



Chapter 7: Dimensionality Reduction

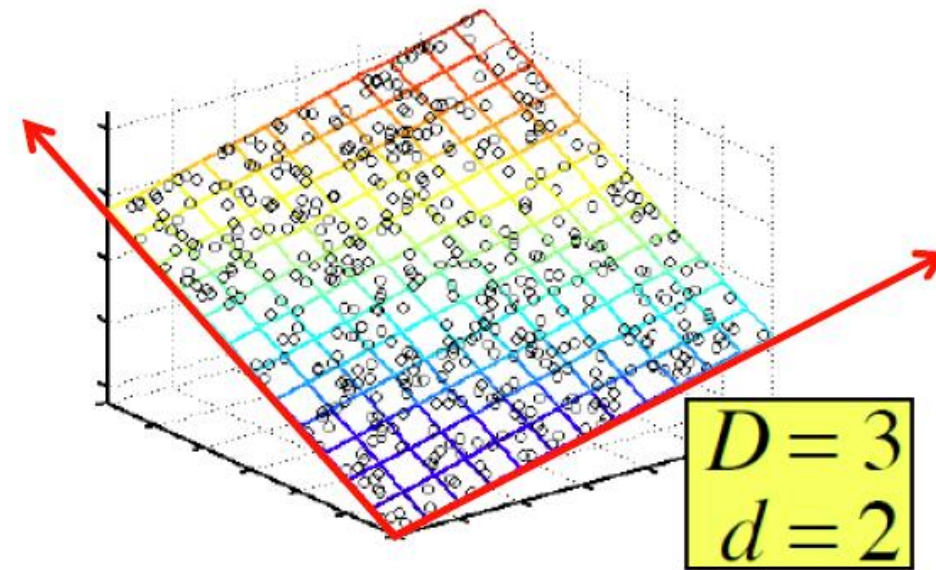
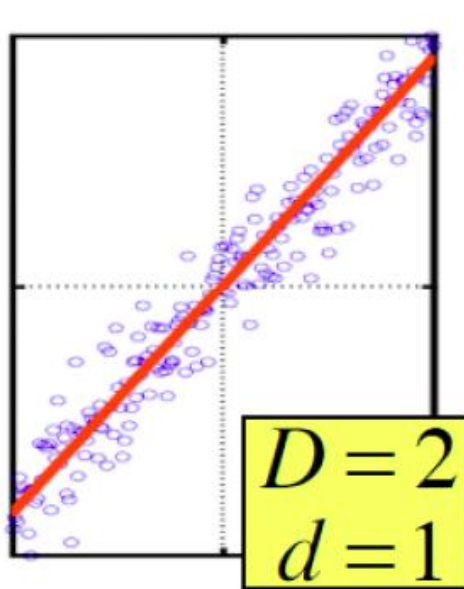
崔金华

电子邮箱: jhcui@hust.edu.cn

个人主页: <https://csjhcui.github.io/>

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
- We can read any cell(s) by reading that targeting cell(s)
- **Some case:** Random access to any cell(s); **small error: OK**

	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	2
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

- User d rated (Alien, Serenity) and user q rated (Matrix). Then, d and q have **zero ratings** in common.
- However, if we can **reduce dimensionality** properly, their similarity will not zero!

	Matrix	Alien	Serenity	Casablanca	Amelie		
d	=	$\begin{bmatrix} 0 & 4 & 5 & 0 & 0 \end{bmatrix}$	----->	$\begin{bmatrix} 5.2 & 0.4 \end{bmatrix}$			
q	=	$\begin{bmatrix} 5 & 0 & 0 & 0 & 0 \end{bmatrix}$	----->	$\begin{bmatrix} 2.8 & 0.6 \end{bmatrix}$			
		Similarity = 0		Similarity \neq 0			

备注:
Matrix黑客帝国
Alien异形
Serenity惊涛迷局
Casablanca卡萨布兰卡
Amelie天使爱美丽

❑ **Q:** What is **rank** (矩阵的秩) of a matrix **A**?

❑ **A:** Number of **linearly independent** columns of **A**

❑ For example: Matrix $\mathbf{A} = \begin{bmatrix} 1 & 2 & 1 \\ -2 & -3 & 1 \\ 3 & 5 & 0 \end{bmatrix}$ 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?**

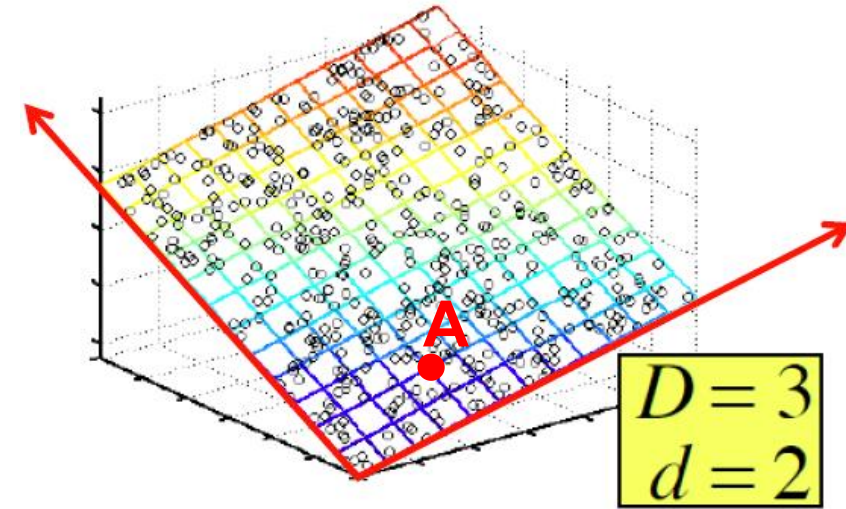
- We can write **A** as two “basis” vectors: $[1 \ 2 \ 1] \ [-2 \ -3 \ 1]$
- And new coordinates of : $[1 \ 0] \ [0 \ 1] \ [1 \ 1]$

Rank is “Dimensionality”

Cloud of points 3D space:

- Think of point positions as a matrix:

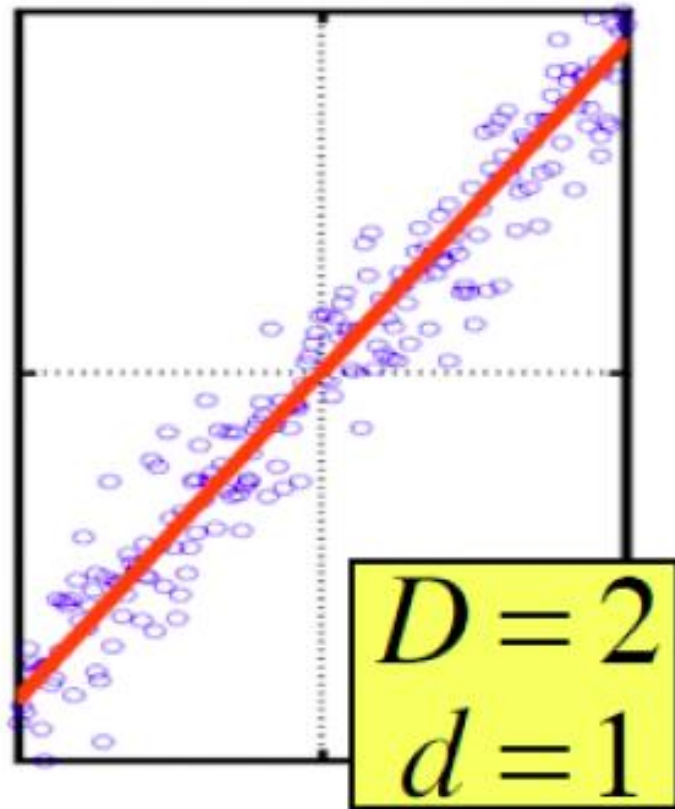
1 row per point:
$$\begin{bmatrix} 1 & 2 & 1 \\ -2 & -3 & 1 \\ 3 & 5 & 0 \end{bmatrix} \begin{matrix} \mathbf{A} \\ \mathbf{B} \\ \mathbf{C} \end{matrix}$$



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!

- 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?

□ 1) Discover hidden correlations/topics

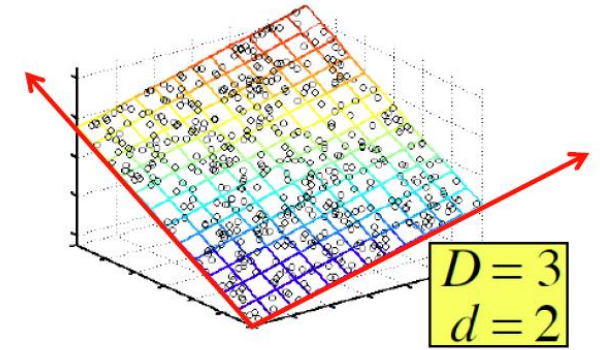
➤ Words that occur commonly together

□ 2) Remove redundant and noisy features

➤ Not all words are useful

□ 3) Interpretation and visualization

□ 4) Easier storage and processing of the data



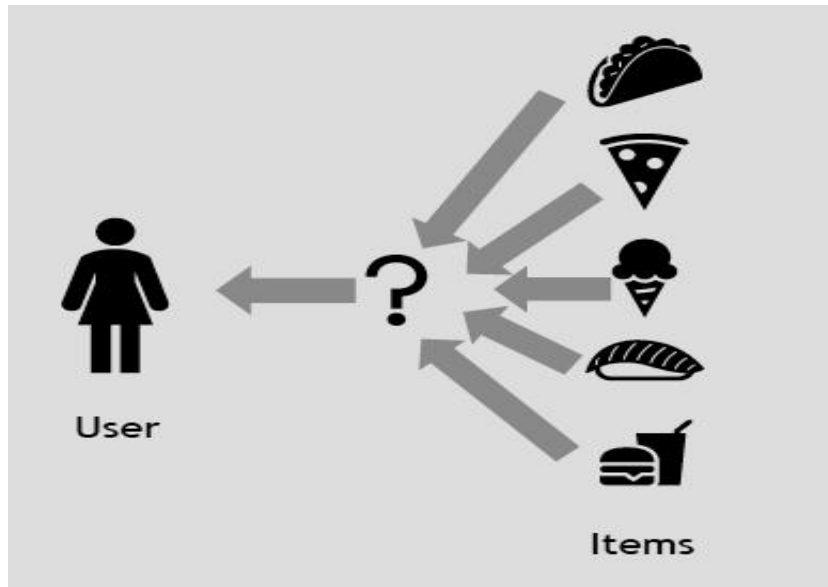
□ **Two dimensionality reduction approaches:**

□ **1) SVD (Singular-Value Decomposition, 奇异值分解)**

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

□ **2) CUR (CUR Decomposition)**

$$A = CUR$$



SVD: Singular-Value Decomposition

Singular-Value Decomposition (奇异值分解 SVD)

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

□ **A: Input data matrix**

➤ $m \times n$ matrix (e.g., m users, n movies)

□ **U: Left singular vectors**

➤ $m \times r$ matrix (m users, 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 movies, r concepts)

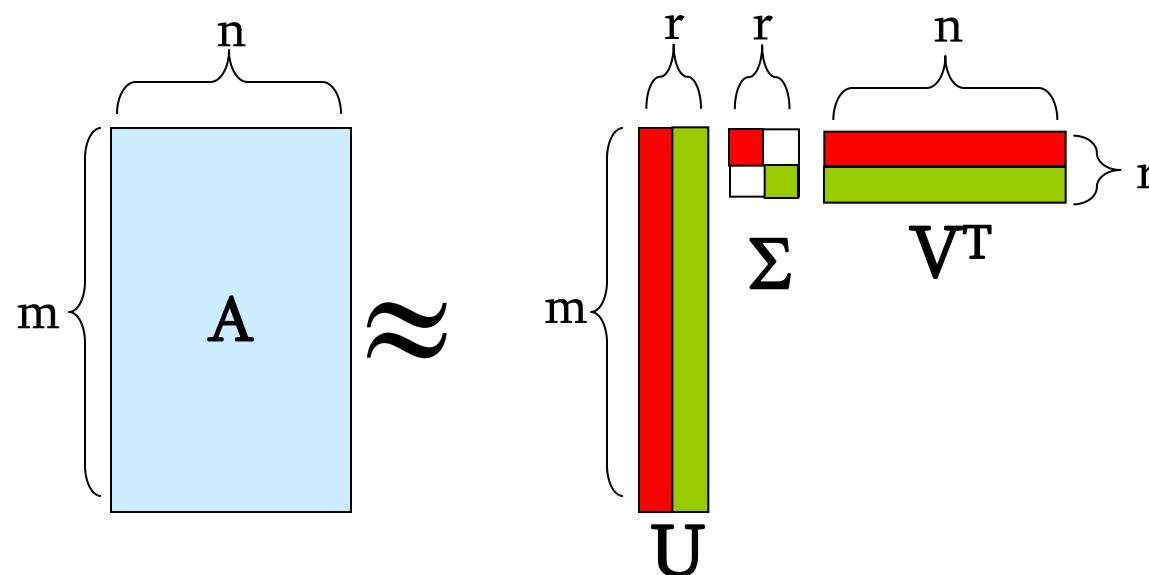
SVD example

□ Example:

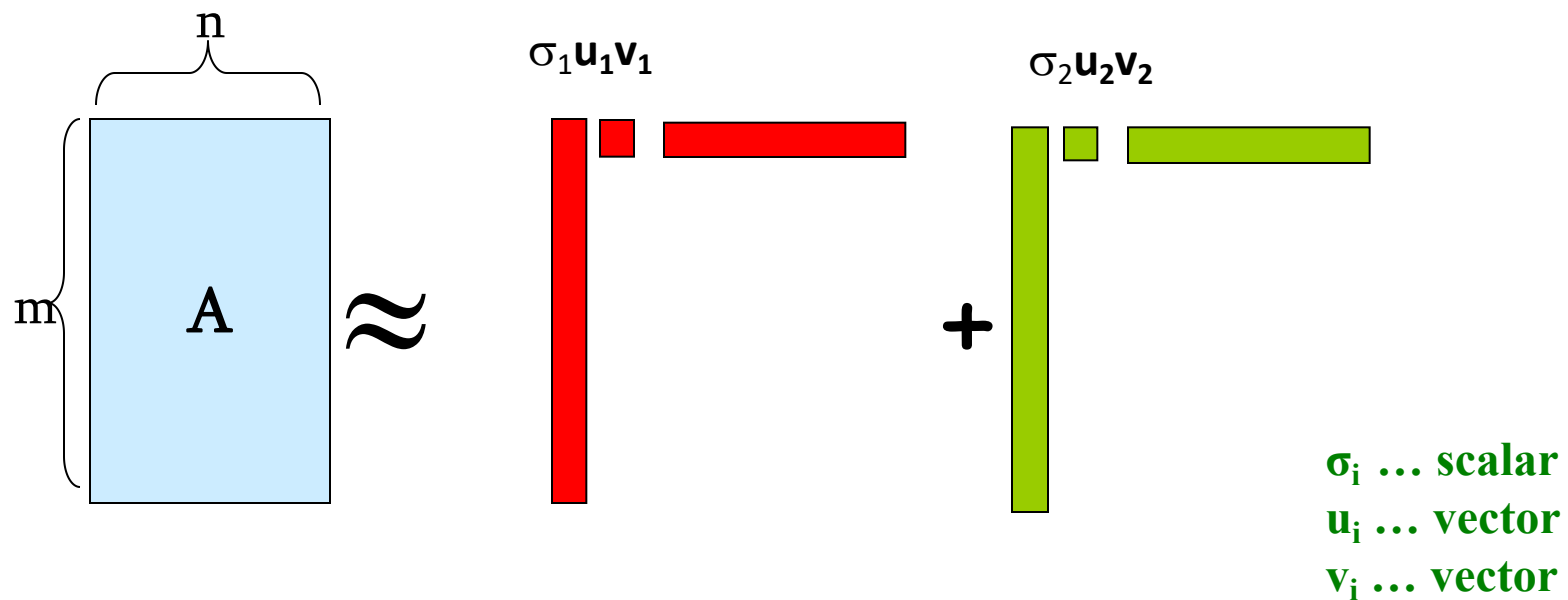
$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 &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

$\mathbf{A} \qquad \mathbf{U} \qquad \mathbf{\Sigma} \qquad \mathbf{V}^T$

$$\mathbf{A} \approx \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$$



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



□ It is **always** possible to decompose a real matrix A into

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

□ U, Σ, V : **unique**

□ U, V : **column orthonormal**(列正交矩阵)

- $U^T U = I; V^T V = I$ (I : identity matrix, 单位矩阵, 主对角线上的值为1, 其他位置的值为0)
- Columns are orthogonal unit vectors (任意一列都是单位向量并且任意两列的内积都是0)

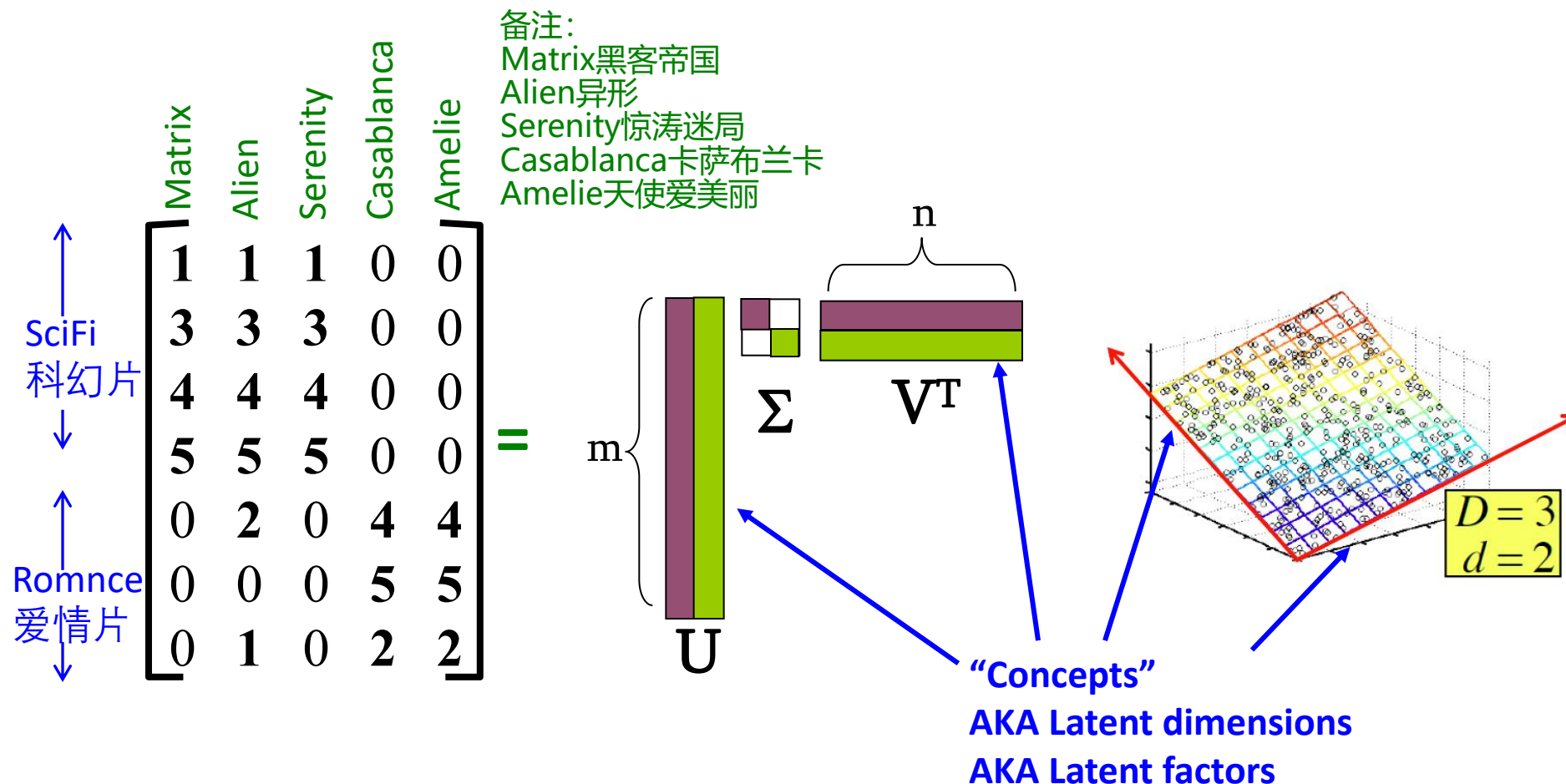
□ Σ : **diagonal**(对角矩阵)

- Entries (**singular values, 奇异值**) are **positive**, and sorted in decreasing order ($\sigma_1 \geq \sigma_2 \geq \dots \geq 0$)

备注: Nice proof of uniqueness: <http://www.mpi-inf.mpg.de/~bast/ir-seminar-ws04/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

Matrix Alien Serenity Casablanca Amelie

SciFi ↑ ↓ ↑ ↓

Romnce ↑ ↓ ↑ ↓

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

SVD – Example: Users-to-Movies

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

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

SVD – Example: Users-to-Movies

□ $A = U \Sigma V^T$ - example:

U is “user-to-concept”
similarity matrix

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

SVD – Example: Users-to-Movies

□ $A = U \Sigma V^T$ - example:

$$\begin{array}{c}
 \begin{array}{c} \uparrow \\ \text{SciFi} \\ \downarrow \\ \uparrow \\ \text{Romnce} \\ \downarrow \end{array}
 \begin{array}{c} \text{Matrix} \\ \text{Alien} \\ \text{Serenity} \\ \text{Casablanca} \\ \text{Amelie} \end{array}
 \begin{bmatrix}
 1 & 1 & 1 & 0 & 0 \\
 3 & 3 & 3 & 0 & 0 \\
 4 & 4 & 4 & 0 & 0 \\
 5 & 5 & 5 & 0 & 0 \\
 0 & 2 & 0 & 4 & 4 \\
 0 & 0 & 0 & 5 & 5 \\
 0 & 1 & 0 & 2 & 2
 \end{bmatrix}
 =
 \begin{array}{c} \text{SciFi-concept} \\ \downarrow \end{array}
 \begin{bmatrix}
 0.13 & 0.02 & -0.01 \\
 0.41 & 0.07 & -0.03 \\
 0.55 & 0.09 & -0.04 \\
 0.68 & 0.11 & -0.05 \\
 0.15 & -0.59 & 0.65 \\
 0.07 & -0.73 & -0.67 \\
 0.07 & -0.29 & 0.32
 \end{bmatrix}
 \times
 \begin{array}{c} \text{"strength" of the SciFi-concept} \\ \downarrow \end{array}
 \begin{bmatrix}
 12.4 & 0 & 0 \\
 0 & 9.5 & 0 \\
 0 & 0 & 1.3
 \end{bmatrix}
 \times
 \begin{bmatrix}
 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\
 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\
 0.40 & -0.80 & 0.40 & 0.09 & 0.09
 \end{bmatrix}
 \end{array}$$

SVD – Example: Users-to-Movies

□ $A = U \Sigma V^T$ - example:

SciFi

Romnce

Matrix

Alien

Serenity

Casablanca

Amelie

SciFi-concept

V is “movie-to-concept” similarity matrix

SciFi-concept

\times

\times

0.56

SciFi-concept

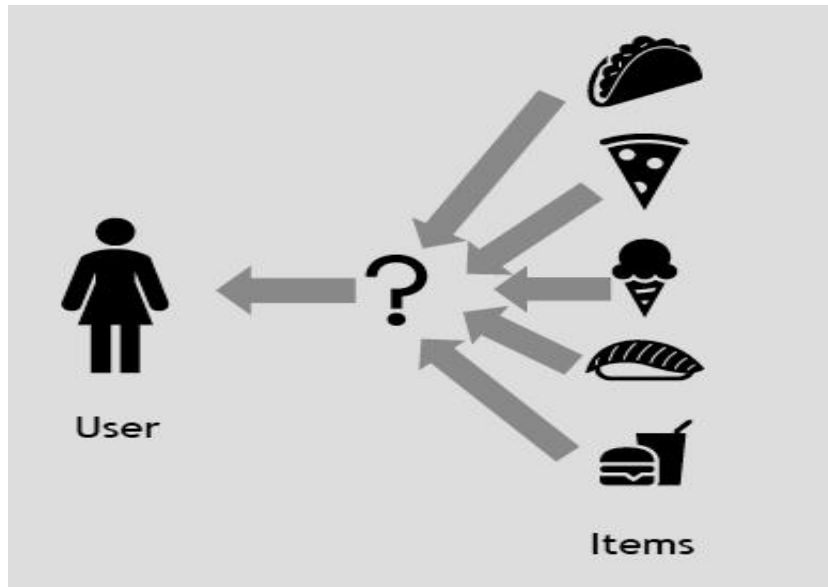
$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 &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

备注:
Matrix黑客帝国
Alien异形
Serenity惊涛迷局
Casablanca卡萨布兰卡
Amelie天使爱美丽

SVD - Interpretation #1

'movies' , 'users' and 'concepts' :

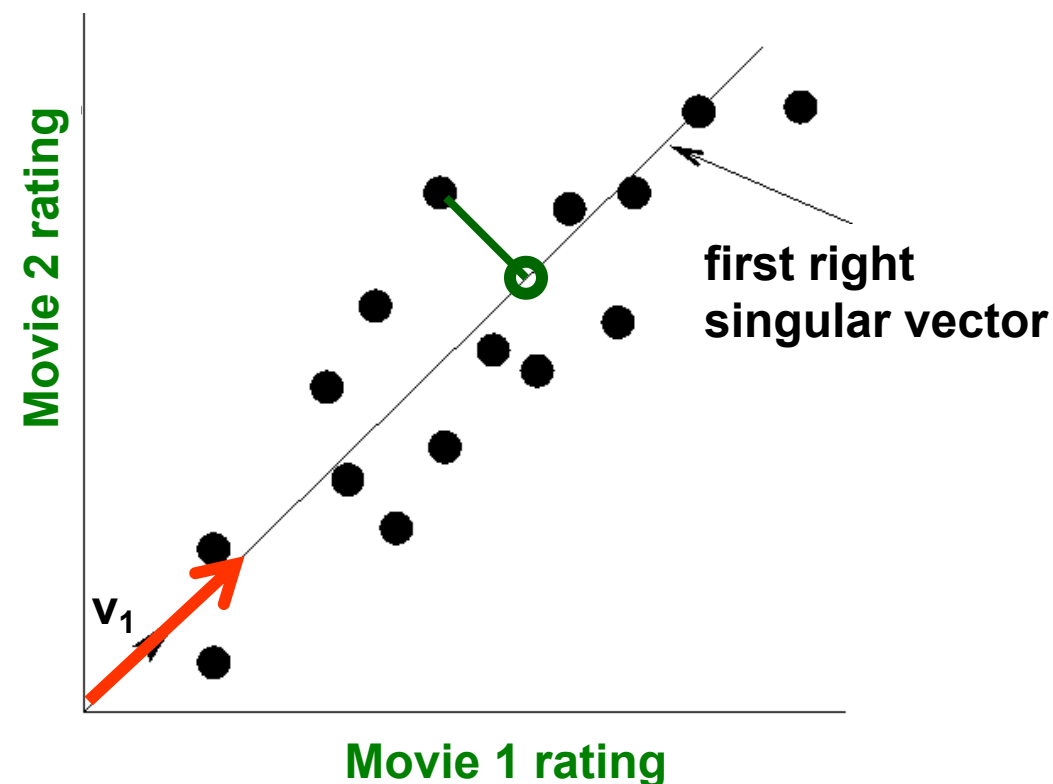
- U : user-to-concept similarity matrix
- V : movie-to-concept similarity matrix
- Σ : its diagonal elements: 'strength' of each concept



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
- **Q: How to choose v_1 ?**
- **A: Minimize reconstruction error**

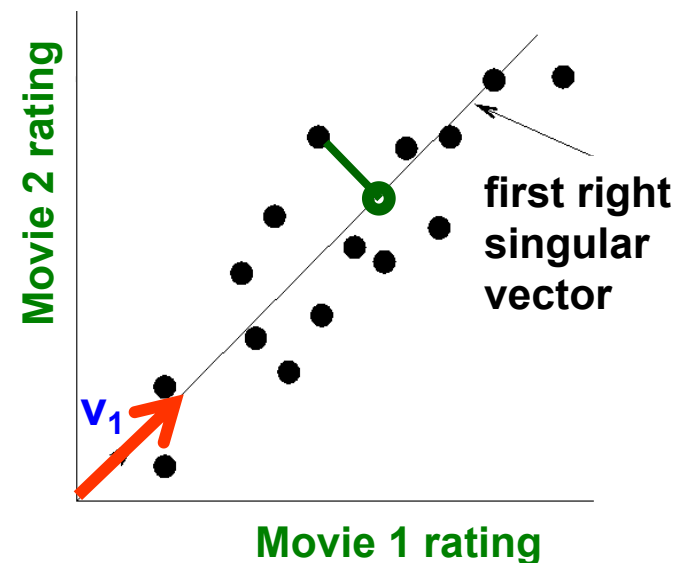


SVD – Dimensionality Reduction

□ **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

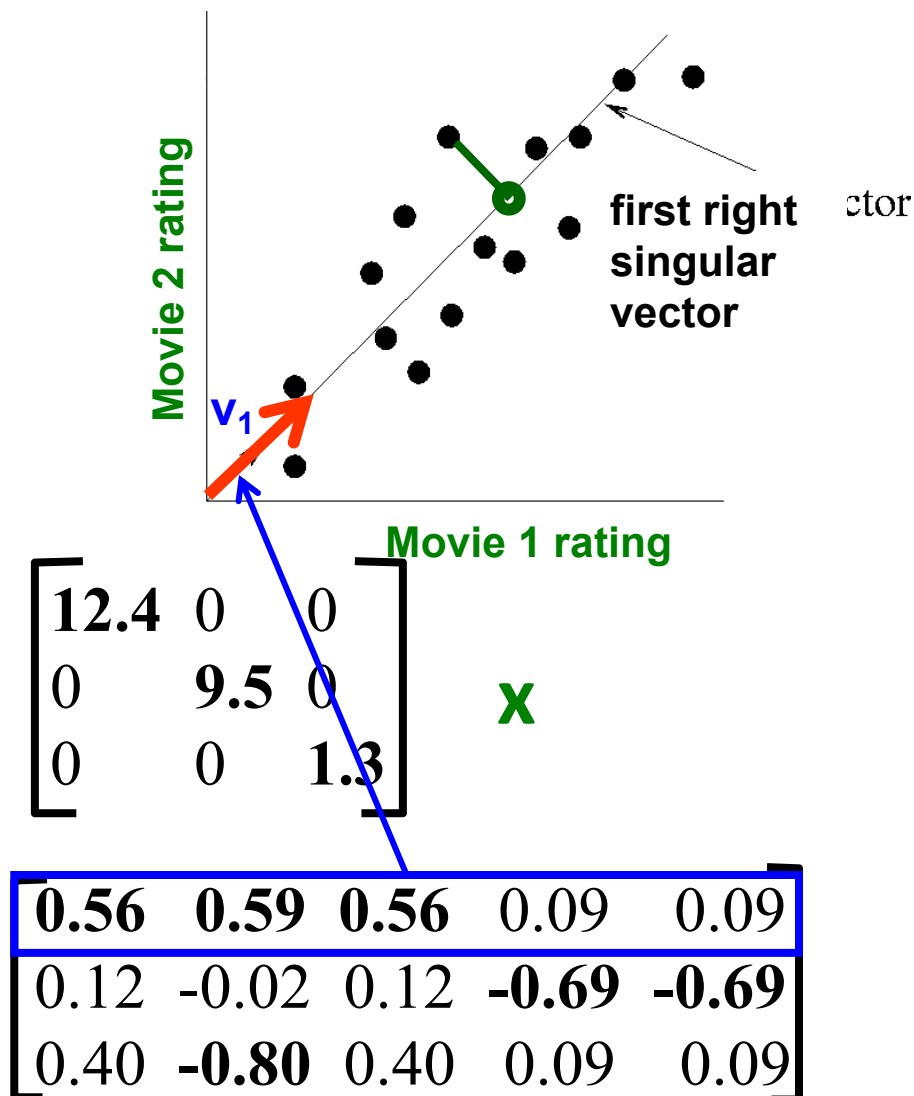
□ **In other words, minimum reconstruction error**

SVD - Interpretation #2

□ $A = U \Sigma V^T$ - example:

- V : "movie-to-concept" matrix
- U : "user-to-concept" matrix

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



SVD - Interpretation #2

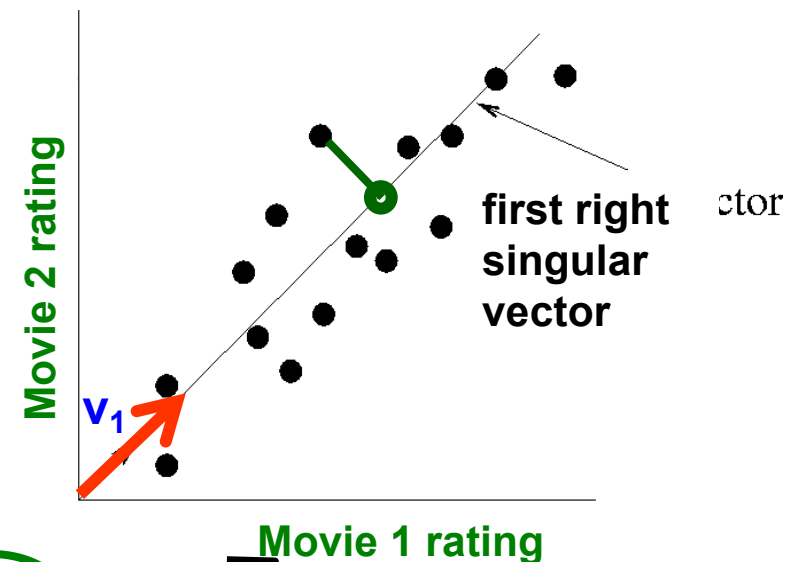
□ $A = U \Sigma V^T$ - example:

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

$$\begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix} \times$$

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

variance ('spread')
on the v_1 axis



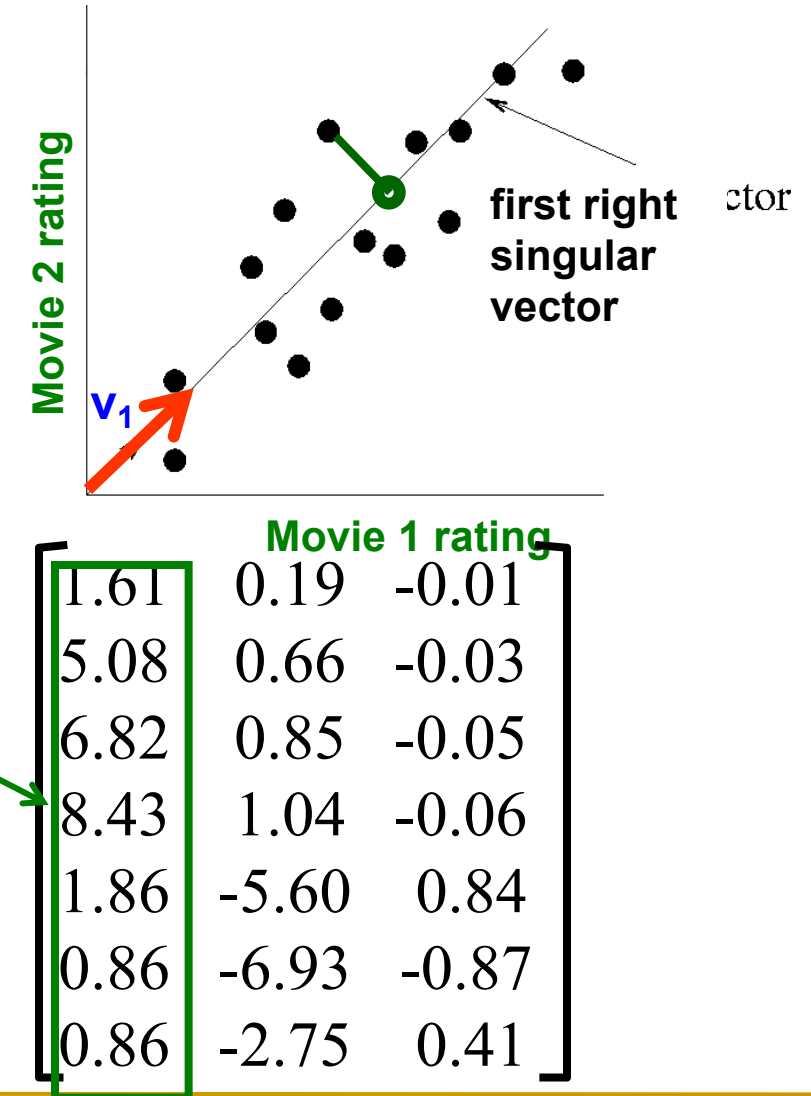
SVD - Interpretation #2

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



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

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

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

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

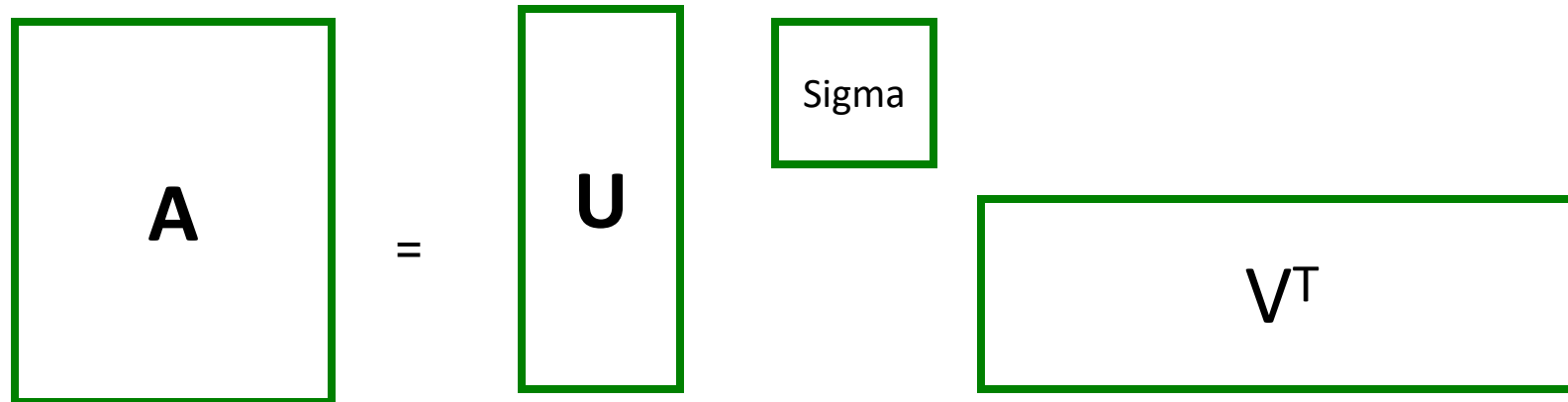
Frobenius norm:

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

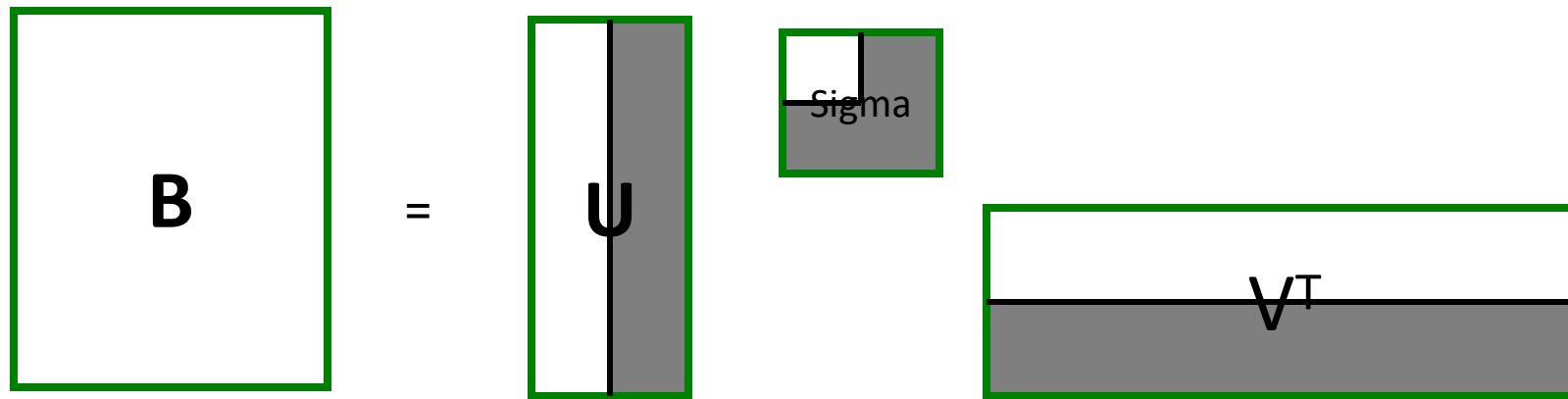
$$\|A-B\|_F = \sqrt{\sum_{ij} (A_{ij}-B_{ij})^2}$$

is "small"

SVD – Best Low Rank Approx.



B is best approximation of A



SVD – Best Low Rank Approx.

□ Theorem:

□ Let $\mathbf{A} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T$ and $\mathbf{B} = \mathbf{U} \mathbf{S} \mathbf{V}^T$ where $\mathbf{S} = \text{diagonal } r \times r \text{ matrix}$ with $s_i = \sigma_i$ ($i=1 \dots k$) else $s_i = 0$ then \mathbf{B} is a **best rank(B)=k approx. to A**

What do we mean by “best” :

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

$$\begin{pmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & \\ \vdots & \vdots & \ddots & \\ x_{m1} & & & x_{mn} \end{pmatrix}_{m \times n} = \begin{pmatrix} u_{11} & \dots & & \\ \vdots & \ddots & & \\ u_{m1} & & & \end{pmatrix}_{m \times r} \begin{pmatrix} \sigma_{11} & 0 & \dots \\ 0 & \ddots & \\ \vdots & & \sigma_r \end{pmatrix}_{r \times r} \begin{pmatrix} v_{11} & \dots & v_{1n} \\ \vdots & \ddots & \\ \vdots & & \end{pmatrix}_{r \times n}$$

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

SVD - Interpretation #2

□ **Equivalent:** 'spectral decomposition' (谱分解, 又称特征分解) of the matrix

$$Av = \lambda v$$

$$\text{特征分解: } A = X \Lambda X^{-1}$$

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

SVD - Interpretation #2

□ **Equivalent:** 'spectral decomposition' (谱分解, 又称特征分解) of the matrix

$$\begin{array}{c} \xleftarrow{m} \xrightarrow{\hspace{1cm}} \\ \uparrow \hspace{0.5cm} \downarrow \hspace{0.5cm} n \\ \left[\begin{array}{ccccc} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{array} \right] \end{array} = \begin{array}{c} \xleftarrow{\hspace{1cm}} \xrightarrow{k \text{ terms}} \\ \sigma_1 \quad \begin{array}{c} u_1 \\ \swarrow \quad \searrow \\ n \times 1 \quad 1 \times m \end{array} \quad v_1^T + \sigma_2 \quad u_2 \quad v_2^T + \dots \\ \text{Assume: } \sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \dots \geq 0 \end{array}$$

Why is setting small σ_i to 0 the right thing to do?
Vectors u_i and 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' $= \sum_i \sigma_i^2$

$$\begin{array}{c} \updownarrow \\ n \end{array} \begin{array}{c} \leftarrow m \rightarrow \\ \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} \end{array} = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^T + \sigma_2 \mathbf{u}_2 \mathbf{v}_2^T + \dots$$

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

□ 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 ...

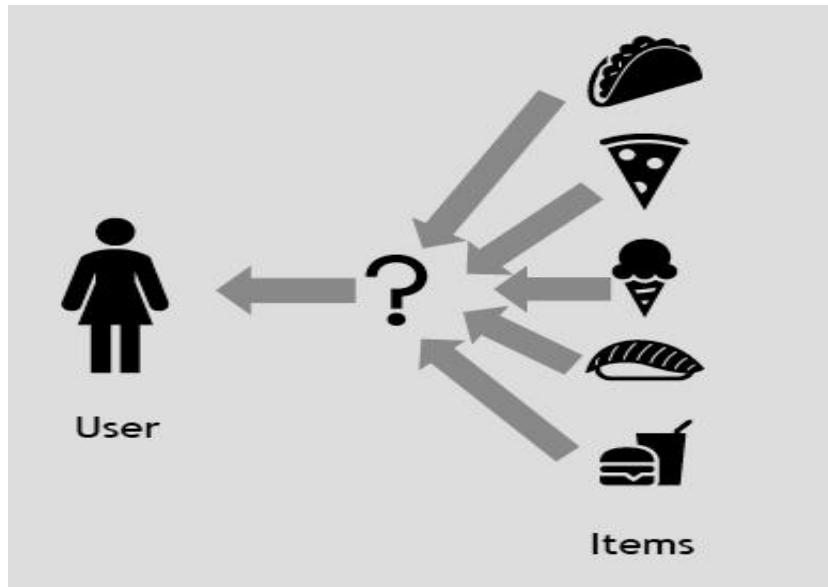
SVD - Conclusions so far

□ SVD: $\mathbf{A} = \mathbf{U} \Sigma \mathbf{V}^T$: **unique**

- \mathbf{U} : user-to-concept similarities
- \mathbf{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



How to compute SVD?

□ 定义: 令 M 为方阵, λ 是一个常数, e 是一个维度等于 M 行数的非零列向量. 那么如果

$$Me = \lambda e$$

λ 则称是方阵 M 的**特征值(eigenvalue)**, e 为方阵 M 的对应的特征值 λ 的**特征向量(eigenvector)**.

□ 例: 设 $A = \begin{bmatrix} 3 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 3 \end{bmatrix}$, 由于 $A = 3I$ (I 表示单位矩阵, 这儿是 3×3 大小, 其主对角线的值为1, 其他位置的值均为0), 那么对于 e 都有 $Ae = 3Ie = 3e$ 成立. 所以3是 A 的特征值, 任何一个非零的三维向量 e 都是 A 的特征向量.

□ 为避免向量大小造成的歧义, 我们进一步要求每个**特征向量**都是**单位向量**(**单位向量中元素的平方和为1**). 当然即使如此也无法完全保证特征向量的唯一性. 此外, 我们通常还要求特征向量的**第一个非零元素为正数**.

□ 例: 设 $M = \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix}$, 该矩阵存在一个特征向量 $e = \begin{bmatrix} 1/\sqrt{5} \\ 2/\sqrt{5} \end{bmatrix}$, 其对应的特征值为7. 因

为 $\begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \begin{bmatrix} 1/\sqrt{5} \\ 2/\sqrt{5} \end{bmatrix} = 7 \begin{bmatrix} 1/\sqrt{5} \\ 2/\sqrt{5} \end{bmatrix}$. 上式两边都等于 $\begin{bmatrix} 7/\sqrt{5} \\ 14/\sqrt{5} \end{bmatrix}$. 并且我们也能发现特征向量 e 是单位向量, 因为 $(1/\sqrt{5})^2 + (2/\sqrt{5})^2 = 1/5 + 4/5 = 1$

□ 定义: **特征对(eigenpair)** 表示特征值及其对应特征向量组成的对.

□ 将先前定义 $Me = \lambda e$ 重写为

$$(M - \lambda I)e = 0$$

- I 是 $n \times n$ 的单位矩阵(主对角线的值为1, 其他位置的值为0)
- 0 是一个所有值为0的向量.

□ 线性代数的一个客观事实是, 要使 $(M - \lambda I)e = 0$ 对于某个非零向量 e 成立, 那么 $|M - \lambda I|$ ($M - \lambda I$ 的**行列式**, determinant) 必须为0.

□ $n \times n$ 的矩阵的行列式有 $n!$ 项求和值, 所以有多种 $O(n^3)$ 时间复杂度的算法能求解矩阵的行列式, 例如中枢压缩法.

□ 这儿列举一些常见的结论:

➤ $A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$, 那么 $|A|(\text{行列式}) = ad - bc$

➤ $A = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$, 那么 $|A| = a \begin{vmatrix} e & f \\ h & i \end{vmatrix} - b \begin{vmatrix} d & f \\ g & i \end{vmatrix} + c \begin{vmatrix} d & e \\ g & h \end{vmatrix}$

➤

□ 一种求解特征对的方法过程:

□ 如果 $(M - \lambda I)e = 0$, 那么 $|M - \lambda I| = 0$

- 1、 $|M - \lambda I|$ 是 λ 的 n 阶多项式. 通过该值等于 0 就能够求解得到 n 个 λ , 也就是矩阵 M 的特征值(尽管有些情况下某些特征值相等).
- 2、对于每个特征值 c , 求解方程 $Me = ce$. 求解结果后, 我们通过将所有元素的平方和为 1 进行归一化, 从而可以求解得到特征值 c 对应的特征向量.
- 3、循环处理, 找到所有特征值对应的特征向量, 从而求解出所有的特征对.

□例: 求解 $M = \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix}$ 的特征对.

□解:

- $M - \lambda I = \begin{bmatrix} 3 - \lambda & 2 \\ 2 & 6 - \lambda \end{bmatrix}$, 那么 $|M - \lambda I| = (3 - \lambda)(6 - \lambda) - 4 = 0$
- 该等式可以得到方程 $\lambda^2 - 9\lambda + 14 = 0$, 求解方程可得 $\lambda = 7$ 和 $\lambda = 2$.
- 其中 $\lambda = 7$ 的值更大, 所以该值是主特征值(principal eigenvalue, 最大的特征值). 令 e 为未知分量 x 和 y 构成的向量 $\begin{bmatrix} x \\ y \end{bmatrix}$, 那么满足 $Me = ce$. 也就是 $\begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 7 \begin{bmatrix} x \\ y \end{bmatrix}$, 求解可得方程组 $3x + 2y = 7x$ 和 $2x + 6y = 7y$. 它们都表达同一个意思, 即 $y = 2x$.
- 那么一个可能的特征向量为 $\begin{bmatrix} 1 \\ 2 \end{bmatrix}$. 但是它不是单位向量, 于是我们对每个分量都除以 $\sqrt{1^2 + 2^2} = \sqrt{5}$. 所以主特征值 $\lambda = 7$ 对应的主特征向量(principal eigenvector, 最大特征值对应的特征向量)为 $\begin{bmatrix} 1/\sqrt{5} \\ 2/\sqrt{5} \end{bmatrix}$

□解(续):

- 针对特征值 $\lambda=2$. 令 e 为未知分量 x 和 y 构成的向量 $\begin{bmatrix} x \\ y \end{bmatrix}$, 那么满足 $Me = ce$. 也就是 $\begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 2 \begin{bmatrix} x \\ y \end{bmatrix}$, 求解可得方程组 $3x+2y=2x$ 和 $2x+6y=2y$. 它们都表达同一个意思, 即 $x = -2y$.
- 那么一个可能的特征向量为 $\begin{bmatrix} 2 \\ -1 \end{bmatrix}$ (注意这儿不是 $\begin{bmatrix} -2 \\ 1 \end{bmatrix}$, 因为我们前面提及特征向量第一个非零元素为正数). 但是它不是单位向量, 于是我们对每个分量都除以 $\sqrt{2^2 + (-1)^2} = \sqrt{5}$. 所以特征值 $\lambda=2$ 对应的特征向量为 $\begin{bmatrix} 2/\sqrt{5} \\ -1/\sqrt{5} \end{bmatrix}$

□ 定义: $n \times n$ 的矩阵 M , 其特征向量 e_1, e_2, \dots, e_n 看成一系列列向量. **特征向量矩阵** E (matrix of eigenvectors) 是第 i 个列向量为 e_i 的矩阵. 且 $EE^T = E^T E = I$.

□ 例: $M = \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix}$, 对应的特征向量矩阵 E 为

$$\begin{bmatrix} 1/\sqrt{5} & 2/\sqrt{5} \\ 2/\sqrt{5} & -1/\sqrt{5} \end{bmatrix}$$

➤ $E^T = \begin{bmatrix} 1/\sqrt{5} & 2/\sqrt{5} \\ 2/\sqrt{5} & -1/\sqrt{5} \end{bmatrix}$. 那么 $EE^T = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

➤ 类似地, $E^T E = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. 这是因为结果的主对角线上的1刚好是每个特征向量元素的平方和.

□ SVD gives us:

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

□ Eigen-decomposition(特征分解):

$$\triangleright A = X \Lambda X^T \quad Ae = \lambda e$$

- A is symmetric; U, V, X are orthonormal (e.g., $U^T U = I$); Λ, Σ are diagonal
- \diamond is matrix of eigenvalue (特征值) of A, \diamond is the corresponding eigenvector (特征向量); X is matrix of eigenvectors(特征向量组成的矩阵) of A. Λ entries are the corresponding eigenvalues(特征值)

□ Now let's calculate:

$$\triangleright AA^T = U \Sigma V^T (U \Sigma V^T)^T$$

$$\triangleright A^T A = V \Sigma^T U^T (U \Sigma V^T)$$

Relation to Eigen-decomposition

□ SVD gives us:

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

□ Eigen-decomposition(特征分解):

$$\triangleright A = X \Lambda X^T \quad Ae = \lambda e$$

- A is symmetric; U, V, X are orthonormal (e.g., $U^T U = I$); Λ, Σ are diagonal
- \diamond is matrix of eigenvalue (特征值) of A, \diamond is the corresponding eigenvector (特征向量); X is matrix of eigenvectors(特征向量组成的矩阵) of A. Λ entries are the corresponding eigenvalues(特征值)

□ Now let's calculate:

$$\triangleright AA^T = U \Sigma V^T (U \Sigma V^T)^T = U \Sigma V^T (V \Sigma^T U^T) = U \Sigma \Sigma^T U^T$$

$$\triangleright A^T A = V \Sigma^T U^T (U \Sigma V^T) = \underset{\substack{\uparrow \\ X}}{V} \underset{\substack{\uparrow \\ \Lambda}}{\Sigma \Sigma^T} \underset{\substack{\uparrow \\ X^T}}{V^T}$$

$$\underset{\downarrow}{X} \underset{\downarrow}{\Lambda} \underset{\downarrow}{X^T}$$

Shows how to compute
SVD using eigenvalue
decomposition!

Computing the SVD of the matrix A

□ **Goal:** $A = U\Sigma V^T$

□ **How to compute U, V, Σ ?**

□ **Step:** Computing the eigenpairs (特征对, 特征值和对应特征向量组成的对) for $A^T A$, then we get matrix V, Σ

➤ $A^T A V = V \Sigma^2$, V 特征向量矩阵, Σ 特征值开平方

□ **Step:** Computing the eigenpairs for $A A^T$, then we get matrix U

➤ $A A^T = U \Sigma^2 U^T$, then $A A^T U = U \Sigma^2 U^T U \longrightarrow A A^T U = U \Sigma^2$

➤ so U is the matrix of eigenvectors (特征向量组成的矩阵) of $A A^T$

$$\begin{array}{c} X \Lambda X^T \\ \downarrow \quad \downarrow \quad \downarrow \\ A A^T = U \Sigma \Sigma^T U^T \\ A^T A = V \Sigma \Sigma^T V^T \\ \uparrow \quad \uparrow \quad \uparrow \\ X \Lambda X^T \end{array}$$

□ **Example:** Find the matrices U , Σ , V for $A = \begin{bmatrix} 3 & 0 \\ 4 & 5 \end{bmatrix}$, the rank is $r=2$.

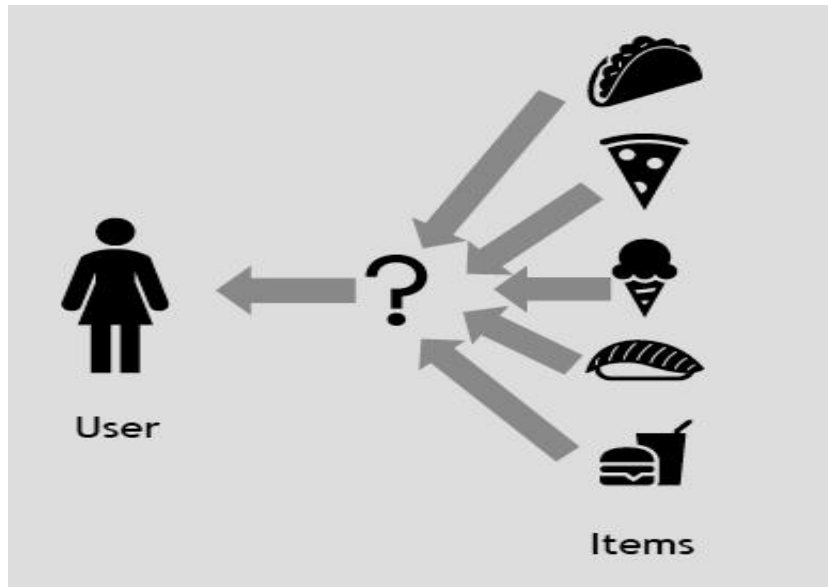
□ **Solution:**

- Step 1: $A^T A = \begin{bmatrix} 25 & 20 \\ 20 & 25 \end{bmatrix}$, so $|A^T A - \lambda I| = (25 - \lambda)(25 - \lambda) - 20 \cdot 20 = 0$. $\lambda = 45$ and $\lambda = 5$.
- for $\lambda = 45$, $\begin{bmatrix} 25 & 20 \\ 20 & 25 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 45 \begin{bmatrix} x \\ y \end{bmatrix}$. We get eigenvector is $\begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix}$
- For $\lambda = 5$, $\begin{bmatrix} 25 & 20 \\ 20 & 25 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 5 \begin{bmatrix} x \\ y \end{bmatrix}$. We get eigenvector is $\begin{bmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{bmatrix}$.

Example of SVD

□ Solution(续):

- Then $V = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}$, $\Sigma = \begin{bmatrix} \sqrt{45} & 0 \\ 0 & \sqrt{5} \end{bmatrix}$
- Step 2: $AA^T = \begin{bmatrix} 9 & 12 \\ 12 & 41 \end{bmatrix}$, so $|AA^T - \lambda I| = (9 - \lambda)(41 - \lambda) - 12 \cdot 12 = 0$. $\lambda = 45$ and $\lambda = 5$.
- for $\lambda = 45$, $\begin{bmatrix} 9 & 12 \\ 12 & 41 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 45 \begin{bmatrix} x \\ y \end{bmatrix}$. We get eigenvector is $\begin{bmatrix} 1/\sqrt{10} \\ 3/\sqrt{10} \end{bmatrix}$
- for $\lambda = 5$, $\begin{bmatrix} 9 & 12 \\ 12 & 41 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 5 \begin{bmatrix} x \\ y \end{bmatrix}$. We get eigenvector is $\begin{bmatrix} 3/\sqrt{10} \\ -1/\sqrt{10} \end{bmatrix}$
- So, $U = \begin{bmatrix} 1/\sqrt{10} & 3/\sqrt{10} \\ 3/\sqrt{10} & -1/\sqrt{10} \end{bmatrix}$
- In summary, $A = U \Sigma V^T$, $U = \begin{bmatrix} 1/\sqrt{10} & 3/\sqrt{10} \\ 3/\sqrt{10} & -1/\sqrt{10} \end{bmatrix}$, $\Sigma = \begin{bmatrix} \sqrt{45} & 0 \\ 0 & \sqrt{5} \end{bmatrix}$, $V = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}$



SVD Applications

Case study: How to query?

❑ Q: Find users that like 'Matrix' (黑客帝国)

❑ A: Map query into a 'concept space' – how?

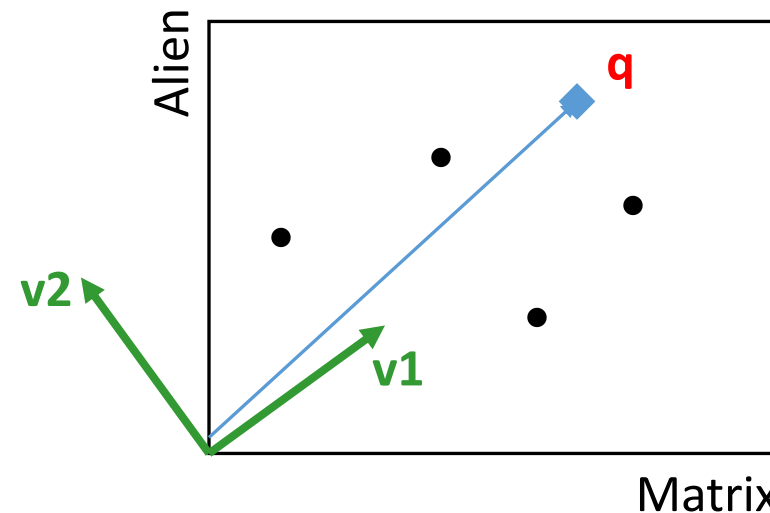
$$\begin{array}{c} \uparrow \text{SciFi} \\ \downarrow \text{Romnce} \end{array} \begin{array}{c} \text{Matrix} \\ \text{Alien} \\ \text{Serenity} \\ \text{Casablanca} \\ \text{Amelie} \end{array} \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 &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

Case study: How to query?

- ❑ Q: Find users that like 'Matrix' (黑客帝国)
- ❑ A: Map query into a 'concept space' – how?

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

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

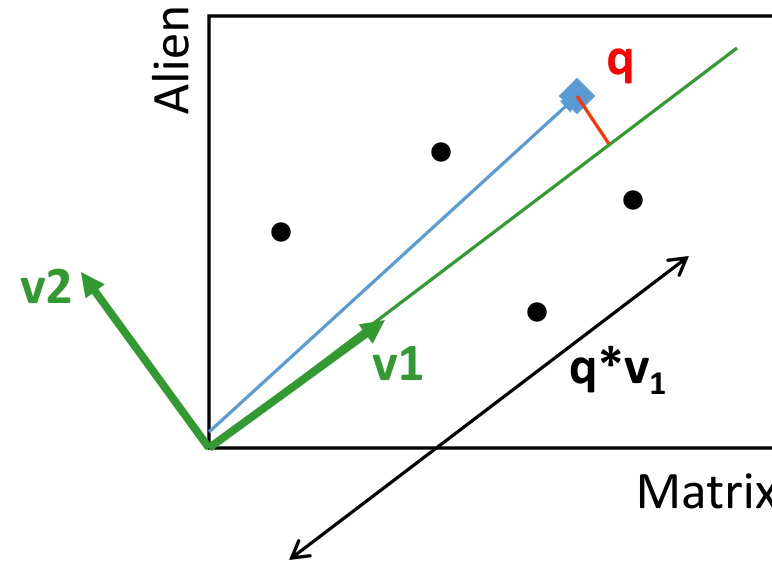


Case study: How to query?

- ❑ Q: Find users that like 'Matrix' (黑客帝国)
- ❑ A: Map query into a 'concept space' – how?

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

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



Case study: How to query?

Compactly, we have:

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

E.g.:

$$\mathbf{q} = \begin{matrix} & \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\ \begin{bmatrix} 5 & 0 & 0 & 0 & 0 \end{bmatrix} & \times & \begin{bmatrix} 0.56 & 0.12 \\ 0.59 & -0.02 \\ 0.56 & 0.12 \\ 0.09 & -0.69 \\ 0.09 & -0.69 \end{bmatrix} & = & \begin{bmatrix} 2.8 & 0.6 \end{bmatrix} \end{matrix}$$

movie-to-concept similarities (V)

SciFi-concept

↓

$$= \begin{bmatrix} 2.8 & 0.6 \end{bmatrix}$$

$\mathbf{q}_{\text{concept}} \mathbf{V}^T$ we get \mathbf{q} rate for
Alien, Serenity...

Case study: How to query?

□ Another sort of query we can perform in concept space is to find **users similar to user q** .

□ **How would user d that rated ('Alien' , 'Serenity') be handled?** $\mathbf{d}_{\text{concept}} = \mathbf{d} \mathbf{V}$

E.g.:

$$\mathbf{d} = \begin{matrix} & \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\ \begin{bmatrix} 0 & 4 & 5 & 0 & 0 \end{bmatrix} & \times & \begin{bmatrix} 0.56 & 0.12 \\ 0.59 & -0.02 \\ 0.56 & 0.12 \\ 0.09 & -0.69 \\ 0.09 & -0.69 \end{bmatrix} & = & \begin{bmatrix} 5.2 & 0.4 \end{bmatrix} \end{matrix}$$

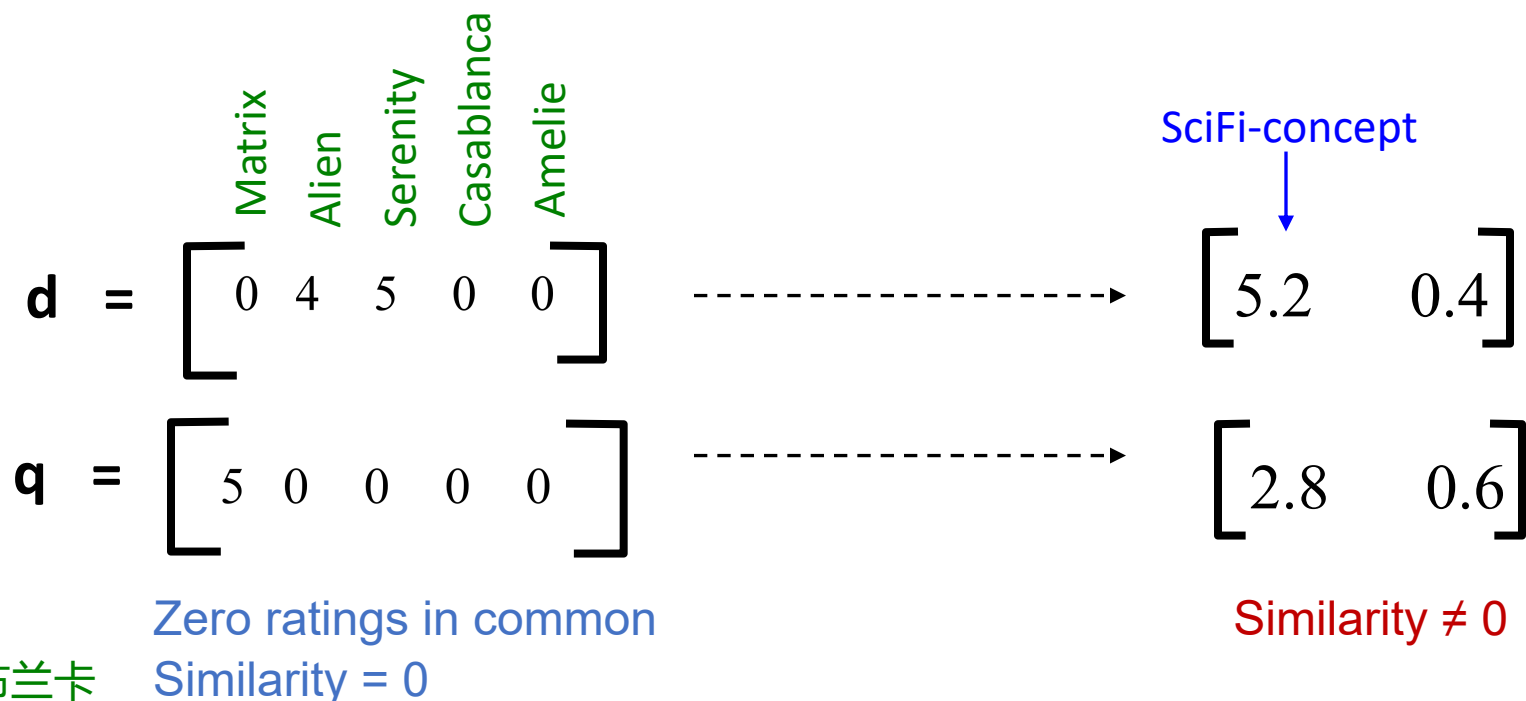
movie-to-concept similarities (V)

SciFi-concept
↓

备注:
Matrix黑客帝国
Alien异形
Serenity惊涛迷局
Casablanca卡萨布兰卡
Amelie天使爱美丽

Case study: How to query?

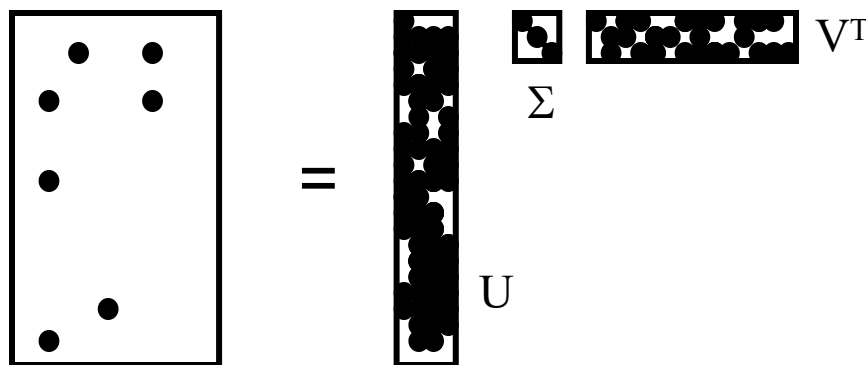
❑ **Observation:** User d that rated ('Alien' , 'Serenity') will be **similar** to user q that rated ('Matrix'), although d and q have **zero ratings in common**!



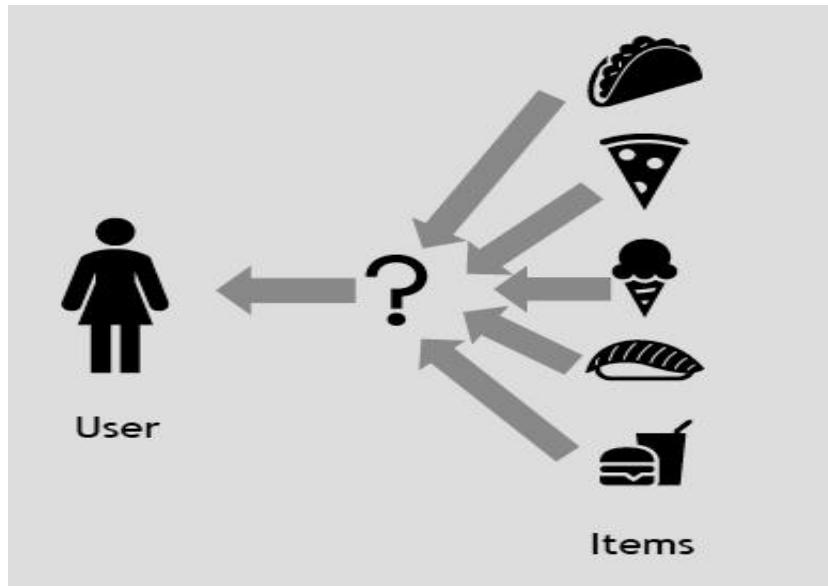
备注:
Matrix黑客帝国
Alien异形
Serenity惊涛迷局
Casablanca卡萨布兰卡
Amelie天使爱美丽

SVD: Advantage and 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**!


$$\begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix} = \begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix} \begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix} \begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix}^T$$

The diagram shows the SVD decomposition of a matrix. On the left is a sparse matrix with 4 rows and 3 columns, containing 6 dots. This is equal to the product of three matrices: a dense matrix U (4x3, filled with dots), a diagonal matrix Σ (3x3, with dots on the diagonal), and a dense matrix V^T (3x3, filled with dots).



CUR Decomposition

CUR Decomposition

Frobenius norm:

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

- **CUR Decomposition(CUR分解)** --- 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**:

$$\left(\begin{array}{c} \text{Red bar} \\ \text{Blue bar} \\ \text{Purple bar} \end{array} \right) \approx \left(\begin{array}{c} \text{Red bar} \\ \text{Red bar} \\ \text{Red bar} \\ \text{Blue bar} \\ \text{Purple bar} \\ \text{Purple bar} \end{array} \right) \cdot \left(\begin{array}{c} \text{Blue bar} \end{array} \right) \cdot \left(\begin{array}{c} \text{Purple bar} \end{array} \right)$$

A **C** **U** **R**

CUR Decomposition

Frobenius norm:

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

华中科技大学
计算机科学与技术学院
School of Computer Science & Technology, HUST

- **CUR Decomposition(CUR分解)** --- 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**:

$$\begin{pmatrix} \text{red bar} \\ \text{purple bar} \\ \text{blue bar} \end{pmatrix} \begin{matrix} \\ A \\ \end{matrix} \approx \begin{pmatrix} \\ C \\ \end{pmatrix} \cdot \begin{pmatrix} U \end{pmatrix} \cdot \begin{pmatrix} \text{red bar} \\ \text{red bar} \\ \text{red bar} \\ \text{purple bar} \\ \text{purple bar} \\ \text{blue bar} \end{pmatrix} \begin{matrix} \\ \\ R \\ \end{matrix}$$

A **C** **U** **R**

Pseudo-inverse (广义逆矩阵, 也称伪逆矩阵) of the intersection (交集矩阵) of **C** and **R**

CUR: Provably good approx. to SVD

□ **Let:** \mathbf{A}_k be the “best” rank k approximation to \mathbf{A} (that is, \mathbf{A}_k is SVD of \mathbf{A})

Theorem [Drineas et al.]

CUR in $O(\mathbf{m} \cdot \mathbf{n})$ time achieves $\|\mathbf{A} - \mathbf{CUR}\|_F \leq \|\mathbf{A} - \mathbf{A}_k\|_F + \varepsilon \|\mathbf{A}\|_F$ with probability at least $1 - \delta$, by picking

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

In practice:
Pick $4k$ cols/rows

□ Sampling columns for matrix **C** (similarly for rows, for matrix **R** in CUR):

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

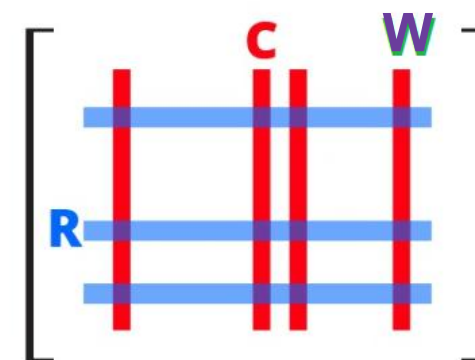
□ Computing U in 4 steps:

□ **Step 1:** Let **W** be the “intersection” of sampled columns **C** and rows **R**

□ **Step 2:** Let SVD of **W** = **X Z Y^T**

□ **Step 3:** **Z⁺** (广义逆矩阵, 也称伪逆矩阵): reciprocals of non-zero singular values, $Z^+_{ii} = 1/Z_{ii}$

□ **Step 4:** Then **U** = **W⁺** = **Y (Z⁺)² X^T**
➤ **W⁺** is the “**pseudoinverse**” (伪逆矩阵)



CUR: Provably good approx. to SVD

□ For example:

- Select $c = O\left(\frac{k \log k}{\epsilon^2}\right)$ columns of A using ColumnSelect algorithm
- Select $r = O\left(\frac{k \log k}{\epsilon^2}\right)$ rows of A using RowSelect algorithm
- Set $U = W^+$

□ Then:

with probability 98%

$$\|A - \overset{\text{CUR error}}{CUR}\|_F \leq (2 + \epsilon) \|A - \overset{\text{SVD error}}{A_k}\|_F$$

In practice:

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

+ Easy interpretation

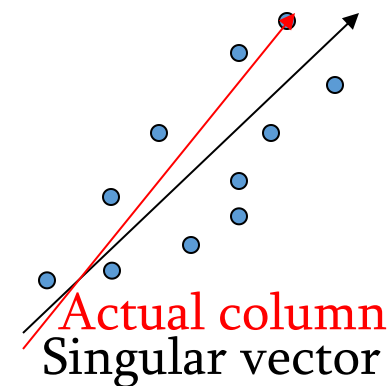
- Since the basis vectors are actual columns and rows

+ Sparse basis

- Since the basis vectors are actual columns and rows

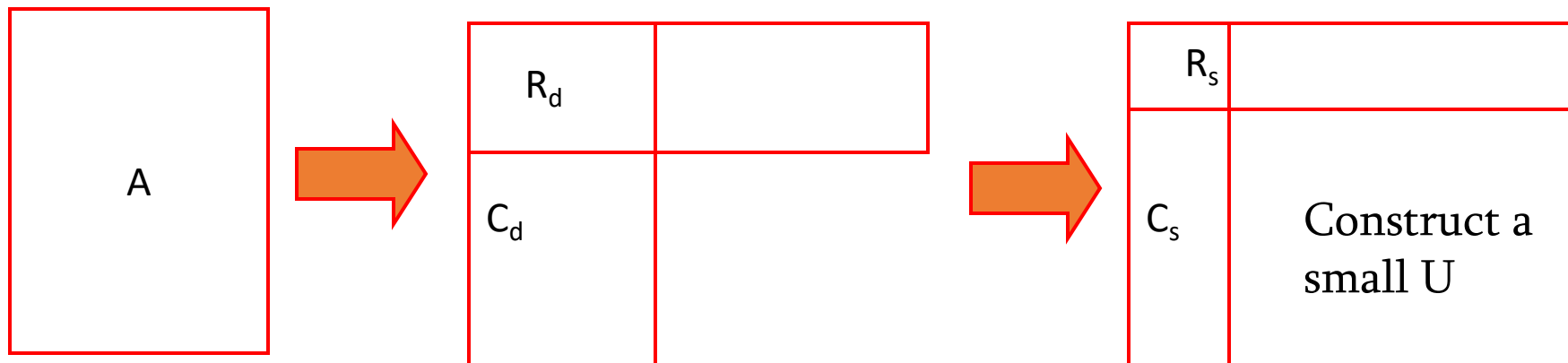
- Duplicate columns and rows

- Columns of large norms will be sampled many times
- But, we can solve it.



□ 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 (最后得到的向量的每个元素乘以重复次数 k 的平方根)



SVD vs. CUR

sparse and small

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

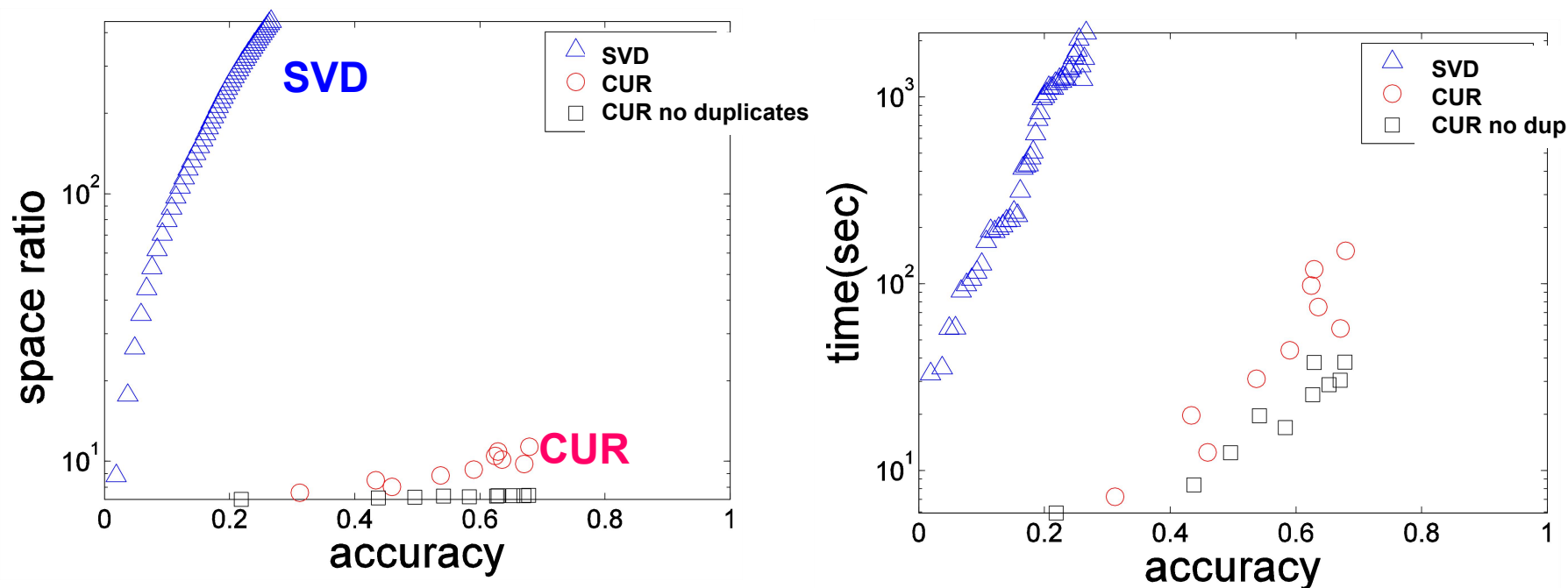
Huge but sparse Big and dense

dense but small

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

Huge but sparse Big but sparse

DBLP author-to-conference big sparse matrix



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

更多详情请见Sun, Faloutsos: *Less is More: Compact Matrix Decomposition for Large Sparse Graphs*, SDM07

□ **Dimensionality reduction**: For a large matrix, can be summarized by finding **narrower matrices** that in some sense are close to the original large matrix.

- These narrow matrices have only small number of rows/ columns, and therefore can be used much more efficiently than can the original large matrix.

□ **Two algorithms:**

- **SVD**
- **CUR**