Neural Networks
Group 04
Andrea Jemmett (2573223)
Bas van der Drift (1707507)

Assignment 6 - Self-Organising Maps

The main Matlab script for this assignment can be found attached as ‘ass6.m’.

One thousand coordinates are randomly generated by the ‘generate_star.m’ script. These coordinates all fall within the shape of a five-pointed star. A two-dimensional Self-Organising Map (SOM) can then be used to fit the shape of the star. In this exercise we will attempt to create a map that fits the star shape as perfectly as possible.

Starting out the size of the lattice of the SOM needs to be determined. Because of shape of the star, we decided to start testing some square maps. Before plotting anything, it would make sense that a 2 x 2 lattice is not able to fit the data well: the five-pointed star is a shape too complicated to capture with any sort of accuracy within four coordinate points. In order to try to get a feel for the minimum lattice size, a plot was made of a variety of lattice sizes. For this 1000 iterations were used, the final neighbourhood size had a value of 1 and the learning rate started out at 0.9, gradually decreasing to 0.05 during the iterations. The initial neighbourhood size was set dynamically for each lattice size by using the following formula:

$$initial\ size \approx (\sqrt{x^2 + y^2}) / 2 \ ,$$

where x and y are the respective dimensions of the lattice as suggested by the teacher. For example, in case of a lattice of 4 x 4, the initial neighbourhood size would be 3, whereas for a lattice of 15 x 15 it would be 11.

Figure 1 displays the fit of different sizes of square lattices onto the five-pointed star. The original data is plotted in blue. The neurons’ weights are then plotted in yellow with red lines defining the lattice connecting those neurons.

As expected, smaller lattice sizes do not perform very well on this dataset. A larger map is required in order to fully map the five points of the star. In our opinion, a minimum lattice size of 16 x 16 is able to map the data well, when looking at the plots.

The evaluation of the performance of a map is not trivial, since the standard error measurements do not take into consideration how well the characteristics of the training data have been preserved. For example, a 2 x 2 lattice could have zero error when error is defined as a lattice point falling outside of the shape of the star, while not mapping well to the data at all. Conversely a 25 x 25 lattice could have a relatively high error while still ‘filling out’ the shape of the star nicely. In this case we would prefer the lattice with a higher error measurement that still maps the data well. Because of this we will ‘measure’ performance simply by looking at plots. Not the most scientifically accurate method, but sufficient for our purposes.

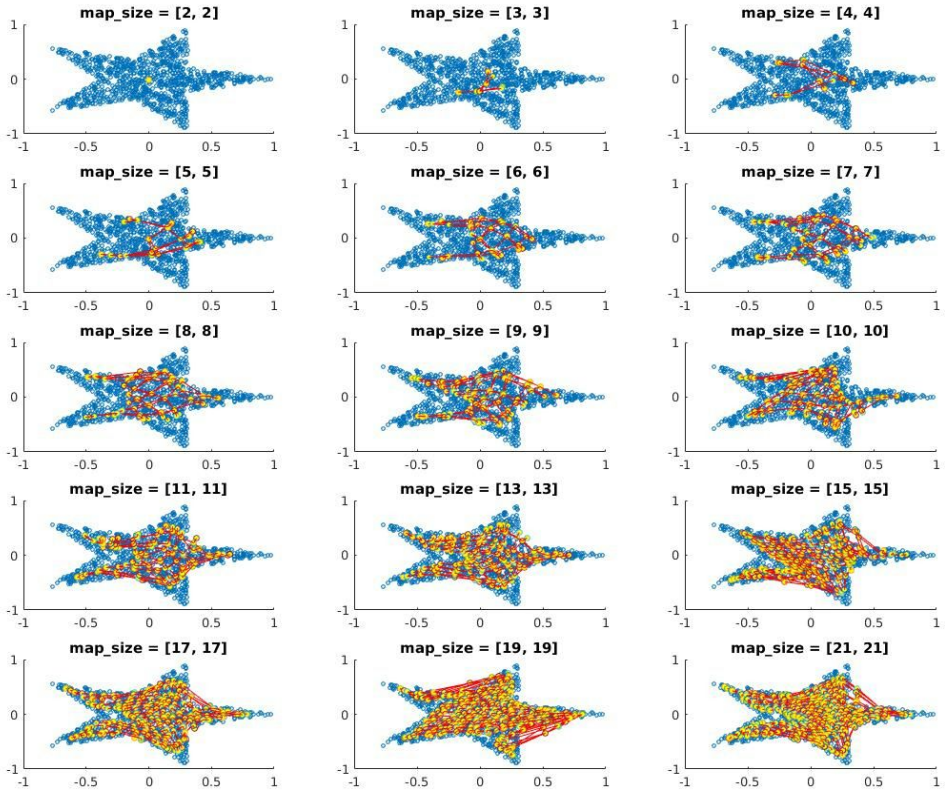

Figure 1: Star-shaped data with fitted SOM for different map sizes

Then we decided to try a SOM with a one dimensional map, namely 1 x i where i = { 5, 10, 25, 50, 100, 200, 400, 800, 1000 }; we used these numbers because we thought that there wouldn't have been that large of an improvement in the resulting map for a higher number of neurons than the number of data points. The results are visible in figure 2. As shown, the lattice covers the complete star shape very well, reaching all the extremities starting from 50 neurons; then, approaching the number of training points (1000), it's noticeable that the lattice spreads uniformly over the training data.

Then for completeness' sake we decided to test a SOM with map size 3 x i where i = { 5, 10, 25, 50, 100, 200, 400, 800, 1000 }. In our expectations this would not have been a good candidate because is unlikely to fit a lattice of that shape on a "squared" shape like the star. Figure 3 shows that we were right. We show that a lattice of this kind would not reach the extremities of the star and all neurons stay fairly centered.

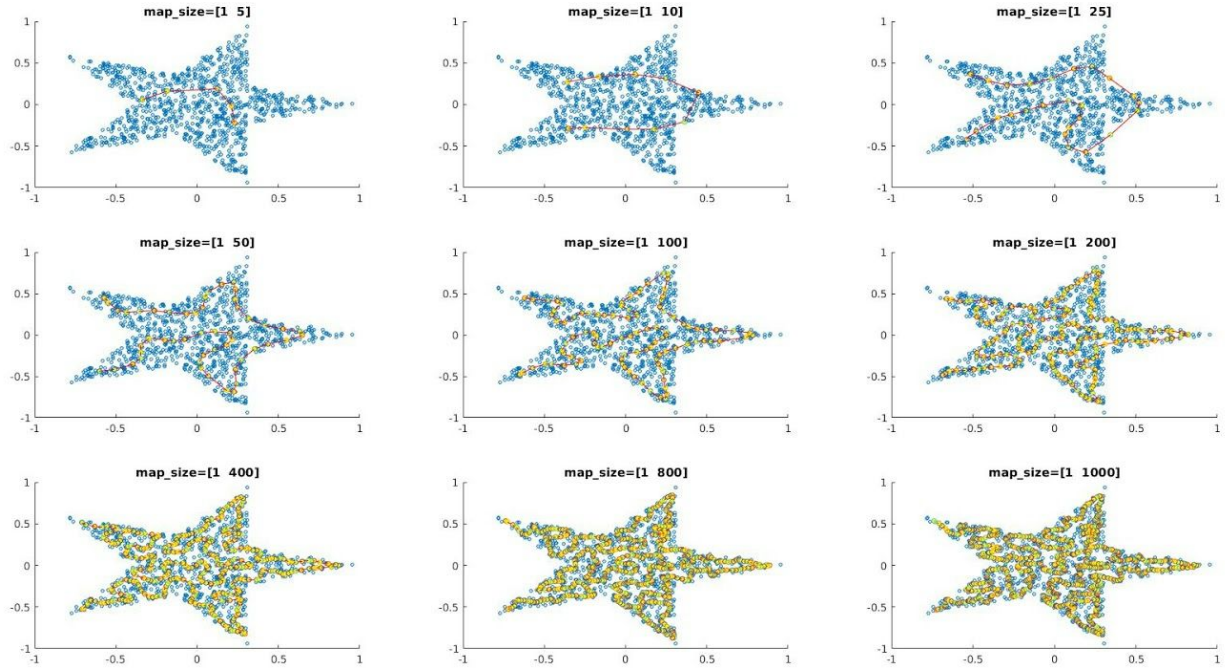

Figure 2: One dimensional lattices

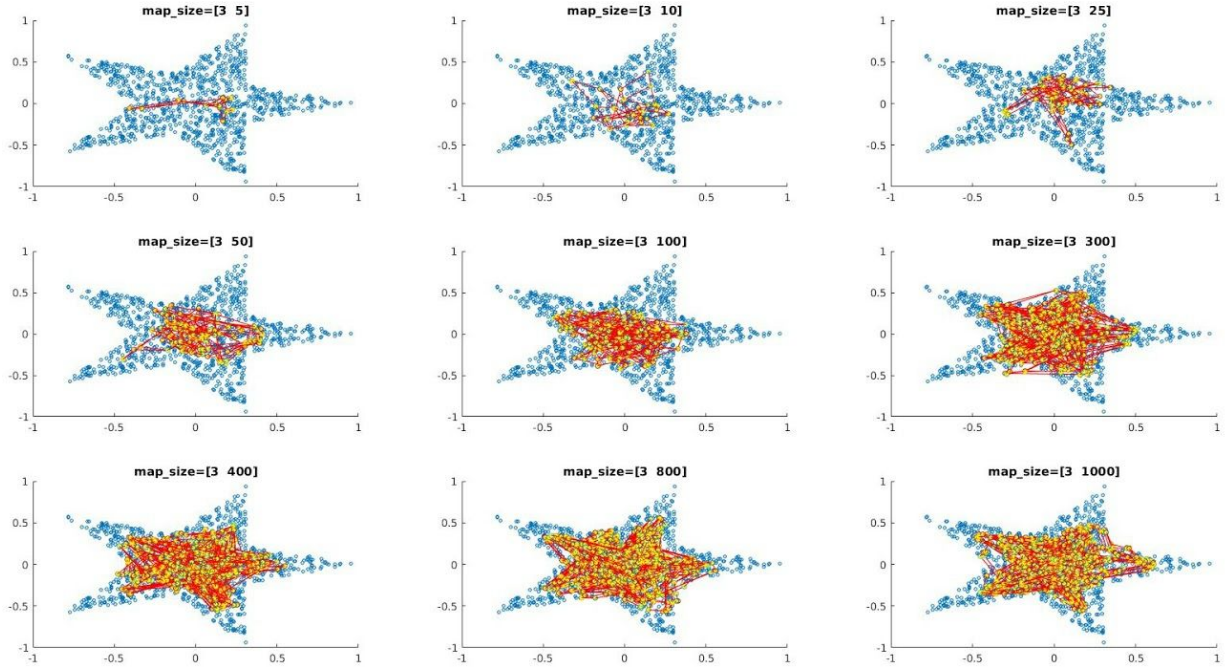

Figure 3: Lattices of size 3 x i

The training of a SOM is composed of two phases: ordering and convergence. To illustrate the entire training process we created the plot in Figure 4 for a map of size 16 x 16.

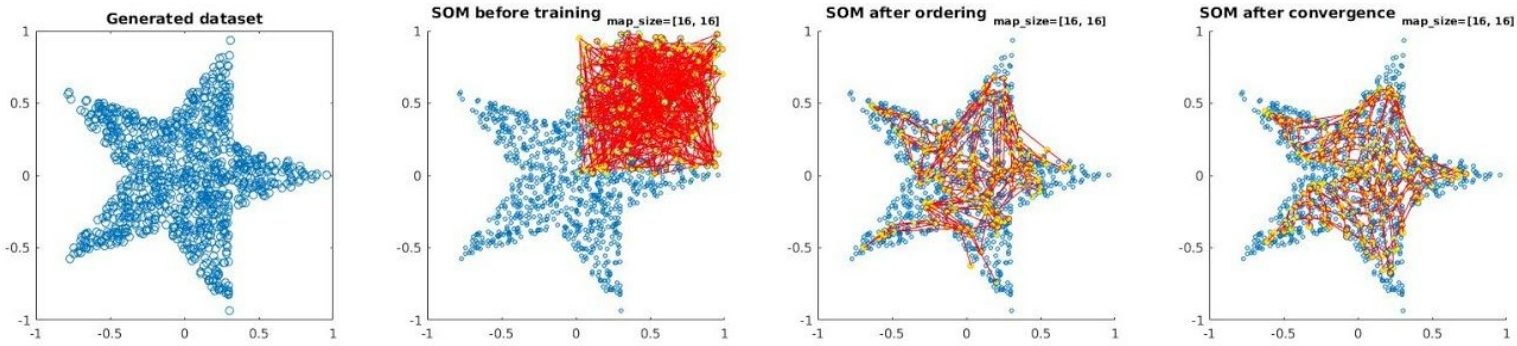

Figure 4: The ordering and convergence phases of training a SOM

Initially all the neurons' weights are initialized randomly. Then there's the ordering phase where the learning rate is set high and lasts for few iterations (learning rate from 0.9 to 0.5 for 50 iterations in the figure) to quickly allow neurons to spread along the shape. The last phase is the convergence phase where the learning rate is lowered and more iterations are performed to allow the lattice to align better with the training pattern (for the SOM in figure 4 was used a learning rate from 0.5 to 0.05 for 1000 more iterations). From the figure the training phases are clearly evident.

In order to gain some more insight about the influence of the learning rate of the maps, we varied both the initial learning rate and the final learning rate. In SOMs the initial learning rate is often set to a relatively high value, decreasing as iterations are performed. This necessity of this is easily imagined when thinking about how the lattice 'folds over' the training data. At first the shape of the data has to be scouted and populated, which is more easily done with a high learning rate, while in the later iterations we would want the lattice to spread across the data in a well-distributed manner, which requires a lower learning rate.

Due to this we would expect the SOM to perform poorly when either a low initial learning rate or a high final learning rate is used. In figure 5 the results of varying the initial learning from 0.9 to 0.1 are shown, while keeping the final learning rate constant at 0.05. A 16 x 16 lattice was used. As shown in the figure, varying the initial learning rate doesn't affect the resulting map very strongly.

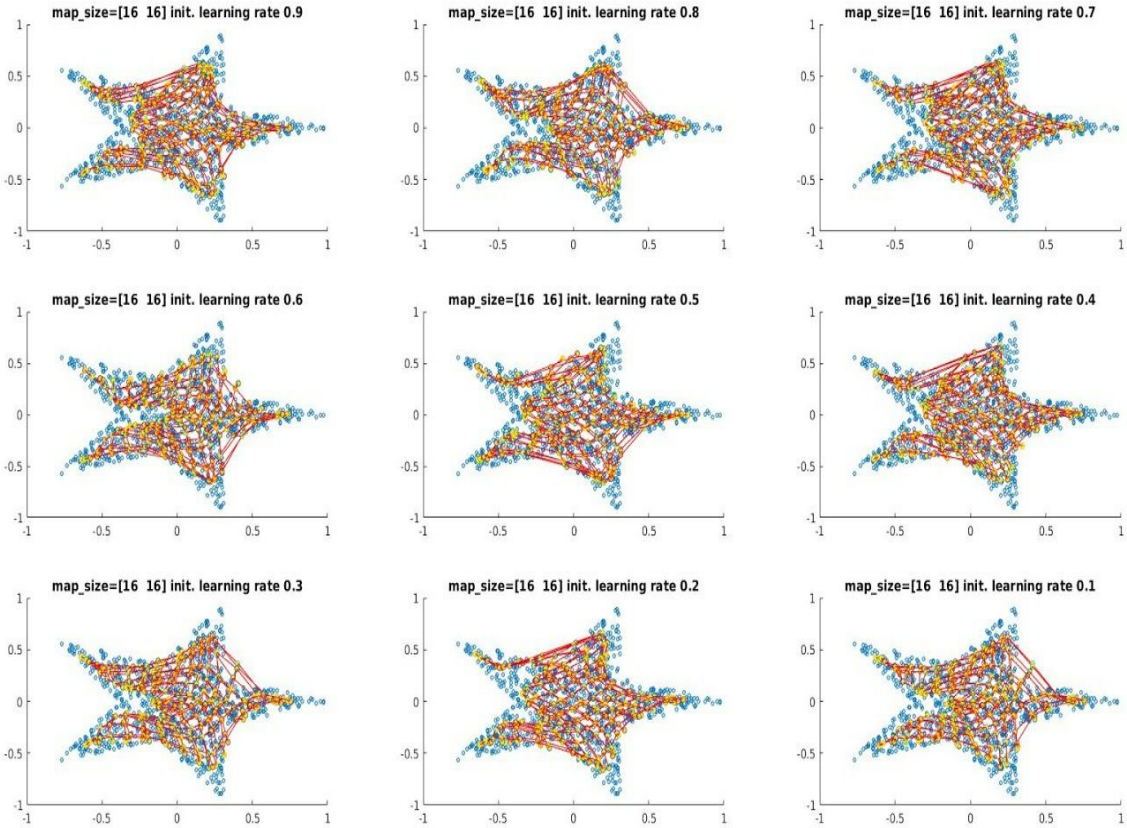

Figure 5: Varying the initial learning rate while keeping final learning rate constant

We then conducted the same experiment for the final learning rate, testing different values for it while keeping the initial learning rate constant (at 0.9). A map size of 16 x 16 was used. The results are visible in figure 6. As the plot shows for higher values of the final learning rate, the lattice does not spread itself well over the star (as expected due to the lack of a convergence phase).

The final experiment we made was testing different values for equal initial and final learning rates. The same map size of 16 x 16 was used. The results are visible in figure 7. The picture shows how for higher values of the learning rate we observe a behaviour similar to the one for high final learning rates: the lattice doesn't "unfold" in a well-distributed manner over the dataset.

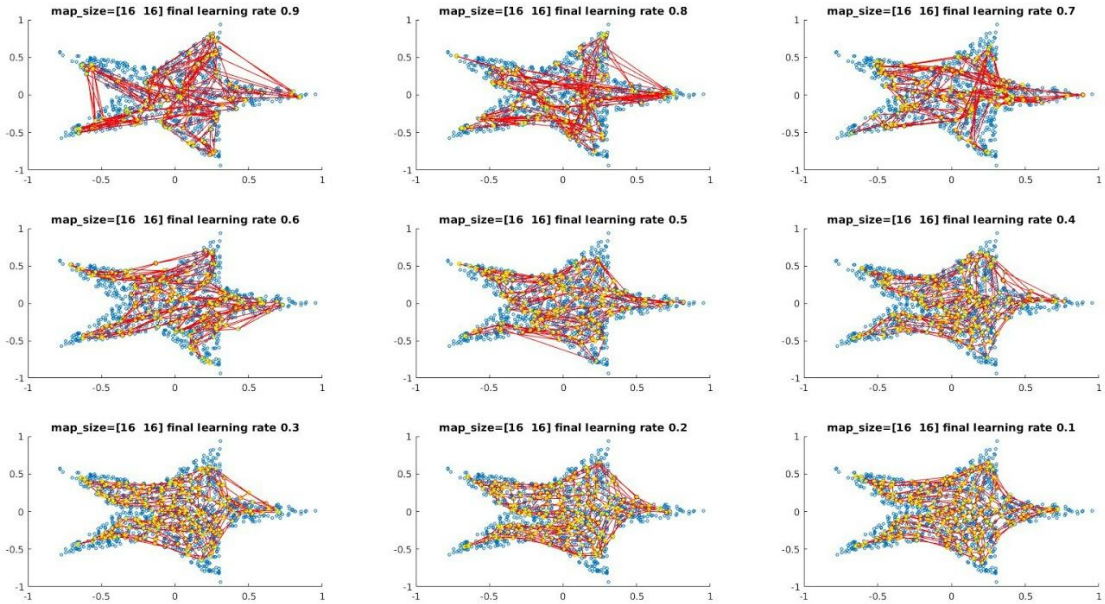

Figure 6: Varying the final learning rate while keeping initial learning rate constant

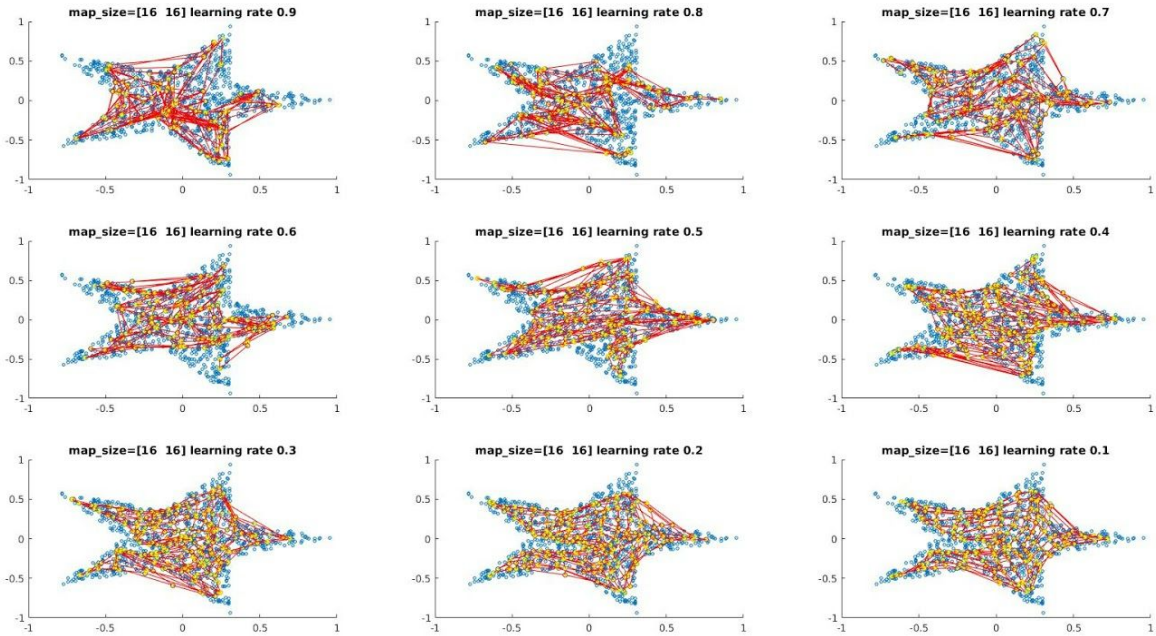

Figure 7: Varying the final learning rate and initial learning rate equally