Introduction to Machine Learning
10701
Independent Component Analysis

Barnabás Póczos & Aarti Singh
Independent Component Analysis
Independent Component Analysis

\[ x_1(t) = a_{11}s_1(t) + a_{12}s_2(t) \]
\[ x_2(t) = a_{21}s_1(t) + a_{22}s_2(t) \]

Model

Observations (Mixtures)

ICA estimated signals

original signals
Independent Component Analysis

Model

\[ x_1(t) = a_{11}s_1(t) + a_{12}s_2(t) \]
\[ x_2(t) = a_{21}s_1(t) + a_{22}s_2(t) \]

We observe

\[
\begin{pmatrix}
  x_1(1) \\
  x_2(1)
\end{pmatrix},
\begin{pmatrix}
  x_1(2) \\
  x_2(2)
\end{pmatrix}, \ldots,
\begin{pmatrix}
  x_1(t) \\
  x_2(t)
\end{pmatrix}
\]

We want

\[
\begin{pmatrix}
  s_1(1) \\
  s_2(1)
\end{pmatrix},
\begin{pmatrix}
  s_1(2) \\
  s_2(2)
\end{pmatrix}, \ldots,
\begin{pmatrix}
  s_1(t) \\
  s_2(t)
\end{pmatrix}
\]

But we don’t know \( \{a_{ij}\} \), nor \( \{s_i(t)\} \)

Goal: Estimate \( \{s_i(t)\} \), (and also \( \{a_{ij}\} \))
The Cocktail Party Problem
SOLVING WITH PCA

Sources

Mixing

Observation

PCA Estimation

Sources

Mixing

Observation

PCA Estimation

$s(t)$

$x(t) = As(t)$

$y(t)=Wx(t)$

$A \in \mathbb{R}^{M \times M}$
The Cocktail Party Problem
SOLVING WITH ICA

Sources
Mixing
Observation
ICA Estimation

\[ A \in \mathbb{R}^{M \times M} \]
\[ x(t) = As(t) \]
\[ y(t) = Wx(t) \]
ICA vs PCA, Similarities

- Perform linear transformations
- Matrix factorization

**PCA:** *low rank* matrix factorization for *compression*

\[ \begin{align*}
N \begin{bmatrix} X \end{bmatrix} & = \begin{bmatrix} U \end{bmatrix} \begin{bmatrix} S \end{bmatrix} \\
 & \quad \text{M} < \text{N}
\end{align*} \]

Columns of U = PCA vectors

**ICA:** *full rank* matrix factorization to *remove dependency* among the rows

\[ \begin{align*}
N \begin{bmatrix} X \end{bmatrix} & = \begin{bmatrix} A \end{bmatrix} \begin{bmatrix} S \end{bmatrix} \\
 & \quad \text{N}
\end{align*} \]

Columns of A = ICA vectors
ICA vs PCA, Similarities

- **PCA**: $X = US$, $U^TU = I$
- **ICA**: $X = AS$, $A$ is invertible

- PCA **does** compression
  - $M < N$

- ICA **does not** do compression
  - same # of features ($M = N$)

- PCA just removes correlations, **not** higher order dependence
- ICA removes correlations, **and** higher order dependence

- PCA: some components are **more important** than others (based on eigenvalues)
- ICA: components are **equally important**
Note

- **PCA** vectors are orthogonal
- **ICA** vectors are **not** orthogonal
ICA vs PCA
ICA basis vectors extracted from natural images

Gabor wavelets, edge detection, receptive fields of V1 cells..., deep neural networks
PCA basis vectors extracted from natural images
Some ICA Applications

**STATIC**
- Image denoising
- Microarray data processing
- Decomposing the spectra of galaxies
- Face recognition
- Facial expression recognition
- Feature extraction
- Clustering
- Classification
- Deep Neural Networks

**TEMPORAL**
- Medical signal processing – fMRI, ECG, EEG
- Brain Computer Interfaces
- Modeling of the hippocampus, place cells
- Modeling of the visual cortex
- Time series analysis
- Financial applications
- Blind deconvolution
ICA Application, Removing Artifacts from EEG

- EEG ~ *Neural cocktail party*
- Severe *contamination* of EEG activity by:
  - eye movements
  - blinks
  - muscle
  - heart, ECG artifact
  - vessel pulse
  - electrode noise
  - line noise, alternating current (60 Hz)

- ICA can improve signal
  - effectively *detect, separate and remove* activity in EEG records from a wide variety of artifactual sources.
    (Jung, Makeig, Bell, and Sejnowski)

- ICA weights (mixing matrix) help find *location* of sources
Removing Artifacts from EEG

Summed Projection of Selected Components

C1

C2

C3

C4

mixing $W^{-1}$

Artifact-corrected EEG

$X_0 = W^{-1}u_0$

Fig from Jung
ICA for Image Denoising

- Original
- Noisy
- Wiener filtered
- Median filtered
- ICA denoised
  (Hoyer, Hyvarinen)
Method for analysis and synthesis of human motion from motion captured data

Provides perceptually meaningful “style” components

109 markers, (327dim data)

Motion capture \(\Rightarrow\) data matrix

**Goal:** Find motion style components.

ICA \(\Rightarrow\) 6 independent components (emotion, content, …)

Statistical (in)dependence

**Definition (Independence)**

\[ Y_1, Y_2 \text{ are independent } \Leftrightarrow p(y_1, y_2) = p(y_1)p(y_2) \]

**Definition (Shannon entropy)**

\[ H(Y) = H(Y_1, \ldots, Y_m) = - \int p(y_1, \ldots, y_m) \log p(y_1, \ldots, y_m) dy. \]

**Definition (KL divergence)**

\[ 0 \leq KL(f \| g) = \int f(x) \log \frac{f(x)}{g(x)} dx \]

**Definition (Mutual Information)**

\[ 0 \leq I(Y_1, \ldots, Y_M) = \int p(y_1, \ldots, y_M) \log \frac{p(y_1, \ldots, y_M)}{p(y_1) \cdots p(y_M)} dy \]
Solving the ICA problem with i.i.d. sources

ICA problem: \( x = As, \ s = [s_1; \ldots; s_M] \) are jointly independent.

Ambiguity:
\( s = [s_1; \ldots; s_M] \) sources can be recovered only up to sign, scale and permutation.

Proof:
- \( P = \) arbitrary permutation matrix,
- \( \Lambda = \) arbitrary diagonal scaling matrix.

\[ \Rightarrow x = [AP^{-1}\Lambda^{-1}][\Lambda P s] \]
Solving the ICA problem

**Lemma:**
We can assume that $E[s] = 0$.

**Proof:**
Removing the mean does not change the mixing matrix.

$$x - E[x] = A(s - E[s]).$$

In what follows we assume that $E[ss^T] = I_M$, $E[s] = 0$. 
Whitening

- Let $\Sigma \doteq \text{cov}(x) = E[xx^T] = A E[ss^T] A^T = AA^T$. (We assumed centered data)

- Do **SVD**: $\Sigma \in \mathbb{R}^{N \times N}$, $\text{rank}(\Sigma) = M$,
  $$\Rightarrow \Sigma = U D U^T,$$
  where $U \in \mathbb{R}^{N \times M}$, $U^T U = I_M$, **Signular vectors**
  $D \in \mathbb{R}^{M \times M}$, diagonal with rank $M$. **Singular values**
Whitening

- Let $Q = D^{-1/2}U^T \in \mathbb{R}^{M \times N}$ whitening matrix
- Let $A^* = QA$
- $x^* = Qx = QAs = A^*s$ is our new (whitened) ICA task.

We have,

$$E[x^*x^{*T}] = E[Qxx^TQ^T] = Q\Sigma Q^T = (D^{-1/2}U^T)UDU^T(UD^{-1/2}) = I_M$$

$$\Rightarrow E[x^*x^{*T}] = I_M, \text{ and } A^*A^{*T} = I_M.$$
Whitening solves half of the ICA problem

Note:
The number of free parameters of an $N \times N$ orthogonal matrix is $(N-1)(N-2)/2$.

$\Rightarrow$ whitening solves half of the ICA problem

After whitening it is enough to consider orthogonal matrices for separation.
Solving ICA

ICA task: Given $\mathbf{x}$,
- find $\mathbf{y}$ (the estimation of $\mathbf{s}$),
- find $\mathbf{W}$ (the estimation of $\mathbf{A}^{-1}$)

ICA solution: $\mathbf{y} = \mathbf{Wx}$
- Remove mean, $E[\mathbf{x}] = 0$
- Whitening, $E[\mathbf{xx}^T] = \mathbf{I}$
- Find an orthogonal $\mathbf{W}$ optimizing an objective function
  - Sequence of 2-d Jacobi (Givens) rotations

![Original](image1.png)
![Mixed](image2.png)
![Whitened](image3.png)
![Rotated (demixed)](image4.png)
Optimization Using Jacobi Rotation Matrices

\[ G(p, q, \theta) = \begin{pmatrix} 1 & \ldots & 0 & \ldots & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \ldots & \cos(\theta) & \ldots & -\sin(\theta) & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \ldots & \sin(\theta) & \ldots & \cos(\theta) & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \ldots & 0 & \ldots & 0 & \ldots & 1 \end{pmatrix} \]

\[ \in \mathbb{R}^{M \times M} \]

**Observation**: \( x = As \)

**Estimation**: \( y = Wx \)

\[ W = \arg \min_{\tilde{W} \in \mathcal{W}} J(\tilde{W}x), \quad \tilde{W} \in \mathcal{W} \]

where \( \mathcal{W} = \{ W | W = \prod_{i} G(p_i, q_i, \theta_i) \} \)
ICA Cost Functions

Let \( y = Wx \), \( y = [y_1; \ldots; y_M] \), and let us measure the dependence using Shannon’s mutual information:

\[
J_{ICA_1}(W) \doteq I(y_1, \ldots, y_M) \doteq \int p(y_1, \ldots, y_M) \log \frac{p(y_1, \ldots, y_M)}{p(y_1) \cdots p(y_M)} \, dy,
\]

Let \( H(y) \doteq H(y_1, \ldots, y_m) \doteq -\int p(y_1, \ldots, y_m) \log p(y_1, \ldots, y_m) \, dy \).

**Lemma**

\[
H(Wx) = H(x) + \log |\det W| \quad \text{Proof: Homework}
\]

Therefore,

\[
I(y_1, \ldots, y_M) = \int p(y_1, \ldots, y_M) \log \frac{p(y_1, \ldots, y_M)}{p(y_1) \cdots p(y_M)}
\]

\[
= -H(y_1, \ldots, y_M) + H(y_1) + \ldots + H(y_M)
\]

\[
= -H(x_1, \ldots, x_M) - \log |\det W| + H(y_1) + \ldots + H(y_M).
\]
ICA Cost Functions

\[ I(y_1, \ldots, y_M) = \int p(y_1, \ldots, y_M) \log \frac{p(y_1, \ldots, y_M)}{p(y_1) \cdots p(y_M)} \]
\[ = -H(y_1, \ldots, y_M) + H(y_1) + \ldots + H(y_M) \]
\[ = -H(x_1, \ldots, x_M) - \log |\det W| + H(y_1) + \ldots + H(y_M). \]

\( H(x_1, \ldots, x_M) \) is constant, \( \log |\det W| = 0. \)

Therefore,

\[ J_{ICA_2}(W) \doteq H(y_1) + \ldots + H(y_M) \]

The covariance is fixed: 1. Which distribution has the largest entropy?

\( \Rightarrow \) go away from normal distribution
The sum of independent variables converges to the normal distribution

⇒ For separation go far away from the normal distribution
⇒ Negentropy, |kurtozis| maximization

Figs from Ata Kaban
ICA Algorithms
Maximum Likelihood ICA Algorithm

- simplest approach
- requires knowing densities of hidden sources \( \{f_i\} \)

\[
x(t) = A s(t), \quad s(t) = W x(t), \quad \text{where} \quad A^{-1} = W = [w_1; \ldots; w_M] \in \mathbb{R}^{M \times M}
\]
Maximum Likelihood ICA Algorithm

\[ \Delta W \propto [W^T]^{-1} + \frac{1}{T} \sum_{t=1}^{T} g(Wx(t))x^T(t), \quad \text{where } g_i = \frac{f'_i}{f_i} \]
ICA algorithm based on Kurtosis maximization

Kurtosis = 4\textsuperscript{th} order cumulant

Measures

• the distance from normality
• the degree of peakedness

\[ \kappa_4(y) = E\{y^4\} - \frac{3}{2} \left( E\{y^2\} \right)^2 \]

= 3 if \( E\{y\} = 0 \) and whitened

\( \kappa_4(y) = -\frac{2}{15} \)

\( \kappa_4(y) = 0 \)

\( \kappa_4(y) = 12 \)
The Fast ICA algorithm (Hyvarinen)

- Given whitened data $z$
- Estimate the $1^{st}$ ICA component:

\[
* \ y = w^T z, \ ||w|| = 1, \quad \iff \ w^T = 1^{st} \text{ row of } W
\]

\[
* \text{maximize kurtosis } f(w) = \kappa_4(y) = \mathbb{E}[y^4] - 3
\]

with constraint $h(w) = ||w||^2 - 1 = 0$

\[
* \text{At optimum } f'(w) + \lambda h'(w) = 0^T \quad (\lambda \text{ Lagrange multiplier})
\]

\[
\Rightarrow 4\mathbb{E}[(w^T z)^3 z] + 2\lambda w = 0
\]

Solve this equation by Newton–Raphson’s method.
Newton method for finding a root
Newton Method for Finding a Root

**Goal:** \( \phi : \mathbb{R} \rightarrow \mathbb{R} \)

\[ \phi(x^*) = 0 \]

\[ x^* =? \]

**Linear Approximation (1st order Taylor approx):**

\[ \phi(x + \Delta x) = \phi(x) + \phi'(x)\Delta x + o(|\Delta x|) \]

Therefore,

\[ 0 \approx \phi(x) + \phi'(x)\Delta x \]

\[ x^* - x = \Delta x = -\frac{\phi(x)}{\phi'(x)} \]

\[ x_{k+1} = x_k - \frac{\phi(x)}{\phi'(x)} \]
Illustration of Newton’s method

**Goal:** finding a root

\[ \hat{f}(x) = f(x_0) + f'(x_0)(x - x_0) \]

\[ x = x_0 + \Delta x_{NT} \]

In the next step we will linearize here in \( x \)
Example: Finding a Root

http://en.wikipedia.org/wiki/Newton%27s_method
Newton Method for Finding a Root

This can be generalized to multivariate functions

\[ F : \mathbb{R}^n \to \mathbb{R}^m \]

\[ 0_m = F(x^*) = F(x + \Delta x) = F(x) + \nabla F(x) \Delta x + o(|\Delta x|) \]

Therefore,

\[ 0_m = F(x) + \nabla F(x) \Delta x \]

\[ \Delta x = -[\nabla F(x)]^{-1} F(x) \]

[Pseudo inverse if there is no inverse]

\[ \Delta x = x_{k+1} - x_k, \text{ and thus} \]

\[ x_{k+1} = x_k - [\nabla F(x_k)]^{-1} F(x_k) \]

Newton method: Start from \( x_0 \) and iterate.
Newton method for FastICA
The Fast ICA algorithm (Hyvarinen)

Solve: \[ F(w) = 4\mathbb{E}[(w^T z)^3 z] + 2\lambda w = 0 \]

Note:
\[ y = w^T z, \quad \|w\| = 1, \quad z \text{ white} \Rightarrow \mathbb{E}[(w^T z)^2] = 1 \]

The derivative of \( F \):
\[
F'(w) = 12\mathbb{E}[(w^T z)^2 z z^T] + 2\lambda I \\
\sim 12\mathbb{E}[(w^T z)^2] \mathbb{E}[z z^T] + 2\lambda I \\
= 12\mathbb{E}[(w^T z)^2] I + 2\lambda I \\
= 12I + 2\lambda I
\]
The Jacobian matrix becomes diagonal, and can easily be inverted.

\[ w(k + 1) = w(k) - [F'(w(k))]^{-1} F(w(k)) \]

\[ w(k + 1) = w(k) - \frac{4\mathbb{E}[(w(k)^T z)^3 z] + 2\lambda w(k)}{12 + 2\lambda} \]

\[(12 + 2\lambda)w(k + 1) = (12 + 2\lambda)w(k) - 4\mathbb{E}[(w(k)^T z)^3 z] - 2\lambda w(k) \]

\[-\frac{12 + 2\lambda}{4} w(k + 1) = -3w(k) + \mathbb{E}[(w(k)^T z)^3 z] \]

Therefore,

Let \( w_1 \) be the fix pont of:

\[ \tilde{w}(k + 1) = \mathbb{E}[(w(k)^T z)^3 z] - 3w(k) \]

\[ w(k + 1) = \frac{\tilde{w}(k + 1)}{\|\tilde{w}(k + 1)\|} \]

- Estimate the 2\(^{nd}\) ICA component similarly using the \( w \perp w_1 \) additional constraint... and so on...
Other Nonlinearities
Other Nonlinearities

Newton method:

Algorithm:
Fast ICA for several units