# COMPSCI 514: ALGORITHMS FOR DATA SCIENCE

Cameron Musco

University of Massachusetts Amherst. Spring 2020.
Lecture 10

# LOGISTICS

- Problem Set 2 is due Sunday 3/8.
- Midterm on Thursday, 3/12. Will cover material through today.
- I have posted a study guide and practice questions on the course schedule.
- Next Tuesday I can't do office hours after class. I will hold them before class on Tuesday (10:00am - 11:15am) and after class on Thursday (12:45pm-2:00pm).

**Last Class: Dimensionality Reduction**

- Finished up Count-Min Sketch and Frequent Items.
- Applications and examples of dimensionality reduction in data science (PCA, LSA, autoencoders, etc.)
- Low-distortion embeddings and some simple cases of when no-distortion embeddings are possible.

**The Johnson-Lindenstrauss Lemma.**

- Any data set can be embedded with low distortion into low-dimensional space.
- Prove the JL Lemma.
- Discuss algorithmic considerations, connections to other methods (SimHash), etc.

**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 $i, j \in [n]$:

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

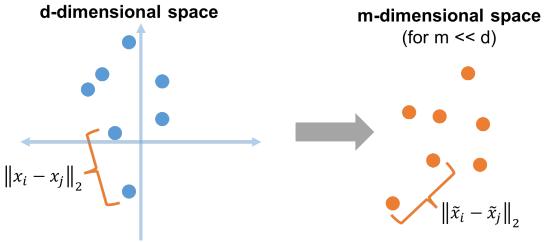
**Euclidean Low Distortion Embedding:** Given $\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$ and error parameter $\epsilon \geq 0$, find $\tilde{x}_1, \ldots, \tilde{x}_n \in \mathbb{R}^m$ (where $m \ll d$) such that for all $i, j \in [n]$:

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

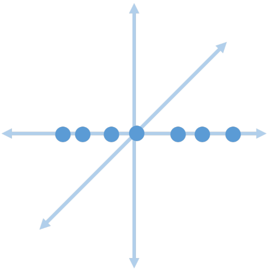
We will primarily focus on this restricted notion in this class.

**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 $i, j \in [n]$:

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



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., $\tilde{x}_i$ is just a single number).

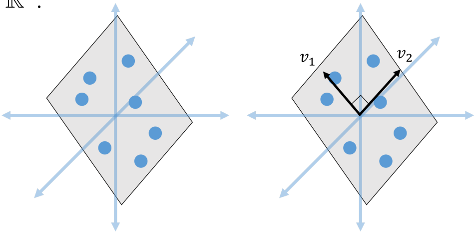
- $\|\tilde{x}_i - \tilde{x}_j\|_2 = \sqrt{[\vec{x}_i(1) - \vec{x}_j(1)]^2} = |\vec{x}_i(1) - \vec{x}_j(1)| = \|\vec{x}_i - \vec{x}_j\|_2$.
- An embedding with no distortion from any $d$ into $m = 1$.

# EMBEDDING WITH ASSUMPTIONS

Assume that $\vec{x}_1, \ldots, \vec{x}_n$ all lie on the unit circle in $\mathbb{R}^2$.


- Admits a low-distortion embedding to 1 dimension by letting $\tilde{x}_i = \theta(\vec{x}_i)$.
- Does it admit a low-distortion Euclidean embedding? No! Send me a proof on Piazza for 3 bonus points on Problem Set 2.

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

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

- 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.

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 $\epsilon$-distortion embedding into $m \ll d$ dimensions for $\epsilon > 0$? Yes! Always, with $m$ depending on $\epsilon$.

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

**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 $\mathbf{\Pi} : \mathbb{R}^d \to R^m$ 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.

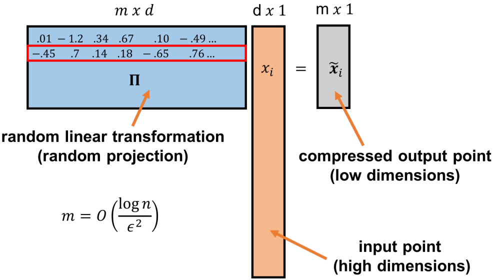
For $d = 1$ trillion, $\epsilon = .05$, and $n = 100,000$, $m \approx 6600$.

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 $\tilde{\mathbf{x}}_i = \mathbf{\Pi}\vec{x}_i$:

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



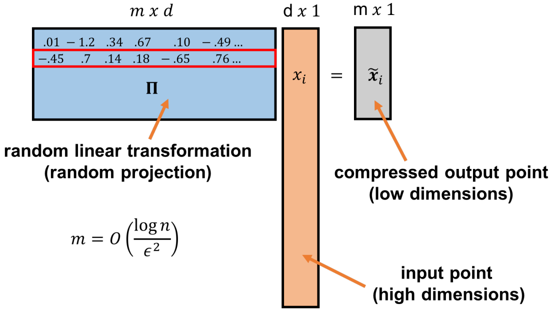

- $\mathbf{\Pi}$ is known as a random projection. It is a random linear function, mapping length $d$ vectors to length $m$ vectors.
- $\mathbf{\Pi}$ is data oblivious. Stark contrast to methods like PCA.

- Many alternative constructions: $\pm 1$ entries, sparse (most entries 0), Fourier structured (Problem Set 2), etc. $\implies$ more efficient computation of $\tilde{\mathbf{x}}_i = \mathbf{\Pi}\vec{x}_i$.
- Data oblivious property means that once $\mathbf{\Pi}$ is chosen, $\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_n$ can be computed in a stream with little memory.
- Memory needed is just $O(d + nm)$ vs. $O(nd)$ to store the full data set.
- Compression can also be easily performed in parallel on different servers.
- When new data points are added, can be easily compressed, without updating existing points.

Compression operation is $\tilde{\mathbf{x}}_i = \mathbf{\Pi}\vec{x}_i$, so for any $j$,

$$\tilde{\mathbf{x}}_i(j) = \langle \mathbf{\Pi}(j), \vec{x}_i \rangle = \sum_{k=1}^{d} \mathbf{\Pi}(j,k) \cdot \vec{x}_i(k).$$

$\mathbf{\Pi}(j)$ is a vector with independent random Gaussian entries.




The Johnson-Lindenstrauss Lemma is a direct consequence of a closely related lemma:

**Distributional JL Lemma:** Let $\mathbf{\Pi} \in \mathbb{R}^{m \times d}$ have each entry chosen i.i.d. as $\mathcal{N}(0, 1/m)$. If we set $m = O\left(\frac{\log(1/\delta)}{\epsilon^2}\right)$, then for any $\vec{y} \in \mathbb{R}^d$, with probability $\geq 1 - \delta$

$$(1-\epsilon)\|\vec{y}\|_2 \leq \|\mathbf{\Pi}\vec{y}\|_2 \leq (1+\epsilon)\|\vec{y}\|_2$$

Applying a random matrix $\mathbf{\Pi}$ to any vector $\vec{y}$ preserves $\vec{y}$'s norm with high probability.

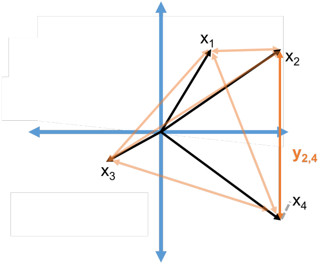
- Like a low-distortion embedding, but for the length of a compressed vector rather than distances between vectors.
- Can be proven from first principles. Will see next.

$\mathbf{\Pi} \in \mathbb{R}^{m \times d}$: random projection matrix. $d$: original dimension. $m$: compressed dimension, $\epsilon$: embedding error, $\delta$: embedding failure prob.

**Distributional JL Lemma $\implies$ JL Lemma:** Distributional JL show that a random projection $\mathbf{\Pi}$ preserves the norm of any $y$. The main JL Lemma says that $\mathbf{\Pi}$ preserves distances between vectors.

Since $\mathbf{\Pi}$ is linear these are the same thing!

**Proof:** Given $\vec{x}_1, \ldots, \vec{x}_n$, define $\binom{n}{2}$ vectors $\vec{y}_{ij}$ where $\vec{y}_{ij} = \vec{x}_i - \vec{x}_j$.




- If we choose $\mathbf{\Pi}$ with $m = O\left(\frac{\log 1/\delta}{\epsilon^2}\right)$, for each $\vec{y}_{ij}$ with probability $\geq 1 - \delta$ we have:

**Claim:** If we choose $\mathbf{\Pi}$ with i.i.d. $\mathcal{N}(0, 1/m)$ entries and $m = O\left(\frac{\log(1/\delta')}{\epsilon^2}\right)$, letting $\tilde{\mathbf{x}}_i = \mathbf{\Pi}\vec{x}_i$, for each pair $\vec{x}_i, \vec{x}_j$ with probability $\geq 1 - \delta'$ we have:

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

With what probability are all pairwise distances preserved?

**Union bound:** With probability $\geq 1 - \binom{n}{2} \cdot \delta'$ all pairwise distances are preserved.

Apply the claim with $\delta' = \delta / \binom{n}{2}$. $\implies$ for $m = O\left(\frac{\log(1/\delta')}{\epsilon^2}\right)$, all pairwise distances are preserved with probability $\geq 1 - \delta$.

$$m = O\left(\frac{\log(1/\delta')}{\epsilon^2}\right) = O\left(\frac{\log(\binom{n}{2}/\delta)}{\epsilon^2}\right) = O\left(\frac{\log(n^2/\delta)}{\epsilon^2}\right) = O\left(\frac{\log(n/\delta)}{\epsilon^2}\right)$$

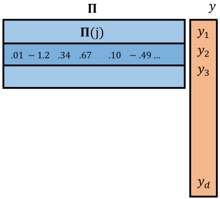
Yields the JL lemma.

# DISTRIBUTIONAL JL PROOF

**Distributional JL Lemma:** Let $\mathbf{\Pi} \in \mathbb{R}^{m \times d}$ have each entry chosen i.i.d. as $\mathcal{N}(0, 1/m)$. If we set $m = O\left(\frac{\log(1/\delta)}{\epsilon^2}\right)$, then for any $\vec{y} \in \mathbb{R}^d$, with probability $\geq 1 - \delta$

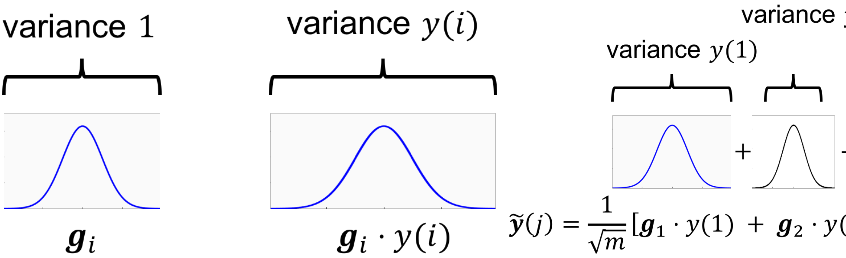
$$(1-\epsilon)\|\vec{y}\|_2 \leq \|\mathbf{\Pi}\vec{y}\|_2 \leq (1+\epsilon)\|\vec{y}\|_2$$

- Let $\tilde{\mathbf{y}}$ denote $\mathbf{\Pi}\vec{y}$ and let $\mathbf{\Pi}(j)$ denote the $j^{th}$ row of $\mathbf{\Pi}$.
- For any $j$, $\tilde{\mathbf{y}}(j) = \langle \mathbf{\Pi}(j), \vec{y} \rangle = \frac{1}{\sqrt{m}} \sum_{i=1}^{d} \mathbf{g}_i \cdot \vec{y}(i)$ where $\mathbf{g}_i \sim \mathcal{N}(0, 1/m1)$.




$\vec{y} \in \mathbb{R}^d$: arbitrary vector, $\tilde{\mathbf{y}} \in \mathbb{R}^m$: compressed vector, $\mathbf{\Pi} \in \mathbb{R}^{m \times d}$: random projection. $d$: original dim. $m$: compressed dim, $\epsilon$: error, $\delta$: failure prob.

- Let $\tilde{\mathbf{y}}$ denote $\mathbf{\Pi}\vec{y}$ and let $\mathbf{\Pi}(j)$ denote the $j^{th}$ row of $\mathbf{\Pi}$.
- For any $j$, $\tilde{\mathbf{y}}(j) = \langle \mathbf{\Pi}(j), \vec{y} \rangle = \frac{1}{\sqrt{m}} \sum_{i=1}^{d} \mathbf{g}_i \cdot \vec{y}(i)$ where $\mathbf{g}_i \sim \mathcal{N}(0,1)$.
- $\mathbf{g}_i \cdot \vec{y}(i) \sim \mathcal{N}(0, \vec{y}(i)^2)$: a normal distribution with variance $\vec{y}(i)^2$.




What is the distribution of $\tilde{\mathbf{y}}(j)$? Also Gaussian!

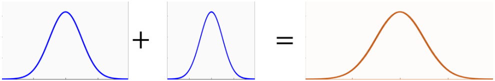
$\vec{y} \in \mathbb{R}^d$: arbitrary vector, $\tilde{\mathbf{y}} \in \mathbb{R}^m$: compressed vector, $\mathbf{\Pi} \in \mathbb{R}^{m \times d}$: random projection mapping $\vec{y} \to \tilde{\mathbf{y}}$. $\mathbf{\Pi}(j)$: $j^{th}$ row of $\mathbf{\Pi}$, $d$: original dimension. $m$: compressed dimension, $\mathbf{g}_i$: normally distributed random variable

Letting $\tilde{\mathbf{y}} = \mathbf{\Pi}\vec{y}$, we have $\tilde{\mathbf{y}}(j) = \langle \mathbf{\Pi}(j), \vec{y} \rangle$ and:

$$\tilde{\mathbf{y}}(j) = \frac{1}{\sqrt{m}} \sum_{i=1}^{d} \mathbf{g}_i \cdot \vec{y}(i) \text{ where } \mathbf{g}_i \cdot \vec{y}(i) \sim \mathcal{N}(0, \vec{y}(i)^2).$$

**Stability of Gaussian Random Variables.** For independent $a \sim \mathcal{N}(\mu_1, \sigma_1^2)$ and $b \sim \mathcal{N}(\mu_2, \sigma_2^2)$ we have:

$$a + b \sim \mathcal{N}(\mu_1 + \mu_2, \sigma_1^2 + \sigma_2^2)$$



Thus, $\tilde{\mathbf{y}}(j) \sim \frac{1}{\sqrt{m}} \mathcal{N}(0, \vec{y}(1)^2 + \vec{y}(2)^2 + \ldots + \vec{y}(d)^2 \|\vec{y}\|_2^2) \mathcal{N}(0, \|\vec{y}\|_2^2/m)$. I.e., $\tilde{\mathbf{y}}$ itself is a random Gaussian vector. Rotational invariance of the Gaussian distribution.

Stability is another explanation for the central limit theorem.

**So far:** Letting $\mathbf{\Pi} \in \mathbb{R}^{d\times m}$ have each entry chosen i.i.d. as $\mathcal{N}(0, 1/m)$, for any $\vec{y} \in \mathbb{R}^d$, letting $\tilde{\mathbf{y}} = \mathbf{\Pi}\vec{y}$:

$$\tilde{\mathbf{y}}(j) \sim \mathcal{N}(0, \|\vec{y}\|_2^2/m).$$

What is $\mathbb{E}[\|\tilde{\mathbf{y}}\|_2^2]$?

$$\mathbb{E}[\|\tilde{\mathbf{y}}\|_2^2] = \mathbb{E}\left[\sum_{j=1}^{m} \tilde{\mathbf{y}}(j)^2\right] = \sum_{j=1}^{m} \mathbb{E}[\tilde{\mathbf{y}}(j)^2]$$

$$= \sum_{j=1}^{m} \frac{\|\vec{y}\|_2^2}{m} = \|\vec{y}\|_2^2$$

So $\tilde{\mathbf{y}}$ has the right norm in expectation.

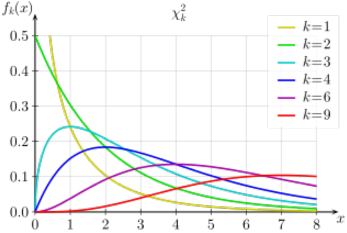
How is $\|\tilde{\mathbf{y}}\|_2^2$ distributed? Does it concentrate?

$\vec{y} \in \mathbb{R}^d$: arbitrary vector, $\tilde{\mathbf{y}} \in \mathbb{R}^m$: compressed vector, $\mathbf{\Pi} \in \mathbb{R}^{m\times d}$: random projection mapping $\vec{y} \to \tilde{\mathbf{y}}$. $\mathbf{\Pi}(j)$: $j^{th}$ row of $\mathbf{\Pi}$, $d$: original dimension. $m$: compressed dimension, $\mathbf{g}_i$: normally distributed random variable

**So far:** Letting $\mathbf{\Pi} \in \mathbb{R}^{d \times m}$ have each entry chosen i.i.d. as $\frac{1}{\sqrt{m}} \cdot \mathcal{N}(0,1)$, for any $\vec{y} \in \mathbb{R}^d$, letting $\tilde{\mathbf{y}} = \mathbf{\Pi}\vec{y}$:

$$\tilde{\mathbf{y}}(j) \sim \mathcal{N}(0, \|\vec{y}\|_2^2/m) \text{ and } \mathbb{E}[\|\tilde{\mathbf{y}}\|_2^2] = \|\vec{y}\|_2^2$$

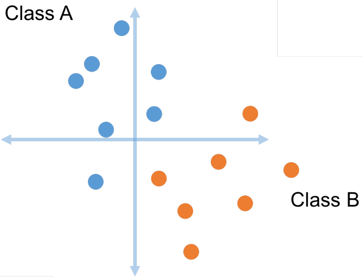
$\|\tilde{\mathbf{y}}\|_2^2 = \sum_{i=1}^m \tilde{\mathbf{y}}(j)^2$ a Chi-Squared random variable with $m$ degrees of freedom (a sum of $m$ squared independent Gaussians)

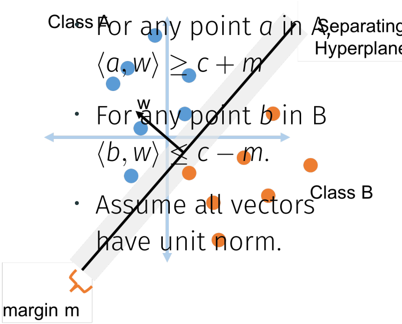



**Lemma:** (Chi-Squared Concentration) Letting $\mathbf{Z}$ be a Chi-Squared random variable with $m$ degrees of freedom,

$$\Pr\left[|\mathbf{Z} - \mathbb{E}\mathbf{Z}| \geq \epsilon \mathbb{E}\mathbf{Z}\right] \leq 2e^{-m\epsilon^2/8}.$$

**Support Vector Machines:** A classic ML algorithm, where data is classified with a hyperplane.




- For any point $a$ in A, $\langle a, w\rangle \geq c + m$
- For any point $b$ in B $\langle b, w\rangle \leq c - m$.
- Assume all vectors have unit norm.

JL Lemma implies that after projection into $O\left(\frac{\log n}{m^2}\right)$ dimensions, still have $\langle \tilde{a}, \tilde{w}\rangle \geq c + m/4$ and $\langle \tilde{b}, \tilde{w}\rangle \leq c - m/4$.

**Upshot:** Can random project and run SVM (much more efficiently) in the lower dimensional space to find separator $\tilde{w}$.

Questions?