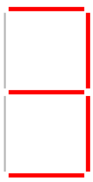
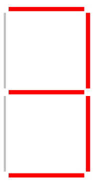
James O'Reilly

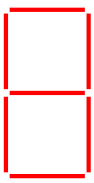
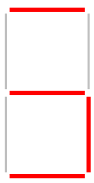
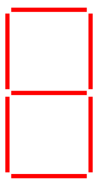
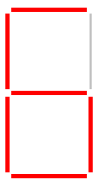
Weight matrix

$$W = \begin{pmatrix} 0.0 & 0.33 & -0.33 & 1.0 & 0.33 & 0.33 & 1.0 & -0.33 & 1.0 & 0.33 & -0.33 \\ 0.33 & 0.0 & -1.0 & 0.33 & 1.0 & -0.33 & 0.33 & -1.0 & 0.33 & 1.0 & 0.33 \\ -0.33 & -1.0 & 0.0 & -0.33 & -1.0 & 0.33 & -0.33 & 1.0 & -0.33 & -1.0 & -0.33 \\ 1.0 & 0.33 & -0.33 & 0.0 & 0.33 & 0.33 & 1.0 & -0.33 & 1.0 & 0.33 & -0.33 \\ 0.33 & 1.0 & -1.0 & 0.33 & 0.0 & -0.33 & 0.33 & -1.0 & 0.33 & 1.0 & 0.33 \\ 0.33 & -0.33 & 0.33 & 0.33 & -0.33 & 0.0 & 0.33 & 0.33 & 0.33 & -0.33 & -1.0 \\ 1.0 & 0.33 & -0.33 & 1.0 & 0.33 & 0.33 & 0.0 & -0.33 & 1.0 & 0.33 & -0.33 \\ -0.33 & -1.0 & 1.0 & -0.33 & -1.0 & 0.33 & -0.33 & 0.0 & -0.33 & -1.0 & -0.33 \\ 1.0 & 0.33 & -0.33 & 1.0 & 0.33 & 0.33 & 1.0 & -0.33 & 0.0 & 0.33 & -0.33 \\ 0.33 & 1.0 & -1.0 & 0.33 & 1.0 & -0.33 & 0.33 & -1.0 & 0.33 & 0.0 & 0.33 \\ -0.33 & 0.33 & -0.33 & -0.33 & 0.33 & -1.0 & -0.33 & -0.33 & -0.33 & 0.33 & 0.0 \end{pmatrix}$$

Test 1


0 -4.3

3 -16.3

Test 2


0 -0.3

7 -7.0

2 -8.3

6 -23.0

1 Some Extra Stuff

1.1 Graphing the Energy function

Below I have included the output for the test patterns from my Hopfield network that did not fit the submission layout. At each iteration I append both the energy and the current state to a list which is then printed once the pattern has converged. The list of energies is used to plot the change in energy over time and shows that the energy is never increasing. The energy is not always decreasing as the pattern may converge to an unstable attractor state where it is flipping between two configurations.

```
hopRunfield(test1, weight_matrix)
```

```
Attractor found after 1 iterations.
List of States: [[1, -1, 1, 1, -1, 1, 1, -1, -1, -1, -1], [1, -1, 1, 1, -1, 1, 1, 1, 1, -1, -1]]
Attractor State: [1, -1, 1, 1, -1, 1, 1, 1, 1, -1, -1]
List of energies at each iteration: [-4.2999999999999998, -16.300000000000001]

  _
  _|
  _|
3
```

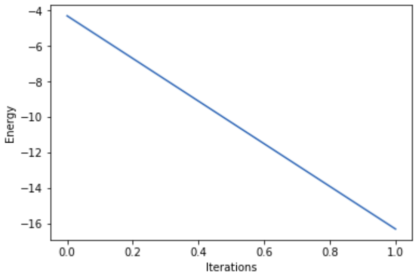


Figure 1: Output from first test configuration.

```
hopRunfield(test2, weight_matrix)
```

```
Attractor found after 3 iterations.
List of States: [[1, 1, 1, 1, 1, 1, 1, -1, -1, -1, -1], [1, -1, -1, 1, -1, 1, 1, 1, 1, 1, -1], [1, 1, 1, 1, 1, 1,
1, -1, 1, -1, -1], [1, 1, -1, 1, 1, 1, 1, -1, 1, 1, -1]]
Attractor State: [1, 1, -1, 1, 1, 1, 1, -1, 1, 1, -1]
List of energies at each iteration: [-0.29999999999999999, -7.0, -8.3000000000000007, -23.0]

 _
|_
|_|
6
```

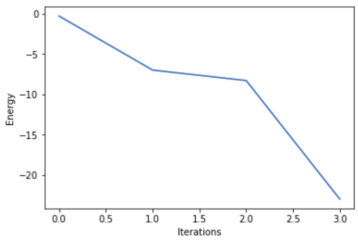


Figure 2: Output from second test configuration.

1.2 Unstable Attractor States

Both of the test patterns converge to stable attractor states. However, this is not the case for every initial configuration. To test this, I updated my Hopfield network function to scan the list of states at each iteration and test whether the current state had been reached before. If the current state had been reached before, then I test when that state was last reached. If the state was reached on the previous iteration, then the pattern has converged to an unstable attractor state. If the state is the same as the state reached two iterations ago, then the pattern has converged to a set of unstable attractor states and will constinuously flip between them.

I then generated an array of every possible initial configuration (there are $2^11 = 2048$ such configurations) and ran my Hopfield network on each configuration. Below I have included an example of a configuration that reaches a set of unstable attractor states.

```
unstable attractor states found after 3 iterations.
List of States: [[-1, -1, -1, -1, -1, -1, 1, -1, 1, -1, -1], [1, -1, 1, 1, -1, 1, -1, 1, -1, 1, 1], [-1, -1, 1, -1,
-1, 1, 1, 1, 1, -1, -1], [1, -1, 1, 1, -1, 1, -1, 1, -1, -1, -1]]
Attractor States: [-1, -1, 1, -1, -1, 1, 1, 1, 1, -1, -1] , [1, -1, 1, 1, -1, 1, -1, 1, -1, -1, -1]
List of energies at each iteration: [5.0, 1.0, -12.300000000000001, -12.300000000000001]
```

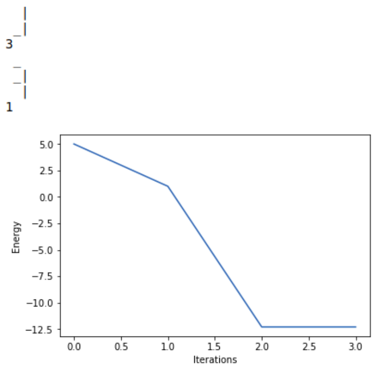


Figure 3: Output for unstable attractor states.

Note that the energy does not change between the two final states, as can be seen in the graph.

1.3 Distance from Training Patterns

I noticed when looking through the final configurations that they were mostly one or two bit flips away from one of the configurations that were used to train the weight matrix. It would be interesting to calculate the Hamming distance from each attractor state to the training configurations. As to why they seem to converge to a state very near to these training states, I don't know. I guess we're effectively crafting an 11-dimensional landscape when we train the weight matrix, and as the gradients in this landscape are determined by our training configurations, the vast majority of our initial configurations will tend toward these attractor states. In some sense, I view the unstable attractor states as little "pockets" near the training configurations that are rarely reached. If I had a bit more time I'd look into this a bit more deeply, I might get back to it later.