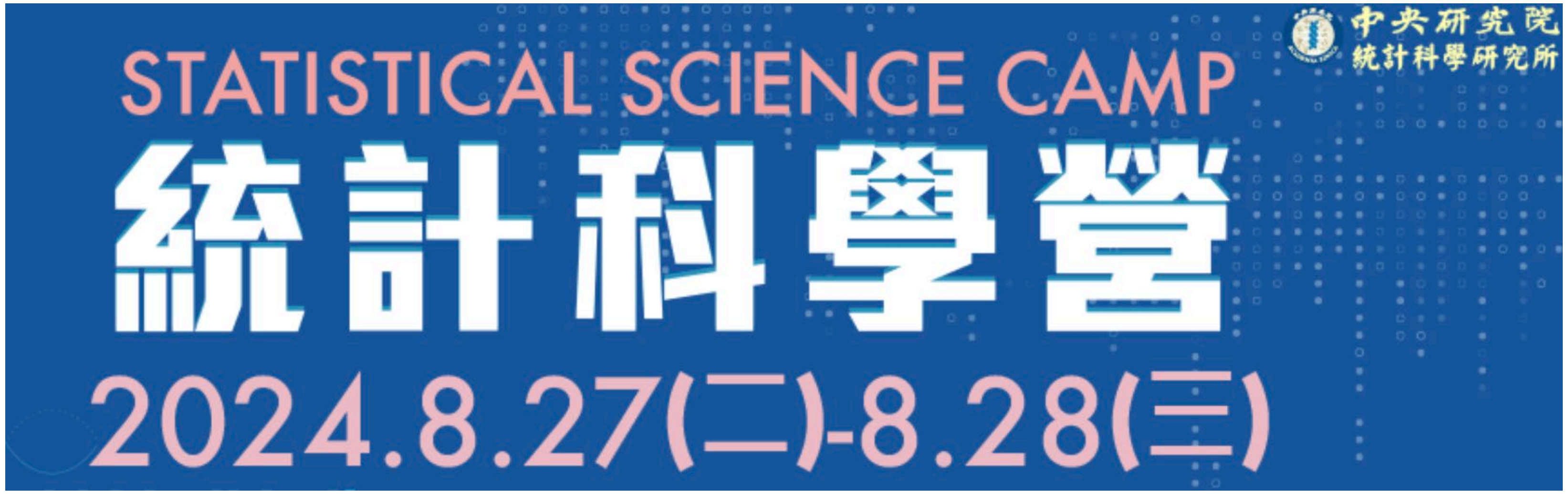
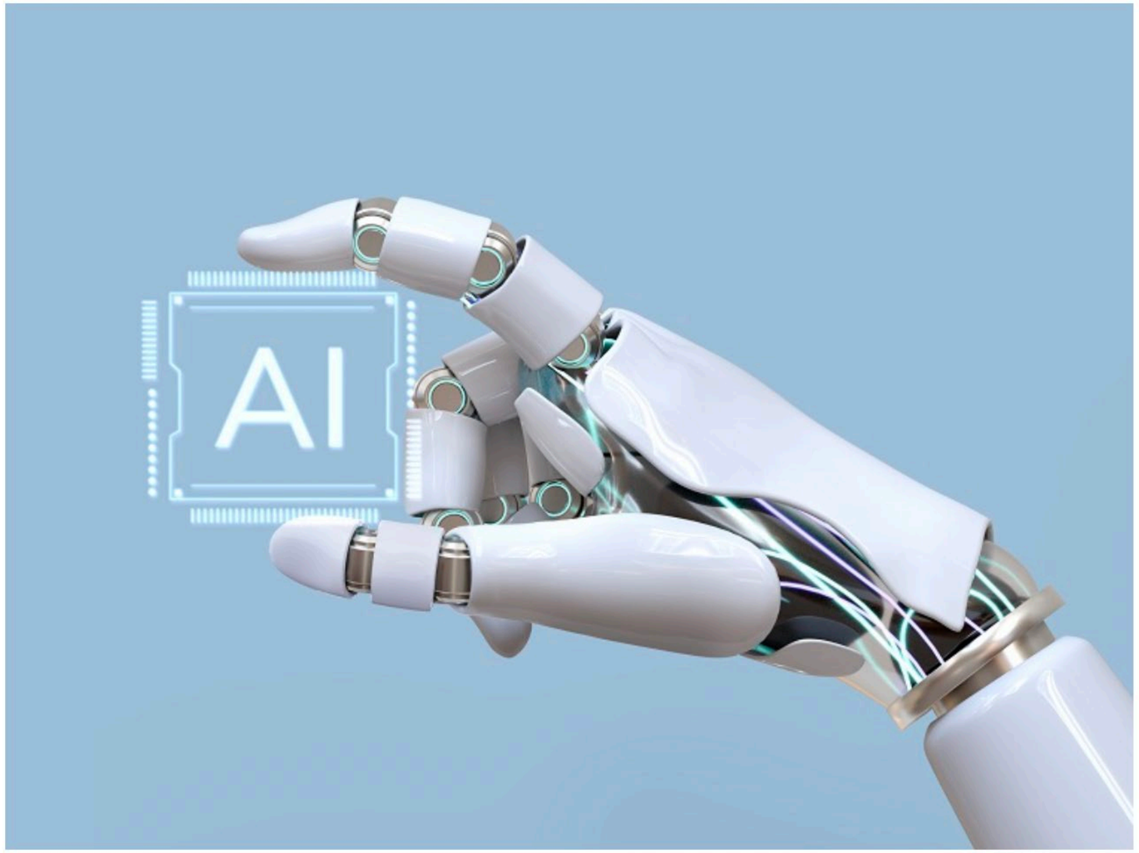
高維度空間上的最佳化問題---走在流行



講者: 王紹宣 ｜ 中央大學統計所

全球趨勢

AI讓碳排飆升三成？微軟：供應商2030全面使用零碳能源



微軟最新永續報告指出，2023碳排量激增30%，主要原因是AI發展導致用電量大增。
freepik by rawpixel.com

拓展AI版圖，碳排激增三成

隨著AI PC與生成式AI浪潮席捲全球，微軟近年來積極拓展AI版圖，業務不斷擴大。微軟目前是全球第二大雲端服務供應商，2024年市值突破三兆美元，在全球擁有61個 Azure區域，微軟也宣布在台灣設立資料中心，以加速推動台灣的數位轉型。

然而，**隨著AI技術提升與應用變廣，雲端運算能力更高，同時連帶著耗能增加。**根據報告指出，由於2023年不斷投資與打造新科技基礎設施，造成整體碳排量增加29.1%。其中，超過 96%碳放量來自範疇三，包括供應鏈、硬體和設備的生命週期，以及其他間接排放源。

範疇三碳排減半，供應商須提供零碳電力

為實現2050年碳排放量減半至淨零及100%使用再生能源的目標，微軟制定了階段性計畫，盼於2030年達成範疇三碳排放減半以上，並全面採用無碳能源。為此，微軟提出80多項減排措施，涵蓋五大方向為：

1. **強化數據監測**：藉由數位科技精準掌握碳排放數據，為採取有效行動奠定基礎。
2. **提高效率**：採用創新技術提升資料中心營運效率，降低能源消耗。
3. **合作研發綠色科技**：透過投資和人工智慧能力，加速研發突破性綠色科技，例如更環保的鋼材、混凝土和燃料。
4. **建立綠色市場**：利用採購力，加速市場對綠色科技的需求，推動產業轉型。
5. **倡導氣候政策**：積極推動有利於氣候變遷改善的公共政策制定，促成全球共同減碳行動。

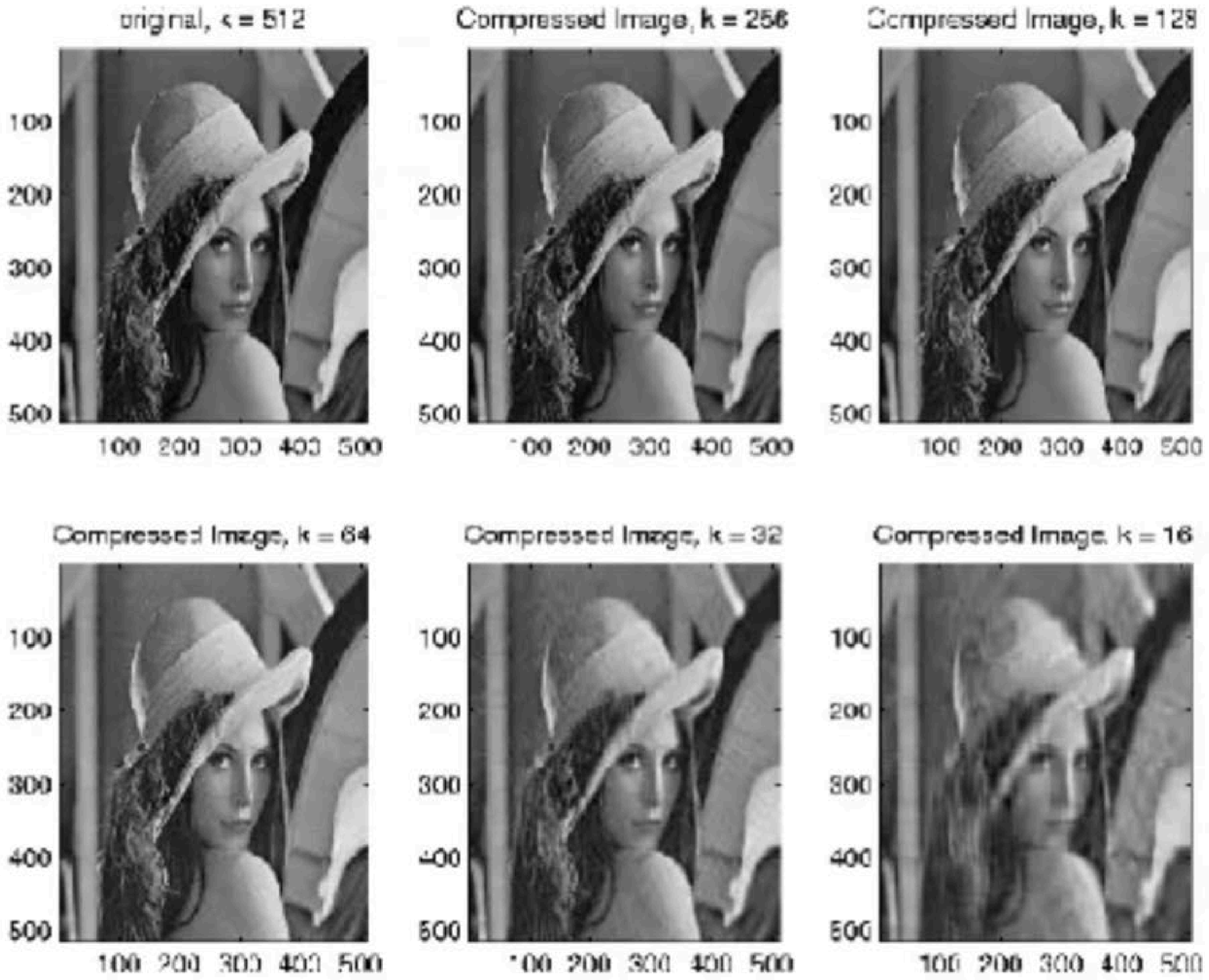
提高效率

1. 統計方法-
數據降維

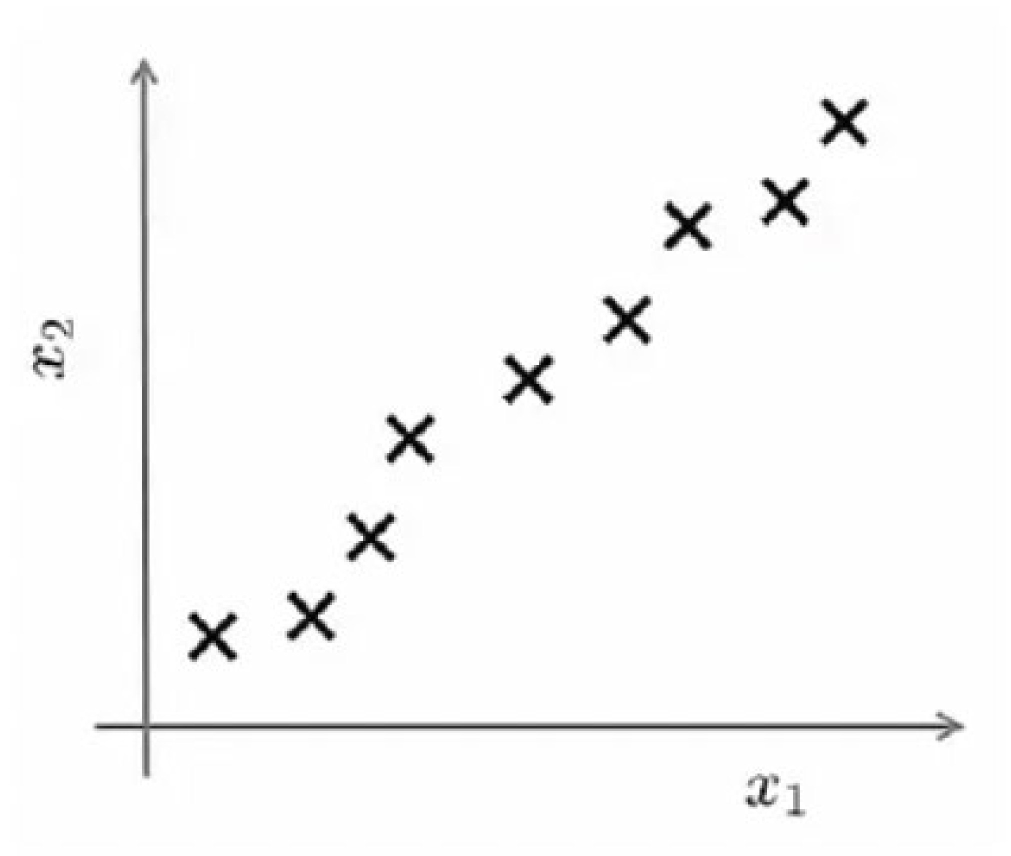
2. 數值方法-
高維度空間隱藏起來的維度

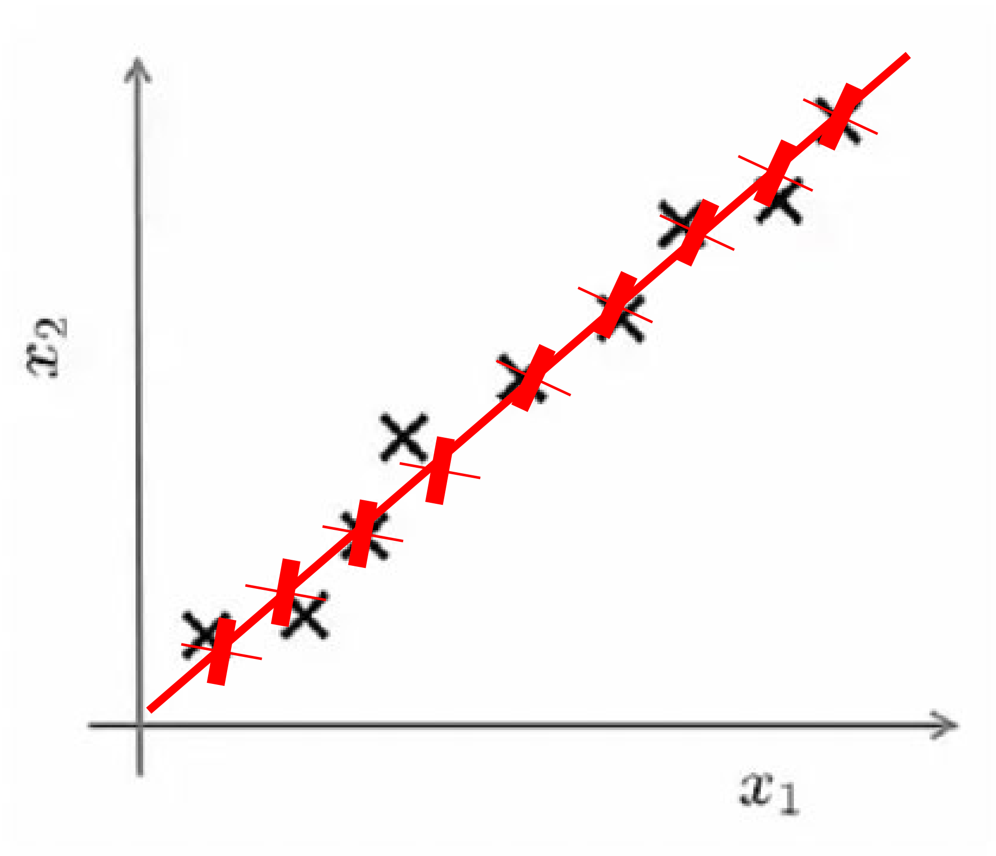
數據降維

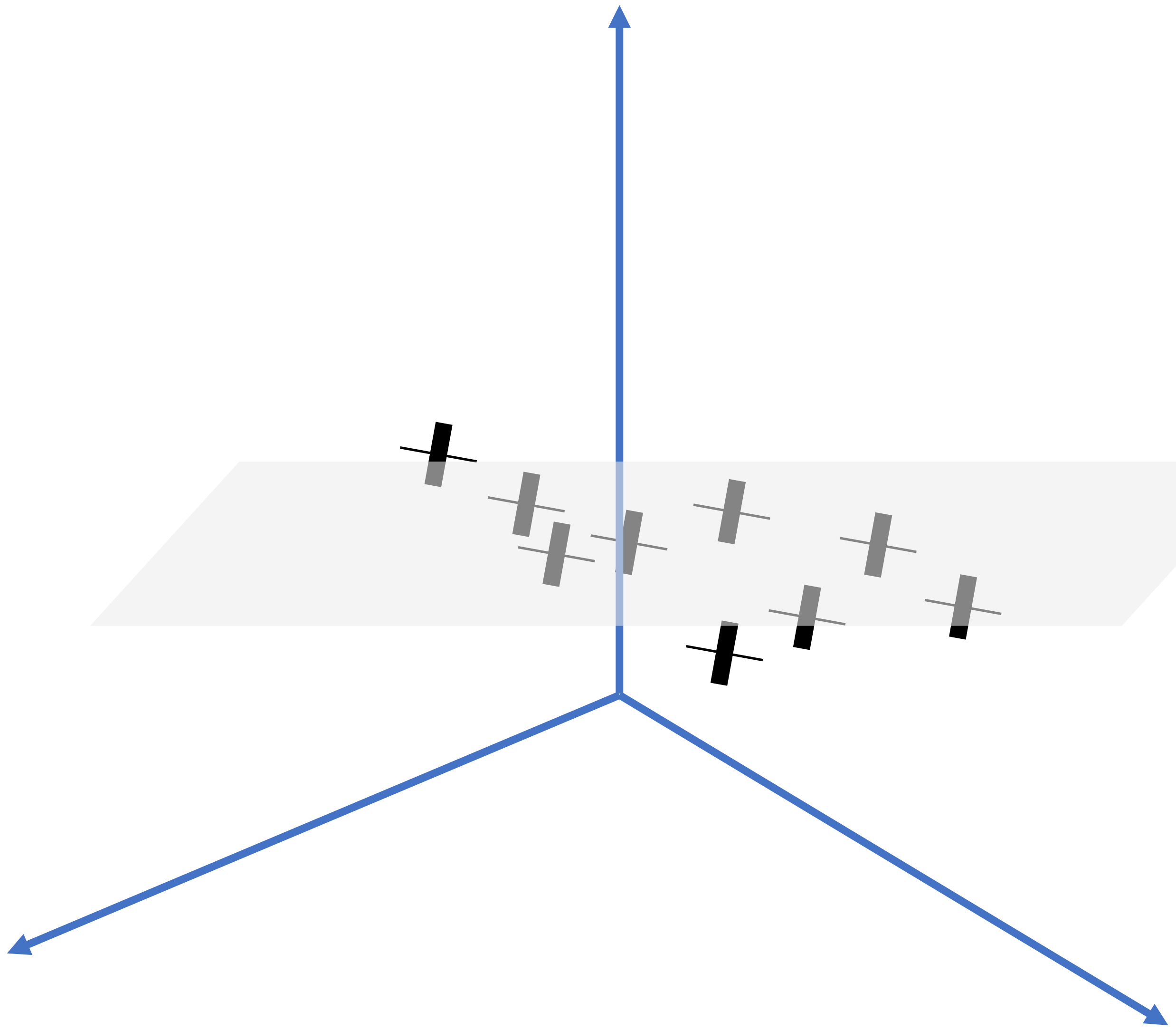
對高維度數據**保留最重要特徵**的方法，同時**去除噪聲和不重要的特徵**，來提升數據處理效率。在一定的信息損失成本下，可以**節省大量的時間和成本**。



圖片來源：fourier.eng.hmc.edu


x_2
x_1


x_2
x_1

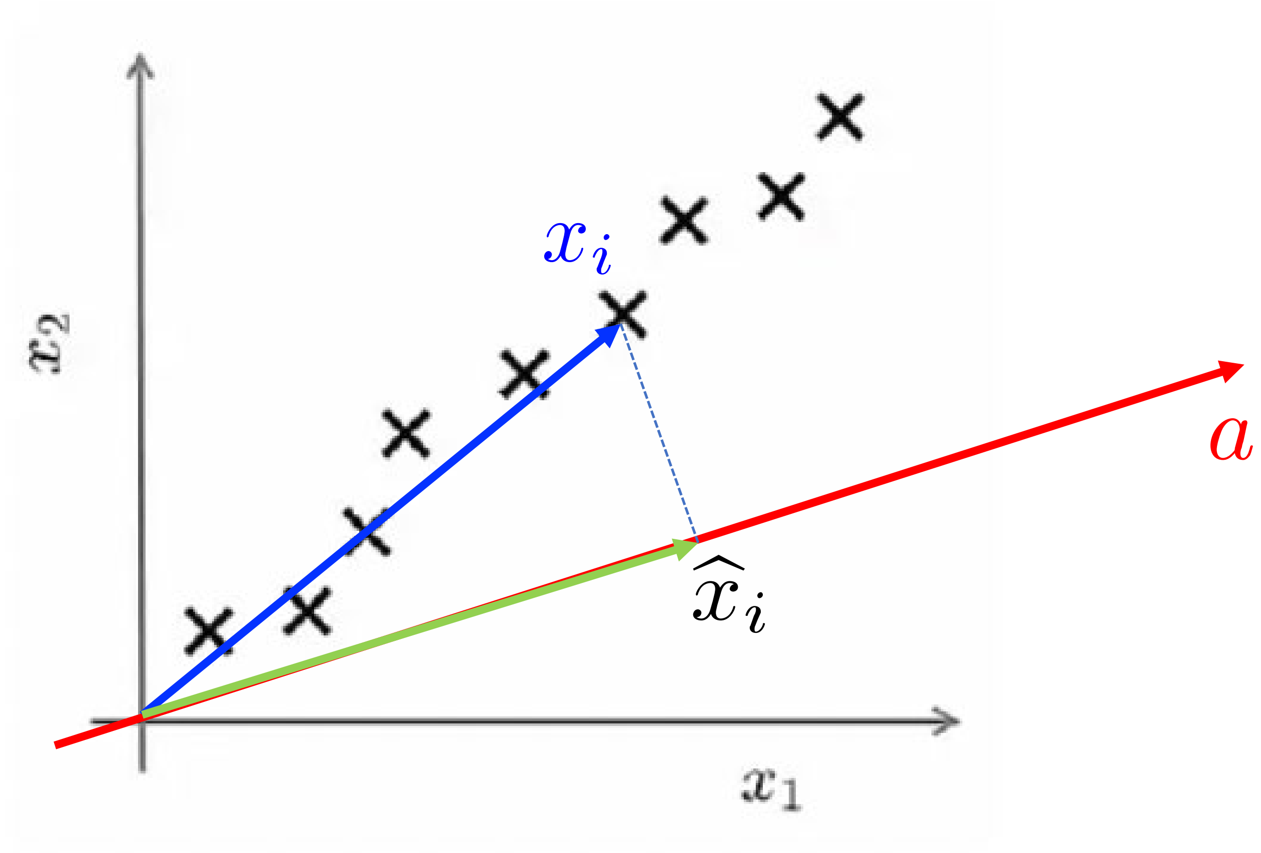


樣本點: $x_1, \ldots, x_n,\ x_i \in \mathbb{R}^p$

樣本平均: $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$

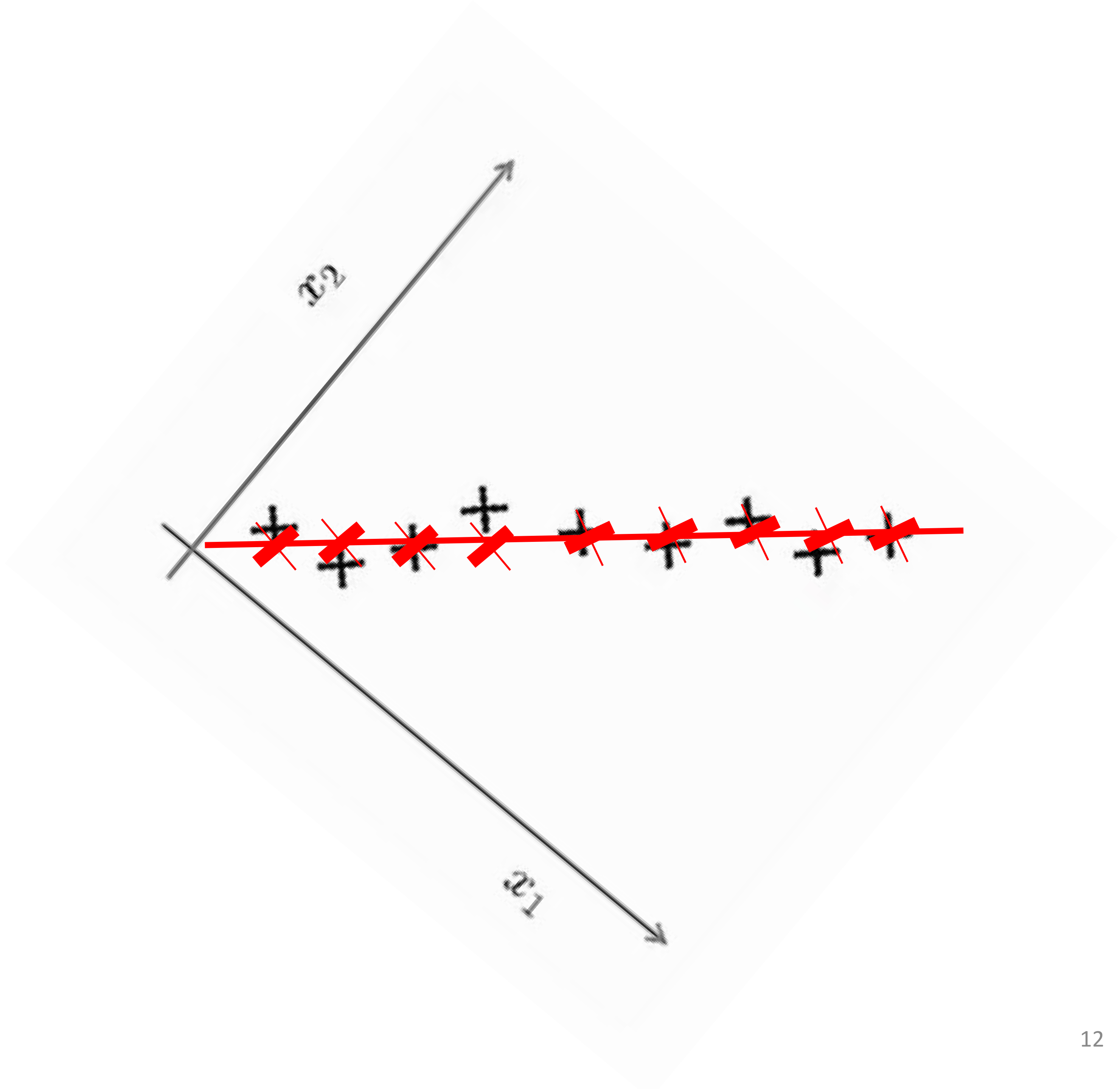
樣本方差(變異矩陣): $S = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})^\top$

$$\widehat{x}_i = \frac{a^\top x_i}{a^\top a} a$$



$$\widehat{x}_i = (a^\top x_i) a = a a^\top x_i \qquad \text{if } \|a\| = 1$$

觀察資料投影後的分散程度

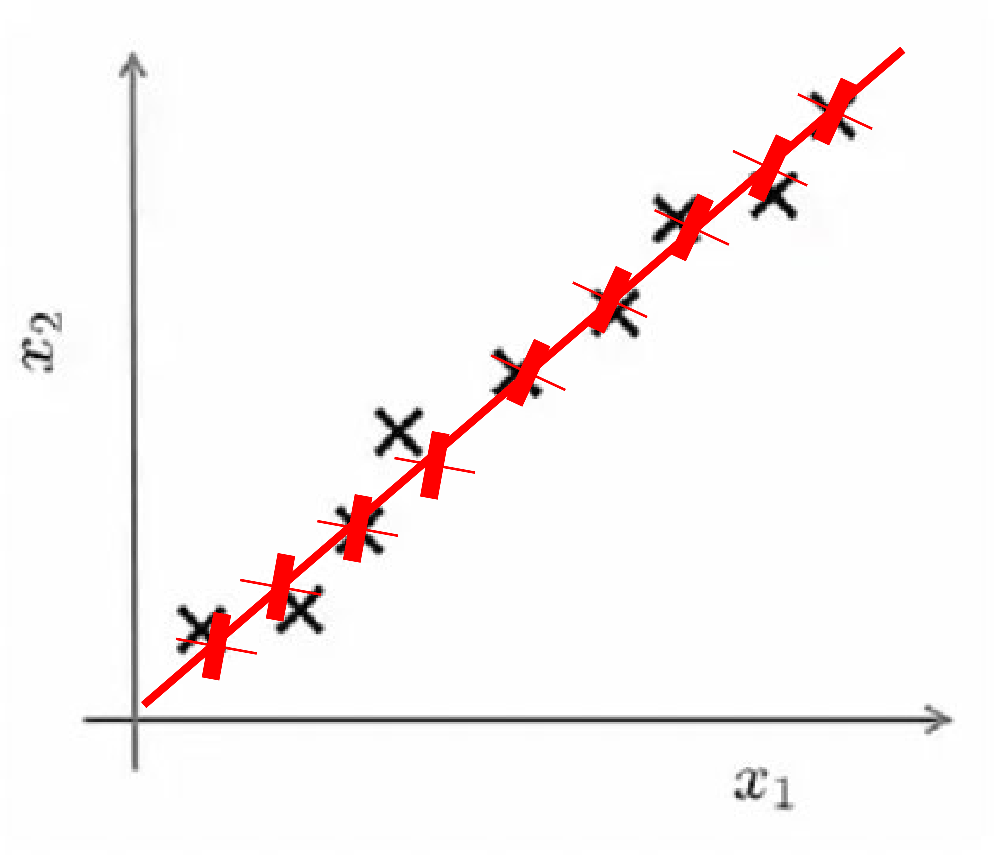


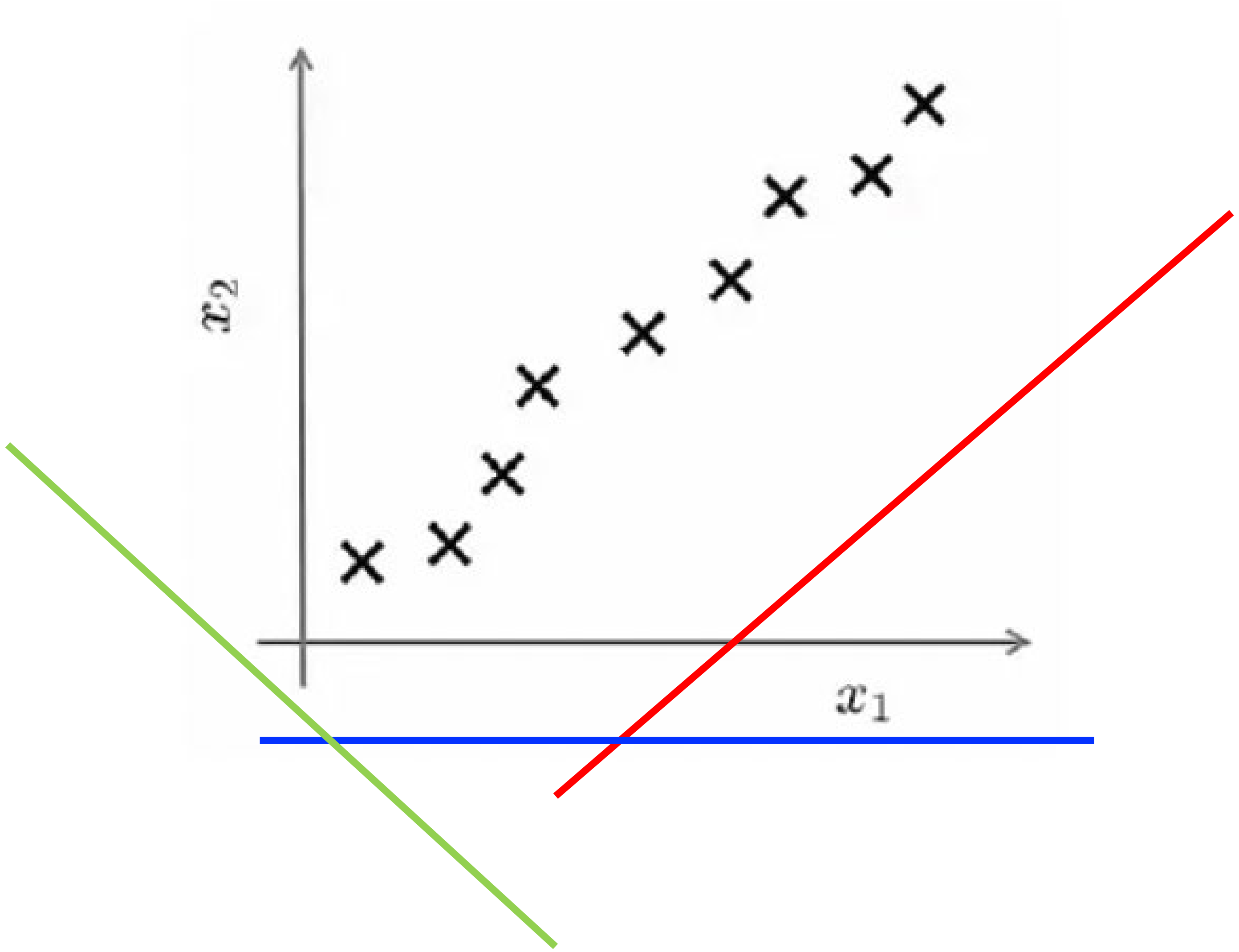
樣本投影點: $z_1, \ldots, z_n,\ z_i = a^\top x_i \in \mathbb{R}$

樣本投影平均: $\bar{z} = \frac{1}{n}\sum_{i=1}^{n} z_i$

樣本投影方差:

$$S_a = \frac{1}{n-1}\sum_{i=1}^{n}(z_i - \bar{z})(z_i - \bar{z})^\top$$




x_2
x_1

Find $\widehat{a} = \mathrm{argmax}_{a \in \mathbb{R}^p} a^\top S a$

$$x_i \Rightarrow \widehat{x}_i = \widehat{a}\widehat{a}^\top x_i$$

方向

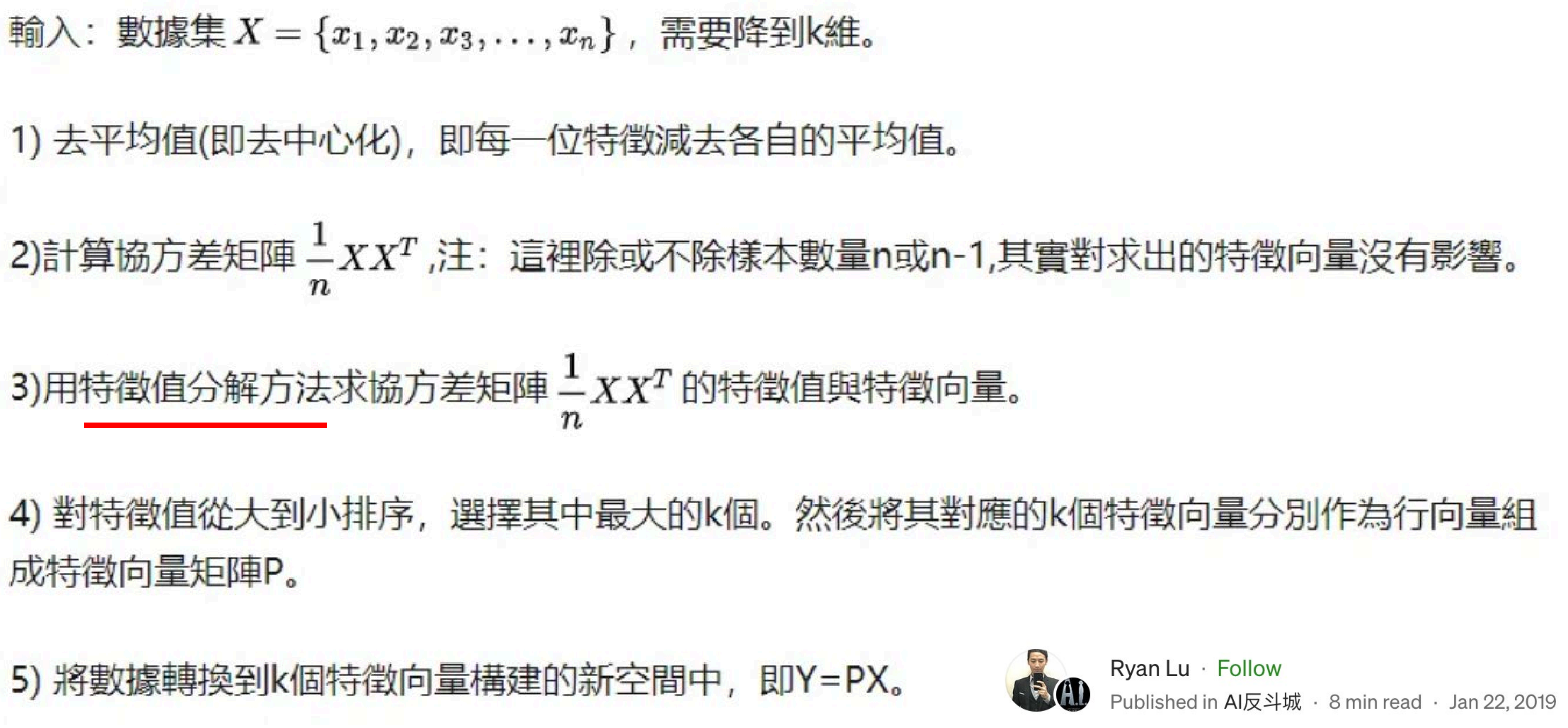
大小

In general

$$\text{Find}\quad \widehat{A} = \operatorname{argmax}_{A \in \mathcal{O}_{p,k}} A^\top S A$$

$$x_i \Longrightarrow \hat{x}_i = \underbrace{\widehat{A}}_{\text{方向}}\,\underbrace{\widehat{A}^\top x_i}_{\text{大小}}$$

Principal Component Analysis (PCA, 主成分分析)



輸入：數據集 $X = \{x_1, x_2, x_3, \ldots, x_n\}$，需要降到k維。

1) 去平均值(即去中心化)，即每一位特徵減去各自的平均值。

2)計算協方差矩陣 $\frac{1}{n}XX^T$,注：這裡除或不除樣本數量n或n-1,其實對求出的特徵向量沒有影響。

3)用特徵值分解方法求協方差矩陣 $\frac{1}{n}XX^T$ 的特徵值與特徵向量。

4) 對特徵值從大到小排序，選擇其中最大的k個。然後將其對應的k個特徵向量分別作為行向量組成特徵向量矩陣P。

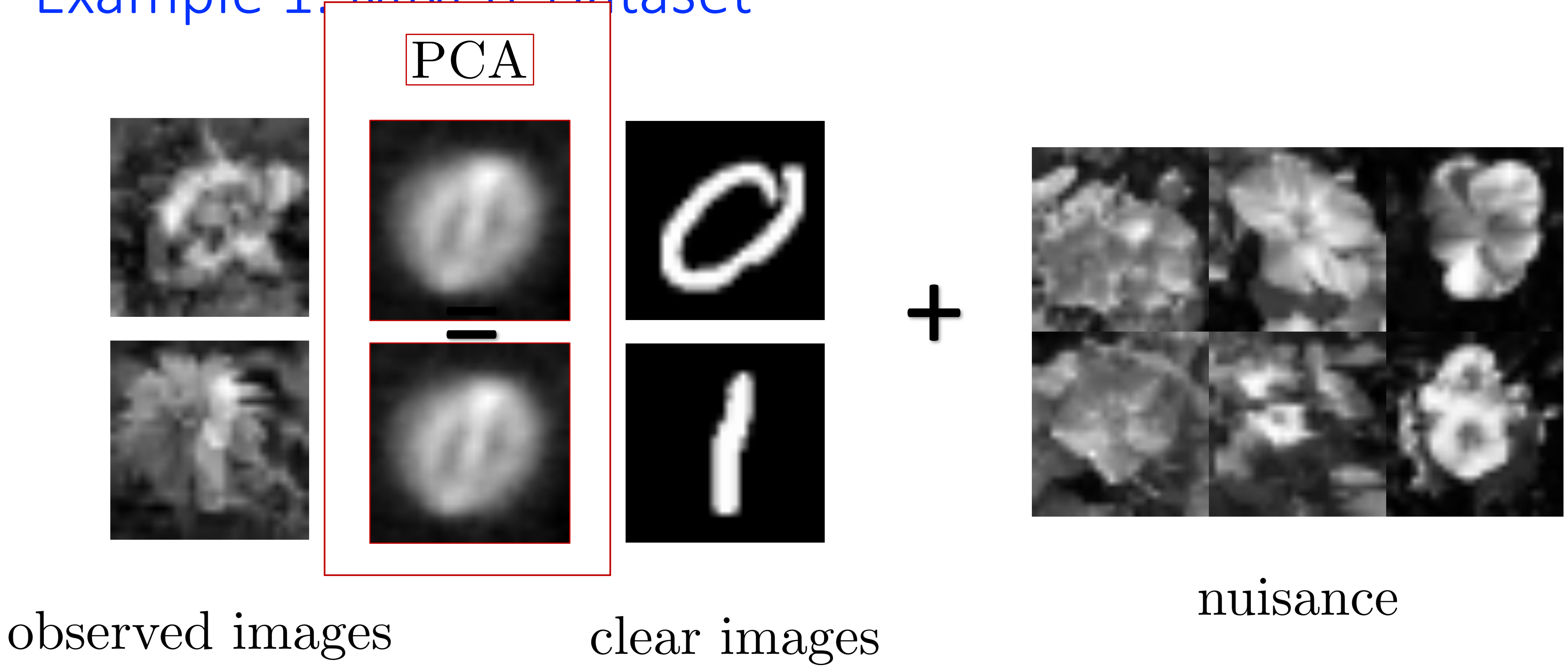
5) 將數據轉換到k個特徵向量構建的新空間中，即Y=PX。

Ryan Lu · Follow
Published in AI反斗城 · 8 min read · Jan 22, 2019

Principal Component Analysis is NOT WORK

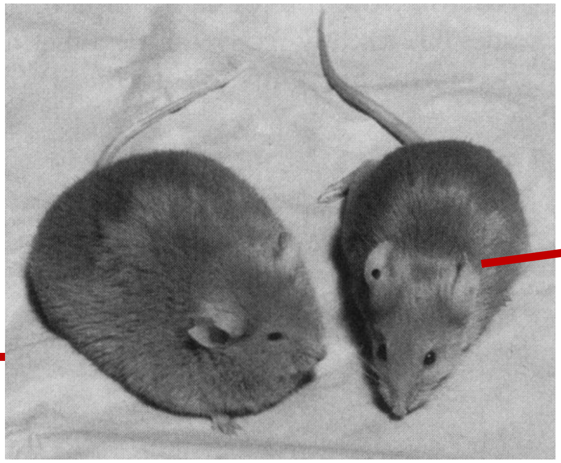
- we start with two datasets

Example 1. MNIST Dataset



Example 2. Mice protein expression

Down Syndrome (DS)
ダウン症

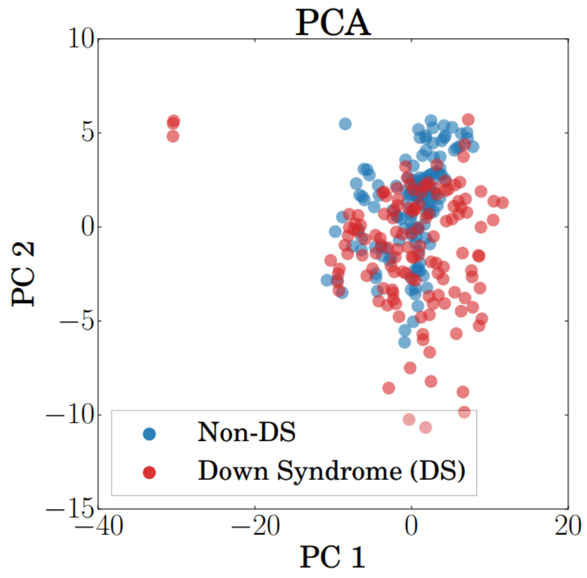


Normal

mice that have received **shock therapy**

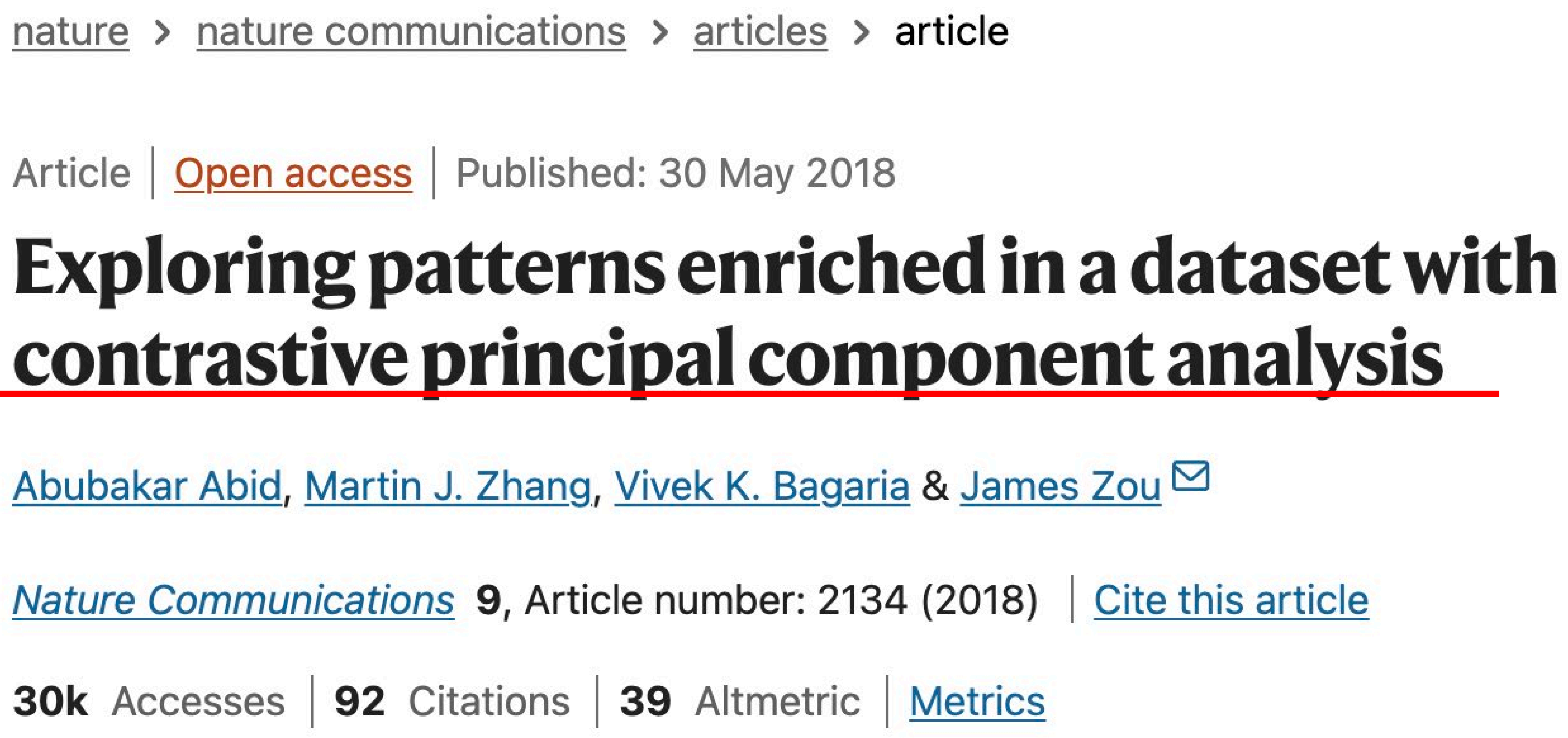
Down Syndrome (DS)

Normal



visualization

Question: how to impove the PCA method



nature > nature communications > articles > article

Article | Open access | Published: 30 May 2018

Exploring patterns enriched in a dataset with contrastive principal component analysis

Abubakar Abid, Martin J. Zhang, Vivek K. Bagaria & James Zou

Nature Communications **9**, Article number: 2134 (2018) | Cite this article

30k Accesses | **92** Citations | **39** Altmetric | Metrics

Input

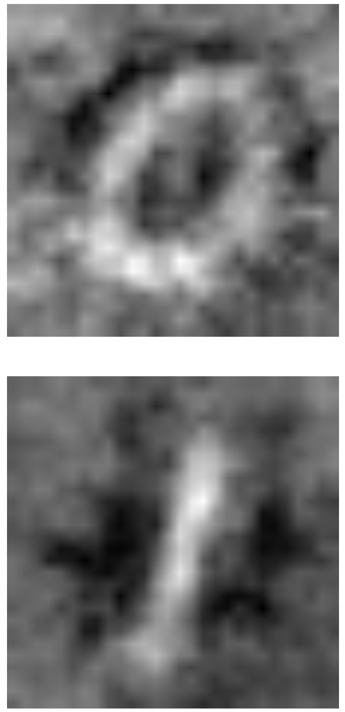
observed images
(target dataset)

X

background dataset

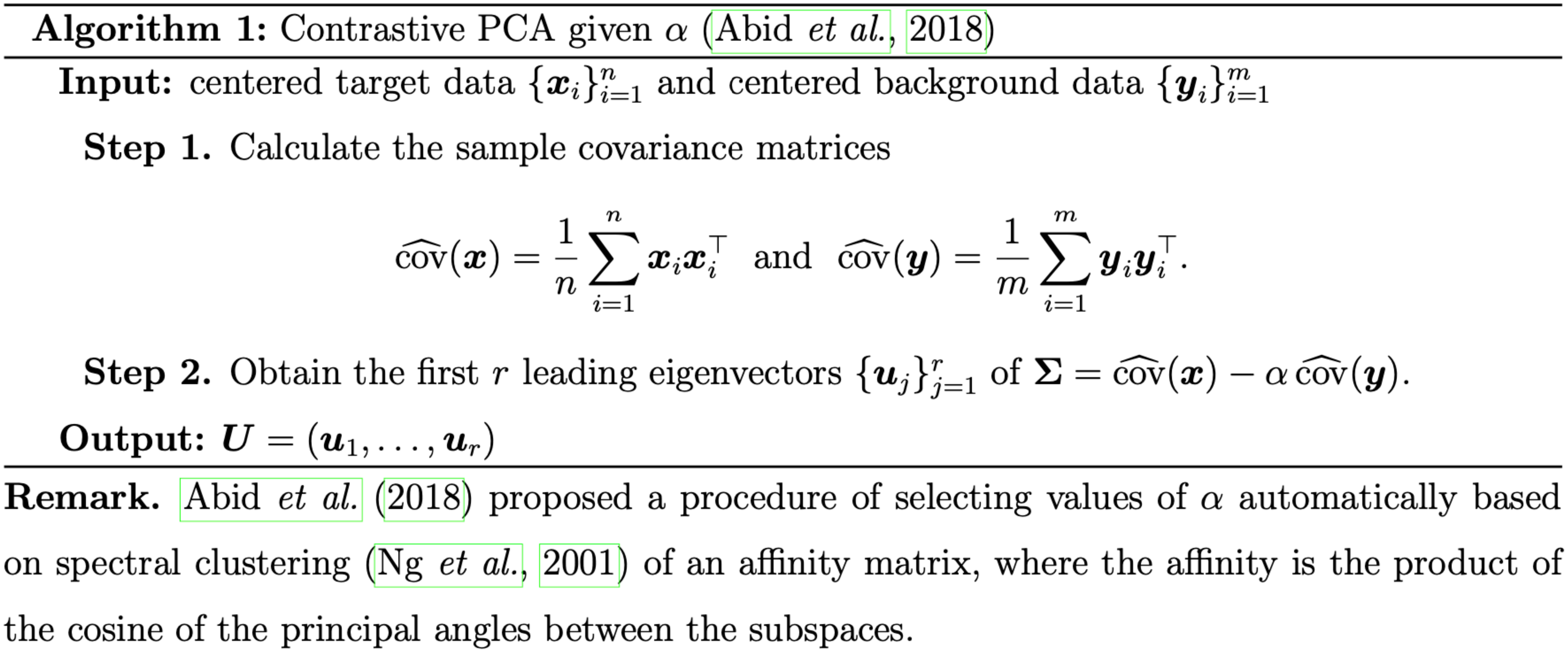
Y

Output



clearer images

Contrastive PCA method (Abid et al. (2017, 2018))



Algorithm 1: Contrastive PCA given α (Abid *et al.*, 2018)

Input: centered target data $\{\boldsymbol{x}_i\}_{i=1}^{n}$ and centered background data $\{\boldsymbol{y}_i\}_{i=1}^{m}$

Step 1. Calculate the sample covariance matrices

$$\widehat{\text{cov}}(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i\boldsymbol{x}_i^\top \quad \text{and} \quad \widehat{\text{cov}}(\boldsymbol{y}) = \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{y}_i\boldsymbol{y}_i^\top .$$

Step 2. Obtain the first r leading eigenvectors $\{\boldsymbol{u}_j\}_{j=1}^{r}$ of $\boldsymbol{\Sigma} = \widehat{\text{cov}}(\boldsymbol{x}) - \alpha\, \widehat{\text{cov}}(\boldsymbol{y})$.

Output: $\boldsymbol{U} = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_r)$

Remark. Abid *et al.* (2018) proposed a procedure of selecting values of α automatically based on spectral clustering (Ng *et al.*, 2001) of an affinity matrix, where the affinity is the product of the cosine of the principal angles between the subspaces.

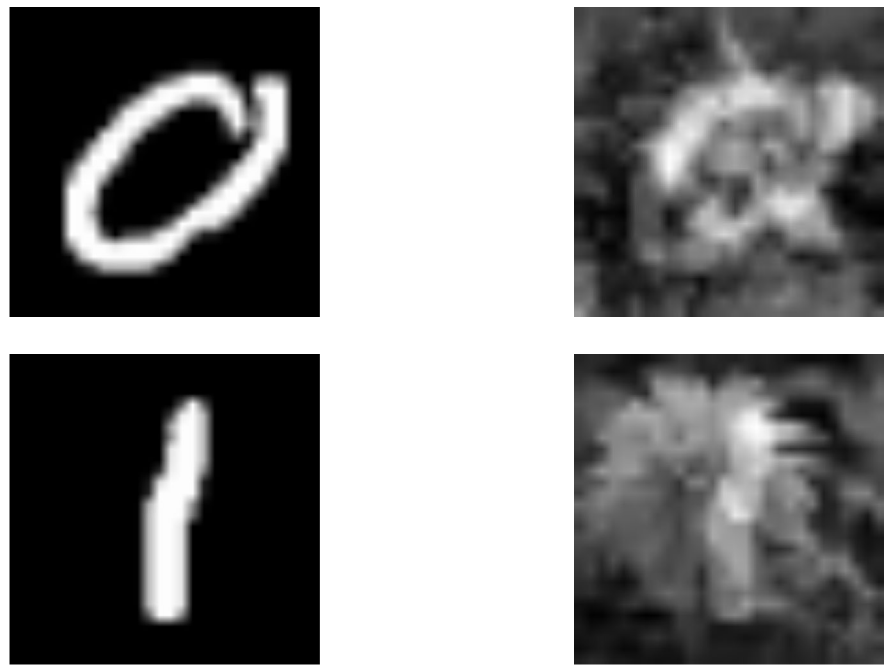
contrastive matrix

$$\boldsymbol{\Sigma} = \underbrace{\widehat{\operatorname{cov}}(\boldsymbol{x})}_{\text{target}} - \alpha \underbrace{\widehat{\operatorname{cov}}(\boldsymbol{y})}_{\text{background}}$$

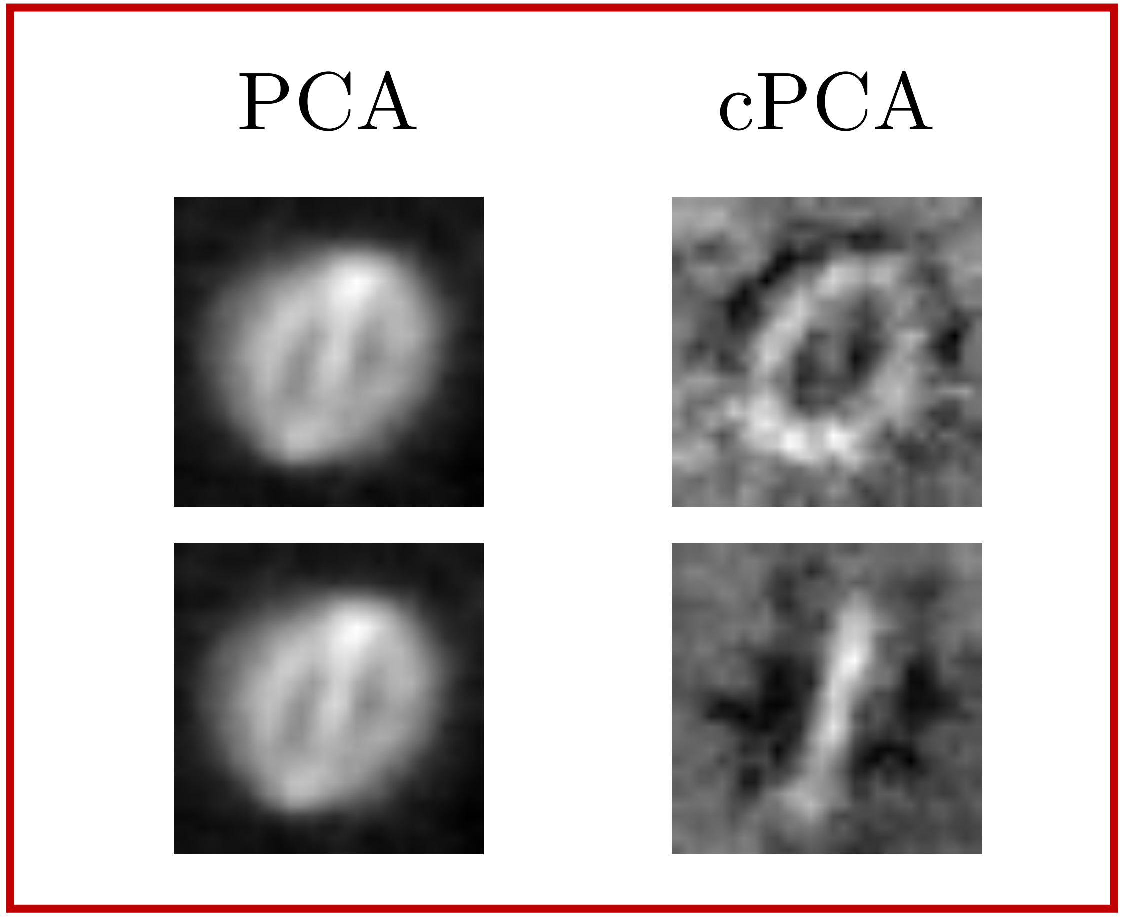
- target data $\{\boldsymbol{x}_i\}_{i=1}^{n}$ is the original data
- background data $\{\boldsymbol{y}_i\}_{i=1}^{m}$ is the data which we select
- cPCA represents the trade-off between maximizing target variance and minimizing the background variance.

Two datasets again!

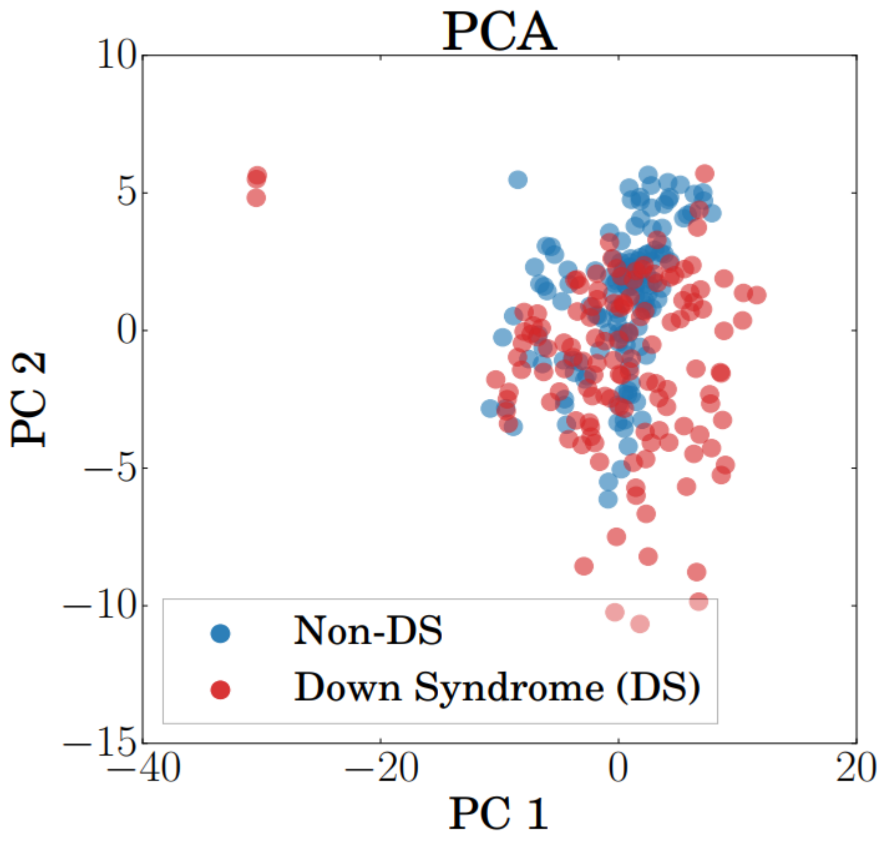
Example 1. MNIST Dataset

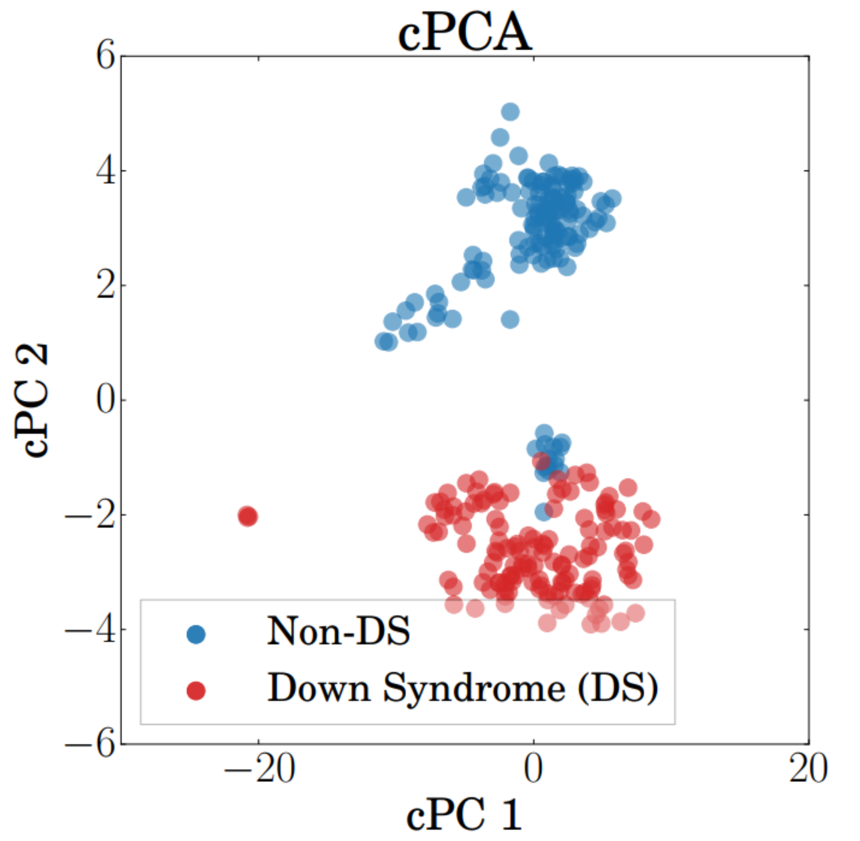


clear images observed images



Example 2. Mice protein expression





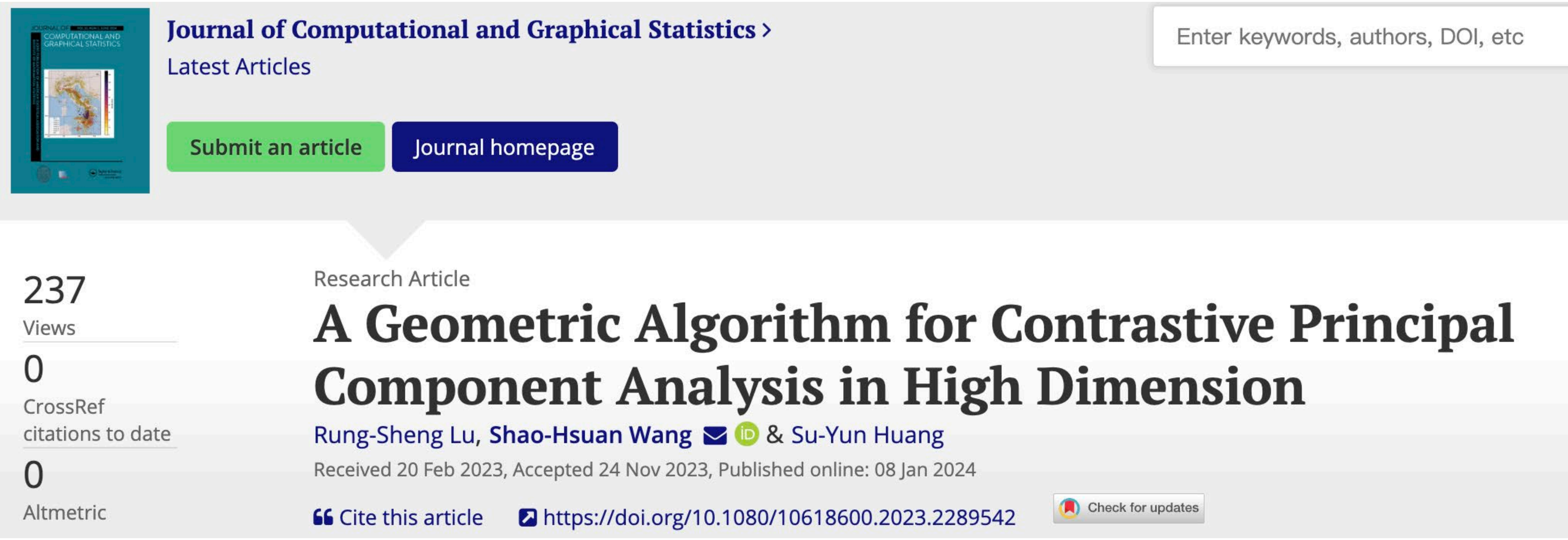
visualization

cPCA is significatly better than PCA in the real data

However, the existing algorithm for the cPCA requires heavy computational loading in high dimension

We would like to investigate cPCA for

developing an efficient algorithm in high dimension



Journal of Computational and Graphical Statistics ›

Latest Articles

Submit an article

Journal homepage

Enter keywords, authors, DOI, etc

237
Views

0
CrossRef
citations to date

0
Altmetric

Research Article

A Geometric Algorithm for Contrastive Principal Component Analysis in High Dimension

Rung-Sheng Lu, Shao-Hsuan Wang & Su-Yun Huang

Received 20 Feb 2023, Accepted 24 Nov 2023, Published online: 08 Jan 2024

Cite this article https://doi.org/10.1080/10618600.2023.2289542 Check for updates

高維度空間隱藏起來的維度

Part I. Geometric curvilinear-search algorithm for cPCA

Part II. Numerical examples

Part I. Geometric curvilinear-search algorithm for cPCA

We turn the optimization to the following problem

$$\boldsymbol{\Sigma} = \widehat{\operatorname{cov}}(\boldsymbol{x}) - \alpha\widehat{\operatorname{cov}}(\boldsymbol{y})$$

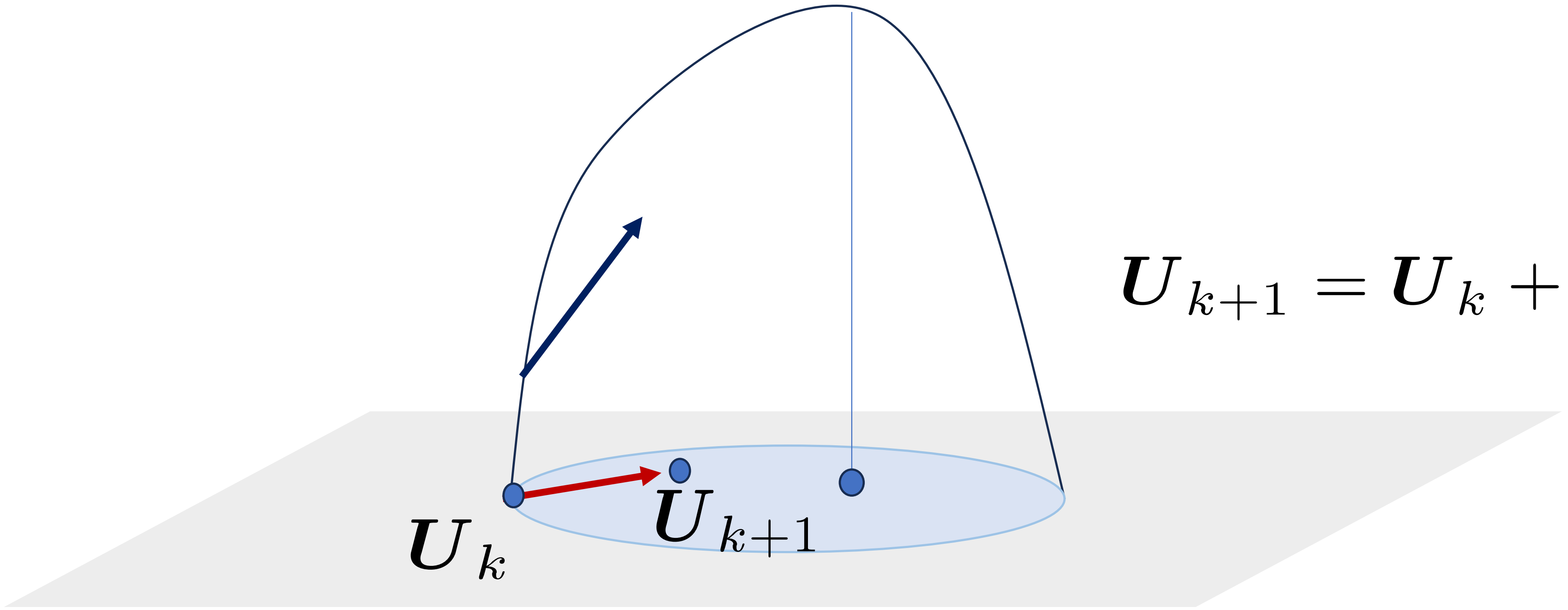
$$\boldsymbol{U}^* = \underset{\boldsymbol{U}\in\mathrm{St}(p,r)}{\operatorname{argmax}} F(\boldsymbol{U}), \quad \text{where } F(\boldsymbol{U}) \equiv \frac{1}{2}\operatorname{tr}\left(\boldsymbol{U}^\top\boldsymbol{\Sigma}\boldsymbol{U}\right).$$

Stiefel manifold

$\mathrm{St}(p,r) \equiv \left\{\boldsymbol{U}\in\mathbb{R}^{p\times r} | \boldsymbol{U}^\top\boldsymbol{U} = \boldsymbol{I}_r\right\}$, which is a submanifold of $\mathbb{R}^{p\times r}$.

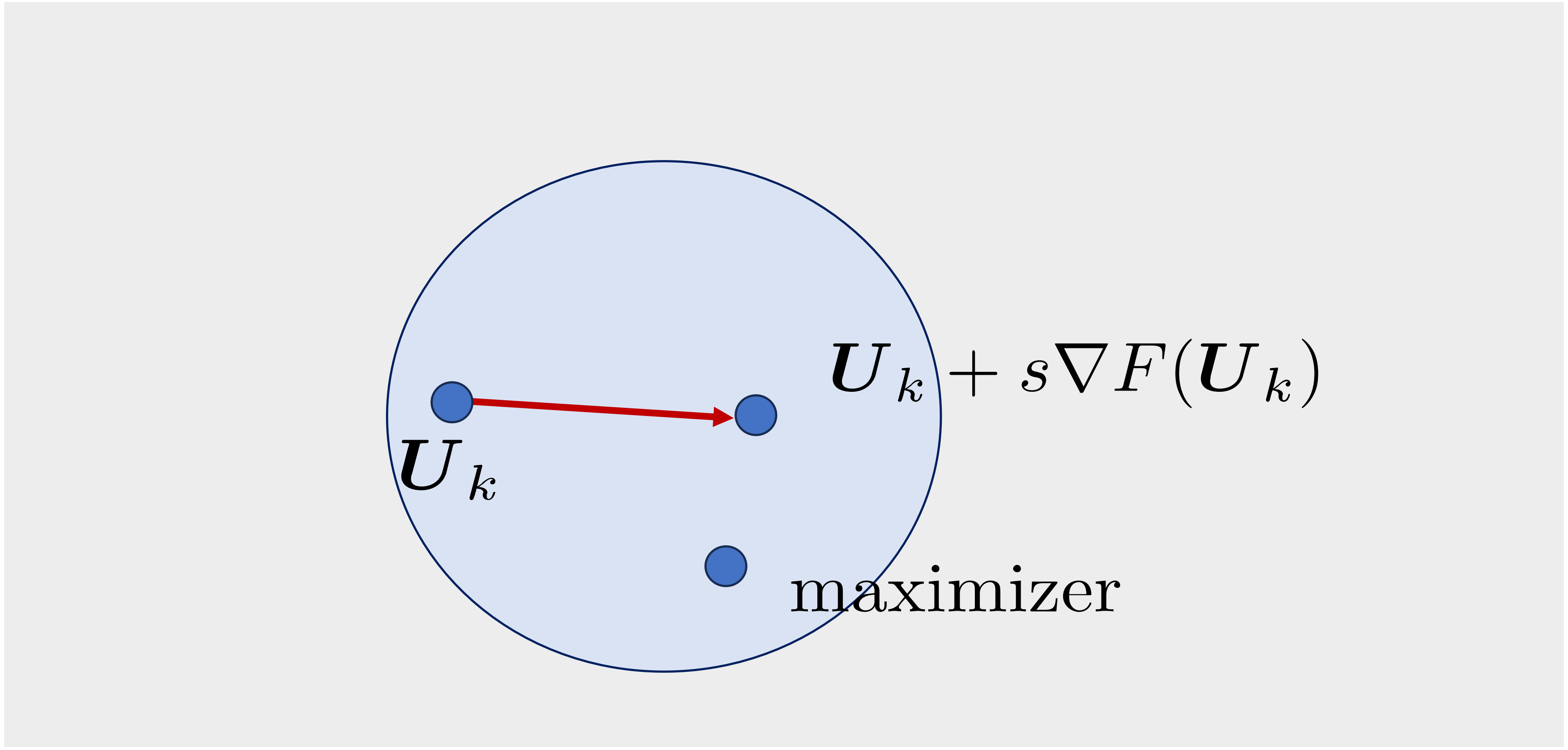
For unconstrained optimization problem, the fastest rising direction of F is its gradient, which is a $p \times r$ matrix given by

$$\nabla F(\boldsymbol{U}) = \left[\frac{\partial F}{\partial U_{ij}}(\boldsymbol{U})\right]_{p\times r} = \boldsymbol{\Sigma U}$$

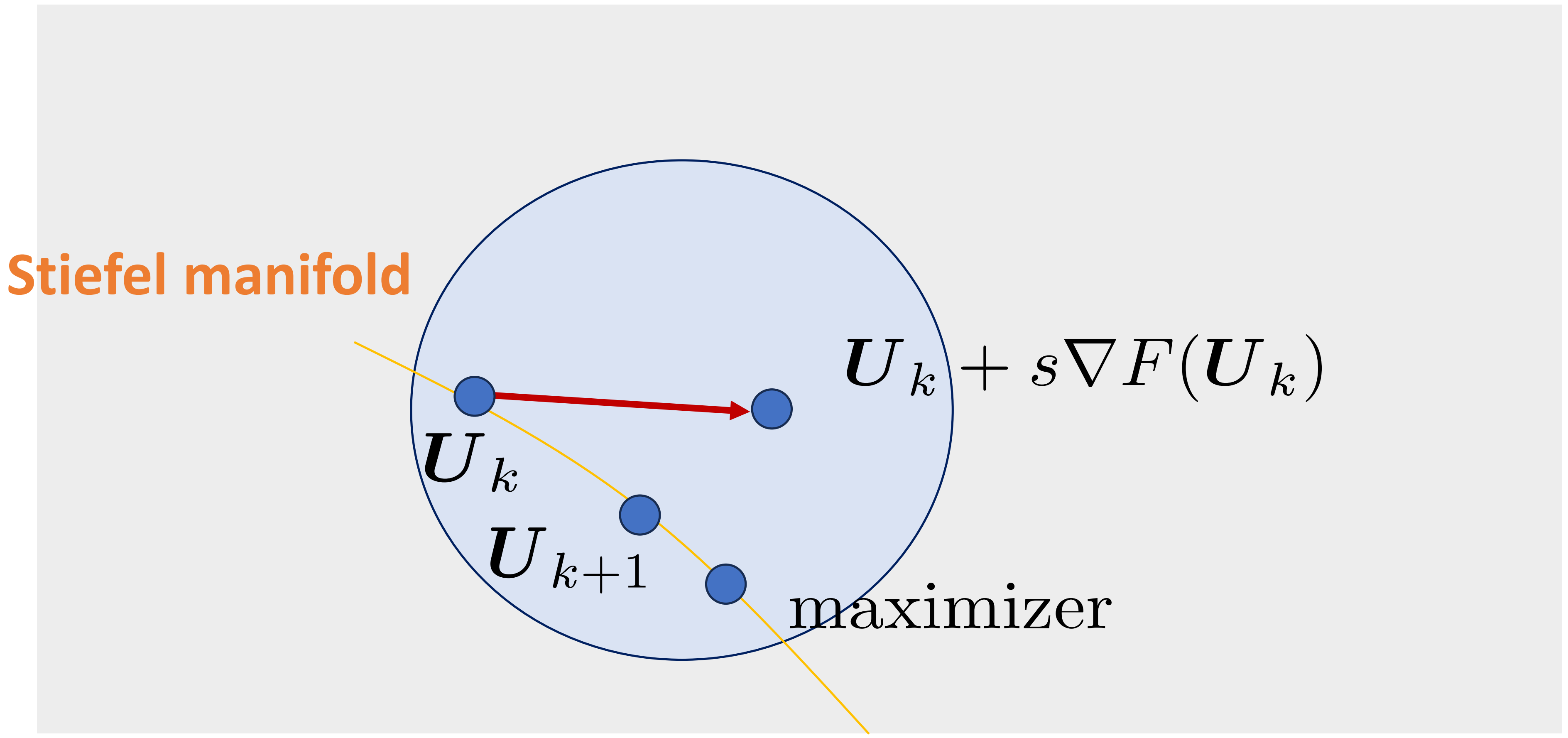


$$\boldsymbol{U}_{k+1} = \boldsymbol{U}_k + s\nabla F(\boldsymbol{U}_k),\ s > 0$$

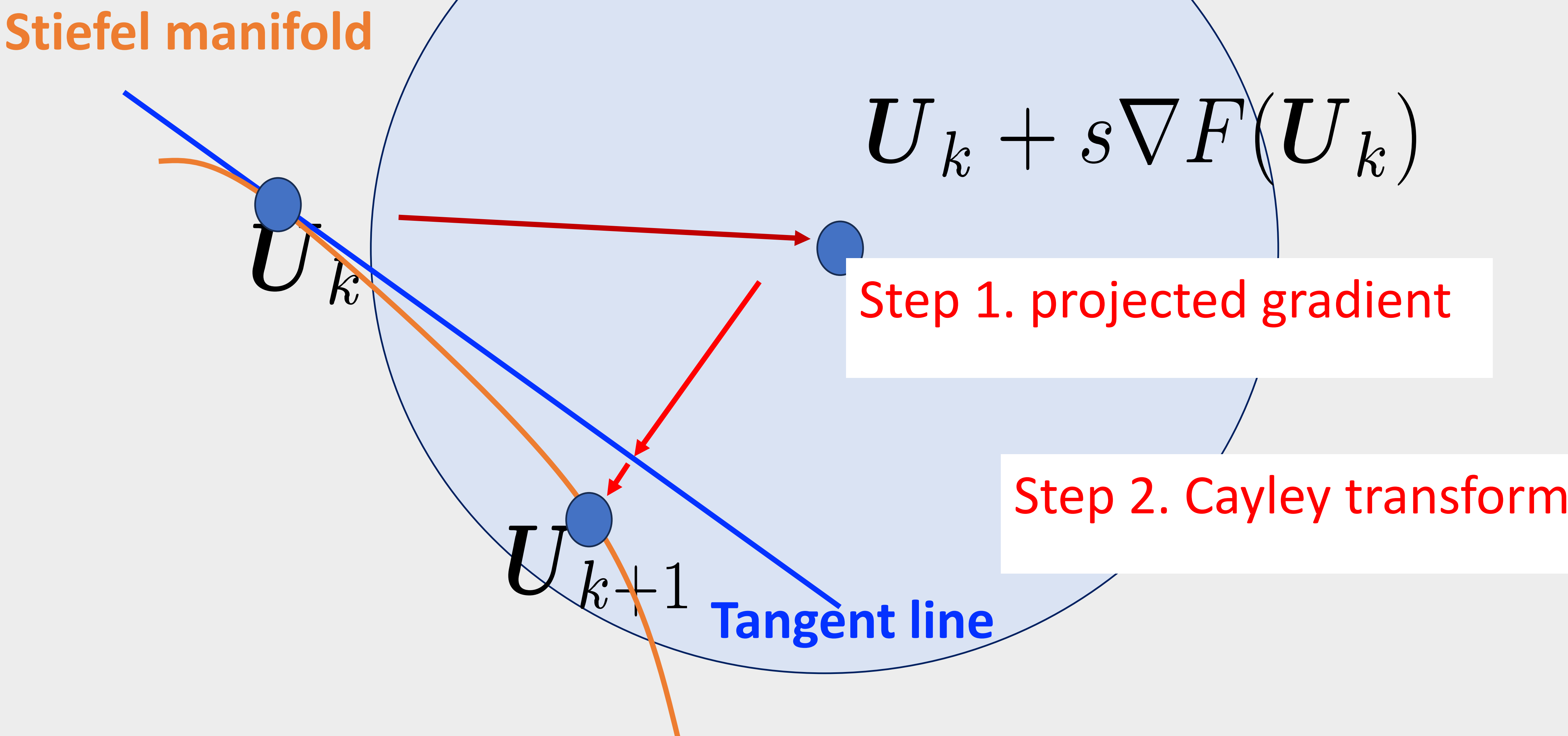
For unconstrained optimization,



For constrained optimization,



The proposed geometric algorithm



The procedure of geometric algorithm

At the k-th iteration,

1. Calculate the projected gradient $\mathcal{P}^{(c)}_{\mathcal{T}_{\boldsymbol{U}^{(k)}}\mathrm{St}(p,r)}\left(\nabla F\left(\boldsymbol{U}^{(k)}\right)\right)$.
2. Generate a Cayley transformed curve $\boldsymbol{C}_{\boldsymbol{U}}(\tau \boldsymbol{T})$, defined as in , which passes $\boldsymbol{U}^{(k)}$ based on $\boldsymbol{T} = \mathcal{P}^{(c)}_{\mathcal{T}_{\boldsymbol{U}^{(k)}}\mathrm{St}(p,r)}\left(\nabla F\left(\boldsymbol{U}^{(k)}\right)\right)$.
3. Determine the step size by using the Armijo rule.
4. Move from $\boldsymbol{U}^{(k)}$ to $\boldsymbol{U}^{(k+1)}$ along the curve in step 2 with the step size determined in step 3.

Projected gradient

1 Define: $\mathcal{T}_{\boldsymbol{U}}\mathrm{St}(p,r) = \{\boldsymbol{T} \in \mathbb{R}^{p\times r} | \boldsymbol{U}^\top \boldsymbol{T} + \boldsymbol{T}^\top \boldsymbol{U} = 0\}.$

2 For any given $\boldsymbol{G} \in \mathbb{R}^{p\times r}$,

$$\boxed{\langle \boldsymbol{A}_1,\ \boldsymbol{A}_2\rangle_e = \mathrm{tr}(\boldsymbol{A}_1^\top \boldsymbol{A}_2)}$$

let $\mathcal{P}^{(e)}_{\mathcal{P}_{\boldsymbol{U}}\mathrm{St}(p,r)}(\boldsymbol{G})$ (Euclidean metric)

and $\mathcal{P}^{(c)}_{\mathcal{P}_{\boldsymbol{U}}\mathrm{St}(p,r)}(\boldsymbol{G})$ (canonical metric)

$$\boxed{\langle \boldsymbol{A}_1,\ \boldsymbol{A}_2\rangle_c = \mathrm{tr}\left(\boldsymbol{A}_1^\top(\boldsymbol{I}_p - \frac{1}{2}\boldsymbol{U}\boldsymbol{U}^\top)\boldsymbol{A}_2\right)}$$

denote the projection of $\boldsymbol{G}$ onto $\mathcal{T}_{\boldsymbol{U}}\mathrm{St}(p,r)$

3 In our case, $\mathcal{P}^{(e)}_{\mathcal{T}_{\boldsymbol{U}}\mathrm{St}(p,r)}(\nabla F(\boldsymbol{U})) = \mathcal{P}^{(c)}_{\mathcal{T}_{\boldsymbol{U}}\mathrm{St}(p,r)}(\nabla F(\boldsymbol{U})) = (\boldsymbol{I}_p - \boldsymbol{U}\boldsymbol{U}^\top)\boldsymbol{\Sigma}\boldsymbol{U}.$

Remark. The matrix $\boldsymbol{T}$, which is tangent to $\mathrm{St}(p,r)$ at $\boldsymbol{U}$, has a general form

$$\boldsymbol{T} = \boldsymbol{U}\boldsymbol{A} + \boldsymbol{U}_{\perp}\boldsymbol{B},$$

where $\boldsymbol{A}$ is $r \times r$ skew-symmetric and $\boldsymbol{B}$ is any $p \times r$ matrix. Using the representation of tangent vectors given in (6), we have

$$\langle \boldsymbol{T},\ \boldsymbol{T} \rangle_e \ = \ \mathrm{tr}(\boldsymbol{A}^\top \boldsymbol{A}) + \mathrm{tr}(\boldsymbol{B}^\top \boldsymbol{B}) = 2\sum_{i<j} a_{ij}^2 + \sum_{ij} b_{ij}^2.$$

For cannoical metric,

$$\langle \boldsymbol{A}_1,\ \boldsymbol{A}_2 \rangle_c = \mathrm{tr}\left(\boldsymbol{A}_1^\top (\boldsymbol{I}_p - \frac{1}{2}\boldsymbol{U}\boldsymbol{U}^\top)\boldsymbol{A}_2 \right)$$

$$\Rightarrow \quad \langle \boldsymbol{T},\ \boldsymbol{T} \rangle_c = \textstyle\sum_{i<j} a_{ij}^2 + \sum_{ij} b_{ij}^2$$

Cayley transform

Given $\boldsymbol{U} \in \mathrm{St}\,(p, r)$ and $\boldsymbol{T} \in \mathcal{T}_{\boldsymbol{U}}\mathrm{St}\,(p, r)$, the Cayley retraction mapping

$\boldsymbol{C}_{\boldsymbol{U}}(\boldsymbol{T})$ is defined as

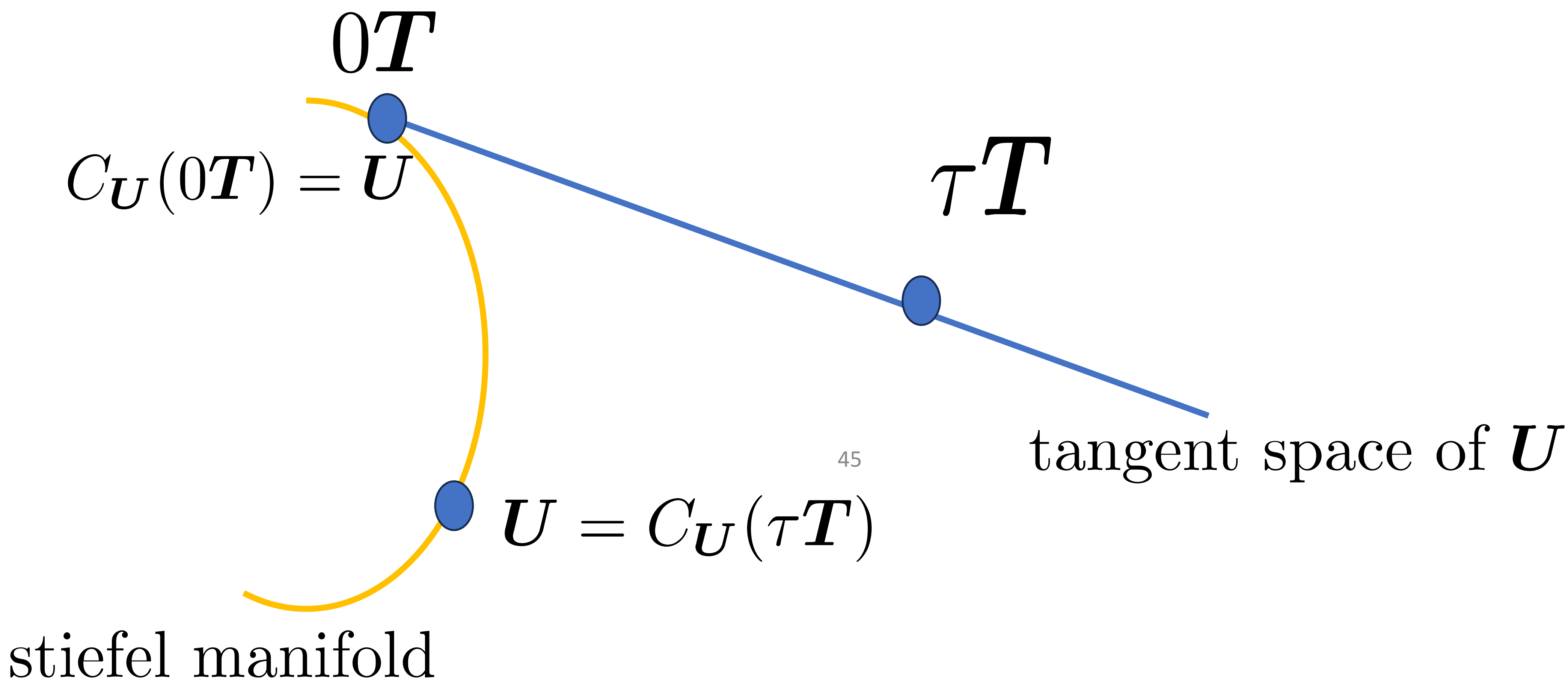
$$\boldsymbol{C}_{\boldsymbol{U}}\left(\boldsymbol{T}\right) \equiv \left(\boldsymbol{I}_p + \frac{1}{2}\boldsymbol{W}\right)\left(\boldsymbol{I}_p - \frac{1}{2}\boldsymbol{W}\right)^{-1}\boldsymbol{U},$$

where $\boldsymbol{W} = \boldsymbol{\Pi}_{\boldsymbol{U}}\left(\boldsymbol{T}\right)\boldsymbol{U}^\top - \boldsymbol{U}\boldsymbol{\Pi}_{\boldsymbol{U}}\left(\boldsymbol{T}\right)^\top$ with $\boldsymbol{\Pi}_{\boldsymbol{U}}\left(\boldsymbol{T}\right) = \left(\boldsymbol{I}_p - \frac{1}{2}\boldsymbol{U}\boldsymbol{U}^\top\right)\boldsymbol{T}$.

Cayley transform

The Cayley transformed curve $\boldsymbol{C}_{\boldsymbol{U}}(\tau\boldsymbol{T})$ satisfies the following properties.

1. $\boldsymbol{C}_{\boldsymbol{U}}(\tau\boldsymbol{T})$ stays in $\mathrm{St}(p,r)$ for all $\tau \in \mathbb{R}$.
2. $\boldsymbol{C}_{\boldsymbol{U}}(\mathbf{0}) = \boldsymbol{U}$.
3. $\frac{d}{d\tau}\boldsymbol{C}_{\boldsymbol{U}}(\tau\boldsymbol{T})\Big|_{\tau=0} = \boldsymbol{T}$.


$0\boldsymbol{T}$
$C_{\boldsymbol{U}}(0\boldsymbol{T}) = \boldsymbol{U}$
$\tau\boldsymbol{T}$
tangent space of $\boldsymbol{U}$
45
$\boldsymbol{U} = C_{\boldsymbol{U}}(\tau\boldsymbol{T})$
stiefel manifold

Algorithm 2: Geometric algorithm for maximizer of $F(\boldsymbol{U})$.

```
Input: initial point $\boldsymbol{U}^{(0)} \in \mathrm{St}\,(p, r)$
    Step 1. Calculate the gradient $\nabla F$ and the projected gradient $\mathcal{P}^{(c)}_{\mathcal{T}_{\boldsymbol{U}^{(k)}}\mathrm{St}(p,r)}(\nabla F)$.
    Step 2. Obtain the corresponding $\boldsymbol{C}(\boldsymbol{U}^{(k)},\ \tau) \overset{\text{def}}{=} \boldsymbol{C}_{\boldsymbol{U}^{(k)}}\left(\tau \mathcal{P}^{(c)}_{\mathcal{T}_{\boldsymbol{U}^{(k)}}\mathrm{St}(p,r)}\left(\nabla F\left(\boldsymbol{U}^{(k)}\right)\right)\right)$.
    Step 3. Implement the Armijo rule to find a suitable step size $\tau^{(k)}$.
    Step 4. Obtain $\boldsymbol{U}^{(k+1)} = \boldsymbol{C}(\boldsymbol{U}^{(k)},\ \tau^{(k)})$.
    Step 5. Iterate Steps 1-4 until $|F(\boldsymbol{U}^{(k+1)}) - F(\boldsymbol{U}^{(k)})|/|F(\boldsymbol{U}^{(k)})|$ is smaller than an
        error tolerance. Notationally, set $k = \infty$ to indicate that the iteration has stopped.
Result: the maximizer $\boldsymbol{U}^{(\infty)}$ of $F(\boldsymbol{U})$.
```

Computational efficiencies

Let

- n: the sample size of the target data
- m: the sample size of the background data
- p: the data dimension
- r: the number of leading components.

cPCA

	eigen-decom	geometric
Time complexity	$O((n \vee m)p^2)$	$O((n \vee m)pr)$ Wins
Space complexity	$O(p^2)$	$O((n \vee m)p)$ Wins

wheh p is very large (with respect to m and n)

Numerical examples

MNIST dataset

Case 1. Not-large dimension

The MNIST dataset consists of 70000 images of handwritten digits, 60000 for training and 10000 for testing.

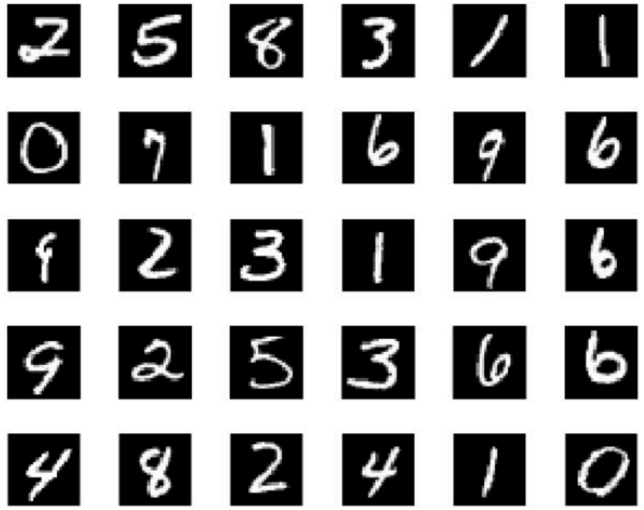
- Each image is superimposed with a different complex background, namely, an image of a section of grass from a lawn.

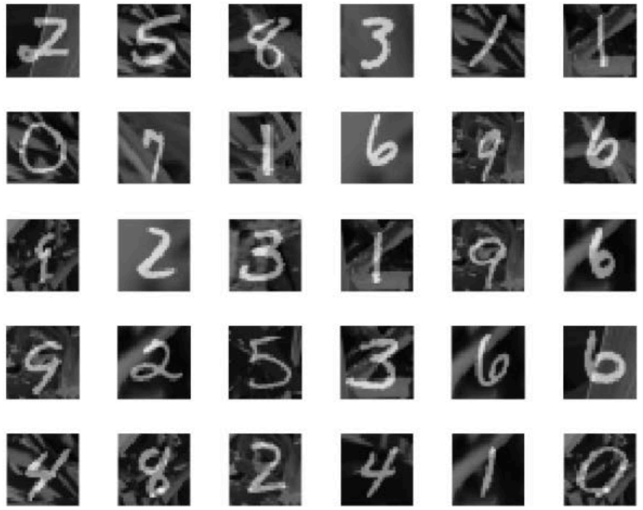
- The images are of size 28×28, which are transformed into a 784×1 covariate vector.

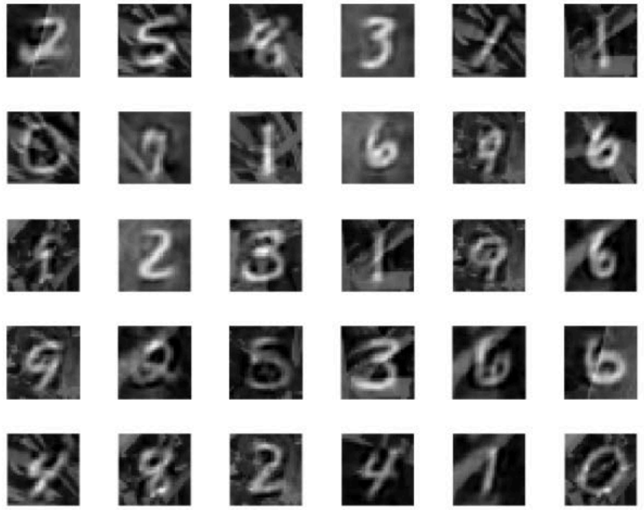
Goal: The commonly used method for cPCA — **the eigenvalue decomposition** works in not-large dimension. In this case, we can compare the performance of reconstruction from the two cPCA algorithms.

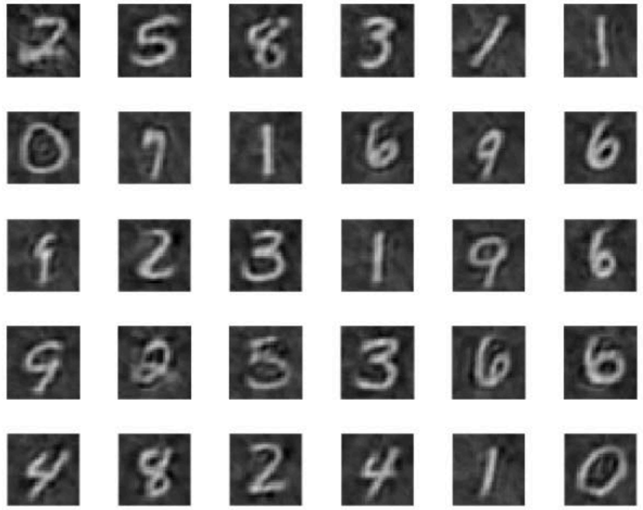
(a) cPCA via the eigenvalue decomposition

(b) cPCA via the proposed algorithm


(a) Clear test images


(b) Test images mixed with grass


(c) Test image reconstructions by PCA


(d) Test image reconstructions by cPCA

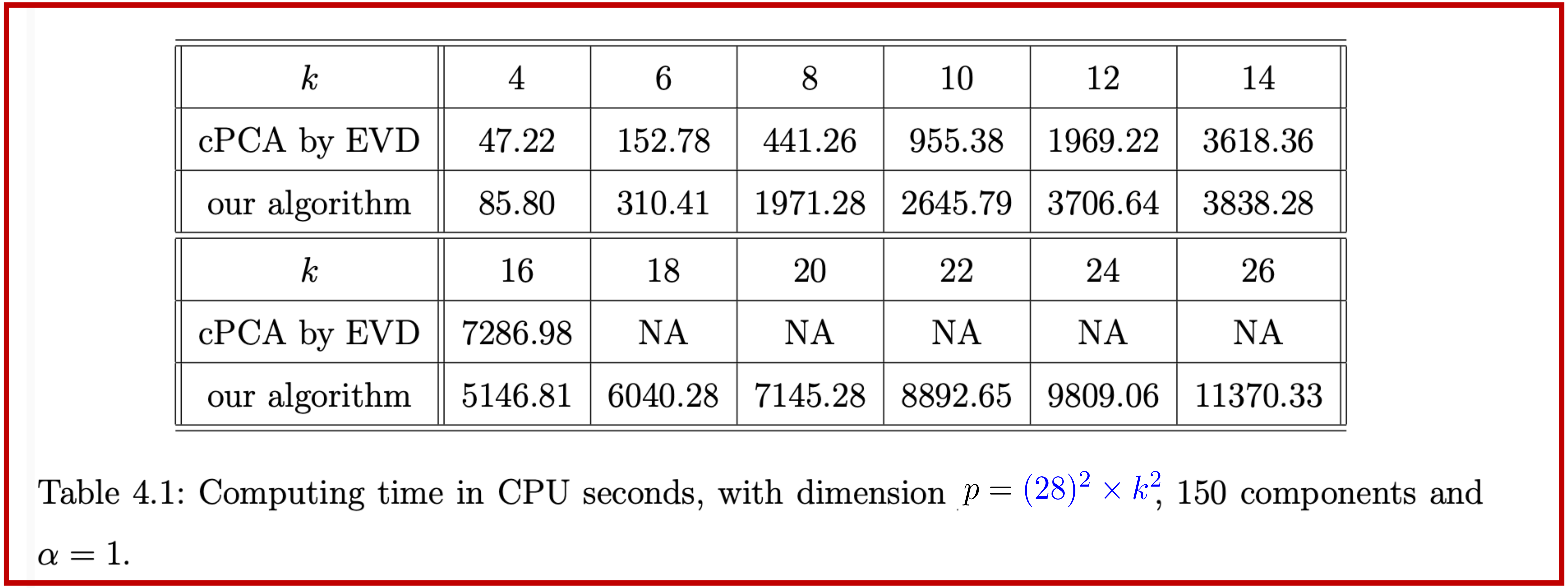
Case 2. High dimension

The MNIST dataset consists of 70000 images of handwritten digits, 60000 for training and 10000 for testing.

- To further illustrate the advantage of the proposed method in high dimensions, we use larger images. Larger images are formed by combining 16 images of 28×28 leading to image size 112×112. Then, the dimensionality becomes 12,544.

$$p = (28)^2 \times 4^2$$

Using “super computer” — an Nvidia DGX A100 server, featuring an AMD EPYC 7742 64-Core CPU and 1008 GB of RAM.



k	4	6	8	10	12	14	16	18	20	22	24	26
cPCA by EVD	47.22	152.78	441.26	955.38	1969.22	3618.36	7286.98	NA	NA	NA	NA	NA
our algorithm	85.80	310.41	1971.28	2645.79	3706.64	3838.28	5146.81	6040.28	7145.28	8892.65	9809.06	11370.33

Table 4.1: Computing time in CPU seconds, with dimension $p = (28)^2 \times k^2$, 150 components and $\alpha = 1$.

- the symbol “NA” indicating cases where the problem size exceeded the machine’s capacity, resulting in unavailable results.

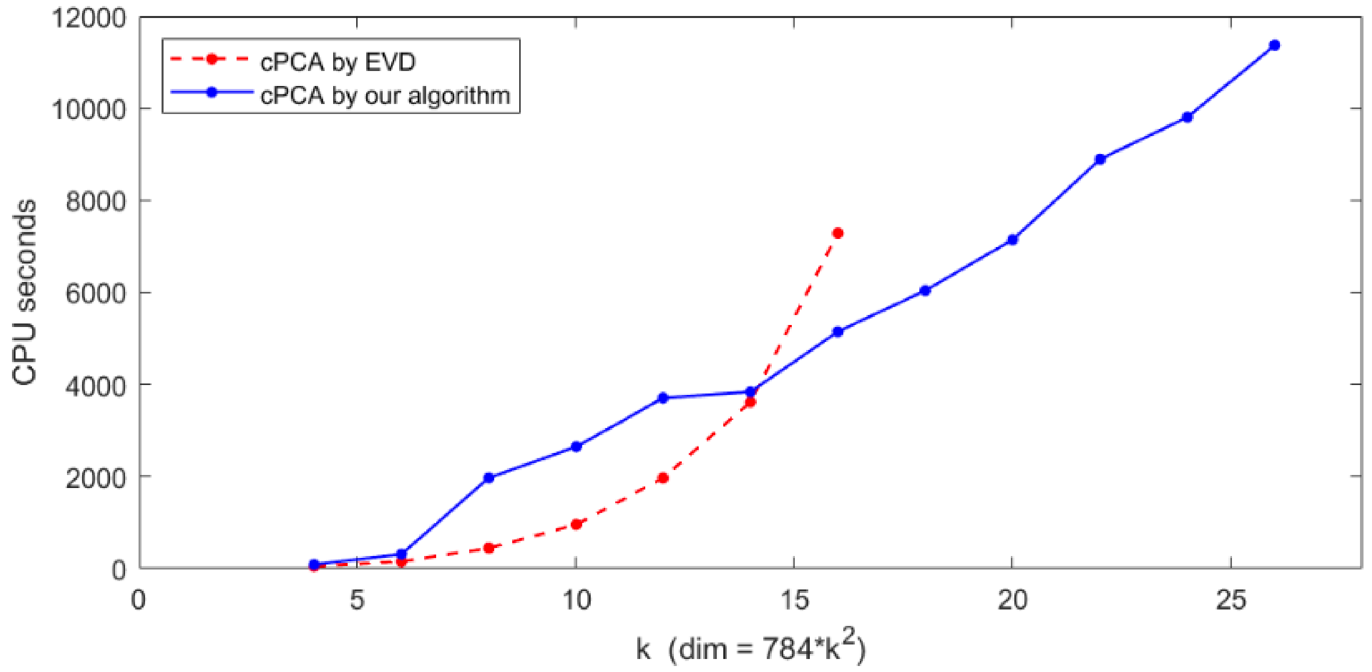


Figure 4.5: Computing time comparison, with 150 components and $\alpha = 1$.

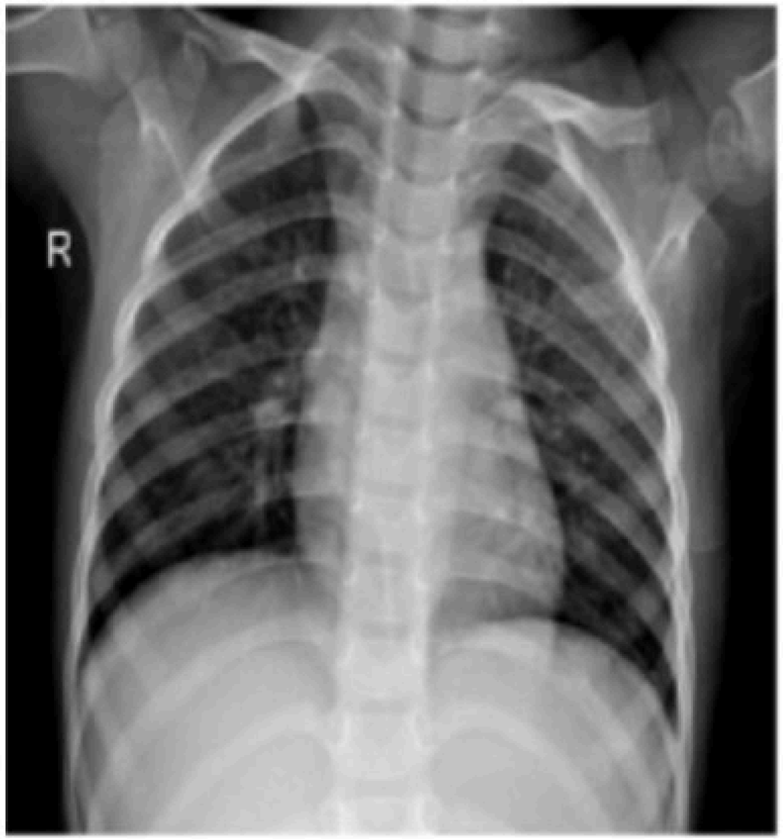
Chest X-ray dataset

Chest X-ray dataset — high dimenaionl case

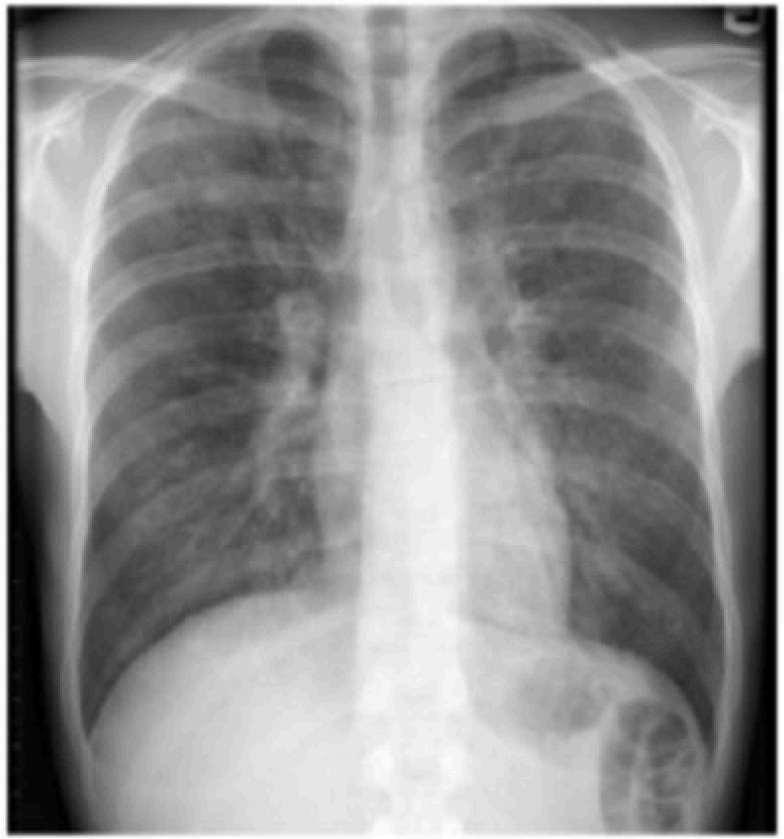
The dataset contains X-ray images with two categories (Pneumonia/Normal). The training dataset consists of 3875 pneumonia images and 1341 normal images, and the testing dataset consists of 390 pneumonia images and 234 normal images.

Each image is resized to 224×224, which leads to a high dimensionality 50176. In this data example,

- pneumonia images (target dataset)
- normal images (background dataset)



(a) Normal



(b) Pneumonia

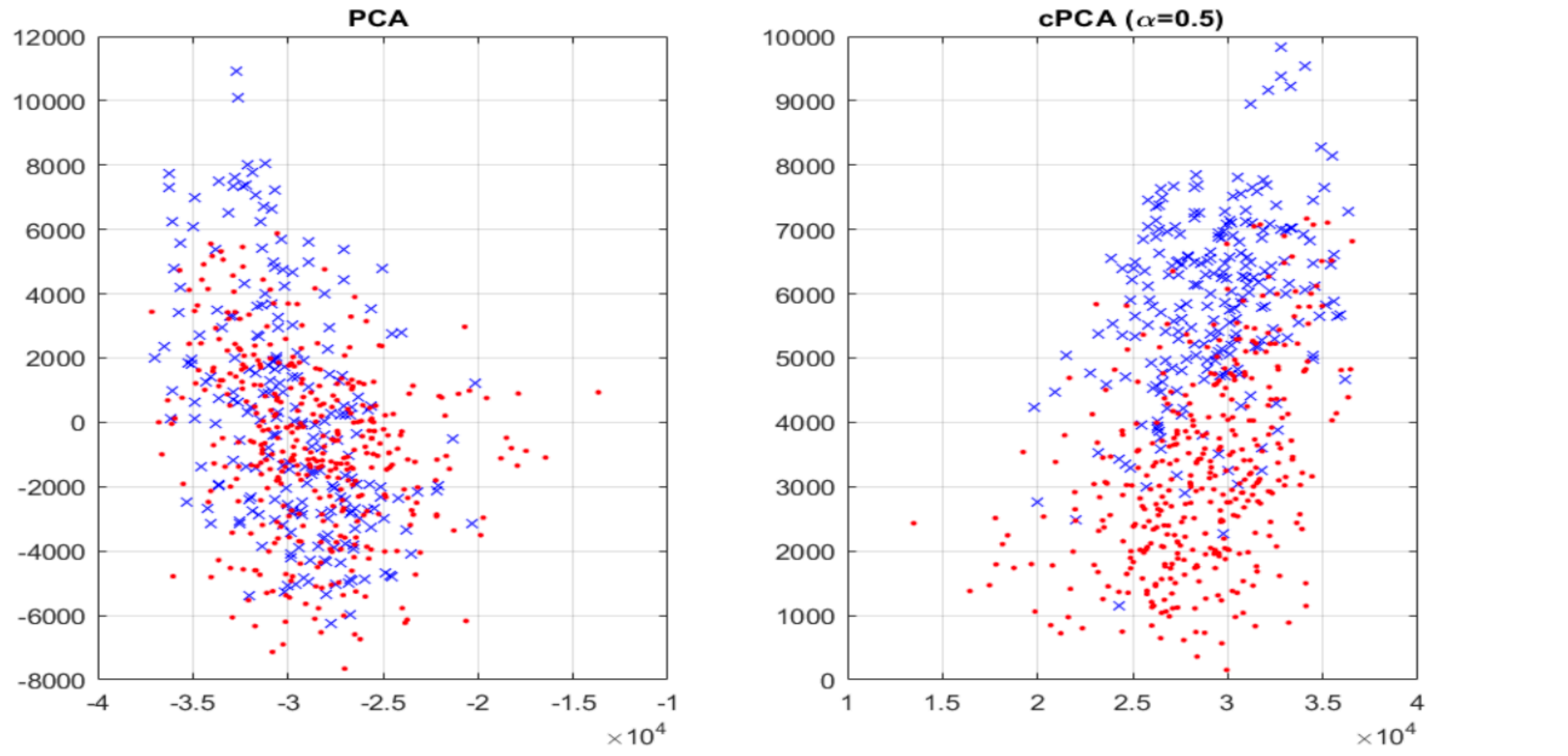


Figure 4.6: Visualization of test data projected onto the leading 2D subspaces obtained, respectively, by PCA (left) and by the proposed cPCA algorithm (right).