Dimensionality Reduction: SVD & CUR

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

Dimensionality Reduction

- High-dimension == many features
- Find concepts/topics/genres:
  - Documents:
    - Features: thousands of words, millions of word pairs

term / document	data	information	retrieval	brain	lung
CS-TR1	1	1	1	0	0
CS-TR2	2	2	2	0	0
CS-TR3	1	1	1	0	0
CS-TR4	5	5	5	0	0
MED-TR1	0	0	0	2	2
MED-TR2	0	0	0	3	3
MED-TR3	0	0	0	1	1

  - Surveys – Netflix: 480k users x 177k movies

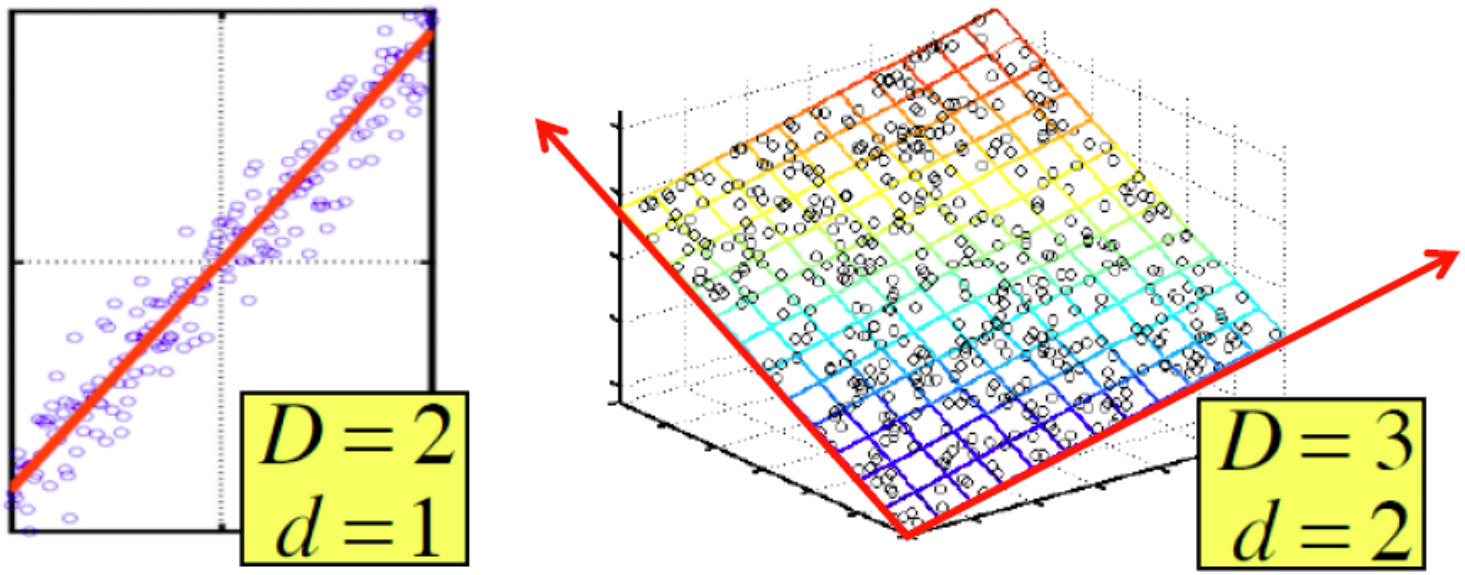
	movie 1	movie 2	movie 3	movie 4	movie 5	movie 6
Tom	5	?	?	1	3	?
George	?	?	3	1	2	5
Susan	4	3	1	?	5	1
Beth	4	3	?	2	4	2

Dimensionality Reduction

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

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

Dimensionality Reduction



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

Why Reduce Dimensions

- Why reduce dimensionality?
  - 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

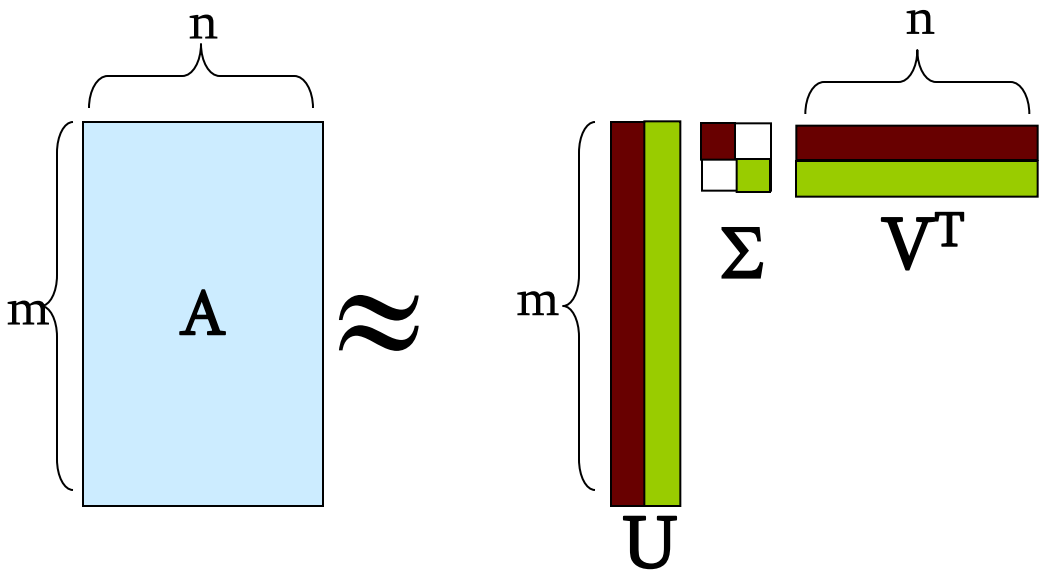
SVD - Definition

$$A_{[n \times m]} = U_{[n \times r]} \Sigma_{[r \times r]} (V_{[m \times r]})^T$$

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

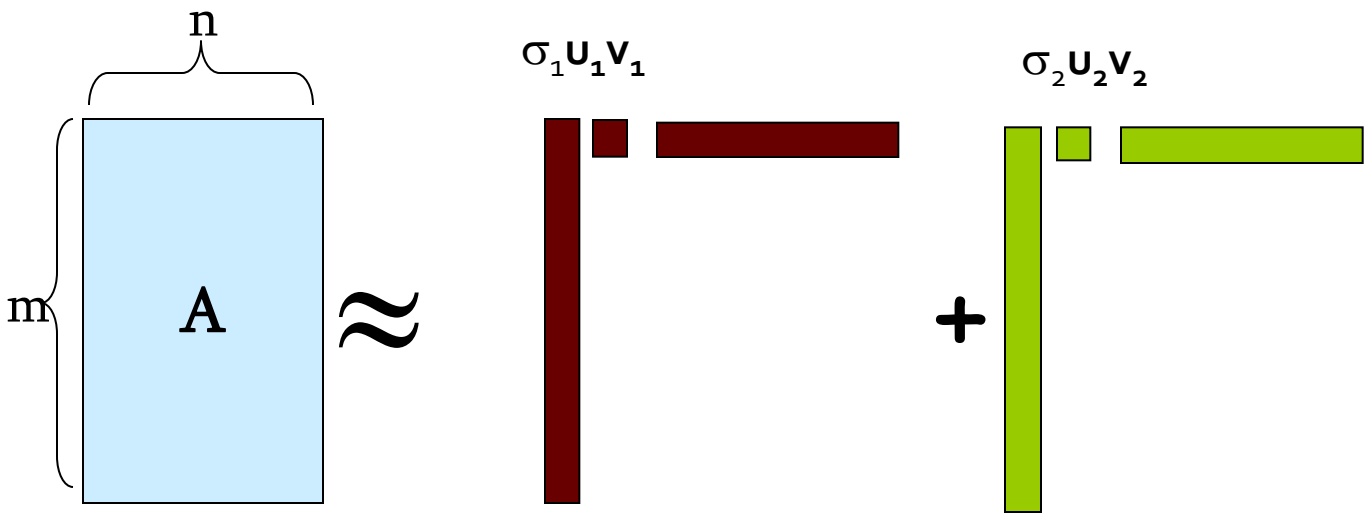
SVD

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



SVD

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



σ_i … scalar
u_i … vector
v_i … vector

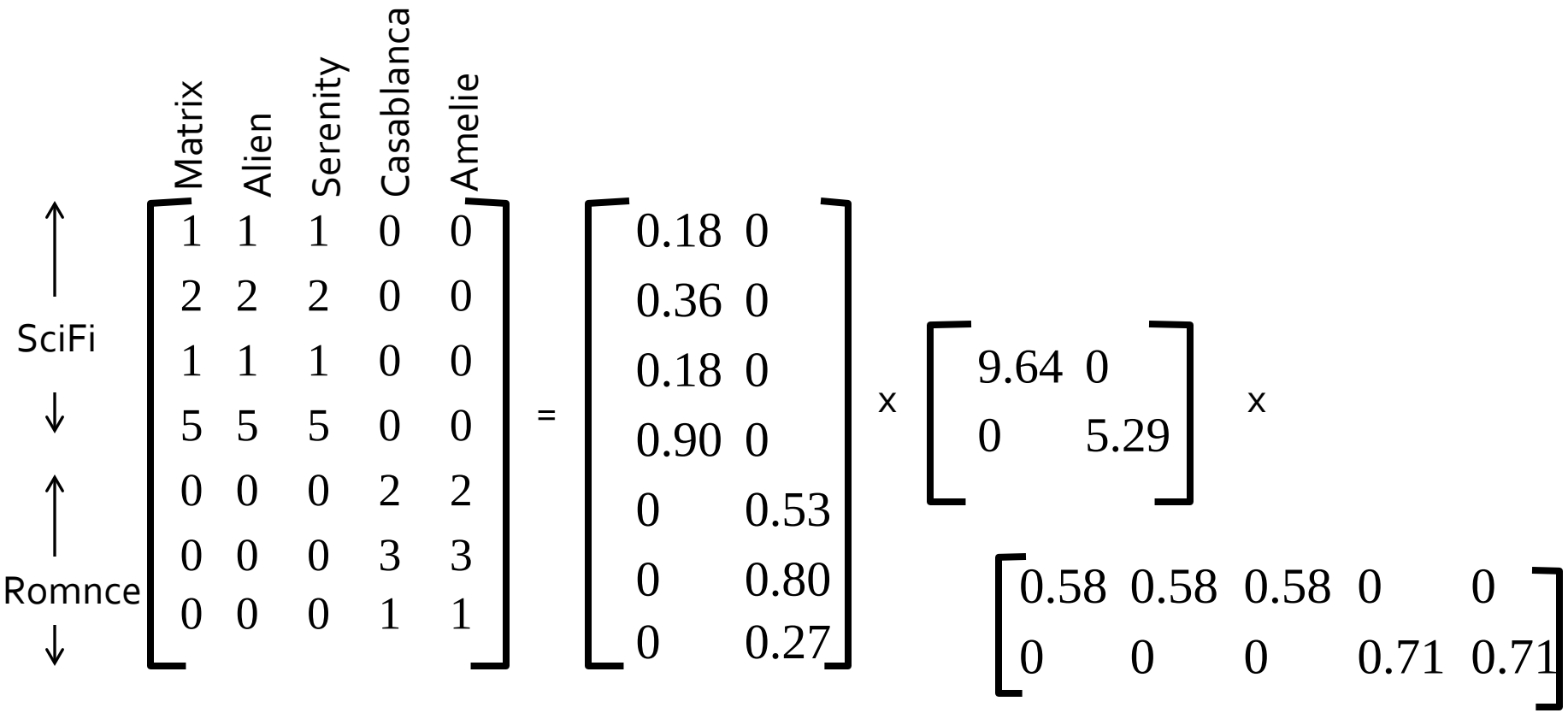
SVD - Properties

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

- **U, Σ, V**: unique
- **U, V**: column orthonormal:
  - $\mathbf{U}^T\mathbf{U} = \mathbf{I}$; $\mathbf{V}^T\mathbf{V} = \mathbf{I}$ (**I**: identity matrix)
  - (Cols. are orthogonal unit vectors)
- **Σ**: diagonal
  - Entries (singular values) are positive, and sorted in decreasing order ($\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \ldots$)

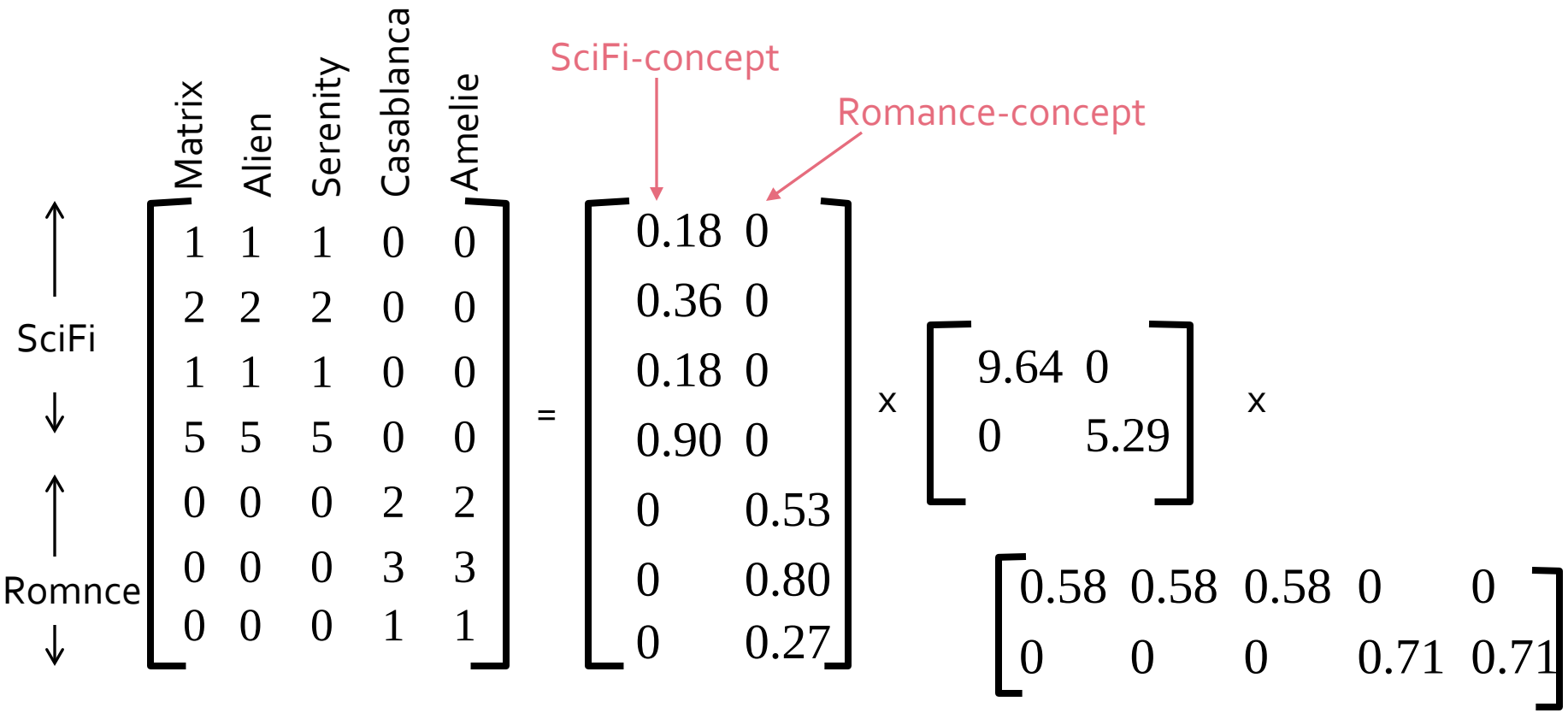
SVD – Example: Users-to-Movies

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



SVD – Example: Users-to-Movies

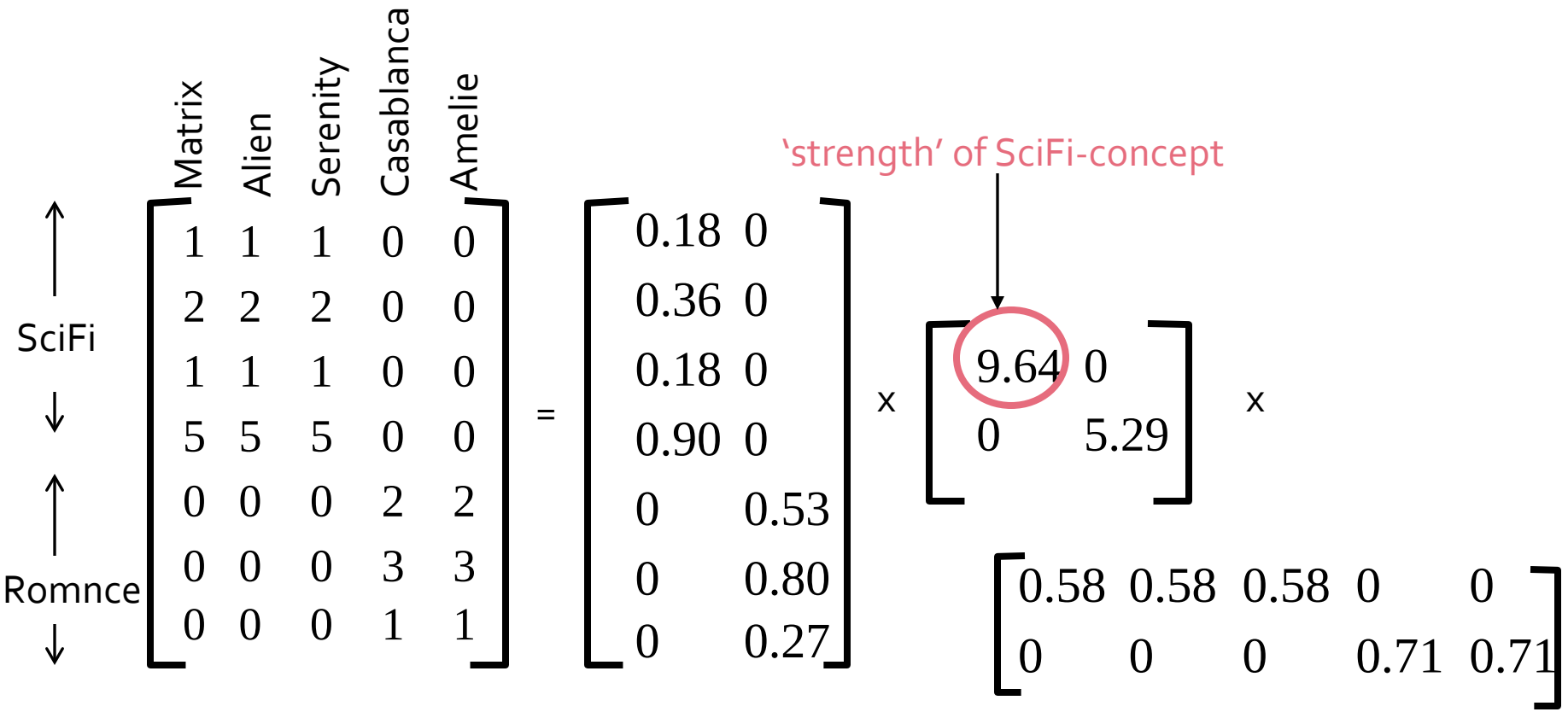
- $A = U \Sigma V^T$ - example:



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

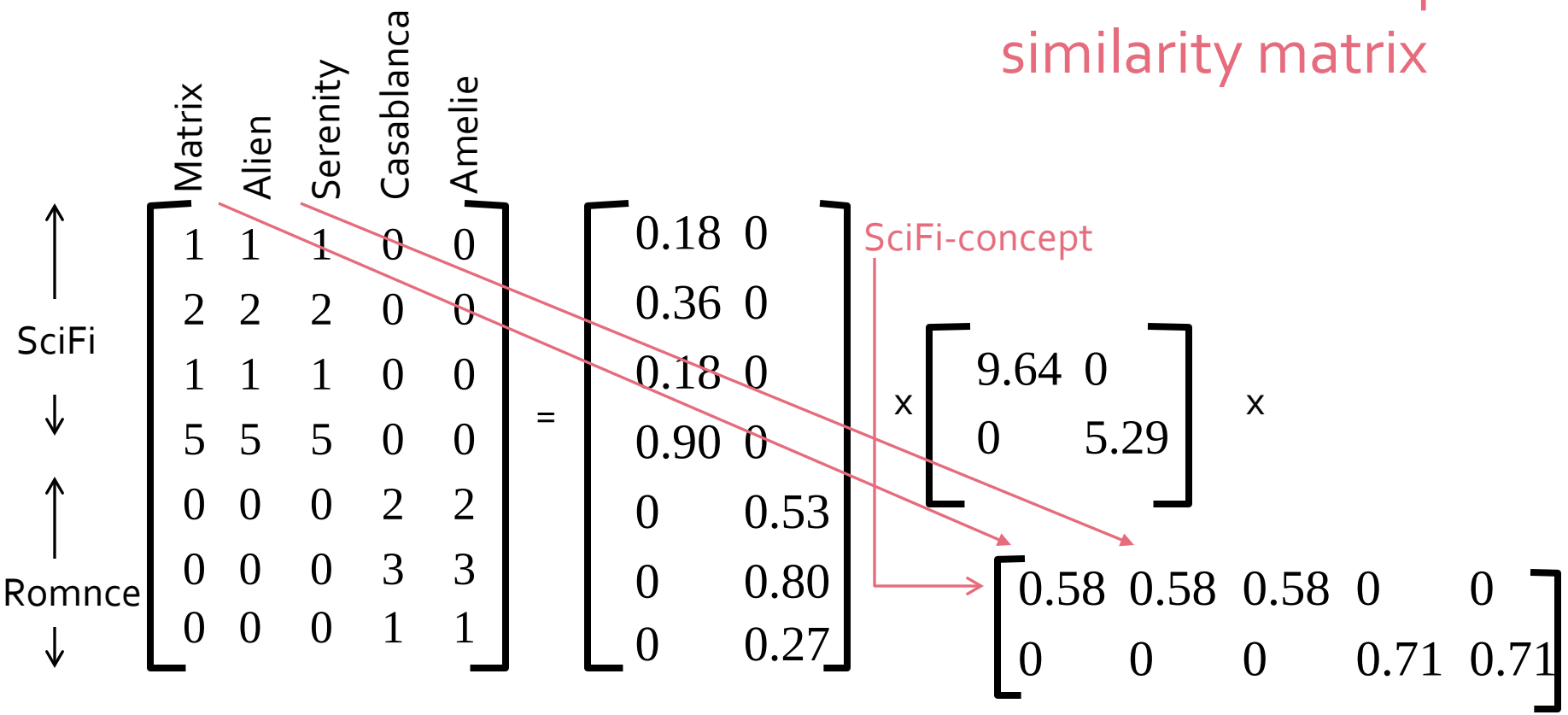
user-to-concept similarity matrix

- **A = U Σ V^T** - example:



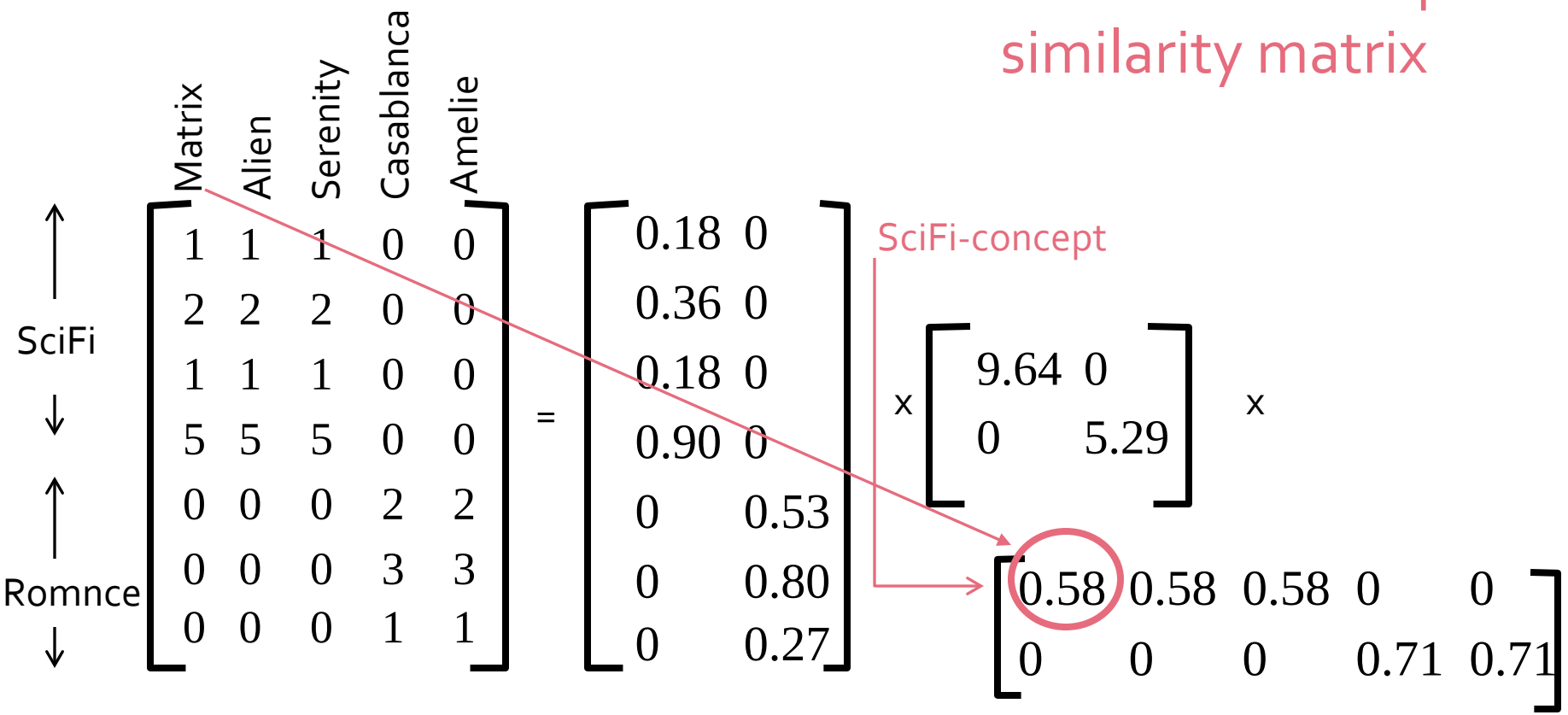
- **A = U Σ V^T** - example:

movie-to-concept similarity matrix



- **A = U Σ V^T** - example:

movie-to-concept similarity matrix



SVD - Interpretation #1

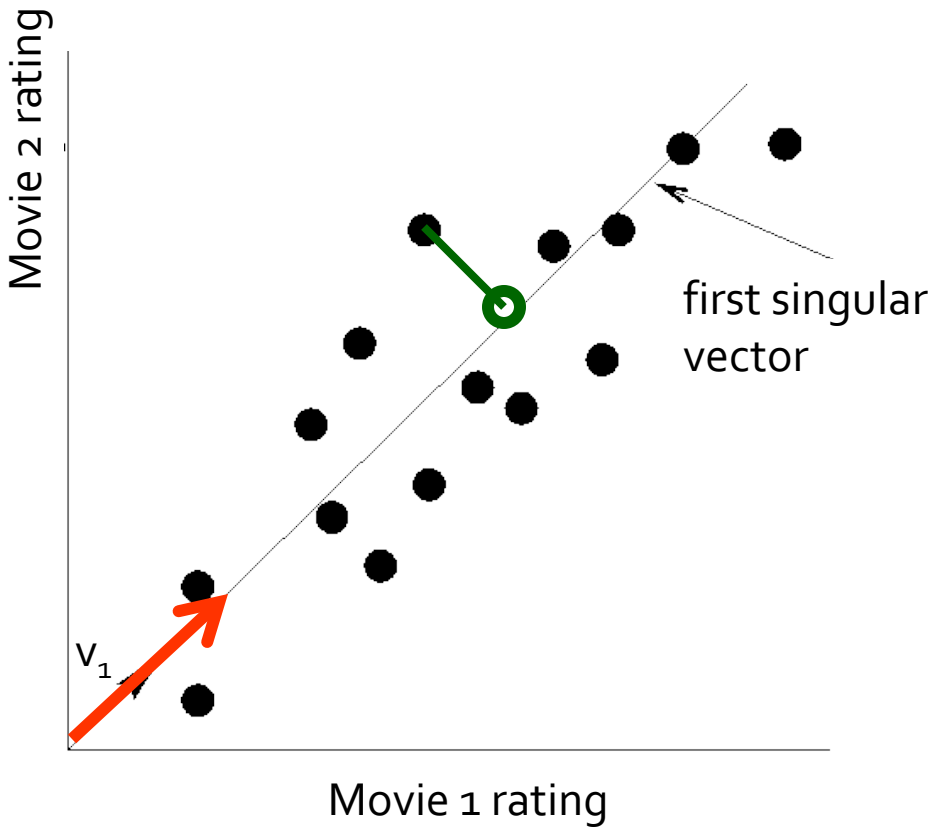
'movies', 'users' and 'concepts':

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

SVD - interpretation #2

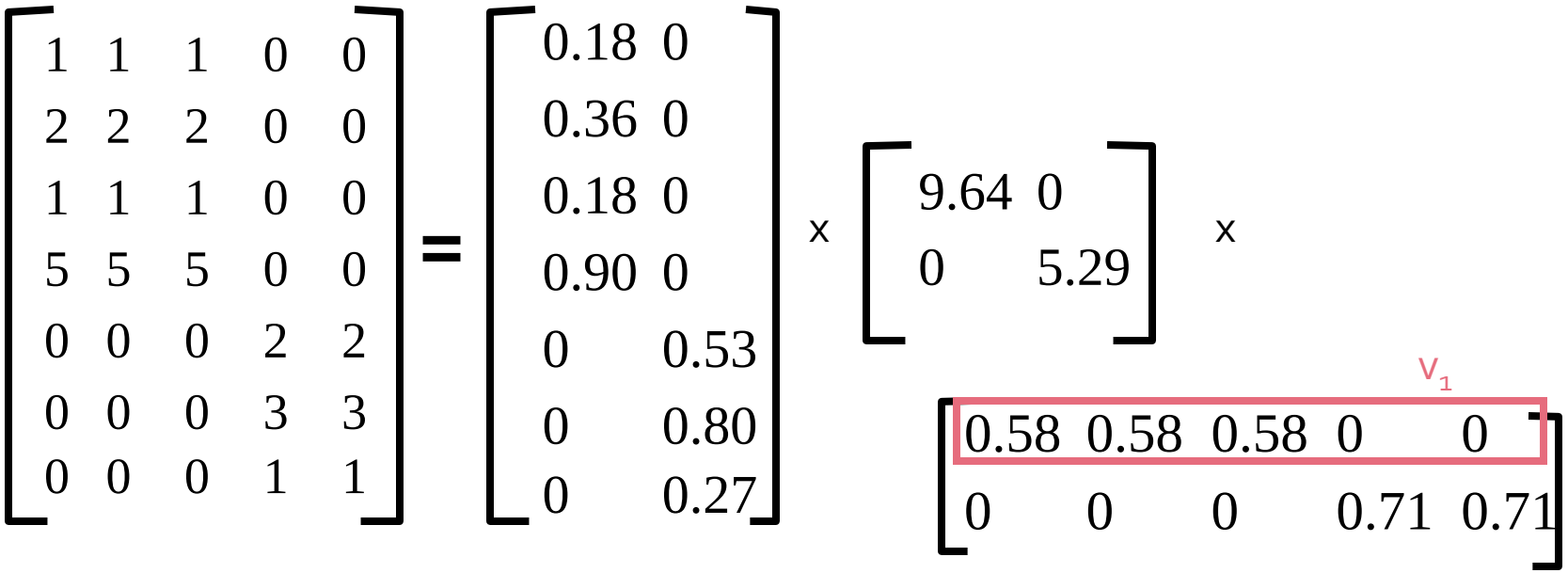
SVD gives best axis to project on:

- 'best' = min sum of squares of projection errors
- minimum reconstruction error



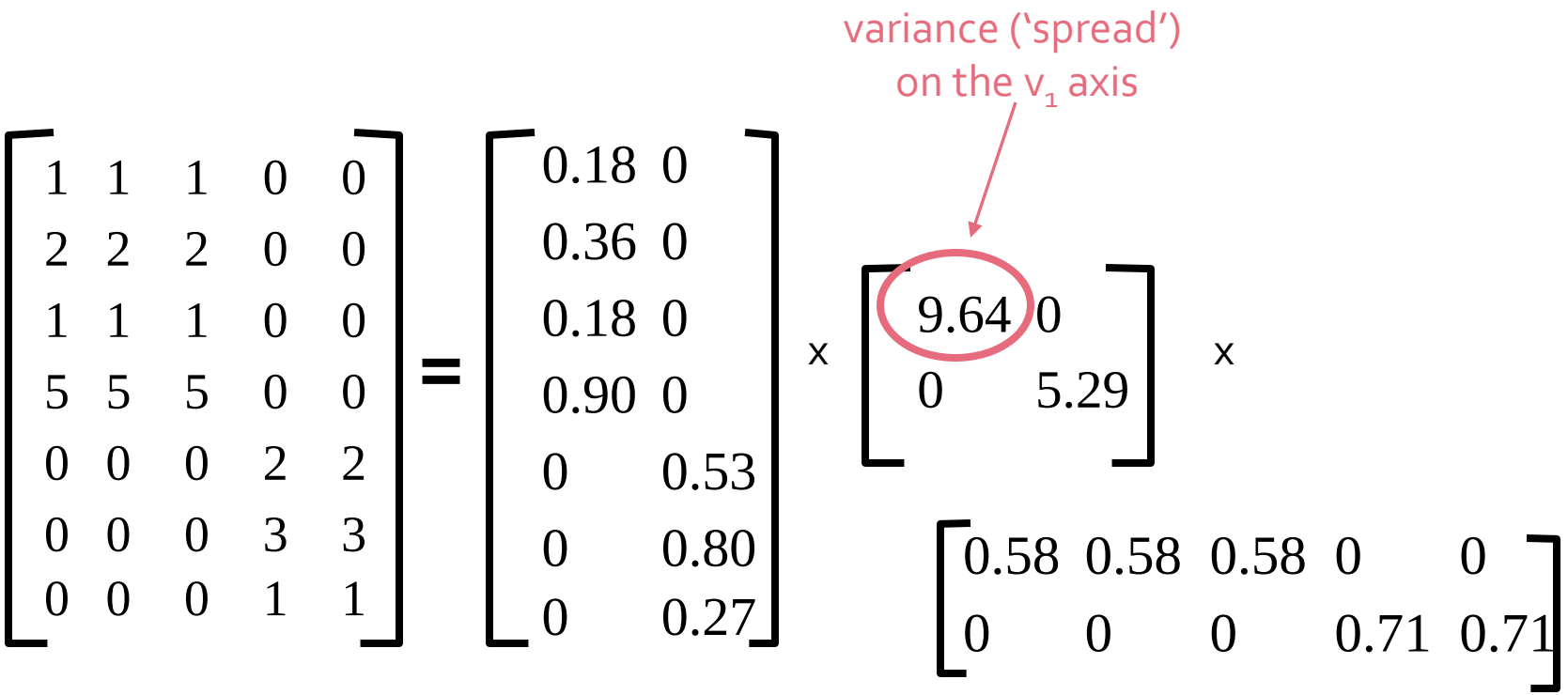
SVD - Interpretation #2

- **A = U Σ V^T** - example:



SVD - Interpretation #2

- **A = U Σ V^T** - example:



SVD - Interpretation #2

- **A** = **U** Σ $\mathbf{V}^T$ - example:
  - **U**Σ: gives the coordinates of the points in the projection axis

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 0.18 & 0 \\ 0.36 & 0 \\ 0.18 & 0 \\ 0.90 & 0 \\ 0 & 0.53 \\ 0 & 0.80 \\ 0 & 0.27 \end{bmatrix} \times \begin{bmatrix} 9.64 & 0 \\ 0 & 5.29 \end{bmatrix} \times \begin{bmatrix} 0.58 & 0.58 & 0.58 & 0 & 0 \\ 0 & 0 & 0 & 0.71 & 0.71 \end{bmatrix}$$

SVD - Interpretation #2

More details

- Q: How exactly is dim. reduction done?

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 0.18 & 0 \\ 0.36 & 0 \\ 0.18 & 0 \\ 0.90 & 0 \\ 0 & 0.53 \\ 0 & 0.80 \\ 0 & 0.27 \end{bmatrix} \times \begin{bmatrix} 9.64 & 0 \\ 0 & 5.29 \end{bmatrix} \times \begin{bmatrix} 0.58 & 0.58 & 0.58 & 0 & 0 \\ 0 & 0 & 0 & 0.71 & 0.71 \end{bmatrix}$$

SVD - Interpretation #2

More details

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

$$A = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 0.18 & 0 \\ 0.36 & 0 \\ 0.18 & 0 \\ 0.90 & 0 \\ 0 & 0.53 \\ 0 & 0.80 \\ 0 & 0.27 \end{bmatrix} \times \begin{bmatrix} 9.64 & 0 \\ 0 & 5.29 \end{bmatrix} \times \begin{bmatrix} 0.58 & 0.58 & 0.58 & 0 & 0 \\ 0 & 0 & 0 & 0.71 & 0.71 \end{bmatrix}$$

SVD - Interpretation #2

More details

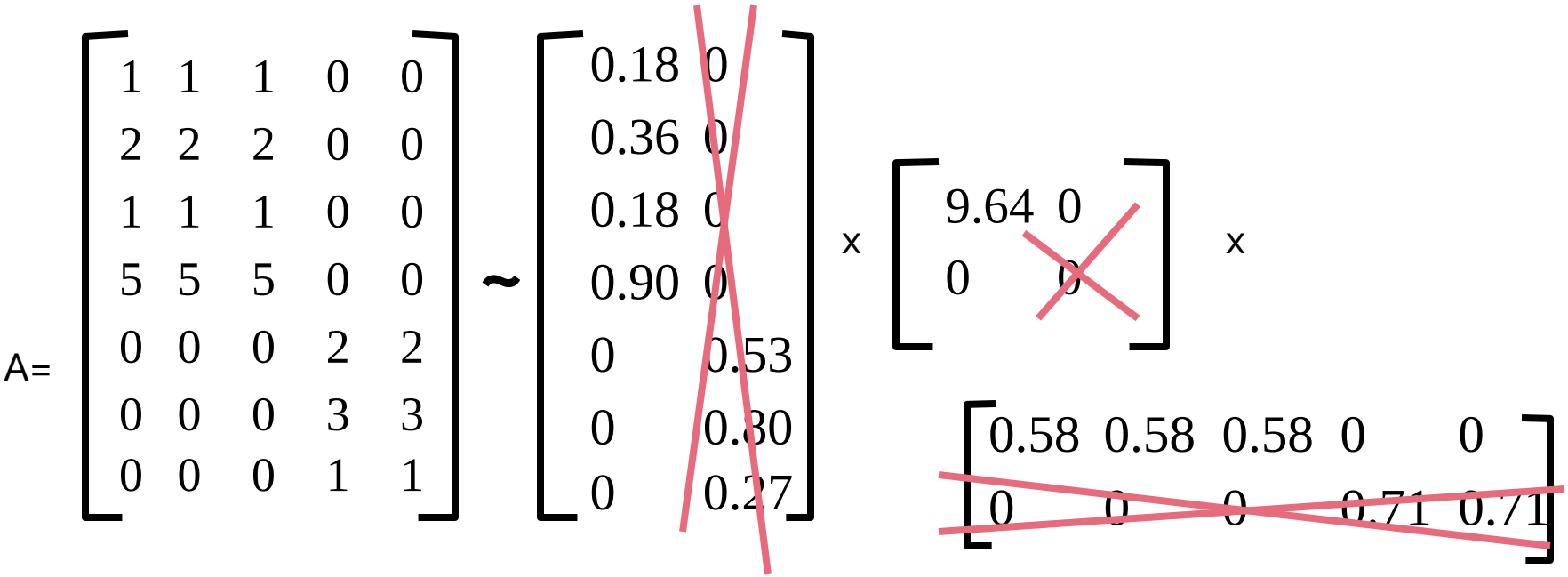
- Q: How exactly is dim. reduction done?
- A: Set the smallest singular values to zero

$$A = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \sim \begin{bmatrix} 0.18 & 0 \\ 0.36 & 0 \\ 0.18 & 0 \\ 0.90 & 0 \\ 0 & 0.53 \\ 0 & 0.80 \\ 0 & 0.27 \end{bmatrix} \times \begin{bmatrix} 9.64 & 0 \\ 0 & 0 \end{bmatrix} \times \begin{bmatrix} 0.58 & 0.58 & 0.58 & 0 & 0 \\ 0 & 0 & 0 & 0.71 & 0.71 \end{bmatrix}$$

SVD - Interpretation #2

More details

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



SVD - Interpretation #2

More details

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

$$A = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \sim \begin{bmatrix} 0.18 \\ 0.36 \\ 0.18 \\ 0.90 \\ 0 \\ 0 \\ 0 \end{bmatrix} \times \begin{bmatrix} 9.64 \end{bmatrix} \times \begin{bmatrix} 0.58 & 0.58 & 0.58 & 0 & 0 \end{bmatrix}$$

SVD - Interpretation #2

More details

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

$$A = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \sim B = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

SVD – best low rank approx

- Theorem: Let $A = U \Sigma V^T$ ($\sigma_1 \geq \sigma_2 \geq \ldots$, rank(A)=n) then $B = U S V^T$
  - S = diagonal nxn matrix where $s_i = \sigma_i$ (i=1…k) else $s_i = 0$

  is a best rank-k approximation to A:
  - B is solution to $\min_B \|A-B\|_F$ where rank(B)=k
- Why?

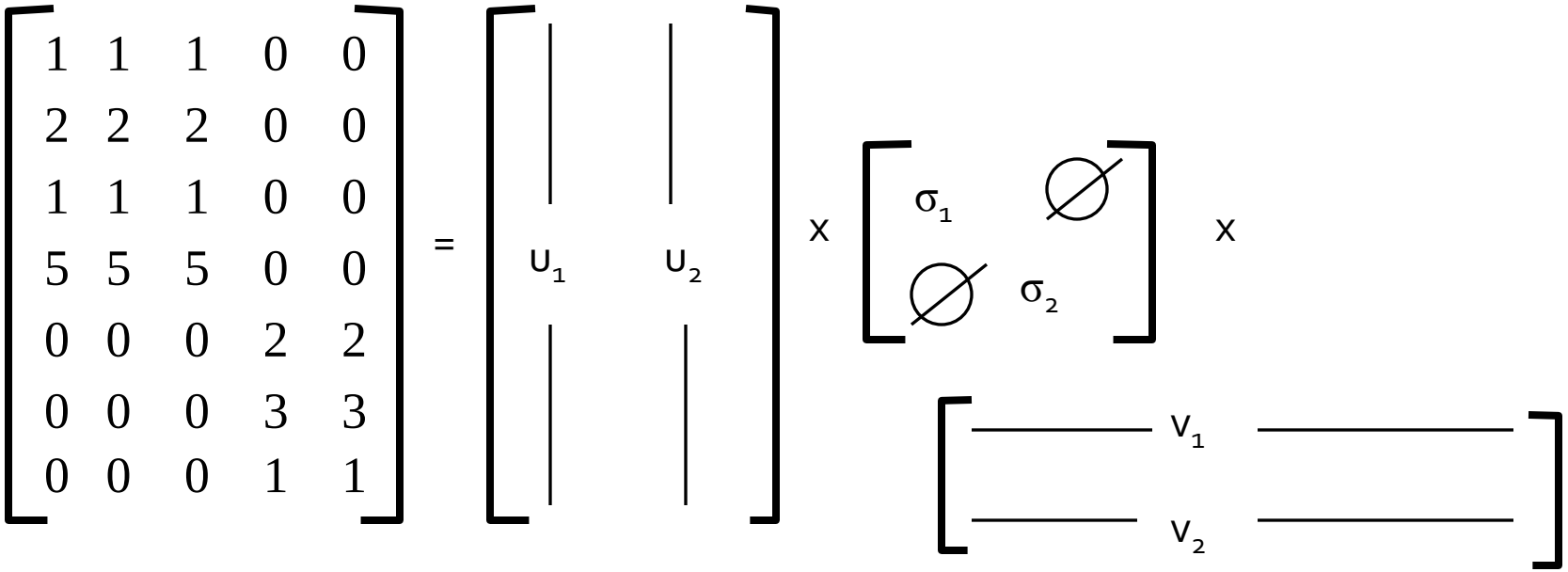
$$\min_{B,rank(B)=k} \|A-B\|_F = \min \|\Sigma - S\|_F = \min_{s_i} \sum_{i=1}^{n} (\sigma_i - s_i)^2$$

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

SVD - Interpretation #2

Equivalent:

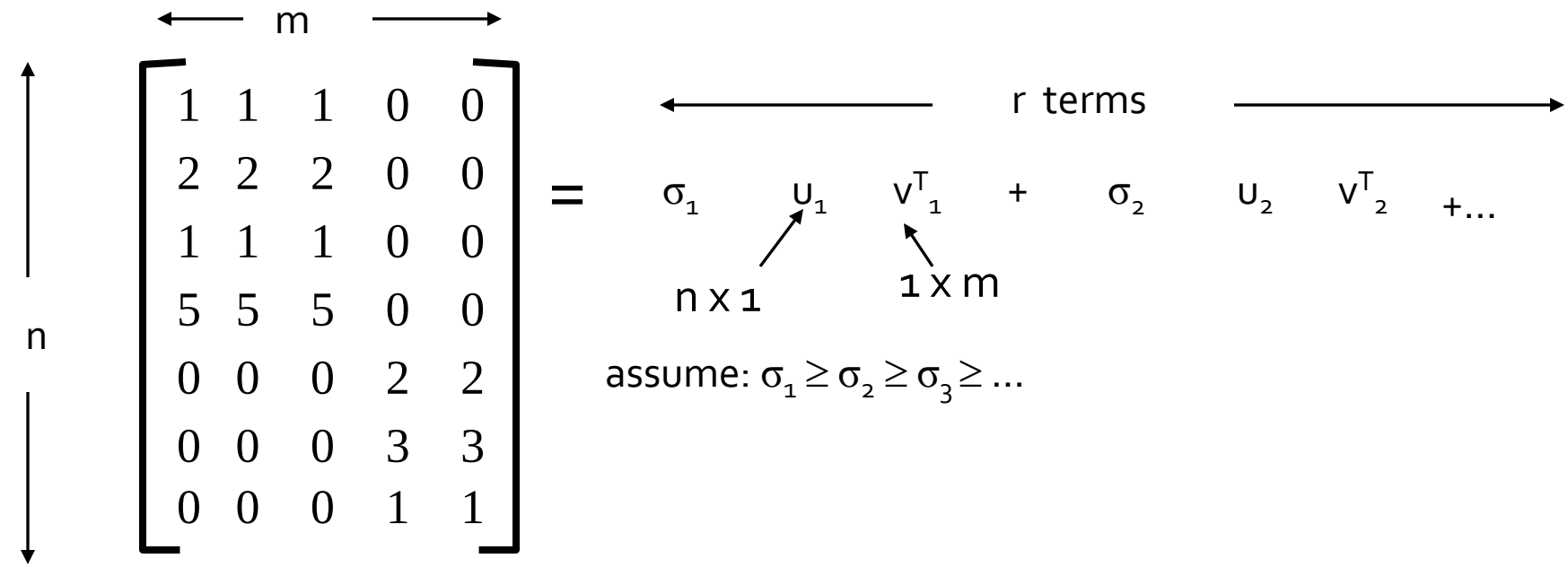
'spectral decomposition' of the matrix:



SVD - Interpretation #2

Equivalent:

'spectral decomposition' of the matrix:

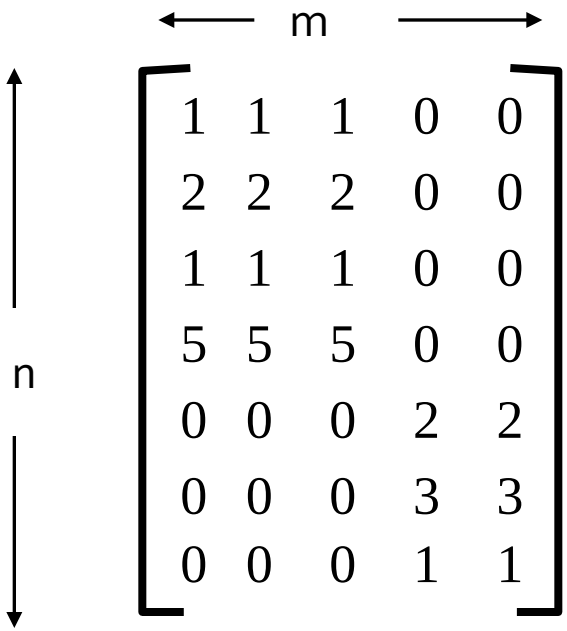


SVD - Interpretation #2

Q: How many σ_s to keep?

A: Rule-of-a thumb:

keep 80-90% of 'energy' ($=\sum\sigma_i^2$)



$$= \sigma_1 \; u_1 \; v^T_1 + \sigma_2 \; u_2 \; v^T_2 + \ldots$$

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

SVD - Complexity

- To compute SVD:
  - $O(nm^2)$ or $O(n^2m)$ (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:
  - Linear algebra packages like: LINPACK, Matlab, SPlus, Mathematica ...

SVD - conclusions so far

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

Relation to Eigen-decomposition

- SVD gives us:
  - $\boldsymbol{A = U\,\Sigma\,V^T}$
- Eigen-decomposition:
  - $\boldsymbol{A = X\,L\,X^T}$
    - A is symmetric
    - U, V, X are orthonormal ($U^TU=I$),
    - Λ, Σ are diagonal
- What is:
  - $\mathbf{AA^T = U\Sigma\,V^T(U\Sigma\,V^T)^T = U\Sigma\,V^T(V\Sigma^TU^T) = U\Sigma\Sigma^T\,U^T}$
  - $\mathbf{A^TA = V\Sigma^T\,U^T\,(U\Sigma\,V^T) = V\,\Sigma\Sigma^T\,V^T}$

$\boldsymbol{X\ \ L\ \ X^T}$ So, $\lambda_i = \sigma_i^2$

SVD: Properties

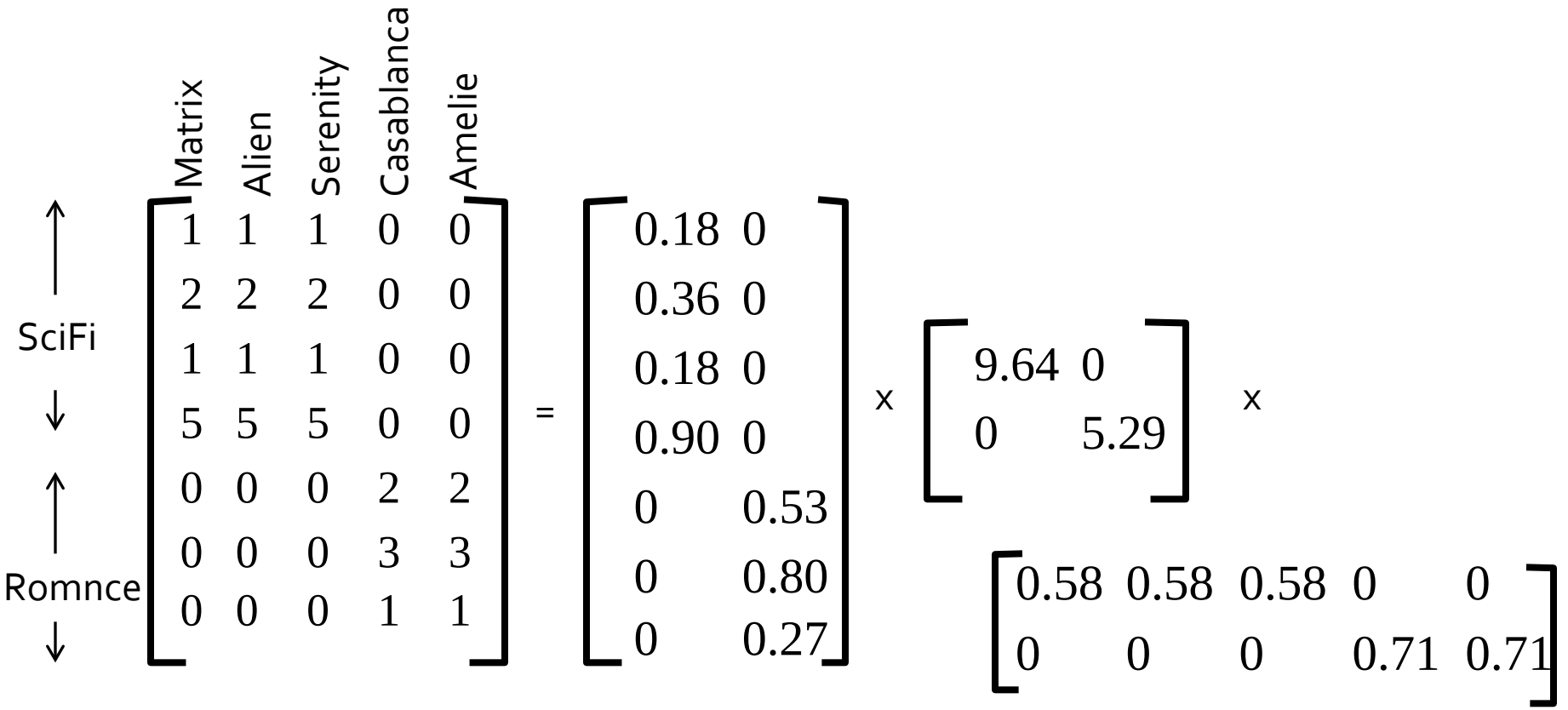
- $\mathbf{A}\,\mathbf{A}^T = \mathbf{U}\,\Sigma^2\,\mathbf{U}^T$
- $\mathbf{A}^T\mathbf{A} = \mathbf{V}\,\Sigma^2\,\mathbf{V}^T$

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

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

Case study: How to query?

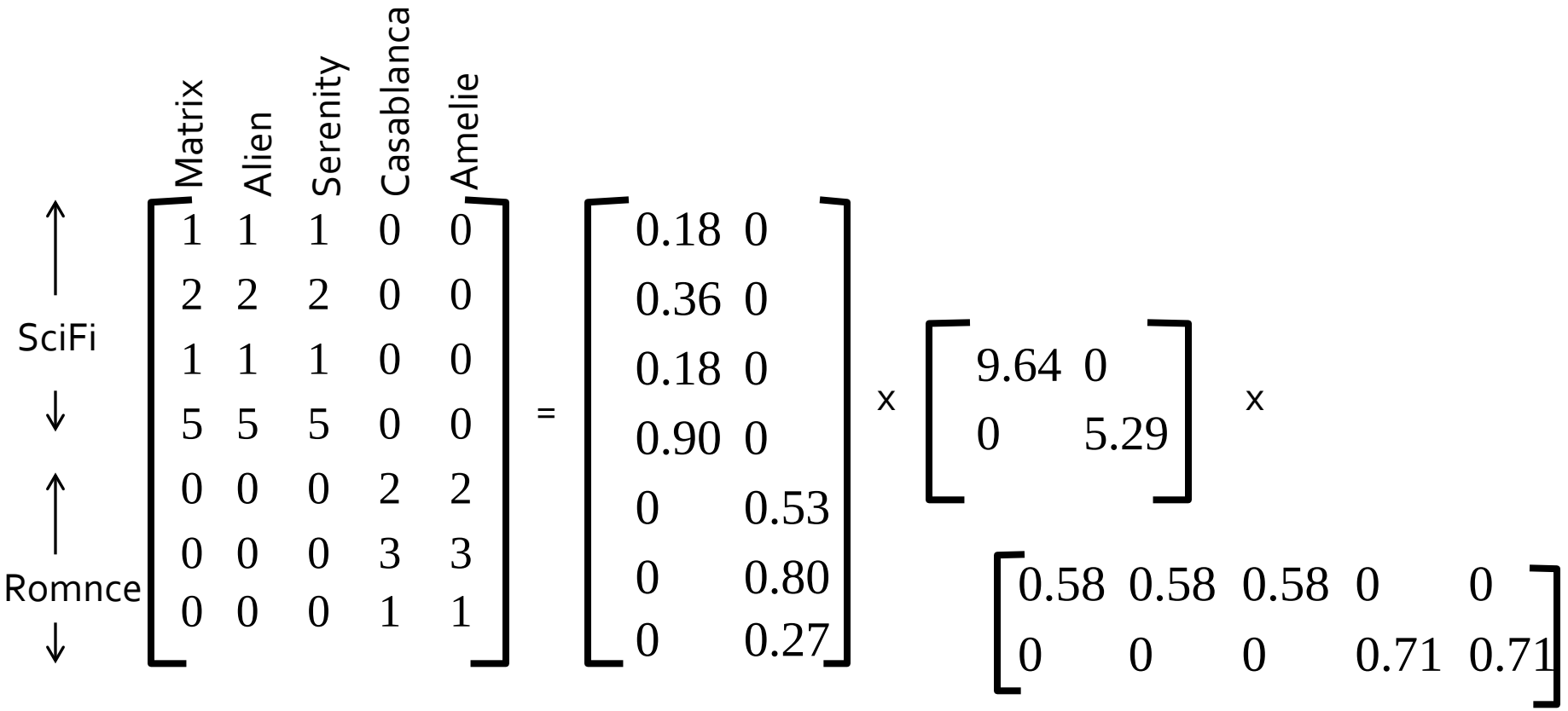
Q: Find users that like 'Matrix' and 'Alien'



Case study: How to query?

Q: Find users that like 'Matrix' and 'Alien'

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



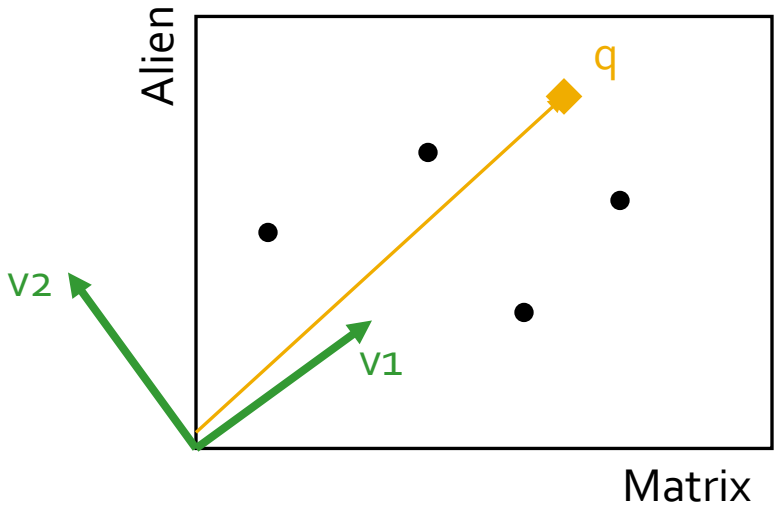
Case study: How to query?

Q: Find users that like 'Matrix'

A: map query vectors into 'concept space' – how?

q=

Matrix	Alien	Serenity	Casablanca	Amelie
5	0	0	0	0



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

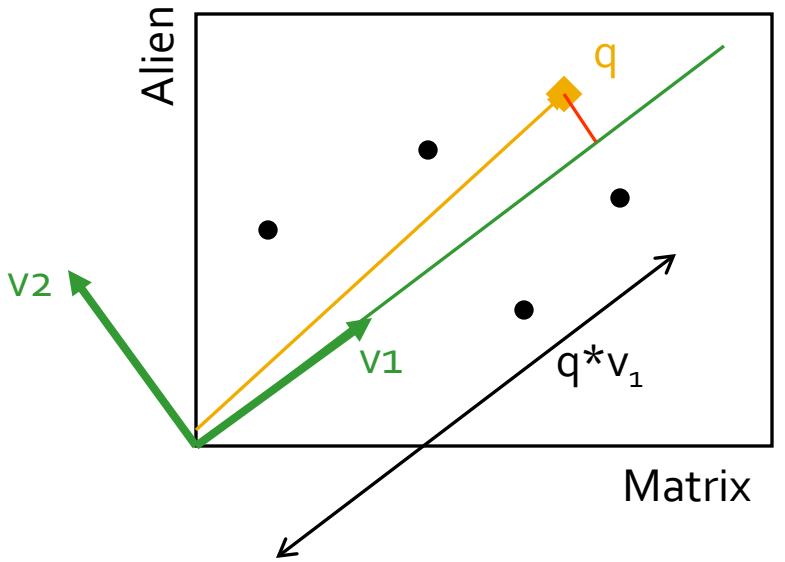
Case study: How to query?

Q: Find users that like 'Matrix'

A: map query vectors into 'concept space' – how?

q=

Matrix	Alien	Serenity	Casablanca	Amelie
5	0	0	0	0



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

Case study: How to query?

Compactly, we have:

$q_{concept} = q \, V$

E.g.:

	Matrix	Alien	Serenity	Casablanca	Amelie
q=	5	0	0	0	0

$$\begin{bmatrix} 0.58 & 0 \\ 0.58 & 0 \\ 0.58 & 0 \\ 0 & 0.71 \\ 0 & 0.71 \end{bmatrix} = \begin{bmatrix} 2.9 & 0 \end{bmatrix}$$

SciFi-concept

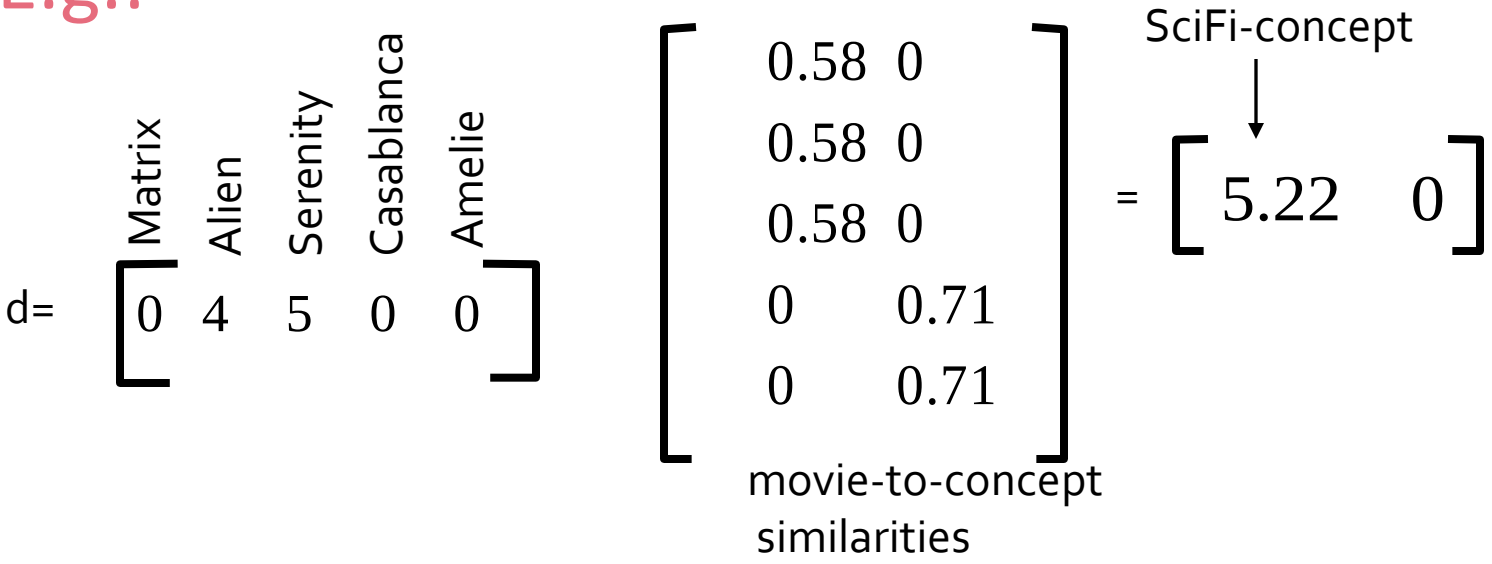
movie-to-concept similarities

Case study: How to query?

How would the user ('Alien', 'Serenity') be handled?

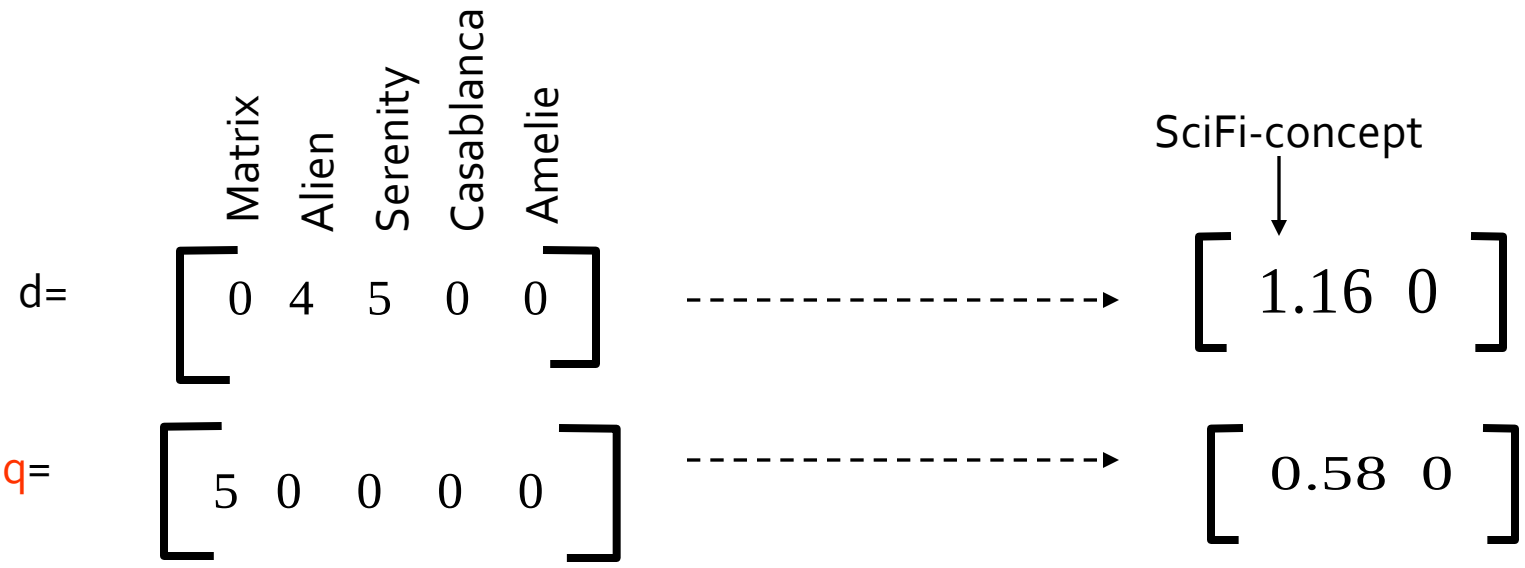
$d_{concept} = d \mathbf{V}$

E.g.:



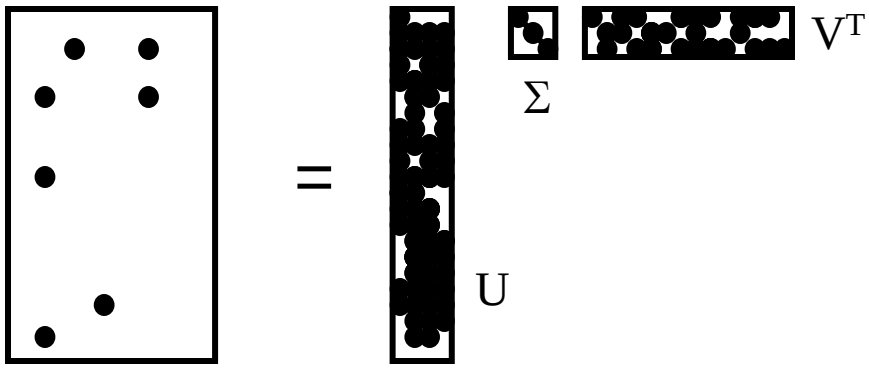
Case study: How to query?

Observation: User ('Alien', 'Serenity') will be retrieved by query ('Matrix'), although it did not rate 'Matrix'!



SVD: Drawbacks

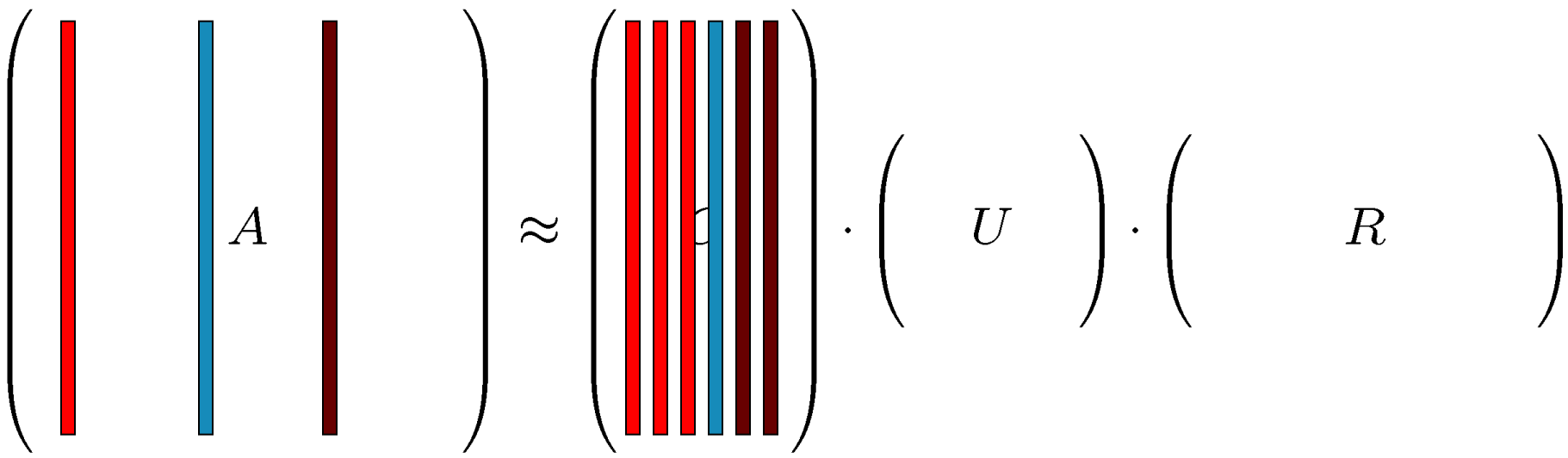
+ Optimal low-rank approximation:
  - in L2 norm
- Interpretability problem:
  - A singular vector specifies a linear combination of all input columns or rows
- Lack of Sparsity:
  - Singular vectors are **dense**



CUR Decomposition

- Goal:
  Make $\|A\text{-}CUR\|_F$ small

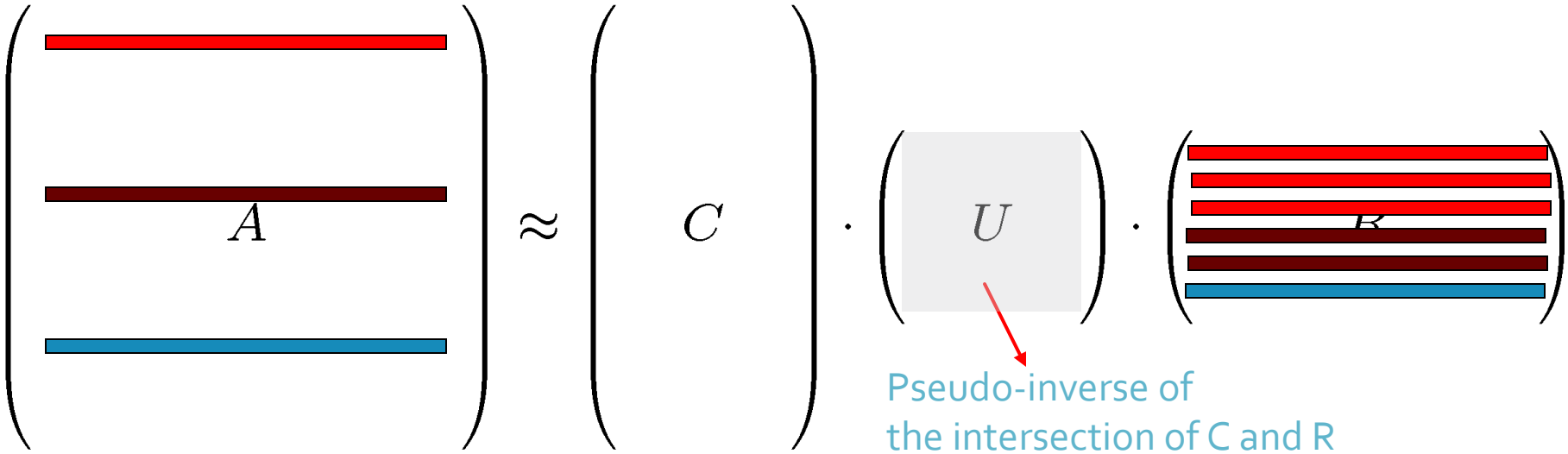
Frobenius norm:
$\|X\|_F = \Sigma_{ij} X_{ij}^2$



CUR Decomposition

- Goal:
  Make $\|A\text{-}CUR\|_F$ small

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



CUR: provably good approx. to SVD

- Let:

  $\mathbf{A}_k$ be the "best" rank k approximation to $\mathbf{A}$ (e.i., SVD)

Theorem [Drineas et al.]:

CUR in O(mn) time achieves

- $\|A\text{-}CUR\|_F \leq \|A\text{-}A_k\|_F + \varepsilon\|A\|_F$

with probability at least $1\text{-}\delta$, by picking

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

CUR: How it Works

- Sample columns (similarly for rows):

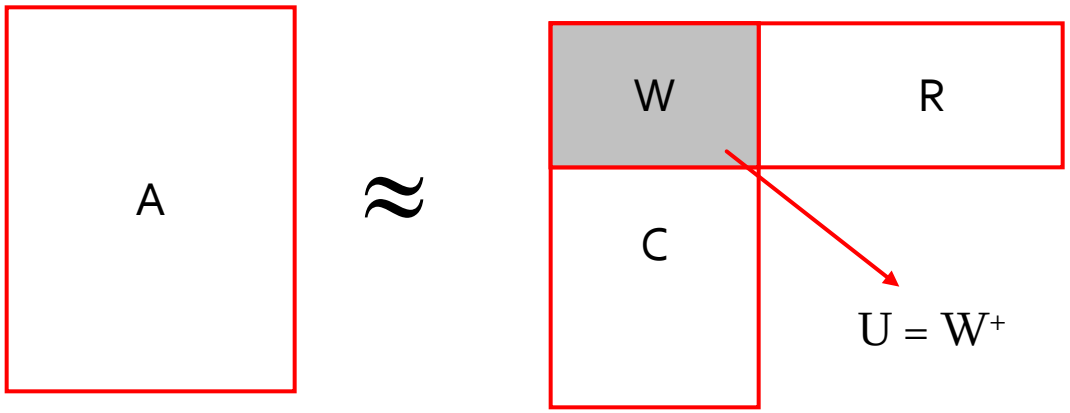
Input: matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, sample size c
Output: $\mathbf{C}_d \in \mathbb{R}^{m \times c}$

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 $\mathbf{C}_d(:,i) = \mathbf{A}(:,j) / \sqrt{cP(j)}$

Computing U

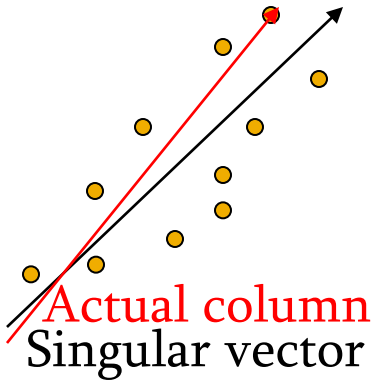
- Let W be the "intersection" of sampled columns C and rows R
  - Let SVD of $W = X \Sigma Y^T$
- Then: $U = W^+ = X \Sigma^+ Y^T$
  - Σ^+: reciprocals of non-zero singular values: $\Sigma^+_{ii} = 1/\Sigma_{ii}$

    i.e., Moore–Penrose pseudoinverse



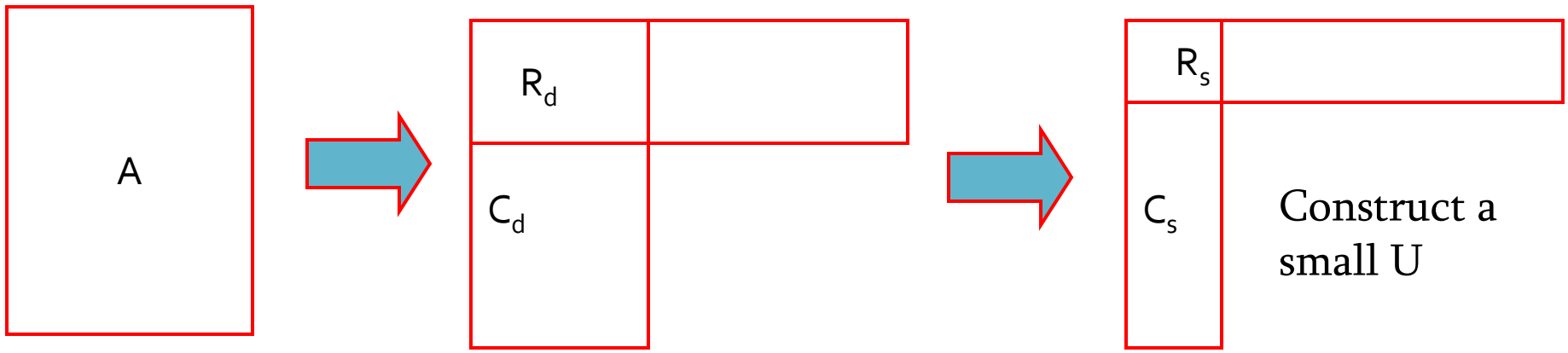
CUR: Pros & Cons

+ 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

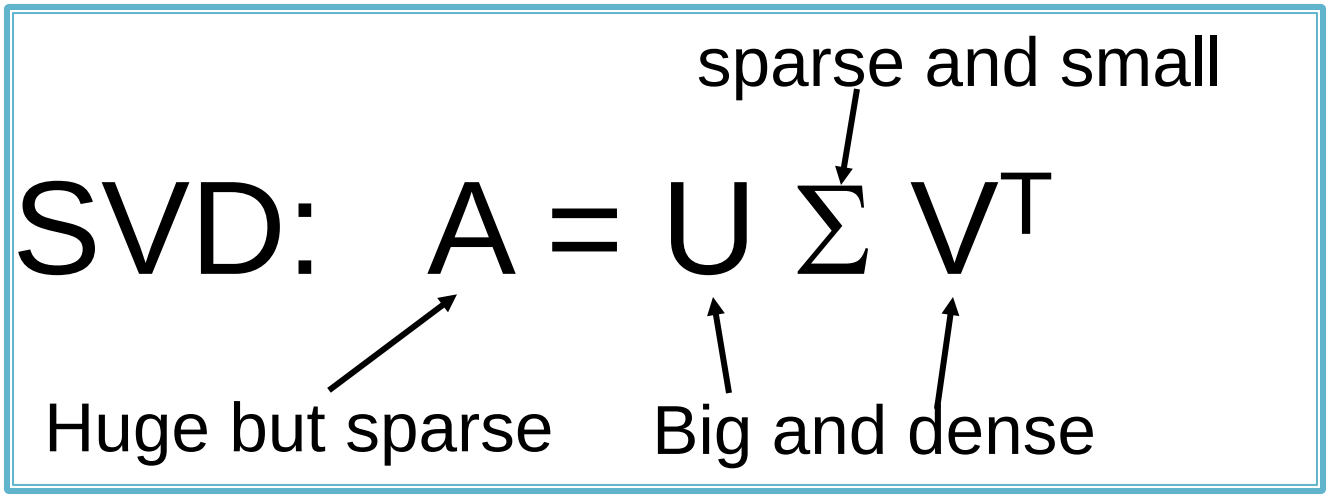


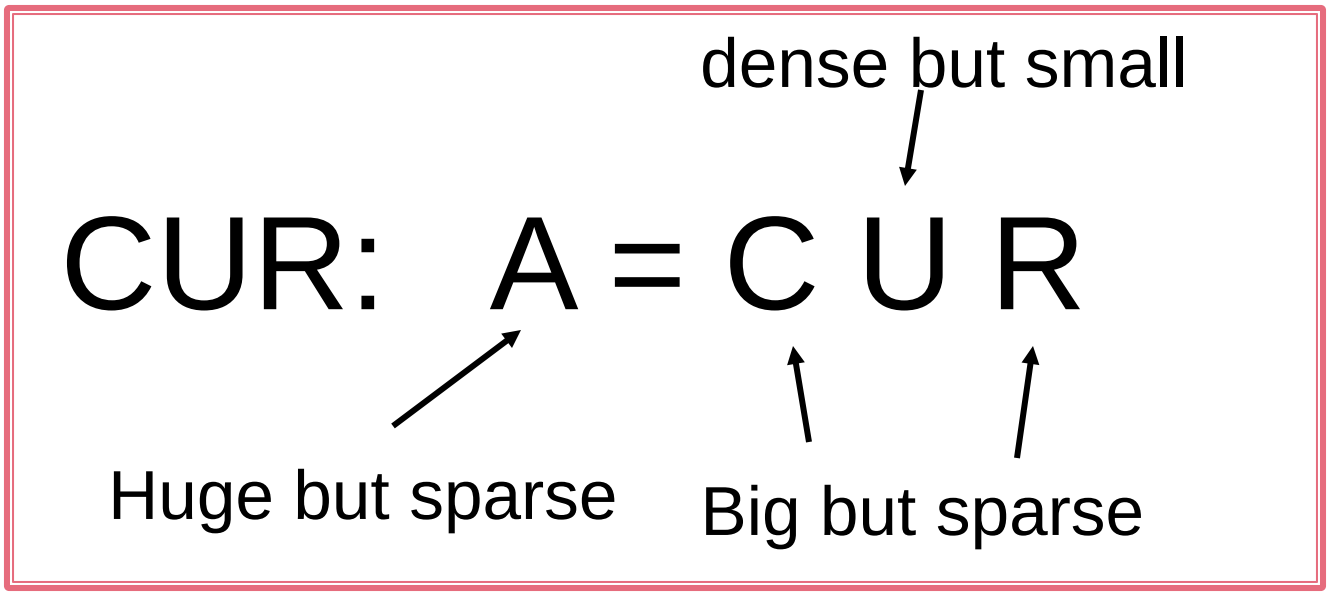
Solution

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



SVD vs. CUR

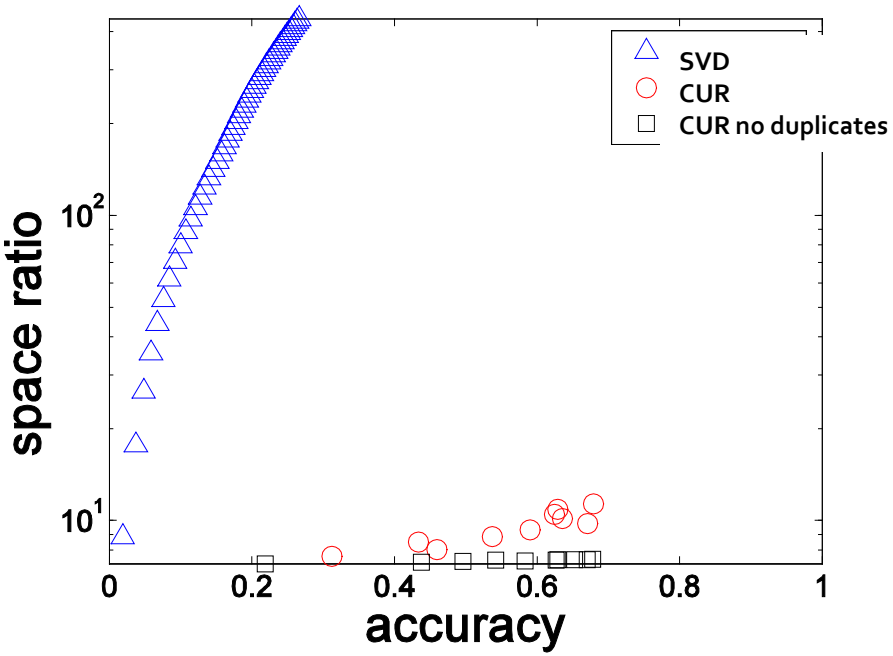

sparse and small
SVD: A = U Σ V^T
Huge but sparse
Big and dense

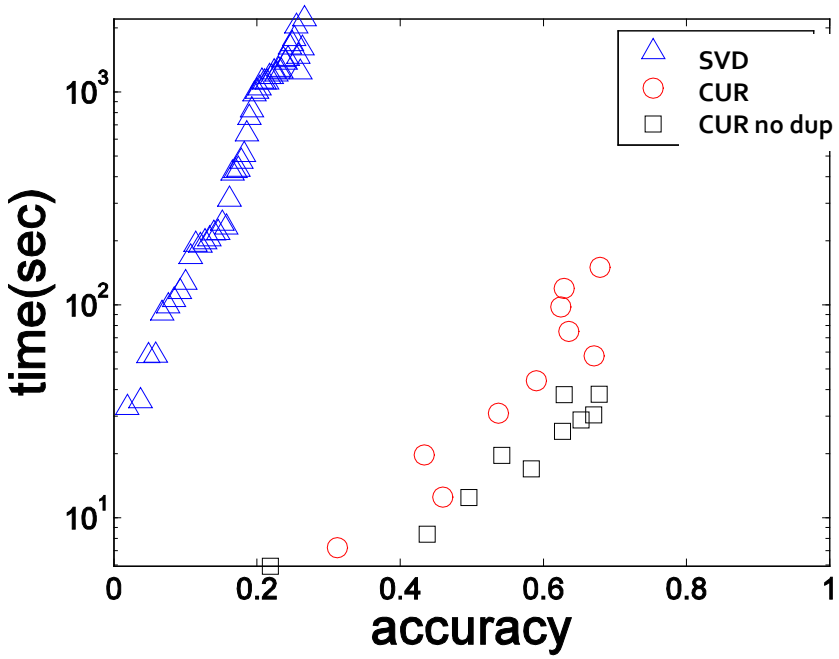

dense but small
CUR: A = C U R
Huge but sparse
Big but sparse

Simple Experiment

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

Results: DBLP- big sparse matrix



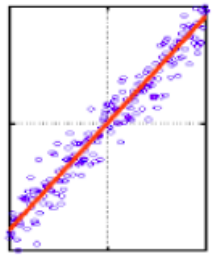


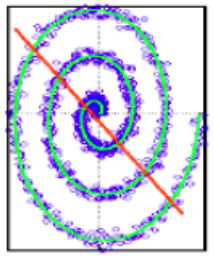
- Accuracy: 1 – relative sum square error
- Space ratio:
  - #output matrix entries / #input matrix entries
- CPU time

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

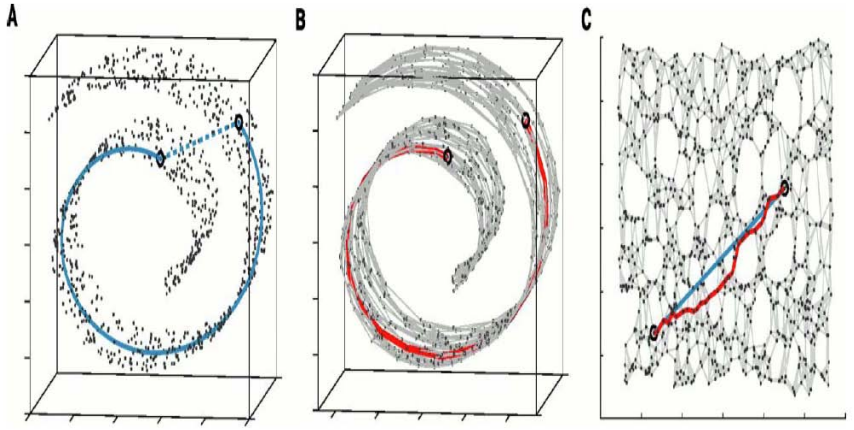
What about linearity assumtion?

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





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



References: 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)