# Learning From Data
# Lecture 8: Unsupervised Learning II

Yang Li    yangli@sz.tsinghua.edu.cn

TBSI

11/14/2019

# Today's Lecture

Midterm Statistics

Unsupervised Learning (Part II)

- Kernel PCA (Cont')
- Independent Component Analysis (ICA)
- Canonical Correlation Analysis (CCA)

# PCA Review

## PCA Dimension reduction

- Find principal components $u_1, \ldots, u_n$ that are mutually orthogonal (uncorrelated)
- Most of the variations in $x$ will be accounted for by $k$ principal components where $k \ll n$.

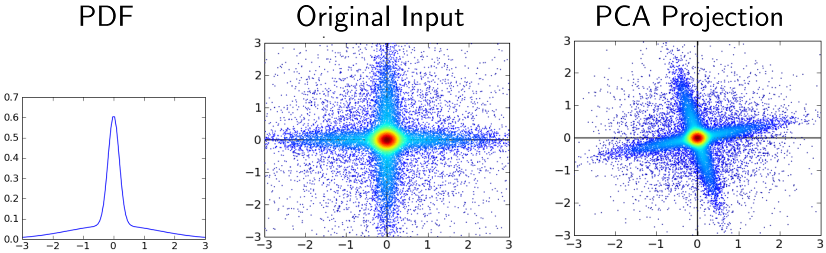
## Main steps

1. Standardize $x$ such that $Mean(x) = 0, Var(x_j) = 1$ for all $j$
2. Compute $\Sigma = cov(x)$
3. Find principal components $u_1, \ldots, u_n$ by eigenvalue decomposition: $\Sigma = U \Lambda U^T$. ← *$U$ is an orthogonal basis in $\mathbb{R}^n$*
4. Project data on first the $k$ principal components: $Z_k = XU_k$

# PCA Limitations

- Assumes input data is real and continuous
- Assumes **approximate normality** of input space (but may still work well on non-normally distributed data in practice)
  $\leftarrow$ *sample mean & covariance must be sufficient statistics*

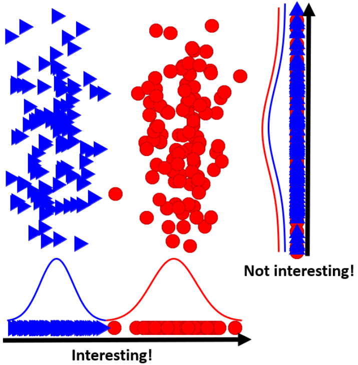
Example of strongly non-normal distributed input:

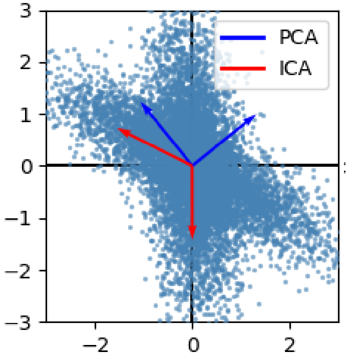


# PCA Limitations

PCA results may not be useful when

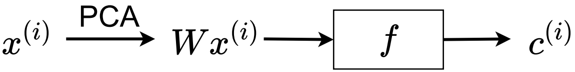
- Axes of larger variance is less ‘interesting‘ than smaller ones.
- Axes of variations are not orthogonal;
- Data has non-linear relationships (see kernel PCA)



# Kernel PCA

Feature extraction using PCA




Linear PCA assumes data are separable in $\mathbb{R}^n$

## A non-linear generalization

- Project data into higher dimension using feature mapping $\phi : \mathbb{R}^n \to \mathbb{R}^d$ $(d \geq n)$
- Feature mapping is defined by a kernel function $K\left(x^{(i)}, x^{(j)}\right) = \phi(x^{(i)})^T \phi(x^{(j)})$ or kernel matrix $K \in \mathbb{R}^{m \times m}$
- We can now perform standard PCA in the feature space

# Kernel PCA

(Bernhard Schoelkopf, Alexander J. Smola, and Klaus-Robert Mueller. 1999. *Kernel principal component analysis*. In Advances in kernel methods)

Sample covariance matrix of feature mapped data (assuming $\phi(x)$ is centered)

$$\Sigma = \frac{1}{m} \sum_{i=1}^{m} \phi(x^{(i)}) \phi(x^{(i)})^T \in \mathbb{R}^{d \times d}$$

Let $(\lambda_k, u_k), k = 1, \ldots, d$ be the eigen decomposition of $\Sigma$:

$$\Sigma u_k = \lambda_k u_k$$

PCA projection of $x^{(l)}$ onto the *kth* principal component $u_k$:

$$\phi(x^{(l)})^T u_k$$

How to avoid evaluating $\phi(x)$ explicitly?

# The Kernel Trick

Represent projection $\phi(x^{(l)})^T u_k$ using kernel function $K$:

- Write $u_k$ as a linear combination of $\phi(x^{(1)}), \ldots, \phi(x^{(m)})$:

$$u_k = \sum_{i=1}^{m} \alpha_k^i \phi(x^{(i)})$$

- PCA projection of $x^{(l)}$ using kernel function $K$:

$$\phi(x^{(l)})^T u_k = \phi(x^{(l)})^T \sum_{i=1}^{m} \alpha_k^i \phi(x^{(i)}) = \sum_{i=1}^{m} \alpha_k^i K(x^{(l)}, x^{(i)})$$

How to find $\alpha_k^i$'s directly ?

# The Kernel Trick

Kth eigenvector equation:

$$\Sigma u_k = \left( \frac{1}{m} \sum_{i=1}^{m} \phi(x^{(i)}) \phi(x^{(i)})^T \right) u_k = \lambda_k u_k$$

- Substitute $u_k = \sum_{i=1}^{m} \alpha_k^{(i)} \phi(x^{(i)})$, we obtain

  $$K \alpha_k = \lambda_k m \alpha_k$$

  where $\alpha_k = \begin{bmatrix} \alpha_k^1 \\ \vdots \\ \alpha_k^m \end{bmatrix}$ can be solved by eigen decomposition of $K$

- Normalize $\alpha_k$ such that $u_k^T u_k = 1$:

  $$u_k^T u_k = \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_k^i \alpha_k^j \phi(x^{(i)})^T \phi(x^{(j)}) = \alpha_k^T K \alpha_k = \lambda_k m (\alpha_k^T \alpha_k)$$

  $$\|\alpha_k\|^2 = \frac{1}{\lambda_k m}$$

# Kernel PCA

When $\mathbb{E}[\phi(x)] \neq 0$ , we need to center $\phi(x)$:

$$\widetilde{\phi}(x^{(i)}) = \phi(x^{(i)}) - \frac{1}{m}\sum_{l=1}^{m} \widetilde{\phi}(x^{(l)})$$

The "centralized" kernel matrix is

$$\tilde{K}_{i,j} = \widetilde{\phi}(x^{(i)})^T \widetilde{\phi}(x^{(j)})$$

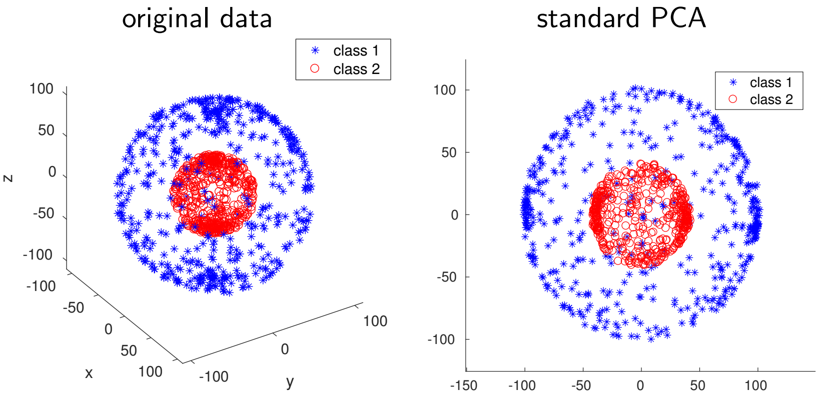
In matrix notation:

$$\widetilde{K} = K - \mathbf{1}_m K - K\mathbf{1}_m + \mathbf{1}_m K \mathbf{1}_m$$

where $\mathbf{1}_m = \begin{bmatrix} 1/m & \dots & 1/m \\ \vdots & \ddots & \vdots \\ 1/m & \dots & 1/m \end{bmatrix} \in \mathbb{R}^{mxm}$

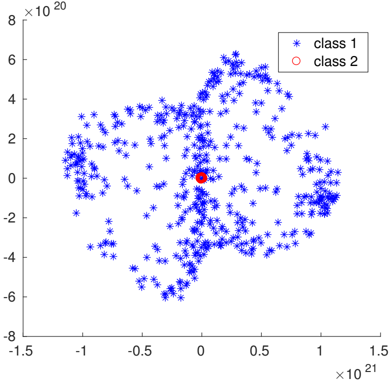
Use $\widetilde{K}$ to compute PCA

# Kernel PCA Example



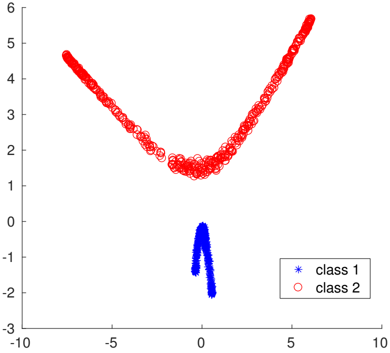
# Kernel PCA Example

Polynomial kernel PCA




$$k(x, x') = (x \cdot x' + 1)^5$$

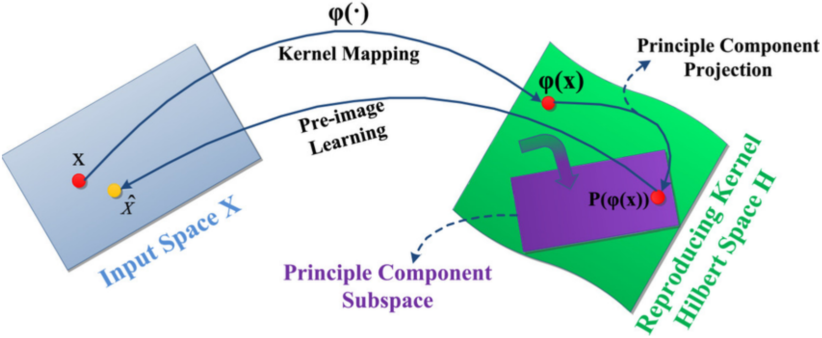
Gaussian kernel PCA




$$k(x, x') = \exp\left(-\frac{||x - x'||^2}{2\sigma^2}\right)$$

# Discussions of kernel PCA

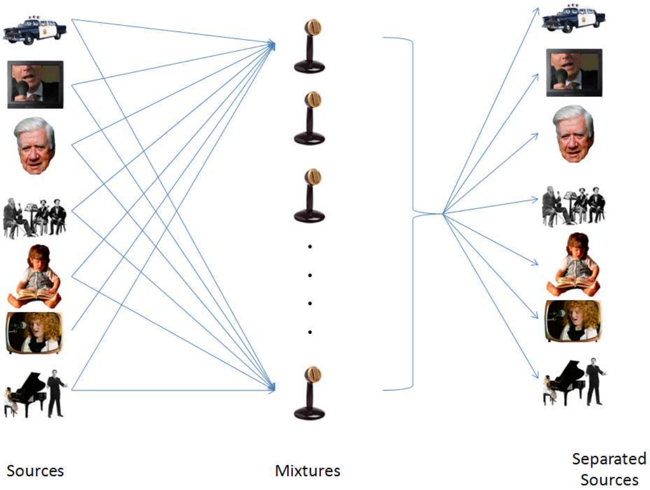
- Often used in clustering, abnormality detection, etc
- Requires finding eigenvectors of $m \times m$ matrix instead of $n \times n$
- Dimension reduction by projecting to k-dimensional principal subspace is generally not possible




**The Pre-Image problem**: reconstruct data in input space $x$ from feature space vectors $\phi(x)$

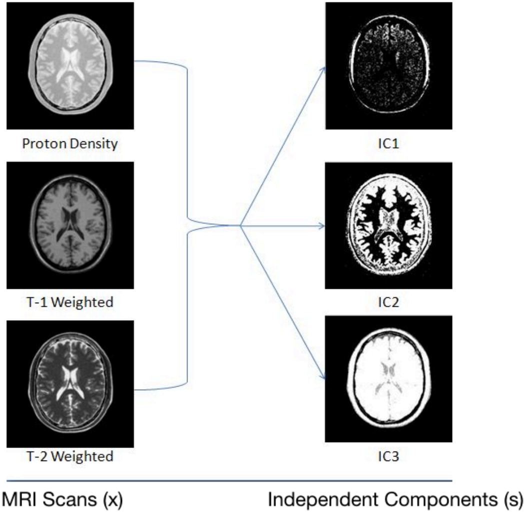
# The cocktail party problem

- $n$ microphones at different locations of the room, each recording a mixture of $n$ sound sources
- How to "unmix" the sound mixtures?



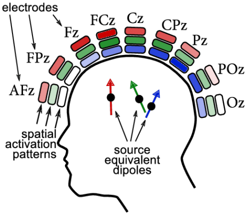
# Brian imaging

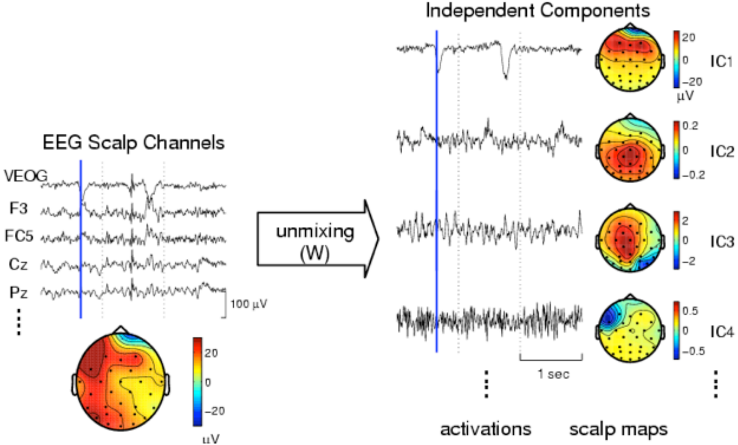
- Different brain matters: gray matter, white matter, cerebrospinal fluid (CSF), fat, muscle/skin, glial matter etc.
- An MRI scan is a mixture of different brain matters



# EEG Analysis

- Electrodes on patient scalp measure a mixture of different brain activations
- Finding independent activation sources helps removing artifacts in the signal



# Problem Model

Case: $n = 2$

- Observed random variables: $x_1, x_2$
- Independent sources: $s_1, s_2 \in \mathbb{R}$

$$x_1 = a_{11}s_1 + a_{12}s_2$$
$$x_2 = a_{21}s_1 + a_{22}s_2$$

$A$ is called the **mixing matrix**

$$x = As$$

## The blind source separation (cocktail party) problem

Given repeated observation $\{x^{(i)}; i = 1, \ldots, m\}$, recover sources $s^{(i)}$ that generated the data ($x^{(i)} = As^{(i)}$)

# Independent Component Analysis (ICA)

## The blind source separation (cocktail party) problem

Given repeated observation $\{x^{(i)}; i = 1, \ldots, m\}$, recover sources $s^{(i)}$ that generated the data ($x^{(i)} = As^{(i)}$)

Let $W = A^{-1}$ be the **unmixing matrix**
Goal of ICA: Find $W$, such that given $x^{(i)}$, the sources can be recovered by $s^{(i)} = Wx^{(i)}$

$$W = \begin{bmatrix} -w_1^T- \\ \vdots \\ -w_n^T- \end{bmatrix}$$

# ICA Ambiguities

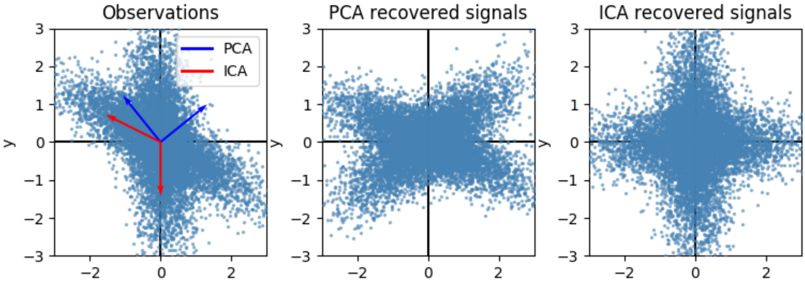
Assume data is **non Gaussian**, ICA has two ambiguities:

- Permutation of original sources $s_1, \ldots, s_n$
- Scaling of $w_i$

*Why is Gaussian data problematic?*

As long as the data is non-Gaussian, given enough data, we can recover the $n$ independent sources.

# ICA vs PCA




| PCA | ICA |
|---|---|
| approximately Gaussian data | non-Gaussian data |
| removes correlation (low order dependence) | removes correlations and higher order dependence |
| ordered importance | all components are equally important |
| orthogonal | not orthogonal |

# Densities and Linear Transformations

Theorem 1

*If random vector $s$ has density $p_s$, and $x = As$ for a square, invertible matrix $A$, then the density of $x$ is*

$$p_x(x) = p_s(Wx)|W|,$$

*where $W = A^{-1}$*

# ICA Algorithm

Joint distributions of *independent* sources $s = \{s_1, \ldots, s_n\}$:

$$p(s) = \prod_{i=1}^{n} p_s(s_i)$$

The density on $x = As = W^{-1}s$:

$$p(x) = \prod_{i=1}^{n} p_s(w_i^T x)|W|$$

Choose the sigmoid function $g(s) = \frac{1}{1+e^{-s}}$ as the *non-Gaussian* cdf for $p_s$, then

$$p_s(s) = g'(s)$$

# ICA Algorithm

Given a training set $\{x^{(1)}, \ldots, x^{(m)}\}$, the log likelihood is

$$l(W) = \sum_{i=1}^{m} \left( \sum_{j=1}^{n} \log g'(w_j{}^T x^{(i)}) + \log |W| \right)$$

Stochastic gradient ascent learning rule for sample $x^{(i)}$:

$$W := W + \alpha \left( \begin{bmatrix} 1 - 2g(w_1{}^T x^{(i)}) \\ \vdots \\ 1 - 2g(w_n{}^T x^{(i)}) \end{bmatrix} x^{(i)\,T} + \left(W^T\right)^{-1} \right)$$

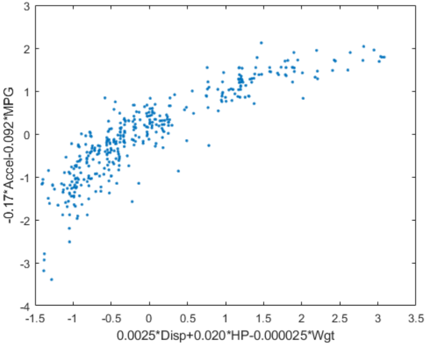
*Check this at home!*

# Canonical Correlation Analysis

**Canonical correlation analysis (CCA)** finds the associations among two sets of variables.

Example: two sets of measurements of 406 cars:

- Specification: Engine displacement (Disp), horsepower (HP), weight (Wgt)
- Measurement: Acceleration (Accel), MPG




find important features that explain covariation between sets of variables

# CCA Definitions

- Random vectors $X = \begin{bmatrix} x_1 \\ \vdots \\ x_{n_1} \end{bmatrix}$ and $Y = \begin{bmatrix} y_1 \\ \vdots \\ y_{n_2} \end{bmatrix}$
- Covariance matrix $\Sigma_{XY} = cov(X, Y)$
- CCA finds vectors $a$ and $b$ such that the random variables $a^T X$ and $b^T Y$ maximize the correlation

$$\rho = corr(a^T X, b^T Y)$$

- $U = a^T X$ and $V = b^T Y$ are called **the first pair of canonical variables**
- Subsequent pairs of canonical variables maximizes $\rho$ while being *uncorrelated* with all previous pairs

# Review: Singular Value Decomposition

A generalization of eigenvalue decomposition to rectangle ($m \times n$) matrices $M$.

$$M = U \Sigma V^T = \sum_{i=1}^{r} \sigma_i u_i v_i^T$$

- $U \in \mathbb{R}^{m \times m}$, $V \in \mathbb{R}^{n \times n}$ are orthogonal matrices
- $\Sigma \in \mathbb{R}^{m \times n}$ is a **rectangular diagonal matrix**.
  Examples:

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \\ 0 & 0 & 0 \end{bmatrix} \quad \Sigma = \begin{bmatrix} \sigma_1 & 0 & 0 & 0 \\ 0 & \sigma_2 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 \end{bmatrix}$$

Diagonal entries $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_k$, $k = \min(n, m)$ are called **singular values of** $M$.

# Review: Singular Value Decomposition

A non-negative real number $\sigma$ is a singular value for $M \in \mathbb{R}^{m \times n}$ if and only if there exist unit-length $u \in \mathbb{R}^m$ and $v \in \mathbb{R}^n$ such that

$$Mv = \sigma u$$
$$M^T u = \sigma v$$

$u$ is called the **left singular vector** of $\sigma$, $v$ is called the **right singular vector** of $\sigma$

## Connection to eigenvalue decomposition

Given SVD of matrix $M = U\Sigma V^T$,

- $M^T M = (V\Sigma^T U^T)(U\Sigma V^T) = V(\Sigma^T\Sigma)V^T$ ← *$v_i$ is an eigenvector of $M^T M$ with eigenvalue $\sigma_i^2$*
- $MM^T = (U\Sigma V^T)(V^T\Sigma^T U) = U(\Sigma\Sigma^T)U^T$ ← *$u_i$ is an eigenvector of $MM^T$ with eigenvalue $\sigma_i^2$*

# CCA Derivations

The original problem:

$$(a_1, b_1) = \underset{a \in \mathbb{R}^{n_1}, b \in R^{n_2}}{\operatorname{argmax}} \; corr(a^T X, b^T Y) \tag{1}$$

Assume $\mathbb{E}[x_1] = \ldots = \mathbb{E}[x_{n_1}] = \mathbb{E}[y_1] = \ldots = \mathbb{E}[y_{n_2}] = 0$,

$$\begin{aligned} corr(a^T X, b^T X) &= \frac{\mathbb{E}[(a^T X)(b^T Y)]}{\sqrt{\mathbb{E}[(a^T X)^2]\mathbb{E}[(a^T Y)^2]}} \\ &= \frac{a^T \Sigma_{XY} b}{\sqrt{a^T \Sigma_{XX} a}\sqrt{b^T \Sigma_{YY} b}} \end{aligned}$$

(1) is equivalent to:

$$(a_1, b_1) = \underset{\substack{a \in \mathbb{R}^{n_1}, b \in R^{n_2} \\ a^T \Sigma_{XX} a = b^T \Sigma_{YY} b = 1}}{\operatorname{argmax}} \; a^T \Sigma_{XY} b \tag{2}$$

# CCA Derivations

Define $\Omega \in R^{n_1 \times n_2}$, $c \in \mathbb{R}^{n_1}$ and $d \in \mathbb{R}^{n_2}$,

$$\Omega = \Sigma_{XX}^{-\frac{1}{2}} \Sigma_{XY} \Sigma_{YY}^{-\frac{1}{2}}$$
$$c = \Sigma_{XX}^{\frac{1}{2}} a$$
$$d = \Sigma_{YY}^{\frac{1}{2}} b$$

(2) can be written as

$$(c_1, d_1) = \underset{\substack{c \in \mathbb{R}^{n_1}, d \in \mathbb{R}^{n_2} \\ ||c||^2 = ||d||^2 = 1}}{\text{argmax}} \quad c^T \Omega d \tag{3}$$

$(c_1, d_1)$ can be solved by SVD, then the first pair of canonical variables are

$$a_1 = \Sigma_{XX}^{-\frac{1}{2}} c_1, \quad b_1 = \Sigma_{YY}^{-\frac{1}{2}} d_1$$

# CCA Derivations

$$(c_1, d_1) = \underset{\substack{c \in \mathbb{R}^{n_1}, d \in \mathbb{R}^{n_2} \\ ||c||^2 = ||d||^2 = 1}}{\text{argmax}} \quad c^T \Omega d$$

### Proposition 1

*$c_1$ and $d_1$ are the left and right unit singular vectors of $\Omega$ with the largest singular value.*

### Theorem 2

*$c_i$ and $d_i$ are the left and right unit singular vectors of $\Omega$ with the $i$th largest singular value.*

# CCA Algorithm

**Input:** Covariance matrices for centered data $X$ and $Y$:

- $\Sigma_{XY}$ , invertible $\Sigma_{XX}$ and $\Sigma_{YY}$
- Dimension $k \leq \min(n_1, n_2)$

**Output:** CCA projection matrices $A_k$ and $B_k$:

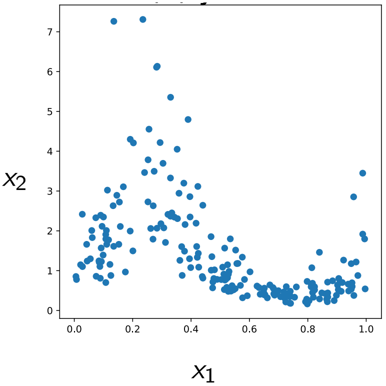
- Compute $\Omega = \Sigma_{XX}^{-\frac{1}{2}} \Sigma_{XY} \Sigma_{YY}^{-\frac{1}{2}}$
- Compute SVD decomposition of $\Omega$

$$\Omega = \begin{bmatrix} | & \dots & | \\ c_1 & \dots & c_{n_1} \\ | & \dots & | \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_r \\ & 0 & \end{bmatrix} \begin{bmatrix} -d_1^T- \\ \vdots \\ -d_{n_2}^T- \end{bmatrix}$$

- $A_k = \Sigma_{XX}^{-\frac{1}{2}}[c_1, \dots, c_k]$ and $B_k = \Sigma_{YY}^{-\frac{1}{2}}[d_1, \dots, d_k]$

# Discussion of CCA

- CCA only measures linear dependencies
- Non-linear generalizations:
  - Kernel CCA (KCCA)
  - Deep CCA (DCCA)
  - Maximal HGR Correlation




Non-linear dependency between $x_1$ and $x_2$