# 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 $r$



$$A \approx U \times \Sigma \times V^T$$
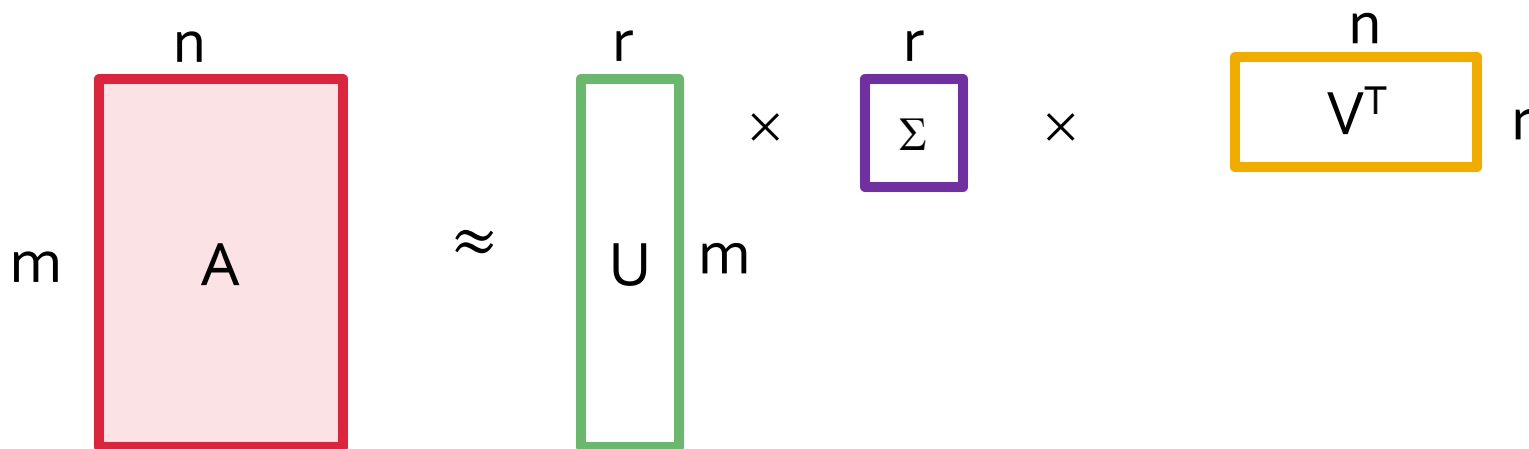
# Dimensionality Reduction

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

| day customer | Wc 7/10/96 | Th 7/11/96 | Fr 7/12/96 | Sa 7/13/96 | Su 7/14/96 | New representation |
|---|---|---|---|---|---|---|
| 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



$D = 2$
$d = 1$

$D = 3$
$d = 2$
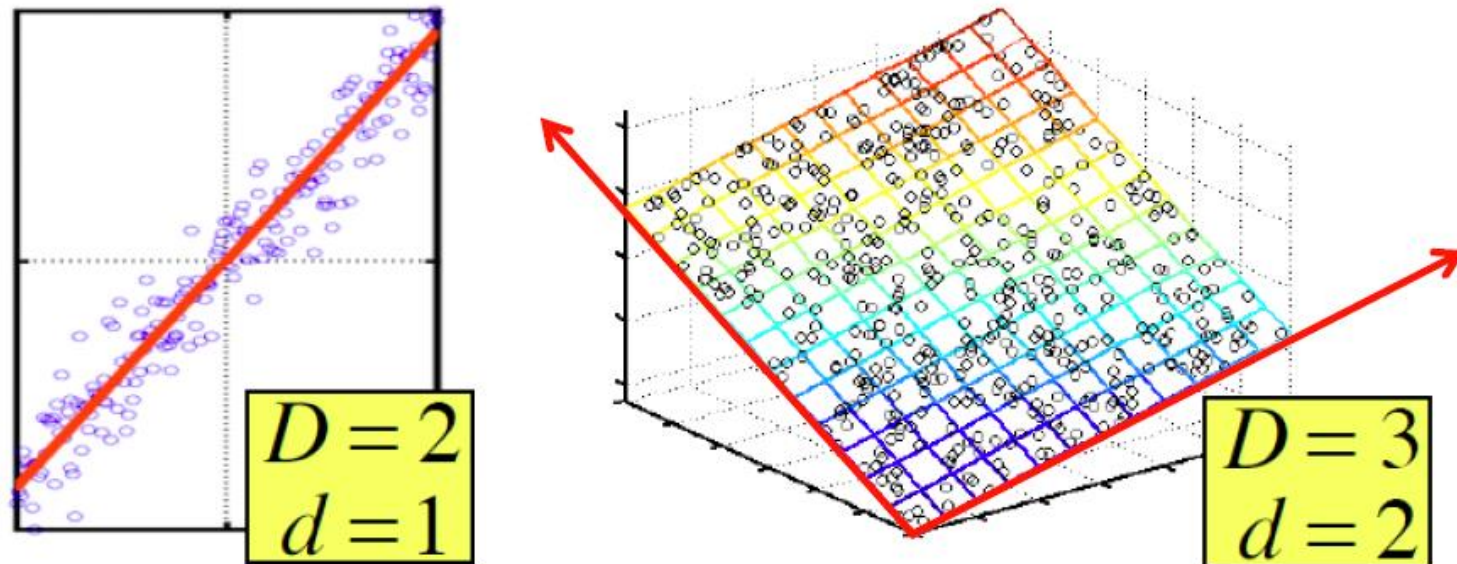
There are hidden, or **latent factors, latent dimensions** that – to a close approximation – explain why the values are as they appear in the data matrix

# 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$^{nd}$ greatest variance

- And so on…, until you have enough dimensions that variance is really low
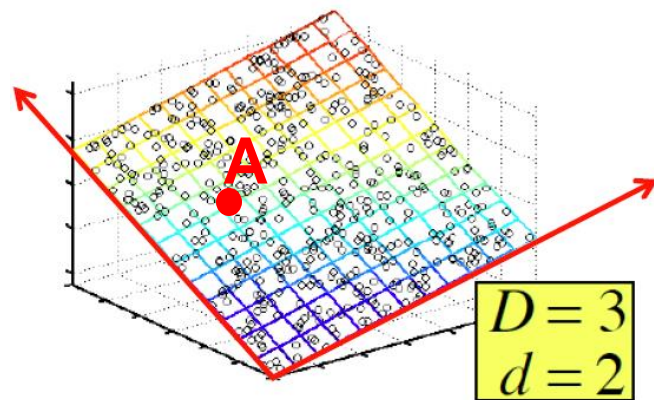


$D = 2$
$d = 1$

$D = 3$
$d = 2$
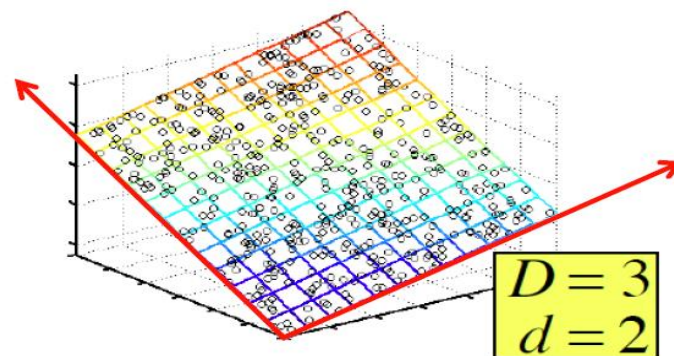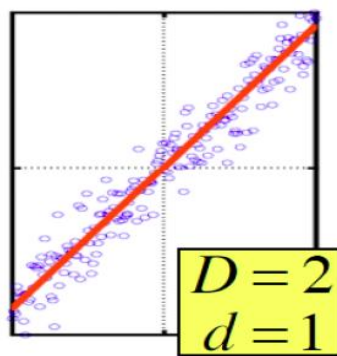
# Rank is "Dimensionality"

- **Q:** What is **rank** of a matrix **A**?
- **A:** Number of **linearly independent** rows of **A**
- **Cloud of points in 3D space:**
  - Think of point coordinates as a matrix:

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

    $D = 3$
    $d = 2$

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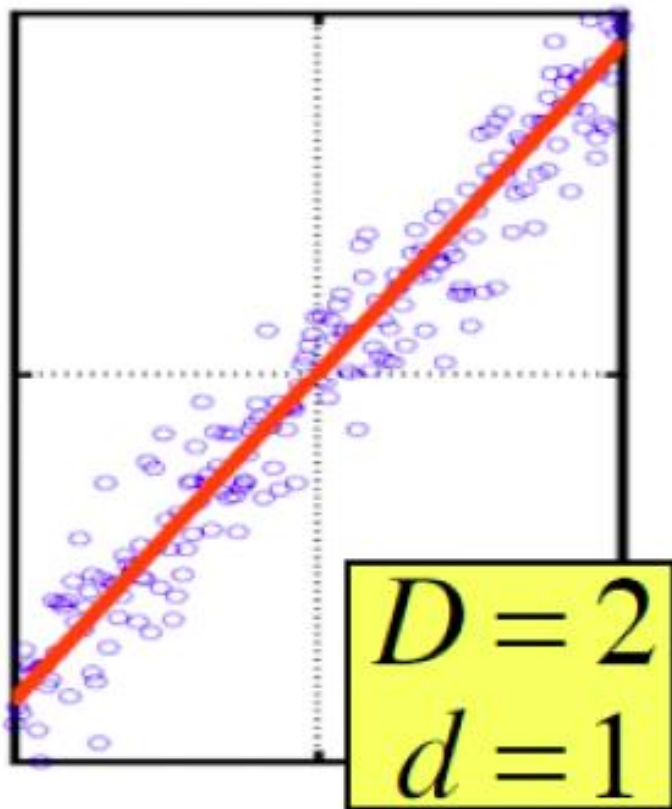
Rather than representing every point with 2 coordinates we represent each point with 1 coordinate (corresponding to the position of the point on the red line).

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

$$D = 2$$
$$d = 1$$

# SVD: Singular Value Decomposition

# Reducing Matrix Dimension

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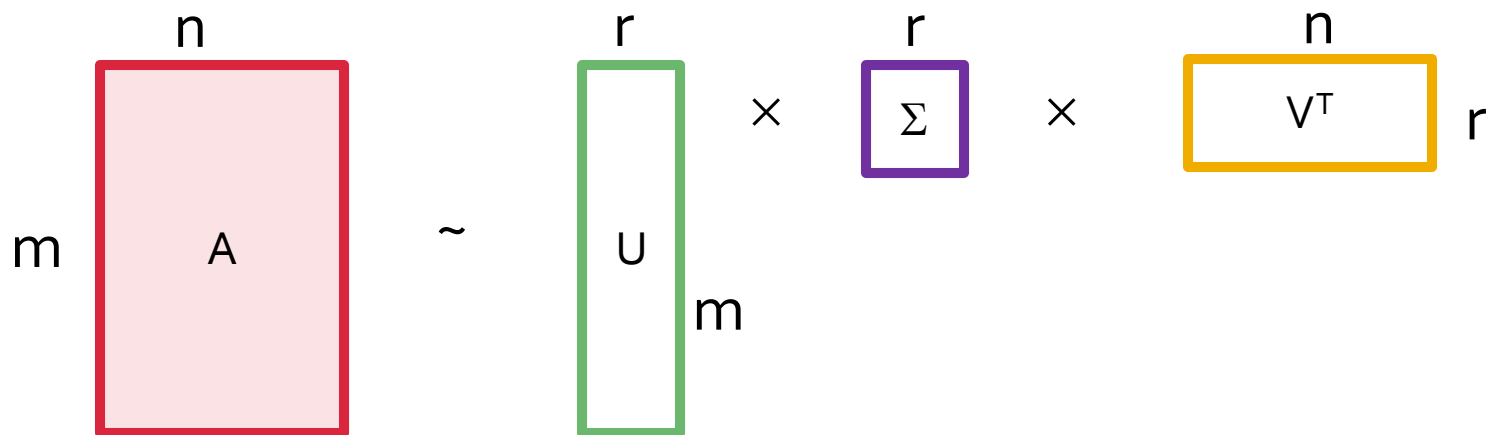


- There are strong constraints on the form of each of these matrices

  - 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

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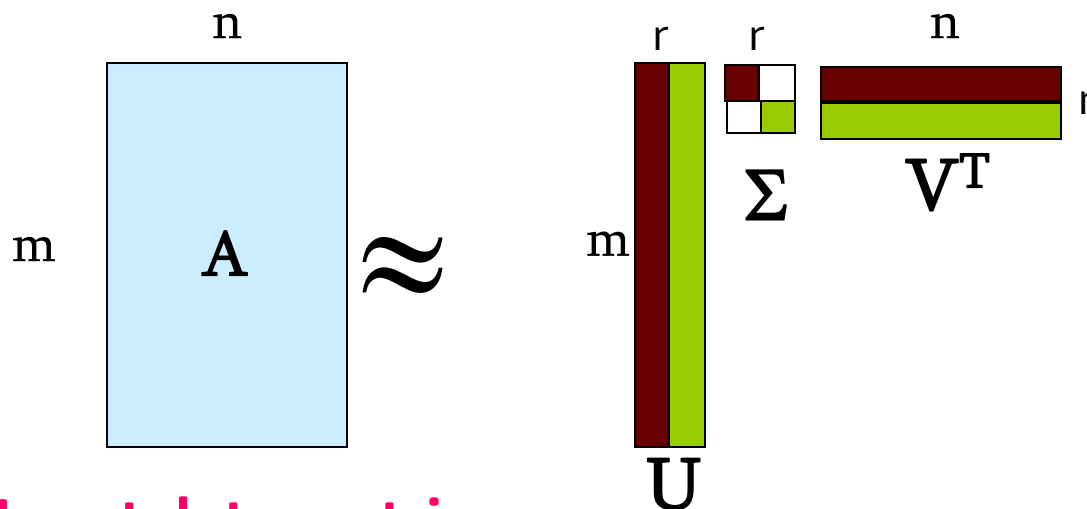


- **A**: **Input data matrix**
  - *m* x *n* matrix (e.g., *m* documents, *n* terms)
- **U**: **Left singular vectors**
  - *m* x *r* matrix  (*m* documents, *r* concepts)
- **Σ**: **Singular values**
  - *r* x *r* diagonal matrix (strength of each 'concept')
    (*r* : rank of the matrix **A**)
- **V**: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

# SVD

$$A \approx U\Sigma V^T = \sum_i \sigma_i u_i \circ v_i^{\mathsf{T}}$$



$\sigma_1 u_1 v_1$

$\sigma_2 u_2 v_2$

n

m

A

≈

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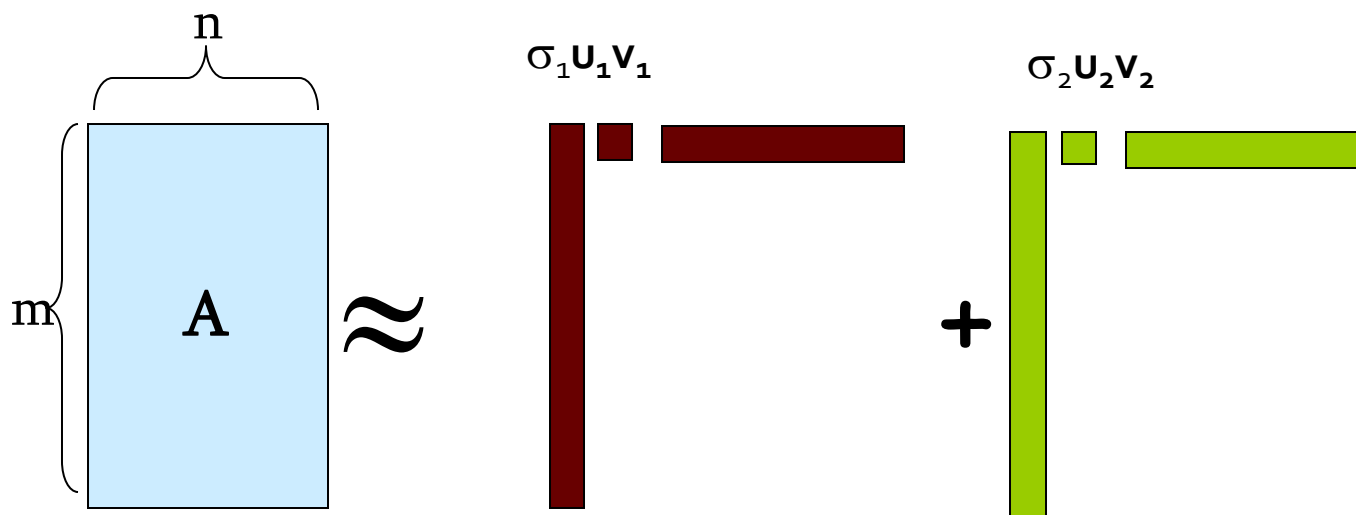
If we set $\sigma_2 = 0$, then the green columns may as well not exist.

$\sigma_i$ … scalar
$u_i$ … vector
$v_i$ … vector

# SVD – Properties

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

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

Nice proof of uniqueness: https://www.cs.cornell.edu/courses/cs322/2008sp/stuff/TrefethenBau_Lec4_SVD.pdf

# SVD – Example: Users-to-Movies

- **Consider a matrix. What does SVD do?**

$$
\begin{array}{c}
\phantom{=}
\begin{array}{ccccc}
\text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie}
\end{array}\\
\begin{bmatrix}
1 & 1 & 1 & 0 & 0\\
3 & 3 & 3 & 0 & 0\\
4 & 4 & 4 & 0 & 0\\
5 & 5 & 5 & 0 & 0\\
0 & 2 & 0 & 4 & 4\\
0 & 0 & 0 & 5 & 5\\
0 & 1 & 0 & 2 & 2
\end{bmatrix}
\end{array}
$$

SciFi

Romance

**=**

$m$ 

$\mathbf{U}$  $\Sigma$  $\mathbf{V^T}$  $n$

$D = 3$
$d = 2$

**"Concepts"**
**AKA Latent dimensions**
**AKA Latent factors**

Ratings matrix where each column corresponds to a movie and each row to a user. First 4 users prefer SciFi, while others prefer Romance.

- **A = U ∑ V$^T$ - example: Users to Movies**

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

Movies: Matrix, Alien, Serenity, Casablanca, Amelie

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

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

SciFi-concept

Romance-concept

|            | 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      |

=

| 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 |

X

| 12.4 | 0 | 0 |
|------|---|---|
| 0 | 9.5 | 0 |
| 0 | 0 | 1.3 |

X

| 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 Σ V^T - example:**

*U* is "user-to-concept" factor matrix

SciFi-concept    Romance-concept

|  | 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 |

=

| | | |
|---|---|---|
| **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** |

X

| | | |
|---|---|---|
| **12.4** | 0 | 0 |
| 0 | **9.5** | 0 |
| 0 | 0 | **1.3** |

X

| | | | | |
|---|---|---|---|---|
| **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 |

- **A = U ∑ V$^T$ - example:**

SciFi-concept

"strength" of the SciFi-concept

|  | 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** |

=

$$\begin{bmatrix} \mathbf{0.13} & 0.02 & -0.01 \\ \mathbf{0.41} & 0.07 & -0.03 \\ \mathbf{0.55} & 0.09 & -0.04 \\ \mathbf{0.68} & 0.11 & -0.05 \\ 0.15 & \mathbf{-0.59} & \mathbf{0.65} \\ 0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\ 0.07 & \mathbf{-0.29} & \mathbf{0.32} \end{bmatrix}$$

X

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

X

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

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

*V* is "movie-to-concept" factor matrix

SciFi-concept

$$
\begin{array}{c}
\text{Matrix} \quad \text{Alien} \quad \text{Serenity} \quad \text{Casablanca} \quad \text{Amelie}
\end{array}
$$

SciFi

Romance

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

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

SciFi-concept
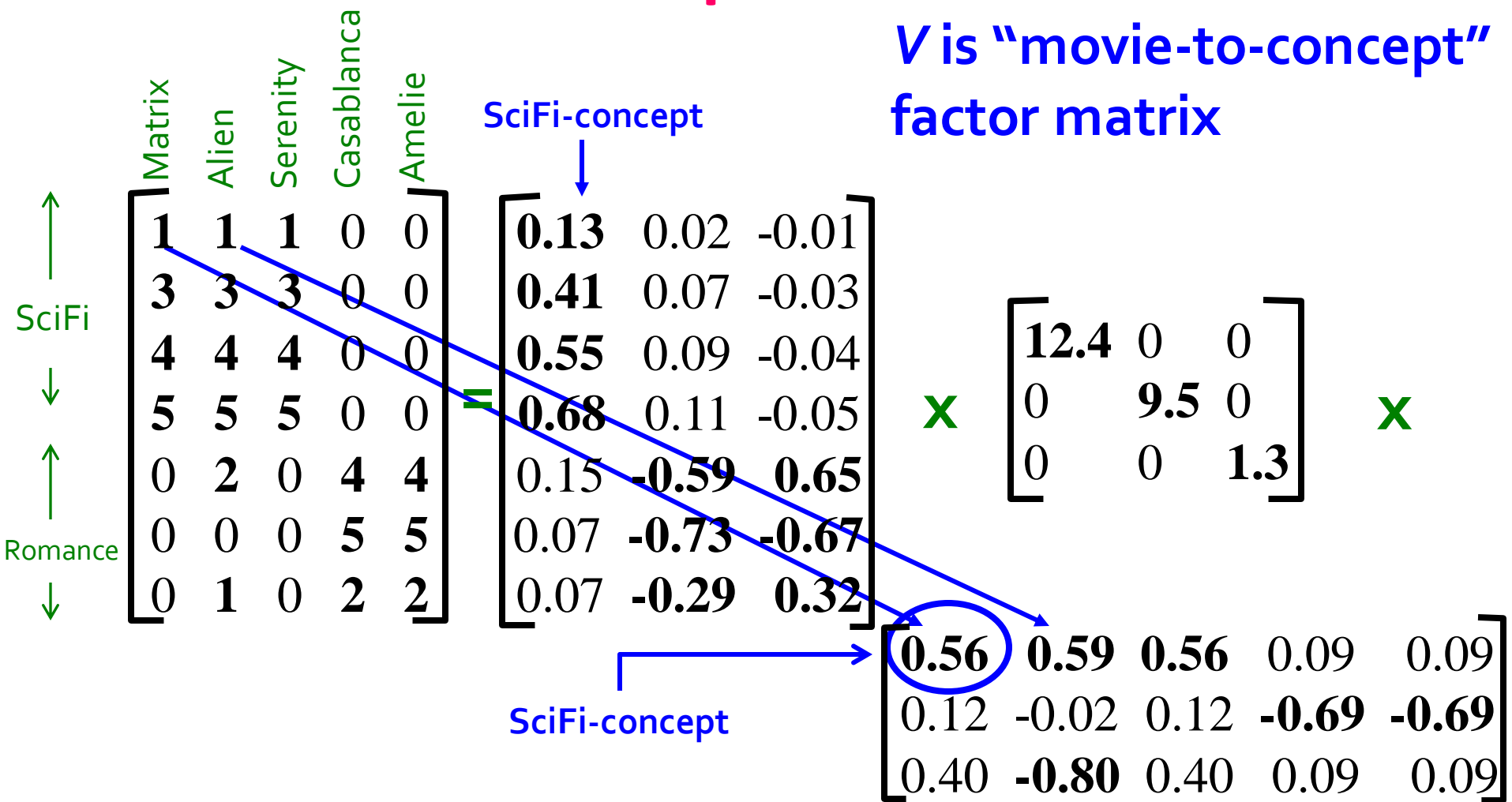
# SVD – Interpretation #1

**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

**Movie 2 rating** (y-axis)

**Movie 1 rating** (x-axis)

**first right singular vector**

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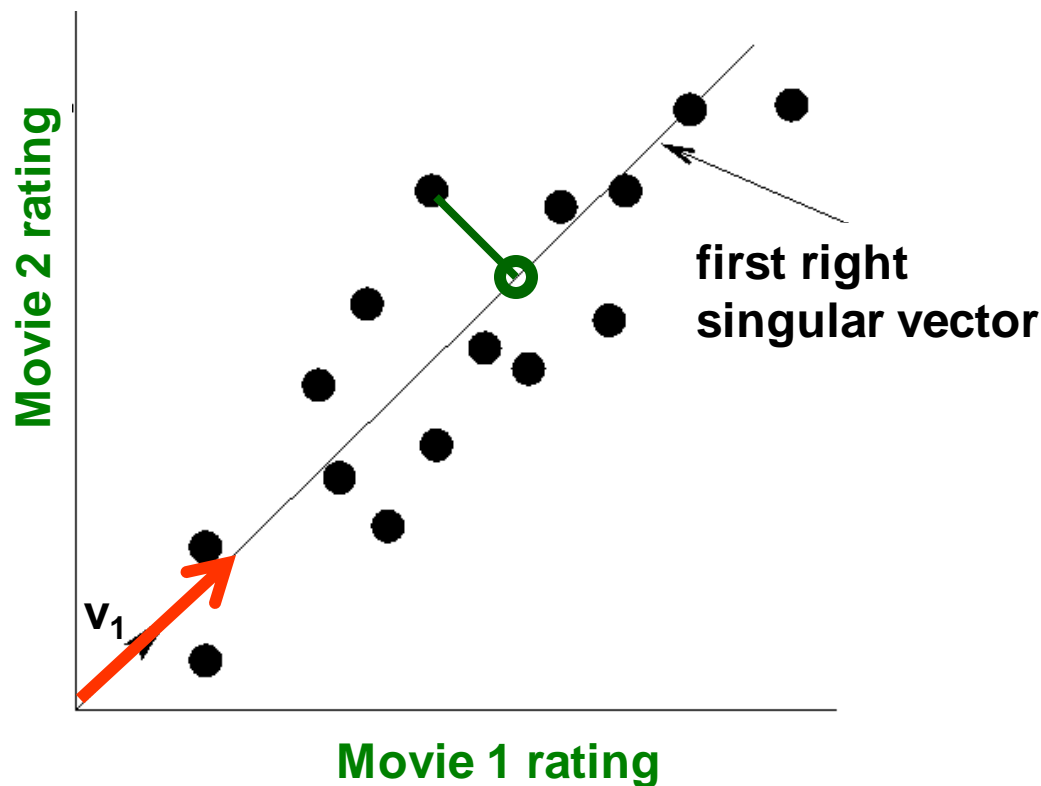
- Instead of using two coordinates $(x, y)$ to describe point positions, let's use only one coordinate
- Point's position is its location along vector $v_1$

# SVD – Dimensionality Reduction

- **A = U $\Sigma$ V$^T$ - example:**
  - **U**: "user-to-concept" matrix
  - **V**: "movie-to-concept" matrix



Movie 2 rating / Movie 1 rating
first right singular vector
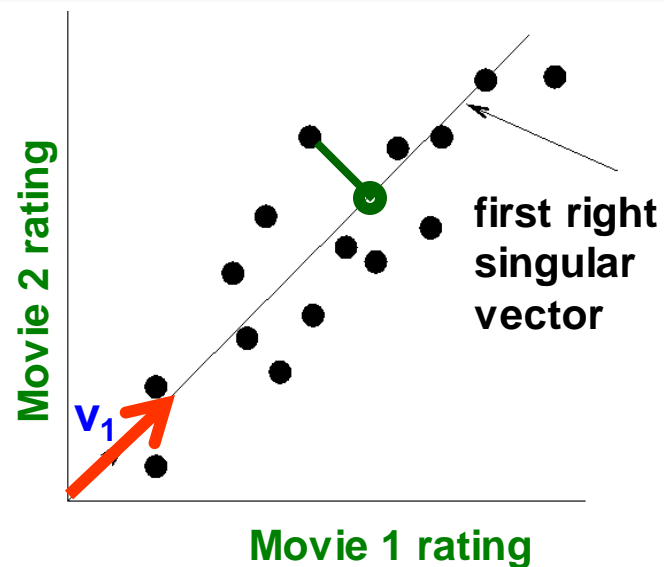$v_1$

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

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

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

Movie 2 rating

**first right singular vector**

$v_1$

Movie 1 rating

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

# SVD – Dimensionality Reduction

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

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



**Movie 2 rating**

**v$_1$**

**first right singular vector**

**Movie 1 rating**

**Projection of users on the "Sci-Fi" axis**
**U $\Sigma$:**

$$
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 \\
3 & 3 & 3 & 0 & 0 \\
4 & 4 & 4 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 2 & 0 & 4 & 4 \\
0 & 0 & 0 & 5 & 5 \\
0 & 1 & 0 & 2 & 2
\end{bmatrix}
$$
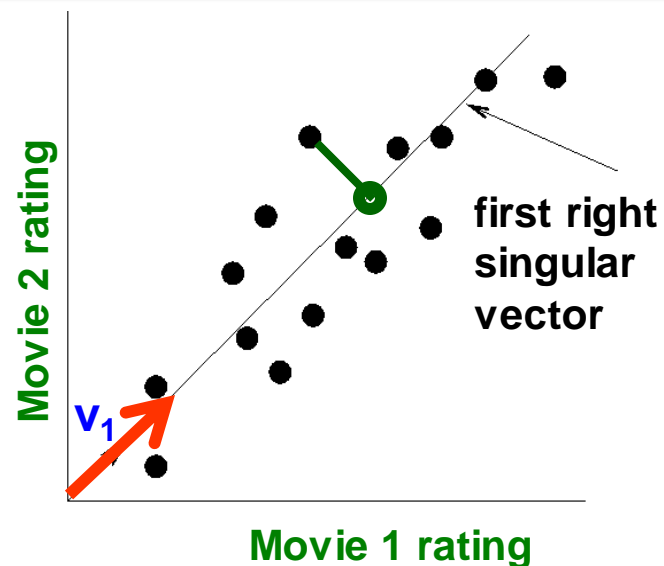
$$
\begin{bmatrix}
1.61 & 0.19 & -0.01 \\
5.08 & 0.66 & -0.03 \\
6.82 & 0.85 & -0.05 \\
8.43 & 1.04 & -0.06 \\
1.86 & -5.60 & 0.84 \\
0.86 & -6.93 & -0.87 \\
0.86 & -2.75 & 0.41
\end{bmatrix}
$$

## More details

- **Q: How is dim. reduction done?**

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

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

## More details

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

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

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

## More details

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

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

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

# 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**

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

# SVD – Interpretation #2

## More details

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

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

Reconstructed data matrix B

**Reconstruction Error is quantified by the Frobenius norm:**

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

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

is "small"

# SVD – Best Low Rank Approx.

- **Fact: SVD gives 'best' axis to project on:**
  - '**best**' = minimizing the sum of reconstruction errors
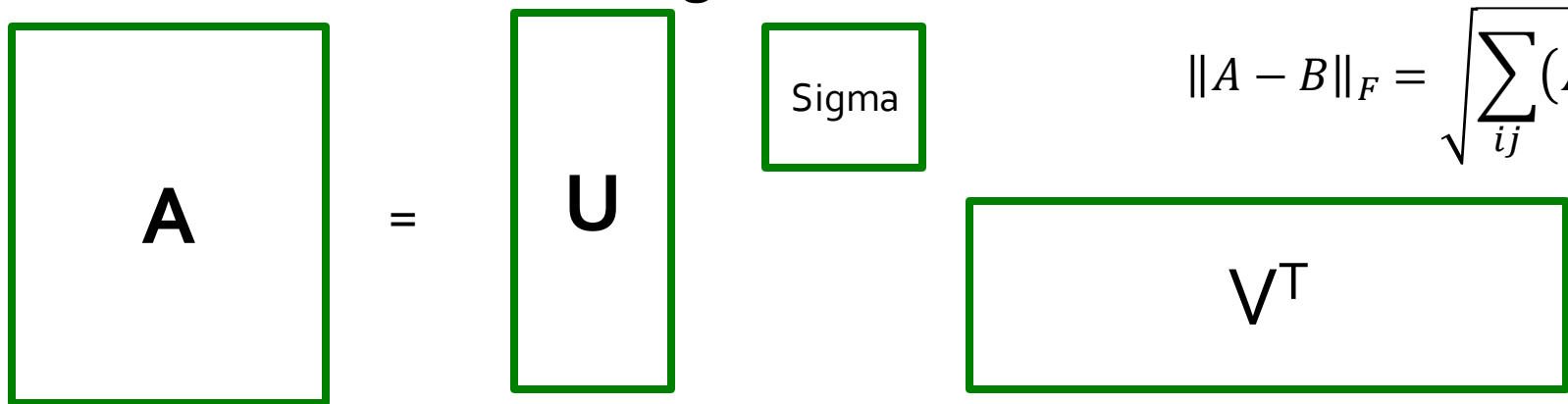
$$\|A - B\|_F = \sqrt{\sum_{ij} (A_{ij} - B_{ij})^2}$$

A = U Sigma $V^\mathsf{T}$

**B is best approximation of A:**
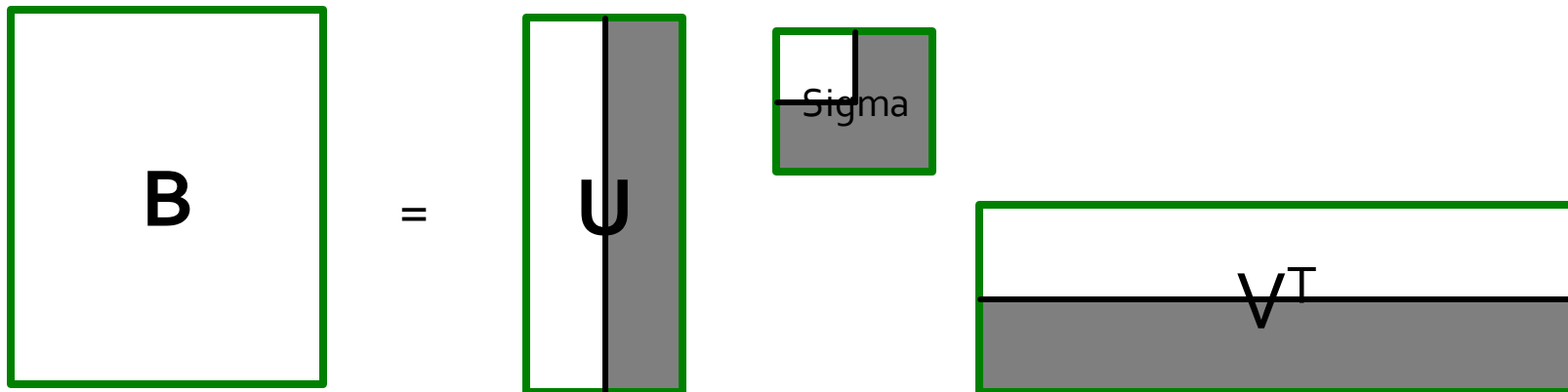
B = U Sigma $V^\mathsf{T}$

# 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 $r$ (# of latent factors)?**
- 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" $\boldsymbol{x}_0$

  - Construct $\boldsymbol{x}_{k+1} = \dfrac{M\boldsymbol{x}_k}{||M\boldsymbol{x}_k||}$ for $k = 0, 1, \ldots$
    - $|| \ldots ||$ denotes the Frobenius norm
  - Stop when consecutive $\boldsymbol{x}_k$ show little change

# Example: Iterative Eigenvector

$$M = \begin{pmatrix} 1 & 2 \\ 2 & 3 \end{pmatrix} \qquad \mathbf{x}_0 = \begin{matrix} 1 \\ 1 \end{matrix}$$

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

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

…..

# Finding the Principal Eigenvalue

- Once you have the principal eigenvector $\boldsymbol{x}$, you find its eigenvalue $\lambda$ by $\lambda = \boldsymbol{x}^T M \boldsymbol{x}$.
  - In proof: We know $\boldsymbol{x}\lambda = M\boldsymbol{x}$ if $\lambda$ is the eigenvalue; multiply both sides by $\boldsymbol{x}^T$ on the left.
  - Since $\boldsymbol{x}^T \boldsymbol{x} = 1$ we have $\lambda = \boldsymbol{x}^T M \boldsymbol{x}$

- **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 $\boldsymbol{x}$:
$$M^* := M - \lambda\, x\, x^T$$

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

- **Example:**

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

# 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(nm²)** or **O(n²m)** (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

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

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

Column headers: Matrix, Alien, Serenity, Casablanca, Amelie

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



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

(columns: Matrix, Alien, Serenity, Casablanca, Amelie)

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

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

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

**Project into concept space:**
Inner product with each
'concept' vector $v_i$
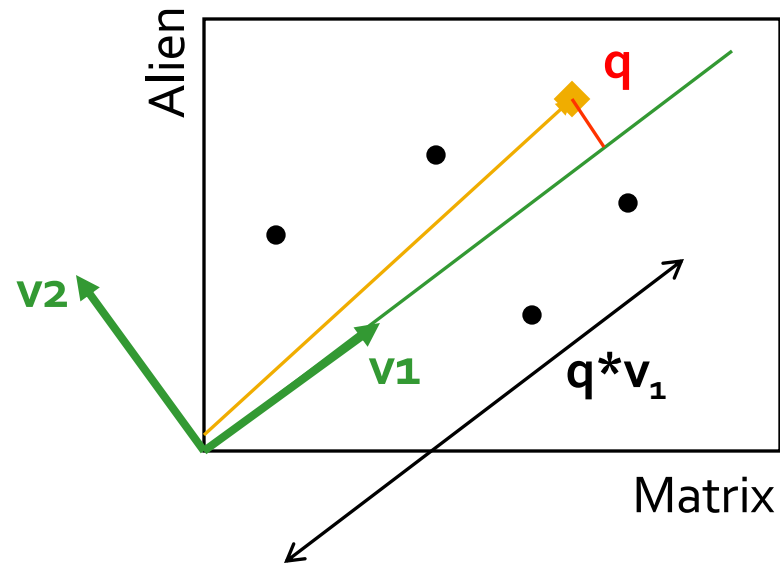
## Compactly, we have:
$q_{concept} = q\ V$

## E.g.:

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

SciFi-concept

**movie-to-concept factors (V)**

- **How would the user *d* that rated ('Alien', 'Serenity') be handled? $d_{concept} = d\ V$**

E.g.:

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

SciFi-concept

movie-to-concept factors (V)

- **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**!



SciFi-concept

$$d = \begin{bmatrix} 0 & 4 & 5 & 0 & 0 \end{bmatrix} \dashrightarrow \begin{bmatrix} 5.2 & 0.4 \end{bmatrix}$$

$$q = \begin{bmatrix} 5 & 0 & 0 & 0 & 0 \end{bmatrix} \dashrightarrow \begin{bmatrix} 2.8 & 0.6 \end{bmatrix}$$

**Zero ratings in common**                    Similarity > 0
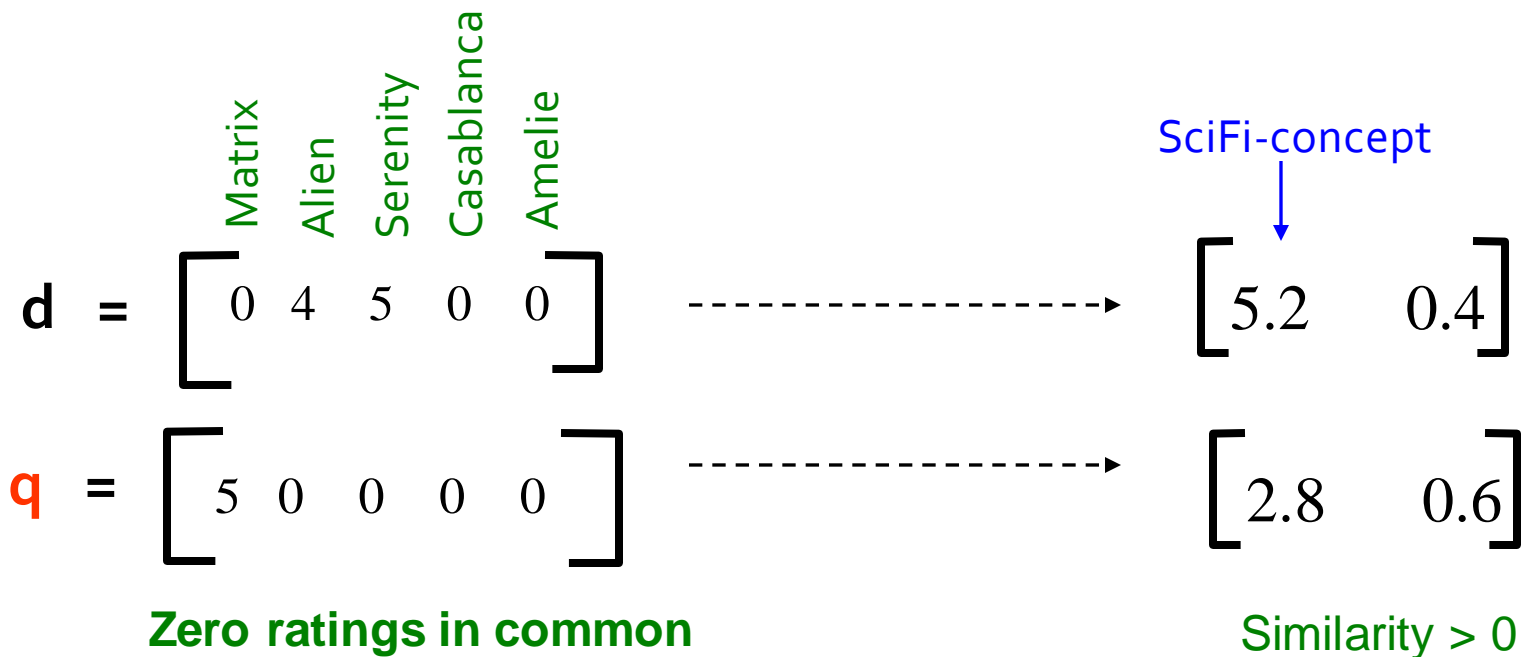
# SVD: Drawbacks

+ **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!**

# 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

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



$$A \approx C \cdot U \cdot R$$

A          C          U          R

# CUR Decomposition

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



$$A \approx C \cdot U \cdot R$$

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

A        C        U        R

# Computing U

- Let $W$ be the "intersection" of sampled columns $C$ and rows $R$

- **Def: W⁺** is the **pseudoinverse**

  - Let SVD of $W = X Z Y^T$

  - **Then:** $W^+ = Y Z^+ X^T$

    - $Z^+$: **reciprocals of non-zero singular values:** $Z^+_{ii} = 1/Z_{ii}$

- **Let:** $U = Y (Z^+)^2 X^T$
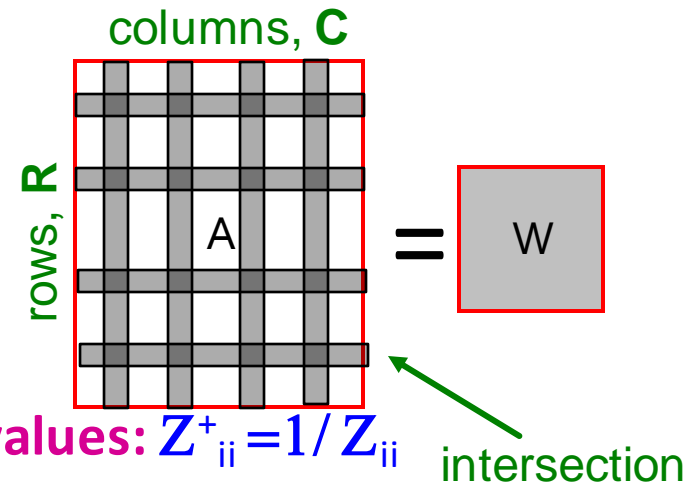
**Why the intersection?** These are high magnitude numbers

**Why pseudoinverse works?**

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

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

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

**Thus**, if **W** is nonsingular, pseudoinverse is the true inverse

columns, **C**

rows, **R**

A

= W

intersection

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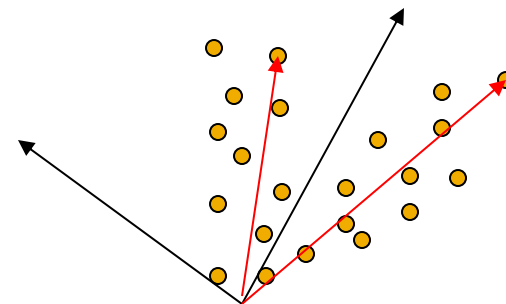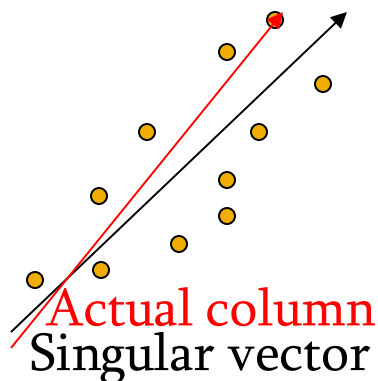
- **Sampling columns (similarly for rows):**

**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



Actual column
Singular vector

- **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 $c = O\left(\dfrac{k\,log\,k}{\varepsilon^2}\right)$ columns of A using ColumnSelect algorithm** (slide 56)

  - **Select $r = O\left(\dfrac{k\,log\,k}{\varepsilon^2}\right)$ rows of A using RowSelect algorithm** (slide 56)

  - **Set $U = Y\,(Z^+)^2 X^T$**

  - **Then:** $\underbrace{\left|\left|A - CUR\right|\right|_F}_{\text{CUR error}} \leq (2 + \varepsilon)\underbrace{\left|\left|A - A_K\right|\right|_F}_{\text{SVD error}}$

    with probability 98%

    **In practice:** Pick $4k$ cols/rows for a "rank-k" approximation

# CUR: Pros & Cons

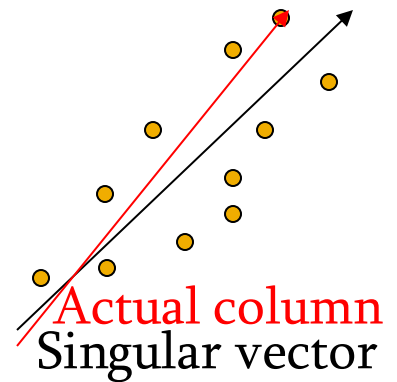+ **Easy interpretation**

  - 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

Actual column
Singular vector

# SVD vs. CUR

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

sparse and small
Huge but sparse
Big and dense

CUR:  $A = C U R$

dense but small
Huge but sparse
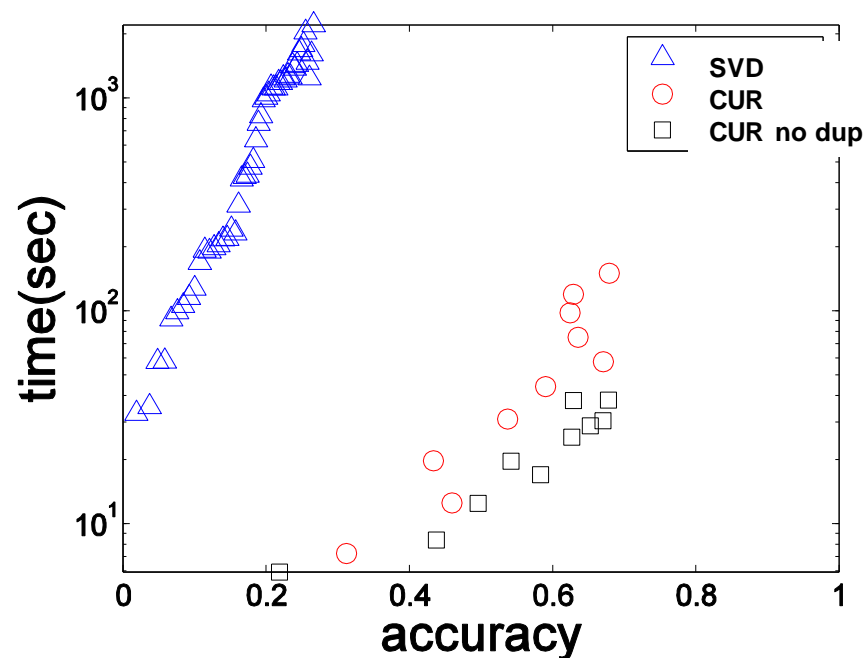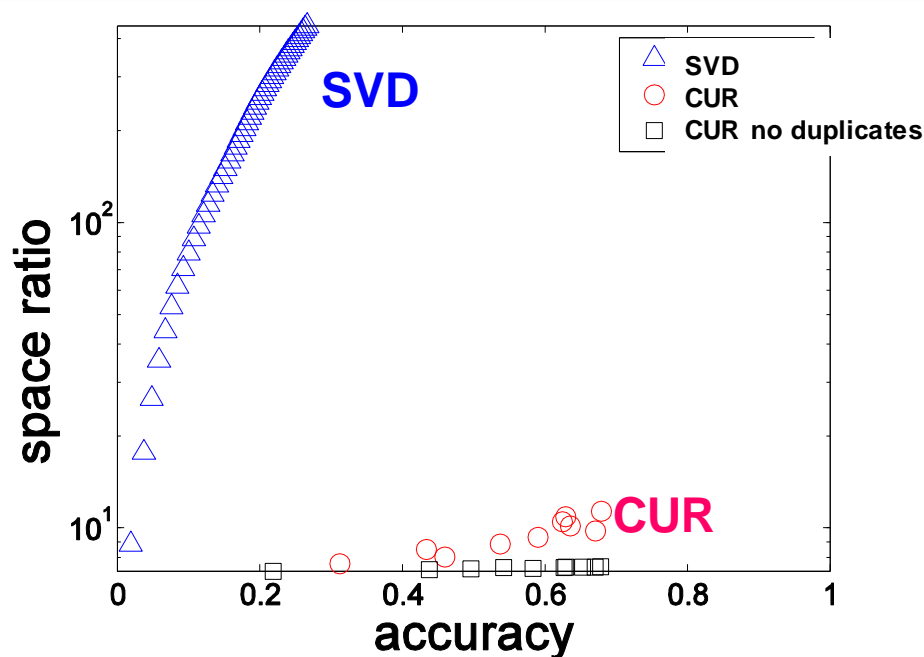Big but sparse

# SVD vs. CUR: Simple Experiment

- ## DBLP bibliographic data

  - Author-to-conference big sparse matrix

  - $A_{ij}$: Number of papers published by author $i$ at conference $j$

  - 428K authors (rows), 3659 conferences (columns)
    - **Very sparse**

- ## Want to reduce dimensionality

  - How much time does it take?

  - What is the reconstruction error?

  - How much space do we need?

- **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.