Kernel PCA and ICA

Oleg Ivanov

Linear PCA

Given a set of centered observations i.e. $\sum_{k=1}^{M} x_k = 0$

Diagonalization of covariance matrix *C*

$$C = \frac{1}{M}\sum_{j=1}^{M} x_j x_j^T \qquad (1)$$

via the solution of eigenvalue equation

$$\lambda v = Cv \text{ where } \lambda \geq 0 \text{ and } v \in R^N \setminus \{0\} \qquad (2)$$

therefore

$$Cv = \frac{1}{M}\sum_{j=1}^{M} x_j x_j^T v$$

Linear PCA

since $(xx^T)v = (x \cdot v)x$ derivation (?)

A dot product formulation

$$Cv = \frac{1}{M}\sum_{j=1}^{M}(x_j \cdot v)x_j = \lambda v \qquad (x_j \cdot v) \text{ - scalar}$$

Therefore all solutions v lie in the span of $(x_1 \dots x_M)$

And eigenvalue equation for each data point:

$$\lambda(x_k \cdot v) = (x_k \cdot Cv) \qquad \forall k = 1, \dots, M$$

Non-Linear PCA

F – feature space, related to the input space by a non-linear map Φ

$$\Phi : R^N \to F$$

Given a set of centered observations i.e. $\sum_{k=1}^{M}\Phi(x_k) = 0$

Covariance matrix *C* in *F*

$$\overline{C} = \frac{1}{M}\sum_{j=1}^{M}\Phi(x_j)\Phi(x_j)^T$$

Non-Linear PCA

Again we have to solve the eigenvalue equation

$\lambda V = \bar{C}V$ where $\lambda \geq 0$ and $V \in F \setminus \{0\}$

By the analogy with the linear PCA the solutions *V* lie in the span of non-linear input mappings

$$(\Phi(x_1) \ldots \Phi(x_M))$$

Non-Linear PCA

Eigenvalue equation for each data point:

$$\lambda(\Phi(x_k) \cdot V) = (\Phi(x_k) \cdot \bar{C}V) \quad (3)$$

$\forall k = 1, \ldots, M$

Since *V* are linearly related to inputs $\Phi(x_k)$ we can define coefficients α

$$V = \sum_{i=1}^{M} \alpha_i \Phi(x_i) \quad (4)$$

Non-Linear PCA

Combining (3), (4) and defining matrix *K* (macro step !):

$$K_{ij} = (\Phi(x_i) \cdot \Phi(x_j))$$

We get:

$$M\lambda\boldsymbol{\alpha} = K\boldsymbol{\alpha} \qquad (4)$$

K is a positive semidefinite → diagonalize it to get the solutions for the equation (4)

Non-Linear PCA

$\boldsymbol{\alpha}'s$ should be normalized

If λ_p is the first eigenvalue>0 then the normalized vectors should satisfy:

$$(V^k \cdot V^k) = 1 \quad \forall k = p, \dots, M$$

Using equation (3) $V = \sum_{i=1}^{M} \alpha_i \Phi(x_i)$

Normalization condition for $\boldsymbol{\alpha}^p, \dots, \boldsymbol{\alpha}^M$

$$\sum_{i,j=1}^{M} \alpha_i^k \alpha_j^k (\Phi(x_i) \cdot \Phi(x_j)) = 1$$

Non-Linear PCA

$$\sum_{i,j=1}^{M} \alpha_i^k \alpha_j^k (\Phi(x_i) \cdot \Phi(x_j)) = 1 \qquad k = p, \dots, M$$

$$\sum_{i,j=1}^{M} \alpha_i^k \alpha_j^k K_{ij} = 1$$

$$(\boldsymbol{\alpha}^k \cdot K \boldsymbol{\alpha}^k) = 1$$

$\lambda_k (\boldsymbol{\alpha}^k \cdot \boldsymbol{\alpha}^k) = 1$ the normalization condition

Non-Linear PCA

Extracting non-linear principal components

Let x be a test point, $\Phi(x)$ is image of x in *F*

$$(V^k \cdot \Phi(x)) = \sum_{i=1}^{M} \alpha_i^k (\Phi(x_i) \cdot \Phi(x))$$

Non-Linear PCA

Three steps of non-linear PCA:

1. Compute the dot product matrix *K*

   $$K_{ij} = (\Phi(x_i)\cdot\Phi(x_j))$$

2. Compute Eigenvectors of *K* and normalize them in *F*

   $$\lambda_k(\boldsymbol{\alpha}^k\cdot\boldsymbol{\alpha}^k) = 1$$

3. Compute projections of a test point onto the Eigenvectors

   $$PC_k(x) = (V^k\cdot\Phi(x)) = \sum_{i=1}^{M}\alpha_i^k(\Phi(x_i)\cdot\Phi(x))$$

Non-Linear

BIG PROBLEM (!) makes non-linear PCA almost impracticable

Problem:

Φ can be a map into high-dimensional space *F*

Example: polynomial map of degree 5 of an image 16X16 would yield dimensionality 10^{10}
Computing dot product in such a space would be too expensive (!)

Kernel PCA

Solution:

kernel PCA where dot products can be represented using the kernel function

$$k(x, y) = (\Phi(x) \cdot \Phi(y))$$

This allows to compute $(\Phi(x) \cdot \Phi(y))$ without explicitly mapping x into *F*

Kernel function

How does a kernel work:

Suppose $\Phi(x)$ is a quadratic basis function and x is the input vector of dimensionality d then the full quadratic expansion is…

$$
\Phi(x) = \begin{bmatrix} 1 \\ \sqrt{2}\,x_1 \\ \sqrt{2}\,x_2 \\ \vdots \\ \sqrt{2}\,x_d \\ x_1^2 \\ x_2^2 \\ \vdots \\ x_d^2 \\ \sqrt{2}\,x_1 x_2 \\ \sqrt{2}\,x_1 x_3 \\ \vdots \\ \sqrt{2}\,x_1 x_d \\ \sqrt{2}\,x_2 x_3 \\ \vdots \\ \sqrt{2}\,x_2 x_d \\ \vdots \\ \sqrt{2}\,x_{d-1} x_d \end{bmatrix}
$$

- constant term
- linear terms
- pure quadratic terms
- quadratic cross-terms

Number of terms =(d+2)(d+1)/2

$$
\Phi(x) \bullet \Phi(y) = \begin{bmatrix} 1 \\ \sqrt{2}\,x_1 \\ \sqrt{2}\,x_2 \\ \vdots \\ \sqrt{2}\,x_d \\ x_1^2 \\ x_2^2 \\ \vdots \\ x_d^2 \\ \sqrt{2}\,x_1 x_2 \\ \sqrt{2}\,x_1 x_3 \\ \vdots \\ \sqrt{2}\,x_1 x_d \\ \sqrt{2}\,x_2 x_3 \\ \vdots \\ \sqrt{2}\,x_2 x_d \\ \vdots \\ \sqrt{2}\,x_{d-1} x_d \end{bmatrix} \bullet \begin{bmatrix} 1 \\ \sqrt{2}\,y_1 \\ \sqrt{2}\,y_2 \\ \vdots \\ \sqrt{2}\,y_d \\ y_1^2 \\ y_2^2 \\ \vdots \\ y_d^2 \\ \sqrt{2}\,y_1 y_2 \\ \sqrt{2}\,y_1 y_3 \\ \vdots \\ \sqrt{2}\,y_1 y_d \\ \sqrt{2}\,y_2 y_3 \\ \vdots \\ \sqrt{2}\,x_2 x_d \\ \vdots \\ \sqrt{2}\,y_{d-1} y_d \end{bmatrix}
$$

$$
\mathbf{1} + \sum_{i=1}^{d} 2x_i y_i + \sum_{i=1}^{d} x_i^2 y_i^2 + \sum_{i=1}^{d} \sum_{j=i+1}^{d} 2x_i x_j y_i y_j
$$

So…

Kernel function

$$\Phi(x)\cdot\Phi(y) = 1 + \sum_{i=1}^{d} 2x_i y_i + \sum_{i=1}^{d} x_i^2 y_i^2 + \sum_{i=1}^{d}\sum_{j=i+1}^{d} 2x_i x_j y_i y_j$$

Let's consider another function of *x* and *y:*

$$(x\cdot y+1)^2$$

Kernel function

$$(x\cdot y+1)^2 =$$
$$(x\cdot y)^2 + 2x\cdot y + 1 =$$
$$\left(\sum_{i=1}^{d} x_i y_i\right)^2 + 2\sum_{i=1}^{d} x_i y_i + 1 =$$
$$\sum_{i=1}^{d}\sum_{j=1}^{d} x_i y_i x_j y_j + 2\sum_{i=1}^{d} x_i y_i + 1 =$$
$$\sum_{i=1}^{d}(x_i y_i)^2 + 2\sum_{i=1}^{d}\sum_{j=i+1}^{d} x_i y_i x_j y_j + 2\sum_{i=1}^{d} x_i y_i + 1 = \Phi(x)\cdot\Phi(y)(!)$$

Kernel function

Definition of a kernel function:

$$k(x, y) = (\Phi(x) \cdot \Phi(y))$$

So $(x \cdot y + 1)^2$ is the kernel function of x and y given $\Phi()$ is the mapping function into quadratic feature space *F*

Polynomial kernel function of degree *p:*

$$(x \cdot y + 1)^p$$

Kernel function

Polynomial kernel functions and number of terms

Polynomial	# terms $\Phi(x)$	# terms d=100	Kernel	# terms kernel	# terms d=100
Quadratic	$d^2/2$	5000	$(x \cdot y+1)^2$	*d*/2	50
Cubic	$d^3/6$	166,000	$(x \cdot y+1)^3$	*d*/2	50
Quartic	$d^4/24$	~4,000,000	$(x \cdot y+1)^4$	*d*/2	50

Other kernel functions

Radial basis

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

Neural Network type

$$k(x, y) = \tanh((x \cdot y) + b)$$

Kernel PCA

Three steps of kernel PCA:

1. Compute the dot product matrix *K* using kernel function

   $$K_{ij} = (k(x_i, x_j))_{ij}$$

1. Compute Eigenvectors of *K* and normalize them

   $$\lambda_k (\boldsymbol{\alpha}^k \cdot \boldsymbol{\alpha}^k) = 1$$

2. Compute projections of a test point onto the Eigenvectors using kernel function

   $$kPC_k(x) = (V^k \cdot \Phi(x)) = \sum_{i=1}^{M} \alpha_i^k k(x_i, x)$$

Kernel PCA

Properties of kernel PCA:

1. Remains the orthogonal basis transformation in feature space *F*
2. The first Q PCs carry more variance than any other Q PCs
3. Min MSE of reconstruction
4. PCs are uncorrelated
5. The representation entropy is minimized
6. #PCs can exceed *d (input dimensions)*

ICA

Independent component analysis (ICA) decomposes the multivariate data $y \in R^N$ into a linear sum of statistically independent components:

$$y = \sum_{i=1}^{N} x_i a_i = Ax$$

where x_i is the basis coefficient (source) and a_i is the basis vector

The task is to estimate parameters *A* from data

ICA

The estimation of the data model of independent component analysis is usually performed by formulating an objective function (contrast function) and then minimizing or maximizing it.

Therefore:

ICA = Contrast Function + Optimization

F-correlation

In the paper by Bach and Jordan (2001) “Kernel ICA”, a new contrast function, based on *F*-correlation, was developed

This new function is based on the non-linear function space not on just one function

F-correlation

F-correlation – measures dependence between $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ using correlation of functions of the variables $\boldsymbol{f_1(x_1)}$ and $\boldsymbol{f_2(x_2)}$ for $\boldsymbol{f_1}$ and $\boldsymbol{f_2}$ belonging to some space $\boldsymbol{F}$

$$\rho_F = \max_{f_1, f_2 \in F} corr\,(f_1(x_1), f_2(x_2))$$

$$\rho_F = \max_{f_1, f_2 \in F} \frac{\mathrm{cov}(f_1(x_1), f_2(x_2))}{(\mathrm{var}\ f_1(x_1))^{1/2}(\mathrm{var}\ f_2(x_2))^{1/2}}$$

F-correlation

If $\rho_F = 0$ and *F* is large enough then $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ are independent
Large enough?
If *F* contains the Fourier basis i.e. all functions of the form:

$$x \mapsto e^{i\omega x}$$

where $\omega \in R$

How to make this tractable?

“kernelize” *F*-correlation

“kernelized” *F*-correlation is equivalent to canonical correlation

Canonical Correlation

Given two multivariate random variables $x_1 \in R^{N_1}$ and $x_2 \in R^{N_2}$

CCA finds the pair of directions w_2 and w_1 with maximum correlation.

$$\rho(x_1, x_2) = \max_{w_1, w_2} corr(w_1^T x_1, w_2^T x_2)$$

$$= \max_{w_1, w_2} \frac{w_1^T C_{12} w_2}{(w_1^T C_{11} w_1)^{\frac{1}{2}} (w_2^T C_{22} w_2)^{\frac{1}{2}}}$$

Where $C = \begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix}$ is the covariance matrix of (x_1, x_2)

Canonical Correlation

The CCA reduces to the generalized eigenvalue problem:

$$\begin{pmatrix} 0 & C_{12} \\ C_{21} & 0 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = \rho \begin{pmatrix} C_{11} & 0 \\ 0 & C_{22} \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$$

This problem has N_1+N_2 generalized eigenvalues ρ

"Kernelized" *F*-correlation is equivalent to canonical correlation

RKHS

RKHS – reproducing kernel Hilbert spaces

Let $K(x,y)$ be a Mercer kernel on $X = R^p$ i.e. a function for which the Gram matrix

$$K_{ij} = K(x_i, x_j)$$

is positive definite for any collection $\{x_i\}_{i=1,\ldots,N}$ in X

Corresponding to any such kernel K there is a map Φ from X to a feature space F such that

$$K(x,y) = \langle \Phi(x), \Phi(y) \rangle$$

RKHS

RKHS is the instantiation of F with the following property called "reproducing property"

$$f(x) = \langle K(\cdot,x), f \rangle$$
$$\forall f \in F$$

If $\Phi(x) = K(\cdot,x)$ is a map from the input space into the RKHS then

$$\langle \Phi(x), \Phi(y) \rangle = \langle K(\cdot,x), K(\cdot,y) \rangle = K(x,y)$$

Kernel

Isotropic Gaussian kernel – Mercer kernel with the feature space *F* the space of smooth functions

$$K(x, y) = \exp(-\frac{1}{2\sigma^2} \| x - y \|^2)$$

Theorem 1

Theorem 1
Let x_1 and x_2 be random variables in $X = R^p$ Let K_1 and K_2 be Mercer kernels with feature maps Φ_1 and Φ_2 and feature spaces $F_1, F_2 \in R^X$
Then the canonical correlation between $\Phi_1(x_1)$ and $\Phi_2(x_2)$ which is defined as

$$\rho_F = \max_{(f_1, f_2) \in F_1 x F_2} corr(\langle \Phi_1(x_1), f_1 \rangle, \langle \Phi_2(x_2), f_2 \rangle)$$

is equal

$$\rho_F = \max_{(f_1, f_2) \in F_1 x F_2} corr(f_1(x_1), f_2(x_2))$$

Theorem 2

Theorem 2

(Independence and F-correlation)

If F is the RKHS corresponding to Gaussian kernel $\rho_F = 0$

iff x_1 and x_2 are independent

Kernelized CCA

$$\rho(x_1, x_2) = \max_{w_1, w_2 \in R^N} \frac{w_1^T K_1 K_2 w_2}{(w_1^T K_1^2 w_1)^{\frac{1}{2}} (w_1^T K_2^2 w_1)^{\frac{1}{2}}}$$

Where K_1 and K_2 are Gram matrices of x_1 and x_2

This is equivalent to performing CCA on two vectors with covariance matrix

$$\begin{pmatrix} K_1^2 & K_1 K_2 \\ K_2 K_1 & K_2^2 \end{pmatrix}$$

Kernelized CCA

The kernelized CCA reduces to the generalized eigenvalue problem:

$$\begin{pmatrix} 0 & K_1 K_2 \\ K_2 K_1 & 0 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = \rho \begin{pmatrix} K_1^2 & 0 \\ 0 & K_2^2 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$$

Kernel ICA algorithm

Input:

Data $y^1, y^2, \ldots y^N$

Kernel K(x,y)

1. Whiten the data
2. Compute Gram matrices $K_1, K_2, \ldots, K_m$ of the estimated sources $\{x_1, x_2, \ldots, x_N\}$, where $x_i = W y_i$ (Cholesky decomposition)
3. Define $\lambda_F(K_1, \ldots, K_m)$ as the first eigenvalue of the generalized eigenvector equation $K\alpha = \lambda D\alpha$
4. Minimize $M_{\lambda_F}(K_1, \ldots, K_m) = \frac{1}{2} \log \lambda_F(K_1, \ldots, K_m)$ with respect to W (Stiefel manifold)

Output: W