Assignment 2: ICA

Xiyu Wang

xw5638 xiyu_wang@utexas.edu

Feb. 20, 2020

1 Introduction

Signal recognition problems can be solved with machine learning technologies including classification, neural networks and so on. The main idea is to recognize several signals and regard the sound in chaos as a mixture of them. The ideal result is to recover exactly the source signals regardless of the noises. But in most case, we have no idea about how the source signals look like, which is blind source problem. In this case we need to solve the blind source separation problem. One of the practical algorithms about this is independent component analysis (ICA), which project the data to a lower dimensional subspace with reconstructed signals.

In this report, two different signal sets are used to implement the ICA algorithm. First ICA is applied to a test data with three simple short signals with a given mixture matrix A. Then a larger sound data is used for more challenging and practical cases. This is accomplished by assuming that there is no correlation between source signals. During which process some figures will be generated for comparison and analyses. At last, the accuracy of the classification is analyzed and compared with different parameters.

2 Methods

Independent component analysis (ICA) recover the mixed signals to original signals as estimation. In all, a mixed signal matrix X should be generated. After that, the ICA algorithm would deal with X according to a given learning rate η and a given iteration number.

When mapping the source signals into mixture signals, there exists a mixture matrix A. The relationship between source signal matrix U and mixed signal matrix X is:

$$X = AU$$

For the test case, U and A are both given so it's easy to generate X. But for the sound signal, the A matrix will be generated by ourselves.

During ICA, a matrix W is initialized with small random values such as from 0 to 0.01 since the overall signal values are mostly within 1.

The estimate source signals Y can be expressed as:

$$Y = WX$$

Then calculate matrix Z to help with traverse the gradient of maximum information separation:

$$Z = 1 / (1 + \exp(-Y))$$

The best update of matrix W can be generated:

$$\Delta W = \eta \, (I + (1 - 2Z) \, Y^T) \, W$$

where η is the learning rate, which is in range 0~1. The common value of it is 0.01~0.0001.

For better result, it also can be computed as:

$$\Delta W = \eta \, (I * t + (1 - 2Z) \, Y^T) \, W$$

where t is the length of the signals.

When the update of W is finished, one iteration is completed. The estimated signals Y can be computed again and evaluated if necessary. If the iterations satisfied the requirement, the new Y can also be output and it's the result of the ICA algorithm.

$$W = W + \Delta W$$

$$Y = WX \text{ (start a new iteration or output)}$$

After a set of reconstructed source signals obtained, since the real source signals are given, it's necessary to compare and evaluate the reconstructed signals. Also, plot the figures of the real source signals, mixed signals and reconstructed signals separately after scaling them to the range of 0~1.

3 Results

3 (1) Evaluation

The reconstructed source signals Y are blind separated and the order of signals may vary. It's a problem to find the corresponding real source signal and do the evaluation process. A special algorithm is designed to recognize the matching source signal.

For the Y and U with same dimension n, compare the signals one by one with a two-dimension for loop. For each signal in Y, go through all signals in U and calculate the correlation coefficient. Select the highest one among others and recognize the matching source signal. In this experiment, all correlation coefficient above 0.5 are printed to observe how everything work. After that the highest of all is regarded as the matching one. In very rare case, if more than one reconstructed signal get matched to the same real source signal, it always has relatively low coefficient and the ICA accuracy is not good, which also means the reconstructed signals are unrecognizable.

The correlation coefficient between two vectors x and y is computed as follows:

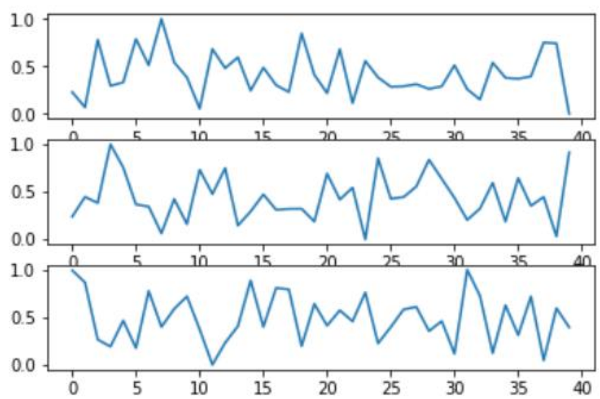
$$r = \frac{\sum(x - m_x)(y - m_y)}{\sqrt{\sum(x - m_x)^2 \sum(y - m_y)^2}}$$

where m_x is the mean of the vector x and m_y is the mean of the vector y.

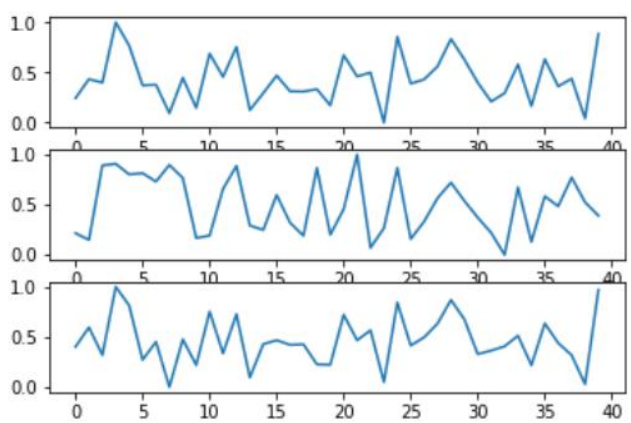
3 (2) Test case

In test case, a small set of signals Ut are given with a mixture matrix At. The shape of them are (3 * 40) and (3 * 3).

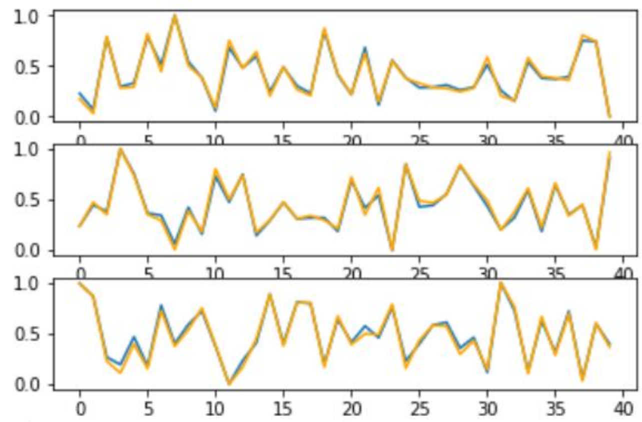
The real source signals Ut can be plot:



And generated Xt (3 * 40) can be plot in comparison:



In the test case, the learning rate η is set as 0.001. Since the algorithm can run quite fast, after 1,000,000 iterations, the reconstructed source signals are shown, with the blue lines being the real source signals and orange ones being reconstructed estimated source signals.



And the correlation coefficient are:

	Ut[0]	Ut[1]	Ut[2]
Correlation Coefficient	0.9906	0.9918	0.9924

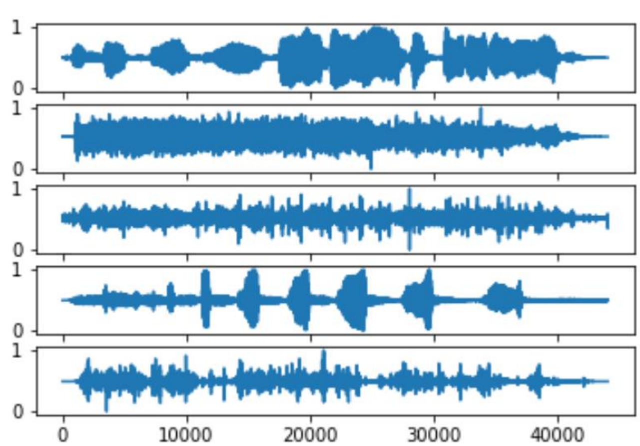
which is quite satisfying.

Also, in the figure it can be found that for estimated source signals, the trend of the signal is mostly accurate. It's the peaks and crests of signal wave more difficult to recognize and separate. Also, the frequency is also quite accurate. In the test case we can for-see how the larger sound signals would perform. The main sound of the signals may get recognized but the volume of them may get disturbed by others and have some error.

3 (3) Sounds signals

For the sounds signals, a large matrix "sounds" is given with shape (5 * 44000). The real source signals U can be generated by selecting some or all the signals within "sounds". Assume in U there are n signals.

The original sounds signals can be ploted for a start:

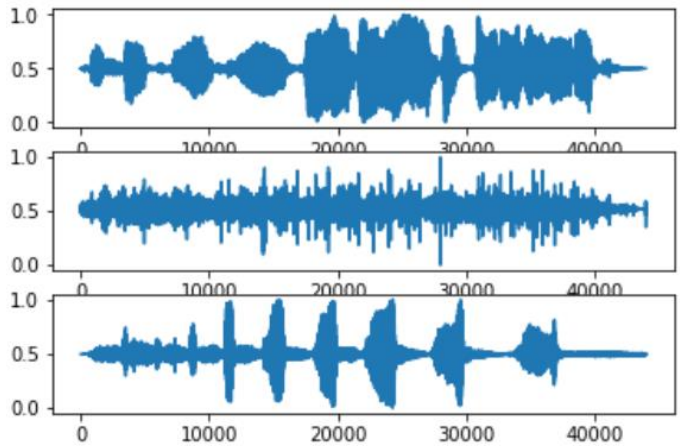


Listen to the signals separately, and they generally are: a man talking, sound of a chain saw, applause, a woman laughing and sound of some scratching.

To get a mixed signal X, in this experiment A is generated randomly within 0~1. For the X with m separate signals, the shape of A is (m * n) since U is (n * 44000).

3 (3) 1 U = sounds[0, 2, 3]

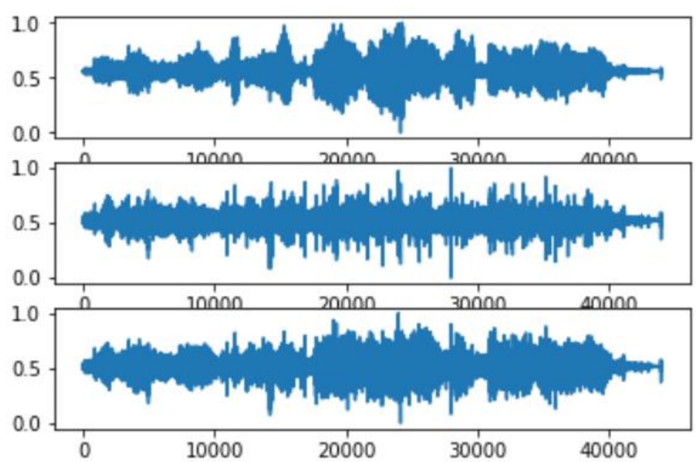
Selecting the n = 3, U is formed with sounds[0], sounds[2] and sounds[3]. U can be plot as follows:



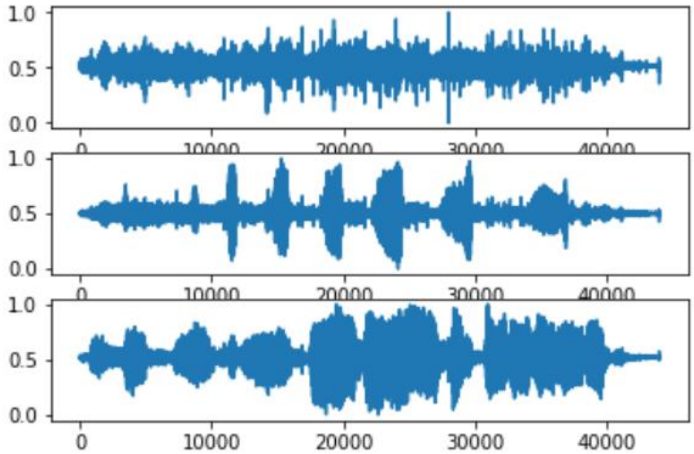
The generated random A :

```
[[0.38337807, 0.34219559, 0.64150542],
 [0.09689993, 0.69389635, 0.13623670],
 [0.31019958, 0.54628450, 0.19681414]])
```

And the X mixed up:



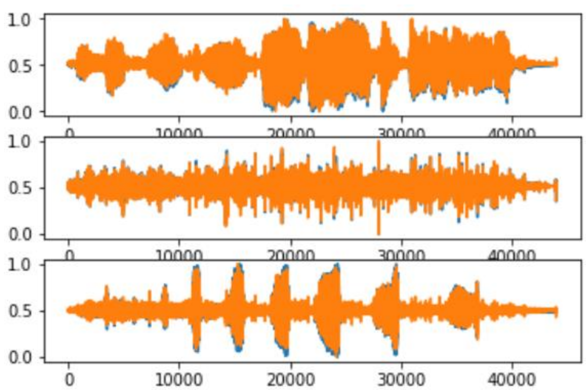
The η is set 0.001 and the iteration runs 5000 times. Separated source signals Y are shown below:

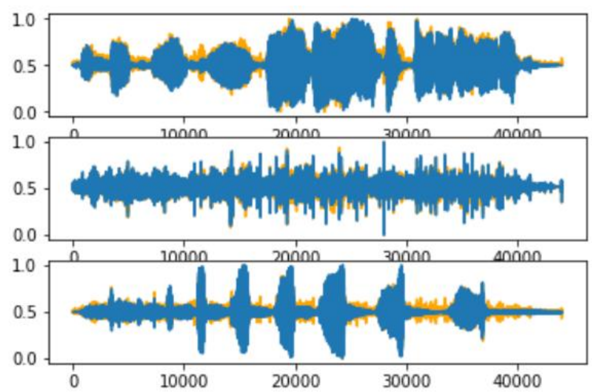


And the correlation coefficient are:

	U[0]	U[2]	U[3]
Correlation Coefficient	0.9603	0.9094	0.9433

For visual compare, the blue one is the real source signals U and orange one is reconstructed source signals Y:



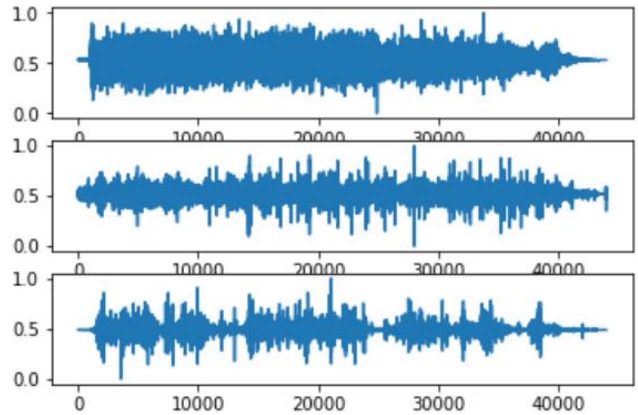


Above two plots are the same figure with different signal on the top level. It can be observed that the main trend of signals can be separated well, with some peaks inaccurate and some white noise included in the reconstructed signals.

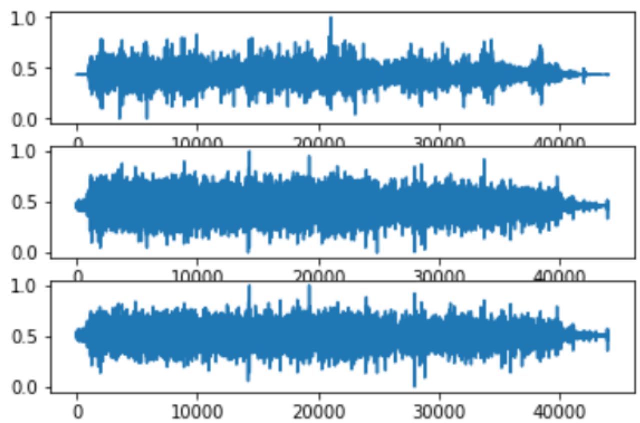
3 (3) 2 U = sounds[1, 2, 4]

Next, try another source set: sounds [1], sounds [2] and sounds [4]. The selected three signals are all closer to noise and have less features to recognize.

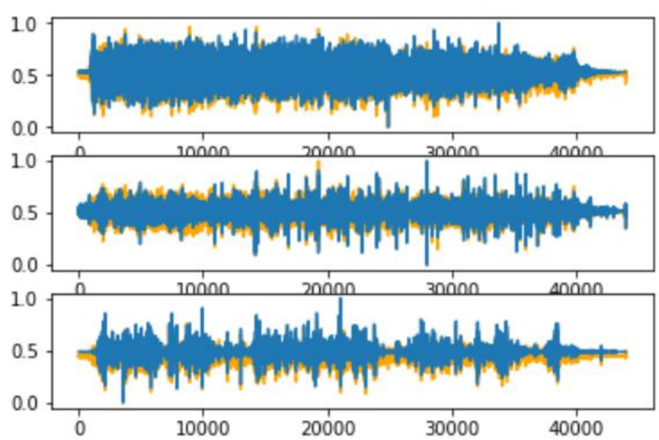
The U, A and X are shown below:



```
[[0.47212101, 0.05380419, 0.92220406],
 [0.98615490, 0.79405393, 0.68595349],
 [0.79679835, 0.94071038, 0.29492918]])
```



The η is set 0.001 and the iteration runs 5000 times. Separated source signals Y are shown below, with the blue one is the real source signals U and orange one is reconstructed source signals Y:



And the correlation coefficient are:

	U[1]	U[2]	U[4]
Correlation Coefficient	0.9056	0.8113	0.8183

From the figure the reconstructed signals matched well with the real signals. But from the coefficient, this case is less than the former case. It's possible that the signal 1,2 and 4 are more similar in wave shape and some other features so it's difficult to set them apart than the former case.

3 (3) X shape (6, t)

Keeping U (n * t) as fixed value (U = sounds[0, 2, 3] for instance), the shape of X (m * t) varies with the shape of A (m * n). The basic requirement is m >= n. Set m = 6 for instance with n = 3, compare the coefficient ICA algorithm returns:

#iterations	U[0]	U[2]	U[3]
5000	0.9624	0.6693	0.6949
7500	0.9601	0.8003	0.8253
10000	0.9909	0.9913	0.9862
15000	0.9579	0.9421	0.9115

According to the table, when the mixed X contains more signals, it may take more iterations to separate the source signals. In this case 10000 iterations is obviously more proper than 5000. Although taking more iterations, the accuracy or the reconstruction is higher for X provided more information. In 15000 iterations it's a little bit over fitting because the learning rate is not small enough for the large iterations so it's possibly unnecessary.

Also, in this experiment it's shown the signal 0 is much easier than the other two to separate. When it's 5000 iterations it's already with 0.96 coefficient while the other two are still below 0.7.

4 Summary

ICA can separate source signals from a mixture of signals if the number of signals are equal or more than the number of sources. Without knowing the value of sources, it can operate the blind source separation and approach the target source signals with gradient descent method.

For different source signals, the difficulty of separation is different. Within the same iteration, some signals will get a high coefficient faster than others, which indicates that it's easier to be separated from the mixture. In this case, the signal 0 with a man talking is the easier one and the plastic sound is more similar with environment noise and more difficult to find.

When the number of signals in mixture is more than the number of sources, it always takes more iteration to get a great separation. However, with more information contained in X, the final coefficient is also higher than the ones with less mixture signals.

If the initialized W is not small enough compared with the signal we are analyzing, it's much more difficult to get a great separation, which is one of the mistakes I made. Also, scaling the signal between 0 ~ 1 is not helpful during the analyze if it's not normalized. For instance, I tried to simply scale the range within 1 by magnify the signals with (max – min). It's value during separation is changed actually and it increased the difficulty of ICA. Some of them are nearly impossible to recognize. Instead, use normalization has the least affect on the separation. Scaling only before plotting is another idea.