EE16B, Spring 2018
UC Berkeley EECS

Maharbiz and Roychowdhury

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

Data Analysis

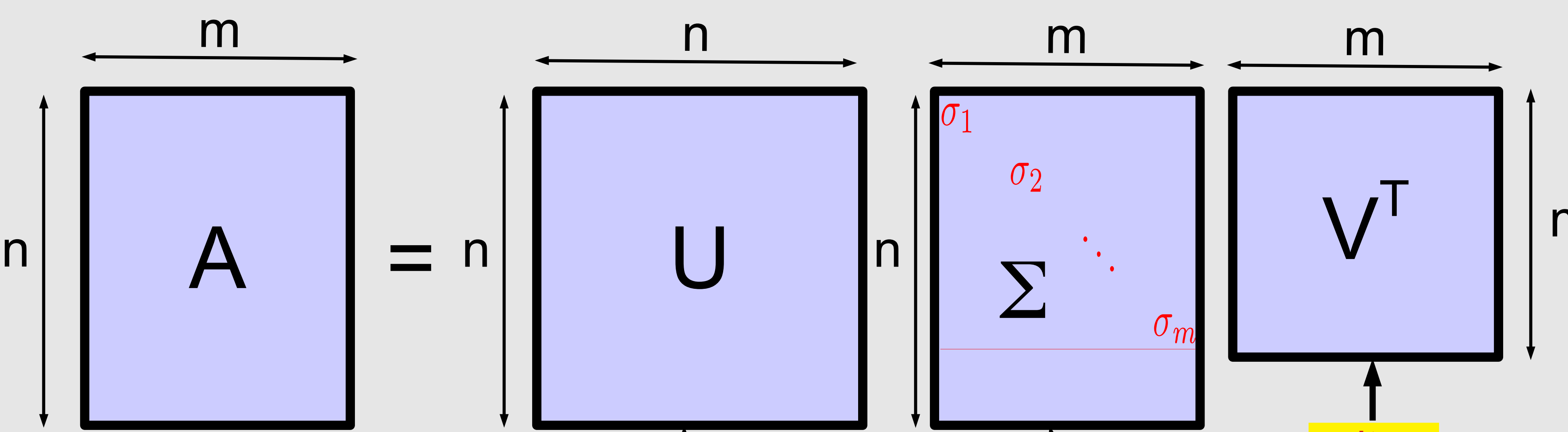
Singular Value Decomposition
and
Principal Component Analysis

The SVD
(Singular Value Decomposition)

Singular Value Decomposition

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

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



SINGULAR VALUES

$\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \cdots \geq \sigma_m \geq 0$

unitary

$U^T U = I$

diagonal
σ_i real and +ve

unitary

$V^T V = I$

Unitary Matrices: Orthonormality

$$
\underbrace{\begin{bmatrix} \longleftarrow & \vec{u}_1^T & \longrightarrow \\ \longleftarrow & \vec{u}_2^T & \longrightarrow \\ \longleftarrow & \vec{u}_3^T & \longrightarrow \\ & \vdots & \\ \longleftarrow & \vec{u}_n^T & \longrightarrow \end{bmatrix}}_{\text{U}^\text{T}}
\underbrace{\begin{bmatrix} \uparrow & \uparrow & \uparrow & & \uparrow \\ \vec{u}_1 & \vec{u}_2 & \vec{u}_3 & \cdots & \vec{u}_n \\ \downarrow & \downarrow & \downarrow & & \downarrow \end{bmatrix}}_{\text{U}}
=
\underbrace{\begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix}}_{\text{I}}
$$

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

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

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

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

Similarly,
$\vec{v}_1, \vec{v}_2, \cdots, \vec{v}_m$
are **ORTHONORMAL**

$||\vec{v}_j|| = 1$

Rank 1 Matrices and Outer Products

rank=1 col

- Consider $A = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 3 & 3 & 3 & 3 & 3 \\ 2 & 2 & 2 & 2 & 2 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix}$ row
  - rank-1 matrix can be written as $\vec{x}\vec{y}^T$: an **outer product**
- outer product: product of col and row vectors

rank=1

- $\begin{bmatrix} x \\ y \\ z \end{bmatrix} \begin{bmatrix} a & b & c & d & e \end{bmatrix} = \begin{bmatrix} xa & xb & xc & xd & xe \\ ya & yb & yc & yd & ye \\ za & zb & zc & zd & ze \end{bmatrix}$
- rank-1: a very "simple" type of matrix
  - its "data" can be "compressed" very easily
    - ➔ can be written as outer product: $A = \vec{x}\vec{y}^T$

nm numbers nxm nx1 1xm

n+m << nm: **data compression**

n+m numbers

Matrix Multiplication using Outer Products

$$\underbrace{\begin{bmatrix} \uparrow & \uparrow & \uparrow & & \uparrow \\ \vec{x}_1 & \vec{x}_2 & \vec{x}_3 & \cdots & \vec{x}_n \\ \downarrow & \downarrow & \downarrow & & \downarrow \end{bmatrix}}_{X} \underbrace{\begin{bmatrix} \longleftarrow & \vec{y}_1^T & \longrightarrow \\ \longleftarrow & \vec{y}_2^T & \longrightarrow \\ \longleftarrow & \vec{y}_3^T & \longrightarrow \\ & \vdots & \\ \longleftarrow & \vec{y}_n^T & \longrightarrow \end{bmatrix}}_{Y^T}$$

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:

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

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

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

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

SVD: Sum of Outer Products Form

$$A = \overset{U}{\begin{bmatrix} \uparrow & \uparrow & \uparrow & & \uparrow \\ \vec{u}_1 & \vec{u}_2 & \vec{u}_3 & \cdots & \vec{u}_n \\ \downarrow & \downarrow & \downarrow & & \downarrow \end{bmatrix}} \overset{\Sigma}{\begin{bmatrix} \sigma_1 & & & & \\ & \sigma_2 & & & \\ & & \sigma_3 & & \\ & & & \ddots & \\ & & & & \sigma_m \\ & & & & \end{bmatrix}} \overset{V^T}{\begin{bmatrix} \longleftarrow & \vec{v}_1^T & \longrightarrow \\ \longleftarrow & \vec{v}_2^T & \longrightarrow \\ & \vdots & \\ \longleftarrow & \vec{v}_m^T & \longrightarrow \end{bmatrix}}$$

biggest weight | next biggest weight | smallest weight

$$= \sigma_1 \begin{bmatrix} \uparrow \\ \vec{u}_1 \\ \downarrow \end{bmatrix} \begin{bmatrix} \longleftarrow & \vec{v}_1^T & \longrightarrow \end{bmatrix} + \sigma_2 \begin{bmatrix} \uparrow \\ \vec{u}_2 \\ \downarrow \end{bmatrix} \begin{bmatrix} \longleftarrow & \vec{v}_2^T & \longrightarrow \end{bmatrix} + \ldots + \sigma_m \begin{bmatrix} \uparrow \\ \vec{u}_m \\ \downarrow \end{bmatrix} \begin{bmatrix} \longleftarrow & \vec{v}_m^T & \longrightarrow \end{bmatrix}$$

outer product nxm rank-1 matrix $\vec{u}_1\vec{v}_1^T$

outer product nxm rank-1 matrix $\vec{u}_2\vec{v}_2^T$

outer product nxm rank-1 matrix $\vec{u}_m\vec{v}_m^T$

Frobenius norm (sqrt(sum of squares) = 1)

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

$$\sigma_1 \begin{bmatrix} \uparrow \\ \vec{u}_1 \\ \downarrow \end{bmatrix} \begin{bmatrix} \longleftarrow & \vec{v}_1^T & \longrightarrow \end{bmatrix}$$

6.75x10^4

$$\sigma_2 \simeq 10^{-10}$$

$$\vec{u}_1 = \begin{bmatrix} 1 \\ \vdots \\ 1 \\ 0.2 \\ \vdots \\ 0.2 \end{bmatrix}$$

(actual values are normalized)

$$\vec{v}_1^T = \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}$$

(actual values are normalized)

This is a RANK-1 FLAG

Example: Polish Flag as a Matrix

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

A =

original: 10MB

rank 1: 17.5kB

$$\sigma_1 \begin{bmatrix} \uparrow \\ \vec{u}_1 \\ \downarrow \end{bmatrix} \begin{bmatrix} \longleftarrow & \vec{v}_1^T & \longrightarrow \end{bmatrix}$$

$(8.5,6.4,6.6) \times 10^4$

$$\begin{bmatrix} 1,1,1 \\ \vdots \\ 1,1,1 \\ 1,0,0 \\ \vdots \\ 1,0,0 \end{bmatrix}$$

RGB components (actual ones are normalized)

$$\begin{bmatrix} 1,1,1 & \cdots & 1,1,1 \end{bmatrix}$$

RGB components (actual ones are normalized)

$$\sigma_2 \simeq 10^{-10} \quad \text{for R, G, B}$$

This is a RANK-1 FLAG

Example: SVD of the Austrian Flag

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

A =

original: 73MB

rank 1: 48.5kB

$$\sigma_1 \begin{bmatrix} \uparrow \\ \vec{u}_1 \\ \downarrow \end{bmatrix} \begin{bmatrix} \longleftarrow & \vec{v}_1^T & \longrightarrow \end{bmatrix}$$

$2.3\text{x}10^5$

$$\begin{bmatrix} 1,0,0 \\ \vdots \\ 1,0,0 \\ 1,1,1 \\ \vdots \\ 1,1,1 \\ 1,0,0 \\ \vdots \\ 1,0,0 \end{bmatrix}$$

$$\begin{bmatrix} 1,1,1 & \cdots & 1,1,1 \end{bmatrix}$$

RGB components
(actual ones are normalized)

RGB components
(actual ones are normalized)

This is ALSO a RANK-1 FLAG

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

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

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

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

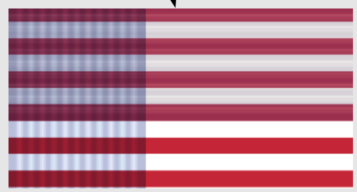
This is a RANK-3 FLAG

Example: SVD of the US Flag

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

original: 8.8MB

strongest "feature"



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

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

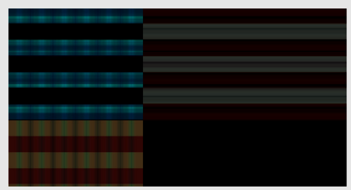
rank 5: 83kB

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

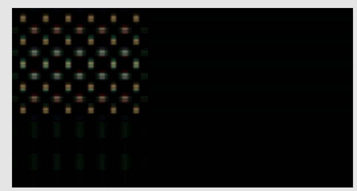
rank 10: 167kB

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

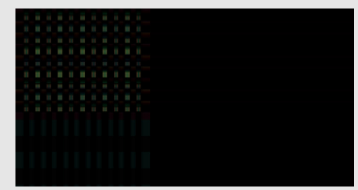
rank 15: 253kB



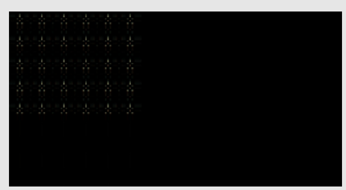
$\sigma_2 \vec{u}_2 \vec{v}_2^T$ 16.5kb



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



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



$\sigma_5 \vec{u}_5 \vec{v}_5^T$ 16.5kb

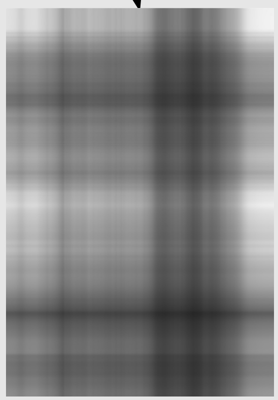
$\sigma_6 \vec{u}_6 \vec{v}_6^T$ 16.5kb

Example: SVD of Michel Maharbiz

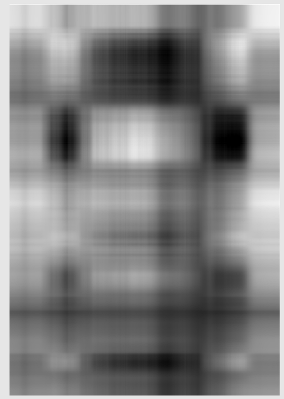
- size: 1100x757 (x 3 colours: R, G, B)

original

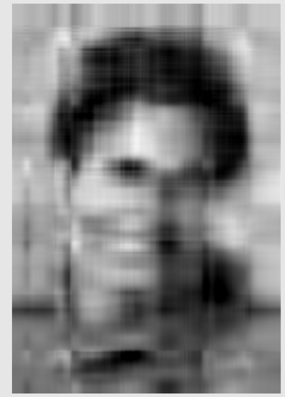
strongest **"feature"** ?!



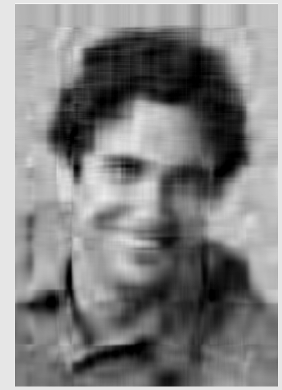
$\sigma_1 \vec{u}_1 \vec{v}_1^T$ 15kB



$\sum_{i=1}^{2} \sigma_i \vec{u}_i \vec{v}_i^T$



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

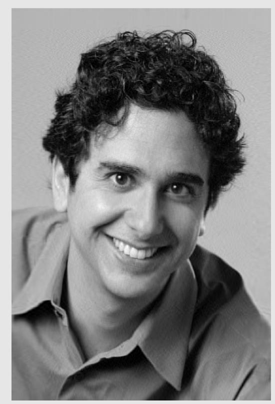


$\sum_{i=1}^{10} \sigma_i \vec{u}_i \vec{v}_i^T$

$\sum_{i=1}^{300} \sigma_i \vec{u}_i \vec{v}_i^T$

$\sum_{i=1}^{20} \sigma_i \vec{u}_i \vec{v}_i^T$

$\sum_{i=1}^{50} \sigma_i \vec{u}_i \vec{v}_i^T$



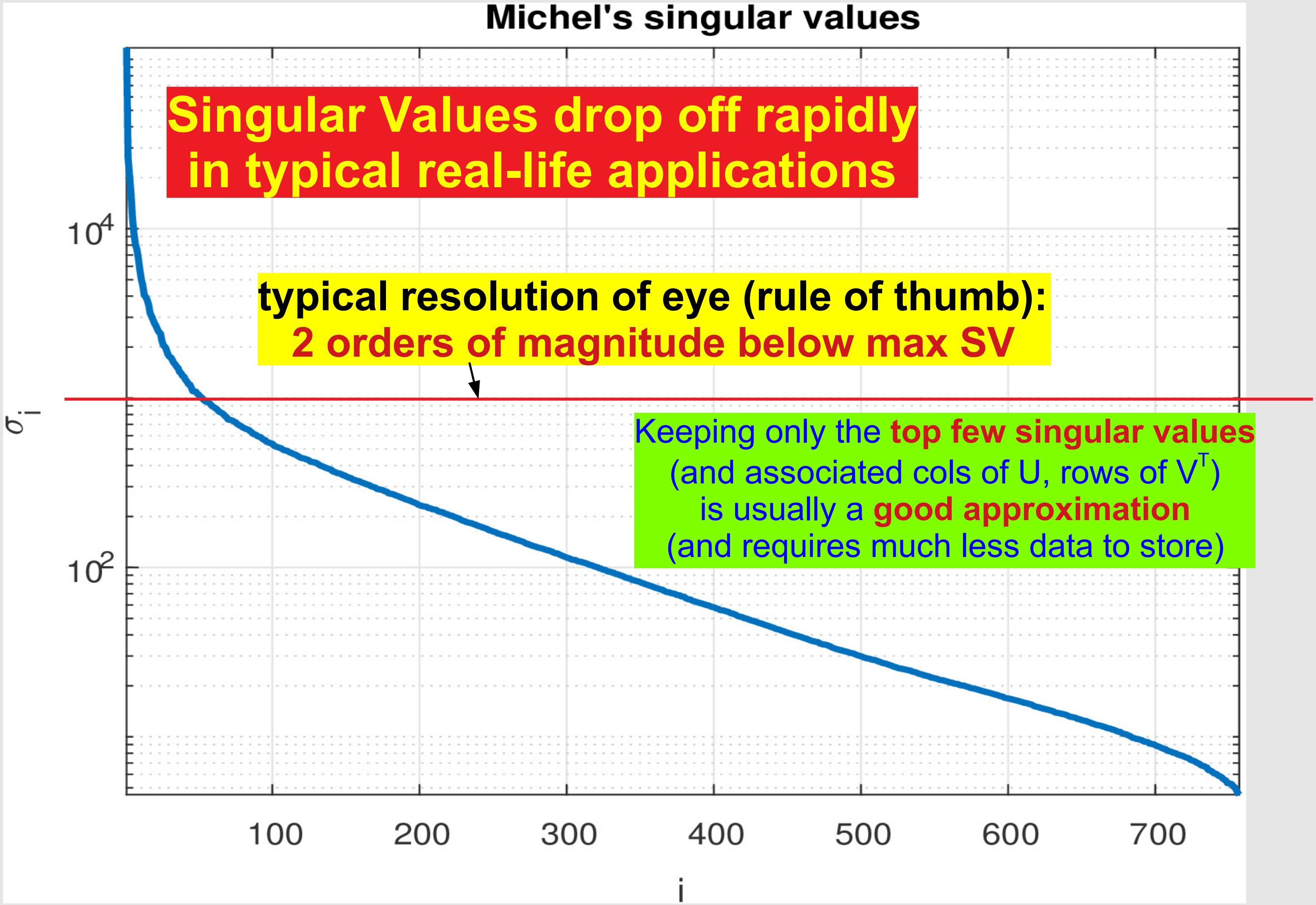
$\sum_{i=1}^{100} \sigma_i \vec{u}_i \vec{v}_i^T$

$\sum_{i=1}^{200} \sigma_i \vec{u}_i \vec{v}_i^T$

Features not always intuitive

Michel’s Singular Values

- How Michel’s singular values drop off



Geometric View of Orthogonality

Projection onto Orthonormal Bases

Geometric View of Unitary Operations

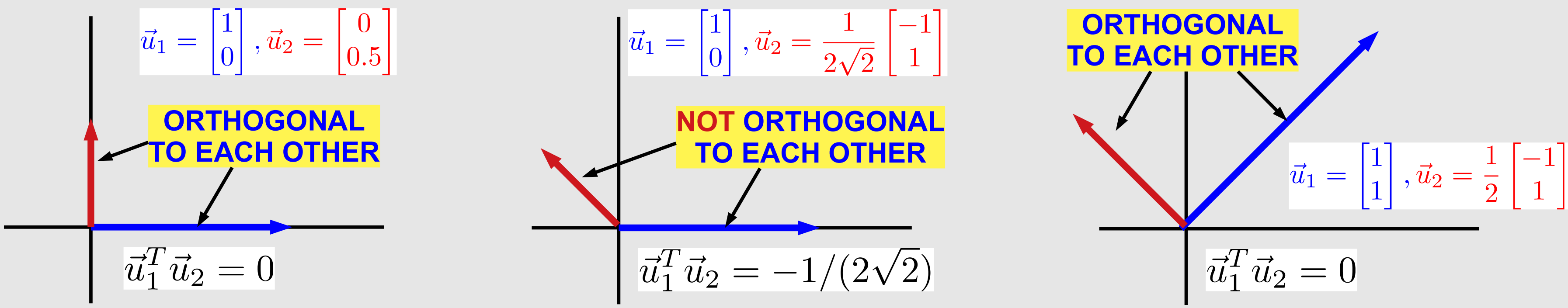
Geometric View of Orthogonality

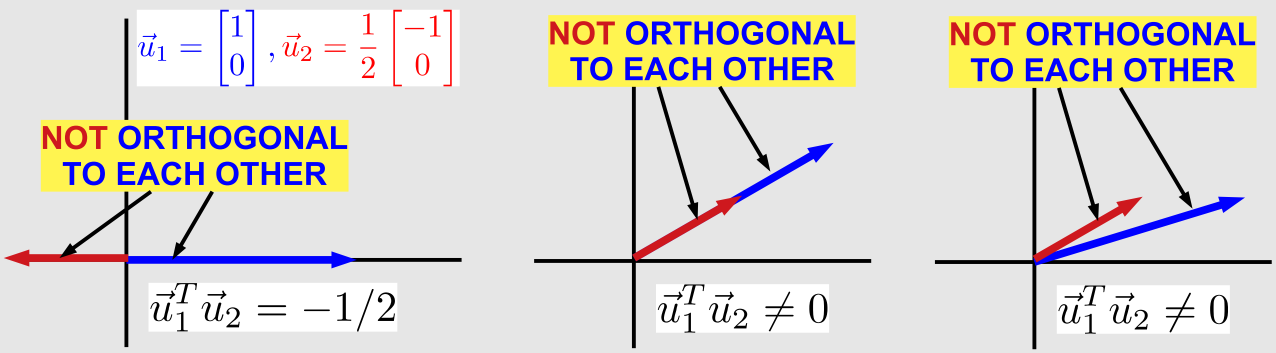
if not necessarily = 1 (but ≠ 0): then called **ORTHOGONAL**

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

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

- In 2D:

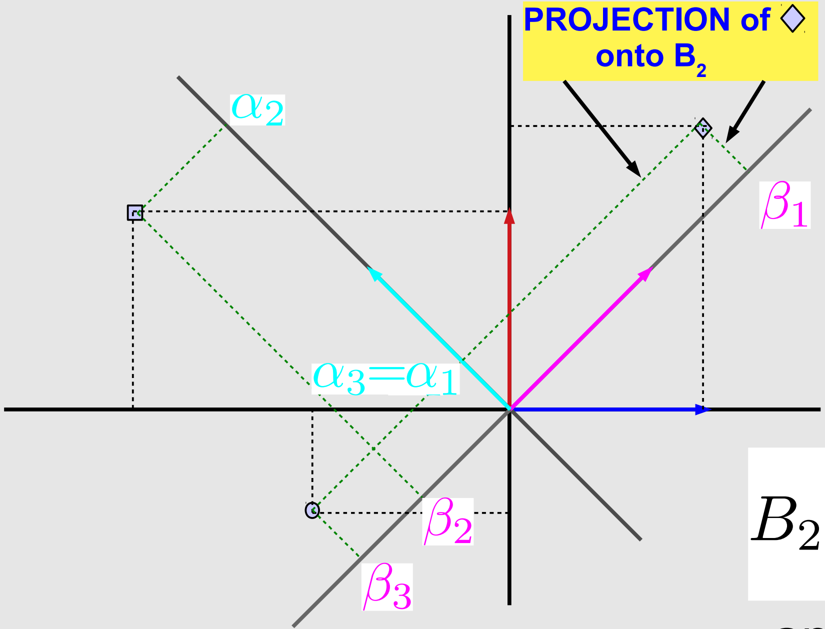




3D: orthogonality also means **at right angles**

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

Projection onto Orthonormal Bases



orthonormal basis

$$B_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

SAMPLES

$$\begin{array}{cccc} & x & & y \\ D = & \begin{bmatrix} 1 & \diamond & 1.5 \\ -2 & \square & 1 \\ -1 & \circ & -0.5 \end{bmatrix} \end{array}$$

$$\begin{array}{ccc} & \vec{p}_1 & \vec{p}_2 \\ B_2 = \frac{1}{\sqrt{2}} & \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \end{array} \qquad \|\vec{p}_i\| = 1 \qquad \vec{p}_1^T \vec{p}_2 = 0$$

another orthonormal basis

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

projecting the data D onto the basis B_2

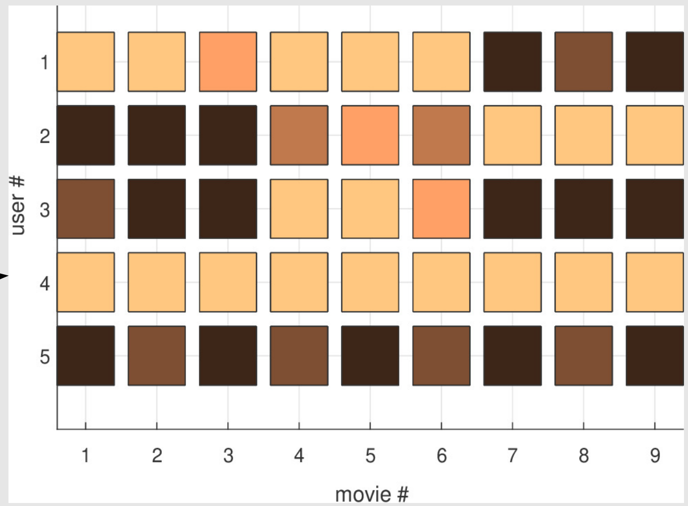
Using the SVD for Data Analysis, Feature Extraction and Clustering

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



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

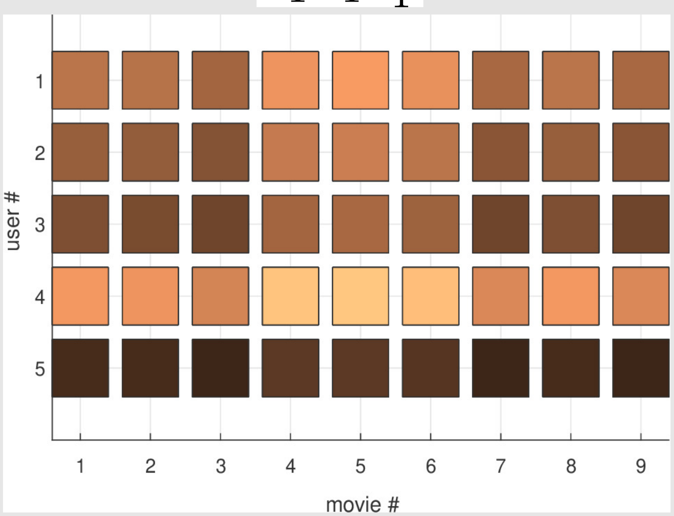
Features of Rating Matrices

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

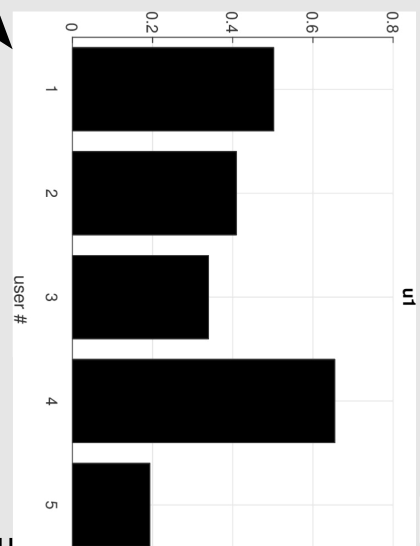
"most typical" user feature:
65% are like D, 50% are like A,
40% are like B, 35% are like C
20% like E

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

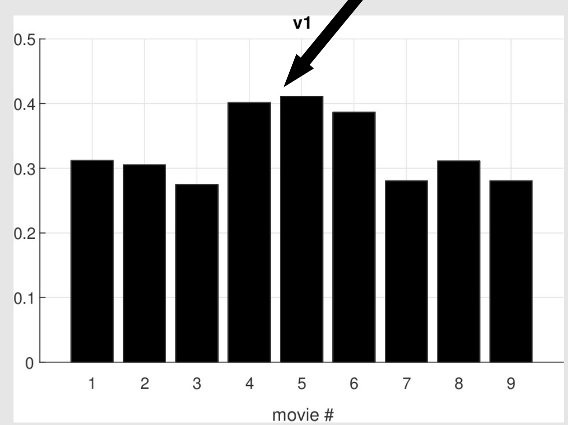
$\sigma_1 \vec{u}_1 \vec{v}_1^T$



=



$\vec{u}_1$



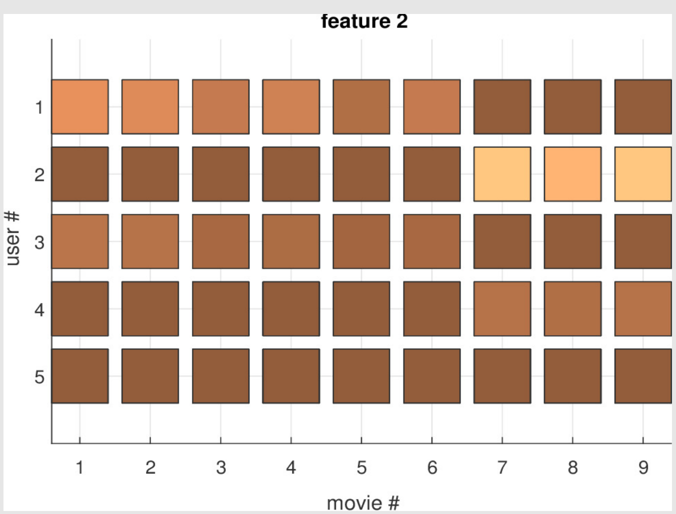
$\vec{v}_1^T$

$\sigma_1 = 22.6$

Features of Rating Matrices (contd.)

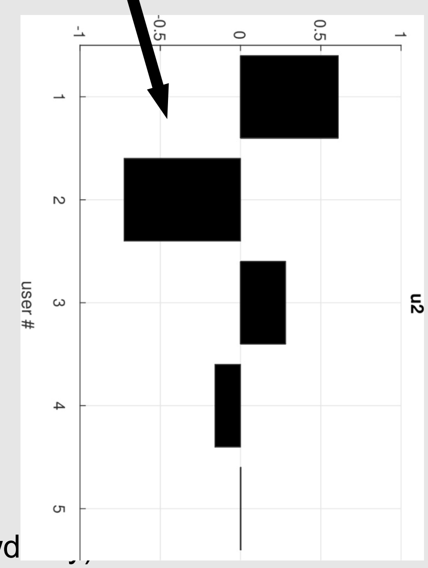
Movie → User Name ↓	Full Metal Jacket	Die Hard	Yojimbo	2001: A Space Odyssey	The Quiet Earth	On The Beach	Would I Lie to You	Dr. Strangelove	Hokkabaz
A	5	5	4	5	5	5	1	2	1
B	1	1	1	3	4	3	5	5	5
C	2	1	1	5	5	4	2	1	1
D	5	5	5	5	5	5	5	5	5
E	1	2	1	2	1	2	2	2	1

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



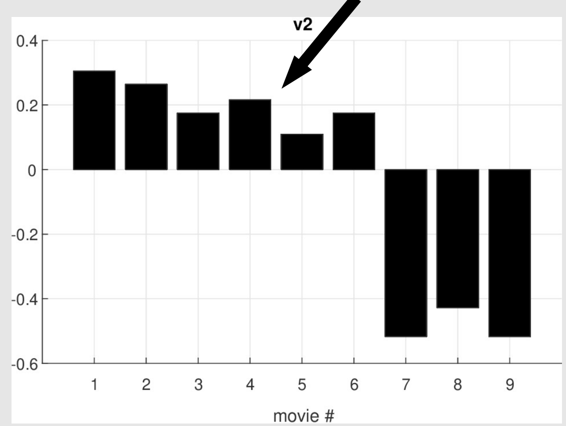
=

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



$\vec{u}_2$

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



$\vec{v}_2^T$

$\sigma_2 = 6.8$

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:

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

user features

"how much the most typical user likes FMJ"

"how much the 3rd most typical user likes FMJ"

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

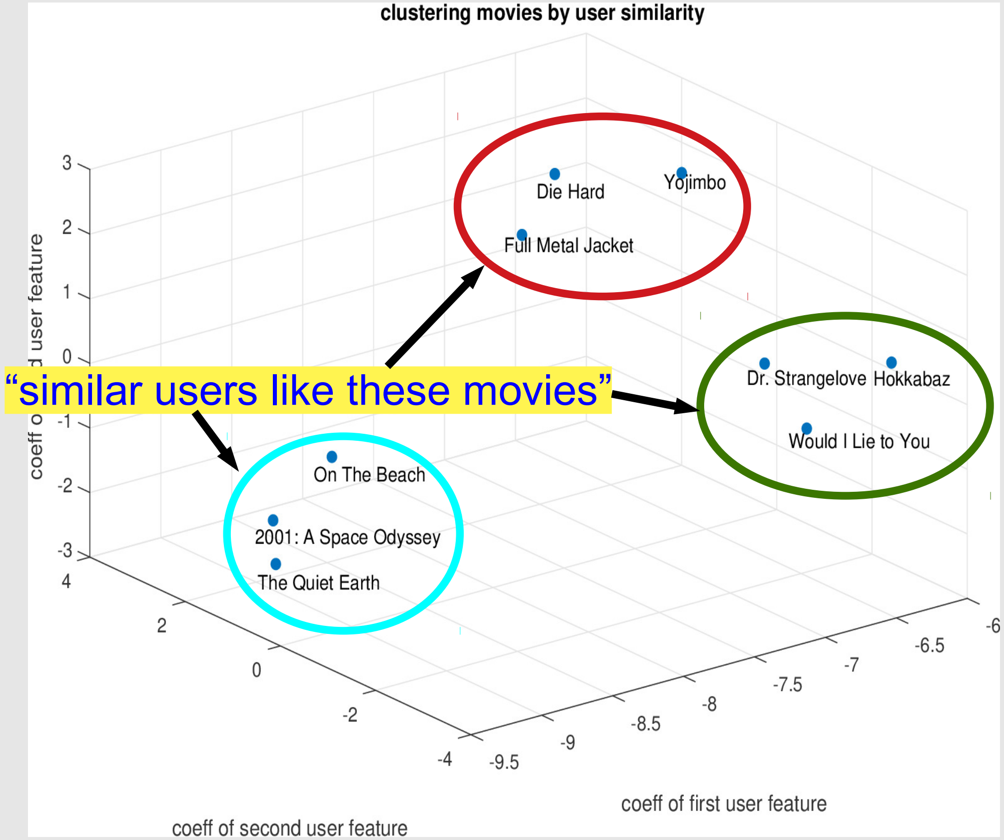
Projection of FMJ col. onto user feature basis

Clustering in Feature Bases

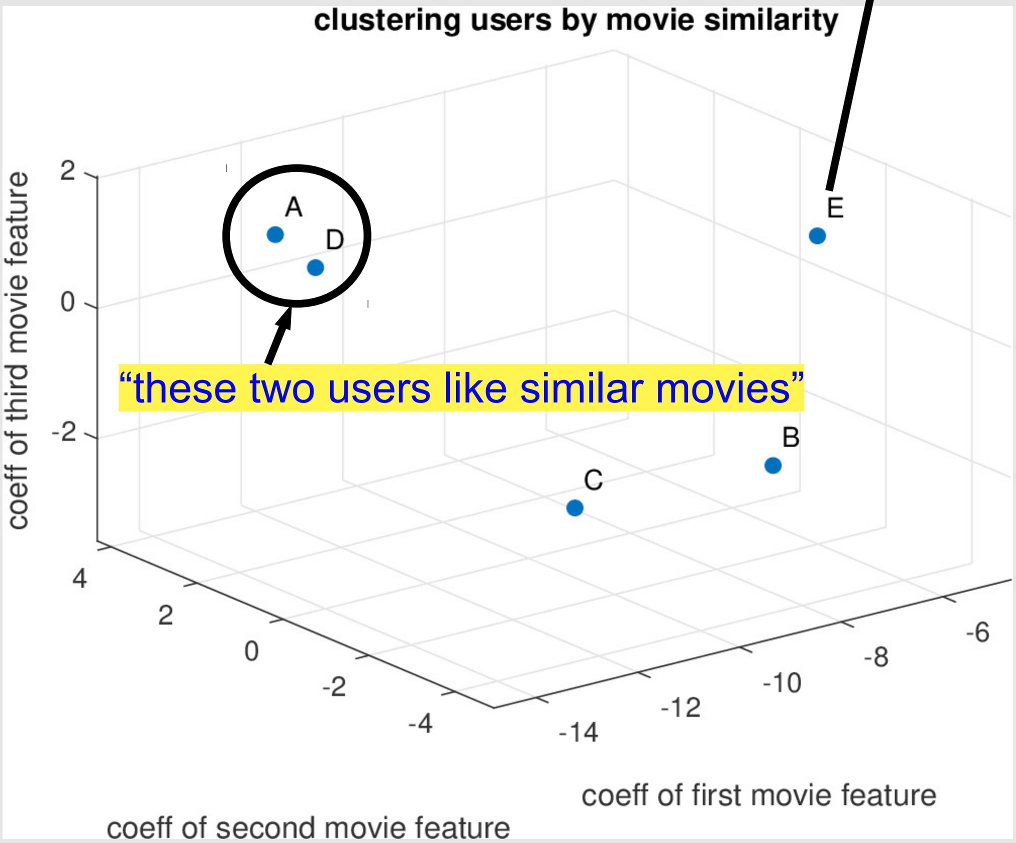
- Scatter plot of α_{11}, α_{21}, and α_{31} for all movies

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

Movies classified by user features

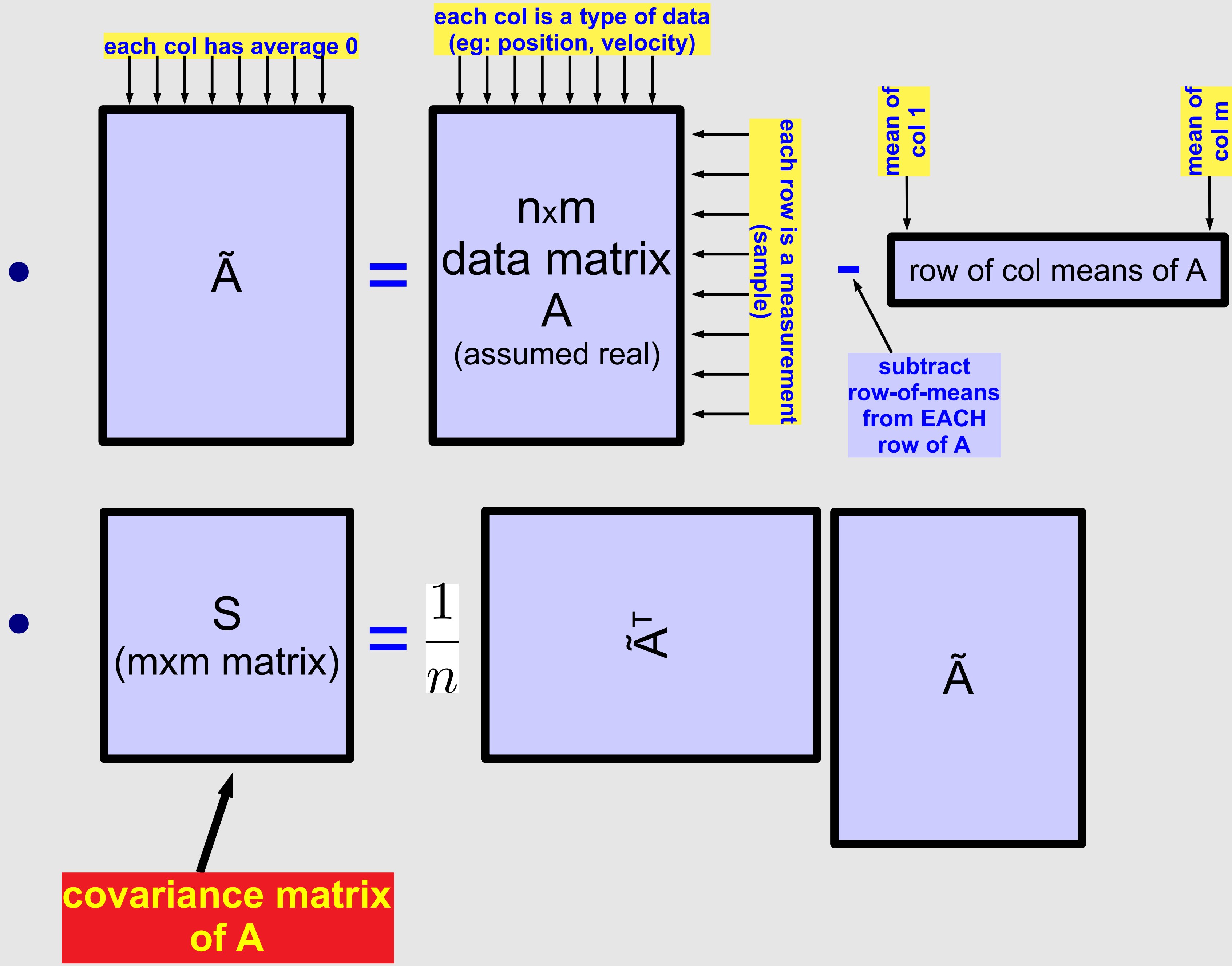


Users classified by movie features



Principal Component Analysis (PCA)

Covariance Matrices



Covariance Matrices: Properties

- $S \triangleq \dfrac{1}{n}\tilde{A}^T\tilde{A}$
- S is square and **symmetric**: $S = S^T$ or $s_{ij} = s_{ji}$
- The **diagonal entries of S** are **real and ≥ 0**
  - $s_i^2 \triangleq s_{ii} = \dfrac{1}{n}\sum_{j=1}^{n} \tilde{a}_{ij}^2 \geq 0$: **variance of i^th row of A**
  - $$S = \begin{bmatrix} s_1^2 & s_{12} & s_{13} & \cdots & s_{1m} \\ s_{21} & s_2^2 & s_{23} & \cdots & s_{2m} \\ s_{31} & s_{32} & s_3^2 & \cdots & s_{3m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ s_{m1} & s_{m2} & s_{m3} & \cdots & s_m^2 \end{bmatrix}$$

covariance matrix

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

The Correlation Matrix

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

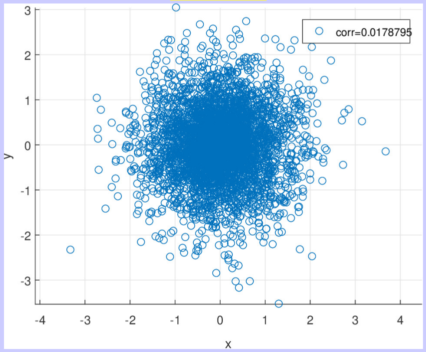
  correlation

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

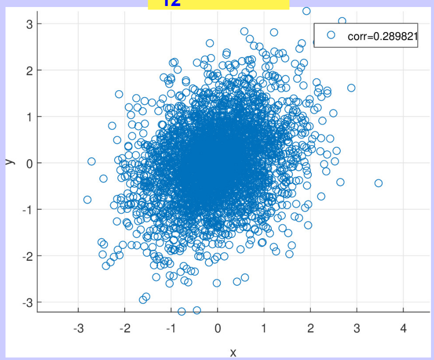
  correlation matrix

Correlation: Geometric Intuition

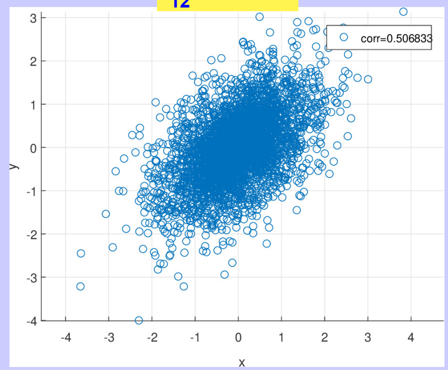
$r_{12} = 0$



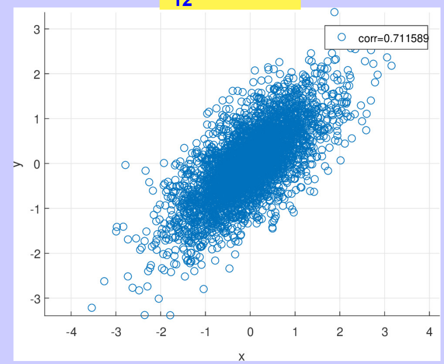
$r_{12} = 0.29$



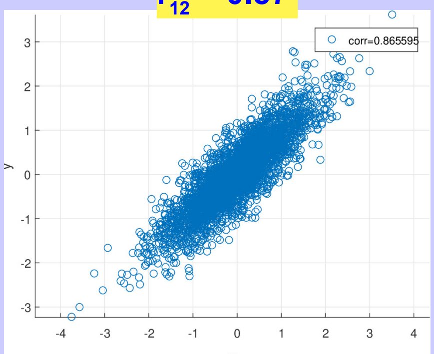
$r_{12} = 0.51$



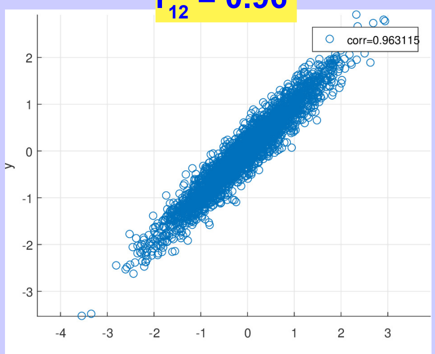
$r_{12} = 0.71$



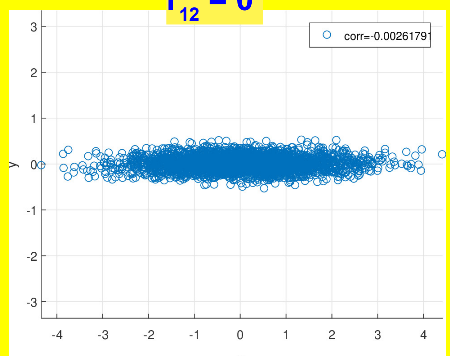
$r_{12} = 0.87$



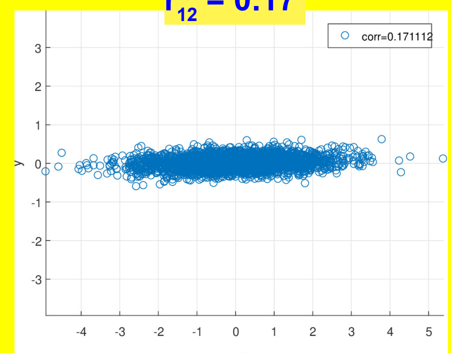
$r_{12} = 0.96$



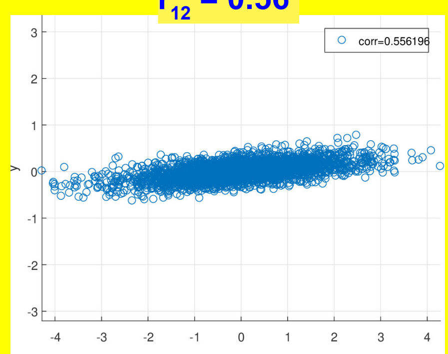
$r_{12} = 0$



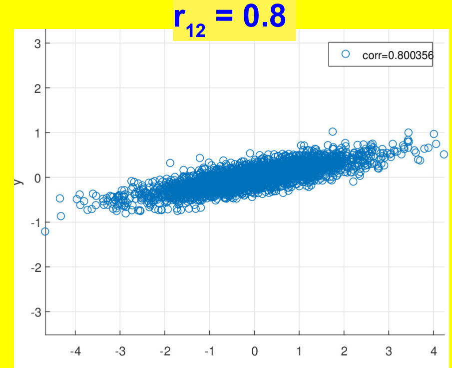
$r_{12} = 0.17$



$r_{12} = 0.56$



$r_{12} = 0.8$



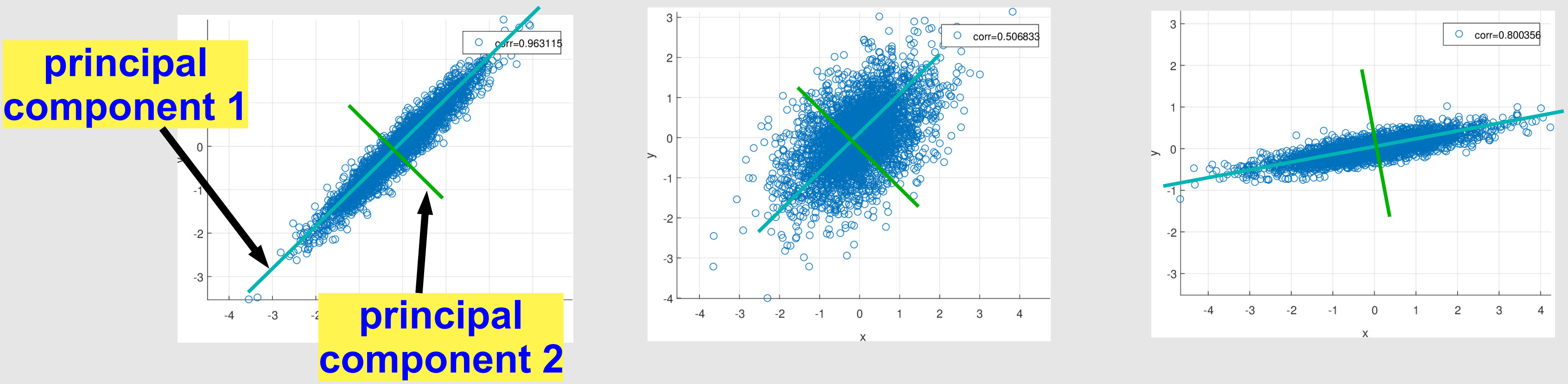
5000 x 2 matrices
(each point is a row)

correlation provides some insight ...

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

$$P = \begin{bmatrix} \uparrow & \uparrow & \uparrow & & \uparrow \\ \vec{p}_1 & \vec{p}_2 & \vec{p}_3 & \cdots & \vec{p}_n \\ \downarrow & \downarrow & \downarrow & & \downarrow \end{bmatrix}$$

cols will be orthonormal

these are the PCs

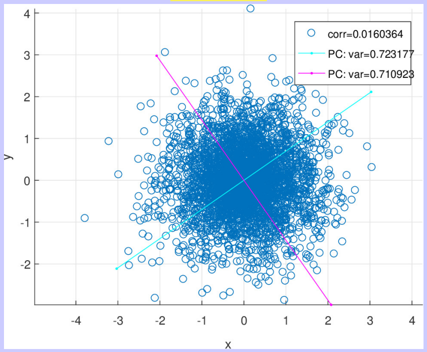
$$\Lambda = \begin{bmatrix} \lambda_1 & & & & \\ & \lambda_2 & & & \\ & & \lambda_3 & & \\ & & & \ddots & \\ & & & & \lambda_m \end{bmatrix}$$

will be real and ≥0

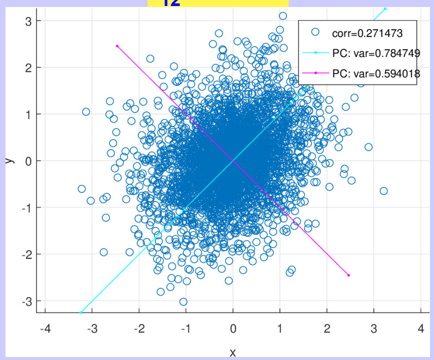
- $\sqrt{\lambda_i}$ = the **weights**
  - **(i.e., the variances)**
  - **with** $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \cdots \geq \lambda_m \geq 0$
- eigenvectors $\vec{p}_i$ = the **principal components**

Principal Components of the Data

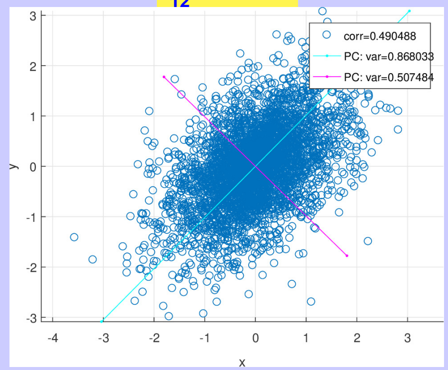
$r_{12} = 0$



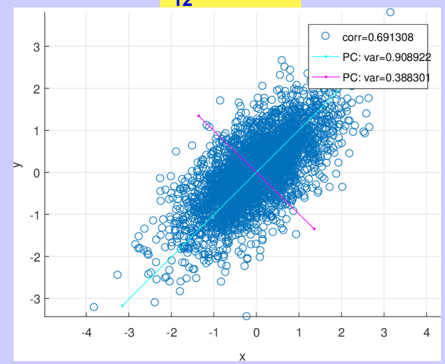
$r_{12} = 0.27$



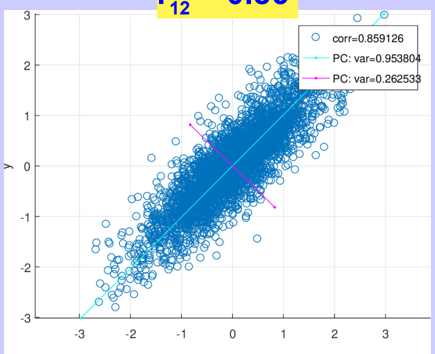
$r_{12} = 0.49$



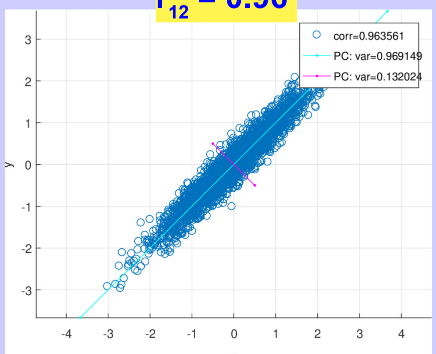
$r_{12} = 0.69$



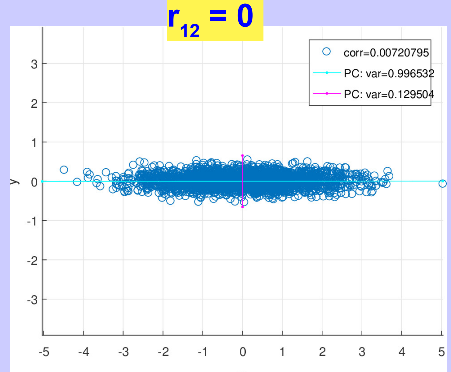
$r_{12} = 0.86$



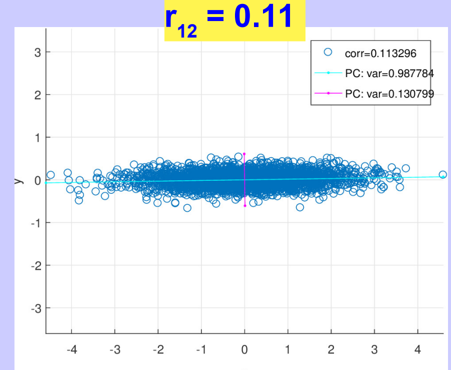
$r_{12} = 0.96$



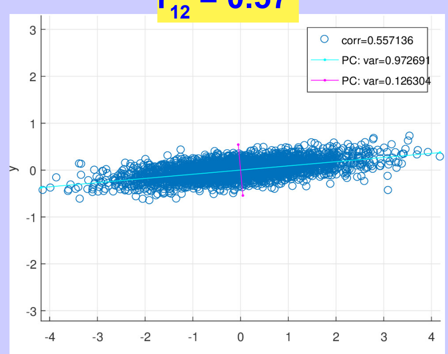
$r_{12} = 0$



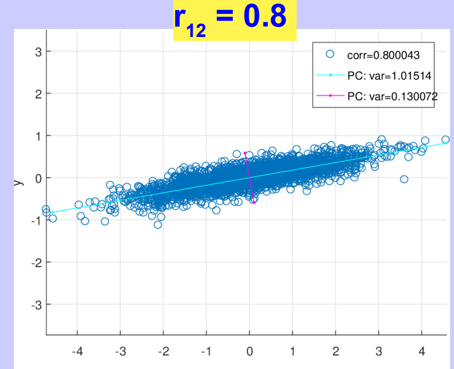
$r_{12} = 0.11$



$r_{12} = 0.57$



$r_{12} = 0.8$

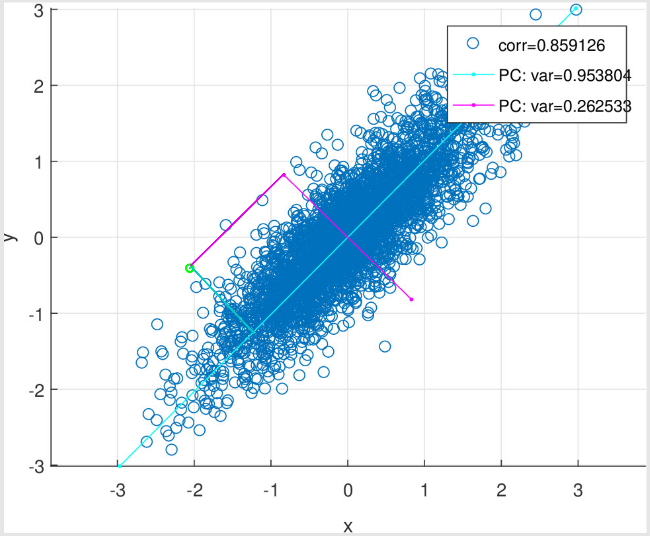


5000 x 2 matrices
(each point is a row)

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
  - 2^{nd} PC maximizes remaining variance; and so on

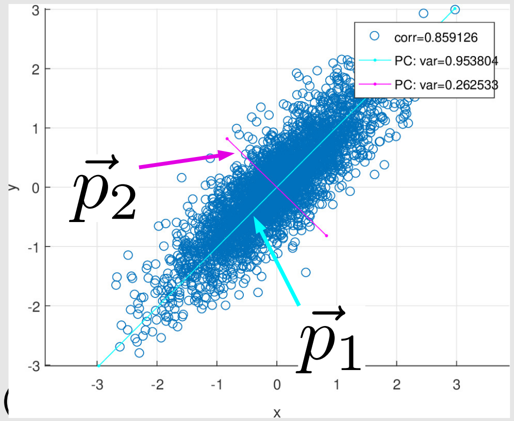
Properties of Covariance Matrices

- If S is a real mxm symmetric matrix ($s_{ij} = s_{ji}$)
  - 1. its eigenvalues are all real
    - $S\vec{p} = \lambda\vec{p}$. S symmetric $\rightarrow$ $\vec{p}^T S = \lambda\vec{p}^T$. $\rightarrow$ $\vec{p}^T S\overline{\vec{p}} = \lambda\vec{p}^T\overline{\vec{p}} = \lambda\|\vec{p}\|^2$
    - S real $\rightarrow$ $S\overline{\vec{p}} = \overline{\lambda}\,\overline{\vec{p}}$. $\rightarrow$ $\vec{p}^T S\overline{\vec{p}} = \overline{\lambda}\vec{p}^T\overline{\vec{p}} = \overline{\lambda}\|\vec{p}\|^2$.
    - hence $\lambda\|\vec{p}\|^2 = \overline{\lambda}\|\vec{p}\|^2 \rightarrow \lambda = \overline{\lambda} \rightarrow$ λ 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^TA (A real)
  - 4. its eigenvalues are all ≥ 0.
    - $A^T A\vec{p} = \lambda\vec{p} \longrightarrow \vec{p}^T A^T A\vec{p} = \lambda\vec{p}^T\vec{p} \longrightarrow (A\vec{p})^T A\vec{p} = \lambda\vec{p}^T\vec{p}$
    - $\longrightarrow \|A\vec{p}\|^2 = \lambda\|\vec{p}\|^2 \longrightarrow \lambda = \dfrac{\|A\vec{p}\|^2}{\|\vec{p}\|^2} \geq 0$.

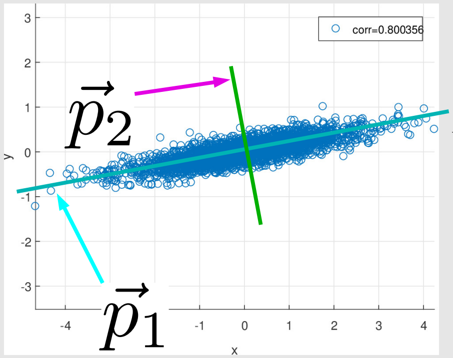
PCA Basis Diagonalises the Data

- eigenvectors orthonormal $\longrightarrow$ $PP^T = I \longrightarrow$ $P^T = P^{-1}$
- 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: $G \triangleq (F^T F)/n$
  - $nG = F^T F = P^T \tilde{A}^T \tilde{A} P = nP^T S P = nP^T P \Lambda P^T P = n\Lambda$

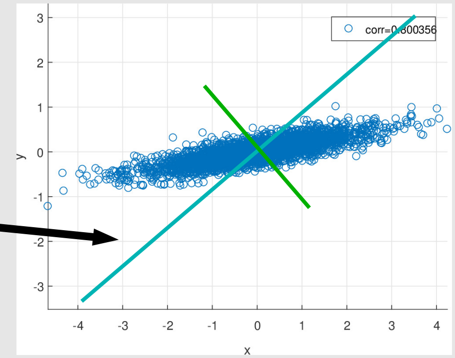
    $= \Lambda$ (diagonal) ← **Data projected on PC basis becomes UNCORRELATED**
  - the diagonal entries are the variances of the data projected along $\vec{p}_i$ (recall: from defn. of covariance matrix)



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:
    - $S \triangleq \frac{1}{n}\tilde{A}^T\tilde{A} = \frac{1}{n}V\Sigma^T U^T U\Sigma V^T = \boxed{V \boxed{\frac{\Sigma^T\Sigma}{n}} V^T}$ $\quad \frac{1}{n}\begin{bmatrix} \sigma_1^2 & & & & \\ & \sigma_2^2 & & & \\ & & \sigma_3^2 & & \\ & & & \ddots & \\ & & & & \sigma_m^2 \end{bmatrix}$
    - recall PCA: $S = \boxed{P\boxed{\Lambda}P^T}$ IDENTICAL FORM
      diagonal and ≥ 0
  - i.e., can use the SVD of Ã for PCA:
    - **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 $\sigma_i \triangleq \sqrt{\lambda_i}$, $V \triangleq P$
  - what about U?

    more work, because (we had assumed) n ≥ m

    nxn

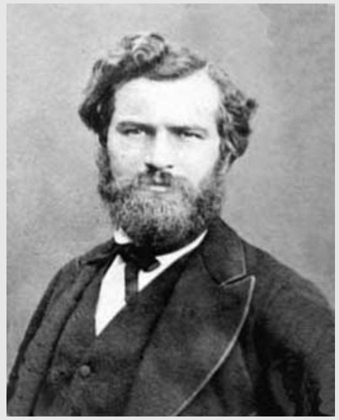
    - just eigendecompose $\hat{S} \triangleq AA^T = Q\hat{\Lambda}Q^T$; then $U \triangleq Q$
    - can also get V from the **same** eigendecomposition
      - $A = U\Sigma V^T \rightarrow U^T A = \Sigma V^T \rightarrow A^T U = V \Sigma^T \rightarrow$ $\vec{v}_i = \frac{A^T \vec{u}_i}{\sigma_i}$, $i = 1, \cdots, m$
  - set $\sigma_i \triangleq \sqrt{\lambda_i}$

    if $\sigma_i = 0$, choose v_i arbitrarily to complete orthonormal basis for V

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

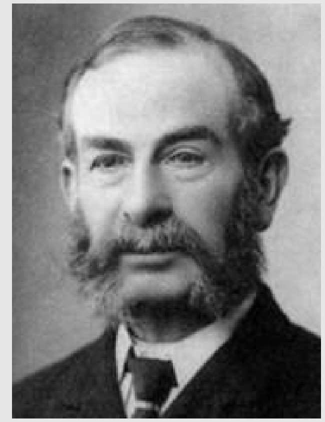
Who Invented the SVD?

- SVD: "**Swiss Army Knife**" of numerical analysis

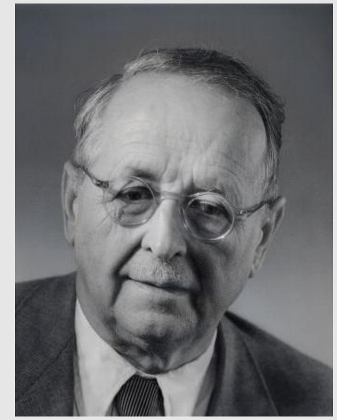


Camille Jordan
1838-1922

Erhardt Schmidt
1878-1959



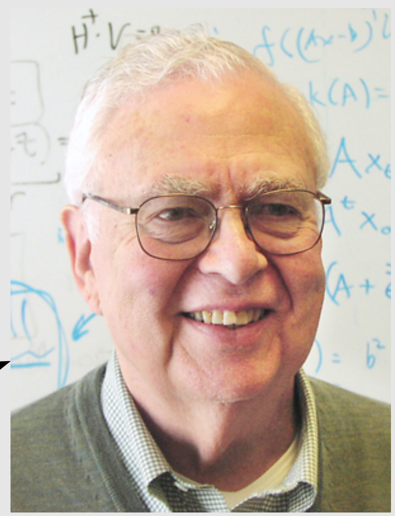
James Joseph
Sylvester 1814-97



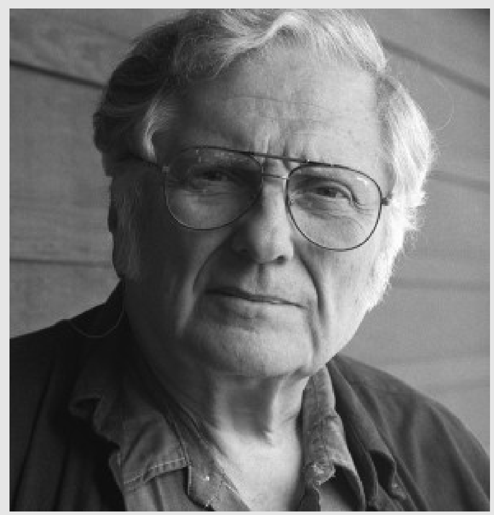
Hermann Weyl
1885-1955

Eugenio Beltrami
1835-1990

proposed the SVD
via eigendecomposition
of $A^T A$ or $A A^T$



Gene Golub
1932-2007



Bill Kahan
UCB EECS

Jim Demmel
UCB EECS

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