COMPSCI 514: ALGORITHMS FOR DATA SCIENCE

Andrew McGregor

Lecture 10

(ϵ, k)-FREQUENT ITEMS PROBLEM

Given stream of n items $x_1, \ldots, x_n$ where each $x_i \in U$. Return a set F, such that for every $x \in U$:

1. If $f(x) \geq n/k$ then $x \in F$
2. If $f(x) < (1 - \epsilon)n/k$ then $x \notin F$

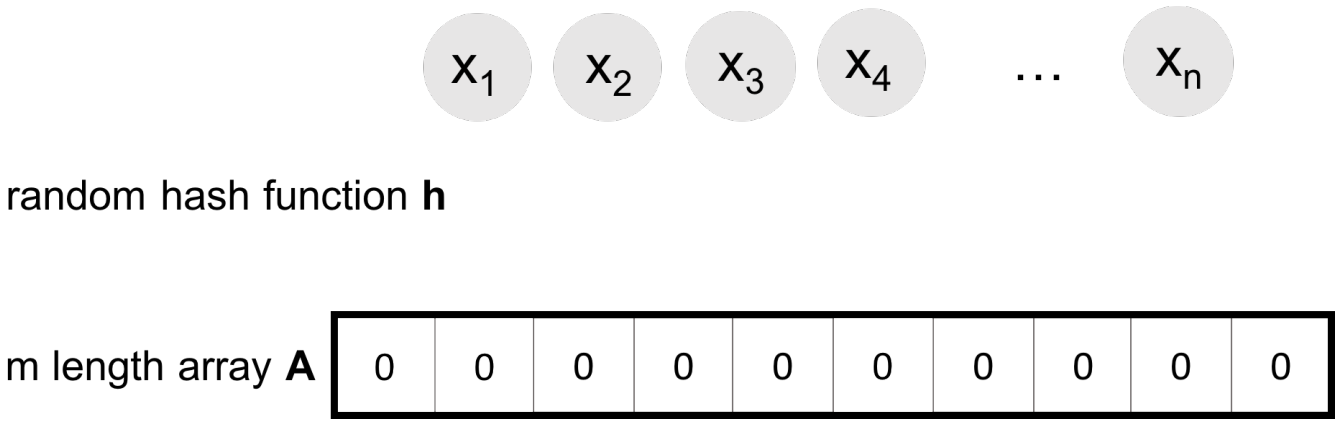
where $f(x)$ is the number of times x appears in the stream.

FREQUENT ELEMENTS WITH COUNT-MIN SKETCH

Count-Min Sketch: A random hashing based method closely related to bloom filters.

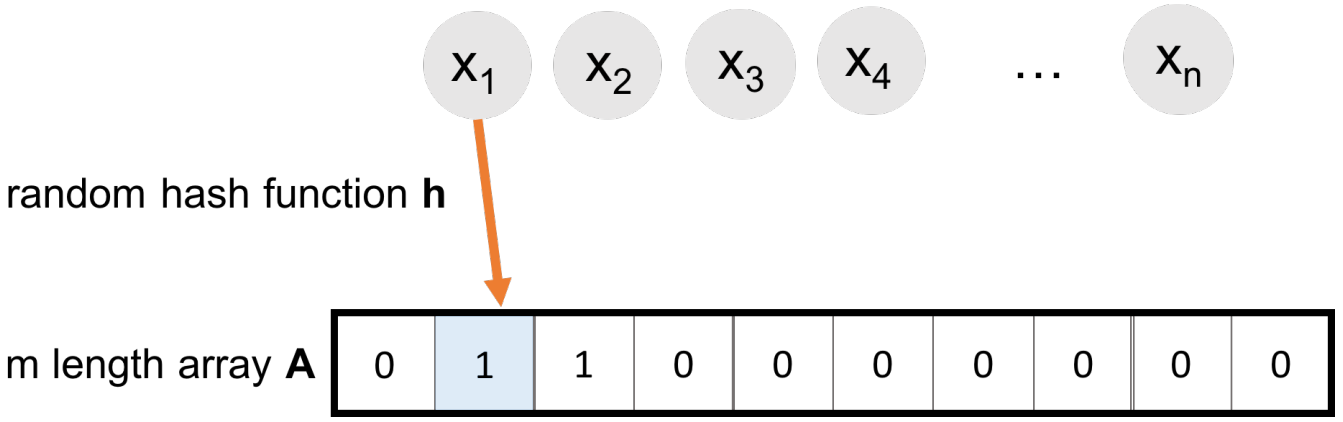
FREQUENT ELEMENTS WITH COUNT-MIN SKETCH

Count-Min Sketch: A random hashing based method closely related to bloom filters.



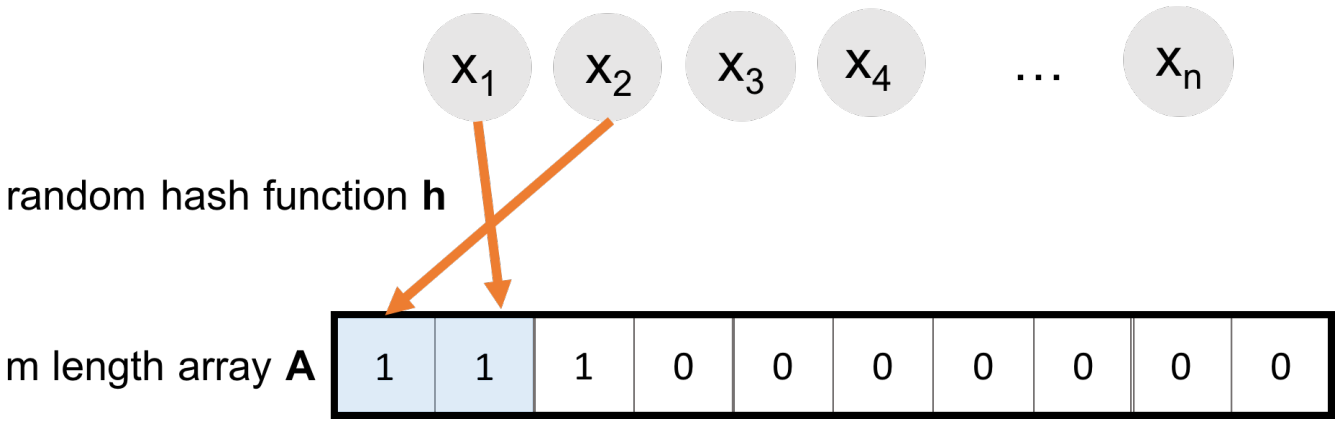
FREQUENT ELEMENTS WITH COUNT-MIN SKETCH

Count-Min Sketch: A random hashing based method closely related to bloom filters.



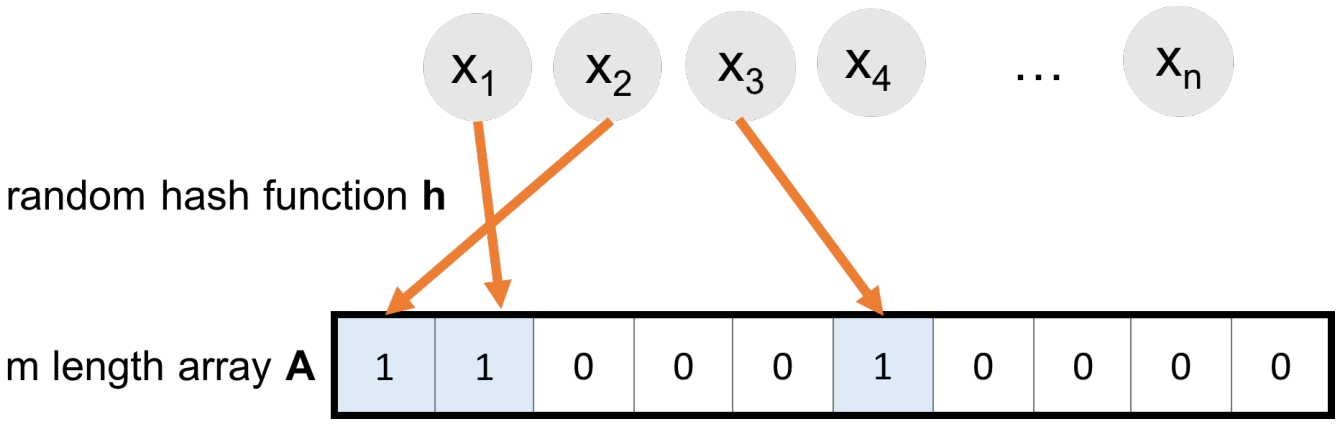
FREQUENT ELEMENTS WITH COUNT-MIN SKETCH

Count-Min Sketch: A random hashing based method closely related to bloom filters.



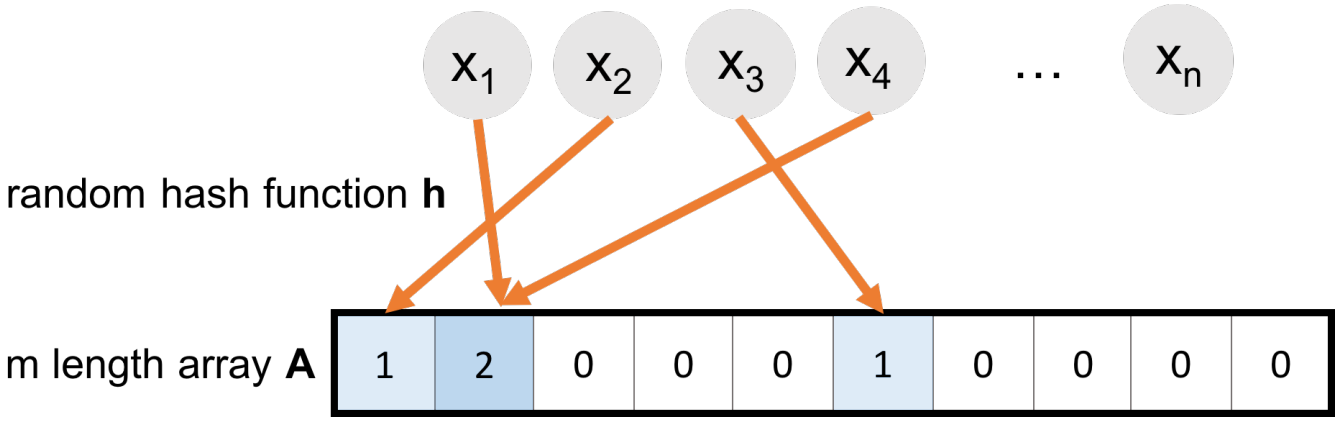
FREQUENT ELEMENTS WITH COUNT-MIN SKETCH

Count-Min Sketch: A random hashing based method closely related to bloom filters.



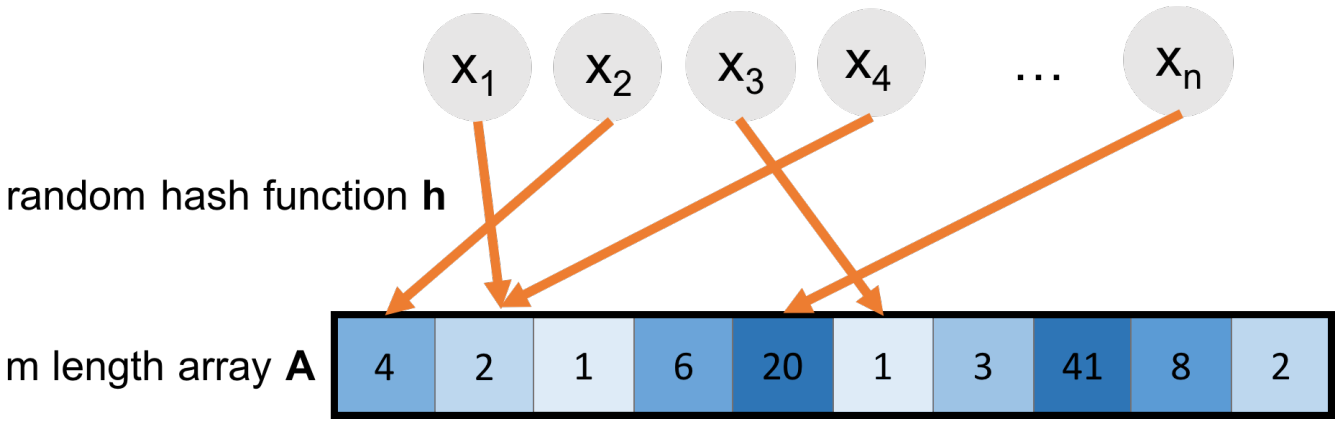
FREQUENT ELEMENTS WITH COUNT-MIN SKETCH

Count-Min Sketch: A random hashing based method closely related to bloom filters.



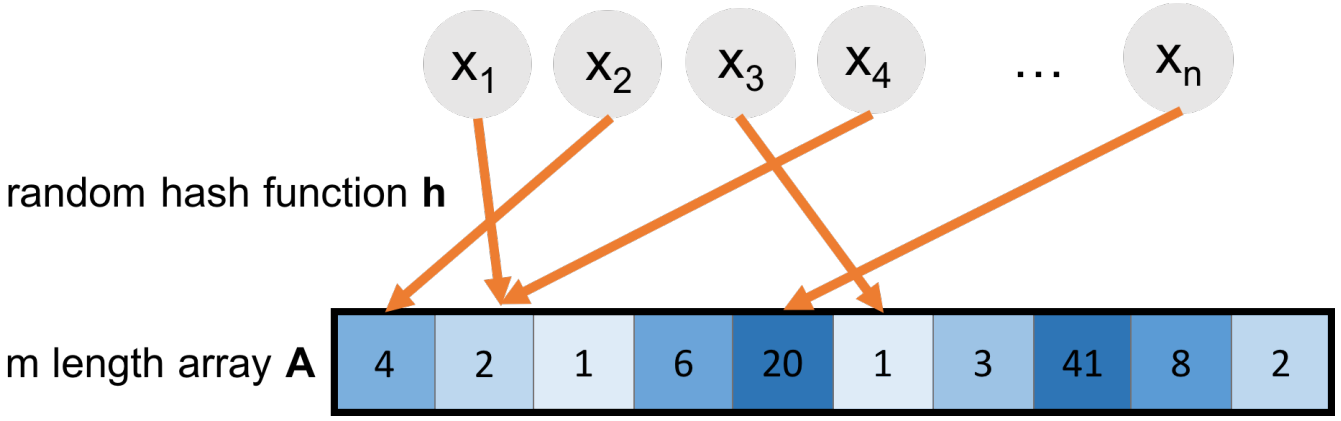
FREQUENT ELEMENTS WITH COUNT-MIN SKETCH

Count-Min Sketch: A random hashing based method closely related to bloom filters.



FREQUENT ELEMENTS WITH COUNT-MIN SKETCH

Count-Min Sketch: A random hashing based method closely related to bloom filters.

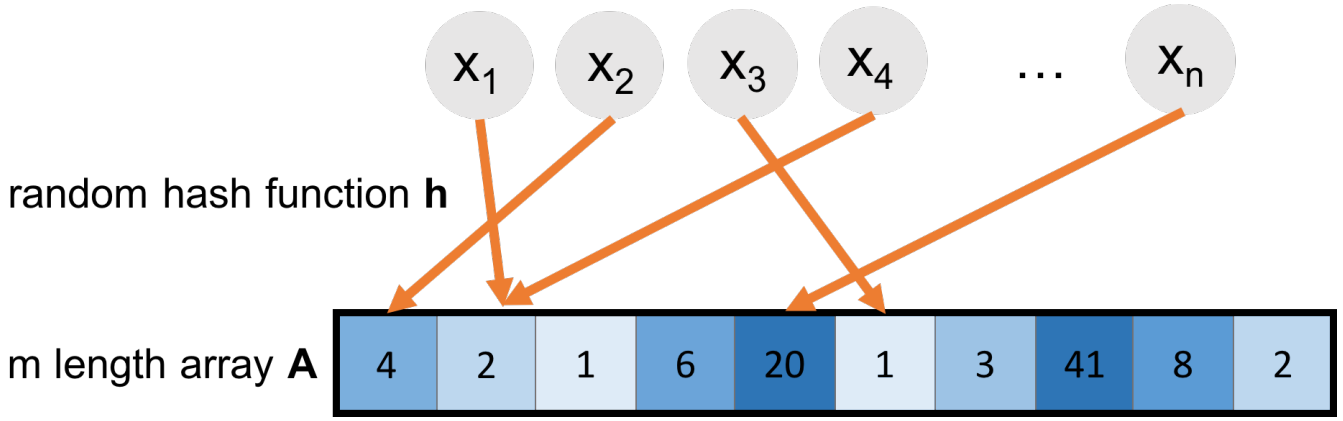


Use $A[\mathbf{h}(x)]$ to estimate $f(x)$, the frequency of x in the stream.

- **Claim:** $A[\mathbf{h}(x)] \geq f(x)$.
- **Claim:** $A[\mathbf{h}(x)] \leq f(x) + 2n/m$ with probability at least $1/2$.

FREQUENT ELEMENTS WITH COUNT-MIN SKETCH

Count-Min Sketch: A random hashing based method closely related to bloom filters.

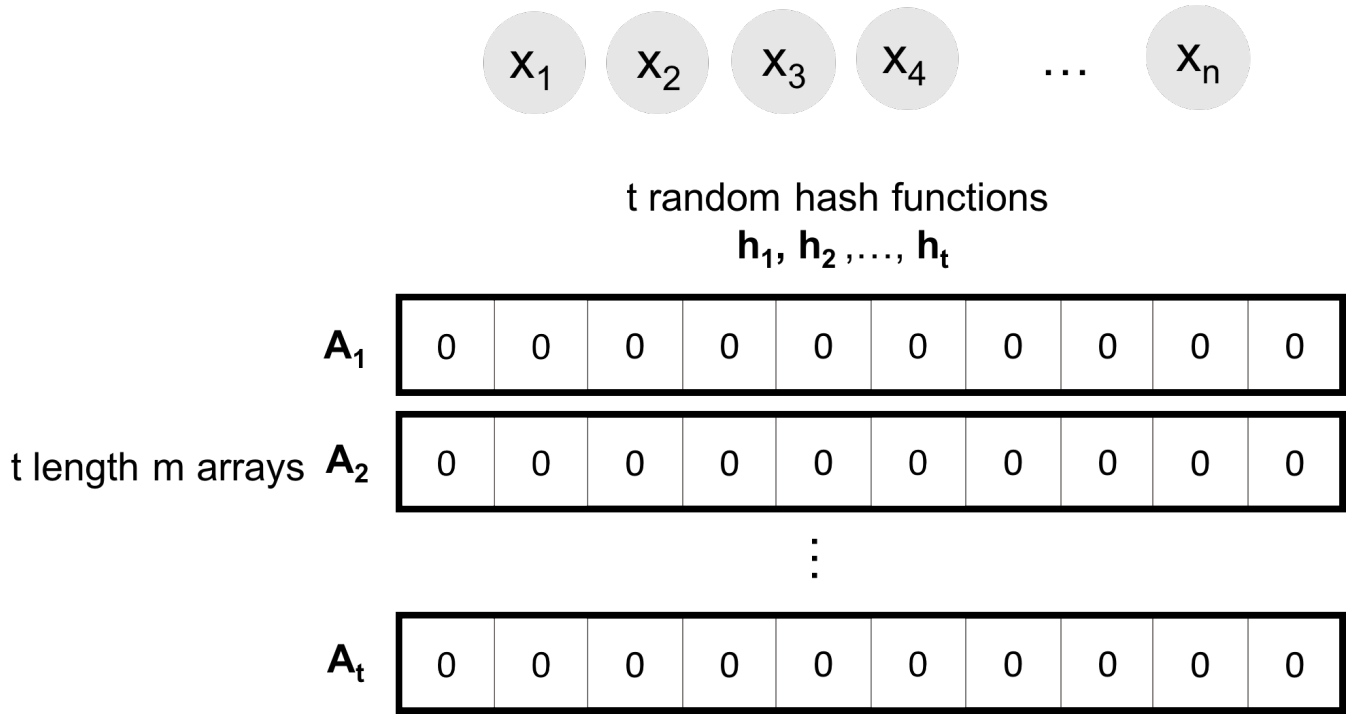


Use $A[\mathbf{h}(x)]$ to estimate $f(x)$, the frequency of x in the stream.

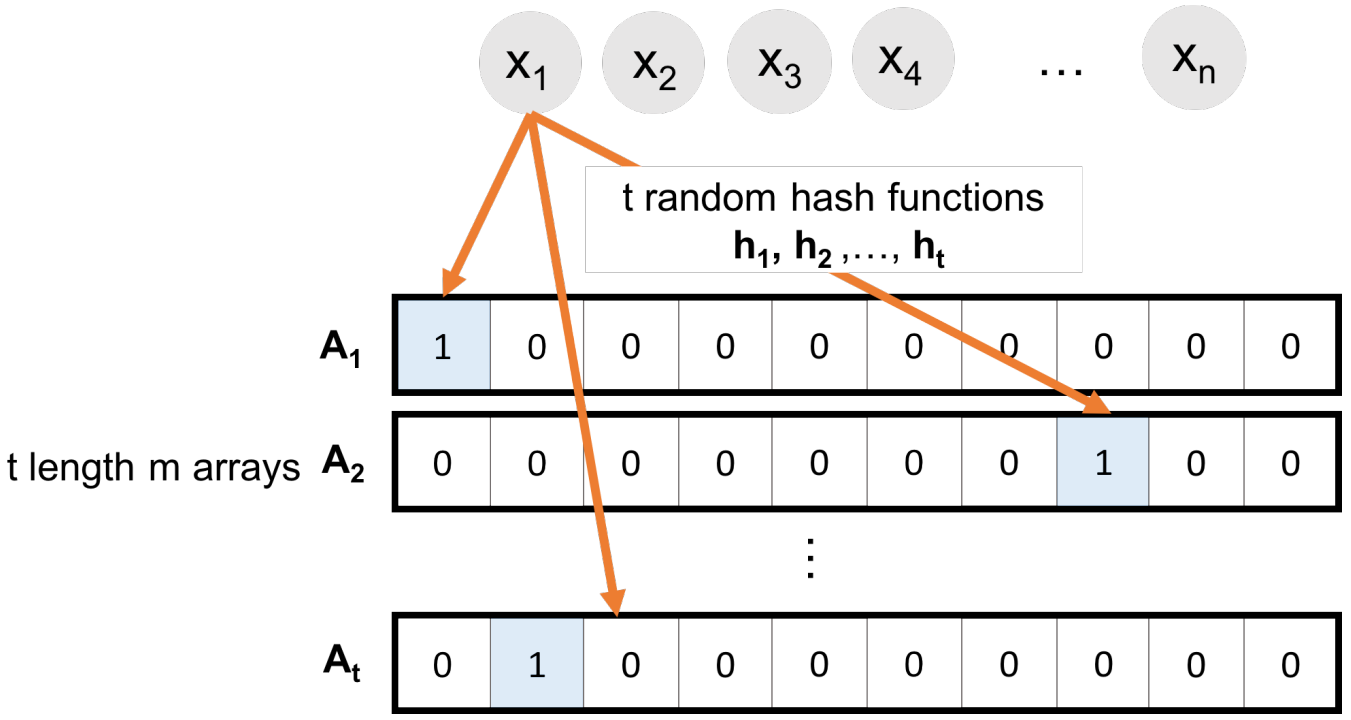
- **Claim:** $A[\mathbf{h}(x)] \geq f(x)$.
- **Claim:** $A[\mathbf{h}(x)] \leq f(x) + 2n/m$ with probability at least $1/2$.

How can we increase this probability to $1 - \delta$ for arbitrary $\delta > 0$?

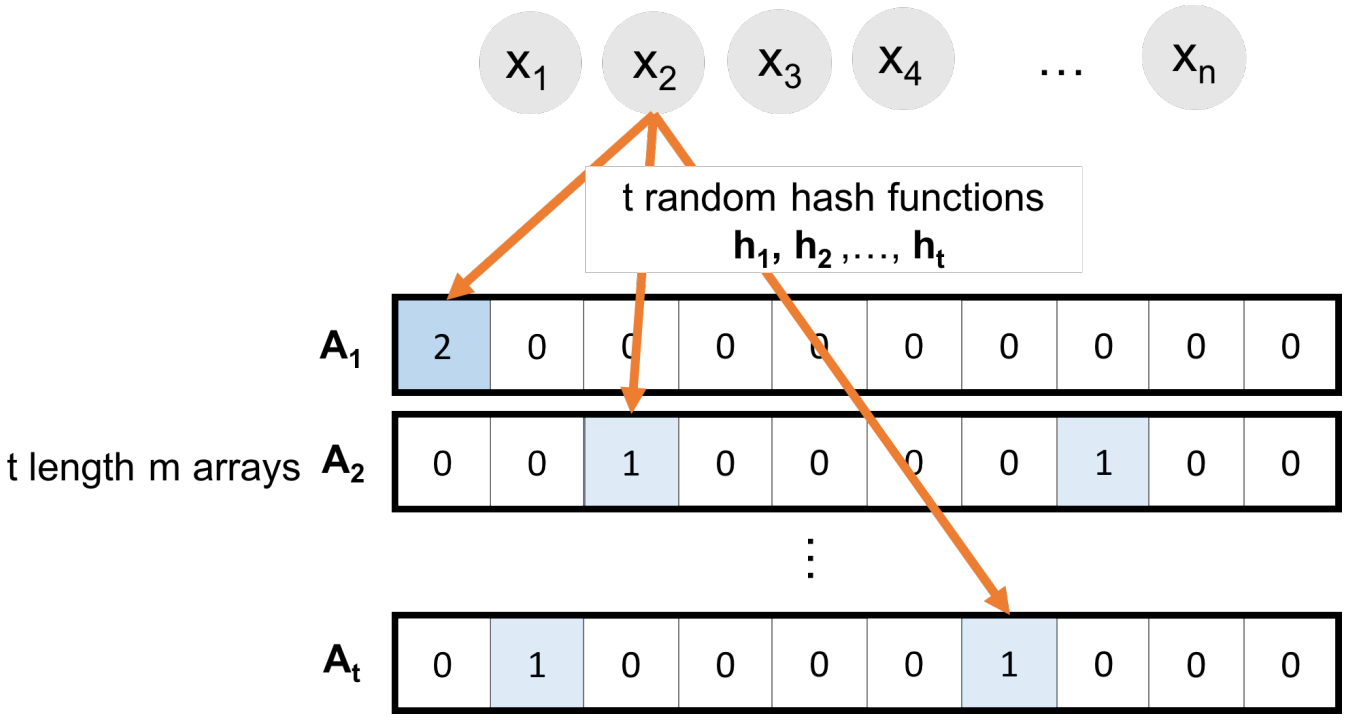
COUNT-MIN SKETCH ACCURACY



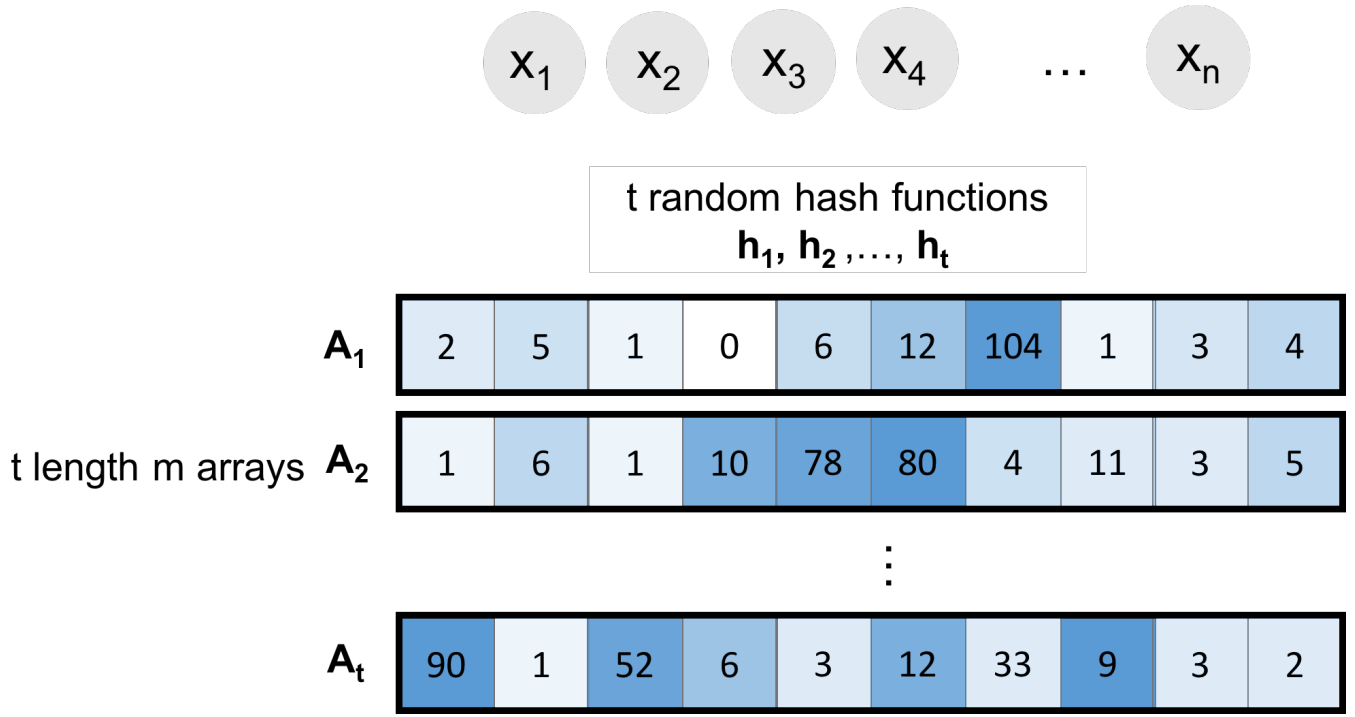
COUNT-MIN SKETCH ACCURACY



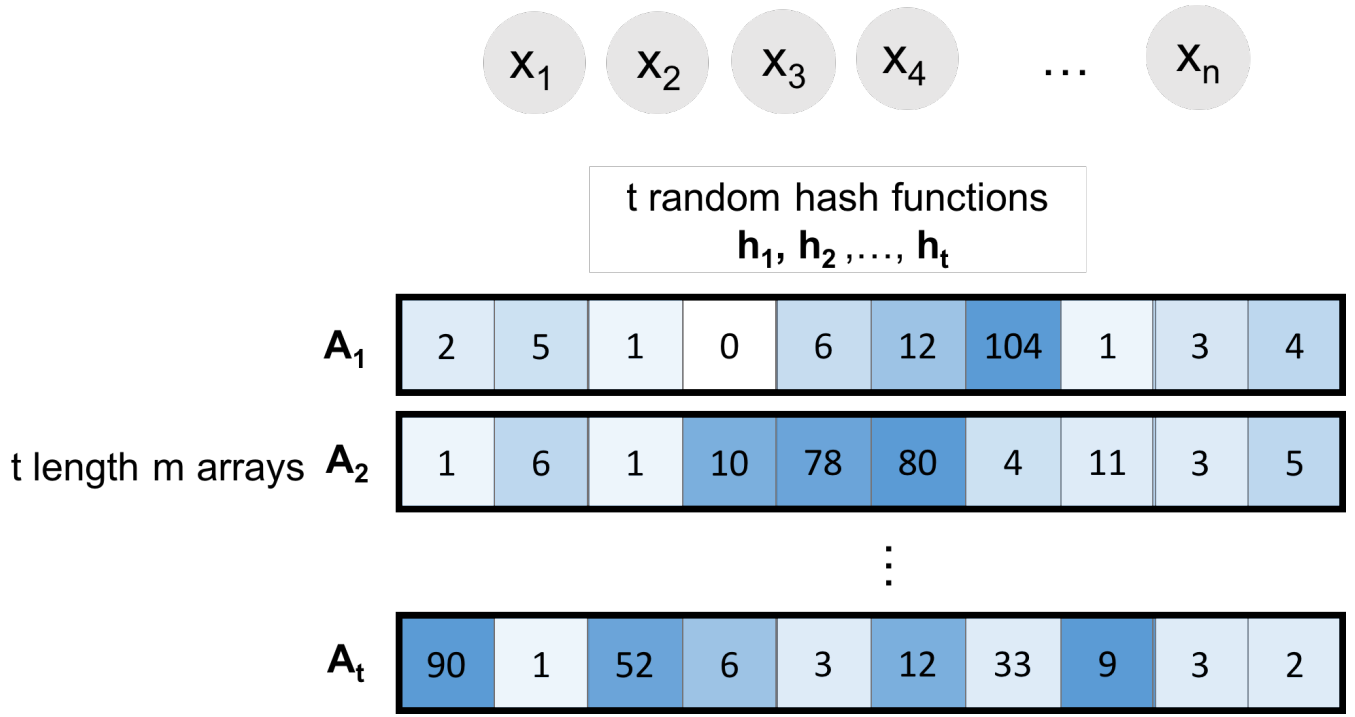
COUNT-MIN SKETCH ACCURACY



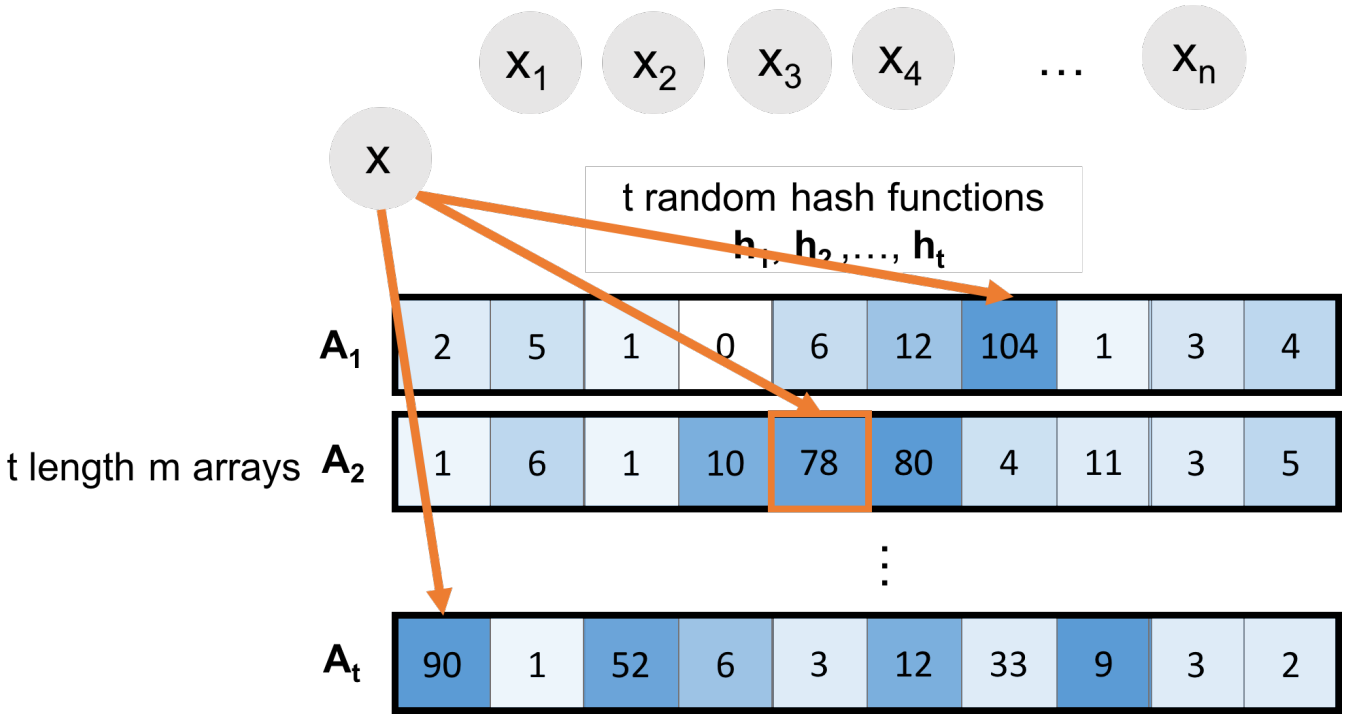
COUNT-MIN SKETCH ACCURACY



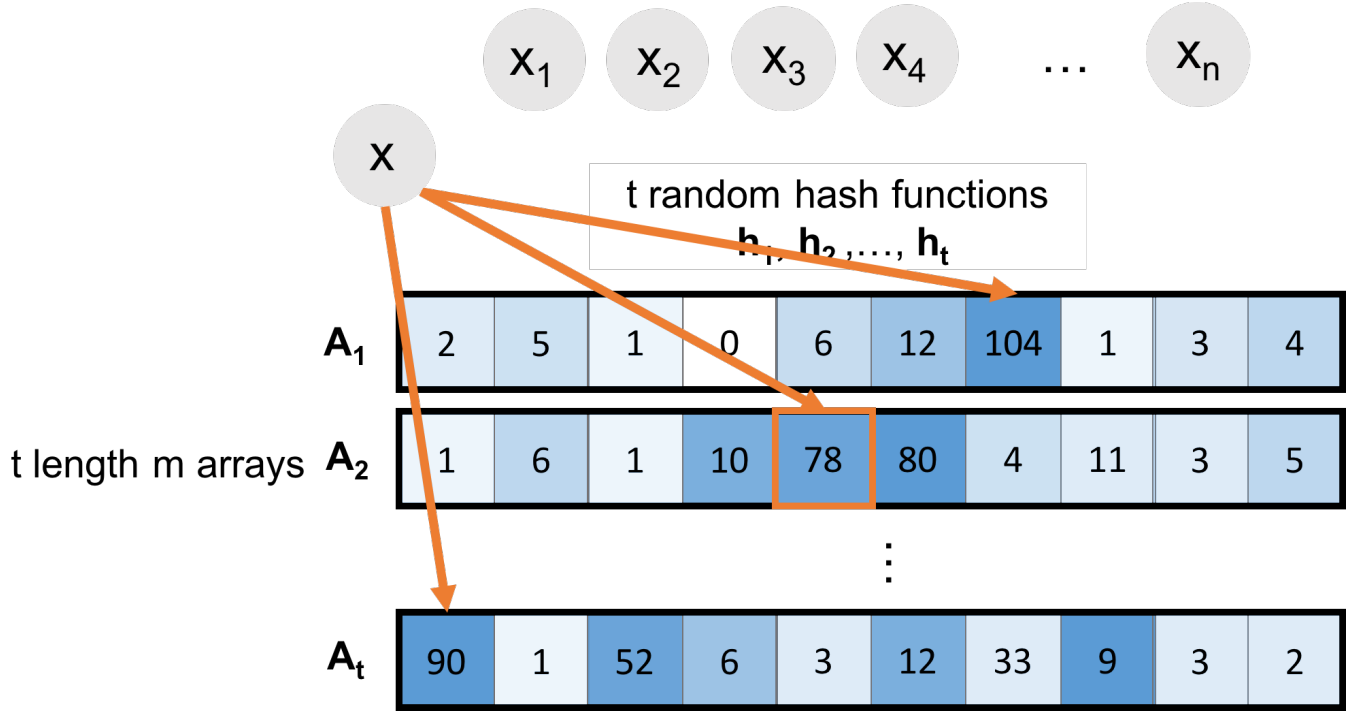
COUNT-MIN SKETCH ACCURACY



COUNT-MIN SKETCH ACCURACY

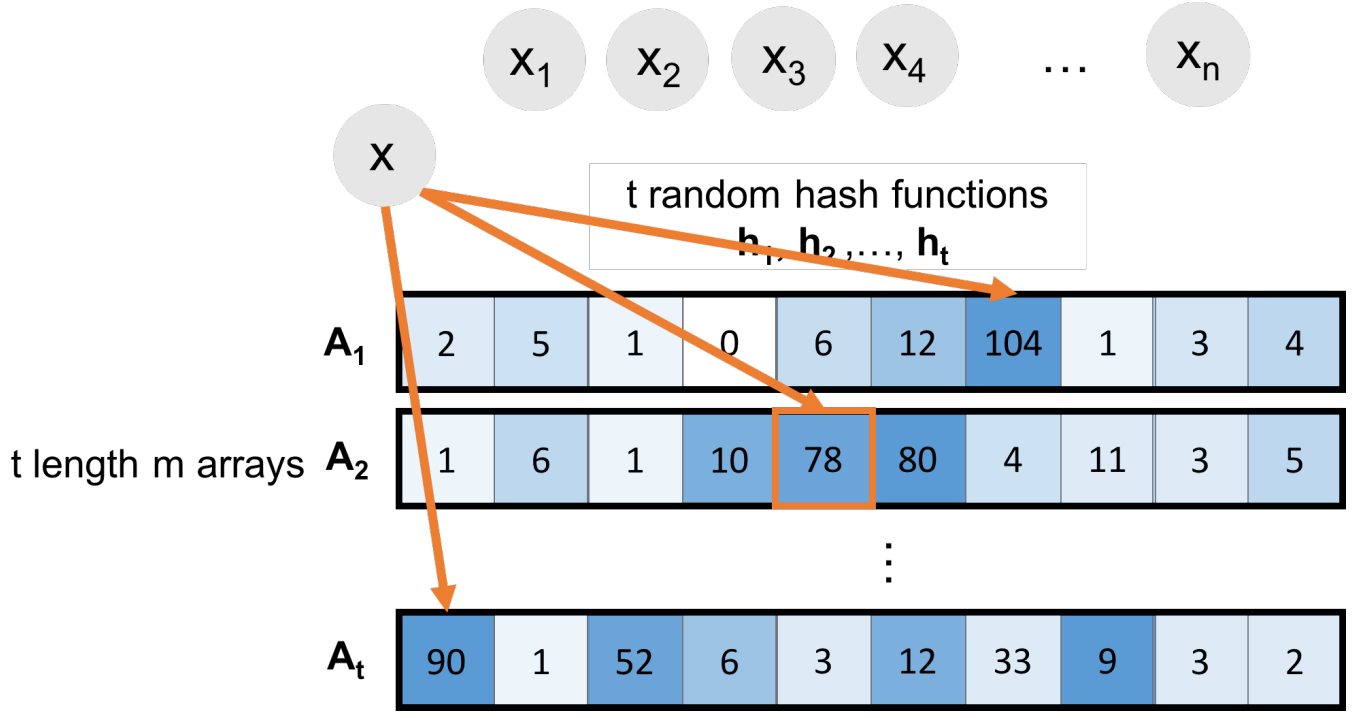


COUNT-MIN SKETCH ACCURACY



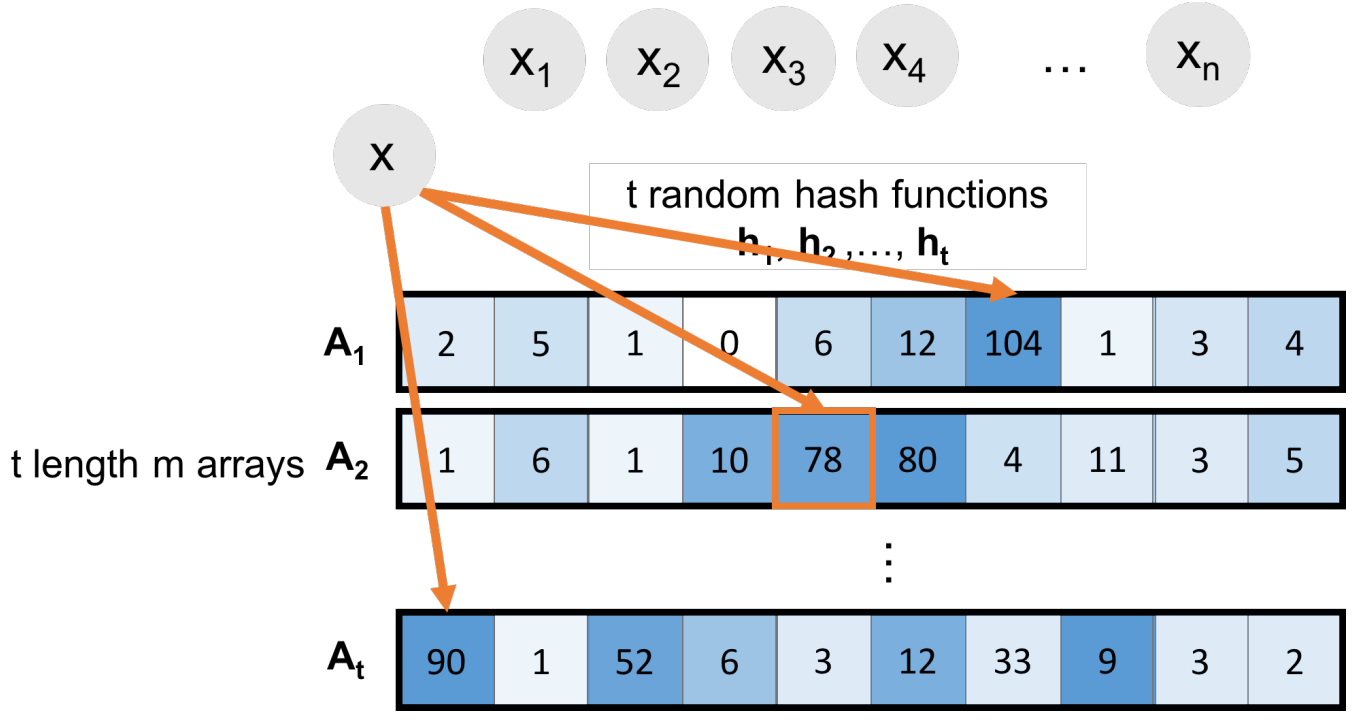
- Estimate $f(x)$ with $\tilde{f}(x) = \min_{i \in [t]} A_i[\mathbf{h}_i(x)]$.

COUNT-MIN SKETCH ACCURACY



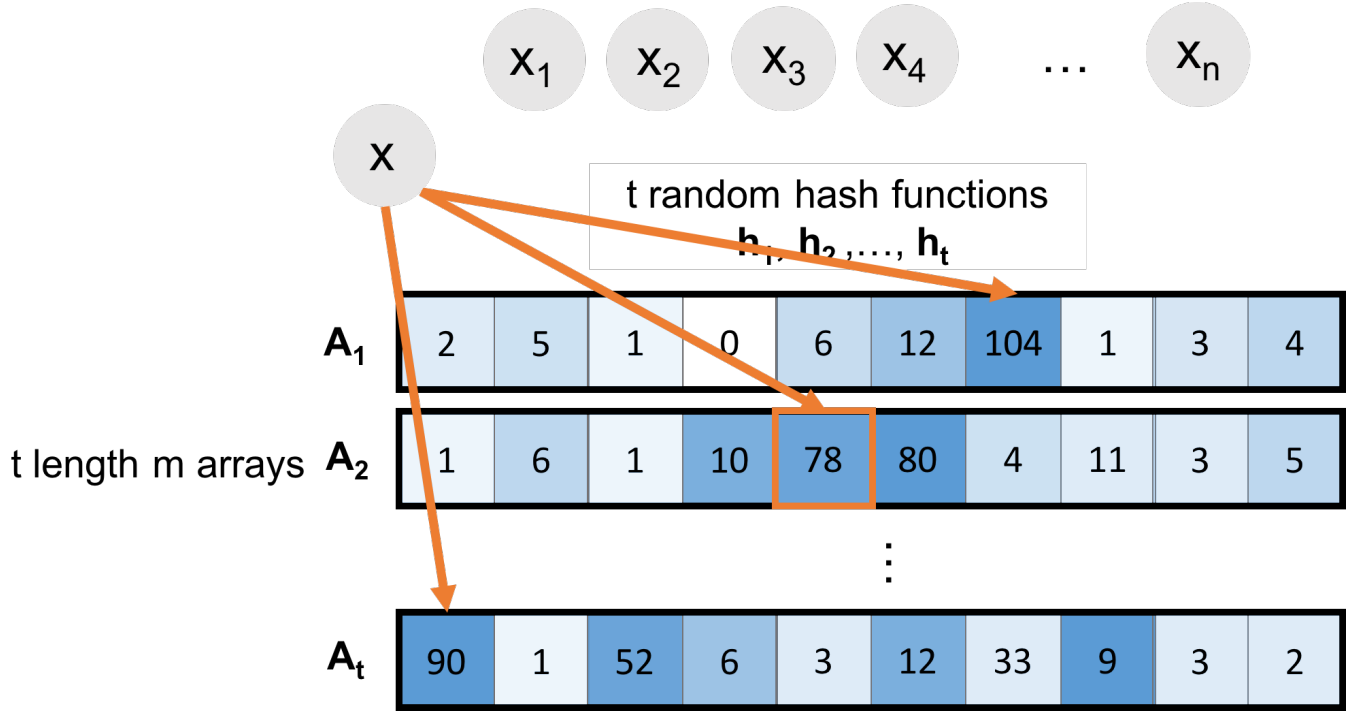
- Estimate $f(x)$ with $\tilde{f}(x) = \min_{i \in [t]} A_i[\mathbf{h}_i(x)]$.
- What is $\Pr[f(x) \leq \tilde{f}(x) \leq f(x) + 2n/m]$?

COUNT-MIN SKETCH ACCURACY



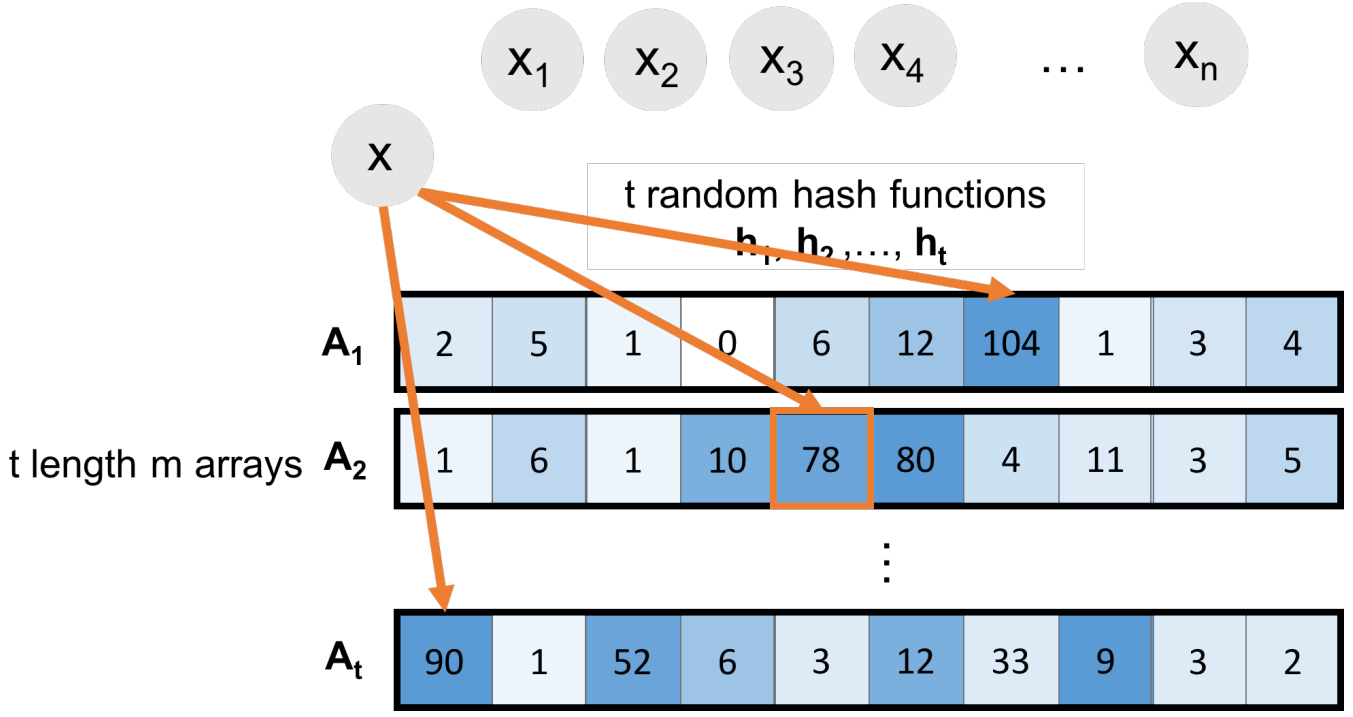
- Estimate $f(x)$ with $\tilde{f}(x) = \min_{i \in [t]} A_i[\mathbf{h}_i(x)]$.
- What is $\Pr[f(x) \leq \tilde{f}(x) \leq f(x) + 2n/m]$? Answer: $\geq 1 - 1/2^t$.

COUNT-MIN SKETCH ACCURACY



- Estimate $f(x)$ with $\tilde{f}(x) = \min_{i \in [t]} A_i[\mathbf{h}_i(x)]$.
- What is $\Pr[f(x) \leq \tilde{f}(x) \leq f(x) + 2n/m]$? Answer: $\geq 1 - 1/2^t$.
- Setting $t = \log(1/\delta)$ ensures probability is at least $1 - \delta$.

COUNT-MIN SKETCH ACCURACY



- Estimate $f(x)$ with $\tilde{f}(x) = \min_{i \in [t]} A_i[\mathbf{h}_i(x)]$.
- What is $\Pr[f(x) \leq \tilde{f}(x) \leq f(x) + 2n/m]$? Answer: $\geq 1 - 1/2^t$.
- Setting $t = \log(1/\delta)$ ensures probability is at least $1 - \delta$.
- Setting $m = 2k/\epsilon$ ensures the error $2n/m$ is $\epsilon n/k$ and this is enough to determine whether we need to output the element.

COUNT-MIN SKETCH

Upshot: Count-min sketch lets us estimate the frequency of every item in a stream up to error $\frac{\epsilon n}{k}$ with probability $\geq 1 - \delta$ in $O\left(\log(1/\delta) \cdot k/\epsilon\right)$ space.

COUNT-MIN SKETCH

Upshot: Count-min sketch lets us estimate the frequency of every item in a stream up to error $\frac{\epsilon n}{k}$ with probability $\geq 1 - \delta$ in $O\left(\log(1/\delta) \cdot k/\epsilon\right)$ space.

- Accurate enough to solve the (ϵ, k)-Frequent elements problem: Can distinguish between items with frequency $\frac{n}{k}$ and those with frequency $< (1 - \epsilon)\frac{n}{k}$.

COUNT-MIN SKETCH

Upshot: Count-min sketch lets us estimate the frequency of every item in a stream up to error $\frac{\epsilon n}{k}$ with probability $\geq 1 - \delta$ in $O\left(\log(1/\delta) \cdot k/\epsilon\right)$ space.

- Accurate enough to solve the (ϵ, k)-Frequent elements problem: Can distinguish between items with frequency $\frac{n}{k}$ and those with frequency $< (1 - \epsilon)\frac{n}{k}$.
- How should we set δ if we want a good estimate for all items at once, with 99% probability?

Upshot: Count-min sketch lets us estimate the frequency of every item in a stream up to error $\frac{\epsilon n}{k}$ with probability $\geq 1 - \delta$ in $O\left(\log(1/\delta) \cdot k/\epsilon\right)$ space.

- Accurate enough to solve the (ϵ, k)-Frequent elements problem: Can distinguish between items with frequency $\frac{n}{k}$ and those with frequency $< (1-\epsilon)\frac{n}{k}$.
- How should we set δ if we want a good estimate for all items at once, with 99% probability? $\delta = 0.01/|U|$ ensures

$$\begin{aligned}&\Pr[\text{there exists } x \in U \text{ with a bad estimate}]\\ &\leq \sum_{x \in U} \Pr[\text{estimate for } x \text{ is bad}] \leq \sum_{x \in U} 0.01/|U| = 0.01\end{aligned}$$

IDENTIFYING FREQUENT ELEMENTS

Count-min sketch gives an accurate frequency estimate for every item in the stream. But how do we identify the frequent items without having to look up the estimated frequency for $x \in U$?

IDENTIFYING FREQUENT ELEMENTS

Count-min sketch gives an accurate frequency estimate for every item in the stream. But how do we identify the frequent items without having to look up the estimated frequency for $x \in U$?

One approach:

- Maintain a set F while processing the stream:
- At step i:
  - Add ith stream element to F if it's estimated frequency is $\geq i/k$ and it isn't already in F.
  - Remove any element from F whose estimated frequency is $< i/k$.
- Store at most k items at once and have all items with frequency $\geq n/k$ stored at the end of the stream.

Questions on Frequent Elements?

HIGH DIMENSIONAL DATA

'Big Data' means not just many data points, but many measurements per data point. I.e., very high dimensional data.

HIGH DIMENSIONAL DATA

'Big Data' means not just many data points, but many measurements per data point. I.e., very high dimensional data.

- Twitter has 321 million active monthly users. Records (tens of) thousands of measurements per user: who they follow, who follows them, when they last visited the site, timestamps for specific interactions, how many tweets they have sent, the text of those tweets, etc.

HIGH DIMENSIONAL DATA

'Big Data' means not just many data points, but many measurements per data point. I.e., very high dimensional data.

- Twitter has 321 million active monthly users. Records (tens of) thousands of measurements per user: who they follow, who follows them, when they last visited the site, timestamps for specific interactions, how many tweets they have sent, the text of those tweets, etc.
- A 3 minute Youtube clip with a resolution of 500×500 pixels at 15 frames/second with 3 color channels is a recording of $\geq$ 2 billion pixel values. Even a 500×500 pixel color image has $750,000$ pixel values.

HIGH DIMENSIONAL DATA

'Big Data' means not just many data points, but many measurements per data point. I.e., very high dimensional data.

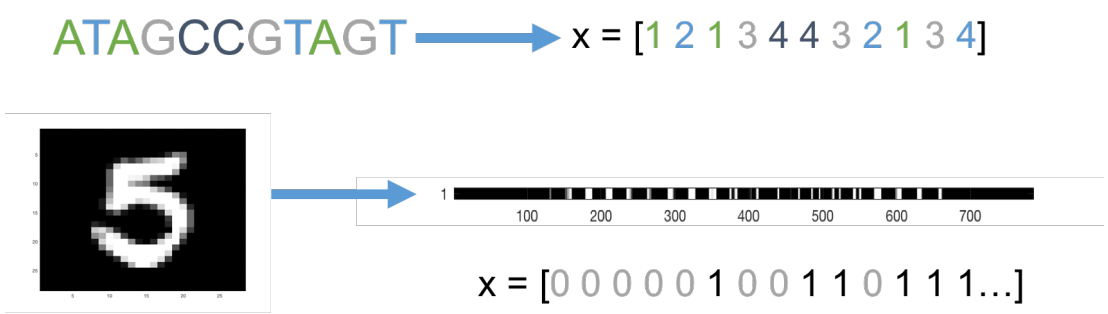
- Twitter has 321 million active monthly users. Records (tens of) thousands of measurements per user: who they follow, who follows them, when they last visited the site, timestamps for specific interactions, how many tweets they have sent, the text of those tweets, etc.
- A 3 minute Youtube clip with a resolution of 500×500 pixels at 15 frames/second with 3 color channels is a recording of $\geq$ 2 billion pixel values. Even a 500×500 pixel color image has $750,000$ pixel values.
- The human genome contains 3 billion+ base pairs. Genetic datasets often contain information on 100s of thousands+ mutations and genetic markers.

DATA AS VECTORS AND MATRICES

In data analysis and machine learning, data points with many attributes are often stored, processed, and interpreted as high dimensional vectors, with real valued entries.

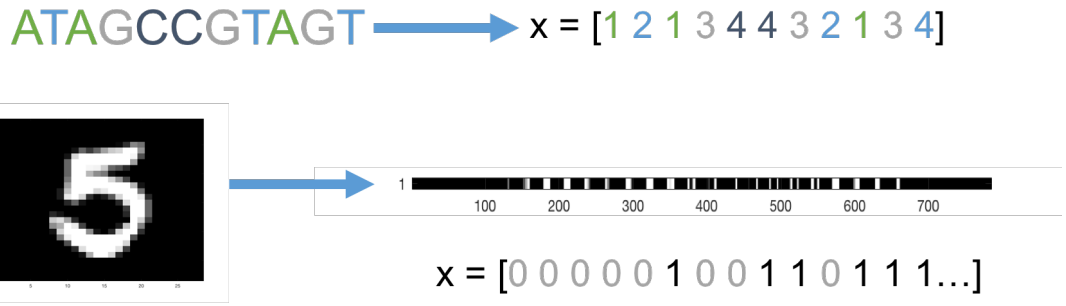
DATA AS VECTORS AND MATRICES

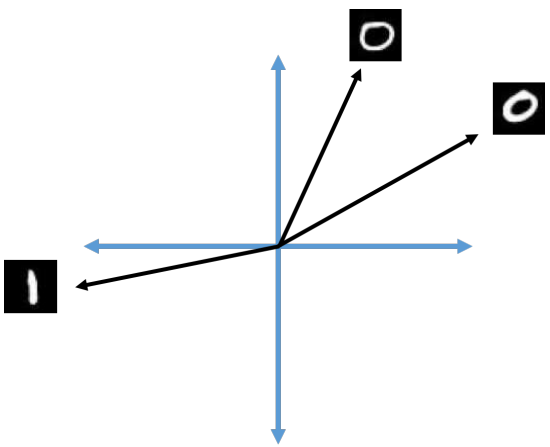
In data analysis and machine learning, data points with many attributes are often stored, processed, and interpreted as high dimensional vectors, with real valued entries.



DATA AS VECTORS AND MATRICES

In data analysis and machine learning, data points with many attributes are often stored, processed, and interpreted as high dimensional vectors, with real valued entries.





Similarities/distances between vectors (e.g., $\langle x, y \rangle$, $\|x - y\|_2$) have meaning for underlying data points.

DATASETS AS VECTORS AND MATRICES

Data points are interpreted as high dimensional vectors, with real valued entries. Data set is interpreted as a matrix.

Data Points: $\vec{x_1}, \vec{x_2}, \ldots, \vec{x_n} \in \mathbb{R}^d$.

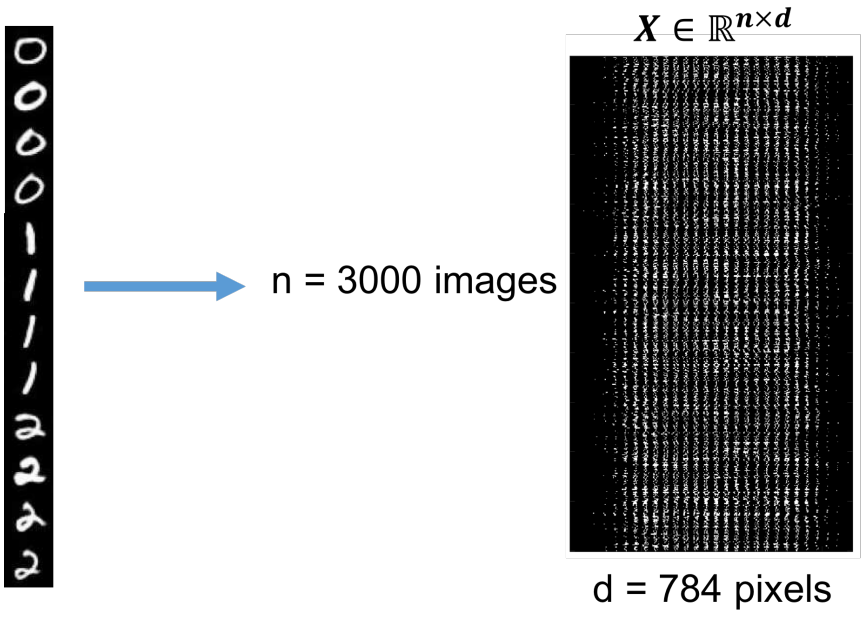
Data Set: $X \in \mathbb{R}^{n \times d}$ with i^{th} rows equal to $\vec{x_i}$.

DATASETS AS VECTORS AND MATRICES

Data points are interpreted as high dimensional vectors, with real valued entries. Data set is interpreted as a matrix.

Data Points: $\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n \in \mathbb{R}^d$.

Data Set: $X \in \mathbb{R}^{n \times d}$ with i^{th} rows equal to $\vec{x}_i$.

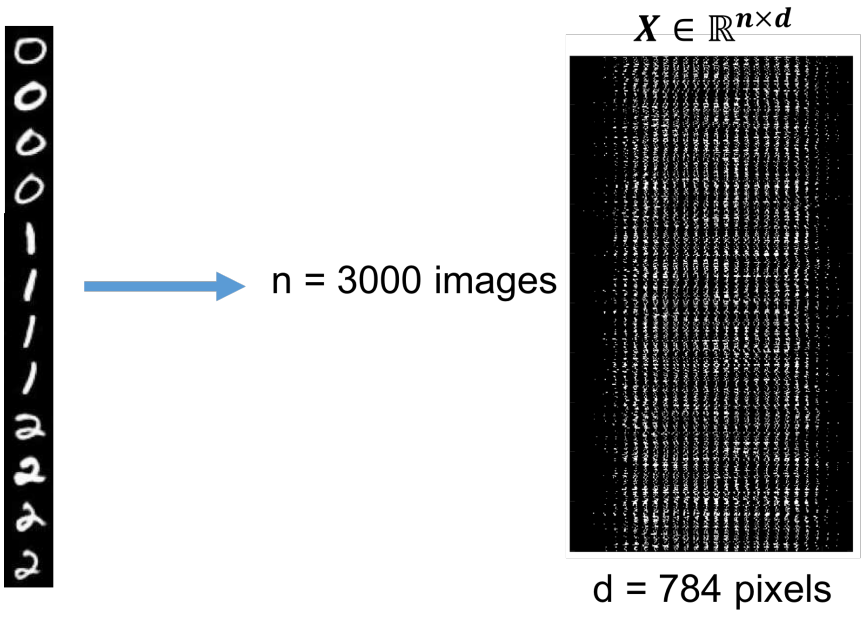


DATASETS AS VECTORS AND MATRICES

Data points are interpreted as high dimensional vectors, with real valued entries. Data set is interpreted as a matrix.

Data Points: $\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n \in \mathbb{R}^d$.

Data Set: $X \in \mathbb{R}^{n \times d}$ with i^{th} rows equal to $\vec{x}_i$.



Many data points $n \implies$ tall. Many dimensions $d \implies$ wide.

DIMENSIONALITY REDUCTION

Dimensionality Reduction: Compress data points so that they lie in many fewer dimensions.

DIMENSIONALITY REDUCTION

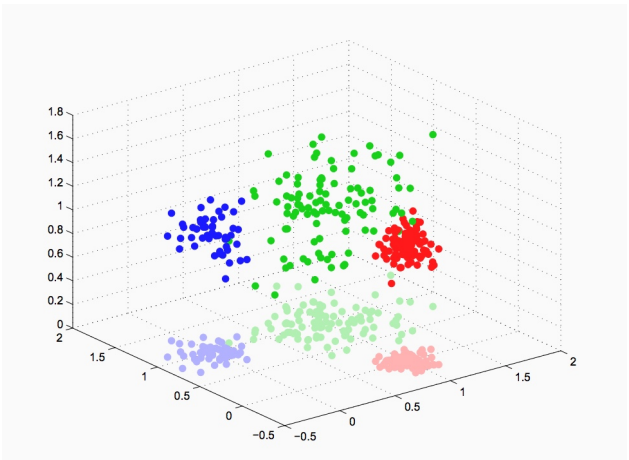
Dimensionality Reduction: Compress data points so that they lie in many fewer dimensions.

$$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d \to \tilde{x}_1, \ldots, \tilde{x}_n \in \mathbb{R}^m \text{ for } m \ll d.$$

Dimensionality Reduction: Compress data points so that they lie in many fewer dimensions.

$$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d \rightarrow \tilde{x}_1, \ldots, \tilde{x}_n \in \mathbb{R}^m \text{ for } m \ll d.$$

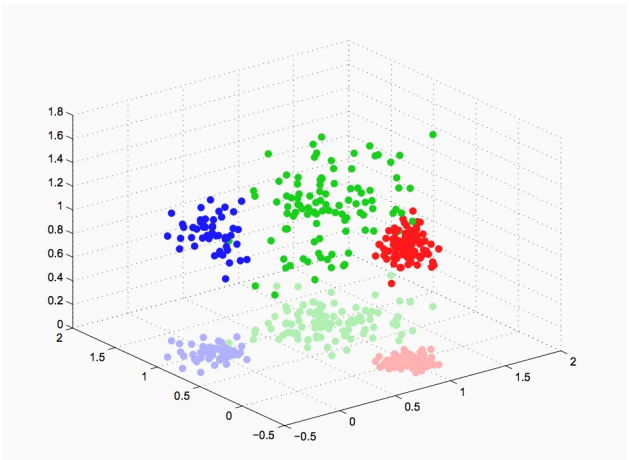
'Lossy compression' that still preserves important information about the relationships between $\vec{x}_1, \ldots, \vec{x}_n$.



Dimensionality Reduction: Compress data points so that they lie in many fewer dimensions.

$$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d \rightarrow \tilde{x}_1, \ldots, \tilde{x}_n \in \mathbb{R}^m \text{ for } m \ll d.$$

'Lossy compression' that still preserves important information about the relationships between $\vec{x}_1, \ldots, \vec{x}_n$.



Generally will not consider directly how well $\tilde{x}_i$ approximates $\vec{x}_i$.

LOW DISTORTION EMBEDDING

Low Distortion Embedding: Given $\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$, distance function D, and error parameter $\epsilon \geq 0$, find $\tilde{x}_1, \ldots, \tilde{x}_n \in \mathbb{R}^m$ (where $m \ll d$) and distance function $\tilde{D}$ such that for all $i, j \in [n]$:

$$(1-\epsilon)D(\vec{x}_i, \vec{x}_j) \leq \tilde{D}(\tilde{x}_i, \tilde{x}_j) \leq (1+\epsilon)D(\vec{x}_i, \vec{x}_j).$$

We'll focus on the case where D and $\tilde{D}$ are Euclidean distances. I.e., the distance between two vectors x and y is defined as

$$\|\vec{x} - \vec{y}\|_2 = \sqrt{\sum_i (\vec{x}(i) - \vec{y}(i))^2}$$

This is related to the Euclidean norm, $\|\vec{z}\|_2 = \sqrt{\sum_{i=1}^n \vec{z}(i)^2}$.

THE JOHNSON-LINDENSTRAUSS LEMMA

Johnson-Lindenstrauss Lemma: For any set of points $\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$ and $\epsilon > 0$ there exists a linear map $\boldsymbol{M} : \mathbb{R}^d \to \mathbb{R}^m$ such that $m = O\left(\frac{\log n}{\epsilon^2}\right)$ and letting $\tilde{x}_i = \boldsymbol{M}\vec{x}_i$:

For all i, j : $(1-\epsilon)\|\vec{x}_i - \vec{x}_j\|_2 \leq \|\tilde{x}_i - \tilde{x}_j\|_2 \leq (1+\epsilon)\|\vec{x}_i - \vec{x}_j\|_2$.

THE JOHNSON-LINDENSTRAUSS LEMMA

Johnson-Lindenstrauss Lemma: For any set of points $\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$ and $\epsilon > 0$ there exists a linear map $\boldsymbol{M} : \mathbb{R}^d \to \mathbb{R}^m$ such that $m = O\left(\frac{\log n}{\epsilon^2}\right)$ and letting $\tilde{x}_i = \boldsymbol{M}\vec{x}_i$:

For all i, j : $(1-\epsilon)\|\vec{x}_i - \vec{x}_j\|_2 \leq \|\tilde{x}_i - \tilde{x}_j\|_2 \leq (1+\epsilon)\|\vec{x}_i - \vec{x}_j\|_2$.

For $d = 1$ trillion, $\epsilon = .05$, and $n = 100,000$, $m \approx 6600$.

THE JOHNSON-LINDENSTRAUSS LEMMA

Johnson-Lindenstrauss Lemma: For any set of points $\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$ and $\epsilon > 0$ there exists a linear map $\boldsymbol{M} : \mathbb{R}^d \to \mathbb{R}^m$ such that $m = O\left(\frac{\log n}{\epsilon^2}\right)$ and letting $\tilde{x}_i = \boldsymbol{M}\vec{x}_i$:

$$\text{For all } i, j: \ (1-\epsilon)\|\vec{x}_i - \vec{x}_j\|_2 \leq \|\tilde{x}_i - \tilde{x}_j\|_2 \leq (1+\epsilon)\|\vec{x}_i - \vec{x}_j\|_2.$$

For $d = 1$ trillion, $\epsilon = .05$, and $n = 100,000$, $m \approx 6600$.

Very surprising! Powerful result with a simple construction: applying a random linear transformation to a set of points preserves distances between all those points with high probability.