COMPSCI 514: Algorithms for Data Science

Cameron Musco

University of Massachusetts Amherst. Fall 2022.

Lecture 11

Logistics

4.1 tables of size $O(n)$

tables of size $n = O(k)$

- Problem Set 2 is due on Friday at 11:59pm.
- Midterm is in class next Thursday, 10/20.
- I have posted a study guide and practice questions on the course schedule.

$\frac{n_k}{m}$ $\quad n - \sum_{i=1}^{k} f_i$

Summary

Last Class:

- Introduced the k-frequent elements problem – identify all elements of a stream of n elements that occur $\geq n/k$ times.
- Saw how to solve approximately in $O(k \log n/\epsilon)$ space using the Count-min sketch algorithm.
- Simple analysis based on Markov's inequality and repeated random hashing.

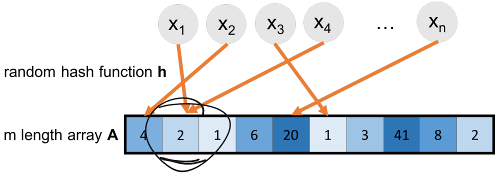
This Class:

- Recap and finish up Count-min sketch
- Randomized methods for dimensionality reduction.
- The Johnson-Lindenstrauss Lemma.

Count-Min Sketch

(ε, k) - Freq. Items Problem

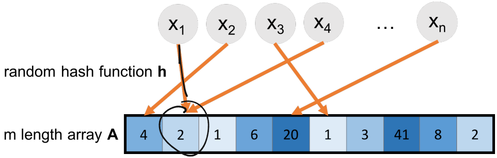
Goal: Return all items in a stream of n elements with frequency at least n/k. Don't return any with frequency $\leq (1-\epsilon)\cdot\frac{n}{k}$.



$F(x_1)$

Count-Min Sketch

Goal: Return all items in a stream of n elements with frequency at least n/k. Don't return any with frequency $\leq (1-\epsilon)\cdot\frac{n}{k}$.



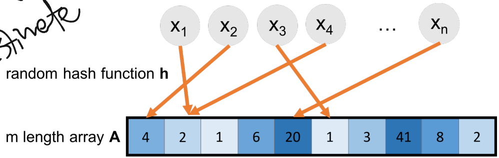
- $A[\mathbf{h}(x)] = f(x) + \sum_{y\neq x:\mathbf{h}(y)=\mathbf{h}(x)} f(y)$ where $\mathbb{E}[\sum_{y\neq x:\mathbf{h}(y)=\mathbf{h}(x)} f(y)] \leq \frac{n}{m}$.

Count-Min Sketch

Goal: Return all items in a stream of n elements with frequency at least n/k. Don't return any with frequency $\leq (1-\epsilon) \cdot \frac{n}{k}$.

$f(x)$ = true freq. of x

$\tilde{f}(x)$ is an estimate



- $A[\mathbf{h}(x)] = f(x) + \sum_{y \neq x : \mathbf{h}(y) = \mathbf{h}(x)} f(y)$ where $\mathbb{E}[\sum_{y \neq x : \mathbf{h}(y) = \mathbf{h}(x)} f(y)] \leq \frac{n}{m}$.

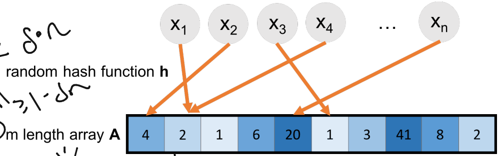
  $\Pr\left(\tilde{f}(x) \geq f(x) + \frac{2n}{m}\right) \leq \frac{1}{2}$

- If we let $\tilde{f}(x)$ be the minimum of $t = \log_2(1/\delta)$ estimates, $f(x) \leq \tilde{f}(x) \leq f(x) + \frac{2n}{m}$ with probability at least $1 - \delta$.

  $1 - \frac{1}{2^t} = 1 - \delta$

Count-Min Sketch

Goal: Return all items in a stream of n elements with frequency at least n/k. Don't return any with frequency $\leq (1-\epsilon)\cdot\frac{n}{k}$.



- $A[\mathbf{h}(x)] = f(x) + \sum_{y \neq x : \mathbf{h}(y)=\mathbf{h}(x)} f(y)$ where $\mathbb{E}[\sum_{y \neq x : \mathbf{h}(y)=\mathbf{h}(x)} f(y)] \leq \frac{n}{m}$.
- If we let $\tilde{f}(x)$ be the minimum of $t = \log_2(1/\delta)$ estimates, $f(x) \leq \tilde{f}(x) \leq f(x) + \frac{2n}{m}$ with probability at least $1-\delta$.
- Setting $m = O(k/\epsilon)$, $\delta = O(\delta'/n)$, and applying a union bound, we have a good estimate for all $f(x)$ with probability at least $1-\delta'$.

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 store/look up the estimated frequency for all elements in the stream?

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 store/look up the estimated frequency for all elements in the stream?

One approach:

- When a new item comes in at step i, check if its estimated frequency is $\geq i/k$ and store it if so.
- At step i remove any stored items whose estimated frequency drops below i/k.
- Store at most $O(k)$ items at once and have all items with frequency $\geq n/k$ stored at the end of the stream, no items with frequency $< (1-\epsilon) \cdot \frac{n}{k}$.

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 ≥ 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.

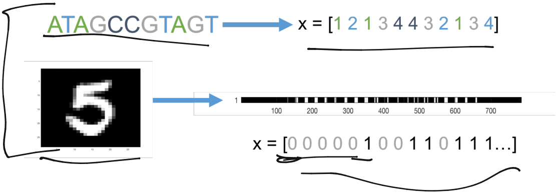
- Twitter has 321 million active monthly users. Records (tens of) thousands of measurements per user: who they follow, who follows them, when they last visited the site, timestamps for specific interactions, how many tweets they have sent, the text of those tweets, etc.
- A 3 minute Youtube clip with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

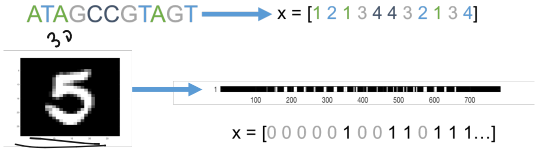
Data as Vectors and Matrices

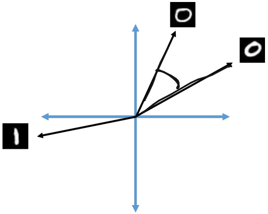
In data analysis and machine learning, data points with many attributes are often stored, processed, and interpreted as high dimensional vectors, with real valued entries.



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

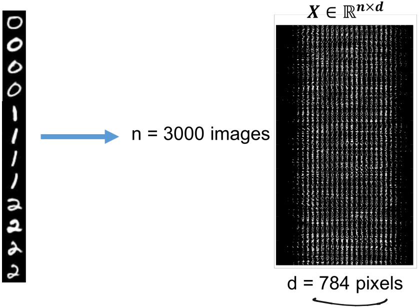
Data Set: $X \in \mathbb{R}^{n \times d}$ with i^{th} row equal to $\vec{x}_i$.

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} row equal to $\vec{x}_i$.

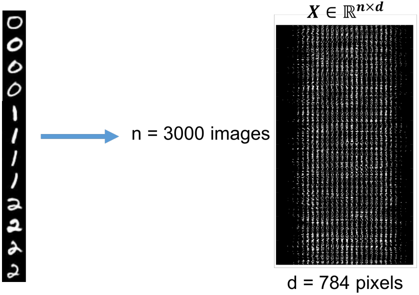


Datasets as Vectors and Matrices

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} row 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.

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

 → x = [0 0 0 0 0 1 0 0 1 1 0 1 1 1...] → $\tilde{x}$ = [-5.5 4 3.2 -1]

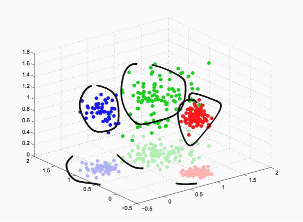
Dimensionality Reduction

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

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

x = [0 0 0 0 0 1 0 0 1 1 0 1 1 1...] → $\tilde{x}$ = [-5.5 4 3.2 -1]

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



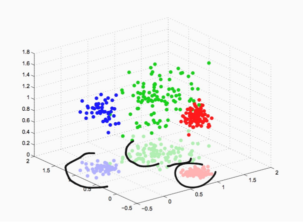
Dimensionality Reduction

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

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

x = [0 0 0 0 0 1 0 0 1 1 0 1 1 1...] → $\tilde{x}$ = [-5.5 4 3.2 -1]

'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 $\vec{x}_i$.

Dimensionality Reduction

Dimensionality reduction is one of the most important techniques in data science. What methods have you heard of?

PCA → LSA

T-SNE

Auto encoders, Nueral networks

self-organizing maps

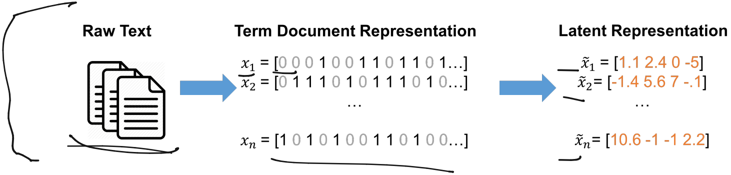
clustering

JPEG, mp3

Dimensionality Reduction

Dimensionality reduction is one of the most important techniques in data science. What methods have you heard of?

- Principal component analysis
- Latent semantic analysis (LSA)

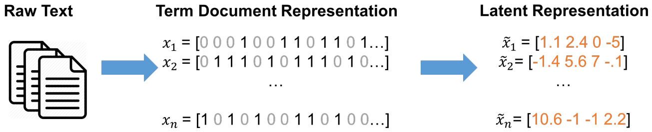


- Linear discriminant analysis
- Autoencoders

Dimensionality Reduction

Dimensionality reduction is one of the most important techniques in data science. What methods have you heard of?

- Principal component analysis
- Latent semantic analysis (LSA)



- Linear discriminant analysis
- Autoencoders

Compressing data makes it more efficient to work with. May also remove extraneous information/noise.

Embeddings for Euclidean Space

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

$$(1-\epsilon)\|\vec{x}_i - \vec{x}_j\|_2 \leq \|\tilde{x}_i - \tilde{x}_j\|_2 \leq (1+\epsilon)\|\vec{x}_i - \vec{x}_j\|_2.$$

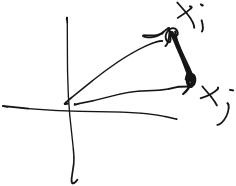
m-dimension   d-dimension

Embeddings for Euclidean Space

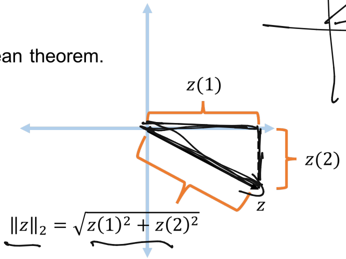
Euclidean Low Distortion Embedding: Given $\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$ and error parameter $\epsilon \geq 0$, find $\tilde{x}_1, \ldots, \tilde{x}_n \in \mathbb{R}^m$ (where $m \ll d$) such that for all $i, j \in [n]$:

$$(1-\epsilon)\|\vec{x}_i - \vec{x}_j\|_2 \leq \|\tilde{x}_i - \tilde{x}_j\|_2 \leq (1+\epsilon)\|\vec{x}_i - \vec{x}_j\|_2.$$

Recall that for $\vec{z} \in \mathbb{R}^n$, $\|\vec{z}\|_2 = \sqrt{\sum_{i=1}^n \vec{z}(i)^2}$.



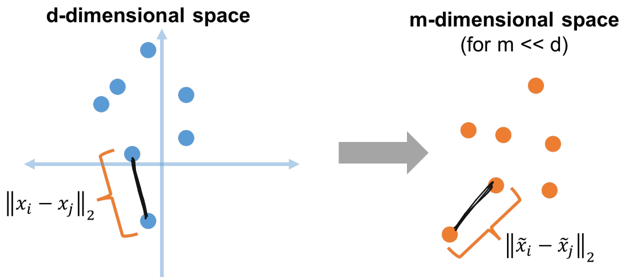
Pythagorean theorem.



Embeddings for Euclidean Space

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

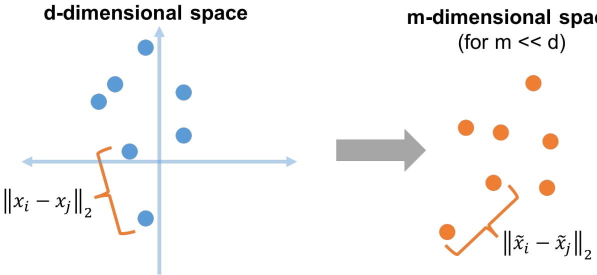
$$(1-\epsilon)\|\vec{x}_i - \vec{x}_j\|_2 \leq \|\tilde{x}_i - \tilde{x}_j\|_2 \leq (1+\epsilon)\|\vec{x}_i - \vec{x}_j\|_2.$$



Embeddings for Euclidean Space

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

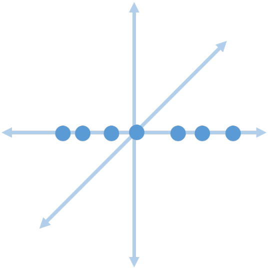
$$(1-\epsilon)\|\vec{x}_i - \vec{x}_j\|_2 \leq \|\tilde{x}_i - \tilde{x}_j\|_2 \leq (1+\epsilon)\|\vec{x}_i - \vec{x}_j\|_2.$$



Can use $\tilde{x}_1, \ldots, \tilde{x}_n$ in place of $\vec{x}_1, \ldots, \vec{x}_n$ in clustering, SVM, linear classification, near neighbor search, etc.

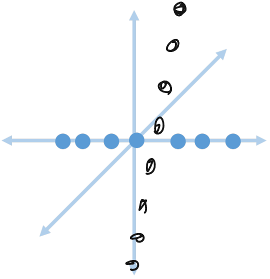
Embedding with Assumptions

A very easy case: Assume that $\vec{x}_1, \ldots, \vec{x}_n$ all lie on the 1^{st} axis in $\mathbb{R}^d$.



Embedding with Assumptions

A very easy case: Assume that $\vec{x}_1, \ldots, \vec{x}_n$ all lie on the 1^{st} axis in $\mathbb{R}^d$.

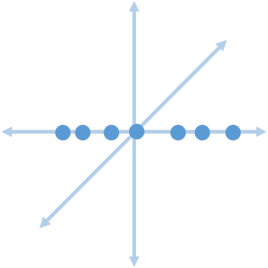


Set $m = 1$ and $\tilde{x}_i = [\vec{x}_i(1)]$ (i.e., $\tilde{x}_i$ contains just a single number).

- $\|\tilde{x}_i - \tilde{x}_j\|_2 = \sqrt{[\vec{x}_i(1) - \vec{x}_j(1)]^2} = |\vec{x}_i(1) - \vec{x}_j(1)| = \|\vec{x}_i - \vec{x}_j\|_2$.

Embedding with Assumptions

A very easy case: Assume that $\vec{x}_1, \ldots, \vec{x}_n$ all lie on the 1^{st} axis in $\mathbb{R}^d$.

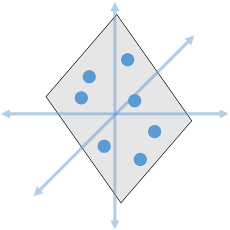


Set $m = 1$ and $\tilde{x}_i = [\vec{x}_i(1)]$ (i.e., $\tilde{x}_i$ contains just a single number).

- $\|\tilde{x}_i - \tilde{x}_j\|_2 = \sqrt{[\vec{x}_i(1) - \vec{x}_j(1)]^2} = |\vec{x}_i(1) - \vec{x}_j(1)| = \|\vec{x}_i - \vec{x}_j\|_2$.
- An embedding with no distortion from any d into $m = 1$.

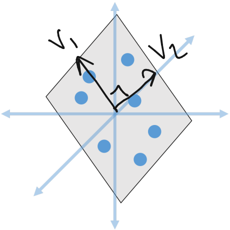
Embedding with Assumptions

Assume that $\vec{x}_1, \ldots \vec{x}_n$ lie in any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.



Embedding with Assumptions

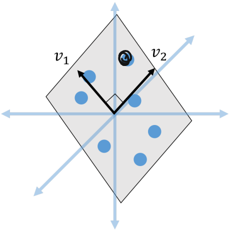
Assume that $\vec{x}_1, \ldots \vec{x}_n$ lie in any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.



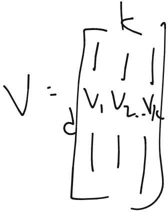
- Let $\vec{v}_1, \vec{v}_2, \ldots \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and let $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns.

Embedding with Assumptions

Assume that $\vec{x}_1, \ldots \vec{x}_n$ lie in any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.



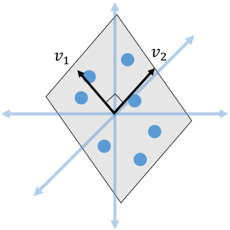
$c_1 \vec{v}_1 + c_2 \vec{v}_2$



- Let $\vec{v}_1, \vec{v}_2, \ldots \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and let $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns.

Embedding with Assumptions

Assume that $\vec{x}_1, \ldots \vec{x}_n$ lie in any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.

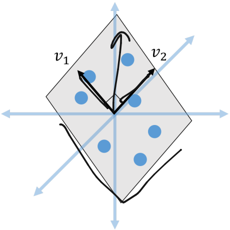


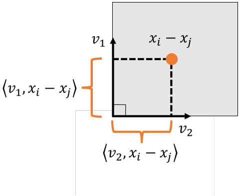
- Let $\vec{v}_1, \vec{v}_2, \ldots \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and let $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns.
- For all i, j we have $\vec{x}_i - \vec{x}_j \in \mathcal{V}$ and (a good exercise!):

$$\|\vec{x}_i - \vec{x}_j\|_2 = \sqrt{\sum_{\ell=1}^{k} \langle v_\ell, \vec{x}_i - \vec{x}_j \rangle^2} \qquad .$$

Embedding with Assumptions

Assume that $\vec{x}_1, \ldots \vec{x}_n$ lie in any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.



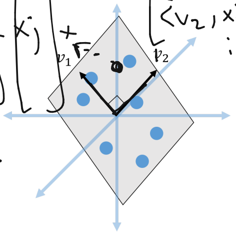


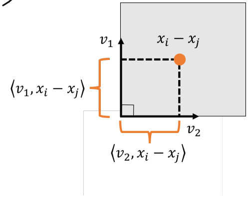
- Let $\vec{v}_1, \vec{v}_2, \ldots \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and let $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns.
- For all i, j we have $\vec{x}_i - \vec{x}_j \in \mathcal{V}$ and (a good exercise!):

$$\|\vec{x}_i - \vec{x}_j\|_2 = \sqrt{\sum_{\ell=1}^{k} \langle v_\ell, \vec{x}_i - \vec{x}_j \rangle^2}.$$

Embedding with Assumptions

Assume that $\vec{x}_1, \ldots \vec{x}_n$ lie in any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.





- Let $\vec{v}_1, \vec{v}_2, \ldots \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and let $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns.
- For all i, j we have $\vec{x}_i - \vec{x}_j \in \mathcal{V}$ and (a good exercise!):

$$\|\vec{x}_i - \vec{x}_j\|_2 = \sqrt{\sum_{\ell=1}^{k} \langle \vec{v}_\ell, \vec{x}_i - \vec{x}_j \rangle^2} = \|\mathbf{V}^T(\vec{x}_i - \vec{x}_j)\|_2.$$

Embedding with Assumptions

Assume that $\vec{x}_1, \ldots \vec{x}_n$ lie in any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.

- Let $\vec{v}_1, \vec{v}_2, \ldots \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and let $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns.
- For all i, j we have $\vec{x}_i - \vec{x}_j \in \mathcal{V}$ and (a good exercise!):

$$\|\vec{x}_i - \vec{x}_j\|_2 = \sqrt{\sum_{\ell=1}^{k} \langle v_\ell, \vec{x}_i - \vec{x}_j \rangle^2} = \|\mathbf{V}^T(\vec{x}_i - \vec{x}_j)\|_2.$$

Embedding with Assumptions

Assume that $\vec{x}_1, \ldots \vec{x}_n$ lie in any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.

- Let $\vec{v}_1, \vec{v}_2, \ldots \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and let $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns.
- For all i, j we have $\vec{x}_i - \vec{x}_j \in \mathcal{V}$ and (a good exercise!):

$$\|\vec{x}_i - \vec{x}_j\|_2 = \sqrt{\sum_{\ell=1}^{k} \langle v_\ell, \vec{x}_i - \vec{x}_j \rangle^2} = \|\mathbf{V}^T(\vec{x}_i - \vec{x}_j)\|_2.$$

- If we set $\tilde{x}_i \in \mathbb{R}^k$ to $\tilde{x}_i = \mathbf{V}^T \vec{x}_i$ we have:

$$\|\tilde{x}_i - \tilde{x}_j\|_2 = \|\mathbf{V}^T \vec{x}_i - \mathbf{V}^T \vec{x}_j\|_2$$

Embedding with Assumptions

Assume that $\vec{x}_1, \ldots \vec{x}_n$ lie in any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.

- Let $\vec{v}_1, \vec{v}_2, \ldots \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and let $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns.
- For all i, j we have $\vec{x}_i - \vec{x}_j \in \mathcal{V}$ and (a good exercise!):

$$\|\vec{x}_i - \vec{x}_j\|_2 = \sqrt{\sum_{\ell=1}^{k} \langle v_\ell, \vec{x}_i - \vec{x}_j \rangle^2} = \|\mathbf{V}^T(\vec{x}_i - \vec{x}_j)\|_2.$$

- If we set $\tilde{x}_i \in \mathbb{R}^k$ to $\tilde{x}_i = \mathbf{V}^T \vec{x}_i$ we have:

$$\|\tilde{x}_i - \tilde{x}_j\|_2 = \|\mathbf{V}^T\vec{x}_i - \mathbf{V}^T\vec{x}_j\|_2 = \|\mathbf{V}^T(\vec{x}_i - \vec{x}_j)\|_2$$

Assume that $\vec{x}_1, \ldots \vec{x}_n$ lie in any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.

- Let $\vec{v}_1, \vec{v}_2, \ldots \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and let $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns.
- For all i, j we have $\vec{x}_i - \vec{x}_j \in \mathcal{V}$ and (a good exercise!):

$$\|\vec{x}_i - \vec{x}_j\|_2 = \sqrt{\sum_{\ell=1}^{k} \langle v_\ell, \vec{x}_i - \vec{x}_j \rangle^2} = \|\mathbf{V}^T(\vec{x}_i - \vec{x}_j)\|_2.$$

$$\tilde{x}_1 = [\mathbf{V}^T] [x_1] = [\tilde{x}_1]$$

- If we set $\tilde{x}_i \in \mathbb{R}^k$ to $\tilde{x}_i = \mathbf{V}^T\vec{x}_i$ we have:

$k \times d \cdot d \times 1 = k \times 1$

$$\|\tilde{x}_i - \tilde{x}_j\|_2 = \|\mathbf{V}^T\vec{x}_i - \mathbf{V}^T\vec{x}_j\|_2 = \|\mathbf{V}^T(\vec{x}_i - \vec{x}_j)\|_2 = \|\vec{x}_i - \vec{x}_j\|_2.$$

$$\sqrt{\sum_{\ell=1}^{k} \mathbf{V}^T(x_i - x_j)[\ell]^2}$$

$$\langle v_\ell, x_i - x_j \rangle$$

Embedding with Assumptions

Assume that $\vec{x}_1, \ldots \vec{x}_n$ lie in any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.

- Let $\vec{v}_1, \vec{v}_2, \ldots \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and let $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns.
- For all i, j we have $\vec{x}_i - \vec{x}_j \in \mathcal{V}$ and (a good exercise!):

$$\|\vec{x}_i - \vec{x}_j\|_2 = \sqrt{\sum_{\ell=1}^{k} \langle v_\ell, \vec{x}_i - \vec{x}_j \rangle^2} = \|\mathbf{V}^T(\vec{x}_i - \vec{x}_j)\|_2.$$

- If we set $\tilde{x}_i \in \mathbb{R}^k$ to $\tilde{x}_i = \mathbf{V}^T \vec{x}_i$ we have:

$$\|\tilde{x}_i - \tilde{x}_j\|_2 = \|\mathbf{V}^T \vec{x}_i - \mathbf{V}^T \vec{x}_j\|_2 = \|\mathbf{V}^T(\vec{x}_i - \vec{x}_j)\|_2 = \|\vec{x}_i - \vec{x}_j\|_2.$$

- An embedding with no distortion from any d into $m = k$.

Embedding with Assumptions

Assume that $\vec{x}_1, \ldots \vec{x}_n$ lie in any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.

- Let $\vec{v}_1, \vec{v}_2, \ldots \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and let $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns.
- For all i, j we have $\vec{x}_i - \vec{x}_j \in \mathcal{V}$ and (a good exercise!):

$$\|\vec{x}_i - \vec{x}_j\|_2 = \sqrt{\sum_{\ell=1}^{k} \langle v_\ell, \vec{x}_i - \vec{x}_j \rangle^2} = \|\mathbf{V}^T(\vec{x}_i - \vec{x}_j)\|_2.$$

- If we set $\tilde{x}_i \in \mathbb{R}^k$ to $\tilde{x}_i = \mathbf{V}^T\vec{x}_i$ we have:

$$\|\tilde{x}_i - \tilde{x}_j\|_2 = \|\mathbf{V}^T\vec{x}_i - \mathbf{V}^T\vec{x}_j\|_2 = \|\mathbf{V}^T(\vec{x}_i - \vec{x}_j)\|_2 = \|\vec{x}_i - \vec{x}_j\|_2.$$

- An embedding with no distortion from any d into $m = k$.
- $\mathbf{V}^T : \mathbb{R}^d \to \mathbb{R}^k$ is a linear map giving our embedding.

Embedding with No Assumptions

What about when we don't make any assumptions on $\vec{x}_1, \ldots, \vec{x}_n$. I.e., they can be scattered arbitrarily around d-dimensional space?

- Can we find a no-distortion embedding into $m \ll d$ dimensions?

Embedding with No Assumptions

What about when we don't make any assumptions on $\vec{x}_1, \ldots, \vec{x}_n$. I.e., they can be scattered arbitrarily around d-dimensional space?

- Can we find a no-distortion embedding into $m \ll d$ dimensions? No. Require $m = d$.

Embedding with No Assumptions

What about when we don't make any assumptions on $\vec{x}_1, \ldots, \vec{x}_n$. I.e., they can be scattered arbitrarily around d-dimensional space?

- Can we find a no-distortion embedding into $m \ll d$ dimensions? No. Require $m = d$.
- Can we find an ϵ-distortion embedding into $m \ll d$ dimensions for $\epsilon > 0$?

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

Embedding with No Assumptions

What about when we don't make any assumptions on $\vec{x}_1, \ldots, \vec{x}_n$. I.e., they can be scattered arbitrarily around d-dimensional space?

- Can we find a no-distortion embedding into $m \ll d$ dimensions? No. Require $m = d$.
- Can we find an ϵ-distortion embedding into $m \ll d$ dimensions for $\epsilon > 0$? Yes! Always, with m depending on ϵ.

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

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 $\mathbf{\Pi} : \mathbb{R}^d \to \mathbb{R}^m$ such that $m = O\left(\frac{\log n}{\epsilon^2}\right)$ and letting $\tilde{x}_i = \mathbf{\Pi}\vec{x}_i$:

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

Further, if $\mathbf{\Pi} \in \mathbb{R}^{m \times d}$ has each entry chosen i.i.d. from $\mathcal{N}(0, 1/m)$, it satisfies the guarantee with high probability.

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 $\mathbf{\Pi} : \mathbb{R}^d \rightarrow \mathbb{R}^m$ such that $m = O\left(\frac{\log n}{\epsilon^2}\right)$ and letting $\tilde{x}_i = \mathbf{\Pi}\vec{x}_i$:

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

Further, if $\mathbf{\Pi} \in \mathbb{R}^{m \times d}$ has each entry chosen i.i.d. from $\mathcal{N}(0, 1/m)$, it satisfies the guarantee with high probability.

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 $\mathbf{\Pi} : \mathbb{R}^d \to \mathbb{R}^m$ such that $m = O\left(\frac{\log n}{\epsilon^2}\right)$ and letting $\tilde{x}_i = \mathbf{\Pi}\vec{x}_i$:

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

Further, if $\mathbf{\Pi} \in \mathbb{R}^{m \times d}$ has each entry chosen i.i.d. from $\mathcal{N}(0, 1/m)$, it satisfies the guarantee with high probability.

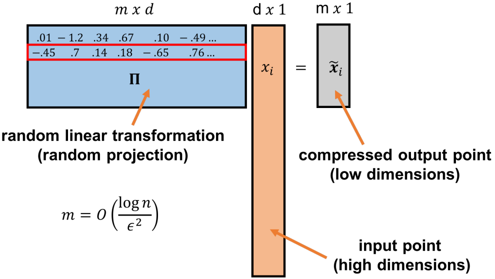
For $d = 1$ trillion, $\epsilon = .05$, and $n = 100,000$, $m \approx 6600$.

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.

Random Projection

For any $\vec{x}_1, \ldots, \vec{x}_n$ and $\mathbf{\Pi} \in \mathbb{R}^{m \times d}$ with each entry chosen i.i.d. from $\mathcal{N}(0, 1/m)$, with high probability, letting $\tilde{\mathbf{x}}_i = \mathbf{\Pi}\vec{x}_i$:

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

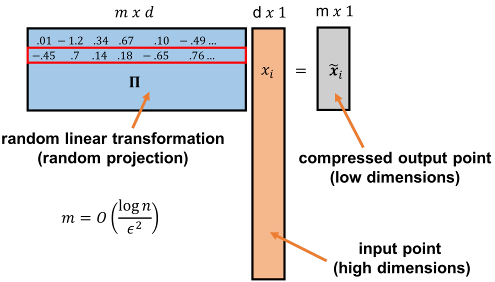


$$m = O\left(\frac{\log n}{\epsilon^2}\right)$$

Random Projection

For any $\vec{x}_1, \ldots, \vec{x}_n$ and $\mathbf{\Pi} \in \mathbb{R}^{m \times d}$ with each entry chosen i.i.d. from $\mathcal{N}(0, 1/m)$, with high probability, letting $\tilde{\mathbf{x}}_i = \mathbf{\Pi}\vec{x}_i$:

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

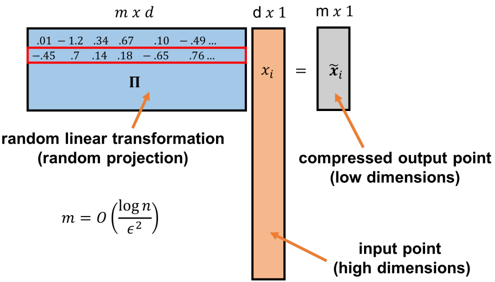


- $\mathbf{\Pi}$ is known as a random projection. It is a random linear function, mapping length d vectors to length m vectors.

Random Projection

For any $\vec{x}_1, \ldots, \vec{x}_n$ and $\mathbf{\Pi} \in \mathbb{R}^{m \times d}$ with each entry chosen i.i.d. from $\mathcal{N}(0, 1/m)$, with high probability, letting $\tilde{\mathbf{x}}_i = \mathbf{\Pi}\vec{x}_i$:

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



- $\mathbf{\Pi}$ is known as a random projection. It is a random linear function, mapping length d vectors to length m vectors.
- $\mathbf{\Pi}$ is data oblivious. Stark contrast to methods like PCA.