Machine Learning for Data Science (CS4786)
Lecture 5

Random Projections

Course Webpage :
http://www.cs.cornell.edu/Courses/cs4786/2016sp/

Recap

Dimensionality Reduction

Given feature vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$, compress the data points into low dimensional representation $\mathbf{y}_1, \ldots, \mathbf{y}_n \in \mathbb{R}^K$ where $K << d$

Principal Component Analysis:

- **Find directions that maximize variance (spread)**
- **Find directions that minimize reconstruction error**

Principal Component Analysis

1. $\Sigma = \text{cov}\left(X \right)$

2. $W = \text{eigs}\left(\Sigma, K \right)$

3. $Y = X - \mu \times W$

Reconstruction

4. $$\widehat{X} = Y \times W^\top + \mu$$

Two View Dimensionality Reduction

- Data can be split into pairs $(\mathbf{x}_1, \mathbf{x}'_1), \ldots, (\mathbf{x}_n, \mathbf{x}'_n)$ where $\mathbf{x}_t$'s are d_1 dimensional and $\mathbf{x}'_t$'s are d_2 dimensional

- Goal: Compress $\mathbf{x}_1, \ldots, \mathbf{x}_n$ into K dimensional vectors $\mathbf{y}_1, \ldots, \mathbf{y}_n$ (or $\mathbf{x}'_1, \ldots, \mathbf{x}'_n$ into $\mathbf{y}'_1, \ldots, \mathbf{y}'_n$ or both)
  - Retain information redundant between the two views

Canonical Correlation Analysis:

- **Find directions that maximize correlations between the projections in the two views**

CCA Algorithm

1. $X = \left({}_{n} X_1 {}_{d_1}, X_2 {}_{d_2} \right)$

2. $\Sigma = \begin{matrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{matrix} = \mathrm{cov}\left(X \right)$

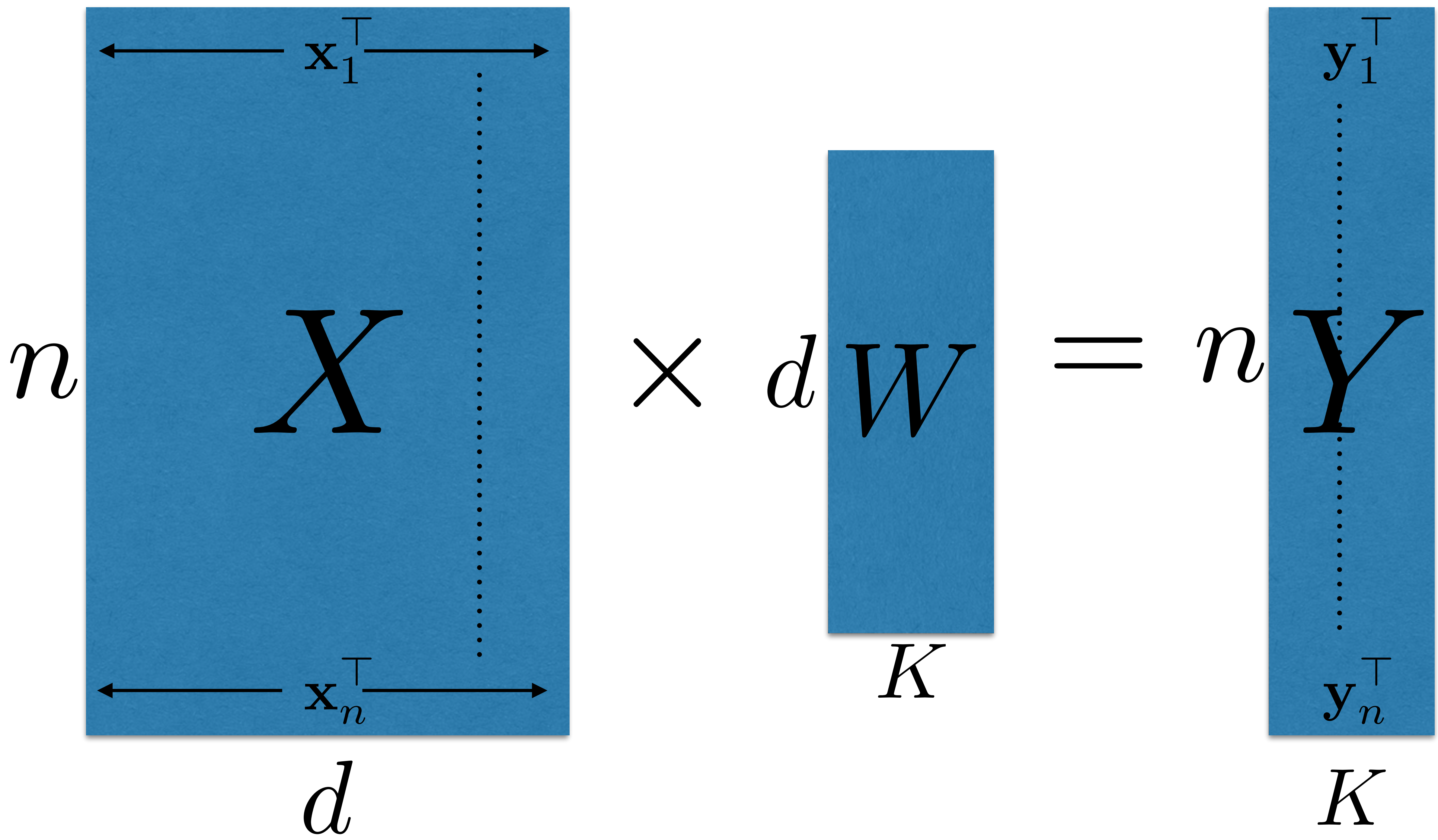
3. $W_1 = \mathrm{eigs}\left(\Sigma_{11}^{-1}\Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}, K \right)$

CCA Algorithm

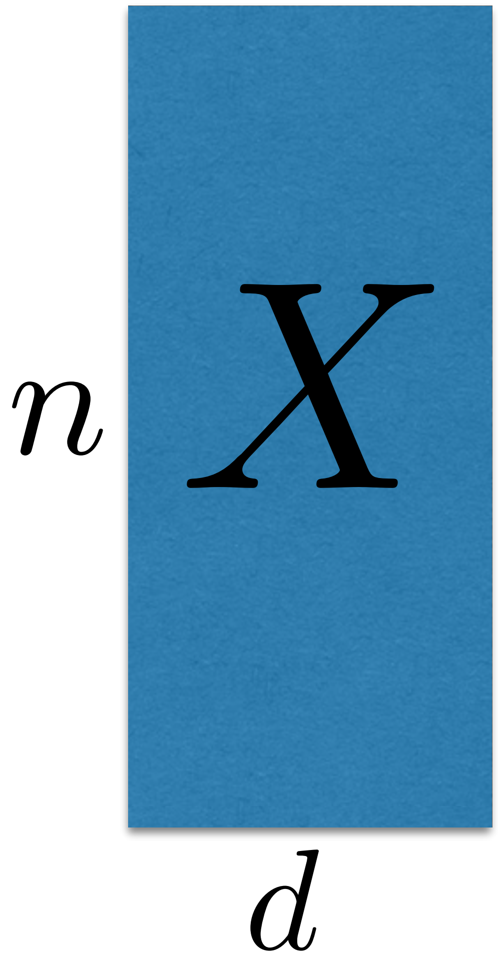
4. $Y_1 = X_1 - \mu_1 \times W_1$

CCA Demo Redo?

Back to Single View: Recap



The Tall, the Fat and the Ugly

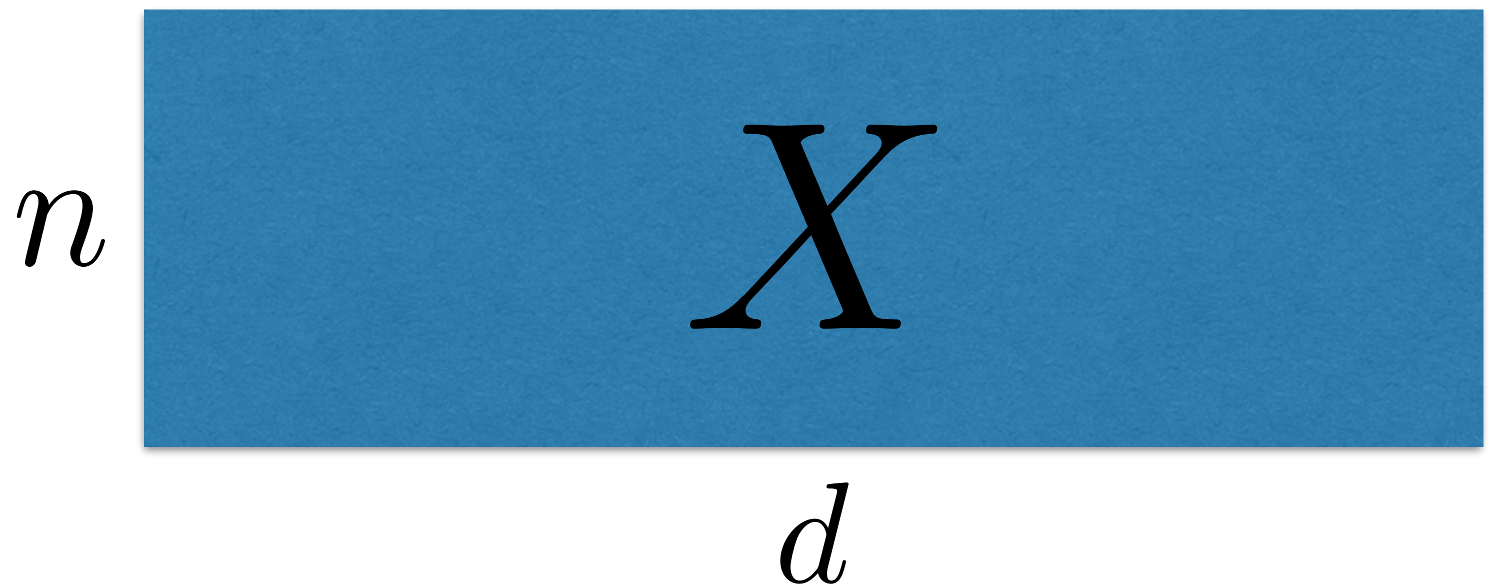


The Tall, the Fat and the Ugly

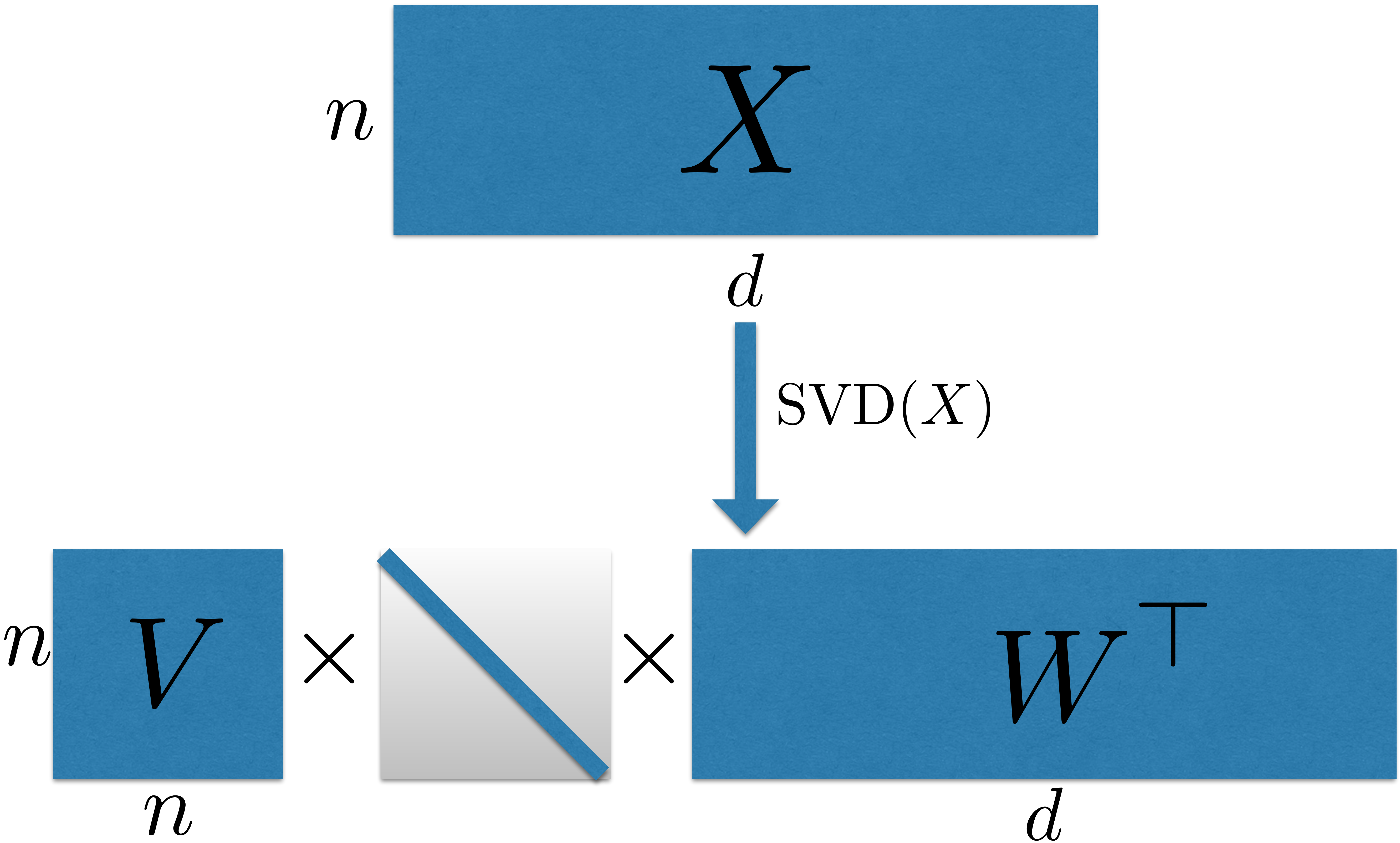
$$\underbrace{X^\top}_{d \times n} \times \underbrace{X}_{n \times d} \Big/ n = \underbrace{\Sigma}_{d \times d}$$

$$\underbrace{W}_{d \times K} = \mathrm{Eigs}\left(\Sigma, K\right)$$

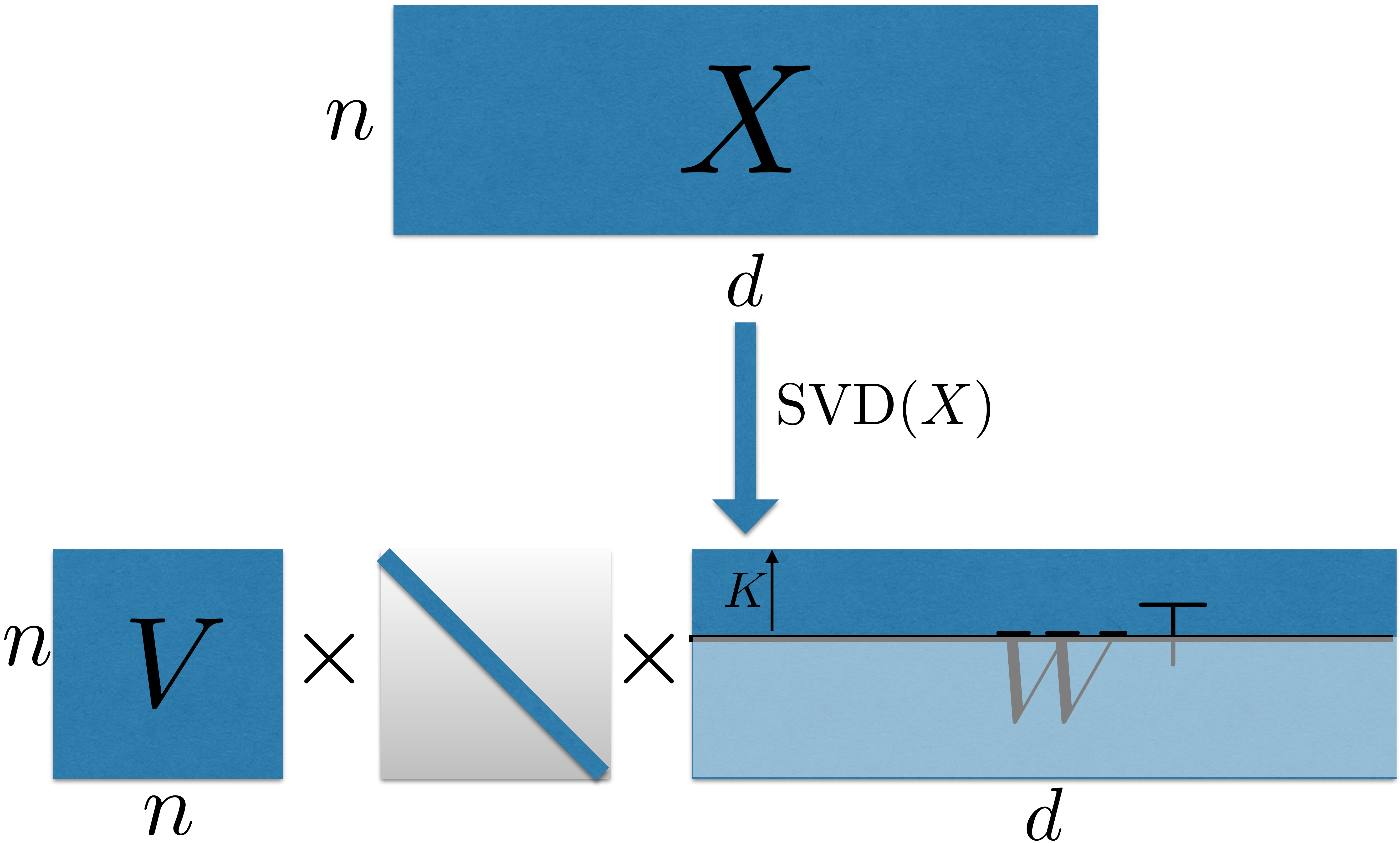
The Tall, **the Fat** and the Ugly



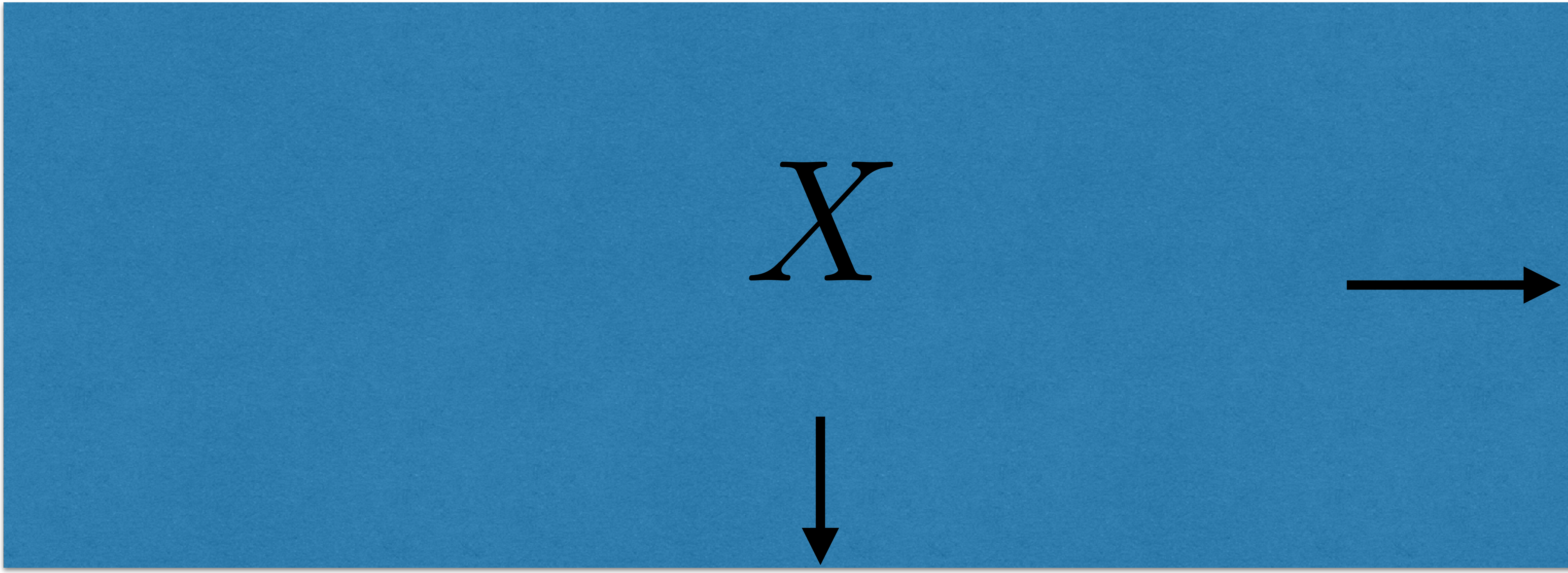
THE TALL, **the Fat** AND THE UGLY



THE TALL, **the Fat** AND THE UGLY



The Tall, the Fat and **the Ugly**



- d and n so large we can't even store in memory
- Only have time to be linear in $\text{size}(X) = n \times d$

I there any hope?

Pick a Random W

$$Y = X \times \left.\underbrace{\begin{bmatrix} +1 & \dots & -1 \\ -1 & \dots & +1 \\ +1 & \dots & -1 \\ & \vdots & \\ +1 & \dots & -1 \end{bmatrix}}_{K} d \middle/ \sqrt{K} \right.$$

Random Projection

- What does "it works" even mean?

  Distances between all pairs of data-points in low dim. projection is roughly the same as their distances in the high dim. space.

  That is, when K is "large enough", with "high probability", for all pairs of data points $i, j \in \{1, \ldots, n\}$,

$$(1 - \epsilon) \left\| \mathbf{y}_i - \mathbf{y}_j \right\|_2 \leq \left\| \mathbf{x}_i - \mathbf{x}_j \right\|_2 \leq (1 + \epsilon) \left\| \mathbf{y}_i - \mathbf{y}_j \right\|_2$$

Why should Random Projections even work?!

Consider any vector $\tilde{\mathbf{x}} \in \mathbb{R}^d$ and let $\tilde{\mathbf{y}} = W^\top \tilde{\mathbf{x}}$. Note that

$$\tilde{\mathbf{y}}[j]^2 = \left(\sum_{i=1}^{d} W[i,j] \cdot \tilde{\mathbf{x}}[i] \right)^2 = \sum_{i,i'} \left(W[i,j] \cdot \tilde{\mathbf{x}}[i] \right) \cdot \left(W[i',j] \cdot \tilde{\mathbf{x}}[i'] \right)$$

$$= \sum_{i,i'} \left(W[i,j] \cdot W[i',j] \right) \cdot \left(\tilde{\mathbf{x}}[i] \cdot \tilde{\mathbf{x}}[i'] \right)$$

Why should Random Projections even work?!

Hence,

$$\mathbb{E}\left[\tilde{\mathbf{y}}[j]^2\right] = \sum_{i,i'=1}^{d} \mathbb{E}\left[\left(W[i,j]\cdot W[i',j]\right)\right]\cdot\left(\tilde{\mathbf{x}}[i]\cdot\tilde{\mathbf{x}}[i']\right)$$

if $i \neq i'$, $W[i,j]$ and $W[i',j]$ are independent and so

$$= \sum_{i=1}^{d} \mathbb{E}\left[\left(W[i,j]^2\right)\right]\tilde{\mathbf{x}}[i]^2 + \sum_{i\neq i'}\left(\mathbb{E}[W[i,j]]\cdot\mathbb{E}\left[W[i',j]\right]\right)\cdot\left(\tilde{\mathbf{x}}[i]\cdot\tilde{\mathbf{x}}[i']\right)$$

$$= \sum_{i=1}^{d} \tilde{\mathbf{x}}[i]^2/\sqrt{K}^2 = \|\tilde{\mathbf{x}}\|_2^2/K$$

Why should Random Projections even work?!

Hence,

$$\mathbb{E}\left[\|\tilde{\mathbf{y}}\|_2^2\right] = \sum_{j=1}^{K} \mathbb{E}\left[\tilde{\mathbf{y}}[j]^2\right] = \sum_{j=1}^{K} \|\tilde{\mathbf{x}}\|_2^2 / K = \|\tilde{\mathbf{x}}\|_2^2$$

If we let $\tilde{\mathbf{x}} = \mathbf{x}_s - \mathbf{x}_t$ then

$$\tilde{\mathbf{y}} = W^\top \tilde{\mathbf{x}} = W^\top \mathbf{x}_s - W^\top \mathbf{x}_t = \mathbf{y}_s - \mathbf{y}_t$$

Hence for any $s, t \in \{1, \ldots, n\}$,

$$\mathbb{E}\left[\|\mathbf{y}_s - \mathbf{y}_t\|_2^2\right] = \|\mathbf{x}_s - \mathbf{x}_t\|_2^2$$

Lets try this in Matlab ...

Why should Random Projections even work?!

For large K, not only true in expectation but also with high probability

For any $\epsilon > 0$, if $K \approx \log(n/\delta)/\epsilon^2$, with probability $1 - \delta$ over draw of W, for all pairs of data points $i, j \in \{1, \ldots, n\}$,

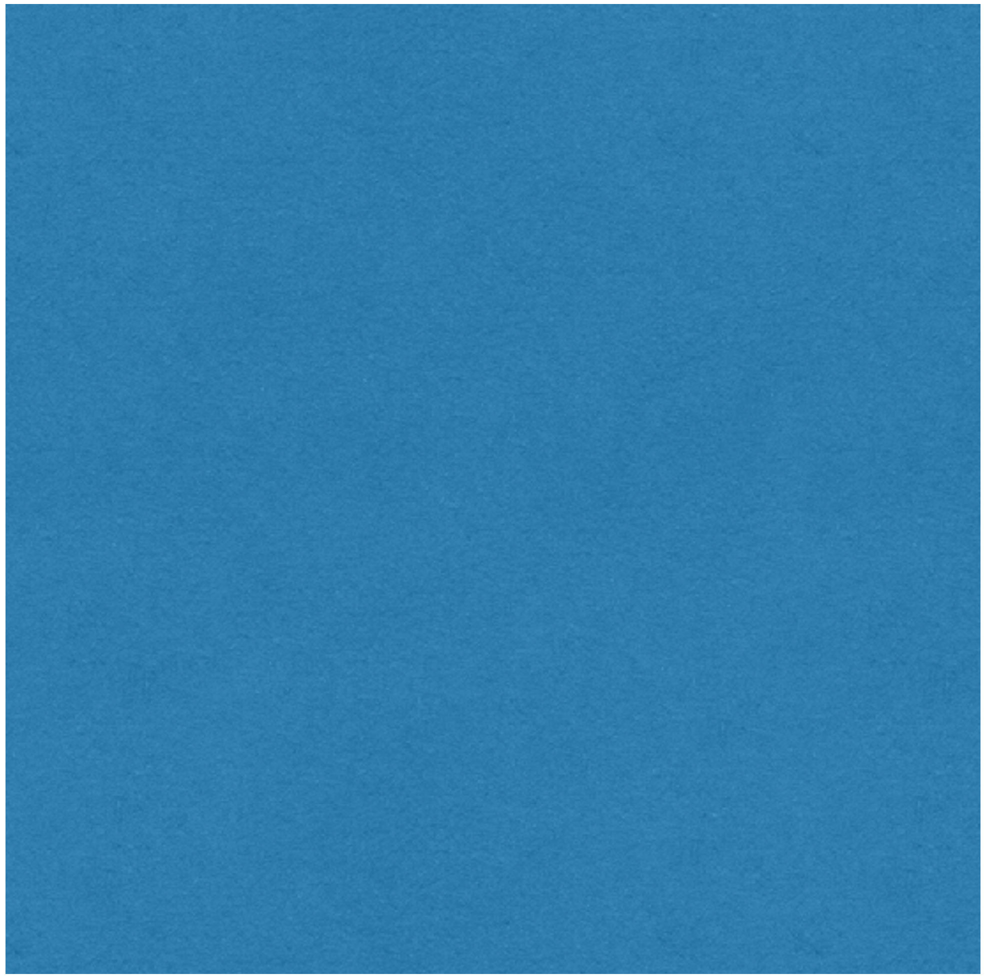
$$(1-\epsilon)\left\|\mathbf{y}_i - \mathbf{y}_j\right\|_2 \leq \left\|\mathbf{x}_i - \mathbf{x}_j\right\|_2 \leq (1+\epsilon)\left\|\mathbf{y}_i - \mathbf{y}_j\right\|_2$$

Lets try on Matlab …

This is called the Johnson-Lindenstrauss lemma or JL lemma for short.

Why is this so Ridiculously Magical?

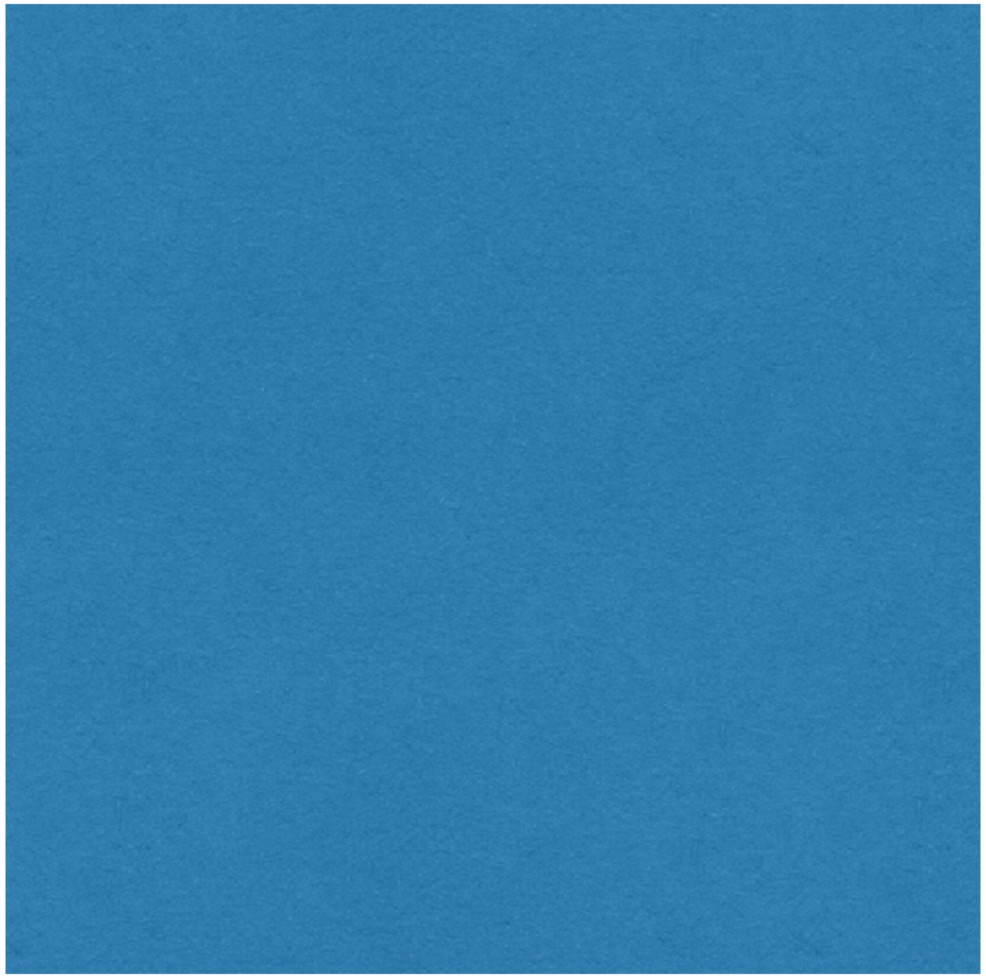
n=
1000



d = 1000

Why is this so Ridiculously Magical?

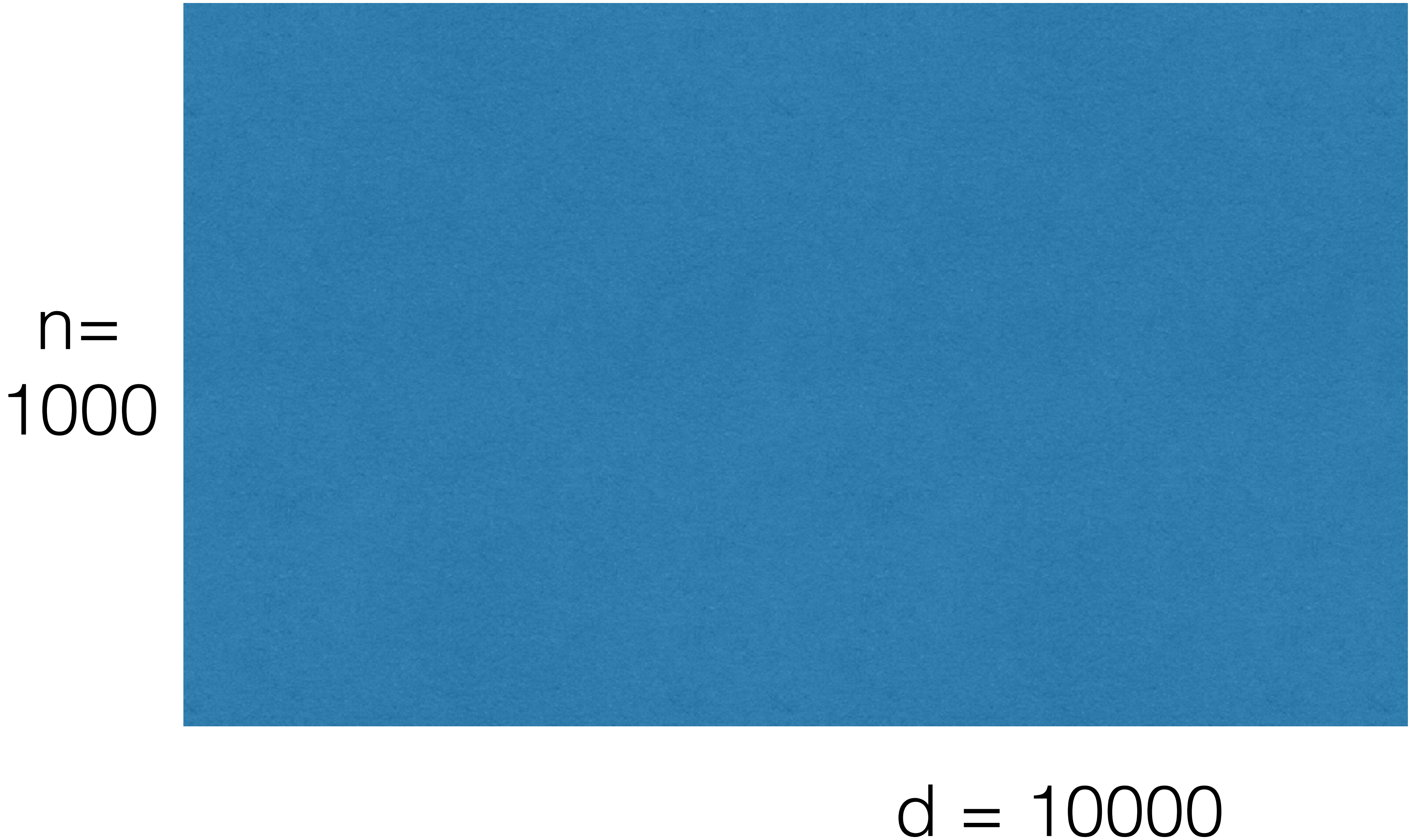
n=
1000



d = 1000

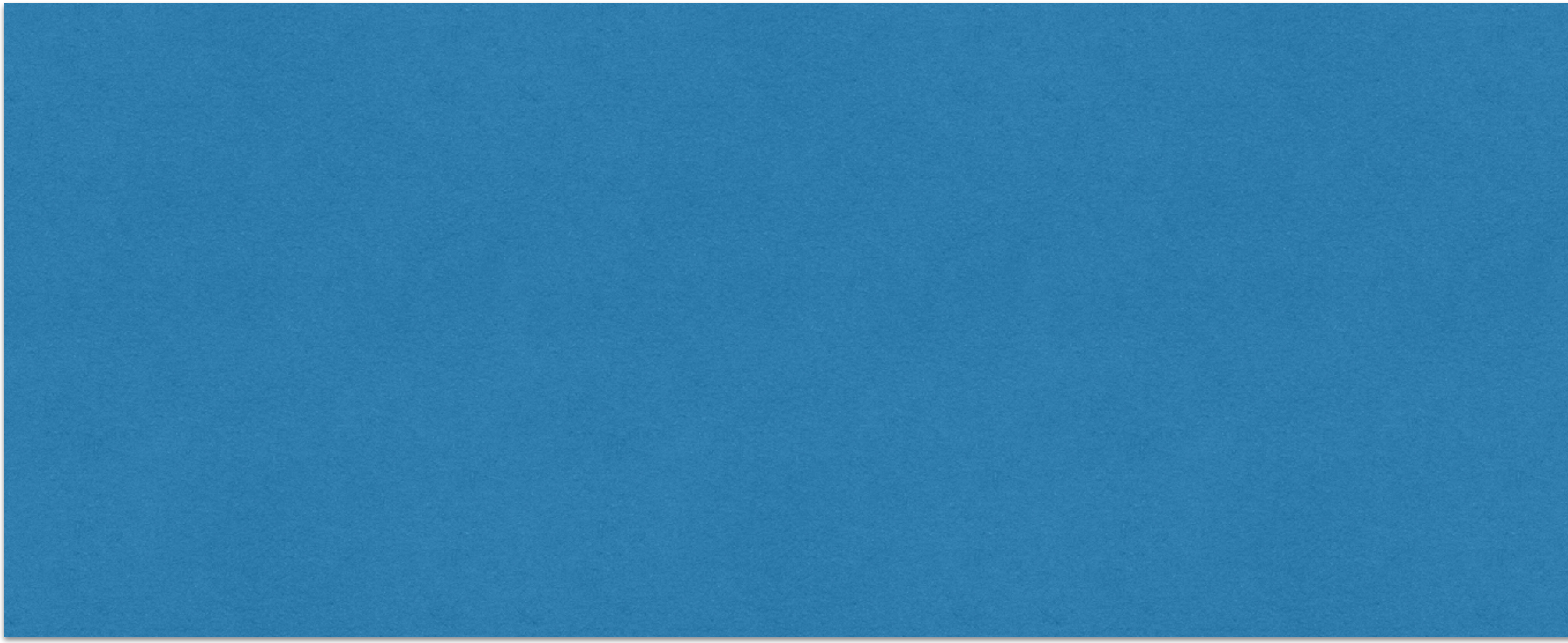
If we take $K = 69.1/\epsilon^2$, with probability 0.99 distances are preserved to accuracy ϵ

Why is this so Ridiculously Magical?



If we take $K = 69.1/\epsilon^2$, with probability 0.99 distances are preserved to accuracy ϵ

WHY IS THIS SO RIDICULOUSLY MAGICAL?



If we take $K = 69.1/\epsilon^2$, with probability 0.99 distances are preserved to accuracy ϵ