CS 15-859: Algorithms for Big Data Fall 2019

Lecture Lecture 11 — November 14

Prof. David Woodruff *Scribe: Alex Zheng*

1 Nonnegative Matric Factorization

The **Nonnegative Matrix Factorization Problem** is as follows:

Input: $A \in \mathbb{R}^{n \times n}$, $k \in \mathbb{N}$ where $k \leq n$.

Output: Are there two matrices $U, V^T \in \mathbb{R}^{n \times k}$ such that

$$UV = A, U \geq 0, V \geq 0.$$

Example: Let $A = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 \\ 2 & 4 & 2 & 2 \\ 3 & 5 & 2 & 2 \end{bmatrix}$, and $k = 0$. Then the output would be YES because

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 1 \\ 2 & 2 \\ 3 & 2 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}.$$

This question is equivalent of computing the nonnegative rank of A, $rank_+(A)$. It is a fundamental question in machine learning and has a bunch of applications in fields such as text mining, spectral data analysis, scalable internet distance prediction, non-stationary speech denoising, bioinformatics, nuclear imaging, etc. Note that $rank(A) \leq rank_+(A)$, which is trivial. The question of whether equality holds is NP-hard, and the hardness was proven by **Vavasis'09**.

Example of Inequality:

$$A = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}.$$

In this case,

$$3 = rank(A) < rank_+(A) = 4.$$

1.1 Results

To solve NMF, we first introduce the **Polynomial System Verifier Problem** which is as follows:

Input: Given $P(x)$ a real polynomial system with v variables along with:
- $x = (x_1, x_2, ..., x_v)$

- m polynomial contraints $f_i(x) \geq 0$ where $i \in [m]$
- the maximum degree of all polynomial constraints d
- the bitsize of the coefficients of the polynomials H.

Output: Whether or not there exists a solution to the polynomial system P. We can decide this in $(md)^{O(v)}poly(H)$ time.

We can solve Nonnegative Matrix Factorization by writing

$$\min_{U,V^T \in \mathbb{R}^{n \times k}, U,V \geq 0} \|UV - A\|_F^2$$

as a polynomial system that has $poly(k)$ variables and $poly(n)$ constraints and degree and use a polynomial system verifier to solve it. Currently we have the following algorithms:

- A $n^{2^{O(k)}}$ time algorithm by **Arora-Ge-Kannan-Moitra'12**. [1]
- A $2^{O(k^3)}n^{O(k^2)}$ time algorithm by **Moitra'13**. [2]

1.2 Hardness

For hardness results, we first introduce the **k-SUM Problem** which is as follows:

Input: A set of n values $\{s_1, ..., s_n\}$ each in range $[0, 1]$.

Output: Whether there is a set of k numbers that sum to exactly $k/2$.

It turns out that, assuming that 3-SAT on n variables cannot be solved in $2^{o(n)}$ time, k-SUM can't be solved in $n^{o(k)}$ time. Using this result, we can deduct that Nonnegative Matrix Factorization takes $n^{\Omega(k)}$ time. **(Patrascu-Williams'10)**

2 Weighted Low Rank Approximation

The **Weighted Low Rank Approximation Problem** is as follows:

Input: A matrix $A \in \mathbb{R}^{n \times n}$, a weight matrix $W \in \mathbb{R}^{n \times n}$, $k \in \mathbb{N}$, and $\epsilon > 0$.

Output: A rank-k matrix $\hat{A}$ such that

$$\|W \circ (\hat{A} - A)\|_F^2 \leq \min_{A' : \text{rank}-k} \|W \circ (\hat{A} - A)\|_F^2.$$

We will define

$$OPT = \min_{A' : \text{rank}-k} \|W \circ (A' - A)\|_F^2.$$

Since $\hat{A}$ is rank k, the problem is equivalent to outputting $U, V \in \mathbb{R}^{n \times k}$ such that

$$\|W \circ (UV^T - A)\|_F^2 \leq (1 + \epsilon) \cdot OPT.$$

2.1 Motivation

Suppose some website (with a bunch of users) has a bunch of movies split into several categories and each user can be rated by users. Each user has a distribution he uses to assign ratings to, so it

seems natural to scale each users score according to that users distribution's standard deviation. We can use a weight matrix to do this.

Another motivation comes from a related problem **Matrix Completion**. The problem is as follows:

Input: $A \in \{\mathbb{R}, ?\}^{n \times n}$, a matrix with some entries missing and the missing entries are denoted using "?", $\Omega \subset [n] \times [n]$ which are the observed entries, and $k \in \mathbb{R}$.

Output: We want to fill in the missing entries of A such that $rank(A) = k$.

We can solve this problem by assigning weight 0 to the missing entries and solve WLRA with the input k.

2.2 Results (with assumptions)

We will make some assumptions on the weight matrix W to simplify the problem:

- If W **has** r **distinct rows and columns**, then we can output $\hat{A} \in \mathbb{R}^{n \times n}$ that satisfies

$$\|W \circ (\hat{A} - A)\|_F^2 \leq (1 + \epsilon) \cdot OPT$$

with probability at least $\frac{9}{10}$ in $O((nnz(A) + nnz(W)) \cdot n^\gamma) + n \cdot 2^{\tilde{O}(k^2 r/\epsilon)}$ time, for an arbitrarily small constant $\gamma > 0$.

- If W **has** r **distinct columns**, then we can output $\hat{A} \in \mathbb{R}^{n \times n}$ that satisfies

$$\|W \circ (\hat{A} - A)\|_F^2 \leq (1 + \epsilon) \cdot OPT$$

with probability at least $\frac{9}{10}$ in $O((nnz(A) + nnz(W)) \cdot n^\gamma) + n \cdot 2^{\tilde{O}(k^2 r^2/\epsilon)}$ time, for an arbitrarily small constant $\gamma > 0$.

- If W **has rank** r, then we can output $\hat{A} \in \mathbb{R}^{n \times n}$ that satisfies

$$\|W \circ (\hat{A} - A)\|_F^2 \leq (1 + \epsilon) \cdot OPT$$

with probability at least $\frac{9}{10}$ in $n^{O(k^2 r^2/\epsilon)}$ time, for an arbitrarily small constant $\gamma > 0$.

The above results are from [3].

2.3 Hardness

We will assume the **Random 4-SAT Hypothesis** which is as follows: We are given $\mathcal{S}$, a 4-SAT formula with n variables and m clauses with 4 literals in each clause. Suppose that each of the $\Theta(n^4)$ clauses is picked independently with probability $\Theta(1/n^3)$, then any algorithm that:

- Outputs 1 with probability 1 when $\mathcal{S}$ is satisfiable.
- Outputs 0 with probability $\geq 1/2$.

requires $2^{\Omega(n)}$ time.

We will also use the **Maximum Edge Biclique** problem which is as follows:

Input: We are given $G = (V, U, E)$ a bipartite graph with $|U| = |V| = n$.

Output: A k_1-by-k_2 complete bipartite subgraph of G such that $k_1 \cdot k_2$ is maximized.

Assuming that the Random 4-SAT Hypothesis is true, then there exists $\epsilon_1 > \epsilon_2 > 0$ such that any algorithm that distinguishes between bipartite graphs $G = (V, U, E)$ with $|V| = |U| = n$ in the following cases:

- there is a bipartite clique of size $\geq \left(\frac{n}{16}\right)^2 (1 + \epsilon_1)$
- all bipartite cliques have size $\leq \left(\frac{n}{16}\right)^2 (1 + \epsilon_2)$

requires $2^{\Omega(n)}$ time.

2.3.1 Reduction

We will reduce Maximum Edge Biclique to Weighted Low Rank Approximation. Suppose we are given an instance of MEB, so namely a bipartite graph $G = (V, U, E)$ with $|V| = |U| = n$ and we wish to output a k_1-by-k_2 complete bipartite subgraph of G such that $k_1 \cdot k_2$ is maximized. Note that it suffices to solve the complement problem - output a k_1-by-k_2 complete bipartite subgraph such that $|E| - k_1 \cdot k_2$ is minimized. We will transform this new problem to a WLRA problem.

Define matrix A such that $A_{i,j}$ is equal to 1 if the edge (V_i, U_j) exists otherwise is equal to 0. Define matrix W such that $W_{i,j}$ is equal to 1 if the edge (V_i, U_j) otherwise is equal to n^6. Then, a $(1 + \epsilon)$ approximation to OPT in WLRA, even with $r = 1$ and with an arbitrary W, will translate into a $(1 + \epsilon)$ approximation to OPT in MEB complement. However, assuming the Random 4-SAT Hypothesis, we know that the MEB problem takes $2^{\Omega(n)}$ time, which shows a lower bound on the time complexity of WLRA.

We conclude that given input matrix $A \in \mathbb{R}^{n \times n}$, weight matrix $W \in \mathbb{R}^{n \times n}$ with r distinct columns, $k \in \mathbb{N}$, $W_{i,j} \in \{0, 1, 2, ..., poly(n)\}$, there exists ϵ_0 such that any algorithm that outputs a rank $k \geq 1$ matrix $\hat{A} \in \mathbb{R}^{n \times n}$ such that

$$\|W \circ (UV^T - A)\|_F^2 \leq (1 + \epsilon) \cdot OPT$$

with $\epsilon_0 > \epsilon > 0$ probability at least $\frac{9}{10}$ takes $2^{\Omega(r)}$ time.

2.4 Algorithmic Techniques

To improve the time complexity, we first introduce the following tools:

2.4.1 Tools

Polynomal System Verifier: Given $P(x)$ a real polynomial system with v variables, $x = (x_1, x_2, ..., x_v)$, m polynomial contraints $f_i(x) \geq 0$ where $i \in [m]$, the maximum degree of all polynomial constraints d, and the bitsize of the coefficients of the polynomials H, then it takes $(md)^{O(v)} poly(H)$ time to decide if there exists a solution to P.

Lower Bound on the Cost: This Let $T = \{x \in \mathbb{R}^v \mid f_1(x) \geq 0, ..., f_m \geq 0\}$. Suppose we have a non-negative polynomial $G(x)$ and we evaluate $G(x)$ for all $x \in T$. Suppose we are given $\min_{x \in T} G(x) > 0$. How small can $\min_{x \in T} G(x)$ be? It turns out it can get arbitrarily small. For

example, consider $x_1, x_2 \in \mathbb{R}$, no f_i constraints, and let $G(x_1, x_2) = (x_1x_2 - 1)^2 + x_2^2$. $G(x_1, x_2) > 0$ since $G(x_1, x_2) = 0$ implies $x_2 = 0$ which implies $(x_1x_2 - 1)^2 = 1$, which is a contradiction. However, we can make $G(x_1, x_2)$ arbitrarily small by setting $x_1 \to \infty$ and $x_2 = \frac{1}{x_1} \to 0$.

To fix this, we intersect T with a ball, $B = \{x : \|x\|_\infty \leq 2^H\}$. This prevents x_1 from increasing towards infinity and would thus set a lower bound on $G(x_1, x_2)$. Then the minimum value that nonnegative G takes over $T \cap B$ is either 0 or $\geq (2^H + m)^{-d^v}$. This will be our lower bound on the cost of the polynomial system. With it, we can perform binary search over the cost.

Multiple Regression Sketch: Given matrices $A^{(1)}, ..., A^{(m)} \in \mathbb{R}^{n\times k}$ and $b^{(1)}, ..., b^{(m)} \in \mathbb{R}^{n\times 1}$, let

$$x^{(j)} = \arg\min_{x\in\mathbb{R}^{k\times 1}} \|A^{(j)}x - b^{(j)}\|_2^2$$

where $j \in [m]$. Let S be a random Gaussian matrix of size $t \times n$, and let

$$y^{(j)} = \arg\min_{y\in\mathbb{R}^{k\times 1}} \|SA^{(j)}x - Sb^{(j)}\|_2^2$$

where $j \in [m]$. Then, for all $\epsilon \in (0, 1/2)$, we can set $t = O(k/\epsilon)$ such that:

$$\sum_{j=1}^{m} \|A^{(j)}x - b^{(j)}\|_2^2 \leq (1 + \epsilon) \sum_{j=1}^{m} \|A^{(j)}x - b^{(j)}\|_2^2$$

with probability at least $\frac{9}{10}$.

2.4.2 Warmup, Inefficient WLRA Algoritm

Input: $A \in \mathbb{R}^{n\times n}$, $W \in \mathbb{R}^{n\times n}$, $k \in \mathbb{N}$, $\epsilon > 0$, suppose $A_{i,j} \in \{0, \pm 1, \pm 2, ..., \pm\Delta\}$, $W_{i,j} = \{0, 1, 2, ..., \Delta\}$.

Output: We want to output a rank-k matrix $\hat{A} \in \mathbb{R}^{n\times n}$ such that

$$\|W \circ (\hat{A} - A)\|_F^2 \leq (1 + \epsilon) \cdot OPT$$

with probability at least $\frac{9}{10}$.

Naive Algorithm:

- Create $2nk$ variables for $U, V \in \mathbb{R}^{n\times k}$.
- Set the polynomial $g(x_1, ..., x_{2nk}) = \|W \circ (A - UV^T)\|_F^2$.
- Pick $C \in [L^-, L^+]$, run polynomial verifier $g(x) \leq C$.
- Optimize C by binary search over $[L^-, L^+]$.

The runtime for the above algorithm is $2^{\Omega(nk)}$. How can we do better? The step which incurs too much is step 1. Note that the polynomial verifier runs in $(\text{constrants} \cdot \text{degree})^{O(\text{variables})}$ and the lower bound of the cost is $(\text{constraints})^{-\text{degree}^{O(\text{variables})}}$. We would like to write a polynomial with a few number of variables, $poly(kr/\epsilon)$ without increasing degree and number of constraints (by too much).

2.4.3 Guessing a Sketch

To reduce the number of variables to $poly(kr/\epsilon)$, given that we can do Multiple regression sketch with $O(k/\epsilon)$ rows and the Weight matrix W has rank at most r. Consider the following algorithm:

- Given the same inputs A, W, r, k, ϵ. Let W_j be the j-th column of W and set D_{W_j} to be a diagonal matrix with vector W_j.
- The objective function is:

$$\|W \circ (UV^T - A)\|_F^2 \leq (1+\epsilon) \cdot OPT$$

which is equivalent with

$$\sum_{j=1}^{n} \|D_{W_j} U V_j^T - D_{W_j} A_j)\|_F^2 \leq (1+\epsilon) \cdot OPT$$

which is in the form of a Multiple Regression problem.
- We can sketch by Gaussian Matrix $S \in \mathbb{R}^{t \times n}$ and turn the problem into

$$\sum_{j=1}^{n} \|SD_{W_j} U V_j^T - SD_{W_j} A_j)\|_F^2$$

and we can guess $SD_{W_j}U \in \mathbb{R}^{t \times k}$ by creating $t \times k$ variables for each of the n $SD_{W_j}U$s.

However, this would take $n \times t \times k$ variables. To reduce this, notice that W has rank r, which mean we actually do not have to create variables for all n $SD_{W_j}U$s since some columns are linear combinations of others. We only need to create $SD_{W_j}U$ for columns that form a basis for the column space of W, and for the remaining columns we can express them as a linear combination of the existing variables in the column span. Then, we create $t \times k$ variables for each $SD_{W_j}U \in \mathbb{R}^{t \times k}$, which gives a total of $r \times t \times k$ variables.

2.4.4 Wrapping Up

We can use an explicit formula for the regression solution in terms of the variables created. The regression solution is a rational function, but can use tricks to clear the denominator. Multiple Regression Sketch + Bounded Rank Weight Matrix imply a small number of variables, and runs in time $n^{O(rk^2/\epsilon)}$.

3 Topics at a Survey Level

3.1 Projection onto Complicated Objects and Gaussian Mean Width

Least squares regression find the closest point y in a subspace K to a given point b. More formally, given a possibly infinite set of points K and a point b, least square regression computes

$$y' = \min_{y \in K} |y - b|.$$

Note that all norms were Euclidean norms in the above expression.

Let S be a sketching matrix. We want that if $y' = \arg\min_{y \in K} |Sy - Sb|$, then

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

More generally, we want to preserve distances of all vectors in a set K, that is

$$|S(y - y') = (1 \pm \epsilon)|y - y'|$$

for all $y, y' \in K$. The question is, what properties of K determine the dimension and sparsity of S?

3.1.1 Example: Preserving Distances in a Set

What is the answer to the previous question when K is:

- A set of n arbitrary points in $\mathbb{R}^d$?
- A set of n arbitrary points on a line in $\mathbb{R}^d$?

Hint: Johnson-Lindensruass Lemma]

3.1.2 Spherical Mean Width

Let K be a bounded subset in $\mathbb{R}^n$. We define the **width** in direction u for a unit vector u as:

$$u = \sum_{p,q \in K} \langle u, p - q \rangle.$$

Then the **spherical mean width** of K is just the expectation over (infinitely many) directions u of the above width direction:

$$s(K) = E_u[\sup_{p,q \in K} \langle u, p - q \rangle].$$

3.1.3 Gaussian Mean Width

We now formally define the **gaussian mean width**, $g(K)$, which can be thought of as describing the ℓ_2 complexity of the subspace K. Formally it is defined as:

$$g(K) = E_g[\sup_{p,q \in K} \langle g, p - q \rangle]$$

where $g \in \mathbb{R}^n$ is an i.i.d Gaussian vector $g \sim \mathcal{N}(0, I_n)$.

Examples: It is easy to compute the gaussian mean width for:

- $K = S^{n-1}$. In this case, $g(K) = \Theta(\sqrt{n})$.
- $K = \{$set of unit vectors in a d-dimensional subspace of $\mathbb{R}^n\}$. In this case, $g(K) = \Theta(\sqrt{d})$.
- $K = \{$t arbitrary unit vectors in $\mathbb{R}^n\}$. In this case, $g(K) = \Theta(\sqrt{\log(t)})$.

We will only present the analysis for the last case.

Analysis for t arbitrary unit vectors: Let $u_1, ..., u_t \in \mathbb{R}^n$ such that $\|u_i\|_2 = 1$. Let $g \in \mathbb{R}^n$ be an i.i.d. gaussian random vector. Let $Z_j = \langle u_j, g \rangle$ for all $j \in [t]$. We want to bound

$$E_g[\max_j Z_j].$$

We have, for any $\lambda > 0$,

$$E[e^{\lambda \max_j Z_j}] \leq \sum_{j=1}^{t} E[e^{\lambda Z_j}] = te^{\lambda^2/2}$$

where equality uses the fact that $E[e^{\lambda w}] = e^{\lambda^2/2}$ for a $\mathcal{N}(0,1)$ variable w. Then the above gives us

$$E_g[\max_j Z_j] \leq \frac{1}{\lambda} \log E[e^{\max_j Z_j}] \leq \frac{\log t}{\lambda} + \frac{\lambda}{2} = 2\sqrt{\log t}.$$

3.1.4 Sketching Bounds

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

$$|S(y - y')|^2 = (1 \pm \epsilon)|y - y'|^2$$

for all $y, y' \in K$.

[Bourgain, Dirksen, Nelson] - For sparse sketching matrices, S can have $m = g(K)^2 poly(\log n)/\epsilon^2$ rows and $s = poly(\log n)/\epsilon^2$ non-zeros per column if m and s satisfy a condition related to higher moments of $\sup_{p,q} \langle g, p - q \rangle$. [4]

3.2 M-Estimator Loss Functions for Regression

We have seen methods to use linear sketching and obtain fast, approximate randomized algorithms for ℓ_1 and ℓ_2 regression respectively. ℓ_1 regression can be solved efficiently using Linear Programming and is less sensitive to outliers, whereas ℓ_2^2 regression has desirable smoothness properties a closed form solution.

In practice, we can use fitness measures to try to combine the strengths of both ℓ_1 and ℓ_2^2 regression. A very common one used in practice is **Huber loss**:

$$M(x) = \begin{cases} \frac{x^2}{2c} \text{ for } |x| \leq c \\ |x| - \frac{c}{2} \text{ for } |x| > c \end{cases}.$$

The Huber loss function enjoys the smoothness properties of ℓ_2^2 and robustness properties of l_1. Here are some other examples of other fitness measures:

$L_1 - L_2$:

$$M(x) = 2\left(\sqrt{\frac{1 + x^2}{2}} - 1\right).$$

Fair Estimator:

$$M(x) = c^2\left(\frac{|x|}{c} - \log\left(1 + \frac{|x|}{c}\right)\right).$$

Tukey Estimator:

$$M(x) = \begin{cases} \frac{c^2}{6}\left(1-(1-(x/c)^2)^3\right) & \text{if } |x| \le c \\ \frac{c^2}{6} & \text{otherwise} \end{cases}.$$

3.2.1 Nice M-Estimators

We call an M-estimator **nice** if there exists a constant $C_M > 0$ such that for all a, a' with $|a| \ge |a'| > 0$,

$$C_M|\frac{a}{a'}| \le \frac{M(a)}{M(a')} \le |\frac{a}{a'}|^2.$$

Intuitively, an M-Estimator is nice if it has at least linear growth and at most quadratic growth. Furthermore, an M-estimator defined to have value 0 at 0.

Note that any convex M satisfies the lower bound of the above condition:

$$M(a') = M\left(\frac{a'}{a}\cdot a + (1-\frac{a'}{a})\cdot 0\right) \le \frac{a'}{a}M(a) + \left(1-\frac{a'}{a}\right)M(0) = \frac{a'}{a}\cdot M(a).$$

We call an M-estimator **sketchable** if there is a distribution on matrices $S \in \mathbb{R}^{k\times n}$ for which $|Sx|_M = \Theta(|x|_M)$ with good probability and k is a slow growing function of n. Any Sketchable M satisfies the quadratic lower bound.

3.2.2 Nice M-Estimator Theorem

Theorem 1. *(Woodruff, Clarkson, 2013) There exists an algorithm that uses sketching, runs in $O(nnz(A) + poly(d\log n))$ time and outputs $x' \in \mathbb{R}^d$ such that for any constant $C > 1$, with probability at least 0.99,*

$$\|Ax' - b\|_M \le C\min_x \|Ax - b\|_M.$$

Remark 1. For convex nice M-estimators, we can solve the M regression problem using convex programming in $poly(nd)$ time which is slow in practice.

Remark 2. The sketch of the approximate algorithm is universal, as in the same M-sketch works for all nice M-estimators.

3.2.3 M-Sketch

The universal M-sketch is the following block matrix:

$$T = \begin{bmatrix} S_0\cdot D_0 \\ S_1\cdot S_1 \\ \vdots \\ S_{\log n}\cdot D_{\log n} \end{bmatrix}$$

where each S_i is an independent CountSketch matrix with $poly(d)$ rows and n columns, and each D_i are diagonal and uniformly samples a $\frac{1}{(\log n)^i}$ fraction of the n rows. For more information on this please see [5].

3.3 Compressed Sensing

In this problem, we are trying to estimate a vector $x \in \mathbb{R}^n$ by taking random linear measurements of said vector. This might be because of physical constraints.

In other words, we are taking a random $r \times n$ sketching matrix S and observing $S \cdot x$. Our goal is to output a vector $x' \in \mathbb{R}^n$ such that:

$$\|x - x'\|_p = D \cdot \min_{\text{k-sparse}} |x - z|_q$$

where D is the distortion, which is also known as the ℓ_p/ℓ_q-guarantee. Let x_k denote the best k-sparse approximation to x, i.e. the largest k coordinates in magnitude.

For estimating x', there are deterministic (for all) and randomized (for each) schemes. The famous CountSketch matrix is a randomized scheme achieving ℓ_2/ℓ_2 with high probability:

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

3.3.1 CountSketch for Compressed Sensing

A CountSketch matrix S is a random linear map with $O(k \log n)$ rows. Multiplying by a CountSketch matrix S with $O(k \log n)$ rows can be thought of as $O(\log n)$ repetitions of hashing into $O(k)$ buckets. As we have seen before, S estimates every coordinate x_i of some $x \in \mathbb{R}^n$ up to an additive error of $\frac{|x-x_k|_2}{\sqrt{k}}$.

Suppose we output the $2k$-sparse vector $x' \in \mathbb{R}^n$ that consists of the top $2k$ (biggest magnitude) estimates given by CountSketch. We will prove that

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

is satisfied with high probability. Before proving this, let us define two useful terms.

- We call a coordinate, i, **heavy** if

$$|x_i| \geq \frac{\|x - x_k\|_2}{\sqrt{k}}.$$

It is easy to see that there can be at most $2k$ heavy coordinates.

- We call a coordinate, i, **super-heavy** if

$$|x_i| \geq 3 \cdot \frac{\|x - x_k\|_2}{\sqrt{k}}.$$

Note that the set of super-heavy coordinates is in the support of x'. Let said set be T.

Then we have

$$\begin{aligned}
|x - x'|_2 \leq & |(x - x')_T|_2 + |(x - x')_{[n]/T}|_2 \\
\leq & \sqrt{2k} \cdot \frac{|x - x_k|_2}{\sqrt{k}} + |(x - x_k)_{[n]/T}|_2 + |(x_k - x')_{[n]/T}|_2 \\
= & O(|x - x_k|_2)
\end{aligned}$$

as desired.

3.3.2 No Deterministic Algorithm Achieves ℓ_2/ℓ_2

Recall that ℓ_2/ℓ_2 requires us to output x' with

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

Let us consider $k = 1$ and suppose that there is some S that is a deterministic sketching matrix with $r = o(n)$ rows. It suffices to show that there is a vector $x \in kernel(S)$ such that

$$|x|_\infty \geq C|x - x_1|_2$$

for any constant $C > 0$. WLOG, assume that S has orthonormal rows. Because $\sum_i |Se_i|_2^2 = r$, there exists some i such that $|Se_i|_2^2 \leq \frac{r}{n}$. Now let $x = e_i - S^T Se_i$, so $x \in kernel(S)$. Then

$$|x|_\infty^2 \geq |x_i|^2 = (e_i^T e_i - e_i^T S^T Se_i)^2 \geq (1 - r/n)^2$$

while

$$|x - x_1|_2 \leq |x - e_i|_2 = |S^T Se_i|_2 = |Se_i|_2 \leq \sqrt{\frac{r}{n}} = o(1).$$

3.3.3 Deterministic Algorithms Achieve ℓ_2/ℓ_1

Recall that ℓ_2/ℓ_1 requires us to output x' with

$$\|x - x'\|_2 = O(1/\sqrt{k}) \cdot \|x - x_k\|_1.$$

We will say that the matrix S has the (ϵ, k)**-restricted isometry property** if for all k-sparse vectors $x \in \mathbb{R}^n$:

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

It can be proved that if S has the (ϵ, k)-restricted isometry property then one can efficiently output $x' \in \mathbb{R}^n$ such that

$$\|x - x'\|_2 = O(1/\sqrt{k})\|x - x_k\|_1$$

by solving the Linear Programming problem:

$$\min_{z \in \mathbb{R}^n} |z|_1 \text{ such that } Sz = Sx.$$

If x' is the solution, then

$$\|x - x'\|_2 = O(1/\sqrt{k})\|x - x_k\|_1.$$

The proof that x' satisfies the above equation uses the (ϵ, k)-RIP property and elementary norm manipulations.

There are deterministic, but not explicit matrices with $O(k \log(n/k))$ rows that have the (ϵ, k)-RIP property for constant ϵ.

4 Open Problems

- For NMF, the upper bound is $n^{O(k^2)}$ while the lower bound is $n^{\Omega(k)}$ - what is the right answer?

- For a rank r weight matrix W, the upper bound for WLRA is $n^{O(k^2 r/\epsilon)}$ but the lower bound is only $2^{\Omega(r)}$ - can we close this gap?

- Can we prove a hardness result with respect to the parameter k, e.g., a $2^{\Omega(k)}$ lower bound for WLRA problem?

- Is there an explicit matrix with (ϵ, k)-RIP property that has only $o(k^2)$ rows? Bourgain et al: can get $k^{2-\gamma}$ rows for constant $\gamma > 0$ and $k \approx \sqrt{n}$. [6]

5 References

[1] Sanjeev Arora, Rong Ge, Ravi Kannan, Ankur Moitra. Computing a Nonnegative Matrix Factorization – Provably. URL: https://arxiv.org/abs/1111.0952

[2] Ankur Moitra. A Singly-Exponential Time Algorithm for Computing Nonnegative Rank. URL: https://eccc.weizmann.ac.il//report/2012/053/

[3] Ilya Razenshteyn, Zhao Song, and David P. Woodruff. Weighted low rank approximations with provable guarantees. URL: http://www.cs.cmu.edu/afs/cs/user/dwoodruf/www/rsw16.pdf

[4] Jean Bourgain, Sjoerd Dirksen, Jenali Nelson. Toward a unified theory of sparse dimensionality reduction in Euclidean space. arxiv preprint arXiv:1311.2542, 2014

[5] Kenneth L. Clarkson, David P. Woodruff. Input Sparsity for Robust Subspace Approximation. arxiv preprint arxiv:1510.06073

[6] J.Bourgain, S.Dilworth, K.Ford, S.Konyagin and D.Kutzarova, Explicit construction of RIP matrices and related problems, Duke Mathematical Journal, Vol.159 (2011), no.1, pages 145-185.