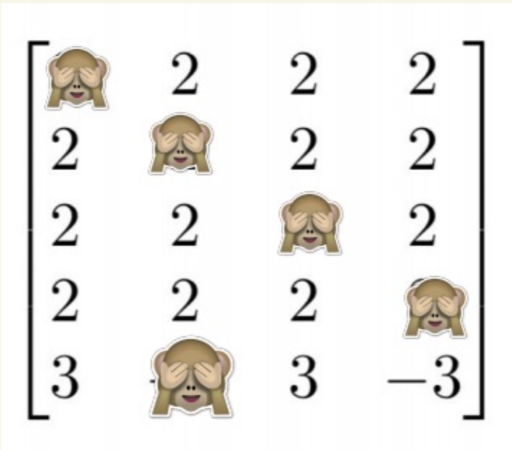
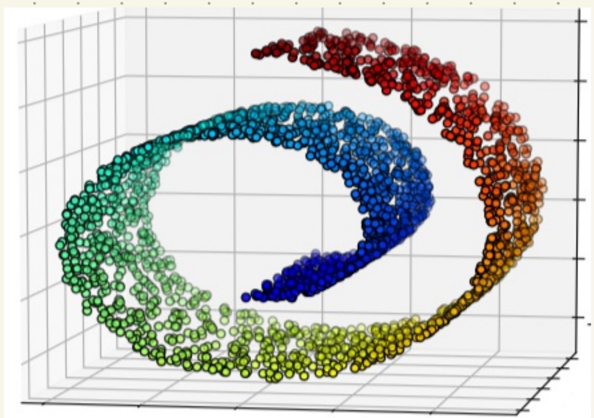
More on Principal Component Analysis



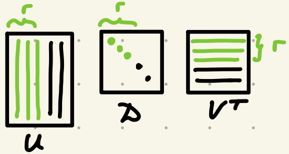
https://nhuang37.github.io/project/matrix/



https://www.thekerneltrip.com/statistics/tsne-vs-pca/

Low-rank matrix approximation

$$\underset{\hat{X}}{\text{minimize}}\ \underbrace{\|X-\hat{X}\|_F^2}_{\substack{x_1 \dots x_n \text{ represent}\\ \text{the observed signal}\\ \text{appoximate by } \hat{x}_1 .. \hat{x}_n}} \quad \text{subject to } \underbrace{\text{rank}(\hat{X})=r}_{\substack{\hat{x}_1 \dots \hat{x}_n \text{ belong to an}\\ r\text{-dimensional plane}\\ \Leftrightarrow \hat{X} \text{ low rank}}}$$

Solution: $X =$ 

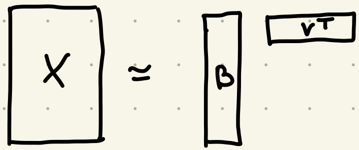
,

$$\hat{X} = \underbrace{U_{(r)} D_{(r)}}_{\text{"B"}} \cdot \underbrace{V_{(r)}^T}_{\text{"}V^T\text{"}} = SVD_r(X)$$

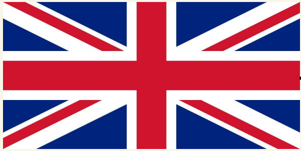
Applications:

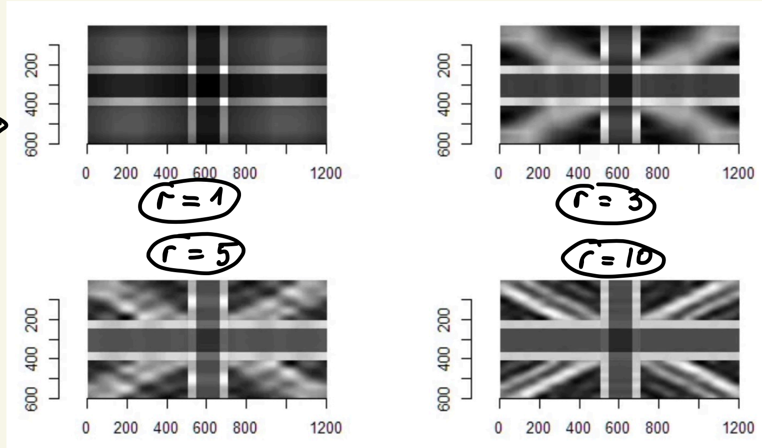
① Image compression/ dimension reduction

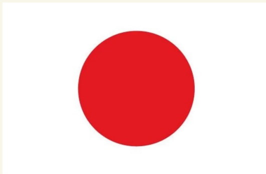
Instead of storing $n \times p$ matrix X
store $n \times r$ matrix B and $p \times r$ matrix V

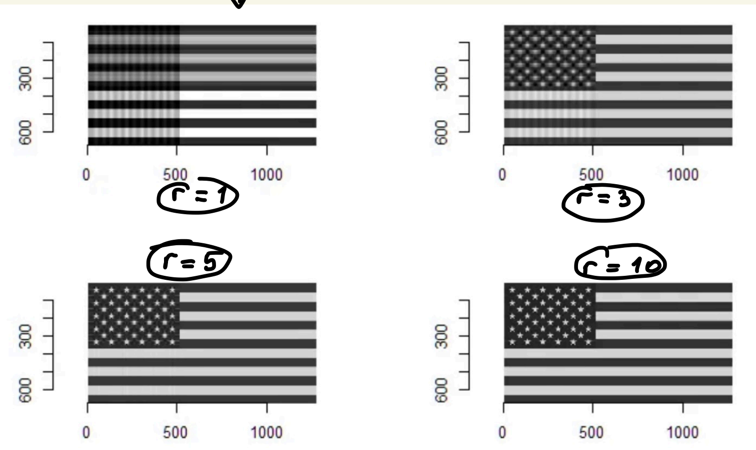


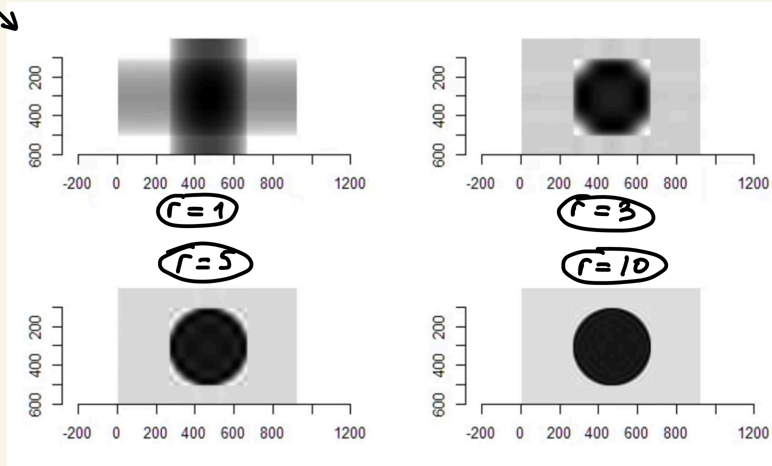
Example: flags









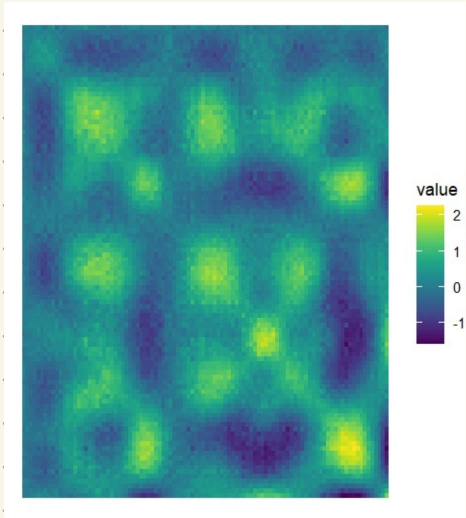


https://medium.com/towards-data-science/low-rank-approximation-for-flags-316cb19f7aba

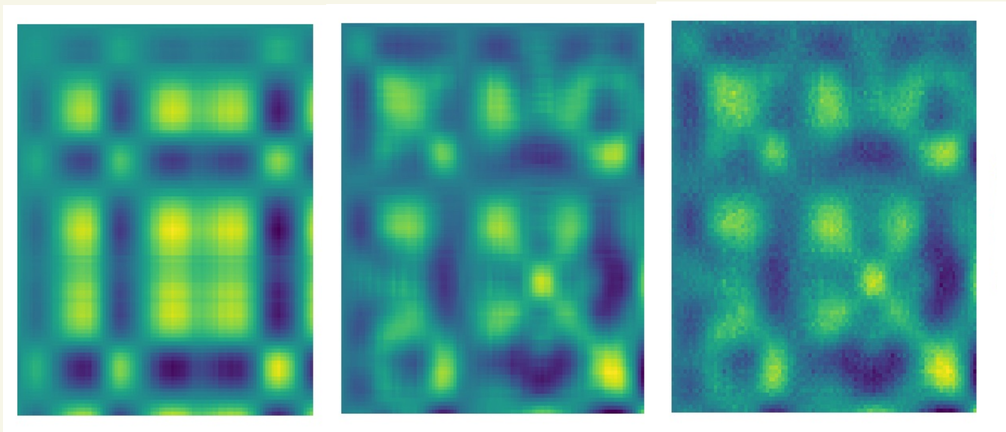
② De-noising

X is actually low-rank, but we observe only "noisy" version of X.

Input data X.



Approximation $\hat{X} = SVD_r(X)$



$r=1$ $r = 3$ $r = 20$

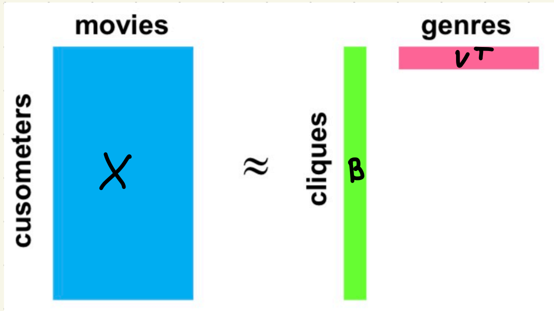
③ Imputation of missing values.

Netflix competition: build a recommendation system

Data: $n = 480189$ customers, $p = 17770$ movies

1% of values observed

	My Octopus Teacher	Invictus	Tsotsi	Catching Feelings	Mandela	The Kingfisher Caper	Skin	Escape from Pretoria	District 9	Angeliena
Customer 1	•	•	•	•	4	•	•	•	•	•
Customer 2	•	•	3	•	•	•	3	•	•	3
Customer 3	•	2	•	4	•	•	•	•	2	•
Customer 4	3	•	•	•	•	•	•	•	•	•
Customer 5	5	5	•	•	4	•	•	•	•	•
Customer 6	•	•	•	•	•	2	4	•	•	•
Customer 7	•	•	5	•	•	•	•	3	•	•
Customer 8	•	•	•	•	•	2	•	•	•	3
Customer 9	3	•	•	•	5	•	•	5	•	•
Customer 10	•	•	•	•	•	•	•	•	•	•



$$x_i \simeq \beta_{i1} V_1 + \dots + \beta_{ir} V_r$$

movie ranks for customer i

thriller

romance

does customer i likes thriller?

romance?

Low-rank matrix completion

Denote $\Omega \subseteq \{1 \dots n\} \times \{1 \dots p\}$ the set of observed entries in X.

Example: $X = \begin{pmatrix} NA & 1 \\ 0 & NA \\ NA & 2 \end{pmatrix}$ $\Omega = \{(1,2), (2,1), (3,2)\}$

The approximation error:

$$\sum_{(i,j)\in\Omega} (X_{ij} - \hat{X}_{ij})^2 = \|W * (X - \hat{X})\|_F^2$$

Here W is a "mask" matrix with $W_{ij} = \begin{cases} 1, & (i,j) \in \Omega \\ 0, & (i,j) \notin \Omega \end{cases}$ and $A * B$ denotes the Hadamard (elemen-wise) product between A and B, i.e. $(A * B)_{ij} = A_{ij} \cdot B_{ij}$.

Example: $W = \begin{pmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}$

$$\sum_{(i,j)\in\Omega} (X_{ij} - \hat{X}_{ij}) = \sum_{i=1}^{n} \sum_{j=1}^{p} \left(W_{ij} \cdot (X_{ij} - \hat{X}_{ij})\right)^2$$

Hard-impute algorithm

$$\underset{\hat{X}}{\text{minimize}} \; \| W * (X - \hat{X}) \|_F^2 \quad \text{subject to} \; \text{rank}(\hat{X}) = r$$

Input: matrix $X \in \mathbb{R}^{n \times p}$ and rank r,
and arbitrary $\hat{X} \in \mathbb{R}^{n \times p}$.

$$\left.\begin{array}{ll} \underline{\text{Step 1}} & Y = W * X + (1 - W) * \hat{X} \\ \underline{\text{Step 2}} & \hat{X} = SVD_r(Y) \end{array}\right\} \text{repeat until } \hat{X} \text{ converges}$$

Output: $\hat{X}$

Hard-impute steps:

- impute missing values in X with the elements from $\hat{X}$
- make $\hat{X}$ low-rank.

Example $X = \begin{pmatrix} NA & 1 \\ 0 & NA \\ NA & 2 \end{pmatrix}$ $W = \begin{pmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}$ $\hat{X} = \begin{pmatrix} a & b \\ c & d \\ e & f \end{pmatrix}$

Step 1 $Y = W * X + (1-W) * \hat{X}$

| $1-W = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{pmatrix} \Rightarrow Y = \begin{pmatrix} 0 & 1 \\ 0 & 0 \\ 0 & 2 \end{pmatrix} + \begin{pmatrix} a & 0 \\ 0 & d \\ e & 0 \end{pmatrix} = \begin{pmatrix} a & 1 \\ 0 & d \\ e & 2 \end{pmatrix}$

Step 2 $\hat{X} = SVD_r(Y)$

| $Y = UDV^T$ $\hat{X} = U_{(r)} D_{(r)} V_{(r)}^T = \begin{pmatrix} a' & b' \\ c' & d' \\ e' & f' \end{pmatrix}$

Computationae trick: if there are many missing values and r is small, use $Y = \underbrace{W * (X - \hat{X})}_{\text{sparse}} + \underbrace{\hat{X}}_{\text{low-rank}}$

| Store only $(X_{ij} - \hat{X}_{ij})$ for $(i,j) \in \Omega$
Store only $U_{(r)}, D_{(r)}, V_{(r)}$

"Soft" low-rank problem

① $\underset{\hat{X}}{\text{minimize}} \; \| X - \hat{X} \|_F^2$ subject to $\text{rank}(\hat{X}) = r$

Controlling $\text{rank}(\hat{X})$ $\Longleftrightarrow$ Controlling number of non-zero singular values (i.e. $d_i > 0$)

<u>Idea</u>: lets control $\sum_{i=1}^{p} d_i$ instead $\Longleftrightarrow$ control the nuclear norm $\|\hat{X}\|_*$

② $\underset{\hat{X}}{\text{minimize}} \; \| X - \hat{X} \|_F^2 + \lambda \|\hat{X}\|_*$

penalty factor

① $\mathrm{rank}(\hat{X}) = r$

Solution: $\hat{X} = SVD_r(X)$

$$X = \underset{U}{} \; D \; V^T$$

$$\downarrow$$

$$\hat{X} = U \; D^* \; V^T$$

$$D = \mathrm{diag}(d_1 \dots d_r, d_{r+1} \dots \dots d_p)$$

$$D^* = \mathrm{diag}(\underbrace{d_1 \dots d_r}_{r}, \underbrace{0 \dots\dots 0}_{p-r})$$

② $\dots + \lambda \|\hat{X}\|_*$

Solution: $\hat{X} = S_\lambda(X)$

$$X = U \; D \; V^T$$

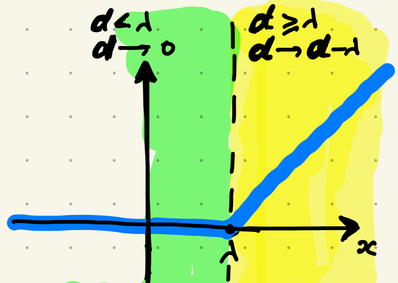
$$\downarrow$$

$$\hat{X} = U \; D^* \; V^T$$

$$D = \mathrm{diag}(d_1 \dots\dots\dots d_p)$$

$$D^* = \mathrm{diag}((d_1 - \lambda)_+ \dots (d_p - \lambda)_+)$$

Here $(d-\lambda)_+ = \max(d-\lambda, 0)$



Soft-impute algorithm

$$\underset{\hat{X}}{\text{minimize}} \; \|W * (X - \hat{X})\|_F^2 + \lambda \|\hat{X}\|_*$$

<u>Input</u>: matrix $X \in \mathbb{R}^{n \times p}$ and penalty λ.
and arbitrary $\hat{X} \in \mathbb{R}^{n \times p}$.

<u>Step 1</u> $\quad Y = W * X + (1-W) * \hat{X}$

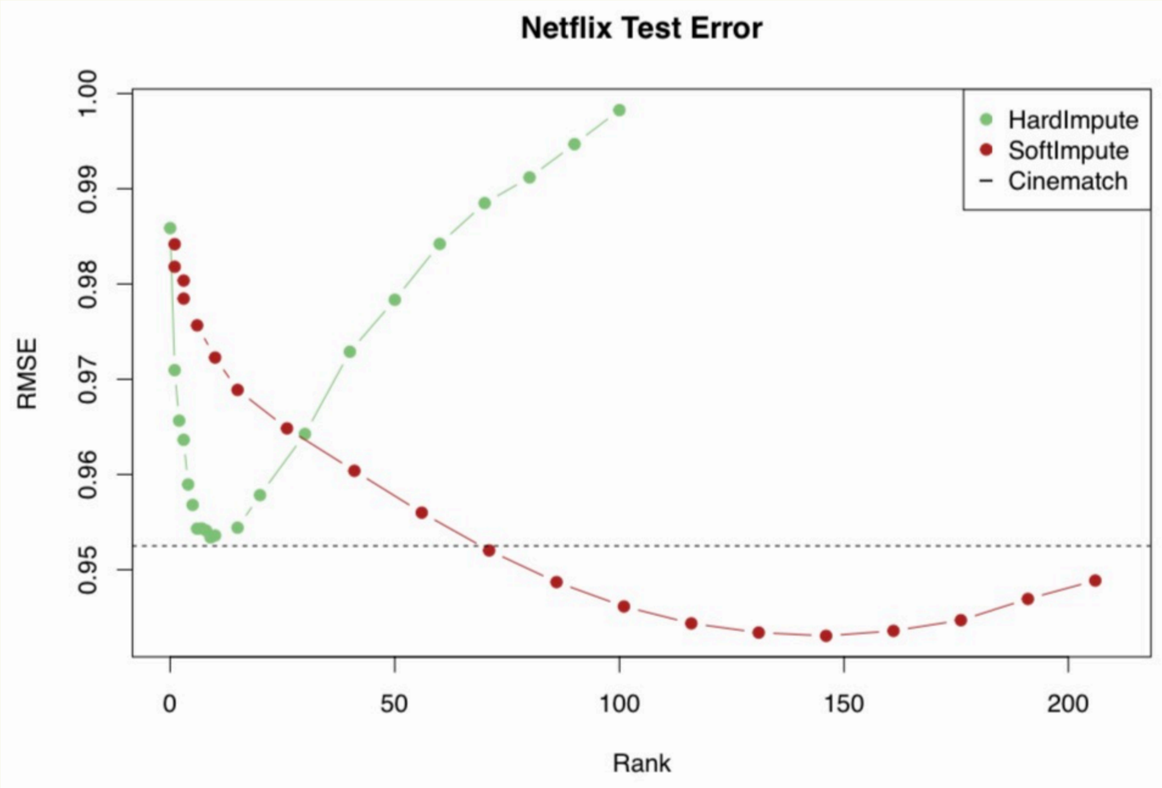
<u>Step 2</u> $\quad \hat{X} = S_\lambda(Y)$

$\Big\}$ repeat until $\hat{X}$ converges

<u>Output</u>: $\hat{X}$

Parameter λ balances off the approximation error and the "rank" of $\hat{X}$

| $\lambda \to 0 \Rightarrow X = \hat{X}$ and $\lambda \uparrow \Rightarrow \text{rank}(\hat{X}) \downarrow$



Cinematch: in-house Netflix algorithm