$$

Claim 2.11. $\|nY_iY_i^\top\|_2 \le n \cdot \max_{j\in[n]} \|v_j\|_2^2$.

Proof. We can write

$$nY_iY_i^\top = \frac{Y_i}{\|Y_i\|_2} \cdot n\|Y_i\|_2^2 \cdot \frac{Y_i^\top}{\|Y_i\|_2},$$

a rank-one matrix whose only nonzero eigenvalue is $n\|Y_i\|_2^2$, so $\|nY_iY_i^\top\|_2 = n\|Y_i\|_2^2$. Since $Y_i = v_j$ for some $j \in [n]$, $\|Y_i\|_2 \le \max_j \|v_j\|_2$. $\square$

Claim 2.12. $\|X_i\|_2 \le \gamma$ *for* $\gamma = \Theta(d \log(nd/\delta))$.

Proof. By the triangle inequality for the operator norm, Claim 2.11, and Corollary 2.8,

$$\begin{aligned}\|X_i\|_2 = \|I_d - nY_iY_i^\top\|_2 &\le \|I_d\|_2 + \|nY_iY_i^\top\|_2 \le \|I_d\|_2 + n \cdot \max_j \|v_j\|_2^2 \\ &\le 1 + n \cdot \left(C\sqrt{d\log(nd/\delta)/n}\right)^2 = 1 + C^2 d \log(nd/\delta) = \Theta(d\log(nd/\delta)). \square\end{aligned}$$

To bound $\|\mathbb{E}[X^\top X]\|_2$ we use the Loewner order: for symmetric matrices A, B, $A \preceq B$ means that $B - A$ is positive semidefinite, that is, $x^\top Ax \le x^\top Bx$ for all vectors x.

Fact 2.13. *If A and B are positive semidefinite and $A \preceq B$, then $\|A\|_2 \le \|B\|_2$.*

This is because, for a positive semidefinite matrix, $\|A\|_2 = \max_{\|x\|_2=1} x^\top Ax$, and $x^\top Ax \le x^\top Bx$ for every x.

Claim 2.14. $\|\mathbb{E}[X^\top X]\|_2 \le \sigma^2$ *for* $\sigma^2 = O(d \log(nd/\delta))$.

Proof. We first compute $\mathbb{E}[X^\top X]$. Each X_i is symmetric, so

$$\begin{aligned}\mathbb{E}[X^\top X] = \mathbb{E}[X_i X_i] &= \mathbb{E}[(I_d - nY_iY_i^\top)^2] = \mathbb{E}[I_d - 2nY_iY_i^\top + n^2 Y_iY_i^\top Y_iY_i^\top] \\ &= I_d - 2n\,\mathbb{E}[Y_iY_i^\top] + n^2\,\mathbb{E}[Y_iY_i^\top Y_iY_i^\top].\end{aligned}$$

We computed $\mathbb{E}[Y_iY_i^\top] = \frac{1}{n} I_d$ in Claim 2.10. Since $v_j^\top v_j = \|v_j\|_2^2$,

$$\mathbb{E}[Y_iY_i^\top Y_iY_i^\top] = \sum_{j=1}^{n} \frac{1}{n} v_j (v_j^\top v_j) v_j^\top = \frac{1}{n} \sum_{j=1}^{n} \|v_j\|_2^2\, v_j v_j^\top.$$

Therefore

$$\mathbb{E}[X^\top X + I_d] = 2I_d - 2n \cdot \frac{1}{n} I_d + n^2 \cdot \frac{1}{n} \sum_{j=1}^{n} \|v_j\|_2^2\, v_j v_j^\top = n \sum_{j=1}^{n} \|v_j\|_2^2\, v_j v_j^\top. \tag{2.3}$$

Now define

$$Z = n\sum_{j=1}^{n} v_j v_j^\top \cdot C^2 \log\Big(\frac{nd}{\delta}\Big) \cdot \frac{d}{n} = C^2 d \log\Big(\frac{nd}{\delta}\Big) \sum_{j=1}^{n} v_j v_j^\top = C^2 d \log\Big(\frac{nd}{\delta}\Big) I_d,$$

using $\sum_j v_j v_j^\top = V^\top V = I_d$. In particular $\|Z\|_2 = C^2 d \log(nd/\delta)$.

Both $\mathbb{E}[X^\top X + I_d]$ and Z are real symmetric matrices with non-negative eigenvalues. We claim that $\mathbb{E}[X^\top X + I_d] \preceq Z$. For every vector $y \in \mathbb{R}^d$, by (2.3),

$$y^\top \mathbb{E}[X^\top X + I_d] y = n\sum_{j=1}^{n} \|v_j\|_2^2 \, y^\top v_j v_j^\top y = n\sum_{j=1}^{n} \langle v_j, y\rangle^2 \|v_j\|_2^2,$$

while

$$y^\top Z y = n\sum_{j=1}^{n} y^\top v_j v_j^\top y \cdot C^2 \log\Big(\frac{nd}{\delta}\Big) \cdot \frac{d}{n} = n\sum_{j=1}^{n} \langle v_j, y\rangle^2 \cdot C^2 \log\Big(\frac{nd}{\delta}\Big) \cdot \frac{d}{n}.$$

Since v_j is the j-th row of $V = HDA$, Corollary 2.8 gives

$$\|v_j\|_2 \le C\frac{\sqrt{d\log(nd/\delta)}}{\sqrt{n}}, \quad \text{so} \quad \|v_j\|_2^2 \le C^2 \log\Big(\frac{nd}{\delta}\Big) \cdot \frac{d}{n},$$

and comparing the two sums term by term, $y^\top \mathbb{E}[X^\top X + I_d] y \le y^\top Z y$.

By Fact 2.13, $\|\mathbb{E}[X^\top X + I_d]\|_2 \le \|Z\|_2 = C^2 d \log(nd/\delta)$. Finally, by the triangle inequality,

$$\begin{aligned}\|\mathbb{E}[X^\top X]\|_2 = \|\mathbb{E}[X^\top X] + I_d - I_d\|_2 &\le \|\mathbb{E}[X^\top X] + I_d\|_2 + \|I_d\|_2 = \|\mathbb{E}[X^\top X + I_d]\|_2 + 1 \\ &\le C^2 d \log\Big(\frac{nd}{\delta}\Big) + 1 = O\Big(d \log \frac{nd}{\delta}\Big).\end{aligned}$$

□

Now we apply the matrix Chernoff bound. The matrix W is

$$W = \frac{1}{m}\sum_{i\in[m]} X_i = \frac{1}{m}\sum_{i\in[m]} (I_d - nY_iY_i^\top) = \frac{1}{m}\Big(mI_d - n\sum_{i\in[m]} Y_iY_i^\top\Big) = I_d - \sum_{i\in[m]} \Big(\sqrt{\frac{n}{m}}\, Y_i\Big)\Big(\sqrt{\frac{n}{m}}\, Y_i\Big)^\top.$$

Here Y_i is the i-th sampled row of $V = HDA$, and $\sqrt{n/m}$ is exactly the scaling factor in P, so $\sqrt{n/m}\, Y_i^\top$ is the i-th row of the sketch $PHDA$. Thus

$$W = I_d - (PHDA)^\top (PHDA).$$

By Theorem 2.9, with $\gamma = \Theta(d\log(nd/\delta))$ and $\sigma^2 = O(d\log(nd/\delta))$, for $\varepsilon \le 1$ the denominator $2\sigma^2 + 2\gamma\varepsilon/3$ is $O(d\log(nd/\delta))$, so

$$\Pr[\|I_d - (PHDA)^\top(PHDA)\|_2 > \varepsilon] \le 2d \cdot \exp\Big(-\frac{m\varepsilon^2}{2\sigma^2 + 2\gamma\varepsilon/3}\Big) \le 2d \cdot \exp\Big(-\frac{m\varepsilon^2}{O(d\log(nd/\delta))}\Big).$$

Setting $m = c\, d \log(nd/\delta)\log(d/\delta)/\varepsilon^2$ for a large enough constant c makes this probability less than $\delta/2$.

2.5 Wrapping up the SRHT

We have shown, using the matrix Chernoff bound, that with $m = \Theta(\varepsilon^{-2} d \log(nd/\delta) \log(d/\delta))$ samples, $\|I_d - (PHDA)^\top (PHDA)\|_2 < \varepsilon$ with probability at least $1 - \delta/2$, conditioned on the event of Corollary 2.8. Since that event fails with probability at most $\delta/2$, a union bound shows that $\|I_d - (PHDA)^\top (PHDA)\|_2 < \varepsilon$ with probability at least $1 - \delta$ overall. For every unit vector x, multiplying $I_d - (PHDA)^\top (PHDA)$ by $x^\top$ on the left and by x on the right gives

$$|1 - \|PHDAx\|_2^2| = |x^\top x - x^\top (PHDA)^\top (PHDA) x| < \varepsilon,$$

so $\|PHDAx\|_2^2 \in 1 \pm \varepsilon$ for all unit vectors x. This proves that the SRHT is a subspace embedding. As in Lecture 1, we can then solve the regression problem by applying the sketch to the column span of A adjoined with b.

Computing Sb takes $O(n \log n)$ time and computing SA takes $O(nd \log n)$ time; solving the sketched least squares problem then takes $\text{poly}(d \log(n)/\varepsilon)$ time. The total time is $O(nd \log n) + \text{poly}(d \log(n)/\varepsilon)$, which is nearly optimal in the matrix dimensions when $n \gg d$.

2.6 CountSketch

With the SRHT we have a nearly optimal running time for least squares regression on dense matrices A. A natural follow-up question is whether we can do better when A is sparse. CountSketch as a subspace embedding is due to [CW17, MM13, NN13]. The simple proof below is due to [Ngu13], and the matrix product result is due to [KN14].

Definition 2.15 (CountSketch)**.** A CountSketch matrix is an $m \times n$ matrix S, with $m = O(d^2/\varepsilon^2)$, that has a single randomly chosen nonzero entry in each column. That is, $S_{h(i),i} = \sigma(i)$ for each $i \in [n]$, and all other entries are 0, where $h\colon [n] \to [m]$ is a random hash function and $\sigma\colon [n] \to \{-1, 1\}$ gives random signs. In the analysis, h is pairwise independent and σ is 4-wise independent (see Lecture 3).

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & -1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Figure 2: An example of a 4×8 CountSketch matrix.

Claim 2.16. *SA can be computed in $O(\text{nnz}(A))$ time, where $\text{nnz}(A)$ is the number of nonzero entries of A.*

A simple algorithm uses a sparse representation of A (for example, a list of the nonzero entries of A with their positions), and iterates over the nonzero entries of A: the entry A_{ij} contributes $S_{h(i),i} A_{ij}$ to the entry $(h(i), j)$ of SA. Since each column of S has only one nonzero entry, this takes constant time per nonzero entry of A, for a total of $O(\text{nnz}(A))$ time.

2.6.1 CountSketch is a subspace embedding

As in our previous proofs of subspace embeddings, we can assume that the columns of A are orthonormal, and it suffices to show that $\|SAx\|_2 = 1 \pm \varepsilon$ for all unit vectors x. For regression, we then apply S to the matrix $[A, b]$, with b adjoined to the columns of A. Let $m = 6d^2/(\delta\varepsilon^2)$, so that SA is a $6d^2/(\delta\varepsilon^2) \times d$ matrix.

Claim 2.17. *If A has orthonormal columns and $\|A^\top S^\top SA - I\|_F \le \varepsilon$, then $\|SAx\|_2 = 1 \pm O(\varepsilon)$ for all unit vectors x.*

Proof. Since $\|A^\top S^\top SA - I\|_2 \le \|A^\top S^\top SA - I\|_F \le \varepsilon$, for every unit vector x,

$$|x^\top A^\top S^\top SAx - x^\top x| \le \varepsilon, \quad \text{that is,} \quad |\|SAx\|_2^2 - 1| \le \varepsilon.$$

So $\|SAx\|_2^2 = 1 \pm \varepsilon$, and hence $\|SAx\|_2 = 1 \pm O(\varepsilon)$. □

Lemma 2.18 (Matrix product result)**.** *Let S be an $m \times n$ CountSketch matrix. For any fixed matrices A and B with n rows,*

$$\Pr\Big[\|A^\top S^\top SB - A^\top B\|_F^2 \le \frac{6}{\delta m}\|A\|_F^2\|B\|_F^2\Big] \ge 1 - \delta.$$

We use the matrix product result now, and prove it later (in Lecture 3). Note that the proof there gives the constant 18 in place of 6. This changes m below only by a constant factor, to $m = 18d^2/(\delta\varepsilon^2)$ instead of $6d^2/(\delta\varepsilon^2)$. Apply it with $B = A$, where A has orthonormal columns. Each column of A has norm 1, so the squared Frobenius norm of A is the number of columns, $\|A\|_F^2 = d$; also $A^\top A = I$. With $m = 6d^2/(\delta\varepsilon^2)$,

$$\begin{aligned}\Pr\Big[\|A^\top S^\top SA - I\|_F^2 \le \frac{6}{\delta \cdot 6d^2/(\delta\varepsilon^2)} \cdot d \cdot d\Big] &= \Pr[\|A^\top S^\top SA - I\|_F^2 \le \varepsilon^2] \\ &= \Pr[\|A^\top S^\top SA - I\|_F \le \varepsilon] \ge 1 - \delta.\end{aligned}$$

So, by Claim 2.17, S is a subspace embedding with probability at least $1 - \delta$.

2.6.2 Toward the matrix product result

It remains to prove Lemma 2.18 for CountSketch. The proof goes through the following property of a distribution on sketching matrices.

Definition 2.19 (JL moment property)**.** A distribution on matrices $S \in \mathbb{R}^{m\times n}$ has the $(\varepsilon, \delta, \ell)$-JL moment property if for all $x \in \mathbb{R}^n$ with $\|x\|_2 = 1$,

$$\mathop{\mathbb{E}}_{S}\Big[\big|\|Sx\|_2^2 - 1\big|^\ell\Big] \le \varepsilon^\ell \cdot \delta.$$

The plan is to show first that the JL moment property implies a matrix product result, and then that CountSketch has the JL moment property. Both steps are carried out in Lecture 3; the first is the following theorem.

Theorem 2.20 (From vectors to matrices)**.** *For $\varepsilon, \delta \in (0, 1/2)$, let $\mathcal{D}$ be a distribution on matrices S with m rows and n columns that has the $(\varepsilon, \delta, \ell)$-JL moment property for some $\ell \ge 2$. Then for matrices A, B with n rows,*

$$\Pr_S\Big[\|A^\top S^\top SB - A^\top B\|_F \ge 3\varepsilon\|A\|_F\|B\|_F\Big] \le \delta.$$

The proof of Theorem 2.20 uses a norm on random variables. For a random scalar X and $p \ge 1$, define

$$\|X\|_p = \big(\mathbb{E}[|X|^p]\big)^{1/p}.$$

We will also apply it to $X = \|T\|_F$ for a random matrix T, writing $\big\|\|T\|_F\big\|_p = (\mathbb{E}[\|T\|_F^p])^{1/p}$. This is a norm for $p \ge 1$; the triangle inequality is Minkowski's inequality, which we prove now.

Lemma 2.21 (Minkowski's inequality)**.** *For random variables X and Y and $p \geq 1$,*

$$\|X + Y\|_p \leq \|X\|_p + \|Y\|_p.$$

We will use Hölder's inequality.

Theorem 2.22 (Hölder's inequality)**.** *For vectors u, v and $p, q > 1$ with $\frac{1}{p} + \frac{1}{q} = 1$,*

$$\langle u, v \rangle \leq \|u\|_p \|v\|_q = \Big(\sum_i |u_i|^p\Big)^{1/p} \Big(\sum_i |v_i|^q\Big)^{1/q}.$$

More generally, for functions f, g and a measure μ,

$$\int |fg| \, d\mu \leq \Big(\int |f|^p \, d\mu\Big)^{1/p} \Big(\int |g|^q \, d\mu\Big)^{1/q}.$$

We use this form below, for random variables and their joint distribution.

Proof of Lemma 2.21. Suppose that $\|X\|_p$ and $\|Y\|_p$ are finite. We first show that $\|X + Y\|_p$ is finite. The function $f(x) = |x|^p$ is convex for $p \geq 1$, so $f(\frac{x+y}{2}) \leq \frac{1}{2} f(x) + \frac{1}{2} f(y)$, and for any fixed x and y,

$$\Big|\frac{1}{2}x + \frac{1}{2}y\Big|^p \leq \Big(\frac{1}{2}|x| + \frac{1}{2}|y|\Big)^p \leq \frac{1}{2}|x|^p + \frac{1}{2}|y|^p.$$

Multiplying by 2^p,

$$|x + y|^p \leq 2^{p-1}(|x|^p + |y|^p).$$

So $\mathbb{E}[|X + Y|^p] \leq 2^{p-1}(\mathbb{E}[|X|^p] + \mathbb{E}[|Y|^p])$ is finite, and since $\|X + Y\|_p^p = \mathbb{E}[|X + Y|^p]$, $\|X + Y\|_p$ is finite.

For $p = 1$, the inequality is the triangle inequality $|x + y| \leq |x| + |y|$, integrated, so we can assume $p > 1$. Let μ be the joint distribution of (X, Y), and write x, y for the values of X, Y. Now

$$\begin{aligned}\|X + Y\|_p^p &= \int |x + y|^p \, d\mu = \int |x + y| \cdot |x + y|^{p-1} \, d\mu \\ &\leq \int (|x| + |y|)|x + y|^{p-1} \, d\mu = \int |x||x + y|^{p-1} \, d\mu + \int |y||x + y|^{p-1} \, d\mu.\end{aligned}$$

Applying Hölder's inequality with exponents p for the first factor and $\frac{p}{p-1}$ for the second,

$$\begin{aligned}\int |x||x + y|^{p-1} \, d\mu &\leq \Big(\int |x|^p \, d\mu\Big)^{1/p} \Big(\int (|x + y|^{p-1})^{\frac{p}{p-1}} \, d\mu\Big)^{(p-1)/p}, \\ \int |y||x + y|^{p-1} \, d\mu &\leq \Big(\int |y|^p \, d\mu\Big)^{1/p} \Big(\int (|x + y|^{p-1})^{\frac{p}{p-1}} \, d\mu\Big)^{(p-1)/p}.\end{aligned}$$

So

$$\begin{aligned}\|X + Y\|_p^p &\leq \Big(\Big(\int |x|^p \, d\mu\Big)^{1/p} + \Big(\int |y|^p \, d\mu\Big)^{1/p}\Big)\Big(\int |x + y|^p \, d\mu\Big)^{(p-1)/p} \\ &= \Big((\mathbb{E}[|X|^p])^{1/p} + (\mathbb{E}[|Y|^p])^{1/p}\Big)\Big((\mathbb{E}[|X + Y|^p])^{1/p}\Big)^{p-1} = (\|X\|_p + \|Y\|_p)\|X + Y\|_p^{p-1}.\end{aligned}$$

If $\|X + Y\|_p = 0$ there is nothing to prove; otherwise, dividing both sides by the finite positive number $\|X + Y\|_p^{p-1}$ gives $\|X + Y\|_p \leq \|X\|_p + \|Y\|_p$. □

With Minkowski's inequality in hand, the next lecture proves Theorem 2.20, shows that CountSketch has the JL moment property, and so completes the proof of Lemma 2.18 (Lecture 3).

3 CountSketch, Affine Embeddings, and Low Rank Approximation

Lecture 3, September 24, 2020, by David P. Woodruff. Compiled from the scribe notes of Divyansh Jhunjhunwala (first half) and Dongyu Li and Arvind Mahankali (second half), and from the lecture slides.

In Lecture 2 we showed that CountSketch is a subspace embedding, assuming that CountSketch satisfies the JL moment property, which allowed us to use the approximate matrix product result. In this lecture we prove the two missing pieces:

- the JL moment property implies the approximate matrix product result (Lecture 3.1);
- CountSketch satisfies the JL moment property (Lecture 3.2).

We then introduce affine embeddings (Lecture 3.3) and apply them to low rank approximation (Lecture 3.4).

3.1 From vectors to matrices

We recall the definitions from Lecture 2.

Definition 3.1 (JL moment property)**.** A distribution on matrices $S \in \mathbb{R}^{m\times n}$ has the $(\varepsilon, \delta, \ell)$-*JL moment property* if for all $x \in \mathbb{R}^n$ with $\|x\|_2 = 1$,

$$\mathop{\mathbb{E}}_{S}\left[\left|\|Sx\|_2^2 - 1\right|^\ell\right] \le \varepsilon^\ell \cdot \delta.$$

The approximate matrix product result we are after says that for fixed matrices C and D of compatible dimensions (C has n columns and D has n rows), a random $S \in \mathbb{R}^{m\times n}$ satisfying the JL moment property, and a fixed δ,

$$\Pr\left[\|CS^\top SD - CD\|_F^2 \le \frac{c}{\delta m}\|C\|_F^2\|D\|_F^2\right] \ge 1 - \delta \tag{3.1}$$

for an absolute constant c (the argument below gives $c = 18$ rather than the $c = 6$ of Lemma 2.18; see Remark 3.8).

Recall from Lecture 2 that for a real random variable Z and $p \ge 1$, $\|Z\|_p = \left(\mathbb{E}|Z|^p\right)^{1/p}$.

Fact 3.2 (Minkowski's inequality; Lecture 2)**.** *For random variables Y, Z and $p \ge 1$, $\|Y + Z\|_p \le \|Y\|_p + \|Z\|_p$.*

So $\|\cdot\|_p$ satisfies the triangle inequality on random variables. To prove the approximate matrix product result we first prove the following theorem.

Theorem 3.3 (From vectors to matrices)**.** *For $\varepsilon, \delta \in (0, \frac{1}{2})$, let $\mathcal{D}$ be a distribution on matrices $S \in \mathbb{R}^{m\times n}$ that satisfies the $(\varepsilon, \delta, \ell)$-JL moment property for some $\ell \ge 2$. Then for matrices A, B with n rows,*

$$\Pr_{S}\left[\|A^\top S^\top SB - A^\top B\|_F \ge 3\varepsilon\|A\|_F\|B\|_F\right] \le \delta.$$

We first show that the JL moment property implies that S almost preserves inner products.

Lemma 3.4. *Under the hypotheses of Theorem 3.3, for all $x, y \in \mathbb{R}^n$,*

$$\left\|\langle Sx, Sy\rangle - \langle x, y\rangle\right\|_\ell \le 3\varepsilon \cdot \delta^{1/\ell}\|x\|_2\|y\|_2. \tag{3.2}$$

Proof. First let x and y be unit vectors. Note that $\|S(x-y)\|_2^2 = \|Sx\|_2^2 + \|Sy\|_2^2 - 2\langle Sx, Sy\rangle$ and $\|x-y\|_2^2 = 2 - 2\langle x, y\rangle$, so $2(\langle Sx, Sy\rangle - \langle x, y\rangle)$ is the expression inside the norm on the right-hand side of the first line below. Hence

$$\begin{aligned}
\big\|\langle Sx, Sy\rangle - \langle x, y\rangle\big\|_\ell &= \frac{1}{2}\Big\|(\|Sx\|_2^2 - 1) + (\|Sy\|_2^2 - 1) - (\|S(x-y)\|_2^2 - \|x-y\|_2^2)\Big\|_\ell \\
&\le \frac{1}{2}\Big[\big\|\|Sx\|_2^2 - 1\big\|_\ell + \big\|\|Sy\|_2^2 - 1\big\|_\ell + \big\|\|S(x-y)\|_2^2 - \|x-y\|_2^2\big\|_\ell\Big] \qquad (3.3)\\
&\le \frac{1}{2}\Big(\varepsilon\cdot\delta^{1/\ell} + \varepsilon\cdot\delta^{1/\ell} + \|x-y\|_2^2\,\varepsilon\cdot\delta^{1/\ell}\Big) \qquad (3.4)\\
&\le 3\varepsilon\cdot\delta^{1/\ell}, \qquad (3.5)
\end{aligned}$$

where (3.3) follows from the triangle inequality (Fact 3.2), (3.4) follows from the JL moment property (applied to x, to y, and to the unit vector $(x-y)/\|x-y\|_2$, which brings out the factor $\|x-y\|_2^2$), and (3.5) follows from bounding the squared norm of the difference of two unit vectors, $\|x-y\|_2^2 \le 4$.

For an arbitrary pair of vectors x, y, we normalize by the product of their norms, so that (3.5) still holds for $x/\|x\|_2$ and $y/\|y\|_2$; multiplying through by $\|x\|_2\|y\|_2$ gives (3.2). □

Proof of Theorem 3.3. We use Lemma 3.4 to bound the product of sketched matrices. Let A be any matrix with columns $A_{*,1}, \dots, A_{*,d}$ and B any matrix with columns $B_{*,1}, \dots, B_{*,d'}$. The (i,j)-th entry of the matrix $(SA)^\top(SB) - A^\top B$ is $\langle SA_{*,i}, SB_{*,j}\rangle - \langle A_{*,i}, B_{*,j}\rangle$. We can assume that no column of A or B is zero, since a zero column contributes nothing to the error. Define

$$Z_{i,j} = \frac{1}{\|A_{*,i}\|_2\|B_{*,j}\|_2}\big(\langle SA_{*,i}, SB_{*,j}\rangle - \langle A_{*,i}, B_{*,j}\rangle\big).$$

Then

$$\|(SA)^\top(SB) - A^\top B\|_F^2 = \sum_i\sum_j \|A_{*,i}\|_2^2\|B_{*,j}\|_2^2 Z_{i,j}^2.$$

For any $\ell \ge 2$ we can now write

$$\begin{aligned}
\Big\|\|(SA)^\top(SB) - A^\top B\|_F^2\Big\|_{\ell/2} &= \Big\|\sum_i\sum_j \|A_{*,i}\|_2^2\|B_{*,j}\|_2^2 Z_{i,j}^2\Big\|_{\ell/2} \\
&\le \sum_i\sum_j \big\|\|A_{*,i}\|_2^2\|B_{*,j}\|_2^2 Z_{i,j}^2\big\|_{\ell/2} \qquad (3.6)\\
&= \sum_i\sum_j \|A_{*,i}\|_2^2\|B_{*,j}\|_2^2\big\|Z_{i,j}^2\big\|_{\ell/2} \\
&= \sum_i\sum_j \|A_{*,i}\|_2^2\|B_{*,j}\|_2^2\|Z_{i,j}\|_\ell^2 \\
&\le \big(3\varepsilon\cdot\delta^{1/\ell}\big)^2\sum_i\sum_j \|A_{*,i}\|_2^2\|B_{*,j}\|_2^2 \qquad (3.7)\\
&= \big(3\varepsilon\cdot\delta^{1/\ell}\big)^2\|A\|_F^2\|B\|_F^2, \qquad (3.8)
\end{aligned}$$

where (3.6) is the triangle inequality (Fact 3.2 with $p = \ell/2 \ge 1$), the next two steps use $\|cY\|_p = |c|\|Y\|_p$ and $\|Z^2\|_{\ell/2} = (\mathbb{E}|Z|^\ell)^{2/\ell} = \|Z\|_\ell^2$, and (3.7) follows from (3.2).

Since $\mathbb{E}\left[\|A^\top S^\top SB - A^\top B\|_F^\ell\right] = \left\|\|A^\top S^\top SB - A^\top B\|_F^2\right\|_{\ell/2}^{\ell/2}$, we can write

$$\begin{aligned}
\Pr\left[\|A^\top S^\top SB - A^\top B\|_F \geq 3\varepsilon\|A\|_F\|B\|_F\right] &= \Pr\left[\|A^\top S^\top SB - A^\top B\|_F^\ell \geq (3\varepsilon\|A\|_F\|B\|_F)^\ell\right] \\
&\leq \left(\frac{1}{3\varepsilon\|A\|_F\|B\|_F}\right)^\ell \mathbb{E}\left[\|A^\top S^\top SB - A^\top B\|_F^\ell\right] \qquad (3.9) \\
&\leq \left(\frac{1}{3\varepsilon\|A\|_F\|B\|_F}\right)^\ell \left((3\varepsilon\cdot\delta^{1/\ell})^2\|A\|_F^2\|B\|_F^2\right)^{\ell/2} \qquad (3.10) \\
&= \delta,
\end{aligned}$$

where (3.9) is Markov's inequality and (3.10) follows from (3.8). □

In the next section we bound the number m of rows of a CountSketch in terms of ε^2 and δ, which recovers the approximate matrix product result.

3.2 CountSketch satisfies the JL moment property

For $\ell = 1$ the JL moment property asks that S approximately preserve squared norms. We prove the JL moment property for CountSketch with $\ell = 2$.

A CountSketch matrix $S \in \mathbb{R}^{m\times n}$ is described by two hash functions:

- $h\colon [n] \to [m]$, a pairwise independent hash function; $h(i)$ is the row index of the nonzero entry in the i-th column of S.
- $\sigma\colon [n] \to \{-1, 1\}$, a 4-wise independent hash function, independent of h; $\sigma(i)$ is the sign of the nonzero entry in the i-th column of S.

So $S_{h(i),i} = \sigma(i)$ and all other entries of S are zero. Pairwise independence of h means that each $h(i)$ is uniform on $[m]$ and $\Pr[h(i_1) = j_1 \wedge h(i_2) = j_2] = 1/m^2$ for all $i_1 \neq i_2$ and all j_1, j_2. For an event E, let $\mathbf{1}[E] = 1$ if E holds and $\mathbf{1}[E] = 0$ otherwise. Then $(Sx)_j = \sum_{i\in[n]} \mathbf{1}[h(i) = j]\,\sigma(i)x_i$.

Lemma 3.5. *For a CountSketch matrix S and every $x \in \mathbb{R}^n$, $\mathbb{E}\left[\|Sx\|_2^2\right] = \|x\|_2^2$.*

Proof.

$$\begin{aligned}
\mathbb{E}\left[\|Sx\|_2^2\right] &= \sum_{j\in[m]} \mathbb{E}\left[\Big(\sum_{i\in[n]} \mathbf{1}[h(i) = j]\,\sigma(i)x_i\Big)^2\right] \\
&= \sum_{j\in[m]} \sum_{i_1,i_2\in[n]} \mathbb{E}\left[\mathbf{1}[h(i_1) = j]\,\mathbf{1}[h(i_2) = j]\,\sigma(i_1)\sigma(i_2)\right] x_{i_1}x_{i_2} \qquad (3.11) \\
&= \sum_{j\in[m]} \sum_{i\in[n]} \mathbb{E}\left[\mathbf{1}[h(i) = j]^2\right] x_i^2 \qquad (3.12) \\
&= \sum_{j\in[m]} \frac{1}{m} \sum_{i\in[n]} x_i^2 = \|x\|_2^2.
\end{aligned}$$

Equation (3.11) follows from expanding the square and linearity of expectation. Equation (3.12) uses the following observations:

- h and σ are independent of each other;
- when $i_1 \neq i_2$, $\mathbb{E}[\sigma(i_1)\sigma(i_2)] = \mathbb{E}[\sigma(i_1)]\,\mathbb{E}[\sigma(i_2)]$;

- $\mathbb{E}[\sigma(i)] = 0$, and $\sigma(i)^2 = 1$ always.

So all cross terms in (3.11) vanish, leaving only the expression in (3.12). □

Thus CountSketch preserves norms in expectation.

Theorem 3.6 (CountSketch has the JL moment property)**.** *Let $S \in \mathbb{R}^{m\times n}$ be a CountSketch matrix. For every unit vector $x \in \mathbb{R}^n$, $\mathbb{E}\left[(\|Sx\|_2^2 - 1)^2\right] \le 2/m$. In particular, if $m \ge 2/(\varepsilon^2\delta)$, then S has the $(\varepsilon, \delta, 2)$-JL moment property.*

Proof. For $\ell = 2$ we have

$$\mathbb{E}\left[\left(\|Sx\|_2^2 - 1\right)^2\right] = \mathbb{E}\left[\|Sx\|_2^4\right] - 2\,\mathbb{E}\left[\|Sx\|_2^2\right] + 1. \tag{3.13}$$

We now bound $\mathbb{E}\left[\|Sx\|_2^4\right]$:

$$\begin{aligned}
\mathbb{E}\left[\|Sx\|_2^4\right] &= \mathbb{E}\left[\left(\|Sx\|_2^2\right)^2\right] \\
&= \mathbb{E}\Big[\sum_{j\in[m]}\sum_{j'\in[m]}\Big(\sum_{i\in[n]}\mathbf{1}[h(i)=j]\,\sigma(i)x_i\Big)^2\Big(\sum_{i'\in[n]}\mathbf{1}[h(i')=j']\,\sigma(i')x_{i'}\Big)^2\Big] \\
&= \sum_{j_1,j_2,i_1,i_2,i_3,i_4} \mathbb{E}\left[\mathbf{1}[h(i_1)=j_1]\,\mathbf{1}[h(i_2)=j_1]\,\mathbf{1}[h(i_3)=j_2]\,\mathbf{1}[h(i_4)=j_2]\right. \\
&\qquad\qquad \left.\cdot\,\sigma(i_1)\sigma(i_2)\sigma(i_3)\sigma(i_4)\right] x_{i_1}x_{i_2}x_{i_3}x_{i_4}.
\end{aligned}$$

By the 4-wise independence of σ (and its independence from h), the expectation of a term is zero unless the indices $\{i_1, i_2, i_3, i_4\}$ can be partitioned into equal pairs: if some index occurs exactly once, its sign has mean zero and is independent of the other signs. This leaves the following cases.

Case 1: $i_1 = i_2 = i_3 = i_4$. Since there is only one nonzero in each column of S, the term is nonzero only if $j_1 = j_2$. These terms give

$$\sum_{j\in[m]}\sum_{i\in[n]}\mathbb{E}\left[\mathbf{1}[h(i)=j]^4\right]x_i^4 = \sum_{j\in[m]}\sum_{i\in[n]}\Pr[h(i)=j]\,x_i^4 = \sum_{j\in[m]}\frac{1}{m}\sum_{i\in[n]}x_i^4 = \|x\|_4^4.$$

Case 2: $i_1 = i_2$, $i_3 = i_4$, $i_1 \ne i_3$. We get

$$\begin{aligned}
&\sum_{j_1,j_2\in[m]}\sum_{\substack{i_1,i_3\in[n]\\ i_1\ne i_3}}\mathbb{E}\left[\mathbf{1}[h(i_1)=j_1]^2\,\mathbf{1}[h(i_3)=j_2]^2\right]x_{i_1}^2x_{i_3}^2 \\
&\quad= \sum_{j_1,j_2\in[m]}\sum_{\substack{i_1,i_3\in[n]\\ i_1\ne i_3}}\Pr\left[h(i_1)=j_1\wedge h(i_3)=j_2\right]x_{i_1}^2x_{i_3}^2 \\
&\quad= \sum_{j_1,j_2\in[m]}\frac{1}{m^2}\sum_{\substack{i_1,i_3\in[n]\\ i_1\ne i_3}}x_{i_1}^2x_{i_3}^2 \qquad \text{(pairwise independence of } h\text{)} \\
&\quad= \sum_{i_1,i_3\in[n]}x_{i_1}^2x_{i_3}^2 - \sum_{i\in[n]}x_i^4 = \|x\|_2^4 - \|x\|_4^4.
\end{aligned}$$

Case 3: $i_1 = i_3$, $i_2 = i_4$, $i_1 \neq i_2$. Again $j_1 = j_2$ is necessary, by the same reasoning as in Case 1, which gives

$$\begin{aligned}
&\sum_{j\in[m]} \sum_{\substack{i_1,i_2\in[n]\\ i_1\neq i_2}} \mathbb{E}\left[\mathbf{1}[h(i_1)=j]^2\,\mathbf{1}[h(i_2)=j]^2\right] x_{i_1}^2 x_{i_2}^2 \\
&\quad= \sum_{j\in[m]} \sum_{\substack{i_1,i_2\in[n]\\ i_1\neq i_2}} \Pr\left[h(i_1)=j \wedge h(i_2)=j\right] x_{i_1}^2 x_{i_2}^2 \\
&\quad= \sum_{j\in[m]} \frac{1}{m^2} \sum_{\substack{i_1,i_2\in[n]\\ i_1\neq i_2}} x_{i_1}^2 x_{i_2}^2 \qquad \text{(pairwise independence of } h\text{)} \\
&\quad= \frac{1}{m} \sum_{\substack{i_1,i_2\in[n]\\ i_1\neq i_2}} x_{i_1}^2 x_{i_2}^2 \le \frac{1}{m} \sum_{i_1,i_2\in[n]} x_{i_1}^2 x_{i_2}^2 = \frac{1}{m}\|x\|_2^4.
\end{aligned}$$

Case 4: $i_1 = i_4$, $i_2 = i_3$, $i_1 \neq i_2$. This is equivalent to Case 3 and adds a further $\frac{1}{m}\|x\|_2^4$ to the total.

These four cases are disjoint and cover every way of partitioning $\{i_1, i_2, i_3, i_4\}$ into equal pairs. Adding them up, for a unit vector x we get $\mathbb{E}\left[\|Sx\|_2^4\right] \le (1 + \frac{2}{m})\|x\|_2^4 = 1 + \frac{2}{m}$ (and $\mathbb{E}\left[\|Sx\|_2^4\right] \ge \|x\|_2^4 = 1$, since Cases 1 and 2 add up to $\|x\|_2^4$ and Cases 3 and 4 are nonnegative). Together with $\mathbb{E}\left[\|Sx\|_2^2\right] = 1$ (Lemma 3.5), we can now bound (3.13):

$$\mathbb{E}\left[\left(\|Sx\|_2^2 - 1\right)^2\right] = \mathbb{E}\left[\|Sx\|_2^4\right] - 2\,\mathbb{E}\left[\|Sx\|_2^2\right] + 1 \le \left(1 + \frac{2}{m}\right) - 2 + 1 = \frac{2}{m}.$$

For the JL moment property with $\ell = 2$ we need this to be at most $\varepsilon^2\delta$, so we set $m = 2/(\varepsilon^2\delta)$ (or larger). □

Substituting $\varepsilon^2 = 2/(\delta m)$ into Theorem 3.3 gives the approximate matrix product result for CountSketch.

Corollary 3.7 (Approximate matrix product for CountSketch)**.** *Let $S \in \mathbb{R}^{m\times n}$ be a CountSketch matrix and $\delta \in (0, \frac{1}{2})$, with $m > 8/\delta$. For matrices A, B with n rows,*

$$\Pr\left[\|A^\top S^\top SB - A^\top B\|_F^2 \le \frac{18}{\delta m}\|A\|_F^2\|B\|_F^2\right] \ge 1 - \delta.$$

Proof. Let $\varepsilon = \sqrt{2/(\delta m)}$, so that $\varepsilon < \frac{1}{2}$ because $m > 8/\delta$. By Theorem 3.6, S has the $(\varepsilon, \delta, 2)$-JL moment property. So by Theorem 3.3, with probability at least $1 - \delta$,

$$\|A^\top S^\top SB - A^\top B\|_F^2 < 9\varepsilon^2\|A\|_F^2\|B\|_F^2 = \frac{18}{\delta m}\|A\|_F^2\|B\|_F^2.$$ □

Remark 3.8. Note that the argument above gives (3.1) with $c = 18$ rather than $c = 6$: squaring the bound $3\varepsilon\|A\|_F\|B\|_F$ of Theorem 3.3 gives $9\varepsilon^2$, not $3\varepsilon^2$. The constant does not matter for the application in Lecture 2. There we take $A = B = U$, where $U \in \mathbb{R}^{n\times d}$ has orthonormal columns spanning the column span of the data matrix, so that $U^\top U = I_d$ and $\|U\|_F^2 = d$. If $m \ge 18d^2/(\delta\varepsilon^2)$, then by Corollary 3.7, with probability at least $1 - \delta$,

$$\|U^\top S^\top SU - I_d\|_2 \le \|U^\top S^\top SU - I_d\|_F \le \varepsilon.$$

So CountSketch with $O(d^2/(\delta\varepsilon^2))$ rows is a subspace embedding with probability at least $1 - \delta$, as claimed in Lecture 2.

3.3 Affine embeddings

Consider the generalized regression problem

$$\min_X \|AX - B\|_F^2,$$

where $A \in \mathbb{R}^{n\times d}$ is a tall and thin matrix with d columns, and $B \in \mathbb{R}^{n\times d'}$ is a matrix with a large number d' of columns, potentially much larger than d; the variable is $X \in \mathbb{R}^{d\times d'}$.

We could solve the regression problem for each column of B separately and form the matrix X, but this is slow when B has many columns. We also cannot directly apply subspace embeddings, because the dimension of the relevant subspace (the column span of A together with the columns of B) grows with the number of columns of B.

Definition 3.9 (Affine embedding)**.** A matrix $S \in \mathbb{R}^{m\times n}$ is an *affine embedding* for A and B if

$$\|S(AX - B)\|_F^2 = (1 \pm \varepsilon)\|AX - B\|_F^2 \quad \text{for all } X \in \mathbb{R}^{d\times d'}.$$

Let us try to show $\|SAX - SB\|_F = (1 \pm \varepsilon)\|AX - B\|_F$ for all X, and see what properties we need of S. We work with the squared form of Definition 3.9. Just as for subspace embeddings, we can assume that A has orthonormal columns. Let X^* be the optimum, and let

$$B^* = AX^* - B.$$

The normal equations $A^\top(AX^* - B) = 0$ say that $A^\top B^* = 0$: the columns of B^* are orthogonal to the column span of A. We write $a \in b \pm c$ for $b - c \le a \le b + c$. We use two basic facts about the Frobenius norm.

Lemma 3.10. *(i) For two matrices $C, D \in \mathbb{R}^{p\times q}$, $\|C + D\|_F^2 = \|C\|_F^2 + \|D\|_F^2 + 2\operatorname{tr}(C^\top D)$.*

(ii) For matrices $C \in \mathbb{R}^{p\times q}$ and $D \in \mathbb{R}^{q\times p}$, $|\operatorname{tr}(CD)| \le \|C\|_F\|D\|_F$.

Proof. (i) For $i \in [q]$, let $C_{*,i}$ and $D_{*,i}$ be the i-th columns of C and D. Then

$$\begin{aligned}\|C + D\|_F^2 &= \sum_{i=1}^q \|C_{*,i} + D_{*,i}\|_2^2 = \sum_{i=1}^q \Big(\|C_{*,i}\|_2^2 + \|D_{*,i}\|_2^2 + 2\langle C_{*,i}, D_{*,i}\rangle\Big) \\ &= \sum_{i=1}^q \|C_{*,i}\|_2^2 + \sum_{i=1}^q \|D_{*,i}\|_2^2 + 2\sum_{i=1}^q \langle C_{*,i}, D_{*,i}\rangle = \|C\|_F^2 + \|D\|_F^2 + 2\operatorname{tr}(C^\top D),\end{aligned}$$

since the entry of $C^\top D$ in row i and column i is $\langle C_{*,i}, D_{*,i}\rangle$.

(ii) Writing $C_{i,*}$ for the rows of C and $D_{*,i}$ for the columns of D,

$$|\operatorname{tr}(CD)| = \Big|\sum_{i=1}^p \langle C_{i,*}, D_{*,i}\rangle\Big| \le \sum_{i=1}^p \|C_{i,*}\|_2\|D_{*,i}\|_2 \le \Big(\sum_{i=1}^p \|C_{i,*}\|_2^2\Big)^{1/2}\Big(\sum_{i=1}^p \|D_{*,i}\|_2^2\Big)^{1/2} = \|C\|_F\|D\|_F,$$

where the first inequality follows from the triangle inequality and the Cauchy–Schwarz inequality (applied to each of the inner products), and the second inequality follows from the Cauchy–Schwarz inequality as well, this time applied to the vectors $(\|C_{1,*}\|_2, \dots, \|C_{p,*}\|_2)$ and $(\|D_{*,1}\|_2, \dots, \|D_{*,p}\|_2)$. □

Theorem 3.11 (Sufficient conditions for an affine embedding)**.** *Let $A \in \mathbb{R}^{n\times d}$ have orthonormal columns, let $B \in \mathbb{R}^{n\times d'}$, let X^* and $B^* = AX^* - B$ be as above, and let $\varepsilon \in (0, 1)$. Suppose $S \in \mathbb{R}^{m\times n}$ satisfies:*

(a) S *is a subspace embedding for the column span of* A*:* $\|SAy\|_2^2 = (1 \pm \varepsilon)\|Ay\|_2^2$ *for all* $y \in \mathbb{R}^d$*;*

(b) $\|A^\top S^\top S B^*\|_F \le \varepsilon \|B^*\|_F$ *(this follows from the approximate matrix product result, see below);*

(c) $\|SB^*\|_F^2 = (1 \pm \varepsilon)\|B^*\|_F^2$.

Then $\|S(AX - B)\|_F^2 = (1 \pm 3\varepsilon)\|AX - B\|_F^2$ *for all* $X \in \mathbb{R}^{d \times d'}$.

Proof. Let us first analyze the expression $\|S(AX - B)\|_F^2 - \|SB^*\|_F^2$:

$$\begin{aligned}
\|S(AX - B)\|_F^2 - \|SB^*\|_F^2 &= \|SA(X - X^*) + S(AX^* - B)\|_F^2 - \|SB^*\|_F^2 \\
&= \|SA(X - X^*)\|_F^2 + \|S(AX^* - B)\|_F^2 \\
&\quad + 2\,\mathrm{tr}\left[(X - X^*)^\top A^\top S^\top S(AX^* - B)\right] - \|SB^*\|_F^2 && (3.14) \\
&= \|SA(X - X^*)\|_F^2 + 2\,\mathrm{tr}\left[(X - X^*)^\top A^\top S^\top S B^*\right] && (3.15) \\
&\in \|SA(X - X^*)\|_F^2 \pm 2\|X - X^*\|_F \|A^\top S^\top S B^*\|_F && (3.16) \\
&\in \|SA(X - X^*)\|_F^2 \pm 2\varepsilon \|X - X^*\|_F \|B^*\|_F && (3.17) \\
&\in \|A(X - X^*)\|_F^2 \\
&\quad \pm \varepsilon\Big(\|A(X - X^*)\|_F^2 + 2\|X - X^*\|_F \|B^*\|_F\Big). && (3.18)
\end{aligned}$$

The steps use the following facts:

- (3.14): Lemma 3.10(i).
- (3.15): $\|S(AX^* - B)\|_F^2 = \|SB^*\|_F^2$, because $B^* = AX^* - B$.
- (3.16): Lemma 3.10(ii).
- (3.17): assumption (b).
- (3.18): assumption (a), applied to each column of $A(X - X^*)$, which lies in the column span of A; so the Frobenius norm is preserved up to a multiplicative $(1 \pm \varepsilon)$.

Now consider the normal equations, analogous to those for least squares regression in Lecture 1: since the columns of B^* are orthogonal to the column span of A,

$$\|AX - B\|_F^2 = \|A(X - X^*)\|_F^2 + \|B^*\|_F^2. \tag{3.19}$$

We subtract the unsketched difference $\|AX - B\|_F^2 - \|B^*\|_F^2$ from the sketched difference:

$$\begin{aligned}
&\|S(AX - B)\|_F^2 - \|SB^*\|_F^2 - (\|AX - B\|_F^2 - \|B^*\|_F^2) \\
&\quad \in \pm\varepsilon\Big(\|A(X - X^*)\|_F^2 + 2\|X - X^*\|_F \|B^*\|_F\Big) \\
&\quad \in \pm\varepsilon\Big(\|A(X - X^*)\|_F^2 + 2\|A(X - X^*)\|_F \|B^*\|_F + \|B^*\|_F^2\Big) && (3.20) \\
&\quad = \pm\varepsilon\big(\|A(X - X^*)\|_F + \|B^*\|_F\big)^2 \\
&\quad \in \pm 2\varepsilon\Big(\|A(X - X^*)\|_F^2 + \|B^*\|_F^2\Big) && (3.21) \\
&\quad = \pm 2\varepsilon \|AX - B\|_F^2.
\end{aligned}$$

In (3.20) we used that A has orthonormal columns, so $\|X - X^*\|_F = \|A(X - X^*)\|_F$ (and added the nonnegative term $\|B^*\|_F^2$); in (3.21) we used $(a+b)^2 \le 2a^2 + 2b^2$, because $ab \le \frac{a^2+b^2}{2}$; the last step is (3.19).

Finally, by assumption (c),

$$\begin{aligned}\|S(AX-B)\|_F^2 &\in \|AX-B\|_F^2 + (\|SB^*\|_F^2 - \|B^*\|_F^2) \pm 2\varepsilon\|AX-B\|_F^2 \\ &\subseteq (1\pm 2\varepsilon)\|AX-B\|_F^2 \pm \varepsilon\|B^*\|_F^2 = (1\pm 2\varepsilon)\|AX-B\|_F^2 \pm \varepsilon\|AX^*-B\|_F^2 \\ &\subseteq (1\pm 3\varepsilon)\|AX-B\|_F^2.\end{aligned}$$

The last step holds because at the optimum X^*, $\|AX^* - B\|_F^2$ is at most $\|AX - B\|_F^2$ for any other X, by definition of being optimal. □

So S is an affine embedding if it satisfies these properties:

- S is a subspace embedding for the column span of A;
- S satisfies the approximate matrix product result;
- S preserves the Frobenius norm of the fixed matrix B^* up to a factor $1 \pm \varepsilon$, with constant probability.

For the second property, note that $\|A^\top S^\top SB^* - A^\top B^*\|_F^2 = \|A^\top S^\top SB^* - 0\|_F^2 = \|A^\top S^\top SB^*\|_F^2$, because the columns of B^* are orthogonal to the column span of A. If S satisfies the approximate matrix product result (3.1), then with constant probability $\|A^\top S^\top SB^*\|_F^2 \le O(\frac{1}{m})\|A\|_F^2 \cdot \|B^*\|_F^2$. Since A is an $n \times d$ matrix with orthonormal columns, $\|A\|_F^2 = d$. Therefore $\|A^\top S^\top SB^*\|_F^2 \le \varepsilon^2\|B^*\|_F^2$ once the number of rows of S is $m = O(d/\varepsilon^2)$ (with a large enough constant), and then $\|A^\top S^\top SB^*\|_F \le \varepsilon\|B^*\|_F$.

For the third property, $B^* = AX^* - B$ does not depend on X, so we only need it for one fixed matrix, which is not much to ask. CountSketch satisfies it with $O(1/\varepsilon^2)$ rows.

Lemma 3.12. *Let $B^* \in \mathbb{R}^{n\times d'}$ be a fixed matrix and $S \in \mathbb{R}^{m\times n}$ a CountSketch matrix. If $m = O(1/\varepsilon^2)$, then $\|SB^*\|_F^2 = (1\pm\varepsilon)\|B^*\|_F^2$ with constant probability.*

This lemma was Problem 3 of Homework 1 in Fall 2017. The key idea of the proof is to use Chebyshev's inequality to bound the failure probability, by first computing the expectation and the variance of $\|SB^*\|_F^2$.

Proof. For the expectation, by Lemma 3.5 applied to each column,

$$\mathbb{E}\left[\|SB^*\|_F^2\right] = \sum_{j\in[d']} \mathbb{E}\left[\|SB^*_{*,j}\|_2^2\right] = \sum_{j\in[d']} \|B^*_{*,j}\|_2^2 = \|B^*\|_F^2. \tag{3.22}$$

To bound the variance of $\|SB^*\|_F^2$, it is enough to compute $\mathbb{E}\left[\|SB^*\|_F^4\right]$. This is similar to the computation of $\mathbb{E}\left[\|Sx\|_2^4\right]$ in the proof of Theorem 3.6.

Write $b_i = B^*_{i,*}$ for the i-th row of B^*. Row j of SB^* is $\sum_{i\,:\,h(i)=j}\sigma(i)b_i$, so

$$\|SB^*\|_F^2 = \sum_{j\in[m]}\Big\|\sum_{i\,:\,h(i)=j}\sigma(i)b_i\Big\|_2^2 = \sum_{j\in[m]}\sum_{i,i'\,:\,h(i)=h(i')=j}\sigma(i)\sigma(i')\langle b_i, b_{i'}\rangle.$$

For any fixed h, taking expectation over σ gives $\mathbb{E}[\sigma(i)\sigma(i')] = 0$ unless $i = i'$, in which case $\mathbb{E}[\sigma(i)\sigma(i')] = 1$; this gives (3.22) again. We also have

$$\|SB^*\|_F^4 = \sum_{j_1,j_2\in[m]}\Big(\sum_{i_1,i_2\,:\,h(i_1)=h(i_2)=j_1}\sigma(i_1)\sigma(i_2)\langle b_{i_1}, b_{i_2}\rangle\Big)\Big(\sum_{i_3,i_4\,:\,h(i_3)=h(i_4)=j_2}\sigma(i_3)\sigma(i_4)\langle b_{i_3}, b_{i_4}\rangle\Big),$$

so $\mathbb{E}\left[\|SB^*\|_F^4\right]$ equals

$$\sum_{\substack{j_1,j_2\in[m]\\ i_1,i_2,i_3,i_4\in[n]}} \mathbb{E}\left[\mathbf{1}[h(i_1)=j_1]\,\mathbf{1}[h(i_2)=j_1]\,\mathbf{1}[h(i_3)=j_2]\,\mathbf{1}[h(i_4)=j_2]\right.$$
$$\left.\cdot\,\sigma(i_1)\sigma(i_2)\sigma(i_3)\sigma(i_4)\right]\langle b_{i_1},b_{i_2}\rangle\langle b_{i_3},b_{i_4}\rangle.$$

Taking expectation with respect to σ, a term can be nonzero only if $\{i_1,i_2,i_3,i_4\}$ can be partitioned into equal pairs. This drives the following cases.

Case: $j_1\neq j_2$. Then $\{i_1,i_2\}$ must be disjoint from $\{i_3,i_4\}$, since we cannot have $h(i)=j_1$ and $h(i)=j_2$ for one i. So $i_1=i_2$ and $i_3=i_4$ with $i_1\neq i_3$ are the only terms that contribute, and the total contribution of the terms with $j_1\neq j_2$ is

$$\sum_{\substack{j_1,j_2\in[m]\\ j_1\neq j_2}}\sum_{\substack{i_1,i_3\in[n]\\ i_1\neq i_3}}\frac{1}{m^2}\|b_{i_1}\|_2^2\|b_{i_3}\|_2^2 \le \|B^*\|_F^4-\sum_{i\in[n]}\|b_i\|_2^4.$$

Case: $j_1=j_2$ *and* $i_1=i_2=i_3=i_4$. The total contribution of these terms is

$$\sum_{j\in[m]}\sum_{i\in[n]}\frac{1}{m}\|b_i\|_2^4=\sum_{i\in[n]}\|b_i\|_2^4.$$

Case: $j_1=j_2$, *and* $i_1=i_2$, $i_3=i_4$, $i_1\neq i_3$. The total contribution of these terms is

$$\sum_{j\in[m]}\sum_{\substack{i_1,i_3\in[n]\\ i_1\neq i_3}}\frac{1}{m^2}\|b_{i_1}\|_2^2\|b_{i_3}\|_2^2\le\frac{1}{m}\|B^*\|_F^4.$$

Case: $j_1=j_2$, *and* $i_1=i_3$, $i_2=i_4$, $i_1\neq i_2$. The total contribution of these terms is

$$\sum_{j\in[m]}\sum_{\substack{i_1,i_2\in[n]\\ i_1\neq i_2}}\frac{1}{m^2}\langle b_{i_1},b_{i_2}\rangle^2\le\frac{1}{m}\|B^*\|_F^4,$$

by the Cauchy–Schwarz inequality $\langle b_{i_1},b_{i_2}\rangle^2\le\|b_{i_1}\|_2^2\|b_{i_2}\|_2^2$.

Case: $j_1=j_2$, *and* $i_1=i_4$, $i_2=i_3$, $i_1\neq i_2$. This case is the same as the previous one, and contributes at most $\frac{1}{m}\|B^*\|_F^4$.

In total, $\mathbb{E}\left[\|SB^*\|_F^4\right]\le\|B^*\|_F^4+\frac{3}{m}\|B^*\|_F^4$. Hence

$$\mathrm{Var}\left[\|SB^*\|_F^2\right]=\mathbb{E}\left[\|SB^*\|_F^4\right]-\mathbb{E}\left[\|SB^*\|_F^2\right]^2\le\frac{3}{m}\|B^*\|_F^4.$$

By Chebyshev's inequality,

$$\Pr\left[\left|\|SB^*\|_F^2-\|B^*\|_F^2\right|\ge\varepsilon\|B^*\|_F^2\right]\le\frac{3\|B^*\|_F^4/m}{\varepsilon^2\|B^*\|_F^4}\le\frac{1}{10}$$

for a suitably chosen $m=\Theta(1/\varepsilon^2)$ (any $m\ge 30/\varepsilon^2$). □

Remark 3.13. For CountSketch, property (a) of Theorem 3.11 holds with constant probability once $m=O(d^2/\varepsilon^2)$ (Remark 3.8), property (b) once $m=O(d/\varepsilon^2)$, and property (c) once $m=O(1/\varepsilon^2)$ (Lemma 3.12). We choose the constants so that each property fails with probability at most 1/10. Then by a union bound, a CountSketch with $O(d^2/\varepsilon^2)$ rows is an affine embedding with probability at least 7/10, after replacing ε by $\varepsilon/3$. Note that the number of rows does not depend on the number d' of columns of B.

3.4 Low rank approximation

We now consider an application of affine embeddings which arises often when dealing with large datasets. Consider a matrix $A \in \mathbb{R}^{n \times d}$, where n and d may both be large; think of its rows as n points in $\mathbb{R}^d$. For example, A is a customer-product matrix, where A_{ij} is how many times customer i purchased item j. In many cases, A may be approximated by a low rank matrix UV, where $U \in \mathbb{R}^{n \times k}$, $V \in \mathbb{R}^{k \times d}$, and $k \ll n, d$ (k is an upper bound on the rank of U and V).

This offers several advantages when k is small. First, the amount of space needed to store A decreases from $O(nd)$ to $O(nk+kd)$. In addition, multiplication of A by a vector $x \in \mathbb{R}^d$ can be done in $O(nk + kd)$ time, by first multiplying x by V and then by U. Finally, a low rank approximation may remove noise which had artificially increased the rank of A, and can improve the interpretability of the data.

3.4.1 Exact algorithms with the SVD

Consider the singular value decomposition $A = U\Sigma V^\top$. If it can be computed, then we can obtain a good rank-k approximation of A as follows. Let $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$ be the nonzero singular values of A, where r is the rank of A. Define Σ_k to be the diagonal matrix with $\sigma_1, \sigma_2, \ldots, \sigma_k$ on its diagonal. In addition, take $V_k^\top$ to be the matrix consisting of the first k rows of $V^\top$ (in other words, the top k right singular vectors, which are the top k principal components). Similarly, take U_k to be the matrix consisting of the k leftmost columns of U. Then $A_k = U_k \Sigma_k V_k^\top$ is a matrix of rank k (and $A = U_k \Sigma_k V_k^\top + E$ for a residual matrix E), which is in fact the best rank-k approximation to A in the sense that

$$A_k = \underset{\text{rank-}k \text{ matrices } A'}{\arg\min} \|A - A'\|_F$$

(here $\|C\|_F = \left(\sum_{i,j} C_{ij}^2\right)^{1/2}$). (This holds under other norms as well, in addition to the Frobenius norm.)

However, computing the SVD of A takes $O(nd^2)$ time (for $n \geq d$). To obtain faster algorithms, we relax the problem as we did for regression in Lecture 1. More specifically, our goal is to compute a rank-k matrix A' such that

$$\|A - A'\|_F \leq (1+\varepsilon)\|A - A_k\|_F,$$

with high probability. This can be done in $O(\text{nnz}(A) + (n+d)\,\text{poly}(k/\varepsilon))$ time, as shown in [Sar06] and [CW17]. This is a significant improvement, as even if A is dense, $\text{nnz}(A) = O(nd)$.

3.4.2 Low rank approximation with sketching

The idea is as follows. View the rows of A as points in $\mathbb{R}^d$. In addition, let $S \in \mathbb{R}^{m \times n}$ be a sketching matrix with $m = \text{poly}(k/\varepsilon) \ll n$ rows (we are perfectly fine with $\text{poly}(k/\varepsilon)$ terms in our running time). The rows of SA are linear combinations of the rows of A, so the row span of SA is a lower-dimensional subspace of the row span of A. From here, the algorithm proceeds as follows:

1. Compute SA, which takes $O(\text{nnz}(A))$ time if S is a CountSketch matrix. (S can be any of the random matrices we considered earlier: a $(k/\varepsilon) \times n$ matrix of i.i.d. normal random variables, the SRHT with $\widetilde{O}(k/\varepsilon)$ rows [Sar06], or a CountSketch matrix with $\text{poly}(k/\varepsilon)$ rows [CW17].)

2. Project the rows of A onto the row span of SA.

3. Find a best rank-k approximation of the projected rows (in other words, find a k-dimensional subspace inside the row span of SA that approximates the projected rows of A).

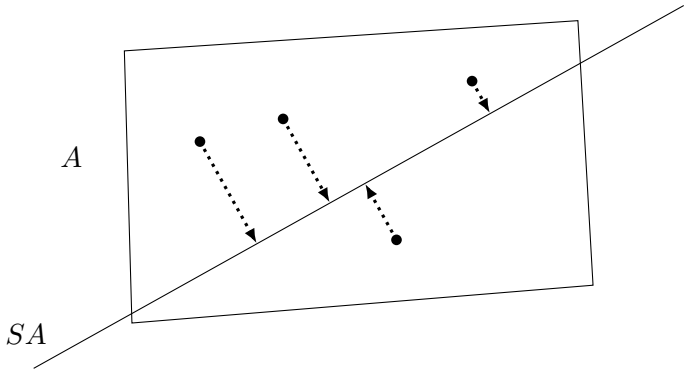


Figure 3: The rows of A (points) are projected onto the row span of SA (the line); then a best rank-k approximation of the projected points is found inside the row span of SA.

Figure 3 illustrates the second step.

To do this, we solve the optimization problem

$$\min_{\text{rank-}k\ X} \|XSA - A\|_F^2. \tag{3.23}$$

Why is this a useful objective function, that is, why does the row span of SA contain a good solution? Consider a different objective:

$$\min_X \|A_k X - A\|_F.$$

Clearly, this is minimized when X is the identity, since the columns of $A_k X$ lie in the column span of U_k, and projecting A onto that span gives $U_k U_k^\top A = A_k$. Now consider the sketched version of this objective, where S is an affine embedding for A_k and A (for instance, a CountSketch matrix):

$$\arg\min_X \|SA_k X - SA\|_F.$$

For all matrices X, $\|SA_k X - SA\|_F = (1 \pm \varepsilon)\|A_k X - A\|_F$.

We can solve the sketched objective using the normal equations, which give the minimizer $\hat{X} = (SA_k)^\dagger SA$. Why does this hold? The i-th column of $SA_k X$ is $SA_k X_{*,i}$, where $X_{*,i}$ is the i-th column of X. Therefore, we can independently choose $X_{*,i} = (SA_k)^\dagger (SA)_{*,i}$ for each i, where $(SA)_{*,i}$ is the i-th column of SA.

Now, since S is an affine embedding, this minimizer is an approximate solution to the objective $\|A_k X - A\|_F^2$; that is,

$$\|A_k (SA_k)^\dagger SA - A\|_F \le (1+\varepsilon)\|A_k - A\|_F. \tag{3.24}$$

To see this, note that

$$\|A_k \hat{X} - A\|_F \le \frac{1}{1-\varepsilon}\|SA_k \hat{X} - SA\|_F \le \frac{1}{1-\varepsilon}\|SA_k - SA\|_F \le \frac{1+\varepsilon}{1-\varepsilon}\|A_k - A\|_F,$$

where we used the affine embedding, then the optimality of $\hat{X}$ (compared with $X = I$), then the affine embedding again. Since $\frac{1+\varepsilon}{1-\varepsilon} = 1 + O(\varepsilon)$, replacing ε by a small constant multiple of itself gives (3.24). We do the same in (3.25) below.

This shows that our original objective (3.23) is a good one. Indeed, $A_k (SA_k)^\dagger SA$ is a matrix of rank at most k, and its rows are linear combinations of the rows of SA. Therefore,

$$\min_{\text{rank-}k\ X} \|XSA - A\|_F^2 \le \|A_k (SA_k)^\dagger SA - A\|_F^2 \le (1+\varepsilon)\|A - A_k\|_F^2, \tag{3.25}$$

and it is useful to find solutions X to our original objective.

We now solve the original objective. Let $P = (SA)^\dagger SA$, the orthogonal projection onto the row span of SA. The rows of $XSA - AP$ lie in the row span of SA, and the rows of $AP - A = -A(I-P)$ are orthogonal to it. Using the normal equations (the Pythagorean theorem) gives

$$\|XSA - A\|_F^2 = \|XSA - A(SA)^\dagger SA\|_F^2 + \|A(SA)^\dagger SA - A\|_F^2,$$

meaning

$$\min_{\text{rank-}k\ X} \|XSA - A\|_F^2 = \|A(SA)^\dagger SA - A\|_F^2 + \min_{\text{rank-}k\ X} \|XSA - A(SA)^\dagger SA\|_F^2.$$

Now we write $SA = U\Sigma V^\top$ in its *thin* SVD form, meaning that we remove all zero singular values from Σ, and remove the corresponding columns of U and of V. Say $\Sigma \in \mathbb{R}^{r' \times r'}$, where $r' = \text{rank}(SA)$. The second term of the above objective becomes

$$\begin{aligned}\min_{\text{rank-}k\ X} \|XSA - A(SA)^\dagger SA\|_F^2 &= \min_{\text{rank-}k\ X} \|XU\Sigma - A(SA)^\dagger U\Sigma\|_F^2 \\ &= \min_{\text{rank-}k\ Y} \|Y - A(SA)^\dagger U\Sigma\|_F^2,\end{aligned}$$

where the first equality is obtained by replacing SA with its thin SVD: we can ignore $V^\top$ because its rows are orthonormal, so multiplying by $V^\top$ on the right does not change the singular values of the matrix inside the Frobenius norm, and the Frobenius norm is determined by the singular values. The second equality holds because we can go back and forth between X and $Y = XU\Sigma \in \mathbb{R}^{n \times r'}$: if $\text{rank}(X) \le k$ then $\text{rank}(XU\Sigma) \le k$, and conversely every $Y \in \mathbb{R}^{n \times r'}$ with $\text{rank}(Y) \le k$ equals $XU\Sigma$ for $X = Y\Sigma^{-1}U^\top$, which has rank at most k (using $U^\top U = I$).

To compute the optimal Y, it suffices to compute the SVD of $A(SA)^\dagger U\Sigma$ and discard all but the k largest singular values. Even though we can compute the matrix SA in $\text{nnz}(A)$ time for a CountSketch matrix S, the matrix $(SA)^\dagger$ might be dense, which can make the computation of the matrix $A(SA)^\dagger U\Sigma$ very slow. In the next lecture (Lecture 4) we will see that we can use affine embeddings to solve this problem as well.

4 Fast Low Rank Approximation, High Precision Regression, and Leverage Score Sampling

Lecture 4, October 1, 2020, by David P. Woodruff. Compiled from the scribe notes of Lichen Zhang (first half) and Jie Jiao (second half), and from the lecture slides.

4.1 Recap: low rank approximation via sketching

Recall the *low rank approximation* problem from Lecture 3: given $A \in \mathbb{R}^{n\times d}$, find a matrix of rank at most k that minimizes the distance $\|A - A'\|_F$ over all matrices A' of rank at most k; the minimizer is A_k. For interpretation, one thinks of A as n data points, each in $\mathbb{R}^d$. In the last lecture we saw that A_k can be found from the SVD of A, but this takes $O(nd^2)$ time, so we turned to a sketching-based method. Take a CountSketch matrix S with $\text{poly}(k/\varepsilon) \ll n$ rows, and do the following:

- Compute SA, which takes $\text{nnz}(A)$ time.
- Project the rows of A onto the row span of SA.
- Find a best rank-k approximation of the projected rows, using a method such as the SVD.

In Lecture 3 we showed that, if S is an affine embedding for A_k and A, then

$$\min_{\text{rank}(X)\le k} \|XSA - A\|_F^2 \le (1+\varepsilon)\|A - A_k\|_F^2, \tag{4.1}$$

and that solving this problem is equivalent to solving

$$\min_{\text{rank}(Y)\le k} \|Y - A(SA)^\dagger U\Sigma\|_F,$$

where $SA = U\Sigma V^\top$ is the thin SVD of SA (all zero singular values are removed from Σ, together with the corresponding columns of U and V). One can solve this problem by computing the SVD of the matrix $A(SA)^\dagger U\Sigma$. However, computing the product of A and $(SA)^\dagger U\Sigma$ might take up to $\text{nnz}(A)\cdot\text{poly}(k/\varepsilon)$ time, which is too expensive. We now give an algorithm that takes only $\text{nnz}(A) + (n+d)\cdot\text{poly}(k/\varepsilon)$ time to solve this problem.

4.2 Approximate projection via affine embeddings

The computational bottleneck of the scheme above is projecting the rows of A onto the row span of SA. The projection problem is just regression, so the problem reduces to finding a fast algorithm for approximately solving the regression problem

$$\min_{\text{rank}(X)\le k} \|X(SA) - A\|_F^2. \tag{4.2}$$

Readers might recognize this as exactly the affine embedding problem $\min_X \|AX - B\|_F^2$ of Lecture 3, with SA playing the role of A and A playing the role of B, after transposing, since $\|XSA - A\|_F = \|(SA)^\top X^\top - A^\top\|_F$. So we can choose an affine embedding R and solve the minimization problem

$$\min_{\text{rank}(X)\le k} \|X(SA)R - AR\|_F^2 \tag{4.3}$$

instead. Here the sketch R multiplies $XSA - A$ from the right, and we take $R \in \mathbb{R}^{d \times \text{poly}(k/\varepsilon)}$ to be the transpose of a CountSketch matrix. Recall that for R to be an affine embedding, it suffices that R have $\text{poly}(\text{rank}(SA)/\varepsilon)$ columns; since SA has $\text{poly}(k/\varepsilon)$ rows, $\text{rank}(SA) \leq \text{poly}(k/\varepsilon)$, and therefore R has $\text{poly}(k/\varepsilon)$ columns. Since R is an affine embedding, writing $m = \text{poly}(k/\varepsilon)$ for the number of rows of S,

$$\|XSAR - AR\|_F^2 = (1 \pm \varepsilon)\|XSA - A\|_F^2 \quad \text{for all } X \in \mathbb{R}^{n \times m}.$$

Why does it suffice to solve (4.3)? Let $\tilde{X}$ be an optimal solution of (4.3) and $\hat{X}$ an optimal solution of (4.2). Then

$$\|\tilde{X}SA - A\|_F^2 \leq \frac{\|\tilde{X}SAR - AR\|_F^2}{1-\varepsilon} \leq \frac{\|\hat{X}SAR - AR\|_F^2}{1-\varepsilon} \leq \frac{1+\varepsilon}{1-\varepsilon}\|\hat{X}SA - A\|_F^2 \leq \frac{(1+\varepsilon)^2}{1-\varepsilon}\|A - A_k\|_F^2,$$

where we used the affine embedding, the optimality of $\tilde{X}$, the affine embedding again, and (4.1). This is $(1 + O(\varepsilon))\|A - A_k\|_F^2$, so $\tilde{X}SA$ is a good rank-k approximation.

We now analyze the new optimization problem. By the normal equations, the optimal solution of the unconstrained version of (4.3) (with no rank constraint on X) is

$$X^* = AR(SAR)^\dagger,$$

and X^*SAR is the projection of the rows of AR onto the row span of SAR, so the rows of $AR - X^*SAR$ are orthogonal to that row span. Since the rows of $XSAR - X^*SAR$ lie in that row span, the Pythagorean theorem gives, for arbitrary X,

$$\|XSAR - AR\|_F^2 = \|X^*SAR - AR\|_F^2 + \|XSAR - X^*SAR\|_F^2,$$

and hence

$$\min_{\text{rank}(X) \leq k} \|XSAR - AR\|_F^2 = \|X^*SAR - AR\|_F^2 + \min_{\text{rank}(X) \leq k} \|XSAR - X^*SAR\|_F^2.$$

Expanding $\|XSAR - X^*SAR\|_F^2$, we get

$$\|XSAR - AR(SAR)^\dagger SAR\|_F^2 = \|Y - AR(SAR)^\dagger SAR\|_F^2 \qquad \text{for } Y = XSAR.$$

We have reduced the problem to

$$\min_{\text{rank}(Y) \leq k} \|Y - AR(SAR)^\dagger SAR\|_F^2,$$

which can be solved by computing the SVD of the matrix $AR(SAR)^\dagger SAR$. We claim that the optimal Y necessarily lies in the row span of SAR, since otherwise extra cost would be incurred, by the Pythagorean theorem. Therefore $Y = XSAR$ for some X, namely $X = Y(SAR)^\dagger$ (which has rank at most $\text{rank}(Y) \leq k$). Recall that our goal is to compute XSA, so we can first compute $X = Y(SAR)^\dagger$, and then $XSA = Y(SAR)^\dagger SA$.

We now consider the running time of the above algorithm. We can break it into the following steps:

- Compute AR, SAR, and $(SAR)^\dagger$.
- Compute $AR(SAR)^\dagger SAR$.

- Compute the thin SVD of $AR(SAR)^\dagger SAR$, and find Y.
- Compute $Y(SAR)^\dagger SA$.

Since R is the transpose of a CountSketch matrix, computing AR takes only $\text{nnz}(A)$ time, and computing SAR takes $\text{nnz}(SA) \le d \cdot \text{poly}(k/\varepsilon)$ time. To compute $(SAR)^\dagger$, we can compute its SVD and multiply the respective parts, which takes $\text{poly}(k/\varepsilon)$ time since $SAR \in \mathbb{R}^{\text{poly}(k/\varepsilon)\times\text{poly}(k/\varepsilon)}$. Computing $AR(SAR)^\dagger SAR$ takes at most $n \cdot \text{poly}(k/\varepsilon)$ time, and computing its SVD takes the same amount of time. Finally, we output $Y(SAR)^\dagger SA$ in factored form, and we are done. The total running time is

$$\text{nnz}(A) + d \cdot \text{poly}(k/\varepsilon) + n \cdot \text{poly}(k/\varepsilon) = \text{nnz}(A) + (n+d) \cdot \text{poly}(k/\varepsilon).$$

We summarize the algorithm in Algorithm 4.1.

Algorithm 4.1 Low rank approximation in $\text{nnz}(A) + (n+d) \cdot \text{poly}(k/\varepsilon)$ time [CW17]

1: Compute SA.
2: Compute SAR and AR.
3: Compute $Y = \arg\min_{\text{rank}(Y)\le k} \|Y - AR(SAR)^\dagger SAR\|_F^2$ using the SVD.
4: Output $Y(SAR)^\dagger SA$ in factored form.

The overall running time is $\text{nnz}(A) + (n+d) \cdot \text{poly}(k/\varepsilon)$.

4.3 High precision regression

Recall the approximate regression problem: given $A \in \mathbb{R}^{n\times d}$ and $b \in \mathbb{R}^n$, output x' such that

$$\|Ax' - b\|_2 \le (1+\varepsilon) \min_x \|Ax - b\|_2$$

with high probability. Using the sketch-and-solve paradigm with a CountSketch matrix as the subspace embedding, we achieve a running time of $\text{nnz}(A) + \text{poly}(d/\varepsilon)$ (Lectures 1 and 2). However, if ε is very small, the factor $\text{poly}(1/\varepsilon)$ can be unacceptably large. Our goal is an algorithm with running time $O(\text{nnz}(A) + \text{poly}(d)) \log(1/\varepsilon)$.

To get there, we introduce a gradient-based algorithm, which relies heavily on the following quantity.

Definition 4.1 (Condition number)**.** Let $A \in \mathbb{R}^{n\times d}$. The *condition number* of A, denoted by $\kappa(A)$, is

$$\kappa(A) = \frac{\sup_{\|x\|_2=1} \|Ax\|_2}{\inf_{\|x\|_2=1} \|Ax\|_2} = \frac{\sigma_{\max}(A)}{\sigma_{\min}(A)}.$$

The running time of our algorithm will depend on $\kappa(A)$: a large condition number implies a large running time. We will use sketching to reduce $\kappa(A)$ to $O(1)$.

4.3.1 A preconditioner from a small QR decomposition

Assume that A has full column rank d, so that the matrix R below is invertible. Pick a constant $\varepsilon_0 < 1$, and let S be an ε_0-subspace embedding for A. We compute a QR factorization of SA, so that $SA = QR^{-1}$, where Q has orthonormal columns and R is an invertible matrix. One way to compute a QR factorization is via the thin SVD $SA = U\Sigma V^\top$: here U has orthonormal columns, and $\Sigma V^\top$ has inverse $V\Sigma^{-1}$, so we may take $Q = U$ and $R = V\Sigma^{-1}$. Since it is a thin SVD, Σ

has no zero entry on its diagonal, so it is invertible. Also, V is a $d \times d$ orthogonal matrix, since $\text{rank}(SA) = \text{rank}(A) = d$: S is a subspace embedding, so $SAx = 0$ only if $Ax = 0$.

We make two remarks.

1. S has $\text{poly}(d/\varepsilon_0) = \text{poly}(d)$ rows, so SA is a $\text{poly}(d) \times d$ matrix.
2. $\kappa(SAR) = 1$. To see this, notice that $SAR = Q$, and Q has orthonormal columns, so all of its singular values are 1. So by multiplying SA on the right by R, we squeeze the condition number of SA to the smallest possible value.

This motivates the following claim.

Claim 4.2. $\kappa(AR) \leq \dfrac{1+\varepsilon_0}{1-\varepsilon_0}$.

Proof. Since S is a subspace embedding, we have the following facts.

- For all unit x,
$$(1-\varepsilon_0)\|ARx\|_2 \leq \|SARx\|_2 = \|Qx\|_2 = \|x\|_2 = 1.$$
- Similarly, for all unit x, $(1+\varepsilon_0)\|ARx\|_2 \geq \|SARx\|_2 = 1$.

We hence obtain, for all unit x,

$$\frac{1}{1+\varepsilon_0} \leq \|ARx\|_2 \leq \frac{1}{1-\varepsilon_0}. \tag{4.4}$$

Thus the condition number of AR is

$$\kappa(AR) = \frac{\sup_{\|x\|_2=1}\|ARx\|_2}{\inf_{\|x\|_2=1}\|ARx\|_2} \leq \frac{1/(1-\varepsilon_0)}{1/(1+\varepsilon_0)} = \frac{1+\varepsilon_0}{1-\varepsilon_0}. \qquad \square$$

By Claim 4.2, R is a good *preconditioner* for A, in the sense that it squeezes the condition number of AR to a small constant.

4.3.2 A constant factor solution and the iteration

With S in hand, a natural thing to do is to compute

$$x_0 = \arg\min_x \|SARx - Sb\|_2.$$

Let $x^* = \arg\min_x \|ARx - b\|_2$. Then x_0 gives a $1+O(\varepsilon_0)$ approximation to $\|ARx^* - b\|_2$ (as in sketch-and-solve, Lecture 1), and consequently Rx_0 gives a $1+O(\varepsilon_0)$ approximation to $\min_x \|Ax - b\|_2$, since R is invertible and so $\min_x \|Ax - b\|_2 = \min_y \|ARy - b\|_2$. For the first claim we take S to be a subspace embedding for the column span of $[A, b]$, which equals the column span of $[AR, b]$ since R is invertible. Starting from x_0, consider the following iterative scheme:

$$x_{t+1} \leftarrow x_t + R^\top A^\top (b - ARx_t). \tag{4.5}$$

We will show that $\|AR(x_t - x^*)\|_2$ decays at a rate of $O(\varepsilon_0)$. Combining this fact with the initial gap $\|AR(x_0 - x^*)\|_2$, we will show that $O(\log(1/\varepsilon))$ iterations suffice to get the desired error bound.

We first examine the vector $AR(x_{t+1} - x^*)$. Let $U\Sigma V^\top = AR$ be the SVD of AR. Then

$$\begin{aligned}
AR(x_{t+1} - x^*) &= AR(x_t + R^\top A^\top (b - ARx_t) - x^*) \\
&= ARx_t - ARx^* + ARR^\top A^\top b - ARR^\top A^\top ARx_t \\
&= AR(x_t - x^*) + ARR^\top A^\top ARx^* - ARR^\top A^\top ARx_t \\
&= (AR - ARR^\top A^\top AR)(x_t - x^*) \\
&= (U\Sigma V^\top - U\Sigma V^\top V\Sigma U^\top U\Sigma V^\top)(x_t - x^*) \\
&= (U\Sigma V^\top - U\Sigma^3 V^\top)(x_t - x^*) \\
&= U(\Sigma - \Sigma^3)V^\top (x_t - x^*).
\end{aligned}$$

The third equality holds since x^* satisfies the normal equations $(AR)^\top ARx^* = (AR)^\top b$.

What are the singular values of AR? In the proof of Claim 4.2, (4.4) in fact shows that all singular values of AR lie in $1 \pm O(\varepsilon_0)$, and $(1 \pm O(\varepsilon_0))^3 = 1 \pm O(\varepsilon_0)$; this means each diagonal entry of $\Sigma - \Sigma^3$ is a small value in the range $\pm O(\varepsilon_0)$. Now we are ready to bound the ℓ_2 norm of this vector:

$$\begin{aligned}
\|AR(x_{t+1} - x^*)\|_2 &= \|U(\Sigma - \Sigma^3)V^\top (x_t - x^*)\|_2 \\
&= \|(\Sigma - \Sigma^3)V^\top (x_t - x^*)\|_2 \\
&\le O(\varepsilon_0)\|V^\top (x_t - x^*)\|_2 \\
&\le \frac{O(\varepsilon_0)}{1 - O(\varepsilon_0)}\|\Sigma V^\top (x_t - x^*)\|_2 \\
&\le (1 + O(\varepsilon_0))O(\varepsilon_0)\|\Sigma V^\top (x_t - x^*)\|_2 \\
&= (O(\varepsilon_0)^2 + O(\varepsilon_0))\|U\Sigma V^\top (x_t - x^*)\|_2 \\
&= O(\varepsilon_0)\|AR(x_t - x^*)\|_2 \\
&\le \frac{1}{2}\|AR(x_t - x^*)\|_2.
\end{aligned}$$

The third step holds because a diagonal matrix scales each entry of a vector; the fourth step uses that every diagonal entry of Σ is at least $1 - O(\varepsilon_0)$; and the last step holds by choosing ε_0 so that the factor is at most $1/2$. We get a decay rate of $O(\varepsilon_0) \le \frac{1}{2}$ per step, so by induction on t,

$$\|AR(x_t - x^*)\|_2 \le 2^{-t}\|AR(x_0 - x^*)\|_2.$$

Picking $t = O(\log(1/\varepsilon))$, the discrepancy becomes

$$\|AR(x_t - x^*)\|_2 \le O(\varepsilon)\|AR(x_0 - x^*)\|_2.$$

Finally, let us compute $\|ARx_t - b\|_2^2$. By the normal equations, $ARx^* - b$ is orthogonal to the column span of AR, which contains $AR(x_t - x^*)$. This gives the first and third lines below.

$$\begin{aligned}
\|ARx_t - b\|_2^2 &= \|ARx^* - b\|_2^2 + \|AR(x_t - x^*)\|_2^2 \\
&\le \|ARx^* - b\|_2^2 + O(\varepsilon)\|AR(x_0 - x^*)\|_2^2 \\
&= \|ARx^* - b\|_2^2 + O(\varepsilon)\Big(\|ARx_0 - b\|_2^2 - \|ARx^* - b\|_2^2\Big) \\
&\le \|ARx^* - b\|_2^2 + O(\varepsilon)\Big((1 + O(\varepsilon_0))\|ARx^* - b\|_2^2 - \|ARx^* - b\|_2^2\Big) \\
&\le \|ARx^* - b\|_2^2 + \frac{1}{2}O(\varepsilon)\|ARx^* - b\|_2^2 \\
&= (1 + O(\varepsilon))\|ARx^* - b\|_2^2.
\end{aligned}$$

This shows that our iterative scheme finds a $1 + O(\varepsilon)$ approximation to the optimal solution. We output $x' = Rx_t$, so that $\|Ax' - b\|_2^2 = \|ARx_t - b\|_2^2 \leq (1 + O(\varepsilon)) \min_x \|Ax - b\|_2^2$.

4.3.3 Running time

We now analyze the running time of the algorithm.

- Compute SA: picking S as a CountSketch matrix, this takes $\text{nnz}(A)$ time.
- Compute R, which can be done by an SVD of SA in $\text{poly}(d)$ time. Notice that R is a $d \times d$ matrix.
- Compute SAR, which takes at most $\text{poly}(d)$ time.
- Compute x_0: one can simply solve the normal equations, which takes $\text{poly}(d)$ time.
- Iteratively compute $x_{t+1} \leftarrow x_t + R^\top A^\top (b - ARx_t)$. We have upper bounded the number of iterations by $O(\log(1/\varepsilon))$, so it suffices to analyze the running time of a single update. The idea is to perform the matrix-vector products one at a time, starting from Rx_t, then $A(Rx_t)$, and so on, so that each matrix-vector product takes either $\text{nnz}(A)$ or $\text{poly}(d)$ time, and the vector subtraction takes $n \leq \text{nnz}(A)$ time. In total, this process takes $O(\text{nnz}(A) + \text{poly}(d)) \log(1/\varepsilon)$ time.

Thus our algorithm takes $O(\text{nnz}(A) + \text{poly}(d)) \log(1/\varepsilon)$ time in total, as desired.

4.3.4 An alternative perspective: preconditioned gradient descent

We now give an interpretation of the high precision regression algorithm, which motivates its design. The idea is to view the objective function of the minimization problem as a *cost function*:

$$F(x) = \frac{1}{2}\|Ax - b\|_2^2.$$

So the problem is to minimize the cost function. The function F is convex, so a standard algorithm is to run *gradient descent* on F. The minimum satisfies the condition that the first derivative is 0:

$$\frac{dF}{dx} = \frac{1}{2}\frac{d(x^\top A^\top Ax - 2x^\top A^\top b + b^\top b)}{dx} = A^\top Ax - A^\top b = 0.$$

This gives the normal equations, and yields an alternative proof of the optimality of the normal equations. Instead of solving the normal equations directly, we iteratively improve the quality of x by moving in the negative direction of the gradient:

$$x_{t+1} = x_t - \gamma \frac{dF}{dx}(x_t),$$

where γ is the step size. Plugging in the expression for $\frac{dF}{dx}$, we get

$$x_{t+1} = x_t - \gamma A^\top (Ax_t - b).$$

A quick fact: if γ is sufficiently small, then this algorithm converges at a rate of $1 - 1/\kappa(A^\top A)$. This makes the algorithm almost impractical to run: since $(1 - 1/\kappa(A^\top A))^{\kappa(A^\top A)} \approx 1/e$, it requires around $\kappa(A^\top A) = \kappa(A)^2$ iterations to improve the solution by a constant factor, and if $\kappa(A)$ is

large, the scheme takes too long to converge. This is where the preconditioner R comes in: AR has a constant condition number, which means the quality of the solution improves by a constant factor after only a constant number of iterations. Moreover, by using sketching we find a good initial solution, which is only a constant factor away from the optimal solution; this further reduces the number of iterations required.

To summarize, our sketching-based high precision regression algorithm can be viewed as a preconditioned gradient descent algorithm. One big win from using sketching, especially CountSketch, is that we can efficiently find a high-quality preconditioner and compute a good initial point. It also allows us to use the step size $\gamma = 1$, which significantly improves the running time. Note that the update (4.5) is exactly $x_{t+1} = x_t - \nabla G(x_t)$ for $G(y) = \frac{1}{2}\|ARy - b\|_2^2$, whose gradient is $R^\top A^\top (ARy - b)$.

4.4 Leverage score sampling

All the subspace embedding matrices we have seen so far are given by families of random matrices, such as Gaussian matrices, CountSketch and the SRHT. One issue with these subspace embeddings is that the product SA is hard to interpret; another is that even though A might be a sparse matrix, SA might not be sparse. We now study a *non-oblivious* way to compute a subspace embedding matrix, which samples and rescales rows of the matrix A itself. So we can interpret the sampled rows as being *important* in some way, and we can also ensure that if A has sparse rows, then SA has sparse rows. Note that this is a non-oblivious construction: the distribution of S depends on A.

4.4.1 Leverage scores and the sampling matrix

Definition 4.3 (Leverage score sampling matrix)**.** Let $A = U\Sigma V^\top$ be an $n \times d$ matrix of rank d, written in its SVD.

- The i-th leverage score of A is $\ell_i = \|U_{i,*}\|_2^2$. Since $\sum_{i\in[n]} \ell_i = \|U\|_F^2 = d$, the vector $(\ell_1/d, \ldots, \ell_n/d)$ is a probability distribution.
- Let $(q_1, \ldots, q_n)$ be a distribution with $q_i \geq \beta \ell_i/d$ for all i, where β is a parameter. Note that necessarily $\beta \leq 1$, since $1 = \sum_i q_i \geq \beta \sum_i \ell_i/d = \beta$.
- Define the sampling matrix $S = D \cdot \Omega^\top$, where $D \in \mathbb{R}^{m\times m}$ is a diagonal rescaling matrix and $\Omega \in \mathbb{R}^{n\times m}$ is a sampling matrix; the number m of rows of S is a parameter that we choose to get good enough probability bounds.
- For each column j of Ω and D, independently and with replacement, pick a row index $i \in [n]$ with probability q_i, and set $\Omega_{i,j} = 1$ and $D_{j,j} = 1/\sqrt{q_i \cdot m}$. All other entries of Ω and D are 0.

We first argue that this construction does not depend on the choice of orthonormal basis for the column span of A.

Claim 4.4. *Let U and U' be two matrices with orthonormal columns that both span the column span of A. Then $\|e_i^\top U\|_2^2 = \|e_i^\top U'\|_2^2$ for all i.*

Proof. Since both U and U' have column span equal to that of A, we have $U = U'Z$ for some change of basis matrix $Z \in \mathbb{R}^{d\times d}$. Since U and U' each have orthonormal columns, Z is a rotation matrix (it has orthonormal rows and columns, and all of its singular values are 1). Indeed, suppose for contradiction that some singular value of Z is not equal to 1, $\sigma_i(Z) \neq 1$. Then for the corresponding unit right singular vector y of Z, we have $\|Zy\|_2 \neq 1$. However, $\|Uy\|_2 = 1$, which contradicts $\|Uy\|_2 = \|U'Zy\|_2 = \|Zy\|_2 \neq 1$. Hence $\|e_i^\top U\|_2^2 = \|e_i^\top U'Z\|_2^2 = \|e_i^\top U'\|_2^2$, because multiplying a row vector on the right by the orthogonal matrix Z preserves its norm. □

4.4.2 Leverage score sampling gives a subspace embedding

We now show that the sampling matrix S is a subspace embedding for the column span of A with high probability. To get the desired probability bound, we again use the matrix Chernoff bound, which was also used in Lecture 2.

Theorem 4.5 (Matrix Chernoff bound)**.** *Let $X_1, \dots, X_m$ be independent copies of a symmetric random matrix $X \in \mathbb{R}^{d\times d}$ with $\mathbb{E}[X] = 0$, $\|X\|_2 \le \gamma$, and $\|\mathbb{E}[X^\top X]\|_2 \le \sigma^2$. Let $W = \frac{1}{m}\sum_{j\in[m]} X_j$. Then for any $\varepsilon > 0$,*

$$\Pr\left[\|W\|_2 > \varepsilon\right] \le 2d \cdot \exp\Big(-\frac{m\varepsilon^2}{2\sigma^2 + 2\gamma\varepsilon/3}\Big),$$

where $\|W\|_2 = \sup_{x\neq 0}\|Wx\|_2/\|x\|_2$. Since W is symmetric, $\|W\|_2 = \sup_{\|x\|_2=1}|x^\top W x|$. Note that, as in Lecture 2, the denominator of the exponent is $2\sigma^2 + 2\gamma\varepsilon/3$ and not $\sigma^2 + \gamma\varepsilon/3$. The factor 2 is needed, and it changes only the constants below.

Let $i(j)$ denote the index of the row of U sampled in the j-th trial, so that $U_{i(j),*}$ is the j-th sampled row of U.

Claim 4.6. *The random matrices*

$$X_j = I_d - \frac{U_{i(j),*}^\top U_{i(j),*}}{q_{i(j)}}, \qquad j \in [m],$$

together with $\gamma = 1 + \frac{d}{\beta}$ and $\sigma^2 = \frac{d}{\beta} - 1$, satisfy the premise of the matrix Chernoff bound (Theorem 4.5).

Proof. Since the trials in the construction of S are independent, the X_j are independent copies of a symmetric $d \times d$ random matrix X. For each j,

$$\mathbb{E}[X_j] = I_d - \mathbb{E}\Big[\frac{U_{i(j),*}^\top U_{i(j),*}}{q_{i(j)}}\Big] = I_d - \sum_{i\in[n]} q_i \cdot \frac{U_{i,*}^\top U_{i,*}}{q_i} = I_d - I_d = 0,$$

since $\sum_i U_{i,*}^\top U_{i,*} = U^\top U = I_d$, as U has orthonormal columns. Also,

$$\|X_j\|_2 \le \|I_d\|_2 + \frac{\|U_{i(j),*}^\top U_{i(j),*}\|_2}{q_{i(j)}} \le 1 + \max_i \frac{\|U_{i,*}\|_2^2}{q_i} \le 1 + \frac{d}{\beta},$$

where the last inequality holds because our distribution has $q_i \ge \beta\|U_{i,*}\|_2^2/d$. In addition, for each j, using $U_{i,*}U_{i,*}^\top = \|U_{i,*}\|_2^2 = \ell_i$,

$$\begin{aligned}
\mathbb{E}[X_j^\top X_j] &= I_d - 2\,\mathbb{E}\Big[\frac{U_{i(j),*}^\top U_{i(j),*}}{q_{i(j)}}\Big] + \mathbb{E}\Big[\frac{U_{i(j),*}^\top U_{i(j),*} U_{i(j),*}^\top U_{i(j),*}}{q_{i(j)}^2}\Big] \\
&= \mathbb{E}\Big[\frac{U_{i(j),*}^\top U_{i(j),*} U_{i(j),*}^\top U_{i(j),*}}{q_{i(j)}^2}\Big] - I_d \\
&= \mathbb{E}\Big[\frac{\ell_{i(j)}\, U_{i(j),*}^\top U_{i(j),*}}{q_{i(j)}^2}\Big] - I_d = \sum_{i\in[n]} \frac{\ell_i}{q_i} U_{i,*}^\top U_{i,*} - I_d \\
&\preceq \frac{d}{\beta}\sum_{i\in[n]} U_{i,*}^\top U_{i,*} - I_d = \Big(\frac{d}{\beta} - 1\Big) I_d,
\end{aligned}$$

where $C \preceq D$ means that $x^\top C x \le x^\top D x$ for all x. Since $\mathbb{E}[X_j^\top X_j]$ is also positive semidefinite, this gives $\|\mathbb{E}[X_j^\top X_j]\|_2 \le \frac{d}{\beta} - 1$. □

Using the random matrices X_j of Claim 4.6, we now show that S is a subspace embedding.

Theorem 4.7 (Leverage score sampling gives a subspace embedding)**.** *Let $\varepsilon \in (0,1)$. If $m = \Theta(d \log d/(\beta\varepsilon^2))$ with a large enough constant, then with probability at least $1 - 1/\operatorname{poly}(d)$, $\|SAx\|_2^2 = (1 \pm \varepsilon)\|Ax\|_2^2$ for all $x \in \mathbb{R}^d$.*

Proof. Consider the SVD $A = U\Sigma V^\top$. Because U and A have the same column span, the claim is equivalent to showing $\|SUy\|_2^2 = (1 \pm \varepsilon)\|Uy\|_2^2 = (1 \pm \varepsilon)\|y\|_2^2$ for all $y \in \mathbb{R}^d$. We can show this by showing that $\|U^\top S^\top SU - I_d\|_2 \le \varepsilon$ with probability at least $1 - 1/\operatorname{poly}(d)$.

We use the matrix Chernoff bound on the mean-zero random matrices X_j defined above. By Claim 4.6, we have $\gamma = 1 + \frac{d}{\beta}$ and $\sigma^2 = \frac{d}{\beta} - 1$. By construction of S we have

$$U^\top S^\top SU = \frac{1}{m}\sum_{j=1}^{m} \frac{1}{q_{i(j)}} U_{i(j),*}^\top U_{i(j),*}.$$

This implies

$$W = \frac{1}{m}\sum_{j\in[m]} X_j = I_d - \frac{1}{m}\sum_{j=1}^{m} \frac{1}{q_{i(j)}} U_{i(j),*}^\top U_{i(j),*} = I_d - U^\top S^\top SU.$$

Since $\varepsilon < 1$ and $\beta \le 1 \le d$, we have $2\sigma^2 + 2\gamma\varepsilon/3 \le \frac{2d}{\beta} + \frac{2}{3}\cdot\frac{2d}{\beta} = \frac{10d}{3\beta}$. Hence the matrix Chernoff bound gives

$$\Pr\left[\|I_d - U^\top S^\top SU\|_2 > \varepsilon\right] \le 2d \cdot e^{-m\varepsilon^2\Theta(\beta/d)}. \tag{4.6}$$

Recall that we control the number m of trials, so we can set $m = \Theta(d \log d/(\beta\varepsilon^2))$ to get $\Pr\left[\|I_d - U^\top S^\top SU\|_2 > \varepsilon\right] \le 2d \cdot e^{-c \ln d} = 2d^{1-c}$, where the constant c can be made as large as we like by choosing the constant in m large enough; for $c > 2$ this is $1/\operatorname{poly}(d)$. More generally, (4.6) is at most δ once $m = \Theta(d \log(d/\delta)/(\beta\varepsilon^2))$. □

4.4.3 Fast computation of leverage scores

Naively, we need to compute the SVD of A to compute the leverage scores. Instead, suppose we compute SA for an ε_0-subspace embedding S, and then compute $SA = QR^{-1}$ as in Lecture 4.3. As shown in the first part of this lecture (Claim 4.2), $\kappa(SAR) = 1$ and $\kappa(AR) \le \frac{1+\varepsilon_0}{1-\varepsilon_0}$. Instead of computing the leverage score ℓ_i as a squared row norm of a matrix with orthonormal columns spanning the column span of A, we approximate it by the squared row norm of AR, and compute

$$\ell_i' = \|e_i^\top AR\|_2^2.$$

Claim 4.8. *$\ell_i = (1 \pm O(\varepsilon_0))\ell_i'$ for all $i \in [n]$.*

Proof. Since AR has the same column span as A, we can write a QR decomposition of AR as $AR = UT^{-1}$, where U has orthonormal columns and T is invertible. Since S is a subspace embedding for the column span of A, for any vector x we have

$$(1-\varepsilon_0)\|ARx\|_2 \le \|SARx\|_2 = \|x\|_2 \quad \text{and} \quad (1+\varepsilon_0)\|ARx\|_2 \ge \|SARx\|_2 = \|x\|_2.$$

Combining these two statements, for any vector x,

$$(1 \pm O(\varepsilon_0))\|x\|_2 = \|ARx\|_2 = \|UT^{-1}x\|_2 = \|T^{-1}x\|_2.$$

So all the singular values of T^{-1}, and hence of T, lie in the interval $[1 - O(\varepsilon_0), 1 + O(\varepsilon_0)]$, and $\|zT\|_2 = (1 \pm O(\varepsilon_0))\|z\|_2$ for every row vector z. Therefore, for the i-th leverage score (computed with the basis U, which is allowed by Claim 4.4),

$$\ell_i = \|e_i^\top U\|_2^2 = \|e_i^\top ART\|_2^2 = (1 \pm O(\varepsilon_0))\|e_i^\top AR\|_2^2 = (1 \pm O(\varepsilon_0))\ell_i'.$$ □

But note that R may be a dense matrix, so we cannot compute AR in $\mathrm{nnz}(A)$ time. Instead, we approximate the squared row norms of AR by those of ARG, where G is a $d \times O(\log n)$ matrix of i.i.d. normal random variables with variance $1/t$, where $t = O(\log n)$ is the number of columns of G. Here it suffices to set ε_0 to be a constant. Since G is Gaussian, for every fixed row vector z,

$$\Pr\left[\|zG\|_2^2 = \left(1 \pm \frac{1}{2}\right)\|z\|_2^2\right] \geq 1 - \frac{1}{n^2}.$$

This means we can instead set $\ell_i' = \|e_i^\top ARG\|_2^2$, and still get every ℓ_i' within a constant factor of ℓ_i with high probability (by a union bound over the n rows). Computing $\ell_i' = \|e_i^\top ARG\|_2^2$ only takes $(\mathrm{nnz}(A) + d^2)\log n$ time (first compute the $d \times O(\log n)$ matrix RG, then $A(RG)$).

To use these approximations for sampling, we set $q_i = \ell_i' / \sum_{i'} \ell_{i'}'$. If $c_1\ell_i \leq \ell_i' \leq c_2\ell_i$ for all i, then $\sum_{i'} \ell_{i'}' \leq c_2 d$, so $q_i \geq (c_1/c_2)\ell_i/d$. Thus q satisfies Definition 4.3 with the constant $\beta = c_1/c_2$, and Theorem 4.7 applies with $m = O(d \log d/\varepsilon^2)$. So with leverage score sampling done this way, we can solve regression in $\mathrm{nnz}(A)\log n + \mathrm{poly}(d(\log n)/\varepsilon)$ time.

5 Distributed Low Rank Approximation and Coresets

Lecture 5, October 8, 2020, by David P. Woodruff. Compiled from the scribe notes of Chris Seiler (first half) and Ziyi Guan (second half), and from the lecture slides. A few short passages marked as added are filled in from the lecture slides or supply missing steps.

So far we have seen sketching-based algorithms for computing rank-k approximations to an input matrix $A \in \mathbb{R}^{n\times d}$ (Lectures 3 and 4). However, there are instances in which A is too large to store on a single server; instead, A is distributed among s servers. In this case we would like to solve the same low rank approximation problem, but now modeling the cost of both computation *and* communication.

5.1 The distributed setting

Definition 5.1 (Arbitrary partition model)**.** In the *arbitrary partition model*, the matrix $A \in \mathbb{R}^{n\times d}$ is distributed among s servers as

$$A = A^1 + A^2 + \cdots + A^s,$$

where $A^t \in \mathbb{R}^{n\times d}$ denotes the matrix of server $t \in [s]$.

Example 5.2. There are s shops, each serving the same set of n customers and d products. The t-th shop tracks a matrix A^t, where A^t_{ij} is the number of times customer i bought product j at this particular shop. Then the customer-product matrix $A = A^1 + \cdots + A^s$ of the arbitrary partition model has entries that count the total number of times each customer purchased each product.

The arbitrary partition model is not the only setting in which we consider the problem; the following, less general, setting is often a useful model for distributed matrices.

Definition 5.3 (Row partition model)**.** In the *row partition model*, each of the s servers holds a subset of the rows of A. Typically A is given by the block matrix

$$A = \begin{bmatrix} A^1 \\ A^2 \\ \vdots \\ A^s \end{bmatrix}.$$

In terms of Example 5.2, the row partition model is the special case in which each customer shops in only one shop.

5.1.1 The communication model

To solve distributed low rank approximation we need a way to account for communication costs. Typically we assume the following model.

Definition 5.4 (Coordinator model)**.** In the *coordinator model*, each of the s servers may communicate in both directions with a coordinator node, and with no one else; see Figure 4.

Remark 5.5. A similarly defined *peer-to-peer* model allows communication between any two nodes. There are two reasons we only consider the coordinator model. First, communication in a peer-to-peer setting is difficult to analyze: there are $\binom{s}{2}$ links, and communication is not necessarily synchronized. Second, we do not lose much power in the coordinator model: we can simulate the peer-to-peer model up to a factor of 2 in the number of messages, plus an additive $O(\log s)$ bits per message. Each server simply sends its message to the coordinator, specifying the target server in $O(\log s)$ bits, and the coordinator forwards the message.

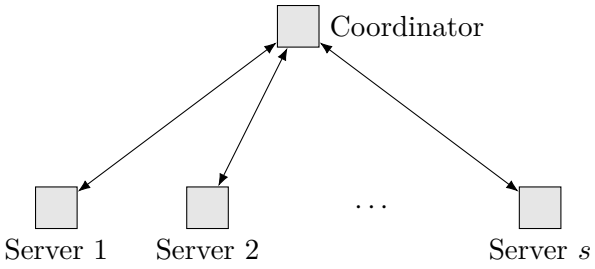


Figure 4: The communication allowed in the coordinator model.

5.1.2 Communication cost of low rank approximation

- **Input:** a matrix $A \in \mathbb{R}^{n \times d}$ stored on s servers in the arbitrary partition model. Server t holds a matrix $A^t \in \mathbb{R}^{n \times d}$, and the entries of each A^t are $O(\log(nd))$-bit integers.
- **Output:** every server must output the same k-dimensional subspace $W \subseteq \mathbb{R}^d$. If $P \in \mathbb{R}^{d \times d}$ denotes the projection matrix onto W, the resulting approximation is

$$C = A^1 P + A^2 P + \cdots + A^s P = AP,$$

  the projection of the rows of A onto W. We want C to be a good estimate of the best rank-k approximation A_k of A; precisely, we want

$$\|A - C\|_F \leq (1 + \varepsilon)\|A - A_k\|_F$$

  for an accuracy parameter ε.
- **Resources:** we want to minimize the total communication (in bits, or in words of $O(\log(nd))$ bits as in the bounds below), as well as the total computation. We also want a protocol with $O(1)$ *round complexity*, the number of back-and-forth rounds of messages needed to compute the answer. Ideally, we also want the computation to run in input sparsity time. One application of distributed low rank approximation is k-means clustering.

5.1.3 Work on distributed low rank approximation

There are several existing approaches to the problem.

- The [FSS20] protocol applies to the row partition model and uses $O(sdk/\varepsilon)$ *real numbers* of communication. Although the communication does not depend on n, communicating real numbers is generally avoided, since an arbitrary real number can encode an unbounded amount of information. With a fixed precision, the bit complexity can be large. Furthermore, [FSS20] needs the running time of an SVD of the input A (see also [BKLW14]).
- The [KVW14] protocol uses $O(skd/\varepsilon)$ words of communication in the arbitrary partition model, and has a faster running time with respect to n. However, [KVW14] is still not optimal in its communication: we would like to remove the ε from the denominator of the leading term skd.
- The [BWZ16] protocol uses $O(skd)+\text{poly}(sk/\varepsilon)$ words of communication, again in the arbitrary partition model, improving on the leading term of [KVW14]. Furthermore, its computation can be done in input sparsity time.

Remark 5.6. The [BWZ16] protocol has a matching lower bound on the leading term of the communication: $\Omega(skd)$ words of communication are required in general. Intuitively, if the subspace W is represented by a $k \times d$ matrix with orthonormal rows (which we also call W), it is possible that most servers have little knowledge of W. Since each server must learn W, loosely speaking the kd entries of W have to be sent to each of the s servers. This idea can be formalized to prove the bound.

Remark 5.7. Several variants of low rank approximation are considered in the literature. The [BLS+16] protocol handles kernel low rank approximation, which seeks a rank-k approximation to A after applying a "kernel map" $f\colon \mathbb{R}^d \to \mathbb{R}^{d'}$ to each row of A. Other variants concern implicit matrices [WZ16] and sparsity [BWZ16].

We now go through three protocols in turn: [FSS20] (Lectures 5.2 to 5.4), [KVW14] (Lecture 5.5) and [BWZ16] (Lecture 5.6).

5.2 Constructing a coreset [FSS20]

The first protocol, [FSS20], is built on the following object. Let $A \in \mathbb{R}^{n\times d}$ and let $A = U\Sigma V^\top$ be its SVD. For an integer m, let Σ_m agree with Σ on the first m diagonal entries (the m largest singular values) and be 0 otherwise, so that $A_m = U\Sigma_m V^\top$ is a best rank-m approximation of A.

Definition 5.8 (Coreset)**.** The matrix $\Sigma_m V^\top$, together with the scalar $c = \|A - A_m\|_F^2$, is called a *coreset* for A.

Throughout, "projection" means orthogonal projection ($P^2 = P = P^\top$). Below, X is the projection onto a k-dimensional subspace of $\mathbb{R}^d$, and $Y = I_d - X$ is the projection onto its orthogonal complement, a $(d-k)$-dimensional subspace.

Claim 5.9 ([FSS20])**.** *Let $m = k + k/\varepsilon$. For all projection matrices $Y = I_d - X$ onto $(d-k)$-dimensional subspaces,*

$$\|AY\|_F^2 \le \|\Sigma_m V^\top Y\|_F^2 + c \le (1+\varepsilon)\|AY\|_F^2,$$

where $c = \|A - A_m\|_F^2$ does not depend on Y.

We can assume that k/ε is an integer, since rounding it up only helps in the proof below.

Before proving this claim, we make a few observations. The coreset satisfies $\Sigma_m V^\top = \Sigma_m V_m^\top$, where $V_m^\top$ is $V^\top$ with all but its first m rows zeroed out, since Σ_m has only m nonzero diagonal entries. A consequence of Claim 5.9 is that it suffices to know $\Sigma_m V_m^\top$, which has only $md \ll nd$ parameters (together with the single number c), to answer queries about $\|AY\|_F^2$ up to a factor $1+\varepsilon$. The guarantee is similar to what we have seen for sketching matrices. Indeed, we can think of the sketching matrix as $S = U_m^\top$, where $U_m \in \mathbb{R}^{n\times m}$ consists of the first m columns of U. Then $SA = U_m^\top U\Sigma V^\top = \Sigma_m V^\top$ is the sketch, which is exactly the coreset (up to the zero rows of $\Sigma_m V^\top$, which SA omits).

One final observation: let $\tilde{Y}$ be a projection onto a $(d-k)$-dimensional subspace that minimizes $\|\Sigma_m V^\top Y\|_F^2$, and let Y^* be one that minimizes $\|AY\|_F^2$. Then

$$\begin{aligned}
\|A\tilde{Y}\|_F^2 &\le \|\Sigma_m V^\top \tilde{Y}\|_F^2 + c && (5.1)\\
&\le \|\Sigma_m V^\top Y^*\|_F^2 + c && (5.2)\\
&\le (1+\varepsilon)\|AY^*\|_F^2 && (5.3)\\
&= (1+\varepsilon)\|A - A_k\|_F^2. && (5.4)
\end{aligned}$$

Inequalities (5.1) and (5.3) follow from Claim 5.9, and (5.2) holds because $\tilde{Y}$ is the minimizer. For (5.4), write $Y^* = I_d - X^*$, where X^* is the projection onto the complementary k-dimensional subspace; since X^* is a k-dimensional projection minimizing $\|A - AX\|_F^2 = \|AY\|_F^2$, we have $AX^* = A_k$.

The following lemma is used in the proof of Claim 5.9.

Lemma 5.10. *A projection matrix P does not increase lengths: for every matrix A, $\|AP\|_F^2 \le \|A\|_F^2$.*

Proof. Recall that all singular values of a projection matrix are 0 or 1. For any matrices A and B,

$$\|AB\|_F^2 = \sum_{\text{rows } i} \|A_{i,*}B\|_2^2 \le \sum_{\text{rows } i} \|A_{i,*}\|_2^2 \|B\|_2^2 = \|A\|_F^2 \|B\|_2^2, \tag{5.5}$$

by the definition of the operator norm. For $B = P$ the operator norm is $\|P\|_2^2 = \sigma_{\max}(P)^2 \le 1$, and the lemma follows. □

Proof of Claim 5.9. First we show $\|AY\|_F^2 \le \|\Sigma_m V^\top Y\|_F^2 + c$. Write

$$\|AY\|_F^2 = \|U\Sigma_m V^\top Y + U(\Sigma - \Sigma_m)V^\top Y\|_F^2.$$

The first term only involves the first m columns of U, and the second term only involves the remaining columns. Since U has orthonormal columns, the columns of the two terms are orthogonal, and by the Pythagorean theorem

$$\|AY\|_F^2 = \|U\Sigma_m V^\top Y\|_F^2 + \|U(\Sigma - \Sigma_m)V^\top Y\|_F^2.$$

Multiplying by U, which has orthonormal columns, does not change the Frobenius norm. In the second term Y is a projection matrix, so by Lemma 5.10,

$$\|AY\|_F^2 \le \|\Sigma_m V^\top Y\|_F^2 + \|U(\Sigma - \Sigma_m)V^\top\|_F^2 = \|\Sigma_m V^\top Y\|_F^2 + \|A - A_m\|_F^2 = \|\Sigma_m V^\top Y\|_F^2 + c,$$

as desired.

Now we prove the second inequality, $\|\Sigma_m V^\top Y\|_F^2 + c \le (1+\varepsilon)\|AY\|_F^2$. Subtracting $\|AY\|_F^2$ from both sides and using the definition of c, it suffices to show

$$\|\Sigma_m V^\top Y\|_F^2 + \|A - A_m\|_F^2 - \|AY\|_F^2 \le \varepsilon \|AY\|_F^2.$$

Since $Y = I_d - X$, we have $MY + MX = M$ for every matrix M. Furthermore, X and Y project onto complementary subspaces, so X and Y have orthogonal rows, and we can apply the Pythagorean theorem: $\|MY\|_F^2 = \|M\|_F^2 - \|MX\|_F^2$. To see this, note that $XY = X - X^2 = 0$, so $(MX)(MY)^\top = MXYM^\top = 0$, and every row of MX is orthogonal to every row of MY. Applying this with $M = \Sigma_m V^\top$ and with $M = A$,

$$\begin{aligned}
&\|\Sigma_m V^\top Y\|_F^2 + \|A - A_m\|_F^2 - \|AY\|_F^2 \\
&\quad = \|\Sigma_m V^\top\|_F^2 - \|\Sigma_m V^\top X\|_F^2 + \|A - A_m\|_F^2 - \|A\|_F^2 + \|AX\|_F^2 \\
&\quad = \|U\Sigma_m V^\top\|_F^2 - \|\Sigma_m V^\top X\|_F^2 + \|A - A_m\|_F^2 - \|A\|_F^2 + \|AX\|_F^2 \\
&\quad = \|A_m\|_F^2 - \|\Sigma_m V^\top X\|_F^2 + \|A - A_m\|_F^2 - \|A\|_F^2 + \|AX\|_F^2,
\end{aligned}$$

using first that U has orthonormal columns and then the definition of A_m. Now we apply the Pythagorean theorem again, noting that $A_m + (A - A_m) = A$ and that A_m and $A - A_m$ have orthogonal columns, so $\|A_m\|_F^2 + \|A - A_m\|_F^2 = \|A\|_F^2$:

$$\begin{aligned}
\|\Sigma_m V^\top Y\|_F^2 + \|A - A_m\|_F^2 - \|AY\|_F^2 &= \|AX\|_F^2 - \|\Sigma_m V^\top X\|_F^2 \\
&= \|U\Sigma V^\top X\|_F^2 - \|U\Sigma_m V^\top X\|_F^2 \\
&= \|U(\Sigma - \Sigma_m)V^\top X\|_F^2,
\end{aligned}$$

where we once again used orthogonality and that U has orthonormal columns. Applying (5.5) to transposes, we get for all matrices C and D

$$\|CD\|_F = \|D^\top C^\top\|_F \le \|D^\top\|_F \|C^\top\|_2 = \|C\|_2 \|D\|_F.$$

Hence

$$\|U(\Sigma - \Sigma_m)V^\top X\|_F^2 \le \|U(\Sigma - \Sigma_m)V^\top\|_2^2 \, \|X\|_F^2.$$

The first factor is in SVD form, so its largest singular value, which equals its operator norm, is σ_{m+1}. In the second factor, X is a rank-k projection matrix, which has exactly k singular values equal to 1 (and the rest 0). Therefore

$$\begin{aligned}
\|U(\Sigma - \Sigma_m)V^\top\|_2^2 \, \|X\|_F^2 = \sigma_{m+1}^2 \sum_{i=1}^{k} 1^2 &= \sigma_{m+1}^2 k \\
&\le \varepsilon \sigma_{m+1}^2 (m-k) \\
&\le \varepsilon \sum_{i=k+1}^{m} \sigma_i^2 \\
&\le \varepsilon \sum_{i=k+1}^{d} \sigma_i^2 = \varepsilon \|A - A_k\|_F^2 \\
&\le \varepsilon \|AY\|_F^2,
\end{aligned}$$

as desired. The second line uses the definition of m, the third uses that the singular values are nonincreasing down the diagonal, and the fourth uses properties of the Frobenius norm. The last line holds because $\|A - A_k\|_F^2 = \|AY^*\|_F^2$, where Y^* is a projection onto a $(d-k)$-dimensional subspace that minimizes $\|AY\|_F^2$. □

5.3 Unions of coresets

Suppose we have matrices $A^1, \dots, A^s$ (with $A^t \in \mathbb{R}^{n_t \times d}$), and for each t we construct the coreset $\Sigma_m^t (V^t)^\top$ of A^t as in Lecture 5.2, together with $c^t = \|A^t - (A^t)_m\|_F^2$. Then, for the matrix A formed by concatenating the rows of $A^1, \dots, A^s$, we have

$$\sum_{t=1}^{s} \Big(\|\Sigma_m^t (V^t)^\top Y\|_F^2 + c^t \Big) = (1 \pm \varepsilon) \|AY\|_F^2.$$

Let B be the matrix obtained by concatenating the rows of $\Sigma_m^1 (V^1)^\top, \dots, \Sigma_m^s (V^s)^\top$. Suppose we compute the SVD $B = U_B \Sigma_B V_B^\top$, the coreset $(\Sigma_B)_m V_B^\top$ of B, and $c^B = \|B - B_m\|_F^2$. Then

$$\|(\Sigma_B)_m V_B^\top Y\|_F^2 + c^B + \sum_t c^t = (1 \pm \varepsilon)\|BY\|_F^2 + \sum_t c^t = (1 \pm O(\varepsilon))\|AY\|_F^2.$$

Thus $(\Sigma_B)_m V_B^\top$ together with the constant $c^B + \sum_t c^t$ is a coreset for A.

To see this, note that since the rows are concatenated, $\|AY\|_F^2 = \sum_t \|A^t Y\|_F^2$ and $\|BY\|_F^2 = \sum_t \|\Sigma_m^t (V^t)^\top Y\|_F^2$. Applying the one-sided bounds of Claim 5.9 first to each A^t and then to B, we get

$$\begin{aligned}
\|AY\|_F^2 &\le \|BY\|_F^2 + \sum_t c^t \le \|(\Sigma_B)_m V_B^\top Y\|_F^2 + c^B + \sum_t c^t \\
&\le (1+\varepsilon)\|BY\|_F^2 + \sum_t c^t \le (1+\varepsilon)\Big(\|BY\|_F^2 + \sum_t c^t \Big) \le (1+\varepsilon)^2 \|AY\|_F^2.
\end{aligned}$$

So the merged coreset satisfies the guarantee of Claim 5.9 with $(1+\varepsilon)^2 = 1 + O(\varepsilon)$ in place of $1+\varepsilon$, and the chain (5.1)–(5.4) applies to it with this factor.

5.4 The [FSS20] row partition protocol

With unions of coresets in hand, we obtain the protocol of Algorithm 5.1 for the row partition model.

Algorithm 5.1 The [FSS20] protocol (row partition model)

1: Server t sends the coordinator the top $k/\varepsilon + k$ principal components of A^t, scaled by the top $k/\varepsilon + k$ singular values of A^t (that is, the coreset $\Sigma_m^t (V^t)^\top$ with $m = k + k/\varepsilon$), together with c^t.
2: The coordinator stacks the received matrices into B and returns $c^B + \sum_t c^t$ and the top $k + k/\varepsilon$ principal components of B, scaled by its top $k + k/\varepsilon$ singular values (the coreset $(\Sigma_B)_m V_B^\top$ for A of Lecture 5.3).

Every server then outputs the span W of the top k principal components of B, which it can read off from $(\Sigma_B)_m V_B^\top$ since $m \geq k$. Let P be the projection onto W. The projection $I_d - P$ onto the orthogonal complement of W minimizes $\|(\Sigma_B)_m V_B^\top Y\|_F^2$, so the chain (5.1)–(5.4) for the merged coreset gives $\|A - AP\|_F^2 \leq (1 + O(\varepsilon))\|A - A_k\|_F^2$.

However, this protocol has several problems:

1. it uses sdk/ε real numbers of communication;
2. the bit complexity can be large;
3. it needs the running time of SVDs;
4. it does not work in the arbitrary partition model.

This is an SVD-based protocol. Maybe our random matrix techniques can improve communication just as they improved computation? The next protocol, [KVW14], handles some of these problems (problems 2, 3 and 4).

5.5 The [KVW14] arbitrary partition protocol

Now consider the arbitrary partition model, $A = A^1 + \cdots + A^s$. The [KVW14] protocol is inspired by the sketching algorithms for low rank approximation discussed earlier in the course (Lectures 3 and 4). Let $S \in \mathbb{R}^{(k/\varepsilon) \times n}$ be one of the random matrices discussed earlier, such as a Gaussian sketch or a CountSketch. The matrix S can be generated pseudorandomly from a small seed, and the coordinator sends this seed to all servers. Server t computes SA^t and sends it to the coordinator, and the coordinator sends $\sum_{t=1}^s SA^t = SA$ back to all servers. In this way every server knows the same S and the same SA, at a relatively small communication cost.

There is a good k-dimensional subspace inside the row span of SA; if we knew it, the t-th server could output the projection of A^t onto it. However, this approach has problems:

- We cannot output the projection of A^t onto the row span of SA, since the rank of SA is larger than k.
- The servers could communicate this projection to the coordinator, who could then find a k-dimensional subspace, but the communication depends on n. (Writing $SA = U\Sigma V^\top$, we would need the SVD of $A(SA)^\dagger U\Sigma$, and each $A^t(SA)^\dagger U\Sigma$ has n rows.) We do not want this, since n can be large.

To fix this, instead of projecting A onto the row span of SA, we solve

$$\min_{\text{rank-}k\ X} \|A(SA)^\top XSA - A\|_F^2, \tag{5.6}$$

over $X \in \mathbb{R}^{(k/\varepsilon)\times(k/\varepsilon)}$. This works because, from the low rank approximation algorithms discussed earlier, we know that

$$\min_{\text{rank-}k\ X} \|XSA - A\|_F^2 \le (1+\varepsilon)\|A - A_k\|_F^2$$

(here the minimum is over $X \in \mathbb{R}^{n\times(k/\varepsilon)}$). Therefore there is a good rank-k subspace inside the row span of SA. Let $Z \in \mathbb{R}^{k\times(k/\varepsilon)}$ be a rank-k matrix such that the rows of ZSA are orthonormal and form a basis of the best k-dimensional subspace inside the row span of SA. Then

$$\|A(ZSA)^\top ZSA - A\|_F^2 = \|A(SA)^\top Z^\top ZSA - A\|_F^2 \le (1+\varepsilon)\|A - A_k\|_F^2.$$

The inequality holds because $A(ZSA)^\top ZSA$ projects the rows of A onto the best k-dimensional subspace inside the row span of SA. For every rank-k matrix X, the rows of XSA lie in some subspace of dimension at most k inside the row span of SA, so the left side is at most $\min_{\text{rank-}k\ X}\|XSA-A\|_F^2$. Since $Z^\top Z$ has rank k, the minimum in (5.6) is at most $(1+\varepsilon)\|A - A_k\|_F^2$.

To solve (5.6), let T_1 and T_2 be affine embeddings, acting on the left and on the right, respectively, and solve

$$\min_{\text{rank-}k\ X} \|T_1A(SA)^\top XSAT_2 - T_1AT_2\|_F^2. \tag{5.7}$$

This optimization problem is small and has a closed form solution. Here the number of rows of T_1 needs to be

$$\text{poly}(\text{rank}(A(SA)^\top)/\varepsilon) = \text{poly}(k/\varepsilon),$$

and the number of columns of T_2 needs to be $\text{poly}(\text{rank}(SA)/\varepsilon) = \text{poly}(k/\varepsilon)$, so the computation is fast. This gives the protocol in Algorithm 5.2.

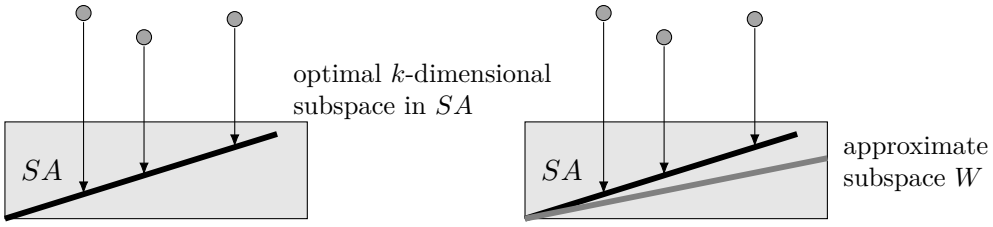
Algorithm 5.2 The [KVW14] protocol (arbitrary partition model)

1: Each server t sends SA^t to the coordinator ($s \cdot kd/\varepsilon$ words of communication).
2: The coordinator sums them and sends SA back to every server ($s \cdot kd/\varepsilon$ words).
3: Each server t sends $T_1A^t(SA)^\top$, SA^tT_2 and $T_1A^tT_2$ to the coordinator ($s \cdot \text{poly}(k/\varepsilon)$ words).
4: The coordinator sums them to get $T_1A(SA)^\top$, SAT_2 and T_1AT_2, and sends them back to all servers ($s \cdot \text{poly}(k/\varepsilon)$ words).
5: Each server solves (5.7) and outputs the row space of $(SA)^\top XSA$, which has dimension at most k since X has rank k.

We can think of the protocol as having two phases (Figure 5). In Phase 1 (steps 1–2), the servers learn the row space of SA, which contains a k-dimensional subspace of cost at most $(1+\varepsilon)\|A-A_k\|_F$. In Phase 2 (steps 3–5), they find an approximately optimal subspace W inside the row space of SA, of cost at most $(1+\varepsilon)^2\|A - A_k\|_F$. In total, [KVW14] uses $O(skd/\varepsilon) + s \cdot \text{poly}(k/\varepsilon)$ words of communication.

5.6 The [BWZ16] protocol

The main problem with [KVW14] is that its communication is $O(skd/\varepsilon) + \text{poly}(sk/\varepsilon)$, while we want $O(skd) + \text{poly}(sk/\varepsilon)$ communication, since d can be quite large in many scenarios. The idea used to obtain this is *projection-cost preserving sketches* [CEM+14].



Phase 1: cost $\leq (1+\varepsilon)\|A - A_k\|_F$ Phase 2: cost $\leq (1+\varepsilon)^2\|A - A_k\|_F$

Figure 5: The two phases of the [KVW14] protocol. The dots are rows of A; the black segment is the optimal k-dimensional subspace inside the row space of SA, and the gray segment is the approximately optimal subspace W found in Phase 2.

Definition 5.11 (Projection-cost preserving sketch)**.** Let A be an $n \times d$ matrix. A matrix $S \in \mathbb{R}^{(k/\varepsilon^2)\times n}$ is a *projection-cost preserving sketch* (PCP sketch) for A if there is a scalar $c \geq 0$ such that for all k-dimensional projection matrices $P \in \mathbb{R}^{d\times d}$,

$$\|SA(I-P)\|_F^2 + c = (1 \pm \varepsilon)\|A(I-P)\|_F^2.$$

Let S be a $(k/\varepsilon^2) \times n$ PCP sketch for A, and let T be a $d \times (k/\varepsilon^2)$ PCP sketch acting on the right (that is, $T^\top$ is a PCP sketch for $(SA)^\top$). The protocol is Algorithm 5.3.

Algorithm 5.3 The [BWZ16] protocol (arbitrary partition model)

1: Every server sends SA^tT to the coordinator.
2: The coordinator sends $SAT = \sum_t SA^tT$ back to all servers.
3: Every server computes the $(k/\varepsilon^2) \times k$ matrix U of the top k left singular vectors of SAT.
4: Every server sends $U^\top SA^t$ to the coordinator.
5: The coordinator returns the space $U^\top SA = \sum_t U^\top SA^t$ (its row span) as the output.

We select U as the top k left singular vectors of SAT, which intuitively look like the top k left singular vectors of SA, so $U^\top SA$ looks like the top k right singular vectors of SA, scaled by the corresponding singular values. The top k right singular vectors of SA work because S is a projection-cost preserving sketch.

In the protocol, SA^tT and SAT are $(k/\varepsilon^2) \times (k/\varepsilon^2)$ matrices, and $U^\top SA^t$ and $U^\top SA$ are $k \times d$ matrices. These are all the matrices communicated, so the communication is $O(sdk) + s \cdot \text{poly}(k/\varepsilon)$ words, which is $O(sdk) + \text{poly}(sk/\varepsilon)$. We continue the discussion of this protocol, and prove its correctness, in Lecture 6.

6 Distributed Low Rank Approximation Protocols and ℓ_1 Regression

Lecture 6, October 15, 2020, by David P. Woodruff. Compiled from the scribe notes of Jeffrey Chen (both halves), and from the lecture slides. A few short passages marked as added are filled in from the lecture slides or supply missing steps.

6.1 The [BWZ16] protocol

Recall the setting of Lecture 5: in the coordinator model, a coordinator has two-way communication with s servers; server $t \in [s]$ holds $A^t \in \mathbb{R}^{n\times d}$, and $A = \sum_{t=1}^s A^t$. The goal is for every server to output a common rank-k projection $P \in \mathbb{R}^{d\times d}$ such that $\|A - AP\|_F \le (1+\varepsilon)\|A - A_k\|_F$, where A_k is a best rank-k approximation of A. The [FSS20] and [KVW14] protocols achieve this with $O(skd/\varepsilon)$ communication; we would like to improve this to $O(skd) + \text{poly}(sk/\varepsilon)$, which the [BWZ16] protocol achieves. Its analysis uses projection-cost preserving (PCP) sketches [CEM+14].

Theorem 6.1 ([CEM+14]). *Let A be an $n \times d$ matrix, and let S be a random $(k/\varepsilon^2) \times n$ matrix, drawn from the distribution of [CEM+14]. Then with constant probability there exists a scalar $c \ge 0$ such that for all k-dimensional projection matrices P,*

$$\|A(I-P)\|_F^2 \le \|SA(I-P)\|_F^2 + c \le (1+\varepsilon)\|A(I-P)\|_F^2. \tag{6.1}$$

We omit the proof of this theorem; see [CEM+14]. To see what it implies, let $\tilde{P}$ be the k-dimensional projection for which $I - \tilde{P}$ minimizes $\|SA(I-P)\|_F^2$, and let P^* be the one for which $I - P^*$ minimizes $\|A(I-P)\|_F^2$. Then

$$\|A(I-\tilde{P})\|_F^2 \le \|SA(I-\tilde{P})\|_F^2 + c \le \|SA(I-P^*)\|_F^2 + c \le (1+\varepsilon)\|A(I-P^*)\|_F^2. \tag{6.2}$$

Note that $[SA]_k = SA\tilde{P}$ and $A_k = AP^*$, where $[SA]_k$ is a best rank-k approximation of SA. So (6.2) states that $\|A(I-\tilde{P})\|_F^2 \le (1+\varepsilon)\|A - A_k\|_F^2$. Then

$$\|SA - [SA]_k\|_F^2 + c = \|SA(I-\tilde{P})\|_F^2 + c \le (1+\varepsilon)\|A(I-\tilde{P})\|_F^2 \le (1+O(\varepsilon))\|A - A_k\|_F^2. \tag{6.3}$$

The protocol from Lecture 5 is Algorithm 6.1.

Algorithm 6.1 The [BWZ16] protocol

1: Let S be a $(k/\varepsilon^2) \times n$ projection-cost preserving sketch.
2: Let T be a $d \times (k/\varepsilon^2)$ projection-cost preserving sketch. (Every server can have the same S and T by receiving the same small pseudorandom seed from the coordinator.)
3: Server t sends the matrix SA^tT to the coordinator.
4: The coordinator sends the matrix $SAT = \sum_{t=1}^s SA^tT$ to every server.
5: Every server computes the $(k/\varepsilon^2) \times k$ matrix U of the top k left singular vectors of SAT. (Since SAT is a small $(k/\varepsilon^2) \times (k/\varepsilon^2)$ matrix, computing its SVD is cheap.)
6: Server t sends the matrix $U^\top SA^t$ to the coordinator.
7: The coordinator returns the matrix $U^\top SA = \sum_{t=1}^s U^\top SA^t$ as the output. The projection P projects onto the row span of $U^\top SA$.

In step 6, each $U^\top SA^t$ is a $k \times d$ matrix, so the s servers together send $O(skd)$ words. In [KVW14] the servers sent the matrices SA^t, which are $(k/\varepsilon) \times d$, for a cost of $O(skd/\varepsilon)$; so we saved a factor of $1/\varepsilon$. In total, this two-round protocol uses $s \cdot \text{poly}(k/\varepsilon) + O(skd)$ words of communication: $s \cdot \text{poly}(k/\varepsilon)$ in the first round and $O(skd)$ in the second.

6.1.1 Intuition

Before proving correctness, we develop some intuition. Intuitively, U looks like the top k left singular vectors of SA. This is because we can think of T as a PCP sketch for SA on the right: by (6.1) (applied to the matrix $(SA)^\top$ and the sketch $T^\top$), there is a scalar $c' \geq 0$ such that for every rank-k projection $\Pi \in \mathbb{R}^{(k/\varepsilon^2)\times(k/\varepsilon^2)}$, acting on the left,

$$\|(I-\Pi)SA\|_F^2 \leq \|(I-\Pi)SAT\|_F^2 + c' \leq (1+\varepsilon)\|(I-\Pi)SA\|_F^2. \tag{6.4}$$

Since U contains the top k left singular vectors of SAT, the rank-k projection that minimizes $\|(I-\Pi)SAT\|_F^2$ is $\Pi = UU^\top$, the projection onto the columns of U. Plugging this Π into the analogue of (6.2) for T, we get

$$\|(I-UU^\top)SA\|_F^2 \leq (1+\varepsilon)\|[SA]_k - SA\|_F^2.$$

So projecting onto U gives a good approximation to the best rank-k approximation of SA, which is given by the top k left singular vectors of SA.

Let $SA = U'\Sigma' V'^\top$ be the SVD of SA, and let U'_k consist of the top k left singular vectors of SA. Then $U_k'^\top U'\Sigma' V'^\top = \Sigma'_k V_k'^\top$ consists of the top k right singular vectors of SA, scaled by the corresponding singular values. Therefore $U^\top SA$ (with U the top k left singular vectors of SAT) looks like the top k right singular vectors of SA, scaled by its singular values, and the coordinator outputs the rows of $U^\top SA$ to project onto. Now, the rank-k projection $\tilde{P}$ that minimizes $\|SA(I-P)\|_F^2$ is the projection onto the top k right singular vectors of SA, that is, $\tilde{P} = V'_k V_k'^\top$. By (6.2), $\|A(I-\tilde{P})\|_F^2 \leq (1+\varepsilon)\|A(I-P^*)\|_F^2$, so $\tilde{P}$ is a good approximation to the optimal projection P^*. Since $U^\top SA$ looks like the top k right singular vectors of SA, we should intuitively project onto the rows of $U^\top SA$ to get a good answer. We now formalize this intuition.

6.1.2 Correctness

Theorem 6.2. *Suppose S satisfies (6.1) for A and T satisfies (6.4). Let the columns of $W \in \mathbb{R}^{d\times k}$ be an orthonormal basis for the row span of $U^\top SA$ (we also write W for this subspace), and let $P = WW^\top$ be the projection onto W. Then*

$$\|A - AP\|_F^2 \leq (1+O(\varepsilon))\|A - A_k\|_F^2.$$

Proof. Recall that T is a PCP sketch for SA, as in (6.4). Let $\tilde{\Pi} = \arg\min_{\text{rank-}k\ \Pi}\|(I-\Pi)SAT\|_F^2$. By the analogue of (6.2) for T, $\|(I-\tilde{\Pi})SA\|_F^2 \leq (1+\varepsilon)\|SA - [SA]_k\|_F^2$. We showed above that the optimal $\tilde{\Pi}$ for SAT is $UU^\top$, so

$$\|SA - UU^\top SA\|_F^2 \leq (1+\varepsilon)\|SA - [SA]_k\|_F^2.$$

Next we show $\|SA - SAP\|_F^2 \leq \|SA - UU^\top SA\|_F^2$. The rows of $UU^\top SA$ are linear combinations of the rows of $U^\top SA$, so they lie in W; the rows of SAP also lie in W, because P projects onto W. Since SAP projects each row of SA onto W, while $UU^\top SA$ only puts it somewhere in W, each row of SAP is at least as close to the corresponding row of SA as the row of $UU^\top SA$ is. Hence $\|SA - SAP\|_F^2 \leq \|SA - UU^\top SA\|_F^2$. Putting the two inequalities together,

$$\|SA - SAP\|_F^2 \leq (1+\varepsilon)\|SA - [SA]_k\|_F^2. \tag{6.5}$$

Recall that S is a PCP sketch for A. By (6.1), there is a scalar $c \geq 0$ such that for all k-dimensional projection matrices Q,

$$\|A - AQ\|_F^2 \leq \|SA - SAQ\|_F^2 + c \leq (1+\varepsilon)\|A - AQ\|_F^2.$$

To complete the proof, add c to both sides of (6.5):

$$\|SA - SAP\|_F^2 + c \leq (1+\varepsilon)\|SA - [SA]_k\|_F^2 + c.$$

Since S is a PCP sketch for A, the left side is at least $\|A(I-P)\|_F^2$. On the right side,

$$(1+\varepsilon)\|SA - [SA]_k\|_F^2 + c \leq (1+\varepsilon)(\|SA - [SA]_k\|_F^2 + c) \leq (1+\varepsilon)(1+\varepsilon)\|A - A_k\|_F^2.$$

For the last step, note that $\|SA - [SA]_k\|_F^2 \leq \|SA - SAP^*\|_F^2$, since SAP^* has rank at most k. By the PCP property with $Q = P^*$, we also have $\|SA - SAP^*\|_F^2 + c \leq (1+\varepsilon)\|A - AP^*\|_F^2 = (1+\varepsilon)\|A - A_k\|_F^2$. Putting these together,

$$\|A - AP\|_F^2 \leq (1 + O(\varepsilon))\|A - A_k\|_F^2. \qquad \square$$

6.1.3 Conclusion and open problems

To summarize, the [BWZ16] protocol achieves the optimal $O(sdk) + \text{poly}(sk/\varepsilon)$ communication for low rank approximation in the arbitrary partition model. We did not resolve the bit complexity: in the protocol, the servers send the matrices $U^\top SA^t$. If A has bounded bit complexity, then so does SA^t, but $U^\top$ may not. It turns out that the bit complexity of U can be reduced by adding noise, though we will not go into detail. The computations done by the coordinator and the servers can be done in input sparsity time. The protocol uses 2 rounds, which is optimal for the $O(sdk)$ bound, since with 1 round there is a lower bound of $\Omega(sdk/\varepsilon)$ on the communication.

There are many open problems about the communication complexity of optimization problems.

- Two servers hold $n \times n$ matrices A and B, and want to find the rank of $A + B$. The only known bounds are an upper bound of $O(n^2)$ and a lower bound of $\Omega(n)$.
- For linear programming, each server has a set of constraints $\langle A_{i,*}, x\rangle \leq b_i$, and the servers want to find $\min_x c^\top x$ subject to $Ax \leq b$ and $x \geq 0$. If A is an $n \times d$ matrix, meaning there are n constraints and d variables, the only known bounds are $O(sd^3)$ and $\Omega(d^2 + sd)$.
- Many servers each hold a subset of the edges of a graph. Can they answer graph questions, such as finding a matching or a vertex cover, without sending all the edges to the coordinator?

6.2 ℓ_1 regression

We now consider ℓ_1 regression, also called the method of least absolute deviations: we want to find an x^* that minimizes

$$\|Ax - b\|_1 = \sum_i |b_i - \langle A_{i,*}, x\rangle|.$$

The ℓ_1 cost is less sensitive to outliers than least squares, since the differences between b and Ax are not squared. If $A \in \mathbb{R}^{n\times d}$, then ℓ_1 regression can be solved exactly by linear programming (LP) in $\text{poly}(nd)$ time.

We can formulate ℓ_1 regression as an LP by introducing variables $\alpha^+, \alpha^- \in \mathbb{R}^n$ and solving

$$\min_{x,\alpha^+,\alpha^-} \mathbf{1}^\top(\alpha^+ + \alpha^-) \quad \text{subject to} \quad \begin{cases} Ax + \alpha^+ - \alpha^- = b, \\ \alpha^+, \alpha^- \geq 0. \end{cases}$$

The objective is the sum of the entries of $\alpha^+ + \alpha^-$. In an optimal solution, for each index i at most one of α_i^+ and α_i^- is nonzero: if both were positive, we could subtract the smaller value from both and still satisfy the constraints at a lower cost. Hence the optimal cost of this LP is $\min_x \|Ax - b\|_1$, and the minimizing x is x^*. There are d variables in x, $2n$ variables in α^+ and α^-, and n equality constraints, so the LP can be solved in $\text{poly}(nd)$ time. Since n can be much larger than d, we want to improve this time by sketching.

6.3 Well-conditioned bases

Throughout, we assume that $A \in \mathbb{R}^{n\times d}$ has full column rank d, so that the matrices W below are invertible.

For an $n \times d$ matrix A, we can always find an $n \times d$ matrix U with orthonormal columns such that $A = UW$ and $\|Ux\|_2 = \|x\|_2$ for all x; for example, write the SVD as $A = U(\Sigma V^\top)$. We want to find a different matrix U such that $A = UW$ and $\|Ux\|_1 \approx \|x\|_1$ for all x. Multiplying A on the right by an invertible change-of-basis matrix does not change its column span, so the question is equivalent to finding an invertible $d \times d$ matrix W such that $U = AW^{-1}$ preserves ℓ_1 norms. The precise requirement is the following.

Definition 6.3 (Well-conditioned basis)**.** A matrix U with $A = UW$ is a *well-conditioned basis* for A if for all $x \in \mathbb{R}^d$,

$$\|x\|_1/\operatorname{poly}(d) \le \|Ux\|_1 \le \operatorname{poly}(d)\|x\|_1.$$

One inequality relating the ℓ_1 and ℓ_2 norms is

$$\|x\|_2 \le \|x\|_1 \le \sqrt{n}\|x\|_2 \qquad \text{for all } x \in \mathbb{R}^n. \tag{6.6}$$

For the first inequality,

$$\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2} \le \sum_{i=1}^n \sqrt{x_i^2} = \sum_{i=1}^n |x_i| = \|x\|_1,$$

and for the second,

$$\|x\|_1 = \sum_{i=1}^n |x_i| = \sum_{i=1}^n |x_i| \cdot 1 \le \sqrt{\sum_{i=1}^n |x_i|^2}\sqrt{\sum_{i=1}^n 1^2} = \sqrt{n}\|x\|_2,$$

where the inequality is the Cauchy–Schwarz inequality.

Consider what happens if we simply let U have orthonormal columns spanning the column span of A. Then, using (6.6) in $\mathbb{R}^n$ and in $\mathbb{R}^d$,

$$\|Ux\|_1 \le \sqrt{n}\|Ux\|_2 = \sqrt{n}\|x\|_2 \le \sqrt{n}\|x\|_1 \quad \text{and} \quad \|Ux\|_1 \ge \|Ux\|_2 = \|x\|_2 \ge \frac{1}{\sqrt{d}}\|x\|_1.$$

Together, $\frac{1}{\sqrt{d}}\|x\|_1 \le \|Ux\|_1 \le \sqrt{n}\|x\|_1$, or $\|x\|_1 \le \|\sqrt{d}\,Ux\|_1 \le \sqrt{nd}\|x\|_1$. This is undesirable, because the approximation is off by a factor that depends on $\sqrt{n}$, which can be large.

6.3.1 The $(Q, 1)$ norm

Let $A = QW$ for any matrix Q with full column rank, and define $\|z\|_{Q,1} = \|Qz\|_1$. We show that $\|\cdot\|_{Q,1}$ is a norm:

- $\|z\|_{Q,1} = 0$ if and only if $z = 0$: indeed, $\|z\|_{Q,1} = \|Qz\|_1 = 0$ if and only if $Qz = 0$, which holds if and only if $z = 0$, since Q has full column rank.
- $\|\lambda z\|_{Q,1} = |\lambda|\|z\|_{Q,1}$ for all scalars λ: indeed, $\|\lambda z\|_{Q,1} = \|\lambda Qz\|_1 = |\lambda|\|Qz\|_1 = |\lambda|\|z\|_{Q,1}$.
- $\|y+z\|_{Q,1} \le \|y\|_{Q,1} + \|z\|_{Q,1}$ for all y, z: indeed, $\|y+z\|_{Q,1} = \|Q(y+z)\|_1 \le \|Qy\|_1 + \|Qz\|_1 = \|y\|_{Q,1} + \|z\|_{Q,1}$.

Let $C = \{z \in \mathbb{R}^d : \|z\|_{Q,1} \le 1\}$ be the unit ball of $\|\cdot\|_{Q,1}$. We claim that C is a convex set that is symmetric about the origin. For convexity, let $y, z \in C$. Any point on the segment between them can be written as $\lambda y + (1-\lambda)z$ for some $\lambda \in [0,1]$, and by the properties of norms,

$$\|\lambda y + (1-\lambda)z\|_{Q,1} \le \lambda\|y\|_{Q,1} + (1-\lambda)\|z\|_{Q,1} \le \lambda \cdot 1 + (1-\lambda)\cdot 1 = 1.$$

So every point on the segment between y and z lies in C. For symmetry, if $z \in C$, then $\|-z\|_{Q,1} = \|Q(-z)\|_1 = \|Qz\|_1 = \|z\|_{Q,1} \le 1$, so $-z \in C$ as well.

Theorem 6.4 (Löwner–John)**.** *Let $C \subseteq \mathbb{R}^d$ be a convex set that is symmetric about the origin (and is compact with nonempty interior). Then there is an ellipsoid $E = \{z \in \mathbb{R}^d : z^\top F z \le 1\}$, where $F = G^\top G$ for some matrix G, such that $E \subseteq C \subseteq \sqrt{d}E$.*

The unit ball C of $\|\cdot\|_{Q,1}$ satisfies these conditions, because $\|\cdot\|_{Q,1}$ is a norm on $\mathbb{R}^d$. Since E is bounded, F is positive definite, so we may take G to be an invertible $d \times d$ matrix (for example $G = F^{1/2}$). By this theorem, for all $z \in \mathbb{R}^d$,

$$\frac{1}{\sqrt{d}}(z^\top F z)^{1/2} \le \|z\|_{Q,1} \le (z^\top F z)^{1/2}. \tag{6.7}$$

To see this, note that if $z^\top F z \le 1$, then $z \in E \subseteq C$, so $\|z\|_{Q,1} \le 1$. Since both sides scale linearly under $z \mapsto \lambda z$, we get $\|z\|_{Q,1} \le (z^\top F z)^{1/2}$ for all z. Similarly, if $\|z\|_{Q,1} \le 1$, then $z \in C \subseteq \sqrt{d}E$, so $z^\top F z \le d$, which gives $(z^\top F z)^{1/2} \le \sqrt{d}\|z\|_{Q,1}$ for all z.

Define $U = QG^{-1}$, where $A = QW$ with Q of full column rank, and $F = G^\top G$ comes from Theorem 6.4. Note that $A = QW = QG^{-1}GW = U(GW)$. Let $z = G^{-1}x$. Then $\|Ux\|_1 = \|QG^{-1}x\|_1 = \|Qz\|_1 = \|z\|_{Q,1}$, and

$$(z^\top F z)^{1/2} = (x^\top (G^{-1})^\top G^\top G G^{-1} x)^{1/2} = (x^\top x)^{1/2} = \|x\|_2.$$

By (6.7), $\frac{1}{\sqrt{d}}\|x\|_2 \le \|Ux\|_1 \le \|x\|_2$. Applying (6.6) in $\mathbb{R}^d$,

$$\frac{\|x\|_1}{d} \le \frac{\|x\|_2}{\sqrt{d}} \le \|Ux\|_1 \le \|x\|_2 \le \|x\|_1,$$

or equivalently

$$\|x\|_1 \le \|dUx\|_1 \le d\|x\|_1.$$

So our chosen U, rescaled to dU, preserves ℓ_1 norms up to a factor of d, which does not depend on n. Here $A = (dU)(GW/d)$. In terms of $\|x\|_2$, the rescaled basis $\sqrt{d}\,U$ satisfies $\|x\|_2 \le \|\sqrt{d}\,Ux\|_1 \le \sqrt{d}\|x\|_2$.

6.4 A net for the ℓ_1 ball

We revisit the idea of a net from Lecture 1, now with the ℓ_1 norm. Consider the set $B = \{x \in \mathbb{R}^d : \|x\|_1 = 1\}$ (the boundary of the unit ℓ_1-ball). A subset $N \subseteq B$ is a *γ-net* if for every $x \in B$ there is a $y \in N$ such that $\|x - y\|_1 \le \gamma$. We can find a net N greedily: while there is a point $x \in B$ at distance greater than γ from every point of N, add x to N. By the triangle inequality, the ℓ_1-ball of radius $\gamma/2$ around every point of N is contained in the ℓ_1-ball of radius $1 + \gamma/2$ around 0. All of these balls are disjoint: if two of them intersected, their centers would be at distance at most $\gamma/2 + \gamma/2 = \gamma$, so the greedy algorithm would not have added the later one to the net. The ratio of

the volumes of similar d-dimensional polytopes is the d-th power of the ratio of their radii. This means

$$|N| \le \frac{\text{Vol}(\ell_1\text{-ball of radius } 1+\gamma/2)}{\text{Vol}(\ell_1\text{-ball of radius } \gamma/2)} = \left(\frac{1+\gamma/2}{\gamma/2}\right)^d.$$

Now let $A = UW$ for a well-conditioned basis U with $\|x\|_1 \le \|Ux\|_1 \le d\|x\|_1$ for all x. Let N be a (γ/d)-net for B, and let $M = \{Ux : x \in N\}$, so that

$$|M| = |N| \le \frac{(1+\gamma/(2d))^d}{(\gamma/(2d))^d}.$$

This is $(d/\gamma)^{O(d)}$ for $\gamma \le 1/2$. We claim that for every $x \in B$ there is a $y \in M$ such that $\|Ux - y\|_1 \le \gamma$. Indeed, there is an $x' \in N$ with $\|x - x'\|_1 \le \gamma/d$, so $\|Ux - Ux'\|_1 \le d\|x - x'\|_1 \le \gamma$ by the well-conditioned basis property, and we can take $y = Ux'$.

6.5 A rough algorithm

To solve ℓ_1 regression we want to minimize $\|Ax - b\|_1$. One way to reduce the size of the problem is to sample some of the rows of $[A, b]$ (as in leverage score sampling, Lecture 4). Uniform sampling is a bad idea for regression, since there may be an important row, very different from the others, that we fail to sample, which leads to a large error. Sampling proportionally to the squared norms of the rows is also insufficient, since the important row may have the same norm as the other rows but point in a different direction. For ℓ_2 regression, sampling proportionally to the squared ℓ_2 norms of the rows of U in the SVD of $[A, b]$ gives more weight to the important rows and leads to a good approximation. For ℓ_1 regression, we will show that the analogous sampling, proportionally to the ℓ_1 norms of the rows of a well-conditioned basis U, gives a good approximation.

The rough algorithm proceeds as follows.

1. Compute a $\text{poly}(d)$-approximation to the solution, that is, find an x' such that $\|Ax' - b\|_1 \le \text{poly}(d) \min_x \|Ax - b\|_1$. Then compute the residual $b' = b - Ax'$, and consider the problem $\min_x \|Ax - b'\|_1$. Note that

   $$\|Ax - b'\|_1 = \|Ax - (b - Ax')\|_1 = \|A(x + x') - b\|_1.$$

   So this problem is equivalent to the original problem: if $y^* = \arg\min_x \|Ax - b'\|_1 = \arg\min_x \|A(x + x') - b\|_1$, then $y^* + x'$ minimizes the original problem $\|Ax - b\|_1$.

2. Compute a well-conditioned basis U for A such that $A = UW$ and, for all $x \in \mathbb{R}^d$,

   $$\|x\|_1 / \text{poly}(d) \le \|Ux\|_1 \le \text{poly}(d)\|x\|_1.$$

   We can then consider the problem $\min_x \|Ux - b'\|_1 = \min_x \|AW^{-1}x - b'\|_1$. This problem is equivalent to minimizing $\|Ax - b'\|_1$: if $\hat{x} = \arg\min_x \|Ux - b'\|_1 = \arg\min_x \|AW^{-1}x - b'\|_1$, then $W^{-1}\hat{x}$ minimizes $\|Ax - b'\|_1$.

3. Approximate the new problem $\min_x \|Ux - b'\|_1$ by sampling $\text{poly}(d/\varepsilon)$ rows of $[U, b']$ proportionally to their ℓ_1 norms. We have shown that this new problem is equivalent to the original one.

4. Solve ℓ_1 regression on the sample, which takes $\text{poly}(d/\varepsilon)$ time with linear programming, and output the solution x.

It remains to show that steps 1 and 2 can be computed efficiently.

6.6 The sketching theorem and its applications

Theorem 6.5 (Sketching theorem)**.** *There is a probability distribution over $(d \log d) \times n$ matrices R such that for every $n \times d$ matrix A, with probability at least $99/100$, simultaneously for all x,*

$$\|Ax\|_1 \le \|RAx\|_1 \le d \log d \cdot \|Ax\|_1.$$

This theorem says that R is a subspace embedding for the ℓ_1 norm: it is linear, it is independent of A, and it preserves the lengths of an infinite number of vectors. Before proving the theorem, we show that it gives both steps 1 and 2 of the rough algorithm.

1. **A $\mathrm{poly}(d)$-approximation.** Let $\hat{x} = \arg\min_x \|RAx - Rb\|_1$, which we can compute efficiently because R reduces the size of the problem to $\mathrm{poly}(d)$. We apply Theorem 6.5 to the $n \times (d+1)$ matrix $[A, b]$, so its guarantee holds for every vector of the form $Ax - b$. With $d+1$ columns in place of d, the matrix R has $(d+1)\log(d+1) = O(d \log d)$ rows, and the factor $d \log d$ below becomes $(d+1)\log(d+1)$, but we keep writing $d \log d$. Then

   $$\|A\hat{x} - b\|_1 \le \|R(A\hat{x} - b)\|_1 \le \|R(Ax^* - b)\|_1 \le d \log d \cdot \|Ax^* - b\|_1,$$

   where $x^* = \arg\min_x \|Ax - b\|_1$ is the optimal solution. So $\hat{x}$ is a $\mathrm{poly}(d)$-approximation, which is step 1.

2. **A well-conditioned basis.** Compute RA, and compute W such that RAW has orthonormal columns (orthonormal in the ℓ_2 sense). This can be done with an SVD: write $RA = Z\Sigma V^\top$ and let $W = (\Sigma V^\top)^{-1}$. Note that RA has rank d, since $\|RAx\|_1 \ge \|Ax\|_1 > 0$ for $x \ne 0$, so $\Sigma V^\top$ is invertible. Then output $U = AW$ as the well-conditioned basis. This is similar to our preconditioning for the ℓ_2 norm in Lecture 4, where we wrote a sketch SA as a matrix with orthonormal columns times an invertible $d \times d$ matrix, multiplied A on the right by the inverse of that factor, and showed that the result is close to having orthonormal columns, meaning its condition number is close to 1.

   To show that $U = AW$ is well-conditioned, note that

   $$\begin{aligned}
   \|Ux\|_1 = \|AWx\|_1 &\le \|RAWx\|_1 && \text{by Theorem 6.5} \\
   &\le (d \log d)^{1/2} \|RAWx\|_2 && \text{by (6.6), since } R \text{ has } d \log d \text{ rows} \\
   &= (d \log d)^{1/2} \|x\|_2 && \text{since } RAW \text{ has orthonormal columns} \\
   &\le (d \log d)^{1/2} \|x\|_1 && \text{by (6.6)}
   \end{aligned}$$

   and

   $$\begin{aligned}
   \|Ux\|_1 = \|AWx\|_1 &\ge \|RAWx\|_1/(d \log d) && \text{by Theorem 6.5} \\
   &\ge \|RAWx\|_2/(d \log d) && \text{by (6.6)} \\
   &= \|x\|_2/(d \log d) && \text{since } RAW \text{ has orthonormal columns} \\
   &\ge \|x\|_1/(d^{3/2} \log d) && \text{by (6.6).}
   \end{aligned}$$

   So

   $$\|x\|_1/(d^{3/2} \log d) \le \|Ux\|_1 \le (d \log d)^{1/2} \|x\|_1,$$

   and U is a well-conditioned basis, which is step 2.

We will prove Theorem 6.5 in the next lecture (Lecture 7). A dense R that works has i.i.d. standard Cauchy entries (density $\frac{1}{\pi(1+z^2)}$), scaled by $1/(d \log d)$.

7 Sketching for ℓ_1 Regression and the Data Stream Model

Lecture 7, October 22, 2020, by David P. Woodruff. Compiled from the scribe notes of Charlie Hou (first half) and Michael Zhang (second half), and from the lecture slides.

The first half of this lecture finishes the ℓ_1 regression algorithm of Lecture 6 by proving the sketching theorem with Cauchy random variables, and then explains how to make the sketch fast. The second half starts the part of the course on data streams.

7.1 Recap: sketching for ℓ_1 regression

Recall from Lecture 6 the ℓ_1 regression problem: given $A \in \mathbb{R}^{n\times d}$ and $b \in \mathbb{R}^n$, find

$$\min_{x\in\mathbb{R}^d} \|Ax - b\|_1.$$

It can be solved by linear programming, and we want to sketch it to make the linear program smaller. The algorithm of Lecture 6 needs two ingredients: a sketching matrix that gives a $\text{poly}(d)$-approximation, and a well-conditioned basis U of the column span of A, that is, a basis that preserves the ℓ_1 norm up to $\text{poly}(d)$ factors. Both ingredients follow from the theorem below, as shown in Lecture 6; we do not repeat that here.

Theorem 7.1 (ℓ_1 sketching theorem)**.** *There is a probability space over $m \times n$ matrices R, with $m = O(d \log d)$, such that for any $n \times d$ matrix A, with probability at least $99/100$ we have for all $x \in \mathbb{R}^d$:*

$$\|Ax\|_1 \le \|RAx\|_1 \le O(d \log d)\|Ax\|_1. \tag{7.1}$$

A dense R that works: the entries of R are i.i.d. standard Cauchy random variables scaled by $1/m$, where $m = c_0 d \log d$ for a large enough constant c_0. (One more constant rescaling of R happens at the end of the proof.)

7.1.1 Cauchy random variables

A standard Cauchy random variable has probability density function

$$f(z) = \frac{1}{\pi(1+z^2)}, \qquad z \in (-\infty, \infty).$$

It has undefined expectation and infinite variance, and it is heavy-tailed. However, it has one very useful property.

Fact 7.2 (1-stability)**.** *If $z_1, \ldots, z_n$ are i.i.d. standard Cauchy random variables, then for every $a \in \mathbb{R}^n$,*

$$\sum_{i=1}^n a_i z_i \sim \|a\|_1 z,$$

where z is a standard Cauchy random variable.

A standard Cauchy random variable can be generated as the ratio of two independent standard Gaussian random variables.

7.2 Proof of the sketching theorem

7.2.1 Reductions

First, the usual tricks. Both sides of (7.1) depend on A only through the vectors Ax, that is, through the column span of A. We can assume that A has full column rank d, since otherwise we can replace d by $\operatorname{rank}(A) \le d$ throughout. If $U = AW$ is another basis of the column span, with W invertible, then $Ax = U(W^{-1}x)$, and $W^{-1}x$ ranges over all of $\mathbb{R}^d$ as x does, so it suffices to prove (7.1) for U in place of A; in particular we may assume that A is a well-conditioned basis. Second, both sides of (7.1) are homogeneous in x, so we may divide by $\|x\|_1$ and assume $\|x\|_1 = 1$.

We take A to be an Auerbach basis, which always exists.

Fact 7.3 (Auerbach basis)**.** *Every d-dimensional subspace of $\mathbb{R}^n$ has a basis $A_{*,1}, \dots, A_{*,d}$ such that the matrix A with these columns satisfies*

(i) $\|x\|_\infty \le \|Ax\|_1$ *for all* $x \in \mathbb{R}^d$*, and*

(ii) $\sum_{i=1}^d \|A_{*,i}\|_1 = d$.

In particular, by the triangle inequality and (ii), $\|Ax\|_1 \le \sum_i |x_i| \|A_{*,i}\|_1 \le d\|x\|_\infty \le d\|x\|_1$ for all x.

7.2.2 Three facts

The proof rests on the following three facts.

Lemma 7.4 (Net, Lecture 6)**.** *For every $\gamma \in (0,1)$ there is a set $M \subseteq \{Ax : \|x\|_1 = 1\}$ with $|M| \le (d/\gamma)^{O(d)}$ such that for every x with $\|x\|_1 = 1$ there is a $y \in M$ with $\|Ax - y\|_1 \le \gamma$ (a γ-net).*

This was proven in the previous lecture (Lecture 6). There we took $M = \{Ax' : x' \in N\}$ for a (γ/d)-net N of the unit ℓ_1 sphere of $\mathbb{R}^d$, and we only used that $\|Ax - Ax'\|_1 \le d\|x - x'\|_1$, which the Auerbach basis satisfies.

Lemma 7.5 (Lower bound for a fixed vector)**.** *For any fixed $y \in \mathbb{R}^n$, with probability at least $1 - \exp(-m/16)$,*

$$\|Ry\|_1 \ge \frac{\|y\|_1}{40}.$$

In particular, for any fixed x, $\|RAx\|_1 = \Omega(\|Ax\|_1)$ with probability $1 - \exp(-\Omega(d \log d))$.

Lemma 7.6 (Upper bound for all vectors)**.** *For any constant $t > 1$, with probability at least $(1 - O(\log d/d))(1 - 1/t)$, simultaneously for all $x \in \mathbb{R}^d$,*

$$\|RAx\|_1 = O(d \log d)\|Ax\|_1.$$

The third fact is exactly the second inequality in (7.1), so it remains to prove the first one. We first show how the three facts give the theorem, and then prove Lemmas 7.5 and 7.6.

7.2.3 Proof of the theorem from the three facts

Proof of Theorem 7.1. Multiply R by the constant 40, so that Lemma 7.5 reads $\|Ry\|_1 \ge \|y\|_1$ for each fixed y; Lemma 7.6 still gives $\|RAz\|_1 = O(d \log d)\|Az\|_1$ for all z, with a larger constant.

Set $\gamma = \frac{1}{d^3 \log d}$, which gives $|M| \le d^{O(d)}$ in Lemma 7.4. By a union bound over M, the probability that some $y \in M$ fails to satisfy $\|Ry\|_1 \ge \|y\|_1$ is at most

$$d^{O(d)} \exp(-m/16) = d^{O(d)} \exp\Big(- \frac{c_0}{16} d \log d \Big) \le \frac{1}{100},$$

where the last step holds because the constant c_0 in $m = c_0 d \log d$ is chosen large enough compared to the constant in the exponent $O(d)$.

Now let x with $\|x\|_1 = 1$ be arbitrary, and let $y \in M$ satisfy $\|Ax - y\|_1 \le \gamma$. Since $y = Az$ for some z with $\|z\|_1 = 1$, the vector $Ax - y = A(x - z)$ is in the column span of A, and Lemma 7.6 applies to it. Hence

$$\begin{aligned}
\|RAx\|_1 &\ge \|Ry\|_1 - \|R(Ax - y)\|_1 \\
&\ge \|y\|_1 - O(d \log d)\|Ax - y\|_1 \\
&\ge \|y\|_1 - O(d \log d)\gamma \\
&\ge \|y\|_1 - O\Big(\frac{1}{d^2}\Big) \\
&\ge \frac{\|y\|_1}{2}.
\end{aligned}$$

The last inequality holds because $y = Az$ for a z with $\|z\|_1 = 1$, so $\|y\|_1 = \|Az\|_1 \ge \|z\|_\infty$ by property (i) of the Auerbach basis, and $\|z\|_\infty \ge \frac{1}{d}$ because otherwise $\|z\|_1 < 1$. So $\|y\|_1 \ge 1/d$, and the $O(1/d^2)$ term is at most $\frac{1}{2d} \le \frac{\|y\|_1}{2}$ once γ is a small enough constant times $\frac{1}{d^3 \log d}$, which still gives $|M| \le d^{O(d)}$. Continuing the chain, by the triangle inequality $\|y\|_1 \ge \|Ax\|_1 - \|Ax - y\|_1 \ge \|Ax\|_1 - \gamma$, and $\|Ax\|_1 \ge \|x\|_\infty \ge \frac{1}{d} \ge 3\gamma$, so

$$\frac{\|y\|_1}{2} \ge \frac{\|Ax\|_1 - \gamma}{2} \ge \frac{\|Ax\|_1}{3}.$$

Altogether, we have

$$\frac{\|Ax\|_1}{3} \le \|RAx\|_1 \le O(d \log d)\|Ax\|_1,$$

and multiplying R by 3 gives (7.1). □

Remark 7.7. The proof above works whenever the union bound succeeds and Lemma 7.6 holds, which happens with probability at least $1 - \frac{1}{100} - O(\frac{\log d}{d}) - \frac{1}{t}$. This is a constant, and adjusting constants makes it at least 99/100. We choose c_0 large enough that the union bound fails with probability at most $\frac{1}{300}$ (instead of $\frac{1}{100}$), and take $t = 300$. We also replace the truncation threshold d^3 in the proof of Lemma 7.6 by a large constant times d^3. This makes the $O(\frac{\log d}{d})$ term at most $\frac{1}{300}$, and does not change the $O(\log d)$ bound there.

So what remains is to prove Lemmas 7.5 and 7.6.

7.2.4 Proof of the lower bound for a fixed vector

Proof of Lemma 7.5. Fix $y \in \mathbb{R}^n$. By 1-stability (Fact 7.2), for every row $R_{j,*}$ of R,

$$\langle R_{j,*}, y \rangle = \frac{\|y\|_1 Z_j}{m},$$

where Z_j is a standard Cauchy random variable, and $Z_1, \ldots, Z_m$ are i.i.d. because the rows of R are independent. So $Ry = (\|y\|_1 Z_1, \ldots, \|y\|_1 Z_m)/m$, and

$$\|Ry\|_1 = \|y\|_1 \sum_{j=1}^{m} \frac{|Z_j|}{m},$$

where the $|Z_j|$ are half-Cauchy.

From the density, $\Pr[|Z_j| \geq \frac{1}{10}] \geq \frac{1}{2}$. Let $Y_j = 1$ if $|Z_j| \geq \frac{1}{10}$ and $Y_j = 0$ otherwise, so that $\mathbb{E}[\sum_j Y_j] \geq \frac{m}{2}$ and $\sum_j |Z_j| \geq \frac{1}{10} \sum_j Y_j$. Then

$$\Pr\Big[\sum_{j=1}^{m} |Z_j| \leq \frac{m}{40}\Big] \leq \Pr\Big[\sum_{j=1}^{m} Y_j \leq \frac{\mathbb{E}[\sum_j Y_j]}{2}\Big] \leq e^{-\mathbb{E}[\sum_j Y_j]/8} \leq e^{-m/16},$$

where the first inequality holds because $\sum_j |Z_j| \leq \frac{m}{40}$ implies $\sum_j Y_j \leq \frac{m}{4} \leq \frac{\mathbb{E}[\sum_j Y_j]}{2}$, and the second is the Chernoff bound. So with probability at least $1 - e^{-m/16}$,

$$\|Ry\|_1 = \|y\|_1 \sum_{j} \frac{|Z_j|}{m} \geq \|y\|_1 \frac{m/40}{m} = \frac{\|y\|_1}{40}.$$

With $y = Ax$ and $m = c_0 d \log d$ this is the second statement. Adjusting R by a constant factor gives the form $\|RAx\|_1 \geq \|Ax\|_1$ used above. □

7.2.5 Proof of the upper bound for all vectors

Proof of Lemma 7.6. By the triangle inequality, then bounding each $|x_i|$ by $\|x\|_\infty$, then property (i) of the Auerbach basis,

$$\|RAx\|_1 \leq \sum_{i=1}^{d} \|RA_{*,i} x_i\|_1 \leq \|x\|_\infty \sum_{i=1}^{d} \|RA_{*,i}\|_1 \leq \Big(\sum_{i=1}^{d} \|RA_{*,i}\|_1\Big) \|Ax\|_1.$$

So it suffices to prove that, with the stated probability,

$$\sum_{i=1}^{d} \|RA_{*,i}\|_1 = O(\log d) \sum_{i=1}^{d} \|A_{*,i}\|_1. \tag{7.2}$$

This implies $\sum_i \|RA_{*,i}\|_1 = O(d \log d)$, because $\sum_i \|A_{*,i}\|_1 = d$ for an Auerbach basis.

The difficulty is the heavy tail: each $|Z|$ has cumulative distribution function $1 - \Theta(1/z)$ for large z, so a sum of half-Cauchy random variables is not concentrated from above, and no Chernoff bound shows that $\|RAx\|_1$ is small for a fixed x, let alone for all x at once. Instead we truncate.

By 1-stability, as in the previous proof,

$$\|RA_{*,i}\|_1 = \|A_{*,i}\|_1 \sum_{j=1}^{m} \frac{|Z_{i,j}|}{m},$$

where each $Z_{i,j}$ is a standard Cauchy random variable. Note that for different i the $Z_{i,j}$ are dependent, since they share the row $R_{j,*}$. This does not matter, because the argument below only uses the union bound and linearity of expectation.

Let $\mathcal{E}_{i,j}$ be the event that $|Z_{i,j}| \leq d^3$, and define $Z'_{i,j} = |Z_{i,j}|$ if $\mathcal{E}_{i,j}$ occurs and $Z'_{i,j} = d^3$ otherwise. We know

$$\mathbb{E}[|Z_{i,j}| \mid \mathcal{E}_{i,j}] = \mathbb{E}[Z'_{i,j} \mid \mathcal{E}_{i,j}] = O(\log d),$$

where the last equality comes from writing out the integral:

$$\mathbb{E}[|Z_{i,j}| \mid \mathcal{E}_{i,j}] = \frac{1}{\Pr[\mathcal{E}_{i,j}]} \int_0^{d^3} \frac{2z}{\pi(1+z^2)}\, dz = \frac{\ln(1+d^6)}{\pi \Pr[\mathcal{E}_{i,j}]} \le \frac{2}{\pi} \ln(1+d^6),$$

using $\Pr[\mathcal{E}_{i,j}] \ge \frac{1}{2}$. Let $\mathcal{E}$ be the event that $\mathcal{E}_{i,j}$ occurs for all i, j. Since $\Pr[|Z_{i,j}| > d^3] = \frac{2}{\pi} \arctan(d^{-3}) \le d^{-3}$, the union bound over the md pairs (i, j) gives

$$\Pr[\mathcal{E}] \ge 1 - \sum_{i,j} \Pr[\text{not } \mathcal{E}_{i,j}] \ge 1 - \frac{md}{d^3} = 1 - O\Big(\frac{\log d}{d}\Big).$$

Since $\mathcal{E} \subseteq \mathcal{E}_{i,j}$, we have $\Pr[\mathcal{E}_{i,j} \mid \mathcal{E}] = 1$ and $\mathbb{E}[Z'_{i,j} \mid \mathcal{E}_{i,j}, \mathcal{E}] = \mathbb{E}[Z'_{i,j} \mid \mathcal{E}]$. Observe that

$$\begin{aligned}
O(\log d) = \mathbb{E}[Z'_{i,j} \mid \mathcal{E}_{i,j}] &= \mathbb{E}[Z'_{i,j} \mid \mathcal{E}_{i,j}, \mathcal{E}] \Pr[\mathcal{E} \mid \mathcal{E}_{i,j}] + \mathbb{E}[Z'_{i,j} \mid \mathcal{E}_{i,j}, \text{not } \mathcal{E}] \Pr[\text{not } \mathcal{E} \mid \mathcal{E}_{i,j}] \\
&\ge \mathbb{E}[Z'_{i,j} \mid \mathcal{E}_{i,j}, \mathcal{E}] \Pr[\mathcal{E} \mid \mathcal{E}_{i,j}] \\
&= \mathbb{E}[Z'_{i,j} \mid \mathcal{E}] \frac{\Pr[\mathcal{E}_{i,j} \mid \mathcal{E}] \Pr[\mathcal{E}]}{\Pr[\mathcal{E}_{i,j}]} \\
&\ge \mathbb{E}[Z'_{i,j} \mid \mathcal{E}] \Pr[\mathcal{E}] \\
&\ge \mathbb{E}[Z'_{i,j} \mid \mathcal{E}] \Big(1 - O\Big(\frac{\log d}{d}\Big)\Big),
\end{aligned}$$

which implies

$$\mathbb{E}[Z'_{i,j} \mid \mathcal{E}] = O(\log d).$$

On $\mathcal{E}$ we have $Z'_{i,j} = |Z_{i,j}|$ for all i, j, so by linearity of expectation

$$\mu := \mathbb{E}\Big[\sum_{i=1}^{d} \|RA_{*,i}\|_1 \;\Big|\; \mathcal{E}\Big] = \sum_{i=1}^{d} \|A_{*,i}\|_1 \sum_{j=1}^{m} \frac{\mathbb{E}[|Z_{i,j}| \mid \mathcal{E}]}{m} = O(\log d) \sum_{i=1}^{d} \|A_{*,i}\|_1.$$

Hence, by Markov's inequality applied conditionally on $\mathcal{E}$,

$$\begin{aligned}
\Pr\Big[\sum_{i=1}^{d} \|RA_{*,i}\|_1 \le t\mu\Big] &\ge \Pr\Big[\sum_{i=1}^{d} \|RA_{*,i}\|_1 \le t\mu \;\Big|\; \mathcal{E}\Big] \Pr[\mathcal{E}] \\
&\ge \Big(1 - \frac{1}{t}\Big)\Big(1 - O\Big(\frac{\log d}{d}\Big)\Big).
\end{aligned}$$

So for a constant t, with this probability $\sum_i \|RA_{*,i}\|_1 = t \cdot O(\log d) \sum_i \|A_{*,i}\|_1 = O(\log d) \sum_i \|A_{*,i}\|_1$, which is (7.2). □

7.3 Faster sketches for ℓ_1 regression

Recall the overall ℓ_1 regression algorithm of Lecture 6:

1. compute a poly(d)-approximation, and, in parallel, compute a well-conditioned basis;
2. sample rows from the well-conditioned basis and the residual of the poly(d)-approximation;
3. solve ℓ_1 regression on the sample, obtaining a vector x, and output x.

We saw how to solve the problems in the first step by sketching with a matrix of i.i.d. Cauchy random variables.

So far we have discussed most of the steps of this algorithm. What remains is to speed up its bottleneck: computing RA, where R is the $O(d \log d) \times n$ matrix of i.i.d. Cauchy random variables. Since R is dense, computing RA takes $O(n \cdot \text{poly}(d))$ time. The other steps of the algorithm are much faster, since they work in a much lower dimension, poly(d).

Sparse Cauchy sketch [CW17, MM13]. We can speed up this step by choosing $R = SC$, where $S \in \mathbb{R}^{m \times n}$ is a CountSketch matrix with $m = \text{poly}(d)$ rows and $C = \text{diag}(C_1, \ldots, C_n)$ is a diagonal matrix of i.i.d. standard Cauchy random variables. Recall from Lecture 3 that each column i of S has a single nonzero entry $S_{h(i),i} = \sigma(i)$, for a hash function $h\colon [n] \to [m]$ and random signs $\sigma\colon [n] \to \{-1, 1\}$. With R chosen this way we get a looser guarantee: for all x,

$$\frac{1}{d^2 \log^2 d}\|Ax\|_1 \le \|RAx\|_1 \le O(d \log d)\|Ax\|_1.$$

However, the overall running time for ℓ_1 regression becomes $O(\text{nnz}(A) + \text{poly}(d/\varepsilon))$. The speedup is in computing $RA = S(CA)$: CA rescales row i of A by C_i, and $S(CA)$ applies a CountSketch to CA, so both take $O(\text{nnz}(A))$ time.

Exponential random variables [WZ13]. We can improve this further by choosing $R = SD$, where S is a CountSketch matrix with $m = \text{poly}(d)$ rows and $D = \text{diag}(1/E_1, \ldots, 1/E_n)$ is a diagonal matrix of reciprocals of i.i.d. exponential random variables $E_i \sim \text{Exp}(1)$. This achieves a better lower bound: for all x,

$$\frac{1}{d^{1/2}\,\text{poly}(\log(nd))}\|Ax\|_1 \le \|RAx\|_1 \le O(d \log d)\|Ax\|_1.$$

With this R, we need to sample fewer rows in the later steps of the algorithm.

Constant distortion from below [WW19]. Recently Wang and Woodruff [WW19] found an even better bound,

$$\Omega(1)\|Ax\|_1 \le \|RAx\|_1 \le O(d \log d)\|Ax\|_1,$$

with a similar running time.

7.4 A fun fact about Cauchy random variables

Suppose $R_1, \ldots, R_n$ are i.i.d. copies of a random variable with mean 0 and variance σ^2. What is the distribution of $\frac{1}{n}\sum_i R_i$? By the Central Limit Theorem, for large n it approaches a normal random variable $\mathcal{N}(0, \sigma^2/n)$. Intuitively, the variance is decreasing and the average is approaching its expectation.

Now suppose instead that $R_1, \ldots, R_n$ are i.i.d. standard Cauchy random variables. Then the theorem does not apply: by 1-stability (Fact 7.2) with $a = (1, \ldots, 1)$,

$$\frac{1}{n}\sum_{i=1}^{n} R_i \sim \frac{n \cdot z}{n} = z,$$

where z is a standard Cauchy random variable. So the average of n Cauchy random variables is still a standard Cauchy random variable: it does not concentrate at all, however large n is.

7.5 The data stream model

In the next part of the course we shift our focus to another important data model: streaming. In many applications we cannot afford the space to store all the data involved in a problem, so the "one pass" streaming model is of great interest in the study of algorithms for big data. From here on, x denotes the vector defined by the stream, not the regression variable of the previous sections.

Definition 7.8 (Turnstile streaming model)**.**

- There is an underlying n-dimensional vector x, initialized to 0^n.
- There is a long stream of updates: the j-th update (i_j, Δ_j) performs $x_{i_j} \leftarrow x_{i_j} + \Delta_j$, with $\Delta_j \in \{-M, -M+1, \ldots, M-1, M\}$, where $M \le \text{poly}(n)$.
- Throughout the stream, x is promised to be in $\{-M, -M+1, \ldots, M-1, M\}^n$, so each entry of x can be stored with $O(\log n)$ bits.
- Output an approximation to $f(x)$, for a given function f, with high probability over our coin tosses (that is, the algorithm may be randomized).
- The goal is to use as little space, in bits, as possible.

The model is useful in many applications where the data are too large to store in memory: packets on the Internet, stock transactions, weather data, and genome sequences.

Next we discuss two example streaming problems and attack them with techniques from earlier in the course.

7.5.1 Example: testing whether $x = 0$

How can we test, with probability at least $9/10$ over our random coin tosses, whether the underlying vector x equals 0^n? A trivial deterministic algorithm stores every entry of x, using $O(n \log n)$ bits of space. First we show that no deterministic algorithm can do better.

Theorem 7.9. *Any deterministic streaming algorithm that tests whether $x = 0^n$ requires $\Omega(n \log n)$ bits of space.*

Proof. Suppose for contradiction that a deterministic algorithm uses $o(n \log n)$ bits of space. Consider a stream whose first half consists of updates that build a vector $a \in \{0, 1, 2, \ldots, \text{poly}(n)\}^n$, and let $\Phi(a)$ be the memory contents of the algorithm after reading the first half of the stream. If $|\Phi(a)| = o(n \log n)$ bits, there are only $2^{o(n \log n)}$ possible memory contents, which is fewer than the $(\text{poly}(n) + 1)^n = 2^{\Omega(n \log n)}$ choices of a. So by the pigeonhole principle there exist $a \neq a'$ with $\Phi(a) = \Phi(a')$.

Suppose the second half of the stream consists of updates that build the vector $b = -a \in \{0, -1, -2, \ldots, -\text{poly}(n)\}^n$, which "undoes" the first half. Since the algorithm is deterministic and its memory contents after the first half are the same in both cases, it ends in the same state, and hence gives the same output, on the streams for $a + b$ and $a' + b$. But $a + b = 0^n$ and $a' + b = a' - a \neq 0^n$, so the algorithm errs on one of the two streams, a contradiction. □

Theorem 7.10. *There is a randomized streaming algorithm that tests whether $x = 0^n$, with one-sided error and success probability at least $9/10$, using $O(\log n)$ bits of space.*

Proof. Let $S \in \mathbb{R}^{m \times n}$ be a CountSketch matrix with $m = O(1/\varepsilon^2)$ rows. We store Sx and update it along the stream: since each column of S has a single nonzero entry and the map $x \mapsto Sx$ is linear, the update $x_i \leftarrow x_i + \Delta_j$ becomes $Sx \leftarrow Sx + \Delta_j S e_i$, which changes one entry of Sx. At the end we output "$x = 0^n$" if and only if $Sx = 0$.

By the JL moment property of CountSketch (Lecture 3), for any fixed vector x, with probability at least $9/10$,

$$\|Sx\|_2^2 = (1 \pm \varepsilon)\|x\|_2^2.$$

If $x = 0$, then $Sx = 0$ with probability 1. If $x \neq 0$, then with probability at least 9/10 we have $\|Sx\|_2^2 \geq (1-\varepsilon)\|x\|_2^2 > 0$, so $Sx \neq 0$. This is a one-sided-error algorithm with success probability at least 9/10. If we set $\varepsilon = \frac{1}{2}$, then Sx has $O(1)$ entries, each an integer of absolute value at most $nM = \text{poly}(n)$, so the overall space is $O(\log n)$ bits. We can also store the pairwise independent hash function and the 4-wise independent sign function that define S in $O(\log n)$ bits. □

7.5.2 Example: recovering a k-sparse vector

Suppose we are promised that x has at most k nonzero entries at the end of the stream, and we want to recover exactly the indices and values of these nonzero entries. The sparsity k is often small: for example, we may see all coordinates of a vector a followed by all coordinates of a *similar* vector b (as negative updates), and $a - b$ has only k nonzero entries.

Theorem 7.11. *There is a deterministic streaming algorithm that recovers any x with at most k nonzero entries using $O(k \log n)$ bits of space.*

The algorithm maintains Ax for a matrix A with few rows and the following property. (This A has nothing to do with the regression matrix of the first half of the lecture.)

Lemma 7.12. *Let A be an $m \times n$ matrix such that any $2k$ columns of A are linearly independent. Then Ax determines x among all vectors with at most k nonzero entries.*

Proof. Suppose for contradiction that there is a $y \neq x$ with at most k nonzero entries and $Ax = Ay$. Then $A(x-y) = 0$. Since x and y each have at most k nonzero entries, $x - y$ has at most $2k$ nonzero entries, so $A(x-y)$ is a linear combination of at most $2k$ columns of A with not all coefficients zero. This contradicts the assumption that any $2k$ columns of A are linearly independent. So x is unique. □

So if we maintain Ax in the stream (each update $x_i \leftarrow x_i + \Delta_j$ adds $\Delta_j A_{*,i}$ to it), then at the end we can recover the support and values of x from Ax, and the algorithm is deterministic given A. But do such matrices A exist with a small number m of rows? Yes: take the $2k \times n$ Vandermonde matrix

$$A_{i,j} = j^{i-1}, \qquad A = \begin{bmatrix} 1 & 1 & 1 & \cdots \\ 1 & 2 & 3 & \cdots \\ 1 & 4 & 9 & \cdots \\ 1 & 8 & 27 & \cdots \\ \vdots & \vdots & \vdots & \end{bmatrix},$$

so $m = 2k$. The $2k \times 2k$ submatrix of A with columns $j_1 < j_2 < \cdots < j_{2k}$ is itself a Vandermonde matrix, with determinant

$$\prod_{1 \leq a < b \leq 2k} (j_b - j_a) \neq 0,$$

so any $2k$ columns of A are linearly independent.

The entries of A grow exponentially, up to n^{2k-1}, so storing Ax exactly might take too much space: each entry of Ax is an integer of $O(k \log n)$ bits, so all of Ax would take $O(k^2 \log n)$ bits. To resolve this, we pick a prime p with $\max(2M, n) < p \leq \text{poly}(n)$ (one exists by Bertrand's postulate) and store $Ax \bmod p$ instead. The proof above still holds modulo p, as follows. Over the field of integers modulo p, any $2k$ columns of A are still linearly independent, because each factor $j_b - j_a$ of the determinant is an integer between 1 and $n - 1 < p$, so the determinant is a product of nonzero elements of the field and is nonzero modulo p. Hence if x and y both have at most k nonzero

entries and $Ax \equiv Ay \pmod{p}$, then $x - y \equiv 0 \pmod{p}$ entrywise. Since every entry of $x - y$ lies in $[-2M, 2M]$ and $p > 2M$, this forces $x - y = 0$. So $Ax \bmod p$ still determines x. Since we only count space here, we can recover x from $Ax \bmod p$ by trying all candidate vectors.

We do not need to store A explicitly, since its entries $A_{i,j} \bmod p = j^{i-1} \bmod p$ are given by a formula and can be computed when needed. We store $2k$ numbers modulo p, each of $O(\log p) = O(\log n)$ bits, so the overall space is $O(k \log n)$ bits. This proves Theorem 7.11.

8 Estimating Norms in a Data Stream

Lecture 8, October 29, 2020, by David P. Woodruff. Compiled from the scribe notes of Lichen Zhang (first half) and Michael Zhang (second half), and from the lecture slides.

8.1 The streaming model and norm estimation

We work in the turnstile streaming model introduced in Lecture 7. There is an underlying vector $x \in \mathbb{Z}^n$, initialized to 0^n; the j-th update (i_j, Δ_j) of a long stream performs $x_{i_j} \leftarrow x_{i_j} + \Delta_j$, with $\Delta_j \in \{-M, \ldots, M\}$ and $M \leq \text{poly}(n)$; and throughout the stream $x \in \{-M, \ldots, M\}^n$, so each entry of x can be stored with $O(\log n)$ bits. We want to output an approximation to a function $f(x)$ with high probability over our coin tosses (so the algorithm may be randomized), using as little space, in bits, as possible. The data stream model has a large number of real-life applications. Today we focus on one specific problem.

Suppose we want to estimate the p-norm of x, $\|x\|_p = \left(\sum_{i=1}^n |x_i|^p\right)^{1/p}$, or equivalently $\|x\|_p^p = \sum_{i=1}^n |x_i|^p$. Our goal is to find an estimate Z such that

$$(1-\varepsilon)\|x\|_p^p \leq Z \leq (1+\varepsilon)\|x\|_p^p \tag{8.1}$$

with probability at least $9/10$. Some remarks on specific values of p and their applications:

- $p = 1$ corresponds to the total variation distance between distributions. For simplicity, consider two discrete probability distributions $\mu = (\mu_1, \ldots, \mu_n)$ and $\nu = (\nu_1, \ldots, \nu_n)$; the total variation distance between μ and ν is
$$d_{\text{tv}}(\mu, \nu) = \frac{1}{2}\|\mu - \nu\|_1.$$
- $p = 2$ is the familiar Euclidean norm, useful for geometric and linear algebraic problems.
- $p = \infty$, the value of the maximum entry, is useful for anomaly detection.

8.2 Estimating the Euclidean norm

The goal is to find Z such that

$$(1-\varepsilon)\|x\|_2^2 \leq Z \leq (1+\varepsilon)\|x\|_2^2.$$

We can simply use a CountSketch matrix $S \in \mathbb{R}^{m \times n}$ with $m = O(1/\varepsilon^2)$ rows. It satisfies the JL moment property (Lecture 3), so if we set $Z = \|Sx\|_2^2$, the above holds with constant probability (at least $9/10$). The algorithm is Algorithm 8.1.

Algorithm 8.1 Estimating $\|x\|_2^2$ in a stream

1: Sample a CountSketch matrix $S \in \mathbb{R}^{m \times n}$ with $m = O(1/\varepsilon^2)$.
2: Initially $x = 0^n$, so $Sx = 0$.
3: **for** each update (i_j, Δ_j) **do**
4: $\quad Sx \leftarrow Sx + \Delta_j S_{*,i_j}$, where S_{*,i_j} is the column of S for the updated coordinate i_j.
5: **end for**
6: **return** $Z = \|Sx\|_2^2$.

How do we store S? Recall that S is determined by two hash functions, $h\colon [n] \to [m]$ and $\sigma\colon [n] \to \{-1, 1\}$, where h needs to be pairwise independent and σ needs to be 4-wise independent, so $O(\log n)$ bits suffice to store S. Each of the m entries of Sx takes $O(\log n)$ bits. Thus, the total space is $O(\frac{1}{\varepsilon^2} \log n)$ bits.

8.3 Estimating the ℓ_1 norm

Once again, we want Z such that

$$(1-\varepsilon)\|x\|_1 \le Z \le (1+\varepsilon)\|x\|_1.$$

A natural thing to try is a random Cauchy matrix S. [KNW10] show that one can store such an S using $O(\frac{1}{\varepsilon}\log n)$ bits, and the update rule is exactly the same as in the ℓ_2 case. Using $\frac{1}{\varepsilon^2}$ rows, we arrive at a space complexity of $O(\frac{1}{\varepsilon^2}\log n)$ bits. However, recall that a Cauchy random variable has undefined expectation and infinite variance, so it has no concentration, and $\|Sx\|_1$ need not be a good approximation of $\|x\|_1$. In fact, the ℓ_1 sketching theorem (Lecture 7), applied with $d=1$, only gives a constant-factor approximation, not the $1\pm\varepsilon$ we want. We use a different strategy.

Recall that the probability density function of a standard Cauchy random variable C is

$$f(u) = \frac{1}{\pi(1+u^2)}.$$

If we only consider $|C|$, the half-Cauchy, its density becomes

$$f(u) = \frac{2}{\pi(1+u^2)}, \qquad u \ge 0,$$

and its cumulative distribution function is

$$F(z) = \int_0^z f(u)\,du = \frac{2}{\pi}\arctan(z).$$

The idea is to consider the *median* of the random variable $|C|$, the value at which its cumulative distribution function equals $1/2$. If $F(z)=\frac{1}{2}$, then $\arctan(z)=\frac{\pi}{4}$, and applying tan to both sides gives $z=\tan(\frac{\pi}{4})=1$. So $F(1)=1/2$, and the median of $|C|$ is 1.

We prove the following key lemma.

Lemma 8.1. *Let $\mathcal{F}$ be a distribution with continuous cumulative distribution function F, let $0<\varepsilon\le 1/4$ and $0<\delta\le 1/2$, and let $m=O(\frac{\log(1/\delta)}{\varepsilon^2})$ with a large enough constant. If we take m independent samples $X_1,X_2,\dots,X_m$ from $\mathcal{F}$ and let $X=\text{median}\{X_1,X_2,\dots,X_m\}$, then with probability at least $1-\delta$,*

$$F(X)\in[1/2-\varepsilon,1/2+\varepsilon].$$

Proof. Consider the indicator random variable Z_i, where $Z_i=1$ if the sample X_i has the property $F(X_i)\le 1/2-\varepsilon$. Since F is continuous, $F(X_i)$ is uniformly distributed on $[0,1]$, so $\Pr[Z_i=1]=\frac{1}{2}-\varepsilon$. Let $Z=\sum_{i=1}^m Z_i$; then

$$\mathbb{E}[Z] = \mathbb{E}\Big[\sum_{i=1}^m Z_i\Big] = \sum_{i=1}^m \Pr[Z_i=1] = \frac{m}{2} - m\varepsilon.$$

Observe that if $F(X)\in[0,1/2-\varepsilon]$ for the median X, then at least half of the Z_i are 1, so $Z\ge m/2$, that is, $Z-\mathbb{E}[Z]\ge m\varepsilon$. By the Chernoff bound,

$$\Pr[Z-\mathbb{E}[Z]\ge m\varepsilon] \le \exp\Big(\frac{-\varepsilon^2\,\mathbb{E}[Z]}{3}\Big).$$

Here we use the multiplicative form $\Pr[Z\ge(1+\eta)\,\mathbb{E}[Z]]\le\exp(-\eta^2\,\mathbb{E}[Z]/3)$, valid for $0<\eta\le 1$, with $\eta = m\varepsilon/\,\mathbb{E}[Z] = \frac{2\varepsilon}{1-2\varepsilon}$. Since $\varepsilon\le 1/4$, this η lies in $[\varepsilon,1]$, so $\eta^2\ge\varepsilon^2$. Recall $m=O(\frac{\log(1/\delta)}{\varepsilon^2})$, so

$$\mathbb{E}[Z] = \frac{m}{2} - m\varepsilon = \Theta(m) = \frac{c\log(1/\delta)}{\varepsilon^2}$$

for a constant c that is as large as we like, and the Chernoff bound gives probability at most $\delta/2$ that $F(X) \in [0, 1/2 - \varepsilon]$. The same argument, with $Z_i = 1$ when $F(X_i) \geq 1/2 + \varepsilon$, shows that the probability that $F(X) \in [1/2 + \varepsilon, 1]$ is at most $\delta/2$. So with probability at least $1 - \delta$, $F(X) \notin [0, 1/2 - \varepsilon) \cup (1/2 + \varepsilon, 1]$, which is equivalent to $F(X) \in [1/2 - \varepsilon, 1/2 + \varepsilon]$. □

Now we know that the median of independent samples has its cumulative distribution function value close to that of the true median. Is the empirical median itself close to the true median? To answer this question, consider the inverse of the cumulative distribution function of $|C|$:

$$F^{-1}(u) = \tan\left(\frac{\pi u}{2}\right), \qquad u \in [0, 1).$$

By Lemma 8.1, $F(X)$ lies between $\frac{1}{2} - \varepsilon$ and $\frac{1}{2} + \varepsilon$, so $X = F^{-1}(F(X))$ lies between the following two endpoints:

$$F^{-1}\left(\frac{1}{2} - \varepsilon\right) = \tan\left(\frac{\pi}{4} - \frac{\varepsilon\pi}{2}\right) \geq 1 - \pi\varepsilon \geq 1 - 4\varepsilon,$$
$$F^{-1}\left(\frac{1}{2} + \varepsilon\right) = \tan\left(\frac{\pi}{4} + \frac{\varepsilon\pi}{2}\right) \leq 1 + 4\varepsilon \qquad (0 < \varepsilon \leq 1/10).$$

The first bound comes from the first-order Taylor expansion of $g(\varepsilon) = \tan(\frac{\pi}{4} - \frac{\varepsilon\pi}{2})$ around $\varepsilon = 0$, which is $1 - \pi\varepsilon$. This is a valid lower bound, since tan is convex on $[0, \pi/2)$ and so lies above its tangent lines. For the same reason, $1 + \pi\varepsilon$ is *not* an upper bound for $\tan(\frac{\pi}{4} + \frac{\varepsilon\pi}{2})$, so the second bound needs ε small. The function $1 + 4\varepsilon - \tan(\frac{\pi}{4} + \frac{\varepsilon\pi}{2})$ is 0 at $\varepsilon = 0$. It increases while $\sec^2(\frac{\pi}{4} + \frac{\varepsilon\pi}{2}) \leq 8/\pi$ and decreases afterward, and at $\varepsilon = 1/10$ it equals $1.4 - \tan(0.3\pi) > 0.02$. So it is nonnegative on $[0, 1/10]$.

Thus, for $\varepsilon \leq 1/10$, the empirical median satisfies $X \in [1 - 4\varepsilon, 1 + 4\varepsilon]$. We remark that if the distribution is very flat near its median, then sampling cannot give such a good approximation to the median. The algorithm, Algorithm 8.2, builds on this property of the median of the half-Cauchy.

Algorithm 8.2 Estimating $\|x\|_1$ in a stream

1: Draw a matrix $S \in \mathbb{R}^{m \times n}$ with independent standard Cauchy entries, $m = O(1/\varepsilon^2)$.
2: Initially $x = 0^n$, so $Sx = 0$.
3: **for** each update (i_j, Δ_j) **do**
4: $\quad Sx \leftarrow Sx + \Delta_j S_{*,i_j}$.
5: **end for**
6: **return** $Z = \text{median}\{|(Sx)_1|, \ldots, |(Sx)_m|\}$.

Why does this work? The entries of Sx are $(Sx)_i = \langle S_{i,*}, x\rangle$, where $S_{1,*}, \ldots, S_{m,*}$ are the rows of S. By the 1-stability of the Cauchy distribution (Lecture 7) and the independence of the rows, we can write

$$\begin{bmatrix} |(Sx)_1| \\ |(Sx)_2| \\ \vdots \\ |(Sx)_m| \end{bmatrix} \sim \begin{bmatrix} \|x\|_1 |C_1| \\ \|x\|_1 |C_2| \\ \vdots \\ \|x\|_1 |C_m| \end{bmatrix},$$

where $|C_1|, |C_2|, \ldots, |C_m|$ are i.i.d. half-Cauchy random variables. By Lemma 8.1 (with $\delta = 1/10$) and the computation above, the median of these m half-Cauchy random variables is a $(1 \pm 4\varepsilon)$-approximation to the true median, which is 1, so with probability at least 9/10 the output Z satisfies

$$(1 - 4\varepsilon)\|x\|_1 \leq Z \leq (1 + 4\varepsilon)\|x\|_1.$$

Replacing ε by $\varepsilon/4$ gives the goal.

8.4 Estimating ℓ_p norms for $0 < p < 2$

We can even achieve $O(\frac{1}{\varepsilon^2} \log n)$ bits of space for p-norm estimation for any $0 < p < 2$. It uses the fact that p-stable distributions exist for $0 < p < 2$: there is a distribution $\mathcal{D}_p$ such that if $W_1, \ldots, W_n \sim \mathcal{D}_p$ are i.i.d., then for every $a \in \mathbb{R}^n$,

$$a_1 W_1 + a_2 W_2 + \cdots + a_n W_n \sim \|a\|_p W,$$

where $W \sim \mathcal{D}_p$. Recall that the Cauchy distribution is 1-stable, and the Gaussian distribution is 2-stable. There is no closed-form expression for the probability density function of a p-stable distribution; instead, we can generate samples relatively easily.

Theorem 8.2. *If we pick $\theta \in [-\frac{\pi}{2}, \frac{\pi}{2}]$ and $r \in [0, 1]$ uniformly at random, then*

$$\frac{\sin(p\theta)}{\cos^{1/p}(\theta)} \left(\frac{\cos(\theta(1-p))}{\ln \frac{1}{r}} \right)^{\frac{1-p}{p}}$$

is a sample from a p-stable distribution.

We remark that if $p < 1$, then $\left(\sum_{i=1}^n |x_i|^p\right)^{1/p}$ is not even a norm, since it does not satisfy the triangle inequality. Also, although Theorem 8.2 requires uniform randomness, it is possible to discretize it and store a sketching matrix of samples from the p-stable distribution using limited independence.

The proof is similar to the 1-norm estimation. We draw $S \in \mathbb{R}^{m \times n}$ with i.i.d. entries from $\mathcal{D}_p$ and $m = O(1/\varepsilon^2)$, maintain Sx, and output $\text{median}_i |(Sx)_i| / \text{median}(|W|)$ for $W \sim \mathcal{D}_p$. By p-stability, each $|(Sx)_i|$ is distributed as $\|x\|_p |W|$, so Lemma 8.1 applies to the distribution of $|W|$ as it did to the half-Cauchy. Since the density of $|W|$ is positive at its median, the same argument with the inverse cumulative distribution function shows that the output is within a factor $1 \pm O(\varepsilon)$ of $\|x\|_p$, with a constant that depends on p. Raising to the power $p < 2$ and rescaling ε gives (8.1).

8.5 Estimating ℓ_p norms for $p > 2$

Unfortunately, for $p > 2$ there is no p-stable distribution. Even more, we will see later that $\Omega(n^{1-2/p})$ bits of space are required to approximate the p-norm, for $p > 2$, up to a constant factor with constant probability. It is possible to achieve $\widetilde{O}(n^{1-2/p})$ (in fact $O(n^{1-2/p} \log^2 n)$) bits of space using exponential random variables, and we will focus on a constant approximation parameter ε.

The sketching matrix is PD, where $P \in \mathbb{R}^{m \times n}$ is a CountSketch matrix (here P is a CountSketch, not a projection), and D is the diagonal matrix

$$D = \begin{bmatrix} 1/E_1^{1/p} & & & \\ & 1/E_2^{1/p} & & \\ & & \ddots & \\ & & & 1/E_n^{1/p} \end{bmatrix},$$

that is, each diagonal entry is the reciprocal of an independent standard exponential random variable, raised to the power $1/p$.

We first introduce, or refresh the reader's memory of, exponential random variables.

Definition 8.3. An *exponential random variable* has a parameter $\lambda > 0$; we write $X \sim \mathrm{Exp}(\lambda)$. Its probability density function is

$$f(u) = \begin{cases} \lambda e^{-\lambda u} & \text{if } u \geq 0, \\ 0 & \text{otherwise,} \end{cases}$$

and its cumulative distribution function is

$$F(u) = \begin{cases} 1 - e^{-\lambda u} & \text{if } u \geq 0, \\ 0 & \text{otherwise.} \end{cases}$$

It is *standard* if $\lambda = 1$.

A nice property of exponential random variables: if $E \sim \mathrm{Exp}(\lambda)$ and $t > 0$, then $t \cdot E \sim \mathrm{Exp}(\lambda/t)$, since for $u \geq 0$

$$\Pr[t \cdot E \leq u] = \Pr\left[E \leq \frac{u}{t}\right] = 1 - e^{-\frac{\lambda}{t}u}.$$

How do exponential random variables give some form of "stability"?

Lemma 8.4 (Stability of exponential random variables)**.** *Let $E_1, \ldots, E_n$ be independent standard exponential random variables, let $x \in \mathbb{R}^n$ have nonzero entries, and let*

$$q = \min\Big(\frac{E_1}{|x_1|^p}, \ldots, \frac{E_n}{|x_n|^p}\Big).$$

Then $q \sim \mathrm{Exp}(\|x\|_p^p)$; equivalently, $q \sim E/\|x\|_p^p$ for a standard exponential random variable E.

Proof. Consider the cumulative distribution function of q: for $u \geq 0$,

$$\begin{aligned} \Pr[q > u] &= \Pr\left[\forall i,\ \frac{E_i}{|x_i|^p} > u\right] \\ &= \prod_{i=1}^{n} \Pr\left[\frac{E_i}{|x_i|^p} > u\right] && \text{since the } E_i \text{ are independent} \\ &= \prod_{i=1}^{n} e^{-u|x_i|^p} && \text{since } \frac{E_i}{|x_i|^p} \sim \mathrm{Exp}(|x_i|^p) \\ &= e^{-u\sum_{i=1}^{n}|x_i|^p} \\ &= e^{-u\|x\|_p^p}. \end{aligned}$$

Thus $q \sim \mathrm{Exp}(\|x\|_p^p)$, and by the scaling property, $q \sim \frac{E}{\|x\|_p^p}$ for $E \sim \mathrm{Exp}(1)$. □

Coordinates with $x_i = 0$ contribute 0 to both Dx and $\|x\|_p$, so we may drop them and apply the lemma to the remaining ones. Next, we relate $\|Dx\|_\infty$ to $\|x\|_p$. Since $(Dx)_i = x_i/E_i^{1/p}$, we have $|(Dx)_i|^p = |x_i|^p/E_i$, and so

$$\|Dx\|_\infty^p = \max_i \frac{|x_i|^p}{E_i} = \frac{1}{\min_i \frac{E_i}{|x_i|^p}} \sim \frac{1}{E \cdot \frac{1}{\|x\|_p^p}} = \frac{\|x\|_p^p}{E}. \tag{8.2}$$

A standard exponential random variable falls in a fairly small range with good probability:

$$\Pr\left[E \in \left[\frac{1}{10}, 10\right]\right] = (1 - e^{-10}) - (1 - e^{-\frac{1}{10}}) = e^{-\frac{1}{10}} - e^{-10} > \frac{4}{5}.$$

Therefore, since $\|Dx\|_\infty = \|x\|_p / E^{1/p}$ in distribution,

$$\|Dx\|_\infty \in \left[\frac{\|x\|_p}{10^{1/p}}, 10^{1/p}\|x\|_p\right] \quad \text{with probability at least } \frac{4}{5}, \tag{8.3}$$

so $\|Dx\|_\infty$ is a good approximation of $\|x\|_p$. The problem is that $D \in \mathbb{R}^{n\times n}$, so no dimensionality reduction has happened.

So we apply a CountSketch matrix P with $\widetilde{O}(n^{1-2/p})$ rows. The intuition for P is the following: P hashes the coordinates of Dx into buckets and takes a signed sum of the entries in each bucket, and we expect everything except the largest entry to cancel out, so that $\|PDx\|_\infty \approx \|Dx\|_\infty$. In the next section we use Bernstein's bound, together with a division of the entries into large and small ones, to prove this.

8.6 Analysis of the estimator for $p > 2$

Our goal is to show that, with good probability, $\|PDx\|_\infty \approx \|Dx\|_\infty$, where $\|Dx\|_\infty$ satisfies (8.3) with probability at least $4/5$. (Lecture 9, Lecture 9, opens with a recap of this analysis.)

8.6.1 Notation and the goal

- Let m be the number of rows of P. We can think of the rows as m hash buckets for the entries of Dx.
- Let $h\colon [n] \to [m]$ and $\sigma\colon [n] \to \{-1, 1\}$ be the two hash functions that describe the CountSketch matrix P. For the analysis we assume they are truly random, although they can be derandomized.

To achieve $\|PDx\|_\infty \approx \|Dx\|_\infty$, we want:

(1) in each bucket i not containing the coordinate j for which $|(Dx)_j| = \|Dx\|_\infty$, we have $|(PDx)_i| \le \frac{\|x\|_p}{100}$;

(2) in the bucket i containing the coordinate j for which $|(Dx)_j| = \|Dx\|_\infty$, we have $\big||(PDx)_i| - \|Dx\|_\infty\big| \le \frac{\|x\|_p}{100}$.

If these two properties hold, then the bucket containing the largest coordinate of Dx gives a close estimate of $\|Dx\|_\infty$, and all other buckets are not too large. This guarantees a good estimate $\|PDx\|_\infty$.

8.6.2 Mean and variance of a bucket

Let $\mathbb{I}[\mathcal{A}] = 1$ if the event $\mathcal{A}$ occurs and $\mathbb{I}[\mathcal{A}] = 0$ otherwise. The value of the i-th bucket is

$$(PDx)_i = \sum_{j=1}^{n} \mathbb{I}[h(j) = i]\, \sigma(j)(Dx)_j. \tag{8.4}$$

Claim 8.5. $\mathbb{E}[(PDx)_i] = 0$.

Proof. This follows from (8.4), since $\mathbb{E}[\sigma(j)] = 0$ for all j. □

Claim 8.6. $\mathbb{E}[(PDx)_i^2] = O(\frac{1}{m})\|x\|_2^2$, *where the expectation is over both P and D.*

Proof. First fix D and take the expectation over P:

$$\begin{aligned}
\mathop{\mathbb{E}}_{P}[(PDx)_i^2] &= \sum_{j,j'} \mathbb{E}\left[\mathbb{I}[h(j)=i]\,\mathbb{I}[h(j')=i]\,\sigma(j)\sigma(j')(Dx)_j(Dx)_{j'}\right] \\
&= \sum_{j,j'} \mathbb{E}\left[\mathbb{I}[h(j)=i]\,\mathbb{I}[h(j')=i]\,\sigma(j)\sigma(j')\right](Dx)_j(Dx)_{j'} \qquad (8.5)\\
&= \sum_{j} \mathbb{E}\left[\mathbb{I}[h(j)=i]\right](Dx)_j^2 \qquad (8.6)\\
&= \frac{1}{m}\|Dx\|_2^2. \qquad (8.7)
\end{aligned}$$

From (8.5) to (8.6) we case on whether $j = j'$. If $j \neq j'$, then $\sigma(j)$ and $\sigma(j')$ are independent with mean 0, so the expectation is 0. So only the terms with $j = j'$ remain, which gives (8.6).

Then, over D,

$$\mathop{\mathbb{E}}_{D}[\|Dx\|_2^2] = \sum_j x_j^2\ \mathbb{E}[D_{j,j}^2]. \qquad (8.8)$$

Now we bound $\mathbb{E}[D_{j,j}^2]$. Recall that $D_{j,j} = 1/E_j^{1/p}$ with $E_j \sim \mathrm{Exp}(1)$, so $\mathbb{E}[D_{j,j}^2] = \mathbb{E}[E_j^{-2/p}] = \int_{t\geq 0} t^{-2/p}e^{-t}\,dt$, and

$$\begin{aligned}
\mathbb{E}[D_{j,j}^2] &= \int_{0\leq t\leq 1} t^{-2/p}e^{-t}\,dt + \int_{t>1} t^{-2/p}e^{-t}\,dt \\
&\leq \int_{0\leq t\leq 1} t^{-2/p}\,dt + \int_{t>1} e^{-t}\,dt \\
&= \frac{1}{1-\frac{2}{p}}\,t^{1-2/p}\Big|_0^1 - e^{-t}\Big|_1^\infty \\
&= O(1), \qquad (8.9)
\end{aligned}$$

where the $O(1)$ (namely $\frac{p}{p-2} + \frac{1}{e}$) uses that $p > 2$ is a constant. Combining (8.7), (8.8) and (8.9) gives $\mathbb{E}[(PDx)_i^2] = O(\frac{1}{m})\|x\|_2^2$. □

Claim 8.7. *$\|x\|_2^2 \leq n^{1-2/p}\|x\|_p^2$, and hence $\mathbb{E}[(PDx)_i^2] = O(\frac{1}{m})n^{1-2/p}\|x\|_p^2$.*

Proof. Recall Hölder's inequality: $\langle u, v\rangle \leq \|u\|_a\|v\|_b$ whenever $a, b \geq 1$ and $\frac{1}{a} + \frac{1}{b} = 1$. Using it with $a = \frac{p}{2}$ and $b = \frac{p}{p-2}$, we get

$$\begin{aligned}
\|x\|_2^2 = \sum_i x_i^2 \cdot 1 &\leq \Big(\sum_i (x_i^2)^{p/2}\Big)^{2/p} \cdot \Big(\sum_i 1^b\Big)^{1/b} \\
&= \Big(\sum_i |x_i|^p\Big)^{2/p} \cdot n^{1-2/p} \\
&= \|x\|_p^2\, n^{1-2/p}.
\end{aligned}$$

Plugging this into Claim 8.6 gives the second statement. □

8.6.3 Bernstein's bound and the choice of K

Now that we have a bound on the variance of $(PDx)_i$, we use Bernstein's bound to show that $(PDx)_i$ is concentrated.

Fact 8.8 (Bernstein's bound)**.** *Suppose $R_1, R_2, \ldots, R_n$ are independent random variables with $|R_j| \le K$ for all j, and $\mathrm{Var}[\sum_j R_j] = v^2$. There are constants $C, c > 0$ such that for all $t > 0$,*

$$\Pr\Big[\Big|\sum_j R_j - \mathbb{E}\Big[\sum_j R_j\Big]\Big| > t\Big] \le C(e^{-ct^2/v^2} + e^{-ct/K}).$$

It is tempting to apply this bound directly to $(PDx)_i$, with $R_j = \mathbb{I}[h(j) = i]\,\sigma(j)(Dx)_j$. But first we need a good value for K. If we simply take K to be the maximum entry of Dx, Bernstein's bound only shows that each $(PDx)_i$ is within the desired bound with constant probability, which is not enough for a union bound over all the entries of PDx. Indeed, bounding K by the maximum entry defeats our purpose: if a bucket contains a large entry of Dx, we want the absolute value of that bucket to be large, not close to 0. Therefore we proceed by a case analysis and apply Bernstein's bound only to the "small" entries.

8.6.4 The large elements

As above, let $R_j = \mathbb{I}[h(j) = i]\,\sigma(j)(Dx)_j$. We handle separately those R_j with $|R_j| > \frac{\alpha\|x\|_p}{\log n}$, for a sufficiently small constant $\alpha > 0$. If $|R_j| > \frac{\alpha\|x\|_p}{\log n}$, then necessarily $|(Dx)_j| \ge \frac{\alpha\|x\|_p}{\log n}$. We call an index j *large* if $|(Dx)_j| \ge \frac{\alpha\|x\|_p}{\log n}$, and *small* otherwise. How many indices are large?

Claim 8.9. *The expected number of large indices j is $O(\log^p n)$. Hence, with constant probability, the number of large indices is $O(\log^p n)$.*

Proof. Let $Y_j = 1$ if j is large and $Y_j = 0$ otherwise. Then $\mathbb{E}[\sum_j Y_j] = \sum_j \Pr[j \text{ is large}]$, and since $|(Dx)_j| = |x_j|/E_j^{1/p}$,

$$\begin{aligned}
\Pr[j \text{ is large}] &= \Pr\Big[\frac{|x_j|}{E_j^{1/p}} \ge \frac{\alpha\|x\|_p}{\log n}\Big] \\
&= \Pr\Big[\frac{|x_j|^p \log^p n}{\alpha^p \|x\|_p^p} \ge E_j\Big] \\
&= 1 - e^{-\frac{|x_j|^p \log^p n}{\alpha^p \|x\|_p^p}} \qquad (8.10) \\
&\le \frac{|x_j|^p \log^p n}{\alpha^p \|x\|_p^p}, \qquad (8.11)
\end{aligned}$$

where from (8.10) to (8.11) we used the inequality $e^{-u} \ge 1 - u$ for all u. Therefore

$$\mathbb{E}\Big[\sum_j Y_j\Big] \le \sum_j \frac{|x_j|^p \log^p n}{\alpha^p \|x\|_p^p} = \frac{\log^p n}{\alpha^p} = O(\log^p n).$$

By Markov's inequality, with constant probability the number of large indices j is $O(\log^p n)$: with probability at least $19/20$ it is at most $20\log^p n/\alpha^p$. □

Now condition on D satisfying (8.3) and having $O(\log^p n)$ large indices. Then, with high probability over h, there is no collision among the large indices: no two large indices are hashed into the same bucket. This holds because $m = \Theta(n^{1-2/p}\log n)$ is much larger than $\log^p n$. Since h is independent of D, the probability that some two of the $L = O(\log^p n)$ large indices collide is at most $\binom{L}{2}\frac{1}{m} = O(\frac{\log^{2p} n}{n^{1-2/p}\log n})$, which tends to 0 since $p > 2$ is a constant. With the large indices dealt with separately, we are ready to apply Bernstein's bound to the small ones.

8.6.5 The small elements

We condition on the following events:

1. $\|Dx\|_\infty \in [\frac{\|x\|_p}{10^{1/p}}, 10^{1/p}\|x\|_p]$, as in (8.3);
2. the number of large indices is $O(\log^p n)$;
3. all large indices are perfectly hashed by h;
4. $\|Dx\|_2^2 = O(\|x\|_2^2)$. By Markov's inequality and $\mathbb{E}_D[\|Dx\|_2^2] = O(\|x\|_2^2)$ (from the proof of Claim 8.6), this holds with probability at least 19/20 for a suitable constant.

We need event 4 because we apply Bernstein's bound over the randomness of P, with D fixed. For fixed D, by (8.7) the variance of the sum of the small terms in bucket i is at most $v^2 = \frac{1}{m}\|Dx\|_2^2$, which is $O(\frac{1}{m}n^{1-2/p}\|x\|_p^2)$ under event 4 and Claim 8.7. Note that event 3 concerns only the hash values of the large indices, which are independent of the hash values and signs of the small ones.

Every small index j has $|R_j| \le |(Dx)_j| < \frac{\alpha\|x\|_p}{\log n}$, so for the small indices we can take $K = \frac{\alpha\|x\|_p}{\log n}$. We set $t = \frac{\|x\|_p}{100}$ and $m = \Theta(n^{1-2/p}\log n)$. By Bernstein's bound, for each bucket i,

$$\Pr\Big[\Big|\sum_{\text{small } j} \mathbb{I}[h(j)=i]\,\sigma(j)(Dx)_j\Big| > \frac{\|x\|_p}{100}\Big] \le C(e^{-\Theta(\log n)} + e^{-c\frac{\log n}{100\alpha}}) \le \frac{1}{n^2}. \tag{8.12}$$

Here $t^2/v^2 = \Omega(m/n^{1-2/p}) = \Omega(\log n)$, with a constant that grows with the constant in $m = \Theta(n^{1-2/p}\log n)$, and $t/K = \frac{\log n}{100\alpha}$. So both terms are at most $\frac{1}{2Cn^2}$ once the constant in m is large enough and α is small enough. By a union bound over the $m \le n$ buckets, with probability at least $1 - \frac{1}{n}$ the "signed sum" of the small indices in every bucket is at most $\frac{\|x\|_p}{100}$ in absolute value.

8.6.6 Wrapping up

Following the analysis of the buckets $(PDx)_i$, we know that for all i:

- $|(PDx)_i| \le \frac{\|x\|_p}{100}$ if there is no large index in the i-th bucket;
- $|(PDx)_i| = |\sigma(j)(Dx)_j| \pm \frac{\|x\|_p}{100} = |(Dx)_j| \pm \frac{\|x\|_p}{100}$ if exactly one large index j is in the i-th bucket;
- no bucket contains more than one large index.

Since we also conditioned on $\|Dx\|_\infty \in [\frac{\|x\|_p}{10^{1/p}}, 10^{1/p}\|x\|_p]$, we conclude that

$$\|PDx\|_\infty \le 10^{1/p}\|x\|_p + \frac{\|x\|_p}{100}, \tag{8.13}$$

$$\|PDx\|_\infty \ge \frac{\|x\|_p}{10^{1/p}} - \frac{\|x\|_p}{100}. \tag{8.14}$$

For (8.13), every bucket is at most $\|Dx\|_\infty + \frac{\|x\|_p}{100}$. For (8.14), note that the coordinate j achieving $\|Dx\|_\infty \ge \|x\|_p/10^{1/p}$ is large, since $\frac{\alpha}{\log n} < 10^{-1/p}$ for n larger than a constant. So its bucket has absolute value at least $\|Dx\|_\infty - \frac{\|x\|_p}{100}$.

Therefore $\|PDx\|_\infty$ is a good estimate of $\|x\|_p$, up to a constant factor: since $p > 2$, (8.13) and (8.14) put it between $0.3\|x\|_p$ and $3.2\|x\|_p$. We output $\|PDx\|_\infty$ as the result of the algorithm. The overall space is $m = O(n^{1-2/p}\log n)$ words, that is, $O(n^{1-2/p}\log^2 n)$ bits.

We make two final remarks. First, the sketch is maintained in the stream like the others: an update (i_j, Δ_j) adds $\Delta_j \sigma(i_j)/E_{i_j}^{1/p}$ to bucket $h(i_j)$. Here we treat the E_i, like h and σ, as truly random, and we do not discuss how to store them or how to round the real-valued buckets to $O(\log n)$-bit words. Second, by a union bound, the estimate is within these constant factors with probability at least $\frac{7}{10} - o(1)$. Event 1 fails with probability at most $\frac{1}{5}$, events 2 and 4 each with probability at most $\frac{1}{20}$, event 3 with probability $o(1)$, and the bound on the small indices in every bucket with probability at most $\frac{1}{n}$.

9 Heavy Hitters

Lecture 9, November 5, 2020, by David P. Woodruff. Compiled from the scribe note of Misha Khodak, and from the lecture slides. One scribe note covers both halves; material marked as added is filled in from the lecture slides, or supplies a standard step that the lecture left out.

9.1 Recap: norm estimation for $p > 2$

The lecture began by recalling the analysis of the estimator for $\|x\|_p$, $p > 2$, which is developed in full in Lecture 8. The goal is to estimate $\|x\|_p$ for the stream vector x up to a constant factor. The sketch is PDx, where $D = \mathrm{diag}(1/E_1^{1/p}, \ldots, 1/E_n^{1/p})$ with $E_1, \ldots, E_n \sim \mathrm{Exp}(1)$ i.i.d., and $P \in \mathbb{R}^{m \times n}$ is a CountSketch matrix (not a projection) with $m = n^{1-2/p} \log n$ rows; we showed that $\|Dx\|_\infty = \Theta(\|x\|_p)$ with constant probability. Suppose $|(Dx)_j| = \|Dx\|_\infty$, that is, j is the coordinate realizing the maximum. CountSketch hashes this coordinate to a bucket, and the estimator is the largest bucket value in absolute value, so we needed to show that the noise in the buckets is small. To do this we split the coordinates j' of Dx into *heavy* ones, with $|(Dx)_{j'}| > \alpha \|x\|_p / \log n$ for a small constant α, and *light* ones (all the others; Lecture 8 calls them large and small). The heavy coordinates do not concentrate, but there are only $O(\log^p n)$ of them with probability $9/10$, so they are likely to go to separate buckets, that is, they are perfectly hashed. For the light coordinates, Bernstein's inequality bounds the noise in every bucket by $\|x\|_p/100$ with high probability. The two bounds are combined at the end of Lecture 8.

9.2 Heavy hitters and their guarantees

The rest of the lecture is about finding the items that occur very frequently in a stream, for example a destination on the web that receives heavy traffic. As before, $x \in \mathbb{Z}^n$ is the vector defined by the stream (Lecture 7). There are several possible guarantees; $\phi \in (0, 1)$ is a threshold and $\varepsilon \in (0, \phi)$ a slack parameter.

Definition 9.1 (Heavy hitter guarantees)**.**

- The ℓ_1-*guarantee*: output a set containing every j with $|x_j| \geq \phi \|x\|_1$, and containing no j with $|x_j| \leq (\phi - \varepsilon)\|x\|_1$.
- The ℓ_2-*guarantee*: output a set containing every j with $x_j^2 \geq \phi \|x\|_2^2$, and containing no j with $x_j^2 \leq (\phi - \varepsilon)\|x\|_2^2$.

Items strictly between the two thresholds may or may not be output. This relaxation is needed: there is a lower bound by a reduction from the indexing problem (Lecture 11), using a stream in which whether or not an item is a heavy hitter depends on whether it occurred in a long stretch of the stream.

The ℓ_2-guarantee can be much stronger than the ℓ_1-guarantee. Consider the vector $x = (\sqrt{n}, 1, 1, \ldots, 1) \in \mathbb{R}^n$. Then $x_1^2 = n \geq \frac{1}{2}(2n - 1) = \frac{1}{2}\|x\|_2^2$, so item 1 is an ℓ_2-heavy hitter for the constant $\phi = 1/2$. But $|x_1|/\|x\|_1 = \sqrt{n}/(\sqrt{n} + n - 1) < 1/\sqrt{n}$, so to report item 1 under the ℓ_1-guarantee we would need $\phi \approx 1/\sqrt{n}$. This is a problem, since the algorithms below use memory proportional to $1/\phi$. In general, every ℓ_1-heavy hitter is an ℓ_2-heavy hitter with ϕ^2 in place of ϕ: if $|x_j| \geq \phi \|x\|_1$, then squaring both sides gives

$$x_j^2 \geq \phi^2 \|x\|_1^2 \geq \phi^2 \|x\|_2^2.$$

In principle one could go beyond, to ℓ_p-guarantees for $p > 2$, but as for norm estimation the memory then grows as $n^{1-2/p}$.

9.3 Intuition: finding a single heavy item

Suppose we are promised that at the end of the stream some coordinate has $x_i = n$, and $x_{i'} \in \{0, 1\}$ for every $i' \neq i$. How do we find i? For each $j \in [\log n]$, let $A_j \subseteq [n]$ be the set of indices whose binary representation has j-th bit equal to 0, and B_j the set of indices whose j-th bit is 1. We maintain

$$a_j = \sum_{i' \in A_j} x_{i'} \qquad \text{and} \qquad b_j = \sum_{i' \in B_j} x_{i'} \qquad \text{for each } j \in [\log n],$$

which takes $\log n$ counters of $O(\log n)$ bits each, so $O(\log^2 n)$ bits in total. We recover i as the number whose j-th bit is 1 if $b_j > a_j$ and 0 otherwise. Why does this work? If the j-th bit of i is 1, then $i \in B_j$, so $b_j \geq x_i = n$, while a_j is a sum of fewer than n entries from $\{0, 1\}$, so $a_j \leq n - 1 < b_j$. If the j-th bit of i is 0, then $a_j > b_j$ in the same way.

Now suppose instead that we are promised $x_i = 100\sqrt{n \log n}$ and $x_{i'} \in \{0, 1\}$ for $i' \neq i$. The previous approach no longer works, since the value x_i is too small compared to the up to $n - 1$ ones. But we can use a similar sketch with random signs: pick $\sigma(i') \sim \text{Unif}\{-1, 1\}$ independently for each $i' \in [n]$, and maintain

$$a_j = \sum_{i' \in A_j} \sigma(i') x_{i'} \qquad \text{and} \qquad b_j = \sum_{i' \in B_j} \sigma(i') x_{i'} \qquad \text{for each } j \in [\log n],$$

again with $O(\log^2 n)$ bits. The "noise" in a counter, the part of the sum coming from coordinates other than i, is a sum of at most n independent random signs, so by the additive Chernoff bound its magnitude is at most $\sqrt{n \log n}$ in every counter, with high probability. So we again recover i, now with j-th bit 1 if $|b_j| > |a_j|$ and 0 otherwise. The counter that contains i has magnitude at least $100\sqrt{n \log n} - \sqrt{n \log n} = 99\sqrt{n \log n}$, while the other counter of the pair has magnitude at most $\sqrt{n \log n}$.

Next we remove both assumptions: the known value of x_i, and the values $\{0, 1\}$ elsewhere.

9.4 CountSketch achieves the ℓ_2-guarantee

Assign each coordinate i a random sign $\sigma(i) \in \{-1, 1\}$, and randomly partition the coordinates into B buckets with a hash function $h \colon [n] \to [B]$. In the j-th bucket we maintain the counter

$$c_j = \sum_{i \colon h(i) = j} \sigma(i) x_i, \qquad j \in [B].$$

In other words, $(c_1, \ldots, c_B) = Sx$ for the CountSketch matrix $S \in \mathbb{R}^{B \times n}$ with $S_{h(i),i} = \sigma(i)$ (Lecture 3). So the counters are easy to maintain in a turnstile stream: the update (i, Δ) adds $\sigma(i)\Delta$ to $c_{h(i)}$. We estimate x_i by $\sigma(i) c_{h(i)}$. Figure 6 shows an example.

Lemma 9.2. *Suppose that h and σ are each pairwise independent, and independent of each other. Then for every $i \in [n]$, $\mathbb{E}[\sigma(i) c_{h(i)}] = x_i$, and the noise $N_i = \sigma(i) c_{h(i)} - x_i$ satisfies $\mathbb{E}[N_i^2] \leq \|x\|_2^2 / B$. Consequently,*

$$\Pr\left[|\sigma(i) c_{h(i)} - x_i| \geq \frac{10 \|x\|_2}{\sqrt{B}}\right] \leq \frac{1}{100}. \tag{9.1}$$

Proof. Since $\sigma(i)^2 = 1$,

$$\mathbb{E}[\sigma(i) c_{h(i)}] = \mathbb{E}\left[\sigma(i) \sum_{i' \colon h(i') = h(i)} \sigma(i') x_{i'}\right] = \mathbb{E}\left[\sigma(i)^2 x_i + \sum_{i' \neq i \colon h(i') = h(i)} \sigma(i)\sigma(i') x_{i'}\right] = x_i,$$

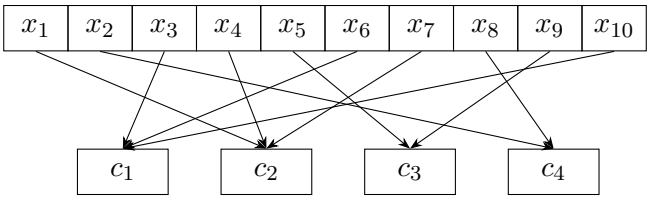


Figure 6: CountSketch with $B = 4$ buckets and an example hash function $h = (2, 4, 1, 2, 3, 1, 2, 4, 3, 1)$: bucket 2 holds $c_2 = \sum_{i:\, h(i)=2} \sigma(i)x_i$.

because $\mathbb{E}[\sigma(i)\sigma(i')] = 0$ for $i' \neq i$. The noise in the bucket of i is $N_i = \sigma(i) \sum_{i' \neq i:\, h(i')=h(i)} \sigma(i')x_{i'}$, and its variance is

$$\mathbb{E}[N_i^2] = \mathbb{E}\left[\Big(\sigma(i) \sum_{i' \neq i:\, h(i')=h(i)} \sigma(i')x_{i'}\Big)^2\right] = \sum_{i' \neq i} x_{i'}^2 \Pr[h(i') = h(i)] \leq \frac{\|x\|_2^2}{B}.$$

When we expand the square, the cross terms $\sigma(i')\sigma(i'')$ with $i' \neq i''$ have expectation 0, and $\Pr[h(i') = h(i)] = 1/B$ by pairwise independence of h. Chebyshev's inequality now gives (9.1). □

So with constant probability the noise in a bucket is $O(\|x\|_2/\sqrt{B})$ in magnitude. To boost the probability, we independently repeat the hashing scheme $O(\log n)$ times and output the median of the estimates across the repetitions:

$$\hat{x}_i = \text{median}\,\{\sigma(i)c_{h(i)} \text{ over the } O(\log n) \text{ repetitions}\}.$$

Since the repetitions are independent, $\hat{x}_i = x_i \pm O(\|x\|_2/\sqrt{B})$ with probability $1 - 1/\operatorname{poly}(n)$. The median is off by more than $10\|x\|_2/\sqrt{B}$ only if at least half of the repetitions are. By the Chernoff bound, this happens with probability at most $1/\operatorname{poly}(n)$, and a union bound over the n coordinates gives the following.

Theorem 9.3 (CountSketch, ℓ_2 error)**.** *With $O(\log n)$ independent repetitions of B buckets each, with probability $1 - 1/\operatorname{poly}(n)$,*

$$|\hat{x}_i - x_i| \leq \frac{10\|x\|_2}{\sqrt{B}} \qquad \textit{for every } i \in [n] \textit{ simultaneously.}$$

The data structure stores $O(B \log n)$ counters of $O(\log n)$ bits each, so $O(B \log^2 n)$ bits, plus $O(\log n)$ bits per repetition for h and σ.

Thus CountSketch approximates every x_i simultaneously, with additive error $O(\|x\|_2/\sqrt{B})$. Using $B = O(1/\phi)$ buckets with a large enough constant, the error is a small fraction of $\sqrt{\phi}\|x\|_2$, so we can distinguish the cases $|x_i| \leq \sqrt{\phi/2}\|x\|_2$ and $|x_i| \geq \sqrt{\phi}\|x\|_2$, by comparing $|\hat{x}_i|$ with a threshold between the two. This solves the ℓ_2-heavy hitters problem with slack $\varepsilon = \phi/2$, in $O(\phi^{-1} \log^2 n)$ bits. Note that the threshold is a multiple of $\|x\|_2$. We can replace $\|x\|_2$ by a (1 ± 0.01)-estimate, maintained with the norm estimators of Lecture 8, and this changes only the constant in B.

9.4.1 The tail guarantee

The guarantee of Theorem 9.3 can be poor. Suppose x_1 is a very large $\operatorname{poly}(n)$, $x_2 = n$, and $x_3 = \cdots = x_n = 1$, and we want to recover both x_1 and x_2. Then $\|x\|_2 \geq x_1 = \operatorname{poly}(n)$, so the

additive error $O(\|x\|_2/\sqrt{B})$ swamps x_2, which is poorly approximated. However, the same data structure has a much better guarantee. Write $x_{-B/4}$ for the vector x with its $B/4$ largest coordinates (in magnitude) set to 0.

Theorem 9.4 (Tail guarantee)**.** *With probability* $1 - 1/\operatorname{poly}(n)$*, the CountSketch estimates satisfy*

$$|\hat{x}_i - x_i| = O\Big(\frac{\|x_{-B/4}\|_2}{\sqrt{B}}\Big) \qquad \text{for every } i \in [n] \text{ simultaneously.}$$

Proof. With probability at least $3/4$, in each repetition none of the top $B/4$ coordinates of x (other than i itself) lands in the same bucket as i. By pairwise independence, each of them lands in the bucket of i with probability $1/B$, so by a union bound over the at most $B/4$ of them, this fails with probability at most $1/4$. Let H be the set of the top $B/4$ coordinates, let G be the event that no $i' \in H \setminus \{i\}$ has $h(i') = h(i)$, and let $N_i' = \sigma(i)\sum_{i' \notin H \cup \{i\}:\, h(i')=h(i)} \sigma(i')x_{i'}$ be the noise from the coordinates outside H. On G the noise N_i equals N_i'. The computation in the proof of Lemma 9.2 gives $\mathbb{E}[(N_i')^2] \le \|x_{-B/4}\|_2^2/B$, so by Chebyshev's inequality $|N_i'| < 10\|x_{-B/4}\|_2/\sqrt{B}$ with probability at least $99/100$. Hence in each repetition

$$\Pr\Big[|\sigma(i)c_{h(i)} - x_i| \ge \frac{10\|x_{-B/4}\|_2}{\sqrt{B}}\Big] \le \Pr[\text{not } G] + \frac{1}{100} \le \frac{1}{4} + \frac{1}{100} < \frac{1}{3}.$$

The median over the $O(\log n)$ independent repetitions, followed by a union bound over i, finishes the proof as for Theorem 9.3. □

Note that this argument requires only pairwise independent hash functions, which can be specified with $O(\log n)$ bits. Furthermore, if x is $B/4$-sparse, then $x_{-B/4} = 0$, so the theorem implies that every x_i is estimated exactly.

9.5 Deterministic ℓ_1 heavy hitters

As computed in Lecture 9.2, every ℓ_1-heavy hitter is an ℓ_2-heavy hitter, so the ℓ_2-guarantee is the stronger one. Why care about the ℓ_1-guarantee, then? Because it can be achieved deterministically, unlike our CountSketch-based data structure. To do so, we consider matrices with the following property.

Definition 9.5 (Incoherent matrix)**.** A matrix $S \in \mathbb{R}^{m \times n}$ is *ε-incoherent* if every column has unit norm, $\|S_{*,i}\|_2 = 1$, and every pair of distinct columns has small inner product, $|\langle S_{*,i}, S_{*,j}\rangle| \le \varepsilon$ for all $i \ne j$.

We further want all entries of S to be specified with $O(\log n)$ bits. The algorithm maintains Sx in the stream, using $O(m \log n)$ bits of space, and at the end outputs the estimates $\hat{x}_i = S_{*,i}^\top Sx$.

Lemma 9.6. *If S is ε-incoherent, then $|\hat{x}_i - x_i| \le \varepsilon\|x\|_1$ for every $i \in [n]$.*

Proof. By ε-incoherence,

$$\hat{x}_i = \sum_{j=1}^{n} \langle S_{*,i}, S_{*,j}\rangle x_j = \|S_{*,i}\|_2^2\, x_i \pm \max_{j \ne i} |\langle S_{*,i}, S_{*,j}\rangle| \cdot \|x\|_1 = x_i \pm \varepsilon\|x\|_1. \tag{9.2}$$

□

So we can tell which $|x_i| \ge \phi\|x\|_1$ and which $|x_i| \le (\phi - \varepsilon)\|x\|_1$, which solves the ℓ_1-heavy hitters problem. We use an $(\varepsilon/3)$-incoherent matrix, and output every i with $|\hat{x}_i| \ge (\phi - \varepsilon/2)\|x\|_1$. An i with $|x_i| \ge \phi\|x\|_1$ has $|\hat{x}_i| \ge (\phi - \varepsilon/3)\|x\|_1$, and an i with $|x_i| \le (\phi - \varepsilon)\|x\|_1$ has $|\hat{x}_i| \le (\phi - 2\varepsilon/3)\|x\|_1$. The threshold needs $\|x\|_1$. When all entries of x are nonnegative, as in an insertion-only stream, $\|x\|_1 = \sum_i x_i$ is the sum of all updates, and one counter maintains it exactly.

9.5.1 Constructing incoherent matrices

But do ε-incoherent matrices exist? We construct one from polynomials over a finite field. (In this section d is a degree bound; the data matrix A of earlier lectures does not appear.)

Lemma 9.7. *For every $\varepsilon \in (0,1)$ there is an ε-incoherent matrix $S \in \mathbb{R}^{m \times n}$ with $m = O(\varepsilon^{-2} \log^2 n)$ rows.*

Proof. Consider a prime $q = \Theta((\log n)/\varepsilon)$, and let $d = \lfloor \varepsilon q \rfloor = \Theta(\log n)$. Let $p_1, \ldots, p_n$ be n distinct nonzero polynomials of degree less than d over the field $\mathbb{F}_q$, $p_i(z) = \sum_{\ell=0}^{d-1} a_{i,\ell} z^\ell \bmod q$. There are $q^d - 1$ such polynomials, and we choose the constant in q so that $q^d - 1 > n$. Associate p_i with the i-th column of S, let $m = q^2$, and partition the rows of S into q groups of q rows each, the groups indexed by $j \in \mathbb{F}_q$ and the rows within a group by $\mathbb{F}_q$. In the j-th group, the i-th column has a single nonzero entry, equal to $1/\sqrt{q}$, in row $p_i(j)$ of the group; all other entries are 0.

Each column $S_{*,i}$ has q nonzero entries equal to $1/\sqrt{q}$, so $\|S_{*,i}\|_2 = 1$. Columns $S_{*,i}$ and $S_{*,i'}$ share a nonzero entry in the j-th group if and only if $p_i(j) = p_{i'}(j)$, so their inner product is the number of $j \in \mathbb{F}_q$ with $p_i(j) = p_{i'}(j)$, multiplied by $1/q$. Since $p_i - p_{i'}$ is a nonzero polynomial of degree at most $d-1$, it has at most $d-1$ roots, so the inner product is at most $(d-1)/q < \varepsilon$, using $d \le \varepsilon q$. Thus S is ε-incoherent, with $m = q^2 = O(\varepsilon^{-2} \log^2 n)$ rows. □

The entries of S are 0 or $1/\sqrt{q}$, so the algorithm can store the integer vector $\sqrt{q}\, Sx$ instead of Sx, and divide by q when it computes $\hat{x}_i$. Also, S itself need not be stored, since the column $S_{*,i}$ can be generated from i: for example, we can take the base-q digits of i as the coefficients of p_i.

9.6 Finding the heavy hitters quickly

The algorithms so far compute an estimate of every x_i and then cycle through all of them to find the heavy hitters. This is memory-efficient, but it needs $\Omega(n)$ time after the stream ends. We can find the top k heavy hitters much faster with a tree-like data structure (Figure 7).

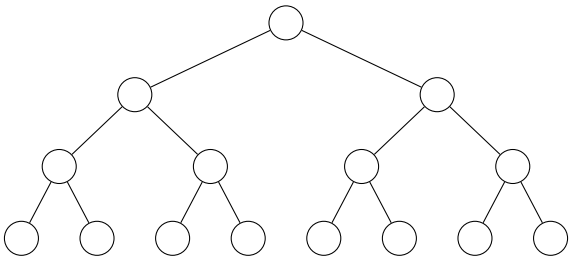


Figure 7: A full binary tree over $[n]$ (here $n = 8$). Level t has 2^t nodes, and a node at level t stands for the $n/2^t$ indices whose binary representations share a t-bit prefix.

Consider the full binary tree whose leaves are the coordinates $[n]$. There are 2^t nodes at level t, and each node at level t corresponds to the subset of $[n]$ of size $n/2^t$ consisting of the indices with the same t-bit prefix. For a node v, let x_v be the restriction of x to the indices of v. We start at the level with $2k$ nodes, which partitions the coordinates into $2k$ groups of size $n/(2k)$. At each level t, we hash the 2^t nodes into $O(k)$ buckets, and repeat this $O(\log n)$ times. This is like CountSketch, except that in each bucket we run an approximation algorithm for the 2-norm (Lecture 8) of the part of x that hashes there, by applying a sketching matrix to the sum of the vectors x_v of the nodes v in that bucket. As for CountSketch, this estimates the norm of every node with additive error $O(\|x\|_2/\sqrt{k})$, so at each level at most k nodes can look heavy. This is because $\sum_v \|x_v\|_2^2 = \|x\|_2^2$

over the nodes v of a level, so at most k nodes of a level have $\|x_v\|_2^2 \geq \|x\|_2^2/k$. Also, a node that contains a heavy hitter i has $\|x_v\|_2 \geq |x_i|$.

At the top level the universe has only $2k$ nodes, so we find the top k of them by computing an estimate for all of them. The main idea is that at the next level we only need to consider the left and right children of each of the k nodes found at the previous level, so only $2k \ll n$ nodes. Going down the $\log n$ levels in this way, we examine $O(k)$ nodes per level, so $O(k \log n)$ nodes in total, and the leaves found at the bottom level are the heavy hitters. Each node estimate is a median over the $O(\log n)$ repetitions, so the search evaluates $O(k \log^2 n)$ bucket sketches, which takes $O(k \log^2 n)$ time when each bucket sketch has $O(1)$ counters (see below). The data structure keeps one such table per level, so its space is $O(k \log^4 n)$ bits: there are $\log n$ levels, $O(k \log n)$ buckets per level, and a 2-norm sketch in each bucket with $O(\log n)$ counters of $O(\log n)$ bits. In fact, a 2-norm sketch with $O(1)$ counters that succeeds with constant probability already suffices, since the median over repetitions boosts the success probability. This gives $O(k \log^3 n)$ bits.

10 Estimating the Number of Non-Zeros, and Information Theory

Lecture 10, November 12, 2020, by David P. Woodruff. Compiled from the scribe notes of Zheng Xu (first half) and Sylvia Zhang (second half), and from the lecture slides.

10.1 Estimating the number of non-zero entries

For the stream vector $x \in \mathbb{Z}^n$ (Lecture 7) we want to estimate

$$\|x\|_0 = |\{i : x_i \neq 0\}|,$$

the number of non-zero entries of x (usually called the ℓ_0 norm, although it is not a norm). As usual, we want to output a number Z with

$$(1-\varepsilon)\|x\|_0 \leq Z \leq (1+\varepsilon)\|x\|_0$$

with probability at least $9/10$, and we would like to use $O((\log n)/\varepsilon^2)$ bits of space.

In the simpler case $\|x\|_0 = O(1/\varepsilon^2)$, we can recover x exactly, using either the algorithm based on a Vandermonde matrix for recovering a k-sparse vector (Lecture 7) or CountSketch (the tail guarantee of Lecture 9), with $k = O(1/\varepsilon^2)$, and then count its non-zero entries. But we also want an algorithm when $\|x\|_0 \gg 1/\varepsilon^2$. We will sample coordinates of x and solve the problem on the sampled vector.

10.1.1 Estimation given a rough estimate

Suppose we are given an estimate Z with $Z \leq \|x\|_0 \leq 2Z$. Given this information, we want to sample coordinates of x so that the sampled vector has $O(1/\varepsilon^2)$ non-zero entries. We independently sample each coordinate of x with probability $p = 100/(Z\varepsilon^2)$. Let Y_i be the indicator random variable of the event that coordinate i is sampled, and let y be the vector x restricted to the coordinates i with $Y_i = 1$. Then

$$\begin{aligned}
\mathbb{E}[\|y\|_0] &= \sum_{i:\, x_i \neq 0} \mathbb{E}[Y_i] = p\|x\|_0 = \frac{100}{Z\varepsilon^2}\|x\|_0 \in \left[\frac{100}{\varepsilon^2}, \frac{200}{\varepsilon^2}\right], \\
\mathrm{Var}[Y_i] &\leq \mathbb{E}[Y_i^2] = \Pr[Y_i = 1] = p, \\
\mathrm{Var}[\|y\|_0] &= \sum_{i:\, x_i \neq 0} \mathrm{Var}[Y_i] \leq p\|x\|_0 \leq \frac{200}{\varepsilon^2}.
\end{aligned}$$

By Chebyshev's inequality,

$$\Pr\left[\left|\|y\|_0 - \mathbb{E}[\|y\|_0]\right| > \frac{100}{\varepsilon}\right] \leq \frac{\mathrm{Var}[\|y\|_0]\,\varepsilon^2}{100^2} \leq \frac{1}{50}.$$

Thus, with probability at least $49/50$, $\|y\|_0 \in \mathbb{E}[\|y\|_0] \pm 100/\varepsilon$, and since $\mathbb{E}[\|y\|_0] \geq 100/\varepsilon^2$, this means $\|y\|_0 = (1 \pm \varepsilon)\,\mathbb{E}[\|y\|_0] = (1 \pm \varepsilon)p\|x\|_0$. In particular y has $O(1/\varepsilon^2)$ non-zero entries, so we can compute $\|y\|_0$ exactly using sparse recovery or CountSketch, run on the updates to the sampled coordinates. Multiplying by $1/p$, we get $\frac{1}{p}\|y\|_0 \in (1 \pm \varepsilon)\|x\|_0$.

10.1.2 Finding the right guess

But we do not know Z. Because $0 \le \|x\|_0 \le n$, there are $O(\log n)$ powers of 2 in the range, so we guess Z in powers of 2 and run the algorithm in parallel for all $O(\log n)$ guesses. The i-th guess $Z = 2^i$ corresponds to sampling each coordinate with probability

$$p_i = \min\Big(1, \frac{100}{2^i \varepsilon^2}\Big).$$

Let S_i be the set of coordinates sampled for $Z = 2^i$. We can ensure that the sets are nested, $[n] = S_0 \supseteq S_1 \supseteq \cdots \supseteq S_{\log n}$. The sets S_i and S_j are not independent for $i \neq j$, but within each set S_i the coordinates are sampled independently. We compute an estimate of $\|x\|_0$ from each set S_i; one of the guesses satisfies $Z \le \|x\|_0 \le 2Z$, and we should use that one.

The problem comes down to deciding which of the answers is correct. Let y^i denote the vector x restricted to the coordinates in S_i. Because the subsets are nested,

$$\|y^0\|_0 \ge \|y^1\|_0 \ge \cdots \ge \|y^{\log n}\|_0.$$

The rule is: use the largest guess i for which

$$\frac{400}{\varepsilon^2} \le \|y^i\|_0 \le \frac{3200}{\varepsilon^2}, \tag{10.1}$$

call it i^*, and output $\|y^{i^*}\|_0 / p_{i^*}$.

Theorem 10.1. *Let $\varepsilon \le 1$, and suppose that $\|x\|_0 \ge 3200/\varepsilon^2$ (smaller values are covered by the sparse case above). With probability at least $4/5$, the output $\|y^{i^*}\|_0/p_{i^*}$ is a $(1 \pm \varepsilon)$-approximation of $\|x\|_0$.*

Proof. For any i, Chebyshev's inequality gives

$$\Pr\Big[\big|\|y^i\|_0 - \mathbb{E}[\|y^i\|_0]\big| \ge t\Big] \le \frac{\mathrm{Var}[\|y^i\|_0]}{t^2} \le \frac{\mathbb{E}[\|y^i\|_0]}{t^2},$$

where the last inequality holds because $\|y^i\|_0$ is a sum of independent indicator random variables. Let i' be such that $800/\varepsilon^2 \le \mathbb{E}[\|y^{i'}\|_0] \le 1600/\varepsilon^2$. Such an i' exists, and has $p_{i'} < 1$. Levels with $p_i = 1$ have $\mathbb{E}[\|y^i\|_0] = \|x\|_0 \ge 3200/\varepsilon^2$. The first level with $p_i < 1$ has $p_i \ge 1/2$, so $\mathbb{E}[\|y^i\|_0] \ge 1600/\varepsilon^2$ there. From that level on, the expectation $p_i\|x\|_0$ halves from one level to the next, down to at most $100/\varepsilon^2$ at level $\log n$. In particular, $\mathbb{E}[\|y^{i'+3}\|_0] = \mathbb{E}[\|y^{i'}\|_0]/8$. Picking $t = 400/\varepsilon^2$, we obtain from the above that

$$\frac{400}{\varepsilon^2} \le \|y^{i'}\|_0 \le \frac{2000}{\varepsilon^2}$$

with probability at least $1 - \varepsilon^2/100$. We also have $100/\varepsilon^2 \le \mathbb{E}[\|y^{i'+3}\|_0] \le 200/\varepsilon^2$, from which we obtain in the same way (with $t = 200/\varepsilon^2$) that $\|y^{i'+3}\|_0 < 400/\varepsilon^2$ with probability at least $1 - \varepsilon^2/200$. Therefore, by a union bound, with probability at least $1 - O(\varepsilon^2)$, both $400/\varepsilon^2 \le \|y^{i'}\|_0 \le 2000/\varepsilon^2$ and $\|y^{i'+3}\|_0 < 400/\varepsilon^2$ hold. Then $i' \le i^* \le i' + 3$, by monotonicity of $\|y^i\|_0$ and the fact that i^* is the largest index satisfying (10.1).

It remains to check that all four guesses $Z = 2^{i'}, 2^{i'+1}, 2^{i'+2}, 2^{i'+3}$ give a $(1 \pm \varepsilon)$-approximation, so that it does not matter which of them is i^*. Each of them has $\mathbb{E}[\|y^i\|_0] \ge 100/\varepsilon^2$. So Chebyshev's inequality with $t = \varepsilon\,\mathbb{E}[\|y^i\|_0]$ shows that $\|y^i\|_0 = (1 \pm \varepsilon)\,\mathbb{E}[\|y^i\|_0] = (1 \pm \varepsilon)p_i\|x\|_0$, except with probability at most $1/(\varepsilon^2\,\mathbb{E}[\|y^i\|_0]) \le 1/100$. By a union bound, all four estimates are $(1 \pm \varepsilon)$-approximations simultaneously with probability at least $1 - 4/100$. So the overall success probability is at least $1 - O(\varepsilon^2) - 4/100 > 4/5$. □

The sparse recovery at levels below i' may be run on vectors with more than $3200/\varepsilon^2$ non-zero entries, and then return garbage. This does not matter: the rule picks the largest level satisfying (10.1), and on the event above every level from i' on has at most $2000/\varepsilon^2$ non-zero entries, where the recovery (with $k = 3200/\varepsilon^2$) is exact.

10.1.3 Overall space complexity

We run k-sparse recovery at each of the $O(\log n)$ levels, and each k-sparse recovery uses $O(k \log n)$ bits. Here $k = O(1/\varepsilon^2)$, so overall $O(\varepsilon^{-2} \log^2 n)$ bits of space are used, ignoring random bits.

Now consider the amount of randomness the algorithm needs. We only used Chebyshev's inequality, and Chebyshev's inequality only requires pairwise independence. We achieve pairwise independence as follows. Suppose we have a pairwise independent hash function $h\colon [n] \to [n]$, and we want to sample each coordinate with probability $1/2$. We can use $h(j) \bmod 2$ to decide whether to sample coordinate j: sample it if $h(j) \bmod 2 = 0$, and do not sample it otherwise. For the set S_i, we sample coordinate j when the last i digits of the binary representation of $h(j)$ are zero. This way of sampling also produces nested subsets, and needs only the $O(\log n)$ bits of space that store h. Note that this samples with probability 2^{-i}, so we round the probabilities p_i above to powers of 2, which changes only the constants.

The space can be improved to

$$O\Big(\frac{\log n\,(\log(1/\varepsilon) + \log\log n)}{\varepsilon^2}\Big)$$

bits by reducing the size of the counters: at the levels that matter, we just need to know the number of non-zero counters, so the counters can shrink from $O(\log n)$ bits to $O(\log(1/\varepsilon) + \log\log n)$ bits.

10.1.4 Reducing the counter size

In sparse recovery we use a Vandermonde matrix (Lecture 7), and we reduce the memory by maintaining the sparse recovery structure modulo a small random prime. In the sampling levels that we care about, we have $O(1/\varepsilon^2)$ counters, each of $O(\log n)$ bits. We want a prime that divides none of the non-zero ones among these $O(1/\varepsilon^2)$ counters, so that every non-zero counter stays non-zero modulo the prime. A non-zero integer of magnitude $\mathrm{poly}(n)$ has at most $O(\log n)$ distinct prime divisors, so there are in total $O(\varepsilon^{-2} \log n)$ primes that can divide any of the counters. We choose a prime q uniformly at random from the primes in $\{2, \dots, L\}$, which avoids these primes with good probability: there are $\Theta(L/\log L)$ primes in $\{2, \dots, L\}$, and we choose

$$L = O\Big(\frac{\log n\,(\log\log n + \log(1/\varepsilon))}{\varepsilon^2}\Big)$$

such that $L/\log L$ exceeds the number of bad primes by a large constant factor. Then it is unlikely that q divides any non-zero counter. In this case, we just need to maintain our sparse recovery structure modulo q, which takes $O(\log q) = O(\log(1/\varepsilon) + \log\log n)$ bits per counter, so $O(\varepsilon^{-2}(\log(1/\varepsilon) + \log\log n))$ bits for each of the $O(\log n)$ sparse recovery instances.

10.2 Information theory

The second half of the lecture develops the tools for a communication lower bound. Consider the indexing problem: Alice has a vector $x \in \{0,1\}^n$, and Bob has an index $i \in [n]$; the goal is $\textsc{Index}(x, i) = x_i$. Alice sends a single message M to Bob, based only on her vector x, and Bob computes an output $f(M, i) = \hat{x}_i$. If $\hat{x}_i = x_i$ with probability at least $2/3$, then we want to show

that $|M| = \Omega(n)$; this is done in Lecture 11. The technique used to prove such lower bounds is *information theory*. In this section, log is base 2.

10.2.1 Entropy

Consider a distribution p over a finite support, which we take to be $[N]$ (we write N for the support size, since n is the length of the stream vector in this lecture): $p = (p_1, \ldots, p_N)$ with $p_i \in [0,1]$ and $\sum_i p_i = 1$. A random variable X has distribution p if $\Pr[X = i] = p_i$. We write $\mathcal{X}$ for the support of X.

Definition 10.2 (Entropy)**.** Let X be a random variable with distribution p on N items. The *entropy* of X is

$$H(X) = \sum_i p_i \log \frac{1}{p_i},$$

where a term with $p_i = 0$ is $p_i \log(1/p_i) = 0$ (as in the limit $p_i \to 0$).

Entropy captures the uncertainty of a distribution. For example, a uniform distribution has high entropy, while a constant one has low entropy.

Claim 10.3. *$H(X) \le \log N$, with equality when $p_i = 1/N$ for all i. If $p_i = 1$ for some i (and 0 elsewhere), then $H(X) = 0$.*

Proof. The function $f(t) = t \log(1/t)$ is concave on $[0,1]$: the average of function values is at most the function value of the average. Applying this to $p_1, \ldots, p_N$,

$$\frac{1}{N}\sum_{i=1}^{N} p_i \log \frac{1}{p_i} = \frac{1}{N}\sum_{i=1}^{N} f(p_i) \le f\Big(\frac{1}{N}\sum_{i=1}^{N} p_i\Big) = f\Big(\frac{1}{N}\Big) = \frac{1}{N}\log N.$$

Multiplying by N gives $H(X) \le \log N$. The uniform distribution attains equality, since then $H(X) = \sum_i \frac{1}{N} \log N = \log N$. If $p_i = 1$ for some i, every term of $H(X)$ is 0. □

Definition 10.4 (Binary entropy)**.** If B is a bit with bias p (that is, $\Pr[B = 1] = p$), then

$$H(B) = H(p) = p \log \frac{1}{p} + (1-p) \log \frac{1}{1-p}.$$

The binary entropy is symmetric, $H(p) = H(1-p)$, and is maximized at $p = 1/2$, where $H(1/2) = 1$ (Figure 8). It is also increasing on $[0, 1/2]$, which we use in Lecture 11.

Let X and Y be two random variables.

Definition 10.5 (Conditional entropy)**.** The *conditional entropy* of X given Y is the uncertainty of X, averaged over the choices of Y:

$$H(X \mid Y) = \sum_y H(X \mid Y = y) \Pr[Y = y].$$

Definition 10.6 (Joint entropy)**.** The *joint entropy* of X and Y is the entropy of the joint distribution:

$$H(X,Y) = \sum_{x,y} \Pr[(X,Y) = (x,y)] \log \frac{1}{\Pr[(X,Y) = (x,y)]}.$$

Fact 10.7 (Chain rule for entropy)**.** $H(X,Y) = H(X) + H(Y \mid X)$.

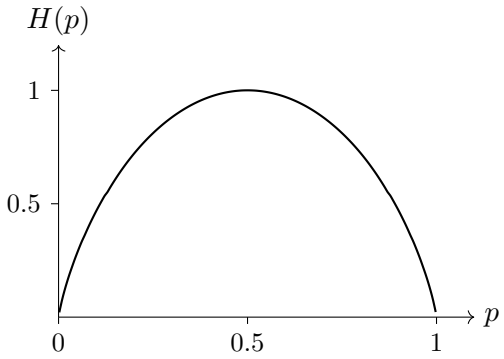


Figure 8: The binary entropy function $H(p)$.

Proof.

$$H(X,Y) = \sum_{x,y} \Pr[(X,Y)=(x,y)] \log \frac{1}{\Pr[(X,Y)=(x,y)]} \tag{10.2}$$

$$= \sum_{x,y} \Pr[X=x] \Pr[Y=y \mid X=x] \log \frac{1}{\Pr[X=x] \Pr[Y=y \mid X=x]} \tag{10.3}$$

$$= \sum_{x,y} \Pr[X=x] \Pr[Y=y \mid X=x] \Big(\log \frac{1}{\Pr[X=x]} + \log \frac{1}{\Pr[Y=y \mid X=x]} \Big) \tag{10.4}$$

$$= H(X) + H(Y \mid X). \tag{10.5}$$

Here (10.2) is the definition of joint entropy, (10.3) uses the definition of conditional probability, (10.4) splits the logarithm, and (10.5) splits the sum over x and y and uses the definitions of entropy and conditional entropy. □

10.2.2 Conditioning cannot increase entropy

Fact 10.8 (Conditioning cannot increase entropy)**.** *$H(X \mid Y) \le H(X)$, with equality if X and Y are independent.*

Informally, you know more after conditioning. The proof uses Jensen's inequality: if f is a continuous concave function and $p_1, \ldots, p_N$ are nonnegative reals that sum to 1, then for any $x_1, \ldots, x_N$,

$$\sum_{i=1}^{N} p_i f(x_i) \le f\Big(\sum_{i=1}^{N} p_i x_i \Big).$$

Recall that f is concave if $f(\frac{a+b}{2}) \ge \frac{f(a)}{2} + \frac{f(b)}{2}$, and that $f(x) = \log x$ is concave.

Proof. All sums over pairs (x, y) below run over the pairs with $\Pr[X = x, Y = y] > 0$.

$$H(X \mid Y) - H(X) = \sum_{x,y} \Pr[Y = y] \Pr[X = x \mid Y = y] \log \frac{1}{\Pr[X = x \mid Y = y]} \tag{10.6}$$

$$- \sum_{x} \Pr[X = x] \log \frac{1}{\Pr[X = x]} \sum_{y} \Pr[Y = y \mid X = x] \tag{10.7}$$

$$= \sum_{x,y} \Pr[X = x, Y = y] \log \frac{\Pr[X = x]}{\Pr[X = x \mid Y = y]} \tag{10.8}$$

$$= \sum_{x,y} \Pr[X = x, Y = y] \log \frac{\Pr[X = x] \Pr[Y = y]}{\Pr[(X, Y) = (x, y)]} \tag{10.9}$$

$$\leq \log \Big(\sum_{x,y} \Pr[X = x, Y = y] \cdot \frac{\Pr[X = x] \Pr[Y = y]}{\Pr[(X, Y) = (x, y)]} \Big) \tag{10.10}$$

$$\leq \log 1 = 0. \tag{10.11}$$

The last sum in (10.7) multiplies $H(X)$ by 1, written in terms of conditional probabilities. (10.8) uses the fact that a difference of logarithms is the logarithm of the ratio, (10.9) uses the definition of conditional probability, and (10.10) is Jensen's inequality for the logarithm. In (10.11), the sum inside the logarithm is the sum of $\Pr[X = x] \Pr[Y = y]$ over the pairs in the support of (X, Y), which is at most 1. If X and Y are independent, the ratio inside the logarithm in (10.9) is 1 for every pair, and $H(X \mid Y) = H(X)$. □

10.2.3 Mutual information

Definition 10.9 (Mutual information)**.** The *mutual information* of X and Y, how much information X reveals about Y, is

$$I(X; Y) = H(X) - H(X \mid Y) = H(Y) - H(Y \mid X) = I(Y; X).$$

Why is it symmetric? By the chain rule, $H(X) + H(Y \mid X) = H(X, Y) = H(Y) + H(X \mid Y)$. If $X = Y$, the mutual information is just the entropy of X, $I(X; X) = H(X) - H(X \mid X) = H(X)$: X reveals everything about Y. If X and Y are independent, the mutual information is 0: X reveals nothing about Y. By Fact 10.8, mutual information is always nonnegative.

Definition 10.10 (Conditional mutual information)**.** $I(X; Y \mid Z) = H(X \mid Z) - H(X \mid Y, Z)$.

How does $I(X; Y \mid Z)$ compare to $I(X; Y)$? Both orders can happen.

Claim 10.11. *For certain X, Y, Z, we have $I(X; Y \mid Z) \leq I(X; Y)$.*

Proof. Consider $X = Y = Z$. Recall that $I(X; Y) = H(X)$. However, once we know Z, there is no information left for Y to reveal about X:

$$I(X; Y \mid Z) = H(X \mid Z) - H(X \mid Y, Z) = 0 - 0 = 0,$$
$$I(X; Y) = H(X) - H(X \mid Y) = H(X) - 0 = H(X).$$ □

Claim 10.12. *For certain X, Y, Z, we have $I(X; Y \mid Z) \geq I(X; Y)$.*

Proof. Let X and Y be independent and uniform in $\{0,1\}$, and let $Z = X + Y \bmod 2$, so that also $X = Y + Z \bmod 2$. Any two of X, Y, Z are independent, but the three are not mutually independent. Then

$$I(X;Y \mid Z) = H(X \mid Z) - H(X \mid Y,Z) = H(X) - 0 = 1 - 0 = 1,$$
$$I(X;Y) = H(X) - H(X \mid Y) = H(X) - H(X) = 1 - 1 = 0. \qquad \square$$

Intuitively, Y only reveals useful information about X after we also condition on Z.

Fact 10.13 (Chain rule for mutual information)**.** $I(X,Y;Z) = I(X;Z) + I(Y;Z \mid X)$.

This is the information that X reveals about Z, plus the information that Y reveals about Z given that we know X.

Proof.

$$\begin{aligned} I(X,Y;Z) &= H(X,Y) - H(X,Y \mid Z) && \text{(definition)} \\ &= H(X) + H(Y \mid X) - H(X \mid Z) - H(Y \mid X,Z) && \text{(chain rule)} \\ &= I(X;Z) + I(Y;Z \mid X) && \text{(definition).} \qquad \square \end{aligned}$$

By induction, this generalizes to n random variables: with $X_{<i} = (X_1, \ldots, X_{i-1})$,

$$I(X_1, \ldots, X_n; Z) = \sum_{i=1}^{n} I(X_i; Z \mid X_{<i}). \tag{10.12}$$

10.2.4 Fano's inequality

Theorem 10.14 (Fano's inequality)**.** *Let $X \to Y \to X'$ be a Markov chain, that is, X' is an estimator of X computed from Y, and X' and X are independent given Y. If $P_e = \Pr[X' \neq X]$, then*

$$H(X \mid Y) \leq H(P_e) + P_e \log(|\mathcal{X}| - 1),$$

where $|\mathcal{X}|$ is the support size of X and $H(P_e)$ is the binary entropy function.

Here $X \to Y \to X'$ being a Markov chain means that X' and X are independent given Y. Think of X as the past, Y as the present, and X' as the future: "past and future are conditionally independent given the present". In the context of Fano's inequality, X' is one's best guess about X given Y.

Consider the special case $X \in \{0,1\}$, so that we are only dealing with one bit going through the channel Y, and Fano's inequality reads $H(X \mid Y) \leq H(P_e)$. Suppose that $Y = X$; then $X' = Y = X$, so $P_e = 0$, and Fano's inequality gives $H(X \mid Y) \leq H(0) = 0$, which checks out. Suppose instead that X is uniform and Y is independent of X. Then every estimator has error probability exactly $1/2$, so $H(P_e) = 1$, and also $H(X \mid Y) = 1$.

To prove Fano's inequality, we need the data processing inequality.

Theorem 10.15 (Data processing inequality)**.** *Suppose $X \to Y \to Z$ is a Markov chain. Then $I(X;Y) \geq I(X;Z)$.*

The moral is that no clever combination of the data can improve the estimation; here Z is the clever combination.

Proof. Expanding $I(X;Y,Z)$ in two ways with the chain rule, taking out Y and Z first respectively,

$$I(X;Y,Z) = I(X;Z) + I(X;Y \mid Z) = I(X;Y) + I(X;Z \mid Y).$$

Therefore it suffices to show that $I(X;Z \mid Y) = 0$, since mutual information is nonnegative (Fact 10.8, applied conditionally on each value of Z). By definition,

$$I(X;Z \mid Y) = H(X \mid Y) - H(X \mid Y,Z).$$

But given Y, the variables X and Z are independent, so $H(X \mid Y,Z) = H(X \mid Y)$. □

Corollary 10.16. *If $X \to Y \to Z$ is a Markov chain, then $H(X \mid Y) \le H(X \mid Z)$.*

Proof. Write the data processing inequality as $H(X) - H(X \mid Y) \ge H(X) - H(X \mid Z)$. □

Proof of Fano's inequality. Let $E = 1$ if $X' \neq X$, and $E = 0$ otherwise. On the one hand,

$$H(E, X \mid X') = H(X \mid X') + H(E \mid X, X') = H(X \mid X'), \tag{10.13}$$

by the chain rule and since E is determined by X and X'. On the other hand, expanding the other way,

$$H(E, X \mid X') = H(E \mid X') + H(X \mid E, X') \le H(P_e) + H(X \mid E, X'), \tag{10.14}$$

where the inequality uses that conditioning cannot increase entropy: $H(E \mid X') \le H(E) = H(P_e)$. Moreover,

$$\begin{aligned} H(X \mid E, X') &= \Pr[E=0]\, H(X \mid X', E=0) + \Pr[E=1]\, H(X \mid X', E=1) \\ &\le (1 - P_e) \cdot 0 + P_e \cdot \log(|\mathcal{X}| - 1), \end{aligned}$$

since given $E = 0$ we have $X = X'$, and given $E = 1$ and $X' = x'$, X takes one of the $|\mathcal{X}| - 1$ values other than x' (Claim 10.3). Combining (10.13) and (10.14), we get $H(X \mid X') \le H(P_e) + P_e \log(|\mathcal{X}| - 1)$. Since $X \to Y \to X'$ is a Markov chain by assumption, Corollary 10.16 gives

$$H(X \mid Y) \le H(X \mid X') \le H(P_e) + P_e \log(|\mathcal{X}| - 1).$$ □

Statistically, Fano's inequality gives a lower bound on the error probability: it measures how well we could predict X, in terms of the entropy of X given Y.

10.2.5 Tightness of Fano's inequality

The following example shows that Fano's inequality is tight. Suppose the distribution p of X on $[N]$ satisfies $p_1 \ge p_2 \ge \cdots \ge p_N$, and suppose Y is a constant, so $I(X;Y) = H(X) - H(X \mid Y) = 0$. The best predictor X' of X is $X' = 1$, with $P_e = \Pr[X' \neq X] = 1 - p_1$. Fano's inequality predicts $H(X \mid Y) \le H(p_1) + (1 - p_1)\log(N-1)$ (recall $H(1-p_1) = H(p_1)$). Since Y is a constant, $H(X \mid Y) = H(X)$, and if $p_2 = p_3 = \cdots = p_N = \frac{1-p_1}{N-1}$, then the inequality is tight:

$$\begin{aligned} X &\sim \Big(p_1, \frac{1-p_1}{N-1}, \ldots, \frac{1-p_1}{N-1}\Big), \\ H(X) &= \sum_i p_i \log \frac{1}{p_i} \\ &= p_1 \log \frac{1}{p_1} + \sum_{i>1} \frac{1-p_1}{N-1} \log \frac{N-1}{1-p_1} && (10.15) \\ &= p_1 \log \frac{1}{p_1} + \frac{1}{N-1} \sum_{i>1} (1-p_1) \log \frac{1}{1-p_1} + (1-p_1)\log(N-1) && (10.16) \\ &= H(p_1) + (1-p_1)\log(N-1). && (10.17) \end{aligned}$$

Steps (10.15) and (10.16) use properties of the logarithm to split off the p_1 term. The first two terms of (10.16) form the binary entropy $H(p_1)$, and the last term is the one in Fano's inequality. This chain of equalities shows that equality in Fano's inequality is attainable.

11 Streaming Lower Bounds, Gaussian Width, and Compressed Sensing

Lecture 11, November 19, 2020, by David P. Woodruff. Compiled from the scribe notes of Jessica Lee (first half) and Charlie Hou (second half), and from the lecture slides. The section on compressed sensing is not in the scribe notes and is filled in from the lecture slides.

In this lecture we use the information theory of Lecture 10 to prove communication lower bounds, and through them lower bounds on the space of streaming algorithms. We then turn to sketching for projections onto general sets, measured by the Gaussian mean width, and end with compressed sensing. As in Lecture 10, log is base 2 and $H(\delta)$ denotes the binary entropy of a number $\delta \in [0,1]$. We will use that H is increasing on $[0,1/2]$ and that $H(\delta) < 1$ for $\delta < 1/2$.

11.1 The one-way communication complexity of Index

In the INDEX problem, Alice has a string $x \in \{0,1\}^n$ and Bob has an index $i \in [n]$. Alice sends a single message M to Bob, and Bob, given M and i, should output x_i with probability at least $2/3$. The probability is over the coin tosses of the protocol, not over the inputs: for every x and every i, $\Pr_{\text{coins}}[\text{Bob outputs } x_i] \geq 2/3$. More generally, $\text{CC}_\delta(f)$ denotes the randomized one-way communication complexity of a function f with error δ. This is the smallest ℓ such that some protocol in which Alice always sends an ℓ-bit message lets Bob output f correctly with probability at least $1-\delta$ on every input. We want to show that for some inputs and coin tosses, M must be $\Omega(n)$ bits long.

Theorem 11.1. $\text{CC}_{1/3}(\text{INDEX}) \geq n(1 - H(1/3)) = \Omega(n)$.

Proof. Consider a protocol with error at most $1/3$, and let X be uniformly distributed on $\{0,1\}^n$ (call this distribution μ). Alice sends the message M (a random variable that depends on X and on Alice's coins), and we can think of Bob's output as a guess X_i' for X_i. Since the protocol is correct with probability at least $2/3$ on every input, also $\Pr[X_i' = X_i] \geq 2/3$ for every i when $X \sim \mu$. Bob computes X_i' from M, i and his own coins, so $X_i \to M \to X_i'$ is a Markov chain, and Fano's inequality (Lecture 10) applies. With $P_e = \Pr[X_i' \neq X_i] \leq 1/3$ and $|\mathcal{X}| = 2$ (a bit), for every i,

$$H(X_i \mid M) \leq H(P_e) + P_e \log(2-1) = H(P_e) \leq H(1/3) < 1, \tag{11.1}$$

where $H(P_e) \leq H(1/3)$ because H is increasing on $[0,1/2]$. So the message reveals a lot of information about each bit. Now consider the mutual information $I(X;M)$. Recall that $X_{<i} = (X_1, \ldots, X_{i-1})$. Then

$$\begin{aligned}
I(X;M) &= \sum_{i=1}^n I(X_i; M \mid X_{<i}) && \text{(chain rule for mutual information)} \\
&= \sum_{i=1}^n \left(H(X_i \mid X_{<i}) - H(X_i \mid M, X_{<i})\right) && \text{(definition of mutual information)} \\
&= n - \sum_{i=1}^n H(X_i \mid M, X_{<i}) && (X_i \text{ is a uniform bit, independent of } X_{<i}) \\
&\geq n - \sum_{i=1}^n H(X_i \mid M) && \text{(conditioning cannot increase entropy)} \\
&\geq n - H(1/3)\, n && \text{(by (11.1))}.
\end{aligned} \tag{11.2}$$

How does this relate to the length of the message? Suppose every message has ℓ bits. Then M takes at most 2^ℓ values, so $H(M) \le \log 2^\ell = \ell$ by the bound $H(M) \le \log(\text{support size})$ of Lecture 10, whatever the distribution of M. Allowing different lengths changes the bound $H(M) \le \ell$ by at most one bit: if the messages have at most ℓ bits each, Alice can pad them to exactly $\ell + 1$ bits by appending a 1 and then 0s. Moreover $I(X; M) = H(M) - H(M \mid X) \le H(M)$. With (11.2),

$$\ell \ge H(M) \ge I(X; M) \ge n(1 - H(1/3)) = \Omega(n). \qquad \square$$

This formalizes that the length of Alice's message has to be at least $\Omega(n)$ bits.

11.2 Lower bounds for streaming algorithms by reduction

It is natural to prove a lower bound for a problem by reducing a problem with a known lower bound, such as INDEX, to it. For streaming algorithms there is a natural framework. Alice has an input $a \in \{0,1\}^n$ and creates a stream $s(a)$; Bob has an input $b \in \{0,1\}^n$ and creates a stream $s(b)$. The lower bound technique is:

1. Alice runs the streaming algorithm Alg on $s(a)$, and transmits the state (the memory contents) of $\text{Alg}(s(a))$ to Bob.
2. Bob continues the algorithm on his stream, and computes $\text{Alg}(s(a), s(b))$, the output on the concatenated stream.
3. If Bob can solve a problem $g(a, b)$ from this output, then the space complexity of Alg is at least the one-way communication complexity of g.

11.2.1 Example: distinct elements

Given a stream of items from $[n]$, how many *distinct* items are there? Note that this is $\|x\|_0$ for the vector x of item frequencies, which Lecture 10 approximates with $O(\varepsilon^{-2} \log^2 n)$ bits. We show that computing the answer exactly requires $\Omega(n)$ bits of space, by a reduction from INDEX: Alice has a bit string $x \in \{0,1\}^n$, Bob has an index $i \in [n]$, and Bob wants to know whether $x_i = 1$.

- Alice's stream $s(a)$ lists the indices $i_1, \ldots, i_r$, where an index i' appears (once) if and only if $x_{i'} = 1$.
- Bob's stream $s(b)$ is the single item i.
- If $\text{Alg}(s(a), s(b)) = \text{Alg}(s(a)) + 1$, then Bob outputs $x_i = 0$; otherwise he outputs $x_i = 1$.

Appending i to the stream creates one more distinct item exactly when i did not appear in $s(a)$, that is, when $x_i = 0$; if $x_i = 1$, then i already appeared in $s(a)$ and the number of distinct items does not change. So the space complexity of Alg is at least the one-way communication complexity of INDEX. If Alg is randomized, Bob should not rely on two of its answers. Instead, Alice appends the exact number $\sum_{i'} x_{i'}$ of distinct items in $s(a)$ to the state, using $O(\log n)$ more bits, and Bob compares $\text{Alg}(s(a), s(b))$ with it. Then the error probability is that of a single answer, and the space is $\Omega(n) - O(\log n) = \Omega(n)$.

11.2.2 Augmented Index

We now strengthen INDEX by giving Bob more information. In the AUGMENTED-INDEX problem, Alice has $x \in \{0,1\}^n$, Bob has $i \in [n]$ together with the bits $x_1, \ldots, x_{i-1}$, and Bob wants to learn x_i. Since Bob knows more, a lower bound for AUGMENTED-INDEX implies the same lower bound for INDEX. A similar proof still gives an $\Omega(n)$ lower bound.

Theorem 11.2. *For every $\delta < 1/2$, $\mathrm{CC}_\delta(\text{AUGMENTED-INDEX}) \geq n(1 - H(\delta))$.*

Proof. Let X be uniform on $\{0,1\}^n$. As in the proof of Theorem 11.1,

$$I(M;X) = \sum_{i=1}^{n} I(M; X_i \mid X_{<i}) = n - \sum_{i=1}^{n} H(X_i \mid M, X_{<i}).$$

This time Bob predicts X_i from M and $X_{<i}$, so we use the Markov chain $X_i \to (M, X_{<i}) \to X_i'$. Since Bob is correct with probability at least $1 - \delta$, Fano's inequality gives $H(X_i \mid M, X_{<i}) \leq H(\delta) + \delta \log(2-1) = H(\delta)$. Therefore

$$\mathrm{CC}_\delta(\text{AUGMENTED-INDEX}) \geq H(M) \geq I(M;X) \geq n - H(\delta)\, n = n(1 - H(\delta)). \qquad \square$$

11.3 An $\Omega(\log n)$ bit lower bound for estimating norms

A basic question in streaming is to maintain a counter c under a stream of updates $c \leftarrow c+1$, $c \leftarrow c-1$, $c \leftarrow c-1$, ... To do so exactly, we need $\log n$ bits. Can we approximate the counter within a factor of 2 with less? Writing down a 2-approximation takes only $O(\log \log n)$ bits (for example, the nearest power of 2), but we now show that the problem is harder than writing down the answer: $\Omega(\log n)$ bits are needed. The counter is the special case of a stream vector x with a single non-zero coordinate, so this is also a lower bound for approximating any norm $\|x\|_p$.

We reduce from AUGMENTED-INDEX on $\log n$ bits, with the order of the coordinates reversed: Bob knows the bits after his index instead of those before it, which does not change Theorem 11.2.

- Alice has $x \in \{0,1\}^{\log n}$. She creates the vector v with a single non-zero coordinate, equal to $\sum_{j=1}^{\log n} 10^j x_j$. She feeds v into the streaming algorithm, and sends the state of the algorithm to Bob.
- Bob has $i \in [\log n]$ and $x_{i+1}, x_{i+2}, \ldots, x_{\log n}$. He creates the vector w with the same single non-zero coordinate, equal to $\sum_{j>i} 10^j x_j$, and feeds $-w$ into the state of the algorithm.
- The stream vector is now $v - w$, whose only non-zero coordinate is $c = \sum_{j \leq i} 10^j x_j$, so every norm of $v - w$ equals c. If the output of the algorithm is at least $10^i/2$, Bob guesses $x_i = 1$, and otherwise he guesses $x_i = 0$.

If $x_i = 1$, then $c \geq 10^i$. If $x_i = 0$, then $c \leq 10 + 100 + \cdots + 10^{i-1} < 10^i/9$. So an output Z with $c/2 \leq Z \leq 2c$ satisfies $Z \geq 10^i/2$ in the first case and $Z < 2 \cdot 10^i/9 < 10^i/2$ in the second: there is a gap between the two cases even for a 2-approximation, and Bob learns x_i. By Theorem 11.2, the algorithm needs $\Omega(\log n)$ bits of space. The entries of the stream are at most $10^{\log n + 1} = \mathrm{poly}(n)$ in magnitude, as the streaming model allows.

11.4 An $\Omega(1/\varepsilon^2)$ bit lower bound for estimating norms

For $x, y \in \{0,1\}^n$, the *Hamming distance* $\Delta(x,y)$ is the number of coordinates i with $x_i \neq y_i$.

Definition 11.3 (Gap-Hamming)**.** In the Gap-Hamming problem, Alice has $x \in \{0,1\}^n$ and Bob has $y \in \{0,1\}^n$, with the promise that either $\Delta(x,y) > n/2 + 2\varepsilon n$ or $\Delta(x,y) < n/2 + \varepsilon n$. The goal is to decide which of the two cases occurs.

There is a lower bound of $\Omega(\varepsilon^{-2})$ bits for the randomized one-way communication complexity of Gap-Hamming [IW03, Woo04, JKS08], and the same holds for two-way communication [CR12]. It gives an $\Omega(\varepsilon^{-2})$ bit lower bound for approximating any norm. To see this, Alice feeds x into the streaming algorithm and Bob feeds $-y$. Since $x - y \in \{-1,0,1\}^n$, $\|x-y\|_p^p = \Delta(x,y)$ for every p. A $(1 \pm \varepsilon/3)$-approximation Z of $\Delta(x,y)$ is at least $(1-\varepsilon/3)(n/2+2\varepsilon n)$ in the first case and less than $(1+\varepsilon/3)(n/2+\varepsilon n)$ in the second. The first bound exceeds the second by $(2\varepsilon/3 - \varepsilon^2)n > 0$ for $\varepsilon < 2/3$, so Bob can tell the two cases apart.

We sketch the reduction from Index to Gap-Hamming of [JKS08], on instances of length $t = \Theta(\varepsilon^{-2})$. Alice has $x \in \{0,1\}^t$ and Bob has $i \in [t]$. They share a public coin: uniformly random vectors $r^1, \dots, r^t \in \{0,1\}^t$ (equivalently, a $t \times t$ matrix of random bits whose k-th row is r^k). Alice and Bob create $a, b \in \{0,1\}^t$ by

$$a_k = \text{Majority}\{r^k_j : x_j = 1\}, \qquad b_k = r^k_i, \qquad k \in [t].$$

In words, b is the i-th column of the public coin matrix, and a is obtained by taking only the columns j with $x_j = 1$ and the majority of the bits in each row.

- If $x_i = 0$, then a and b are independent, since the i-th column is not used for a. Each b_k is a uniform bit independent of a_k, so it differs from a_k with probability $1/2$, and $\mathbb{E}[\Delta(a,b)] = t/2$.
- If $x_i = 1$, then a and b are dependent, since b_k is one of the bits whose majority is a_k. Let w be the number of ones in x (say w is odd). For independent uniform bits $Z_1, \dots, Z_w$,

$$\Pr[\text{Majority}(Z_1,\dots,Z_w) = Z_1] = \frac{1}{2} + \frac{\binom{w-1}{(w-1)/2}}{2^w} = \frac{1}{2} + \Theta\Big(\frac{1}{\sqrt{w}}\Big) \geq \frac{1}{2} + \Theta\Big(\frac{1}{\sqrt{t}}\Big) = \frac{1}{2} + \Theta(\varepsilon), \tag{11.3}$$

  using $w \leq t$. The majority equals Z_1 if and only if at least $(w-1)/2$ of $Z_2, \dots, Z_w$ agree with Z_1. Let $\beta = \binom{w-1}{(w-1)/2} 2^{-(w-1)}$ be the probability that exactly $(w-1)/2$ of them agree. By symmetry, more than $(w-1)/2$ of them agree with probability $\frac{1}{2}(1-\beta)$, and adding β gives (11.3). By Stirling's formula, $\beta = \Theta(1/\sqrt{w})$. So $\mathbb{E}[\Delta(a,b)] = t/2 - \Theta(t/\sqrt{w}) \leq t/2 - \Omega(\varepsilon t) = t/2 - \Omega(1/\varepsilon)$.

Without casing on the value of x_i, and for x with $\Theta(t)$ ones, $\mathbb{E}[\Delta(a,b)] = t/2 - x_i \cdot \Theta(\sqrt{t})$, and since $\sqrt{t} = \Theta(\varepsilon t)$, the two cases differ by $\Theta(\varepsilon t)$, the gap in Gap-Hamming on instances of length t. So a protocol for Gap-Hamming lets Bob learn x_i, and the $\Omega(t)$ lower bound for Index gives an $\Omega(t) = \Omega(\varepsilon^{-2})$ lower bound for Gap-Hamming. Note that this argument compares only expectations. $\Delta(a,b)$ is a sum of t independent bits, so it fluctuates around its mean by $\Theta(\sqrt{t})$, the same order as the gap. So the instances need not satisfy the promise, and the full proof [JKS08] needs more care.

11.5 Index with public randomness, and on average

The reduction above used a public coin. Is Index still hard when Alice and Bob share randomness? Let Alice have $x \in \{0,1\}^n$ and Bob $i \in [n]$, and let Alice send a randomized message M to Bob,

where both know the shared randomness R. In the proof for the hardness of INDEX, we can just carry R along. With X uniform on $\{0,1\}^n$ and independent of R,

$$\begin{aligned} I(M; X \mid R) &= \sum_{i=1}^{n} I(M; X_i \mid X_{<i}, R) \\ &\geq \sum_{i=1}^{n} I(M; X_i \mid R) \\ &= n - \sum_{i=1}^{n} H(X_i \mid M, R). \end{aligned}$$

The inequality holds term by term, since X_i is independent of $(X_{<i}, R)$ and conditioning cannot increase entropy:

$$\begin{aligned} I(M; X_i \mid X_{<i}, R) &= H(X_i \mid X_{<i}, R) - H(X_i \mid M, X_{<i}, R) \\ &\geq H(X_i \mid R) - H(X_i \mid M, R) = I(M; X_i \mid R). \end{aligned}$$

The last line of the first display uses $H(X_i \mid R) = 1$. By Fano's inequality, applied with (M, R) in place of M, $H(X_i \mid M, R) \leq H(\delta)$ if Bob can guess X_i with probability at least $1 - \delta$ (for $\delta < 1/2$). So once again, using $|M| \geq H(M) \geq H(M \mid R) \geq I(M; X \mid R)$,

$$\mathrm{CC}_\delta(\textsc{Index}) \geq I(M; X \mid R) \geq n(1 - H(\delta)).$$

Remark 11.4 (Correct on average). The same lower bound, up to a constant factor, holds if the protocol is only correct on average over x and i drawn independently from the uniform distribution. Suppose that Bob errs with probability at most $1/10$, over the uniformly random x and i and the randomness of the protocol. For each i, let δ_i be the probability that Bob errs on index i, over x and the randomness. Then $\mathbb{E}_i[\delta_i] \leq 1/10$, so by Markov's inequality $\Pr_i[\delta_i \geq 2/10] \leq 1/2$: for at least half of the indices, $\delta_i \leq 1/5$. For those indices the proof above applies with $\delta = 1/5$, and for the other indices we use only $H(X_i \mid M, R) \leq 1$. So $n - nH(\delta)$ becomes $\frac{n}{2} - \frac{n}{2}H(1/5)$, which is still $\Omega(n)$. Thus INDEX is still hard in the distributional setting, where the inputs are drawn uniformly at random.

11.6 Distributional communication complexity

Let Alice have X and Bob have Y, with $(X, Y) \sim \mu$ for a distribution μ. The *μ-distributional complexity* $D_\mu(f)$ is the minimum communication cost of a protocol which outputs $f(X, Y)$ with probability at least $2/3$ for $(X, Y) \sim \mu$, where Alice and Bob are deterministic. (In one-way communication, Alice sends a single message $M(X)$ to Bob.) Let $R(f)$ be the randomized communication complexity of computing $f(X, Y)$, with error probability at most $1/3$ on every input.

Theorem 11.5 (Yao's minimax principle)**.** *$R(f) = \max_\mu D_\mu(f)$ (for protocols with public randomness).*

That is, the randomized communication complexity is lower bounded by every μ-distributional complexity, and the best such bound is tight. This gives a strategy for proving lower bounds: find a hard distribution μ on the inputs, and prove a lower bound on the communication of *deterministic* protocols that compute $f(X, Y)$ with probability $2/3$ under μ.

11.6.1 Indexing is universal for product distributions

The *communication matrix* A_f of a Boolean function $f\colon \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ has rows indexed by $\mathcal{X}$ and columns indexed by $\mathcal{Y}$, and its (x,y) entry is $f(x,y)$. The *VC dimension* of A_f is the dimension of the largest Boolean hypercube that "fits" into A_f: the largest k such that A_f has a $2^k \times k$ submatrix whose rows are all the 2^k vertices of $\{0,1\}^k$.

Theorem 11.6 (Kremer, Nisan, Ron)**.** *For one-way communication,* $\max_{\text{product distributions } \mu} D_\mu(f) = \Theta(\text{VC dimension of } A_f)$.

For example, the matrix

$$\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

has VC dimension 2. In the first two columns, rows 3, 2, 4 and 6 show the patterns 00, 10, 01 and 11. But no row has the pattern 111 (or 011, or 101), so the three columns together do not contain the 3-dimensional hypercube.

If A_f has VC dimension d, then f can be used to solve INDEX on d bits. Consider a $2^d \times d$ submatrix of A_f that contains the d-dimensional hypercube, and label its columns from 1 to d. Each row of the submatrix is a vector in $\{0,1\}^d$, and all vectors occur. Suppose Alice has $x \in \{0,1\}^d$: she picks a row of the submatrix equal to x, which corresponds to a row of A_f, that is, to an input x' of f. Suppose Bob has $i \in [d]$: he takes column i of the submatrix, which corresponds to a column of A_f, that is, to an input y' of f. Then $f(x', y') = x_i$, so a protocol for f also solves INDEX. The uniform distribution on INDEX inputs is a product distribution, so combined with Theorem 11.6, this implies that, when μ is restricted to product distributions, the best lower bound one can prove on $D_\mu(f)$ is the best lower bound one can prove by a reduction from INDEX.

11.7 Index with low error

INDEX with error probability 1/3 and with error probability 0 both need $\Omega(n)$ communication. But sometimes we want lower bounds in terms of the error probability. Indexing on large alphabets is a setting where the lower bound depends on the error probability δ:

- Alice has $x \in \{0,1\}^{n/\delta}$ with exactly n ones, and Bob has $i \in [n/\delta]$.
- Bob wants to decide whether $x_i = 1$, with error probability δ.

An upper bound: there are $\binom{n/\delta}{n}$ possible sets of positions of the ones, so Alice can communicate her set with $\log \binom{n/\delta}{n}$ bits, which by Stirling's formula is $O(n \log(1/\delta))$. We can also see this directly: $\binom{N}{n} \le (eN/n)^n$ gives $\log \binom{n/\delta}{n} \le n \log(e/\delta)$. [JW13] show that the one-way communication complexity is also $\Omega(n \log(1/\delta))$, so this is optimal.

With $n = 1$, Alice has a single position, and Bob has $i \in [1/\delta]$: this is the equality ("matching") problem, which asks whether Alice and Bob have the same index. From the large-alphabet bound, it needs $\Omega(\log(1/\delta))$ bits for success probability $1 - \delta$. This gives a lower bound for norm estimation in streaming. Suppose a streaming algorithm approximates $\|x\|_p$ for some p. Let $x = v - w$, where Alice feeds v into the algorithm and passes its memory contents to Bob, who continues with $-w$. If $v = e_j$ for Alice's index j and $w = e_i$ for Bob's index i, then $\|e_j - e_i\|_p = 0$ if and only if $i = j$

(otherwise it is $2^{1/p}$), so even a crude approximation of the norm solves the matching problem. So norm estimation needs $\Omega(\log(1/\delta))$ bits if success is desired with probability $1-\delta$.

Combining the lower bounds for norm estimation in this lecture, we get an $\Omega(\log n+\varepsilon^{-2}+\log(1/\delta))$ bit lower bound. In fact, there is an $\Omega(\varepsilon^{-2}\log(1/\delta)\log n)$ bit lower bound, which is optimal, since earlier lectures gave an algorithm with $O(\varepsilon^{-2}\log(1/\delta)\log n)$ bits of space: for example, the ℓ_1 estimator of Lecture 8 keeps $O(\varepsilon^{-2}\log(1/\delta))$ counters of $O(\log n)$ bits.

11.8 Projection onto complicated objects

Least squares regression finds the closest point y in a subspace K to a given point b. More generally, given a (possibly infinite) set of points $K \subseteq \mathbb{R}^n$ and a point b, we want to compute $\min_{y\in K}\|y-b\|_2$ (all norms in this section are Euclidean). Let S be a sketching matrix. We want that if $\hat{y} = \arg\min_{y\in K}\|Sy-Sb\|_2$, then

$$\|\hat{y}-b\|_2 \le (1+\varepsilon)\min_{y\in K}\|y-b\|_2.$$

More generally, we want to preserve the distances between all vectors in K:

$$\|S(y-y')\|_2 = (1\pm\varepsilon)\|y-y'\|_2 \qquad \text{for all } y,y'\in K. \tag{11.4}$$

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

11.8.1 Some case studies

What is the number of rows of S needed for (11.4) if K is

- a set of t arbitrary points in $\mathbb{R}^n$? By the JL lemma (Lecture 1), $O(\varepsilon^{-2}\log t)$ rows suffice.
- a set of t arbitrary points on a line in $\mathbb{R}^n$? Then $O(\varepsilon^{-2})$ rows suffice.

This suggests that the right measure of the "difficulty" of a set is some kind of volume. We formalize this next.

11.8.2 Mean widths

Let K be a bounded subset of $\mathbb{R}^n$. For a unit vector u, the *width* of K in direction u is $\sup_{p,q\in K}\langle u, p-q\rangle$ (Figure 9), and the *spherical mean width* of K is

$$\bar{w}(K) = \mathop{\mathbb{E}}_{u}\Big[\sup_{p,q\in K}\langle u, p-q\rangle\Big],$$

where u is a uniformly random unit vector. This is larger the more width the set has over all directions, which is what we want.

There is another measure of the difficulty of a set, the Gaussian mean width. Let $g\sim\mathcal{N}(0,I_n)$ be a vector of i.i.d. standard Gaussians. The *Gaussian mean width* of K is

$$w(K) = \mathop{\mathbb{E}}_{g}\Big[\sup_{p,q\in K}\langle g, p-q\rangle\Big].$$

We can write $g=\chi u$, where $u=g/\|g\|_2$ is a uniformly random unit vector and $\chi=\|g\|_2$ is a random variable that concentrates very close to $\sqrt{n}$. So the Gaussian mean width is essentially $\sqrt{n}$ times the spherical mean width, $w(K)=\Theta(\sqrt{n})\cdot\bar{w}(K)$. In fact, u and χ are independent, so $w(K)=\mathbb{E}[\chi]\cdot\bar{w}(K)$ exactly, and $\mathbb{E}[\chi]=\Theta(\sqrt{n})$.

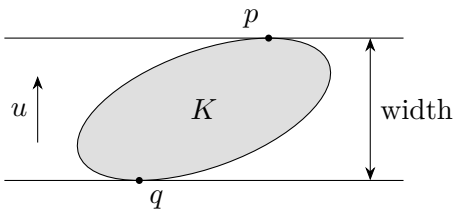


Figure 9: The width of a set K in direction u: the distance between the two supporting hyperplanes orthogonal to u, which touch K at p and q, equals $\langle u, p-q\rangle$.

Example 11.7.

- $K = \mathbb{S}^{n-1}$, the unit sphere. For every direction u, the width is 2 (take $p = u$ and $q = -u$), so $\bar{w}(K) = 2$ and $w(K) = 2\,\mathbb{E}[\chi] = \Theta(\sqrt{n})$.
- K is the set of unit vectors in a d-dimensional subspace of $\mathbb{R}^n$. Then $w(K) = \Theta(\sqrt{d})$. To see this, write $K = \{Uz : z \in \mathbb{S}^{d-1}\}$ for a matrix $U \in \mathbb{R}^{n\times d}$ with orthonormal columns. Then

$$w(K) = \mathop{\mathbb{E}}_{g}\Big[\sup_{z,z'\in\mathbb{S}^{d-1}} \langle g, Uz - Uz'\rangle\Big] = \mathop{\mathbb{E}}_{h}\Big[\sup_{z,z'\in\mathbb{S}^{d-1}} \langle h, z - z'\rangle\Big],$$

  where $h = U^\top g \sim \mathcal{N}(0, I_d)$ is still a standard Gaussian vector, so $w(K) = 2\,\mathbb{E}[\|h\|_2] = \Theta(\sqrt{d})$.
- K is a set of t arbitrary unit vectors in $\mathbb{R}^n$. Then $w(K) = O(\sqrt{\log t})$, because the expected maximum of t standard Gaussians is $O(\sqrt{\log t})$, which we show next.

Lemma 11.8. *Let $u^1, \dots, u^t$ be arbitrary unit vectors in $\mathbb{R}^n$, $t \geq 2$, let $g \in \mathbb{R}^n$ have i.i.d. $\mathcal{N}(0,1)$ entries, and let $Z_j = \langle u^j, g\rangle$, which are $\mathcal{N}(0,1)$ random variables (not necessarily independent). Then*

$$\mathop{\mathbb{E}}_{g}\Big[\max_j Z_j\Big] \leq 2\sqrt{\ln t},$$

and the Gaussian mean width of $K = \{u^1, \dots, u^t\}$ is $w(K) = \mathbb{E}[\max_j Z_j - \min_j Z_j] = 2\,\mathbb{E}[\max_j Z_j] = O(\sqrt{\log t})$.

Proof. For a standard Gaussian random variable W, $\mathbb{E}[e^{\lambda W}] = e^{\lambda^2/2}$. So for any $\lambda > 0$,

$$\mathbb{E}\left[e^{\lambda \max_j Z_j}\right] \leq \sum_{j=1}^{t} \mathbb{E}\left[e^{\lambda Z_j}\right] \leq t e^{\lambda^2/2}.$$

By Jensen's inequality (the logarithm is concave),

$$\mathbb{E}\Big[\max_j Z_j\Big] = \frac{1}{\lambda}\,\mathbb{E}\left[\ln e^{\lambda \max_j Z_j}\right] \leq \frac{1}{\lambda}\ln \mathbb{E}\left[e^{\lambda \max_j Z_j}\right] \leq \frac{\ln t}{\lambda} + \frac{\lambda}{2}.$$

Setting $\lambda = \sqrt{\ln t}$ gives $\mathbb{E}[\max_j Z_j] \leq \frac{3}{2}\sqrt{\ln t} \leq 2\sqrt{\ln t}$. For the width, $\sup_{p,q\in K}\langle g, p-q\rangle = \max_j Z_j - \min_j Z_j$, and $-g$ has the same distribution as g, so $\mathbb{E}[\min_j Z_j] = -\,\mathbb{E}[\max_j Z_j]$. □

This bound is tight. If we choose $u^1, \dots, u^t$ to be standard basis vectors ($t \leq n$), then $Z_1, \dots, Z_t$ are t independent standard Gaussians. Each of them is at least some threshold $s = \Theta(\sqrt{\log t})$ with probability $\Theta(1/t)$, so at least one of them is that large with probability $1 - (1 - \Theta(1/t))^t$, which is at least a positive constant. Since $\max_j Z_j \geq Z_1$, this gives $\mathbb{E}[\max_j Z_j] \geq s \cdot \Pr[\max_j Z_j \geq s] - \mathbb{E}[|Z_1|] = \Omega(\sqrt{\log t})$.

11.8.3 Sketching bounds

Theorem 11.9 (Gordon)**.** *Let K be a subset of $\mathbb{S}^{n-1}$. A random Gaussian matrix $S \in \mathbb{R}^{m\times n}$ with i.i.d. $\mathcal{N}(0, 1/m)$ entries and $m = O(w(K)^2/\varepsilon^2)$ rows satisfies, with constant probability,*

$$\|S(y - y')\|_2^2 = (1 \pm \varepsilon)\|y - y'\|_2^2 \qquad \text{for all } y, y' \in K.$$

This recovers the earlier results. For the unit vectors of a d-dimensional subspace, $w(K) = \Theta(\sqrt{d})$, so $O(d/\varepsilon^2)$ rows give a subspace embedding (Lecture 1). For t arbitrary points, $w(K) = O(\sqrt{\log t})$, and we recover the JL lemma.

Remark 11.10. Note that the statement above is informal. In its precise form, Gordon's theorem bounds $\sup_{z\in T}\big|\|Sz\|_2 - 1\big|$ over a set $T \subseteq \mathbb{S}^{n-1}$ in terms of the width of T. To preserve the distances between points of K, we take for T the set of normalized differences $(y - y')/\|y - y'\|_2$, $y \neq y' \in K$, and it is the width of this set that matters. The width of K itself can be misleading: two very close points of the sphere form a set of tiny width, but preserving their distance within $1 \pm \varepsilon$ needs $\Omega(1/\varepsilon^2)$ Gaussian rows. In the examples above, the two widths agree up to constants. The normalized differences of points of a subspace are again unit vectors of the subspace. Those of t points are at most t^2 unit vectors, of width $O(\sqrt{\log t})$. Those of points on a line are just $\pm u$ for one unit vector u, of width $O(1)$, which matches the $O(\varepsilon^{-2})$ rows of the second case study.

What about sparse sketching matrices S? Bourgain, Dirksen and Nelson [BDN15] show that S can have $m = w(K)^2 \operatorname{poly}(\log n)/\varepsilon^2$ rows and $s = \operatorname{poly}(\log n)/\varepsilon^2$ non-zero entries per column, if m and s satisfy a condition related to higher moments of $\sup_{p,q\in K}\langle g, p - q\rangle$. This has been applied to finite and infinite unions of subspaces.

11.9 Compressed sensing

In compressed sensing we take random "linear measurements" of an n-dimensional vector x. In our language, we choose a random $m \times n$ sketching matrix S and observe Sx. The goal is to output a vector x' with

$$\|x - x'\|_p \leq D \cdot \min_{k\text{-sparse } z} \|x - z\|_q, \tag{11.5}$$

where D is the *distortion*. We call this the ℓ_p/ℓ_q *guarantee*. Let $x_{[k]}$ be the best k-sparse approximation of x, which keeps the k largest coordinates of x in magnitude and sets the others to 0. The minimum in (11.5) is attained at $z = x_{[k]}$, so it equals $\|x - x_{[k]}\|_q = \|x_{-k}\|_q$, where $x_{-k} = x - x_{[k]}$ is x with its k largest coordinates set to 0 (as with $x_{-B/4}$ in Lecture 9).

We distinguish randomized ("for-each") schemes from deterministic ("for-all") schemes. In a for-each scheme, S is random, and for each fixed x the guarantee holds with high probability over S. In a for-all scheme, one fixed S works for all x simultaneously. CountSketch is a randomized scheme that achieves the ℓ_2/ℓ_2 guarantee with high probability:

$$\|x - x'\|_2 \leq O(1) \cdot \|x - x_{[k]}\|_2.$$

11.9.1 CountSketch for compressed sensing

Recall from Lecture 9 that CountSketch hashes the coordinates into B buckets, with $O(\log n)$ independent repetitions. Here $B = O(k)$, so S is a random linear map with $O(k \log n)$ rows. By the tail guarantee of Lecture 9, with a large enough constant in $B = O(k)$, with high probability the estimates $\hat{x}_i$ of all coordinates satisfy

$$|\hat{x}_i - x_i| \leq \eta = \frac{\|x - x_{[k]}\|_2}{\sqrt{k}} \qquad \text{for every } i \in [n]. \tag{11.6}$$

This is because the tail guarantee bounds the error by $O(\|x_{-B/4}\|_2/\sqrt{B})$, and $\|x_{-B/4}\|_2 \le \|x_{-k}\|_2$ once $B \ge 4k$. The output x' is the $2k$-sparse vector consisting of the top $2k$ estimates: let J be the set of the $2k$ indices with the largest $|\hat{x}_i|$, and let $x'_i = \hat{x}_i$ for $i \in J$ and $x'_i = 0$ otherwise.

Theorem 11.11. *If (11.6) holds, then* $\|x - x'\|_2 = O(\|x - x_{[k]}\|_2)$.

Proof. Say that coordinate i is *heavy* if $|x_i| \ge \eta$. How many heavy coordinates can there be? At most k of them are among the k largest coordinates, and every other heavy coordinate contributes at least η^2 to $\|x - x_{[k]}\|_2^2 = k\eta^2$, so there are at most k of those. Hence there are at most $2k$ heavy coordinates.

Say that coordinate i is *super-heavy* if $|x_i| \ge 3\eta$, and let T be the set of super-heavy coordinates. We claim that $T \subseteq J$, so T is in the support of x'. By (11.6), a super-heavy i has $|\hat{x}_i| \ge 2\eta$, while a coordinate j that is not heavy has $|\hat{x}_j| < 2\eta$. If some super-heavy i were not in J, then every $j \in J$ would have $|\hat{x}_j| \ge |\hat{x}_i| \ge 2\eta$, so all $2k$ coordinates of J, together with i, would be heavy, which contradicts the bound of $2k$ heavy coordinates.

Now we split the error into the parts on T and off T, writing $[n] \setminus T$ for the complement of T:

$$\|x-x'\|_2 \le \|(x-x')_T\|_2 + \|(x-x')_{[n]\setminus T}\|_2 \le \sqrt{2k}\cdot\frac{\|x - x_{[k]}\|_2}{\sqrt{k}} + \|(x-x_{[k]})_{[n]\setminus T}\|_2 + \|(x_{[k]}-x')_{[n]\setminus T}\|_2.$$

For the first term, $T \subseteq J$, so $|x_i - x'_i| = |x_i - \hat{x}_i| \le \eta$ for $i \in T$, and $|T| \le 2k$ since super-heavy coordinates are heavy. The second term is at most $\|x - x_{[k]}\|_2$. For the third term, every coordinate outside T has $|x_i| < 3\eta$. So $x_{[k]}$ restricted to $[n] \setminus T$ has at most k non-zero entries, each less than 3η in magnitude, and x' restricted to $[n] \setminus T$ has at most $2k$ non-zero entries, each at most $|x_i| + \eta < 4\eta$ in magnitude. Therefore $\|(x_{[k]} - x')_{[n]\setminus T}\|_2 \le 3\eta\sqrt{k} + 4\eta\sqrt{2k} = (3 + 4\sqrt{2})\|x - x_{[k]}\|_2$. Adding the three terms, $\|x - x'\|_2 \le (4 + 5\sqrt{2})\|x - x_{[k]}\|_2 = O(\|x - x_{[k]}\|_2)$. □

11.9.2 No deterministic scheme achieves ℓ_2/ℓ_2

Recall that the ℓ_2/ℓ_2 guarantee asks for $\|x - x'\|_2 = O(1) \cdot \|x - x_{[k]}\|_2$. This cannot be achieved by a deterministic sketch with few rows.

Theorem 11.12. *Let $k = 1$ and let $C > 0$ be any constant. No deterministic sketching matrix S with $m = o(n)$ rows, together with any recovery procedure that sees only Sx, outputs for every x a vector x' with $\|x - x'\|_2 \le C\|x - x_{[1]}\|_2$.*

Proof. It suffices to show that there is a vector x in the kernel of S with $\|x\|_\infty > C\|x - x_{[1]}\|_2$. Why? Since $Sx = S0$, the procedure outputs the same x' on x as on 0. On input 0 the guarantee forces $x' = 0$. So on input x the error is $\|x\|_2 \ge \|x\|_\infty > C\|x - x_{[1]}\|_2$.

We can assume that S has orthonormal rows, since replacing S by a matrix with orthonormal rows that span the row space of S does not change which information Sx carries about x, and does not increase the number of rows. Then $\sum_{i=1}^n \|Se_i\|_2^2 = \|S\|_F^2 = m$, so there is an i with $\|Se_i\|_2^2 \le m/n$. Let $x = e_i - S^\top Se_i$. Since $SS^\top = I_m$, $Sx = Se_i - Se_i = 0$, so x is in the kernel of S. But

$$\|x\|_\infty^2 \ge x_i^2 = \left(e_i^\top e_i - e_i^\top S^\top Se_i\right)^2 = \left(1 - \|Se_i\|_2^2\right)^2 \ge \left(1 - \frac{m}{n}\right)^2,$$

while, since e_i is 1-sparse and $S^\top$ has orthonormal columns,

$$\|x - x_{[1]}\|_2 \le \|x - e_i\|_2 = \|S^\top Se_i\|_2 = \|Se_i\|_2 \le \sqrt{\frac{m}{n}} = o(1).$$

So $\|x\|_\infty/\|x - x_{[1]}\|_2 \ge (1 - m/n)/\sqrt{m/n} \to \infty$, and exceeds C once n is large. □

11.9.3 Deterministic schemes achieve ℓ_2/ℓ_1

The ℓ_2/ℓ_1 guarantee asks for $\|x-x'\|_2 = O(1/\sqrt{k})\cdot\|x-x_{[k]}\|_1$, and it can be achieved deterministically, through the following property.

Definition 11.13 (Restricted isometry property)**.** A matrix S has the (ε, k)-*restricted isometry property (RIP)* if for all k-sparse vectors x,

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

What are some matrices S with $O(k\log(n/k))$ rows that have the (ε, k)-RIP for a constant ε? A random Gaussian matrix with that many rows has it with high probability. Such matrices are deterministic, but not explicit: a matrix with the RIP exists, and once it is fixed the scheme works for all x, but no efficient deterministic construction with this many rows is known. A major open question is to construct an explicit matrix with the (ε, k)-RIP and $o(k^2)$ rows. Bourgain et al. get $k^{2-\gamma}$ rows for a constant $\gamma > 0$ and $k \approx n^{1/2}$.

Why is k^2 a natural barrier? An (ε/k)-incoherent matrix (Lecture 9) has the (ε, k)-RIP: for a k-sparse x with support Q,

$$\|Sx\|_2^2 = \sum_{i,j\in Q} x_i x_j \langle S_{*,i}, S_{*,j}\rangle = \|x\|_2^2 \pm \frac{\varepsilon}{k}\|x\|_1^2,$$

and $\|x\|_1^2 \le k\|x\|_2^2$. The polynomial construction of Lecture 9 is explicit and gives such a matrix with $O(k^2\varepsilon^{-2}\log^2 n)$ rows.

Theorem 11.14. *If S has the $(\varepsilon, 2k)$-RIP for a small enough constant ε, then one can efficiently output an x' with $\|x-x'\|_2 = O(1/\sqrt{k})\cdot\|x-x_{[k]}\|_1$. In fact, one can just solve the linear program*

$$\min_{z\in\mathbb{R}^n}\|z\|_1 \quad \textit{subject to} \quad Sz = Sx :$$

if x' is its solution, then $\|x-x'\|_2 \le O(1/\sqrt{k})\cdot\|x-x_{[k]}\|_1$.

Minimizing $\|z\|_1$ is a linear program in the variables $z, u \in \mathbb{R}^n$: minimize $\sum_i u_i$ subject to $-u \le z \le u$ and $Sz = Sx$. The proof of Theorem 11.14 uses the restricted isometry property and elementary norm manipulations, and we omit it.

References

[BDN15] Jean Bourgain, Sjoerd Dirksen, and Jelani Nelson. Toward a unified theory of sparse dimensionality reduction in Euclidean space. *Geometric and Functional Analysis*, 25(4):1009–1088, 2015.

[BKLW14] Maria-Florina Balcan, Vandana Kanchanapally, Yingyu Liang, and David P. Woodruff. Improved distributed principal component analysis. In *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, 2014.

[BLS+16] Maria-Florina Balcan, Yingyu Liang, Le Song, David P. Woodruff, and Bo Xie. Communication efficient distributed kernel principal component analysis. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2016)*, pages 725–734, 2016.

[BWZ16] Christos Boutsidis, David P. Woodruff, and Peilin Zhong. Optimal principal component analysis in distributed and streaming models. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing (STOC 2016)*, pages 236–249, 2016.

[CEM+14] Michael B. Cohen, Sam Elder, Cameron Musco, Christopher Musco, and Madalina Persu. Dimensionality reduction for k-means clustering and low rank approximation. *CoRR*, abs/1410.6801, 2014.

[CR12] Amit Chakrabarti and Oded Regev. An optimal lower bound on the communication complexity of Gap-Hamming-Distance. *SIAM Journal on Computing*, 41(5):1299–1317, 2012.

[CW17] Kenneth L. Clarkson and David P. Woodruff. Low-rank approximation and regression in input sparsity time. *Journal of the ACM*, 63(6):54:1–54:45, 2017.

[FSS20] Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: Constant-size coresets for k-means, PCA, and projective clustering. *SIAM Journal on Computing*, 49(3):601–657, 2020.

[IW03] Piotr Indyk and David P. Woodruff. Tight lower bounds for the distinct elements problem. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2003)*, pages 283–288, 2003.

[JKS08] Thathachar S. Jayram, Ravi Kumar, and D. Sivakumar. The one-way communication complexity of Hamming distance. *Theory of Computing*, 4(1):129–135, 2008.

[JW13] Thathachar S. Jayram and David P. Woodruff. Optimal bounds for Johnson–Lindenstrauss transforms and streaming problems with subconstant error. *ACM Transactions on Algorithms*, 9(3):26:1–26:17, 2013.

[KN14] Daniel M. Kane and Jelani Nelson. Sparser Johnson–Lindenstrauss transforms. *Journal of the ACM*, 61(1):4:1–4:23, 2014.

[KNW10] Daniel M. Kane, Jelani Nelson, and David P. Woodruff. On the exact space complexity of sketching and streaming small norms. In *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2010)*, pages 1161–1178, 2010.

[KVW14] Ravi Kannan, Santosh S. Vempala, and David P. Woodruff. Principal component analysis and higher correlations for distributed data. In *Proceedings of the 27th Conference on Learning Theory (COLT 2014)*, volume 35 of *Proceedings of Machine Learning Research*, pages 1040–1057, 2014.

[MM13] Xiangrui Meng and Michael W. Mahoney. Low-distortion subspace embeddings in input-sparsity time and applications to robust linear regression. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC 2013)*, pages 91–100, 2013.

[Ngu13] Huy L. Nguyễn. Personal communication, 2013.

[NN13] Jelani Nelson and Huy L. Nguyễn. OSNAP: Faster numerical linear algebra algorithms via sparser subspace embeddings. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2013)*, pages 117–126, 2013.

[Sar06] Tamás Sarlós. Improved approximation algorithms for large matrices via random projections. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2006)*, pages 143–152, 2006.

[Woo04] David P. Woodruff. Optimal space lower bounds for all frequency moments. In *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2004)*, pages 167–175, 2004.

[WW19] Ruosong Wang and David P. Woodruff. Tight bounds for ℓ_p oblivious subspace embeddings. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2019)*, pages 1825–1843, 2019.

[WZ13] David P. Woodruff and Qin Zhang. Subspace embeddings and ℓ_p-regression using exponential random variables. In *Proceedings of the 26th Annual Conference on Learning Theory (COLT 2013)*, volume 30 of *Proceedings of Machine Learning Research*, pages 546–567, 2013.

[WZ16] David P. Woodruff and Peilin Zhong. Distributed low rank approximation of implicit functions of a matrix. In *Proceedings of the 32nd IEEE International Conference on Data Engineering (ICDE 2016)*, pages 847–858, 2016.