COMPSCI 614: Randomized Algorithms with Applications to Data Science

Prof. Cameron Musco

University of Massachusetts Amherst. Spring 2024.
Lecture 9

Logistics

- Problem Set 2 is due Wednesday at 11:59pm.
- One page project proposal due Tuesday 3/12.

Summary

Last Time:

- Finish up ℓ_0 sampling analysis and applications to distributed and streaming graph connectivity.
- Start on linear sketching for frequency estimation.
- Count-sketch algorithm.

Summary

Last Time:

- Finish up ℓ_0 sampling analysis and applications to distributed and streaming graph connectivity.
- Start on linear sketching for frequency estimation.
- Count-sketch algorithm.

Today:

- Finish up Count-sketch analysis
- If time, start on randomized methods for matrix multiplication

Linear Sketching

- **Linear Sketching:** Compress data via a random linear function (i.e., the random matrix **A**), and prove that we can still recover useful information from the compression.

$$
\begin{array}{ccccc}
\text{Random sketching matrix } \mathbf{A} & & \mathbf{x} & & \mathbf{A}\mathbf{x} \\
\begin{bmatrix}
1 & -1 & 0 & 0 & 1 & -1 & 0 & 1 \\
-1 & 0 & 1 & 1 & 0 & 0 & -1 & 0 \\
1 & 1 & -1 & 0 & -1 & -1 & 0 & 1 \\
0 & -1 & -1 & -1 & 1 & 1 & 1 & 0
\end{bmatrix}
& &
\begin{bmatrix} 1 \\ 0 \\ 0 \\ -2 \\ 0 \\ 0 \\ 3 \\ 0 \end{bmatrix}
& = &
\begin{bmatrix} 1 \\ -2 \\ 1 \\ 5 \end{bmatrix}
\end{array}
$$

Linear Sketching

- **Linear Sketching:** Compress data via a random linear function (i.e., the random matrix **A**), and prove that we can still recover useful information from the compression.

$$\begin{array}{ccccc} & \text{Random sketching matrix } \mathbf{A} & x & & \mathbf{A}x \\ +1: & \begin{bmatrix} 1 & -1 & 0 & 0 & 1 & -1 & 0 & 1 \\ -1 & 0 & 1 & 1 & 0 & 0 & -1 & 0 \\ 1 & 1 & -1 & 0 & -1 & -1 & 0 & 1 \\ 0 & -1 & -1 & -1 & 1 & 1 & 1 & 0 \end{bmatrix} & \begin{bmatrix} 1 \\ 0 \\ 0 \\ -2 \\ 0 \\ 0 \\ 3 \\ 0 \end{bmatrix} +1 & = & \begin{bmatrix} 1 \\ -2 \\ 1 \\ 5 \end{bmatrix} \end{array}$$

- Linearity is useful because it lets us easily aggregate sketches in distributed settings and update sketches in streaming settings.
- May want to recover non-zero entries of x, estimate norms or other aggregate statistics, find large magnitude entries, sample entries with probabilities according to their magnitudes, etc.

Linear Sketching for ℓ_2 Heavy-Hitters

$x(i)$ = i^{th} entry of vector $\tilde{x}(i)$ = approximation given by sketch

Set up: We will show how to estimate each entry of a vector $x \in \mathbb{R}^n$ up to error $\pm\epsilon \cdot \|x\|_2$ with probability at least $1 - \delta$, from a small linear sketch, of size $O\left(\frac{\log(1/\delta)}{\epsilon^2}\right)$.

$\tilde{x}(i) \geq 2\epsilon\|x\|_2 \rightarrow x(i) \geq \epsilon\|x\|_2$

$x(i) \geq 3\epsilon\|x\|_2 \rightarrow \tilde{x}(i) \geq 2\epsilon\|x\|_2$

- This error guarantee allows recovering the indices of all 'heavy-hitter' entries with magnitude $> \cancel{2}3\epsilon\|x\|_2$.
- What are some possible application of this primitive?

$$\begin{bmatrix} 1.01 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

↳ Frequent items estimation

- quickly find high weights, or high activations in NN

$$\begin{bmatrix} 0 \\ 5 \\ 2 \\ 100 \\ 1000 \end{bmatrix}$$

- error log
- sparse Fourier detection
- estimating gradients

Count Sketch Algorithm – Visually

x(1) = 5 x(2) = -3 x(3) = 1 ... x(n) = 0

random hash functions

$\boldsymbol{h}: [n] \to [m]$

$\boldsymbol{s}: [n] \to \{-1,1\}$

m length array **y**

0	0	0	0	0	0	0	0	0	0

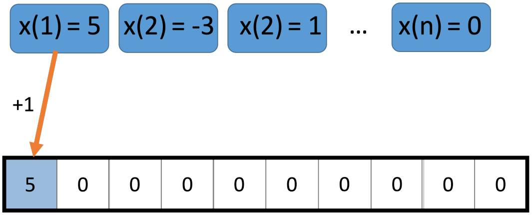
Count Sketch Algorithm – Visually

random hash functions

$\boldsymbol{h}: [n] \rightarrow [m]$

$\boldsymbol{s}: [n] \rightarrow \{-1,1\}$

m length array **y**

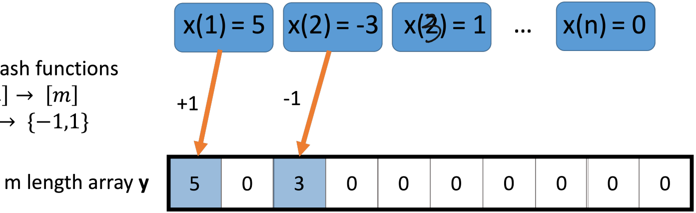


$h(1) = 1$

$s(1) = 1$

Count Sketch Algorithm – Visually

random hash functions
$\boldsymbol{h}: [n] \rightarrow [m]$
$\boldsymbol{s}: [n] \rightarrow \{-1,1\}$



$h(2) = 3$
$s(2) = -1$

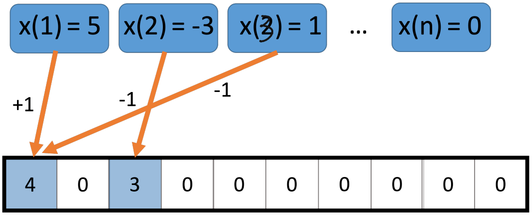
Count Sketch Algorithm – Visually

random hash functions

$\boldsymbol{h}: [n] \rightarrow [m]$

$\boldsymbol{s}: [n] \rightarrow \{-1,1\}$

m length array **y**



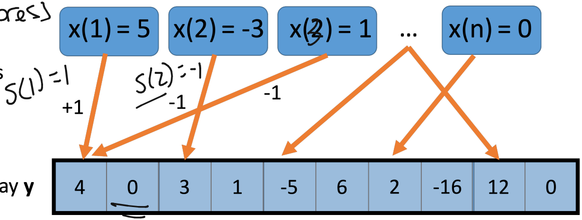
n = length of original vector

m = length of compress

random hash functions

$h: [n] \rightarrow [m]$

$s: [n] \rightarrow \{-1,1\}$

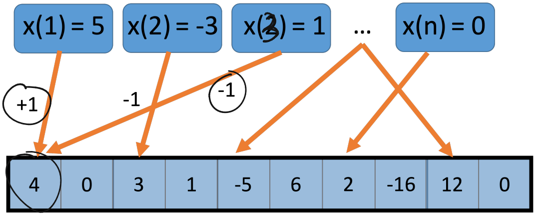


m length array **y**

$m << n$

$m = \frac{1}{\varepsilon^2}$

Count Sketch Algorithm – Visually



random hash functions
$\boldsymbol{h}: [n] \rightarrow [m]$
$\boldsymbol{s}: [n] \rightarrow \{-1,1\}$

Estimate: $x(i) \approx \mathrm{s}(i) \cdot y(\mathrm{h}(i))$

$\tilde{x}(1) = 1 \cdot 4 = 4$

$\tilde{x}(3) = -1 \cdot 4 = -4$

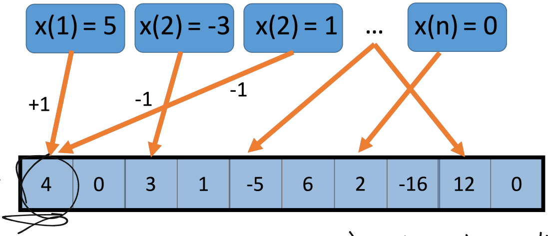
Count Sketch Algorithm – Visually

random hash functions

$\boldsymbol{h}: [n] \to [m]$

$\boldsymbol{s}: [n] \to \{-1,1\}$

m length array **y**



$s(i) \cdot x(i)\, s(i) = x(i)$

Estimate: $x(i) \approx \mathrm{s}(i) \cdot y(\mathrm{h}(i)) = \mathrm{s}(i) \cdot \sum_{k:\mathrm{h}_j(k)=\mathrm{h}_j(i)} x(k) \cdot \mathrm{s}(k)$

+10 w.p 1/2

-10 w.p 1/2

10

$= x(i) + \mathbb{E} \underbrace{\sum_{k \neq i:\mathrm{h}_j(k)=\mathrm{h}_j(i)} x(k) \cdot \mathrm{s}(k) \cdot \mathrm{s}(i)}_{\text{mean } 0 \text{ noise}}$

$\sum \mathbb{E}\, x(k) \cdot s(k) \cdot s(i)$

$\sum x(k) \cdot \mathbb{E}\, \overset{0}{\cancel{s(k)}}\, s(i)$

n columns

Random sketching matrix **A**

m rows

Count Sketch Algorithm – Psuedocode

$h(x)$ + $h(y)$ are independent

by $\{h(x), h(y), h(z)\}$ may not be independent

- Let $m = O(1/\epsilon^2)$ and $t = O(\log(1/\delta))$.
- Pick t random pairwise independent hash functions $\mathrm{h}_1, \ldots, \mathrm{h}_t : [n] \rightarrow [m]$.
- Pick t random pairwise independent hash functions $\mathrm{s}_1, \ldots, \mathrm{s}_t : [n] \rightarrow \{-1, 1\}$.

Count Sketch Algorithm – Psuedocode

- Let $m = O(1/\epsilon^2)$ and $t = O(\log(1/\delta))$.
- Pick t random pairwise independent hash functions $\mathrm{h}_1, \ldots, \mathrm{h}_t : [n] \to [m]$.
- Pick t random pairwise independent hash functions $\mathrm{s}_1, \ldots, \mathrm{s}_t : [n] \to \{-1, 1\}$.

Count Sketch Algorithm – Psuedocode

- Let $m = O(1/\epsilon^2)$ and $t = O(\log(1/\delta))$.
- Pick t random pairwise independent hash functions $\mathbf{h}_1, \ldots, \mathbf{h}_t : [n] \to [m]$.
- Pick t random pairwise independent hash functions $\mathbf{s}_1, \ldots, \mathbf{s}_t : [n] \to \{-1, 1\}$.
- Compute t independent estimates of $x(i)$ as $\tilde{\mathbf{x}}_j(i) = \mathbf{s}_j(i) \cdot \sum_{k:\mathbf{h}_j(k)=\mathbf{h}_j(i)} x(k) \cdot \mathbf{s}_j(k)$.

Count Sketch Algorithm – Psuedocode

- Let $m = O(1/\epsilon^2)$ and $t = O(\log(1/\delta))$.
- Pick t random pairwise independent hash functions $\mathbf{h}_1, \ldots, \mathbf{h}_t : [n] \to [m]$.
- Pick t random pairwise independent hash functions $\mathbf{s}_1, \ldots, \mathbf{s}_t : [n] \to \{-1, 1\}$.
- Compute t independent estimates of $x(i)$ as $\tilde{\mathbf{x}}_j(i) = \mathbf{s}(i) \cdot \sum_{k:\mathbf{h}_j(k)=\mathbf{h}_j(i)} x(k) \cdot \mathbf{s}(k)$.
- Output the median of $\{\tilde{\mathbf{x}}_1(i), \ldots, \tilde{\mathbf{x}}_t(i)\}$ as our final estimate of $x(i)$.

Recall: $\tilde{x}_j(i) = s_j(i) \cdot \sum_{k:h_j(k)=h_j(i)} x(k) \cdot s_j(k)$.

What is $\mathbb{E}[\tilde{x}_j(i)]$? $= x(i)$

Recall: $\tilde{\mathbf{x}}_j(i) = \mathbf{s}(i) \cdot \sum_{k:\mathbf{h}_j(k)=\mathbf{h}_j(i)} x(k) \cdot \mathbf{s}(k)$.

What is $\mathbb{E}[\tilde{\mathbf{x}}_j(i)]$?

$$
\begin{aligned}
\mathbb{E}[\tilde{\mathbf{x}}_j(i)] &= x(i) + \mathbb{E}\left[\sum_{k \neq i:\mathbf{h}_j(k)=\mathbf{h}_j(i)} x(k) \cdot \mathbf{s}(k) \cdot \mathbf{s}(i)\right] \\
&= x(i) + \sum_{k \neq i:\mathbf{h}_j(k)=\mathbf{h}_j(i)} x(k) \cdot \underbrace{\mathbb{E}[\mathbf{s}(k) \cdot \mathbf{s}(i)]}_{\begin{cases} +1 & \text{w.p. } 1/2 \\ -1 & \text{w.p. } 1/2 \end{cases}} \\
&= x(i).
\end{aligned}
$$

0

$\mathbb{E}[s(k) \cdot s(i)] = 0$

Recall: $\tilde{x}_j(i) = s(i) \cdot \sum_{k:h_j(k)=h_j(i)} x(k) \cdot s(k)$.

What is $\text{Var}[\tilde{x}_j(i)]$? $= x(i) + \sum_{\substack{k \neq i \\ h(k)=h(i)}} x(k) \cdot s(k) \cdot s(i)$

$\checkmark$

$\mathbb{E}[\tilde{x}(i)]$

Concentration Analysis

Recall: $\tilde{\mathrm{x}}_j(i) = \mathrm{s}(i) \cdot \sum_{k:\mathrm{h}_j(k)=\mathrm{h}_j(i)} x(k) \cdot \mathrm{s}(k)$.

What is $\mathrm{Var}[\tilde{\mathrm{x}}_j(i)]$?

$$\mathrm{Var}[\tilde{\mathrm{x}}_j(i)] = \mathrm{Var}\left[\sum_{k \neq i:\mathrm{h}_j(k)=\mathrm{h}_j(i)} x(k) \cdot \mathrm{s}(k) \cdot \mathrm{s}(i)\right]$$

Concentration Analysis

Recall: $\tilde{x}_j(i) = s(i) \cdot \sum_{k:h_j(k)=h_j(i)} x(k) \cdot s(k)$.

What is $\mathrm{Var}[\tilde{x}_j(i)]$?

random subset

$$\mathrm{Var}[\tilde{x}_j(i)] = \mathrm{Var}\left[\sum_{k \neq i : h_j(k)=h_j(i)} x(k) \cdot s(k) \cdot s(i)\right]$$

$$= \mathrm{Var}\left[\sum_{k \neq i} \mathbb{I}_{h_j(k)=h_j(i)} \cdot x(k) \cdot s(k) \cdot s(i)\right]$$

$$\begin{cases} 1 & \text{if } h_j(k) = h_j(i) \\ 0 & \text{otherwise.} \end{cases}$$

Recall: $\tilde{\mathbf{x}}_j(i) = \mathbf{s}(i) \cdot \sum_{k:\mathbf{h}_j(k)=\mathbf{h}_j(i)} x(k) \cdot \mathbf{s}(k)$.

What is $\mathrm{Var}[\tilde{\mathbf{x}}_j(i)]$?

$$
\begin{aligned}
\mathrm{Var}[\tilde{\mathbf{x}}_j(i)] &= \mathrm{Var}\left[\sum_{k \neq i:\mathbf{h}_j(k)=\mathbf{h}_j(i)} x(k) \cdot \mathbf{s}(k) \cdot \mathbf{s}(i)\right] \\
&= \mathrm{Var}\left[\sum_{k \neq i} \mathbb{I}_{\mathbf{h}_j(k)=\mathbf{h}_j(i)} \cdot x(k) \cdot \mathbf{s}(k) \cdot \mathbf{s}(i)\right] \\
&= \sum_{k \neq i} \mathrm{Var}\left[\underbrace{\mathbb{I}_{\mathbf{h}_j(k)=\mathbf{h}_j(i)} \cdot x(k) \cdot \mathbf{s}(k) \cdot \mathbf{s}(i)}_{Z}\right]
\end{aligned}
$$

$$
Z = \begin{cases} 0 & \text{w.p. } 1 - \frac{1}{m} \\ x(k) & \text{w.p. } \frac{1}{2m} \\ -x(k) & \text{w.p. } \frac{1}{2m} \end{cases}
$$

$$\mathrm{Var}(Z) = \frac{x(k)^2}{m}$$

Concentration Analysis

Recall: $\tilde{\mathbf{x}}_j(i) = \mathbf{s}(i) \cdot \sum_{k:\mathbf{h}_j(k)=\mathbf{h}_j(i)} x(k) \cdot \mathbf{s}(k)$.

What is $\mathrm{Var}[\tilde{\mathbf{x}}_j(i)]$?

$$\begin{aligned}
\mathrm{Var}[\tilde{\mathbf{x}}_j(i)] &= \mathrm{Var}\left[\sum_{k\neq i:\mathbf{h}_j(k)=\mathbf{h}_j(i)} x(k)\cdot \mathbf{s}(k)\cdot \mathbf{s}(i)\right] \\
&= \mathrm{Var}\left[\sum_{k\neq i} \mathbb{I}_{\mathbf{h}_j(k)=\mathbf{h}_j(i)} \cdot x(k)\cdot \mathbf{s}(k)\cdot \mathbf{s}(i)\right] \\
&= \sum_{k\neq i} \mathrm{Var}\left[\mathbb{I}_{\mathbf{h}_j(k)=\mathbf{h}_j(i)} \cdot x(k)\cdot \mathbf{s}(k)\cdot \mathbf{s}(i)\right] \\
&= \sum_{k\neq i} \frac{1}{m}\cdot x(k)^2
\end{aligned}$$

Concentration Analysis

Recall: $\tilde{\mathbf{x}}_j(i) = \mathbf{s}(i) \cdot \sum_{k:\mathbf{h}_j(k)=\mathbf{h}_j(i)} x(k) \cdot \mathbf{s}(k)$.

What is $\mathrm{Var}[\tilde{\mathbf{x}}_j(i)]$?

$$\begin{aligned}
\mathrm{Var}[\tilde{\mathbf{x}}_j(i)] &= \mathrm{Var}\left[\sum_{k\neq i:\mathbf{h}_j(k)=\mathbf{h}_j(i)} x(k)\cdot\mathbf{s}(k)\cdot\mathbf{s}(i)\right] \\
&= \mathrm{Var}\left[\sum_{k\neq i} \mathbb{I}_{\mathbf{h}_j(k)=\mathbf{h}_j(i)}\cdot x(k)\cdot\mathbf{s}(k)\cdot\mathbf{s}(i)\right] \\
&= \sum_{k\neq i}\mathrm{Var}\left[\mathbb{I}_{\mathbf{h}_j(k)=\mathbf{h}_j(i)}\cdot x(k)\cdot\mathbf{s}(k)\cdot\mathbf{s}(i)\right] \\
&= \sum_{k\neq i}\frac{1}{m}\cdot x(k)^2 \leq \frac{\|x\|_2^2}{m}.
\end{aligned}$$

$|\tilde{x}(i) - x(i)|$

$\sim \frac{\|x\|_2}{\sqrt{m}}$

$m = 1/\varepsilon^2$

$\sim \varepsilon\|x\|_2$

Concentration Analysis

Recall: $\tilde{\mathrm{x}}_j(i) = \mathrm{s}(i) \cdot \sum_{k:\mathrm{h}_j(k)=\mathrm{h}_j(i)} x(k) \cdot \mathrm{s}(k)$.

What is an upper bound on $\Pr[|\tilde{\mathrm{x}}_j(i) - x(i)| \geq \epsilon \|x\|_2]$?

Concentration Analysis

Recall: $\tilde{\mathbf{x}}_j(i) = \mathbf{s}(i) \cdot \sum_{k:\mathbf{h}_j(k)=\mathbf{h}_j(i)} x(k) \cdot \mathbf{s}(k)$.

What is an upper bound on $\Pr[|\tilde{\mathbf{x}}_j(i) - x(i)| \geq \epsilon \|x\|_2]$?

By Chebyshev's inequality:

$$\frac{\|x\|_2^2}{m}$$

$$\Pr[|\tilde{\mathbf{x}}_j(i) - x(i)| \geq \epsilon \|x\|_2] \leq \frac{\mathrm{Var}[\tilde{\mathbf{x}}_j(i)]}{\epsilon^2 \|x\|_2^2} \leq \frac{1}{m\varepsilon^2}$$

Concentration Analysis

Recall: $\tilde{\mathbf{x}}_j(i) = \mathbf{s}(i) \cdot \sum_{k:\mathbf{h}_j(k)=\mathbf{h}_j(i)} x(k) \cdot \mathbf{s}(k)$.

What is an upper bound on $\Pr[|\tilde{\mathbf{x}}_j(i) - x(i)| \geq \epsilon \|x\|_2]$?

By Chebyshev's inequality:

$$\Pr[|\tilde{\mathbf{x}}_j(i) - x(i)| \geq \epsilon \|x\|_2] \leq \frac{\mathrm{Var}[\tilde{\mathbf{x}}_j(i)]}{\epsilon^2 \|x\|_2^2} \leq \frac{1}{\epsilon^2 \cdot m}$$

Concentration Analysis

Recall: $\tilde{\mathbf{x}}_j(i) = \mathbf{s}(i) \cdot \sum_{k:\mathbf{h}_j(k)=\mathbf{h}_j(i)} x(k) \cdot \mathbf{s}(k)$.

What is an upper bound on $\Pr[|\tilde{\mathbf{x}}_j(i) - x(i)| \geq \epsilon \|x\|_2]$?

By Chebyshev's inequality:

$$\Pr[|\tilde{\mathbf{x}}_j(i) - x(i)| \geq \epsilon \|x\|_2] \leq \frac{\mathrm{Var}[\tilde{\mathbf{x}}_j(i)]}{\epsilon^2 \|x\|_2^2} \leq \frac{1}{\epsilon^2 \cdot m} \leq \frac{1}{3}$$

If we set $m = \frac{3}{\epsilon^2}$, then our estimate is good with probability $\geq 2/3$.

Concentration Analysis

$$Var\left(\frac{1}{t}\sum_{j=1}^{t}\tilde{x}_j(i)\right) = \frac{1}{t}\cdot Var(\tilde{x}(i)) = \frac{\|x\|_2^2}{t\cdot m}$$

$\frac{3}{\epsilon^2}$

Recall: $\tilde{\mathbf{x}}_j(i) = \mathbf{s}(i)\cdot\sum_{k:\mathbf{h}_j(k)=\mathbf{h}_j(i)} x(k)\cdot\mathbf{s}(k)$.

What is an upper bound on $\Pr[|\tilde{\mathbf{x}}_j(i) - x(i)| \geq \epsilon\|x\|_2]$?

By Chebyshev's inequality:

$$\Pr[|\tilde{\mathbf{x}}_j(i) - x(i)| \geq \epsilon\|x\|_2] \leq \frac{\mathrm{Var}[\tilde{\mathbf{x}}_j(i)]}{\epsilon^2\|x\|_2^2} \leq \frac{1}{\epsilon^2\cdot m}$$

$1-\delta$

If we set $\boxed{m = \frac{3}{\epsilon^2}}$, then our estimate is good with probability $\geq 2/3$.

How large must we set m to increase our success probability to $\geq 1-\delta$?

$m = \frac{1}{\varepsilon^2\delta}$ — dependence on δ is bad

→ we'll improve to $\log(1/\delta)$

Median Trick for Count Sketch

To achieve $O(\log(1/\delta))$ dependence, Count Sketch uses the 'median trick'.

Median Trick for Count Sketch

To achieve $O(\log(1/\delta))$ dependence, Count Sketch uses the 'median trick'.

- Set $\boxed{m = 3/\epsilon^2}$ so each estimate $\tilde{\mathbf{x}}_j(i)$ is a $\pm\epsilon\|x\|_2$ approximation to $x(i)$ with probability at least 2/3.

Median Trick for Count Sketch

To achieve $O(\log(1/\delta))$ dependence, Count Sketch uses the 'median trick'.

- Set $m = 3/\epsilon^2$ so each estimate $\tilde{\mathbf{x}}_j(i)$ is a $\pm\epsilon\|x\|_2$ approximation to $x(i)$ with probability at least 2/3. $t = O(\log(1/\delta))$
- If we make t such estimates independently, we expect $2/3 \cdot t$ of them to be correct.

Median Trick for Count Sketch

To achieve $O(\log(1/\delta))$ dependence, Count Sketch uses the 'median trick'.

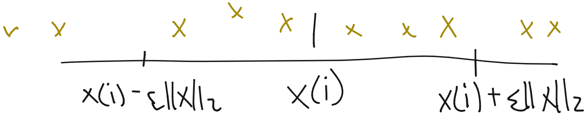
- Set $m = 3/\epsilon^2$ so each estimate $\tilde{\mathbf{x}}_j(i)$ is a $\pm\epsilon\|x\|_2$ approximation to $x(i)$ with probability at least $2/3$.
- If we make t such estimates independently, we expect $2/3 \cdot t$ of them to be correct. $\tilde{x}_1(i), \tilde{x}_2(i) \ldots \tilde{x}_t(i)$
- By a Chernoff bound, for $t = O(\log(1/\delta))$, $> 1/2$ will be correct with probability at least $1 - \delta$.

Median Trick for Count Sketch

"table size" $m = \frac{3}{\varepsilon^2}$ $t = \log(1/\delta)$ "repetitions" $\to$ CS returns $\pm\, \varepsilon \|x\|_2$ estimates of frequencies w.p. $\geq 1-\delta$.

To achieve $O(\log(1/\delta))$ dependence, Count Sketch uses the 'median trick'.

- Set $m = 3/\epsilon^2$ so each estimate $\tilde{\mathbf{x}}_j(i)$ is a $\pm\epsilon\|x\|_2$ approximation to $x(i)$ with probability at least $2/3$.
- If we make t such estimates independently, we expect $2/3 \cdot t$ of them to be correct.
- By a Chernoff bound, for $t = O(\log(1/\delta))$, $> 1/2$ will be correct with probability at least $1-\delta$.
- Thus, the median estimate will be correct with probability at least $1-\delta$.



Approximate Matrix Multiplication

Matrix Multiplication Problem

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

$\|x\|_2 = \sqrt{n}$

$\|x\|_1 = n$

$\tilde{x} = AA^T x$

Given $A, B \in \mathbb{R}^{n \times n}$ would like to compute $C = AB$. Requires n^ω time where $\omega \approx 2.373$ in theory.

Count Sketch Questions

$$t = \log(1/\delta) \left\{ \begin{bmatrix} \} m = \frac{3}{\varepsilon^2} \quad A_1 \\ A_2 \\ \vdots \\ A_t \end{bmatrix} \right. \overbrace{\begin{bmatrix} \\ x \\ \\ \end{bmatrix}}^{n} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_t \end{bmatrix} \rightarrow \begin{bmatrix} \\ \tilde{x} \\ \\ \end{bmatrix}$$

$\boxed{m = \frac{1}{\varepsilon^2} \quad \pm \varepsilon \|x\|_2}$

$\|x - x_k\|_2$

Count-min sketch

$(s(i) = 1$

$\hat{x}(i) = \min_{j=1..t} \hat{x}_j(i)$

$m = \frac{1}{\varepsilon}$

estimates w/ error

$\pm \varepsilon \|x\|_1$

Matrix Multiplication Problem

$$n\left[A\right]\left[d \quad \overset{k}{B}\right] \qquad + \epsilon\|A\|_F\|B\|_F \qquad \frac{1}{2^{2^{1/\epsilon}}} \cdot n^{2+\epsilon} \qquad n^{2+\epsilon} \text{ for } \epsilon > 0$$

Given $A, B \in \mathbb{R}^{n\times n}$ would like to compute $C = AB$. Requires n^ω time where $\omega \approx 2.373$ in theory.

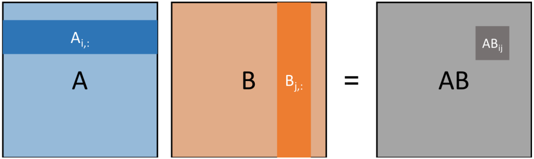
Today: We'll see how to compute an approximation in $O(n^2)$ time via a simple sampling approach.

- One of the most fundamental algorithms in randomized numerical linear algebra. Forms the building block for many other algorithms.

- regression
- low rank approx.
- clustering

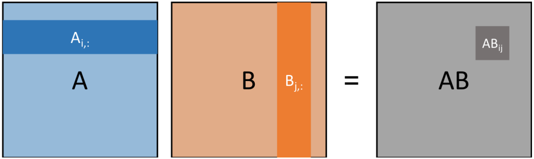
Outer Product View of Matrix Multiplication

Inner Product View: $[AB]_{ij} = \langle A_{i,:}, B_{j,:} \rangle = \sum_{k=1}^{n} A_{ik} \cdot B_{kj}$.

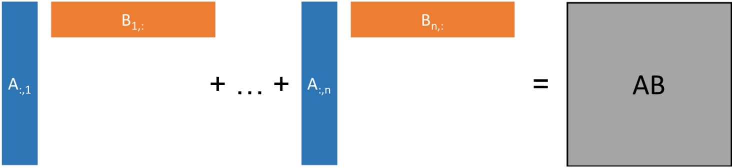


Outer Product View of Matrix Multiplication

Inner Product View: $[AB]_{ij} = \langle A_{i,:}, B_{j,:} \rangle = \sum_{k=1}^{n} A_{ik} \cdot B_{kj}$.

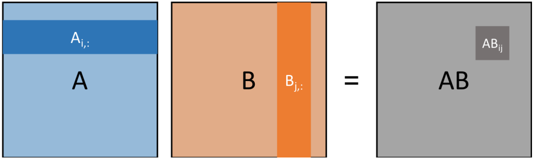


Outer Product View: Observe that $C_k = A_{:,k} B_{k,:}$ is an $n \times n$ matrix with $[C_k]_{ij} = A_{jk} \cdot B_{kj}$. So $AB = \sum_{k=1}^{n} A_{:,k} B_{k,:}$

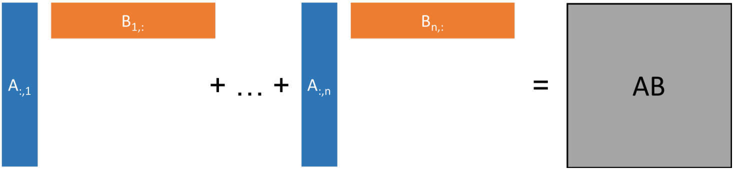


Outer Product View of Matrix Multiplication

Inner Product View: $[AB]_{ij} = \langle A_{i,:}, B_{j,:} \rangle = \sum_{k=1}^{n} A_{ik} \cdot B_{kj}$.



Outer Product View: Observe that $C_k = A_{:,k} B_{k,:}$ is an $n \times n$ matrix with $[C_k]_{ij} = A_{jk} \cdot B_{kj}$. So $AB = \sum_{k=1}^{n} A_{:,k} B_{k,:}$



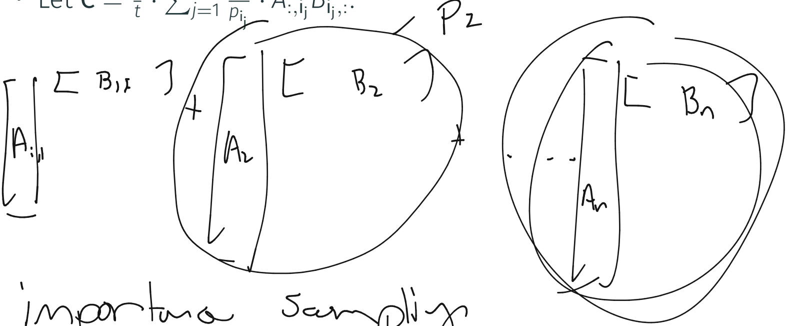
Basic Idea: Approximate **AB** by sampling terms of this sum.

Canonical AMM Algorithm

Approximate Matrix Multiplication (AMM):

? $= \frac{1}{n}$

- Fix sampling probabilities $p_1, \ldots, p_n$ with $p_i \geq 0$ and $\sum_{[n]} p_i = 1$.
- Select $\mathbf{i_1}, \ldots, \mathbf{i_t} \in [n]$ independently, according to the distribution $\Pr[\mathbf{i_j} = k] = p_k$.
- Let $\overline{\mathbf{C}} = \frac{1}{t} \cdot \sum_{j=1}^{t} \frac{1}{p_{\mathbf{i_j}}} \cdot A_{:,\mathbf{i_j}} B_{\mathbf{i_j},:}$.



importance sampling