Machine Learning for Data Science (CS4786)
Lecture 3

Principal Component Analysis

Quiz

- (i,j)'th entry of matrix Σ (X is data matrix with n rows and d columns):

✓ A. Measures how i'th coordinate of data varies w.r.t j'th

B. Measures how it's data point is related to j'th

✓ C. Σ[i,j] = inner product between j'th and the i'th column of matrix X - μ divided by n

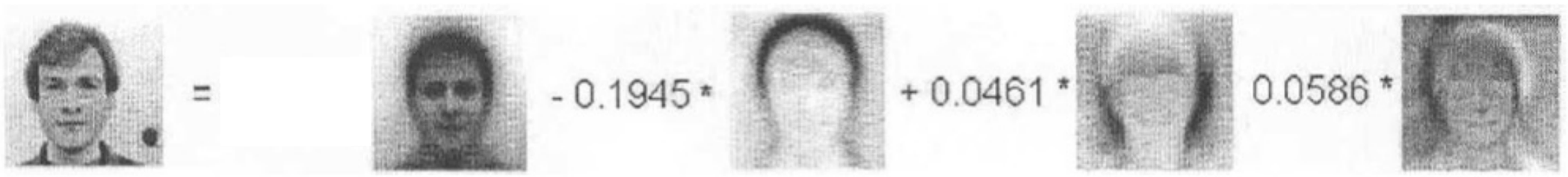
D. Σ[i,j] = inner product between j'th and the i'th row of matrix X - μ divided by n

What if our dataset looked like this?

Principal Component Analysis (PCA)

Turk & Pentland'91

Eigen Face:



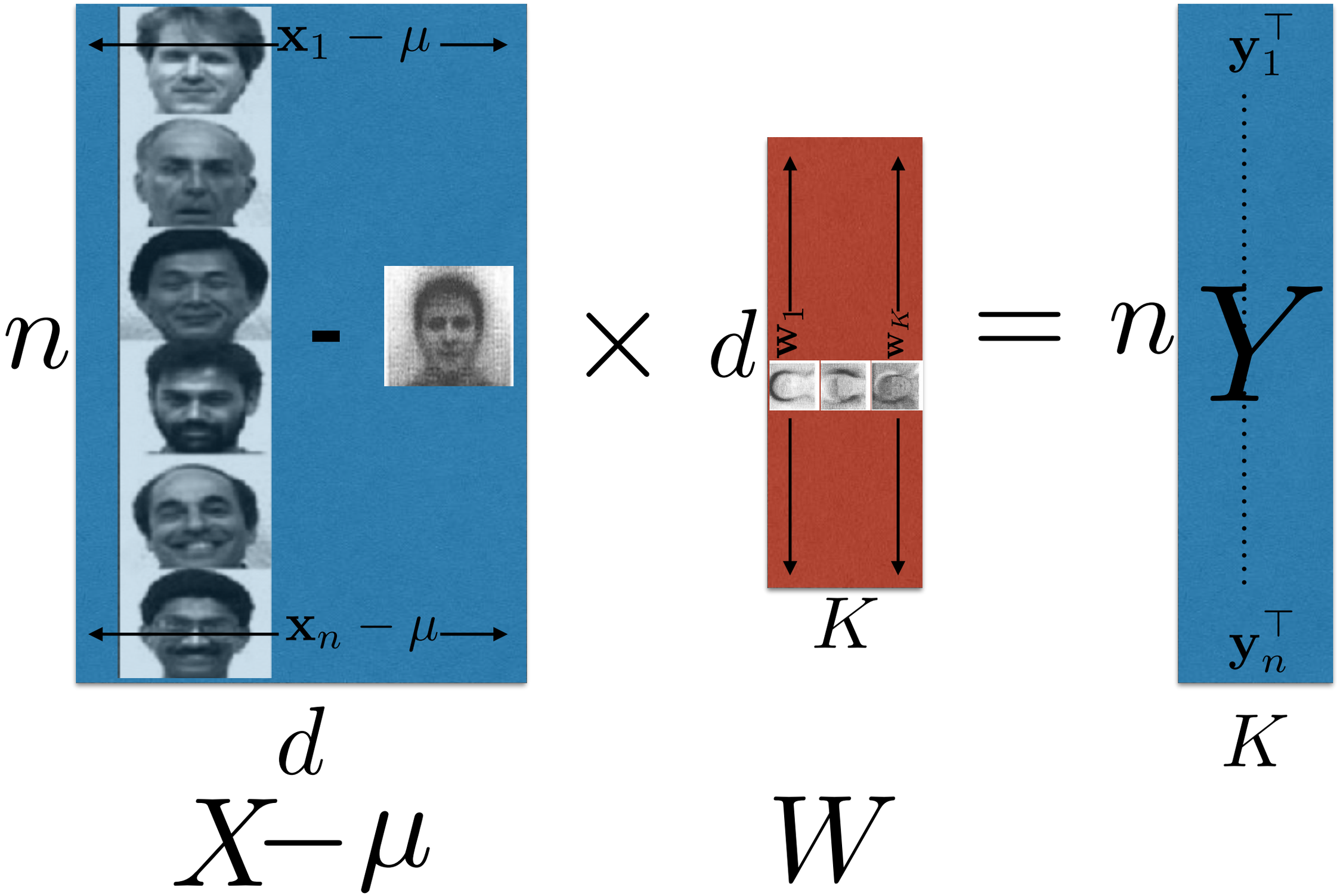
- Each x_t (each row of X) is a face image (vectorized version)

- Each y_t is the set of coefficients we multiply to the eigen face

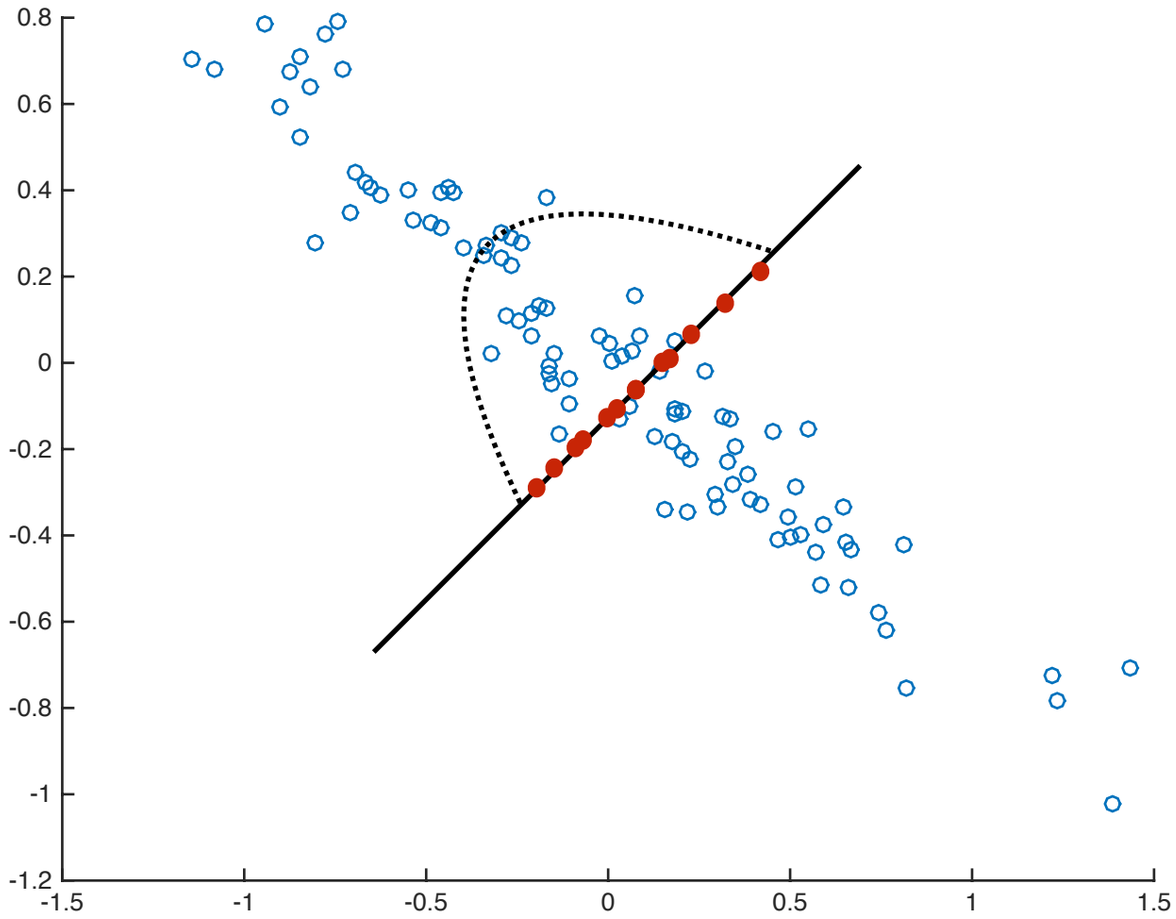
- Each column of W is an Eigenface

Dim Reduction: Linear Transformation

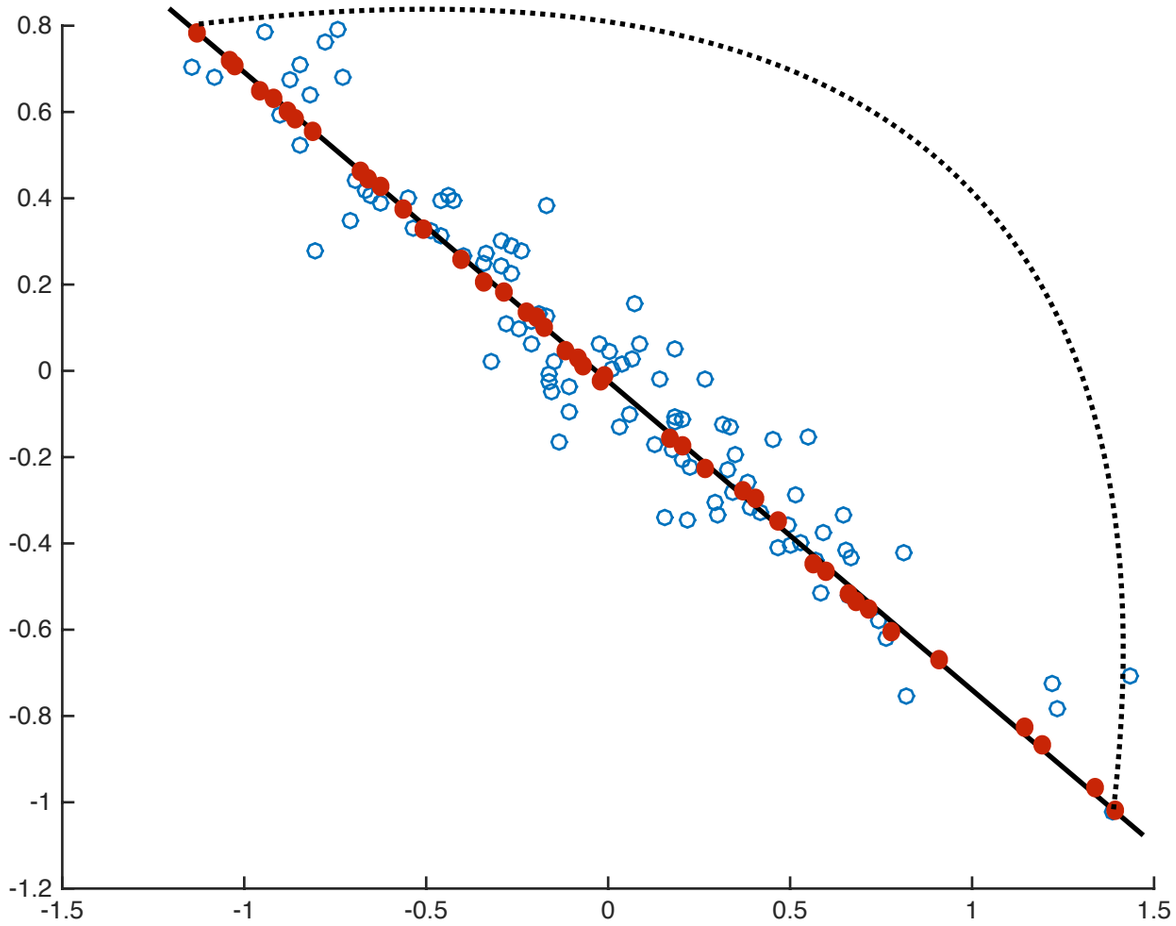
$$\mathbf{y}_i^\top = \mathbf{x}_i^\top W$$



PCA: Variance Maximization



PCA: Variance Maximization



PCA: Variance Maximization

- Pick directions along which data varies the most

$$\begin{aligned}
\text{Variance} &= \frac{1}{n}\sum_{t=1}^{n}\left(y_t - \frac{1}{n}\sum_{s=1}^{n} y_s\right)^2 \\
&= \frac{1}{n}\sum_{t=1}^{n}\left(\mathbf{w}^\top(\mathbf{x}_t - \mu)\right)^2 \\
&= \text{average squared inner product} \\
&= \mathbf{w}^\top \Sigma \mathbf{w}
\end{aligned}$$

Σ is the covariance matrix

PCA: Variance Maximization

Covariance matrix:

$$\Sigma = \frac{1}{n}\sum_{t=1}^{n}(\mathbf{x}_t - \mu)(\mathbf{x}_t - \mu)^\top$$

- Its a $d \times d$ matrix, $\Sigma[i,j]$ measures "covariance" of features i and j

$$\Sigma[i,j] = \frac{1}{n}\sum_{t=1}^{n}(\mathbf{x}_t[i] - \mu[i])(\mathbf{x}_t[j] - \mu[j])$$

PCA: Variance Maximization

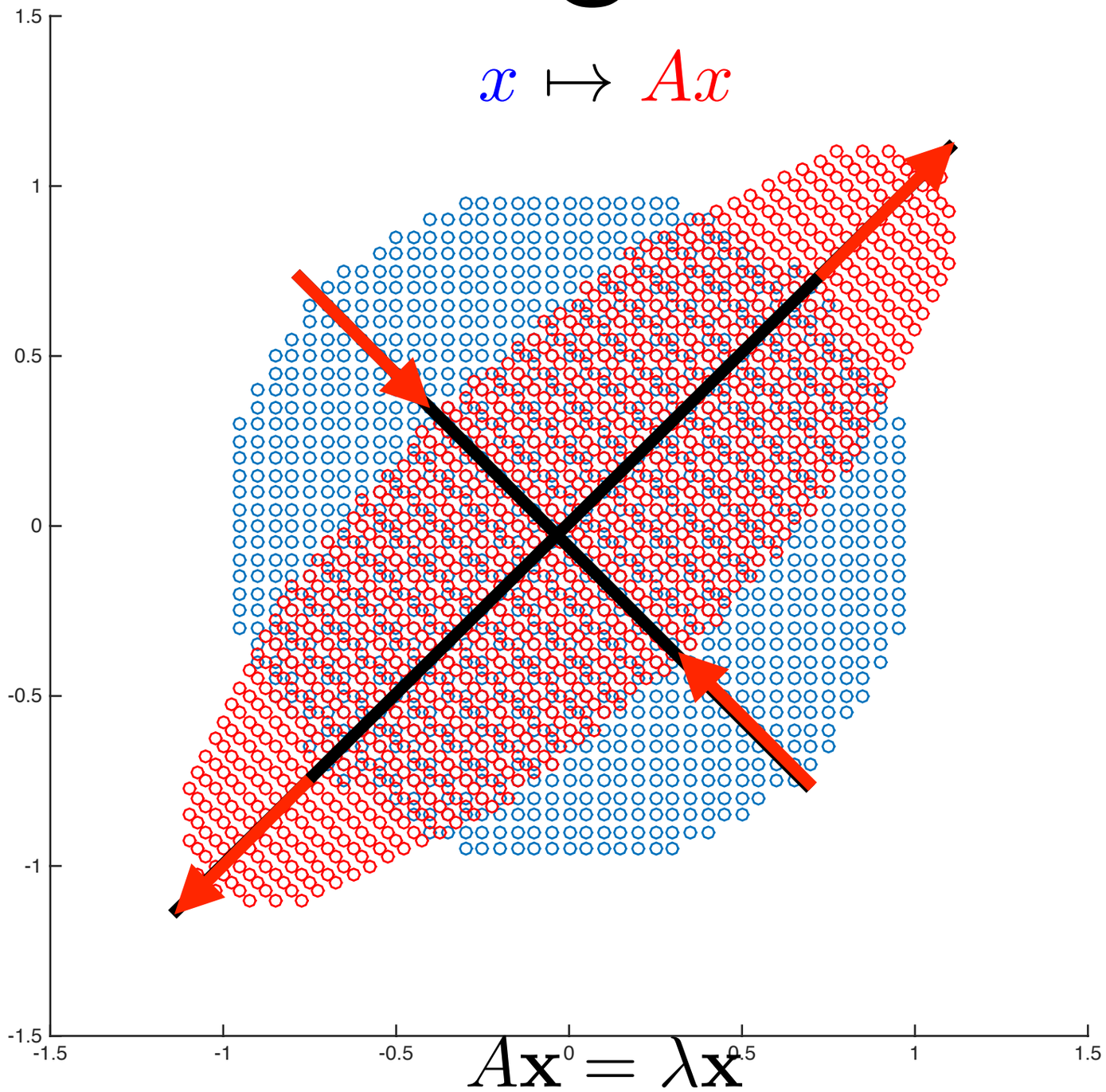
- Pick directions along which data varies the most
- First principal component:

$$\mathbf{w}_1 = \arg \max_{\mathbf{w}:\|\mathbf{w}\|_2=1} \mathbf{w}^\top \Sigma \mathbf{w}$$

Σ is the covariance matrix

Solution: $\mathbf{w}_1 =$ Largest Eigenvector of Σ

What are Eigen Vectors?



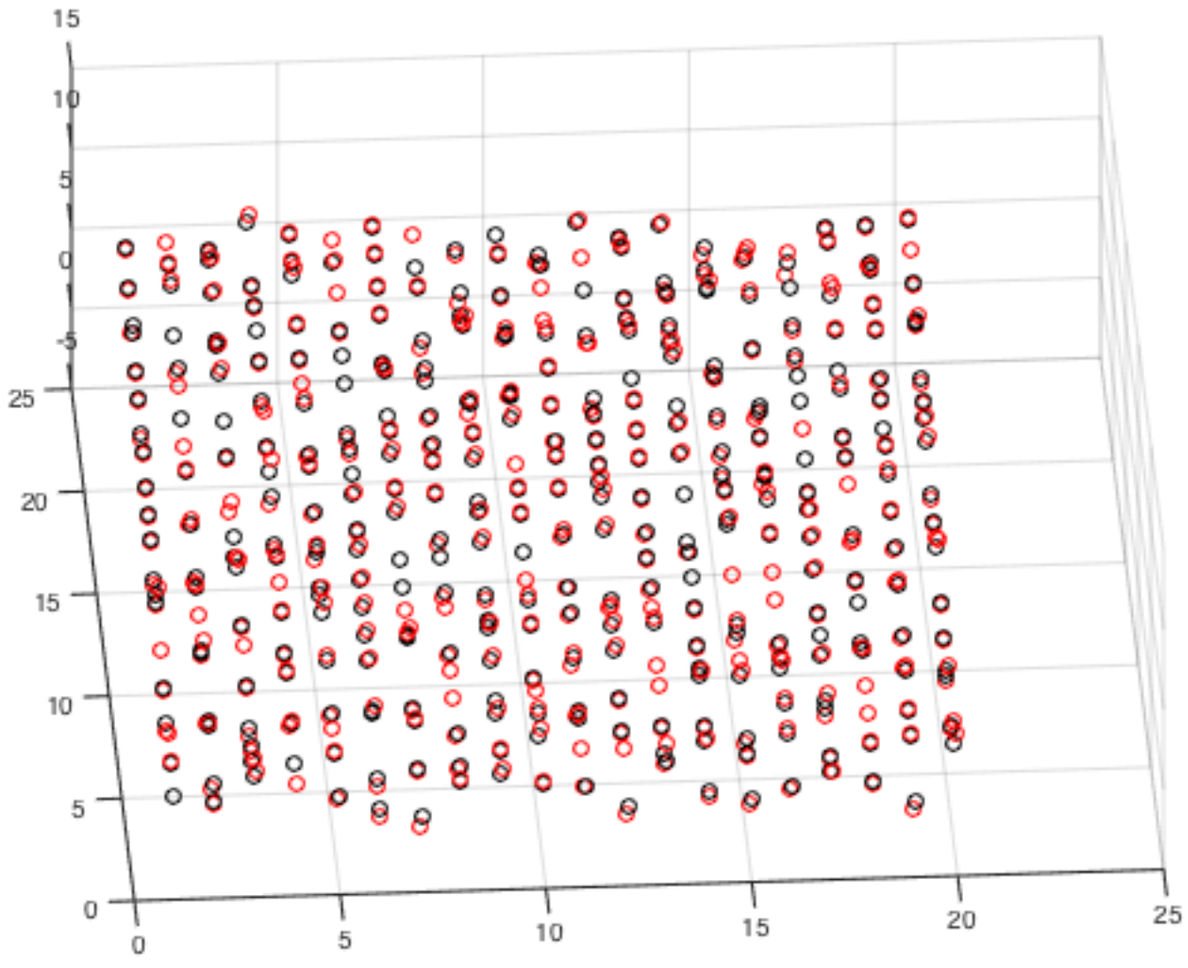
- What if we want more than one number for each data point?

- That is we want to reduce from d to K > 1 dimensions?

PCA: Variance Maximization

- How do we find the K components?

Answer : Maximize sum of spread in the K (orthogonal) directions



PCA: Variance Maximization

Intuition: Remove top direction, now reduce dimension for remaining d-1 dimensions

- How do we find the K components?

- We are looking for orthogonal directions that maximize total spread in each direction $\mathbf{y}_t[j] = \mathbf{w}_j^\top \mathbf{x}_t$

- Find orthonormal W that maximizes $\frac{1}{n}\sum_{t=1}^{n} \text{dist}^2\left(\mathbf{y}_t, \frac{1}{n}\sum_{t=1}^{n}\mathbf{y}_t\right)$

$$\sum_{j=1}^{K}\frac{1}{n}\sum_{t=1}^{n}\left(\mathbf{y}_t[j] - \frac{1}{n}\sum_{t=1}^{n}\mathbf{y}_t[j]\right)^2 = \sum_{j=1}^{K}\frac{1}{n}\sum_{t=1}^{n}\left(\mathbf{w}_j^\top\left(\mathbf{x}_t - \frac{1}{n}\sum_{t=1}^{n}\mathbf{x}_t\right)\right)^2$$

$$= \sum_{j=1}^{K}\mathbf{w}_j^\top \Sigma \mathbf{w}_j$$

- This solutions is given by W = Top K eigenvectors of Σ

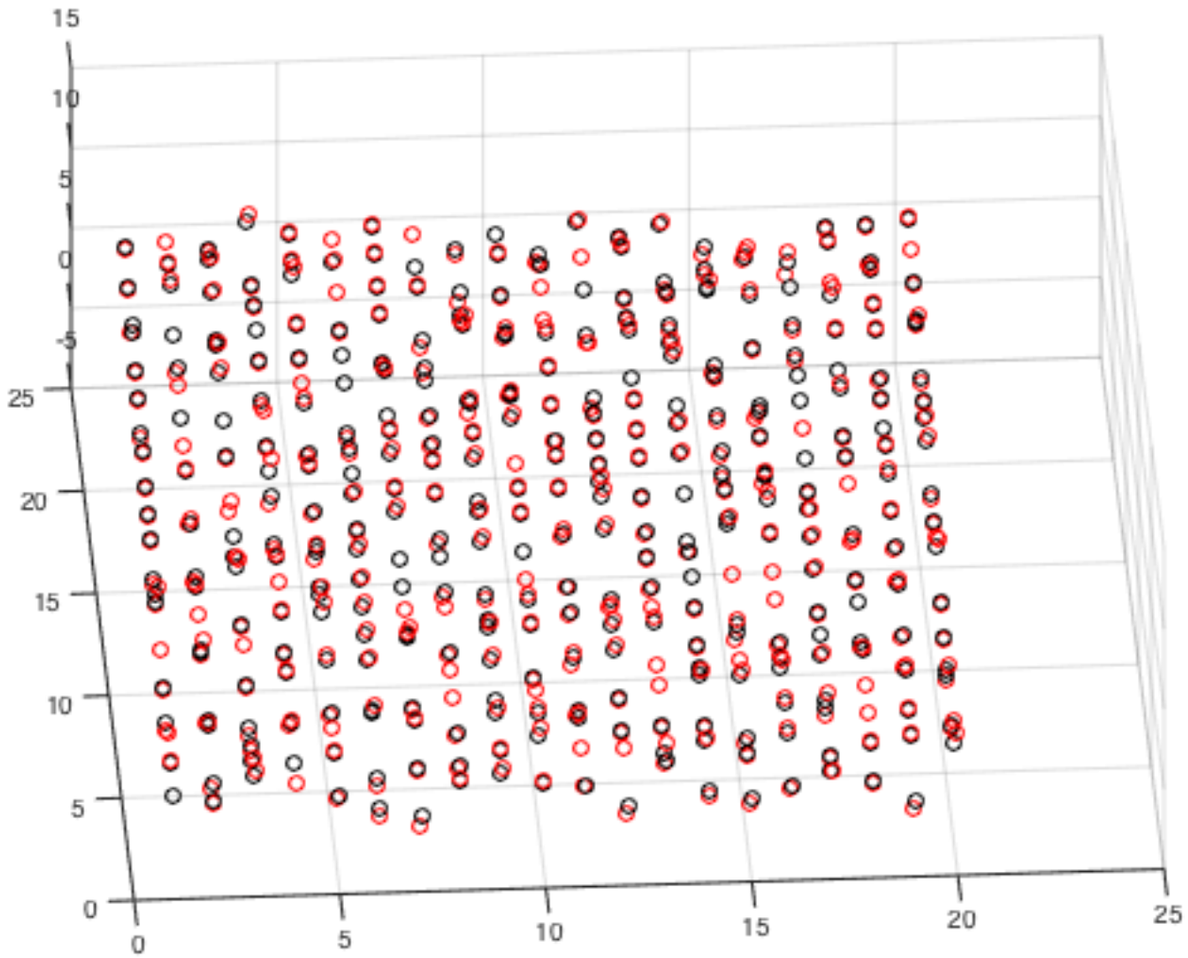
Principal Component Analysis

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

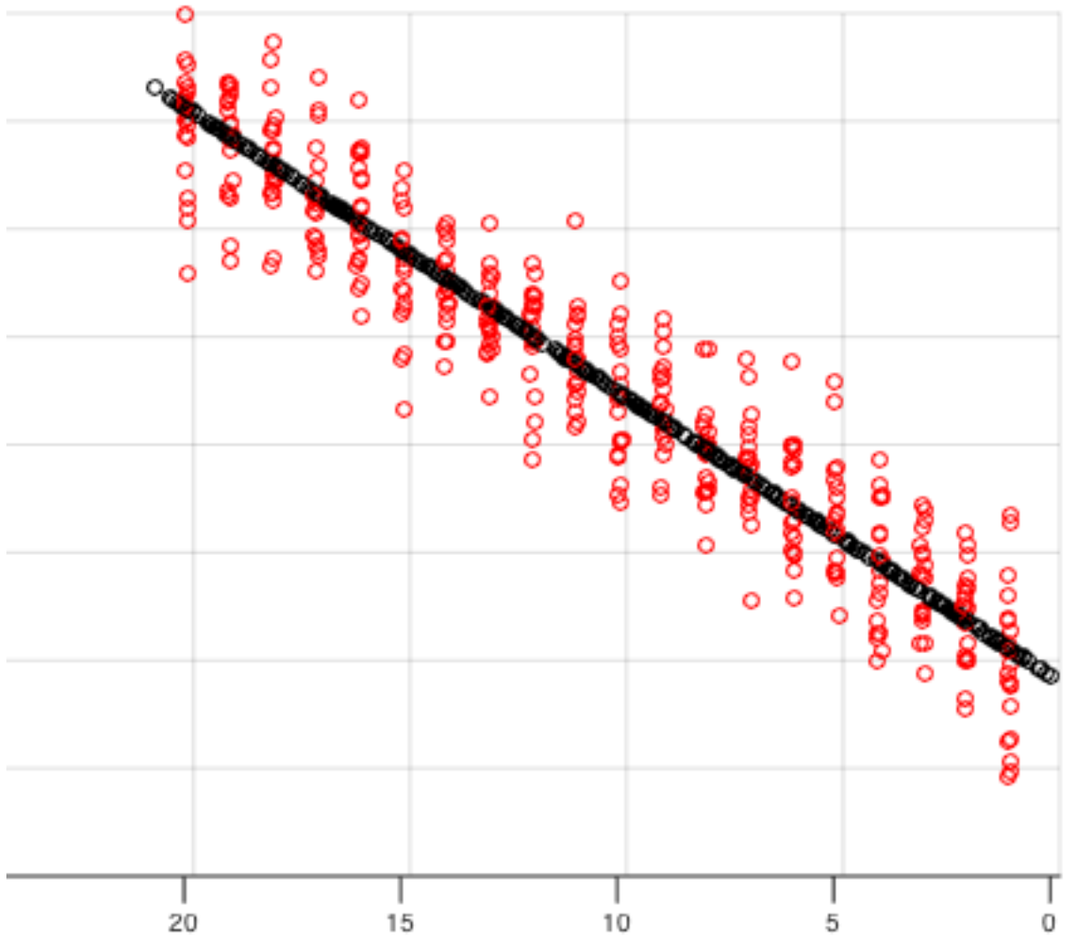
2. $$W = \text{eigs}\left(\Sigma, K\right)$$

3. $$Y = x \times W$$

Maximize Total Spread



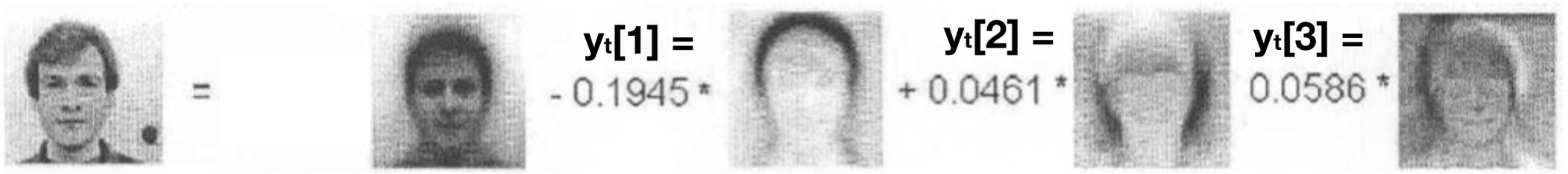
Minimize Reconstruction Error



Principal Component Analysis (PCA)

Turk & Pentland'91

Eigen Face:



- Each x_t (each row of X) is a face image (vectorized version)
- Each y_t is the set of coefficients we multiply to the eigen face
- w_i's are orthogonal to each other and of unit length

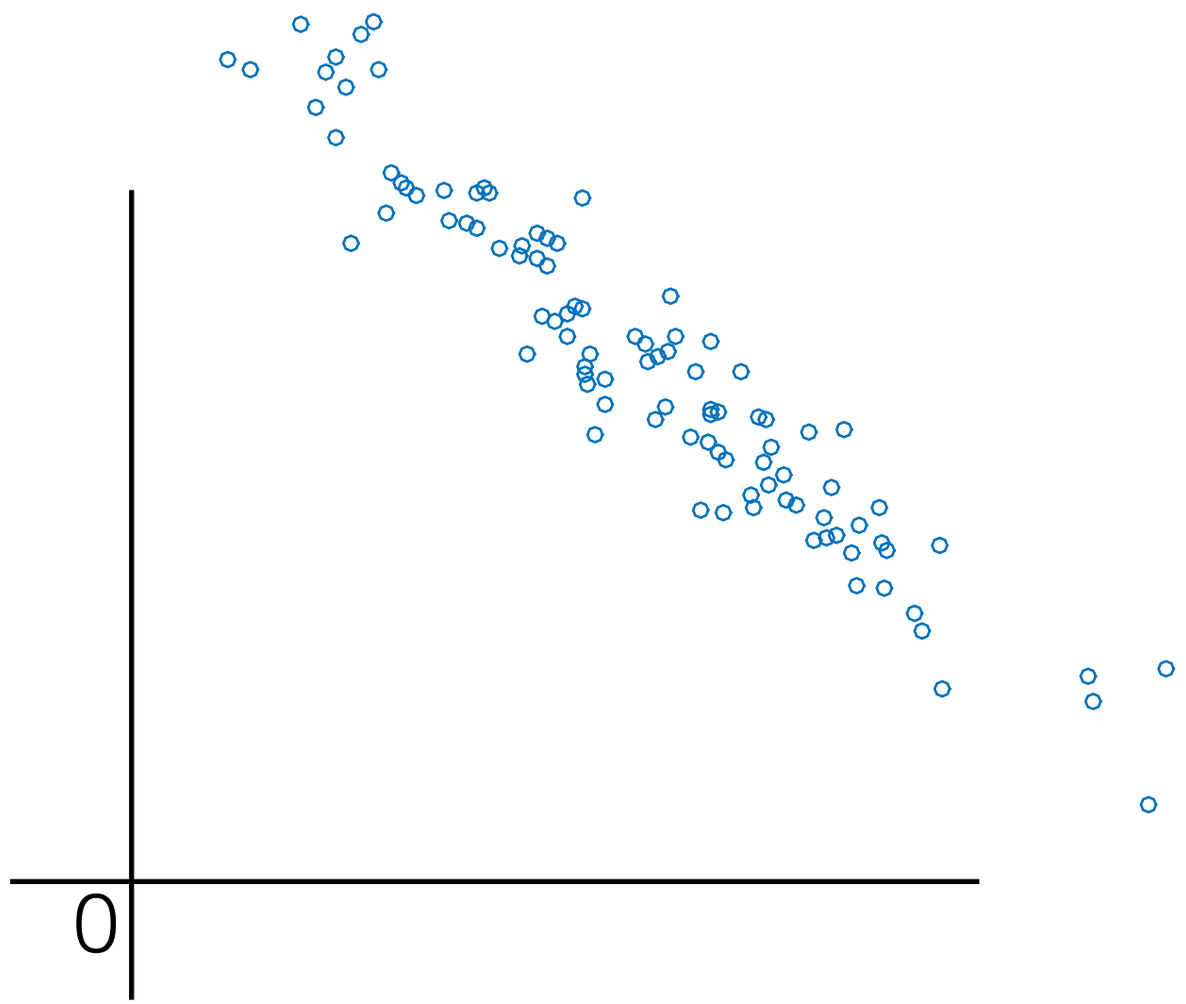
Orthonormal Projections

- Think of $\mathbf{w}_1, \ldots, \mathbf{w}_K$ as coordinate system for PCA (in a K dimensional subspace)
- $\mathbf{y}$ values provide coefficients in this system
- Without loss of generality, $\mathbf{w}_1, \ldots, \mathbf{w}_K$ can be orthonormal, i.e. $\mathbf{w}_i \perp \mathbf{w}_j$ & $\|\mathbf{w}_i\| = 1$.

$$\|\mathbf{w}_i\|_2^2 = \sum_{k=1}^{d} \mathbf{w}_i[k]^2$$

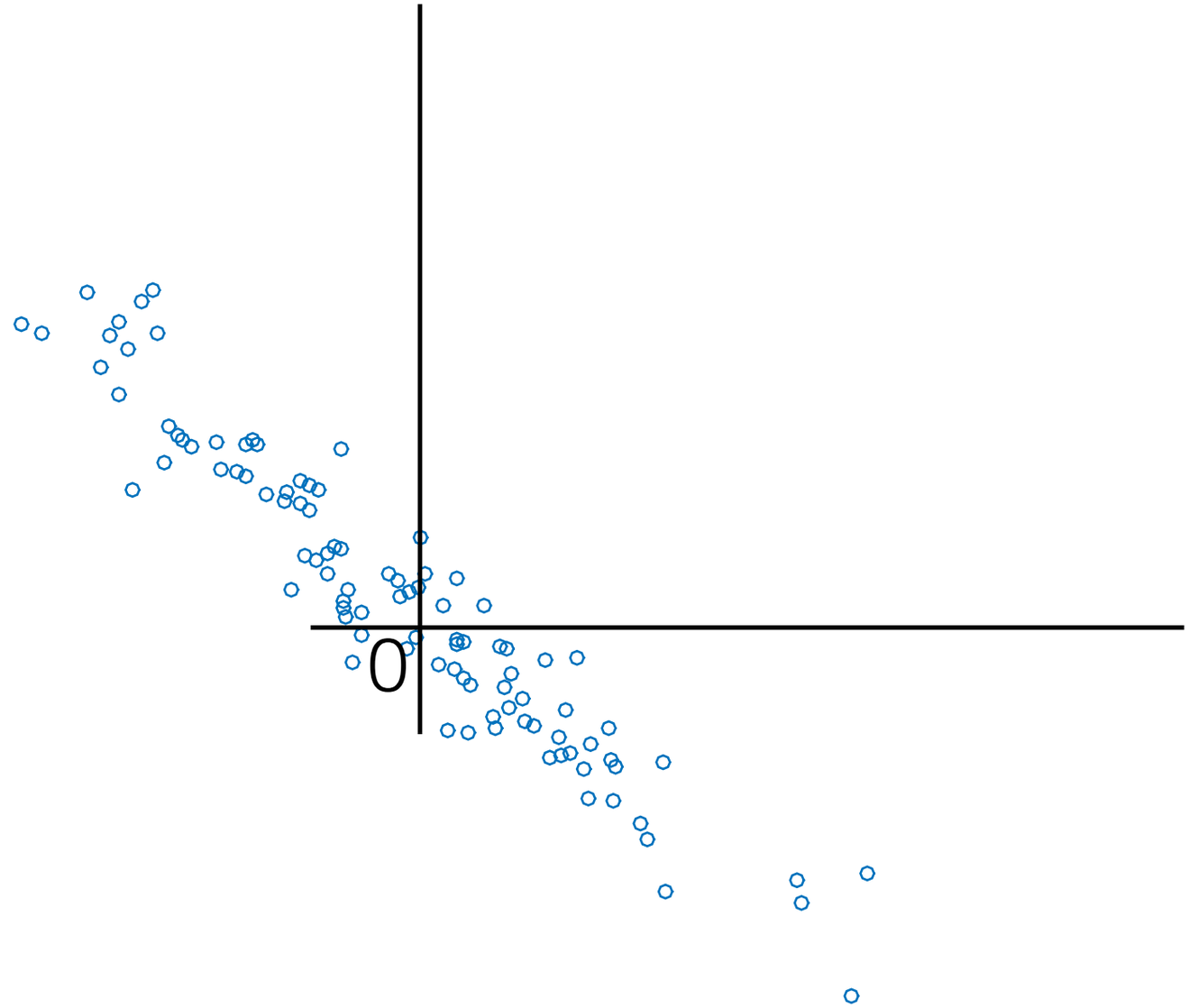
$$\mathbf{w}_i \perp \mathbf{w}_j \Rightarrow \sum_{k=1}^{d} \mathbf{w}_i[k]\mathbf{w}_j[k] = 0$$

Centering Data



Compressing these data points…

Centering Data



… is same as compressing these.

Orthonormal Projections

- (Centered) Data-points as linear combination of some orthonormal basis, i.e.

$$\mathbf{x}_t = \mu + \sum_{j=1}^{d} \mathbf{y}_t[j]\mathbf{w}_j$$

where $\mathbf{w}_1, \ldots, \mathbf{w}_d \in \mathbb{R}^d$ are the orthonormal basis and $\mu = \frac{1}{n}\sum_{t=1}^{n} x_t$.

- Represent data as linear combination of just K orthonormal basis,

$$\hat{\mathbf{x}}_t = \mu + \sum_{j=1}^{K} \mathbf{y}_t[j]\mathbf{w}_j$$

PCA: Minimizing Reconstruction Error

- Goal: find the basis that minimizes reconstruction error,

$$\|\hat{\mathbf{x}}_t - \mathbf{x}_t\|_2^2 = \left\| \sum_{j=1}^{K} \mathbf{y}_t[j]\mathbf{w}_j + \mu - \mathbf{x}_t \right\|_2^2$$

$$= \left\| \sum_{j=1}^{K} \mathbf{y}_t[j]\mathbf{w}_j + \mu - \sum_{j=1}^{d} \mathbf{y}_t[j]\mathbf{w}_j - \mu \right\|_2^2$$

$$= \left\| \sum_{j=K+1}^{d} \mathbf{y}_t[j]\mathbf{w}_j \right\|_2^2 \quad (\text{but } \|a+b\|_2^2 = \|a\|_2^2 + \|b\|_2^2 + 2a^\top b)$$

$$= \left(\sum_{j=K+1}^{d} \mathbf{y}_t[j]^2 \|\mathbf{w}_j\|_2^2 + 2 \sum_{j=K+1}^{d} \sum_{i=j+1}^{d} \mathbf{y}_t[j]\mathbf{y}_t[i]\mathbf{w}_j^\top \mathbf{w}_i \right)$$

$$= \sum_{j=k+1}^{d} \mathbf{y}_t[j]^2 \|\mathbf{w}_j\|_2^2 \quad (\text{last step because } \mathbf{w}_j \perp \mathbf{w}_i)$$

PCA: Minimizing Reconstruction Error

$$\frac{1}{n}\sum_{t=1}^{n}\|\hat{\mathbf{x}}_t - \mathbf{x}_t\|_2^2 = \frac{1}{n}\sum_{t=1}^{n}\sum_{j=k+1}^{d}\mathbf{y}_t[j]^2\|\mathbf{w}_j\|_2^2 \quad (\text{but } \|\mathbf{w}_j\| = 1)$$

$$= \frac{1}{n}\sum_{t=1}^{n}\sum_{j=k+1}^{d}\mathbf{y}_t[j]^2 \quad (\text{now } \mathbf{y}_j = \mathbf{w}_j^\top(\mathbf{x}_t - \mu))$$

$$= \frac{1}{n}\sum_{t=1}^{n}\sum_{j=k+1}^{d}(\mathbf{w}_j^\top(\mathbf{x}_t - \mu))^2$$

$$= \frac{1}{n}\sum_{t=1}^{n}\sum_{j=k+1}^{d}\mathbf{w}_j^\top(\mathbf{x}_t - \mu)(\mathbf{x}_t - \mu)^\top\mathbf{w}_j$$

$$= \sum_{j=k+1}^{d}\mathbf{w}_j^\top\Sigma\mathbf{w}_j$$

PCA: Minimizing Reconstruction Error

$$\text{Claim: } \sum_{j=1}^{d} \mathbf{w}_j^\top \Sigma \mathbf{w}_j = \text{ Constant } = \frac{1}{n} \sum_{t=1}^{n} \|x_t - \mu\|_2^2$$

$$\text{Recall that: } \frac{1}{n} \sum_{t=1}^{n} \|\hat{\mathbf{x}}_t - \mathbf{x}_t\|_2^2 = \sum_{j=K+1}^{d} \mathbf{w}_j^\top \Sigma \mathbf{w}_j$$

$$\text{Take K} = 0 \text{ so that } \hat{\mathbf{x}}_t = \mu$$

PCA: Minimizing Reconstruction Error

Minimize w.r.t. $\mathbf{w}_1, \ldots, \mathbf{w}_K$'s that are orthonormal,

$$\begin{aligned}
&\underset{\forall j,\ \|\mathbf{w}_j\|_2=1}{\operatorname{argmin}} \sum_{j=k+1}^{d} \mathbf{w}_j^\top \Sigma \mathbf{w}_j \\
&= \underset{\forall j,\ \|\mathbf{w}_j\|_2=1}{\operatorname{argmin}} \left(\frac{1}{n} \sum_{t=1}^{n} \|\hat{\mathbf{x}}_t - \mu\|_2^2 - \sum_{j=1}^{K} \mathbf{w}_j^\top \Sigma \mathbf{w}_j \right) \\
&= \underset{\|\mathbf{w}_j\|_2=1, \mathbf{w}_j \perp \mathbf{w}_k}{\operatorname{argmax}} \sum_{j=1}^{K} \mathbf{w}_j^\top \Sigma \mathbf{w}_j
\end{aligned}$$

Maximize Total Spread = Minimize Reconstruction Error

Principal Component Analysis

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

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

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

Reconstruction

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

When $d >> n$

- If $d >> n$ then Σ is large
- But we only need top K eigen vectors.
- Idea: use SVD

$$X - \mu = UDV^T$$

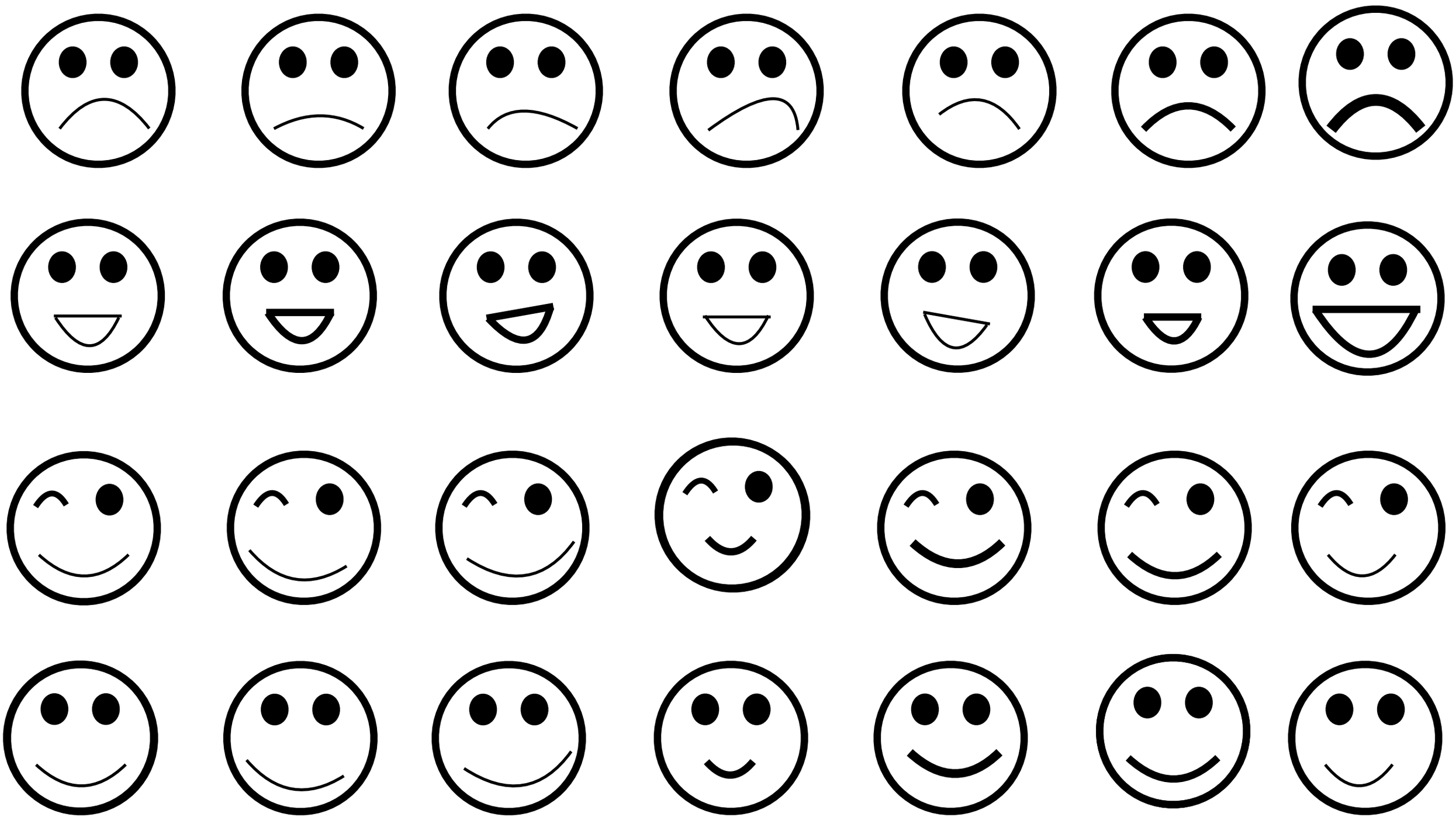
$$V^T V = I$$
$$U^T U = I$$

Then note that, $\Sigma = (X - \mu)^T (X - \mu) = VD^2V$

- Hence, matrix V is the same as matrix W got from eigen decomposition of Σ, eigenvalues are diagonal elements of D^2
- Alternative algorithm:

$$[U, V] = \text{SVD}(X - \mu, K) \quad W = V$$

Principal Component Analysis: Demo

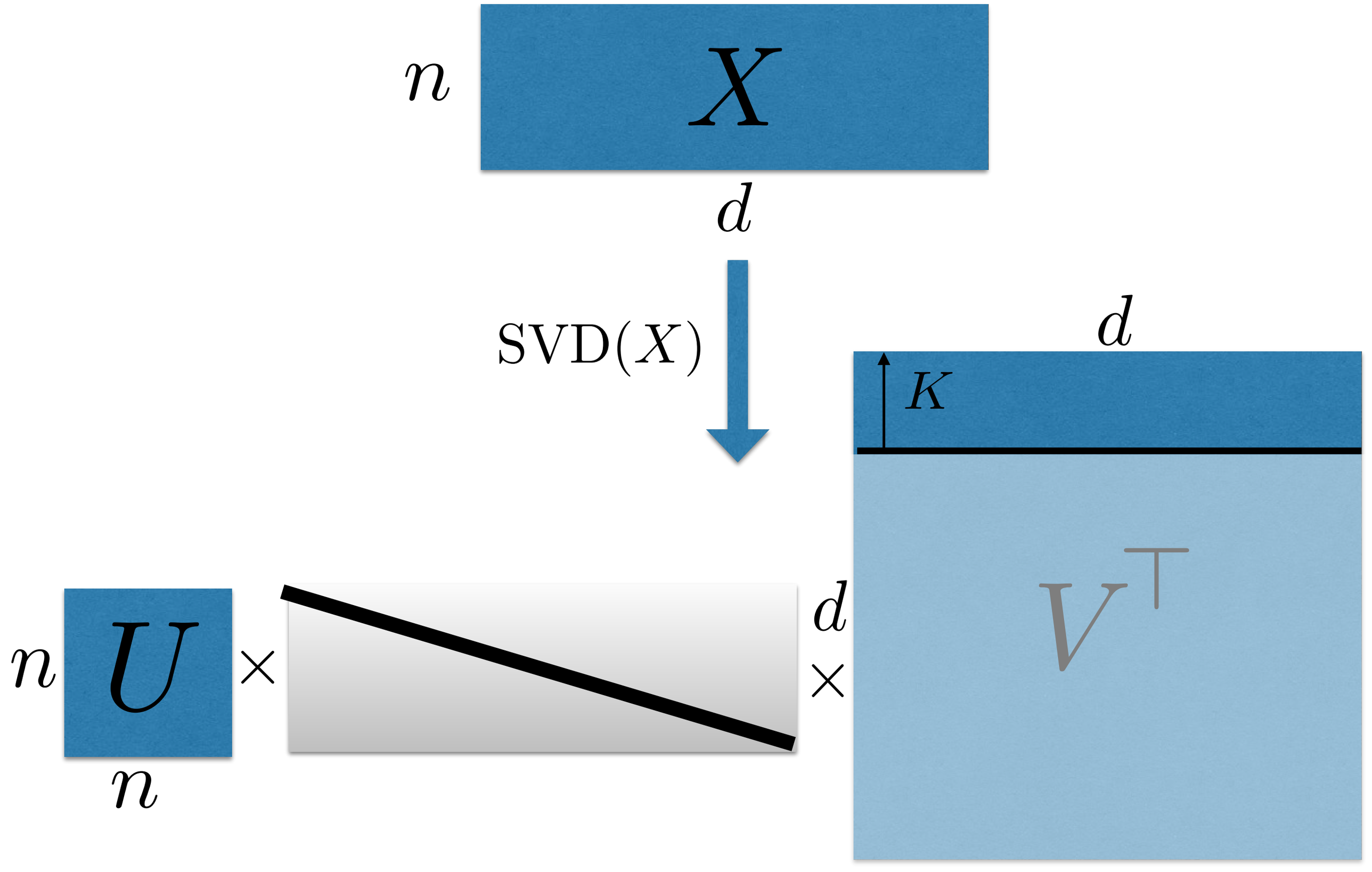


The Tall, the Fat and the Ugly

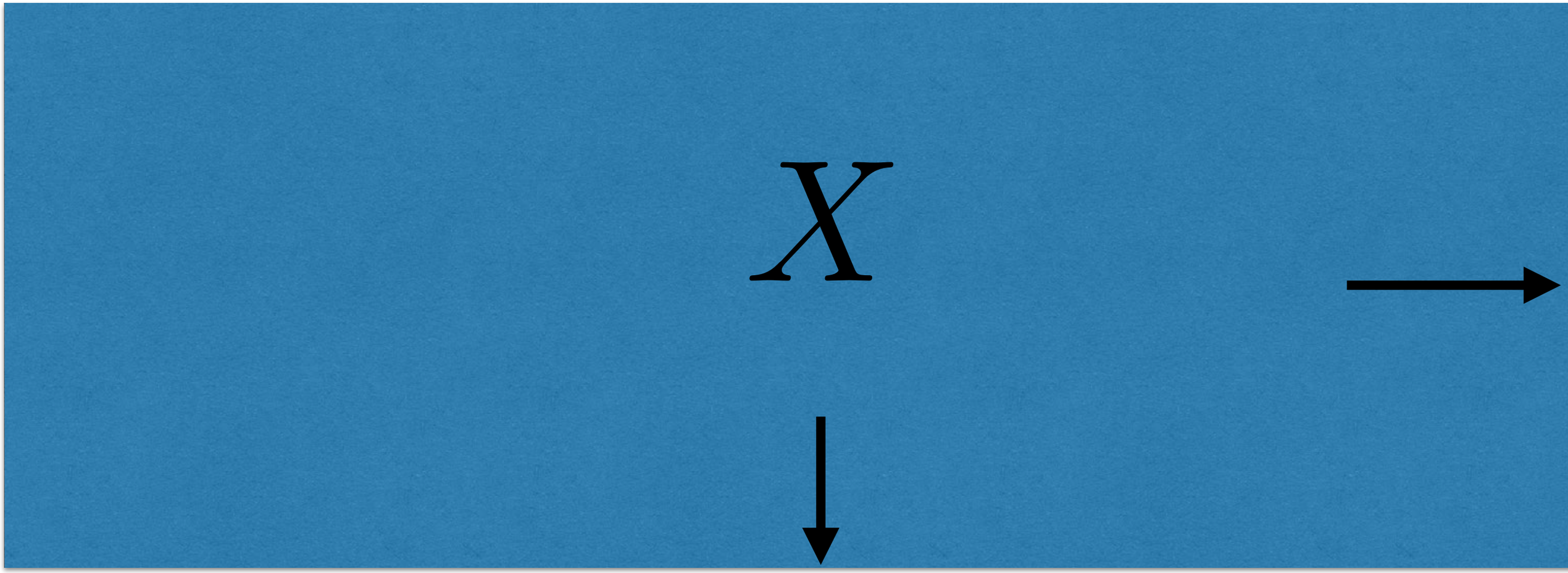
$$\underset{d}{\square}\underset{n}{\square^\top} \times \underset{n}{\square}\underset{d}{\square} \Big/ n = \underset{d}{\square}$$

$$\underset{d}{\square}\underset{K}{\square} = \text{Eigs}\left(\square, K\right)$$

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?