

EE16B, Spring 2018 UC Berkeley EECS

Maharbiz and Roychowdhury

Lectures 8A, 8B & 9A: Overview Slides

Data Analysis

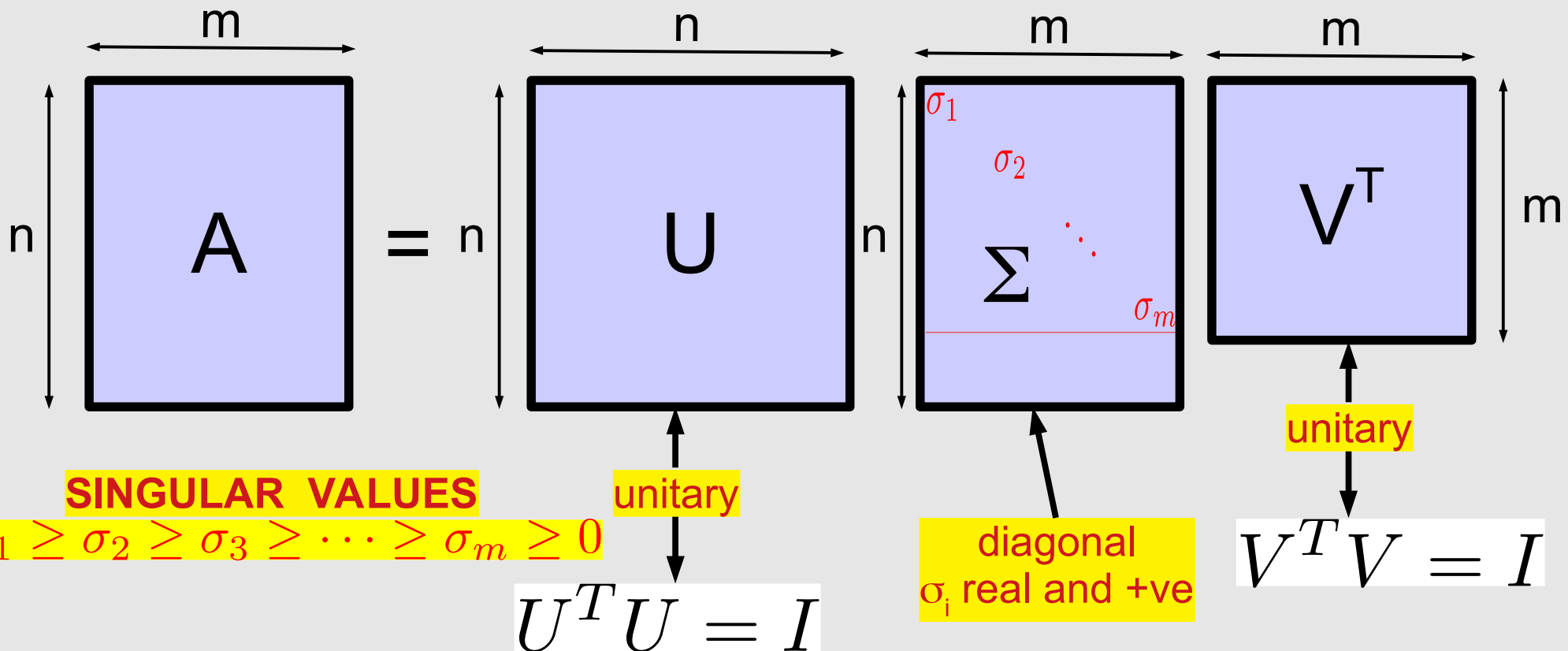
**Singular Value Decomposition
and
Principal Component Analysis**

The SVD **(Singular Value Decomposition)**

Singular Value Decomposition

- A bit like eigendecomposition, **but different**
- **Any matrix A** (no exceptions) **can be decomposed as**

$$A = U \Sigma V^T$$



Unitary Matrices: Orthonormality

U^T

U

I

$$\begin{bmatrix} \leftarrow \vec{u}_1^T \longrightarrow \\ \leftarrow \vec{u}_2^T \longrightarrow \\ \leftarrow \vec{u}_3^T \longrightarrow \\ \vdots \\ \leftarrow \vec{u}_n^T \longrightarrow \end{bmatrix} \begin{bmatrix} \uparrow \\ \uparrow \\ \uparrow \\ \vdots \\ \uparrow \\ \downarrow \\ \downarrow \\ \downarrow \\ \downarrow \end{bmatrix} \begin{bmatrix} \vec{u}_1 & \vec{u}_2 & \vec{u}_3 & \cdots & \vec{u}_n \end{bmatrix} = \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix}$$

$$\begin{array}{ccccc} \vec{u}_1^T \vec{u}_1 = 1 & \vec{u}_1^T \vec{u}_2 = 0 & \vec{u}_1^T \vec{u}_3 = 0 & \cdots & \vec{u}_1^T \vec{u}_n = 0 \\ \vec{u}_2^T \vec{u}_1 = 0 & \vec{u}_2^T \vec{u}_2 = 1 & \vec{u}_3^T \vec{u}_3 = 0 & \cdots & \vec{u}_2^T \vec{u}_n = 0 \\ & \vdots & & \vdots & \\ \vec{u}_n^T \vec{u}_1 = 0 & \vec{u}_n^T \vec{u}_2 = 0 & \vec{u}_n^T \vec{u}_3 = 0 & \cdots & \vec{u}_n^T \vec{u}_n = 1 \end{array}$$

Similarly,
 $\vec{v}_1, \vec{v}_2, \dots, \vec{v}_m$
 are **ORTHONORMAL** $\|\vec{v}_j\| = 1$

$\vec{u}_1, \vec{u}_2, \dots, \vec{u}_n$
 called **ORTHONORMAL**

$$\vec{u}_i^T \vec{u}_j = \begin{cases} 1, & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

$$\|\vec{u}_i\| = 1$$

Rank 1 Matrices and Outer Products

• Consider $A = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 3 & 3 & 3 & 3 & 3 \\ 2 & 2 & 2 & 2 & 2 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix}$

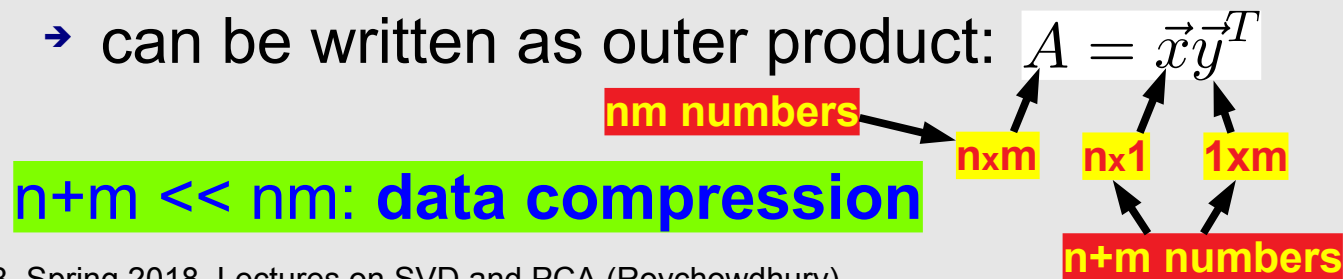
Annotations: **rank=1** points to matrix A; **col** points to the column vector $\begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix}$; **row** points to the row vector $\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix}$.

- rank-1 matrix can be written as $\vec{x}\vec{y}^T$: an **outer product**
- **outer product**: product of col and row vectors

• $\begin{bmatrix} x \\ y \\ z \end{bmatrix} \begin{bmatrix} a & b & c & d & e \end{bmatrix} = \begin{bmatrix} xa & xb & xc & xd & xe \\ ya & yb & yc & yd & ye \\ za & zb & zc & zd & ze \end{bmatrix}$

Annotation: **rank=1** points to the resulting matrix.

- **rank-1**: a very “**simple**” type of matrix
- its “**data**” can be “**compressed**” very easily
 - can be written as outer product: $A = \vec{x}\vec{y}^T$



Matrix Multiplication using Outer Products

$$\begin{array}{c} \mathbf{X} \\ \left[\begin{array}{cccc} \uparrow & \uparrow & \uparrow & \uparrow \\ \vec{x}_1 & \vec{x}_2 & \vec{x}_3 & \cdots & \vec{x}_n \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \end{array} \right] \end{array}
 \quad
 \begin{array}{c} \mathbf{Y}^T \\ \left[\begin{array}{c} \leftarrow \vec{y}_1^T \rightarrow \\ \leftarrow \vec{y}_2^T \rightarrow \\ \leftarrow \vec{y}_3^T \rightarrow \\ \vdots \\ \leftarrow \vec{y}_n^T \rightarrow \end{array} \right] \end{array}$$

each of these is a
rank-1 OUTER PRODUCT

$$= \vec{x}_1 \vec{y}_1^T + \vec{x}_2 \vec{y}_2^T + \vec{x}_3 \vec{y}_3^T + \cdots + \vec{x}_n \vec{y}_n^T$$

• Example:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x & y & z \\ p & q & r \end{bmatrix} = \begin{bmatrix} ax + bp & ay + bq & az + br \\ cx + dp & cy + dq & cz + dr \end{bmatrix}$$

$$\begin{bmatrix} a \\ c \end{bmatrix} \begin{bmatrix} x & y & z \end{bmatrix} = \begin{bmatrix} ax & ay & az \\ cx & cy & cz \end{bmatrix}$$

$$\begin{bmatrix} b \\ d \end{bmatrix} \begin{bmatrix} p & q & r \end{bmatrix} = \begin{bmatrix} bp & bq & br \\ dp & dq & dr \end{bmatrix}$$

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x & y & z \\ p & q & r \end{bmatrix} = \begin{bmatrix} a \\ c \end{bmatrix} \begin{bmatrix} x & y & z \end{bmatrix} + \begin{bmatrix} b \\ d \end{bmatrix} \begin{bmatrix} p & q & r \end{bmatrix}$$

SVD: Sum of Outer Products Form

$$\begin{aligned}
 A &= \begin{matrix} & \begin{matrix} \uparrow & \uparrow & \uparrow & & \uparrow \\ \vec{u}_1 & \vec{u}_2 & \vec{u}_3 & \cdots & \vec{u}_n \end{matrix} \\ \begin{bmatrix} & & & & \\ & & & & \\ & & & & \\ & & & & \end{bmatrix} & \begin{bmatrix} \sigma_1 & & & & \\ & \sigma_2 & & & \\ & & \sigma_3 & & \\ & & & \ddots & \\ & & & & \sigma_m \end{bmatrix} & \begin{matrix} \begin{bmatrix} \leftarrow \vec{v}_1^T \rightarrow \\ \leftarrow \vec{v}_2^T \rightarrow \\ \vdots \\ \leftarrow \vec{v}_m^T \rightarrow \end{bmatrix} \end{matrix} \\
 & \begin{matrix} \text{biggest weight} & & \text{next biggest weight} & & \text{smallest weight} \\ \downarrow & & \downarrow & & \downarrow \\ \sigma_1 & & \sigma_2 & & \sigma_m \end{matrix} \\
 &= \begin{matrix} \begin{bmatrix} \uparrow \\ \vec{u}_1 \\ \downarrow \end{bmatrix} & \begin{bmatrix} \leftarrow \vec{v}_1^T \rightarrow \\ \text{outer product} \\ n \times m \text{ rank-1} \\ \text{matrix} \\ \vec{u}_1 \vec{v}_1^T \end{bmatrix} & + & \begin{bmatrix} \uparrow \\ \vec{u}_2 \\ \downarrow \end{bmatrix} & \begin{bmatrix} \leftarrow \vec{v}_2^T \rightarrow \\ \text{outer product} \\ n \times m \text{ rank-1} \\ \text{matrix} \\ \vec{u}_2 \vec{v}_2^T \end{bmatrix} & + \dots + & \begin{bmatrix} \uparrow \\ \vec{u}_m \\ \downarrow \end{bmatrix} & \begin{bmatrix} \leftarrow \vec{v}_m^T \rightarrow \\ \text{outer product} \\ n \times m \text{ rank-1} \\ \text{matrix} \\ \vec{u}_m \vec{v}_m^T \end{bmatrix} \\
 & \begin{matrix} & \text{Frobenius norm (sqrt(sum of squares) = 1)} & \end{matrix} \end{matrix}
 \end{aligned}$$

SVD splits a matrix into a weighted sum of rank-1 matrices of norm 1

Using the SVD for Image Analysis and Compression

Example: B&W Polish Flag as a Matrix

- size: 281x450

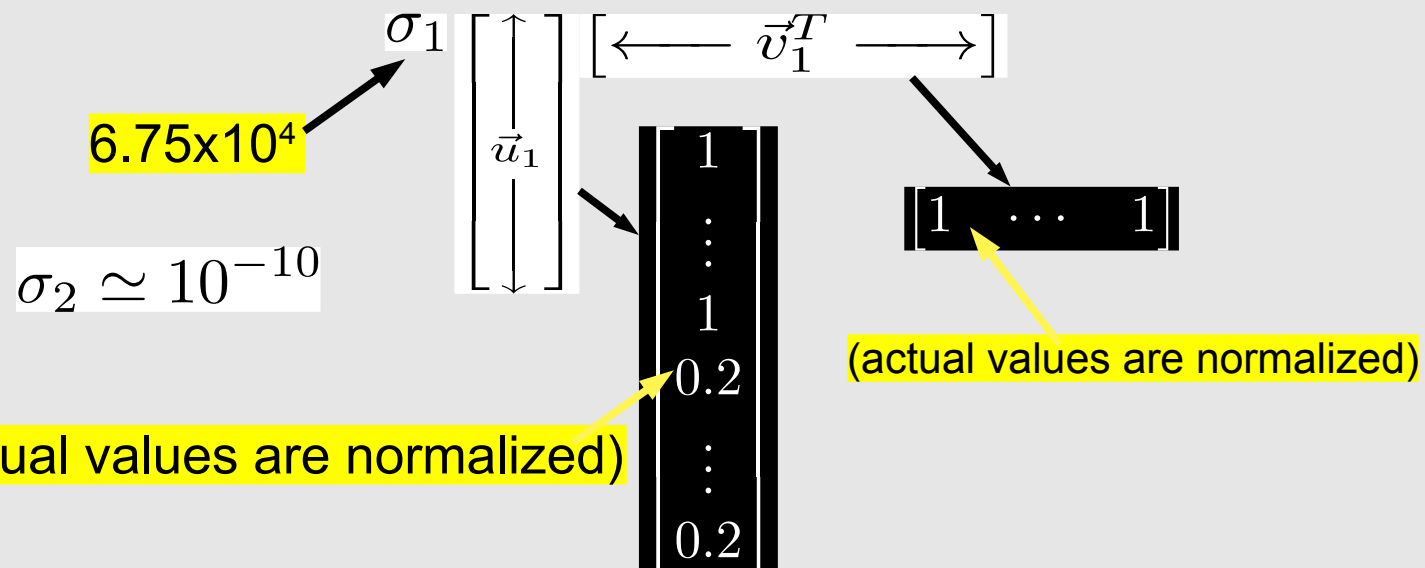
$A =$



original: 3.2MB



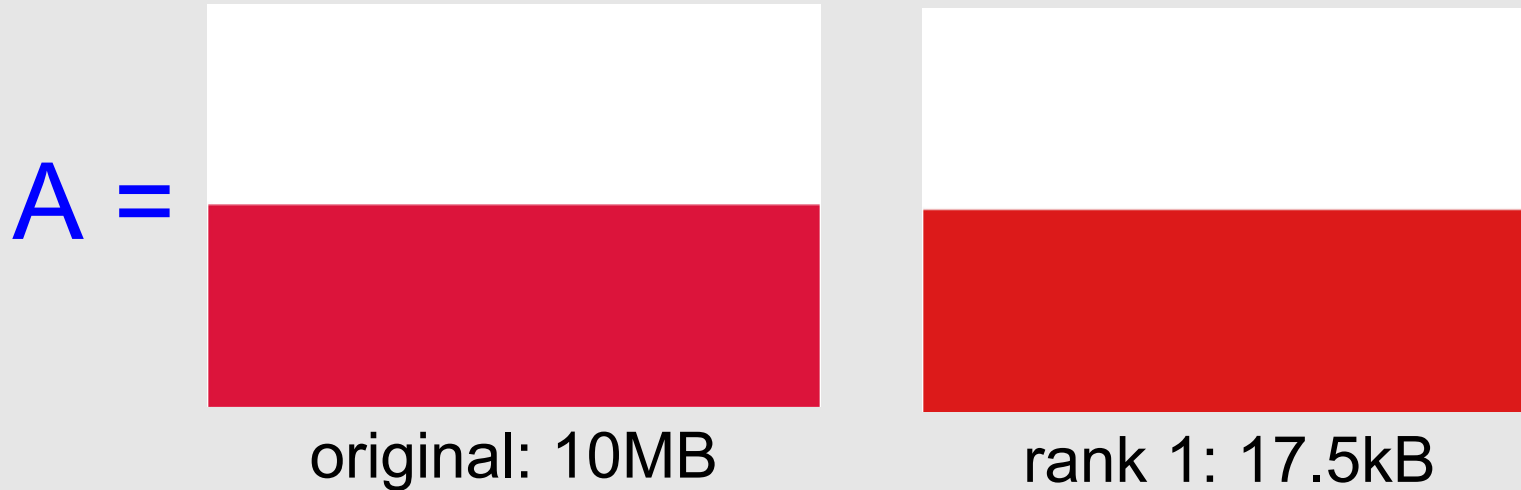
rank=1: 58kB



**This is a
RANK-1 FLAG**

Example: Polish Flag as a Matrix

- size: 281x450 (x 3 colours: R, G, B)



[illegible]

Example: SVD of the Austrian Flag

- size: 281x450 (x 3 colours: R, G, B)

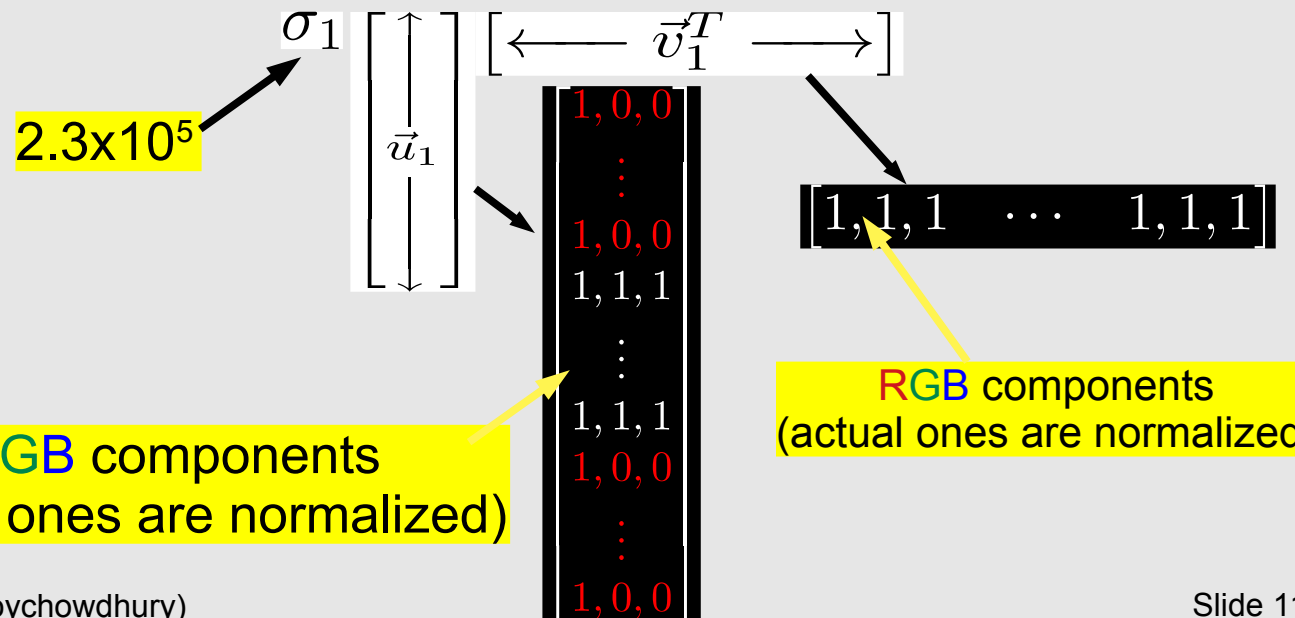
$A =$



original: 73MB



rank 1: 48.5kB



This is ALSO a RANK-1 FLAG

RGB components (actual ones are normalized)

RGB components (actual ones are normalized)

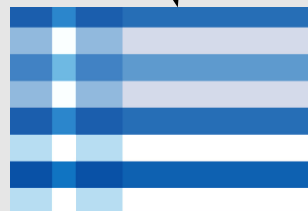
Example: SVD of the Greek Flag

- size: 295x450 (x 3 colours: R, G, B)



original: 10.1MB

strongest
“feature”



rank 1: 18kb

$$\sigma_1 \vec{u}_1 \vec{v}_1^T$$

2nd strongest
“feature”



$$\sigma_2 \vec{u}_2 \vec{v}_2^T$$

3rd strongest
“feature”



$$\sigma_3 \vec{u}_3 \vec{v}_3^T$$



rank 3: 54kb

$$\sum_{i=1}^3 \sigma_i \vec{u}_i \vec{v}_i^T$$



rank 2: 36kB

$$\sigma_1 \vec{u}_1 \vec{v}_1^T + \sigma_2 \vec{u}_2 \vec{v}_2^T$$

This is a
RANK-3 FLAG

Example: SVD of the US Flag

- size: 450x237 (x 3 colours: R, G, B)



original: 8.8MB

strongest
"feature"



$$\sigma_1 \vec{u}_1 \vec{v}_1^T$$

$$\sum_{i=1}^5 \sigma_i \vec{u}_i \vec{v}_i^T$$



rank 5: 83kB

$$\sum_{i=1}^{10} \sigma_i \vec{u}_i \vec{v}_i^T$$

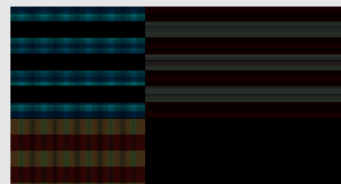


rank 10: 167kB

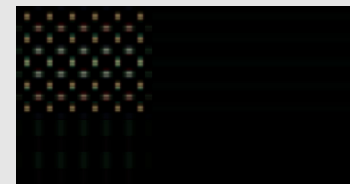
$$\sum_{i=1}^{15} \sigma_i \vec{u}_i \vec{v}_i^T$$



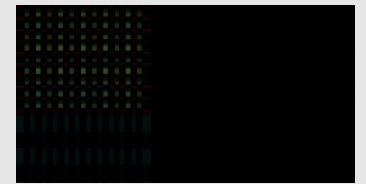
rank 15: 253kB



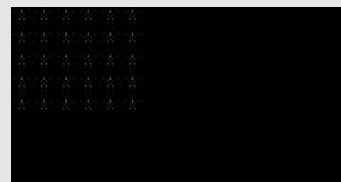
$$\sigma_2 \vec{u}_2 \vec{v}_2^T \quad 16.5\text{kb}$$



$$\sigma_3 \vec{u}_3 \vec{v}_3^T \quad 16.5\text{kB}$$



$$\sigma_4 \vec{u}_4 \vec{v}_4^T \quad 16.5\text{kB}$$



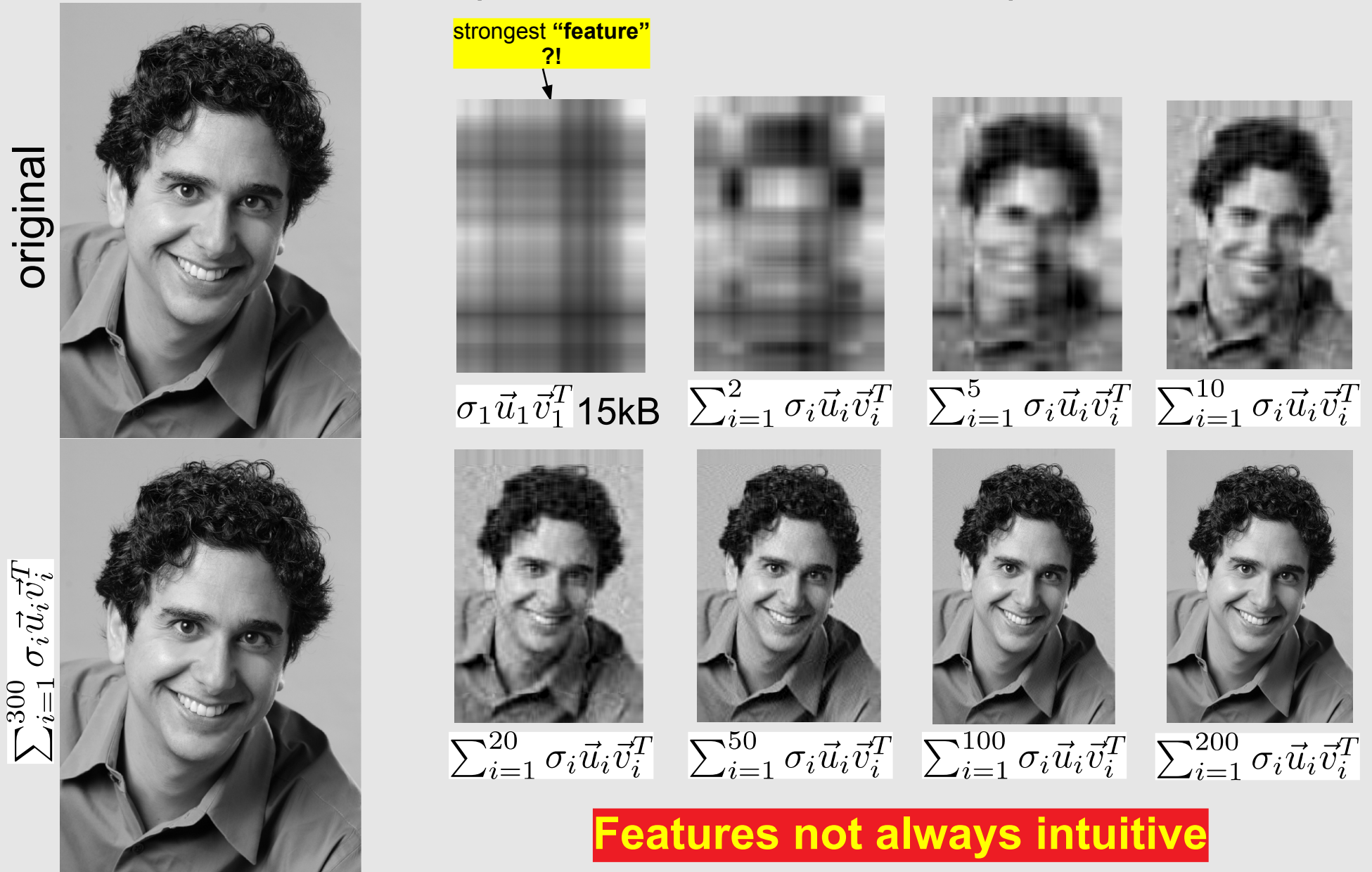
$$\sigma_5 \vec{u}_5 \vec{v}_5^T \quad 16.5\text{kB}$$



$$\sigma_6 \vec{u}_6 \vec{v}_6^T \quad 16.5\text{kB}$$

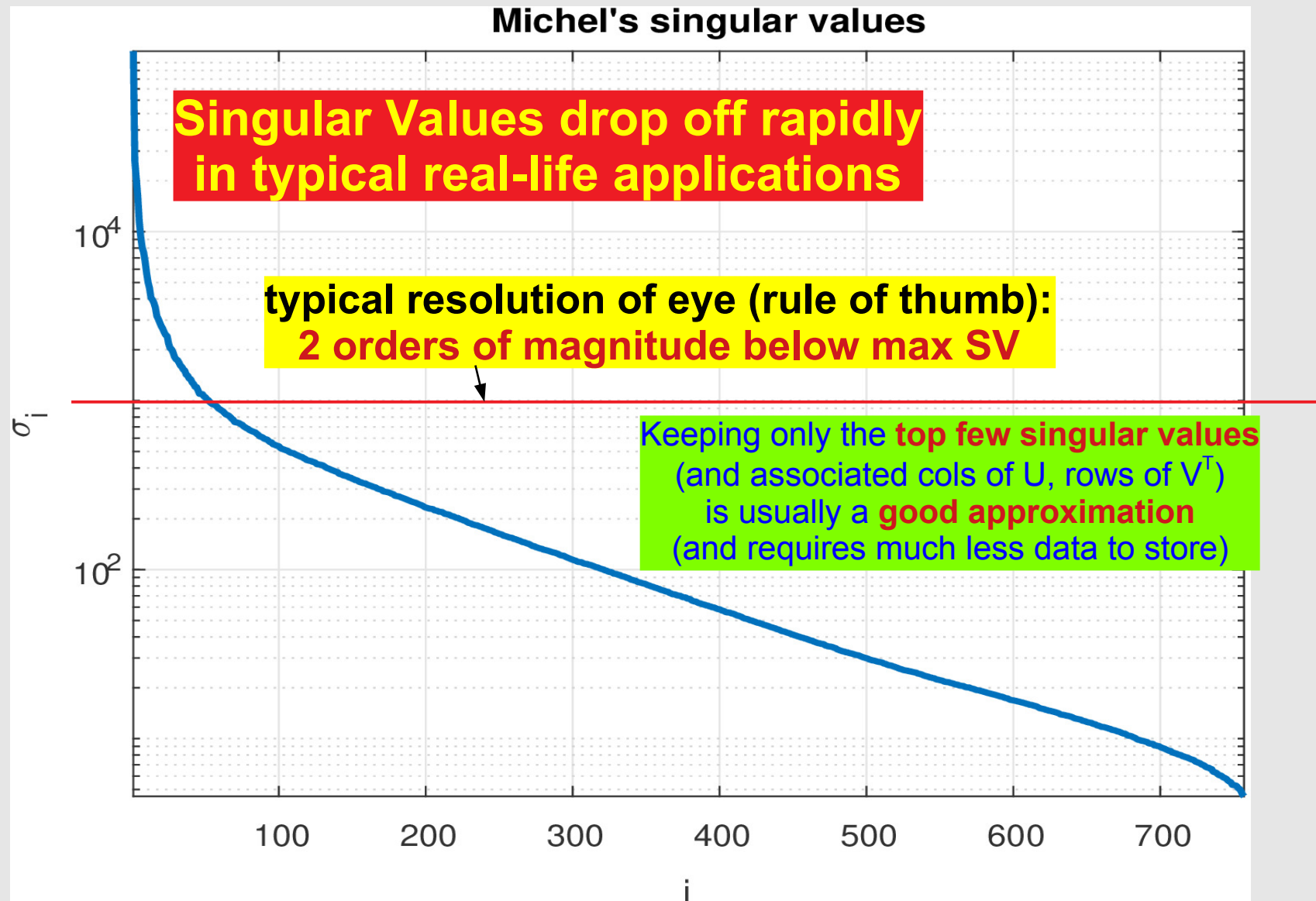
Example: SVD of Michel Maharbiz

- size: 1100x757 (x 3 colours: R, G, B)



Michel's Singular Values

- How Michel's singular values drop off



Geometric View of Orthogonality

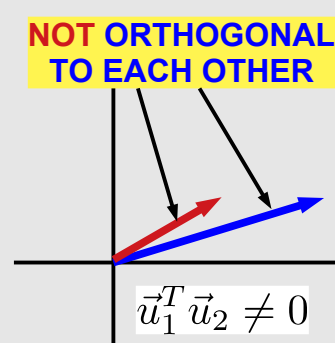
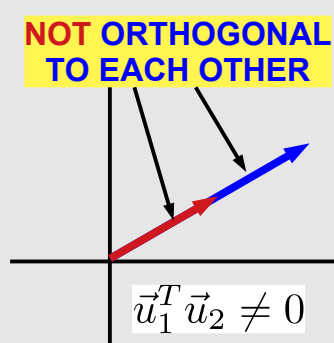
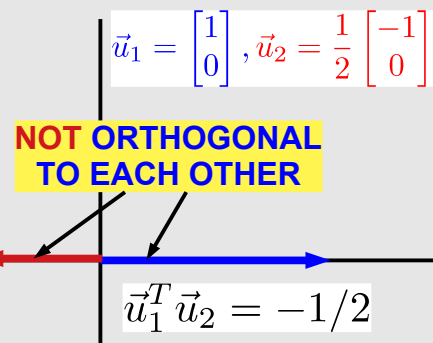
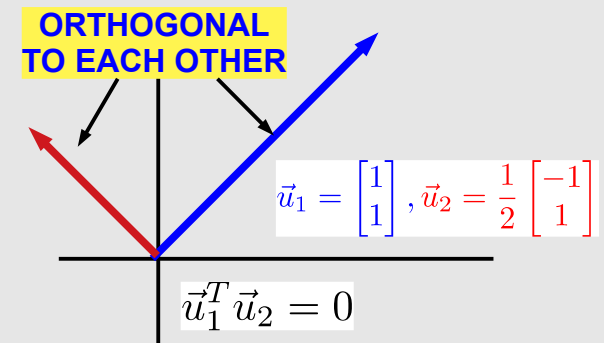
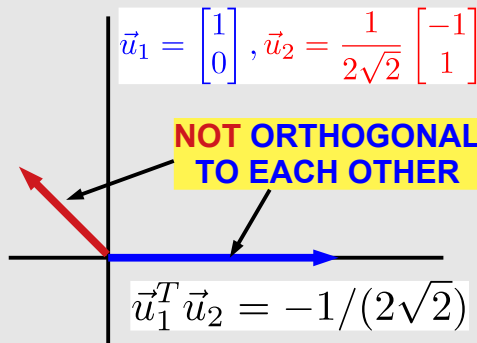
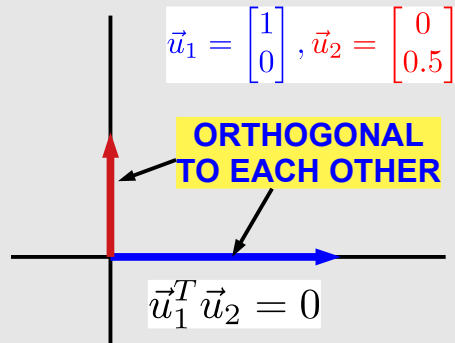
Projection onto Orthonormal Bases

Geometric View of Unitary Operations

Geometric View of Orthogonality

- recall: $\vec{u}_i^T \vec{u}_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$ if not necessarily = 1 (but $\neq 0$): then called **ORTHOGONAL**
- $\vec{u}_1, \vec{u}_2, \dots, \vec{u}_n$ called **ORTHONORMAL**

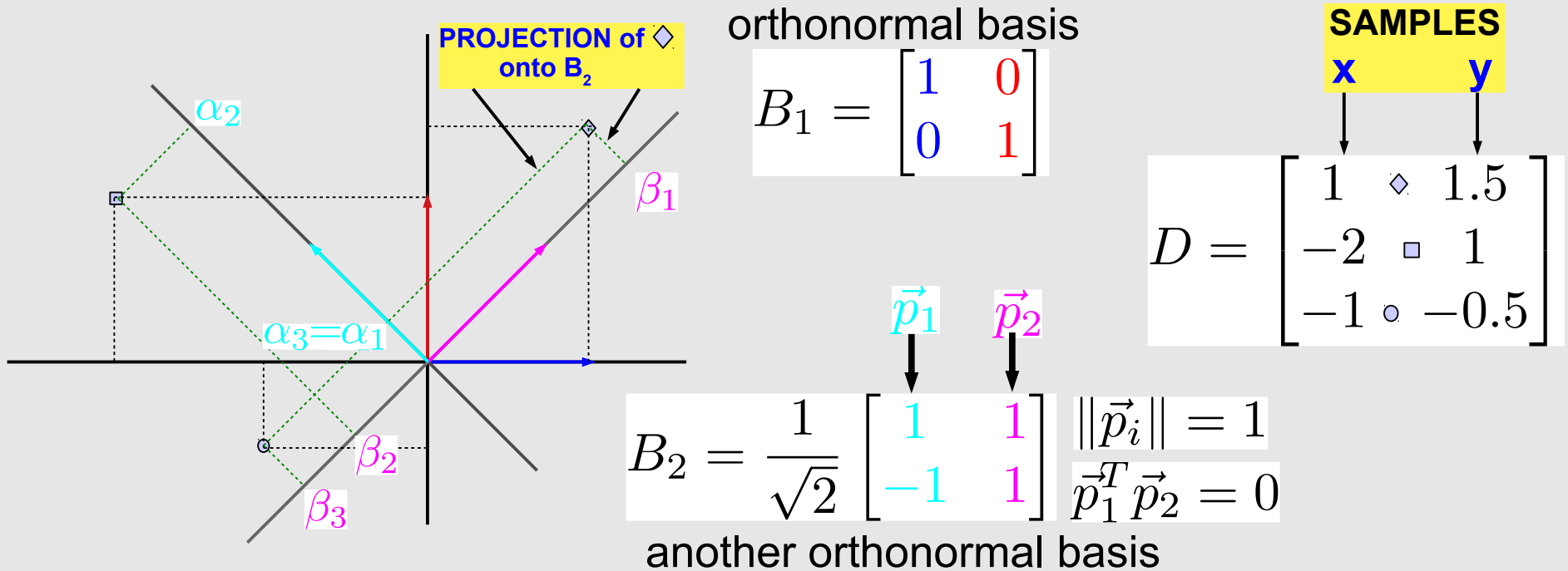
- In 2D:



3D: orthogonality also means at right angles

4D and higher: "right angles" means orthogonality!

Projection onto Orthonormal Bases



• How can we calculate the projections?

- data point: $\begin{bmatrix} x \\ y \end{bmatrix} = \alpha \vec{p}_1 + \beta \vec{p}_2$, or $\begin{bmatrix} x & y \end{bmatrix} = \alpha \vec{p}_1^T + \beta \vec{p}_2^T$
- post-multiply by basis vectors: $\begin{bmatrix} x & y \end{bmatrix} \vec{p}_1 = \alpha$, $\begin{bmatrix} x & y \end{bmatrix} \vec{p}_2 = \beta$

→ or: $\begin{bmatrix} \alpha & \beta \end{bmatrix} = \begin{bmatrix} x & y \end{bmatrix} B_2$; or, for all the data

$$D_2 = \begin{bmatrix} \alpha_1 & \beta_1 \\ \alpha_2 & \beta_2 \\ \alpha_3 & \beta_3 \end{bmatrix} = DB_2$$

projecting the data D
onto the basis B_2

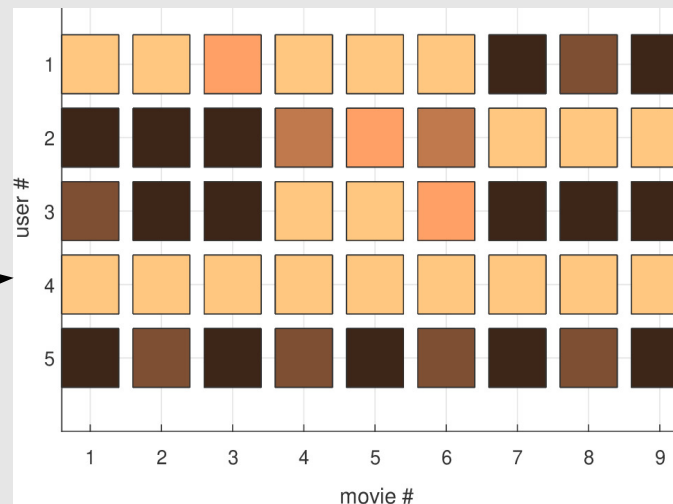
Using the SVD for Data Analysis, Feature Extraction and Clustering

Matrices Representing Ratings

- **Movies rated by Users** (eg, Netflix, Amazon Video)

Movie → User Name ↓	Full Metal Jacket	Die Hard	Yojimbo	2001: A Space Odyssey	The Quiet Earth	On The Beach	Would I Lie to You	Dr. Strangelove	Hokkabaz
A	5	5	4	5	5	5	1	2	1
B	1	1	1	3	4	3	5	5	5
C	2	1	1	5	5	4	2	1	1
D	5	5	5	5	5	5	5	5	5
E	1	2	1	2	1	2	2	2	1

lighter colours
=
stronger ratings



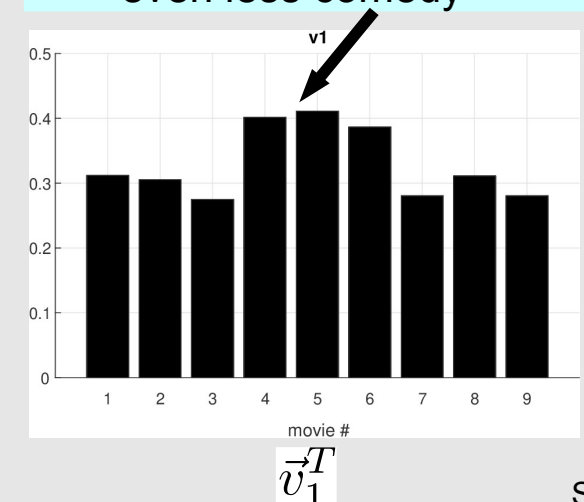
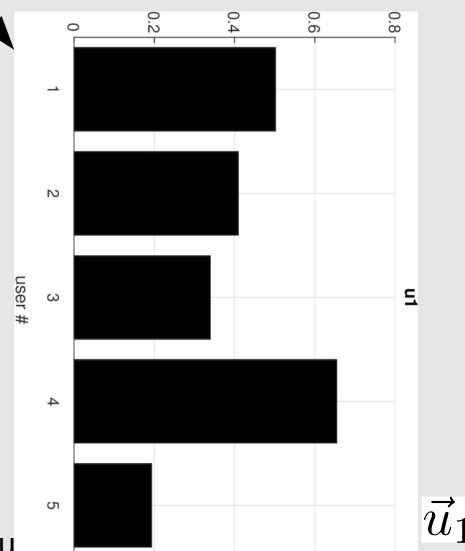
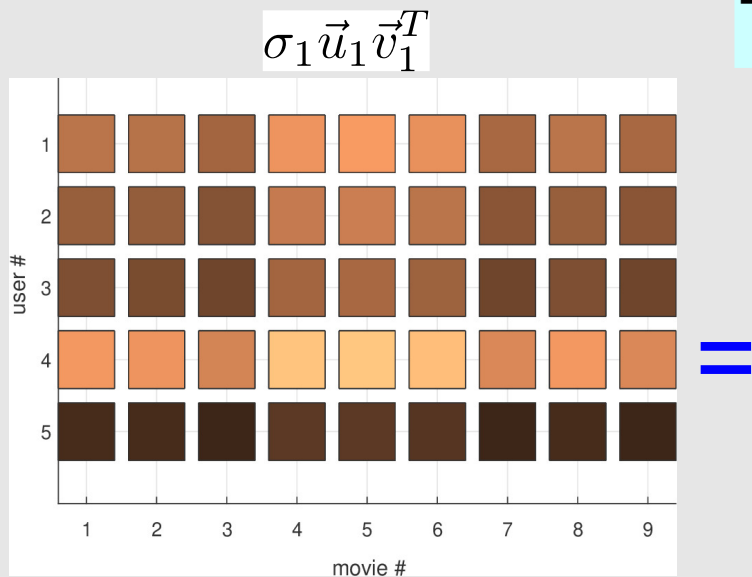
$$= U \Sigma V^T$$

Features of Rating Matrices

Movie → User Name ↓	Full Metal Jacket	Die Hard	Yojimbo	2001:A Space Odyssey	The Quiet Earth	On The Beach	Would I Lie to You	Dr. Strangelove	Hokkabaz
A	5	5	4	5	5	5	1	2	1
B	1	1	1	3	4	3	5	5	5
C	2	1	1	5	5	4	2	1	1
D	5	5	5	5	5	5	5	5	5
E	1	2	1	2	1	2	2	2	1

“most typical” user feature:
65% are like D, 50% are like A,
40% are like B, 35% are like C
20% like E

“most typical” movie feature:
more SF;
rather less action;
even less comedy

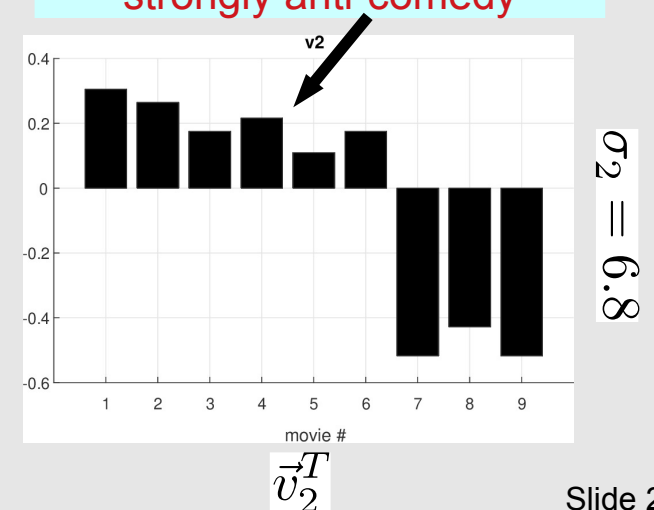
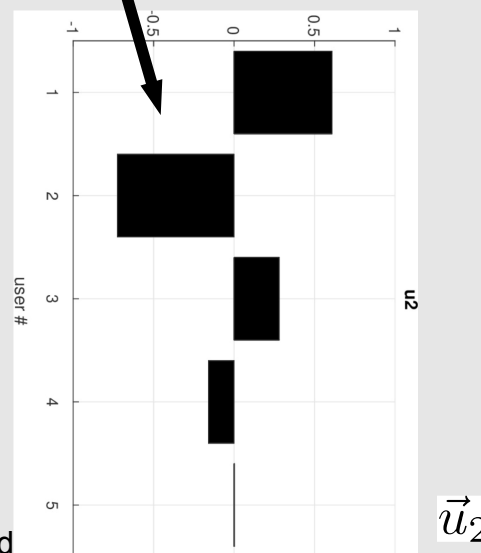
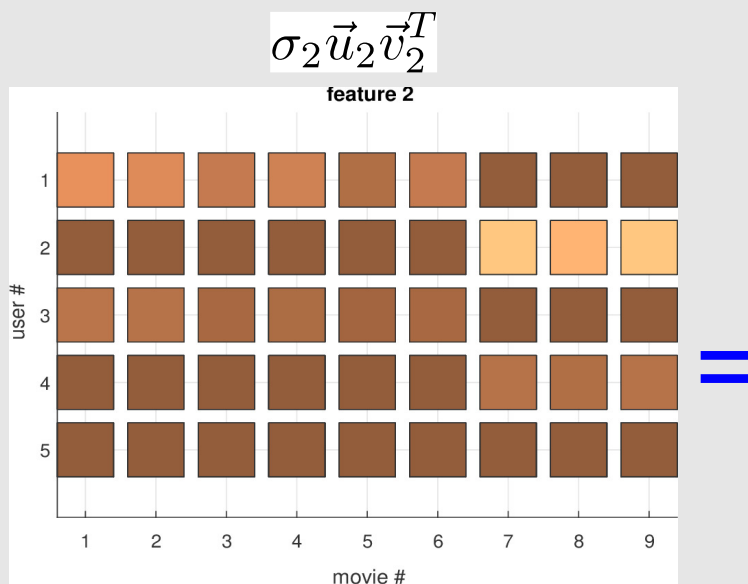


Features of Rating Matrices (contd.)

Movie → User Name ↓	Full Metal Jacket	Die Hard	Yojimbo	2001: A Space Odyssey	The Quiet Earth	On The Beach	Would I Lie to You	Dr. Strangelove	Hokkabaz
A	5	5	4	5	5	5	1	2	1
B	1	1	1	3	4	3	5	5	5
C	2	1	1	5	5	4	2	1	1
D	5	5	5	5	5	5	5	5	5
E	1	2	1	2	1	2	2	2	1

2nd most typical user feature:
55% are like A, 70% unlike B,
35% are like C, 15% unlike C
negligibly like E

2nd most typical movie feature:
more action;
less SF;
strongly anti-comedy



Projection in the Feature Basis

Movie → User Name ↓	Full Metal Jacket	Die Hard	Yojimbo	2001: A Space Odyssey	The Quiet Earth	On The Beach	Would I Lie to You	Dr. Strangelove	Hokkabaz
A	5	5	4	5	5	5	1	2	1
B	1	1	1	3	4	3	5	5	5
C	2	1	1	5	5	4	2	1	1
D	5	5	5	5	5	5	5	5	5
E	1	2	1	2	1	2	2	2	1

- Express each col of A (movie column) as a linear combination of user features
- e.g., **Full Metal Jacket** column:

$$\rightarrow \alpha_{i1} = \vec{u}_i^T \begin{bmatrix} 5 \\ 1 \\ 2 \\ 5 \\ 1 \end{bmatrix}, \quad i = 1, \dots, 5$$

**Projection of FMJ col.
onto user feature basis**

$$\begin{bmatrix} 5 \\ 1 \\ 2 \\ 5 \\ 1 \end{bmatrix} = \alpha_{11} \vec{u}_1 + \alpha_{21} \vec{u}_2 + \alpha_{31} \vec{u}_3 + \dots + \alpha_{51} \vec{u}_5$$

user features

“how much the most typical user likes FMJ”

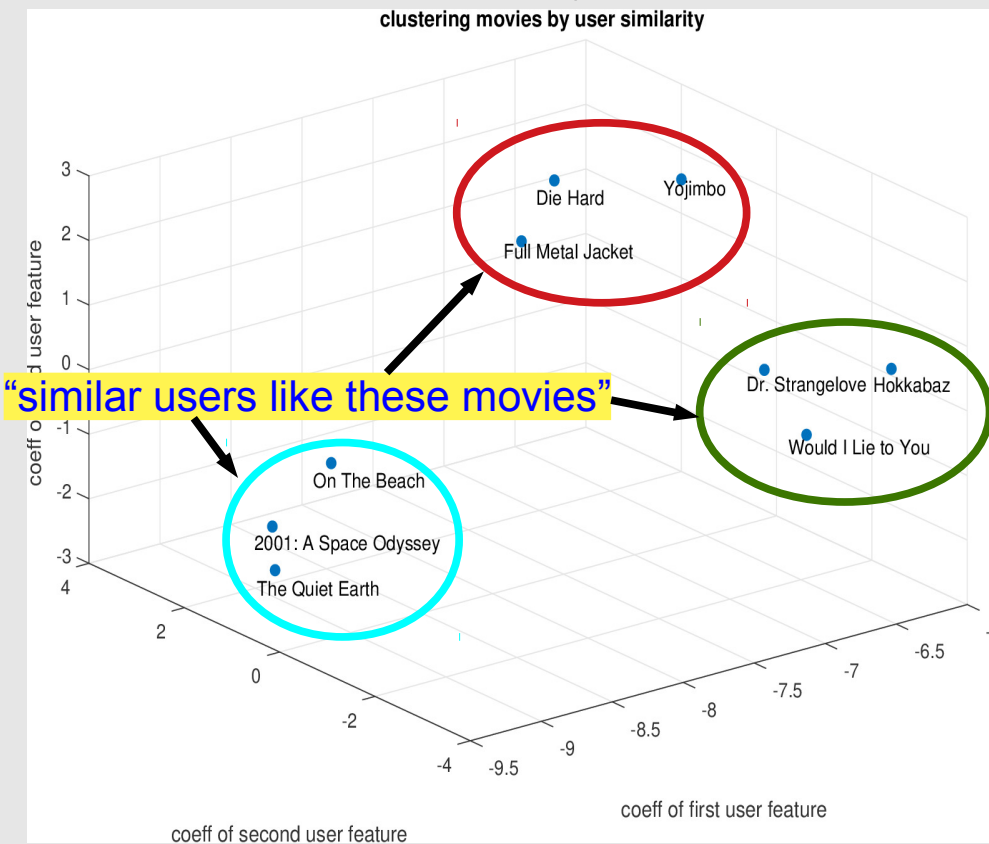
“how much the 3rd most typical user likes FMJ”

Clustering in Feature Bases

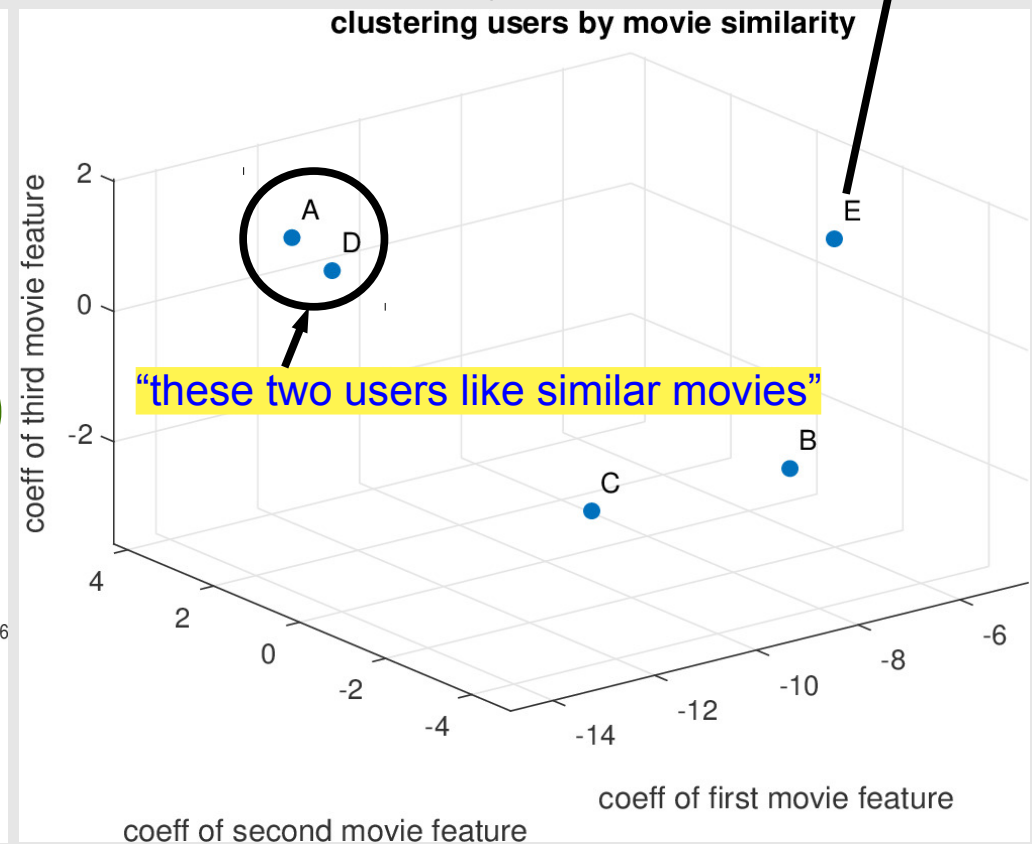
- Scatter plot of α_{11} , α_{21} , and α_{31} for all movies

$$\text{user E row} = [1 \ 2 \ 1 \ 2 \ 1 \ 2 \ 2 \ 2 \ 1] \\ = \beta_{51}\vec{v}_1^T + \beta_{52}\vec{v}_2^T + \beta_{53}\vec{v}_3^T + \cdots + \beta_{59}\vec{v}_9^T$$

Movies classified by user features

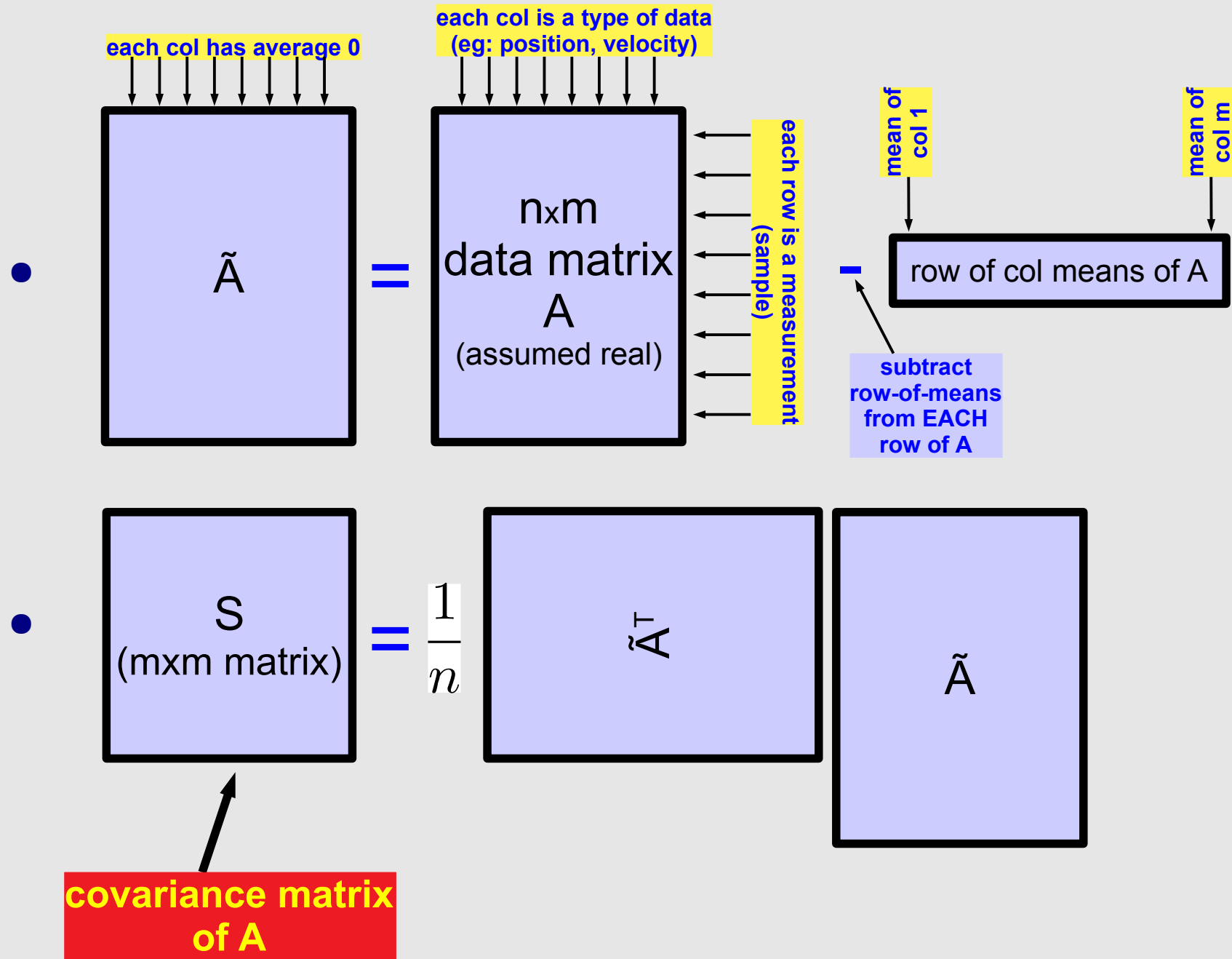


Users classified by movie features



Principal Component Analysis (PCA)


Covariance Matrices



Covariance Matrices: Properties

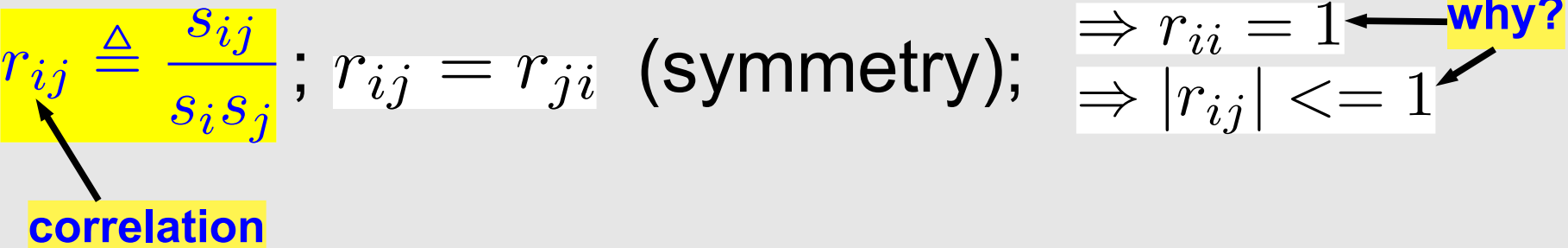
- $S \triangleq \frac{1}{n} \tilde{A}^T \tilde{A}$
- S is square and **symmetric**: $S = S^T$ or $s_{ij} = s_{ji}$
- The **diagonal entries of S are real and ≥ 0**

- $s_i^2 \triangleq s_{ii} = \frac{1}{n} \sum_{j=1}^n \tilde{a}_{ij}^2 \geq 0$: **variance of i^{th} row of A**

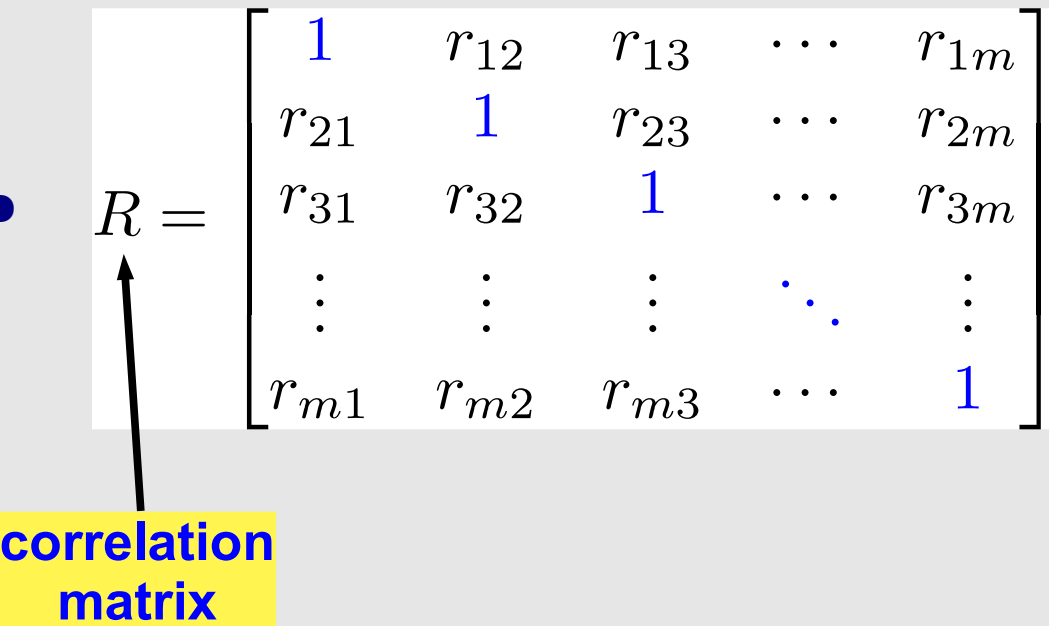
- $S = \begin{bmatrix} s_1^2 & s_{12} & s_{13} & \cdots & s_{1m} \\ s_{21} & s_2^2 & s_{23} & \cdots & s_{2m} \\ s_{31} & s_{32} & s_3^2 & \cdots & s_{3m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ s_{m1} & s_{m2} & s_{m3} & \cdots & s_m^2 \end{bmatrix}$ 

- **can also show:** $|s_{ij}| \leq s_i s_j$
 → using the **Cauchy-Schwartz inequality**

The Correlation Matrix

- $r_{ij} \triangleq \frac{s_{ij}}{s_i s_j}$; $r_{ij} = r_{ji}$ (symmetry); $\Rightarrow r_{ii} = 1$ **why?**
 $\Rightarrow |r_{ij}| \leq 1$


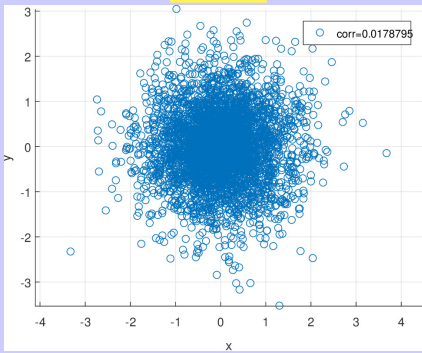
correlation

- $R = \begin{bmatrix} 1 & r_{12} & r_{13} & \cdots & r_{1m} \\ r_{21} & 1 & r_{23} & \cdots & r_{2m} \\ r_{31} & r_{32} & 1 & \cdots & r_{3m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r_{m1} & r_{m2} & r_{m3} & \cdots & 1 \end{bmatrix}$


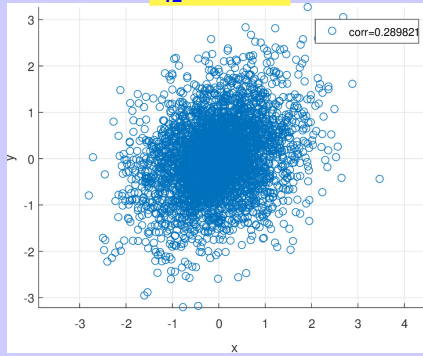
correlation matrix

Correlation: Geometric Intuition

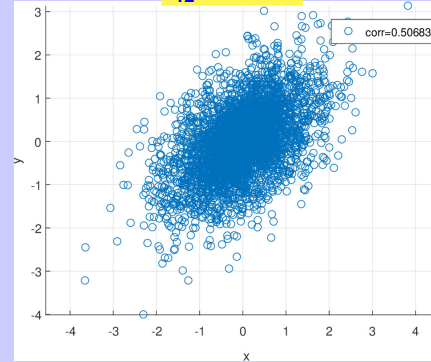
$r_{12} = 0$



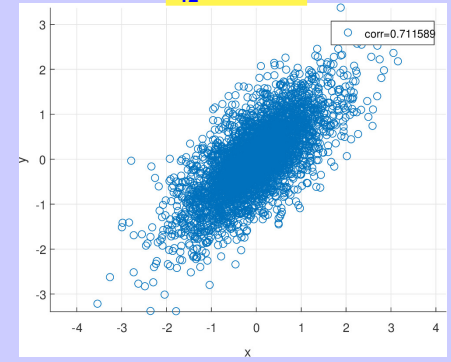
$r_{12} = 0.29$



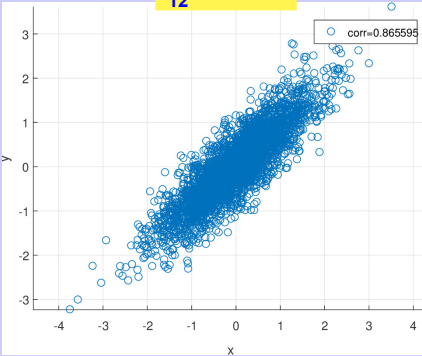
$r_{12} = 0.51$



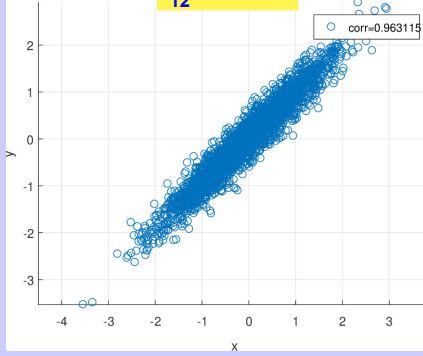
$r_{12} = 0.71$



$r_{12} = 0.87$



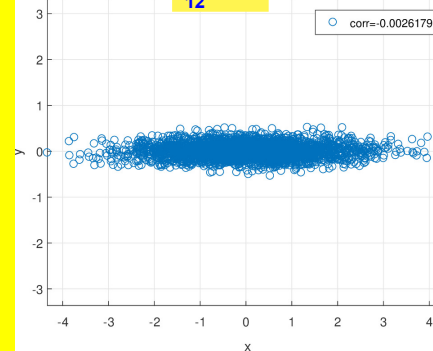
$r_{12} = 0.96$



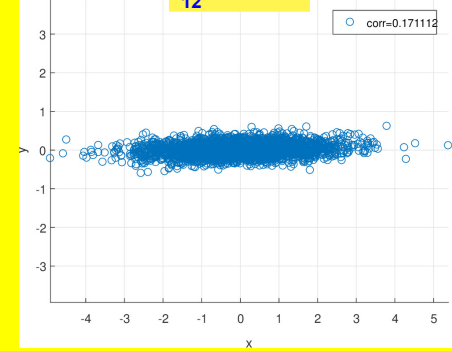
correlation provides
some insight ...

5000 x 2 matrices
(each point is a row)

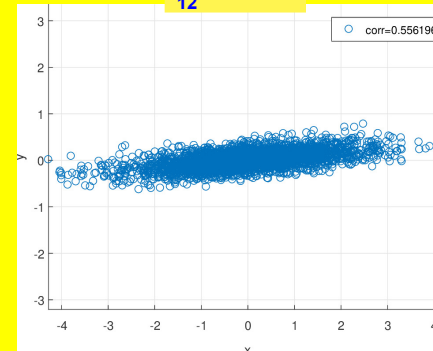
$r_{12} = 0$



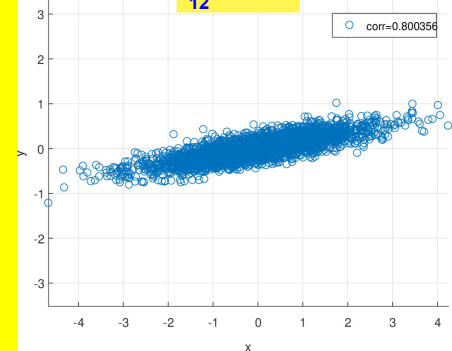
$r_{12} = 0.17$



$r_{12} = 0.56$



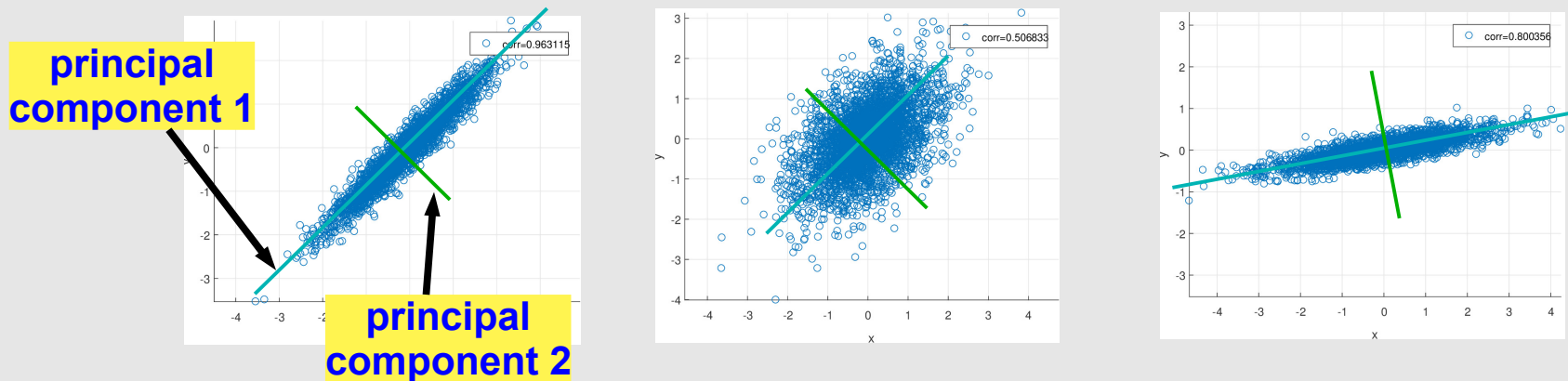
$r_{12} = 0.8$



... but it leaves a lot out

The Intuition behind PCA

- PCA: finds (orthogonal) “**main axes** along which the data lie”: the **principal components**
- provides weights indicating “strength” of each axis



- starting point for PCA: the covariance matrix S

PCA: The Procedure

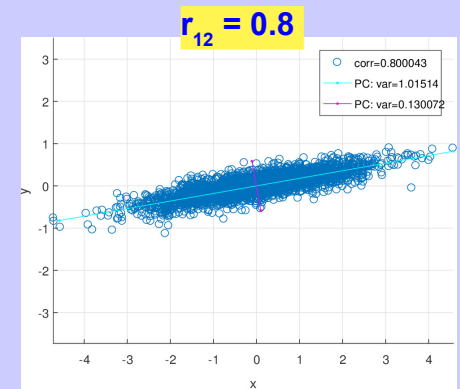
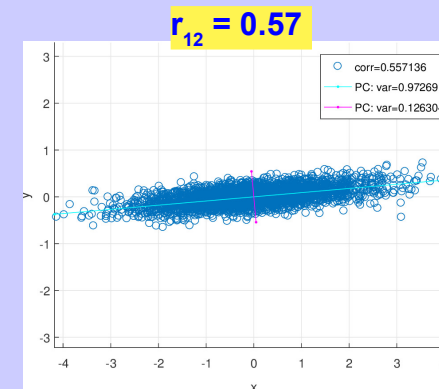
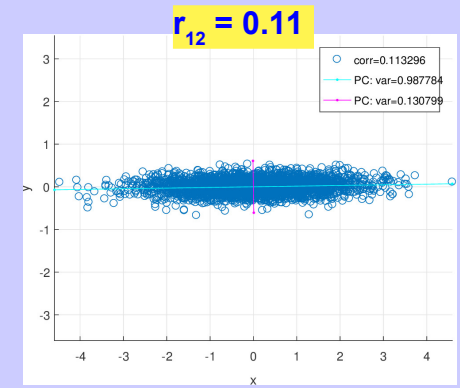
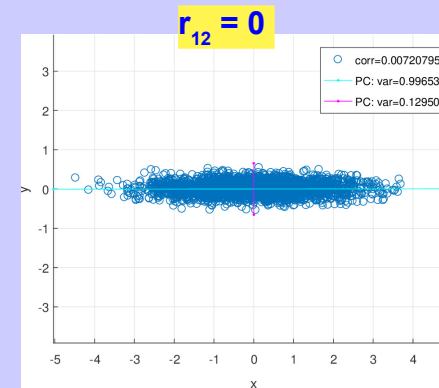
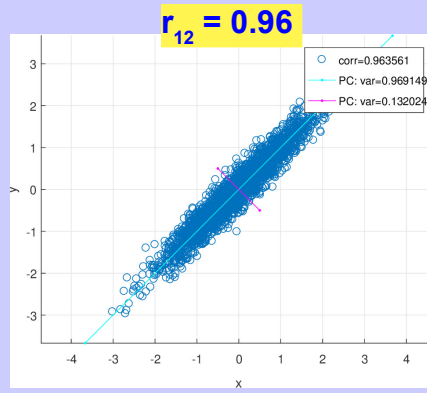
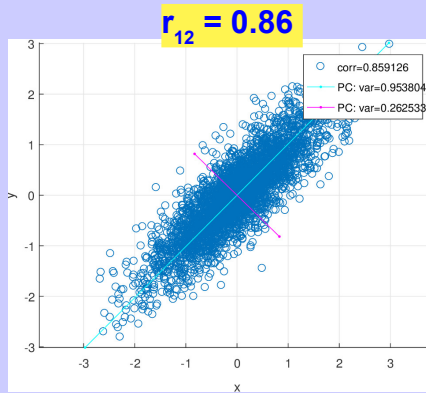
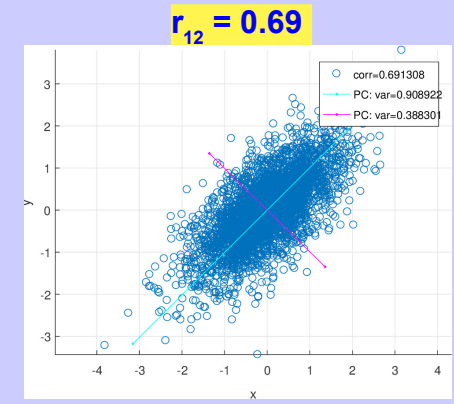
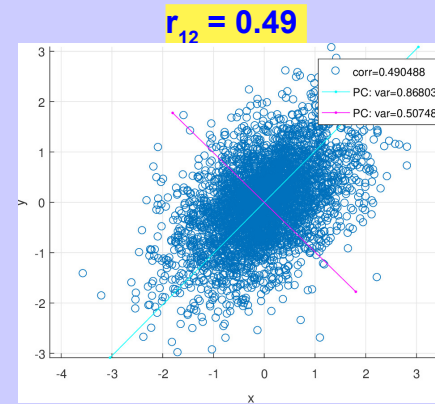
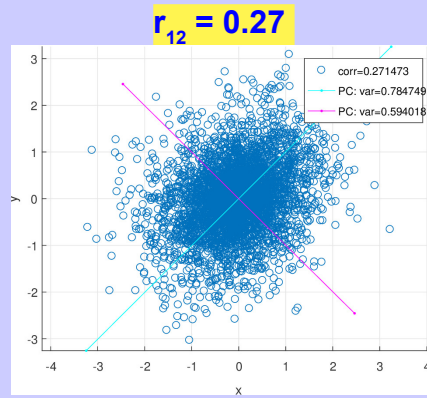
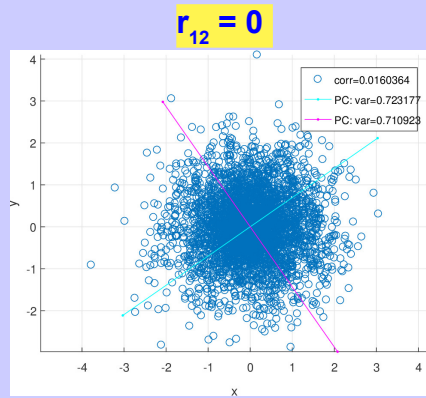
- Eigendecompose the covariance matrix

- $S = \begin{bmatrix} s_1^2 & s_{12} & s_{13} & \cdots & s_{1m} \\ s_{21} & s_2^2 & s_{23} & \cdots & s_{2m} \\ s_{31} & s_{32} & s_3^2 & \cdots & s_{3m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ s_{m1} & s_{m2} & s_{m3} & \cdots & s_m^2 \end{bmatrix} = P\Lambda P^{-1}$

- $\sqrt{\lambda_i}$ = the weights
- (i.e., the variances)
- with $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \cdots \geq \lambda_m \geq 0$

- eigenvectors \vec{p}_i = the principal components

Principal Components of the Data

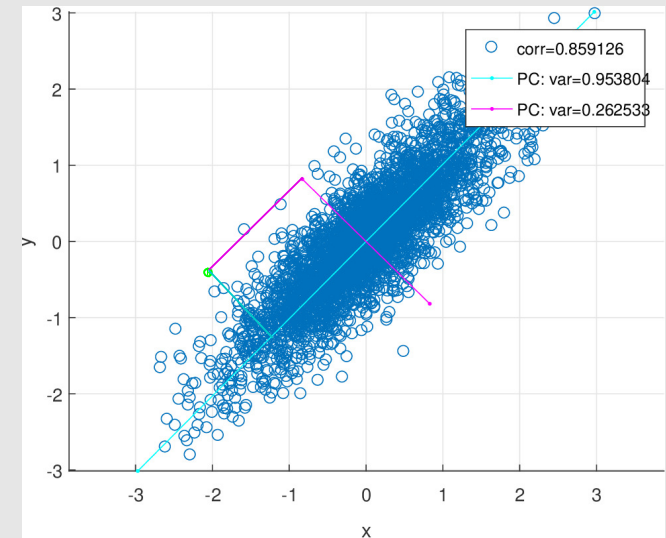


5000 x 2 matrices
(each point is a row)

First PC always captures
direction of
maximum data spread
(2nd PC: max spread in
orthogonal direction)

PCA: Why it Works: The Flow

- First: establish some properties of P and Λ
 - properties of real symmetric matrices
 - real eigenvalues
 - real set of orthonormal eigenvectors
 - properties of real $A^T A$
 - eigenvalues ≥ 0
- Express data in eigenvector basis
 - project each data point onto eigenvectors
- Show that the covariance matrix of the projected data is diagonal
 - the variances of the projections along each axis/PC
- First PC maximizes variance along any 1D projection
 - 2nd PC maximizes remaining variance; and so on

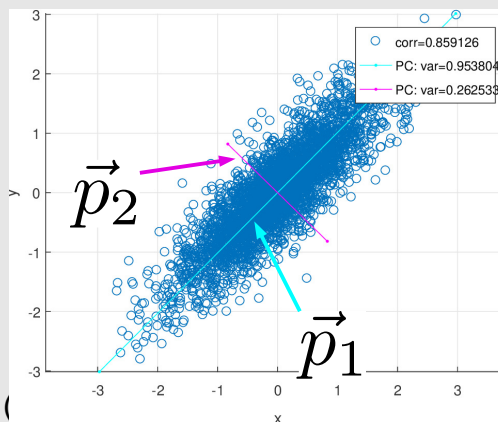


Properties of Covariance Matrices

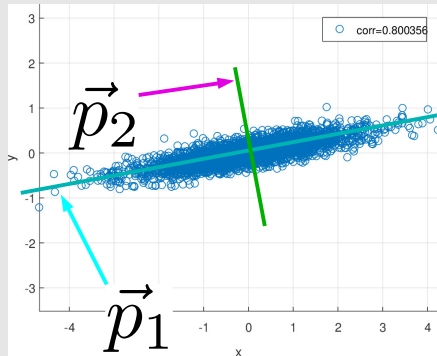
- If S is a **real** $m \times m$ **symmetric** matrix ($s_{ij} = s_{ji}$)
 - 1. **its eigenvalues are all real**
 - $S\vec{p} = \lambda\vec{p}$. **S symmetric** → $\vec{p}^T S = \lambda\vec{p}^T$. → $\vec{p}^T S\vec{p} = \lambda\vec{p}^T\vec{p} = \lambda\|\vec{p}\|^2$
 - **S real** → $S\vec{p} = \bar{\lambda}\vec{p}$. → $\vec{p}^T S\vec{p} = \bar{\lambda}\vec{p}^T\vec{p} = \bar{\lambda}\|\vec{p}\|^2$.
 - **hence** $\lambda\|\vec{p}\|^2 = \bar{\lambda}\|\vec{p}\|^2 \rightarrow \lambda = \bar{\lambda} \rightarrow \lambda \text{ is real.}$
 - 2. A set of **real eigenvectors** can be found (see the notes)
 - 3. The **eigenvectors form an orthonormal set** (basis).
 - (see the notes)
- If S is in the form $A^T A$ (A real)
 - 4. **its eigenvalues are all ≥ 0 .**
 - $A^T A\vec{p} = \lambda\vec{p} \rightarrow \vec{p}^T A^T A\vec{p} = \lambda\vec{p}^T\vec{p} \rightarrow (A\vec{p})^T A\vec{p} = \lambda\vec{p}^T\vec{p}$
 - $\|A\vec{p}\|^2 = \lambda\|\vec{p}\|^2 \rightarrow \lambda = \frac{\|A\vec{p}\|^2}{\|\vec{p}\|^2} \geq 0$.

PCA Basis Diagonalises the Data

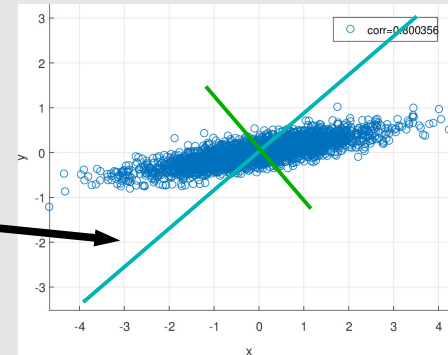
- eigenvectors orthonormal $\rightarrow PP^T = I \rightarrow P^T = P^{-1}$
- eigendecomposition of S: $S = P\Lambda P^T$
- project rows of (zero-mean) A in basis P: $F = \tilde{A}P$
 - columns of F are the projections along \vec{p}_i
- Let G be the co-variance matrix of F: $G \triangleq (F^T F)/n$
 - $nG = F^T F = P^T \tilde{A}^T \tilde{A} P = nP^T S P = nP^T P \Lambda P^T P = n\Lambda$
 $= \Lambda$ (diagonal) \leftarrow Data projected on PC basis becomes UNCORRELATED
- the diagonal entries are the **variances of the data projected along \vec{p}_i** (recall: from defn. of covariance matrix)



Why do PCs Align with Visual Axes?



Why this?
and not this?



- So far: have shown that **PCs are orthonormal**
 - data projected onto them becomes uncorrelated
- but why is the first **PC aligned with the direction of maximum spread?**
- **Key property of PCA** ← [proof](#) → [notes](#)
 - consider **any** norm-1 vector (“direction”) \vec{p}
 - project the data along it: $\tilde{A}\vec{p}$
 - find the variance of the projected data: $\frac{1}{n}(\tilde{A}\vec{p})^T(\tilde{A}\vec{p})$
 - **the first PC \vec{p}_1 maximizes this variance** (the max is $\vec{\lambda}_1$)
 - 2nd PC: maximizes variance along directions orthogonal to \vec{p}_1
 - 3rd PC: maximizes var. along dirs. orthogonal to \vec{p}_1 and \vec{p}_2 ; and so on

PCA: the Connection with the SVD

- Suppose you run an SVD on the data: $\tilde{A} = U\Sigma V^T$
- the covariance matrix is:

$$\rightarrow S \triangleq \frac{1}{n} \tilde{A}^T \tilde{A} = \frac{1}{n} V \Sigma^T U^T U \Sigma V^T = V \frac{\Sigma^T \Sigma}{n} V^T$$

\rightarrow recall PCA: $S = P \Lambda P^T$ **IDENTICAL FORM**

diagonal and ≥ 0

- i.e., can use the SVD of \tilde{A} for PCA:

\rightarrow **just set** $\lambda_i \triangleq \frac{\sigma_i^2}{n}$ **and** $P \triangleq V$ **(no need to even form S)!**

Computing SVDs via Eigendecomposition

- Prev. slide: **SVD**: $S = V \frac{\Sigma^T \Sigma}{n} V^T$; **PCA**: $S = P \Lambda P^T$
- **Q: how to calculate an SVD of a matrix A?**
 - using eigendecomposition
- **A: just use the above insight** (PCA/eigendecomposition)!
 - form $S \triangleq A^T A$, eigendecompose $S = P \Lambda P^T$
 - set $\sigma_i \triangleq \sqrt{\lambda_i}$, $V \triangleq P$
 - more work, because (we had assumed) $n \geq m$
 - what about U?
 - just eigendecompose $\hat{S} \triangleq A A^T = Q \hat{\Lambda} Q^T$; then $U \triangleq Q$
 - can also get **V** from the **same** eigendecomposition
 - $A = U \Sigma V^T \rightarrow U^T A = \Sigma V^T \rightarrow A^T U = V \Sigma^T \rightarrow$

$\vec{v}_i = \frac{A^T \vec{u}_i}{\sigma_i}$
 $i = 1, \dots, m$
 - set $\sigma_i \triangleq \sqrt{\lambda_i}$
 - if $\sigma_i = 0$, choose \vec{v}_i arbitrarily to complete orthonormal basis for V

* why didn't we subtract means from A and normalize by n?

Summary: SVD and PCA

- Singular Value Decomposition (SVD)
 - useful for “low-rank approximations” of matrices
 - image analysis and compression
 - general data analysis, finding important features, clustering
- Covariance, Correlation and PCA
 - visualizing data as scatter plots
 - covariance and correlation matrices of data
 - Principal Component Analysis
 - eigenvecs of covariance matrix: principal components
 - directions along which data varies maximally
 - dropping later PCs can, eg, clean out (small) noise
 - eigenvalues correspond to variances along PCs
 - SVD can be used instead of eigendecomposition
 - eigendecomposition of covariance matrix: performs SVD