Introduction to component analysis

Presenter: Zhe Wang

https://qdata.github.io/deep2Read

Zhe Wang

201909

Content

1. Linear PCA
2. Linear ICA
3. Nonilnear ICA
4. SVD and EIG in graph spectral analysis
5. SVD and EIG in reconstruction

Independent VS Uncorrelated

Independent VS uncorrelated:

Definition: Given two variables X and Y:

- If $E(X_1, X_2) = E(X_1)E(X_2)$, then X_1, X_2 are uncorrelated.
- If $P(X_1, X_2) = P(X_1)P(X_2)$, then X_1, X_2 are independent.

Properties: If X_1, X_2 are independent, X_1, X_2 are uncorrelated. If X_1, X_2 are all normal distributions, X_1, X_2 are uncorrelated iff X_1, X_2 are independent.

PCA and ICA

PCA and ICA are all inverse problem, and can be unified as:

$$X = f(Z),$$

where X is called observation.

The task is to recover Z given X.

Linear case: $X = AZ$, where $A \in R^{m \times p}$ is called sampling matrix, $X \in R^{m \times n}$ and $Z \in R^{p \times n}$

PCA assumes $\{z_i\}_{i=1}^{p}$ are uncorrelated to each other, ICA assumes $\{z_i\}_{i=1}^{p}$ are independent to each other.

For Gaussian variables, ICA and PCA are equivalent.

PCA

Goal: find some directions that data have largest variance.

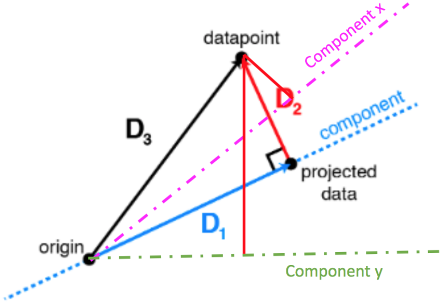
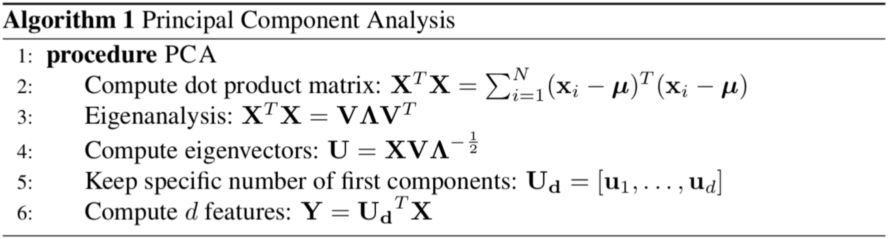


Figure: PCA[1]

[1] from https://bit.ly/2Z1pyAb

Pseudo Code for PCA

The pseudocode for computing PCA



Algorithm 1 Principal Component Analysis

```
1: procedure PCA
2:     Compute dot product matrix: $\mathbf{X}^T\mathbf{X} = \sum_{i=1}^{N} (\mathbf{x}_i - \boldsymbol{\mu})^T (\mathbf{x}_i - \boldsymbol{\mu})$
3:     Eigenanalysis: $\mathbf{X}^T\mathbf{X} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^T$
4:     Compute eigenvectors: $\mathbf{U} = \mathbf{X}\mathbf{V}\boldsymbol{\Lambda}^{-\frac{1}{2}}$
5:     Keep specific number of first components: $\mathbf{U_d} = [\mathbf{u}_1, \ldots, \mathbf{u}_d]$
6:     Compute $d$ features: $\mathbf{Y} = \mathbf{U_d}^T\mathbf{X}$
```

Content

1 Linear PCA

2 Linear ICA

3 Nonilnear ICA

4 SVD and EIG in graph spectral analysis

5 SVD and EIG in reconstruction

ICA example

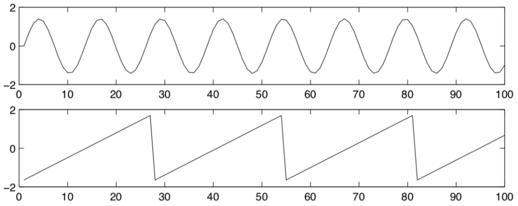


Figure 1: The original signals.

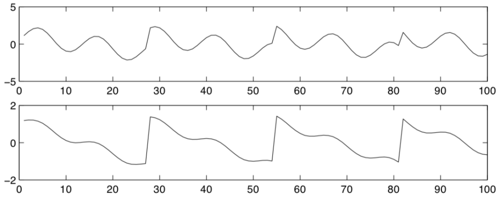


Figure 2: The observed mixtures of the source signals in Fig. 1.

Identifiability of the ICA model

Theorem

Suppose $X = AZ$, variables of Z are mutually independent, and at most one of them is following the gaussian distribution, then Z can be recovered up to an equivalent class of permutation and scaling.

In other words, if $\hat{Z} = BX$, and variables of $\hat{Z}$ are mutually independent, then $\hat{z}_i = s_i * z_j$.

Pseudo Code for ICA

FastICA algorithm

- Firstly, prewhitening the data, suppose data is X, and we use PCA to project X on the new coordinate, such that after projection $X^T X = I$.
- We want to find a direction w that can maximize the non-Gaussianity of the projection $w^T X$. The measure of the non-gaussianity of data will rely on some nonlinear function g
- $w^+ = E(Xg(w^T X)^T) - E(g'(w^T X))w$
- $w = w^+ / ||w||$
- some choices for g, $g(u) = ue^{-u^2/2}$, $g(u) = \tanh u$

Content

1 Linear PCA

2 Linear ICA

3 Nonilnear ICA

4 SVD and EIG in graph spectral analysis

5 SVD and EIG in reconstruction

Nonlinear ICA and VAE

Suppose $P(X, Z) = P_\theta(X|Z)P_\theta(Z)$, where $P_\theta(Z)$ is the prior distribution over latent variable.
The distribution $P_\theta(X|Z)$, often parameterized as a neural network, is called a decoder.
Observed distribution of data:

$$P(X) = \int_Z P(X, Z)dZ$$

We then collect a dataset of observations of X:

$$D = \{x^1, x^2, \cdots, x^N\},$$

where $z^i \sim P_\theta(Z)$ and $x^i \sim P_\theta(X|z^i)$.

VAE and nonlinear ICA[2]

VAE models learn the full data generation process $P_\theta(Z), P_\theta(X|Z), P_\theta(Z|X)$. All we know is $P_\theta(X) = P_{\theta*}(X)$, we don't know what the remaining are.

Goal: under what condition: $P_\theta(X) = P_{\theta*}(X)$ can guarantee $P_\theta(Z|X) = P_{\theta*}(Z), P_\theta(X|Z) = P_{\theta*}(X|Z)$

Main Assumption: A conditionally factorized prior distribution over the latent variables $p_\theta(z|u)$, where u is an additionally observed variable
And the data generation stage is a additive noise model $X = f(Z) + \epsilon$

$P(Z|U)$ is conditionally factorial

$$P(Z|U) = \Pi_{i=1}^n P(Z_i|U),$$

we can always assume $P(Z_i|U)$ is sampled from exponential family.

[2]Ilyes Khemakhem, et.al, AISTATS2020

$$p_{\mathbf{T},\boldsymbol{\lambda}}(\mathbf{z}|\mathbf{u}) = \prod_i \frac{Q_i(z_i)}{Z_i(\mathbf{u})} \exp\left[\sum_{j=1}^{k} T_{i,j}(z_i)\lambda_{i,j}(\mathbf{u})\right]$$

Under some mild differentiable assumptions

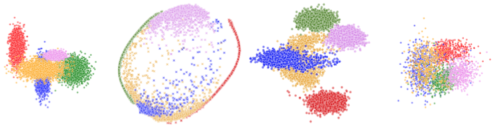
$$P_\theta(X) = P_{\theta*}(X) \rightarrow (f, T, \lambda) \triangleq (f^*, T^*, \lambda^*).$$

If this is true, then it will be really powerful. It can show us the real data generating process.

Connection to ICA, the joint distribution of latent variables are factorial, which implies their mutual independence. Thus if f is linear then, this process is linear ICA, and completeley identifiable.

Experiments

Experiments on 2D dataset.


(a) $p_{\boldsymbol{\theta}^*}(\mathbf{z}|\mathbf{u})$ (b) $p_{\boldsymbol{\theta}^*}(\mathbf{x}|\mathbf{u})$ (c) $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x},\mathbf{u})$ (d) $p_{\text{VAE}}(\mathbf{z}|\mathbf{x})$

Causal analysis

Consider data $x = (x_1, x_2)$. The goal is to establish if the causal direction is $x_1 \rightarrow x_2$, or $x_2 \rightarrow x_1$.
Assume the data generation process is $x1 = f1(n1), x2 = f2(x1, n2)$ where $f = (f1, f2)$ is a (possibly nonlinear) mapping. Then, we can recover the distribution of n_1, n_2. Then we can perform independence analysis to find the causal direction.

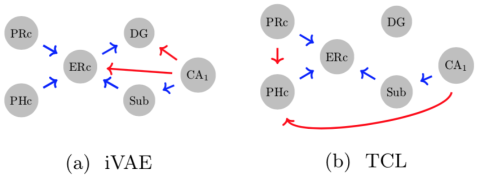


Figure 4: Estimated causal graph on hippocampal fMRI data unmixing of sources is achieved via iVAE (left) or TCL (right). Blue edges are feasible given anatomical connectivity, red edges are not.

Summary:
Given an additional observable variable u, such as class labels and time indices for times series data, we can discover the true latent distributions $p(z)$, data generating process $f(x|z)$ and the inverse.

Limitations:
The dimension of latent variables is required as a prior knowledge.

Some recent work: Some recent results show if the true latent distribution $p(z)$ is multi-Gaussian, their encoder can better reconstruct the latent distribution and discover the dimension of hidden space automatically.

Content

1 Linear PCA

2 Linear ICA

3 Nonilnear ICA

4 SVD and EIG in graph spectral analysis

5 SVD and EIG in reconstruction

Matrix Factorization

Singular Value Decomposition:

$$X = U\Sigma V^T,$$

where U contains the eigenvectors of XX^T, and V contains the eigenvectors of X^TX, Σ is a diagonal matrix of the singular values. If X is a real symmetric matrix, then:

$$X = U\Sigma U^T,$$

which is also called Eigenvalue Decomposition.
The i-th column of the matrix U is called the i-th principle component.
The first few principle components contains most information about X

Both SVD and EIG are of great importance in spectral analysis or image processing like segmentation and denoising.

In spectral analysis, the EIG is a powerful tool:
Given a graph $G = (V, E)$, the laplace matrix is defined as $\Delta = D - E$. Suppose:

$$\Delta = \Phi \Lambda \Phi^T$$

Given a signal $f = (f_1, f_2, \cdots f_n)^T$, then the graph fourier transformation is defined as:

$$\hat{f} = \Phi^T f$$

For two signals f, g on the graph, the graph convolution is defined as:

$$f * g = \Phi[\Phi^T f \odot \Phi^T g] = \Phi Diag(\hat{g}_1, \hat{g}_2, \hat{g}_n)\hat{f}$$

ChebNet

$$f * g = \Phi[\Phi^T g \odot \Phi^T f] = \Phi Diag(\hat{g}_1, \hat{g}_2, \cdots, \hat{g}_n)\hat{f}$$

How to parameterize the operation, so that it can be trained as a layer?
Spectral CNN[3]

$$k * f = \Phi Diag(\theta_1, \theta_2, \cdots, \theta_n)\Phi^T f$$

ChebNet[4]
Since $\Delta = \Phi\Lambda\Phi^T$, the graph convolution kernel is defined as:

$$g_\theta(\Lambda) = \sum_{j=0}^{K-1} \theta_j T_j(\hat{\Lambda})$$

where $\hat{\Lambda} = \frac{2}{\lambda_{max}}\Lambda - I$, $T_j(\hat{\Lambda})$ is chebshev polynomial, and

$$T_0(\hat{\Lambda}) = I, T_1(\hat{\Lambda}) = \hat{\Lambda}, T_{j+1}(\hat{\Lambda}) = 2\hat{\Lambda}T_j(\hat{\Lambda}) - T_{j-1}(\hat{\Lambda})$$

[3] Joan Bruna, et.al, ICLR2014

Now, the parameterized graph convolution can be calculated as:

$$\begin{aligned} k * f &= \Phi \sum_{j=0}^{K-1} \theta_j T_j(\hat{\Lambda}) \Phi^T f = \sum_{j=0}^{K-1} \theta_j \Phi T_j(\hat{\Lambda}) \Phi^T f \\ &= \sum_{j=0}^{K-1} \theta_j T_j(\Phi \hat{\Lambda} \Phi^T) f \\ &= \sum_{j=0}^{K-1} \theta_j T_j(\Delta) f \end{aligned}$$

- No need to do SVD
- only $K < n$ parameters to learn

GCN

Later GCN[5] was proposed: In GCN, set $K = 2$, we have:

$$k * f = \theta_0 f + \theta_1 D^{-1/2} A D^{-1/2} f = \theta(I + D^{-1/2} A D^{-1/2}) f$$

In order to avoid gradient explosion, re-normalization trick will be used:

$$I + D^{-1/2} A D^{-1/2} \rightarrow \hat{D}^{-1/2} \hat{A} \hat{D}^{-1/2}$$

where $\hat{A} = A + I$.
Finally, generalize the update formula from vector to matrix, we have:

$$f = \hat{D}^{-1/2} \hat{A} \hat{D}^{-1/2} f \theta$$

[5] Thomas N. Kipf et.al ICLR2017

Content

1 Linear PCA

2 Linear ICA

3 Nonilnear ICA

4 SVD and EIG in graph spectral analysis

5 SVD and EIG in reconstruction

Low rank as prior knowledge

Low rank is an important prior knowledge in image processing:

$$\arg\min_{\hat{X}} ||X - \hat{X}|| + \lambda||\hat{X}||_*$$

A good solution can be found with SVD.
Low rank is also widely used in inverse problems:

$$\arg\min \frac{1}{2}||Y - XW||_F^2 + \lambda||W||_*$$

There are two methods to solve the inverse problem:

- Proximal method

$$W_{k+1} = D_\tau(W_k + \lambda X^T(Y - XW_k))$$

- Alternative optimization method

$$||W||_* = \min_{U,V} \frac{1}{2}||U||_F^2 + \frac{1}{2}||V||_F^2 \quad s.t. UV = W$$

where $W \in R^{m\times n}, U \in R^{m\times k}, V \in U \in R^{k\times n}$.

$$\arg\min \frac{1}{2}||Y - X(UV)||_F^2 + \lambda_1/2||U||_F^2 + \lambda_2/2||V||_F^2$$

Now, for each variable, there is a closed form solution, we can update them alternatively, and the final $W = UV$.