Separating Sounds using Independent Component Analysis

Daniel Alejandro Noble Hernandez
dan833
The University of Texas at Austin

1. Introduction

Independent component analysis (ICA), widely used for machine learning applications, is a technique that separates a multivariate signal into components, based on the assumption that these components are non-Gaussian signals independent of each other. ICA is a case of the broader term, blind source separation, and is used in this project to separate mixed sound signals into their separate component signals so that they closely represent the original signals.

The purpose of this project was to take a matrix of sound signals – first from a smaller test set and then using full-length signals – and mix them using a randomly generated matrix. Following this, independent component analysis was to be carried out in order to generate a matrix that could separate the mixed signals. This would allow for the recovery of signals based on the original signals, and the accuracy of the algorithm could be evaluated by calculating the Pearson coefficients between each of the pairs of original and recovered signals.

2. Methods

2.1 Test set

The independent component analysis algorithm begins with an *n x t* matrix *U* of *n* sound signals of length *t*; before implementing ICA on the full data set, the author carried it out on a test set of 3 sound signals of length 40. The matrix *U* was left multiplied by a randomly generated *m x n* matrix *A* (in this case 3 *x* 3) to create a matrix *X* of the mixed sound signals. The ICA algorithm steps are outlined below, and the author followed this process to generate a matrix *W* that would then allow for recovery of the original signals:

1. Initialize an *m x n* matrix *W* with small random values drawn from a uniform distribution.
2. Calculate the current estimate of the source signals as $Y = WX$.
3. Calculate the matrix *Z* where $z_{i,j} = \frac{1}{1+e^{-y_{i,j}}}$ for i ∈ [1, …, n] and j ∈ [1, …, t].
4. Using a small learning rate η, find the step $\Delta W = \eta(\mathrm{I} + (1 - 2\mathrm{Z})Y^T)W$.
5. Update $W = W + \Delta W$.
6. 6. Carry out steps 2-5 until convergence or for *R_max* iterations.

These values used for ICA on the test set were 100,000 iterations and a learning rate of 0.001. At this point, one could left multiply the mixed signals X by W to generate the matrix of recovered signals, $recovered = WX$, and then plot them to compare them with the original signals.

In order to validate the accuracy of the recovered signals with respect to the original signals, the author calculated the Pearson coefficient of each pair of original and recovered signals; those that bore the values closest to 1 or -1 were the pairs that corresponded to each other. The Pearson coefficient returns a value between -1 and 1, where a value near -1 indicates a negative correlation – representative of a corresponding signal that has been mirrored – and a value near 1 indicates positive correlation. These results are shown below.

2.2 Full data set

Having validated that the code worked for the test set, the author then carried out the same procedure on the full data set. Given the input matrix U of 5 signals rather than 3, the author chose the first 3 signals and worked with those in the same manner as outlined above. Given a much larger data set with *t* = 44000, the code took significantly longer to run; with that consideration in mind, the learning rate was set a 0.01 and ICA was run for 10,000 iterations. The results are displayed and discussed in the next section.

---

3. Results

3.1 Test set

Figure 1 shows the results of implementation of ICA on the test set using a learning rate of 0.001 and 100,000 iterations.

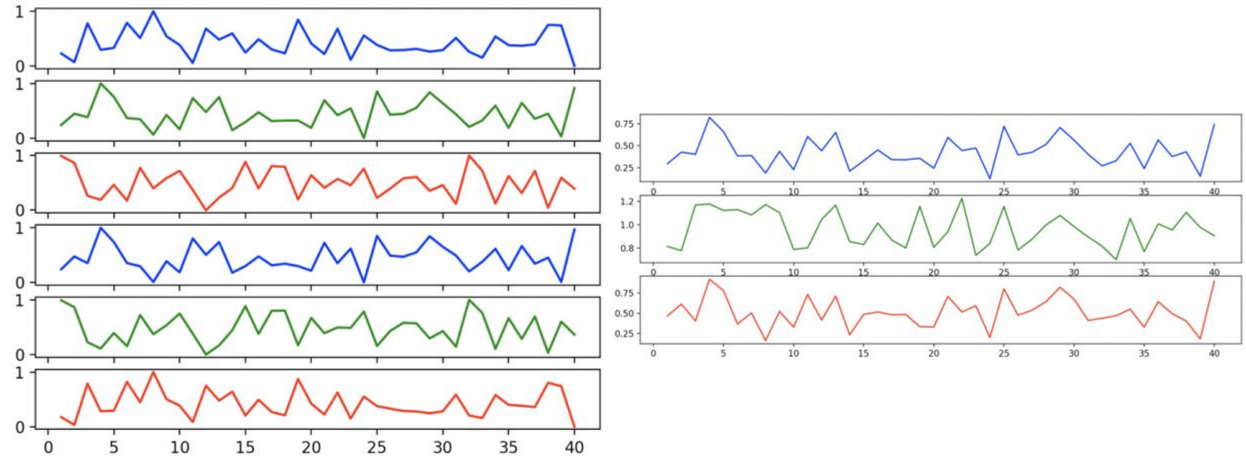

Figure 1(a): Original and recovered signals from the test set. The top 3 are the original signals and the bottom 3 are the recovered signals. Figure 1**(b)** is shown on the right and demonstrates the mixed signals represented by the matrix X.

As shown in the plots, the recovered signals are very similar to the originals; although their order has been mixed up, one can line up which original signal corresponds to which recovered signal. This is facilitated by the table shown in Figure 2, which displays the Pearson coefficients between each of the pairs of original and recovered signals. The bold numbers represent the correct pairs, and we see that they are highly correlated, all having values over 0.99, which indicates that the algorithm carried out the process very efficiently.

	Recovered 1	Recovered 2	Recovered 3
Original 1	-0.4886301	-0.4259943	**0.9906693**
Original 2	**0.99184322**	-0.5442918	-0.3976619
Original 3	-0.484278	**0.99247109**	-0.5055279

Figure 2: The Pearson coefficients between each of the test vector combinations for the original and recovered signals. The bold highlights which of the recovered signals each original signal corresponds to.

3.2 Full data set

The original, mixed, and recovered signals for the full data set using 10,000 iterations and a learning rate of 0.01 are illustrated in Figure 3. These bear less of a striking resemblance to one another than the test set did, indicating that the signals were not recovered as well as for the test set. Nonetheless one can clearly, visually match the first original signal with the last recovered signal, the second original with the first recovered, and the third original with the second recovered signal.

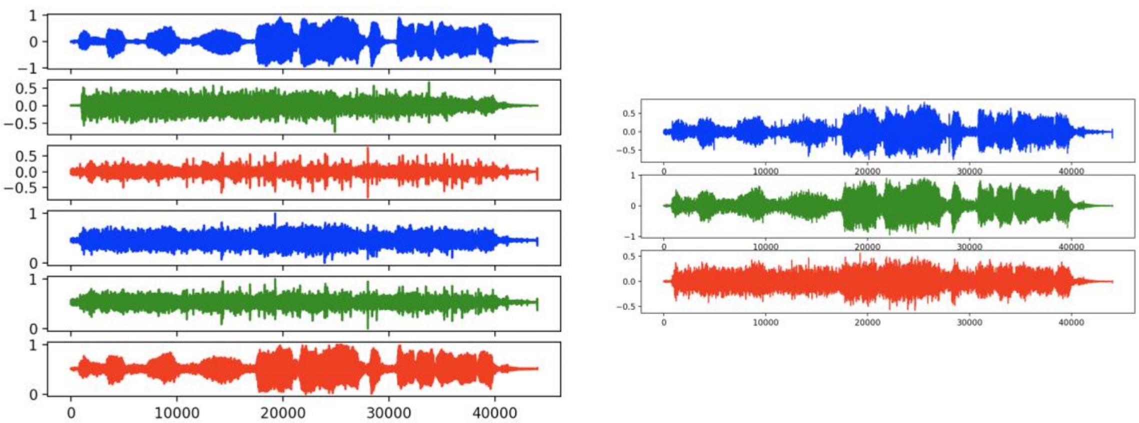

Figure 3(a): The plot on the left shows the original signals plotted on the top and the recovered signals on the bottom. Figure 1**(b)** shows the mixed signals generated from the original signals.

This analysis is further supported by Figure 4, illustrating the correct pairings of signals in bold. These numbers show that there is not as strong a correlation as shown in the test set implementation. It is likely that the reason for this is that a less than ideal combination of learning rate and number of iterations. Not all combinations were explored given how long (~20 minutes) each execution of the code took to run, but visually distinguishable results were developed with the proposed learning rate and number of iterations.

	Recovered 1	Recovered 2	Recovered 3
Original 1	0.56950981	0.08576051	**0.97720045**
Original 2	**0.75491936**	0.55943399	0.19612771
Original 3	0.43322259	**0.87451928**	-0.0702293

Figure 4: The Pearson coefficients between each of the full data vector combinations for the original and recovered signals. The bold highlights which of the recovered signals each original signal corresponds to.

---

4. Conclusion

The implementation of the independent component analysis algorithm was applied correctly to both data sets, with the application to the test set yielding satisfactorily accurate results in a short amount of time; this was largely due to the small size of the test data set, which allowed rapid execution of code and made it easier to find an ideal value for the learning rate and number of iterations. The algorithm implementation was identical for the full data set, but yielded significantly less accurate results. Since we showed that the algorithm was correctly implemented, this discrepancy can be attributed to the values chosen for the learning rate and number of iterations. Nonetheless, the recovered signals were recognizable and the process – mixing the signals, carrying out ICA, and recovering the signals – was relatively efficient and accurate.