CS391L Machine Learning Assignment 2

Srinath Tankasala,
st34546

Abstract

In this assignment, an overview of Independent Component Analysis (ICA) is presented and is applied to separate a mixture of sounds. A set of 5 different sources were taken and mixed to obtain a 5 different ”microphone” recordings. The ICA algorithm was applied to retrieve the individual sounds and the retrieval quality was examined.

I. Introduction

The cocktail party problem is a classic application of ICA. There are many sound sources and focus needs to be given to the relevant ones while ignoring the others. In such problems we assume that we have recordings from different microphones containing the mixture of conversations. The cocktail party problem is solvable if the number of microphone recordings is equal to or greater than the number of sound sources in the room.

For the purposes of this assignment we mix the sounds so as to have equal number of ”microphone” recordings. Additional microphone recordings may help increase the quality of the retrieved sounds, but that is not explored in this work.

II. Mixing matrix

We are given a test data set U containing $n = 5$ sound sources and t number of samples. The aim is to mix these sounds to generate a pseudo microphone output X of $m = 5$ recordings. To do this we use a mixing matrix A. While there is no restriction on the choice of A, for consistency we choose one A for multiple runs

$$X = AU, \text{A is mixing matrix of size (m,n)} \tag{1}$$

Hence X is a m-by-t matrix of the mixed sound signals.

III. Independent Component Analysis

ICA decomposes a mixture of signals into its individual sources. This is different from Principal Component Analysis which tries to find the dominant vector components in the given dataset. Our goal is to find the ”un-mixing” matrix ’W’ of size n-by-m.

$$\begin{aligned} X &= AU \\ \text{determine W s.t.,} \\ WA &= I, \\ U &= WX \end{aligned}$$

We use an algorithm to find an estimate of the matrix W called the maximum likelihood estimate, that is denoted by $\hat{W}$. This method is explained in the next section.

A. Maximum Likelihood Estimation of Unmixing matrix:

There are several methods available in literature that can be used to calculate the unmixing matrix W, some of them use a SVD (singular value decomposition) method to calculate the matrix W. There are advantages and disadvantages to these available approaches. For example, the SVD approach involves calculation of the Eigenvalues of XX^T which may be slow if the number of microphone inputs (m) is large. The Maximum Likelihood Estimate (MLE) of W is calculated by maximizing the parameters of W to match the mixed data X. To do this, the following definition is used for the likelihood of the mixture X:

$$p(x = X) = p_x(X) = p_u(U) \cdot |W| = \prod_{i=1} p_i(u_i)|W| \tag{2}$$

where, $p_u(U)$ is the probability density function of the independent source signals, and $p_i(u_i)$ is the probability density of the i^{th} source component

Since we have t samples of all mixtures in X, we can calculate $P_x(X)$ as:

$$L(W) = \prod_{j=1}^{t}\prod_{i=1}^{n} p_i(w_i^T x_j)|W| \tag{3}$$

where, x_j is the j^{th} column of X.
Thus our goal is to maximize $L(W)$ over all W, i.e.:

$$\max_{W \epsilon R^{(n,m)}} L(W) \tag{4}$$

Given that the probability density function is always positive, we can maximize $L(W)$ by maximizing the log likelihood function. The reason for maximizing log likelihood is because, taking log of $L(W)$ changes the product terms in W into summation terms, i.e.:

$$\ln\left(L(W)\right) = \textstyle\sum_{j=1}^{t}\sum_{i=1}^{n} \ln\left(p_i(w_i^T x_j)|W|\right) \tag{5}$$

$$= \textstyle\sum_{j=1}^{t}\sum_{i=1}^{n} \ln\left(p_i(w_i^T x_j)\right. + \displaystyle\sum_{i=1}^{n} \ln(|W|) \tag{6}$$

$$\ln\left(L(W)\right) = \textstyle\sum_{j=1}^{t}\sum_{i=1}^{n} \ln\left(p_i(w_i^T x_j)\right. + t\ln\left(|W|\right) \tag{7}$$

We can simplify equation 7 further as:

$$\frac{1}{t}\ln\left(L(W)\right) = E\left[\sum_{i=1}^{n} \ln\left(p_i(w_i^T x_j)\right)\right] + \ln\left(|W|\right) \tag{8}$$

where E[] is the expectation/mean observation.
Here we introduce the concept of the cumulative density function (cdf). The cumulative density function $g(X)$ is the integral of the pdf, and gives the net probability of the function having a value below X. Thus the pdf $p(X)$ is simply the derivative of $g(X)$. Thus equation 8 becomes,

$$\frac{1}{t}\ln\left(L(W)\right) = E\left[\ln\left(g'(WX)\right] + \ln\left(|W|\right)\right. \tag{9}$$

There are many algorithms that are available for maximizing the the log-likelihood function. Gradient descent algorithm is a popular approach to maximize $L(W)$. Since we maximize with respect to W, the gradient is taken accordingly, i.e.:

$$\frac{1}{t}\frac{\partial}{\partial W}\ln(L(W)) = E\left[\frac{\partial}{\partial W}(\ln\left(g'(WX)\right)X^T\right] + \frac{\partial}{\partial W}\ln\left(|W|\right) = E\left[\frac{\partial}{\partial W}(\ln\left(g'(WX)\right)X^T\right] + \left[W^T\right]^{-1} \tag{10}$$

B. Gradient Descent:

The gradient descent algorithm is a iterative algorithm which can be used to minimize or maximize a function. If the function being optimized is convex in nature then the gradient descent converges to the function extremum. Proving the convexity of the function $L(W)$ is beyond the scope of this report and it has not been shown here. Gradient descent starts at an initial guess point $\hat{W}_1$ and updates it by moving along the function gradient evaluated at that point. Thus a gradient descent applied to find W results in:

$$\hat{W}_{k+1} = \hat{W}_k + \eta \cdot \left(\frac{1}{t} \frac{\partial}{\partial W} \ln(L(W)) \right)_{W=\hat{W}_k} = \hat{W}_k + \eta \cdot \Delta W \tag{11}$$

where, $\hat{W}_k$ is the estimate of the W matrix after the k^{th} iteration of the gradient descent algorithm. The starting point, i.e. $\hat{W}_1$ is assumed random, however for the sake of consistency of results shown in this report, the value of $\hat{W}_1$ is taken fixed. η is the "learning rate" of the gradient descent algorithm and should be adjusted to achieve convergence.
Now the gradient term contains an inverse operation for the W^T matrix which is computationally expensive to calculate. Thus the gradient step can be conditioned by multiplying it with $W^T W$. This does not affect it's convergence to the optimum. It would involve a lot of algebra to show that the optimality of equation 4 is unaffected by making the above change. For brevity of this report, this has been omitted and it is encouraged to refer the class notes. Hence,

$$\Delta W = E\left[\frac{\partial}{\partial W} (\ln (g'(WX)) X^T \right] W^T W + \left[W^T \right]^{-1} W^T W \tag{12}$$

$$\Delta W = \left(E\left[\left(\frac{\partial}{\partial W} (\ln (g'(WX)) \right) (WX)^T \right] W + W \right)_{W=\hat{W}_k} \tag{13}$$

To facilitate the gradient descent algorithm, it will be ideal to choose a cdf that is differentiable for all possible values of WX and is also easy to calculate. With that in mind, the cdf is defined as

$$g(WX) = \frac{1}{1 + e^{-WX}}$$

this function is differentiable for all WX and is bounded between 0 and 1. It is easy to show that the corresponding pdf, i.e. the derivative of g is given by:

$$g'(WX) = g(WX)(1 - g(WX))$$

With this simplification, we can plug the above derivative into equation 13 to get,

$$\Delta W = \left(\left(E\left[(1 - 2g(WX))(WX)^T \right] + I \right) W \right)_{W=\hat{W}_k} \tag{14}$$

The final algorithm can be summarized as:

1) Start with an intial guess of the W matrix, i.e. $\hat{W}_1$
2) Calculate matrix Z_k which is given by $g(\hat{W}_k X)$
3) Calculate ΔW using W_k as $\left(\frac{1}{t} \left[(1 - 2Z_k)(\hat{W}_k X)^T \right] + I \right) \hat{W}_k$
4) Update the estimate as $\hat{W}_{k+1} = \hat{W}_k + \eta \cdot \Delta W$
5) Check for convergence by seeing if $||\Delta W||/||W||$ is below the tolerance value, if not then return to step 2

IV. Results and Discussion

The original source signals were extracted from the 'sounds.mat' file. They were used to obtain the U matrix. The mixing matrix can be any random 5x5 matrix that is full rank (5). For consistency between multiple runs, the signals were mixed with the following mixing matrix:

$$A = [[0.5, 1, 0.2, 1, 0.3]; [0.5, 0.5, 0.2, 0.6, 0.1]; [0.8, 0.4, 0.5, 0.5, 0.31]; [0.5, 0.4, 0.5, 1, 1]; [0.2, 0.7, 0.7, 0.1, 0.3]]$$

The mixed signals matrix (X) was obtained by multiplying A and U.
For the gradient descent algorithm, the learning rate was set to $\eta = 0.01$ and the tolerance limit for the relative norm, i.e. $||\Delta W||/||W||$, was set to 10^{-6}. The algorithm finished in 6979 iterations and was able to successfully isolate the individual components/sources.

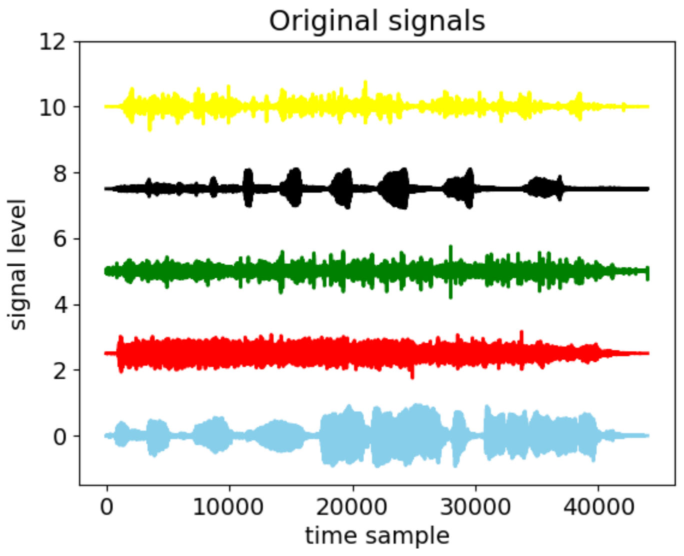

Fig. 1: Original signals in ascending order (y-offset is for plotting purposes)

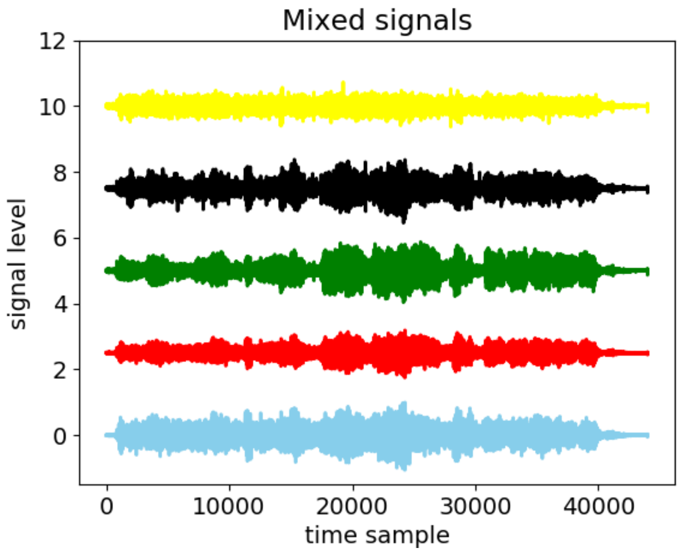

Fig. 2: Mixed signals in ascending order (y-offset is for plotting purposes)

If W is a solution to equation 4, then so is $a \cdot W$ where a is any scalar. This means that the solution is not unique and depends on our chosen starting point $\hat{W}_1$. Hence, for consistency between multiple runs, $\hat{W}_1$ was chosen as a fixed value.
Once we obtain the best estimate $\hat{W}$, we can retrieve the estimate of the original signals by multiplying with X, i.e. $\hat{U} = \hat{W}X$, where $\hat{U}$ is the retrieved estimate

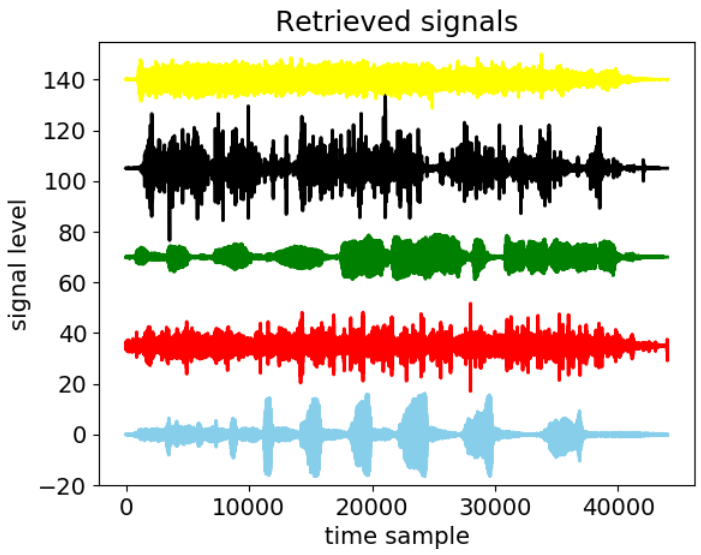

Fig. 3: Retrieved signals in ascending order (y-offset is for plotting purposes), notice the magnitudes

It can be seen from Fig 3, that the retrieved signals are scaled and not in the same order as the source signals. Since we actually know what is the mixing matrix A, we can obtain the scaling factor and order of signals from the product $\hat{W} \cdot A$. This would not be possible normally as A would be unavailable. In such cases, the sources have to manually matched. Then all signals in the retrieved estimate would have their peaks normalized between [-1,1]. The performance of the ICA algorithm can be measure by how well the retrieved signals correlate with the original sources, as shown below,

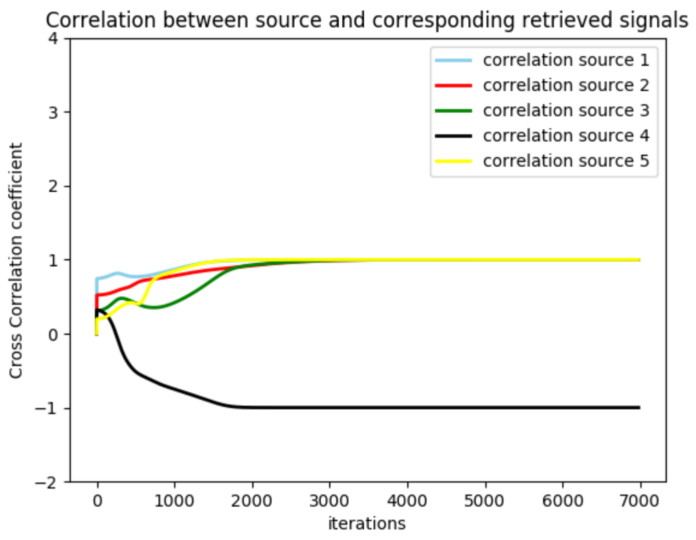

Fig. 4: Correlation between source signals and their corresponding recovered signals

As can be seen in Fig 4, the algorithm improves its estimate as the number of iterations increase. Also notice that the correlation asymptotically reaches -1 or 1. This means that the rate of convergence slows down as iterations increases. The quality of the retrieval can also be visualized by overlaying the source signal and the scaled retrieved signal.

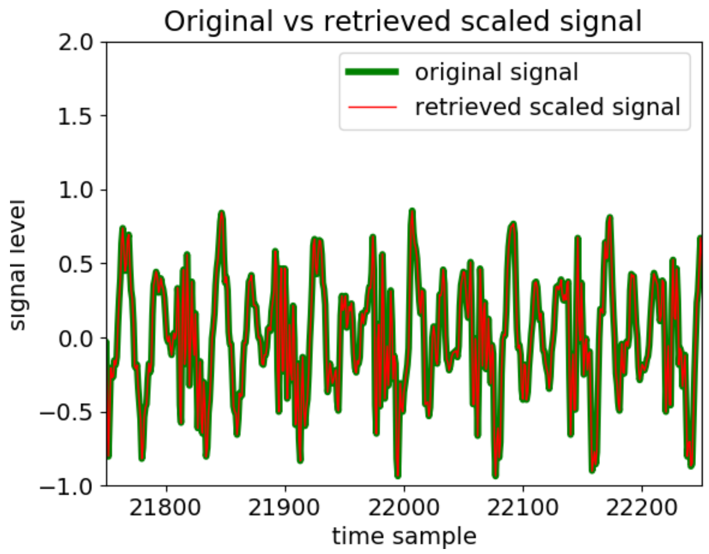

Fig. 5: Comparison of source and retrieved Homer Simpson audio

V. Conclusion

In conclusion, a set of test audio sources were mixed and the source audio was successfully retrieved from the mixture by performing an ICA. The ICA was implemented using a gradient descent method. The performance of the ICA was measured by how well the retrieved signals correlated with their original audio. The estimate of the inverse of the mixing matrix depends on the starting point of the gradient descent and the learning rate. The retrieved signals converge asymptotically to the source audio as the number of iterations increases. Adaptive gradient descent is a possible way to speed up the simulation time and a good future activity.