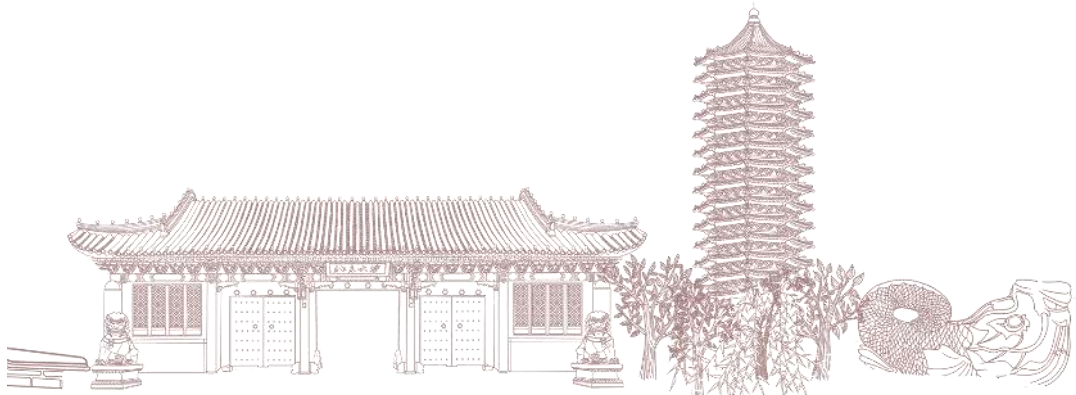
北京大学
PEKING UNIVERSITY

奇异值分解与主成分分析
SVD & PCA

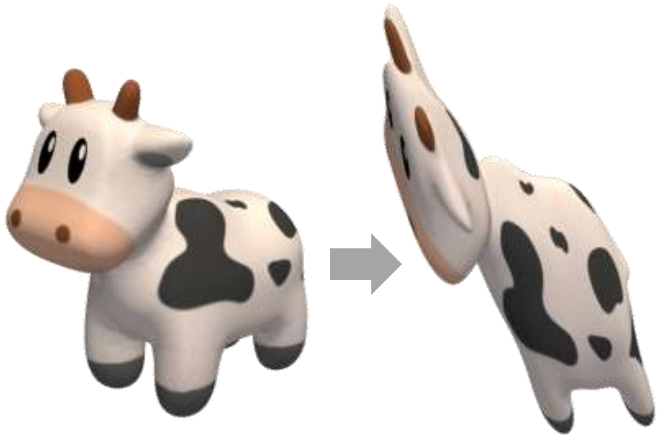
阮良旺

2024.4.1



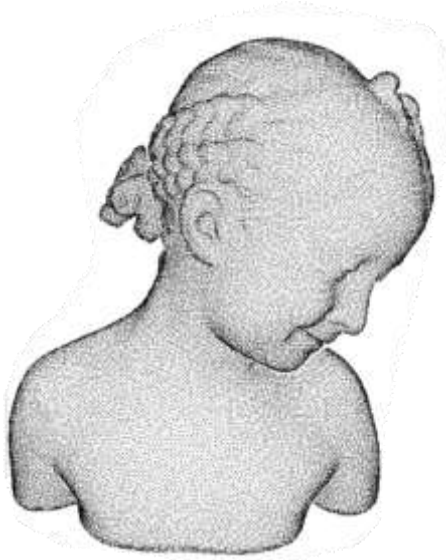
如何理解一个矩阵?

一个矩阵可以表达:



线性变换

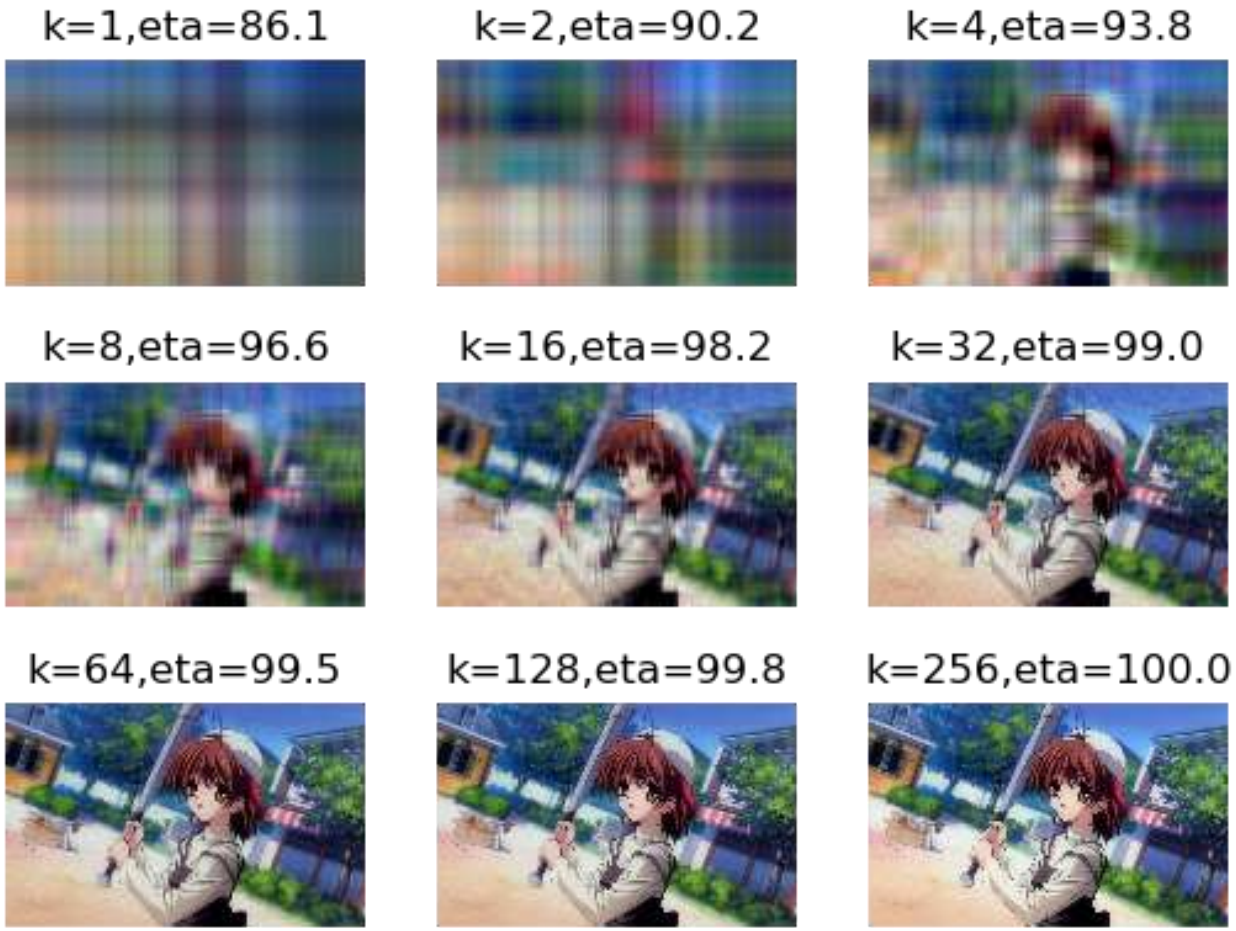
图片



几何数据

...

奇异值分解 (Singular Value Decomposition)



From Heyuan Yao

实对称矩阵特征值

实对称矩阵的特征值是实数

- $Sx = \lambda x, x$是(复)特征向量, $x^T x^* = 1$
- 共轭: $Sx^* = \lambda^* x^*, x^T Sx^* = x^T \lambda^* x^* = \lambda^*$
- 转置: $x^T S = \lambda x^T, x^T Sx^* = x^T \lambda x^* = \lambda$
- $\lambda = \lambda^*$， 特征值是实数

实对称矩阵不同特征值的特征向量相互正交

- $Sx_0 = \lambda_0 x_0, Sx_1 = \lambda_1 x_1$
- $\lambda_0 x_0^{\mathrm{T}} x_1 = x_0^{\mathrm{T}} S^{\mathrm{T}} x_1 = x_0^{\mathrm{T}} Sx_1 = \lambda_1 x_0^{\mathrm{T}} x_1$
- $\lambda_0 \neq \lambda_1, x_0^T x_1 = 0$

实对称矩阵的特征值分解

$$\mathrm{S} = \mathrm{Q}\Lambda\mathrm{Q}^{\mathrm{T}} = (q_0 \quad \cdots \quad q_{n-1}) \begin{pmatrix} \lambda_0 & & \\ & \ddots & \\ & & \lambda_{n-1} \end{pmatrix} \begin{pmatrix} q_0^{\mathrm{T}} \\ \vdots \\ q_{n-1}^{\mathrm{T}} \end{pmatrix}, \mathrm{Q}^{\mathrm{T}}\mathrm{Q} = \mathrm{I}$$

- 特征值λ_i有正有负，当 S 的秩为 r 时，λ_i 中只有 r 个值非零
- 如果λ_i都为非负数，则 S 是半正定矩阵 (Semi Positive Definite, SPD)
- 半正定矩阵满足$\forall x, x^{\mathrm{T}}Sx = x^{\mathrm{T}}\mathrm{Q}\Lambda\mathrm{Q}^{\mathrm{T}}x = \left\|\mathrm{Q}^{\mathrm{T}}x\right\|_{\Lambda}^{2} \geq 0$
- 对于半正定矩阵可以定义实对角矩阵 Σ 满足 $\Sigma^2 = \Lambda$
- 此时有 $\mathrm{S} = \mathrm{Q}\Lambda\mathrm{Q}^{\mathrm{T}} = \mathrm{Q}\Sigma^2\mathrm{Q}^{\mathrm{T}} = \left(\mathrm{Q}\Sigma\mathrm{Q}^{\mathrm{T}}\right)\left(\mathrm{Q}\Sigma\mathrm{Q}^{\mathrm{T}}\right) = \mathrm{P}^2 \Longrightarrow \mathrm{P} = \mathrm{S}^{1/2}$
- $\mathrm{Tr}(\mathrm{S}) = \mathrm{Tr}\left(\mathrm{Q}\Lambda\mathrm{Q}^{\mathrm{T}}\right) = \mathrm{Tr}\left(\mathrm{Q}^{\mathrm{T}}\mathrm{Q}\Lambda\right) = \mathrm{Tr}(\Lambda) = \sum_i \lambda_i$

奇异值分解

- 对于任意矩阵 $\mathrm{A} \in \mathbb{R}^{m \times n}$，秩为$r$，$\mathrm{S} = \mathrm{A}^{\mathrm{T}}\mathrm{A}$ 是对称半正定矩阵
- $Sv_i = \lambda_i v_i$, $\{v_i\}$组成一组r个正交基
- $\mathrm{A}^{\mathrm{T}}\mathrm{A}v_i = \lambda_i v_i$, $v_i^{\mathrm{T}}\mathrm{A}^{\mathrm{T}}\mathrm{A}v_j = \lambda_j v_i^{\mathrm{T}} v_j = \lambda_j \delta_{ij}$
- 定义$u_i = \frac{1}{\sqrt{\lambda_i}} A v_i$，则有$u_i^{\mathrm{T}} u_j = \delta_{ij}$，$\{u_i\}$组成一组$r$个正交基
- 可以通过Schmidt正交化将$\{v_i\}$扩充为$\mathbb{R}^n$的n个正交基，将$\{u_i\}$扩充为$\mathbb{R}^m$的m个正交基，扩充的基满足$\sqrt{\lambda_j} u_j = 0 = \mathrm{A}v_j$
- 将$\{u_i\}$和$\{v_i\}$的关系写成矩阵有

$$\mathrm{AV} = \mathrm{U}\Sigma，\ \mathrm{V}和\mathrm{U}是正交矩阵$$

奇异值分解

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

任意矩阵
$\mathbb{R}^{m\times n}$
$\text{Rank} = r$

正交矩阵
$\mathbb{R}^{m\times m}$

“对角”矩阵

$$\Sigma_{ii} = \begin{cases} \sqrt{\lambda_i}, & i \le r \\ 0, & i > r \end{cases}$$

正交矩阵
$\mathbb{R}^{n\times n}$

奇异值分解

奇异值分解是线性代数的中非常重要的概念，在图形学中的应用也非常广泛。当矩阵的奇异值分解已知时，我们可以得到：

- 矩阵的逆（求解矩阵方程）
- 矩阵的秩
- 矩阵方程的最小二乘解
- 极分解
- …

矩阵的逆与SVD

对于可逆矩阵$\mathrm{A} \in \mathbb{R}^{n \times n}$:

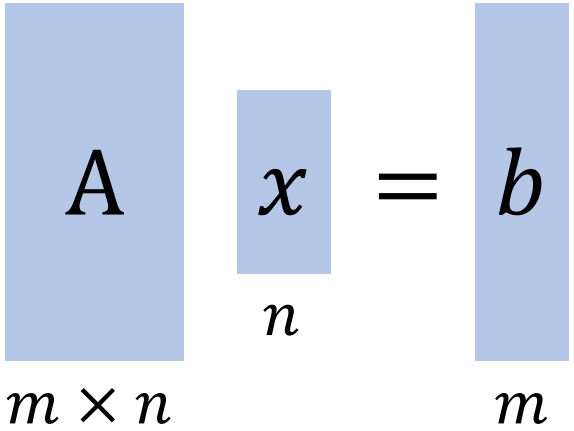
$$\mathrm{A} = \mathrm{U}\Sigma\mathrm{V}^{\mathrm{T}}$$

得到 A 的逆为:

$$\mathrm{A}^{-1} = \left(\mathrm{U}\Sigma\mathrm{V}^{\mathrm{T}}\right)^{-1} = \left(\mathrm{V}^{\mathrm{T}}\right)^{-1}\Sigma^{-1}\mathrm{U}^{-1} = \mathrm{V}\Sigma^{-1}\mathrm{U}^{\mathrm{T}}$$

$$\mathrm{A}^{-1} = \mathrm{V}\begin{pmatrix} 1/\sigma_0 & & \\ & \ddots & \\ & & 1/\sigma_{n-1} \end{pmatrix}\mathrm{U}^{\mathrm{T}}$$

矩阵的伪逆



假设 A 矩阵列满秩，但是行数大于列数 $m > n$

方程个数 m 大于自由度个数 n 时，解不一定存在 （超定情况）

可以寻找最小二乘解：

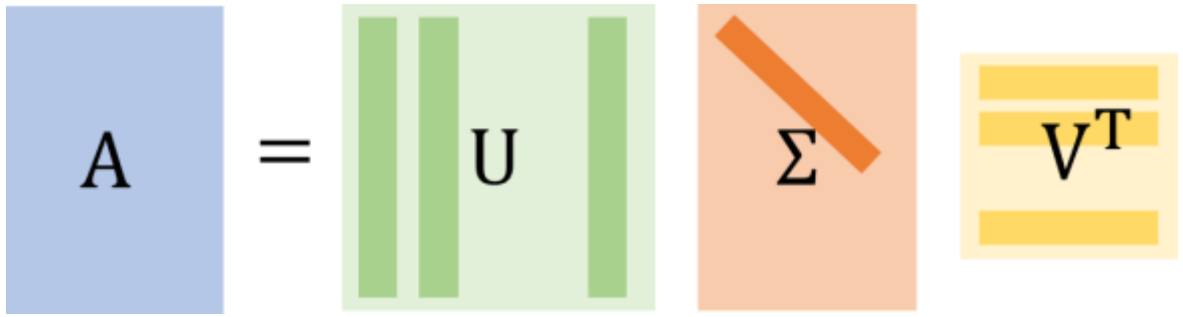
$$\min_{x} \|\mathrm{A}x - b\|^2$$

x 如何取到最小？

矩阵的伪逆

$$\min_{x} \|\mathrm{A}x - b\|^2$$

- $\mathrm{A}x - b = \mathrm{U}\Sigma\mathrm{V}^{\mathrm{T}}x - b = \mathrm{U}\left(\Sigma\mathrm{V}^{\mathrm{T}}x - \mathrm{U}^{\mathrm{T}}b\right)$
- $\|\mathrm{A}x - b\|^2 = \left\|\mathrm{U}\left(\Sigma\mathrm{V}^{\mathrm{T}}x - \mathrm{U}^{\mathrm{T}}b\right)\right\|^2 = \left\|\Sigma\mathrm{V}^{\mathrm{T}}x - \mathrm{U}^{\mathrm{T}}b\right\|^2$
- 令$x' = \mathrm{V}^{\mathrm{T}}x$，$b' = \mathrm{U}^{\mathrm{T}}b$，有$\|\mathrm{A}x - b\|^2 = \|\Sigma x' - b'\|^2$



矩阵的伪逆

- $\Sigma x' = (\sigma_0 x'_0, \cdots, \sigma_{n-1} x'_{n-1}, 0, 0, \cdots)$
- $\|\Sigma x' - b'\|^2$取到最小时：

$$\sigma_0 x'_0 = b'_0, \cdots, \sigma_{n-1} x'_{n-1} = b'_{n-1}$$

- 也即$x' = \hat{\Sigma}^{-1} b'$：

$$x = V\hat{\Sigma}^{-1}U^T b$$

矩阵的伪逆
（Pseudo Inverse）

$$\Sigma = \begin{bmatrix} \sigma_0 & & \\ & \ddots & \\ & & \sigma_{n-1} \end{bmatrix} \qquad \hat{\Sigma}^{-1} = \begin{bmatrix} 1/\sigma_0 & & \\ & \ddots & \\ & & 1/\sigma_{n-1} \end{bmatrix}$$

从矩阵求导推导

$$g = \|\mathrm{A}x - b\|^2 = x^{\mathrm{T}}\mathrm{A}^{\mathrm{T}}\mathrm{A}x - x^{\mathrm{T}}\mathrm{A}^{\mathrm{T}}b - b^{\mathrm{T}}\mathrm{A}x + b^{\mathrm{T}}b$$

$$\frac{\partial g}{\partial x} = 2\mathrm{A}^{\mathrm{T}}\mathrm{A}x - 2\mathrm{A}^{\mathrm{T}}b = 0$$

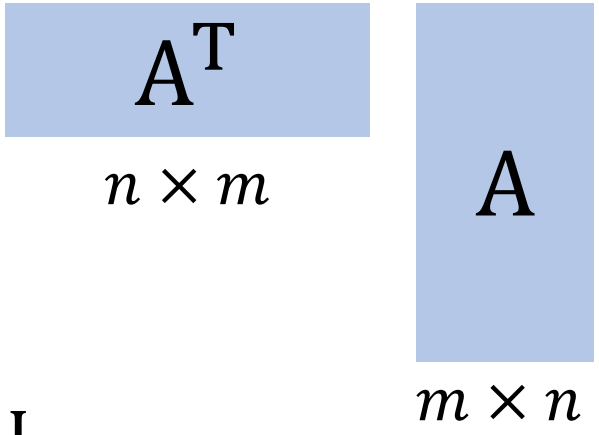
$$\left(\mathrm{A}^{\mathrm{T}}\mathrm{A}\right)x = \mathrm{A}^{\mathrm{T}}b$$

$\mathrm{A}^{\mathrm{T}}\mathrm{A} \in \mathbb{R}^{n\times n}$ 满秩，于是有：

$$x = \left(\mathrm{A}^{\mathrm{T}}\mathrm{A}\right)^{-1}\mathrm{A}^{\mathrm{T}}b$$

这又称为矩阵的左逆 (Left Inverse)，因为：

$$\left(\mathrm{A}^{\mathrm{T}}\mathrm{A}\right)^{-1}\mathrm{A}^{\mathrm{T}}\mathrm{A} = \mathrm{I}, \qquad \mathrm{A}\left(\mathrm{A}^{\mathrm{T}}\mathrm{A}\right)^{-1}\mathrm{A}^{\mathrm{T}} \neq \mathrm{I}$$

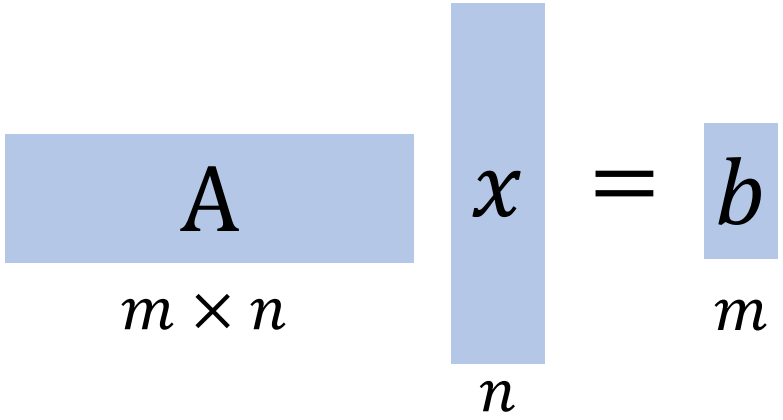


左逆与伪逆

- 左逆: $\left(A^T A\right)^{-1} A^T$
- $A^T A = V\Sigma^T U^T U\Sigma V^T = V\Sigma^T \Sigma V^T = V\Lambda V^T$
- $\left(A^T A\right)^{-1} A^T = V\Lambda^{-1} V^T V\Sigma^T U^T = V\Lambda^{-1}\Sigma^T U^T = V\hat{\Sigma}^{-1} U^T$
- 矩阵列满秩时，左逆与伪逆等价

$$\Sigma = \begin{bmatrix} \sigma_0 & & \\ & \ddots & \\ & & \sigma_{n-1} \end{bmatrix} \quad \Lambda = \begin{bmatrix} \sigma_0^2 & & \\ & \ddots & \\ & & \sigma_{n-1}^2 \end{bmatrix} \quad \hat{\Sigma}^{-1} = \begin{bmatrix} 1/\sigma_0 & & \\ & \ddots & \\ & & 1/\sigma_{n-1} \end{bmatrix}$$

行满秩的情况



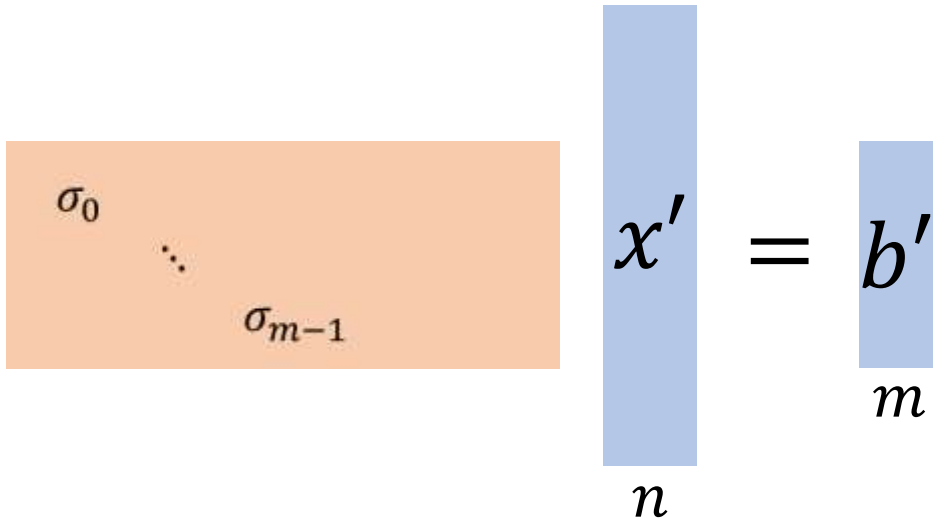
- A 矩阵行满秩，但是 $m < n$
- 此时自由度个数 n 大于方程个数 m，有无穷多组解（欠定情况）
- 可以寻求下面的最小二乘解：

$$\min_{x} \|x\|^2$$
$$s.t. Ax - b = 0$$

行满秩的情况

- $\mathrm{U}\Sigma\mathrm{V}^{\mathrm{T}}x = b, x' = \mathrm{V}^{\mathrm{T}}x, b' = \mathrm{U}^{\mathrm{T}}b$
- $\mathrm{U}\Sigma\mathrm{V}^{\mathrm{T}}x = \mathrm{U}\mathrm{U}^{\mathrm{T}}b \Leftrightarrow \Sigma x' = b'$
- $\sigma_0 x'_0 = b'_0, \cdots, \sigma_{m-1}x'_{m-1} = b'_{m-1}$
- 剩下的$x'_m, \cdots, x'_{n-1}$可以取任意值
- $\|x'\|^2 = x^{\mathrm{T}}\mathrm{V}\mathrm{V}^{\mathrm{T}}x = x^{\mathrm{T}}x = \|x\|^2$
- $\|x\|^2$取最小时，$x'_m = \cdots = x'_{n-1} = 0$
- $x' = \hat{\Sigma}^{-1}b'$
- $x = \mathrm{V}\hat{\Sigma}^{-1}\mathrm{U}^{\mathrm{T}}b$

矩阵的伪逆



$$\hat{\Sigma}^{-1} = \begin{bmatrix} 1/\sigma_0 & & \\ & \ddots & \\ & & 1/\sigma_{m-1} \end{bmatrix}$$

矩阵的右逆

- $\hat{\Sigma}^{-1} = \Sigma^{\mathrm{T}}\left(\Sigma\Sigma^{\mathrm{T}}\right)^{-1}$
- $\mathrm{V}\hat{\Sigma}^{-1}\mathrm{U}^{\mathrm{T}} = \mathrm{A}^{\mathrm{T}}\left(\mathrm{A}\mathrm{A}^{\mathrm{T}}\right)^{-1}$
- $\mathrm{A}^{\mathrm{T}}\left(\mathrm{A}\mathrm{A}^{\mathrm{T}}\right)^{-1}$ 称为矩阵的右逆 (Right Inverse)：

$$\mathrm{A}\mathrm{A}^{\mathrm{T}}\left(\mathrm{A}\mathrm{A}^{\mathrm{T}}\right)^{-1} = \mathrm{I}, \qquad \mathrm{A}^{\mathrm{T}}\left(\mathrm{A}\mathrm{A}^{\mathrm{T}}\right)^{-1}\mathrm{A} \neq \mathrm{I}$$

$$\min_{x} \|x\|^2 \quad s.t. Ax - b = 0$$

直接求解这个带约束的优化问题需要**拉格朗日乘子法**，我们将在后面的课程中介绍

$$\hat{\Sigma}^{-1} = \begin{bmatrix} 1/\sigma_0 & & \\ & \ddots & \\ & & 1/\sigma_{m-1} \end{bmatrix} \qquad \Sigma = \begin{bmatrix} \sigma_0 & & \\ & \ddots & \\ & & \sigma_{m-1} \end{bmatrix} \qquad \Sigma\Sigma^{\mathrm{T}} = \begin{bmatrix} \sigma_0^2 & & \\ & \ddots & \\ & & \sigma_{m-1}^2 \end{bmatrix}$$

左逆，右逆，伪逆

- 当矩阵列满秩时，左逆 $\left(\mathrm{A}^{\mathrm{T}}\mathrm{A}\right)^{-1}\mathrm{A}^{\mathrm{T}}$ 存在，用于求解超定方程
- 当矩阵行满秩时，右逆 $\mathrm{A}^{\mathrm{T}}\left(\mathrm{A}\mathrm{A}^{\mathrm{T}}\right)^{-1}$ 存在，用于求解欠定方程
- 不管在什么情况下，矩阵的伪逆 $\mathrm{V}\hat{\Sigma}^{-1}\mathrm{U}^{\mathrm{T}}$ 一定存在

$$\Sigma = \begin{bmatrix} \sigma_0 & & \\ & \ddots & \\ & & \sigma_{r-1} \end{bmatrix} \qquad \hat{\Sigma}^{-1} = \begin{bmatrix} 1/\sigma_0 & & \\ & \ddots & \\ & & 1/\sigma_{r-1} \end{bmatrix}$$

几何变换与SVD

$$x' \leftarrow \mathrm{A}x = \mathrm{U}\Sigma\mathrm{V}^{\mathrm{T}}x$$

$x' \leftarrow \mathrm{V}^{\mathrm{T}}x$

旋转

$x' \leftarrow \Sigma x$

拉伸

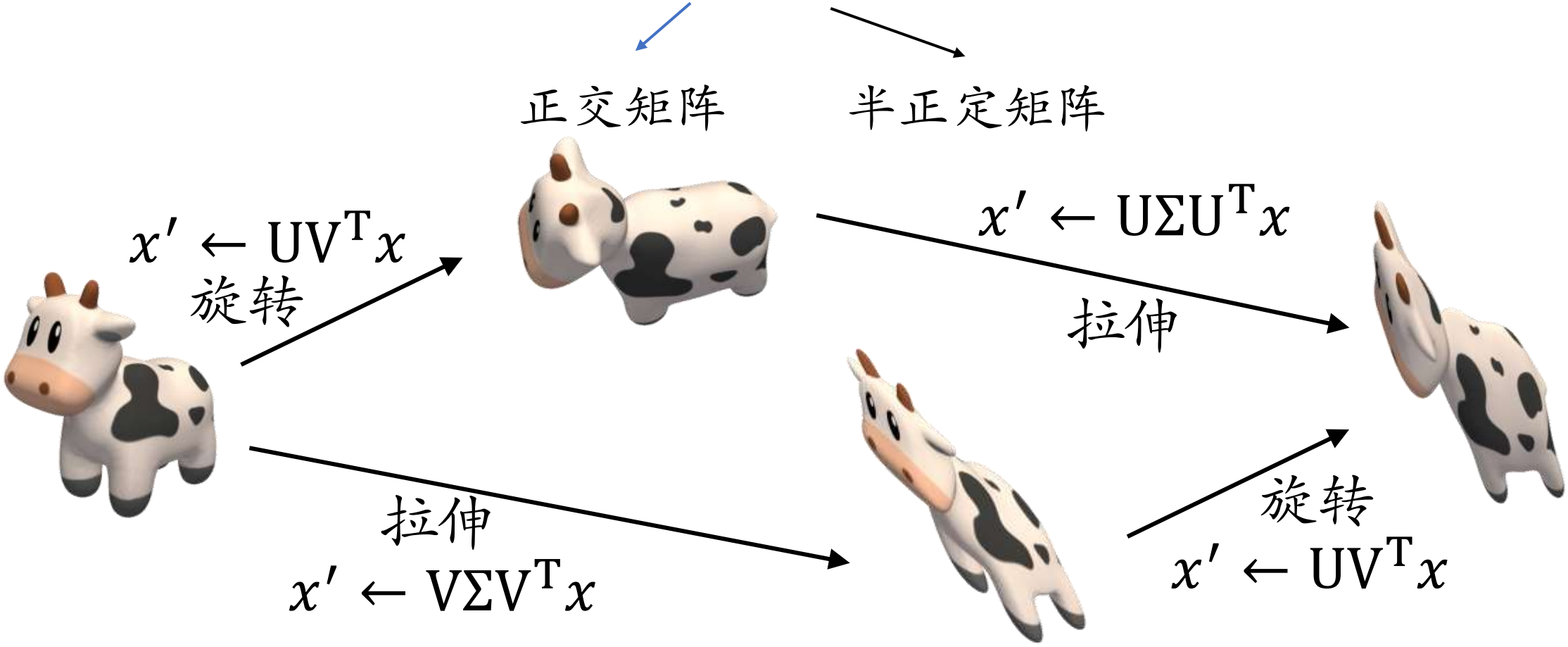
$x' \leftarrow \mathrm{U}x$

旋转

注意通过调节Σ_{ii}的正负号保证U和V的行列式为+1

极分解 (Polar Decomposition)

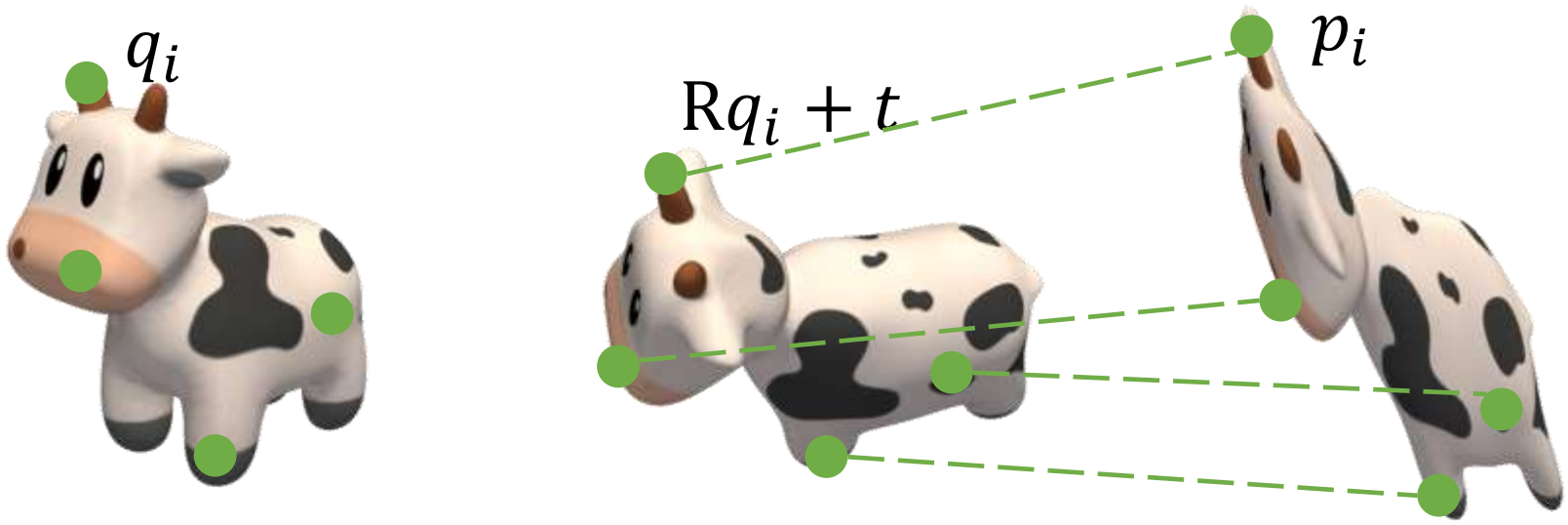
- 对于任意实方阵 A，存在唯一的正交矩阵R和半正定矩阵S, 使得$A = RS$
- 从SVD的角度: $A = U\Sigma V^T = UV^T V\Sigma V^T = U\Sigma U^T UV^T$



形状匹配 (Shape Matching)

对于形变$q \to p$，如何得到刚性变换近似：

$$\mathrm{R}, \mathrm{t} = \operatorname{argmin} \sum_i \|p_i - (\mathrm{R}q_i + t)\|^2$$



形状匹配 (Shape Matching)

- 在R给定时，只考虑t

$$\frac{\partial}{\partial t}\sum_i \|p_i - (\mathrm{R}q_i + t)\|^2 = \sum_i \mathrm{R}q_i + t - p_i = 0$$

$$nt = \sum_i p_i - \mathrm{R}\sum_i q_i$$

形变前后的质心分别为$q_c = \frac{1}{n}\sum q_i, p_c = \frac{1}{n}\sum p_i$，则有

$$t = p_c - \mathrm{R}q_c$$

形状匹配 (Shape Matching)

$$t = p_c - \mathrm{R}q_c$$

$$\sum_i \|p_i - (\mathrm{R}q_i + t)\|^2 = \sum_i \|(p_i - p_c) - R(q_i - q_c)\|^2$$

定义相对质心的偏移 $p_i' = p - p_c, q_i' = q_i - q_c$，则R的优化问题为

$$\mathrm{R} = \mathrm{argmin} \sum_i \|p_i' - \mathrm{R}q_i'\|^2$$

形状匹配 (Shape Matching)

$$\sum_i \|p_i' - \mathrm{R}q_i'\|^2 = \sum_i \Big(\underbrace{p_i'^{\mathrm{T}} p_i'}_{\text{与R无关}} - p_i'^{\mathrm{T}} \mathrm{R} q_i' - (\mathrm{R}q_i')^{\mathrm{T}} p_i' + \underbrace{q_i'^{\mathrm{T}} \mathrm{R}^{\mathrm{T}} \mathrm{R} q_i'}_{\mathrm{R}^{\mathrm{T}}\mathrm{R} = \mathrm{I}} \Big)$$

$$\min \sum_i \|p_i' - \mathrm{R}q_i'\|^2 = \max \sum_i p_i'^{\mathrm{T}} \mathrm{R} q_i'$$

$$\sum_i p_i'^{\mathrm{T}} \mathrm{R} q_i' = \mathrm{Tr}\left(\sum_i p_i'^{\mathrm{T}} \mathrm{R} q_i' \right) = Tr\left(\mathrm{R} \sum_i q_i' p_i'^{\mathrm{T}} \right) = \blacksquare$$

$$\max_{\mathrm{R}} \mathrm{Tr}(\mathrm{RH}), \mathrm{H} = \sum_i q_i' p_i'^{\mathrm{T}}$$

形状匹配 (Shape Matching)

定理：对于对称正定矩阵 M 和正交矩阵 B，一定有

$$\mathrm{Tr}(\mathrm{M}) \geq \mathrm{Tr}(\mathrm{BM})$$

证明：

对于对称正定矩阵 M 进行特征值分解 $\mathrm{Q}\Lambda\mathrm{Q}^{\mathrm{T}}$

存在对称正定矩阵 $\mathrm{S} = \mathrm{Q}\Lambda^{1/2}\mathrm{Q}^{\mathrm{T}}, \mathrm{S}^2 = \mathrm{S}^{\mathrm{T}}\mathrm{S} = \mathrm{S}\mathrm{S}^{\mathrm{T}} = \mathrm{M}$

于是有 $\mathrm{Tr}(\mathrm{BM}) = \mathrm{Tr}(\mathrm{BS}^2) = \mathrm{Tr}(\mathrm{SBS}) = \mathrm{Tr}\left(\mathrm{S}^{\mathrm{T}}\mathrm{BS}\right)$

直接计算对角项 $\left(\mathrm{S}^{\mathrm{T}}\mathrm{S}\right)_{ii} = s_i^{\mathrm{T}}s_i$, $\left(\mathrm{S}^{\mathrm{T}}\mathrm{BS}\right)_{ii} = s_i^{\mathrm{T}}Bs_i$ (s_i是S的第i列)

于是有迹的关系 $\mathrm{Tr}(\mathrm{M}) = \mathrm{Tr}\left(\mathrm{S}^{\mathrm{T}}\mathrm{S}\right) = \sum_i s_i^{\mathrm{T}}s_i$, $\mathrm{Tr}\left(\mathrm{S}^{\mathrm{T}}\mathrm{BS}\right) = \sum_i s_i^{\mathrm{T}}\mathrm{B}s_i$

从而得到

$$\mathrm{Tr}(\mathrm{BM}) = \sum_i s_i^{\mathrm{T}}\mathrm{B}s_i \leq \sum_i s_i^{\mathrm{T}}s_i = \mathrm{Tr}(\mathrm{M})$$

正交矩阵不改变p_i的长度

形状匹配 (Shape Matching)

如果R_*可以使得 R_*H 为对称正定的矩阵，那么一定有

$$\mathrm{Tr}(RH) = \mathrm{Tr}\left(\underline{(RR_*^T)}\,\underline{(R_*H)}\right) \le \mathrm{Tr}(R_*H)$$

正交矩阵 B　对称矩阵 M

RH 何时对称正定?

$$H = \sum_i q_i' p_i'^T = U\Sigma V^T$$

如果$\det(VU^T) = -1$?

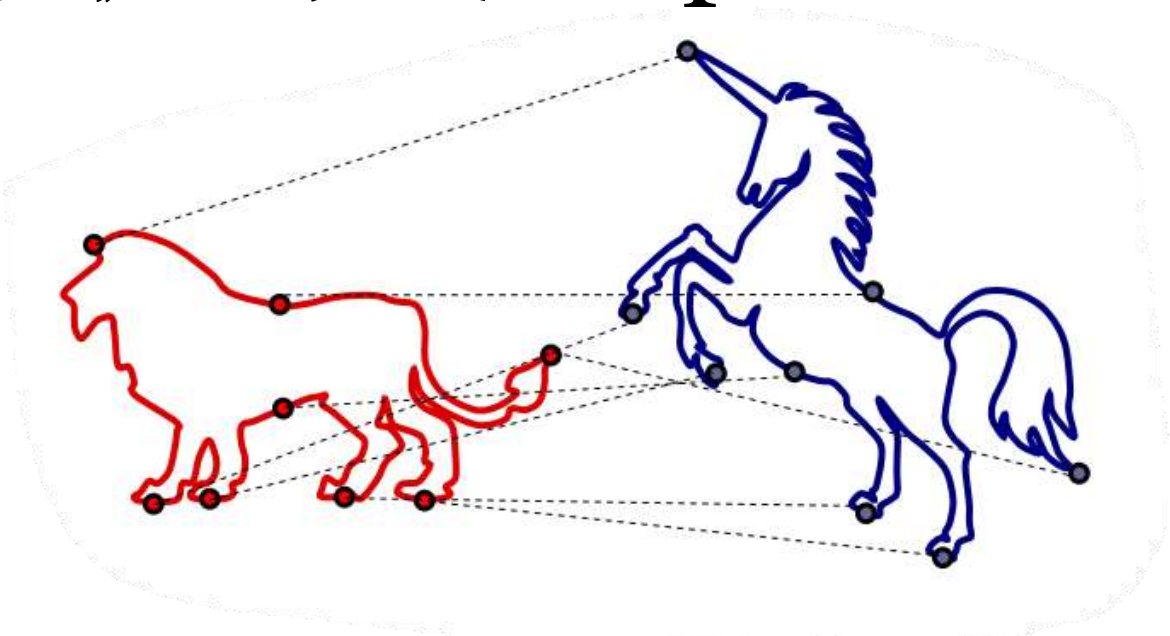
$$R = V\begin{pmatrix} 1 & & \\ & 1 & \\ & & \det(VU^T) \end{pmatrix}U^T$$

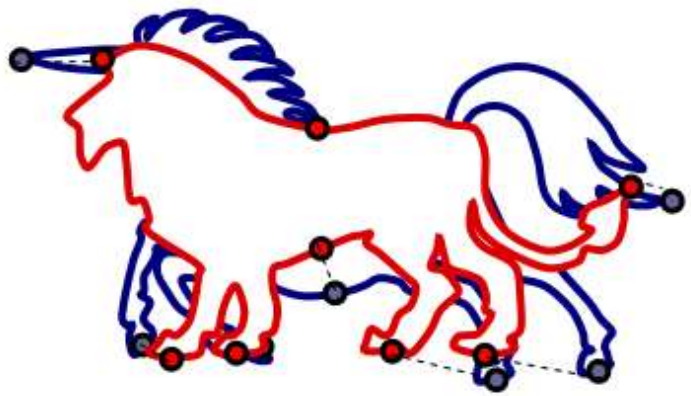
https://igl.ethz.ch/projects/ARAP/svd_rot.pdf

$$R = VU^T$$

$$RH = V\Sigma V^T$$

形状匹配 (Shape Matching)

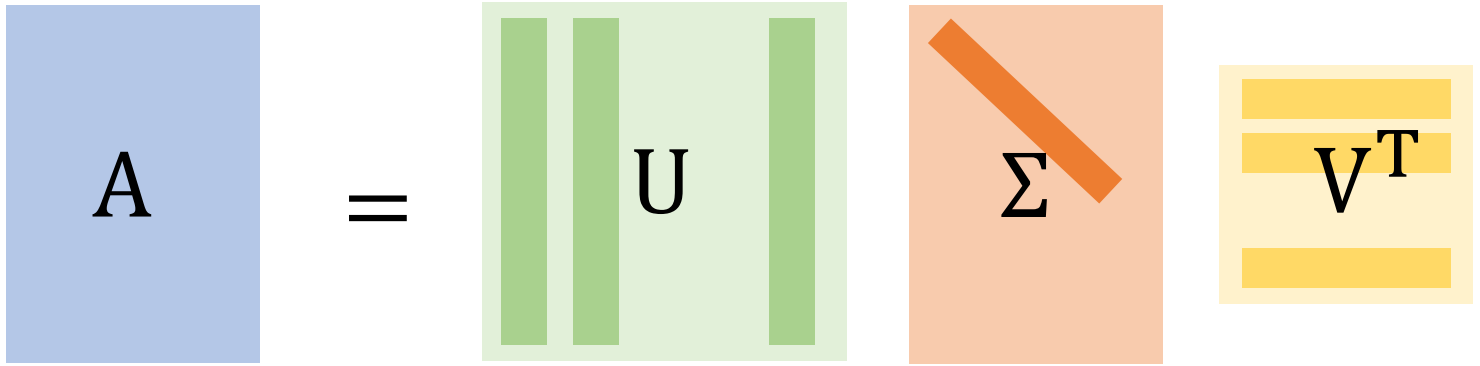


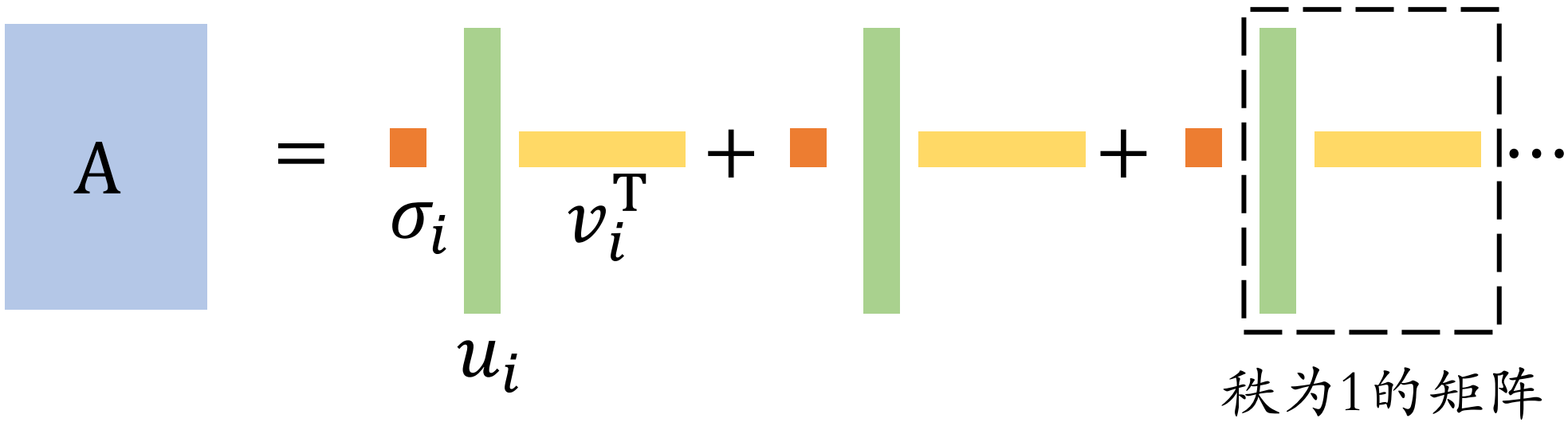


- 计算相对质心的偏移 $q_i' = q_i - q_c, p_i' = p_i - p_c$
- 构造 $\mathrm{H} = \sum_i q_i' p_i'^{\mathrm{T}}$，进行SVD分解 $\mathrm{H} = \mathrm{U}\Sigma\mathrm{V}^{\mathrm{T}}$
- 得到最优旋转 $\mathrm{R} = \mathrm{V}\mathrm{U}^{\mathrm{T}}$
- 得到相对平移 $t = p_c - \mathrm{R}q_c$

示例2：图像压缩

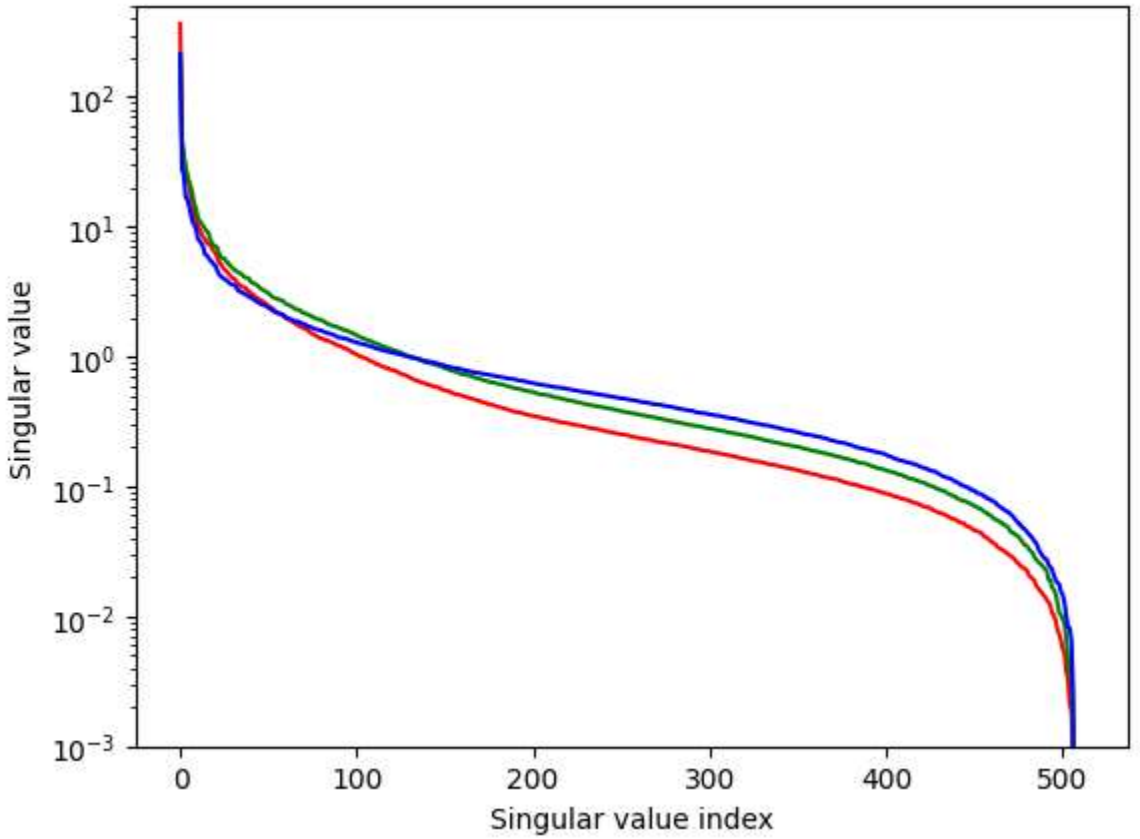
- 分辨率为$m \times n$的图像可以看成一个大型的稠密矩阵 $A \in \mathbb{R}^{m \times n}$



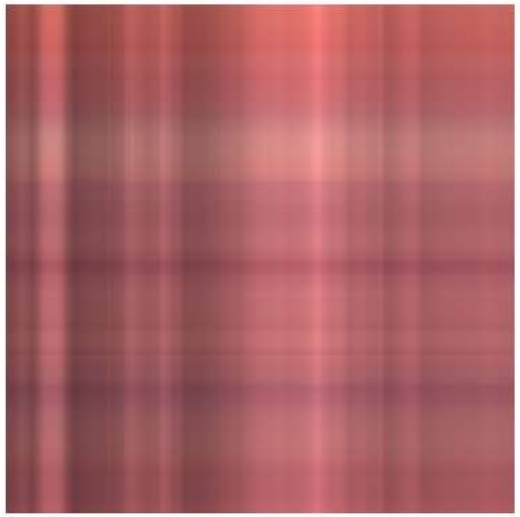
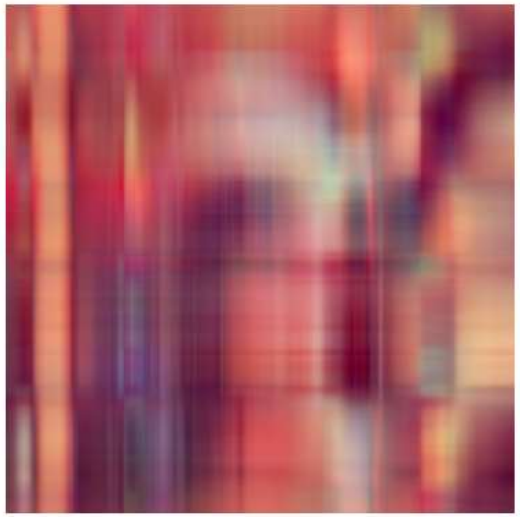


奇异值分布

512×512



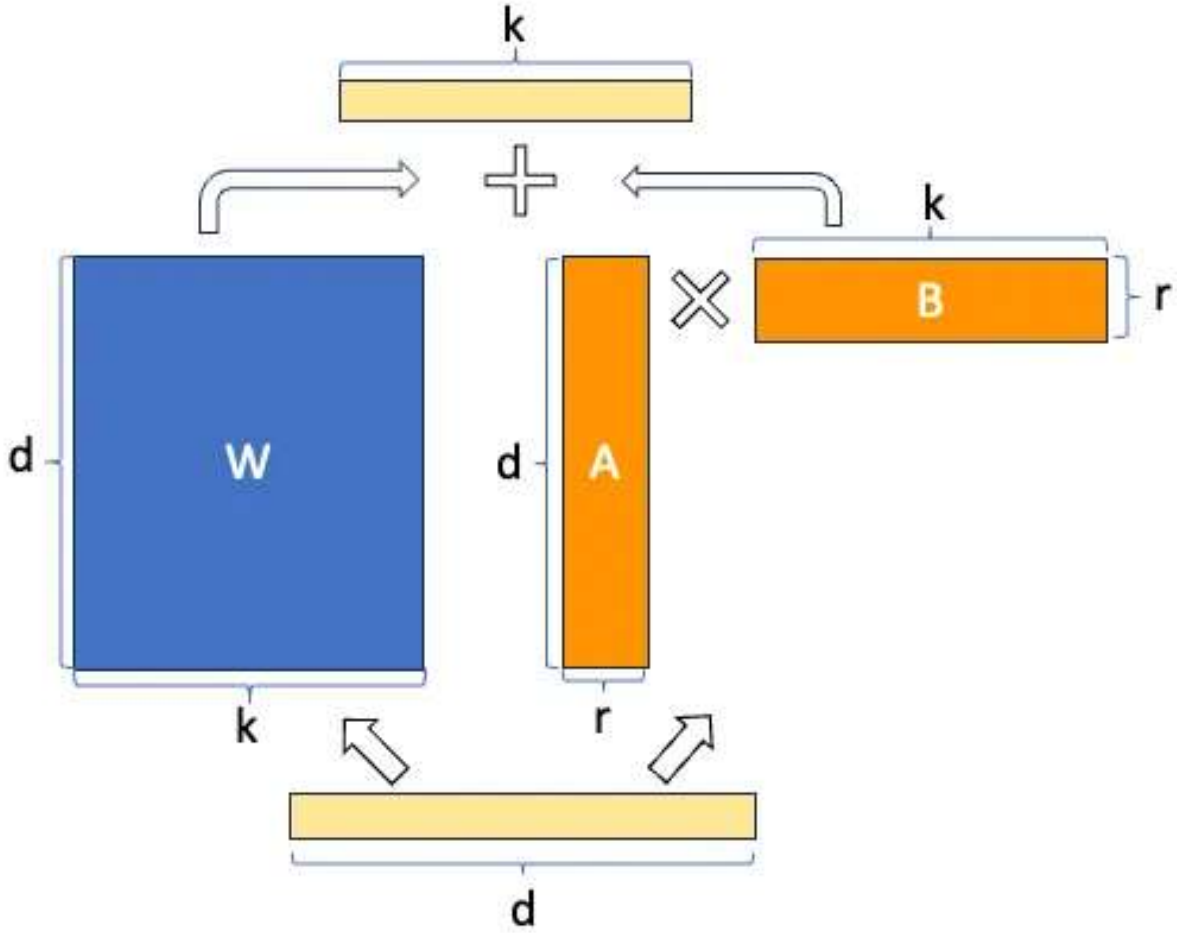
图像压缩

特征值个数	1	4	16	64
压缩率	0.19%	0.78%	3.15%	12.5%

LORA

Low-**R**ank **A**daptation of Large Language Models



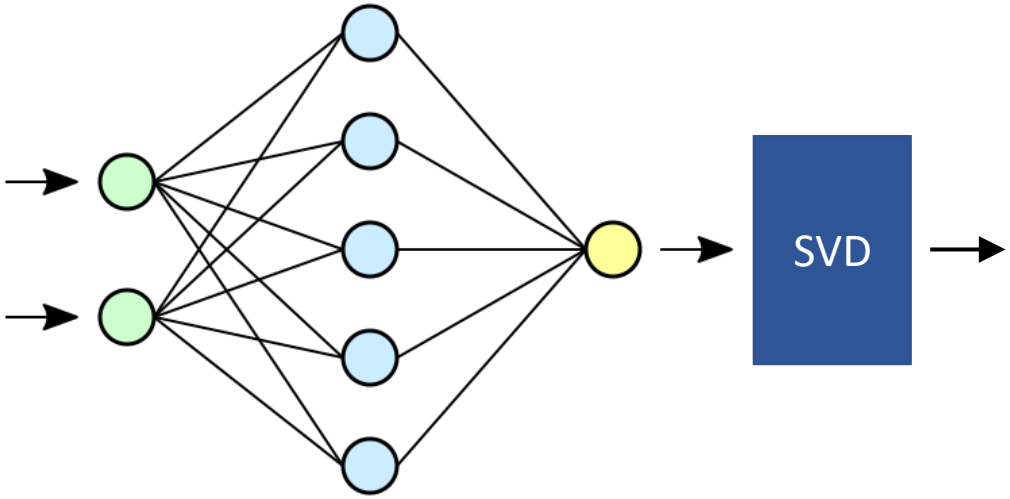
SVD求导

SVD是可微运算

$$A = U\Sigma V^T = RS,\ \frac{\partial \Sigma}{\partial A} = ?,\ \frac{\partial R}{\partial A} = ?$$

梯度反向传播，已知δA，$\delta \Sigma = ?$，$\delta R = ?$

弹性体模拟



神经网络优化

SVD求导 (奇异值)

$$\mathrm{F} = \mathrm{U}\Sigma\mathrm{V}^{\mathrm{T}}$$

$$\delta\mathrm{F} = \delta\mathrm{U}\Sigma\mathrm{V}^{\mathrm{T}} + \mathrm{U}\delta\Sigma\mathrm{V}^{\mathrm{T}} + \mathrm{U}\Sigma\delta\mathrm{V}^{\mathrm{T}}$$

$$\delta\Sigma = \mathrm{U}^{\mathrm{T}}\delta\mathrm{F}\mathrm{V} - \mathrm{U}^{\mathrm{T}}\delta\mathrm{U}\Sigma - \Sigma\left(\mathrm{V}^{\mathrm{T}}\delta\mathrm{V}\right)^{\mathrm{T}}$$

$\mathrm{U}^{\mathrm{T}}\delta\mathrm{U}$和$\mathrm{V}^{\mathrm{T}}\delta\mathrm{V}$是反对称矩阵:

$$\mathrm{U}^{\mathrm{T}}\mathrm{U} = \mathrm{I}, \delta\mathrm{U}^{\mathrm{T}}\mathrm{U} + \mathrm{U}^{\mathrm{T}}\delta\mathrm{U} = 0, \mathrm{U}^{\mathrm{T}}\delta\mathrm{U} = -\left(\mathrm{U}^{\mathrm{T}}\delta\mathrm{U}\right)^{\mathrm{T}}$$

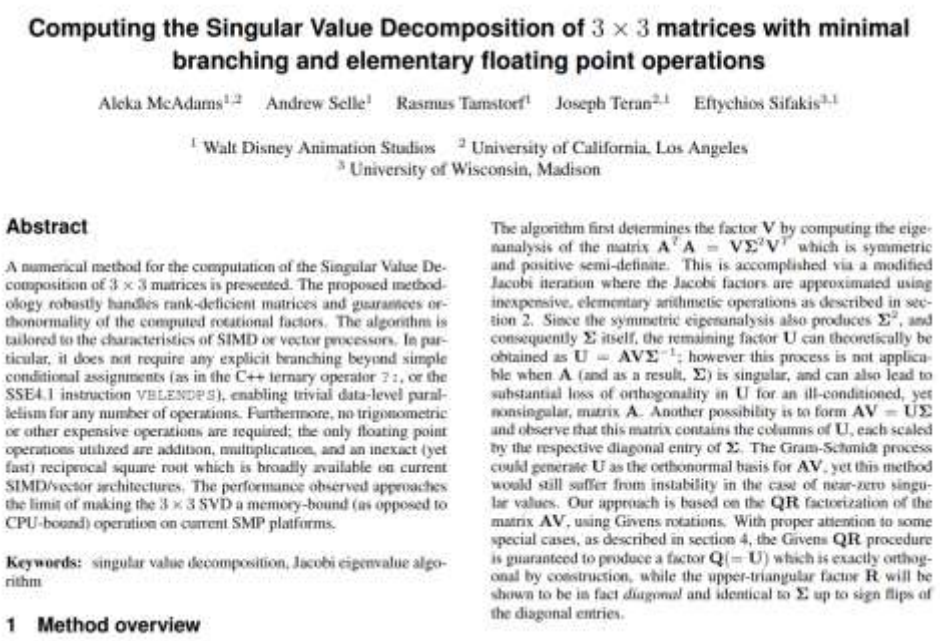
反对称矩阵的对角线为0:

$$\mathrm{Diag}\left(\mathrm{U}^{\mathrm{T}}\delta\mathrm{U}\Sigma\right) = 0, \mathrm{Diag}\left(\Sigma\left(\mathrm{V}^{\mathrm{T}}\delta\mathrm{V}\right)^{\mathrm{T}}\right) = 0$$

$$\boxed{\delta\Sigma = \mathrm{Diag}\left(\mathrm{U}^{\mathrm{T}}\delta\mathrm{F}\mathrm{V}\right)}$$

SVD效率

一般来说，SVD是比较耗时的算法，好在现有的实现比较齐全



Computing the Singular Value Decomposition of 3×3 matrices with minimal branching and elementary floating point operations

Aleka McAdams[1,2] Andrew Selle[1] Rasmus Tamstorf[1] Joseph Teran[2,1] Eftychios Sifakis[3,1]

[1] Walt Disney Animation Studios [2] University of California, Los Angeles
[3] University of Wisconsin, Madison

Abstract

A numerical method for the computation of the Singular Value Decomposition of 3×3 matrices is presented. The proposed methodology robustly handles rank-deficient matrices and guarantees orthonormality of the computed rotational factors. The algorithm is tailored to the characteristics of SIMD or vector processors. In particular, it does not require any explicit branching beyond simple conditional assignments (as in the C++ ternary operator `?:`, or the SSE4.1 instruction `VBLENDPS`), enabling trivial data-level parallelism for any number of operations. Furthermore, no trigonometric or other expensive operations are required; the only floating point operations utilized are addition, multiplication, and an inexact (yet fast) reciprocal square root which is broadly available on current SIMD/vector architectures. The performance observed approaches the limit of making the 3×3 SVD a memory-bound (as opposed to CPU-bound) operation on current SMP platforms.

Keywords: singular value decomposition, Jacobi eigenvalue algorithm

1 Method overview

The algorithm first determines the factor $\mathbf{V}$ by computing the eigenanalysis of the matrix $\mathbf{A}^T\mathbf{A} = \mathbf{V}\mathbf{\Sigma}^2\mathbf{V}^T$ which is symmetric and positive semi-definite. This is accomplished via a modified Jacobi iteration where the Jacobi factors are approximated using inexpensive, elementary arithmetic operations as described in section 2. Since the symmetric eigenanalysis also produces $\mathbf{\Sigma}^2$, and consequently $\mathbf{\Sigma}$ itself, the remaining factor $\mathbf{U}$ can theoretically be obtained as $\mathbf{U} = \mathbf{A}\mathbf{V}\mathbf{\Sigma}^{-1}$; however this process is not applicable when $\mathbf{A}$ (and as a result, $\mathbf{\Sigma}$) is singular, and can also lead to substantial loss of orthogonality in $\mathbf{U}$ for an ill-conditioned, yet nonsingular, matrix $\mathbf{A}$. Another possibility is to form $\mathbf{AV} = \mathbf{U}\mathbf{\Sigma}$ and observe that this matrix contains the columns of $\mathbf{U}$, each scaled by the respective diagonal entry of $\mathbf{\Sigma}$. The Gram-Schmidt process could generate $\mathbf{U}$ as the orthonormal basis for $\mathbf{AV}$, yet this method would still suffer from instability in the case of near-zero singular values. Our approach is based on the $\mathbf{QR}$ factorization of the matrix $\mathbf{AV}$, using Givens rotations. With proper attention to some special cases, as described in section 4, the Givens $\mathbf{QR}$ procedure is guaranteed to produce a factor $\mathbf{Q}(=\mathbf{U})$ which is exactly orthogonal by construction, while the upper-triangular factor $\mathbf{R}$ will be shown to be in fact *diagonal* and identical to $\mathbf{\Sigma}$ up to sign flips of the diagonal entries.

scipy.sparse.linalg.svds

```
scipy.sparse.linalg.svds(A, k=6, ncv=None, tol=0, which='LM', v0=None, maxiter=None,
return_singular_vectors=True, solver='arpack', random_state=None, options=None)
```
[source]

Partial singular value decomposition of a sparse matrix.

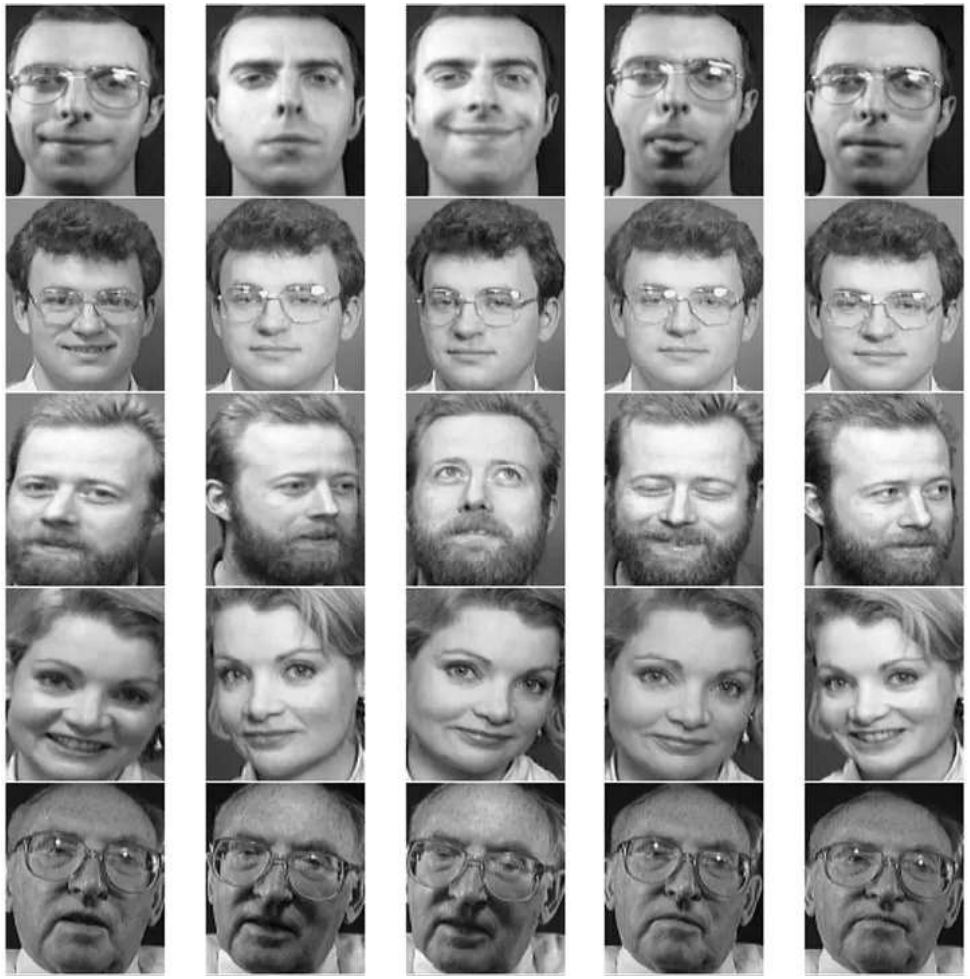
Compute the largest or smallest *k* singular values and corresponding singular vectors of a sparse matrix *A*. The order in which the singular values are returned is not guaranteed.

In the descriptions below, let `M, N = A.shape`.

Notes

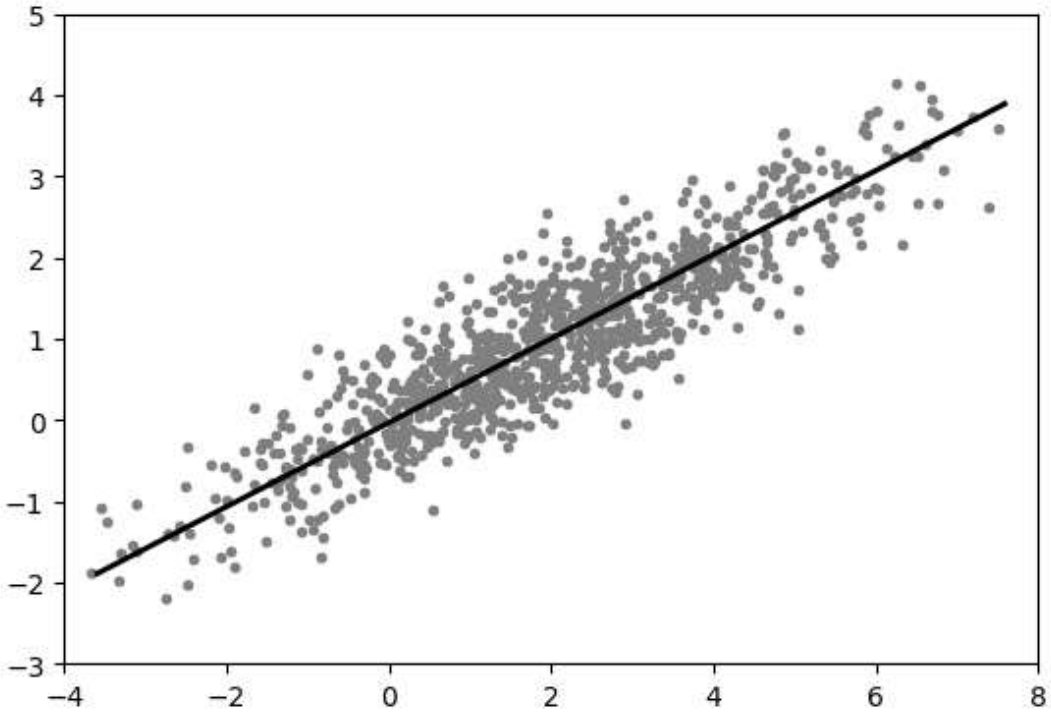
This is a naive implementation using ARPACK or LOBPCG as an eigensolver on the matrix `A.conj().T @ A` or `A @ A.conj().T`, depending on which one is smaller size, followed by the Rayleigh-Ritz method as postprocessing; see Using the normal matrix, in Rayleigh-Ritz method, (2022, Nov. 19), Wikipedia, https://w.wiki/4zms.

主成分分析 (**P**rincipal **C**omponent **A**nalysis)

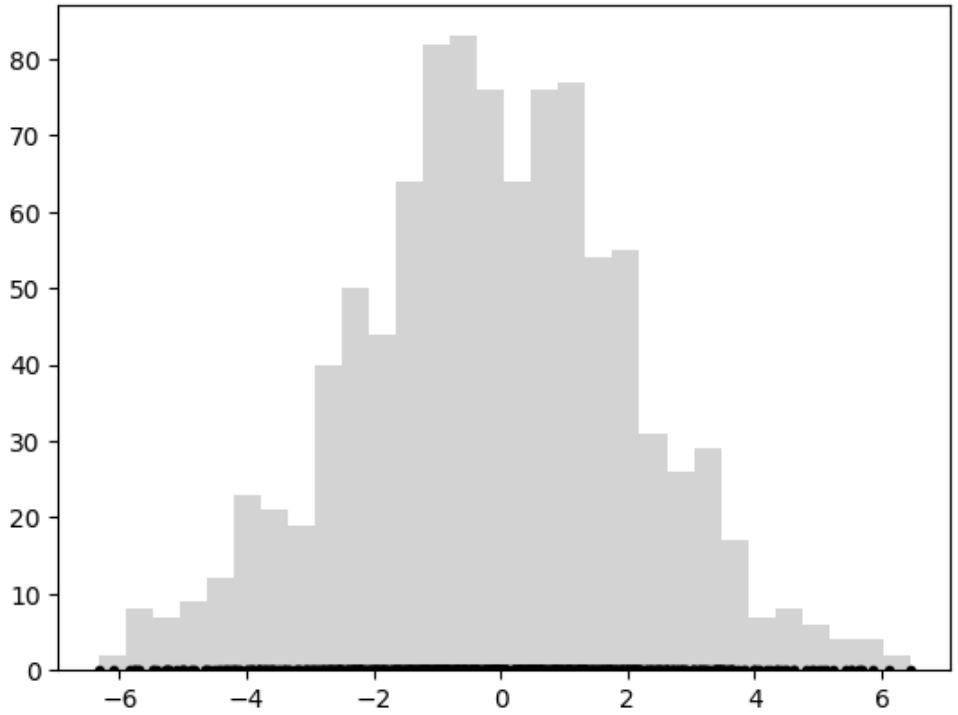


Eigenfaces

降维表达



$\mathbb{R}^2$



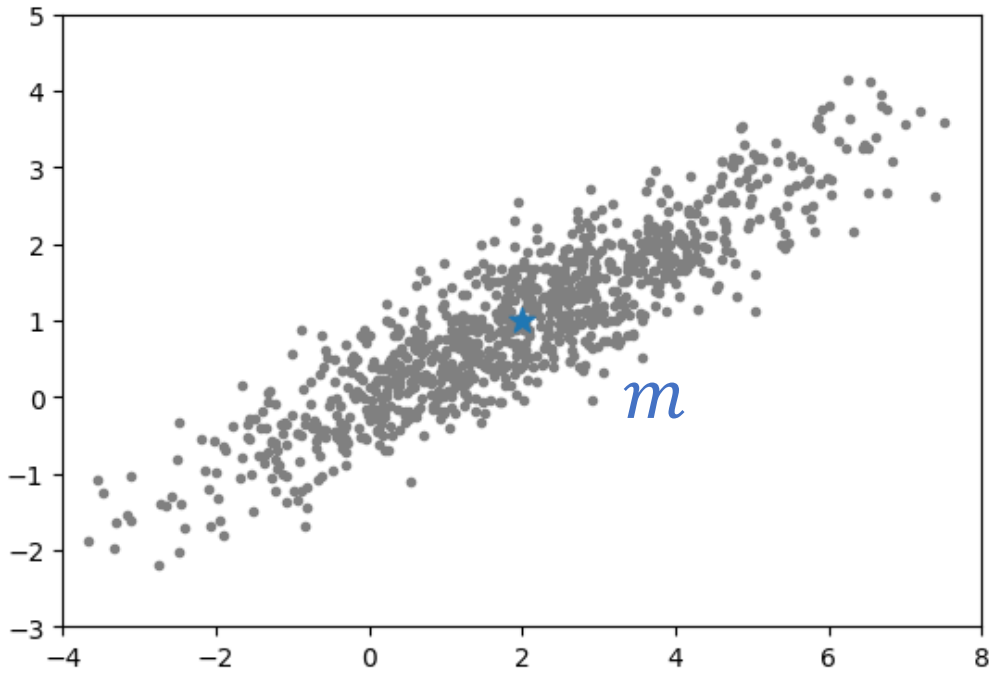
$\mathbb{R}^1$

零阶近似：只有一个点

数据集：$x_0, \cdots, x_{k-1} \in \mathbb{R}^n$

$$m = \arg\min_x \sum_{i=0}^{k-1} \|x_i - x\|^2$$

$$m = \frac{1}{n} \sum_{i=0}^{k-1} x_i$$



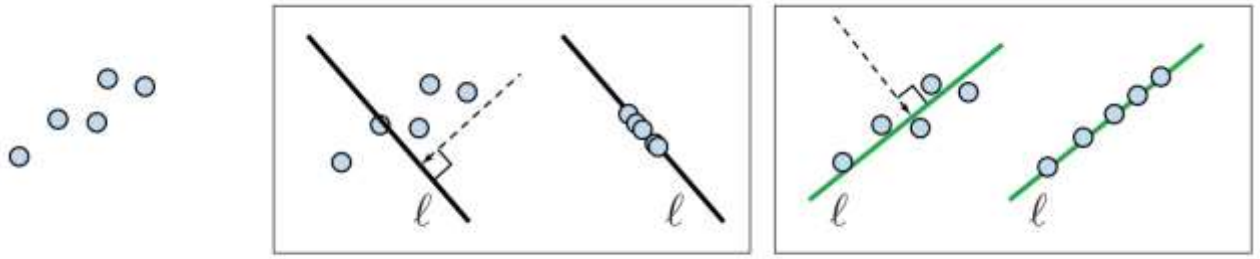
一阶近似：主方向

- 任意方向 $v \in \mathbb{R}^n, \|v\| = 1$ 定义了一条直线: $x = m + tv$
- 将数据点投影到直线上: $x_i' = (x_i - m) \cdot v$
- 数据点在垂直方向的距离: $d_i^2 = \|x_i - m\|^2 - \|x_i'\|^2$

$$\underset{v}{\operatorname{argmin}} \sum_i d_i^2 = \underset{v}{\operatorname{argmax}} \sum_i \|x_i'\|^2$$

$$= \underset{v}{\operatorname{argmax}}\, v^{\mathrm{T}} \left(\sum_i (x_i - m)(x_i - m)^{\mathrm{T}} \right) v$$

方差矩阵 S



一阶近似：主方向

$$p = \underset{v}{\operatorname{argmax}}\, v^{\mathrm{T}} \mathrm{S} v, \qquad \mathrm{S} = \sum_i (x_i - m)(x_i - m)^{\mathrm{T}}$$

$\mathrm{S} \in \mathbb{R}^{n \times n}$ 是半正定矩阵，可以实数对角化，并且特征值非负

$$\mathrm{S} = \mathrm{Q} \Lambda \mathrm{Q}^{\mathrm{T}} = (q_0 \quad \cdots \quad q_{n-1}) \begin{pmatrix} \lambda_0 & & \\ & \ddots & \\ & & \lambda_{n-1} \end{pmatrix} \begin{pmatrix} q_0^{\mathrm{T}} \\ \vdots \\ q_{n-1}^{\mathrm{T}} \end{pmatrix}, \mathrm{Q}^{\mathrm{T}} \mathrm{Q} = \mathrm{I}$$

$$\lambda_0 \geq \lambda_1 \geq \cdots \geq 0$$

$$v^{\mathrm{T}} \mathrm{S} v = v^{\mathrm{T}} \mathrm{Q} \Lambda \mathrm{Q}^{\mathrm{T}} v = \left(\mathrm{Q}^{\mathrm{T}} v\right)^{\mathrm{T}} \Lambda (\mathrm{Q}^{\mathrm{T}} v)$$

当$v = q_0$时，$\mathrm{Q}^T v = (1, 0, \cdots, 0)^T$，$v^{\mathrm{T}} \mathrm{S} v$ 取到最大值λ_0

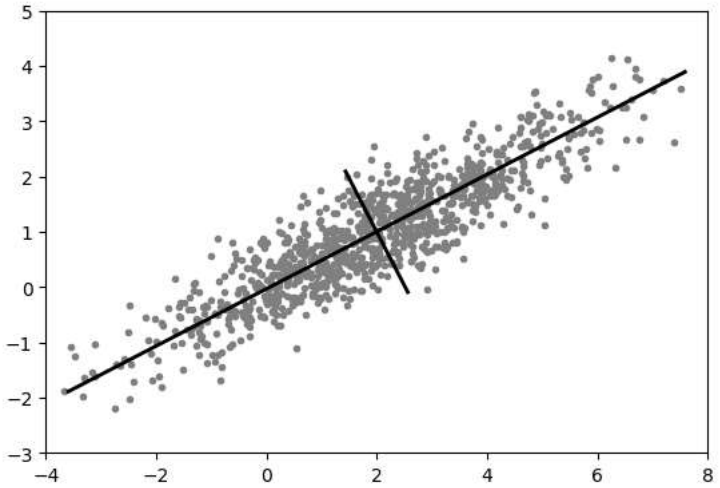
继续下去

- 投影完之后剩下的分量：$x_i'' = (x_i - m) - \big((x_i - m) \cdot p\big)p$
- 继续寻找主分量：

$$\underset{v}{\operatorname{argmax}}\, v^{\mathrm{T}} \left(\sum_i x_i'' x_i''^{\mathrm{T}} \right) v = \underset{v}{\operatorname{argmax}}\, v^{\mathrm{T}} \left(\mathrm{S} - \lambda_0 p p^{\mathrm{T}} \right) v$$

$$\mathrm{S} = \mathrm{Q}\Lambda\mathrm{Q}^{\mathrm{T}} = \lambda_0 q_0 q_0^{\mathrm{T}} + \lambda_1 q_1 q_1^{\mathrm{T}} + \cdots$$

可以尝试自己推导出来



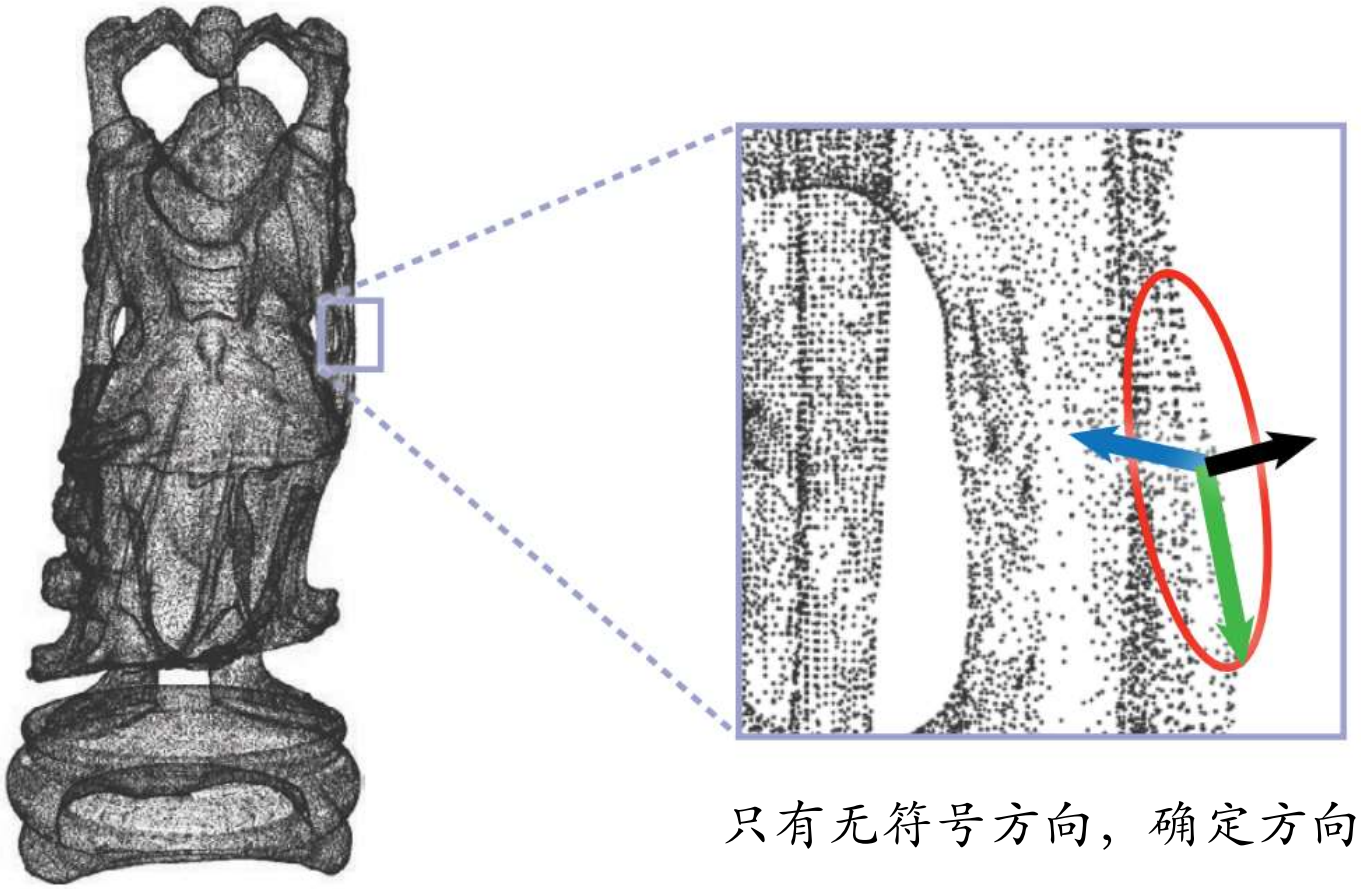
PCA

由 $x_0, \cdots x_{k-1} \in \mathbb{R}^n$ 这 k 个 n 维数据点组成的数据集，前 m 个主成分为方差矩阵 S 的前 m 个特征向量 $q_0, \cdots, q_{m-1}$，其中

$$S = \sum_i (x_i - m)(x_i - m)^T, m = \frac{1}{n} \sum_{i=0}^{k-1} x_i$$

这些向量相互正交，张成了 m 维的子空间。将数据投影到这个子空间，就构成了数据集的低维表达

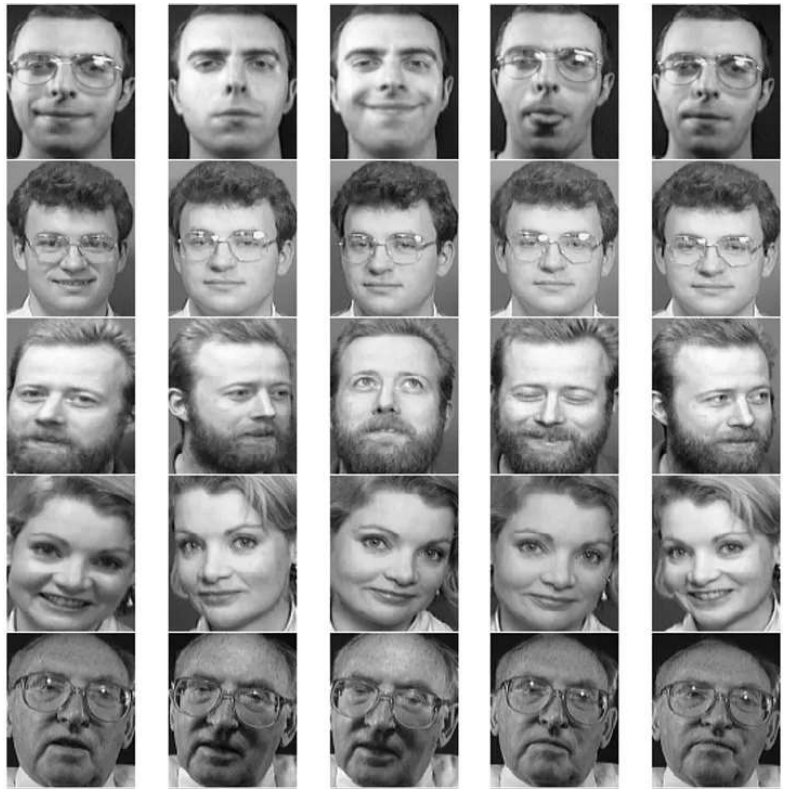
应用：点云法向



只有无符号方向，确定方向需要别的算法

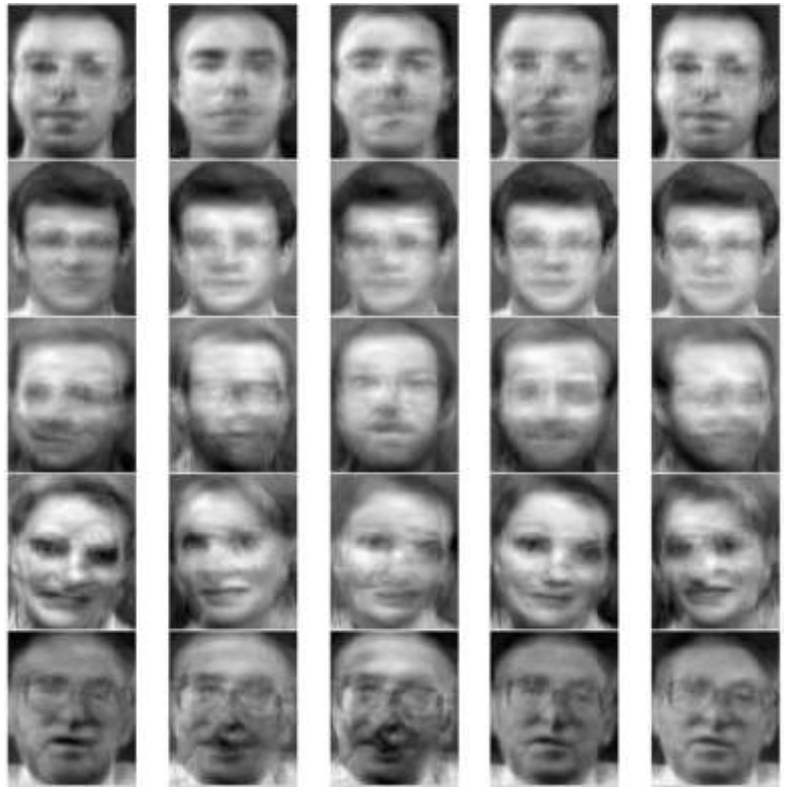
From “A Sampler of Useful Computational Tools for Applied Geometry, Computer Graphics, and Image Processing”

应用：Eigenfaces



400张112x92图片

40张Eigenfaces (PCA主分量)

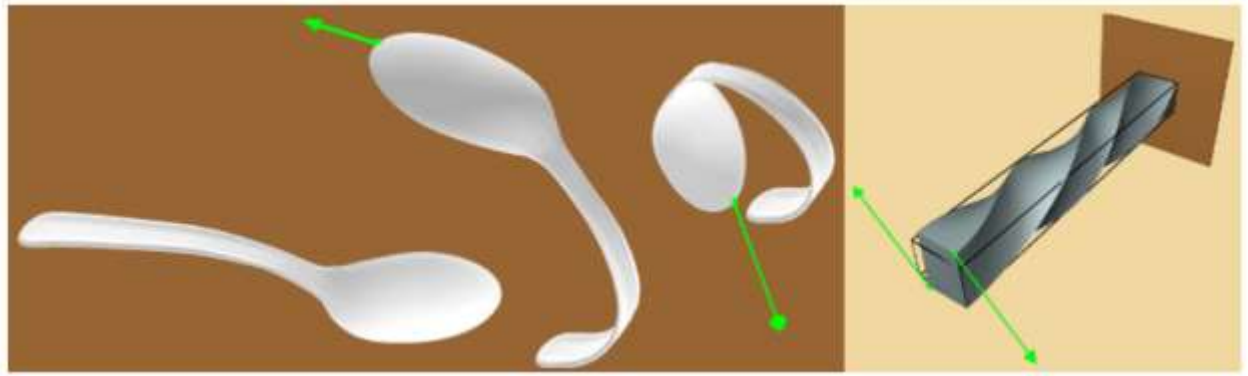


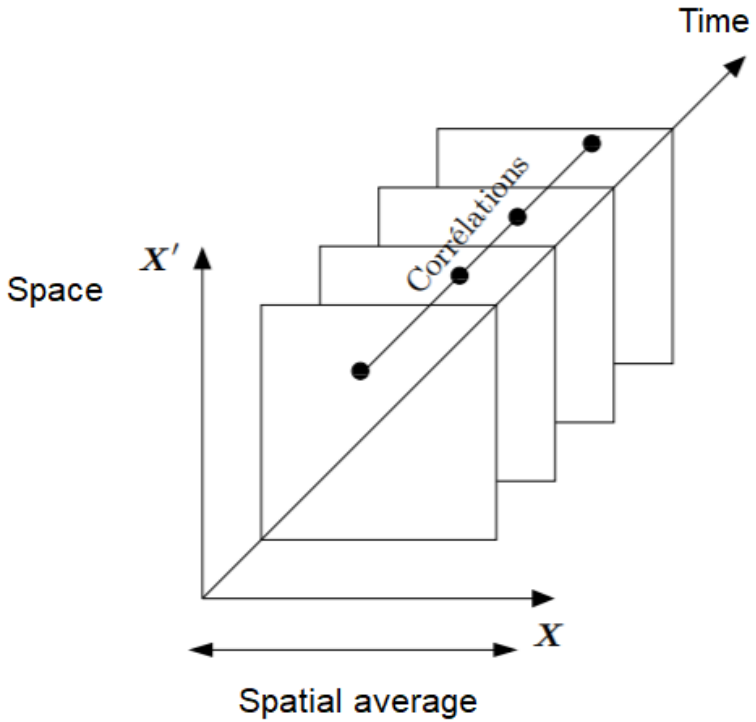
400个40维权重

Maryna Longnickel

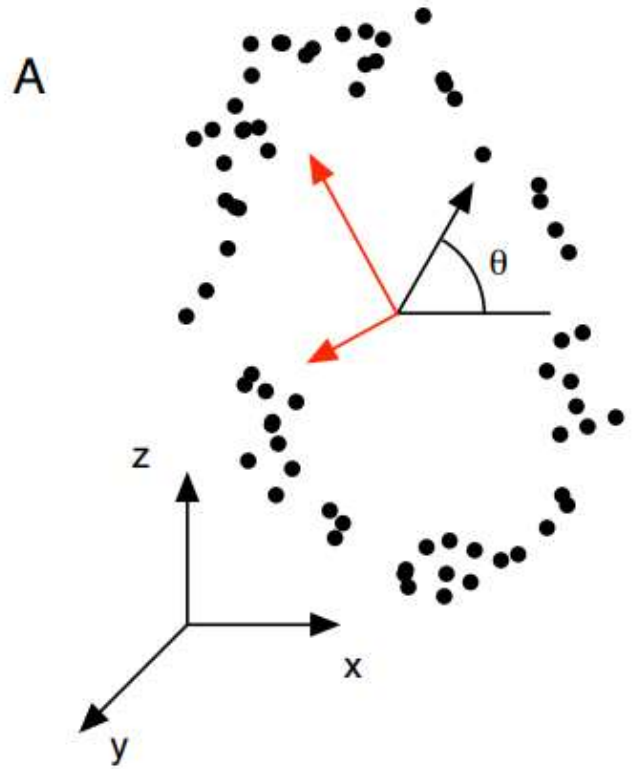
应用：模型降阶 (Model Reduction)

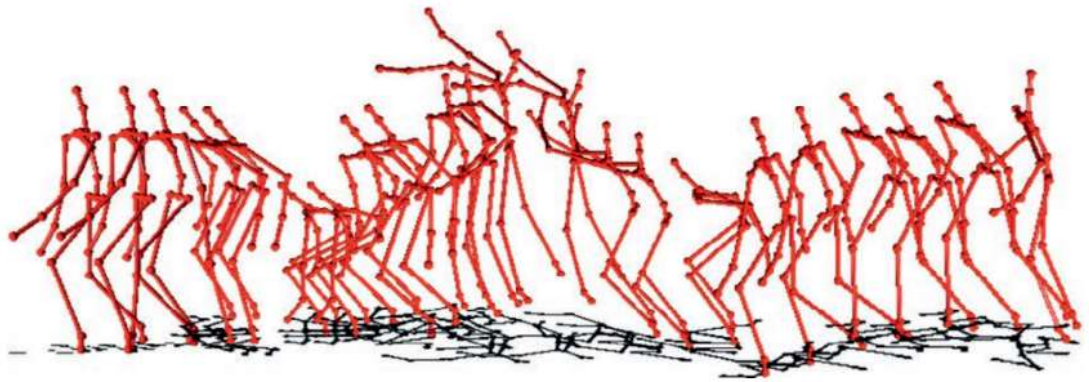
本征正交分解 (Proper orthogonal decomposition)



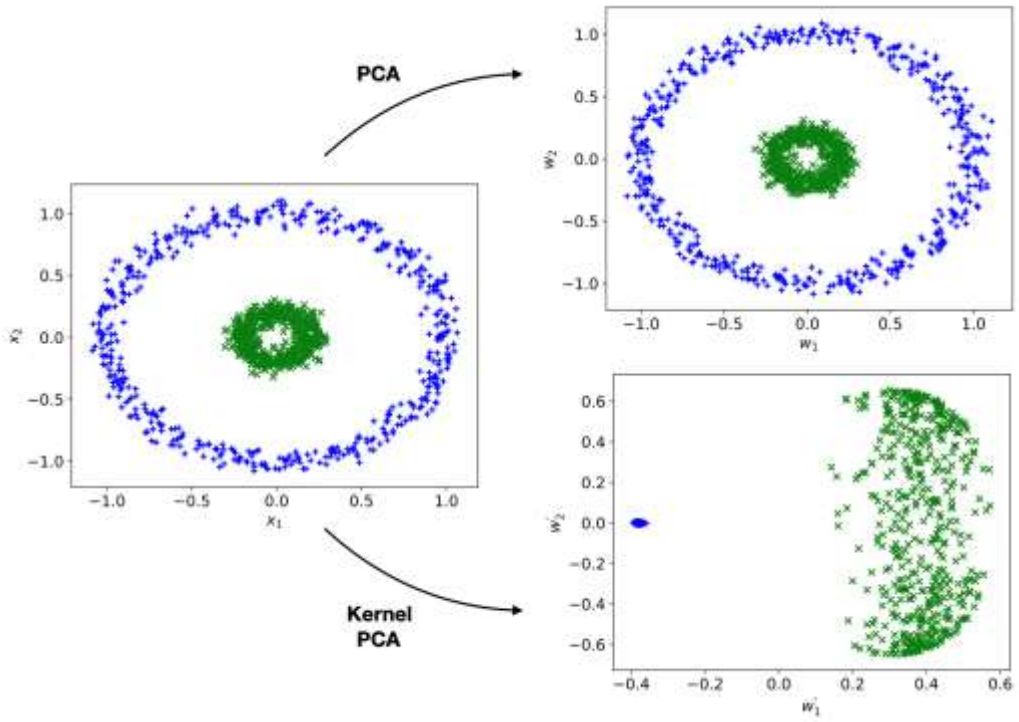


PCA的局限性：非线性

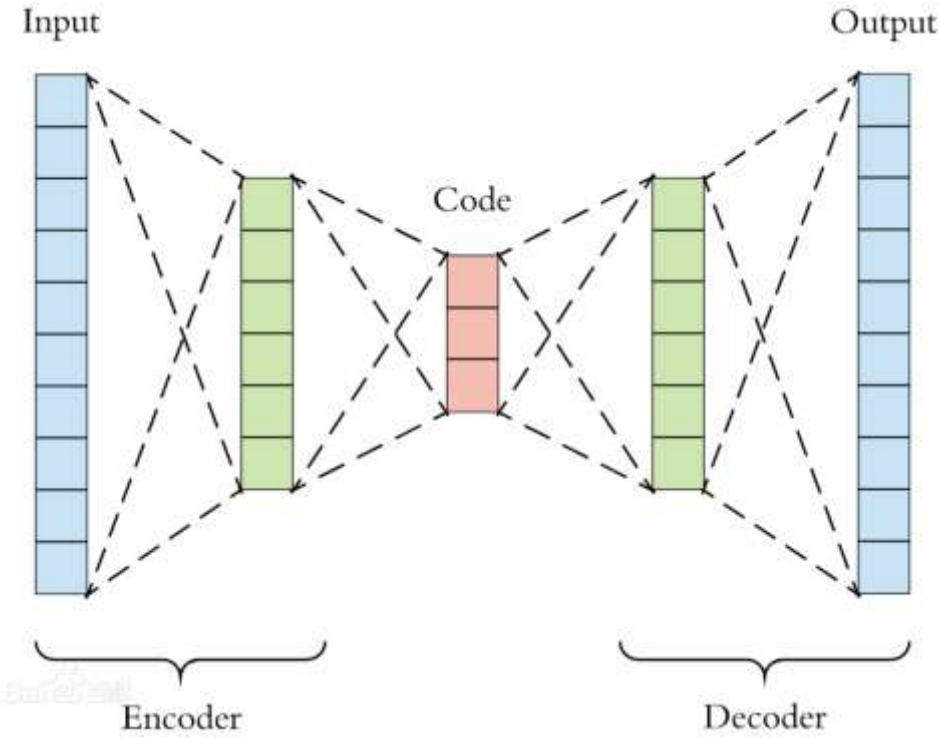




解决方法



Kernel PCA



Neural Network

参考资料

- Daniel Cohen-Or: https://www.cs.tau.ac.il/~dcor/Graphics/graphics2021.html
- Gilbert Strang: https://math.mit.edu/~gs/LectureNotes/
- Eftychios Sifakis: https://viterbi-web.usc.edu/~jbarbic/femdefo/
- Olga Sorkine-Hornung: https://igl.ethz.ch/projects/ARAP/svd_rot.pdf

北京大学
PEKING UNIVERSITY

谢谢