Analysis Of Self-Organizing Maps

Using A Java Based Implementation

Matthew A. Letter

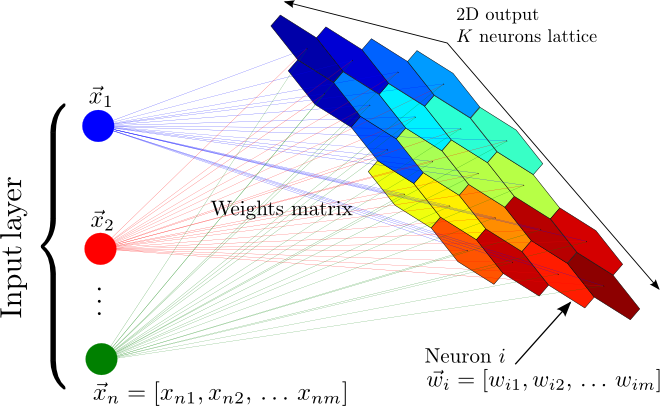
November 13th 2014

Neural Networks 547

**Self-Organizing Map**

Self Organizing Maps (SOM) are based on competitive learning; where output neurons of the network compete among themselves to be activated or fired.1 The result of which is that only one neuron per group is on at any one time.1 In this study, the Self-Organizing map consists of a 2 dimensional matrix with a varying number of neurons. The neurons become selectively tuned to various input patterns during the course of the competitive learning process.1

Java was used to code the SOM network. A Graphical user interface was used to create a real-time updating graph of the weight changes laid out on the inputs. This was done in order to see “weight grouping” and to gauge weather the SOM fit the data by the end of a run. There were 2 given inputs to the SOM with 1600 data points that were pulled at random. 1600 data point pulls was considered one iteration. Below is a pictorial schema for the self organizing map system.2



In the following tables convergence is defined by when the RMS magnitude of the weight change, after one iteration, is less than 0.00001. The following variables and values are set unless otherwise noted based on hayken.1 M=5,tao = 1000/log(sigma0), sigma0=M/2, eta=0.01, and all initial weights are randomly chosen between -0.5 and 0.5.

Tables 1: Investigates different learning rates on SOM convergence. It appears that the lower the learning rates the faster the convergence.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Number of SOM neurons** | **# Of iterations** | **Learning rate** | **Was Shuffled** | **Converged?** |
| 5X5 | 1642 | 0.2 | yes | yes |
| 5X5 | 1014 | 0.1 | yes | yes |
| 5X5 | 796 | 0.05 | yes | yes |
| 5X5 | 646 | 0.01 | yes | yes |
| 5X5 | 302 | 0.001 | yes | yes |

Table 2: Investigates varying the number of SOM neurons effect on convergence. It appears that the larger M is the longer it takes to converge. Notice the anomaly for 50X50.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Number of SOM neurons** | **# Of iterations** | **Learning rate** | **Was Shuffled** | **Converged?** |
| 5X5 | 646 | 0.01 | yes | yes |
| 10X10 | 1686 | 0.01 | yes | yes |
| 20X20 | 2508 | 0.01 | yes | yes |
| 50X50 | 286 | 0.01 | yes | yes |
| 100X100 | 10000 | 0.01 | yes | no |

Table 3: Investigates varying tao and how it affects convergence. It appears that the smaller tao gets the faster the system converges(this doesn’t mean the system is a good fit for the data jus that it converge to our criterion).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Number of SOM neurons** | **# Of iterations** | **Learning rate** | **Was Shuffled** | **Tao** | **Converged?** |
| 5X5 | 128 | 0.01 | yes | 10000/log2.5 | yes |
| 5X5 | 560 | 0.01 | yes | 1000/log2.5 | yes |
| 5X5 | 96 | 0.01 | yes | 100/log2.5 | yes |
| 5X5 | 20 | 0.01 | yes | 10/log2.5 | yes |
| 5X5 | 4 | 0.01 | yes | 1/log2.5 | no |

Table 4: Investigates varying sigma0 and how it affects convergence. It appears that the smaller sigma0 gets the faster the system converges(this doesn’t mean the system is a good fit for the data jus that it converge to our criterion).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Number of SOM neurons** | **# Of iterations** | **Learning rate** | **Was Shuffled** | **Sigma0** | **Converged?** |
| 5X5 | 652 | 0.01 | yes | M | yes |
| 5X5 | 542 | 0.01 | yes | M/2 | yes |
| 5X5 | 92 | 0.01 | yes | M/5 | yes |
| 5X5 | 1 | 0.01 | yes | M/10 | yes |
| 5X5 | 1 | 0.01 | yes | M/20 | yes |

Based on the tables above a few interesting cases have been chosen for graph displays and RMS magnitude change as well as an “ideal case”. First the ideal case will be presented when eta=0.01, tao = 1000/logsima0, sigma0 = M/2, and M = 30X30.

Figure 1: Iteration 0 graph of an “ideal” setup based off the previous tables. As you can see the weights are randomly spread over a certain range.

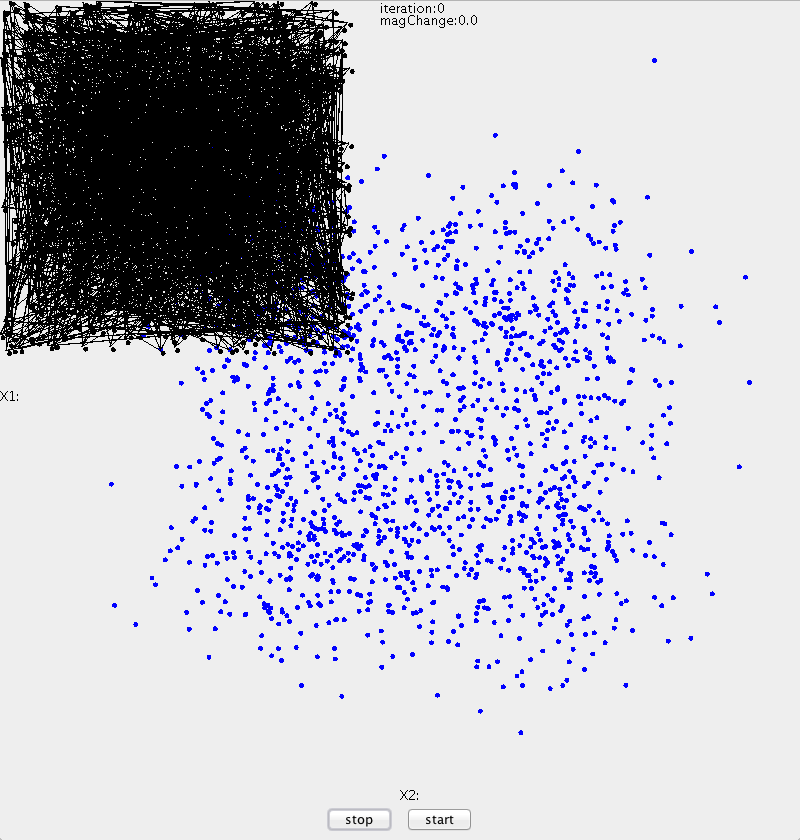


Figure 2: Iteration 1 graph of an “ideal” setup based off the previous tables. The weights immediately snap to the center of the data field.

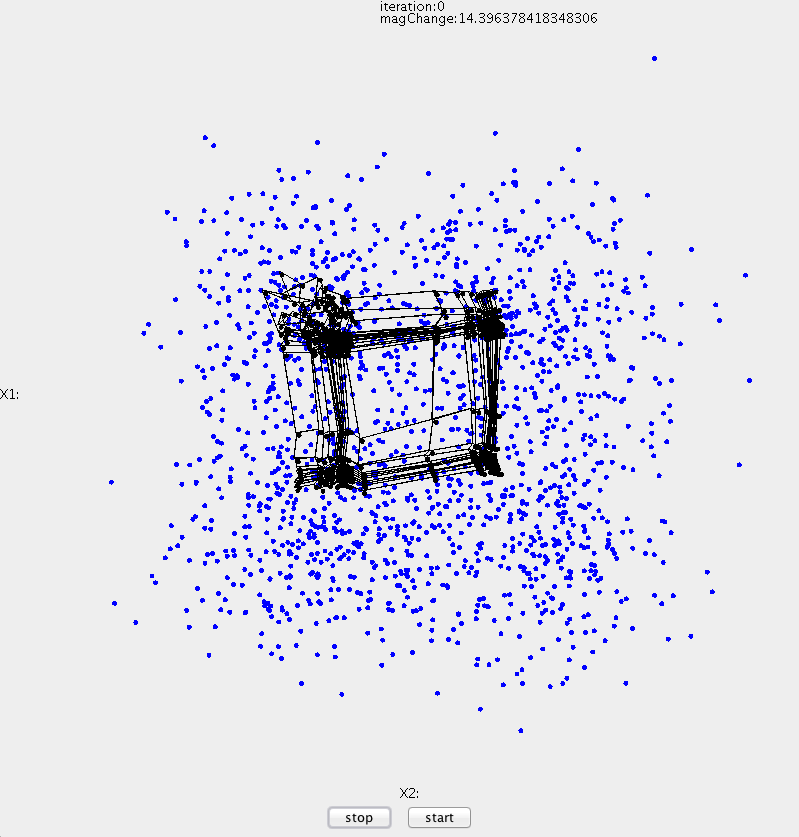


Figure 3: Iteration 300 graph of an “ideal” setup based off the previous tables. Based on the picture its appears as if the weight map matches the data fairly well.

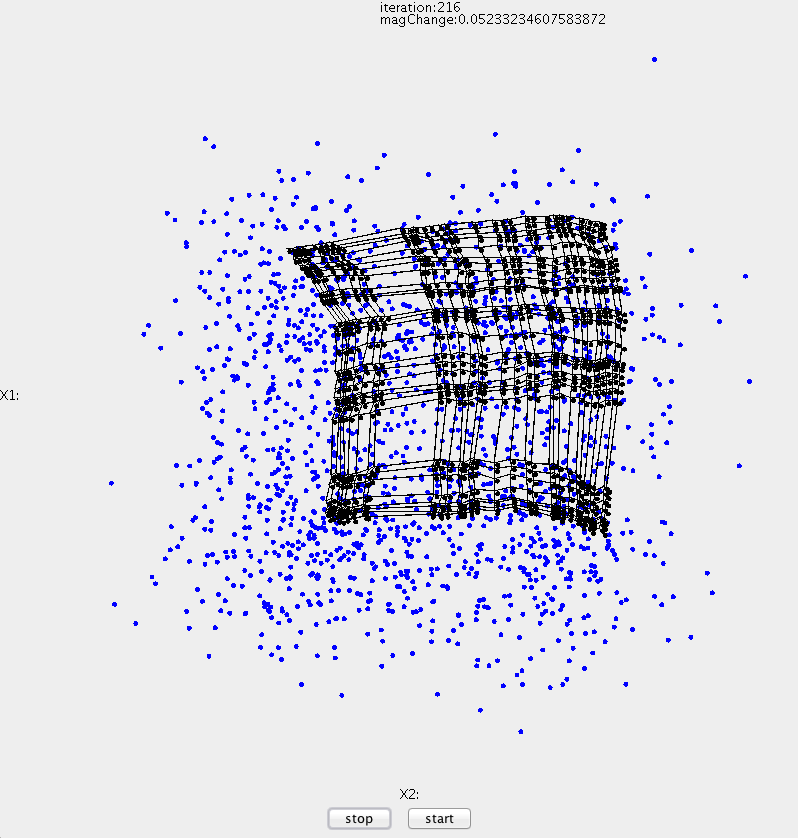


Figure 4: Iteration 972 graph of an “ideal” setup based off the previous tables. Letting the SOM lattice keep running past the point of apparent convergence yielded this interesting graph.

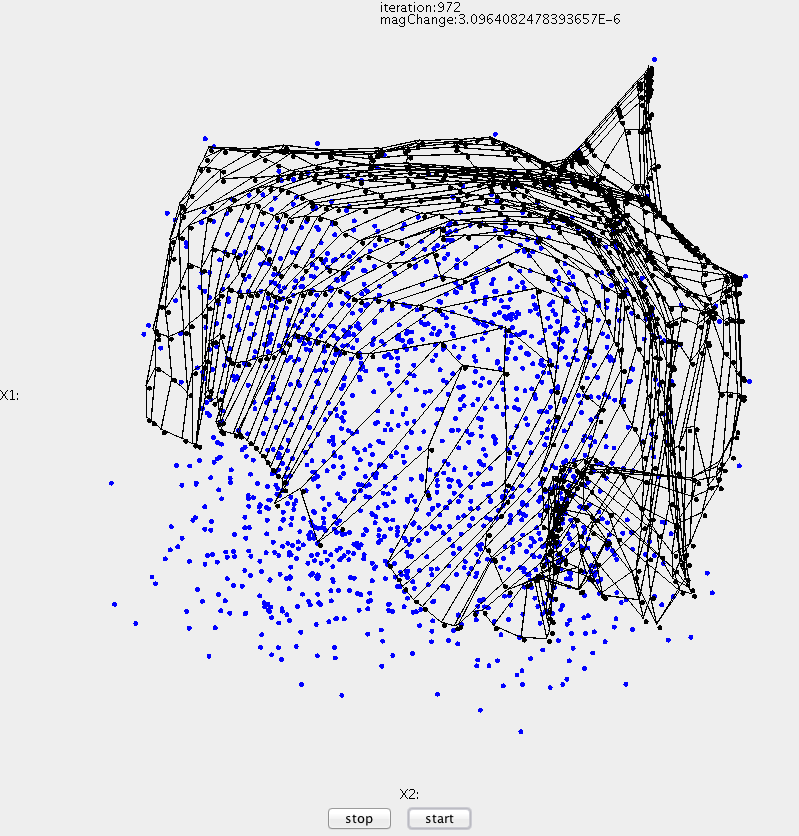


Figure 5: RMS magnitude change graph for figure 3 for the first 50 iterations. It appears as though there is a large initial jump over to the data and then the rest of the iterations are small jump to better match the data

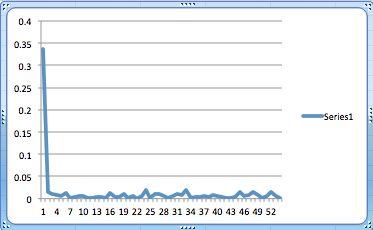


Figure 6: From table 1 a learning rate of 0.001 was chosen for visualization and an M of 10.

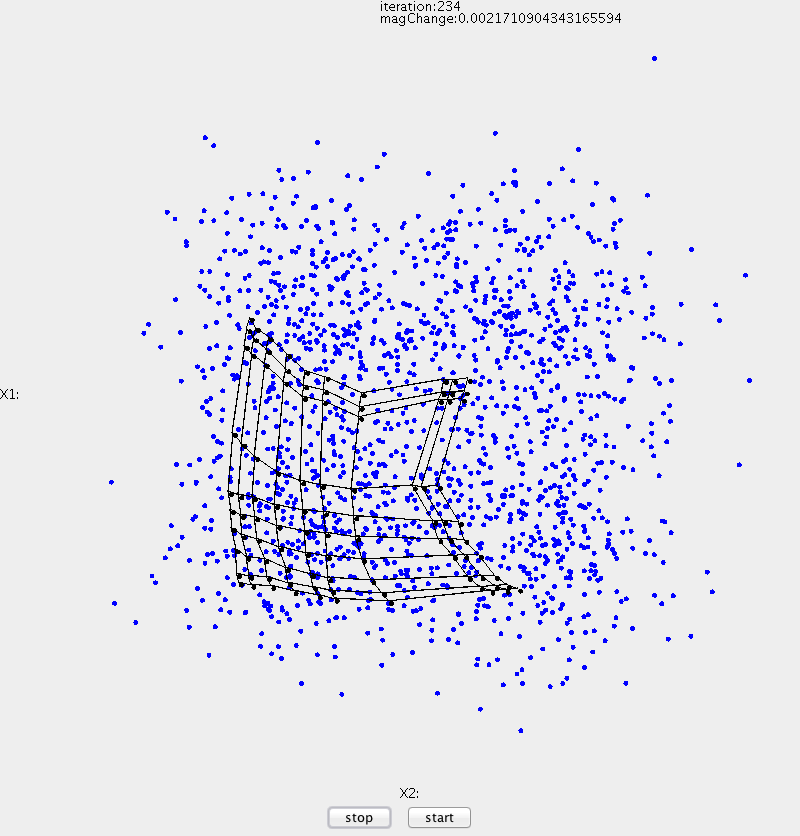


Figure 7: From table 2 an M of 50 was chosen. It seems the map fits the data fairly well.

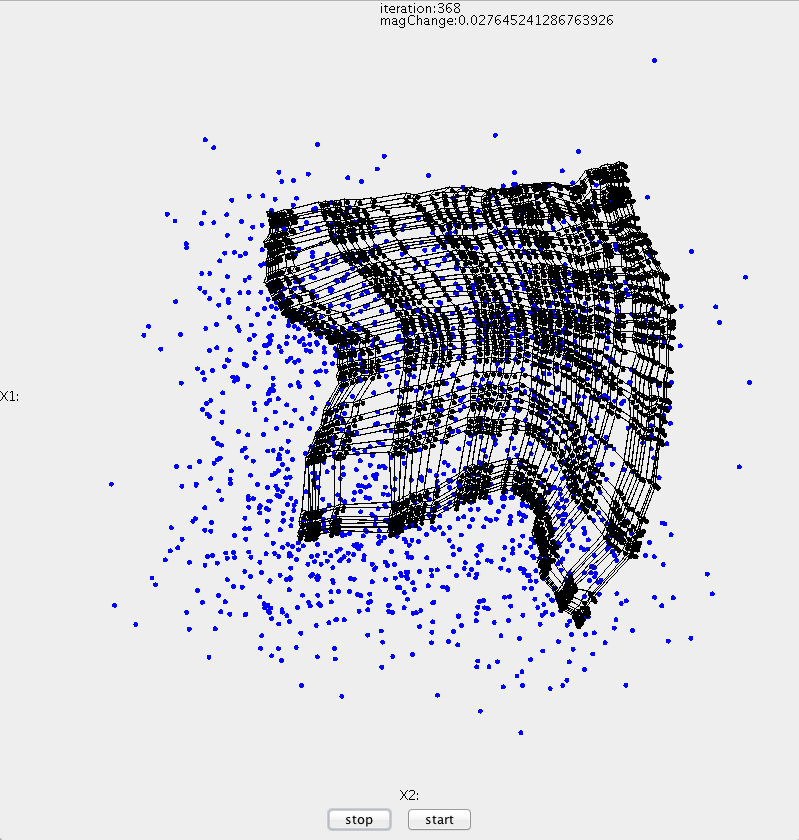
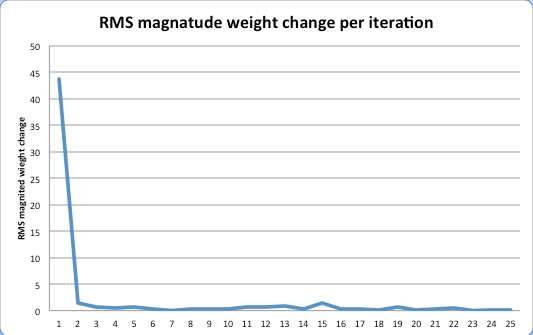


Figure 8: RMS magnitude weight change per iteration for figure 7.



**Results and Conclusions**

Citations

1. Haykin, Simon S. *Neural Networks: A Comprehensive Foundation*. Upper Saddle River, NJ: Prentice Hall, 1999. Print.
2. "Self Organizing Maps." *Self Organizing Maps*. N.p., n.d. Web. <http%3A%2F%2Fwww.stuartreid.co.za%2Fartificial-intelligence-and-statistics-principal-component-analysis-and-self-organizing-maps%2F>.

Source Code