Randomized Algorithms 2015A
Lecture 9 – Dimension Reduction in $\ell_2$, Sketching, and NNS in $\ell_1$

Robert Krauthgamer

1 Dimension Reduction in $\ell_2$

The Johnson-Lindenstrauss (JL) Lemma: Let $x_1, \ldots, x_n \in \mathbb{R}^d$ and fix $\varepsilon > 0$. Then there exist $y_1, \ldots, y_n \in \mathbb{R}^k$, $k = O(\varepsilon^{-2} \log n)$, such that
\[ \forall i, j \in [n], \quad \|y_i - y_j\| \in (1 \pm \varepsilon)\|x_i - x_j\|. \]

Moreover, there is a randomized linear mapping $L : \mathbb{R}^d \rightarrow \mathbb{R}^k$ (oblivious to the given points), such that if we define $y_i = Lx_i$, then with probability at least $1 - 1/n$ all the above inequalities hold.

Remark: Note there is no assumption on the input points (e.g., that they lie on a low-dimensional space).

Idea: The map $L$ is essentially (up to normalization) a matrix of standard Gaussian. In fact, random signs $\pm 1$ would also work!

Since $L$ is linear, $Lx_i - Lx_j = L(x_i - x_j)$, and it suffices to verify that $L$ preserves the norm of any vector (instead of looking at pairs of vectors).

Main Lemma: Let $G : \mathbb{R}^{d \times k}$ be a random matrix of standard gaussians, for suitable $k = O(\varepsilon^{-2} \log n)$.
\[ \forall v \in \mathbb{R}^d, \quad \Pr[\|Gv\| \in (1 \pm \varepsilon)\sqrt{k}\|v\|] \geq 1 - 2/n^3. \]

We saw in class how the theorem’s proof using the Main Lemma, and also how to prove the latter using the following fact and claim.

Fact (Gaussians are 2-stable): Let $X_1, \ldots, X_n$ be independent standard Gaussian $N(0, 1)$, and let $\sigma_1, \ldots, \sigma_n \in \mathbb{R}$. Then $\sum_i \sigma_i X_i \sim N(0, \sum_i \sigma_i^2)$.

Claim: Let $Y$ have chi-squared distribution with parameter $k$, i.e., $Y = \sum_{i=1}^k X_i^2$ for independent $X_1, \ldots, X_k \sim N(0, 1)$. Then
\[ \forall \varepsilon \in (0, 1), \quad \Pr[Y > (1 + \varepsilon)^2 k] \leq e^{-(3/4)\varepsilon^2 k}. \]
Remark: This claim and its proof are similar to Chernoff bounds.

2 Sketching

**What is Sketching:** We have some input \( x \), which we want to “compress” into a sketch \( s(x) \) (much smaller), but want to be able to later compute some \( f(x) \) only from the sketch. Often, randomization helps. We’ll denote it as \( s_r(x) \) where \( r \) is the sequence of random coins.

**Examples:**

1. Sketching \( x \in \mathbb{R}^n \) so that later we could estimate any \( x_i \) (point queries).

2. Sketching for equality testing by hashing and testing whether \( h(x) = h(y) \), using a hash function \( h : \{0,1\}^n \to \{0,1\}^t \), for instance a random function or as in the exercise below (an inner product \( \langle x, r \rangle \) in \( GF[2] \)). It’s important here to choose \( h \) using public randomness, i.e., same \( h \) for both \( x, y \).

Exer: Analyze the hash function \( h_r(x) = \sum_{i=1}^n x_i r_i \pmod{2} \), where \( r \in \{0,1\}^n \) is random, offers a good sketch for equality testing in the sense that

\[
\forall x \neq y, \quad \Pr_r[h_r(x) = h(y)] = 1/2.
\]

3. Sketching for \( \ell_p \) distance, namely, for all \( x, y \in [n]^n \),

\[
\Pr[a(s_r(x), s_r(y)) = (1 \pm \varepsilon)||x - y||_p] \geq 2/3.
\]

We implemented such \( s \) for \( \ell_2 \) norm using a linear sketch \( L : [n]^n \to \mathbb{Z}^k \) for \( k = O(1/\varepsilon^2) \), hence \( |s(x)| \leq O(\varepsilon^{-2} \log n) \) bits.

Question: Can we use (for \( \ell_1 \) or \( \ell_2 \)) only \( O(\varepsilon^{-2}) \) bits? No if we want an estimate. But maybe for a decision version (output is YES/NO)?

**Theorem 1 [Estimating \( \ell_1 \) distance]:** For all \( 0 < \varepsilon < 1 \) there is a randomized sketching algorithm (simulatenous protocol) that can estimate the \( \ell_1 \) (or Hamming) distance between vectors within factor \( 1 + \varepsilon \) in the decision version (i.e., given any parameter \( R > 0 \), it can decide whether \( ||x - y|| \leq R \) or \( > (1 + \varepsilon)R \)) with sketch size \( O(1/\varepsilon^2) \).

The sketching algorithm seen in class had two steps, the first chooses \( I \subset [n] \) to subsample the coordinates with rate \( 1/R \), and the second applies to \( x_I, y_I \) the equality testing mentioned earlier (inner-product in \( GF[2] \)).

**Review of key points:**

1. Design a single-bit sketch with small “advantage”

2. Amplify success probability using Chernoff bounds
3 NNS under $\ell_1$ norm (logarithmic query time)

**Problem definition (NNS):** Preprocess a dataset of $n$ points $x_1, \ldots, x_n \in \mathbb{R}^d$, so that then, given a query point $q \in \mathbb{R}^d$, we can quickly find the closest data point to the query, i.e. report $x_i$ that minimizes $\|q - x_i\|_1$.

Performance measure: Preprocessing (time and space) and query time.

Two naive solutions: exhaustive search with query time $O(n)$, and preparing all answer in advance with preprocessing space $2^d$ (at least).

Challenge: being polynomial in dimension $d$, but still getting query time sublinear (or polylog) in $n$.

Approximate version (factor $c \geq 1$): find $x_j$ such that $\|q - x_j\|_1 \leq c \cdot \min_i \|q - x_i\|_1$.

**Theorem 2 [Indyk-Motwani’98, Kushilevitz-Ostrovsky-Rabani’98]:** For every $\varepsilon > 0$ there is a randomized algorithm for $1+\varepsilon$ approximate NNS in $\mathbb{Z}^d$ under $\ell_1$-norm with preprocessing space $n^{O(1/\varepsilon^2)} \cdot O(d)$ and query time $O(\varepsilon^{-2}d \text{ polylog } n)$.

Remark 1: We shall omit/neglect the precise polynomial dependence on $d$.

Remark 2: The success probability is for a single query (assuming it’s independent of the coins).

Remark 3: We only need to solve the decision version i.e. there is a target distance $R > 0$, and if there is data point $x_j$ such that $\|q - x_j\|_1 \leq R$ then we need to find point $x_i$ such that $\|q - x_i\|_1 \leq cR$. If no point is within distance $cR$, then report NONE. Otherwise, can report either answer. This follows by preparing in advance for all powers of $1+\varepsilon$ as the value of $R$ (then trying all of them or binary search).

Remark 4: WLOG $x_i$ and $q$ are in $\{0,1\}^d$.

**Main idea:** We basically repeat the single-bit sketching algorithm from Theorem 1 $k = O(\varepsilon^{-2} \log n)$ times to reduce the error probability to $1/n^2$, apply it to each $x_i$. We compute at query time $\tilde{s}(q) \in \{0,1\}^k$, but prepare “in advance” an answer for every possible value of $\tilde{s}(q)$, using a table of size $2^k$. 