Machine Learning - Assignment 2

Kaivan Wadia
kaivanwadia@gmail.com
kpw423

February 18th, 2016

1 Introduction

The objective of this assignment is to learn about Independent Component Analysis (ICA) and how it is used to obtain source signals from multiple mixed signals. Independent component analysis aims to decompose a multivariate signal into its individual source signals. The source signals that are mixed together are independent of each other. Independent Component Analysis is used in problems such as the 'cocktail party problem', where the underlying speech signals are separated from a sample data consisting of people talking simultaneously in a room. In ICA the number of mixed signals has to be greater than or equal to the number of source signals. If the number of mixed signals is less than the number of source signals the system is classified as an under-determined system.

In this assignment our goal is to initially mix five source signals which have been sampled at a rate of 11025 samples per second. After mixing these source signals into m mixed signals our aim is to recover the original source signals from the generated mixed signals. For the experiment we have access to 2 different source signal data sets. One data set consists of 5 signals with 44000 samples and the other smaller data set consists of 3 source signals with a total of 40 samples. We use the smaller data set containing 3 source signals to initially test the ICA algorithm to check whether it is working correctly or not. We can then run experiments on the bigger data set by varying different parameters to simulate real world use cases.

The next section involves discussing the ICA algorithm and how I implemented it in MATLAB. After that I describe the experiments I ran and the results I obtained. Finally, I conclude with a section on further discussions and a summary of the experiments.

2 Method

In this task we perform Independent Component Analysis using the gradient descent method. The gradient descent method algorithm is not discussed in detail here as it has been discussed in the instructions of the assignment and in class. In short gradient descent aims to find the local minimum of a function by taking small steps proportional to the negative of the gradient at each iteration. In order to discuss the ICA algorithm I shall first define some parameters that are used in the algorithm:

We can broadly separate the classification task into two major steps:

1. n - The number of source signals that were mixed for the experiment.
2. t - The number of samples gathered for each source signal.
3. m - The number of mixed signals generated from which the source signals need to be recovered.
4. U - A n x t matrix of n source signals with t samples.

5. A - A m x n mixing matrix that mixes n source signals to give m mixed signals.
6. X - A m x t matrix of m mixed signals.
7. W - A n x m matrix that is used to recover the original source signals. It can also be thought of as an initial guess for A^{-1}.
8. Y - A n x t matrix containing the recovered signals after performing the ICA algorithm.
9. η - The learning rate for the gradient descent algorithm.
10. $RMaxIterations$ - The maximum number of iterations that should be run in the gradient descent algorithm.
11. $ConvergenceThreshold$ ϵ - The convergence threshold to be used in the gradient descent algorithm.

2.1 Mixing

The first step in our assignment is to mix the source signals. In order to mix the signals I defined a function to initially select the subset of source signals we want to mix for the experiment. The function takes in the signal numbers of the signals to be considered and also the range of samples of the signal to consider. In order to mix the source signal we randomly generate the A matrix with values between 0 and 1 in order to mix the source signals for the various tests. The mixed signals matrix X is generate by performing $A * U$, where U is the source signals matrix.

The A matrix is also dependent on the number of mixed signals that need to be generated for the experiment. One of the things to consider is that when running multiple experiments on the same source signals with the same number of mixed signals generated the A matrix should be kept constant. Therefore, I declare a constant A matrix before running the experiment for each variation of a parameter such as η. Also in the real world in a blind source separation problem we only have mixed signals and it does not make sense to observe the effect of changing A matrix values.

2.2 Gradient Descent

In order to perform the gradient descent part of the ICA algorithm we first initialize the W matrix with random initial values that are uniformly distributed between 0 and 0.1. Even the W matrix should be initialized to the same random values for multiple experiments and therefore I seed the random number generator with the same value before every experiment to start with the same initial weights.

Gradient descent is performed in the following steps:

1. $Y = W * X$
2. $Z = 1/(1 + e^{-Y_{i,j}}$ for every element [i,j] in the matrix Y.
3. $\Delta W = \eta(I + (1 - 2 * Z)Y^T) * W$
4. $W = W + \Delta W$
5. We repeat steps 1 to 3 either for the maximum number of iterations specified ($RMaxIterations$) or until the norm of ΔW is less than some convergence threshold (ϵ).

In the above mentioned algorithm for gradient descent we can vary two different parameters to produce different results. The learning rate η and the convergence threshold ϵ both can be varied for different results. This is discussed further in the experiments section.

2.3 Correlation Matrix and Accuracy

The final step in the implementation is to calculate the accuracy to which the source signals have been recovered from the mixed signals. For this task we initially calculate the correlation matrix between the source signals and the recovered signals. This matrix denotes how closely the recovered signal matches the source signal. The original source signals are denoted by the rows and the recovered signals by the columns of the matrix. The highest element in each row denotes which recovered signal is matched to which source signal. The higher the correlation value the more accurate is the recovered signal.

To calculate the average accuracy achieved by the algorithm we first obtain the maximum value in each row of the correlation matrix. The maximum value is calculated after taking the absolute value of the matrix as the recovered signal could be inverted. The average of these values is the average accuracy obtained by the algorithm.

In order to check if the above mentioned method was working correctly I ran the algorithm on the smaller data set with just 40 samples per source signal in order to compare if the recovered signals were similar to the source signals. I set $\eta = 0.01$ and used the A matrix provided along with the data set. Given below are the results of my test. The signals on the left are the source signals and the signals on the right are the recovered signals. I have also displayed the mixed signals below the source and recovered signals. The average accuracy achieved for this experiment is 99.16%.

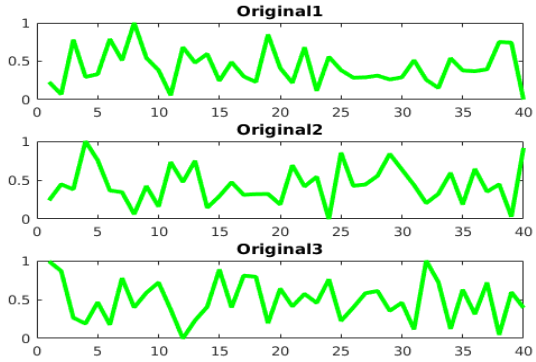

Figure 1: The original source signals

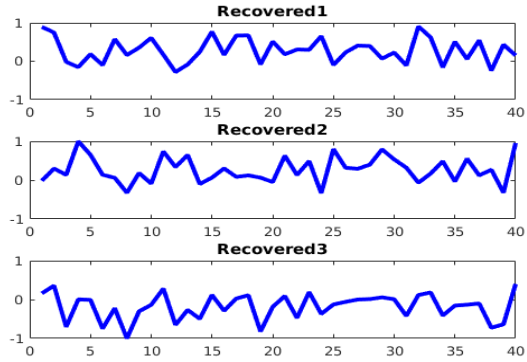

Figure 2: The recovered signals.

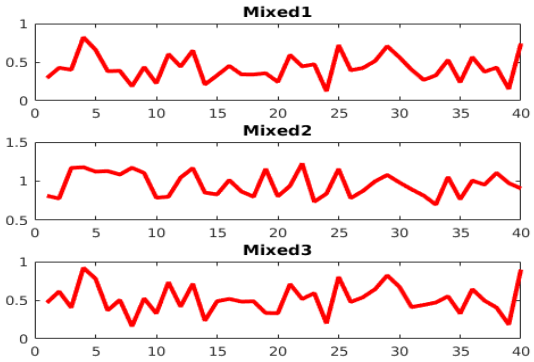

Figure 3: The mixed signals.

As we can see in the figures above the recovered signals are not in the same order as the original source signals. This is because ICA does not guarantee the order of the recovered signals to be the same as the source signals. We can also see that Recovered3 matches Original1 but is inverted. This can also happen as part of the ICA algorithm.

3 Experiments

I conducted various experiments to test the effect of changing various parameters on the accuracy of the resulting recovered signals. All my experiments were conducted on the large data set of 5 source signals with 44000 samples per signal. For the experiments I did not use all 5 source signals for all my experiments. Most of my experiments were conducted using only three source signals that were randomly selected.

The parameters that I varied for the various experiments were the learning rate η, number of mixed signals m and the convergence threshold ϵ. In order to conduct these experiments I needed to keep certain parameters constant throughout the experiments. Parameters such as the A matrix and the W matrix were generated once and used for all the tests conducted. The number of $RMaxIterations$ was set to $100,000$ for all the tests.

While conducting the experiments one of the problems I faced was that a high learning rate would result in the recovered signals being returned as NaN (Not a Number). In order to avoid this problem I had to change the range of random values of the A matrix from between 0 to 1 to 0 to 0.1.

I wrote a script to perform the experiments wherein I could specify which parameter to vary and the range in which to vary the parameter. The script could then run each test with the appropriate parameters. After conducting each test I store the mean accuracy achieved between the recovered and the original signals. Once all the tests were conducted for a certain experiment we can plot the effect of varying a certain parameter on the accuracy of the recovered signals. The results of the experiments are provided in the next section.

4 Results

As mentioned in the previous section I conducted three different experiments to check the effect of changing different parameters. The first experiment was to determine the effect of changing η on the average accuracy of the recovered signals. For this experiment I used three source signals mixed to form $m = 6$ mixed signals. The mixing matrix A was generated randomly and used throughout all the tests. The maximum number of iterations was $100,000$ and the convergence threshold was 10^{-8}. The average accuracy obtained for every test is was plotted in a graph and is shows in Figure 4.

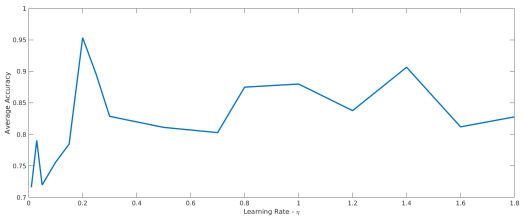

Figure 4: Plot of Learning Rate vs Average Accuracy.

As can be seen in Figure 4 the effect of the learning rate is not very deterministic on the resultant average accuracy of the recovered signals. In general the average accuracy seems to increase with an increase in learning rate but at certain points it drops drastically. The only explanation I could think of for these cases was over-fitting in the gradient descent part of the algorithm.

In the next experiment I varied the number of mixed signals generated while using three source signals and a learning rate of 0.03. The resulting plot is given shown below in Figure 5.

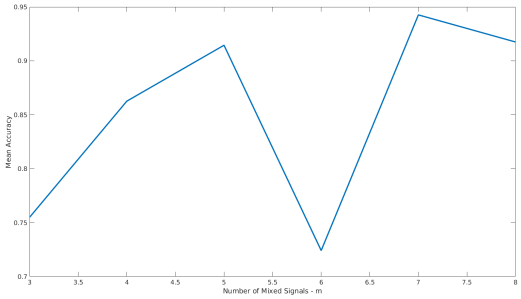

Figure 5: Plot of Number of Mixed Signals vs Average Accuracy.

In general increasing the number of mixed signals generated increases the average accuracy of the recovered signals. This is because we can find more common features between as the number of mixed signals increases. When the number of mixed signals is low we cannot distinguish between he source signals by finding any patterns. There seems to be an anomaly when we generate six mixed signals. The mean accuracy drops rather drastically in that case for which I cannot seem to figure out the reason.

The final experiment I conducted was to vary the convergence threshold of the gradient descent algorithm. For this experiment I fixed the learning rate to 0.03 and the number of mixed signals $m = 5$. The results of that experiment are shown in Figure 6.

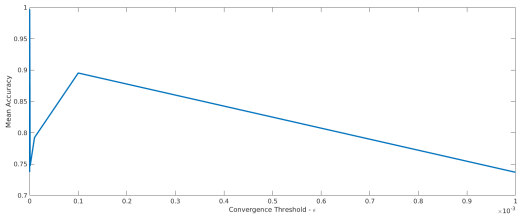

Figure 6: Plot of Convergence Threshold vs Average Accuracy.

As we decrease the convergence threshold we initially see observe an increase in the mean accuracy of the recovered signals. However after a certain point decreasing the convergence threshold leads to a decrease in the accuracy of the recovered signals due to over-fitting.

Finally, I also conducted various tests using all 5 source signals in order to try and achieve the highest average accuracy. I achieved an average accuracy of 94.99%. The correlation matrix for that test case is given below in Table 1. The rows denote the original signals and the columns denote the recovered signals. For this test I used a learning rate of 0.25 and ran the gradient descent algorithm for $250,000$ iterations. The number of mixed signals generated was 8 and the range of the mixing matrix A was between 0 and 0.1. While the convergence threshold was set at 10^{-5} the test did not reach that threshold. This might also indicate that maybe I could have achieved a higher accuracy for a greater number of iterations so that I could reach the convergence threshold.

Signals	Recovered 1	Recovered 2	Recovered 3	Recovered 4	Recovered 5
Original 1	0.0734	0.1938	**0.9721**	0.0380	0.1029
Original 2	-0.2834	-0.1285	-0.0174	**0.9254**	0.2158
Original 3	**0.9237**	-0.0261	-0.0601	0.3244	-0.1928
Original 4	0.2255	0.0108	-0.1090	-0.1470	**0.9568**
Original 5	-0.0215	**0.9715**	-0.2004	0.1244	-0.0044

Table 1: Correlation Matrix for 5 source signals

5 Conclusion

As shown through the various experiments changing the values of the various parameters affects the average accuracy obtained in the recovered signals. One of the interesting points of note was that I could not find a very good correlation between the learning rate and the average accuracy.

If I had more time to conduct more experiments I would have liked to conduct all the tests while using all five of the source signals provided, but in lieu of the time taken to run the tests it was better to run the experiments using only 3 of the signals. I did however run a few tests using all five source signals and they seemed to match the general pattern observed when conducting tests using just 3 of the signals.

I would have also liked to attempt to implement a dynamic learning rate as it would probably have sped up the tests and provided better accuracy in the end. By using a high learning rate early on in the test and then lowering the learning rate we could have reached a minimum faster, however there is always a risk of missing the true minimum and finding a local minimum instead. I currently use only one measure to calculate the convergence threshold. That is also one area that could be explored further and see how it affects the results.

Overall, I do feel that the assignment was a good learning experience. One of the biggest takeaways was that if two signals were even slightly similar and seemed to be dependent on each other the algorithm did not perform all that well, therefore it is necessary that the signals not be dependent on each other.