

# Dimensionality Reduction: SVD & CUR

CS246: Mining Massive Datasets  
Jure Leskovec, Stanford University  
<http://cs246.stanford.edu>



# Reducing Matrix Dimension

- Often, our data can be represented by an  $m$ -by- $n$  matrix
- And this matrix can be closely approximated by the product of three matrices that share a small common dimension  $r$

$$\begin{matrix} n \\ \text{---} \\ m \end{matrix} \boxed{A} \approx \begin{matrix} r \\ \text{---} \\ U \end{matrix} \begin{matrix} m \\ \times \\ r \end{matrix} \begin{matrix} \Sigma \\ \text{---} \\ \Sigma \end{matrix} \begin{matrix} n \\ \text{---} \\ V^T \end{matrix} \begin{matrix} r \\ \text{---} \\ r \end{matrix}$$

The diagram illustrates the Singular Value Decomposition (SVD) of a matrix A. On the left, a pink rectangle labeled 'A' is shown with dimensions m (height) by n (width). An approximation symbol (~) follows. To the right, the decomposition is shown as a product of three matrices: U (green rectangle, r columns by m rows), Sigma (purple square, r columns by r rows), and V^T (yellow rectangle, n columns by r rows). The dimensions of each matrix are indicated by numbers above or below them.

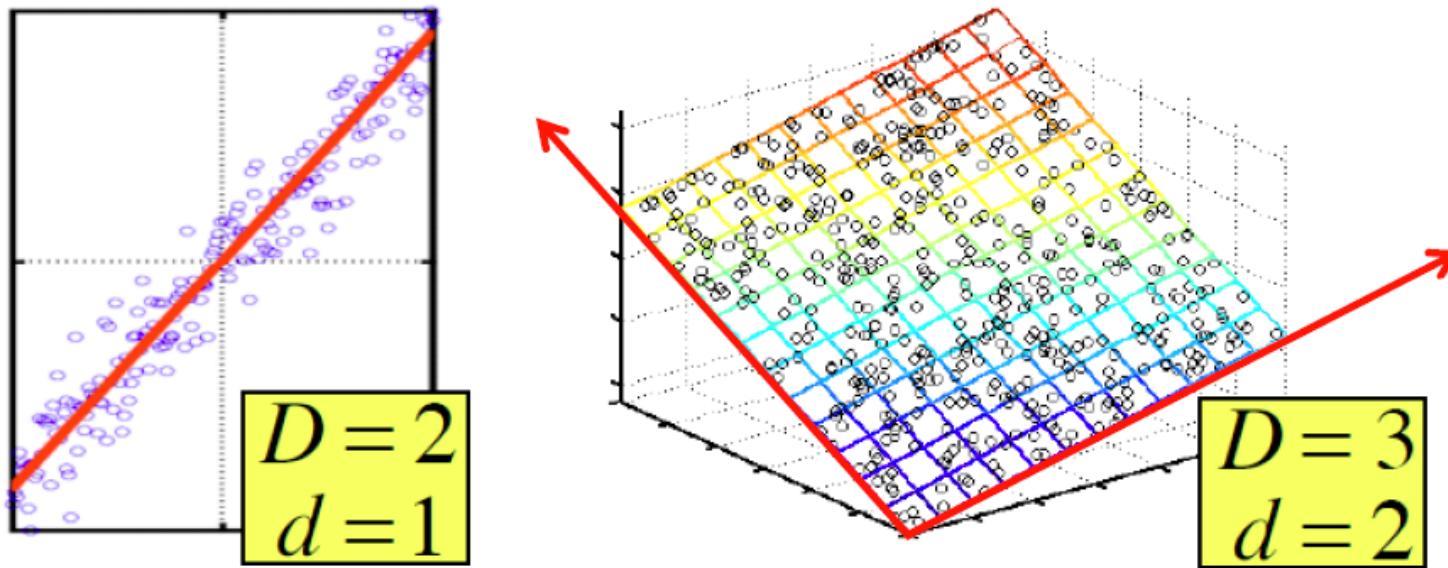
# Dimensionality Reduction

- 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	New representation
		7/10/96	7/11/96	7/12/96	7/13/96	7/14/96	
ABC Inc.		1	1	1	0	0	[1 0]
DEF Ltd.		2	2	2	0	0	[2 0]
GHI Inc.		1	1	1	0	0	[1 0]
KLM Co.		5	5	5	0	0	[5 0]
Smith		0	0	0	2	2	[0 2]
Johnson		0	0	0	3	3	[0 3]
Thompson		0	0	0	1	1	[0 1]

**Note:** 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

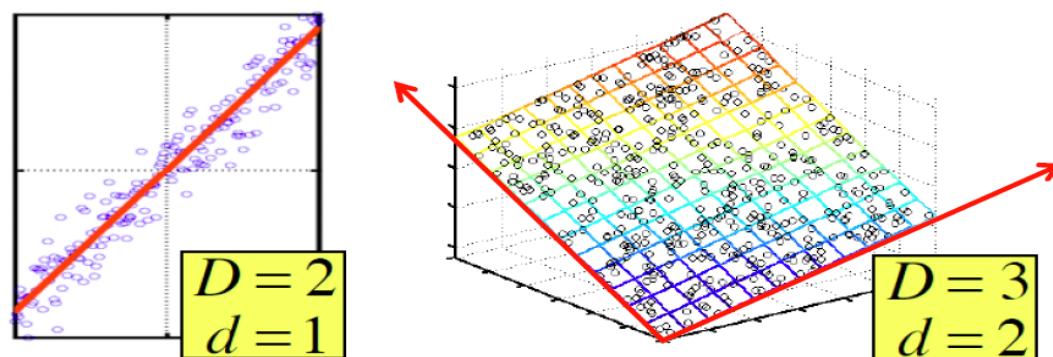


There are hidden, or **latent factors, latent dimensions** that – to a close approximation – explain why the values are as they appear in the data matrix

# Dimensionality Reduction

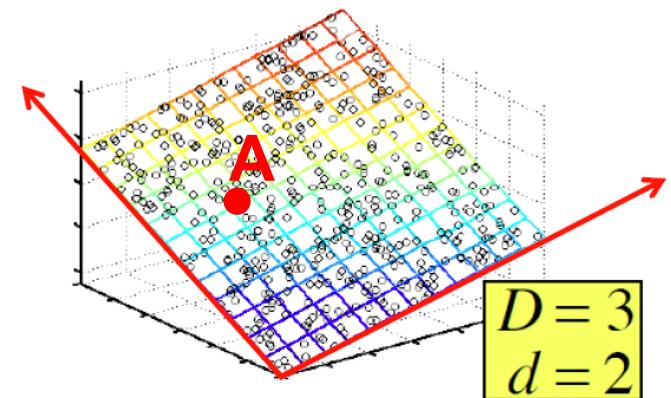
The axes of these dimensions can be chosen by:

- The first dimension is the direction in which the points exhibit the greatest variance
- The second dimension is the direction, orthogonal to the first, in which points show the 2<sup>nd</sup> greatest variance
- And so on..., until you have enough dimensions that variance is really low



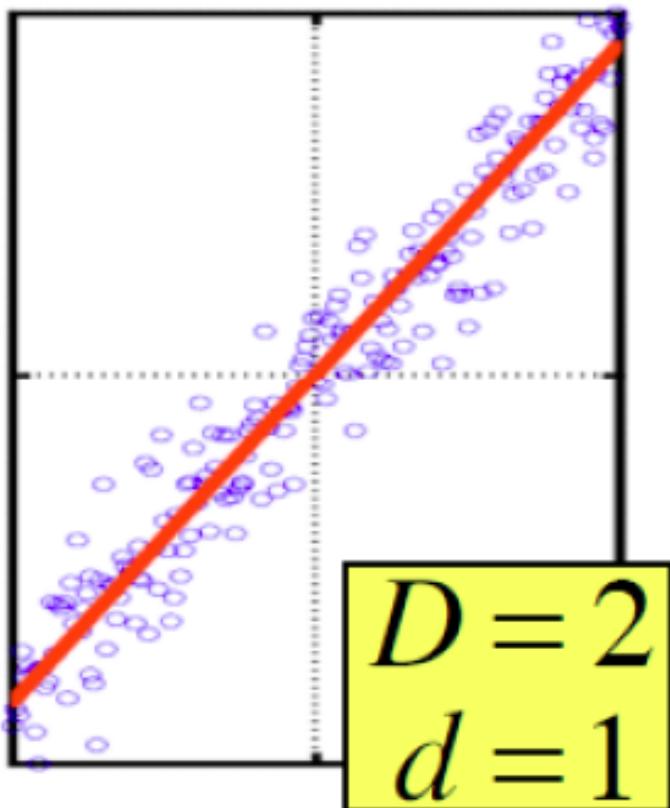
# Rank is “Dimensionality”

- Q: What is **rank** of a matrix A?
- A: Number of **linearly independent** rows of A
- **Cloud of points 3D space:**
  - Think of point positions as a matrix:  
1 row per point:  $\begin{bmatrix} 1 & 2 & 1 \\ -2 & -3 & 1 \\ 3 & 5 & 0 \end{bmatrix}$  **A** **B** **C**
- 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 dimensions/coordinates!



# Dimensionality Reduction

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



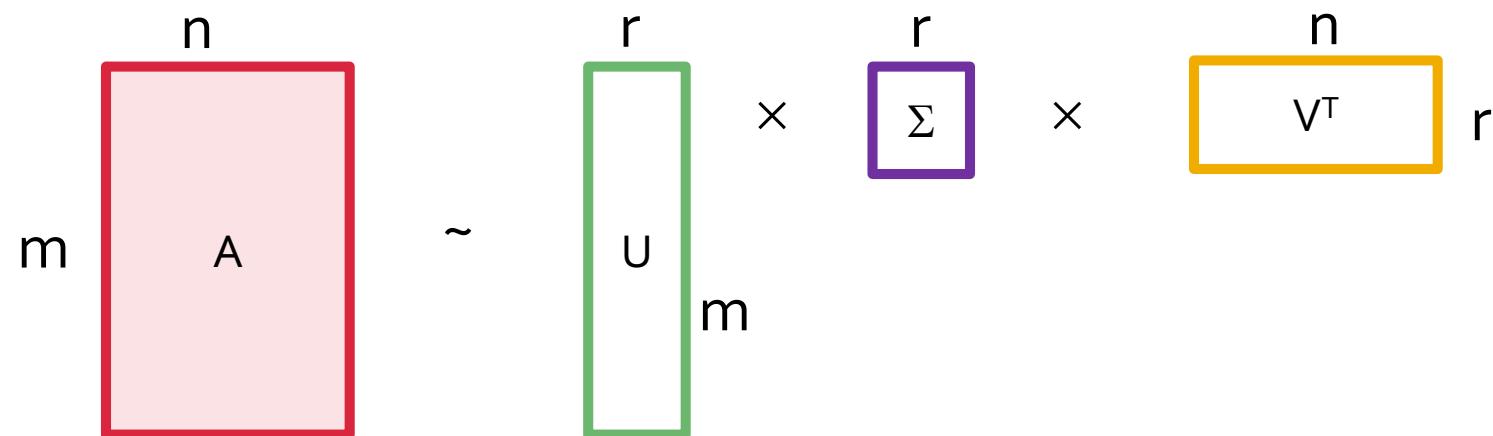
Rather than representing every point with 2 coordinates we represent each point with 1 coordinate (corresponding to the position of the point on the red line).

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

# SVD: Singular Value Decomposition

# Reducing Matrix Dimension

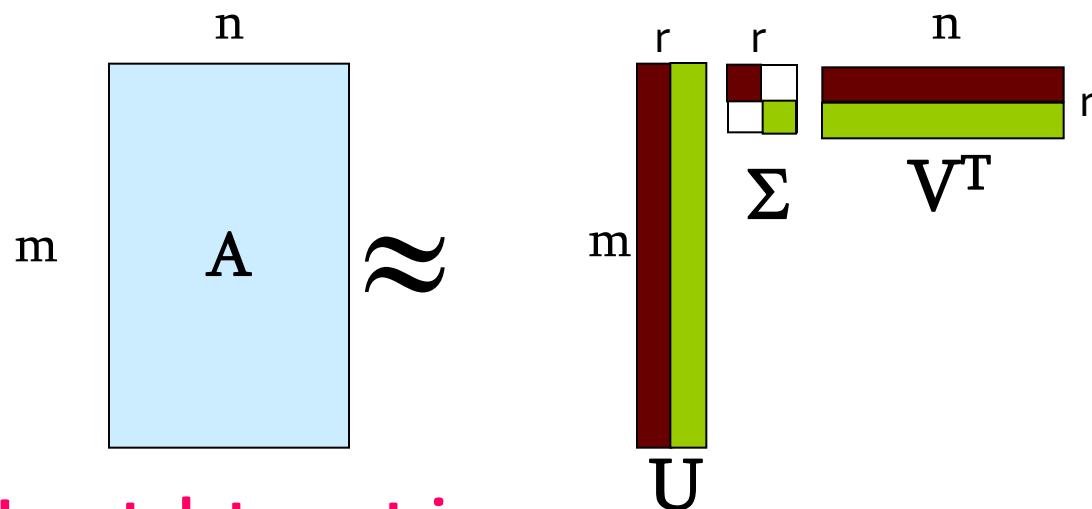
- Gives a decomposition of any matrix into a product of three matrices:



- There are strong constraints on the form of each of these matrices
  - Results in a unique decomposition
- From this decomposition, you can choose any number  $r$  of intermediate concepts (latent factors) in a way that minimizes the reconstruction error

# SVD – Definition

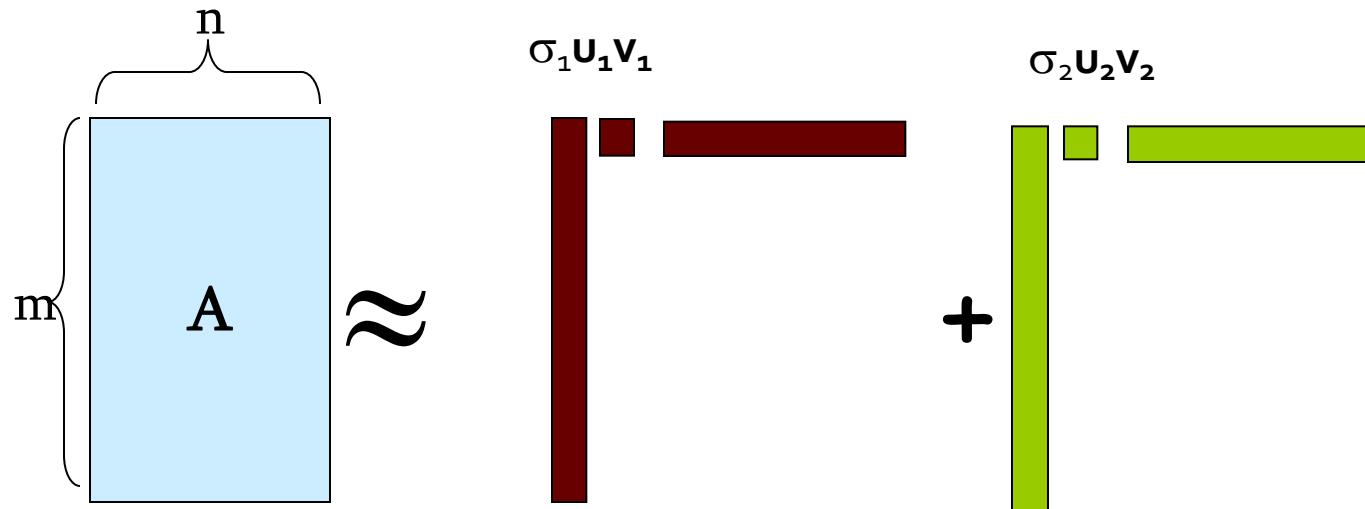
$$A \approx U\Sigma V^T = \sum_i \sigma_i u_i \circ v_i^\top$$



- **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)
- **$\Sigma$ : 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

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



If we set  $\sigma_2 = 0$ , then the green columns may as well not exist.

$\sigma_i$  ... scalar  
 $\mathbf{u}_i$  ... vector  
 $\mathbf{v}_i$  ... vector

# SVD – Properties

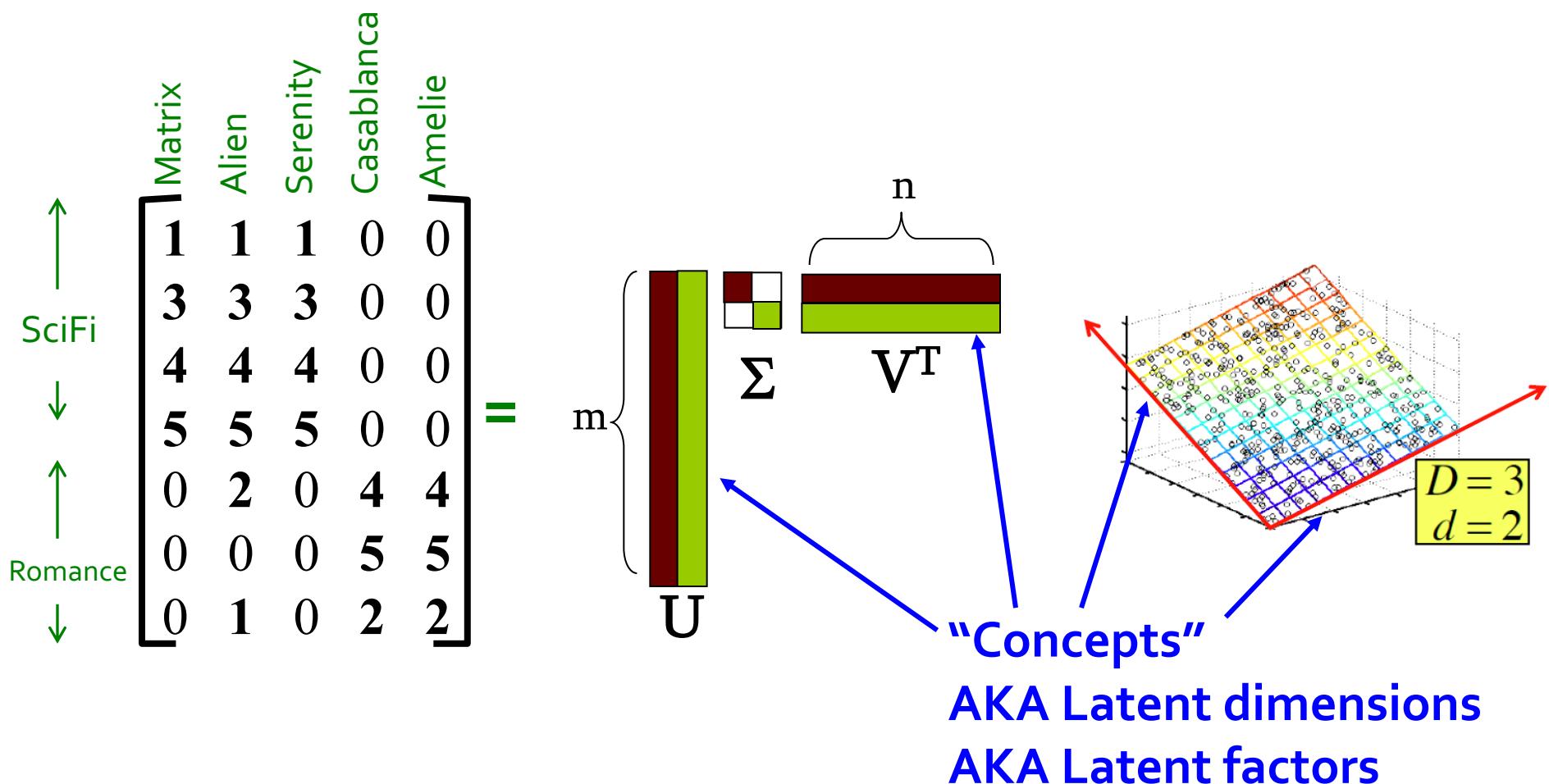
It is **always** possible to decompose a real matrix  $\mathbf{A}$  into  $\mathbf{A} = \mathbf{U} \Sigma \mathbf{V}^T$ , where

- $\mathbf{U}, \Sigma, \mathbf{V}$ : 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)
- $\Sigma$ : 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-ws04/lecture2 pdf>

# SVD – Example: Users-to-Movies

- Consider a matrix. What does SVD do?



# SVD – Example: Users-to-Movies

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

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

# SVD – Example: Users-to-Movies

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

$$\begin{array}{c}
 \text{Matrix} \\
 \begin{bmatrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\ 
 \text{SciFi} & \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} & = & \begin{bmatrix} \text{SciFi-concept} \\ \text{Romance-concept} \end{bmatrix} \\
 \downarrow & & & \\
 \text{Romance} & & & 
 \end{bmatrix}
 \end{array}$$

The diagram illustrates the decomposition of a user-movie rating matrix into three components:  $U$ ,  $\Sigma$ , and  $V^T$ . The original matrix has columns for Alien, Serenity, Casablanca, and Amelie, and rows for SciFi and Romance. The matrix is decomposed into a product of three matrices:  $U$  (containing the SciFi and Romance concepts),  $\Sigma$  (a diagonal matrix with singular values 12.4, 9.5, and 1.3), and  $V^T$  (containing the movie features). The SciFi-concept row in  $U$  has values [0.13, 0.41, 0.55, 0.68, 0.15, 0.07, 0.07]. The Romance-concept row in  $U$  has values [0.02, 0.07, 0.09, 0.11, -0.59, -0.73, -0.29]. The  $\Sigma$  matrix is [12.4 0 0; 0 9.5 0; 0 0 1.3]. The  $V^T$  matrix is [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:

$U$  is “user-to-concept” factor matrix

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

# SVD – Example: Users-to-Movies

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

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:

$V$  is “movie-to-concept” factor matrix

The diagram shows the decomposition of a user-movie rating matrix  $A$  into three components:  $U$ ,  $\Sigma$ , and  $V^T$ .

**Matrix  $A$ :**

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

**SciFi-concept:** A blue arrow points from the SciFi row of  $A$  to the first column of the  $V^T$  matrix.

**SciFi-concept:** A blue arrow points from the SciFi-concept column of the  $V^T$  matrix to the circled value **0.56** in the resulting matrix.

**Matrix  $U$ :**

	SciFi	Romance	SciFi	Romance	SciFi
1	0.13	0.02	-0.01		
3	0.41	0.07	-0.03		
4	0.55	0.09	-0.04		
5	0.68	0.11	-0.05		
0	0.15	-0.59	0.65		
0	0.07	-0.73	-0.67		
0	0.07	-0.29	0.32		

**Matrix  $\Sigma$ :**

12.4	0	0
0	9.5	0
0	0	1.3

**Matrix  $V^T$ :**

	SciFi	Romance	SciFi	Romance	SciFi
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	

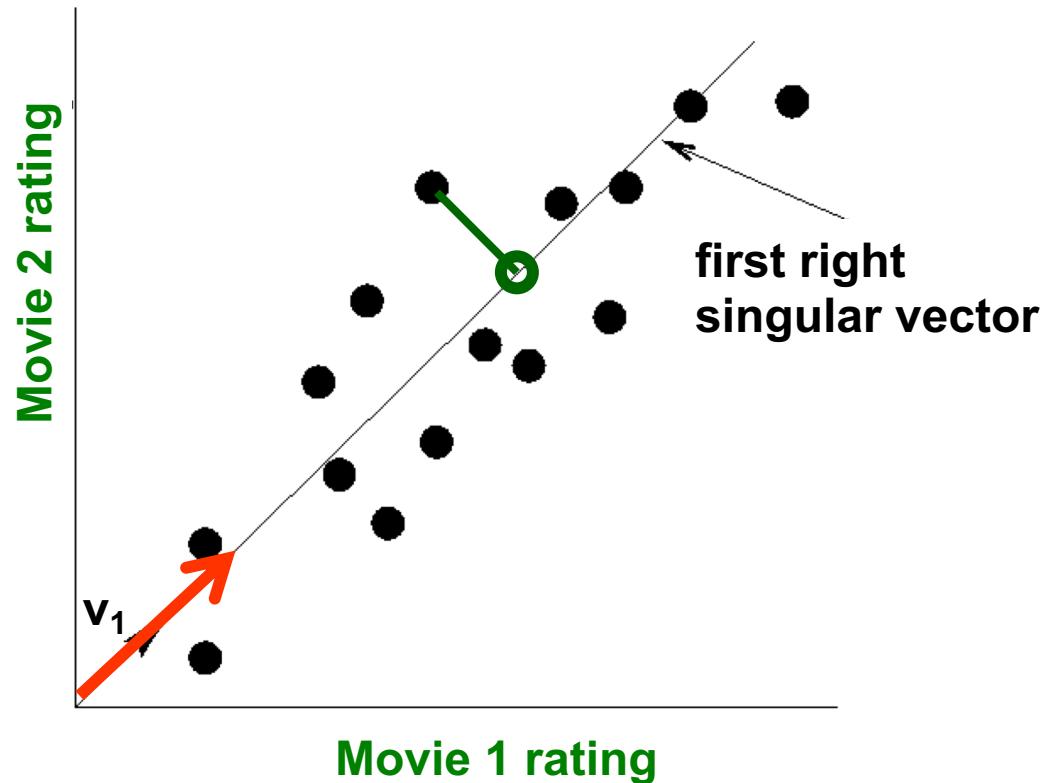
# SVD – Interpretation #1

**Movies, users and concepts:**

- $U$ : user-to-concept matrix
- $V$ : movie-to-concept matrix
- $\Sigma$ : its diagonal elements:  
‘strength’ of each concept

# Dimensionality Reduction with SVD

# SVD – Dimensionality Reduction



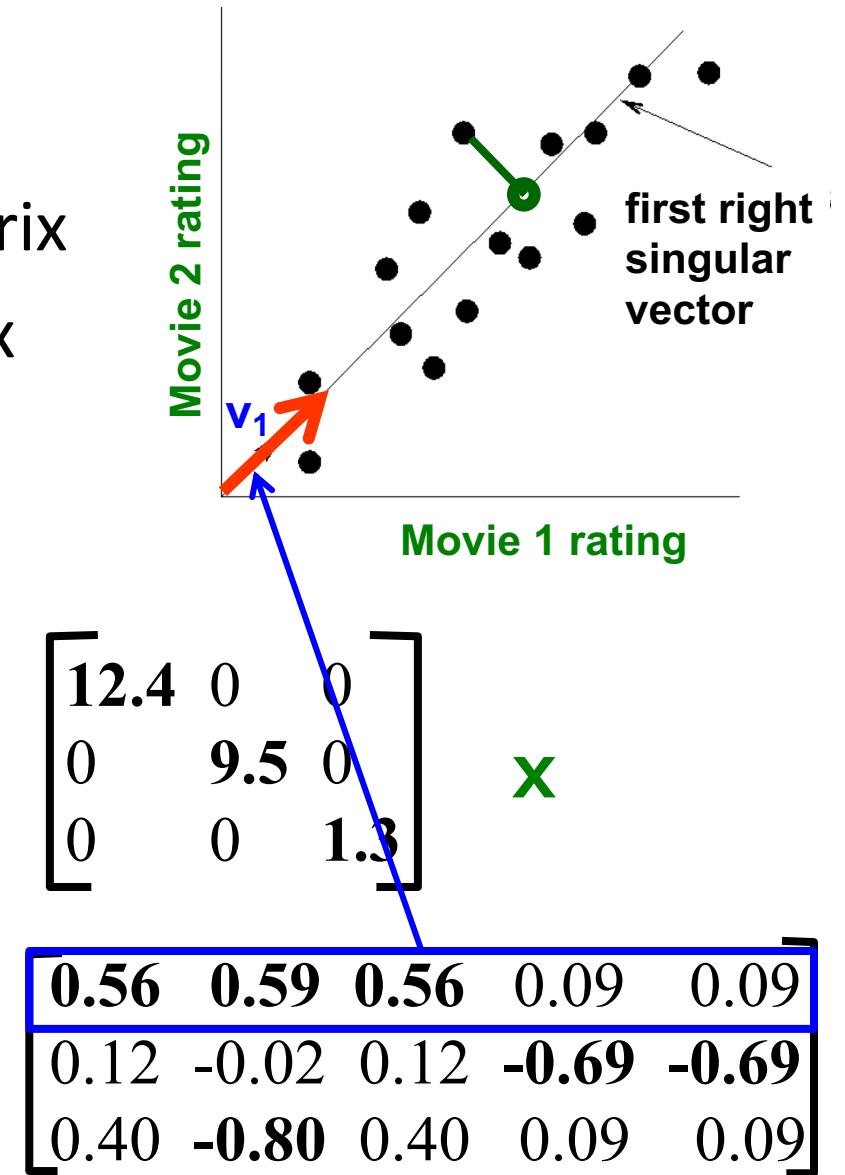
- Instead of using two coordinates  $(x, y)$  to describe point locations, let's use only one coordinate
- Point's position is its location along vector  $v_1$

# SVD – Dimensionality Reduction

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

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

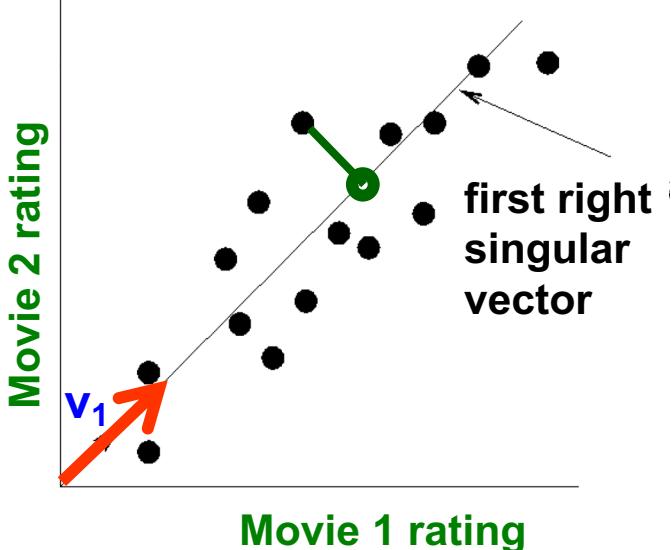
$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 &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



# SVD – Dimensionality Reduction

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

variance ('spread')  
on the  $v_1$  axis



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

**variance ('spread') on the  $v_1$  axis**

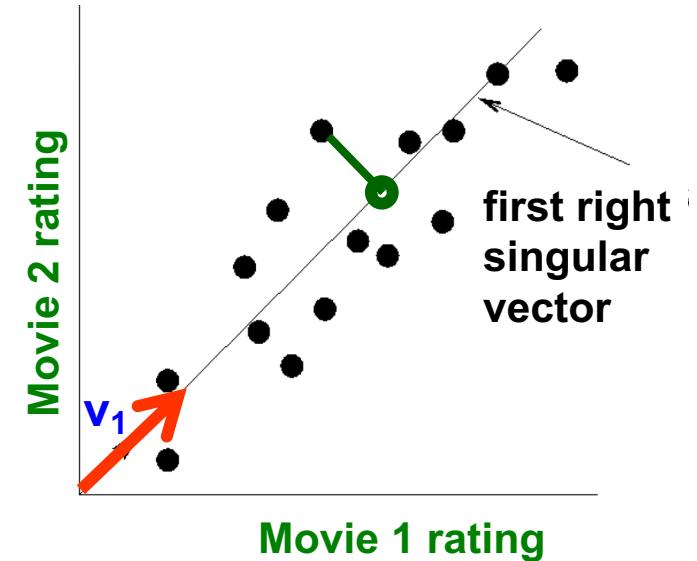
# SVD – Dimensionality Reduction

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



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 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

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

# 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

This is Rank 2 approximation to A.  
We could also do Rank 1 approx.  
The larger the rank  
the more accurate  
the approximation.

## 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

This is Rank 2 approximation to A.  
We could also do Rank 1 approx.  
The larger the rank  
the more accurate  
the approximation.

## 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

This is Rank 2 approximation to A.  
We could also do Rank 1 approx.  
The larger the rank the more accurate the approximation

## More details

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

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

Reconstructed  
data matrix B

Reconstruction Error is quantified by the 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.

- Fact: SVD gives ‘best’ axis to project on:
  - ‘best’ = minimizing the sum of reconstruction errors

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

B is best approximation of A:

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

# SVD – Conclusions so far

- **SVD:  $A = U \Sigma V^T$ : unique**
  - $U$ : user-to-concept factors
  - $V$ : movie-to-concept factors
  - $\Sigma$  : strength of each concept
- **Q: So what's a good value for r?**
- Let the *energy* of a set of singular values be the sum of their squares.
- Pick  $r$  so the retained singular values have at least 90% of the total energy.
- **Back to our example:**
  - With singular values 12.4, 9.5, and 1.3, total energy = 245.7
  - If we drop 1.3, whose square is only 1.7, we are left with energy 244, or over 99% of the total

# How to Compute SVD

# Finding Eigenpairs

- How do we actually compute SVD?
- First we need a method for finding the **principal eigenvalue** (the largest one) and the corresponding **eigenvector** of a symmetric matrix
  - $M$  is *symmetric* if  $m_{ij} = m_{ji}$  for all  $i$  and  $j$
- Method:
  - Start with any “guess eigenvector”  $\mathbf{x}_0$
  - Construct  $\mathbf{x}_{k+1} = \frac{M\mathbf{x}_k}{\|M\mathbf{x}_k\|}$  for  $k = 0, 1, \dots$ 
    - $\| \dots \|$  denotes the Frobenius norm
  - Stop when consecutive  $\mathbf{x}_k$  show little change

# Example: Iterative Eigenvector

$$M = \begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix} \quad \mathbf{x}_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\frac{M\mathbf{x}_0}{\|M\mathbf{x}_0\|} = \begin{bmatrix} 3 \\ 5 \end{bmatrix} / \sqrt{34} = \begin{bmatrix} 0.51 \\ 0.86 \end{bmatrix} = \mathbf{x}_1$$

$$\frac{M\mathbf{x}_1}{\|M\mathbf{x}_1\|} = \begin{bmatrix} 2.23 \\ 3.60 \end{bmatrix} / \sqrt{17.93} = \begin{bmatrix} 0.53 \\ 0.85 \end{bmatrix} = \mathbf{x}_2$$

.....

# Finding the Principal Eigenvalue

- Once you have the principal eigenvector  $\mathbf{x}$ , you find its eigenvalue  $\lambda$  by  $\lambda = \mathbf{x}^T M \mathbf{x}$ .
  - In proof: We know  $\mathbf{x}\lambda = M\mathbf{x}$  if  $\lambda$  is the eigenvalue; multiply both sides by  $\mathbf{x}^T$  on the left.
  - Since  $\mathbf{x}^T \mathbf{x} = 1$  we have  $\lambda = \mathbf{x}^T M \mathbf{x}$
- Example:** If we take  $\mathbf{x}^T = [0.53, 0.85]$ , then

$$\lambda = [0.53 \ 0.85] \begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix} \begin{bmatrix} 0.53 \\ 0.85 \end{bmatrix} = 4.25$$

# Finding More Eigenpairs

- Eliminate the portion of the matrix  $M$  that can be generated by the first eigenpair,  $\lambda$  and  $x$ :

$$M^* := M - \lambda x x^T$$

- Recursively find the principal eigenpair for  $M^*$ , eliminate the effect of that pair, and so on

- **Example:**

$$M^* = \begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix} - 4.25 \begin{bmatrix} 0.53 \\ 0.85 \end{bmatrix} \begin{bmatrix} 0.53 & 0.85 \end{bmatrix} = \begin{bmatrix} -0.19 & 0.09 \\ 0.09 & 0.07 \end{bmatrix}$$

# How to Compute the SVD

- Start by supposing  $A = U\Sigma V^T$
- $A^T = (U\Sigma V^T)^T = (V^T)^T \Sigma^T U^T = V \Sigma U^T$ 
  - Why? (1) Rule for transpose of a product; (2) the transpose of the transpose and the transpose of a diagonal matrix are both the identity functions
- $A^T A = V \Sigma U^T U \Sigma V^T = V \Sigma^2 V^T$ 
  - Why?  $U$  is orthonormal, so  $U^T U$  is an identity matrix
  - Also note that  $\Sigma^2$  is a diagonal matrix whose  $i$ -th element is the square of the  $i$ -th element of  $\Sigma$
- $A^T A V = V \Sigma^2 V^T V = V \Sigma^2$ 
  - Why?  $V$  is also orthonormal

# Computing the SVD –(2)

- Starting with  $(A^T A) V = V \Sigma^2$ 
  - Note that therefore the  $i$ -th column of  $V$  is an eigenvector of  $A^T A$ , and its eigenvalue is the  $i$ -th element of  $\Sigma^2$
- Thus, we can find  $V$  and  $\Sigma$  by finding the eigenpairs for  $A^T A$ 
  - Once we have the eigenvalues in  $\Sigma^2$ , we can find the singular values by taking the square root of these eigenvalues
- Symmetric argument,  $AA^T$  gives us  $U$

# SVD – Complexity

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

# Example of SVD

# Case study: How to query?

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

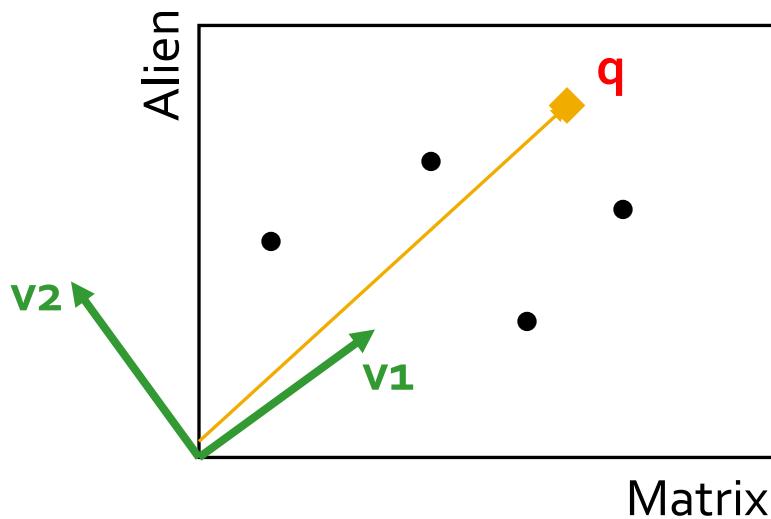
$$\begin{array}{c} \text{SciFi} \\ \uparrow \\ \boxed{\begin{matrix} & \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\ \hline 1 & 1 & 1 & 0 & 0 & \\ 3 & 3 & 3 & 0 & 0 & \\ 4 & 4 & 4 & 0 & 0 & \\ 5 & 5 & 5 & 0 & 0 & \\ \hline 0 & 2 & 0 & 4 & 4 & \\ 0 & 0 & 0 & 5 &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

# 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  $v_i$

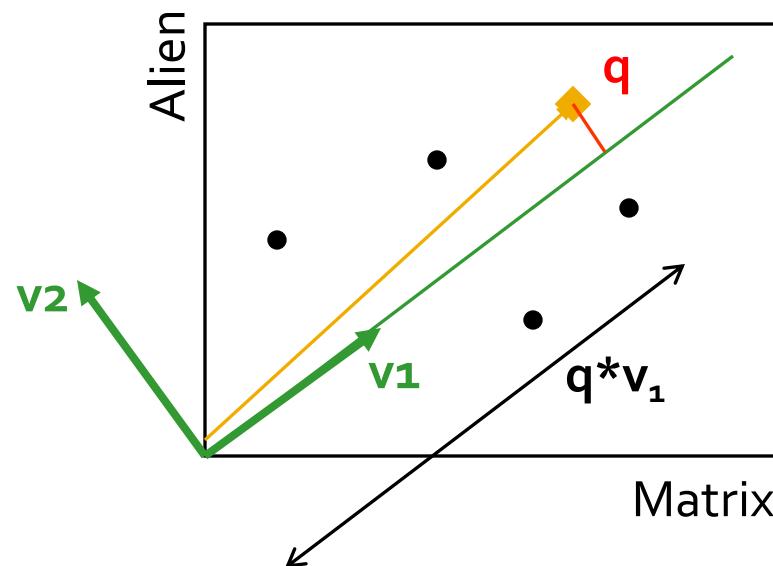


# Case study: How to query?

- 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:

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

E.g.:

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

SciFi-concept

movie-to-concept  
factors ( $\mathbf{V}$ )

# Case study: How to query?

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

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

E.g.:

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

movie-to-concept  
factors (V)

SciFi-concept  
↓  
 $= \begin{bmatrix} 5.2 & 0.4 \end{bmatrix}$

# 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!**

**Matrix Factorization Diagram**

The diagram illustrates the calculation of ratings for movies based on user profiles. It shows two matrices,  $d$  and  $q$ , being multiplied to produce rating vectors for "SciFi-concept" and "Amelie".

**Matrix  $d$ :**

$$d = \begin{bmatrix} & \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\ & 0 & 4 & 5 & 0 & 0 \end{bmatrix}$$

**Matrix  $q$ :**

$$q = \begin{bmatrix} & 5 & 0 & 0 & 0 & 0 \end{bmatrix}$$

**Rating Vectors:**

- SciFi-concept:**  $\begin{bmatrix} 5.2 & 0.4 \end{bmatrix}$
- Amelie:**  $\begin{bmatrix} 2.8 & 0.6 \end{bmatrix}$

**Annotations:**

- Zero ratings in common:** A green box at the bottom left indicates that there are zero ratings in common between the user profiles.
- Similarity > 0:** A green box at the bottom right indicates that the similarity between the user profiles is greater than zero.

# 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!**

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

# CUR Decomposition

# Sparsity

- It is common for the matrix  $A$  that we wish to decompose to be very sparse
- But  $U$  and  $V$  from a SVD decomposition will **not** be sparse
- **CUR** decomposition solves this problem by using only (randomly chosen) rows and columns of  $A$

# 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  $R$ :

$$\left( \begin{array}{c|c|c} & & \\ A & & \\ & & \end{array} \right) \approx \left( \begin{array}{c|c|c|c|c} & & & & \\ C & & & & \\ & & & & \end{array} \right) \cdot \left( \begin{array}{c} U \\ \vdots \\ U \end{array} \right) \cdot \left( \begin{array}{c} R \\ \vdots \\ R \end{array} \right)$$

# 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  $R$ :

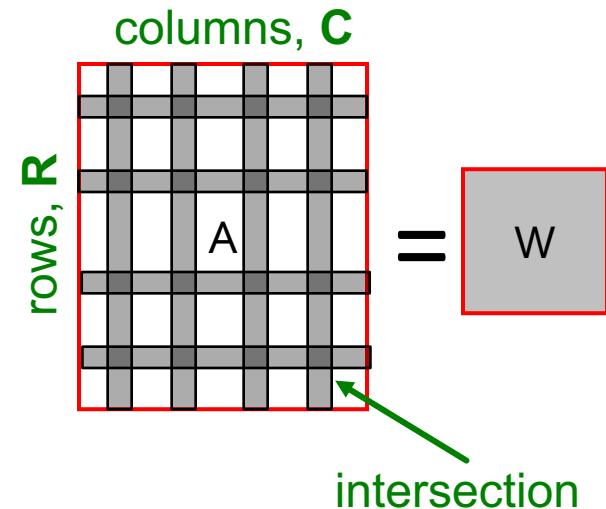
$$\begin{pmatrix} \text{Red row} \\ \text{White row} \\ A \\ \text{Blue row} \end{pmatrix} \approx \begin{pmatrix} C \\ \vdots \end{pmatrix} \cdot \begin{pmatrix} U \\ \vdots \end{pmatrix} \cdot \begin{pmatrix} \text{Red row} \\ \text{Red row} \\ R \\ \text{Red row} \\ \text{Blue row} \end{pmatrix}$$

A                    C                    U                    R

Pseudo-inverse of  
the intersection of  $C$  and  $R$

# Computing U

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



Why the intersection? These are high magnitude numbers  
Why pseudoinverse works?

$$\mathbf{W} = \mathbf{X} \mathbf{Z} \mathbf{Y}^T \text{ then } \mathbf{W}^{-1} = (\mathbf{Y}^T)^{-1} \mathbf{Z}^{-1} \mathbf{X}^{-1}$$

Due to orthonormality:  $\mathbf{X}^{-1} = \mathbf{X}^T$ ,  $\mathbf{Y}^{-1} = \mathbf{Y}^T$

Since  $\mathbf{Z}$  is diagonal  $\mathbf{Z}^{-1} = 1/Z_{ii}$

Thus, if  $\mathbf{W}$  is nonsingular, pseudoinverse is the true inverse

# Which Rows and Columns?

- To decrease the expected error between  $A$  and its decomposition, we must pick rows and columns in a nonuniform manner
- The **importance** of a row or column of  $A$  is the **square of its Frobenius norm**
  - That is, the sum of the squares of its elements.
- When picking rows and columns, the probabilities must be proportional to importance
- **Example:** [3,4,5] has importance 50, and [3,0,1] has importance 10, so pick the first 5 times as often as the second

# CUR: Row Sampling Algorithm

- **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

# Intuition



- **Rough and imprecise intuition behind CUR**
  - CUR is more likely to pick points away from the origin
    - Assuming smooth data with no outliers these are the directions of maximum variation
- **Example:** Assume we have 2 clouds at an angle
  - SVD dimensions are orthogonal and thus will be in the middle of the two clouds
  - CUR will find the two clouds (but will be redundant)

# 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** (slide 56)
  - Select  $r = O\left(\frac{k \log k}{\varepsilon^2}\right)$  rows of A using **RowSelect algorithm** (slide 56)
  - Set  $U = W^+$  SVD error
  - Then:  $\left\| A - CUR \right\|_F \leq (2 + \varepsilon) \left\| A - A_K \right\|_F$  CUR error  
with probability 98%
- In practice:**  
Pick 4k cols/rows  
for a “rank-k” approximation

# CUR: Pros & Cons

## + Easy interpretation

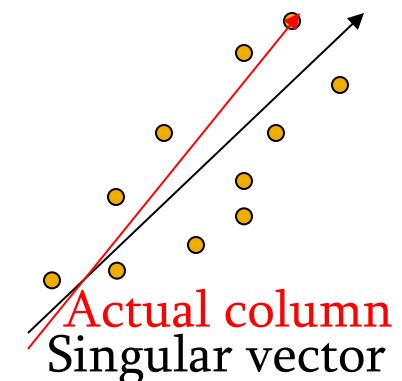
- Since the basis vectors are actual columns and rows

## + Sparse basis

- Since the basis vectors are actual columns and rows

## - Duplicate columns and rows

- Columns of large norms will be sampled many times



# SVD vs. CUR

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

Huge but sparse

sparse and small

Big and dense

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

Huge but sparse

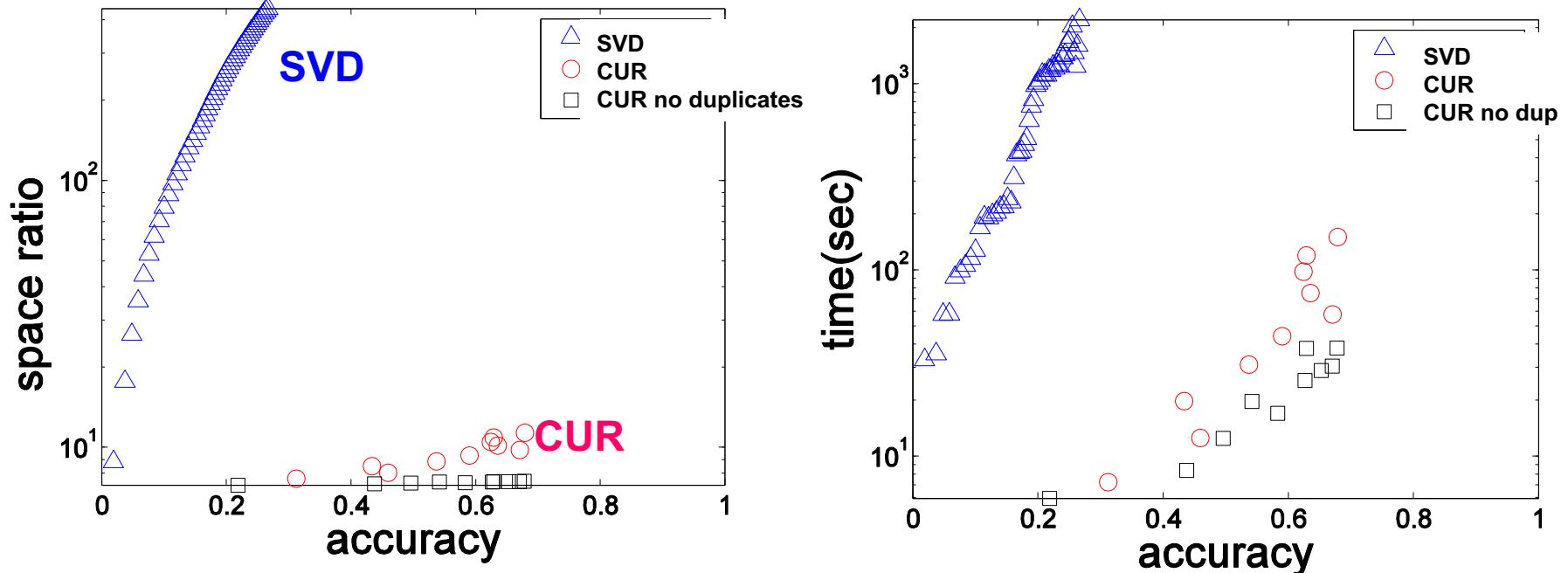
dense but small

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**

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