Linear Sketches

- Johnson-Lindenstrauss, Random Projections for Dimensionality Reduction, and a Geometric Perspective on $F_2$ Sketching Algorithms

Linear Sketches:

- sketch $c \in \mathbb{R}^t$ where $t = O(\frac{1}{\epsilon})$

Sketch matrix $A$

<table>
<thead>
<tr>
<th>$i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c_1$</td>
</tr>
<tr>
<td>$c_2$</td>
</tr>
<tr>
<td>$c_n$</td>
</tr>
</tbody>
</table>

- columns $n$
- rows $t$

Recall the Tug-of-War Sketch from last lecture:

- $t = \frac{c}{\epsilon^3}$ for some $c = O(1)$

- Choose $h_1, \ldots, h_t : [\mathbb{R}] \rightarrow \{\pm 1\}$ at random from a pairwise independent family of hash functions.

- Initialize counters $c_1, \ldots, c_t$ to 0.

- While processing update $(a_j, \ell_j)$:
  - For $i = 1, \ldots, t$
    - $c_i \leftarrow c_i + \ell_j \cdot h_j(a_j)$
  - Output $\frac{1}{t} \cdot \sum_{i=1}^{t} c_i$

Sketch matrix $A$ for Tug-of-War sketch is defined as $A_{i,j} = h_i(j)$. 
Any algorithm of this form is called a linear sketch.

Any linear sketch works in the general turnstile update model.

Proof: Consider update \((a_j, \delta_j)\). What effect does this have on the frequency vector?

\[ f_a \leftarrow f_a + \delta_j \]

In vector notation:

\[ f \leftarrow f + \delta_j \cdot e_{a_j} \]

What effect does this have on the sketch \(\mathbf{A}f\)?

\[ \mathbf{A}f \leftarrow \mathbf{A}f + \mathbf{A}(\delta_j \cdot e_{a_j}) \]

By linearity, this is the same as

\[ \mathbf{A}f \leftarrow \mathbf{A}f + \mathbf{A}(\delta_j \cdot e_{a_j}) \]

\[ = \delta_j \cdot \mathbf{A} \cdot e_{a_j} \]

\[ = \delta_j \cdot \left[ a_j \text{th column of } \mathbf{A} \right] \]

so update to sketch only depends on \(a_j\) and \(\delta_j\), and can be computed in a streaming manner.

*Note:* [LMW14] prove a converse to this: Any turnstile streaming algorithm must be a linear sketch.

*Note:* \(\mathbf{A}\) has \(\ell \cdot n\) entries, so a streaming algorithm cannot afford to store it explicitly. In Tug-of-War sketch, \(\mathbf{A}\) has a succinct implicit representation since it is fully specified by the hash functions \(h_1, \ldots, h_\ell\) (this is \(O(\log^2 n)\) bits total).
Note: We showed last time that with probability at least $\frac{2}{3}$,
$$\frac{1}{t} \sum c_i^2 \in \left[ (1 - \varepsilon) \| A \|_2^2, (1 + \varepsilon) \| A \|_2^2 \right].$$

That is, the linear map $\frac{1}{t}A : \mathbb{R}^n \to \mathbb{R}^t$ preserved the norm
of $x$ up to a factor of $(1 \pm \varepsilon)$ with probability $\geq \frac{2}{3}$.

**Claim:** Suppose for any fixed vector $x \in \mathbb{R}^n$, $\| Ax \| \geq \varepsilon \| x \|$. Then with probability $\geq 1 - \delta$ over the random choice of $A$, then
for any set of $m$ vectors $x_1, \ldots, x_m \in \mathbb{R}^n$, with probability
$\geq 1 - \left( \frac{m}{2} \right) \delta$ over the random choice of $A$, it holds that
$$\forall i \neq j, \quad \| Ax_i - Ax_j \| \in \left[ (1 - \varepsilon) \| x_i - x_j \|, (1 + \varepsilon) \| x_i - x_j \| \right].$$

**Proof:** For each fixed pair $i \neq j$, apply the hypothesis to the
vector $x := x_i - x_j$. Union bound over all $\binom{m}{2}$ pairs $i \neq j$.

**Example application:** Clustering. Say you want to cluster $m$ points,
$x_1, \ldots, x_m \in \mathbb{R}^n$ into $k$ clusters so that all points in the same cluster
are close to each other (i.e., "similar"). Rather than running
a clustering algorithm on $x_1, \ldots, x_m$, which is expensive if $n$ is big,
first project $x_1, \ldots, x_m$ into a much lower dimensional space
and run the algorithm on the low-dimensional vectors. Much
and run the algorithm on the low-dimensional vectors. Much
and run the algorithm on the low-dimensional vectors. Much
A Different F2 Algorithm

Rather than choosing the random matrix A as per the
Thy-of-War sketch, choose each entry of A to be an independent
N(0,1) variable. (Note: A does not have a small-space representation,
but let’s ignore this for now).

Claim. With probability at least \( \frac{2}{3} \),
\[ \|A F^2 \|_2 = (\lceil 1 + \sqrt{2} \rceil - 1) \|A\|_2. \]

This is the Johnson-Lindenstrauss Lemma.

Proof: Fact 1. \[ \mathbb{E} \left[ \frac{1}{\sqrt{d}} \cdot \|A F^2\|_2 \right] = \|A\|_2. \]

Proof. For each \( y \in \mathbb{C}^d \), \[ \mathbb{E} [A F^2]^y = \mathbb{E} \left[ \left( \sum_{j=1}^{2n} A_{ij} f_j \right)^2 \right] \]

\[ = \mathbb{E} \left[ \sum_{j,j'} A_{ij} A_{ij'} f_j f_{j'} \right] \]

Linearity of expectation

\[ = \sum_{j,j'} f_j f_{j'} \mathbb{E} [A_{ij} A_{ij'}] = \sum_{j,j'} f_j f_{j'} \mathbb{E} [A_{ij} A_{ij'}] \]

\[ \sum_{j,j'} = \sum_{j} \mathbb{E} [A_{ij} A_{ij}] \cdot \mathbb{E} [A_{ij}] = 0 \]

\( A_{ij} \) is distributed according to the
\( \chi^2 \)-distribution with 1 degree of freedom.
Its expected value is 1; \( f_j \) the final sum
is just \[ \sum_j f_j^2 \]

Can bound the probability that \( \frac{1}{\sqrt{d}} \|A F^2\|_2 \) deviates significantly
from its expectation. Can be bounded via standard arguments about \( \chi^2 \)
An alternative proof of Fact 1:
Recall \((A_f)_{i} = \sum_{j=1}^{n} A_{ij}\), where each \(A_{ij}\) is independent and distributed \(\sim N(0, 1)\). A standard fact about the normal distribution is that \((A_f)_{i}\) has the same distribution as 
\[1_f \mathcal{N}(\mathbf{y})\] where \(\mathbf{y} \sim N(0, 1)\), 
\[\mathbb{E}(1_f^2 \cdot \mathcal{E} Y^2) = \frac{\pi}{2}\]
Thus, \(\mathbb{E}(A_f^2) = \mathbb{E}(1_f^2 \cdot \mathcal{E} Y^2) = \frac{\pi}{2}\)

---

**Fact:** Stable distributions exist for all \(p \in (0, 2]\). The normal distribution is \(2\)-stable. The Cauchy distribution is \(1\)-stable.

**Key Point:** For any \(p \geq 1\), it is easy to generate a sample from a \(p\)-stable distribution [Chamber-Mallows-Shack (1976)].

*Note: The expected value of the Cauchy distribution is infinite.*
A Basic Estimator for $\|f\|_{L^p}$ for any $0 < p \leq 2$.
(Recall for $p > 2$, estimating $\|f\|_{L^p}$ requires $\Theta(n^{1-\frac{1}{p}})$ space.)

- For each $j \in [n]$, let $z_j \sim D_p \leq \text{requires storing n numbers but let's ignore this for now.}$
- $x \leftarrow 0$
- When processing update $(a_j, \delta_j)$:
  $x \leftarrow x + \delta_j \cdot a_j$

Output $x / \text{median}(D_p) \leq \text{can't use mean}(D_p)$ since this might be infinite (e.g., for $p = 1$), $\left(D_p \text{ is Cauchy}\right)$

Claim: The median of the distribution of the basic estimator is precisely $\|f\|_{L^p}$.

Proof: By p-stability of $D_p$, $x \sim \|f\|_{L^p} \cdot Y$ where $Y \sim D_p$.
So the estimate is distributed as $\left(\frac{\|f\|_{L^p}}{\text{median}(D_p)}\right) \cdot Y$, which has median $\frac{\|f\|_{L^p}}{\text{median}(D_p)}$ of median $(Y) = \|f\|_{L^p}$.
Final Algorithm: Output the median of \( t = O \left( \frac{\log(\frac{d}{\epsilon})}{\epsilon^2} \right) \) copies of the Basic Estimator.

Two additional issues:

1) The Basic Estimator required storing \( n \) numbers drawn from \( D_p \).
2) In addition to these \( n \) numbers, these are real numbers so may take infinitely many bits to represent.

Deal with 2) by rounding to \( O \left( \frac{\log(n \cdot E^{d/2} \cdot \epsilon^{-1}}{\epsilon^2} \right) \) bits of precision and argue that this doesn't introduce too much error.

Deal with 1) by using a pseudo random number generator which require storing only a small random seed and generally bits that "look random" to the Basic Estimator.