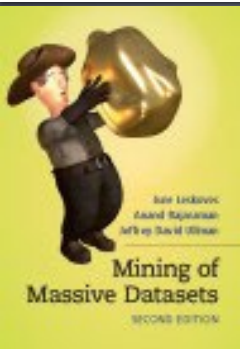


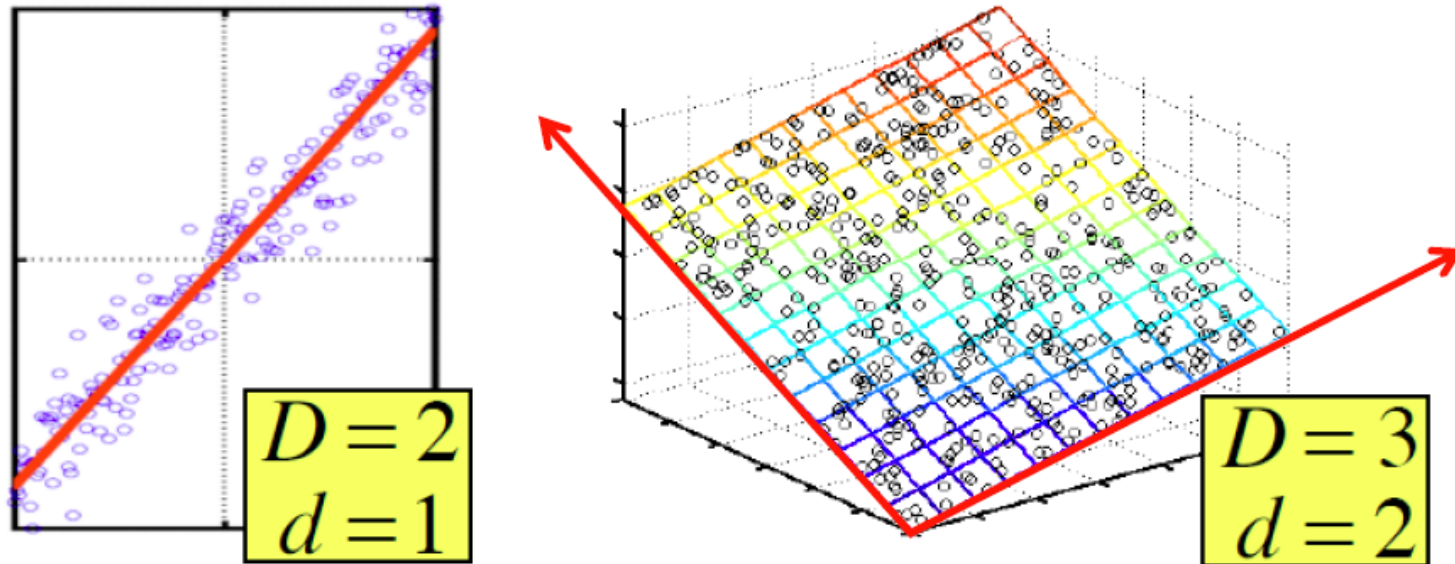
DSTA 7-b: Discovering latent dimensions

A presentation of Ch. 11 of Leskovec et al.,
[Mining of Massive Datasets](#)



Slides adapted from
Jure Leskovec's slides
"Dim. Reduction by
SVD"

Dimensionality Reduction



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

Dimensionality Reduction

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

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

The above matrix is really “2-dimensional.” All rows can be reconstructed by scaling $[1\ 1\ 1\ 0\ 0]$ or $[0\ 0\ 0\ 1\ 1]$

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 \mathbf{A} as two “basis” vectors: $[1 \ 2 \ 1] \ [-2 \ -3 \ 1]$
 - And new coordinates of : $[1 \ 0] \ [0 \ 1] \ [1 \ 1]$

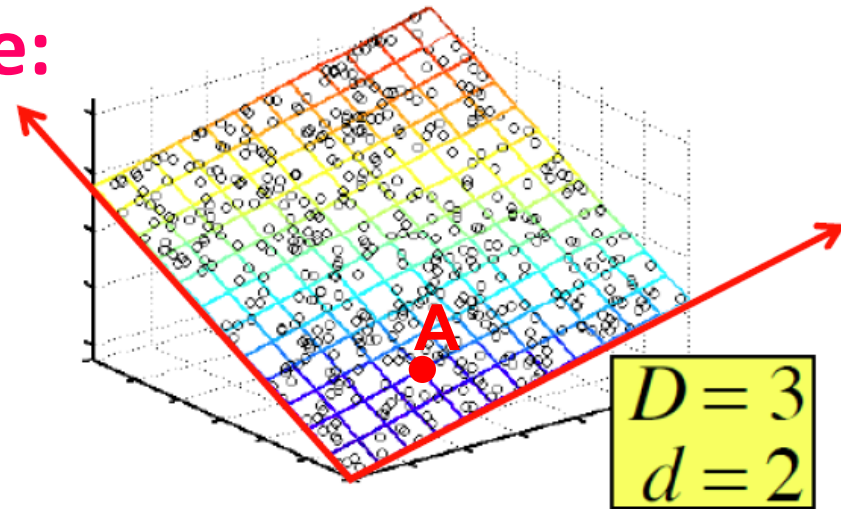
Rank is “Dimensionality”

- **Cloud of points 3D space:**

- Think of point positions

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

1 row per point:



- **We can rewrite coordinates more efficiently!**

- Old basis vectors: $[1 \ 0 \ 0]$ $[0 \ 1 \ 0]$ $[0 \ 0 \ 1]$

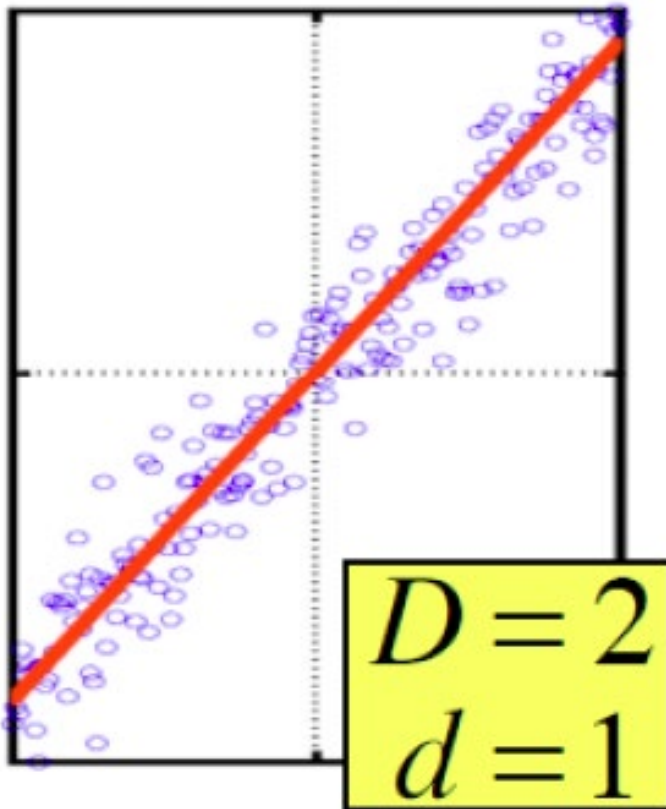
- **New basis vectors: $[1 \ 2 \ 1]$ $[-2 \ -3 \ 1]$**

- 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!



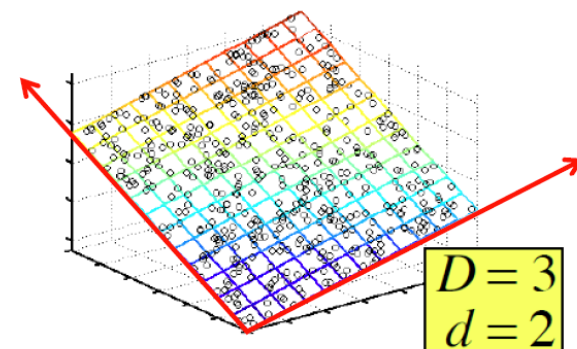
Rather than representing every point with 2 coordinates we represent each point with 1 coordinate (corresponding to the position of the point on the red line).

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**



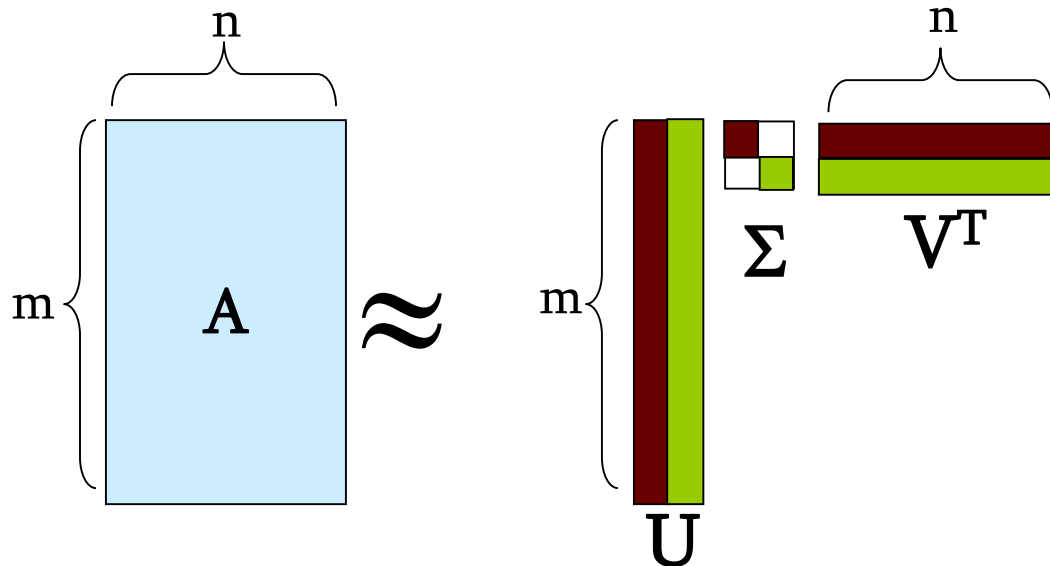
SVD - Definition

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

- **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)

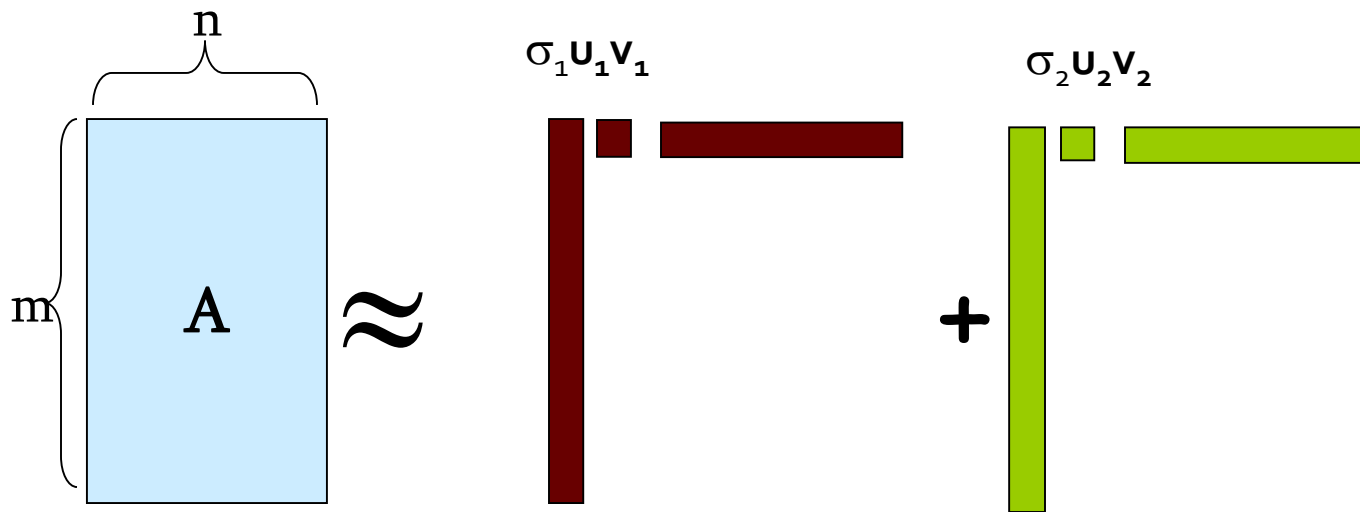
SVD

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



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

	Matrix	Alien	Serenity	Casablanca	Amelie													
↑	1	1	1	0	0	=	[0.13	0.02	-0.01	x	[12.4	0	0	x		
SciFi	3	3	3	0	0			0.41	0.07	-0.03			0	9.5	0			
↓	4	4	4	0	0			0.55	0.09	-0.04			0	0	1.3			
↑	5	5	5	0	0			0.68	0.11	-0.05			0	0	0			
Romnce	0	2	0	4	4			0.15	-0.59	0.65			0	0	0			
↓	0	0	0	5	5			0.07	-0.73	-0.67			0	0	0			
	0	1	0	2	2	0.07	-0.29	0.32	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	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” similarity matrix

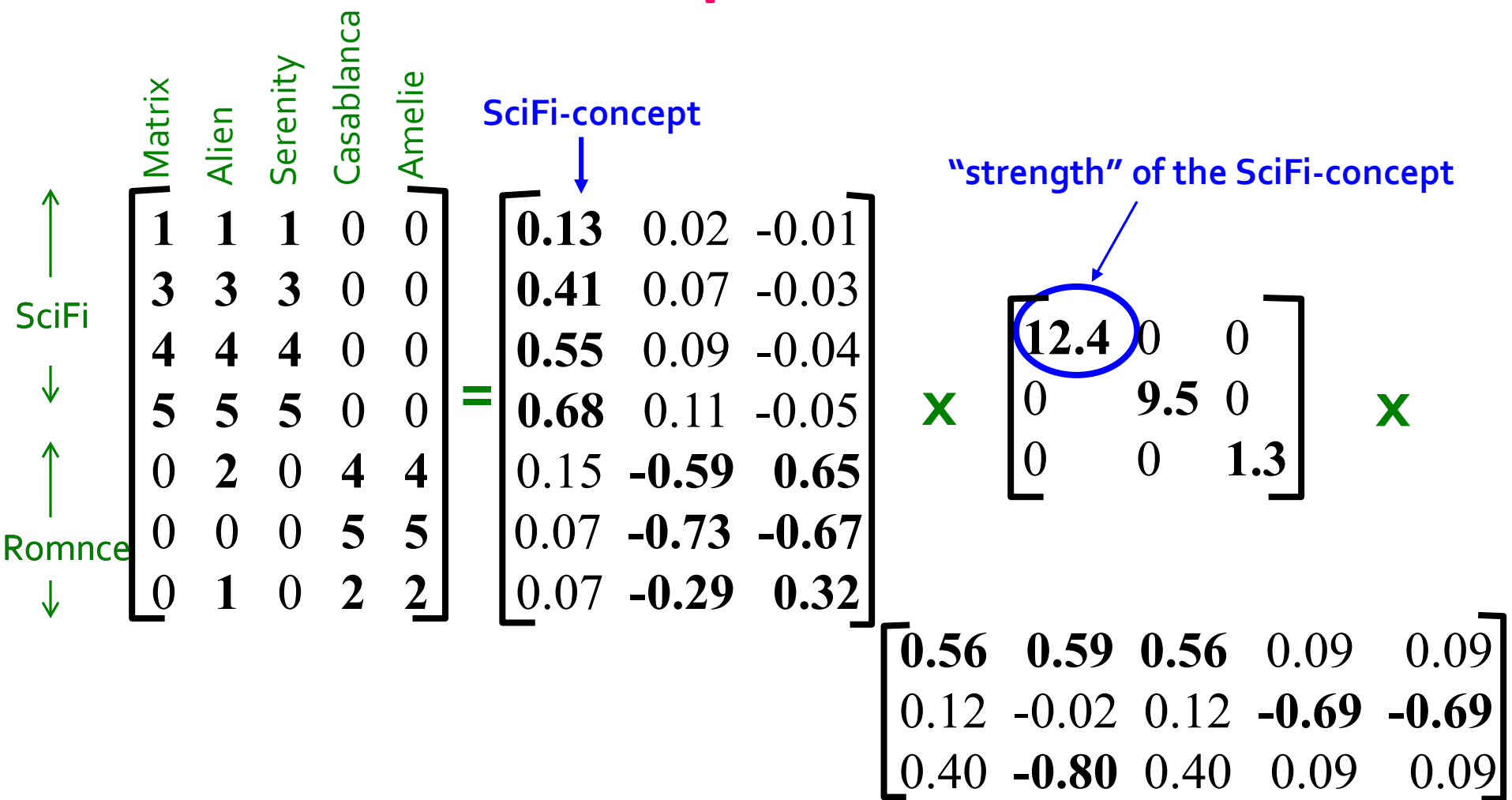
	Matrix	Alien	Serenity	Casablanca	Amelie		SciFi-concept	Romance-concept	
	1	1	1	0	0	=	0.13	0.02	-0.01
	3	3	3	0	0		0.41	0.07	-0.03
	4	4	4	0	0		0.55	0.09	-0.04
	5	5	5	0	0		0.68	0.11	-0.05
↑ SciFi	0	2	0	4	4		0.15	-0.59	0.65
↓	0	0	0	5	5		0.07	-0.73	-0.67
↑ Romnce	0	1	0	2	2		0.07	-0.29	0.32
↓									

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

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

SVD – Example: Users-to-Movies

■ $A = U \Sigma V^T$ - example:



SVD – in practice, I

```
42
43 # ----- SVD -----
44
45 ratings2 = np.array(ratings2)
46
47 print('data matrix ha shape ', ratings2.shape)
48
49 #split the matrix in the three components
50 u, s, vh = np.linalg.svd(ratings2, full_matrices=False)
51
52
53 print('U has shape ', u.shape, s.shape, vh.shape)
54
55 sigma = np.diag(s)
56 print('Sigma has shape ', sigma.shape)
57
58 print(sigma)
59 print('V^t has shape ', vh.shape)
60
61 sigmavh = np.dot(sigma, vh)
62
63 # A = U*Sigma*V^t
64
65 usigmavh = np.dot(u, sigmavh)
66
67 print(usigmavh)
68
69 # verify that the recombination is correct up to roundings
70 print(np.allclose(ratings2, usigmavh))
71
72
73
```

```
34 ratings2 = [[1, 1, 1, 0, 0],
35             [3, 3, 3, 0, 0],
36             [4, 4, 4, 0, 0],
37             [5, 5, 5, 0, 0],
38             [0, 2, 0, 4, 4],
39             [0, 0, 0, 5, 5],
40             [0, 1, 0, 2, 2]]
41
```

```
3 sigma =
4 [[1.24810147e+01 0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00]
5 [0.00000000e+00 9.50861406e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00]
6 [0.00000000e+00 0.00000000e+00 1.34555971e+00 0.00000000e+00 0.00000000e+00]
7 [0.00000000e+00 0.00000000e+00 0.00000000e+00 1.84716760e-16 0.00000000e+00]
8 [0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00 9.74452038e-33]]
```

SVD – in practice, II

```
34 ratings2 = [[1, 1, 1, 0, 0],
35             [3, 3, 3, 0, 0],
36             [4, 4, 4, 0, 0],
37             [5, 5, 5, 0, 0],
38             [0, 2, 0, 4, 4],
39             [0, 0, 0, 5, 5],
40             [0, 1, 0, 2, 2]]
41
42
43 # ----- SVD -----
44
45 ratings2 = np.array(ratings2)
46
47 print('data matrix ha shape ', ratings2.shape)
48
49 #split the matrix in the three components
50 u, s, vh = np.linalg.svd(ratings2, full_matrices=False)
51
52
53 print('U has shape ', u.shape, s.shape, vh.shape)
54
55 sigma = np.diag(s)
56 print('Sigma has shape ', sigma.shape)
57
58 print(sigma)
59 print('V^t has shape ', vh.shape)
60
61 sigmavh = np.dot(sigma, vh)
62
63 # A = U*Sigma*V^t
64
```

Name	Size	Type	Date Modified
> figures		File Folder	06/02/2019
> montecarlo-elections		File Folder	12/02/2019
activity_matrix2.png	86 KB	png File	31/01/2018
dsta-2018-19-class-3-dim_reduction.md	6 KB	md File	08/03/2018

Variable explorer File explorer

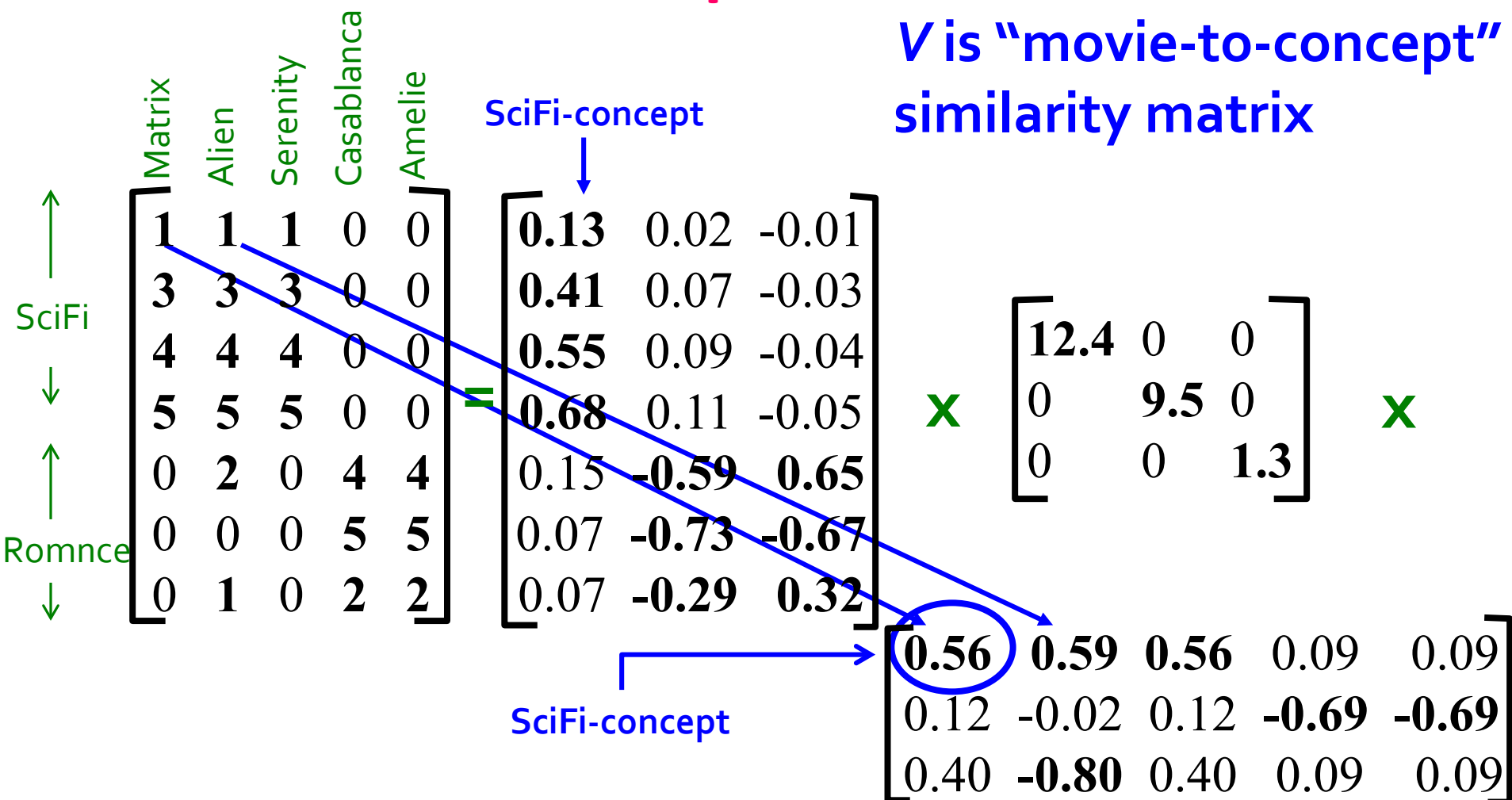
IPython console

Console 1/A

```
data matrix ha shape (7, 5)
U has shape (7, 5) (5,) (5, 5)
Sigma has shape (5, 5)
[[1.24810147e+01 0.00000000e+00 0.00000000e+00 0.00000000e+00
 0.00000000e+00]
 [0.00000000e+00 9.50861406e+00 0.00000000e+00 0.00000000e+00
 0.00000000e+00]
 [0.00000000e+00 0.00000000e+00 1.34555971e+00 0.00000000e+00
 0.00000000e+00]
 [0.00000000e+00 0.00000000e+00 0.00000000e+00 1.84716760e-16
 0.00000000e+00]
 [0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00
 9.74452038e-33]]
V^t has shape (5, 5)
[[ 1.00000000e+00  1.00000000e+00  1.00000000e+00  1.60615687e-16
  1.41232139e-16]
 [ 3.00000000e+00  3.00000000e+00  3.00000000e+00 -6.31770349e-17
 -1.20677157e-16]
 [ 4.00000000e+00  4.00000000e+00  4.00000000e+00  7.38751631e-17
 -2.79166592e-18]
 [ 5.00000000e+00  5.00000000e+00  5.00000000e+00 -9.67102611e-17
```

SVD – Example: Users-to-Movies

■ $A = U \Sigma V^T$ - example:



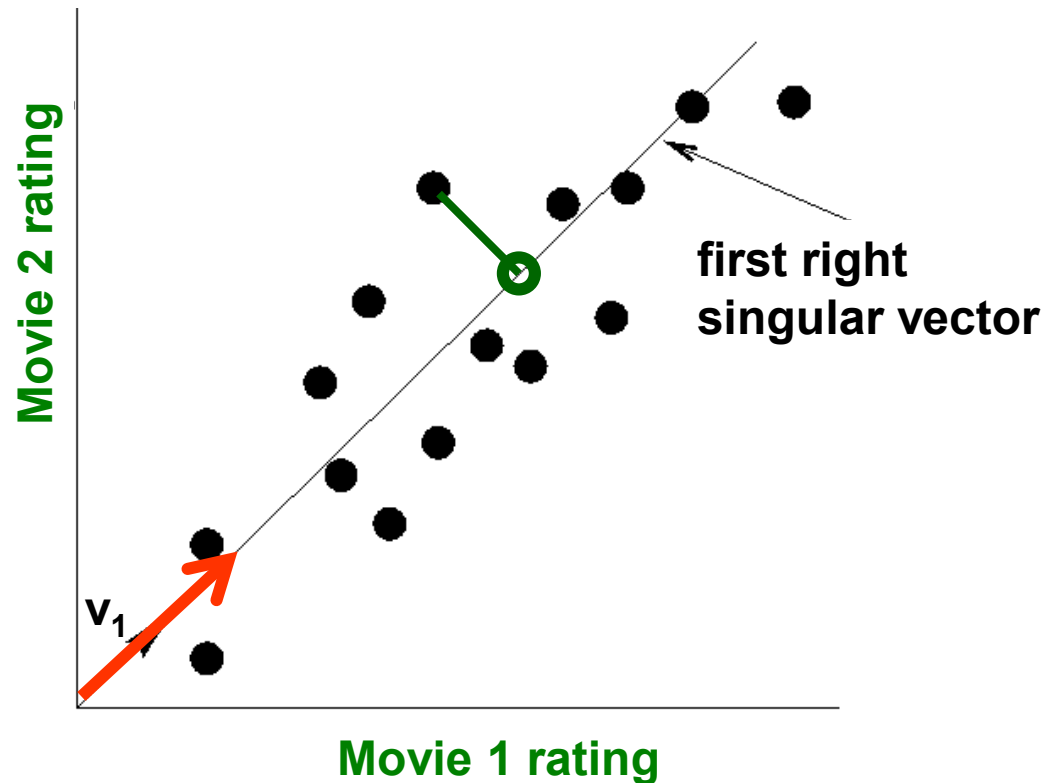
SVD - Interpretation #1

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

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

Dimensionality Reduction with SVD

SVD – Dimensionality Reduction



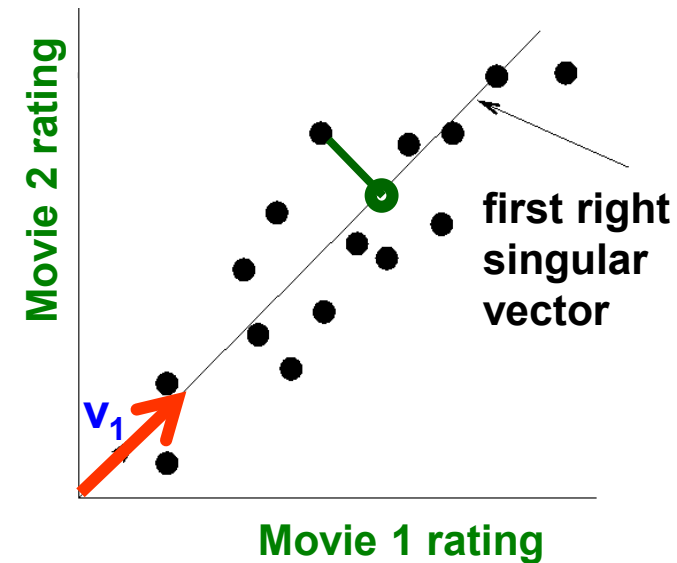
- Instead of using two coordinates (x, y) to describe point locations, let's use only one coordinate (z)
- Point's position is its location along vector v_1
- **How to choose v_1 ? Minimize reconstruction error**

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

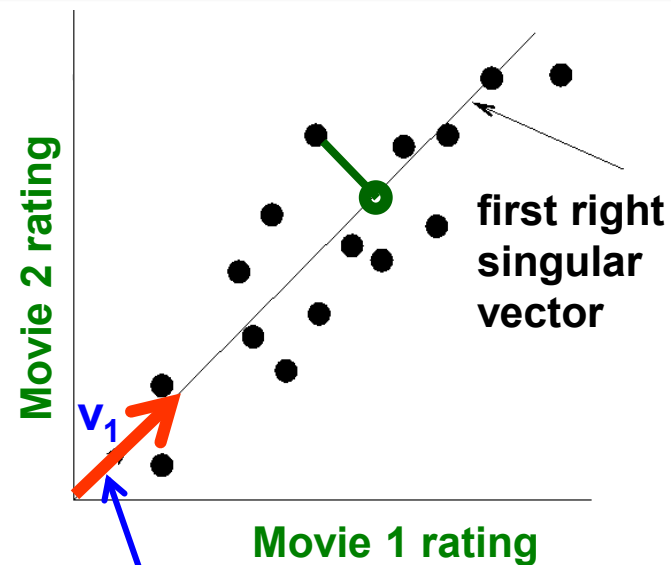
■ $A = U \Sigma V^T$ - example:

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

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

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

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

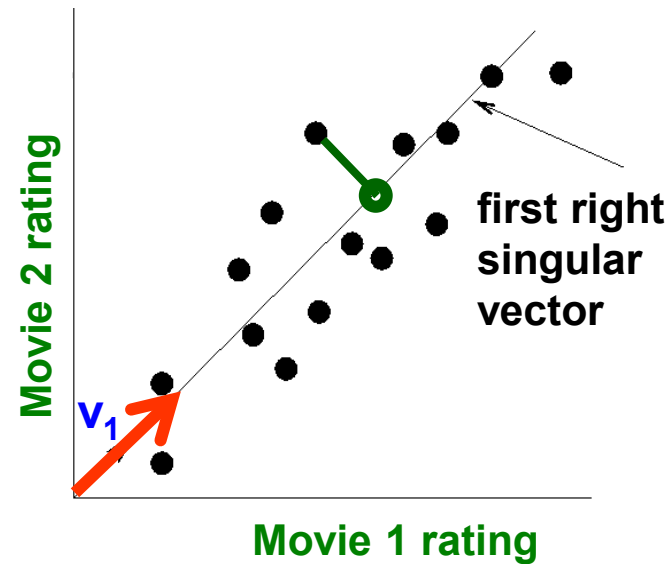


SVD - Interpretation #2

■ $A = U \Sigma V^T$ - example:

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

variance ('spread')
on the v_1 axis



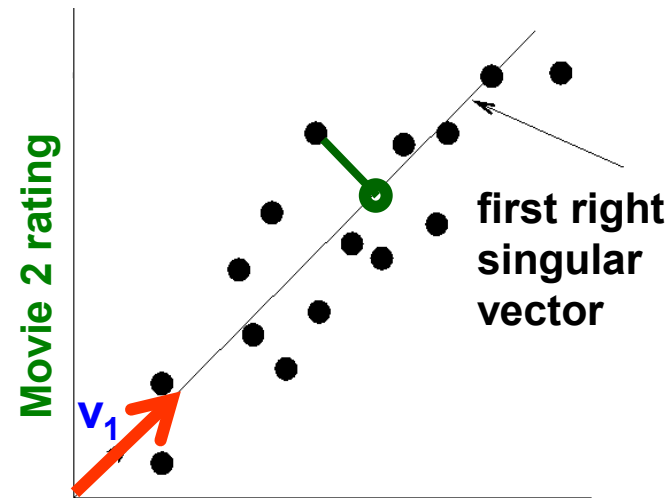
SVD - Interpretation #2

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

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

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

Projection of users
on the “Sci-Fi” axis
($U \Sigma$)^T:



Movie 1 rating

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?

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

SVD - Interpretation #2

More details

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

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

SVD - Interpretation #2

More details

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

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

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

More details

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

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

SVD - Interpretation #2

More details

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

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

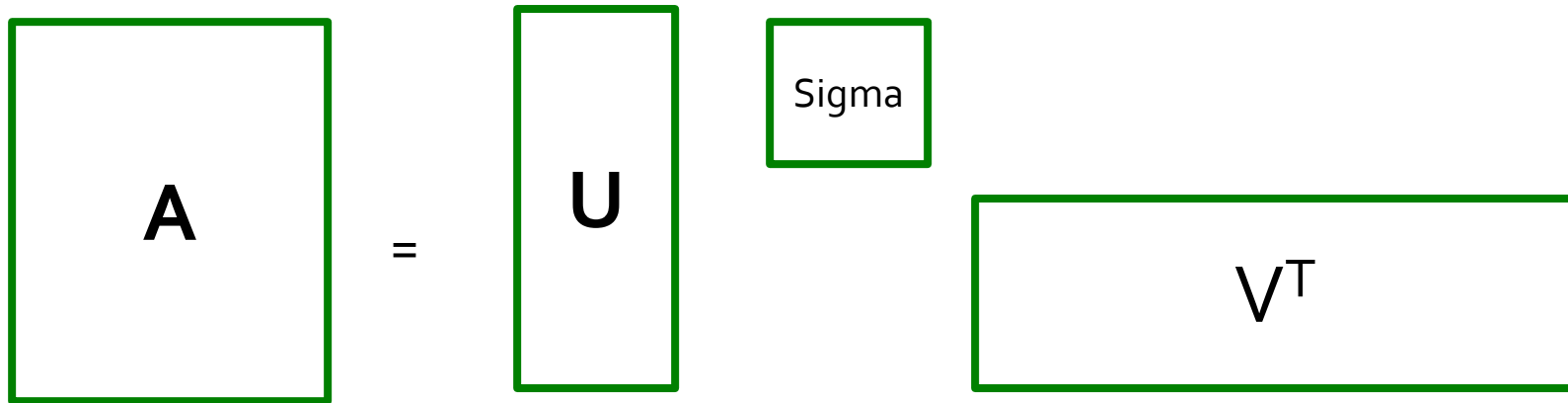
Frobenius norm:

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

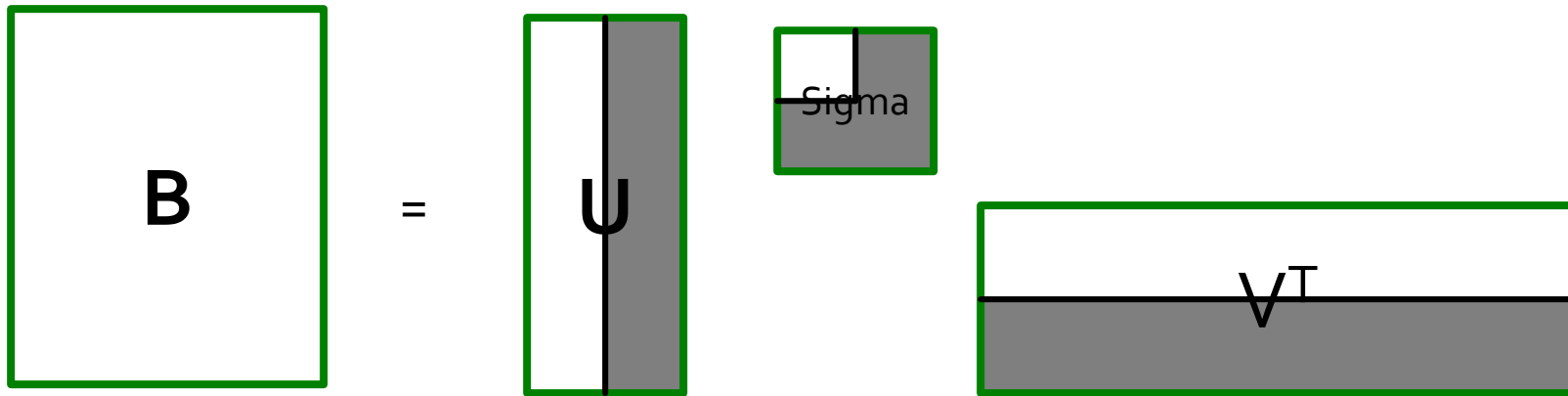
$$\|A-B\|_F = \sqrt{\sum_{ij} (A_{ij}-B_{ij})^2}$$

is "small"

SVD – Best Low Rank Approx.



B is best approximation of A



SVD – Best Low Rank Approx.

- Theorem:

Let $A = U \Sigma V^T$ 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 $\text{rank}(B) = k$

$$\begin{pmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & \\ \vdots & \vdots & \ddots & \\ x_{m1} & & & x_{mn} \end{pmatrix}_{m \times n} = \begin{pmatrix} U & \\ & \text{grey bar} \end{pmatrix}_{m \times r} \begin{pmatrix} \Sigma \\ & \text{grey bar} \end{pmatrix}_{r \times r} \begin{pmatrix} V^T \\ & \text{grey bar} \end{pmatrix}_{r \times n}$$

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

SVD - Interpretation #2

Equivalent:

'spectral decomposition' of the matrix:

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

SVD - Interpretation #2

Equivalent:

'spectral decomposition' of the matrix

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

Assume: $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \dots \geq 0$

Why is setting small σ_i to 0 the right thing to do?

Vectors \mathbf{u}_i and \mathbf{v}_i are unit length, so σ_i scales them.

So, zeroing small σ_i introduces less error.

SVD - Interpretation #2

Q: How many σ_s to keep?

A: Rule-of-a thumb:

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

$$\begin{array}{c} \uparrow \\ \downarrow \\ n \end{array} \begin{array}{c} \leftarrow m \rightarrow \\ \left[\begin{array}{ccccc} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{array} \right] \end{array} = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^T + \sigma_2 \mathbf{u}_2 \mathbf{v}_2^T + \dots$$

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

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

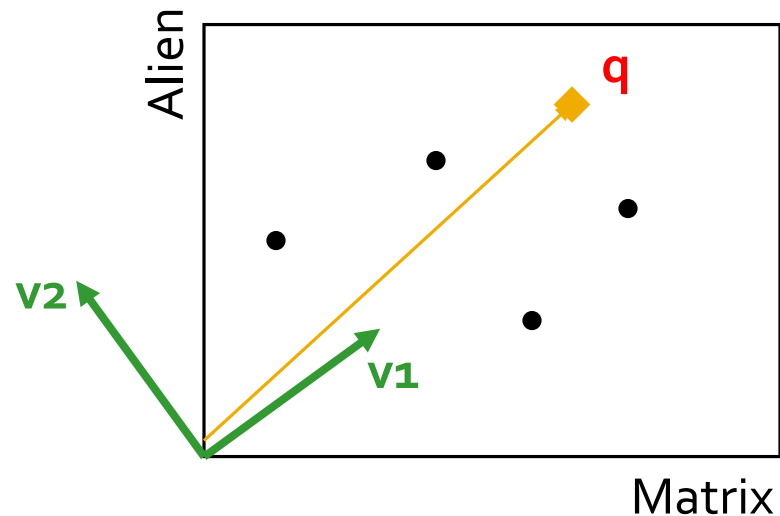
Example of SVD & Conclusion

Case study: How to query?

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

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

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

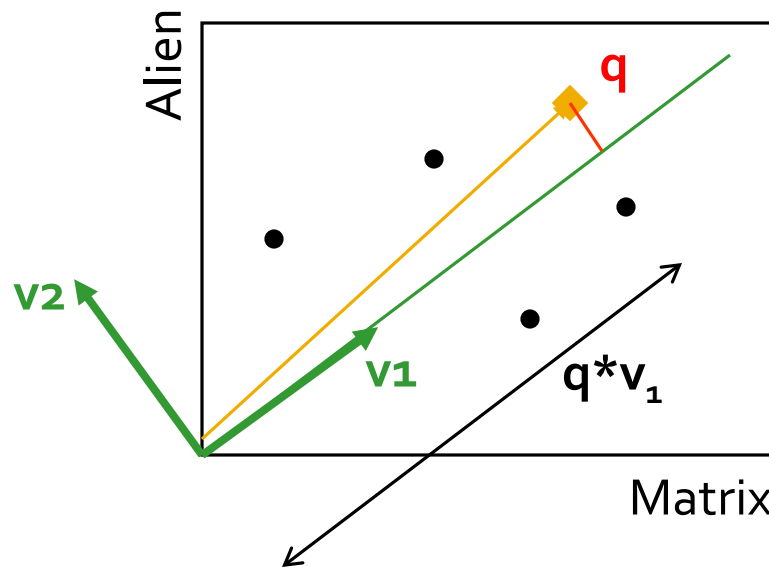


Case study: How to query?

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

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

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



Case study: How to query?

Compactly, we have:

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

E.g.:

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

movie-to-concept similarities (V)

Case study: How to query?

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

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

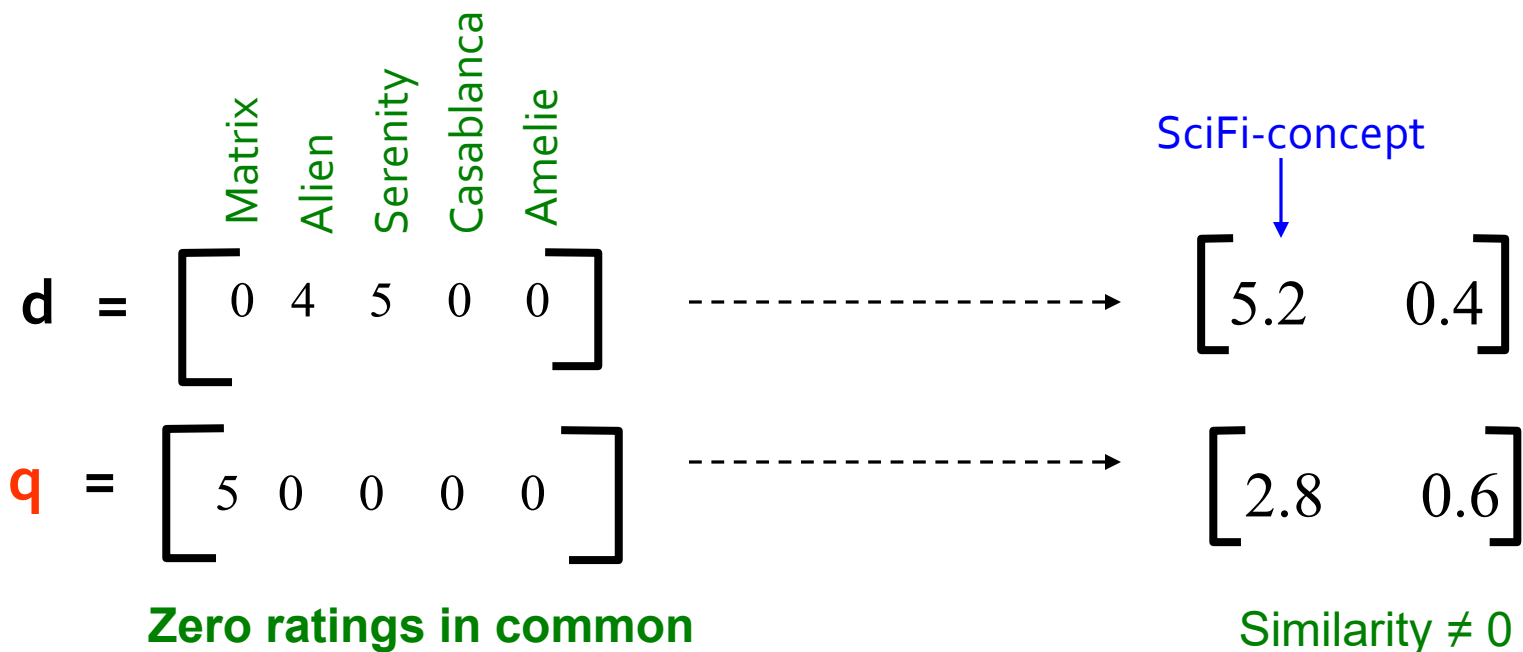
E.g.:

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

movie-to-concept similarities (V)

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!**

