IERG4300
Web-Scale Information Analytics

Dimension Reduction

Prof. Wing C. Lau
Department of Information Engineering
wclau@ie.cuhk.edu.hk

Acknowledgements

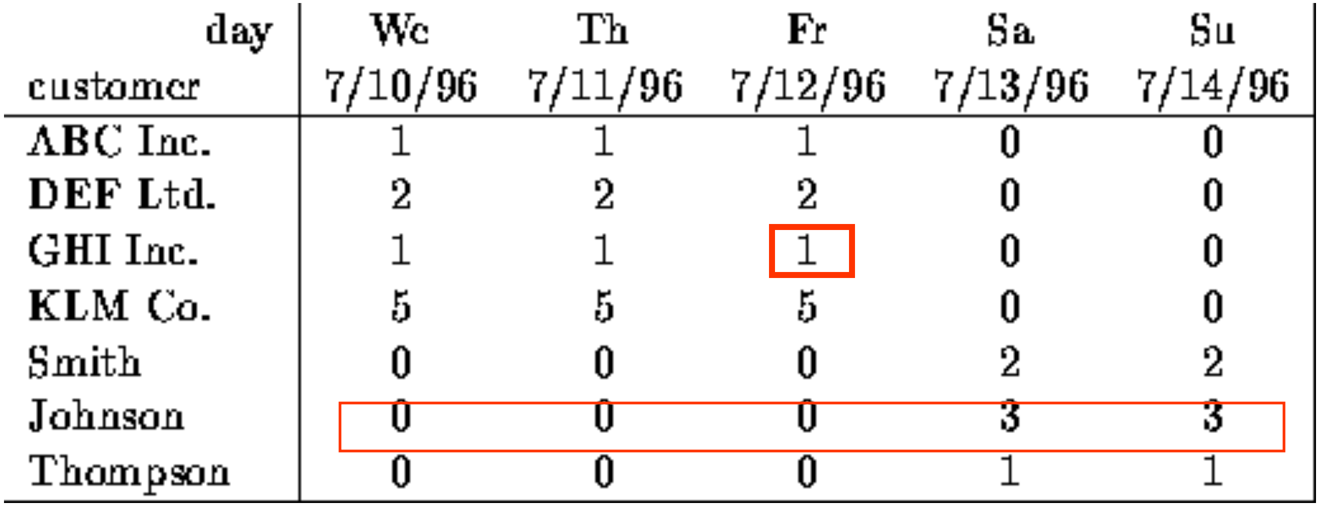
- The slides used in this chapter are adapted from:
  - CS246 Mining Massive Data-sets, by Jure Leskovec, Stanford University.
  - http://www.astro.princeton.edu/~gk/A542/PCA.ppt
  - http://myweb.dal.ca/~hwhitehe/BIOL4062/pca.ppt

with the author’s permission. All copyrights belong to the original author of the material.

Dimensionality Reduction

Compress / reduce dimensionality:

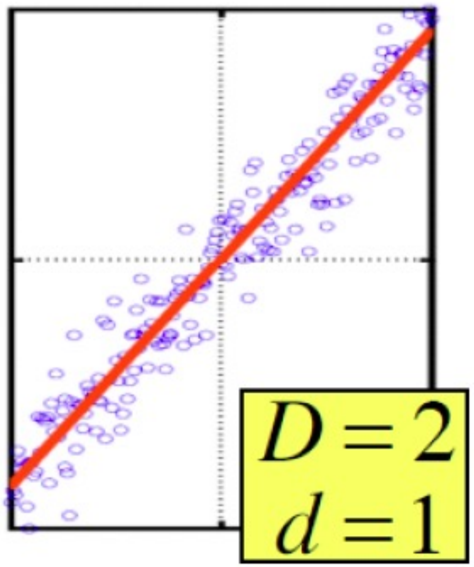
- Matrix of 10^6 rows; 10^3 columns; no updates
- Random access to any cell(s); **small error: OK**

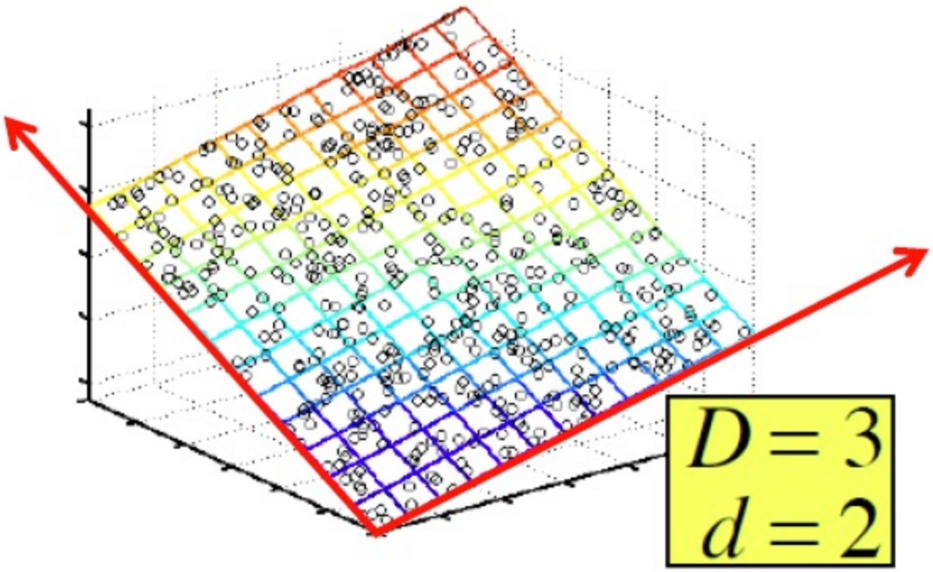


day	We	Th	Fr	Sa	Su
customer	7/10/96	7/11/96	7/12/96	7/13/96	7/14/96
ABC Inc.	1	1	1	0	0
DEF Ltd.	2	2	2	0	0
GHI Inc.	1	1	1	0	0
KLM Co.	5	5	5	0	0
Smith	0	0	0	2	2
Johnson	0	0	0	3	3
Thompson	0	0	0	1	1

The above matrix is really "2-dimensional." All rows can be reconstructed by scaling [1 1 1 0 0] or [0 0 0 1 1]

Dimensionality Reduction



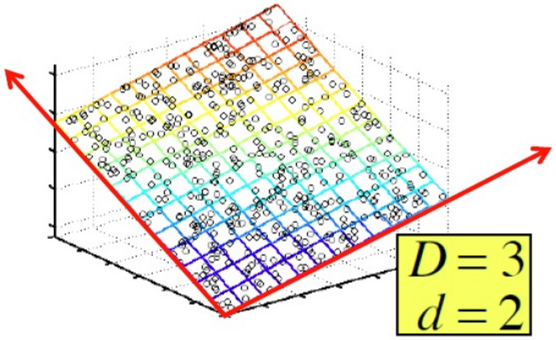


- **Assumption:** Data lies on or near a low *d*-dimensional subspace
- **Axes of this subspace are effective representation of the data**

Why Reduce Dimensions?

Why reduce dimensions?

- **Discover hidden correlations between different attributes of an object**
  - Words that occur commonly together for documents of the same topic
- **Remove redundant and noisy features**
  - Not all words are useful
- **Interpretation and visualization**
- **Easier storage and processing of the data**



An Initial Example

M = U V : a Users-to-Movies Rating matrix

- Consider each of the **6** users as a data point (a row in M) characterized by his/her rating on the **5** movies, i.e. each User is a **5**-dimensional data-point.
- **IF** we can decompose the **6**x**5** matrix (M) into the product of U and V, i.e. the (**6**x**2**) and (**2**x**5**) matrices, we can represent each of the **6** users, as well as the 5 movies, as data-points on a new **2**-dimensional space.

[2 0 3 1 4]
= u31 * [v11 v12 v13 v14 v15] + u32 * [v21 v22 v23 v24 v25]

$$\begin{array}{c} \\ \uparrow \\ \\ \text{Users} \\ \\ \downarrow \end{array}
\begin{array}{ccccc} \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\ \end{array}
\left[\begin{array}{ccccc} 5 & 2 & 4 & 4 & 3 \\ 3 & 1 & 2 & 4 & 1 \\ 2 & 0 & 3 & 1 & 4 \\ 2 & 5 & 4 & 3 & 5 \\ 4 & 4 & 5 & 4 & 0 \\ 1 & 0 & 0 & 5 & 5 \end{array}\right]
= \begin{bmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \\ u_{31} & u_{32} \\ u_{41} & u_{42} \\ u_{51} & u_{52} \\ u_{61} & u_{62} \end{bmatrix}
\times \begin{bmatrix} v_{11} & v_{12} & v_{13} & v_{14} & v_{15} \\ v_{21} & v_{22} & v_{23} & v_{24} & v_{25} \end{bmatrix}$$

Coordinates of User 3 in the new **2**-dimensional space

This vector in the **old** 5-dim. space gives the direction corresponding to the 1st axis of the **new** 2-dim space

An Initial Example

M = U V : a Users-to-Movies Rating matrix

- Consider each of the **6** users as a data point (a row in M) characterized by his/her rating on the **5** movies, i.e. each User is a **5**-dimensional data-point.
- **IF** we can decompose the **6**x**5** matrix (M) into the product of U and V, i.e. the (**6**x**2**) and (**2**x**5**) matrices, we can represent each of the **6** users, as well as the 5 movies, as data-points on a new **2**-dimensional space.

$[5\ 3\ 2\ 2\ 4\ 1]^T$

$= v11 * [u11\ u21\ u31\ u41\ u51\ u61]^T + v21 * [u12\ u22\ u32\ u42\ u42\ u62]^T$

$$\begin{array}{cc} & \begin{array}{ccccc}\text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie}\end{array} \\ \text{Users} & \begin{bmatrix} 5 & 2 & 4 & 4 & 3 \\ 3 & 1 & 2 & 4 & 1 \\ 2 & 0 & 3 & 1 & 4 \\ 2 & 5 & 4 & 3 & 5 \\ 4 & 4 & 5 & 4 & 0 \\ 1 & 0 & 0 & 5 & 5 \end{bmatrix} \end{array} = \begin{bmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \\ u_{31} & u_{32} \\ u_{41} & u_{42} \\ u_{51} & u_{52} \\ u_{61} & u_{62} \end{bmatrix} \mathbf{x} \begin{bmatrix} v_{11} & v_{12} & v_{13} & v_{14} & v_{15} \\ v_{21} & v_{22} & v_{23} & v_{24} & v_{25} \end{bmatrix}$$

This vector in the **old** 6-dim. space gives the direction corresponding to the 1st axi of the **new** 2-dim space

Coordinates of Movie1 in the new **2**-dimensional space

An Initial Example

M = U V : a Users-to-Movies Rating matrix

- Consider each of the **6** users as a data point (a row in M) characterized by his/her rating on the **5** movies, i.e. each User is a **5**-dimensional data-point.
- **IF** we can decompose the **6**x**5** matrix (M) into the product of U and V, i.e. the (**6**x**2**) and (**2**x**5**) matrices, we can represent each of the **6** users, as well as the 5 movies, as data-points on a new **2**-dimensional space.

The rating of User 3 on Movie 1 can be computed as the similarity (recall: projection or dot-product) between User 3 and Movie1 in the new **2**-dimensional space

Coordinates of User 3 in the new **2**-dimensional space

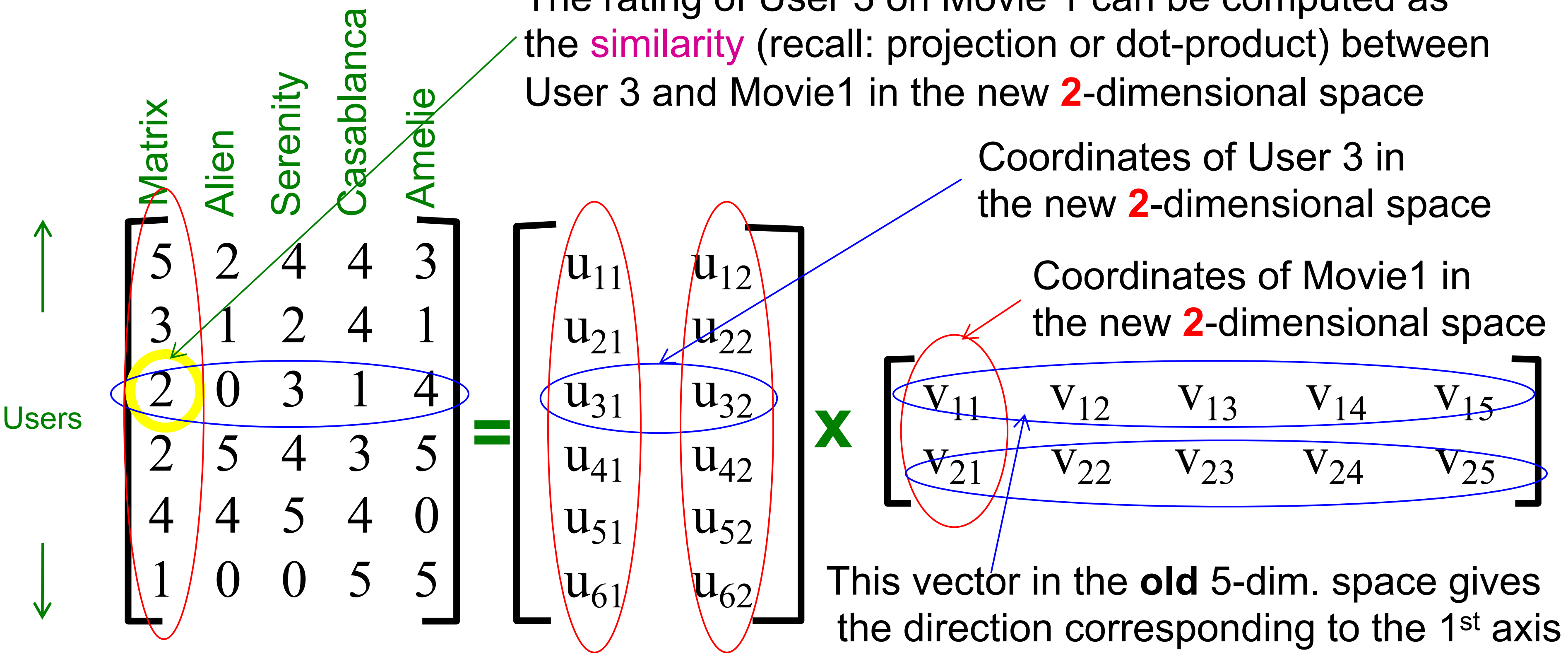
Coordinates of Movie1 in the new **2**-dimensional space

$$\begin{array}{cc} & \begin{array}{ccccc}\text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie}\end{array} \\ \text{Users} & \begin{bmatrix} 5 & 2 & 4 & 4 & 3 \\ 3 & 1 & 2 & 4 & 1 \\ 2 & 0 & 3 & 1 & 4 \\ 2 & 5 & 4 & 3 & 5 \\ 4 & 4 & 5 & 4 & 0 \\ 1 & 0 & 0 & 5 & 5 \end{bmatrix} \end{array} = \begin{bmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \\ u_{31} & u_{32} \\ u_{41} & u_{42} \\ u_{51} & u_{52} \\ u_{61} & u_{62} \end{bmatrix} \times \begin{bmatrix} v_{11} & v_{12} & v_{13} & v_{14} & v_{15} \\ v_{21} & v_{22} & v_{23} & v_{24} & v_{25} \end{bmatrix}$$

An Initial Example

M = U V : a Users-to-Movies Rating matrix

- Consider each of the 6 users as a data point (a row in M) characterized by his/her rating on the 5 movies, i.e. each User is a 5-dimensional data-point.
- IF we can decompose the 6x5 matrix (M) into the product of U and V, i.e. the (6x2) and (2x5) matrices, we can represent each of the 6 users, as well as the 5 movies, as data-points on a new 2-dimensional space.



$$\begin{array}{c|ccccc} & \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\ \hline & 5 & 2 & 4 & 4 & 3 \\ & 3 & 1 & 2 & 4 & 1 \\ \text{Users} & 2 & 0 & 3 & 1 & 4 \\ & 2 & 5 & 4 & 3 & 5 \\ & 4 & 4 & 5 & 4 & 0 \\ & 1 & 0 & 0 & 5 & 5 \end{array} = \begin{bmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \\ u_{31} & u_{32} \\ u_{41} & u_{42} \\ u_{51} & u_{52} \\ u_{61} & u_{62} \end{bmatrix} \mathbf{x} \begin{bmatrix} v_{11} & v_{12} & v_{13} & v_{14} & v_{15} \\ v_{21} & v_{22} & v_{23} & v_{24} & v_{25} \end{bmatrix}$$

The rating of User 3 on Movie 1 can be computed as the similarity (recall: projection or dot-product) between User 3 and Movie1 in the new 2-dimensional space

Coordinates of User 3 in the new 2-dimensional space

Coordinates of Movie1 in the new 2-dimensional space

This vector in the **old** 5-dim. space gives the direction corresponding to the 1st axis of the **new** 2-dim space

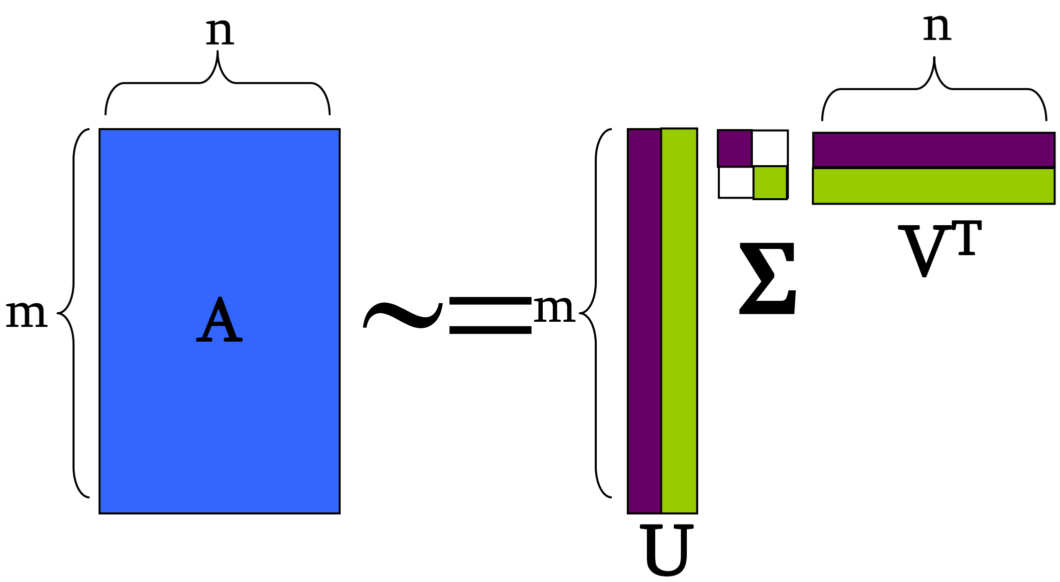
SVD - Definition

$$\boxed{\mathbf{A}_{[\mathbf{m \times n}]} = \mathbf{U}_{[\mathbf{m \times r}]}\, \mathbf{\Sigma}_{[\,\mathbf{r \times r}]}\, (\mathbf{V}_{[\mathbf{n \times r}]})^{T}}$$

- **A**: **Input data matrix**
  - *m* x *n* matrix (e.g., *m* documents, *n* attributes or features, e.g. terms)
- **U**: **Left singular vectors**
  - *m* x *r* , column orthonormal matrix, (*m* documents, *r* hidden/latent concepts)
- **Σ**: **Singular values**
  - *r* x *r* diagonal matrix (strength of each hidden/latent 'concept') (*r* : rank of the matrix **A**)
- **V**: **Right singular vectors**
  - *n* x *r,* column orthonormal matrix (*n* attributes or features, e.g. terms, and *r* hidden/latent concepts)

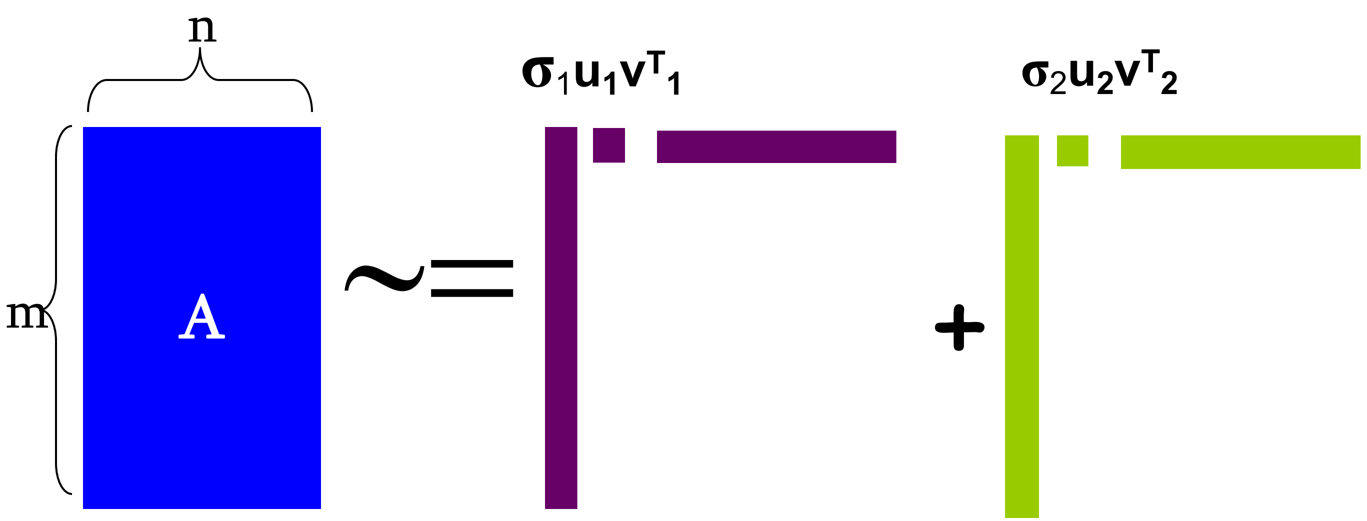
SVD

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T = \sum_i \sigma_i \mathbf{u}_i \circ \mathbf{v}_i^{\mathrm{T}}$$



SVD

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \sum_i \sigma_i \mathbf{u}_i \circ \mathbf{v}_i^{\mathrm{T}}$$



σ_i … scalar

u_i … vector

v_i … vector

SVD - Properties

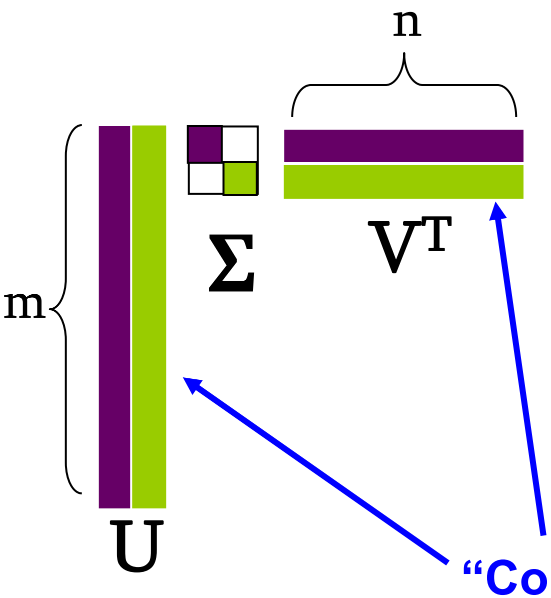
It is **always** possible to decompose a real matrix $\boldsymbol{A}$ into $\boldsymbol{A = U \Sigma V^T}$, where

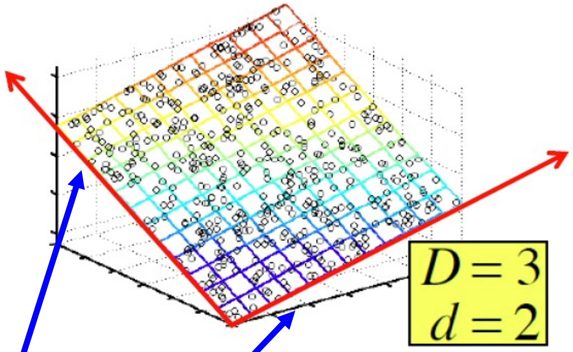
- $\boldsymbol{U, \Sigma, V}$: unique
- $\boldsymbol{U, V}$: column orthonormal
  - $\boldsymbol{U^T U = I; V^T V = I}$ ($\boldsymbol{I}$: identity matrix)
  - (Columns are orthogonal unit vectors)
- $\boldsymbol{\Sigma}$: diagonal
  - Entries (**singular values**) are positive,
    and sorted in decreasing order ($\boldsymbol{\sigma_1 \geq \sigma_2 \geq \ldots \geq 0}$)

SVD – Example: Users-to-Movies

- **$A = U\,\Sigma\,V^T$ - example: Users to Movies**

$$\begin{array}{c|ccccc} & \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\ \hline \text{SciFi Fans} & 1 & 1 & 1 & 0 & 0 \\ & 3 & 3 & 3 & 0 & 0 \\ & 4 & 4 & 4 & 0 & 0 \\ & 5 & 5 & 5 & 0 & 0 \\ \text{Romance Fans} & 0 & 2 & 0 & 4 & 4 \\ & 0 & 0 & 0 & 5 & 5 \\ & 0 & 1 & 0 & 2 & 2 \end{array} =$$





"Concepts"
AKA Latent dimensions
AKA Latent factors

Each row of A represents a User (a data point) who is characterized by the ratings he/she gave to a set of Movies

SVD – Example: Users-to-Movies

- $A = U \Sigma V^T$ - example: **Users to Movies**

$$
\begin{array}{c|ccccc}
 & \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\
\hline
\text{SciFi Fans} & \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\
 & \mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\
 & \mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\
 & \mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\
\text{Romance Fans} & 0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\
 & 0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\
 & 0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2}
\end{array}
=
\begin{bmatrix}
\mathbf{0.13} & 0.02 & -0.01 \\
\mathbf{0.41} & 0.07 & -0.03 \\
\mathbf{0.55} & 0.09 & -0.04 \\
\mathbf{0.68} & 0.11 & -0.05 \\
0.15 & \mathbf{-0.59} & \mathbf{0.65} \\
0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\
0.07 & \mathbf{-0.29} & \mathbf{0.32}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{12.4} & 0 & 0 \\
0 & \mathbf{9.5} & 0 \\
0 & 0 & \mathbf{1.3}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\
0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\
0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09
\end{bmatrix}
$$

SVD – Example: Users-to-Movies

- $A = U \Sigma V^T$ - example: **Users to Movies**

$$
\begin{array}{c|ccccc|}
 & \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\
\hline
\text{SciFi Fans} & \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\
 & \mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\
 & \mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\
 & \mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\
\text{Romance Fans} & 0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\
 & 0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\
 & 0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2}
\end{array}
=
\begin{array}{|ccc|}
\text{SciFi-concept} & \text{Romance-concept} & \\
\hline
\mathbf{0.13} & 0.02 & -0.01 \\
\mathbf{0.41} & 0.07 & -0.03 \\
\mathbf{0.55} & 0.09 & -0.04 \\
\mathbf{0.68} & 0.11 & -0.05 \\
0.15 & \mathbf{-0.59} & \mathbf{0.65} \\
0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\
0.07 & \mathbf{-0.29} & \mathbf{0.32}
\end{array}
\times
\begin{bmatrix}
\mathbf{12.4} & 0 & 0 \\
0 & \mathbf{9.5} & 0 \\
0 & 0 & \mathbf{1.3}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\
0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\
0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09
\end{bmatrix}
$$

SVD – Example: Users-to-Movies

- **A = U Σ V^T - example:**

***U* is "user-to-concept" similarity matrix**

$$
\begin{array}{c|ccccc}
 & \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\
\hline
\text{SciFi Fans} & \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\
 & \mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\
 & \mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\
 & \mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\
\text{Romance Fans} & 0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\
 & 0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\
 & 0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2}
\end{array}
=
\begin{array}{ccc}
\text{SciFi-concept} & \text{Romance-concept} & \\
\hline
\mathbf{0.13} & 0.02 & -0.01 \\
\mathbf{0.41} & 0.07 & -0.03 \\
\mathbf{0.55} & 0.09 & -0.04 \\
\mathbf{0.68} & 0.11 & -0.05 \\
0.15 & \mathbf{-0.59} & \mathbf{0.65} \\
0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\
0.07 & \mathbf{-0.29} & \mathbf{0.32}
\end{array}
\times
\begin{bmatrix}
\mathbf{12.4} & 0 & 0 \\
0 & \mathbf{9.5} & 0 \\
0 & 0 & \mathbf{1.3}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\
0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\
0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09
\end{bmatrix}
$$

SVD – Example: Users-to-Movies

- $A = U \Sigma V^T$ - example:

SciFi-concept

"strength" of the SciFi-concept

$$
\begin{array}{c|ccccc}
 & \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\
\hline
\text{SciFi} & \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\
\text{Fans} & \mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\
 & \mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\
 & \mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\
 & 0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\
\text{Romance} & 0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\
\text{Fans} & 0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2}
\end{array}
=
\begin{bmatrix}
\mathbf{0.13} & 0.02 & -0.01 \\
\mathbf{0.41} & 0.07 & -0.03 \\
\mathbf{0.55} & 0.09 & -0.04 \\
\mathbf{0.68} & 0.11 & -0.05 \\
0.15 & \mathbf{-0.59} & \mathbf{0.65} \\
0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\
0.07 & \mathbf{-0.29} & \mathbf{0.32}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{12.4} & 0 & 0 \\
0 & \mathbf{9.5} & 0 \\
0 & 0 & \mathbf{1.3}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\
0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\
0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09
\end{bmatrix}
$$

SVD – Example: Users-to-Movies

- $A = U\ \Sigma\ V^T$ - example:

V is "movie-to-concept" similarity matrix

SciFi-concept

$$
\begin{array}{c|ccccc|}
 & \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\
\hline
\text{SciFi Fans} & \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\
 & \mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\
 & \mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\
 & \mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\
\text{Romance Fans} & 0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\
 & 0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\
 & 0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2}
\end{array}
=
\begin{bmatrix}
\mathbf{0.13} & 0.02 & -0.01 \\
\mathbf{0.41} & 0.07 & -0.03 \\
\mathbf{0.55} & 0.09 & -0.04 \\
\mathbf{0.68} & 0.11 & -0.05 \\
0.15 & \mathbf{-0.59} & \mathbf{0.65} \\
0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\
0.07 & \mathbf{-0.29} & \mathbf{0.32}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{12.4} & 0 & 0 \\
0 & \mathbf{9.5} & 0 \\
0 & 0 & \mathbf{1.3}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\
0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\
0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09
\end{bmatrix}
$$

SciFi-concept

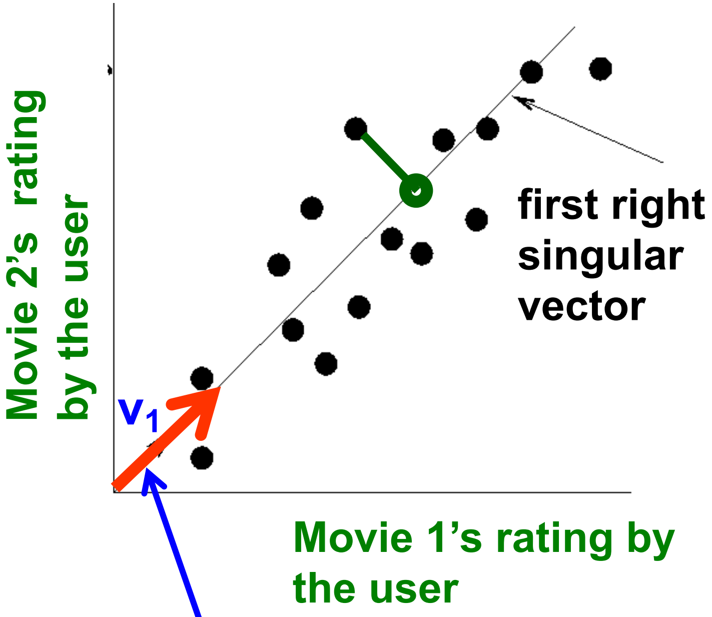
SVD - Interpretation #1

‘**movies**’, ‘**users**’ and ‘**concepts**’:

- ***U***: user-to-concept similarity matrix
- ***V***: movie-to-concept similarity matrix
- **Σ**: its diagonal elements:
  ‘strength’ of each concept

SVD - Interpretation #2 – Choose a new axis to Minimize total “project errors”

- **$A = U\ \Sigma\ V^T$ - example:**
  - **V**: “movie-to-concept” matrix
  - **U**: “user-to-concept” matrix

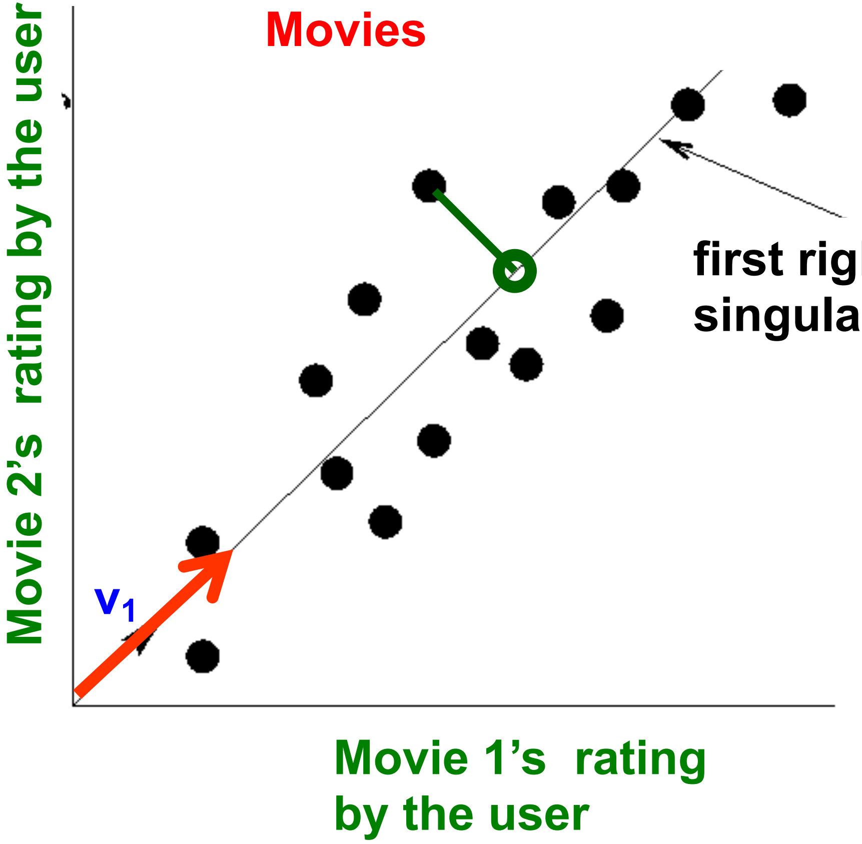


$$\begin{bmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\ \mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\ \mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\ \mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\ 0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\ 0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\ 0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2} \end{bmatrix} = \begin{bmatrix} \mathbf{0.13} & 0.02 & -0.01 \\ \mathbf{0.41} & 0.07 & -0.03 \\ \mathbf{0.55} & 0.09 & -0.04 \\ \mathbf{0.68} & 0.11 & -0.05 \\ 0.15 & \mathbf{-0.59} & \mathbf{0.65} \\ 0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\ 0.07 & \mathbf{-0.29} & \mathbf{0.32} \end{bmatrix} \times \begin{bmatrix} \mathbf{12.4} & 0 & 0 \\ 0 & \mathbf{9.5} & 0 \\ 0 & 0 & \mathbf{1.3} \end{bmatrix} \times \begin{bmatrix} \boxed{\mathbf{0.56} \quad \mathbf{0.59} \quad \mathbf{0.56} \quad 0.09 \quad 0.09} \\ 0.12 \quad -0.02 \quad 0.12 \quad \mathbf{-0.69} \quad \mathbf{-0.69} \\ 0.40 \quad \mathbf{-0.80} \quad 0.40 \quad 0.09 \quad 0.09 \end{bmatrix}$$

SVD - interpretation #2

- **SVD gives ‘best’ axis to project on:**
  - ‘**best**’ = min sum of squares of projection errors
- **In other words, minimum reconstruction error**

Each user (a data point), e.g. in a 2-D space, is characterized by the ratings he/she gave to a set of 2 Movies



first right singular vector

SVD - interpretation #2 (more later)

- **SVD gives ‘best’ axis to project on:**
  - ‘**best**’ = min sum of squares of projection errors
- **In other words, minimum reconstruction error**

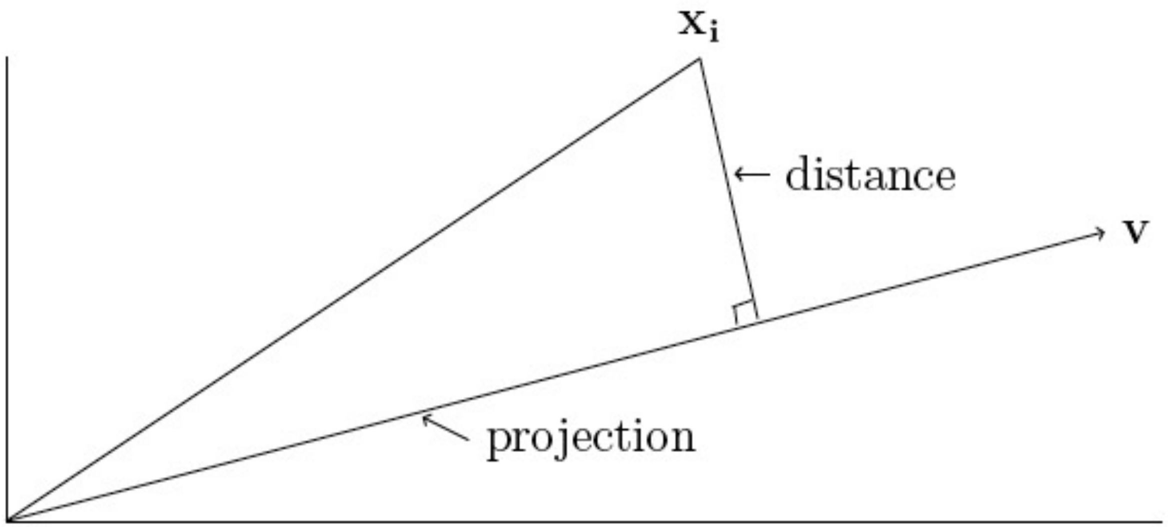
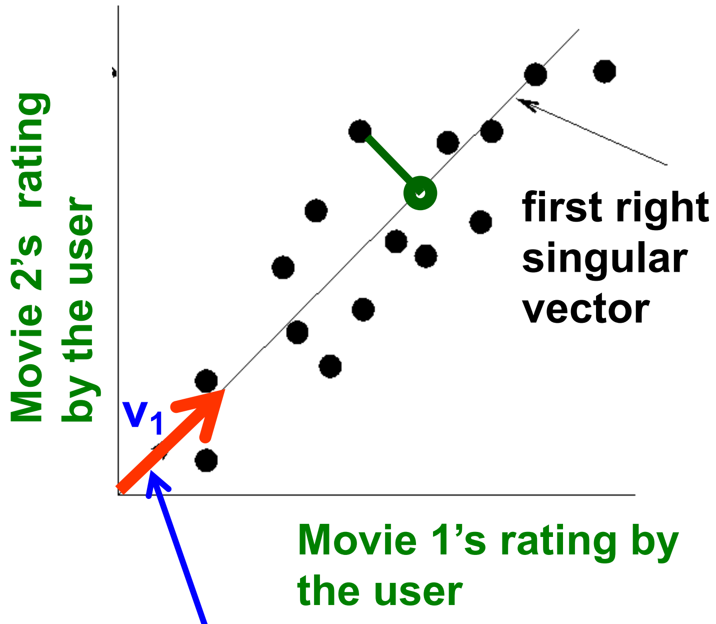


Figure 4.1: The projection of the point $\mathbf{x_i}$ onto the line through the origin in the direction of $\mathbf{v}$

SVD - Interpretation #2 (cont’d)

○ A = U Σ V^T - example:

- **V**: “movie-to-concept” matrix
- **U**: “user-to-concept” matrix



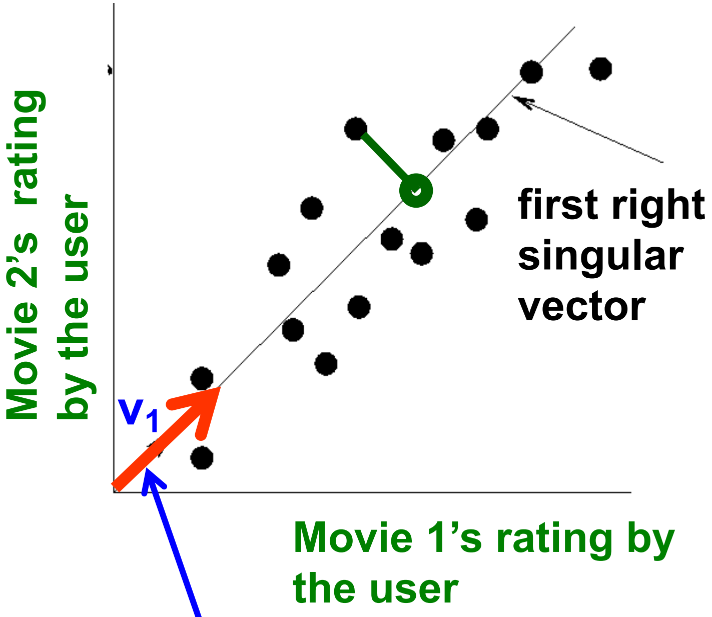
$$\begin{bmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\ \mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\ \mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\ \mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\ 0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\ 0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\ 0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2} \end{bmatrix} = \begin{bmatrix} \mathbf{0.13} & 0.02 & -0.01 \\ \mathbf{0.41} & 0.07 & -0.03 \\ \mathbf{0.55} & 0.09 & -0.04 \\ \mathbf{0.68} & 0.11 & -0.05 \\ 0.15 & \mathbf{-0.59} & \mathbf{0.65} \\ 0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\ 0.07 & \mathbf{-0.29} & \mathbf{0.32} \end{bmatrix} \times \begin{bmatrix} \mathbf{12.4} & 0 & 0 \\ 0 & \mathbf{9.5} & 0 \\ 0 & 0 & \mathbf{1.3} \end{bmatrix} \times \begin{bmatrix} \boxed{\mathbf{0.56} \quad \mathbf{0.59} \quad \mathbf{0.56} \quad 0.09 \quad 0.09} \\ 0.12 \quad -0.02 \quad 0.12 \quad \mathbf{-0.69} \quad \mathbf{-0.69} \\ 0.40 \quad \mathbf{-0.80} \quad 0.40 \quad 0.09 \quad 0.09 \end{bmatrix}$$

SVD - Interpretation #2 (cont’d)

$$\mathbf{A = U\,\Sigma\,V^T}$$

$$\mathbf{\Rightarrow A\,V = U\,\Sigma\,V^T\,V = U\,\Sigma}$$

- $\mathbf{U\,\Sigma}$: Gives the coordinates of the points in the projection axis



first right singular vector

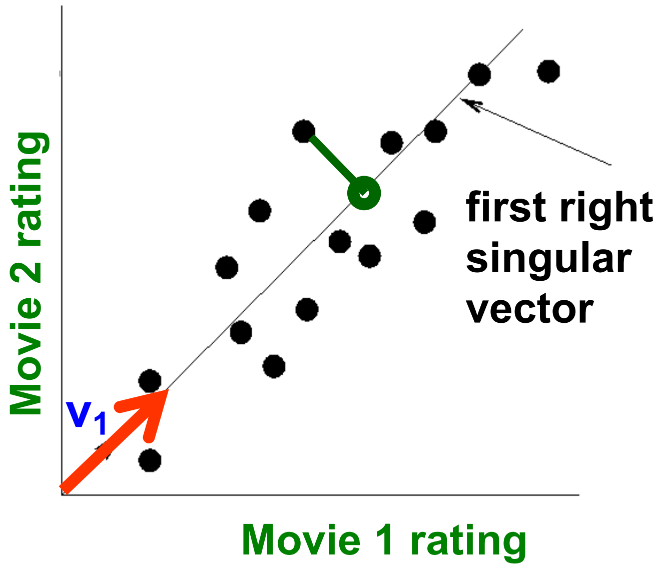
$$\begin{bmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\ \mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\ \mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\ \mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\ 0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\ 0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\ 0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2} \end{bmatrix} = \begin{bmatrix} \mathbf{0.13} & 0.02 & -0.01 \\ \mathbf{0.41} & 0.07 & -0.03 \\ \mathbf{0.55} & 0.09 & -0.04 \\ \mathbf{0.68} & 0.11 & -0.05 \\ 0.15 & \mathbf{-0.59} & \mathbf{0.65} \\ 0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\ 0.07 & \mathbf{-0.29} & \mathbf{0.32} \end{bmatrix} \times \begin{bmatrix} \mathbf{12.4} & 0 & 0 \\ 0 & \mathbf{9.5} & 0 \\ 0 & 0 & \mathbf{1.3} \end{bmatrix} \times \begin{bmatrix} \mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\ 0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

SVD - Interpretation #2 (cont’d)

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

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

- $U \Sigma$: Gives the coordinates of the points in the projection axis



$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} \leftarrow a_1$$

$a_1 \bullet v_1$

$$\begin{bmatrix} 1.61 & 0.19 & -0.01 \\ 5.08 & 0.66 & -0.03 \\ 6.82 & 0.85 & -0.05 \\ 8.43 & 1.04 & -0.06 \\ 1.86 & -5.60 & 0.84 \\ 0.86 & -6.93 & -0.87 \\ 0.86 & -2.75 & 0.41 \end{bmatrix}$$

Projection of users on the “Sci-Fi” axis: $A v_1$

SVD - interpretation #2 (more later)

- **SVD gives 'best' axis to project on:**
  - '**best**' = min sum of squares of projection errors
- **i.e. Choose the axis v to minimize reconstruction error**

  == Choose the axis v to maximize sum of square of projection length of all data points (i.e. each row in A)

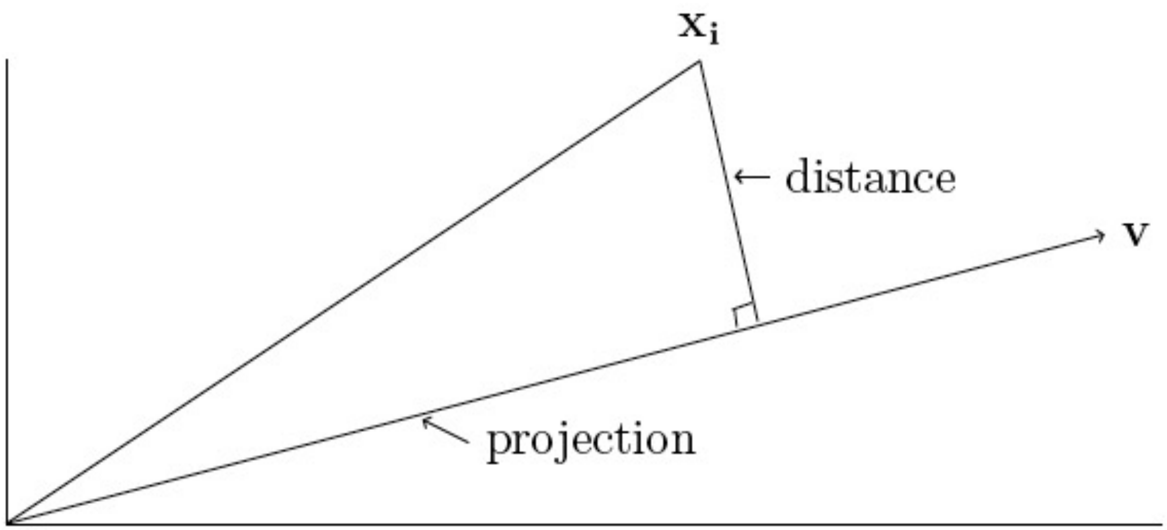
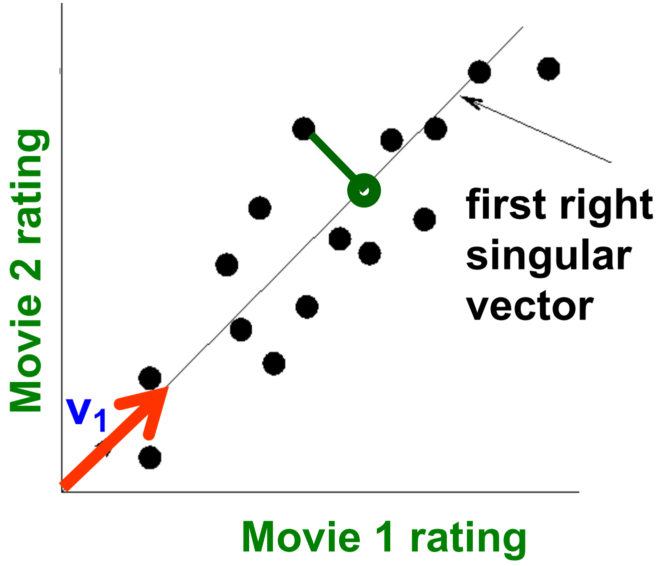


Figure 4.1: The projection of the point $\mathbf{x_i}$ onto the line through the origin in the direction of $\mathbf{v}$

SVD - Interpretation #2 (cont'd)

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

- $U \Sigma$: Gives the coordinates of the points in the projection axis



variance ('spread') on the v_1 axis: Maximize total spread of all data points along the axis defined by v_1 => minimize total projection errors

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} \mathbf{0.13} & 0.02 & -0.01 \\ \mathbf{0.41} & 0.07 & -0.03 \\ \mathbf{0.55} & 0.09 & -0.04 \\ \mathbf{0.68} & 0.11 & -0.05 \\ 0.15 & \mathbf{-0.59} & \mathbf{0.65} \\ 0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\ 0.07 & \mathbf{-0.29} & \mathbf{0.32} \end{bmatrix} \times \begin{bmatrix} \mathbf{12.4} & 0 & 0 \\ 0 & \mathbf{9.5} & 0 \\ 0 & 0 & \mathbf{1.3} \end{bmatrix} \times \begin{bmatrix} \mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\ 0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

SVD - Interpretation #2

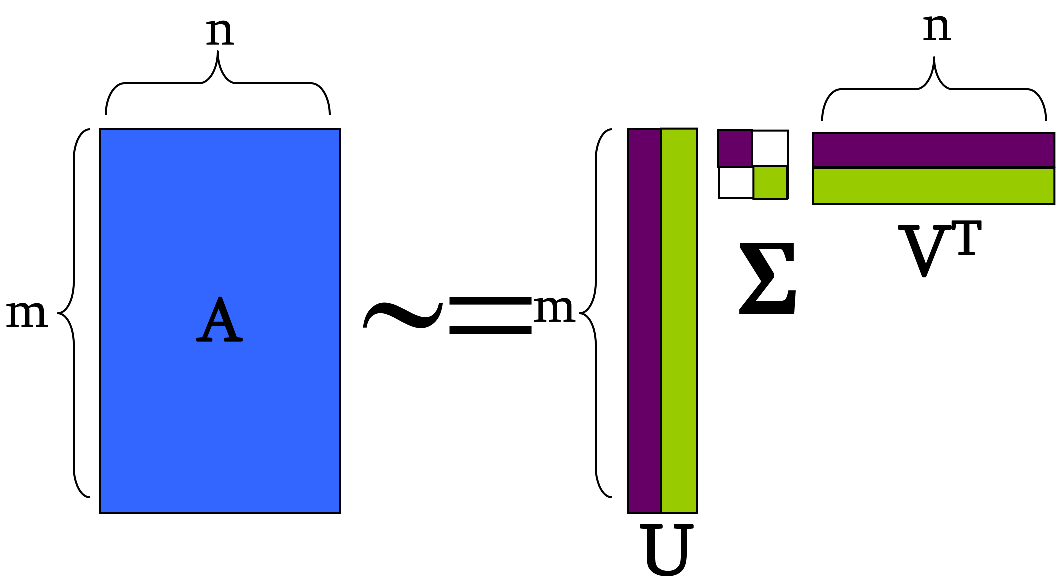
More details

- **Q: How exactly is dim. reduction done?**

$$\begin{bmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\ \mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\ \mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\ \mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\ 0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\ 0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\ 0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2} \end{bmatrix} = \begin{bmatrix} \mathbf{0.13} & 0.02 & -0.01 \\ \mathbf{0.41} & 0.07 & -0.03 \\ \mathbf{0.55} & 0.09 & -0.04 \\ \mathbf{0.68} & 0.11 & -0.05 \\ 0.15 & \mathbf{-0.59} & \mathbf{0.65} \\ 0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\ 0.07 & \mathbf{-0.29} & \mathbf{0.32} \end{bmatrix} \times \begin{bmatrix} \mathbf{12.4} & 0 & 0 \\ 0 & \mathbf{9.5} & 0 \\ 0 & 0 & \mathbf{1.3} \end{bmatrix} \times \begin{bmatrix} \mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\ 0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

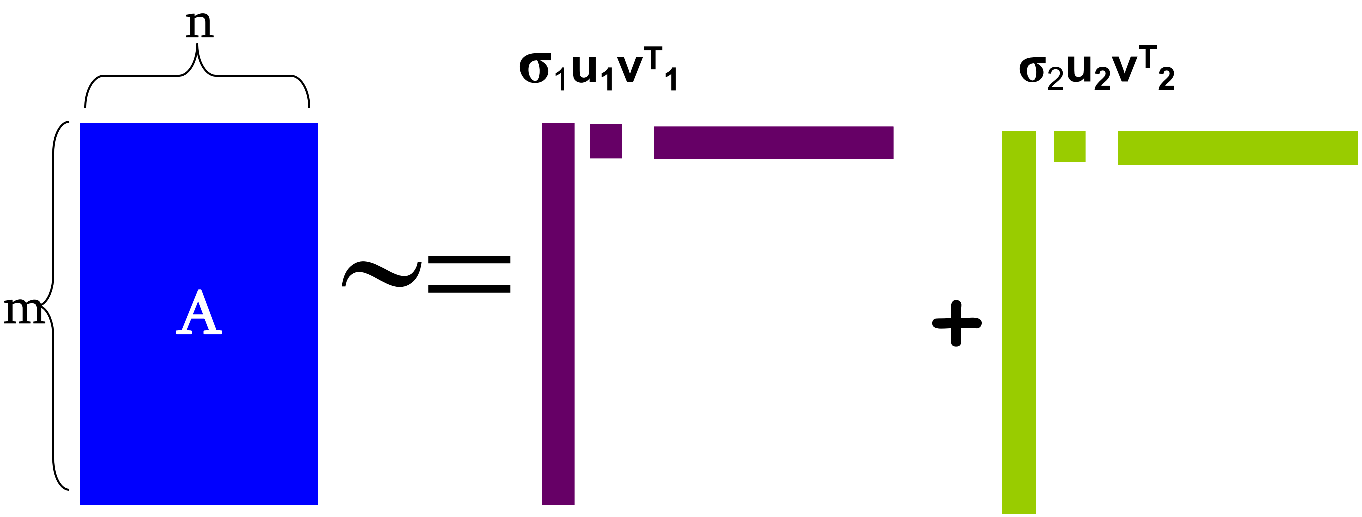
Recall: SVD

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \sum_i \sigma_i \mathbf{u}_i \circ \mathbf{v}_i^{\mathrm{T}}$$



Recall: SVD

$$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T = \sum_i \sigma_i \mathbf{u}_i \circ \mathbf{v}_i^{\mathrm{T}}$$



σ_i … scalar

u_i … vector

v_i … vector

SVD - Interpretation #2

More details

- **Q: How exactly is Dimension Reduction done?**
- **A: Set smallest singular values to zero**

$$
\begin{bmatrix}
\mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\
\mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\
\mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\
\mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\
0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\
0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\
0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2}
\end{bmatrix}
=
\begin{bmatrix}
\mathbf{0.13} & 0.02 & -0.01 \\
\mathbf{0.41} & 0.07 & -0.03 \\
\mathbf{0.55} & 0.09 & -0.04 \\
\mathbf{0.68} & 0.11 & -0.05 \\
0.15 & \mathbf{-0.59} & \mathbf{0.65} \\
0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\
0.07 & \mathbf{-0.29} & \mathbf{0.32}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{12.4} & 0 & 0 \\
0 & \mathbf{9.5} & 0 \\
0 & 0 & \cancel{1.3}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\
0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\
0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09
\end{bmatrix}
$$

SVD - Interpretation #2

More details

- **Q: How exactly is dim. reduction done?**
- **A: Set smallest singular values to zero**

$$\begin{bmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\ \mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\ \mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\ \mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\ 0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\ 0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\ 0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2} \end{bmatrix} \approx \begin{bmatrix} \mathbf{0.13} & 0.02 & -0.01 \\ \mathbf{0.41} & 0.07 & -0.03 \\ \mathbf{0.55} & 0.09 & -0.04 \\ \mathbf{0.68} & 0.11 & -0.05 \\ 0.15 & \mathbf{-0.59} & \mathbf{0.65} \\ 0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\ 0.07 & \mathbf{-0.29} & \mathbf{0.32} \end{bmatrix} \times \begin{bmatrix} \mathbf{12.4} & 0 & 0 \\ 0 & \mathbf{9.5} & 0 \\ 0 & 0 & \cancel{\mathbf{1.3}} \end{bmatrix} \times \begin{bmatrix} \mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\ 0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

SVD - Interpretation #2

More details

- Q: How exactly is dim. reduction done?
- A: Set smallest singular values to zero

$$\begin{bmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\ \mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\ \mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\ \mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\ 0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\ 0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\ 0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2} \end{bmatrix} \approx \begin{bmatrix} \mathbf{0.13} & 0.02 & \xcancel{-0.01} \\ \mathbf{0.41} & 0.07 & \xcancel{-0.03} \\ \mathbf{0.55} & 0.09 & \xcancel{-0.04} \\ \mathbf{0.68} & 0.11 & \xcancel{-0.05} \\ 0.15 & \mathbf{-0.59} & \xcancel{\mathbf{0.65}} \\ 0.07 & \mathbf{-0.73} & \xcancel{\mathbf{-0.67}} \\ 0.07 & \mathbf{-0.29} & \xcancel{\mathbf{0.32}} \end{bmatrix} \times \begin{bmatrix} \mathbf{12.4} & 0 & 0 \\ 0 & \mathbf{9.5} & 0 \\ 0 & 0 & \xcancel{\mathbf{1.3}} \end{bmatrix} \times \begin{bmatrix} \mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\ \xcancel{0.40} & \xcancel{\mathbf{-0.80}} & \xcancel{0.40} & \xcancel{0.09} & \xcancel{0.09} \end{bmatrix}$$

SVD - Interpretation #2

More details

- Q: How exactly is dim. reduction done?
- A: Set smallest singular values to zero

$$
\begin{bmatrix}
\mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\
\mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\
\mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\
\mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\
0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\
0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\
0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2}
\end{bmatrix}
\approx
\begin{bmatrix}
\mathbf{0.13} & 0.02 \\
\mathbf{0.41} & 0.07 \\
\mathbf{0.55} & 0.09 \\
\mathbf{0.68} & 0.11 \\
0.15 & \mathbf{-0.59} \\
0.07 & \mathbf{-0.73} \\
0.07 & \mathbf{-0.29}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{12.4} & 0 \\
0 & \mathbf{9.5}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\
0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69}
\end{bmatrix}
$$

SVD - Interpretation #2

More details

- Q: How exactly is dim. reduction done?
- A: Set smallest singular values to zero

$$\begin{bmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\ \mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\ \mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\ \mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\ 0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\ 0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\ 0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2} \end{bmatrix} \approx \begin{bmatrix} \mathbf{0.92} & \mathbf{0.95} & \mathbf{0.92} & 0.01 & 0.01 \\ \mathbf{2.91} & \mathbf{3.01} & \mathbf{2.91} & -0.01 & -0.01 \\ \mathbf{3.90} & \mathbf{4.04} & \mathbf{3.90} & 0.01 & 0.01 \\ \mathbf{4.82} & \mathbf{5.00} & \mathbf{4.82} & 0.03 & 0.03 \\ 0.70 & \mathbf{0.53} & 0.70 & \mathbf{4.11} & \mathbf{4.11} \\ -0.69 & 1.34 & -0.69 & \mathbf{4.78} & \mathbf{4.78} \\ 0.32 & \mathbf{0.23} & 0.32 & \mathbf{2.01} & \mathbf{2.01} \end{bmatrix}$$

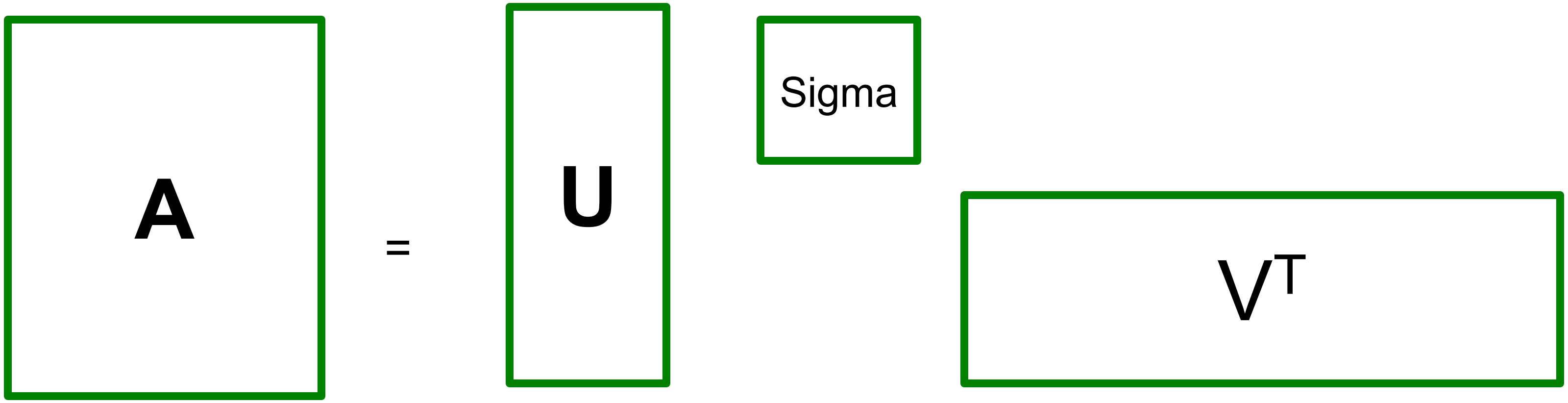
Frobenius norm:

$$\|M\|_F = \sqrt{\Sigma_{ij} M_{ij}^2}$$

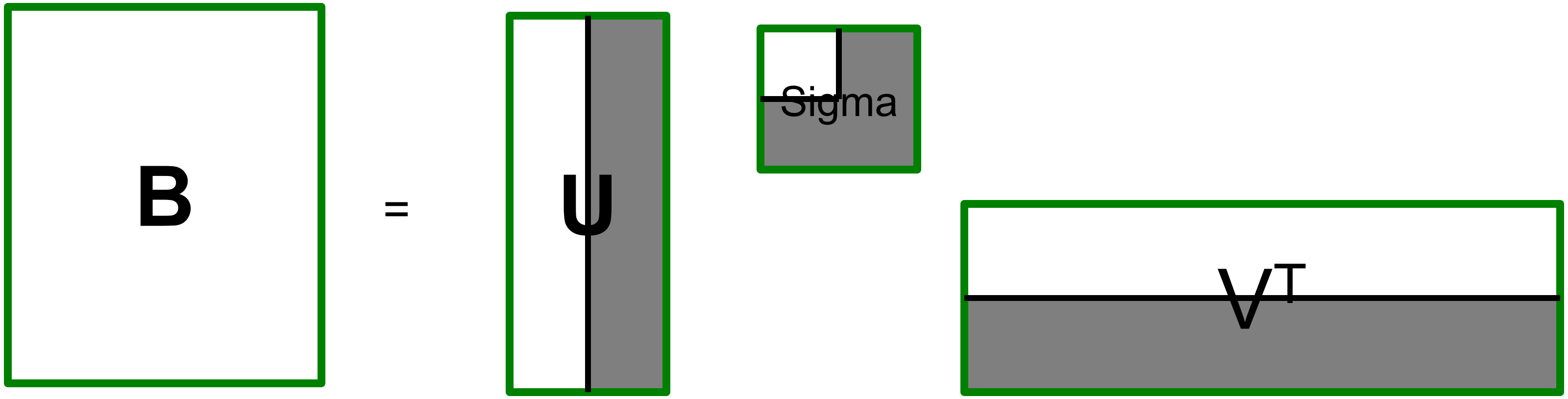
$$\|A-B\|_F = \sqrt{\Sigma_{ij} (A_{ij}-B_{ij})^2}$$

is “small”

SVD – Best Low Rank Approx.



B is best approximation of A



SVD – Best Low Rank Approx.

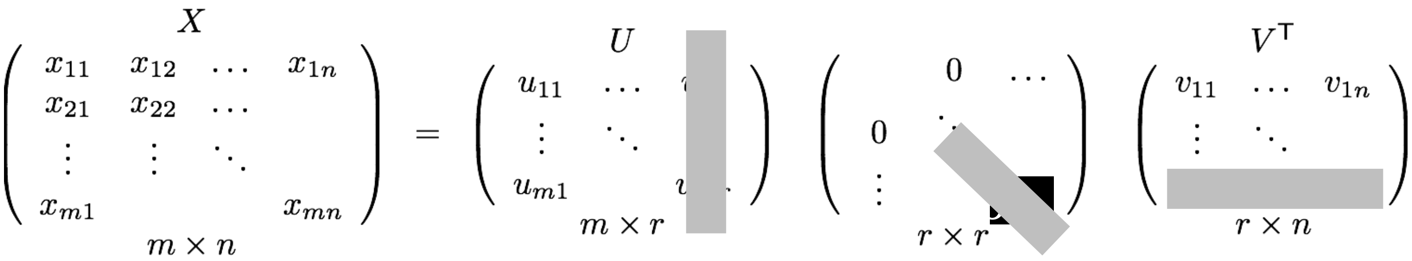
- **Theorem:** Let **A = U Σ V**^{T} ($\sigma_1 \geq \sigma_2 \geq \ldots$, rank(***A***)=***r***)

then B = U S V^T

- **S = diagonal** n**x**n **matrix** where $s_i = \sigma_i$ ($i=1\ldots k$) else $s_i=0$

is a best rank-***k*** approximation to ***A***:

- ***B*** **is a solution to** $\min_B \|A-B\|_F$ **where** $\mathrm{rank}(B)=k$



Backup Slides

Proof of SVD = Best Low Rank Approx. (Refer to Chapter 11.3.4. of [MMDS])

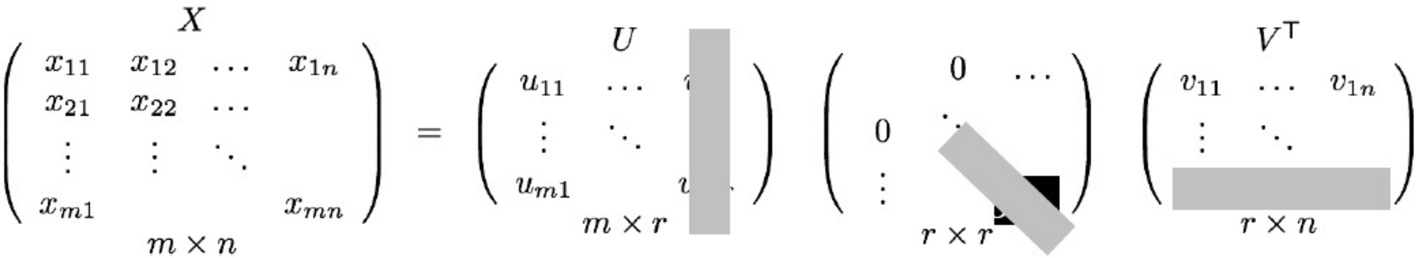
- **Theorem:** Let **A = U Σ V**^{T} $(\sigma_1 \geq \sigma_2 \geq \ldots$, rank(***A***)=***r***)

  then B = U S V^T

  - **S = diagonal** ***n*****x*****n*** **matrix** where $s_i = \sigma_i$ $(i=1 \ldots k)$ else $s_i = 0$

  is a best rank-***k*** approximation to ***A***:

  - ***B*** **is a solution to** $\min_B \|A-B\|_F$ **where** $\mathrm{rank}(B)=k$



- **We will need 2 facts:**
  - $||M||_F = \sum_i (q_{ii})^2$ where ***M* = *P Q R*** is SVD of ***M***
  - **U Σ V**^{T} **- U S V**^{T} **= U (Σ - S) V**^{T}

SVD – Best Low Rank Approx.

- **We will need 2 facts:**
  - $\left\|M\right\|_F = \sum_k (q_{kk})^2$ where ***M*** = ***P Q R*** is SVD of ***M***

$$\|M\| = \sum_i \sum_j (m_{ij})^2 = \sum_i \sum_j \Big(\sum_k \sum_\ell p_{ik} q_{k\ell} r_{\ell j} \Big)^2$$

$$\|M\| = \sum_i \sum_j \sum_k \sum_\ell \sum_n \sum_m p_{ik} q_{k\ell} r_{\ell j} p_{in} q_{nm} r_{mj}$$

$\sum_i p_{ik} p_{in}$ is 1 if $k = n$ and 0 otherwise

- **U Σ V^T - U S V^T = U (Σ - S) V^T**

We apply:

-- P column orthonormal

-- R row orthonormal

-- Q is diagonal

Proof of Fact #1

$$M = PQR$$

$$\left\|M\right\|_F = \sum_i \sum_j (m_{i,j})^2 = \sum_i \sum_j \Big(\sum_k \sum_l p_{i,k} q_{k,l} r_{l,j}\Big)^2 = \sum_i \sum_j \sum_k \sum_l \sum_m \sum_{\mathrm{n}} p_{i,k} q_{k,l} r_{l,j} p_{i,m} q_{m,n} r_{n,j}$$

$$\left\|M\right\|_F = \sum_j \sum_l \sum_n r_{l,j} r_{n,j} \left\{ \sum_k \sum_m q_{k,l} q_{m,n} \sum_i p_{i,k} p_{i,m} \right\}$$

Since $\sum_i p_{i,k} p_{i,m} = 1 \quad \text{if } k = m \;;\text{o.w.} = 0$

$\Rightarrow$ Terms inside the summation $\left\{ \sum_k \sum_m q_{k,l} q_{m,n} \sum_i p_{i,k} p_{i,m} \right\}$ are non-zero only when $k = m$

$$\Rightarrow \left\|M\right\|_F = \sum_j \sum_l \sum_n r_{l,j} r_{n,j} \sum_m q_{m,l} q_{m,n} = \sum_m \sum_l \sum_n q_{m,l} q_{m,n} \sum_j r_{l,j} r_{n,j}$$

Since $\sum_j r_{l,j} r_{n,j} = 1 \quad \text{if } l = n \;;\text{o.w.} = 0,$

$\Rightarrow \left\|M\right\|_F = \sum_m \sum_n q_{m,n} q_{m,n} = \sum_m \sum_n \left(q_{m,n}\right)^2 = \sum_m q_{m,m}^2 = \left\|Q\right\|_F$ because $q_{m,n} = 0$ for $m \neq n$

SVD – Best Low Rank Approx.

- **A = U Σ V^T , B = U S V^T** ($\sigma_1 \geq \sigma_2 \geq \ldots \geq 0$, rank($A$)=$r$)
  - **S** = diagonal nxn matrix where $s_i = \sigma_i$ (i=1…k) else s_i=0

  then B is solution to $\min_B \|A-B\|_F$, rank(B)=k

- **Why?**

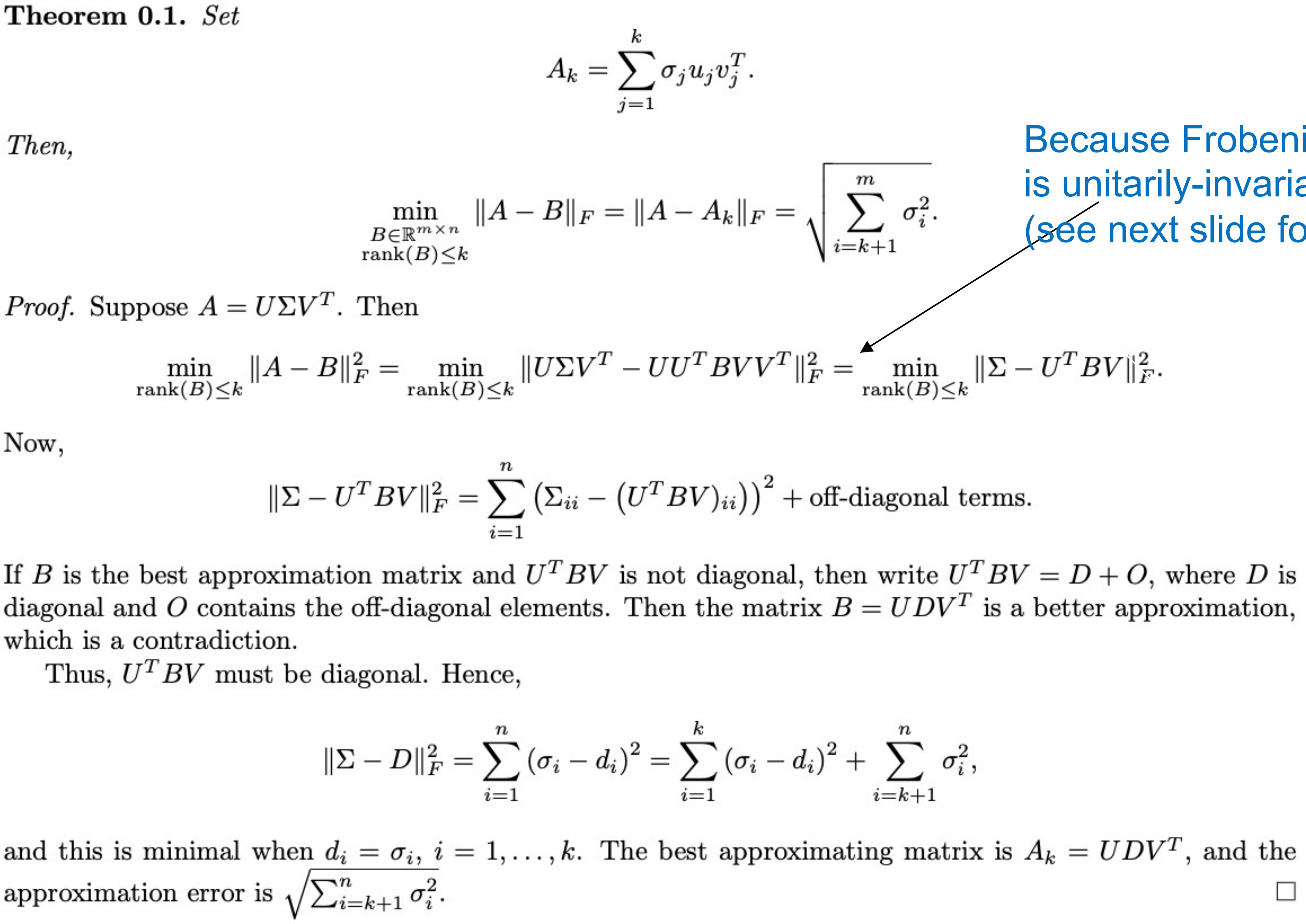
$$\min_{B, rank(B)=k} \|A-B\|_F = \min \|\Sigma - S\|_F = \min_{s_i} \sum_{i=1}^{r} (\sigma_i - s_i)^2$$

We used: U Σ V^T - U S V^T = U (Σ - S) V^T

- We want to choose s_i to minimize
- Solution is to set $s_i = \sigma_i$ (i=1…k) and other s_i=0

$$= \min_{s_i} \sum_{i=1}^{k} (\sigma_i - s_i)^2 + \sum_{i=k+1}^{r} \sigma_i^2 = \sum_{i=k+1}^{r} \sigma_i^2$$

SVD – Best Low Rank Approx.



Theorem 0.1. *Set*

$$A_k = \sum_{j=1}^{k} \sigma_j u_j v_j^T.$$

Then,

$$\min_{\substack{B\in\mathbb{R}^{m\times n}\\ \operatorname{rank}(B)\le k}} \|A-B\|_F = \|A-A_k\|_F = \sqrt{\sum_{i=k+1}^{m} \sigma_i^2}.$$

Because Frobenius norm is unitarily-invariant ; (see next slide for details)

Proof. Suppose $A = U\Sigma V^T$. Then

$$\min_{\operatorname{rank}(B)\le k} \|A-B\|_F^2 = \min_{\operatorname{rank}(B)\le k} \|U\Sigma V^T - UU^TBVV^T\|_F^2 = \min_{\operatorname{rank}(B)\le k} \|\Sigma - U^TBV\|_F^2.$$

Now,

$$\|\Sigma - U^TBV\|_F^2 = \sum_{i=1}^{n} \left(\Sigma_{ii} - (U^TBV)_{ii}\right)^2 + \text{off-diagonal terms.}$$

If B is the best approximation matrix and U^TBV is not diagonal, then write $U^TBV = D + O$, where D is diagonal and O contains the off-diagonal elements. Then the matrix $B = UDV^T$ is a better approximation, which is a contradiction.

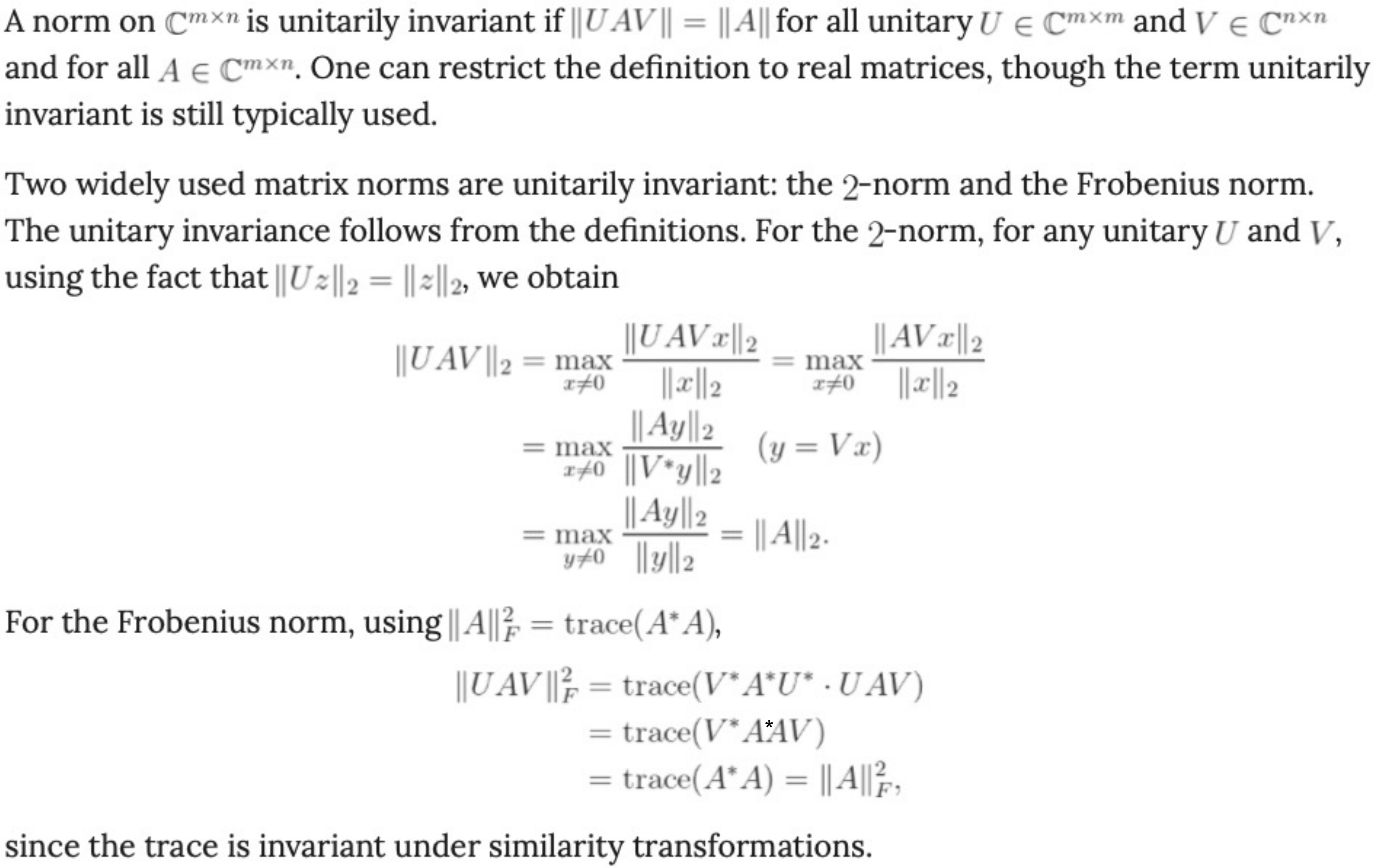
Thus, U^TBV must be diagonal. Hence,

$$\|\Sigma - D\|_F^2 = \sum_{i=1}^{n} (\sigma_i - d_i)^2 = \sum_{i=1}^{k} (\sigma_i - d_i)^2 + \sum_{i=k+1}^{n} \sigma_i^2,$$

and this is minimal when $d_i = \sigma_i$, $i = 1,\ldots,k$. The best approximating matrix is $A_k = UDV^T$, and the approximation error is $\sqrt{\sum_{i=k+1}^{n} \sigma_i^2}$. □

Source: https://www.cs.yale.edu/homes/el327/datamining2013aFiles/07_singular_value_decomposition.pdf

What is a Unitarily Invariant Norm ?



A norm on $\mathbb{C}^{m\times n}$ is unitarily invariant if $\|UAV\| = \|A\|$ for all unitary $U \in \mathbb{C}^{m\times m}$ and $V \in \mathbb{C}^{n\times n}$ and for all $A \in \mathbb{C}^{m\times n}$. One can restrict the definition to real matrices, though the term unitarily invariant is still typically used.

Two widely used matrix norms are unitarily invariant: the 2-norm and the Frobenius norm. The unitary invariance follows from the definitions. For the 2-norm, for any unitary U and V, using the fact that $\|Uz\|_2 = \|z\|_2$, we obtain

$$\begin{aligned} \|UAV\|_2 &= \max_{x\neq 0} \frac{\|UAVx\|_2}{\|x\|_2} = \max_{x\neq 0} \frac{\|AVx\|_2}{\|x\|_2} \\ &= \max_{x\neq 0} \frac{\|Ay\|_2}{\|V^*y\|_2} \quad (y = Vx) \\ &= \max_{y\neq 0} \frac{\|Ay\|_2}{\|y\|_2} = \|A\|_2. \end{aligned}$$

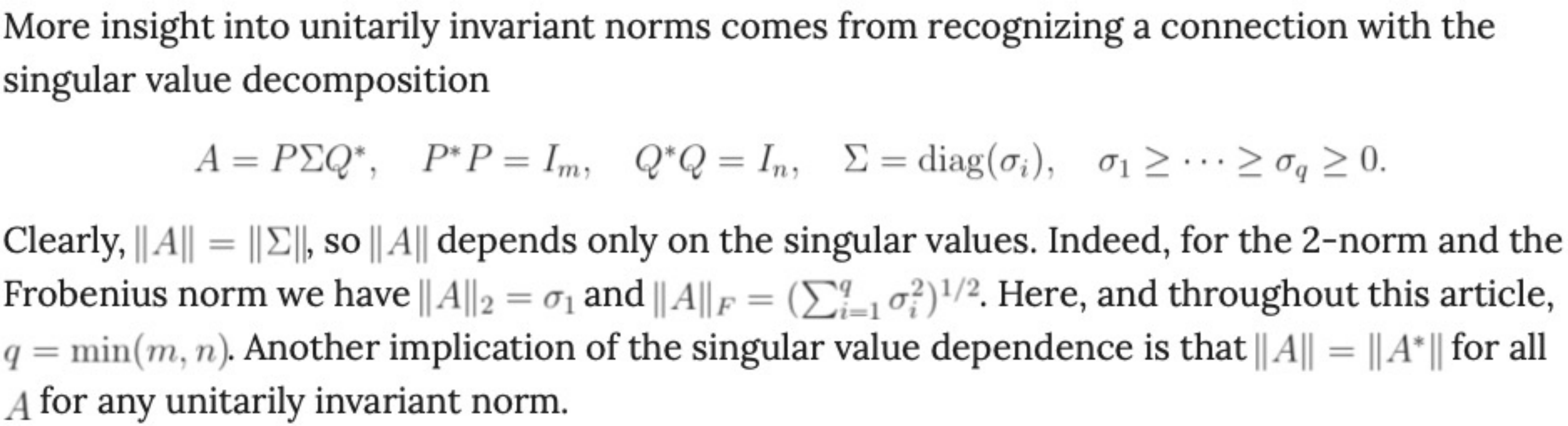
For the Frobenius norm, using $\|A\|_F^2 = \mathrm{trace}(A^*A)$,

$$\begin{aligned} \|UAV\|_F^2 &= \mathrm{trace}(V^*A^*U^* \cdot UAV) \\ &= \mathrm{trace}(V^*A^*AV) \\ &= \mathrm{trace}(A^*A) = \|A\|_F^2, \end{aligned}$$

since the trace is invariant under similarity transformations.

Source: https://nhigham.com/2021/02/02/what-is-a-unitarily-invariant-norm/

Unitarily Invariant Norm and connection to SVD



More insight into unitarily invariant norms comes from recognizing a connection with the singular value decomposition

$$A = P\Sigma Q^*, \quad P^*P = I_m, \quad Q^*Q = I_n, \quad \Sigma = \mathrm{diag}(\sigma_i), \quad \sigma_1 \geq \cdots \geq \sigma_q \geq 0.$$

Clearly, $\|A\| = \|\Sigma\|$, so $\|A\|$ depends only on the singular values. Indeed, for the 2-norm and the Frobenius norm we have $\|A\|_2 = \sigma_1$ and $\|A\|_F = (\sum_{i=1}^{q} \sigma_i^2)^{1/2}$. Here, and throughout this article, $q = \min(m, n)$. Another implication of the singular value dependence is that $\|A\| = \|A^*\|$ for all A for any unitarily invariant norm.

Source: https://nhigham.com/2021/02/02/what-is-a-unitarily-invariant-norm/

End of Backup Slides

SVD - Interpretation #2 (cont'd)

Equivalent:

'spectral decomposition' of the matrix:

$$\begin{bmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\ \mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\ \mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\ \mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\ 0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\ 0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\ 0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2} \end{bmatrix} = \begin{bmatrix} | & | \\ u_1 & u_2 \\ | & | \end{bmatrix} \text{ x } \begin{bmatrix} \sigma_1 & \emptyset \\ \emptyset & \sigma_2 \end{bmatrix} \text{ x } \begin{bmatrix} \text{---} & v_1 & \text{---} \\ \text{---} & v_2 & \text{---} \end{bmatrix}$$

SVD - Interpretation #2

Equivalent:

‘spectral decomposition’ of the matrix

$$m \updownarrow \overset{\longleftarrow \quad n \quad \longrightarrow}{\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix}} = \overbrace{\sigma_1 \; \underset{m \times 1}{u_1} \; \underset{1 \times n}{v^T_1} + \sigma_2 \; u_2 \; v^T_2 + \ldots}^{k \text{ terms}}$$

Assume: $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \ldots \geq 0$

Why is setting small σ_i to 0 the right thing to do?

Vectors $\mathbf{u}_i$ and $\mathbf{v}_i$ are unit length, so σ_i scales them.

So, zeroing small σ_i introduces less error.

SVD - Interpretation #2

Q: How many σs to keep?

A: Rule-of-a thumb:
keep 80-90% of ‘energy’ $(=\Sigma\sigma_i^2)$

$$
m \updownarrow \overbrace{\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix}}^{n} = \sigma_1 \quad u_1 \quad v^T_1 + \sigma_2 \quad u_2 \quad v^T_2 + \ldots
$$

Assume: $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \ldots$

SVD - Complexity

- **To compute SVD:**
  - **O(nm^2)** or **O(n^{2}m)** (whichever is less)
- **But:**
  - Less work, if we just want singular values
  - or if we want first *k* singular vectors
  - or if the matrix is sparse
- **Implemented in** linear algebra packages like
  - LINPACK, Matlab, SPlus, Mathematica ...

SVD - Conclusions so far

- **SVD: A= U Σ V^T: unique**
  - **U**: user-to-concept similarities
  - **V**: movie-to-concept similarities
  - **Σ** : strength of each concept
- **Dimensionality reduction:**
  - keep the few largest singular values (80-90% of ‘energy’)
  - SVD: picks up linear correlations

Relation to Eigen-decomposition

- **SVD gives us:**
  - $A = U \Sigma V^T$
- **Eigen-decomposition:**
  - $S = X \Lambda X^T$
    - S is symmetric
    - U, V, X are orthonormal ($\mathbf{U^T U = I}$),

      Λ, Σ are diagonal
- **What is:**
  - $\mathbf{AA^T = U\Sigma V^T (U\Sigma V^T)^T = U\Sigma V^T (V\Sigma^T U^T) = U\Sigma\Sigma^T U^T}$
  - $\mathbf{A^T A = V \Sigma^T U^T (U\Sigma V^T) = V \Sigma\Sigma^T V^T}$

Relation to Eigen-decomposition

- **SVD gives us:**
  - $A = U\,\Sigma\,V^T$
- **Eigen-decomposition:**
  - $S = X\,\Lambda\,X^T$
    - *S* is symmetric
    - *U, V, X* are orthonormal ($\mathbf{U}^T\mathbf{U}=\mathbf{I}$),

    Λ, Σ are diagonal

Shows how to compute SVD using eigenvalue decomposition!

$\Downarrow$

- **What is:**
  - $\mathbf{AA^T = U\Sigma V^T(U\Sigma V^T)^T = U\Sigma V^T(V\Sigma^T U^T) =} \overset{X}{\overset{\downarrow}{\mathbf{U}}}\,\overset{\Lambda}{\overset{\downarrow}{\mathbf{\Sigma\Sigma^T}}}\,\overset{X^T}{\overset{\downarrow}{\mathbf{U^T}}}$
  - $\mathbf{A^TA = V\,\Sigma^T\,U^T\,(U\Sigma V^T) =} \underset{\underset{X}{}}{\underset{\uparrow}{\mathbf{V}}}\,\underset{\underset{\Lambda}{}}{\underset{\uparrow}{\mathbf{\Sigma\Sigma^T}}}\,\underset{\underset{X^T}{}}{\underset{\uparrow}{\mathbf{V^T}}}$

So, $\lambda_i = \sigma_i^2$

SVD: Properties

- $\mathbf{A}\,\mathbf{A}^T = \mathbf{U}\,\Sigma^2\,\mathbf{U}^T$
- $\mathbf{A}^T\mathbf{A} = \mathbf{V}\,\Sigma^2\,\mathbf{V}^T$

- $(\mathbf{A}^T\mathbf{A})^k = \mathbf{V}\,\Sigma^{2k}\,\mathbf{V}^T$
  - E.g.: $(\mathbf{A}^T\mathbf{A})^2 = \mathbf{V}\,\Sigma^2\,\mathbf{V}^T\mathbf{V}\,\Sigma^2\,\mathbf{V}^T = \mathbf{V}\,\Sigma^4\,\mathbf{V}^T$

- $(\mathbf{A}^T\mathbf{A})^k \sim v_1\,\sigma_1^{2k}\,v_1^T$ for $k>>1$

Case study: How to query?

- **Q: Find users that like 'Matrix'**
- **A: Map query into a 'concept space' – how?**

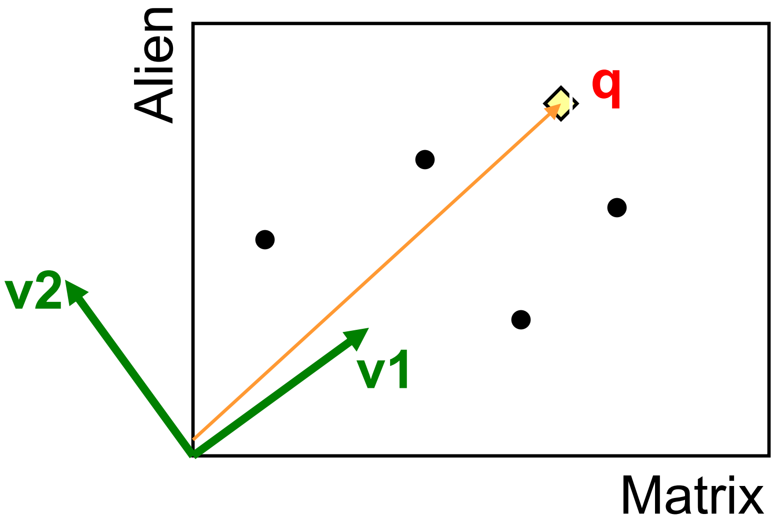
$$
\begin{array}{c|ccccc}
 & \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\
\hline
\text{SciFi Fans} & \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\
 & \mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\
 & \mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\
 & \mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\
\text{Romance Fans} & 0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\
 & 0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\
 & 0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2}
\end{array}
=
\begin{bmatrix}
\mathbf{0.13} & 0.02 & -0.01 \\
\mathbf{0.41} & 0.07 & -0.03 \\
\mathbf{0.55} & 0.09 & -0.04 \\
\mathbf{0.68} & 0.11 & -0.05 \\
0.15 & \mathbf{-0.59} & \mathbf{0.65} \\
0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\
0.07 & \mathbf{-0.29} & \mathbf{0.32}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{12.4} & 0 & 0 \\
0 & \mathbf{9.5} & 0 \\
0 & 0 & \mathbf{1.3}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\
0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\
0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09
\end{bmatrix}
$$

Case study: How to query?

- **Q: Find users that like ‘Matrix’**
- **A: Map query into a ‘concept space’ – how?**

$$\begin{array}{c|ccccc} & \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\ \hline \mathbf{q} & 5 & 0 & 0 & 0 & 0 \end{array}$$

Project into concept space:
Inner product with each ‘concept’ vector $\mathbf{v}_i$

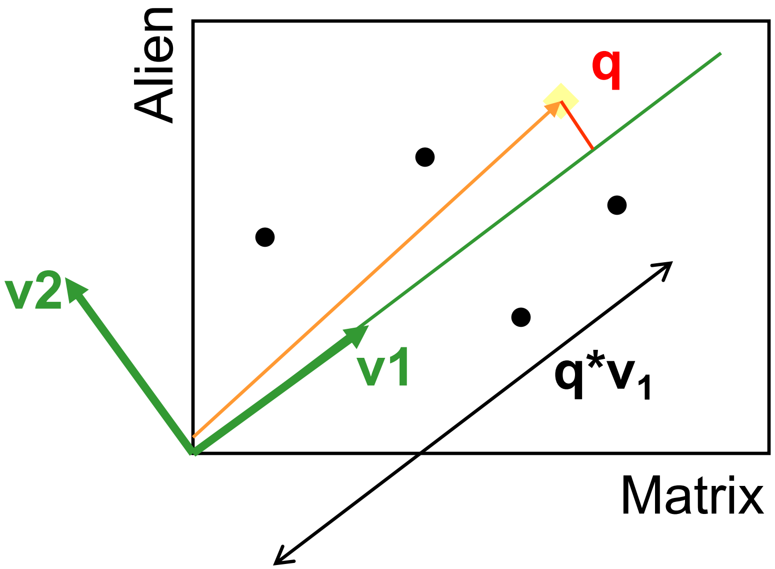


Case study: How to query?

- **Q: Find users that like 'Matrix'**
- **A: Map query into a 'concept space' – how?**

$$\mathbf{q} \quad \begin{array}{ccccc} \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\ \hline 5 & 0 & 0 & 0 & 0 \end{array}$$

Project into concept space:
Inner product with each 'concept' vector $\mathbf{v}_i$



Case study: How to query?

Compactly, we have:

$$q_{concept} = q\, V$$

E.g.:

$$q \quad \begin{array}{ccccc} \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\ \left[\, 5 \right. & 0 & 0 & 0 & \left. 0 \,\right] \end{array} \times \underbrace{\begin{bmatrix} 0.56 & 0.12 \\ 0.59 & -0.02 \\ 0.56 & 0.12 \\ 0.09 & -0.69 \\ 0.09 & -0.69 \end{bmatrix}}_{\text{movie-to-concept similarities (V)}} = \begin{bmatrix} \overset{\substack{\text{SciFi-concept} \\ \downarrow}}{2.8} & 0.6 \end{bmatrix}$$

Case study: How to query?

- **How would the user *d* that rated (‘Alien’, ‘Serenity’) be handled?** $\mathbf{d_{concept} = d\ V}$

E.g.:

$$\mathbf{q} \quad \begin{array}{ccccc} \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\ \left[\;0\right. & 4 & 5 & 0 & \left.0\;\right] \end{array} \;\mathbf{x}\; \underbrace{\begin{bmatrix} 0.56 & 0.12 \\ 0.59 & -0.02 \\ 0.56 & 0.12 \\ 0.09 & -0.69 \\ 0.09 & -0.69 \end{bmatrix}}_{\textbf{movie-to-concept similarities (V)}} \;\mathbf{=}\; \begin{bmatrix} \overset{\text{SciFi-concept}}{\downarrow} \\ 5.2 \end{bmatrix.} \!\!\begin{matrix} \\ 0.4 \end{matrix}\Big]$$

Case study: How to query?

- **Observation:** User ***d*** that rated (‘*Alien*’, ‘*Serenity*’) will be **similar** to user **q** that rated (‘*Matrix*’), although ***d*** and **q** have **zero ratings in common**!

$$\begin{array}{rccccc} & \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\ \mathbf{d}= & 0 & 4 & 5 & 0 & 0 \\ \mathbf{q}= & 5 & 0 & 0 & 0 & 0 \end{array}$$

$$\mathbf{d} \dashrightarrow \begin{bmatrix} \overset{\text{SciFi-concept}}{\overset{\downarrow}{2.8}} & 0.6 \end{bmatrix}$$

$$\mathbf{q} \dashrightarrow \begin{bmatrix} 5.2 & 0.4 \end{bmatrix}$$

Zero ratings in common

Similarity ≠ 0

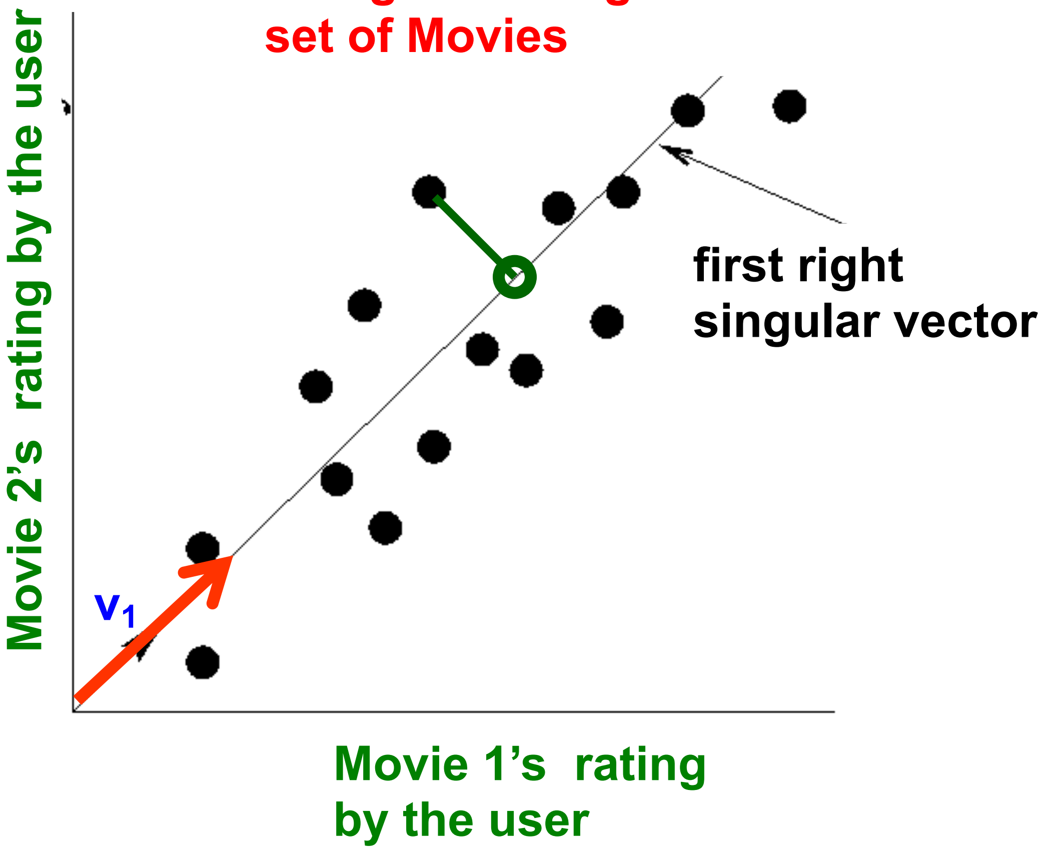
Principal Component Analysis (PCA)

An Application of SVD

Recall: The 2^{nd} Interpretation of SVD

- **SVD gives 'best' axis to project on:**
  - '**best**' = min sum of squares of projection errors
- **In other words, minimum reconstruction error**

Each user (a data point) is characterized by the ratings he/she gave to a set of Movies



Philosophy of PCA

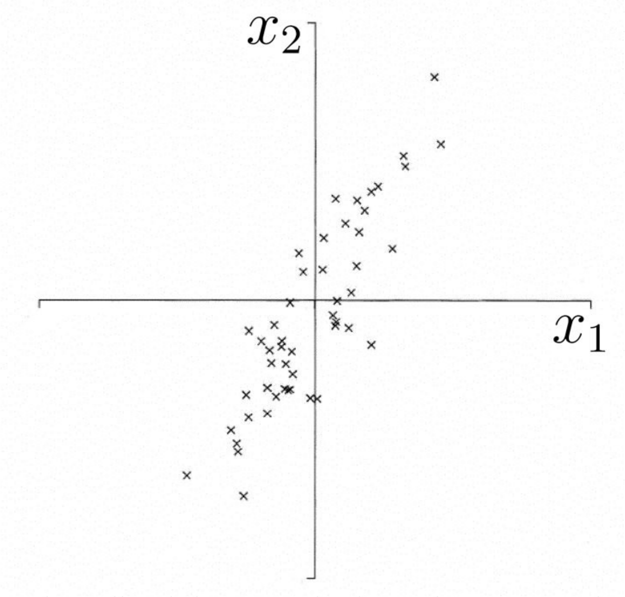
- PCA is concerned with explaining the variance/ covariance structure of a set of variables (features) through a few linear combinations.

- We typically have a *m x n* input data matrix, *A*:
  - Each row of *A* corresponds to one n-dim data-point
  - i.e. *m* observed data-points, each data-point consists of *n* potentially correlated variables (features) $x_1, x_2, ..x_n$

- PCA looks for a transformation of the *n* x_i's into *d* new variables (features) z_i's that are uncorrelated.

- Objective: To replace the old variables (features): $x_1, x_2, \ldots x_n$ with a few new features: z_i's without losing much information.

Geometric picture of principal components (PCs)

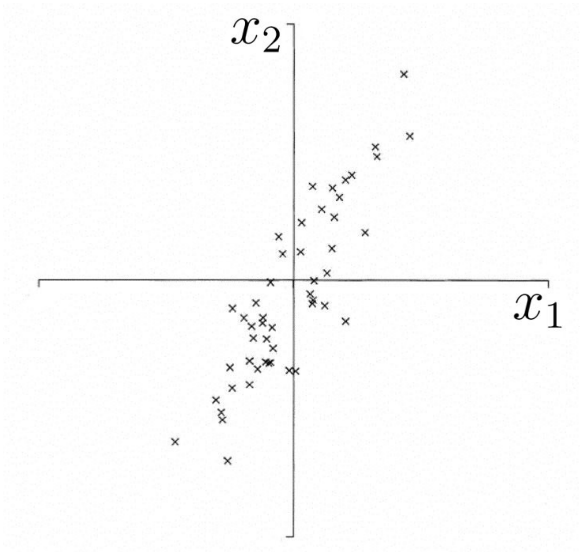


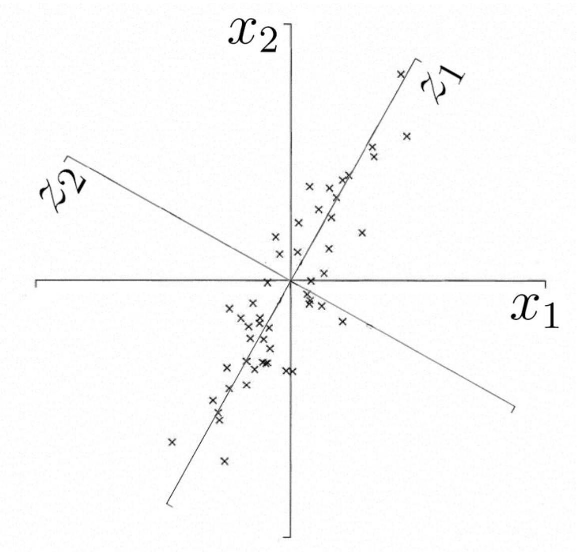
A sample of *m* observations in the old 2-D space $\mathbf{x} = (x_1, x_2)$

Goal: To account for the variation in a sample in as few variables as possible, to some accuracy

Adapted from http://www.astro.princeton.edu/~gk/A542/PCA.ppt

Geometric picture of principal components (PCs)





- The 1st PC z_1 is derived from a minimum distance fit to a line in space ; direction of this line is that of the 1st Principal Vector , say v_1

- The 2nd PC z_2 is derived from a minimum distance fit to another line in the plane perpendicular (orthogonal) to the 1st Principal vector

Adapted from http://www.astro.princeton.edu/~gk/A542/PCA.ppt

PCA: *General methodology*

From n original variables (features): $x_1, x_2, \ldots, x_n$:

Produce d new variables (features): $z_1, z_2, \ldots, z_d$:

$$\left.\begin{array}{l} z_1 = v_{11}x_1 + v_{12}x_2 + \ldots + v_{1n}x_n \\ z_2 = v_{21}x_1 + v_{22}x_2 + \ldots + v_{2n}x_n \\ \ldots \\ z_d = v_{d1}x_1 + v_{d2}x_2 + \ldots + v_{dn}x_n \end{array}\right\}$$

z_i's are the **Principal Components**

N.B: Each of these new variables is a LINEAR combination of the old variables $\boldsymbol{x}_\mathbf{i}$'s

such that:

z_i's are uncorrelated (orthogonal) to each other

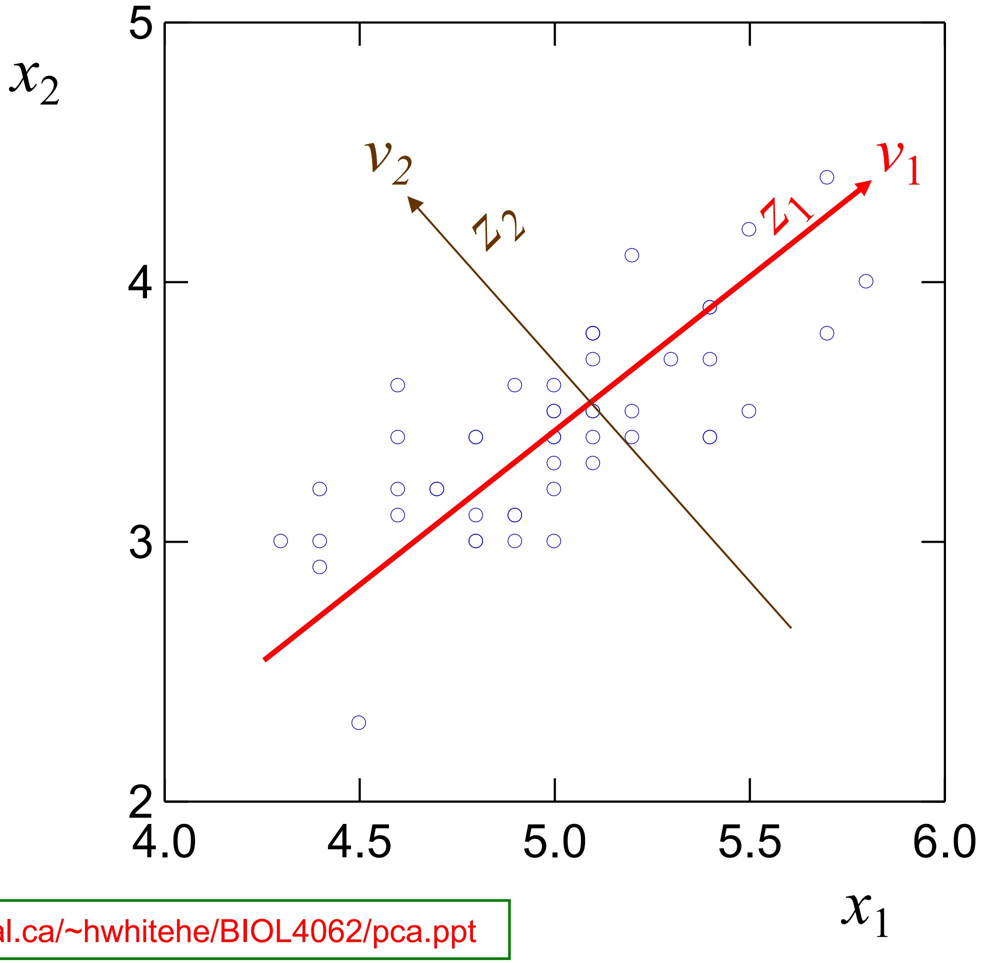
z_1 explains as much as possible of original variance in data set

z_2 explains as much as possible of remaining variance

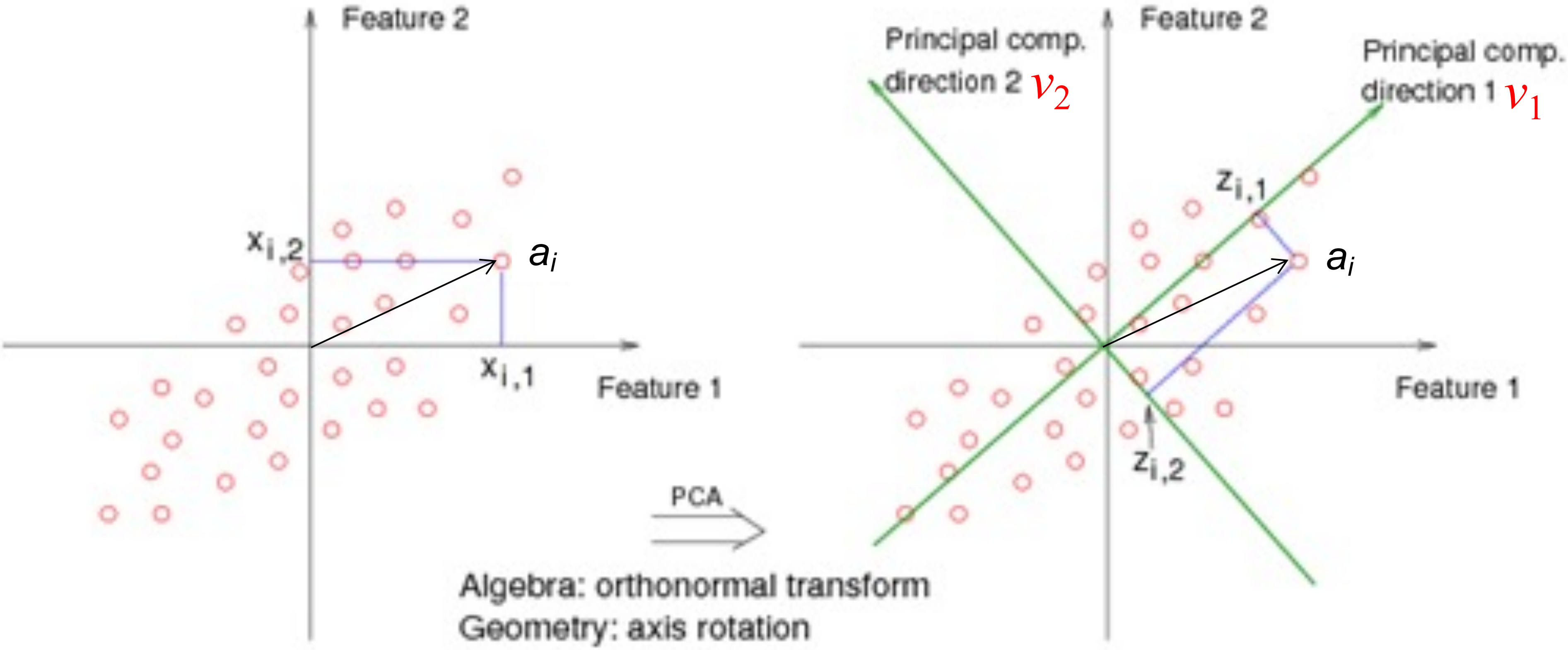
etc.

Adapted from http://myweb.dal.ca/~hwhitehe/BIOL4062/pca.ppt

Principal Components Analysis



Adapted from http://myweb.dal.ca/~hwhitehe/BIOL4062/pca.ppt



Note that:

$z_{i,1} = a_i \cdot \boldsymbol{v_1}$

$z_{i,2} = a_i \cdot \boldsymbol{v_2}$

Adapted from http://myweb.dal.ca/~hwhitehe/BIOL4062/pca.ppt

Terminologies for PCA

- The column vector $\boldsymbol{v_1} = \{v_{11}, v_{12}, \dots, v_{1n}\}'$, sometimes referred as the 1st Principal Vector, defines the direction of the axis for the 1st new variable, z_1 , which is the actual 1st Principal Component (PC)
  - The v_{1j}'s are called the **coefficients** (or **loadings**) of 1st PC
  - It can be shown that the entire vector: $\{v_{11}, v_{12}, \dots, v_{1n}\}$ is the 1st **Eigenvector**, i.e., the one corresponds to the largest eigenvalue of the correlation/covariance matrix (which captures the **correlation between different old features**) of the original input data set

Similarly,

- The column vector $\boldsymbol{v_d} = \{v_{d1}, v_{d2}, \dots, v_{dn}\}'$ defines the direction of the axis for the *d*-th new (derived) variable, z_d , i.e. the *d*-th PC
- The v_{dj}'s are called the **coefficients** (or **loadings**) of *d*-th PC
- $\{v_{d1}, v_{d2}, \dots, v_{dn}\}$ is the *d*-th **Eigenvector**, i.e. the one corresponds to the d-th largest eigenvalue of the correlation/covariance matrix of the input data set…

How to determine v_1 ?

Objective: To find the direction of a new axis which minimizes the sum of squared of errors when the original data points are projected onto this new axis.

Since Minimize Sum of Squared Project Error $\equiv$ Maximize Sum of Squared Projection length of original data points

Thus, it is equivalent to find a new axis which maximizes the sum of squared of projection-length when the original data points are projected onto this new axis.

Let v be the unit column vector which defines the direction of a new axis.

Let a_i be the original data-point represented by the i-th row of the original input matrix A

The length of the projection of the column-vector representing a_i onto the new axis is given by $a_i^T \cdot v$

Sum of Squared Projection length onto the new axis for all data points = $\sum_{i=1}^{m} \left| a_i^T \cdot v \right|^2 = \left| Av \right|^2 = v^T A^T Av$

Thus, the unit-vector defining the direction of the new axis, $v_1 = \arg\max_{|v|=1} \sum_{i=1}^{m} \left| a_i^T \cdot v \right|^2 = \arg\max_{|v|=1} \left| Av \right|^2 = \arg\max_{|v|=1} v^T A^T Av$

In other words, we want to find v_1 which maximizes $v^T A^T Av$ subject to the constraint of $v^T v = 1$

Take the Lagrangian approach, we differentiate

$[v^T A^T Av - \lambda(v^T v - 1)]$ w.r.t. λ and v respectively and set the results to zero to get:

$v_1^T v_1 = 1 \quad and$

$2A^T Av_1 - 2\lambda v_1 = 0 \Rightarrow A^T Av_1 = \lambda v_1 \Rightarrow v_1$ is one of the eigenvectors of the (square) matrix $A^T A$.

Since the objective is to maximize $v^T A^T Av = v^T \lambda v = \lambda$,

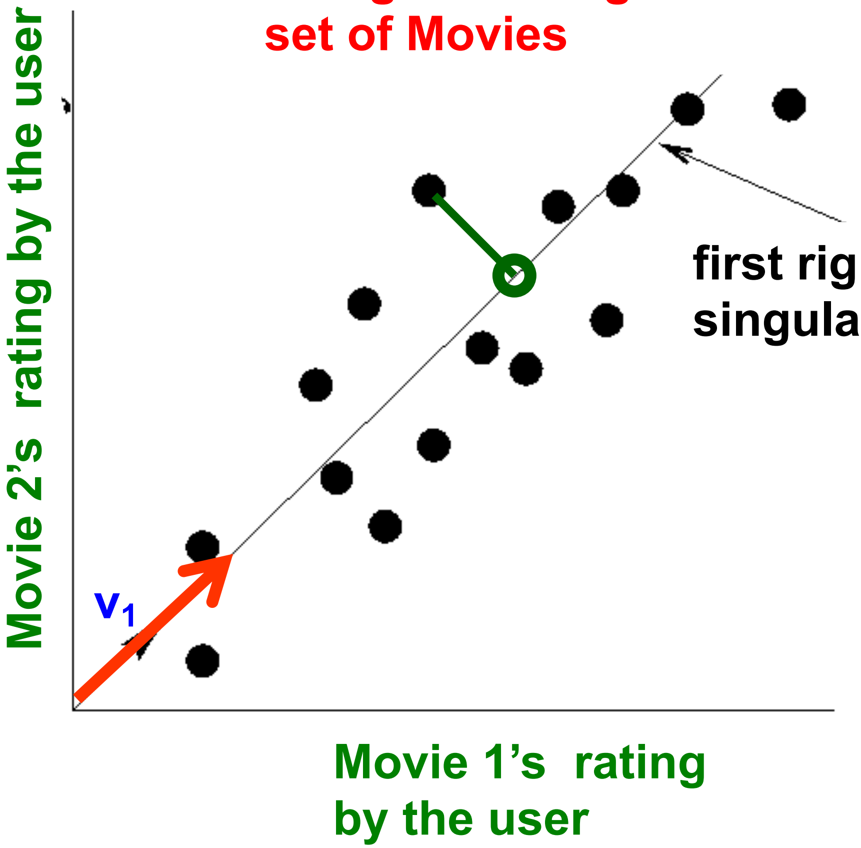
$\Rightarrow \lambda$ and v_1 should be the largest eigenvalue and the corresponding eigenvector of $A^T A$ respectively.

Since each row of A corresponds to a data-point, $A^T A$ is actually the covariance matrix of the data-set as long as the data has already been "centered", i.e. each attribute $x_i \leftarrow \left(x_i - \overline{x}_i\right)$.

Determine v1
by Minimizing Total “Reconstruction Error”

- **SVD gives ‘best’ axis to project on:**
  - **‘Best’ = min sum of squares of projection errors**
  - **In other words, minimizing total reconstruction error**

Each user (a data point) is characterized by the ratings he/she gave to a set of Movies



first right singular vector

Determine v1
by Minimizing Total “Reconstruction Error”

- **Find the ‘Best’ axis (v1) to project on:**
  - ‘**Best**’ = minimize sum of squares of projection errors
    = minimize sum of squares of “distance” for ALL xi’s
    = maximize sum of squares of “projection” for ALL xi’s

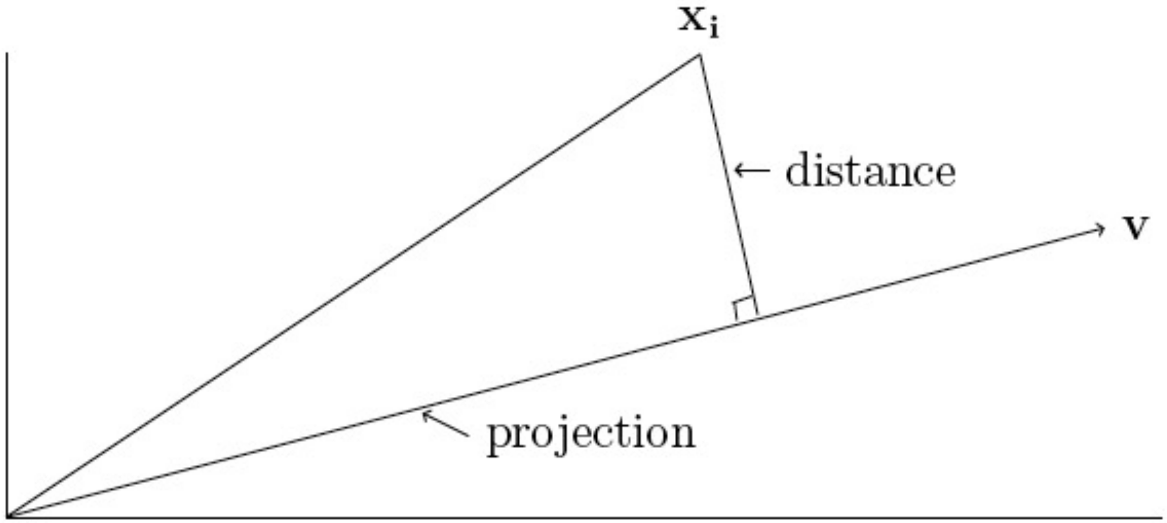


Figure 4.1: The projection of the point $\mathbf{x_i}$ onto the line through the origin in the direction of $\mathbf{v}$

How to determine v_1 ? (cont’d)

Objective: To find the direction of a new axis which minimizes the sum of squared of errors when the original data points are projected onto this new axis.

Since Minimize Sum of Squared Project Error $\equiv$ Maximize Sum of Squared Projection length of original data points

Thus, it is equivalent to find a new axis which maximizes the sum of squared of projection-length when the original data points are projected onto this new axis.

Let v be the unit column vector which defines the direction of a new axis.

Let a_i be the original data-point represented by the i-th row of the original input matrix A

The length of the projection of the column-vector representing a_i onto the new axis is given by $a_i^T \cdot v$

Sum of Squared Projection length onto the new axis for all data points = $\sum_{i=1}^{m} \left| a_i^T \cdot v \right|^2 = \left| Av \right|^2 = v^T A^T Av$

Thus, the unit-vector defining the direction of the new axis, $v_1 = \arg\max_{|v|=1} \sum_{i=1}^{m} \left| a_i^T \cdot v \right|^2 = \arg\max_{|v|=1} \left| Av \right|^2 = \arg\max_{|v|=1} v^T A^T Av$

In other words, we want to find v_1 which maximizes $v^T A^T Av$ subject to the constraint of $v^T v = 1$

Take the Lagrangian approach, we differentiate

$[v^T A^T Av - \lambda(v^T v - 1)]$ w.r.t. λ and v respectively and set the results to zero to get:

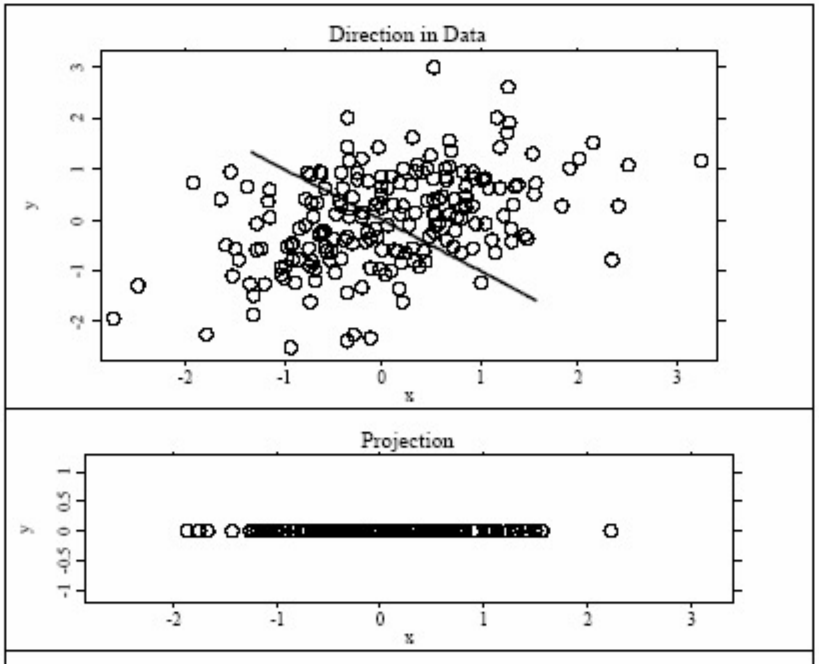
$v_1^T v_1 = 1 \quad and$

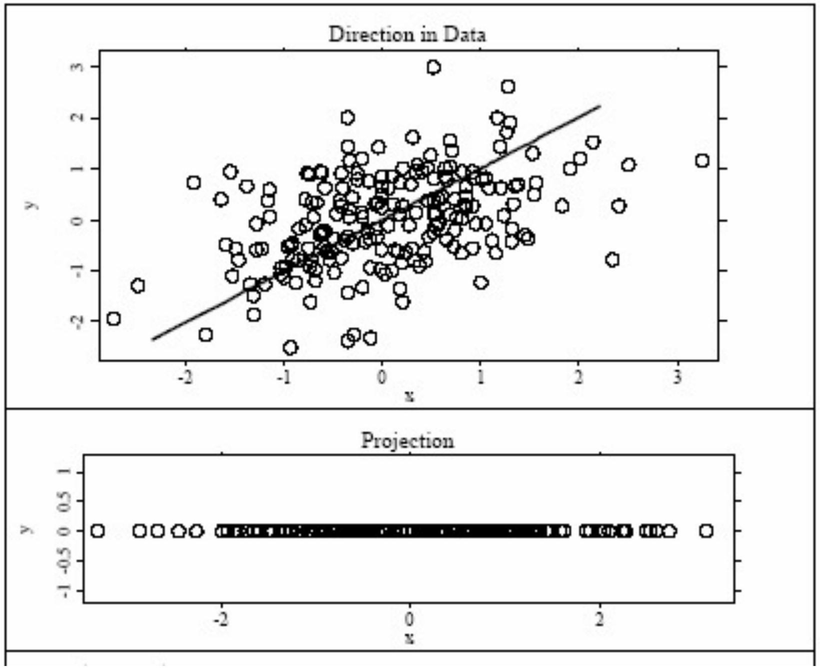
$2A^T Av_1 - 2\lambda v_1 = 0 \Rightarrow A^T Av_1 = \lambda v_1 \Rightarrow v_1$ is one of the eigenvectors of the (square) matrix $A^T A$.

Since the objective is to maximize $v^T A^T Av = v^T \lambda v = \lambda$,

$\Rightarrow \lambda$ and v_1 should be the largest eigenvalue and the corresponding eigenvector of $A^T A$ respectively.

Since each row of A corresponds to a data-point, $A^T A$ is actually the covariance matrix of the data-set as long as the data has already been "centered", i.e. each attribute $x_i \leftarrow \left(x_i - \bar{x}_i \right)$.


Direction in Data
y
3
2
1
0
-1
-2
-2
-1
0
1
2
3
x
Projection
y
1
0.5
0
-0.5
-1
-2
-1
0
1
2
3
x


Direction in Data
y
3
2
1
0
-1
-2
-2
-1
0
1
2
3
x
Projection
y
1
0.5
0
-0.5
-1
-2
0
2
x

How to determine the directions of the 2nd , 3rd , …k-th new axes ?

After projecting the original data-points into the 1st new axis defined by v_1 ,

we want to find another (the 2nd) new axis to account for the "residual components" of each data point.

For the i - th data point represented by its corresponding column-vector a_i^T ,

its residue after projecting to the 1st new axis is given by: $a_i^T - (a_i^T \cdot v_1)v_1$

Objective: To find v_2 which defines the direction of the 2nd new axis which can

Minimize the total projection errors for the "residual components" of each data point

(i.e. after subtracting their projection to the direction of v_1)

$\equiv$ Maximizing Sum of Squared Projection length of the "residual components" of the original data points

In other words, we want to find:

$$v_2 = \arg\max_{v} \sum_{i=1}^{m} \left[v^T \left(a_i^T - (a_i^T \cdot v_1)v_1 \right) \right]^2 \text{with } v_2^T v_2 = 1.$$

or equivalently, we want to find:

$$v_2 = \arg \max_{v \perp v1, |v|=1} v^T A^T Av$$

Here, v_2 is the unit column vector which defines the direction of the 2nd new axis.

Similar to the derivation of v_1 , it can be shown that

v_2 should be the 2nd eigenvector, i.e. the one corresponds to the 2nd largest eigenvalue of $A^T A$ respectively.

In general, v_k ,which defines the direction of the k - th new axis,

is given by the kth-eigenvector of $A^T A$, i.e. the one corresponds to the k - th largest eigenvalue of $A^T A$.

See Chapter 1 of Principal Component Analysis by I.T.Jolliffe [PCA] and the references therein for the detail proof.

In conclusion, we have found that:

- The direction of the 1st PC, z1 is given by the eigenvector $\mathbf{v}_1$ which corresponds to the largest eigenvalue of the covariance matrix $A^T A$.
- The second vector that is orthogonal (uncorrelated) to the first is the one that has the second highest variance which comes to be the eigenvector corresponding to the second largest eigenvalue of $A^T A$.
- And so on …

Relation to between
SVD and PCA (Eigen-decomposition)

- **SVD gives us:**
  - $A = U\,\Sigma\,V^T$
    - *For any matrix A*
- **Eigen-decomposition:**
  - $S = X\,\Lambda\,X^T$
    - *For any symmetric matrix S*
- U, V, X are orthonormal ($U^TU=I$, etc),
- Λ, Σ are diagonal
- **What is:**
  - $\mathbf{AA^T = U\Sigma V^T(U\Sigma V^T)^T = U\Sigma V^T(V\Sigma^T U^T) =} \overset{X}{\overset{\downarrow}{\mathbf{U}}}\,\overset{\Lambda}{\overset{\downarrow}{\mathbf{\Sigma\Sigma^T}}}\,\overset{X^T}{\overset{\downarrow}{\mathbf{U^T}}}$
  - $\mathbf{A^TA = V\,\Sigma^T\,U^T\,(U\Sigma V^T) =} \underset{\uparrow}{\mathbf{V}}\,\underset{\uparrow}{\mathbf{\Sigma\Sigma^T}}\,\underset{\uparrow}{\mathbf{V^T}}$

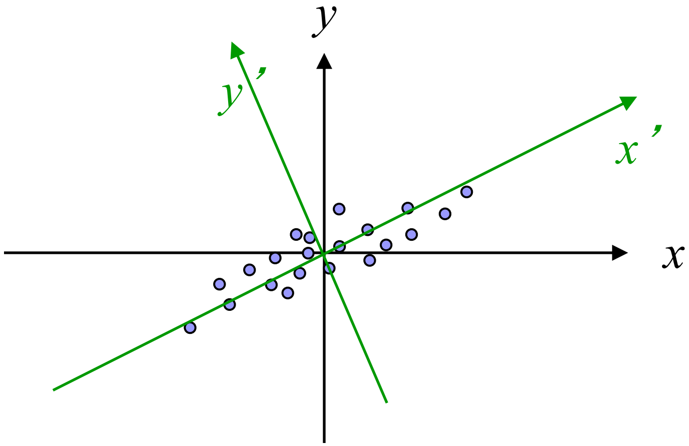
Show how to perform PCA (or eigenvalue decomposition) using SVD in practice !

$\mathbf{A^TA} = S = X\,\Lambda\,X^T$

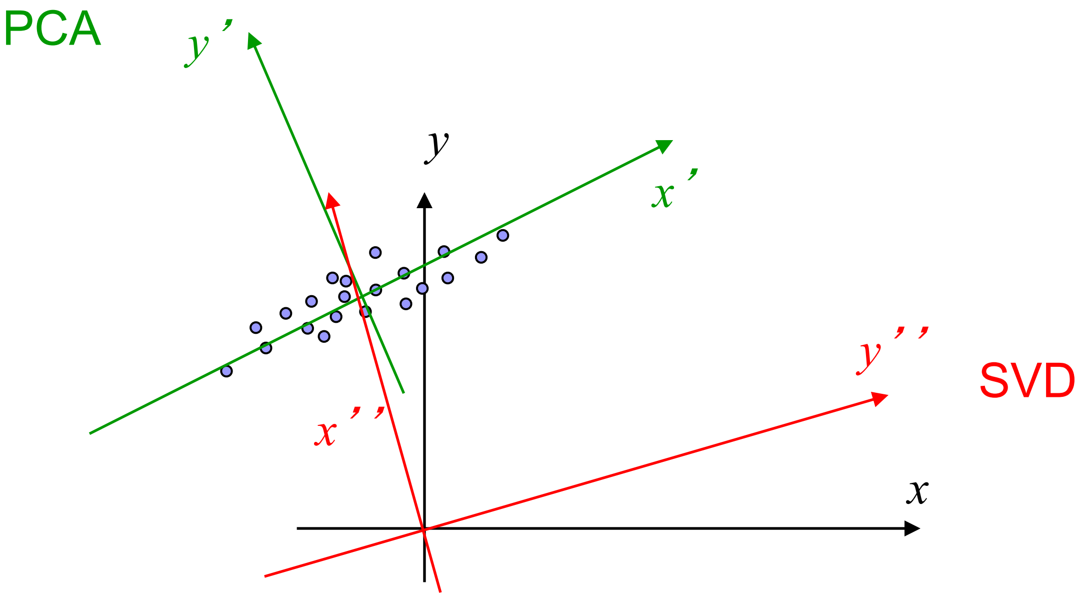
Also: $\lambda_i = \sigma_i^2$

When is SVD = PCA?

- Centered data

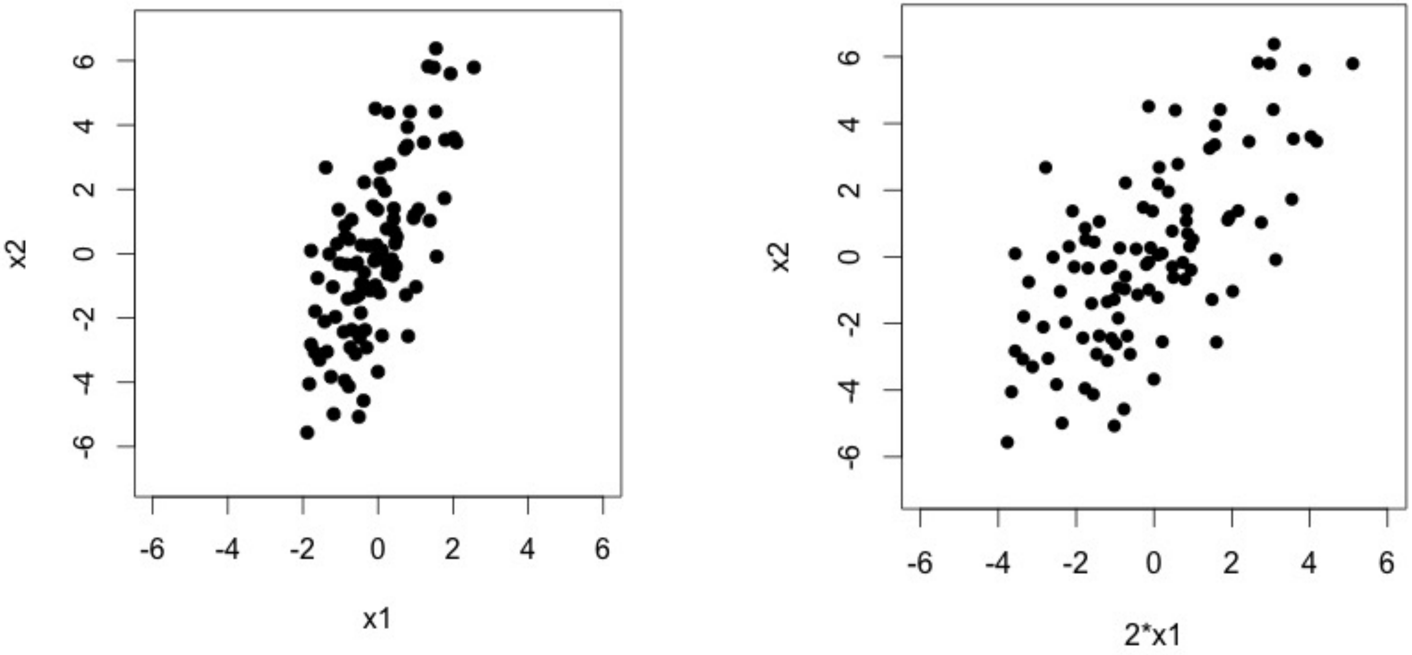


When is SVD different from PCA?

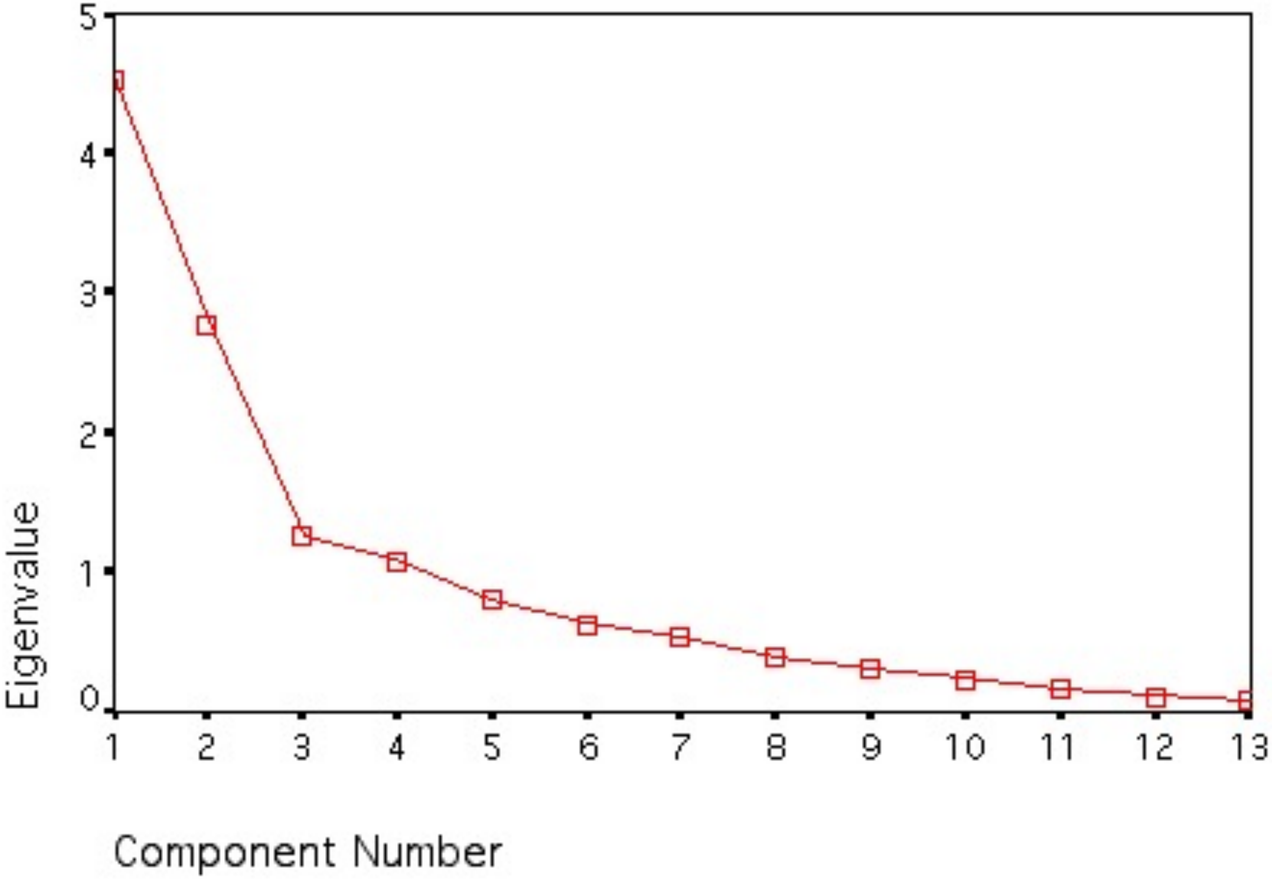


Additional notes for PCA

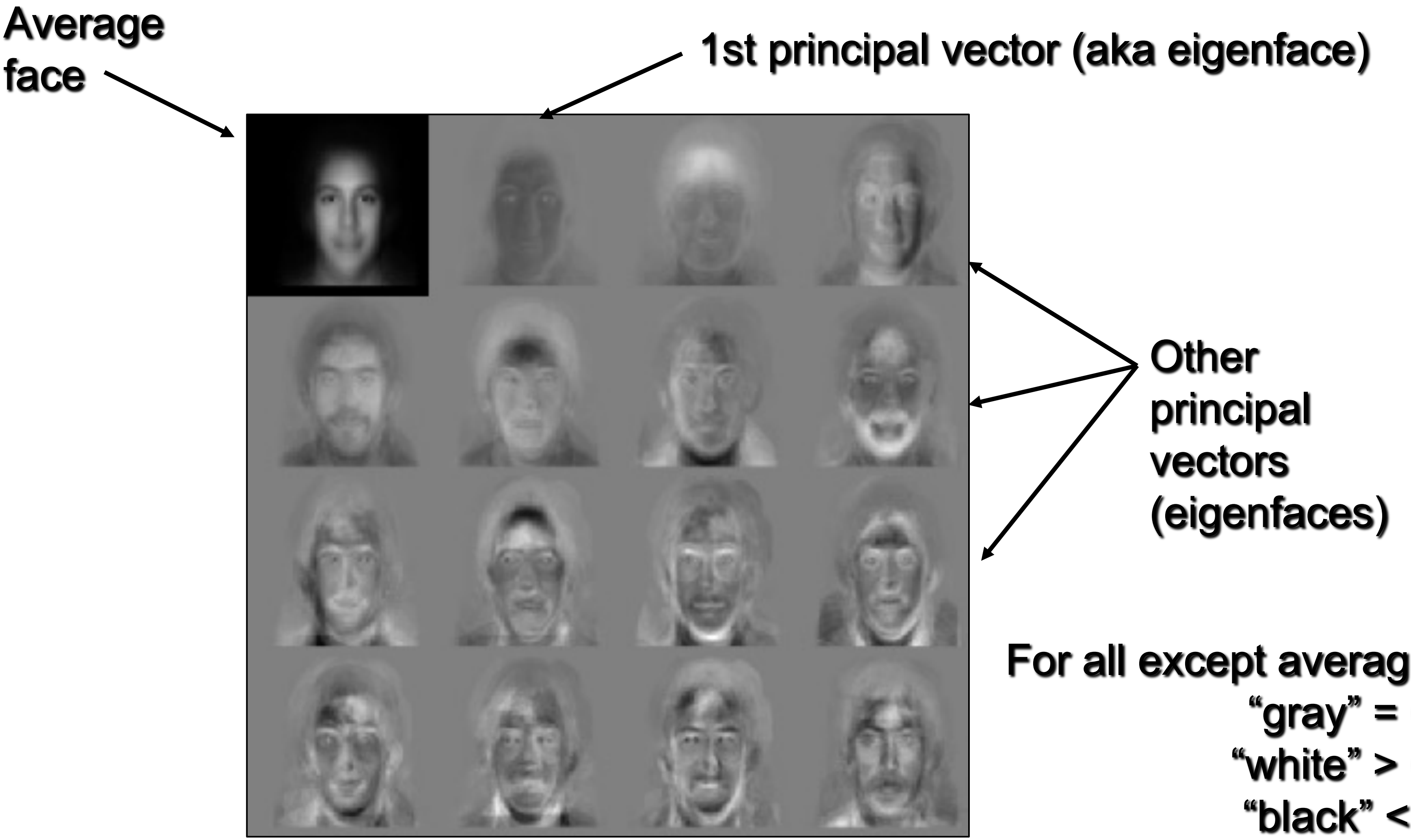
- PCA is sensitive to scale
- PCA should be applied on data that have approximately the same scale in each variable
- Also remember to 'center' each of the attributes, i.e. substracted by the sample mean, to get the covariance matrix before doing eigenvalue decomposition (or SVD).

x2
6
4
2
0
-2
-4
-6
-6
-4
-2
0
2
4
6
x1
x2
6
4
2
0
-2
-4
-6
-6
-4
-2
0
2
4
6
2*x1

How many PCAs to keep



Example: PCA on Faces: “Eigenfaces”


Average face
1st principal vector (aka eigenface)
Other principal vectors (eigenfaces)

For all except average,
“gray” = 0,
“white” > 0,
“black” < 0

http://En.wikipedia.org/wiki/Eigenface

Computational Trick for PCA with Eigenfaces

Each 100x100-pixel sample face is a 10,000 dimension data point, represented as a 1x10,000 row vector.

Stack 300 sample faces together to form a 300x10,000 input data matrix A

$\Rightarrow$ Size of covariance matrix $A^T A \; = 10\text{K}\text{x}10\text{K}$; too big for eigen-decomposition

Instead, do eigen-decomposition on the 300x300 AA^T to get:

λ_i and u_i s.t. $\quad AA^T u_i = \lambda_\text{i} u_i$

Pre-multiply both sides by A^T:

$A^T AA^T u_i = A^T \lambda_\text{i} u_i = \lambda_\text{i} A^T u_i$

$\Rightarrow A^T A\left(A^T u_i\right) = \lambda_\text{i}\left(A^T u_i\right)$

$\Rightarrow \; v_i \; = A^T u_i$ is the eigenvector of the 10,000x10,000 $A^T A$

$\Rightarrow$ We have solved eigen-decomposition for the big $A^T A$

by solving that for the 300x300 AA^T !

$**\, v_i$ is a 10000 x 1 vector, having the same dimension of an input data point (a face)

$\Rightarrow \; v_i$ is (and can be displayed as) the i - th eigenface !

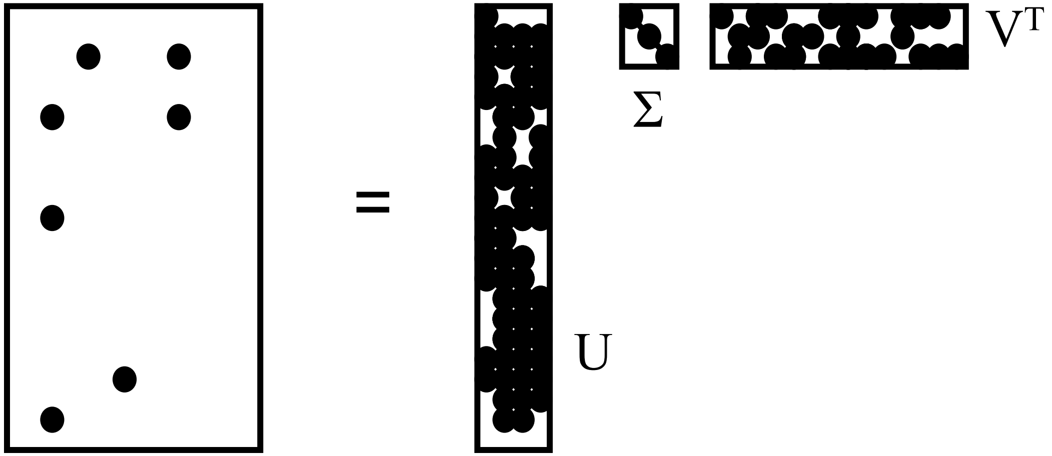
CUR Decomposition

SVD: Strength and Weakness

+ **Optimal low-rank approximation** in terms of Frobenius norm

- **Interpretability problem:**
  - A singular vector specifies a linear combination of all input columns or rows

- **Lack of sparsity:**
  - Singular vectors are **dense!**



Frobenius norm:

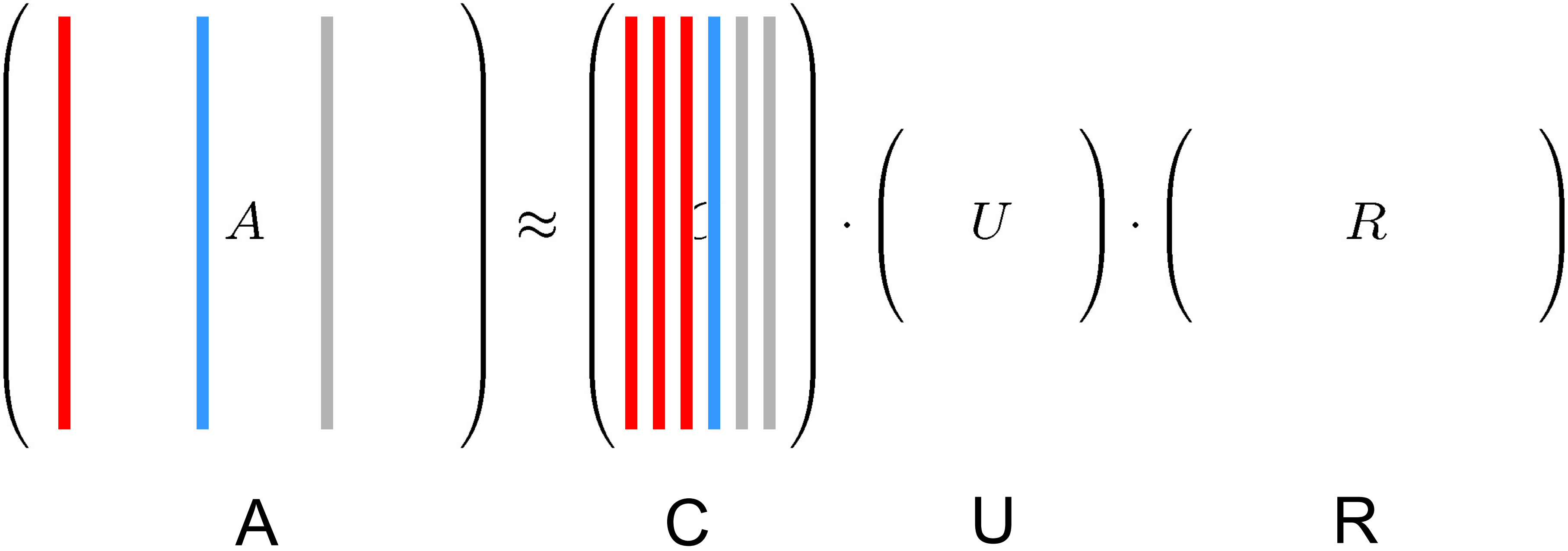
$\|X\|_F = \Sigma_{ij} X_{ij}^2$

CUR Decomposition

- **Goal: Express A as a product of matrices C,U,R**

  Make $\|A\text{-}C\cdot U\cdot R\|_F$ small

- **"Constraints" on C and R:**



CUR Decomposition

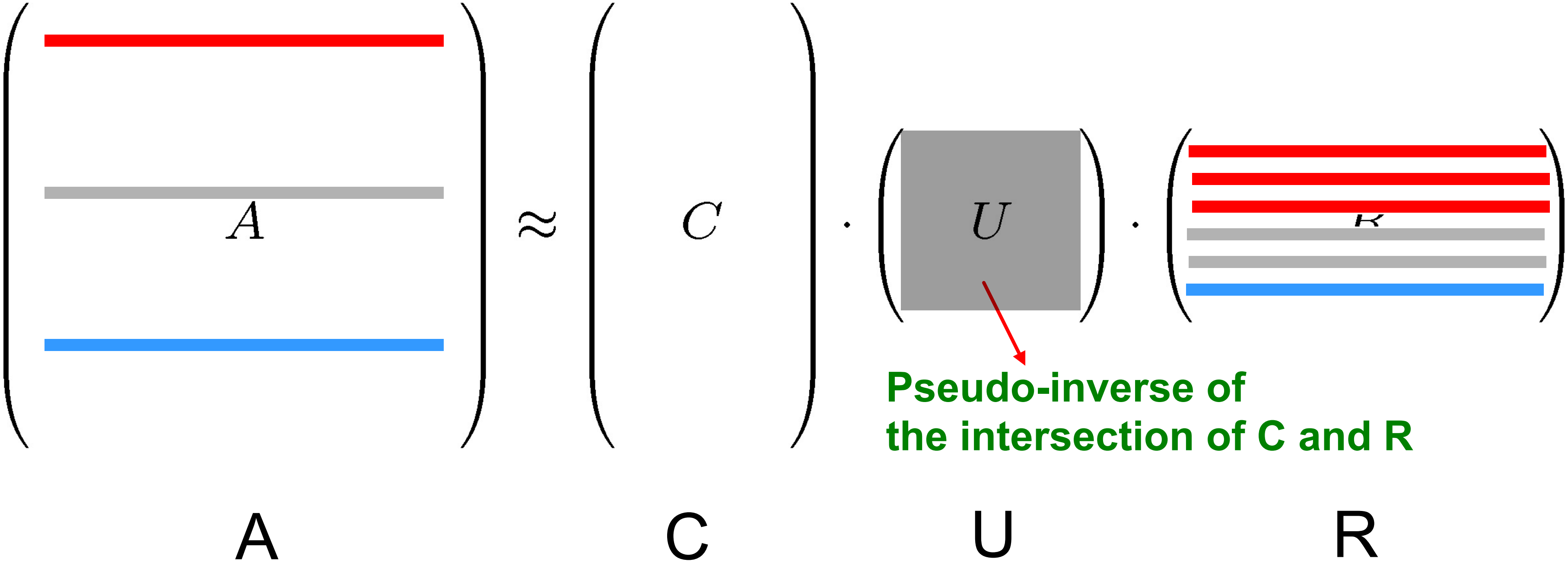
Frobenius norm:

$\|X\|_F = \sqrt{\Sigma_{ij} X_{ij}^2}$

- **Goal: Express A as a product of matrices C,U,R**

Make $\|A\text{-}C\cdot U\cdot R\|_F$ small

- **“Constraints” on C and R:**



Pseudo-inverse of the intersection of C and R

CUR: Provably good approx. to SVD

- **Let:**
  $\mathbf{A}_k$ be the "best" rank ***k*** approximation to **A** (that is, $\mathbf{A}_k$ is SVD of A)

Theorem [Drineas et al.]

CUR in O(**m·n**) time achieves

- $\|\mathbf{A}-\mathbf{CUR}\|_F \le \|\mathbf{A}-\mathbf{A}_k\|_F + \varepsilon\|\mathbf{A}\|_F$

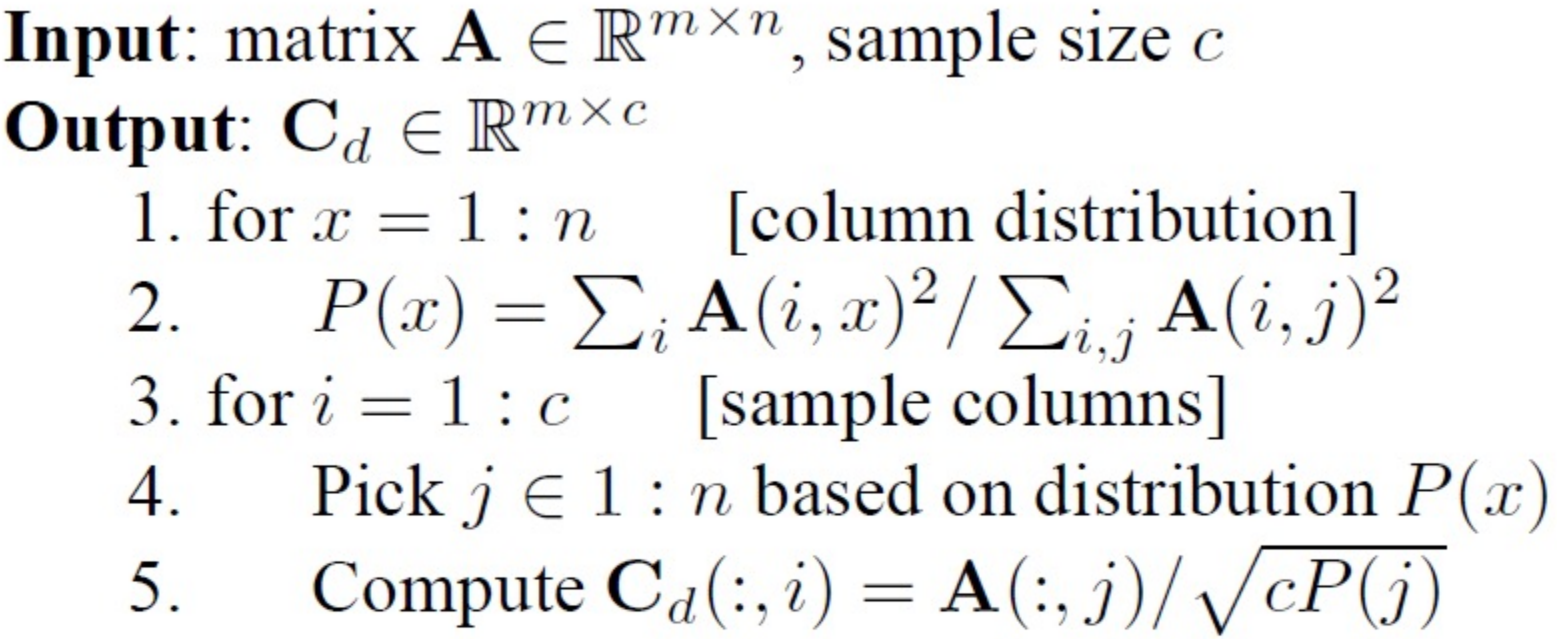
with probability at least **1-**δ, by picking

- **$O(k \log(1/\delta)/\varepsilon^2)$** columns, and
- **$O(k^2 \log^3(1/\delta)/\varepsilon^6)$** rows

In practice:
Pick 4*k* cols/rows

CUR: How it Works

- **Sampling columns (similarly for rows):**



Input: matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, sample size c

Output: $\mathbf{C}_d \in \mathbb{R}^{m \times c}$

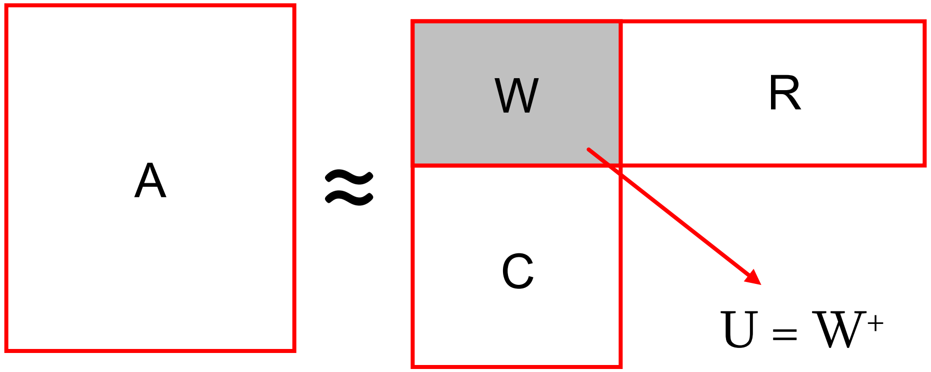
```
for $x = 1 : n$        [column distribution]
    $P(x) = \sum_i \mathbf{A}(i,x)^2 / \sum_{i,j} \mathbf{A}(i,j)^2$
for $i = 1 : c$        [sample columns]
    Pick $j \in 1 : n$ based on distribution $P(x)$
    Compute $\mathbf{C}_d(:,i) = \mathbf{A}(:,j) / \sqrt{cP(j)}$
```

Note this is a randomized algorithm, same column can be sampled more than once

Total power = c * E[C_d(:, i) ^{2}] = c * E [$\boldsymbol{A}^2$(:,j) / [c P(j)]] = c* $\sum_{j=1}^{n}$\{ $\boldsymbol{A}^2$(:,j) P(j)/ c P(j) \} = $\sum_{j=1}^{n}\boldsymbol{A}^2$(:, j)
i.e., same as the total power of the original matrix ***A*** !!

Computing U

- Let **W** be the "intersection" of sampled columns **C** and rows **R**
  - Let SVD of **W** = **X Z Y**^{T}
- **Then: U = W$^+$ = Y Z$^+$ X^T**
  - Z^+: **reciprocals of non-zero singular values:** $Z^+_{ii} = 1/Z_{ii}$
  - W$^+$ is the "**pseudoinverse**"

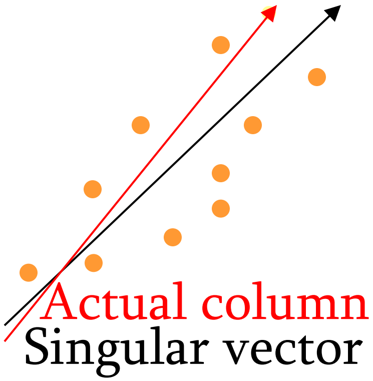


Why pseudoinverse works?
$W = X Z Y^T$
then $W^{-1} = (Y^T)^{-1} Z^{-1} X^{-1}$
$= Y Z^{-1} X^T$

Due to orthonomality
$X^{-1} = X^T$ and $Y^{-1} = Y^T$
Since Z is diagonal $Z^{-1} = 1/Z_{ii}$
Thus, if **W** is nonsingular, pseudoinverse is the true inverse

CUR: Pros & Cons

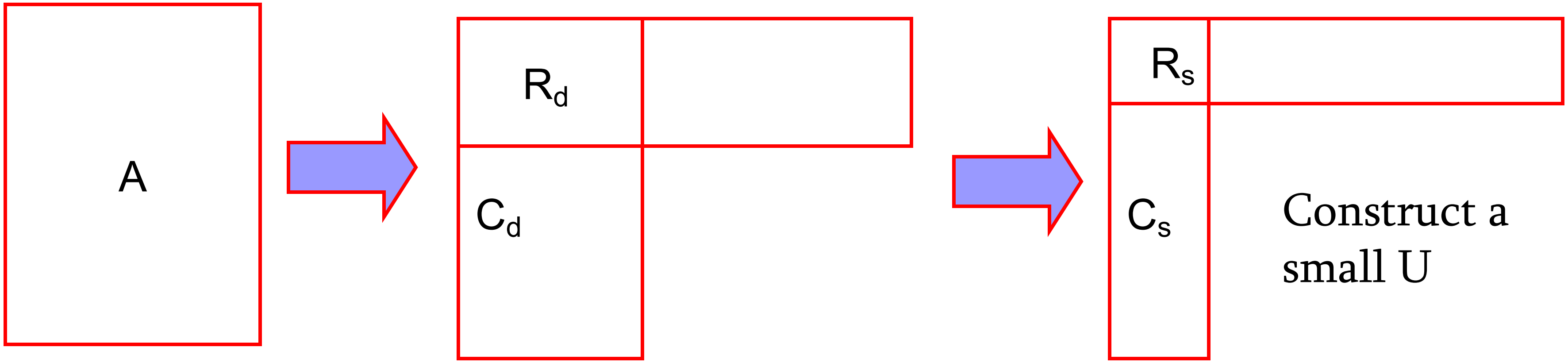
+ **Easy interpretation**
  - Since the basis vectors are actual columns and rows
+ **Sparse basis**
  - Since the basis vectors are actual columns and rows



− **Duplicate columns and rows**
  - Columns of large norms will be sampled many times

Solution

- **If we want to get rid of the duplicates:**
  - Throw them away
  - Scale (multiply) the columns/rows by the square root of the number of duplicates



SVD vs. CUR

sparse and small

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

Huge but sparse

Big and dense

dense but small

CUR: $A = C U R$

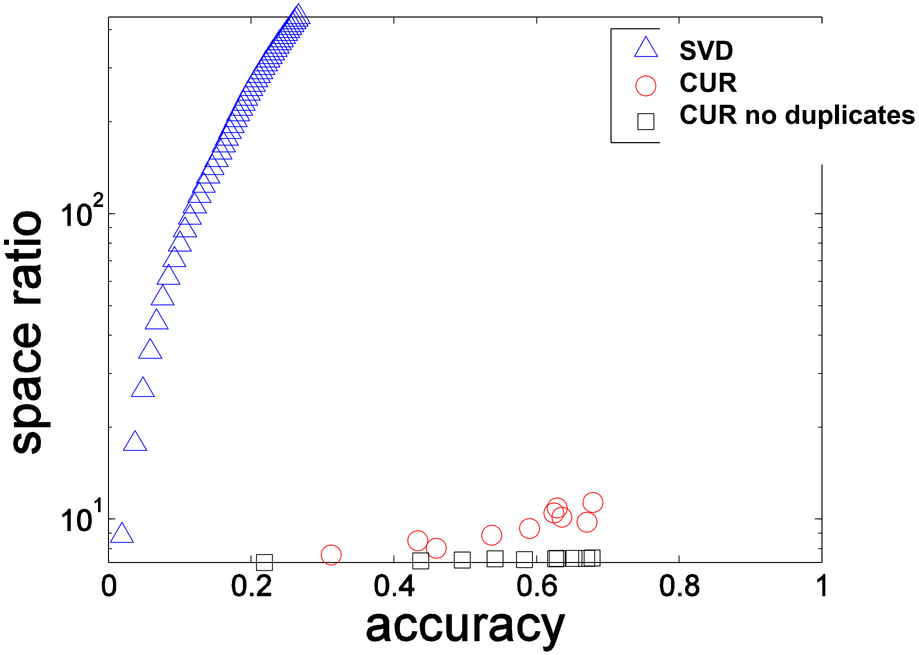
Huge but sparse

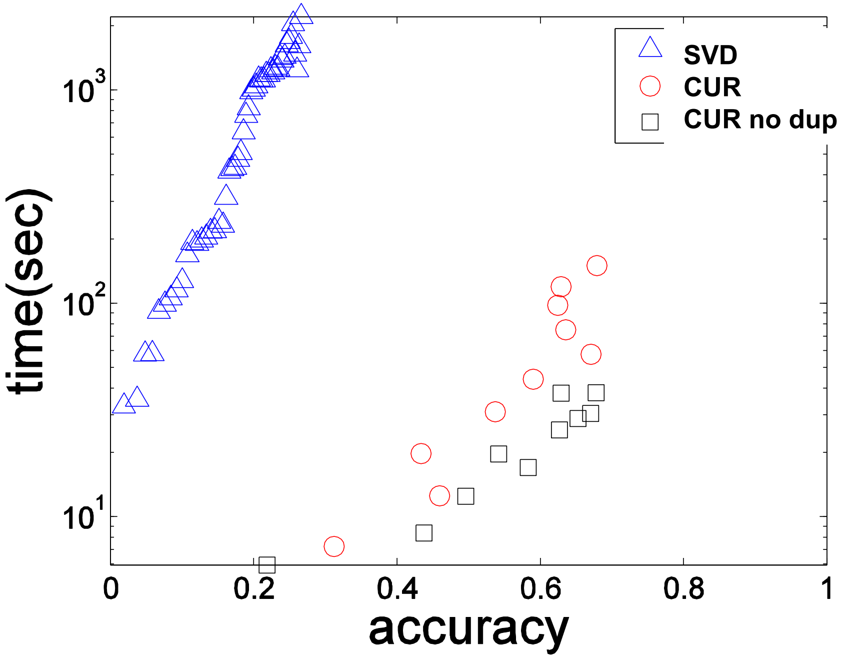
Big but sparse

Simple Experiment

- **DBLP bibliographic data**
  - Author-to-conference big sparse matrix
  - A_{ij}: Number of papers published by author *i* at conference *j*
  - 428K authors (rows), 3659 conferences (columns)
    - **Very sparse**
- **Want to reduce dimensionality**
  - How much time does it take?
  - What is the reconstruction error?
  - How much space do we need?

Results: DBLP- big sparse matrix



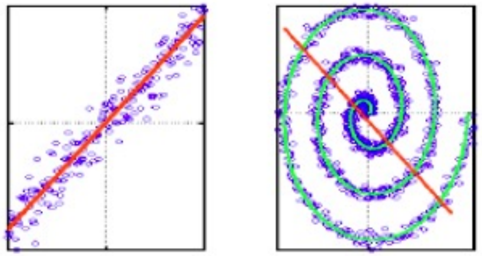


- **Accuracy:**
  - 1 – relative sum squared errors
- **Space ratio:**
  - #output matrix entries / #input matrix entries
- **CPU time**

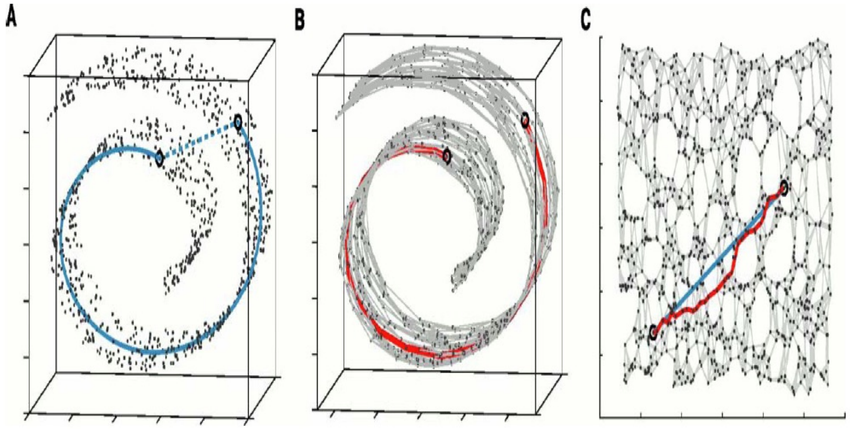
Sun, Faloutsos: *Less is More: Compact Matrix Decomposition for Large Sparse Graphs*, SDM '07

What about linearity assumption?

- **SVD is limited to linear projections:**
  - Lower-dimensional linear projection that preserves Euclidean distances



- Non-linear methods: **Isomap**
  - Data lies on a nonlinear low-dim curve aka manifold
    - Use the distance as measured along the manifold
  - **How?**
    - Build adjacency graph
    - Geodesic distance is graph distance
    - SVD/PCA the graph pairwise distance matrix



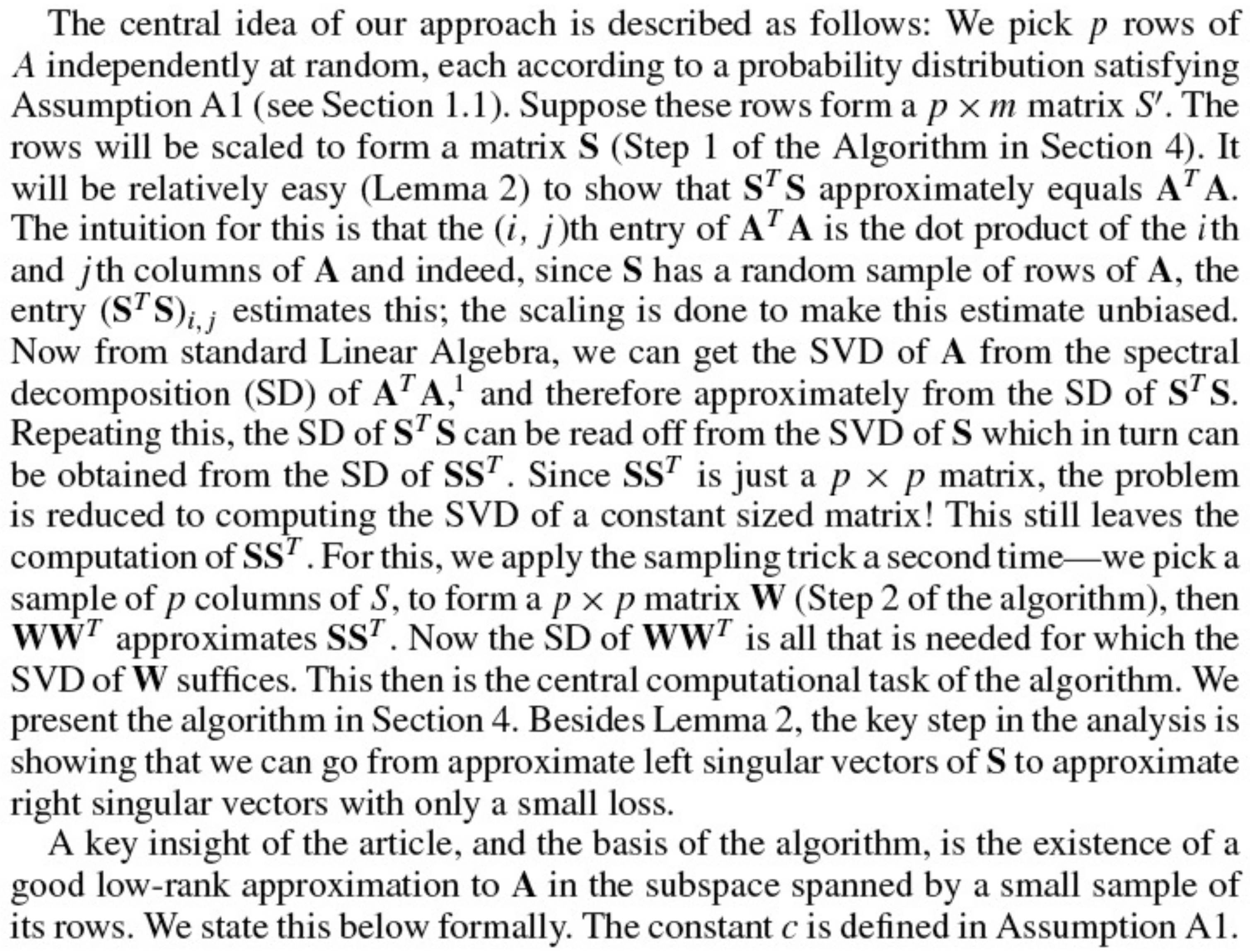
Further Reading for CUR

- *Frieze A, Kannan R, Vempala S (2004) Fast Monte-Carlo algorithms for finding low-rank approximations. J ACM 51(6):1025–1041.*
- Drineas et al., *Fast Monte Carlo Algorithms for Matrices III: Computing a Compressed Approximate Matrix Decomposition*, SIAM Journal on Computing, 2006.
- J. Sun, Y. Xie, H. Zhang, C. Faloutsos: *Less is More: Compact Matrix Decomposition for Large Sparse Graphs*, SDM 2007
- *Intra- and interpopulation genotype reconstruction from tagging SNPs*, P. Paschou, M. W. Mahoney, A. Javed, J. R. Kidd, A. J. Pakstis, S. Gu, K. K. Kidd, and P. Drineas, Genome Research, 17(1), 96-107 (2007)
- *Tensor-CUR Decompositions For Tensor-Based Data*, M. W. Mahoney, M. Maggioni, and P. Drineas, Proc. 12-th Annual SIGKDD, 327-336 (2006)
- *CUR Matrix Decompositions for Improved Data Analysis, M. W. Mahoney and P. Drineas, Proc. Natl. Acad. Sci. USA, 106, 697-702 (2009)*
- *Optimal CUR Matrix Decompositions, C. Boutsidis, D.P. Woodruff, STOC 2014, http://arxiv.org/abs/1405.7910*

Backup Slides

Intuition of CUR

from Frieze A, Kannan R, Vempala S (2004) Fast Monte-Carlo algorithms for finding low-rank approximations. J ACM 51(6):1025–1041.

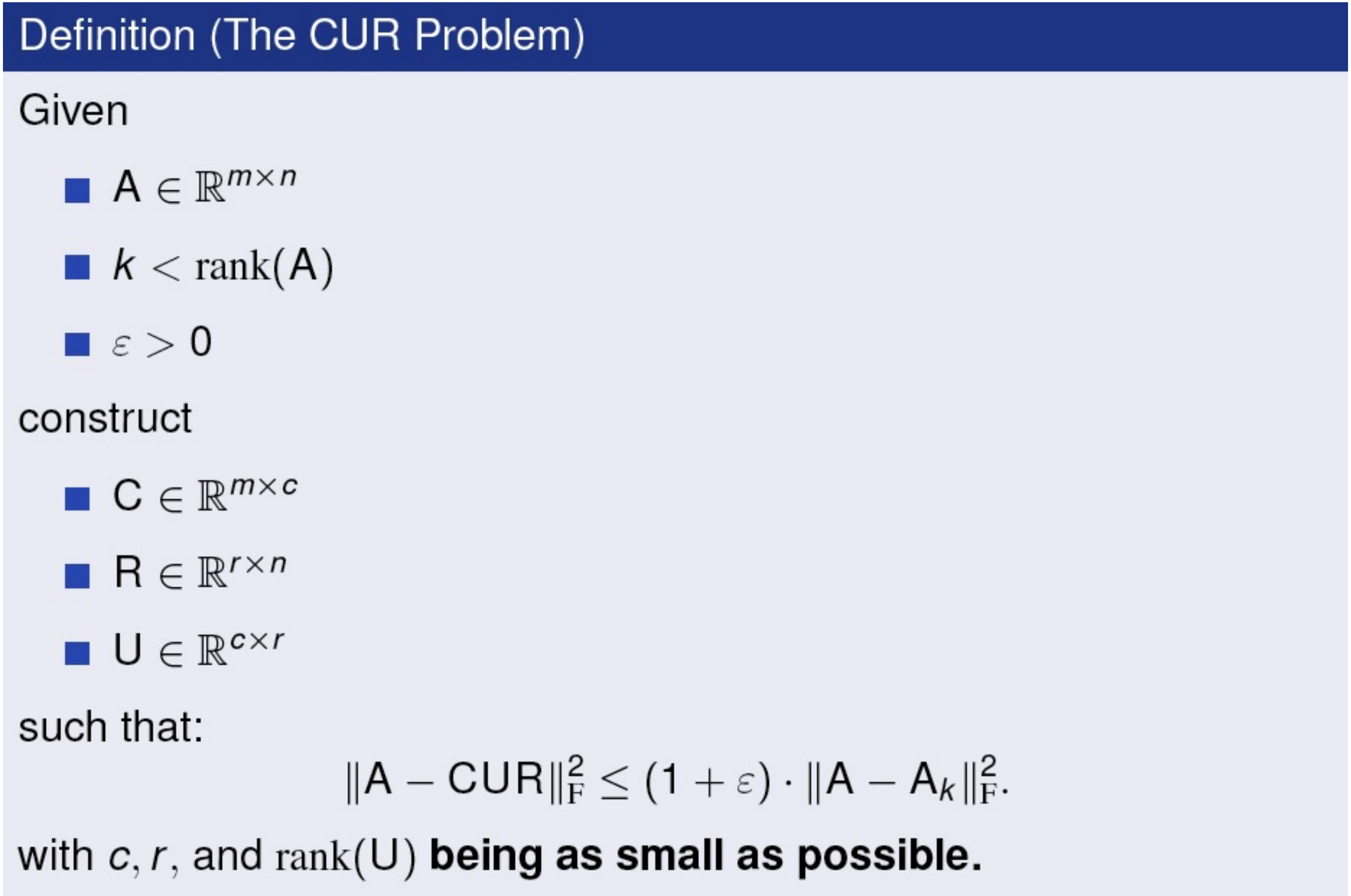


The central idea of our approach is described as follows: We pick p rows of A independently at random, each according to a probability distribution satisfying Assumption A1 (see Section 1.1). Suppose these rows form a $p \times m$ matrix S'. The rows will be scaled to form a matrix $\mathbf{S}$ (Step 1 of the Algorithm in Section 4). It will be relatively easy (Lemma 2) to show that $\mathbf{S}^T\mathbf{S}$ approximately equals $\mathbf{A}^T\mathbf{A}$. The intuition for this is that the (i, j)th entry of $\mathbf{A}^T\mathbf{A}$ is the dot product of the ith and jth columns of $\mathbf{A}$ and indeed, since $\mathbf{S}$ has a random sample of rows of $\mathbf{A}$, the entry $(\mathbf{S}^T\mathbf{S})_{i,j}$ estimates this; the scaling is done to make this estimate unbiased. Now from standard Linear Algebra, we can get the SVD of $\mathbf{A}$ from the spectral decomposition (SD) of $\mathbf{A}^T\mathbf{A}$,[1] and therefore approximately from the SD of $\mathbf{S}^T\mathbf{S}$. Repeating this, the SD of $\mathbf{S}^T\mathbf{S}$ can be read off from the SVD of $\mathbf{S}$ which in turn can be obtained from the SD of $\mathbf{S}\mathbf{S}^T$. Since $\mathbf{S}\mathbf{S}^T$ is just a $p \times p$ matrix, the problem is reduced to computing the SVD of a constant sized matrix! This still leaves the computation of $\mathbf{S}\mathbf{S}^T$. For this, we apply the sampling trick a second time—we pick a sample of p columns of S, to form a $p \times p$ matrix $\mathbf{W}$ (Step 2 of the algorithm), then $\mathbf{W}\mathbf{W}^T$ approximates $\mathbf{S}\mathbf{S}^T$. Now the SD of $\mathbf{W}\mathbf{W}^T$ is all that is needed for which the SVD of $\mathbf{W}$ suffices. This then is the central computational task of the algorithm. We present the algorithm in Section 4. Besides Lemma 2, the key step in the analysis is showing that we can go from approximate left singular vectors of $\mathbf{S}$ to approximate right singular vectors with only a small loss.

A key insight of the article, and the basis of the algorithm, is the existence of a good low-rank approximation to $\mathbf{A}$ in the subspace spanned by a small sample of its rows. We state this below formally. The constant c is defined in Assumption A1.

State of the Art work on Optimal CUR

http://mmds-data.org/presentations/woodruff_mmds14.pdf



Definition (The CUR Problem)

Given

- $A \in \mathbb{R}^{m \times n}$
- $k < \mathrm{rank}(A)$
- $\varepsilon > 0$

construct

- $C \in \mathbb{R}^{m \times c}$
- $R \in \mathbb{R}^{r \times n}$
- $U \in \mathbb{R}^{c \times r}$

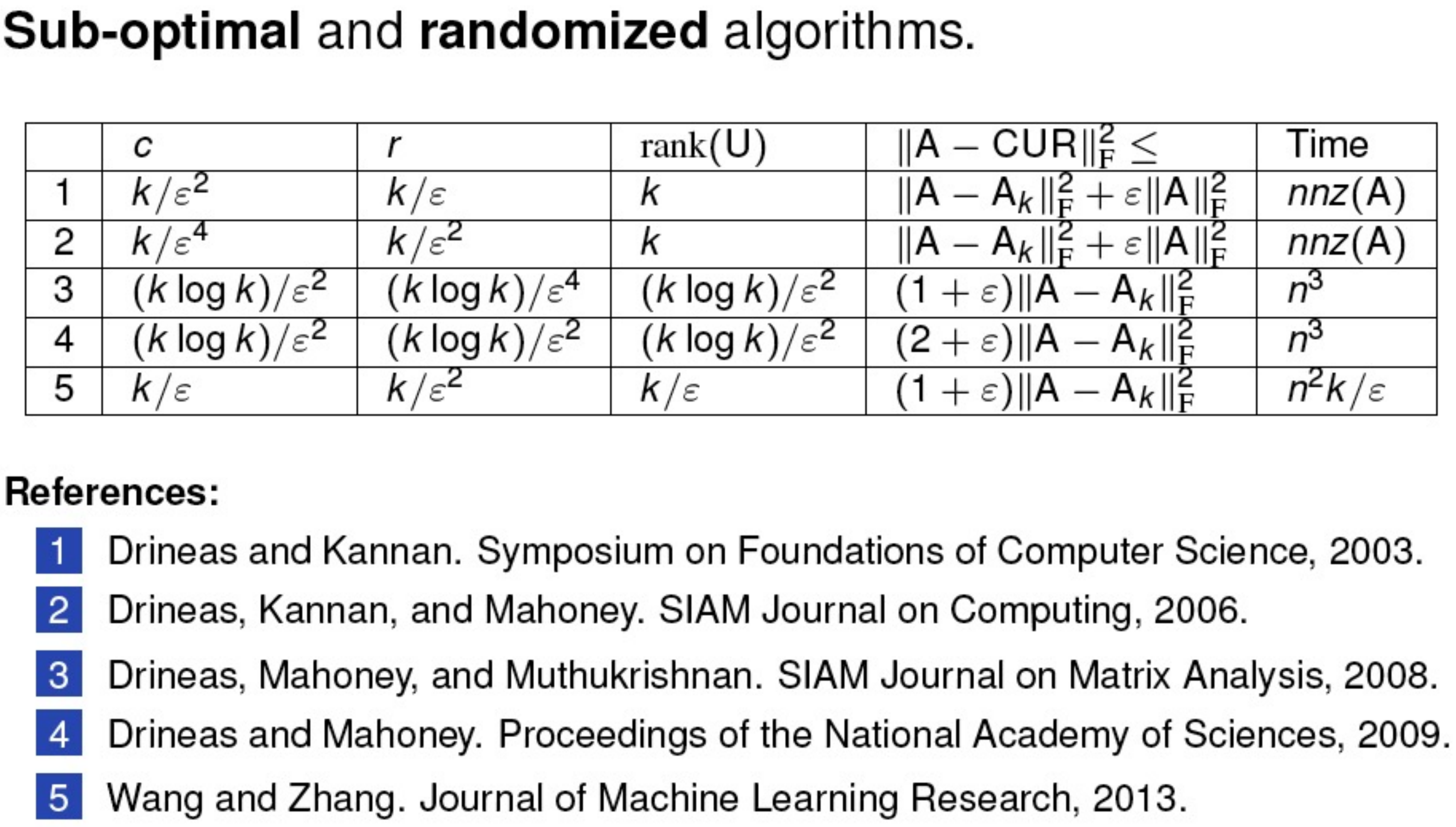
such that:

$$\|A - CUR\|_F^2 \leq (1 + \varepsilon) \cdot \|A - A_k\|_F^2.$$

with c, r, and $\mathrm{rank}(U)$ **being as small as possible.**

Prior Art on CUR

http://mmds-data.org/presentations/woodruff_mmds14.pdf



Sub-optimal and **randomized** algorithms.

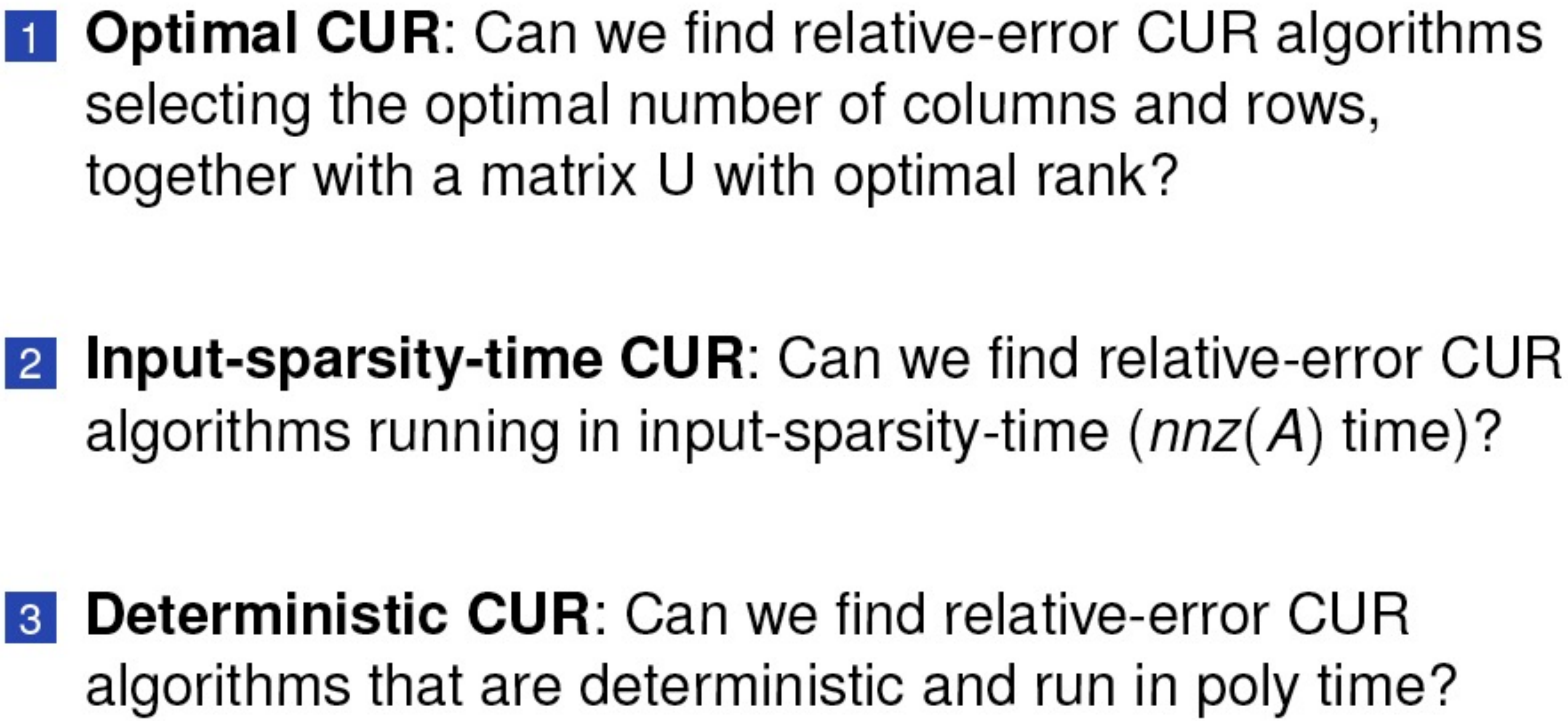
	c	r	$\mathrm{rank}(U)$	$\|A - CUR\|_F^2 \leq$	Time
1	k/ε^2	k/ε	k	$\|A - A_k\|_F^2 + \varepsilon\|A\|_F^2$	$nnz(A)$
2	k/ε^4	k/ε^2	k	$\|A - A_k\|_F^2 + \varepsilon\|A\|_F^2$	$nnz(A)$
3	$(k \log k)/\varepsilon^2$	$(k \log k)/\varepsilon^4$	$(k \log k)/\varepsilon^2$	$(1 + \varepsilon)\|A - A_k\|_F^2$	n^3
4	$(k \log k)/\varepsilon^2$	$(k \log k)/\varepsilon^2$	$(k \log k)/\varepsilon^2$	$(2 + \varepsilon)\|A - A_k\|_F^2$	n^3
5	k/ε	k/ε^2	k/ε	$(1 + \varepsilon)\|A - A_k\|_F^2$	$n^2 k/\varepsilon$

References:

1. Drineas and Kannan. Symposium on Foundations of Computer Science, 2003.
2. Drineas, Kannan, and Mahoney. SIAM Journal on Computing, 2006.
3. Drineas, Mahoney, and Muthukrishnan. SIAM Journal on Matrix Analysis, 2008.
4. Drineas and Mahoney. Proceedings of the National Academy of Sciences, 2009.
5. Wang and Zhang. Journal of Machine Learning Research, 2013.

Prior Open Problems on Optimal CUR

http://mmds-data.org/presentations/woodruff_mmds14.pdf



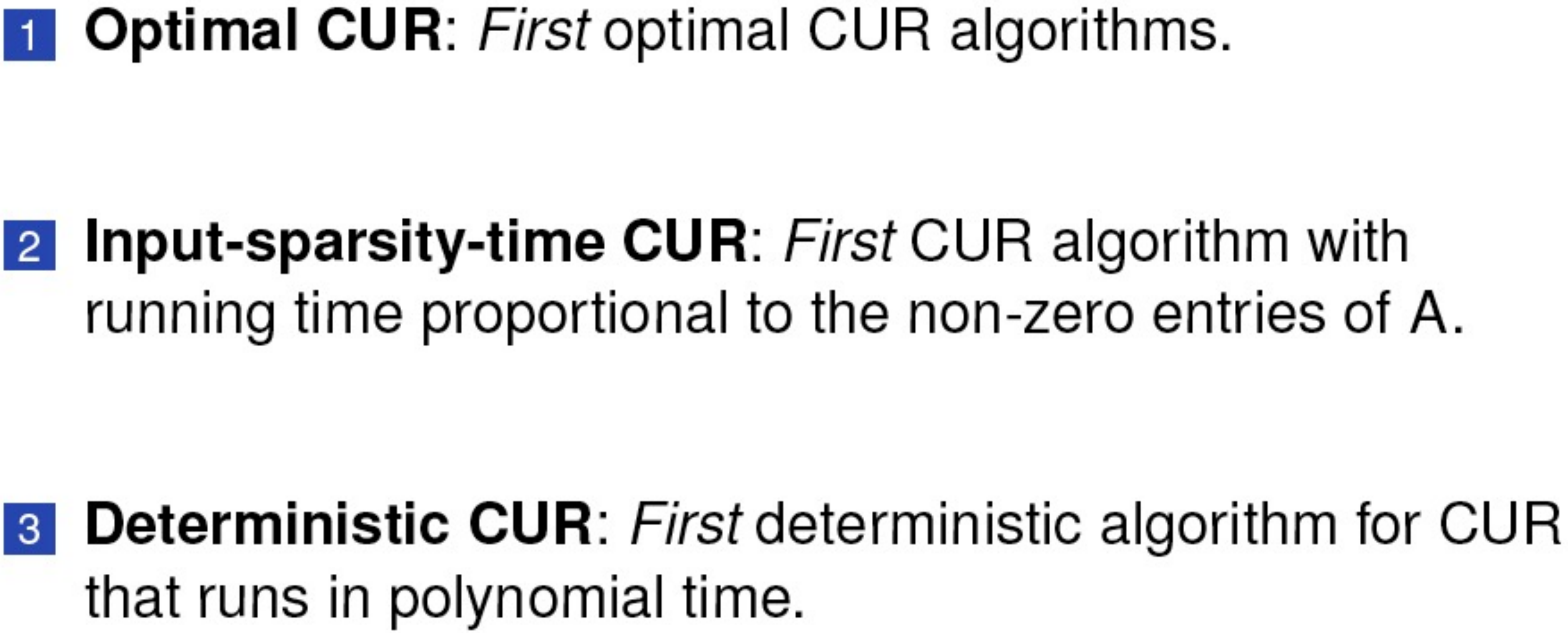
1. **Optimal CUR**: Can we find relative-error CUR algorithms selecting the optimal number of columns and rows, together with a matrix U with optimal rank?

2. **Input-sparsity-time CUR**: Can we find relative-error CUR algorithms running in input-sparsity-time ($nnz(A)$ time)?

3. **Deterministic CUR**: Can we find relative-error CUR algorithms that are deterministic and run in poly time?

Summary of recent Results on Optimal CUR

http://mmds-data.org/presentations/woodruff_mmds14.pdf



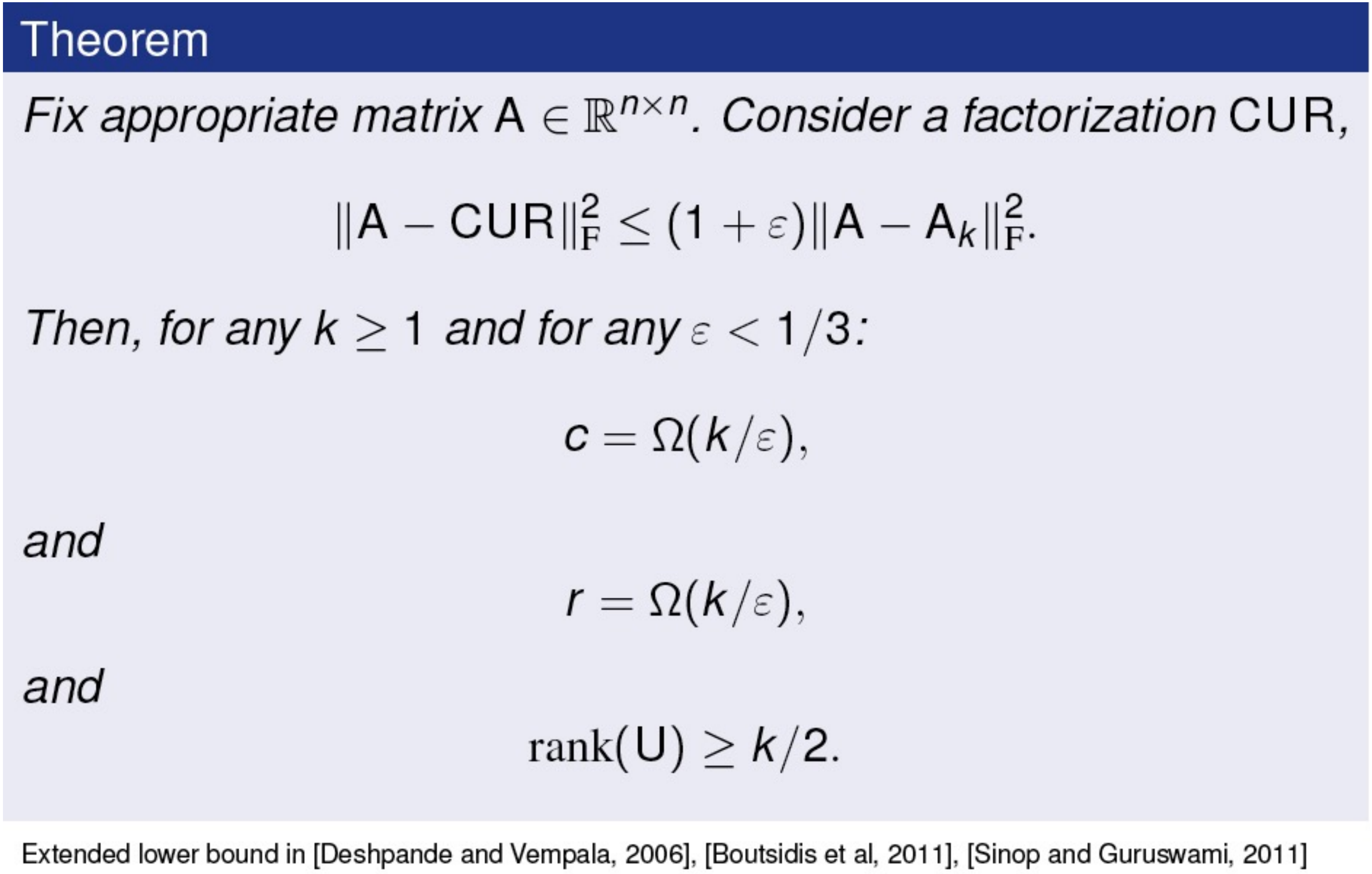
1 **Optimal CUR**: *First* optimal CUR algorithms.

2 **Input-sparsity-time CUR**: *First* CUR algorithm with running time proportional to the non-zero entries of A.

3 **Deterministic CUR**: *First* deterministic algorithm for CUR that runs in polynomial time.

Lower Bound Results on Optimal CUR

http://mmds-data.org/presentations/woodruff_mmds14.pdf



Theorem

Fix appropriate matrix $\mathrm{A} \in \mathbb{R}^{n \times n}$. *Consider a factorization* CUR,

$$\|\mathrm{A} - \mathrm{CUR}\|_{\mathrm{F}}^2 \leq (1+\varepsilon)\|\mathrm{A} - \mathrm{A}_k\|_{\mathrm{F}}^2.$$

Then, for any $k \geq 1$ *and for any* $\varepsilon < 1/3$*:*

$$c = \Omega(k/\varepsilon),$$

and

$$r = \Omega(k/\varepsilon),$$

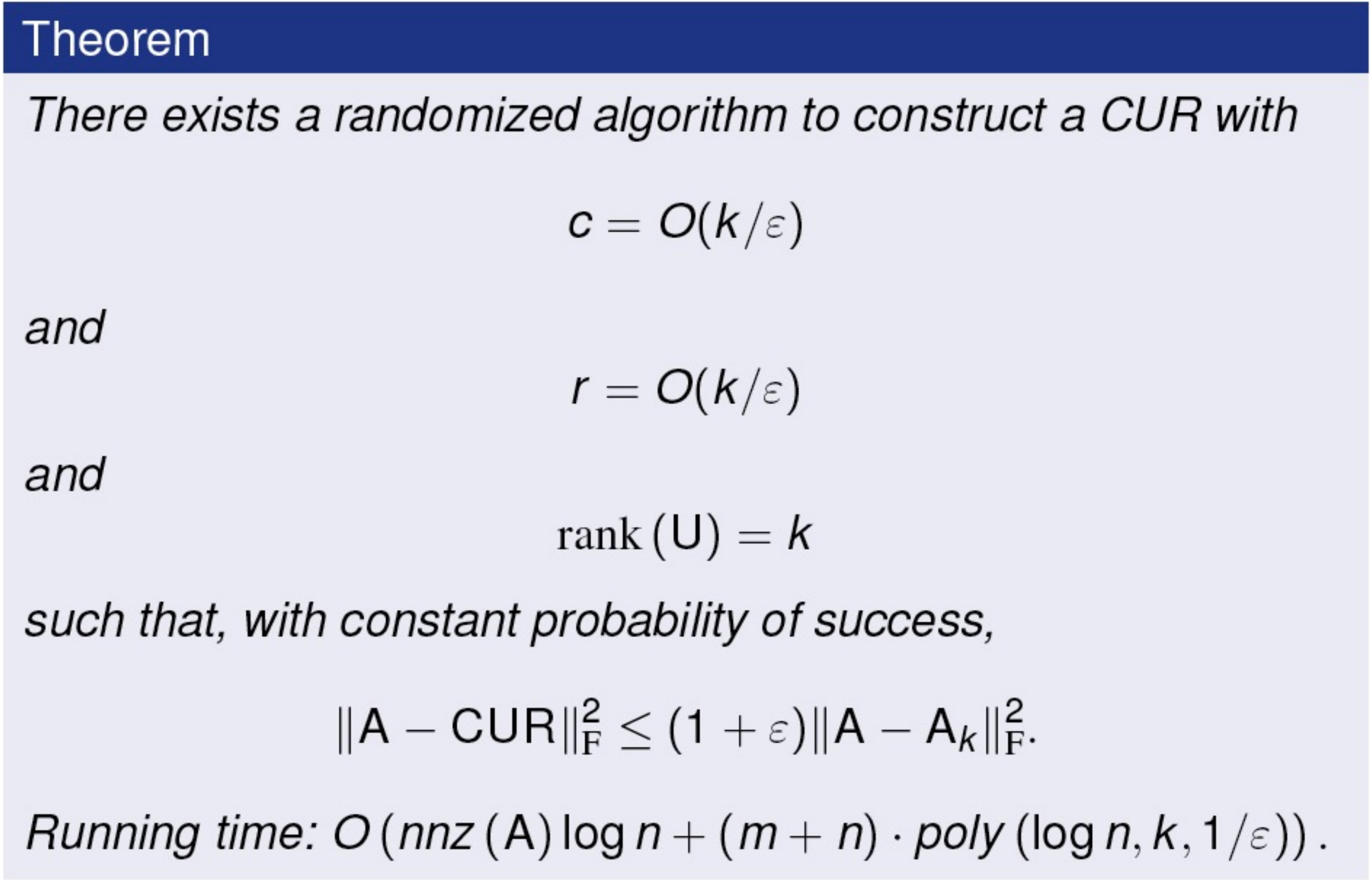
and

$$\mathrm{rank}(\mathrm{U}) \geq k/2.$$

Extended lower bound in [Deshpande and Vempala, 2006], [Boutsidis et al, 2011], [Sinop and Guruswami, 2011]

Input-sparsity-time CUR

http://mmds-data.org/presentations/woodruff_mmds14.pdf



Theorem

There exists a randomized algorithm to construct a CUR with

$$c = O(k/\varepsilon)$$

and

$$r = O(k/\varepsilon)$$

and

$$\mathrm{rank}\,(\mathsf{U}) = k$$

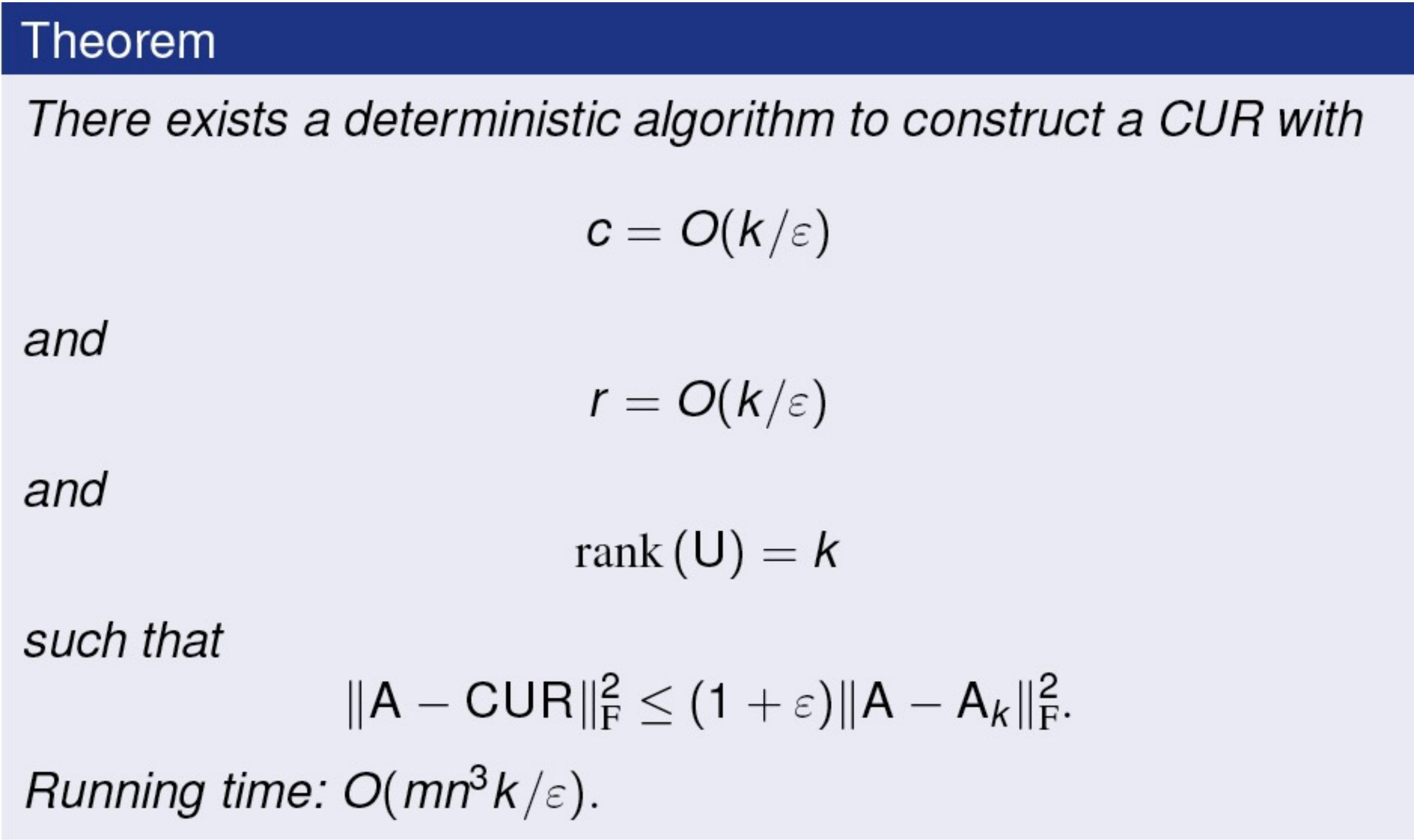
such that, with constant probability of success,

$$\|\mathsf{A} - \mathsf{CUR}\|_{\mathrm{F}}^2 \leq (1+\varepsilon)\|\mathsf{A} - \mathsf{A}_k\|_{\mathrm{F}}^2.$$

Running time: $O\left(nnz\,(\mathsf{A})\log n + (m+n)\cdot poly\,(\log n, k, 1/\varepsilon)\right).$

Deterministic CUR

http://mmds-data.org/presentations/woodruff_mmds14.pdf



Theorem

There exists a deterministic algorithm to construct a CUR with

$$c = O(k/\varepsilon)$$

and

$$r = O(k/\varepsilon)$$

and

$$\mathrm{rank}\,(\mathrm{U}) = k$$

such that

$$\|\mathrm{A} - \mathrm{CUR}\|_{\mathrm{F}}^2 \leq (1 + \varepsilon)\|\mathrm{A} - \mathrm{A}_k\|_{\mathrm{F}}^2.$$

Running time: $O(mn^3k/\varepsilon)$.