

Note to other teachers and users of these slides: We would be delighted if you found this our 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

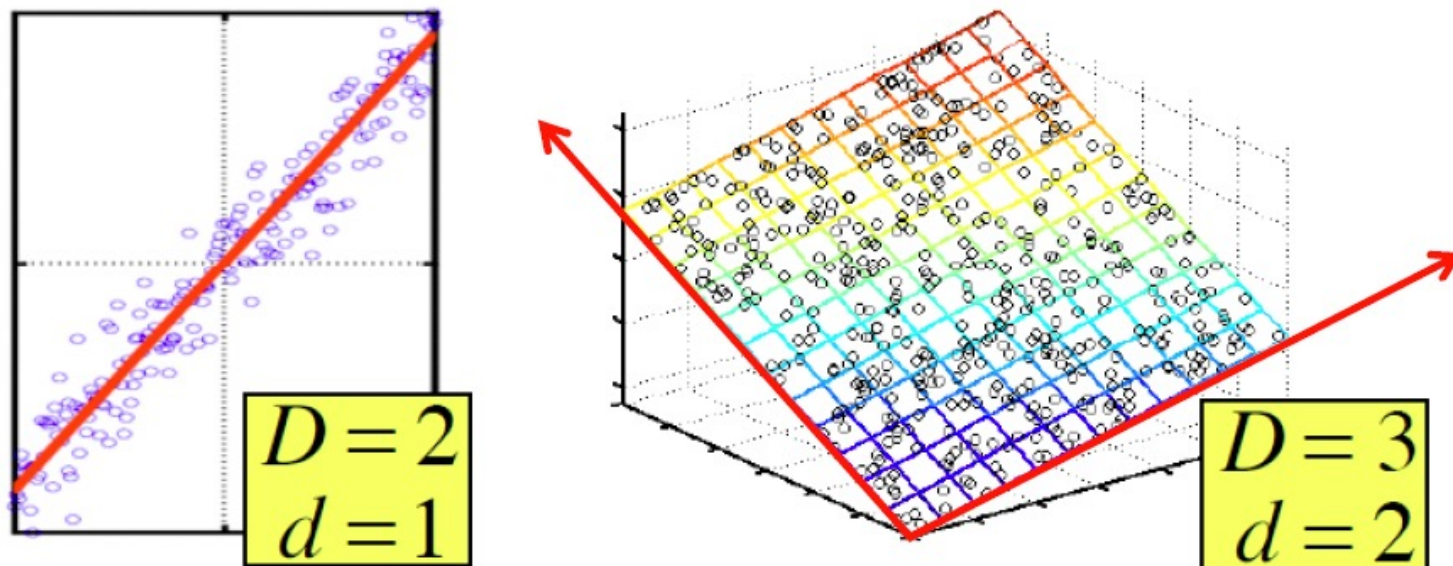
Jure Leskovec, Anand Rajaraman, Jeff Ullman

Stanford University

<http://www.mmds.org>



Linear Dimension Reduction



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

Rank of a Matrix

- **Q:** What is **rank** 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 : $[1 \ 0] \ [0 \ 1] \ [1 \ 1]$

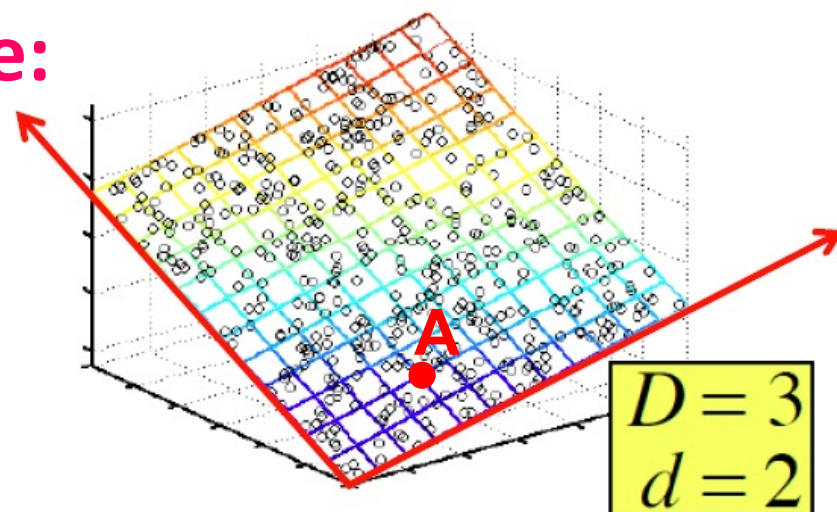
Rank is “Dimensionality”

■ Cloud of points 3D space:

- Think of point positions

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

1 row per point: $\begin{bmatrix} 1 & 2 & 1 \\ -2 & -3 & 1 \\ 3 & 5 & 0 \end{bmatrix}$

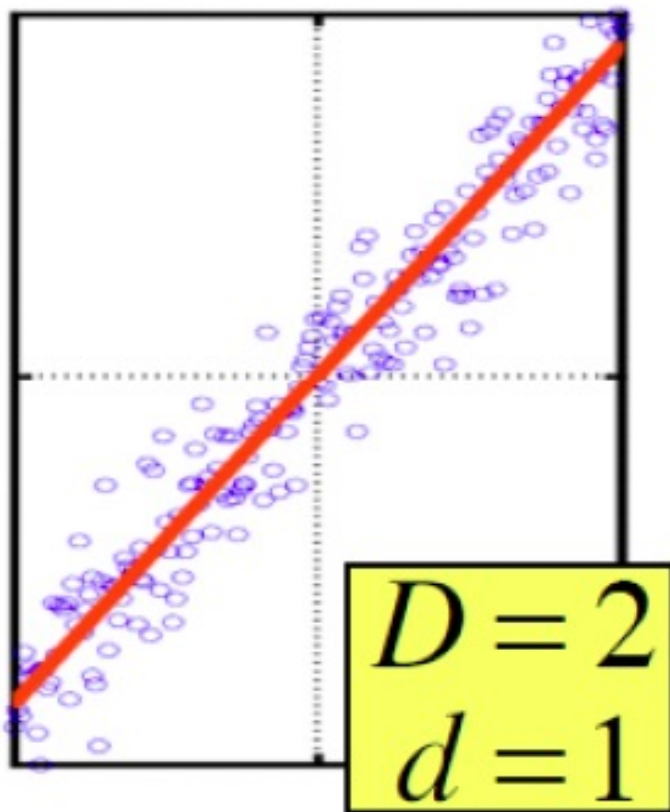


■ We can rewrite coordinates more efficiently!

- 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!



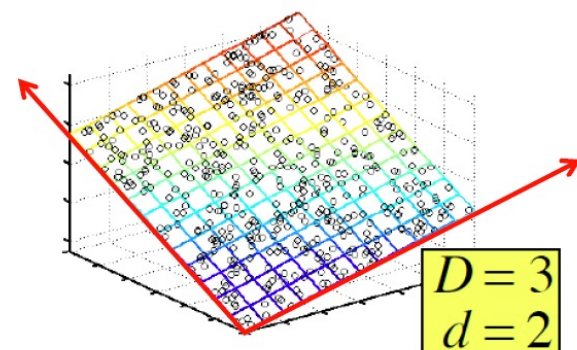
Rather than representing every point with 2 coordinates we represent each point with 1 coordinate (corresponding to the position of the point on the red line).

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**



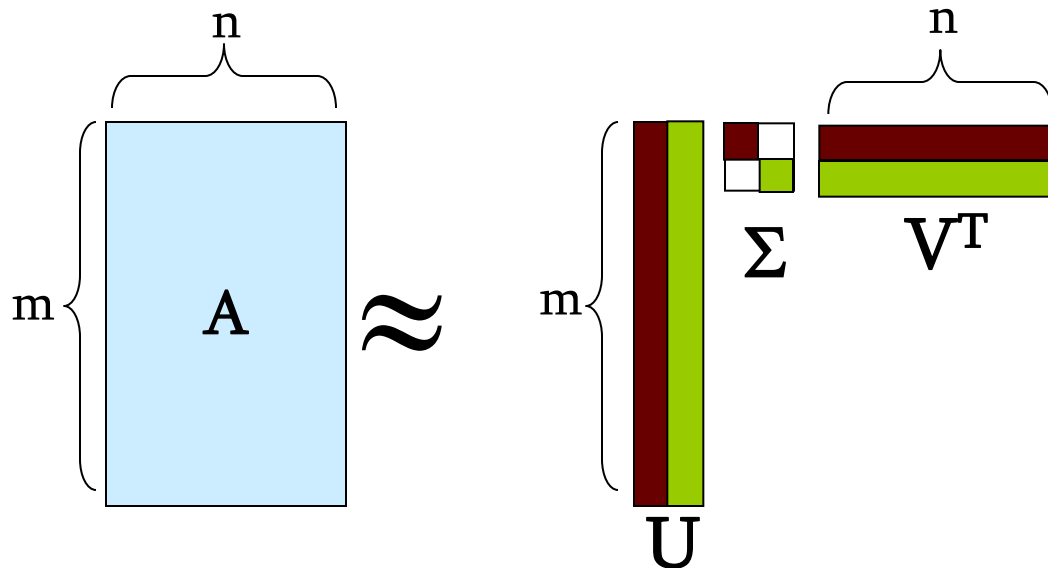
SVD - Definition

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

- **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)

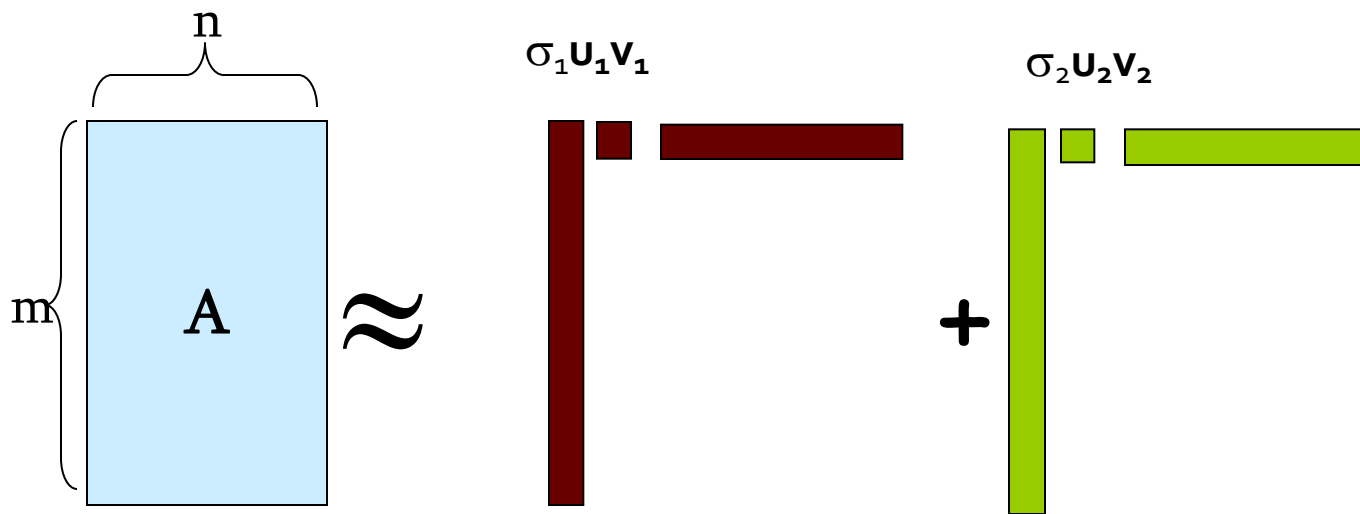
SVD

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



SVD

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



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

SVD - Properties

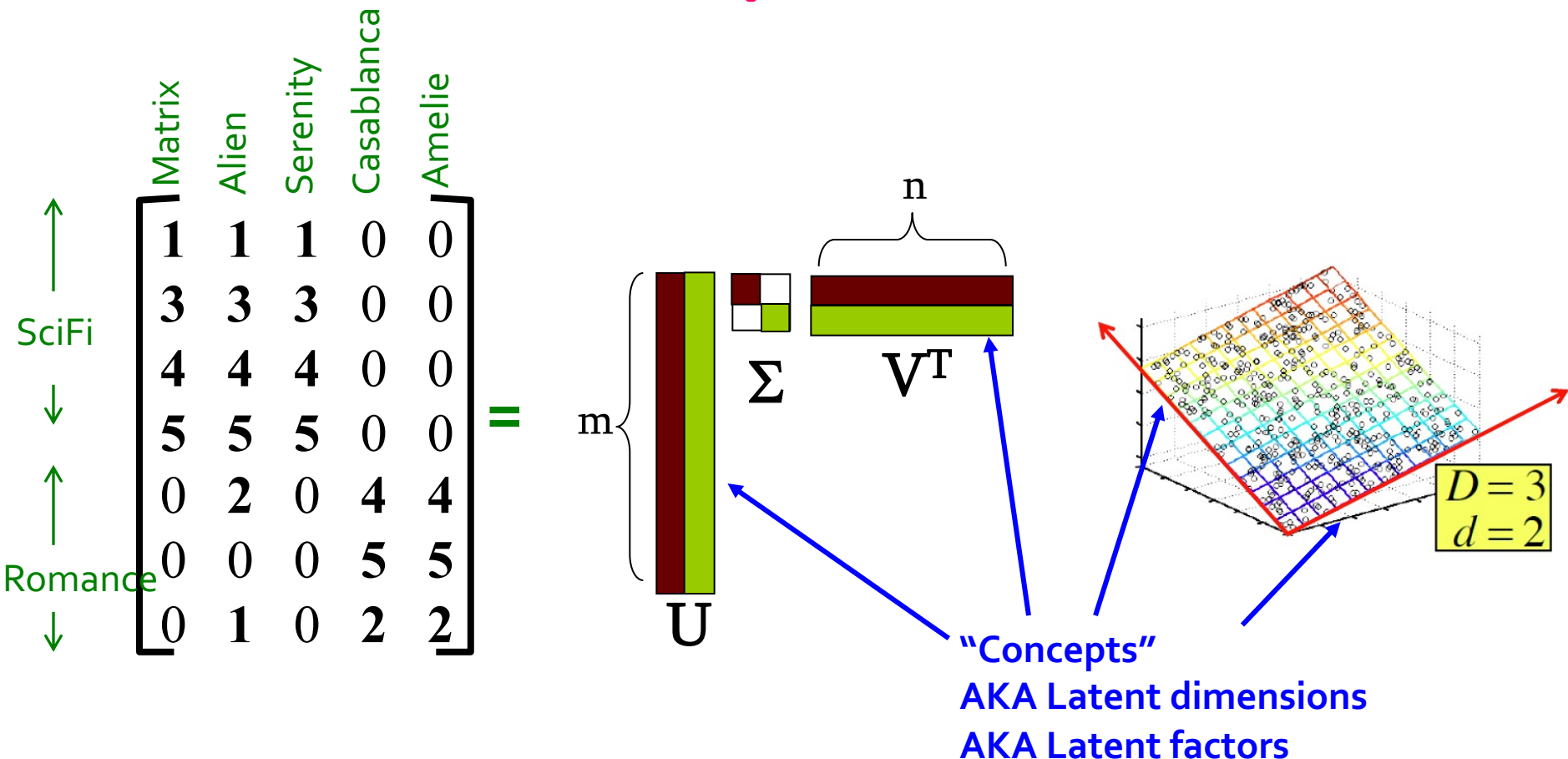
It is **always** possible to decompose a real matrix **A** into **$A = U \Sigma V^T$** , where

- **U, Σ , V: unique**
- **U, V: column orthonormal**
 - **$U^T U = I$; $V^T V = I$** (**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

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

Matrix Alien Serenity Casablanca Amelie

SciFi

Romnce

SciFi-concept

Romance-concept

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

SVD – Example: Users-to-Movies

■ $A = U \Sigma V^T$ - example:

U is “user-to-concept” similarity matrix

Matrix Alien Serenity Casablanca Amelie

SciFi ↑ ↓ ↑ ↓ ↑ ↓

Romnce ↑ ↓ ↑ ↓ ↑ ↓

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

SciFi-concept Romance-concept

SVD – Example: Users-to-Movies

■ $A = U \Sigma V^T$ - example:

Matrix Alien Serenity Casablanca Amelie

SciFi

Romnce

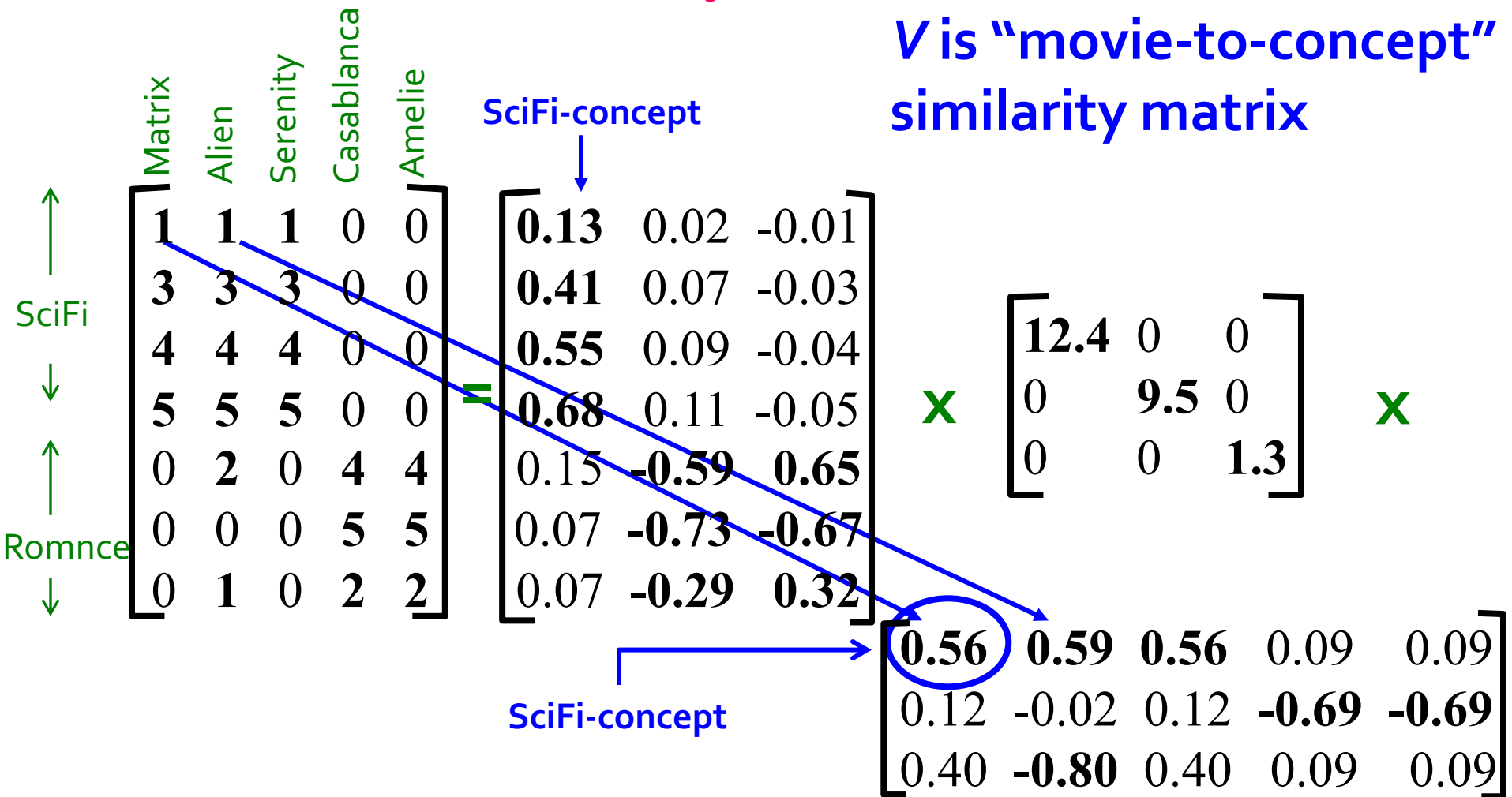
SciFi-concept

"strength/size" of the SciFi-concept

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

SVD – Example: Users-to-Movies

■ $A = U \Sigma V^T$ - example:

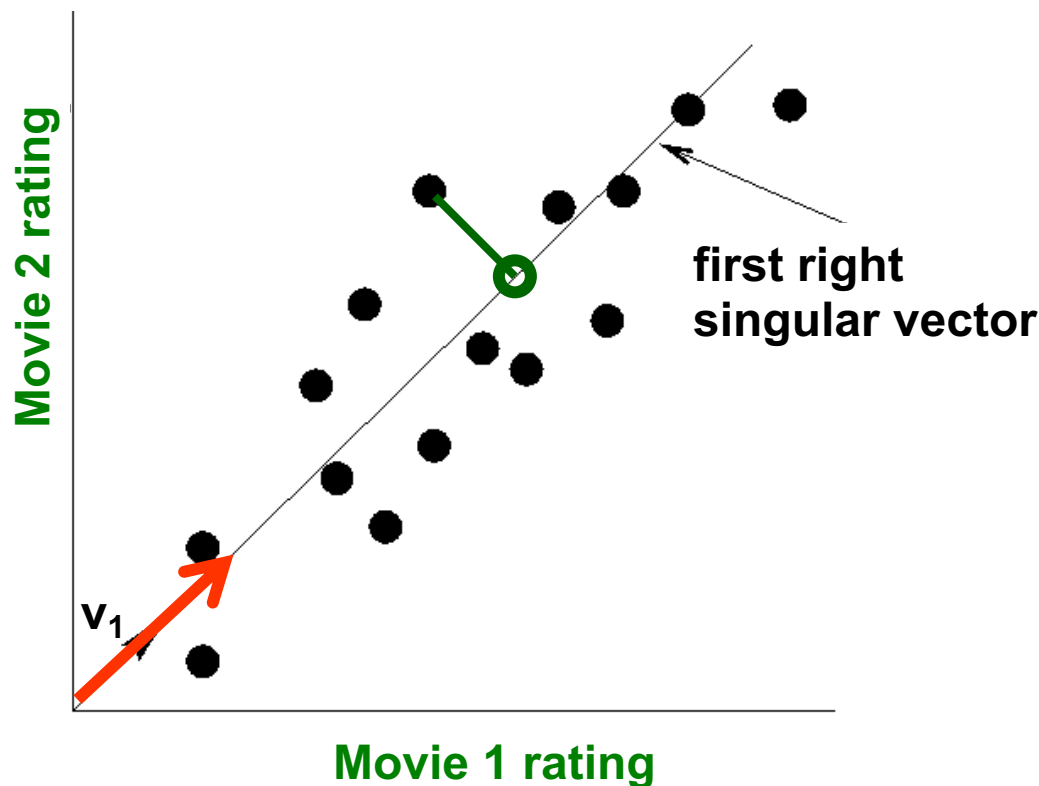


SVD - Interpretation #1

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

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

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**

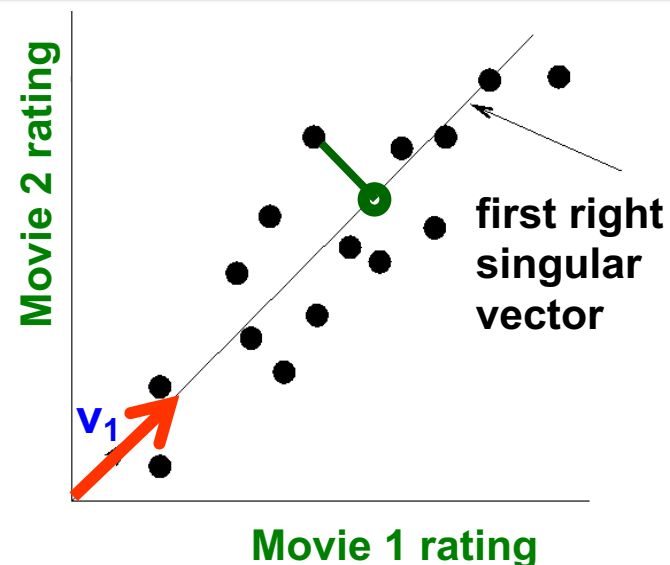
SVD – Dimensionality Reduction

- **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**



SVD - Interpretation #2

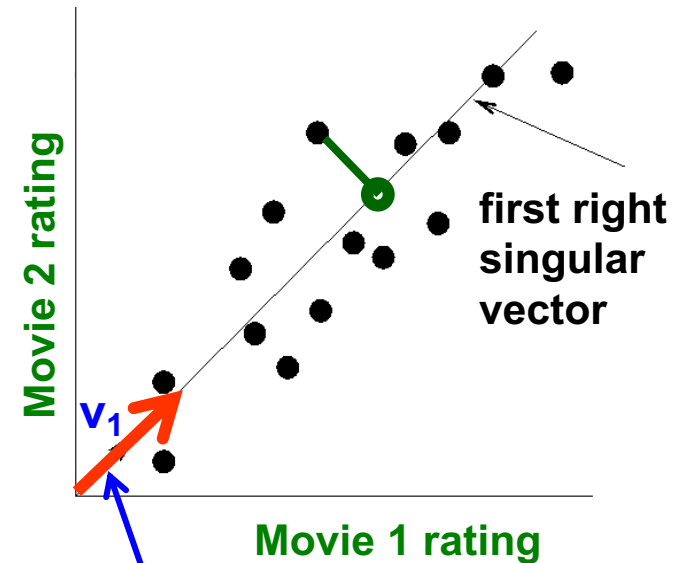
■ $A = U \Sigma V^T$ - example:

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

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

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

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



SVD - Interpretation #2

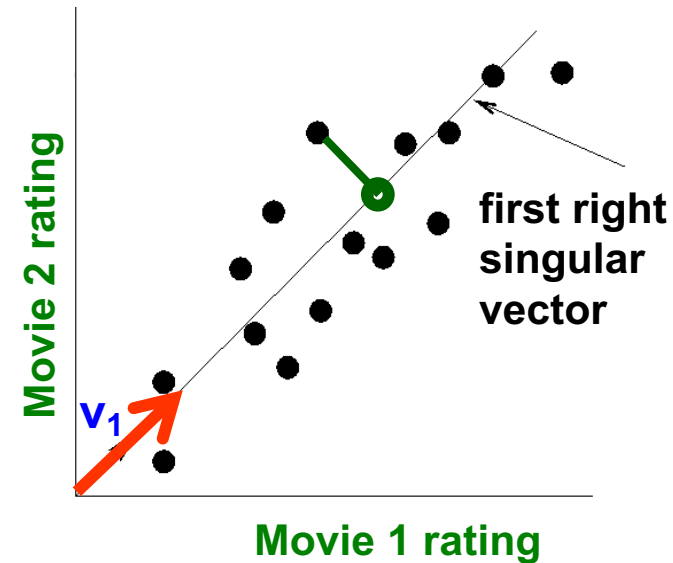
■ $A = U \Sigma V^T$ - example:

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

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

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

stdev ('spread')
on the v_1 axis



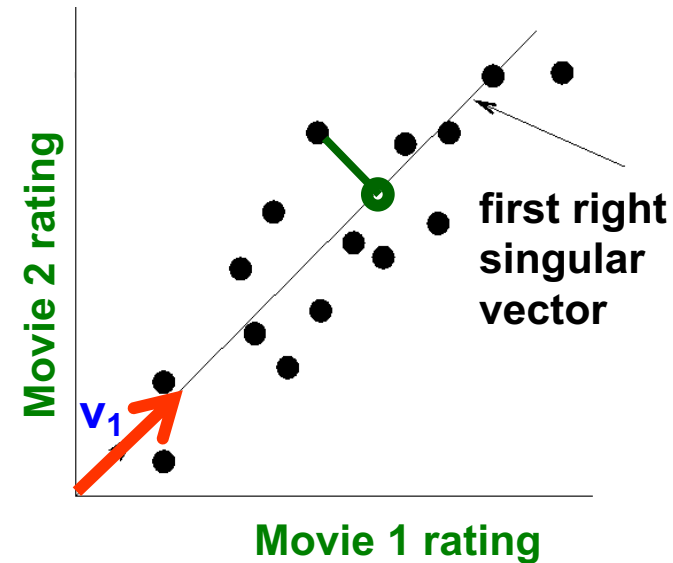
SVD - Interpretation #2

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

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

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

**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?
- **A:** Set smallest singular values to zero

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

The diagram illustrates the SVD decomposition of a matrix. The first matrix is a 7x5 matrix. The second matrix is a 7x3 matrix, with its third column crossed out by a red 'X'. The third matrix is a 3x3 diagonal matrix, with its third diagonal element (1.3) crossed out by a red 'X'. The fourth matrix is a 3x5 matrix, with its third row crossed out by a red 'X'.

SVD - Interpretation #2

More details

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

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

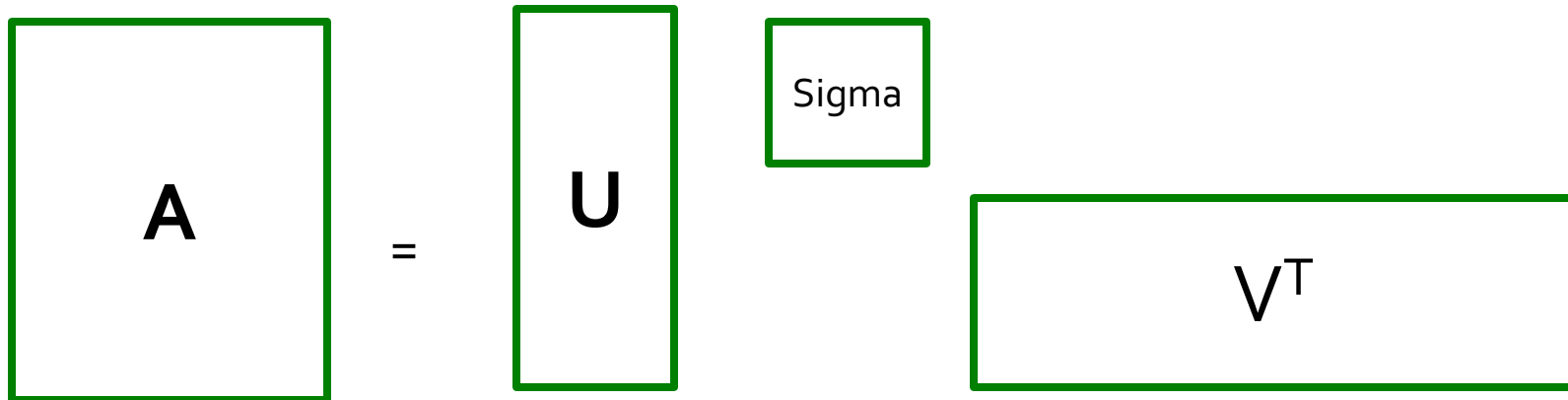
Frobenius norm:

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

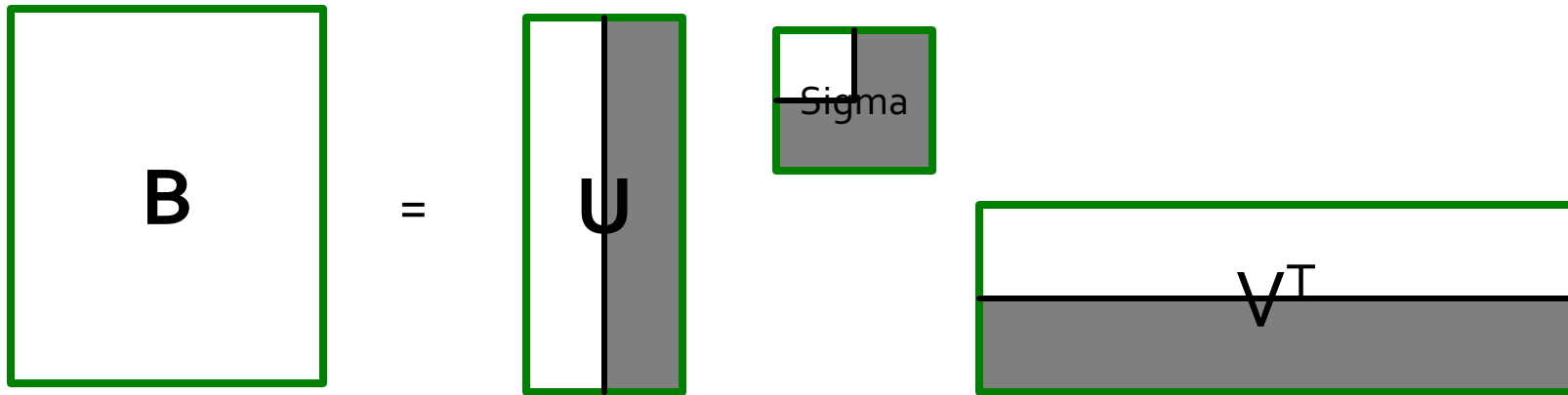
$$\|A-B\|_F = \sqrt{\sum_{ij} (A_{ij}-B_{ij})^2}$$

is “small”

SVD – Best Low Rank Approx.



B is best approximation of A



SVD – Best Low Rank Approx.

■ Theorem:

Let $A = U \Sigma V^T$ and $B = U S V^T$ where

$S = \text{diagonal } r \times r \text{ matrix}$ with $s_i = \sigma_i$ ($i=1 \dots k$) else $s_i=0$

then B is a **best rank(B)= k approx. to A**

What do we mean by “best”:

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

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

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

SVD - Interpretation #2

Q: How many σ_s to keep?

A: Rule-of-a thumb:

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

$$\begin{array}{c} \updownarrow n \\ \left[\begin{array}{ccccc} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{array} \right] \end{array} \quad \begin{array}{c} \longleftarrow m \longrightarrow \\ = \end{array} \quad \sigma_1 \quad \mathbf{U}_1 \quad \mathbf{V}_1^T + \sigma_2 \quad \mathbf{U}_2 \quad \mathbf{V}_2^T + \dots$$

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

Case study: How to query?

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

Diagram illustrating the mapping of a query (Matrix) into a concept space for finding similar items.

Matrix Data (Sci-Fi / Romance):

	Matrix	Alien	Serenity	Casablanca	Amelie
Sci-Fi	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
Romance	0	0	0	5	5
	0	1	0	2	2

Transformation:

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

Resulting Concept Space (3x5 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

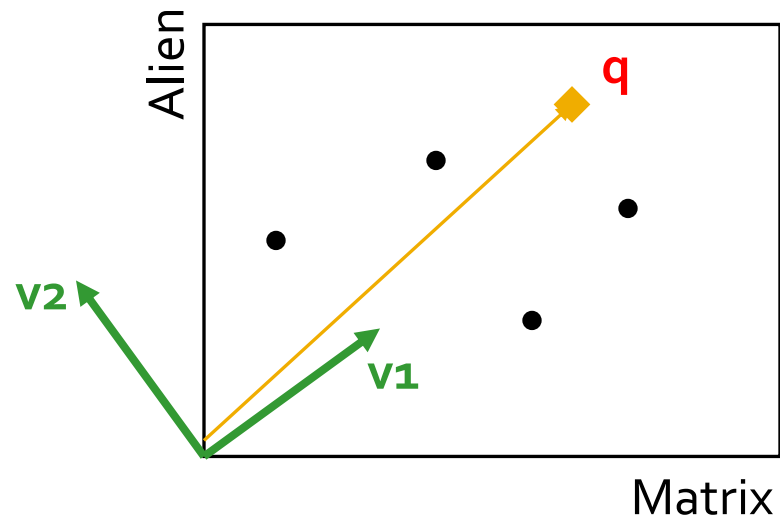
J. Leskovec, A. Rajaraman, J. Ullman: Mining of Massive Datasets, <http://www.mmms.org>

Case study: How to query?

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

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

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

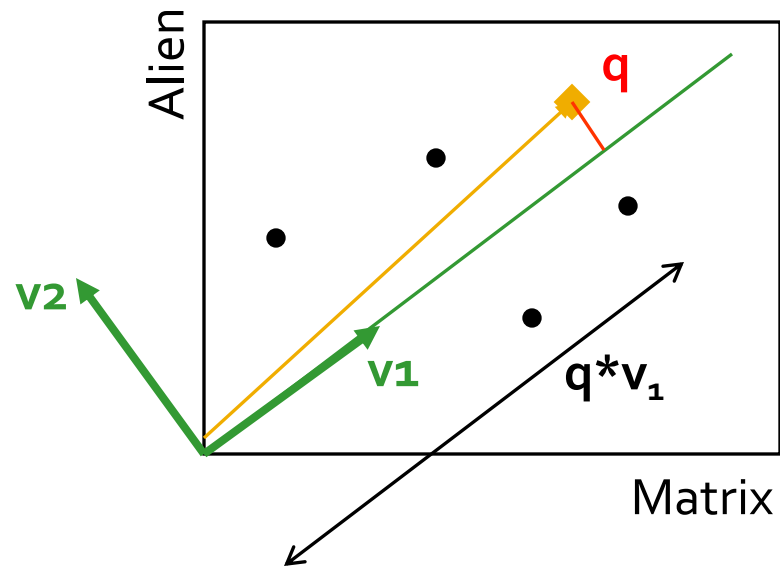


Case study: How to query?

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

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

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
↓

Case study: How to query?

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

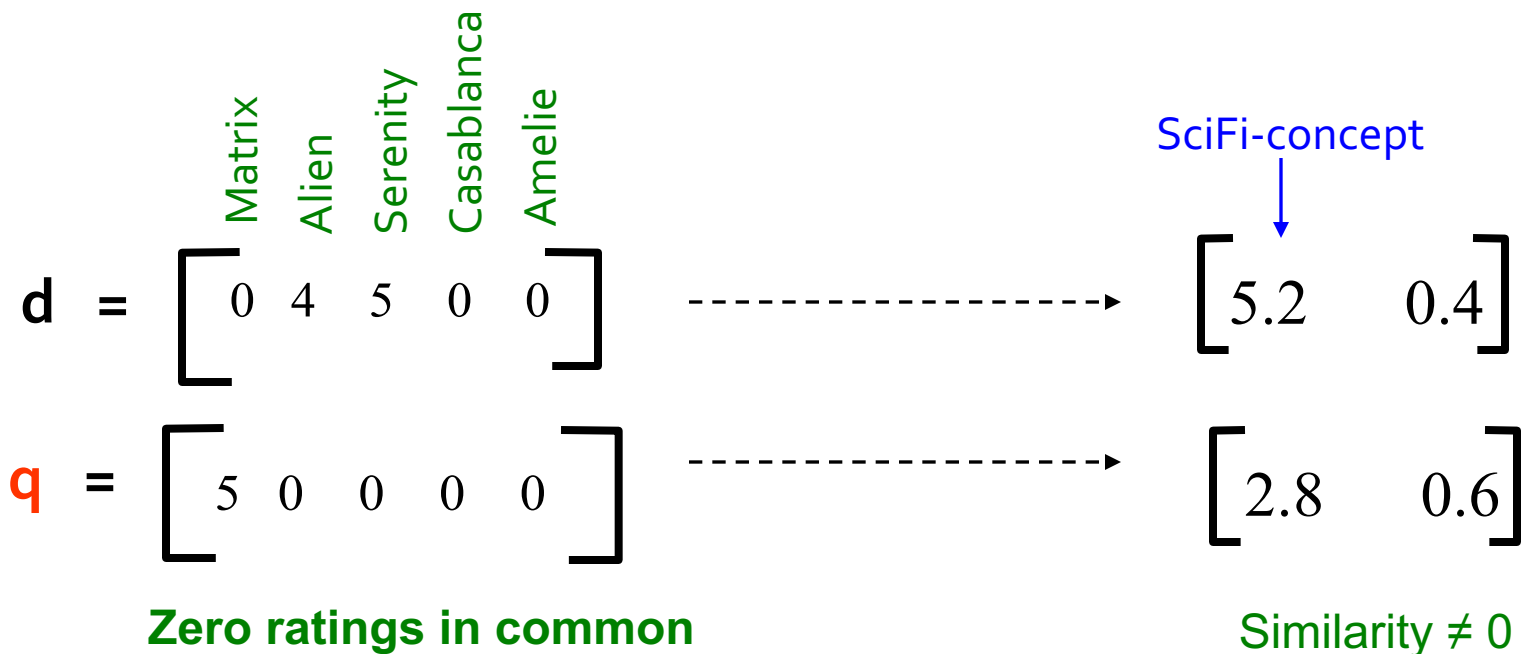
$$\mathbf{d}_{\text{concept}} = \mathbf{d} \mathbf{V}$$

E.g.:

$$\mathbf{q} = \begin{matrix} \text{Matrix} \\ \text{Alien} \\ \text{Serenity} \\ \text{Casablanca} \\ \text{Amelie} \end{matrix} \begin{bmatrix} 0 & 4 & 5 & 0 & 0 \end{bmatrix} \mathbf{x} \begin{matrix} \begin{bmatrix} 0.56 & 0.12 \\ 0.59 & -0.02 \\ 0.56 & 0.12 \\ 0.09 & -0.69 \\ 0.09 & -0.69 \end{bmatrix} \\ \text{movie-to-concept} \\ \text{similarities (V)} \end{matrix} = \begin{matrix} \text{SciFi-concept} \\ \downarrow \\ \begin{bmatrix} 5.2 & 0.4 \end{bmatrix} \end{matrix}$$

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!**

The diagram illustrates the SVD decomposition of a matrix. On the left is a sparse matrix represented by a rectangle containing six black dots. This is followed by an equals sign. To the right of the equals sign is a tall, narrow rectangle labeled U at the bottom, which is densely packed with black dots. To the right of U is a small square labeled Σ at the bottom, containing a few black dots. To the right of Σ is a wide, short rectangle labeled V^T at the bottom, which is also densely packed with black dots.

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 :

$$\left(\begin{array}{c} \text{red bar} \\ \text{blue bar} \\ \text{dark red bar} \end{array} \right) \approx \left(\begin{array}{c} \text{red bar} \\ \text{red bar} \\ \text{red bar} \\ \text{blue bar} \\ \text{dark red bar} \\ \text{dark red bar} \end{array} \right) \cdot \left(\begin{array}{c} U \end{array} \right) \cdot \left(\begin{array}{c} R \end{array} \right)$$

$A \qquad C \qquad U \qquad R$

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:

The diagram illustrates the CUR decomposition of matrix A. Matrix A is shown on the left as a vertical stack of three horizontal bars: red, brown, and blue. It is followed by an approximation symbol $\approx$. Matrix C is shown next as a vertical stack of three horizontal bars: red, brown, and blue. This is followed by a dot operator $\cdot$. Matrix U is shown next as a light gray square. A red arrow points from the text 'Pseudo-inverse of the intersection of C and R' to matrix U. This is followed by another dot operator $\cdot$. Matrix R is shown on the right as a horizontal stack of three horizontal bars: red, brown, and blue. Below each matrix is its label: A, C, U, and R.

$$\begin{pmatrix} \text{red bar} \\ \text{brown bar} \\ \text{blue bar} \end{pmatrix} \approx \begin{pmatrix} \text{red bar} \\ \text{brown bar} \\ \text{blue bar} \end{pmatrix} \cdot \begin{pmatrix} U \end{pmatrix} \cdot \begin{pmatrix} \text{red bar} \\ \text{brown bar} \\ \text{blue bar} \end{pmatrix}$$

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

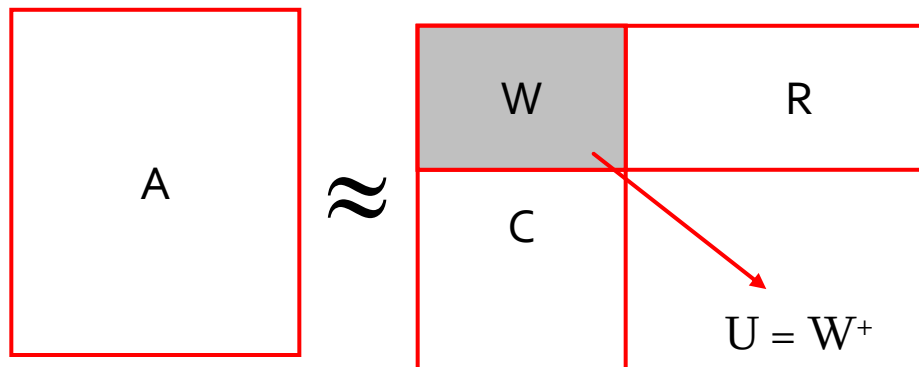
Output: $\mathbf{C}_d \in \mathbb{R}^{m \times c}$

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



Why pseudoinverse works?

$\mathbf{W} = \mathbf{X} \mathbf{Z} \mathbf{Y}$ then $\mathbf{W}^{-1} = \mathbf{X}^{-1} \mathbf{Z}^{-1} \mathbf{Y}^{-1}$

Due to orthonormality

$\mathbf{X}^{-1} = \mathbf{X}^T$ and $\mathbf{Y}^{-1} = \mathbf{Y}^T$

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

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

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 **ColumnSelect** algorithm

- Set $U = W^+$

- **Then:** $\|A - \overset{\text{CUR error}}{CUR}\|_F \leq (2 + \epsilon) \overset{\text{SVD error}}{\|A - A_k\|_F}$
with probability 98%

In practice:

Pick $4k$ cols/rows

for a “rank- k ” approximation

CUR: Pros & Cons

+ Easy interpretation

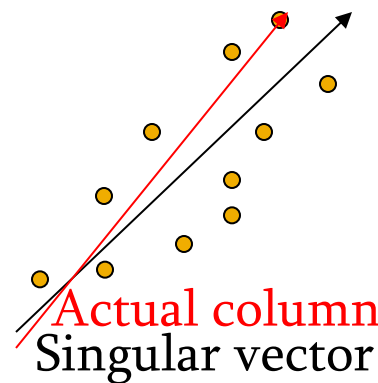
- Since the basis vectors are actual columns and rows

+ Sparse basis

- Since the basis vectors are actual columns and rows

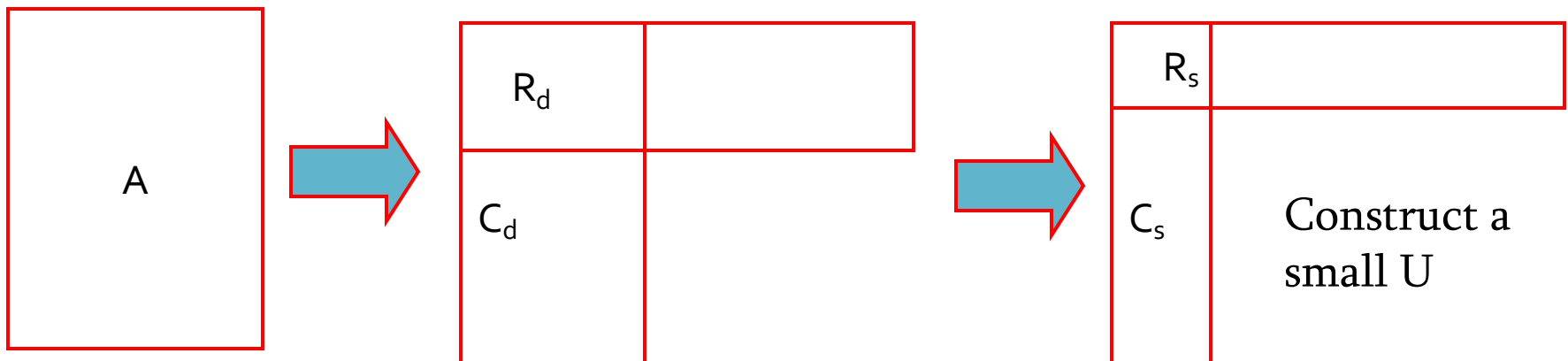
- Duplicate columns and rows

- 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

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

Annotations for SVD:

- A : Huge but sparse
- U : Big and dense
- Σ : sparse and small
- V^T : Big and dense

CUR: $A = C U R$

Annotations for CUR:

- A : Huge but sparse
- C : Big but sparse
- U : dense but small
- R : Big but sparse

SVD vs. CUR: Simple Experiment

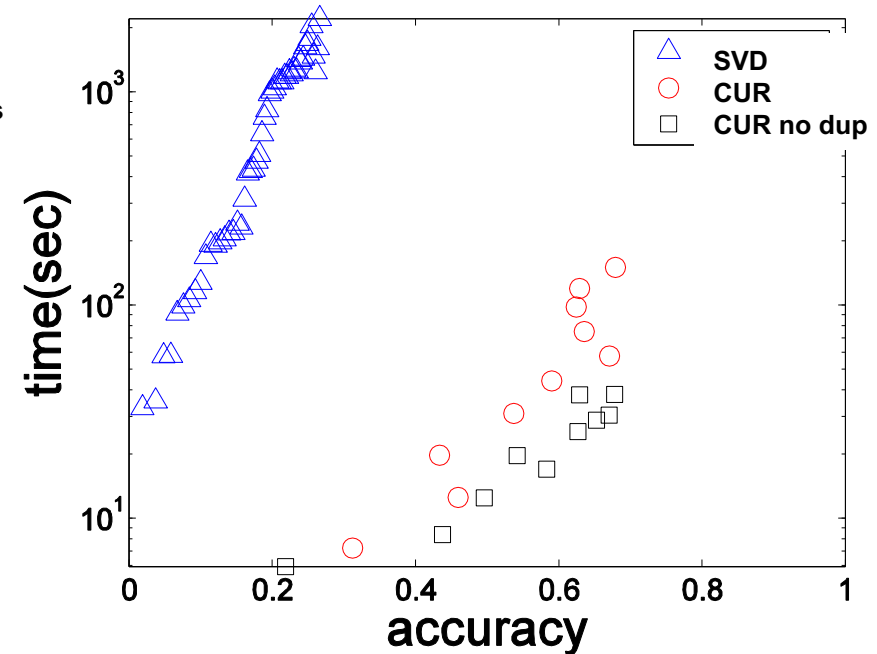
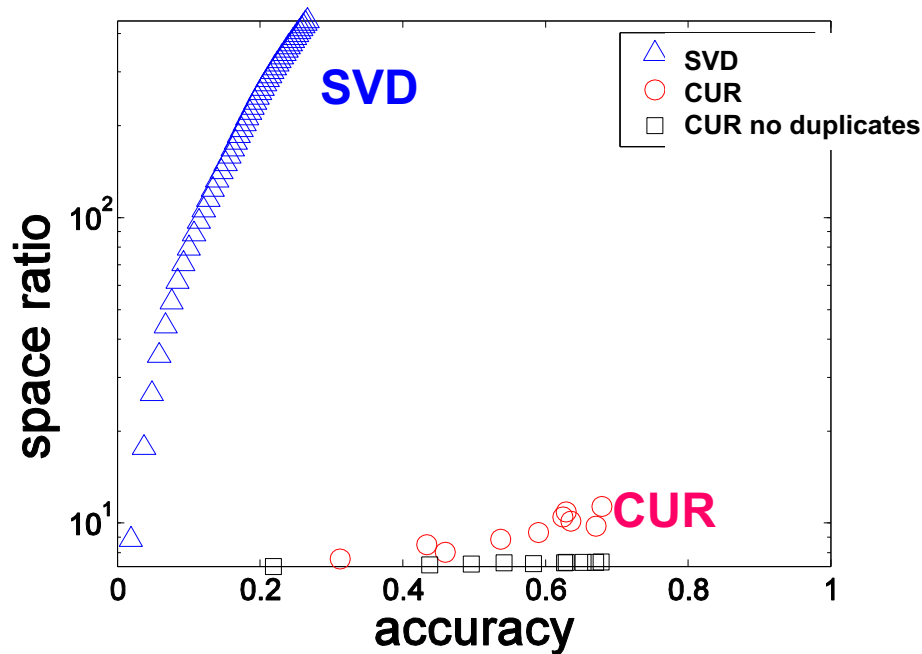
■ DBLP bibliographic data

- 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



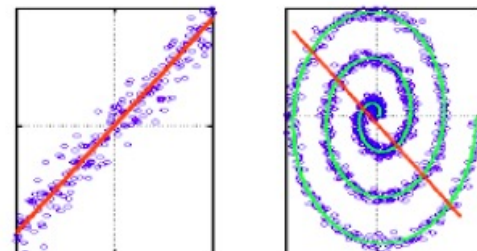
- **Accuracy:**
 - 1 – relative sum squared errors
- **Space ratio:**
 - #output matrix entries / #input matrix entries
- **CPU time**

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

What about linearity assumption?

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

- Lower-dimensional linear projection that preserves Euclidean distances

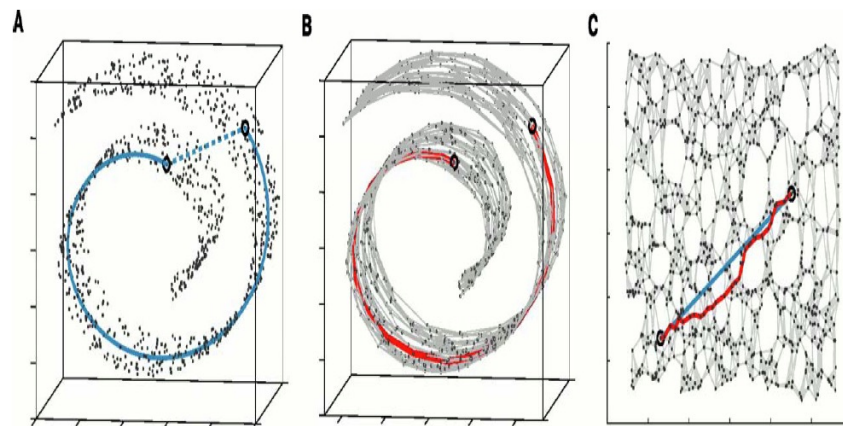


- **Non-linear methods: Manifold Learning**

- 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



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)