CSCE 658: Randomized Algorithms

Lecture 23

Samson Zhou

Previously in the Streaming Model

- Reservoir sampling
- Heavy-hitters
  - Misra-Gries
  - CountMin
  - CountSketch
- Moment estimation
  - AMS algorithm

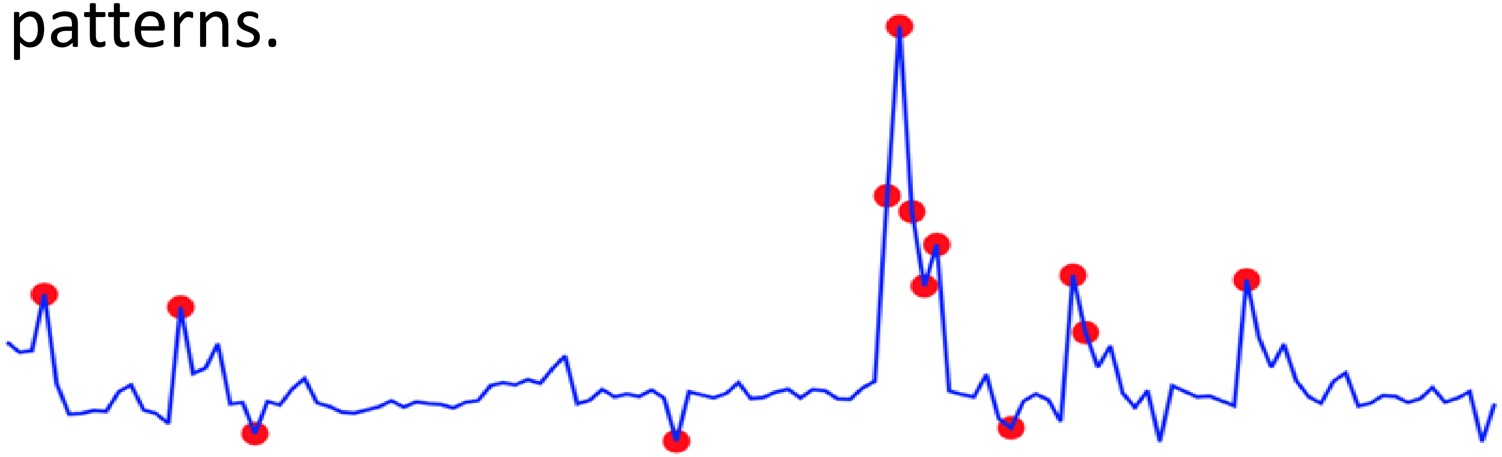
Sparse Recovery

- Suppose we have an insertion-deletion stream of length $m = \Theta(n)$ and at the end we are promised there are at most k nonzero coordinates
- Goal: Recover the k nonzero coordinates and their frequencies

Applications of Sparse Recovery

- Anomaly detection: Noiseless sparse recovery can be used to identify anomalies or outliers in streaming data

- By modeling normal behavior as a sparse signal, deviations from this model can be detected in real-time. This is valuable for cybersecurity, fraud detection, and monitoring network traffic for unusual patterns.

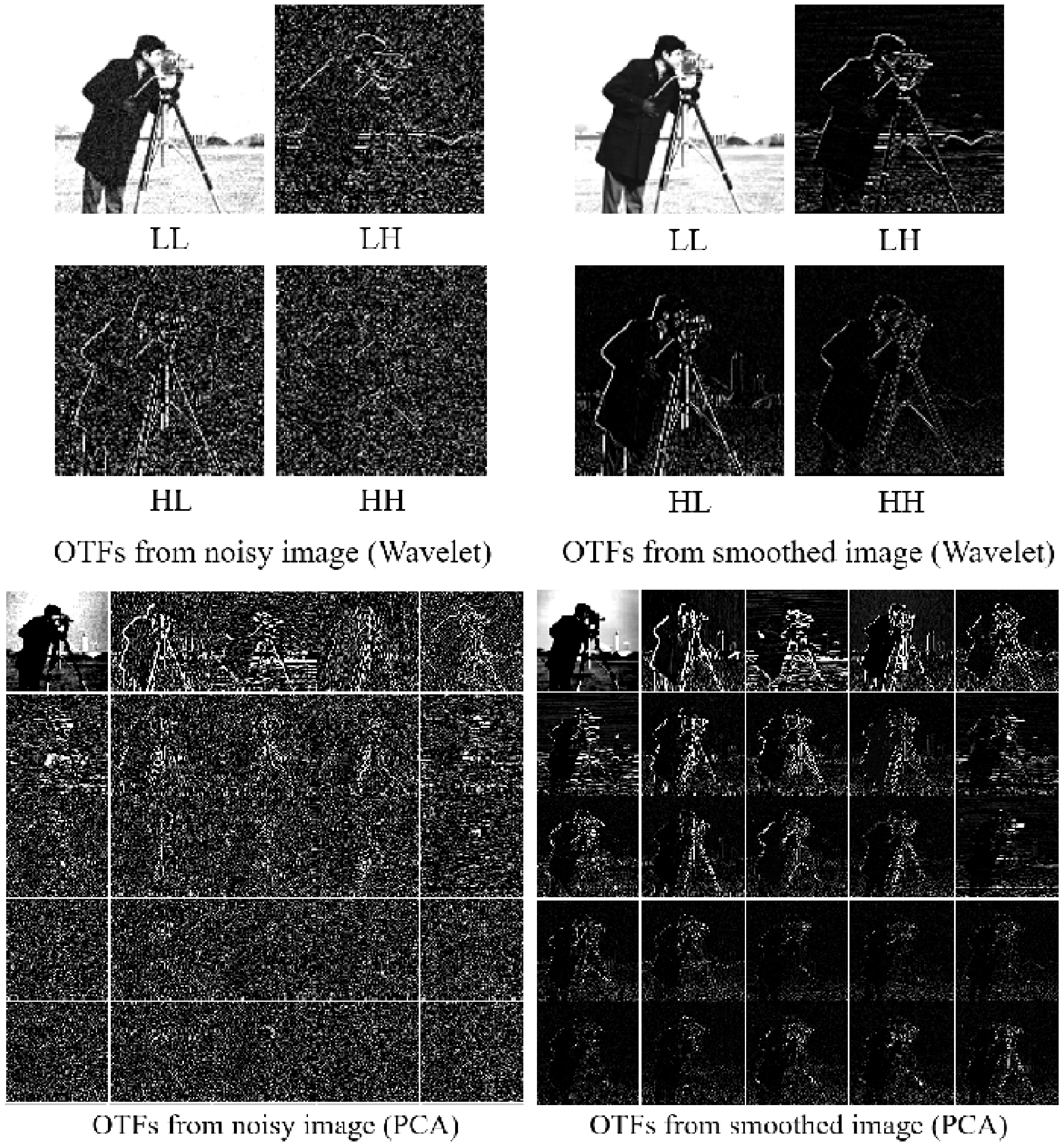


Applications of Sparse Recovery

- Network traffic analysis: Noiseless sparse recovery can be applied to analyze network traffic in real-time, identifying patterns and trends, and helping in network management, intrusion detection, and quality of service (QoS) optimization

Applications of Sparse Recovery

- Real-time compressive imaging: Compressive imaging techniques can be applied to streaming video or image data. By capturing and processing fewer measurements, noiseless sparse recovery can provide real-time reconstruction of high-resolution images or videos.



"Deep Orthogonal Transform Feature for Image Denoising", Shin, et. al. [2020]

Applications of Sparse Recovery

- Online natural language processing (NLP): In real-time natural language processing tasks, noiseless sparse recovery can assist in extracting relevant features or patterns from streaming text data, making it useful for sentiment analysis, topic modeling, and summarization

Sparse Recovery

- Suppose we have an insertion-deletion stream of length $m = \Theta(n)$

- Suppose at the end we are promised there are at most k nonzero coordinates

- How do we recover the vector?

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

u_1: “Increase f_6”

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

u_2: "Increase f_5"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

$$u_3\text{: “Increase } f_2\text{”}$$

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

u_4: "Increase f_7"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

u_5: "Increase f_3"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

u_6: “Increase f_3”

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

u_7: "Increase f_2"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

u_8: "Increase f_8"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

u_9: "Decrease f_3"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

u_{10}: "Decrease f_5"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

u_{11}: “Increase f_1”

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

u_{12}: "Increase f_7"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

u_{13}: "Decrease f_6"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

u_{14}: "Decrease f_8"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

u_{15}: "Decrease f_1"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

u_{16}: “Decrease f_7”

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

u_{17}: "Decrease f_3"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

u_{18}: “Decrease f_2”

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

u_{19}: “Decrease f_7”

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1
- What is left?

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1
- What is left?

$$f_2 = 1$$

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

- Algorithm: Keep running sum of all the coordinates

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ and we are promised the coordinate has frequency 1

- Algorithm: Keep running sum of all the coordinates

- Write each insertion to coordinate $c_i \in [n]$ as $u_i \leftarrow (s_i = 1, c_i)$
- Write each deletion to coordinate $c_i \in [n]$ as $u_i \leftarrow (s_i = -1, c_i)$

Sparse Recovery

- Suppose $k = 1$ and we are promised the coordinate j has frequency 1
- Algorithm: Keep running sum of all the coordinates

- Write each insertion to coordinate $c_i \in [n]$ as $u_i \leftarrow (s_i = 1, c_i)$
- Write each deletion to coordinate $c_i \in [n]$ as $u_i \leftarrow (s_i = -1, c_i)$

- Running sum of coordinates $\sum_{i \in [m]} s_i c_i = j$

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~

- Algorithm: Keep running sum of all the coordinates?

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate~~ $\cancel{j}$ ~~has frequency~~ $\cancel{1}$

- Algorithm: Keep running sum of all the coordinates AND a different linear combination of all the coordinates

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~

- Algorithm: Keep running sum of all the coordinates AND a different linear combination of all the coordinates

- Keep $\sum_{i\in[m]} s_i c_i$ and $\sum_{i\in[m]} s_i c_i^2$

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~

u_1: "Increase f_6"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~

u_2: "Increase f_5"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~

u_3: “Increase f_2”

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~

u_4: "Increase f_7"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~

u_5: “Increase f_3”

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~

u_6: "Increase f_3"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~

u_7: "Increase f_2"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~

u_8: "Increase f_8"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~

u_9: “Decrease f_3”

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~

u_{10}: "Decrease f_5"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~

u_{11}: "Increase f_1"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~

$$u_{12}\text{: “Increase } f_7\text{”}$$

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~

u_{13}: "Decrease f_6"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~

u_{14}: "Decrease f_8"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~

u_{15}: "Decrease f_1"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~

u_{16}: "Decrease f_7"

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~

u_{17}: “Decrease f_3”

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~

$$u_{18}: \text{“Decrease } f_7\text{”}$$

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~
- What is the state of our algorithm?

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~
- What is the state of our algorithm?

$$\sum_{i \in [m]} s_i c_i = 4 \text{ and } \sum_{i \in [m]} s_i c_i^2 = 8$$

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~
- What is the state of our algorithm?

$$\sum_{i\in[m]} s_i c_i = 4 \text{ and } \sum_{i\in[m]} s_i c_i^2 = 8$$

- We know $\sum_{i\in[m]} s_i c_i = j \cdot f_j$ and $\sum_{i\in[m]} s_i c_i^2 = j^2 \cdot f_j$

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~
- What is the state of our algorithm?

$$\sum_{i \in [m]} s_i c_i = 4 \text{ and } \sum_{i \in [m]} s_i c_i^2 = 8$$

- We know $\sum_{i \in [m]} s_i c_i = j \cdot f_j$ and $\sum_{i \in [m]} s_i c_i^2 = j \cdot f_j^2$
- So $f_j = 2$ and $j = 2$

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Suppose $k = 1$ ~~and we are promised the coordinate j has frequency 1~~
- What is the state of our algorithm?

$$\sum_{i \in [m]} s_i c_i = 4 \text{ and } \sum_{i \in [m]} s_i c_i^2 = 8$$

- We know $\sum_{i \in [m]} s_i c_i = j \cdot f_j$ and $\sum_{i \in [m]} s_i c_i^2 = j \cdot f_j^2$
- So $f_j = 2$ and $j = 2$

f_1	f_2	f_3	f_4	f_5	f_6	f_7
0	2	0	0	0	0	0

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates

- Algorithm for $k = 1$: Keep running sum of all the coordinates AND a different linear combination of all the coordinates

Sparse Recovery

- Suppose at the end we are promised there are at most k nonzero coordinates
- Algorithm: Keep $2k$ running sum of different linear combinations of all the coordinates
- We have $2k$ equations and $2k$ unknown variables
- Correctness can be shown (not quite linear algebra)

Sparse Recovery

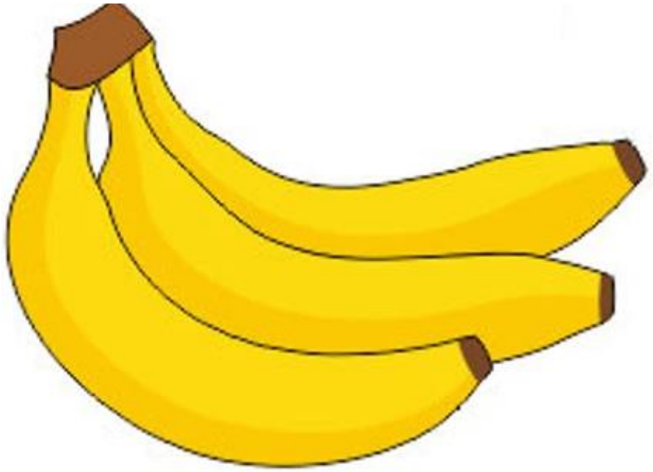
- Suppose at the end we are promised there are at most k nonzero coordinates
- Algorithm: Keep $2k$ running sum of different linear combinations of all the coordinates
- Space: $O(k)$ words of space

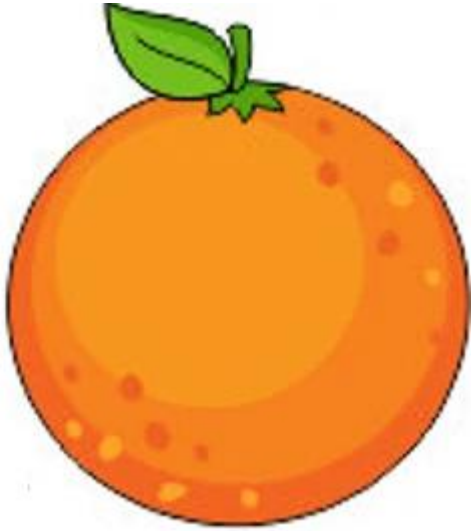
Distinct Elements (F_0 Estimation)

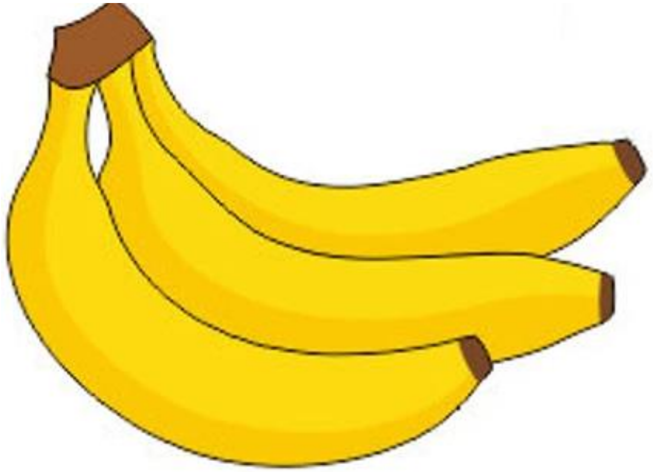
- Given a set S of m elements from $[n]$, let f_i be the frequency of element i. (How often it appears)
- Let F_0 be the frequency moment of the vector:

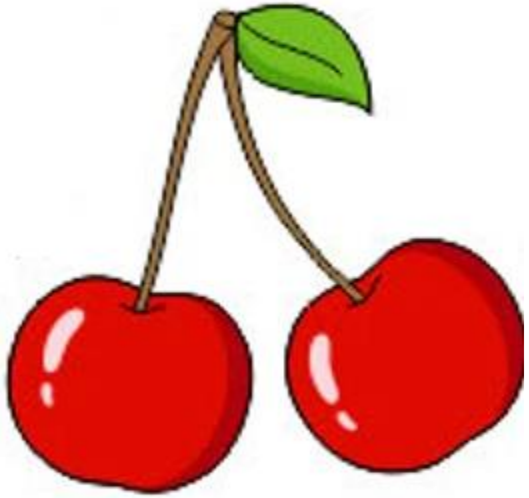
$$F_0 = |\{i : f_i \neq 0\}|$$

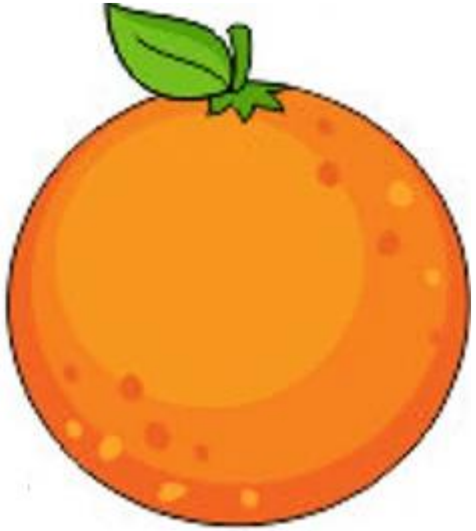
- Goal: Given a set S of m elements from $[n]$ and an accuracy parameter ε, output a $(1 + \varepsilon)$-approximation to F_0

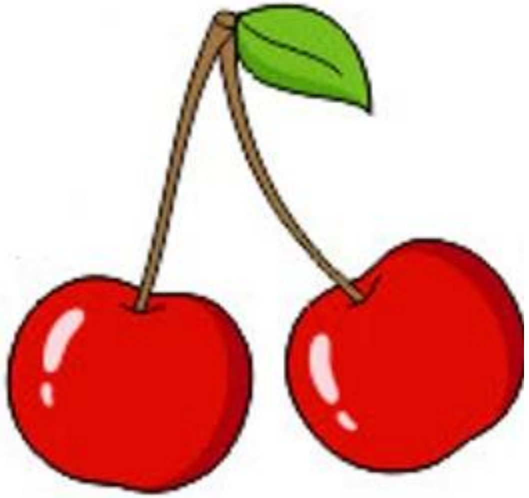



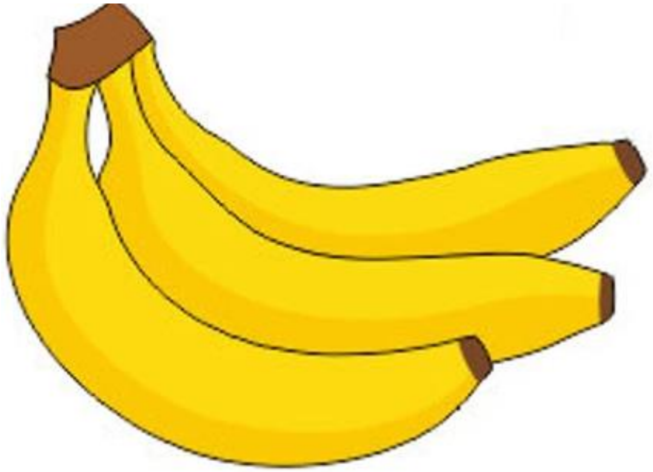

 

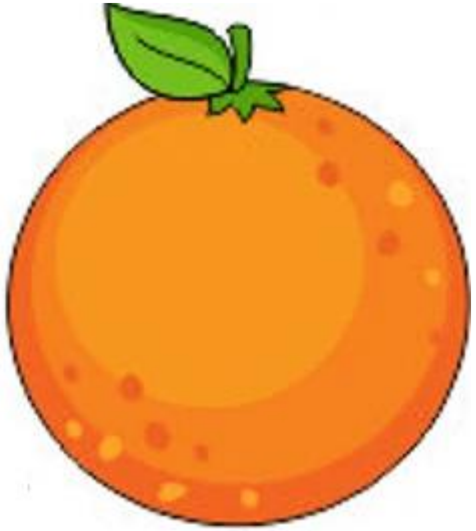


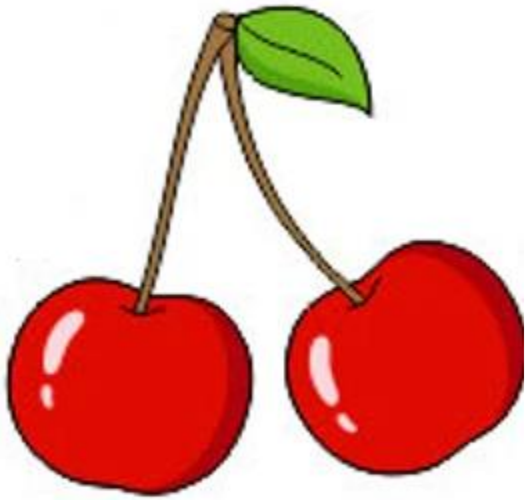
 

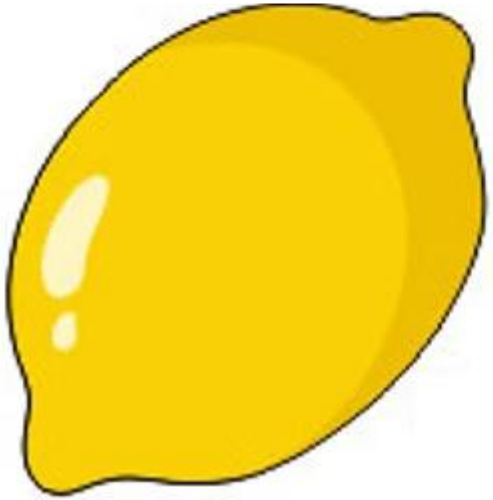








Distinct Elements (F_0 Estimation)

- How many different fruits left in fruit basket?

Distinct Elements (F_0 Estimation)

- How many different fruits left in fruit basket? 8

Distinct Elements (F_0 Estimation)

- Ad allocation: Distinct IP addresses clicking an ad

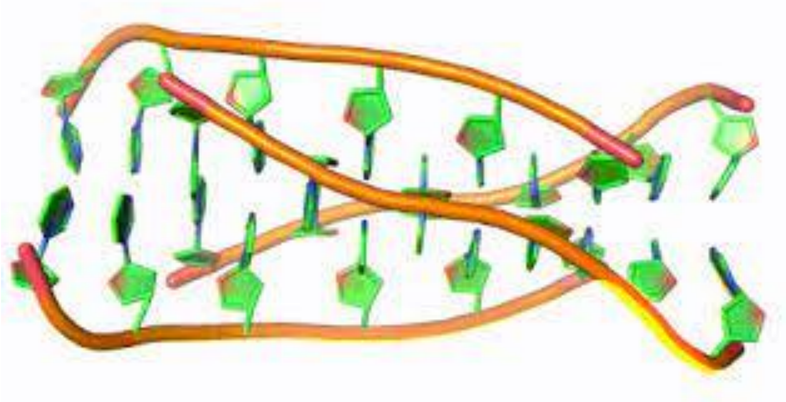
Distinct Elements (F_0 Estimation)

- Traffic monitoring: Distinct IP addresses visiting a site or number of unique search engine queries

3 billion monthly active users

Distinct Elements (F_0 Estimation)

- Computational biology: Counting number of distinct motifs in DNA sequencing



- Sequence motifs are short, recurring patterns in DNA that are presumed to have a biological function

Distinct Elements (F_0 Estimation)

- Let S be a set of N numbers
- Suppose we form set S' by sampling each item of S with probability $\frac{1}{2}$
- How many numbers are in S'?

Distinct Elements (F_0 Estimation)

- Let S be a set of N numbers

- Suppose we form set S' by sampling each item of S with probability $\frac{1}{2}$

- Can we use S' to get a good estimate of N?

Distinct Elements (F_0 Estimation)

- Let S be a set of N numbers, suppose we form set S' by sampling each item of S with probability $\frac{1}{2}$

- We have $\mathrm{E}[|S'|] = \frac{N}{2}$ and $\mathrm{Var}[|S'|] \leq \frac{N}{2}$

Distinct Elements (F_0 Estimation)

- What can we say about $\Pr\left[\left||S'| - \frac{N}{2}\right| \geq t\right]$?

- By Chebyshev's inequality, we have $\Pr\left[\left||S'| - \frac{N}{2}\right| \geq 100\sqrt{N}\right] \leq \frac{1}{10}$

Distinct Elements (F_0 Estimation)

- What can we say about $\Pr\left[\left||S'| - \frac{N}{2}\right| \geq t\right]$?
- By Chebyshev's inequality, we have $\Pr\left[\left||S'| - \frac{N}{2}\right| \geq 100\sqrt{N}\right] \leq \frac{1}{10}$
- With probability at least $\frac{9}{10}$,

$$\frac{N}{2} - 100\sqrt{N} \leq |S'| \leq \frac{N}{2} + 100\sqrt{N}$$

Distinct Elements (F_0 Estimation)

- With probability at least $\frac{9}{10}$,

$$\frac{N}{2} - 100\sqrt{N} \leq |S'| \leq \frac{N}{2} + 100\sqrt{N}$$

- Thus with probability at least $\frac{9}{10}$,

$$N - 200\sqrt{N} \leq 2|S'| \leq N + 200\sqrt{N}$$

Distinct Elements (F_0 Estimation)

- With probability at least $\frac{9}{10}$,

$$N - 200\sqrt{N} \leq 2|S'| \leq N + 200\sqrt{N}$$

- If $200\sqrt{N} \leq \frac{N}{100}$, then $N - 200\sqrt{N} \leq 2|S'| \leq N + 200\sqrt{N}$ implies

$$0.99N \leq 2|S'| \leq 1.01N$$

- Very good approximation to N

Distinct Elements (F_0 Estimation)

- What algorithm does this suggest?

Distinct Elements (F_0 Estimation)

- What algorithm does this suggest?

- Sample each item of the *universe* with probability $\frac{1}{2}$, acquire new universe U'
- Let S' be the items in the data stream that are in U'
- Output $2|S'|$

Distinct Elements (F_0 Estimation)

- Sample each item of the *universe* with probability $\frac{1}{2}$, acquire new universe U'
- Let S' be the items in the data stream that are in U'
- Output $2|S'|$

- What's the problem with this approach?

Distinct Elements (F_0 Estimation)

- Let S be a set of N numbers

- Suppose we form set S' by sampling each item of S with probability $\frac{1}{2}$

- Can we use S' to get a good estimate of N?

Distinct Elements (F_0 Estimation)

- Let S be a set of N numbers

- Suppose we form set S' by sampling each item of S with probability p

- Can we use S' to get a good estimate of N?

Distinct Elements (F_0 Estimation)

- Let S be a set of N numbers, suppose we form set S' by sampling each item of S with probability $\frac{1}{2}$
- We have $\mathrm{E}[|S'|] = \frac{N}{2}$ and $\mathrm{Var}[|S'|] \leq \frac{N}{2}$

Distinct Elements (F_0 Estimation)

- Let S be a set of N numbers, suppose we form set S' by sampling each item of S with probability p

- We have $\mathrm{E}[|S'|] = pN$ and $\mathrm{Var}[|S'|] \leq pN$

Distinct Elements (F_0 Estimation)

- (S' is formed by sampling each item of S with probability $\frac{1}{2}$) With probability at least $\frac{9}{10}$,

$$\frac{N}{2} - 100\sqrt{N} \leq |S'| \leq \frac{N}{2} + 100\sqrt{N}$$

- Thus with probability at least $\frac{9}{10}$,

$$N - 200\sqrt{N} \leq 2|S'| \leq N + 200\sqrt{N}$$

Distinct Elements (F_0 Estimation)

- (S' is formed by sampling each item of S with probability p) With probability at least $\frac{9}{10}$,

$$pN - 100\sqrt{pN} \leq |S'| \leq pN + 100\sqrt{pN}$$

- Thus with probability at least $\frac{9}{10}$,

$$N - \frac{100}{\sqrt{p}}\sqrt{N} \leq \frac{1}{p}|S'| \leq N + \frac{100}{\sqrt{p}}\sqrt{N}$$

Distinct Elements (F_0 Estimation)

- (S' is formed by sampling each item of S with probability p) With probability at least $\frac{9}{10}$,

$$N - \frac{100}{\sqrt{p}}\sqrt{N} \leq \frac{1}{p}|S'| \leq N + \frac{100}{\sqrt{p}}\sqrt{N}$$

- If $\frac{100}{\sqrt{p}}\sqrt{N} \leq \varepsilon N$, then $N - \frac{100}{\sqrt{p}}\sqrt{N} \leq \frac{1}{p}|S'| \leq N + \frac{100}{\sqrt{p}}\sqrt{N}$ implies

$$(1-\varepsilon)N \leq \frac{1}{p}|S'| \leq (1+\varepsilon)N$$

Distinct Elements (F_0 Estimation)

- In other words, with probability at least $\frac{9}{10}$, we have that $\frac{1}{p}|S'|$ is a $(1+\varepsilon)$-approximation of N

- What is p?

Distinct Elements (F_0 Estimation)

- In other words, with probability at least $\frac{9}{10}$, we have that $\frac{1}{p}|S'|$ is a $(1+\varepsilon)$-approximation of N

- What is p?
- Recall, we required $\frac{100}{\sqrt{p}}\sqrt{N} \leq \varepsilon N$

Distinct Elements (F_0 Estimation)

- In other words, with probability at least $\frac{9}{10}$, we have that $\frac{1}{p}|S'|$ is a $(1+\varepsilon)$-approximation of N

- What is p?
- Recall, we required $\frac{100}{\sqrt{p}}\sqrt{N} \leq \varepsilon N$, so $p \geq \frac{1000}{\varepsilon^2 N}$

Distinct Elements (F_0 Estimation)

- In other words, with probability at least $\frac{9}{10}$, we have that $\frac{1}{p}|S'|$ is a $(1+\varepsilon)$-approximation of N

- What is p?
- Recall, we required $\frac{100}{\sqrt{p}}\sqrt{N} \leq \varepsilon N$, so $p \geq \frac{1000}{\varepsilon^2 N}$
- What is the problem here?

Distinct Elements (F_0 Estimation)

- In other words, with probability at least $\frac{9}{10}$, we have that $\frac{1}{p}|S'|$ is a $(1+\varepsilon)$-approximation of N

- What is p?
- Recall, we required $\frac{100}{\sqrt{p}}\sqrt{N} \leq \varepsilon N$, so $p \geq \frac{1000}{\varepsilon^2 N}$
- What is the problem here?

Must know N to set p, but the goal is to find N!

Distinct Elements (F_0 Estimation)

- Observation: We do not need $p = \frac{1000}{\varepsilon^2 N}$, it is also fine to have $p = \frac{2000}{\varepsilon^2 N}$

- How do we find a "good" p?

Finding p

- Observation: We do not need $p = \frac{1000}{\varepsilon^2 N}$, it is also fine to have $p = \frac{2000}{\varepsilon^2 N}$

- How do we find a "good" p?

- What is a "good" p?

Finding p

- What is a “good” p?

- Not too many samples, i.e., S' is small, but enough to find a good approximation to N

- For $p = \Theta\left(\frac{1}{\varepsilon^2 N}\right)$:
  - p is large enough to find a good approximation to N
  - We have $\mathrm{E}[|S'|] = pN = \Theta\left(\frac{1}{\varepsilon^2}\right)$

Finding p

- We want p such that $\mathrm{E}[|S'|] = pN = \Theta\left(\frac{1}{\varepsilon^2}\right)$
- Intuition: Try $p = 1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \dots,$ and see which one has
$$\frac{1000}{\varepsilon^2} \le |S'| \le \frac{2000}{\varepsilon^2}$$
- With high probability, one of these guesses will have $\frac{1000}{\varepsilon^2} \le |S'| \le \frac{2000}{\varepsilon^2}$

Finding p

- Intuition: Try $p = 1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \dots,$ and see which one has

$$\frac{1000}{\varepsilon^2} \leq |S'| \leq \frac{2000}{\varepsilon^2}$$

- However, the wrong guesses will have too many samples

Finding p

- Intuition: Try $p = 1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \ldots,$ and see which one has

$$\frac{1000}{\varepsilon^2} \leq |S'| \leq \frac{2000}{\varepsilon^2}$$

- However, the wrong guesses will have too many samples

- Fix: Dynamically changing guess for p and subsampling

Finding p

- Algorithm: Set $U_0 = [n]$ and for each i, sample each element of U_{i-1} into U_i with probability $\frac{1}{2}$
- Start index $i = 0$ and track the number $|S \cap U_i|$ of elements of S in U_i
- If $|S \cap U_i| > \frac{2000}{\varepsilon^2} \log n$, then increment $i = i + 1$
- At the end of the stream, output $2^i \cdot |S \cap U_i|$

Finding p

- Algorithm: Set $U_0 = [n]$ and for each i, sample each element of U_{i-1} into U_i with probability $\frac{1}{2}$

- Start index $i = 0$ and track the number $|S \cap U_i|$ of elements of S in U_i

- If $|S \cap U_i| > \frac{2000}{\varepsilon^2} \log n$, then increment $i = i + 1$

$\frac{1}{p}$

- At the end of the stream, output $2^i \cdot |S \cap U_i|$

$|S'|$

Finding p

- Recall that $\frac{1}{p}|S'|$ is a $(1+\varepsilon)$-approximation of N

- $2^i \cdot |S \cap U_i|$ is a $(1+\varepsilon)$-approximation of N

$\frac{1}{p}$

- At the end of the stream, output $2^i \cdot |S \cap U_i|$

$|S'|$

Distinct Elements (F_0 Estimation)

- Summary: Algorithm stores at most $\frac{2000}{\varepsilon^2}\log n$ elements from the stream, uses $\Theta\left(\frac{1}{\varepsilon^2}\log n\right)$ words of space