Kohonen algorithm (Self-Organizing Map)

Neurocomputational cognitive modeling Course

Homework Number 2. Prof. Larry Manevitz; Fall, 2019

This is a simple implementation of SOM algorithm, network topography is a line.

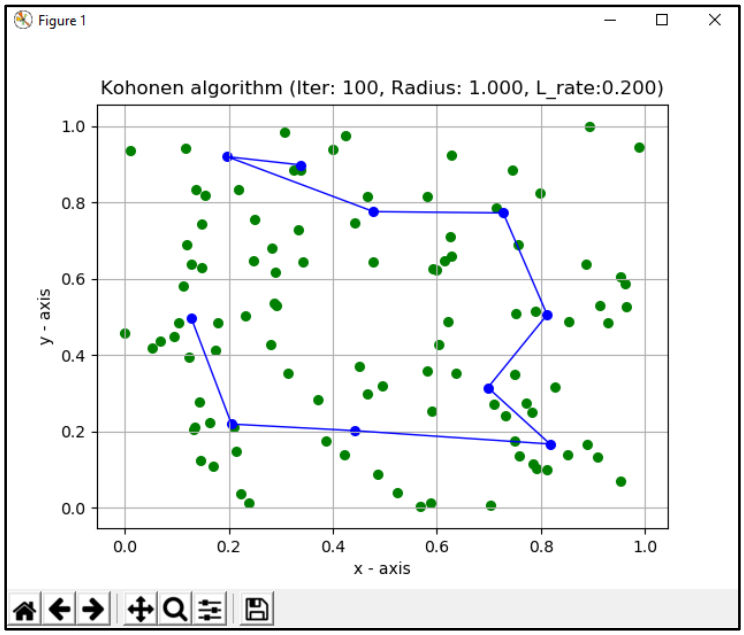
Case number 1:

Fit a line of neurons in a square. That is, the topology of the Kohonen level is a line. Assume the data is two dimensional (on a plane) where the points are given by (x, y) with 0<= x<=1, 0<=y<=1; and the data chosen randomly and uniformly.

When we selected more neurons (100), the line between the data points was more accurate, and the line "occupied" more space between the data points. That means there were more matches between data and the clusters.

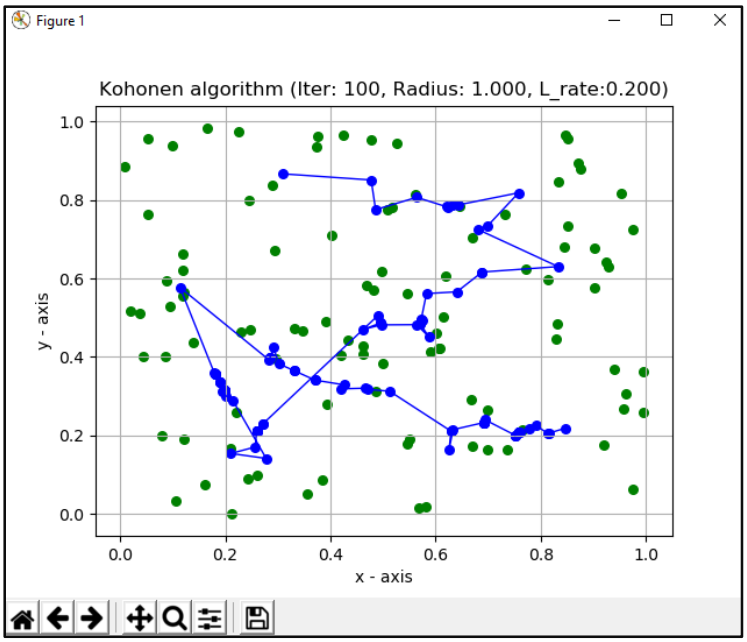
Running with 10 neurons and the next data give the next result:

- **Number of neurons**: 10
- **Iterations:** 100
- **Initial radius:** 5
- **Initial learning rate**: 1
- **Max time learning rate updates:** 50



Updating number of neurons to 100 increase the accuracy:

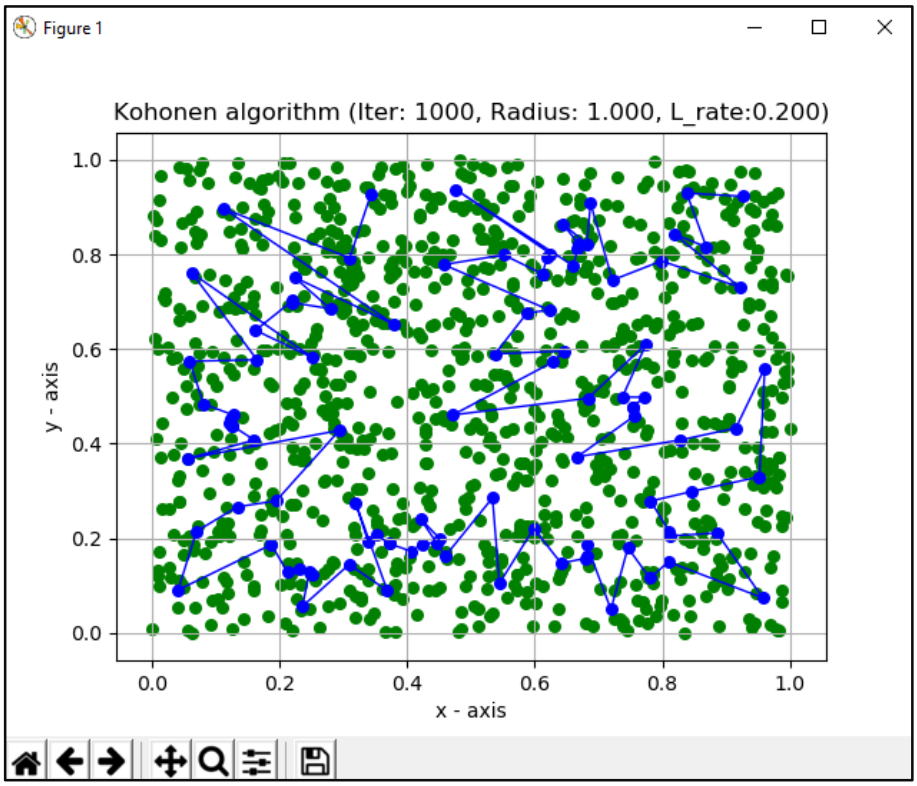
- **Number of neurons**: 100
- **Iterations:** 100
- **Initial radius:** 50
- **Initial learning rate**: 1
- **Max time learning rate updates:** 50



With 100 neurons

With a significant increase in the number of iterations:

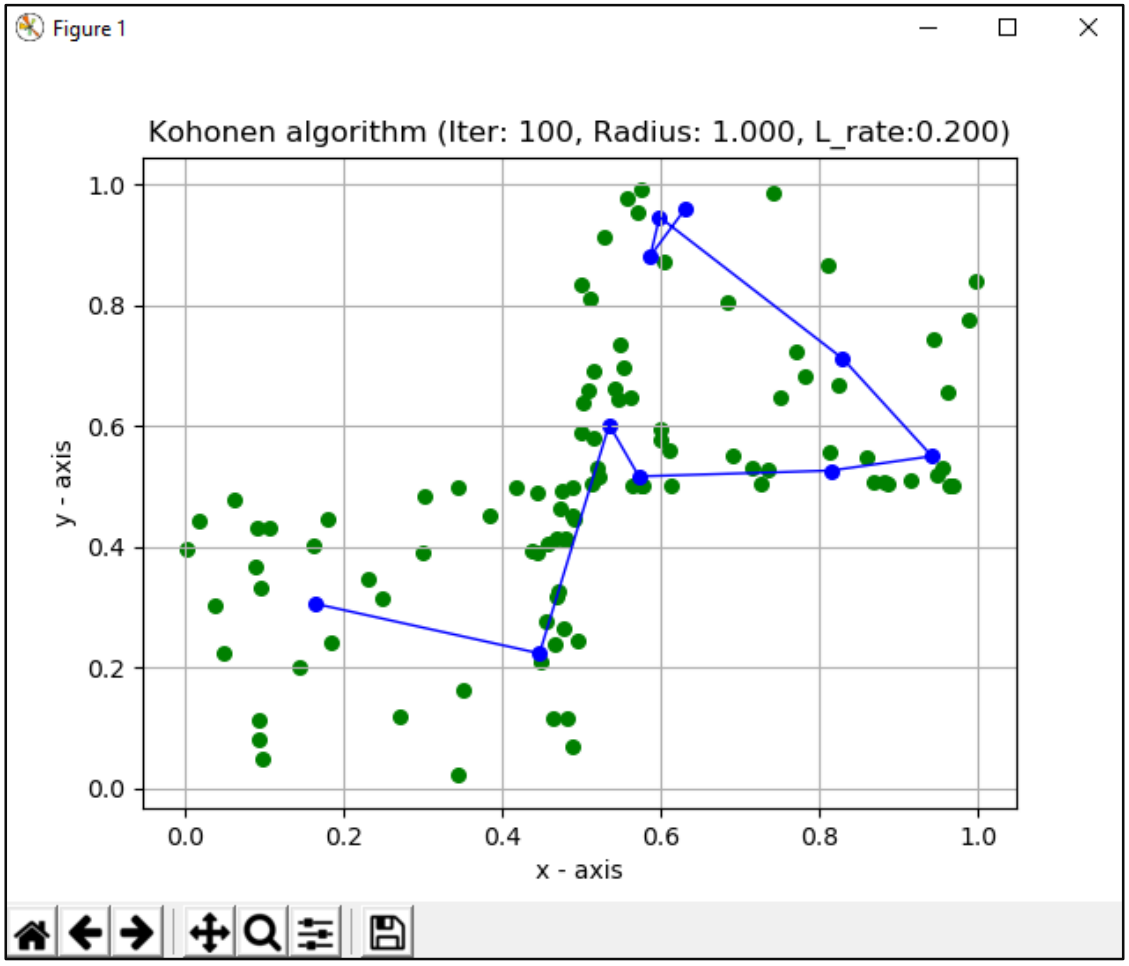
- **Number of neurons**: 100
- **Iterations:** 1000
- **Initial radius:** 50
- **Initial learning rate**: 1
- **Max time learning rate updates:** 200



Case number 2:

Same as above, with the **data chosen non-uniformly**, e.g. higher probability of being chosen the closer you are to (½, ½). (Assume (½, ½) is not part of the data set.

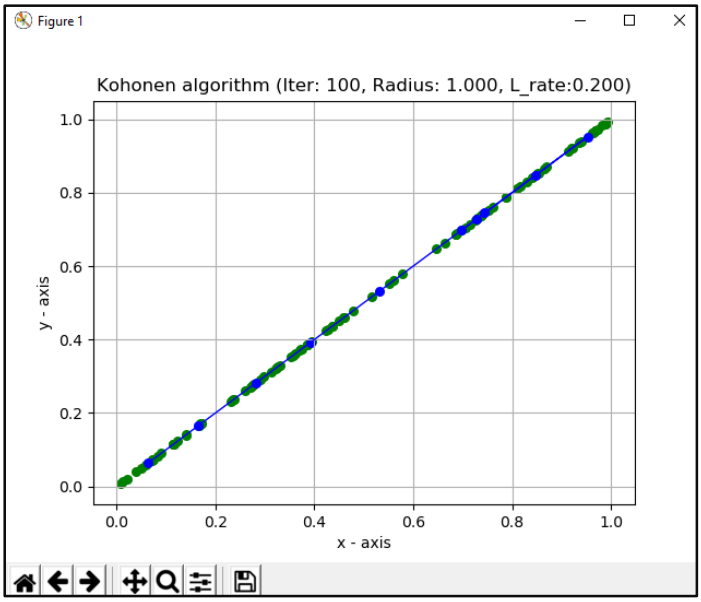
- **Number of neurons**: 10
- **Iterations:** 100
- **Initial radius:** 5
- **Initial learning rate**: 1
- **Max time learning rate updates:** 50



Case number 3:

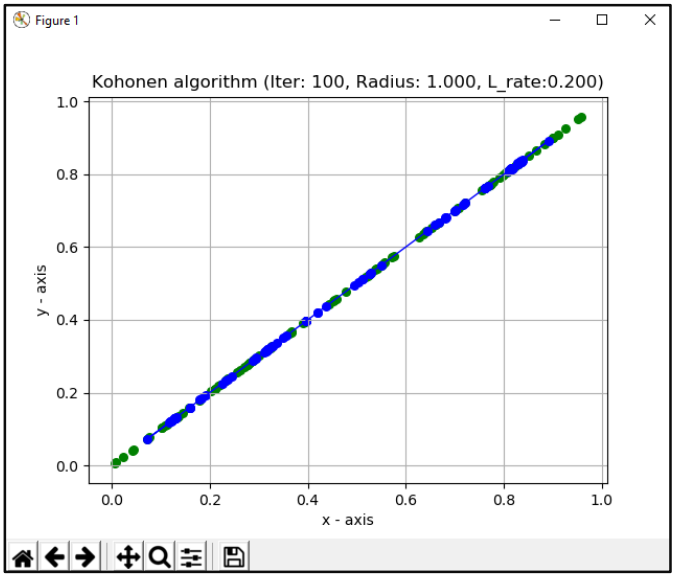
Data are all given uniformly but restricted to a diagonal of the square.

- **Number of neurons**: 10
- **Iterations:** 100
- **Initial radius:** 5
- **Initial learning rate**: 1
- **Max time learning rate updates:** 50



As we will see below - we have increased the number of neurons, but since the data set is distributed quite systematically, the network has reached the same result with only 10 neurons:

- **Number of neurons**: 100
- **Iterations:** 100
- **Initial radius:** 50
- **Initial learning rate**: 1
- **Max time learning rate updates:** 50



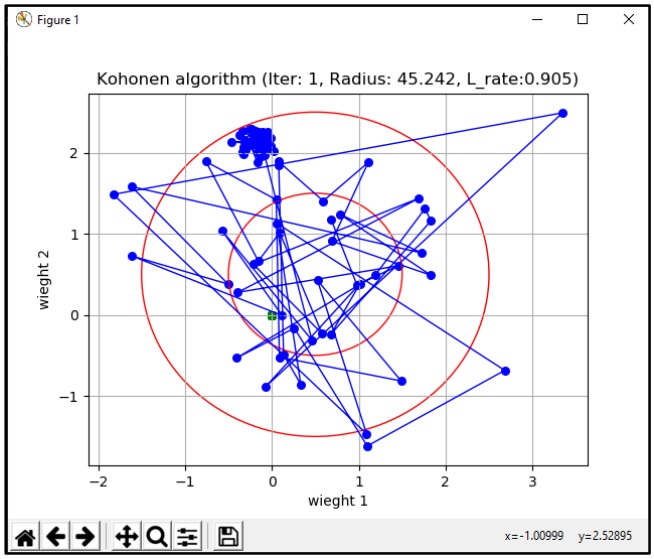
Case number 4:

Data set is chosen uniformly but in the band between two concentric circles (i.e. share the same center); one of radius 1 and one of radius 2.

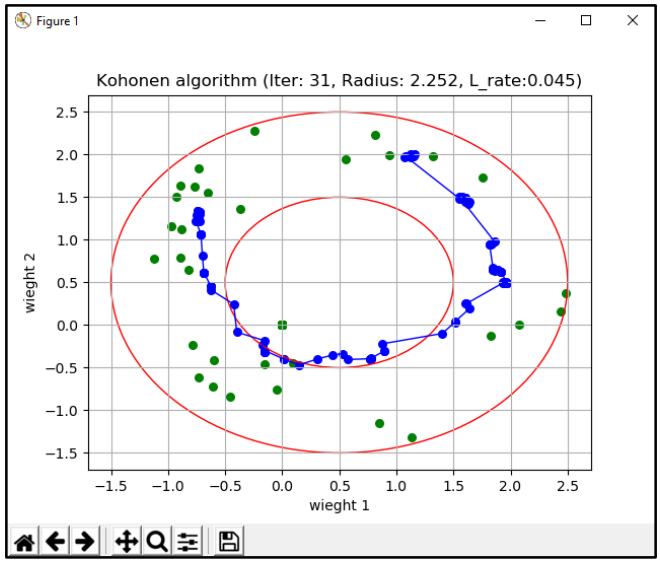
If we select:

- **Number of neurons**: 100
- **Iterations:** 100
- **Initial radius:** 50
- **Initial learning rate**: 1
- **Max time learning rate updates:** 50

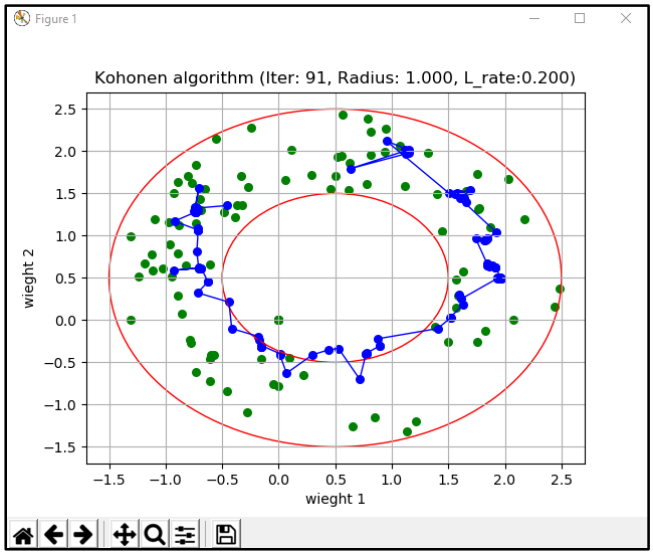
The following results can be seen:



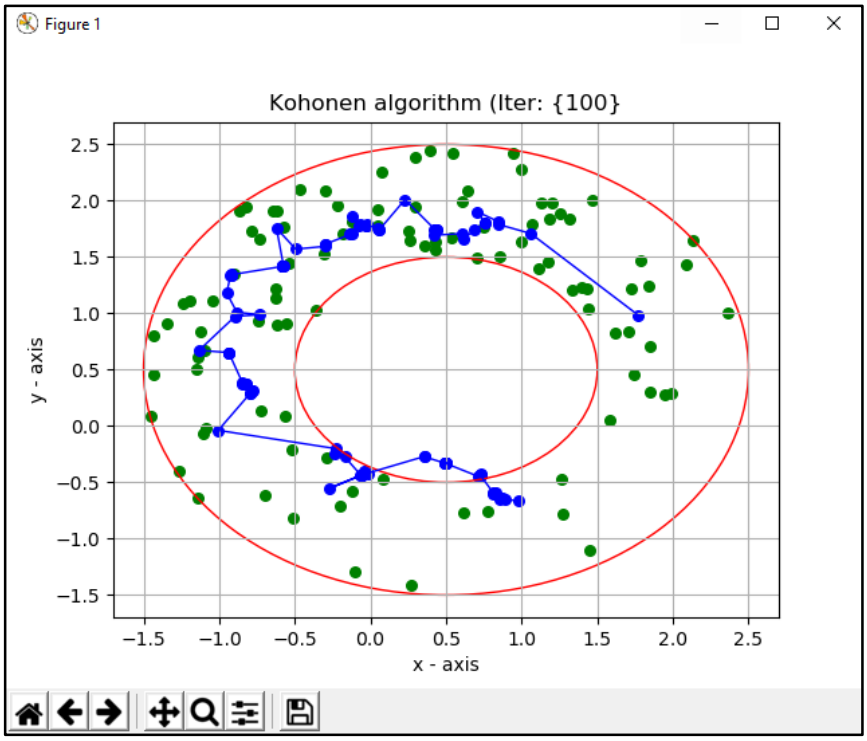
After the first iteration



After the 31's iteration



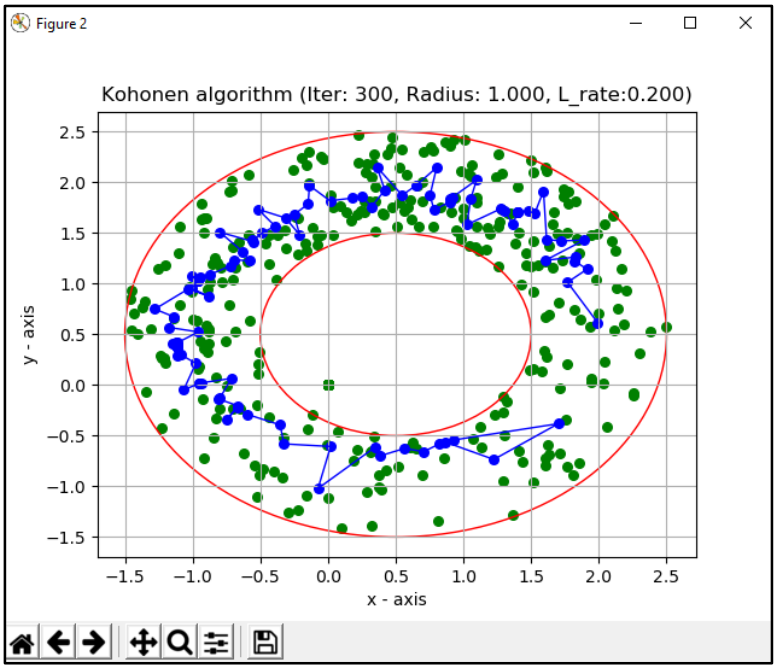
After the 91's iteration



Finally after 100 iterations you can see that the line drawn between the neurons is almost entirely between the two rings, and is adjusted according to the data points.

We've increased the number of iterations:

- **Number of neurons**: 100
- **Iterations:** 300
- **Initial radius:** 50
- **Initial learning rate**: 1
- **Max time learning rate updates:** 150



After 300 iteration. As you can see, the results are more accurate

Once again, we increased the **number of iterations to 1000** .This time we also increased the **initial radius**. You can see that a zigzag line is created that tries to "grab" as many data points as possible.

- **Number of neurons**: 100
- **Iterations:** 1000
- **Initial radius:** 60
- **Initial learning rate**: 1
- **Max time learning rate updates:** 200