Thomas Christinck
260601793

NEUR 603 Assignment 7: Unsupervised Learning

Part 1 –
The code was modified so that when the k-Means clustering code runs using 'kMeans;', we can interactively choose the number of distributions to use (2 or 3) and the k-value to use (2, 3, 4, or 5).

1.A. Two distributions were clustered with k = 3 (Figure 1). When k is erroneously set too high, the k-means algorithm will cluster the data excessively such that there is a number of incorrectly classified data points. We could imagine the extreme case where each data point is assigned to a unique cluster.

1.B. Three distributions were clustered with k = 2 (Figure 2). When k is erroneously set too low, clusters of datapoints that should be separated are grouped together erroneously. We could imagine the extreme case where all data points are assigned to the same cluster.

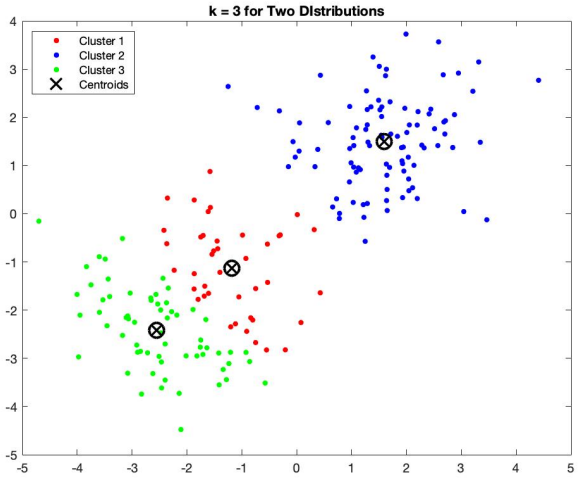

Figure 1 - Two distributions clustered with k = 3

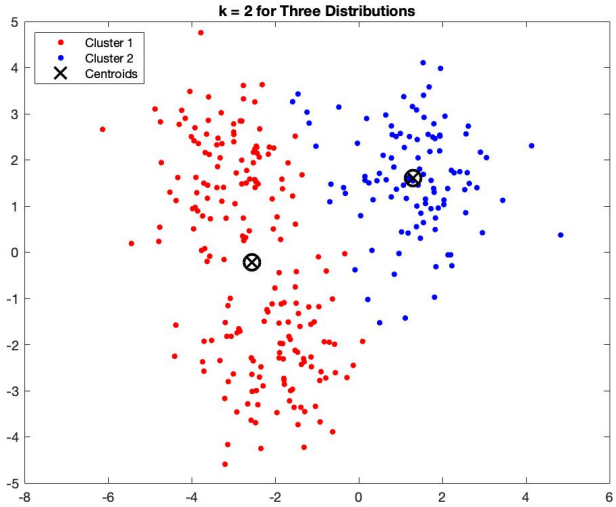

Figure 2 - Three distributions clustered with k = 2

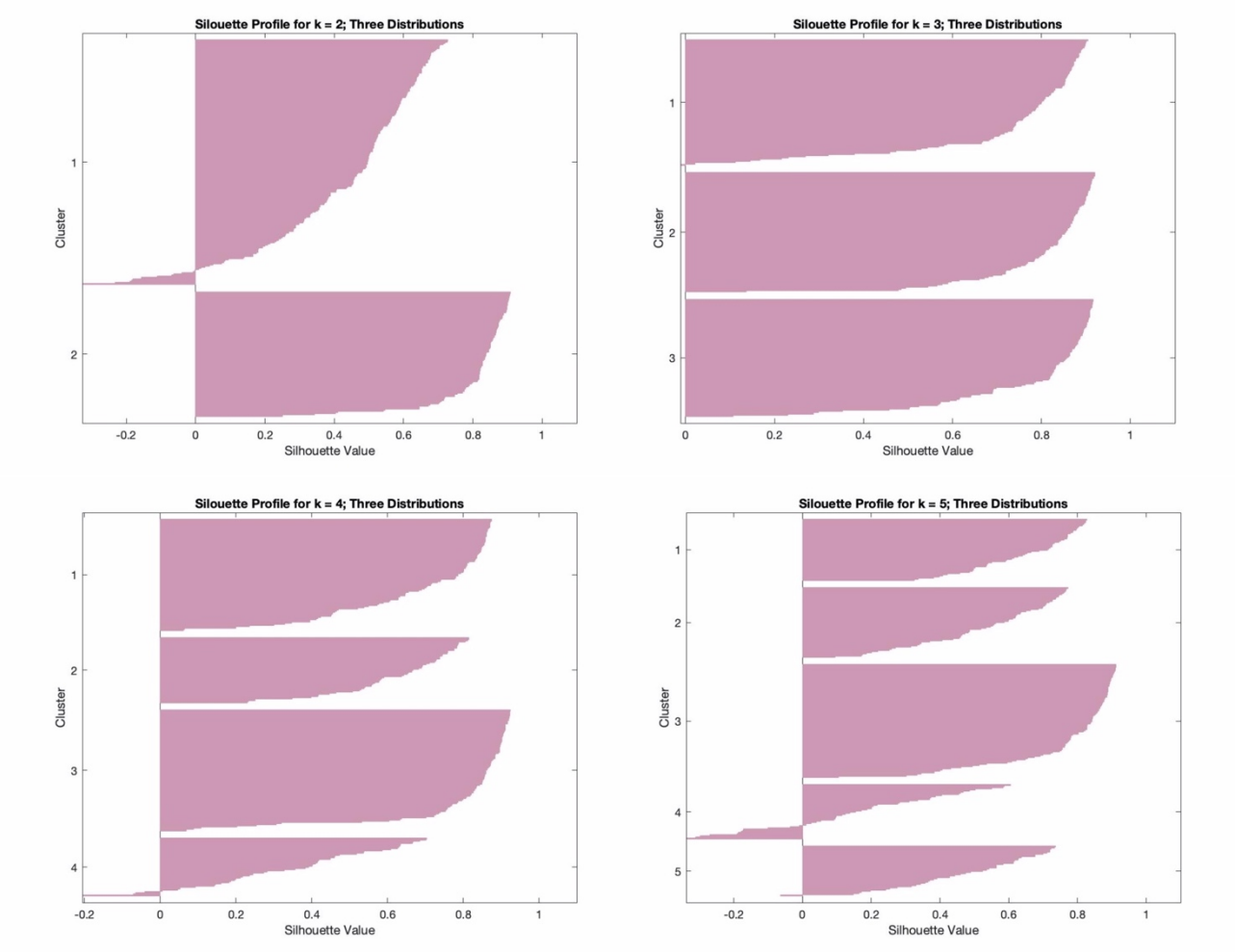

Figure 3 - Silhouette profiles for various k-values with three distributions

1.C. The silhouette measure was used to determine the ideal *k*, with *k* = {2,3,4,5} (Figure 3). From the figure, we can observe that for k-values that are too low *(k* = 2) or too high (*k* = 4, 5) there are a number of data points that are likely wrongly classified (assigned a negative silhouette value) and a large number of data points that exhibit little distinction from one cluster to another (assigned a near-zero silhouette value). The ideal *k*, *k* = 3, results in a broad silhouette profile with high values, suggesting many of the data points are assigned to the correct cluster.

Part 2 –

The code was modified so that when we run the Song et al. 2000 simulation code using 'Song2000_F4', we can interactively choose the initial synaptic conductance g.

2.A. When the initial conductance g is too low (e.g. g = 0.003, Figure 4) synapses are initially too weak to effectively drive the postsynaptic neuron. When the initial synaptic conductance is doubled (e.g. g = 0.006, Figure 5) the postsynaptic neuron is driven more frequently and has a relatively shorter initial latency. Further, the response of the postsynaptic neuron becomes

temporally sharper through STDP, as only a subset of presynaptic neurons is now able to drive postsynaptic action potentials.

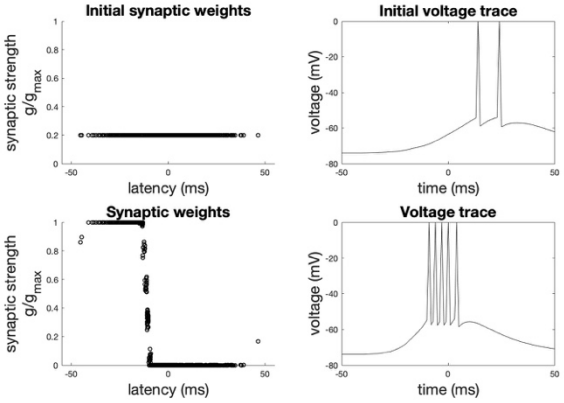

Figure 4 - STDP illustration with initial conductance g = 0.003

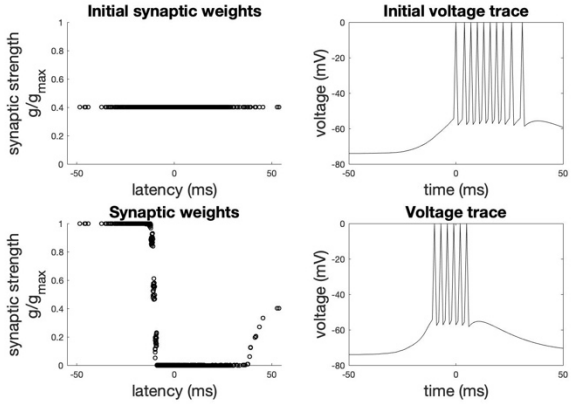

Figure 5 - STDP illustration with initial conductance g=0.006

2.B. A neuron with large initial synaptic conductance *g (g = g_{max})* can be stabilized through STDP as synapses with high-latency input neurons will experience LTD and these synaptic weights will approach 0 (Figure 6). The postsynaptic neuron remains sensitive to low-latency inputs and input neurons that tend to fire before post-synaptic action potentials will have synaptic weights that remain at g_{max}.

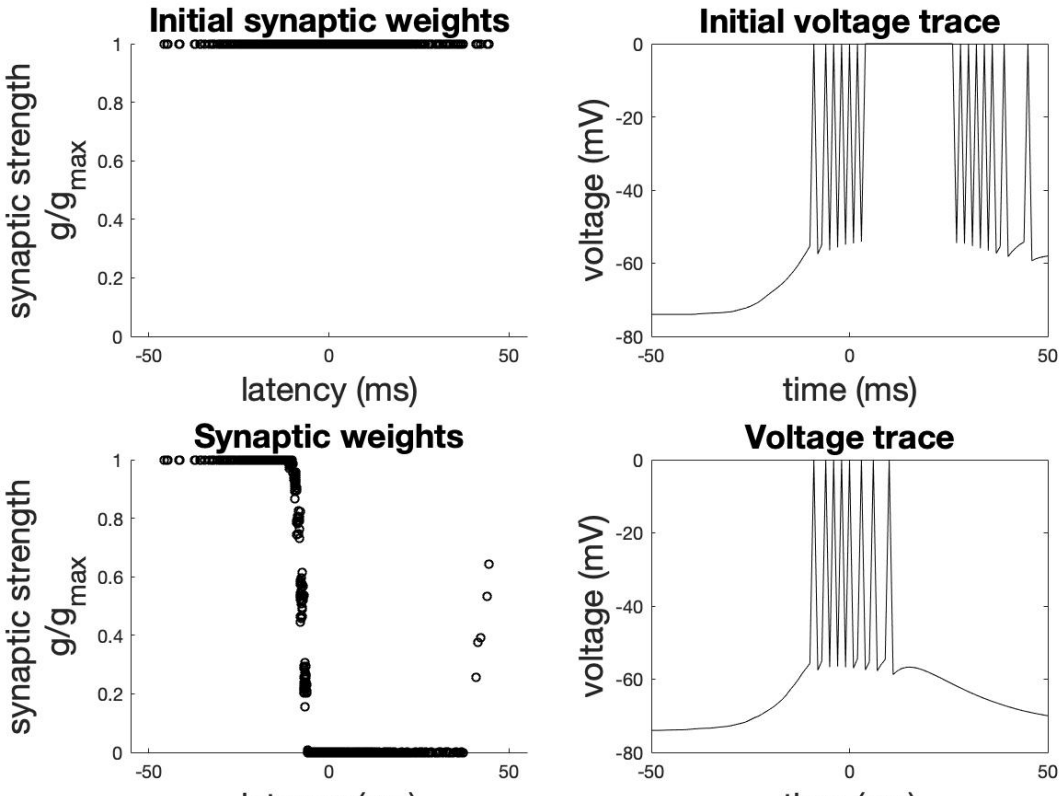

Figure 6 - STDP illustration with large initial synaptic conductance g (g = g_{max})

2.C. Unsupervised learning involves recognizing features or patterns in unlabelled data, and unsupervised classification involves classifying data based on these extracted features or patterns. Figure 7 (Fig 1 of Song et al., 2001) illustrates STDP separating data based on correlation. Input neurons that fire in an uncorrelated manner (Fig 7, top right) are assigned weights randomly. Synapses that are firing in a correlated manner (Fig 7, bottom left) are strengthened and synapses firing in an uncorrelated manner are weakened. The neuron is able to classify unlabelled data as correlated or uncorrelated through STDP (performing unsupervised classification).

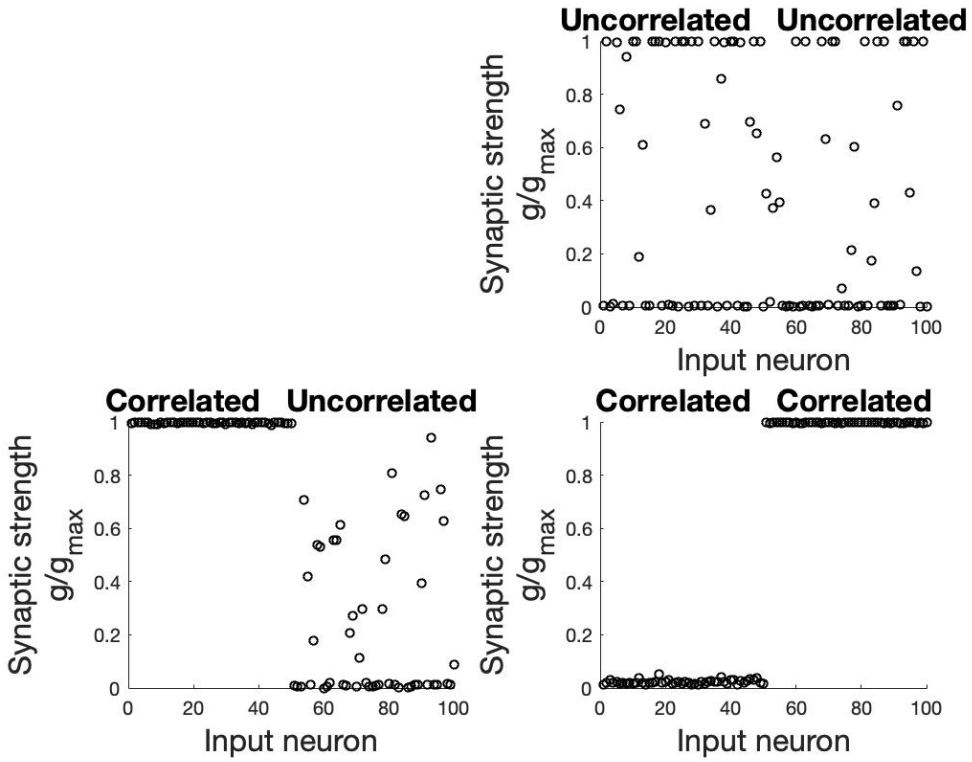

Figure 7 - Equilibrium distributions of synaptic weights for uncorrelated inputs, correlated/uncorrelated inputs, and correlated/correlated inputs (where input neurons are divided into two correlated groups that are uncorrelated with one another)

Part 3 –

The code was not modified for part 3; only the text file was altered to add or change attractor states. The code was run (i.e. hopfield_net.m) using the specified parameters in the responses.

3.A. The noise level used to corrupt memory states was increased to 50; at noise levels from 20 to 60, we see recovered patterns that are mostly different from the original patterns (Figure 8). The recovered pattern is closest in Hamming distance to the corrupted pattern in this case, but far from the original pattern. The noise levels were further increased to 100; at noise levels greater than approximately 60, the noise generation becomes saturated and we see recovered patterns that are the inverse of the originals (Figure 9).

3.B. The text file was altered so that the letter ‘C’ is no longer one of the attractor states, and ‘8’ is stored as a new attractor state. The attractor state ‘8’ is very similar to the attractor state ‘B’. The original patterns ‘B’ and ‘8’ will thus converge to an attractor state that is some hybrid of the two attractor states ‘8’ and ‘B’ (Figure 10).

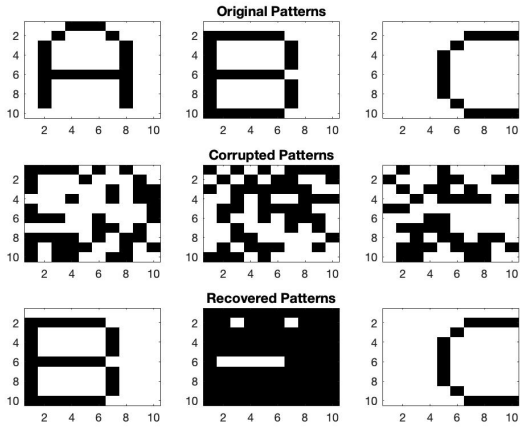

Figure 8 - Hopfield network of size 100 with noise level 50 (note recovered patterns are different from the originals)

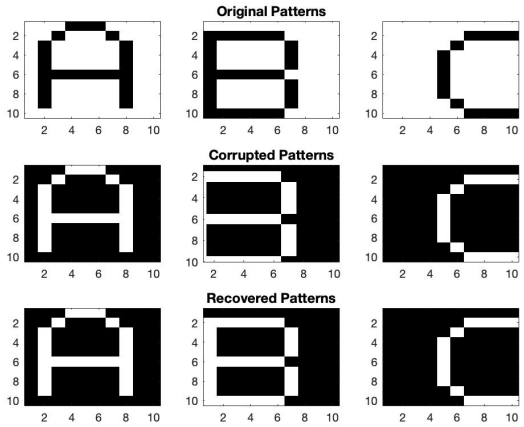

Figure 9 - Hopfield network of size 100 with noise level 100 (recovered patterns are the inverse of the original states)

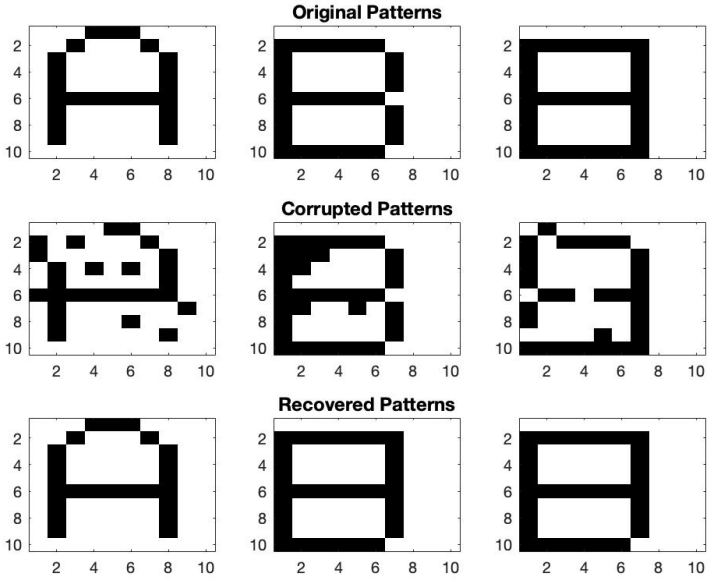

Figure 10 - Hopfield network of size 100 with noise level 10 (note original patterns 'B' and '8' are not recovered; recovered patterns are some hybrid of the two states)

3.C. The Hopfield network had 9 well-separated memories added to it; however, with the 6 added memories, the memory seems to be easily corrupted and spurious additional memories seem to be added to the memory, likely derived from the desired memories (Figure 11). Theoretically, the storage limit of the network should be given by *P* in the following expression, where *N* is the network size:

$$P = \frac{N}{4\log(N)}$$
$$P = 12.5$$

Since we are trying to store 9 memories (close to the maximum theoretical storage) we'll expect to see some interference as the memory is approaching full theoretical capacity.

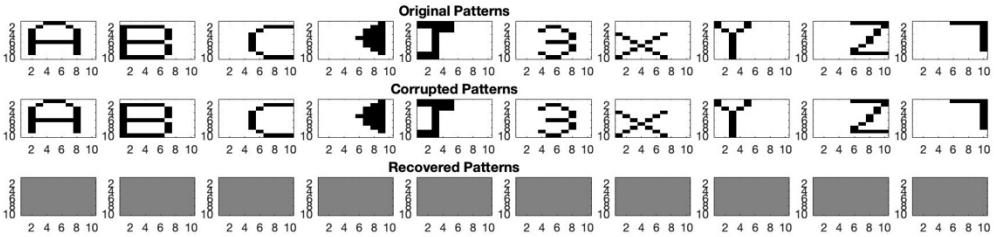

Figure 11 - Hopfield network with 9 original patterns; the retrieved patterns seem to be spurious additional memories derived from the desired memories

3.D. The Lyapunov function was used to show numerically that the network is attracted to a lower energy state as it converges. Here, the Hopfield network converges to state 55 (Figure 12). The Hopfield network can be used as a classifier, as it uses attractor states to recover memories. We could imagine a scenario where the Hopfield network memorizes a training set, and then classifies new unseen examples based on what memory the new examples are associated with.

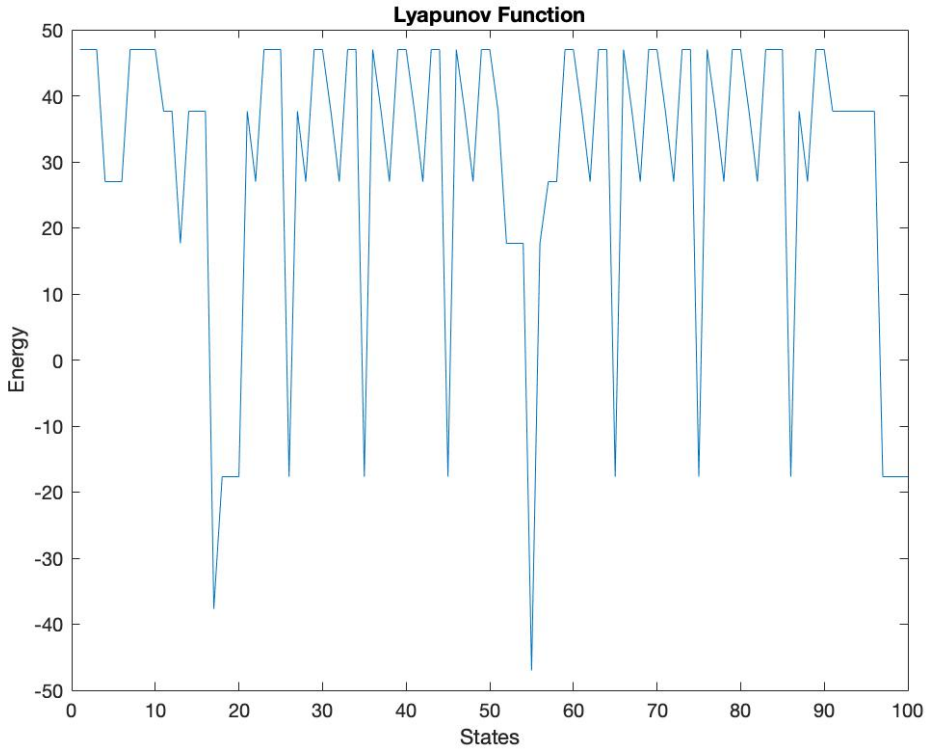

Figure 12 - Lyapunov function illustrating how the Hopfield network is attracted to a low-energy state