# Independent Component Analysis

Qiuping Xu

Department of Math, FSU

## 1 Introduction

Independent component analysis (ICA) is a statistical procedure of finding additive components with the assumption that the components are non-Gaussian and they are statistically independent. This method has great applications in signal decomposition (EEG signal), feature extraction, noise removal and finding hidden factors in financial data.

## 2 Problem Setup

The famous example to illustrate ICA method is so-called cocktail party. Imagine two people are talking in a party and two different microphones are recording. Then the two records $X_1(t), X_2(t)$ from those microphones are both the mixtures of the speech signal $S_1(t), S_2(t)$ from the two speakers. Let us assume that only additive mixed effect exists, at this time, we can use the following equation 1 to describe the scenario.

$$\begin{aligned} X_1(t) &= a_{11}S_1(t) + a_{12}S_2(t) \\ X_2(t) &= a_{21}S_1(t) + a_{22}S_2(t) \end{aligned} \tag{1}$$

Or we can use matrix from $X = AS$ or we put the unknown on the left as $S = WX$. The aim of ICA is to estimate the independent signal $S_1(t), S_2(t)$ and parameters $a_{ij}$ from the recorded data $X_1(t), X_2(t)$.

## 3 Background

### 3.1 PCA VS ICA

PCA is adequate if the data are Gaussian, linear; if not, ICA is more proper. PCA optimizes the covariance of the data (which is second order statistics); on the other hand, ICA optimizes higher-order statistics. PCA finds the uncorrelated components; while ICA finds the independent components (uncorrelatedness is a weaker form of independence). PCA can give the order of the components; while ICA normally can not (Magnitude and scaling ambiguity, Permutation ambiguity).

### 3.2 Uncorrelatedness VS Independence

Consider two scalar-valued random variables $x_1$ and $x_2$, those two variables are said to be independent if information on the value of $x_1$ does not give any information on the value of $x_2$ , and vice verse.

If we define $p(x_1, x_2)$ as the joint probability function of those two variables and $p_1(x_1)$ and $p_2(x_2)$ as the probability functions for $x_1$ and $x_2$ respectively, Then independence is equivalent to the validity of Eq: 2

$$p(x_1, x_2) = p_1(x_1)p_2(x_2) \tag{2}$$

From Eq:2, we can derive that Eq:3 holds for any "proper" function $f$ and $g$

$$E(f(x_1), g(x_2)) = E(f(x_1))E(g(x_2)) \tag{3}$$

However, uncorrelatedness only requires $cov(x_1, x_2) = E(x_1, x_2) - E(x_1)E(x_2) = 0$. If we rewrite this, we can see uncorrelatedness is equivalent to Eq:4

$$E(x_1, x_2) = E(x_1)E(x_2) \tag{4}$$

Compare Eq: 3 and 4, we can clearly see that uncorrelatedness is just a weaker from of Independence, and uncorrelatedness does **not** imply independence.

## 4 Core of ICA

The core of ICA is independence. I prefer thinking there are two different kinds of approaches of independence. One approach is based on non-gaussianity equals independence, so the algorithm focuses on minimization of some measurement(negentropy and kurtosis) of gaussianity to find the independent components. Another approach states independence can be achieved by minimising mutual information, so the algorithm focuses on minimization of mutual information(free to define different measurement upon applications).

### 4.1 Negentropy

The entropy $H_x$ of a continue variable $x$ with density function $p(x)$ is defined as Eq: 5

$$H_x = \int -p(x) \log p(x) dx \tag{5}$$

Negentropy is defined as Eq: 6

$$J_x = H_{x^*} - H_x \tag{6}$$

Where $x^*$ is the Gaussian variable with the same mean and variance as $x$. Out of all distributions with given mean and variance, the Gaussian distribution is the one with the highest entropy. Thus, negentropy is always nonnegative and zero only for a pure Gaussian signal $x$. Negentropy is often used as a measure of distance to gaussianity. This measure is stable but is difficult to evaluate. In practice, approximation is used to estimate this value.

## 5 Example and Computation

### 5.1 Algorithm

**Centering** This is similar to what we did in PCA, just make the the data into zero-mean data. If you like you can add the mean back to the independent components after completing the ICA process, which is $inv(A) * m$, here $A$ is the coefficient matrix and $m$ is the mean we subtracted from the data.

**Whitening** Another pre-processing step we need to do is to whiten the data, which means we need to transform the signals into uncorrelated signals and each of the signal has unit variance. The whole process is to do PCA first, then make the column of the SCORE matrix into unit variance by dividing the column's own standard deviation(component-wise).

If you are not familiar with PCA, the process can also be done by the following steps. Form the zero-mean data $X$, the whiten data $X^*$ is given by $VD^{-\frac{1}{2}}V^TX$, where $V$ and $D$ are the results from singular value decomposition of $X$ and $X = VDV^T$. **One thing to notice, $D$ is a diagonal matrix, so finding $D^{-\frac{1}{2}}$ only requires a component-wised operation.**

**Update** The direct estimation of entropy is generally difficult. The new estimation methods focus on keeping the maximal entropy principle.

$$J1_x = \{E(G(x^*)) - E(G(x))\}^2 \tag{7}$$

Here $x^*$ is the Gaussian variable with the same mean and variance as x. Or after whitening, $x^*$ is the standard Gaussian variable. $E$ is the expectation operation, in computation, it usually replaced by the sample mean(use whole data set to estimate the mean or just a portion to estimate the mean). $J1_x$ shares similar properties to $J_x$, such as $J1_x$ is also always nonnegative and zero only for a pure Gaussian signal $x$.

Given initial guess $W$, the symmetric version(no preference for a particular component) of update rule is given as

$$\begin{aligned} W^* &= E(Xg(W^TX))E(g^{'}(W^TX))W \\ W_{next} &= (W^*W^{*T})^{-\frac{1}{2}}W^* \end{aligned} \tag{8}$$

Where $S = WX$. To remind again, $X$ is the recorded signal and $S$ is the independent components we want to find. $g$ is the derivative of $G$ and $g^{'}$ is the derivative of $g$. The choice of $g$ is application dependent, but there are a few "good" choices such as $g(x) = x\exp(x^2/2), x^3, tanh(x)$.

### 5.2 Code and Example

There is free version of ICA available online called FastICA. The code are well written in several languages including Matlab, R, C++ and Python.

Next, I will use an example to illustrate how to use the code (Of course, download the code first and unzip it into a folder).

**EX1** We first generate two source signal(as speech signal in the cocktail example) which also services as the ground truth for later comparison. The signals show in Fig.1. In order to have better comparison, I made those signals into unit variance(output IC will have unit variance).

```
sig1=0.5*sin([1:0.05:30]);
sig1=sig1./std(sig1);
sig2=rand(size(sig1))-0.5;
sig2=sig2./std(sig2);
figure; % visualize those 2 source signals
subplot(2,1,1)
plot(sig1)
subplot(2,1,2)
plot(sig2)
```

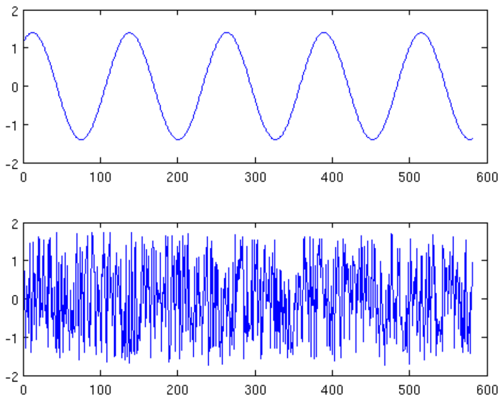

**Fig. 1.** Two source signals

From those two source signals, we can generate mixed signals(as the recording in cocktail example). The mixed signals are showing in Fig.2. And the histograms of those mixed signals are plotted in Fig.3. We see bell shaped histogram from mixed signals, which it is a indication of gaussianity.

```
A=[1,5;0.3,2];
mixedsig=A*[sig1;sig2]; % generate 2 mixed signal
%with 581 observations each
figure; % visualize mixed signals
for i=1:2
subplot(2,1,i)
plot(mixedsig(i,:))
```

```
end
figure
for i=1:2
subplot(2,1,i)
hist(mixedsig(i,:))
end
```

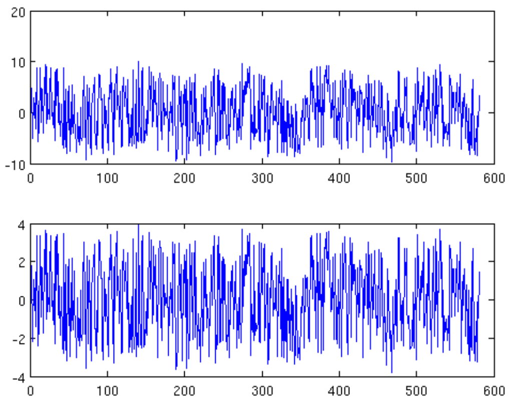

**Fig. 2.** Two observed signals

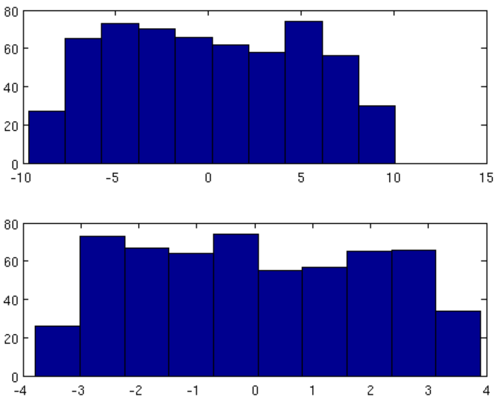

**Fig. 3.** Histograms of Mixed signals

Use the default setting of FastICA to find the independent components. [icasig, A, W]=FASTICA (mixedsig), the input need to be # of signal ∗# time points. The rows of the icasig are the ICs with the mean added back, and $A$ is the coefficient matrix as $X = AS$ and $W * A = I$. Just one thing need to pay attention, if you output just two variables, $A$ and $W$ will output.

The figure of those independent components are Fig. 4 and 5.

```
[icasig A1 W] = FASTICA (mixedsig); % find the ICs by default setting,
% mixedsig is in # of signal * # time points
figure; % visualize those 2 signals
for i=1:2
subplot(2,1,i)
plot(icasig(i,:))
end

figure; % visualize those 2 signals
for i=1:2
subplot(2,1,i)
hist(icasig(i,:))
end
```

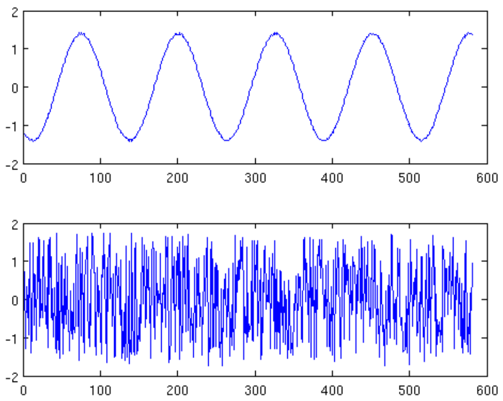

**Fig. 4.** Two independent Signal Lent from FastICA

Several things worth mentioning

– From ICA, there is no natural order of independent components.
– The independent component is unique up to sign. In my example, the first component is the negative of the ground-truth. The learned $A1$ matrix is $[-0.9990, 4.9775; -0.2997, 1.9928]$, the first column is the opposite of the $A$ matrix.

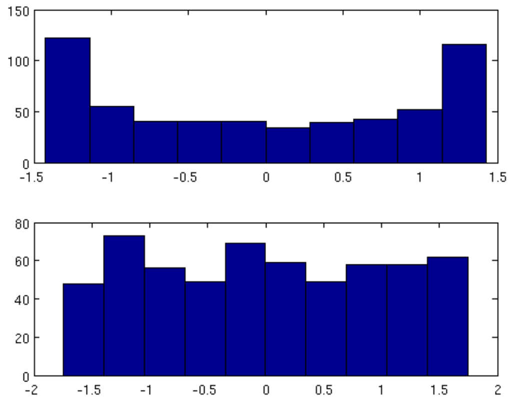

**Fig. 5.** Histogram of independent Signals

– Since the reprocess include whitening, another common thing is to use lower dimensional representation from PCA to avoid information redundancy and get computation efficiency.

**EX2** In this example, I used images as input, the principle is same as EX1. Suppose our source are two colored pictures as in Fig. 6

```
% read in source
I = imread('p1.png');
J = (rgb2gray(I));
figure;
imshow(I)
J_final(1,:)=im2double(reshape(J(1:200,1:200),1,40000));
J_final(1,:)=J_final(1,:)./std(J_final(1,:));

I = imread('p2.png');
J = (rgb2gray(I));
figure;
imshow(I)
J_final(2,:)=reshape(J(1:200,1:200),1,40000);
J_final(2,:)=J_final(2,:)./std(J_final(2,:));
figure
for i=1:2
subplot(2,1,i)
imshow(reshape(J_final(i,:),200,200))
end
```

Since, we just want to illustrate the idea. So I only used the grey image with unite variance as our source.

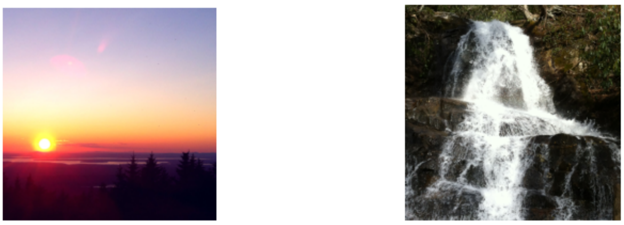

**Fig. 6.** Two color pictures

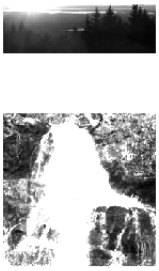

**Fig. 7.** Two grey pictures with unit variance

And generate two mixed images as our observation.

```
A= [ 0.2186    0.2830;
    0.0119    0.3113];
mixedsig=A*(J_final);
figure
for i=1:2
subplot(2,1,i)
imshow((reshape(mixedsig(i,:),200,200)))
end
```

Note,because of the sign ambiguous, even for the unite variance source, we can have $2^{\#IC}$ solution. In Fig 9, I selected the one that is consistent with the source(not applicable in real problem).

```
[icasig A1 W] = FASTICA (mixedsig);
```

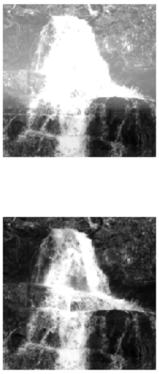
**Fig. 8.** Mixed pictures

```
figure
for i=1:2
subplot(2,1,i)
imshow((reshape(icasig(i,:),200,200)))
end

index=[-1,1];
figure
for i=1:2
subplot(2,1,i)
imshow((reshape(index(i)*icasig(i,:),200,200)))
end

index=[1,-1];
figure
for i=1:2
subplot(2,1,i)
imshow((reshape(index(i)*icasig(i,:),200,200)))
end
index=[-1,-1];
figure
for i=1:2
subplot(2,1,i)
imshow((reshape(index(i)*icasig(i,:),200,200)))
end
```

**Fig. 9.** Learned independent components

## References

1. Aapo Hyvärinen and Erkki Oja, *Independent Component Analysis: Algorithms and Applications*. Neural Networks, 13(4-5):411-430, 2000.
2. Ganesh R. Naik and Dinesh K Kumar, *An Overview of Independent Component Analysis and Its Applications*. Informatica 35 (2011) 6381. 63.