



Erasmus+

Note to other teachers and users of these slides: We would be delighted if you found this material useful in giving your own lectures. Feel free to use these slides verbatim, or to modify them to fit your own needs. If you make use of a significant portion of these slides in your own lecture, please include this message, or a link to our web site: <http://www.mmds.org>

Dimensionality Reduction: SVD & CUR

Mining of Massive Datasets

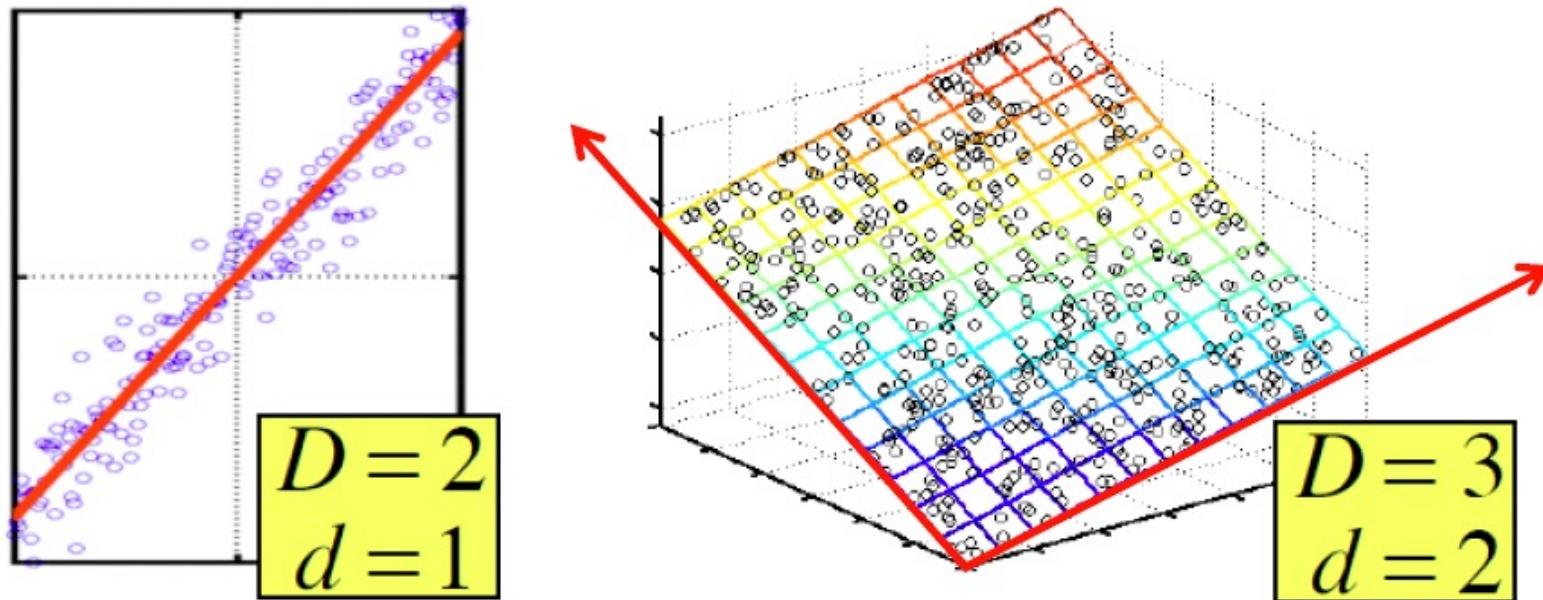
Jure Leskovec, Anand Rajaraman, Jeff Ullman

Stanford University

<http://www.mmds.org>



Dimensionality Reduction



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

Dimensionality Reduction

- **Compress / reduce dimensionality:**
 - 10^6 rows; 10^3 columns; no updates
 - Random access to any cell(s); **small error: OK**

customer	day	We	Th	Fr	Sa	Su
		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]

Rank of a Matrix

- **Q:** What is **rank** of a matrix **A**?
- **A:** Number of **linearly independent columns** of **A**
- **For example:**
 - Matrix **A** = $\begin{bmatrix} 1 & 2 & 1 \\ -2 & -3 & 1 \\ 3 & 5 & 0 \end{bmatrix}$ has rank **r=2**
 - **Why?** The first two rows are linearly independent, so the rank is at least 2, but all three rows are linearly dependent (the first is equal to the sum of the second and third) so the rank must be less than 3.
- **Why do we care about low rank?**
 - We can write **A** as two “basis” vectors: [1 2 1] [-2 -3 1]
 - And new coordinates of : [1 0] [0 1] [1 1]

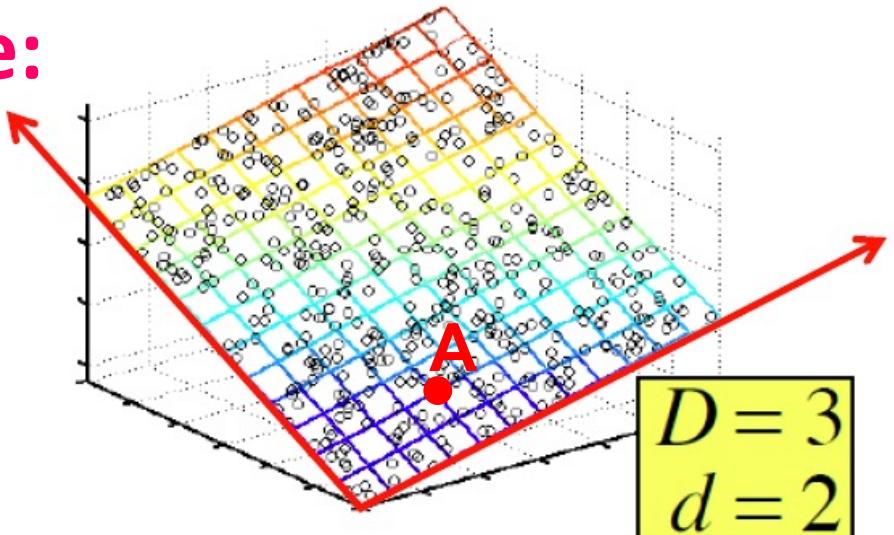
Rank is “Dimensionality”

- Cloud of points 3D space:

- Think of point positions as a matrix:

$$\begin{bmatrix} 1 & 2 & 1 \\ -2 & -3 & 1 \\ 3 & 5 & 0 \end{bmatrix} \begin{matrix} \text{A} \\ \text{B} \\ \text{C} \end{matrix}$$

1 row per point:

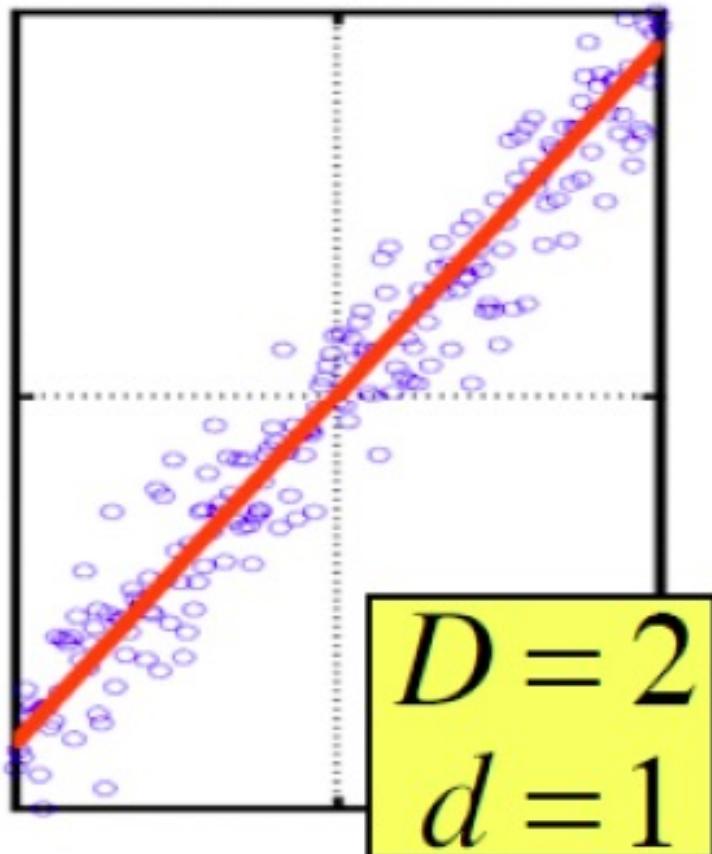


- We can rewrite coordinates more efficiently!

- Old basis vectors: $[1 \ 0 \ 0] \ [0 \ 1 \ 0] \ [0 \ 0 \ 1]$
 - New basis vectors: $[1 \ 2 \ 1] \ [-2 \ -3 \ 1]$
 - Then A has new coordinates: $[1 \ 0]$. B: $[0 \ 1]$, C: $[1 \ 1]$
 - Notice: We reduced the number of coordinates!

Dimensionality Reduction

- Goal of dimensionality reduction is to discover the axis of data!



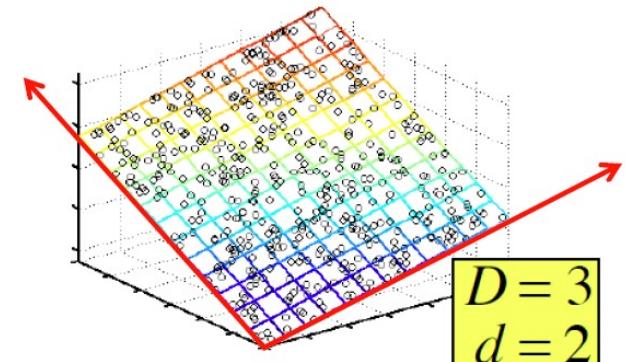
Rather than representing every point with 2 coordinates we represent each point with 1 coordinate (corresponding to the position of the point on the red line).

By doing this we incur a bit of error as the points do not exactly lie on the line

Why Reduce Dimensions?

Why reduce dimensions?

- **Discover hidden correlations/topics**
 - Words that occur commonly together
- **Remove redundant and noisy features**
 - Not all words are useful
- **Interpretation and visualization**
- **Easier storage and processing of the data**



SVD - Definition

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

- **A: Input data matrix**
 - $m \times n$ matrix (e.g., m documents, n terms)
- **U: Left singular vectors**
 - $m \times r$ matrix (m documents, r concepts)
- **Σ : Singular values**
 - $r \times r$ diagonal matrix (strength of each ‘concept’)
(r : rank of the matrix A)
- **V: Right singular vectors**
 - $n \times r$ matrix (n terms, r concepts)

SVD - Definition

		Titanic	Casablanca	Star Wars	Alien	Matrix
Joe	1	1	1	0	0	
Jim	3	3	3	0	0	
John	4	4	4	0	0	
Jack	5	5	5	0	0	
Jill	0	0	0	4	4	
Jenny	0	0	0	5	5	
Jane	0	0	0	2	2	

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 0 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} .14 & 0 \\ .42 & 0 \\ .56 & 0 \\ .70 & 0 \\ 0 & .60 \\ 0 & .75 \\ 0 & .30 \end{bmatrix} \begin{bmatrix} 12.4 & 0 \\ 0 & 9.5 \end{bmatrix} \begin{bmatrix} .58 & .58 & .58 & 0 & 0 \\ 0 & 0 & 0 & .71 & .71 \end{bmatrix}$$

M U Σ V^T

SVD - Properties

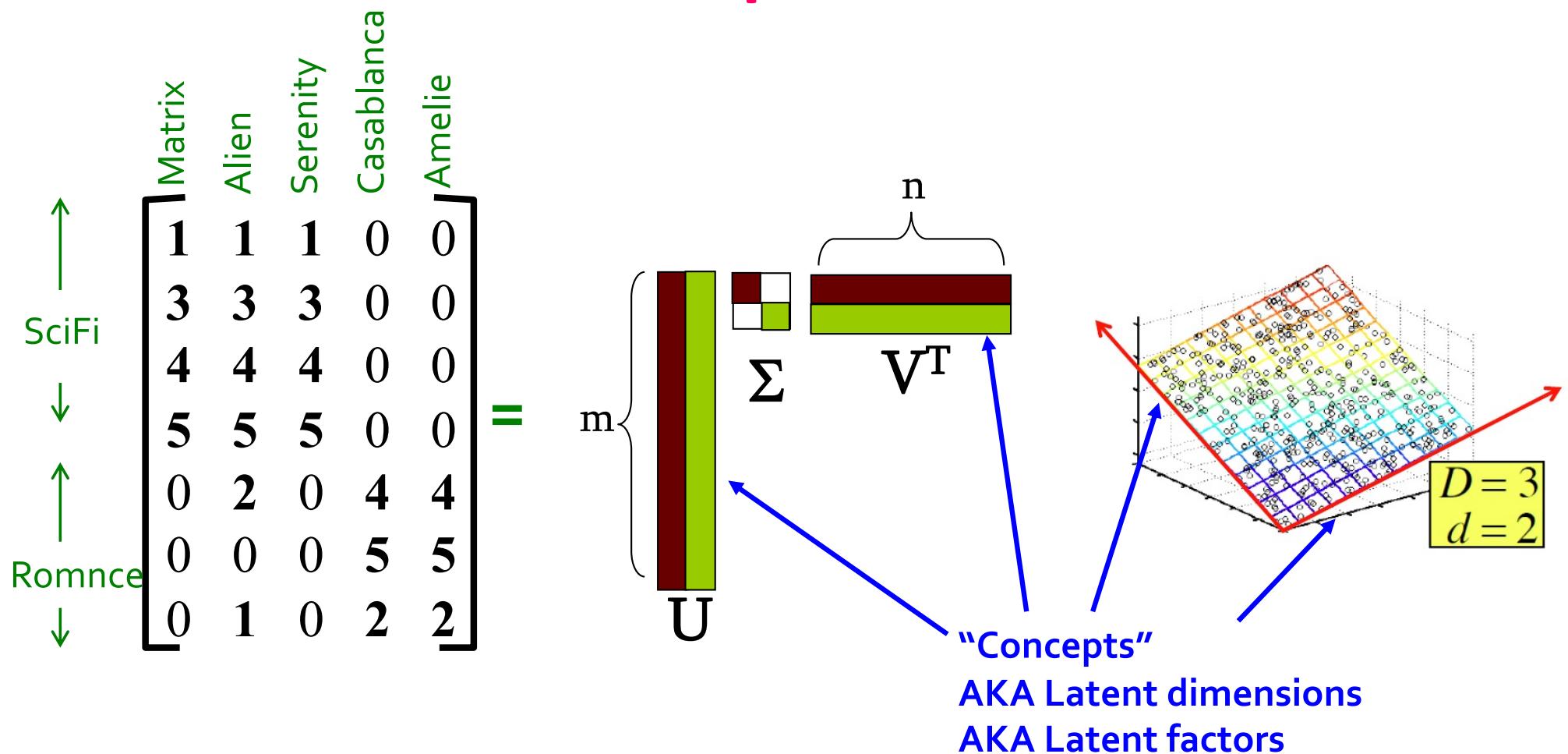
It is **always** possible to decompose a real matrix \mathbf{A} into $\mathbf{A} = \mathbf{U} \Sigma \mathbf{V}^T$, where

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

Nice proof of uniqueness: <http://www.mpi-inf.mpg.de/~bast/ir-seminar-wso4/lecture2.pdf>

SVD – Example: Users-to-Movies

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



SVD – Example: Users-to-Movies

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

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

SVD – Example: Users-to-Movies

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

$$\begin{array}{c}
 \begin{array}{c} \text{Matrix} \\ \uparrow \\ \text{SciFi} \\ \downarrow \\ \uparrow \\ \text{Romance} \\ \downarrow \end{array} & \left[\begin{array}{ccccc} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{array} \right] = \left[\begin{array}{ccc} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{array} \right] & \begin{array}{c} \text{SciFi-concept} \\ \downarrow \\ \text{Romance-concept} \end{array} \\
 & \times \left[\begin{array}{ccc} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{array} \right] \times & \\
 & \left[\begin{array}{ccccc} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{array} \right] &
 \end{array}$$

SVD – Example: Users-to-Movies

■ $A = U \Sigma V^T$ - example:

U is “user-to-concept” similarity matrix

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

SVD – Example: Users-to-Movies

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

Matrix

	Alien	Serenity	Casablanca	Amelie	
SciFi	1	1	1	0	0
	3	3	3	0	0
	4	4	4	0	0
	5	5	5	0	0
Romance	0	2	0	4	4
	0	0	0	5	5
	0	1	0	2	2

=

SciFi-concept

0.13	0.02	-0.01
0.41	0.07	-0.03
0.55	0.09	-0.04
0.68	0.11	-0.05
0.15	-0.59	0.65
0.07	-0.73	-0.67
0.07	-0.29	0.32

"strength" of the SciFi-concept

\times

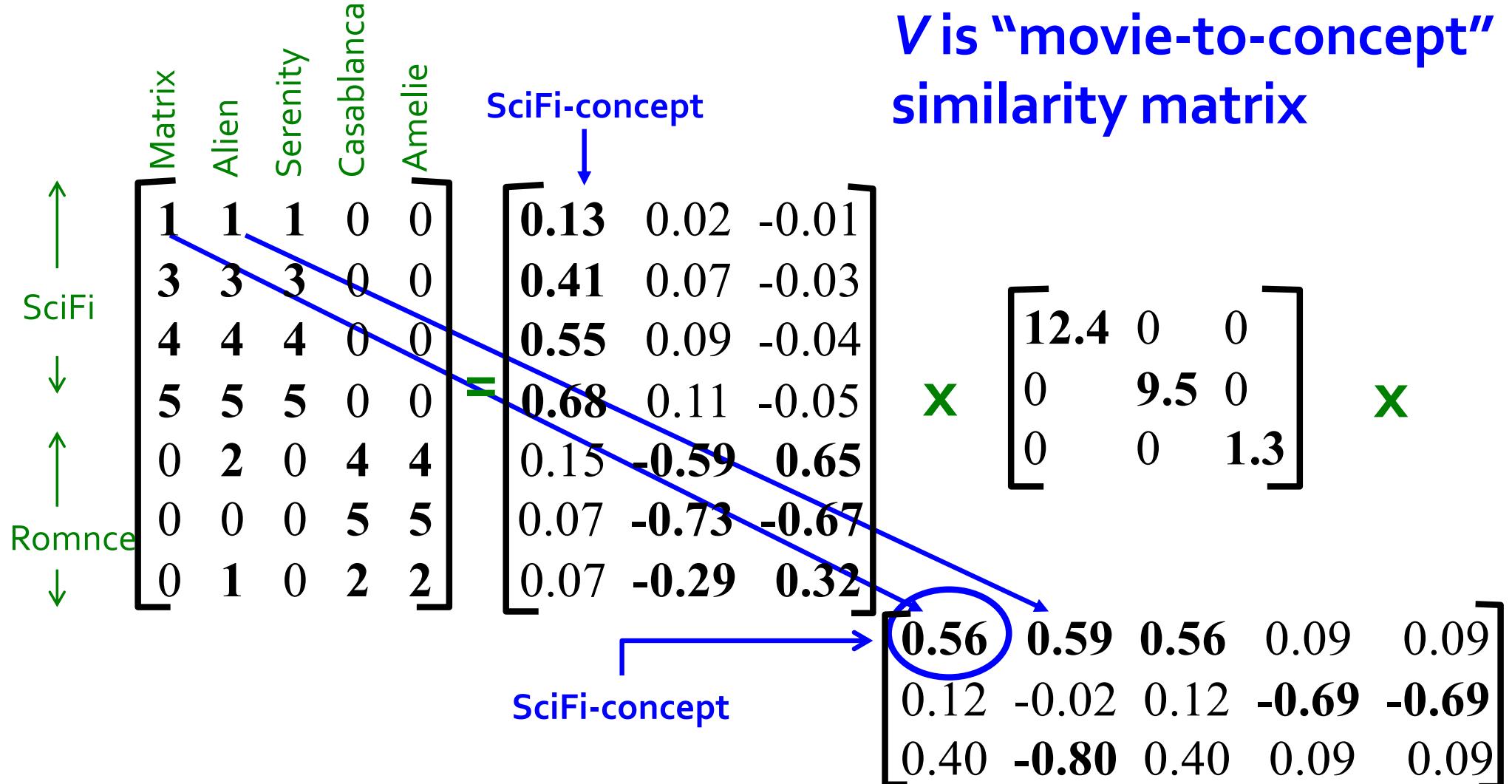
12.4	0	0
0	9.5	0
0	0	1.3

\times

0.56	0.59	0.56	0.09	0.09
0.12	-0.02	0.12	-0.69	-0.69
0.40	-0.80	0.40	0.09	0.09

SVD – Example: Users-to-Movies

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



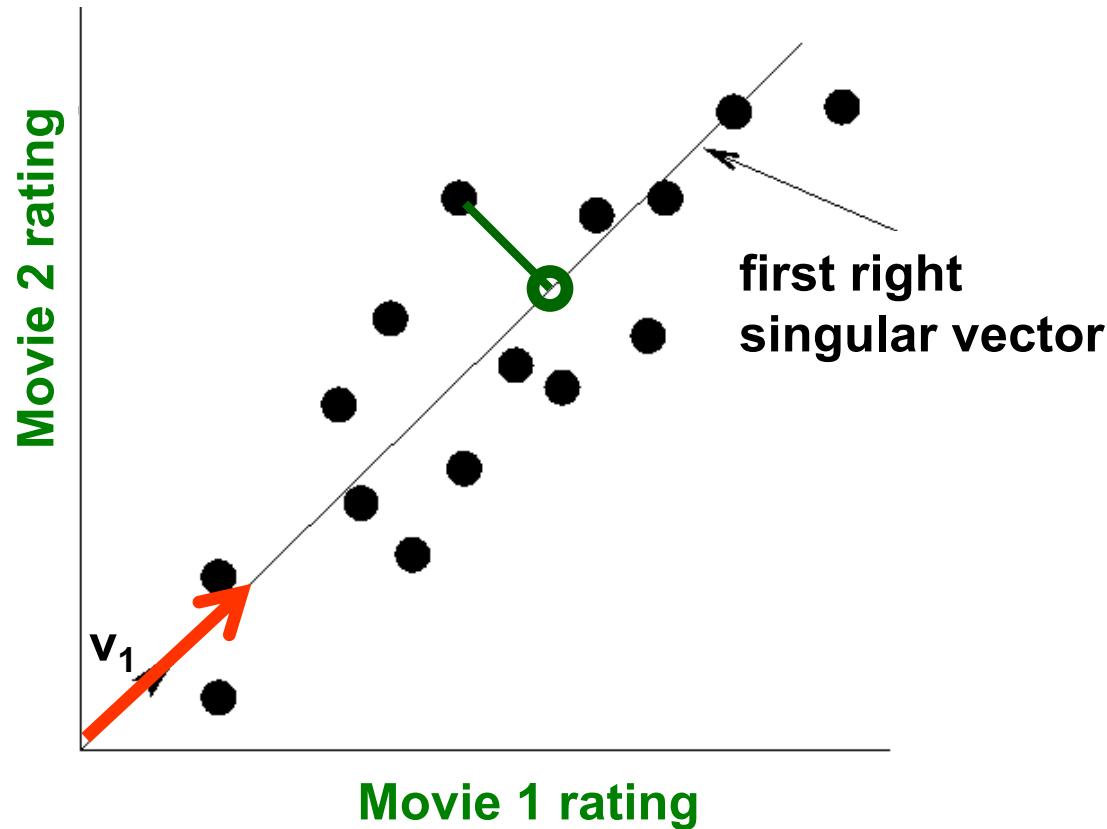
SVD - Interpretation #1

‘movies’, ‘users’ and ‘concepts’:

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

Dimensionality Reduction with SVD

SVD – Dimensionality Reduction



- Instead of using two coordinates (x, y) to describe point locations, let's use only one coordinate (z)
- Point's position is its location along vector v_1
- **How to choose v_1 ? Minimize reconstruction error**

SVD – Dimensionality Reduction

- **Goal:** Minimize the sum of reconstruction errors:

$$\sum_{i=1}^N \sum_{j=1}^D \|x_{ij} - z_{ij}\|^2$$

- where x_{ij} are the “old” and z_{ij} are the “new” coordinates
- **SVD gives ‘best’ axis to project on:**
 - ‘best’ = minimizing the reconstruction errors
 - In other words, **minimum reconstruction error**

Movie 2 rating **Goal:** Minimize the sum of reconstruction errors:
$$\sum_{i=1}^N \sum_{j=1}^D \|x_{ij} - z_{ij}\|^2$$
 first right singular vector
■ where x_{ij} are the “old” and “new” coordinates
■ **SVD gives ‘best’ axis to project on:**
 v_1 ‘best’ = minimizing the reconstruction error
■ In other words, **minimum reconstruction error**

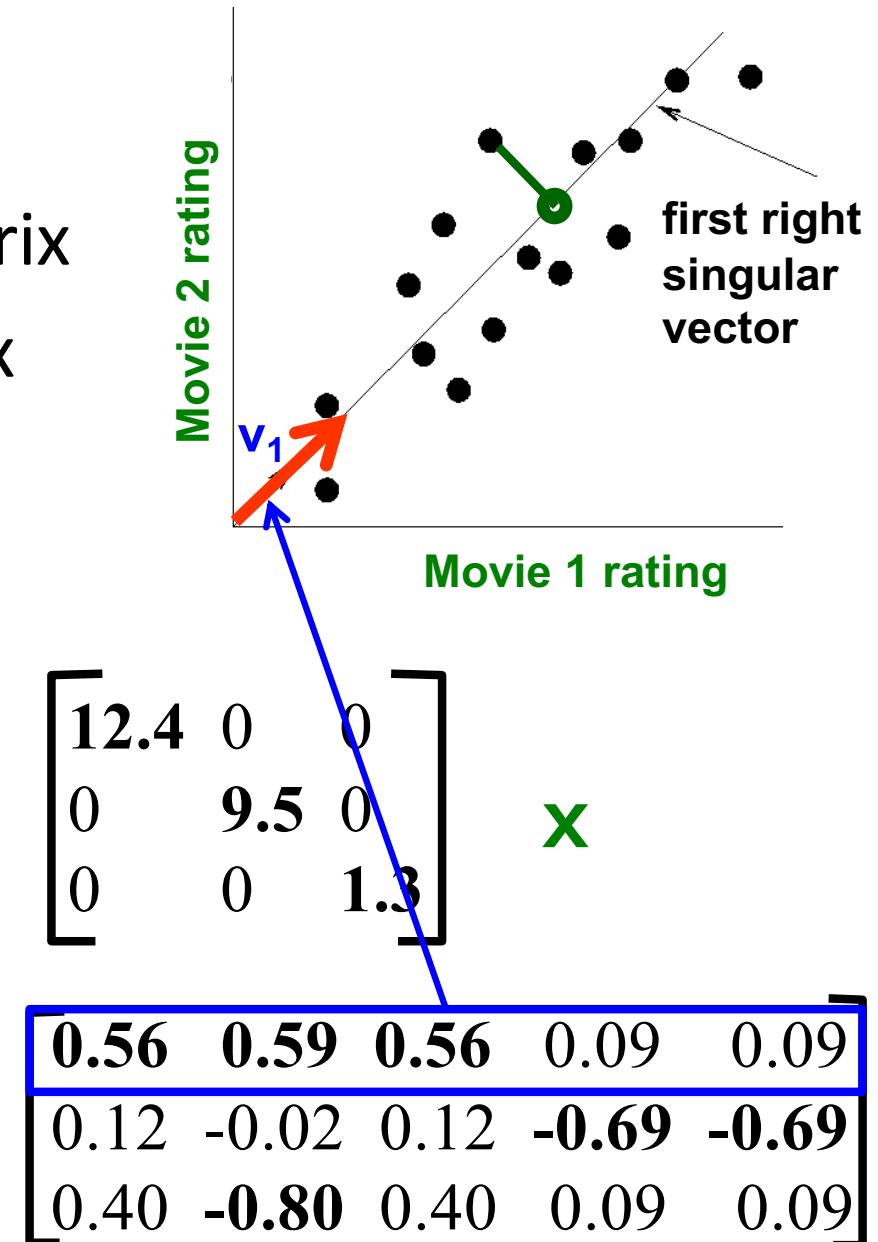
Movie 1 rating

SVD - Interpretation #2

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

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

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



$$\begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

SVD - Interpretation #2

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

variance ('spread')
on the v_1 axis

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

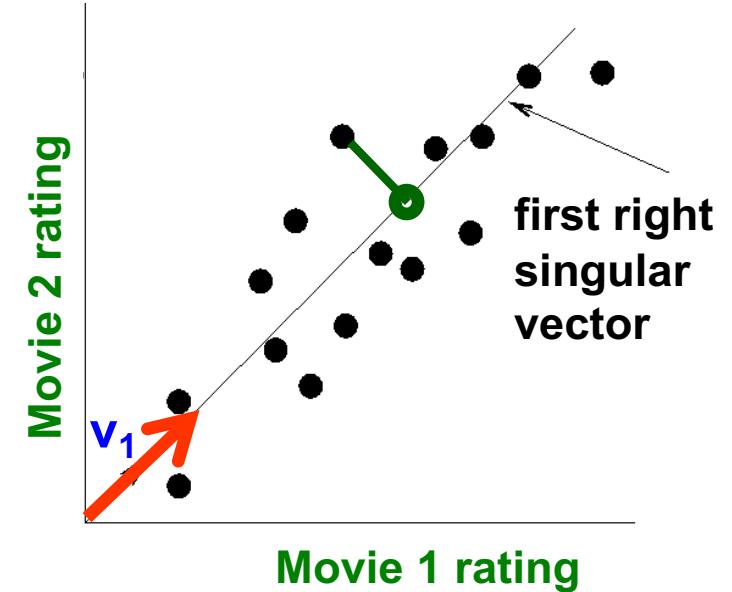
SVD - Interpretation #2

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

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

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

Projection of users on the “Sci-Fi” axis
 $(U \Sigma)^T$:



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

SVD - Interpretation #2

More details

- Q: How exactly is dim. reduction done?

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

SVD - Interpretation #2

More details

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

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

SVD - Interpretation #2

More details

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

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

SVD - Interpretation #2

More details

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

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

SVD - Interpretation #2

More details

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

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

SVD - Interpretation #2

More details

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

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

Frobenius norm:

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

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

is "small"

SVD - Interpretation #2

Equivalent:
'spectral decomposition' of the matrix

$$\begin{array}{c} \text{← } \mathbf{m} \text{ →} \\ \uparrow \quad \downarrow \\ \left[\begin{array}{ccccc} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{array} \right] = \sigma_1 \underbrace{\mathbf{u}_1}_{\mathbf{n} \times 1} \mathbf{v}_1^T + \sigma_2 \underbrace{\mathbf{u}_2}_{\mathbf{1} \times \mathbf{m}} \mathbf{v}_2^T + \dots \end{array}$$

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

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 σ s to keep?

A: Rule-of-a thumb:

keep 80-90% of 'energy' = $\sum_i \sigma_i^2$

$$\begin{array}{c} \xleftarrow{\hspace{-1cm} m } \\ \uparrow \text{n} \\ \left[\begin{array}{ccccc} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{array} \right] = \sigma_1 \quad u_1 \quad v^T_1 + \sigma_2 \quad u_2 \quad v^T_2 + \dots \end{array}$$

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

SVD - Complexity

- **To compute SVD:**
 - $O(nm^2)$ or $O(n^2m)$ (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: $A = U \Sigma V^T$: 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

Example of SVD & Conclusion

Case study: How to query?

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

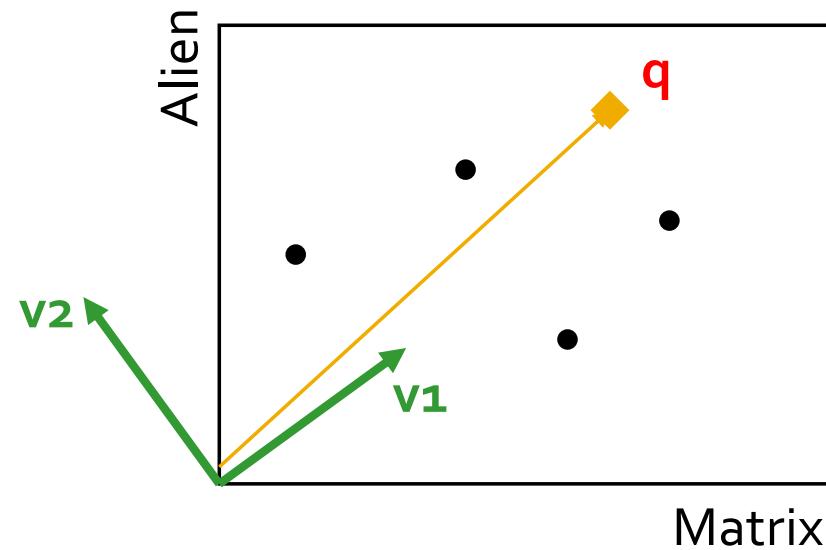
$$\begin{matrix} & \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\ \text{SciFi} \uparrow & \left[\begin{matrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{matrix} \right] & = & \left[\begin{matrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{matrix} \right] & \times & \left[\begin{matrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{matrix} \right] & \times \\ \downarrow & & & & & & & \times \\ \text{Romance} \downarrow & & & & & & & \end{matrix}$$
$$\left[\begin{matrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{matrix} \right]$$

Case study: How to query?

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

$$q = \begin{bmatrix} \text{Matrix} \\ \text{Alien} \\ \text{Serenity} \\ \text{Casablanca} \\ \text{Amelie} \end{bmatrix} \quad \begin{bmatrix} 5 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Project into concept space:
Inner product with each
'concept' vector v_i

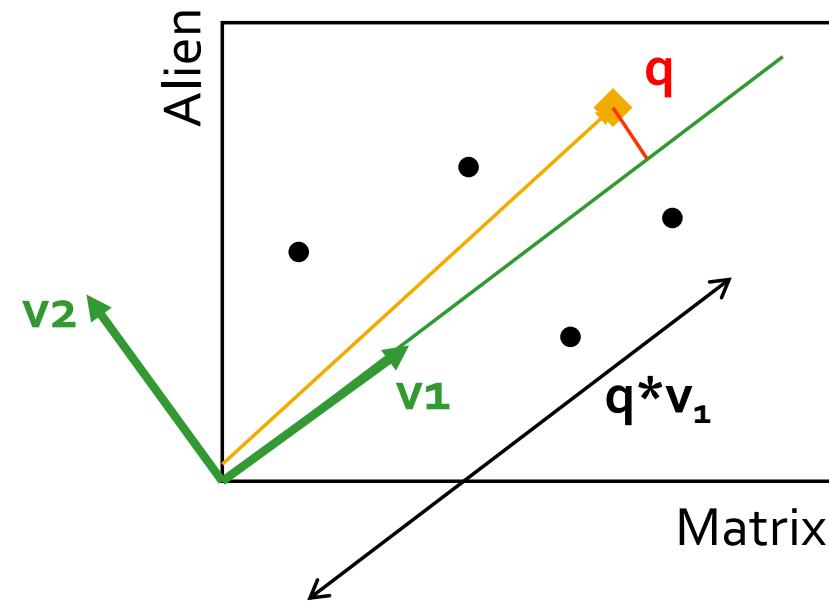


Case study: How to query?

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

$$q = \begin{bmatrix} \text{Matrix} \\ \text{Alien} \\ \text{Serenity} \\ \text{Casablanca} \\ \text{Amelie} \end{bmatrix} \quad \begin{bmatrix} 5 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Project into concept space:
Inner product with each
'concept' vector v_i



Case study: How to query?

Compactly, we have:

$$\mathbf{q}_{\text{concept}} = \mathbf{q} \mathbf{V}$$

E.g.:

$$\mathbf{q} = \begin{bmatrix} \text{Matrix} \\ \text{Alien} \\ \text{Serenity} \\ \text{Casablanca} \\ \text{Amelie} \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.12 \\ 0.59 & -0.02 \\ 0.56 & 0.12 \\ 0.09 & -0.69 \\ 0.09 & -0.69 \end{bmatrix} = \begin{bmatrix} 2.8 \\ 0.6 \end{bmatrix}$$

SciFi-concept

movie-to-concept
similarities (\mathbf{V})

Case study: How to query?

- How would the user d that rated ('Alien', 'Serenity') be handled?

$$d_{\text{concept}} = d \mathbf{V}$$

E.g.:

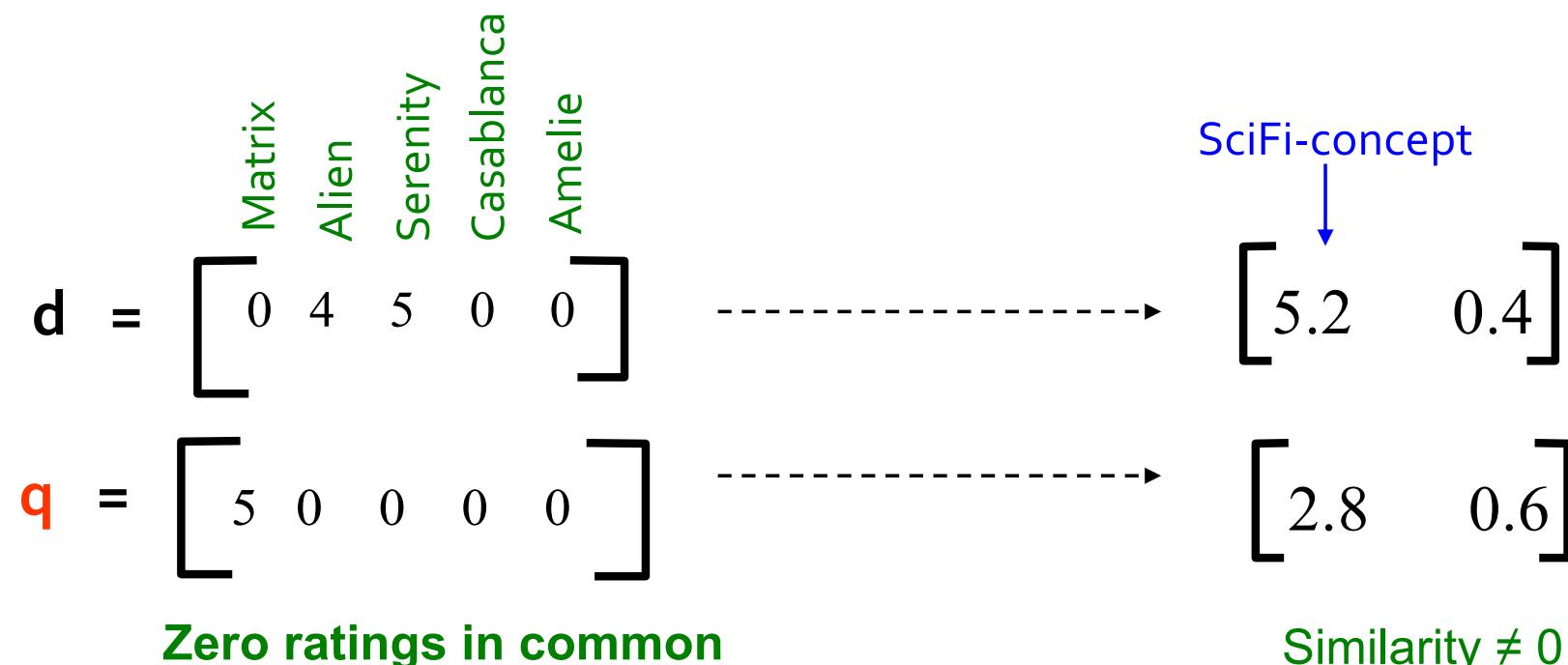
$$\mathbf{q} = \begin{bmatrix} \text{Matrix} \\ \text{Alien} \\ \text{Serenity} \\ \text{Casablanca} \\ \text{Amelie} \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.12 \\ 0.59 & -0.02 \\ 0.56 & 0.12 \\ 0.09 & -0.69 \\ 0.09 & -0.69 \end{bmatrix} = \begin{bmatrix} 5.2 \\ 0.4 \end{bmatrix}$$

SciFi-concept

movie-to-concept
similarities (\mathbf{V})

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!**



SVD: Drawbacks

- + 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!**

$$\begin{matrix} \cdot & \cdot \\ \vdots & \vdots \\ \cdot & \cdot \\ \cdot & \cdot \end{matrix} = \begin{matrix} \Sigma \\ U \end{matrix} \begin{matrix} V^T \\ \cdot \end{matrix}$$

CUR Decomposition

CUR Decomposition

Frobenius norm:
 $\|X\|_F = \sqrt{\sum_{ij} X_{ij}^2}$

- Goal: Express A as a product of matrices C, U, R
Make $\|A - C \cdot U \cdot R\|_F$ small
- “Constraints” on C and R :

$$\left(\begin{array}{c|c|c} \textcolor{red}{|} & \textcolor{blue}{|} & A \\ \hline & & \end{array} \right) \approx \left(\begin{array}{c|c|c|c|c} \textcolor{red}{|} & \textcolor{red}{|} & \textcolor{red}{|} & \textcolor{blue}{|} & \textcolor{darkred}{|} \\ \hline & & & & \end{array} \right) \cdot \left(\begin{array}{c} U \end{array} \right) \cdot \left(\begin{array}{c} R \end{array} \right)$$

A C U R

CUR Decomposition

Frobenius norm:
 $\|X\|_F = \sqrt{\sum_{ij} X_{ij}^2}$

- Goal: Express A as a product of matrices C, U, R
Make $\|A - C \cdot U \cdot R\|_F$ small
- “Constraints” on C and R :

$$\begin{pmatrix} \text{---} \\ \text{---} \\ A \\ \text{---} \end{pmatrix} \approx \begin{pmatrix} \text{---} \\ \text{---} \\ C \\ \text{---} \end{pmatrix} \cdot \begin{pmatrix} \text{---} \\ \text{---} \\ U \\ \text{---} \end{pmatrix} \cdot \begin{pmatrix} \text{---} \\ \text{---} \\ R \\ \text{---} \end{pmatrix}$$

A C U R

Pseudo-inverse of
the intersection of C and R

CUR: How it Works

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

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

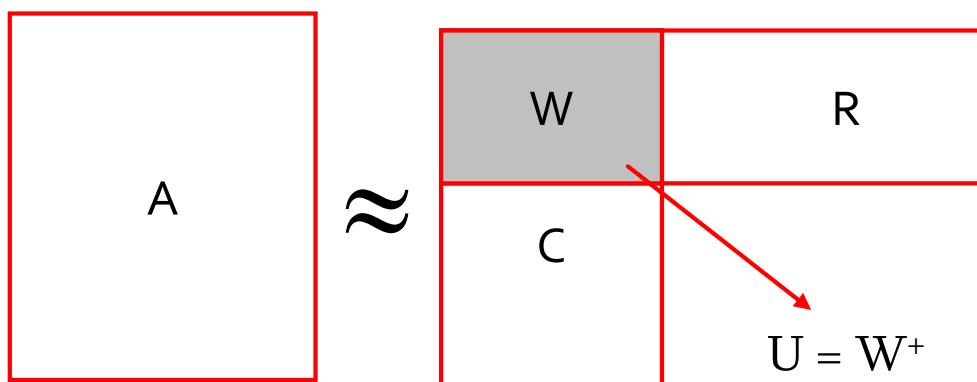
Output: $\mathbf{C}_d \in \mathbb{R}^{m \times c}$

1. for $x = 1 : n$ [column distribution]
2. $P(x) = \sum_i \mathbf{A}(i, x)^2 / \sum_{i,j} \mathbf{A}(i, j)^2$
3. for $i = 1 : c$ [sample columns]
4. Pick $j \in 1 : n$ based on distribution $P(x)$
5. 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

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?
 $W = X Z Y$ then $W^{-1} = X^{-1} Z^{-1} Y^{-1}$
Due to orthonormality
 $X^{-1} = X^T$ 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: Provably good approx. to SVD

- For example:

- Select $c = O\left(\frac{k \log k}{\varepsilon^2}\right)$ columns of A using **ColumnSelect algorithm**
- Select $r = O\left(\frac{k \log k}{\varepsilon^2}\right)$ rows of A using **ColumnSelect algorithm**
- Set $U = W^+$
- Then: $\|A - CUR\|_F \leq (2 + \epsilon) \|A - A_k\|_F$ with probability 98%

In practice:
Pick $4k$ cols/rows
for a “rank- k ” approximation

CUR: Pros & Cons

- + **Easy interpretation**

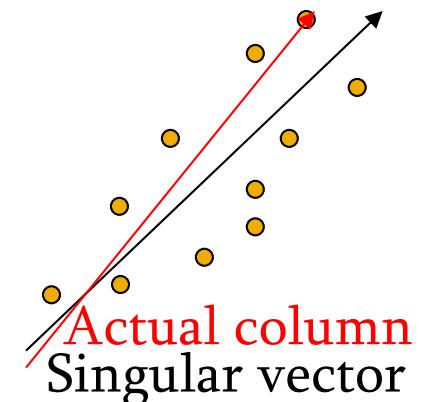
- Since the basis vectors are actual columns and rows

- + **Sparse basis**

- Since the basis vectors are actual columns and rows

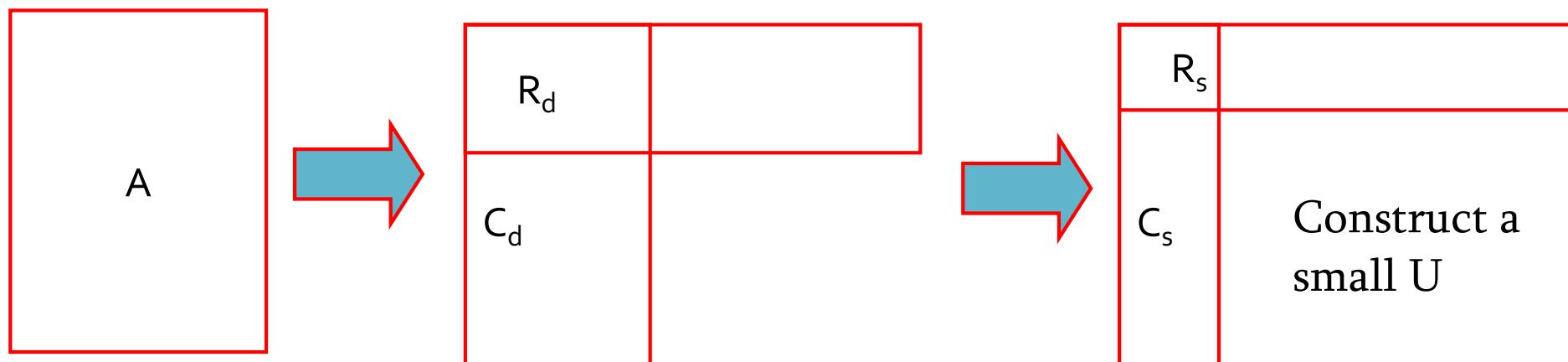
- **Duplicate columns and rows**

- 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

$$\text{SVD: } A = U \Sigma V^T$$

sparse and small
Huge but sparse Big and dense

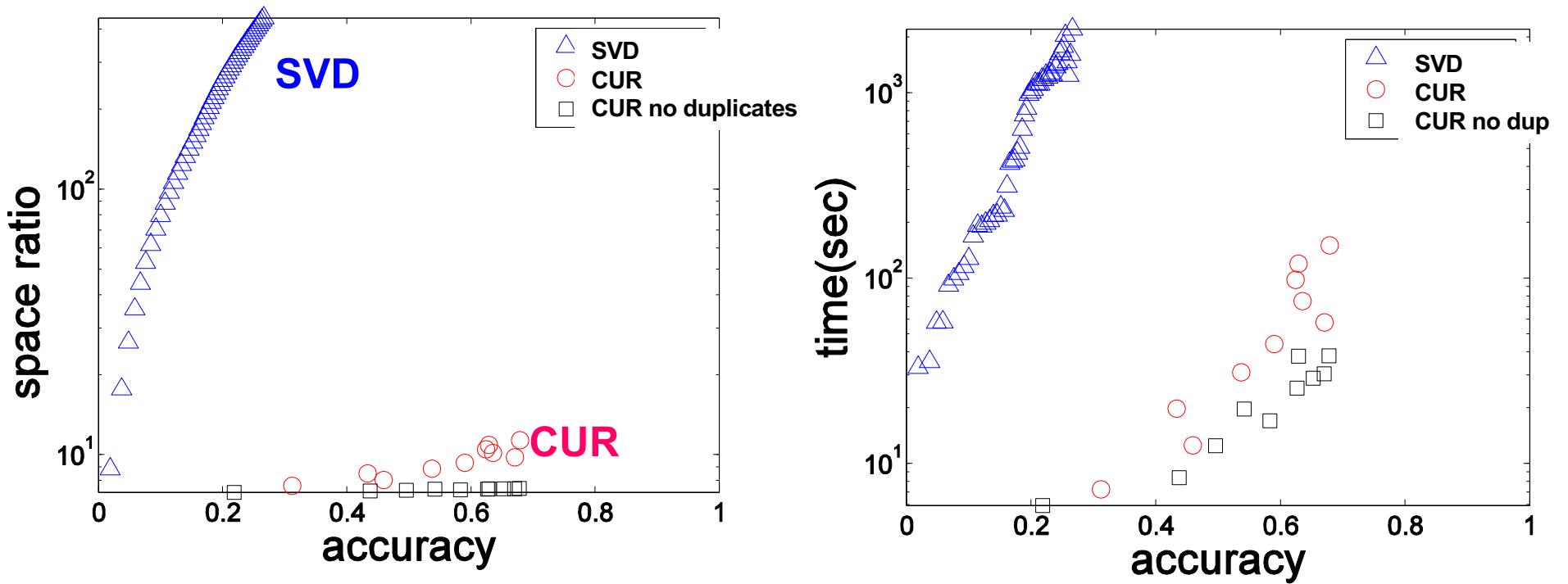
$$\text{CUR: } A = C U R$$

dense but small
Huge but sparse Big but sparse

SVD vs. CUR: 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

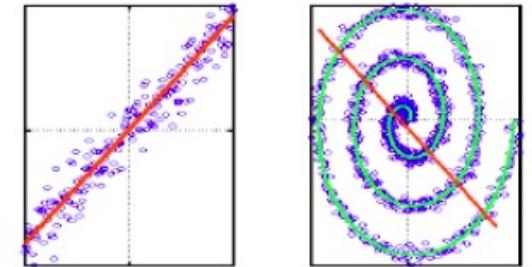


- **Accuracy:**
 - 1 – relative sum squared errors
- **Space ratio:**
 - #output matrix entries / #input matrix entries
- **CPU time**

What about linearity assumption?

- **SVD is limited to linear projections:**

- Lower-dimensional linear projection that preserves Euclidean distances

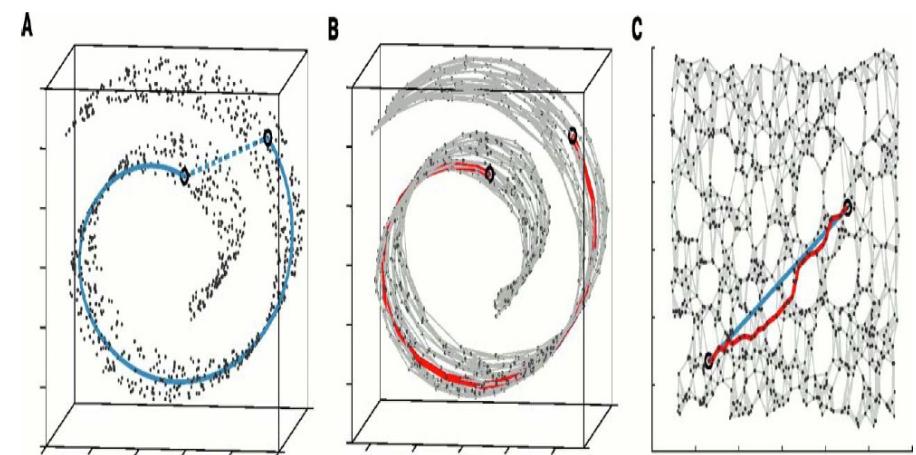


- **Non-linear methods: Isomap**

- 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: CUR

- 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)