a) Create a Hopfield network with 3 attractors T = [1 1; -1 -1; 1 -1] T and an arbitrary number of neurons. Start with various initial vectors and note down obtained attractors after a sufficient number of iterations.

- Is the number of real attractors bigger than the number of attractors used to create the network?

If a point converge to a point that is not an attractor we have a spurious state: an additional attractor is created from the network.

The more data we have, the higher the chance an extra attractor is created (fig1,2), unless initial vector value is taken close to the attractor state values. With same number of steps and data-points if we take initial vale away from ‘T’ e.g 0.5, -0.5, we again have the data points converging to an extra attractor that is not in ’T’.
Increasing the number of epochs (fig 3,4) also would reduce the chance for extra attractors to be created.

- How long does it typically take to reach the attractor?

To reach all three attractors T = [1 1; -1 -1; 1 -1] , it takes 18-25 steps or iterations.
If initial value set is closer to ‘T’ , it takes lesser number of iterations to reach the attractors even with more datapoints that have to reach the attractor state.

3 epochs, 50 datapoints —> lots of spurious states

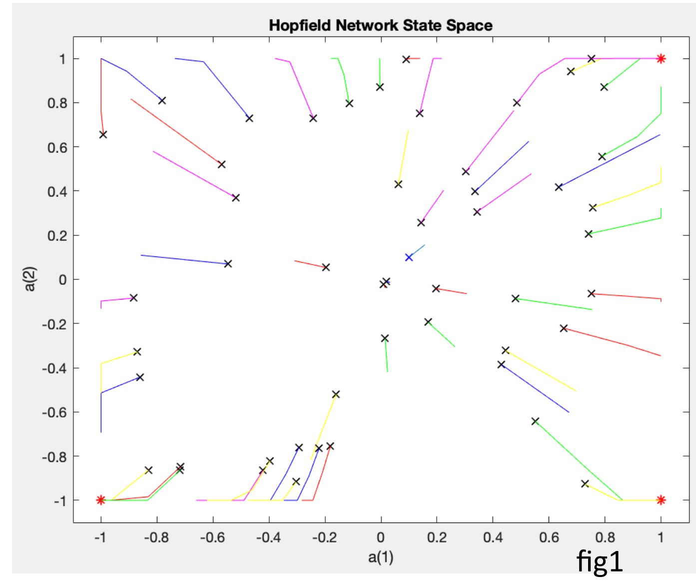

fig1

30 epochs, 50 datapoints —> just one spurious state

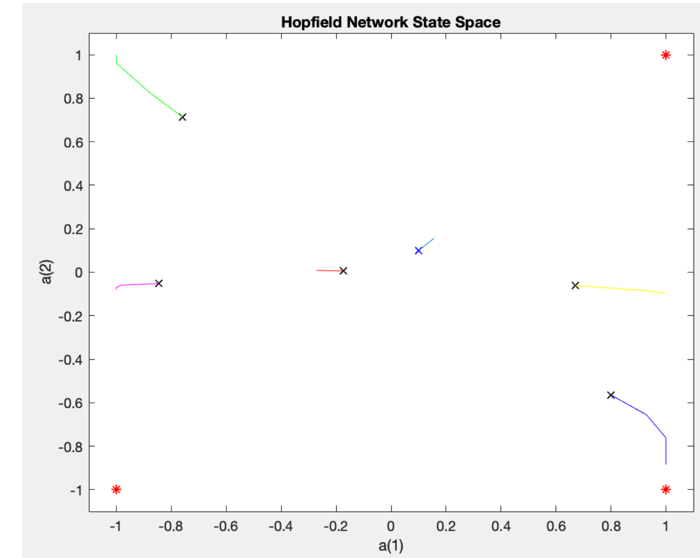

fig2

3 epochs, 50 datapoints —> lots of spurious states

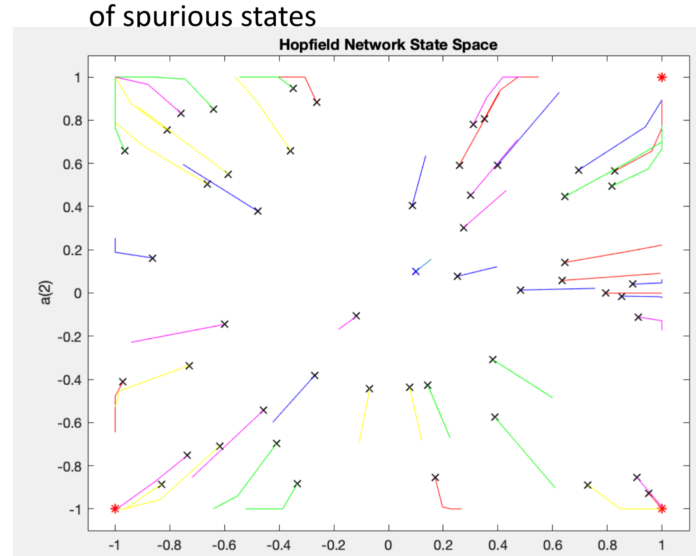

fig3

3 epochs, 5 datapoints —> just one spurious state

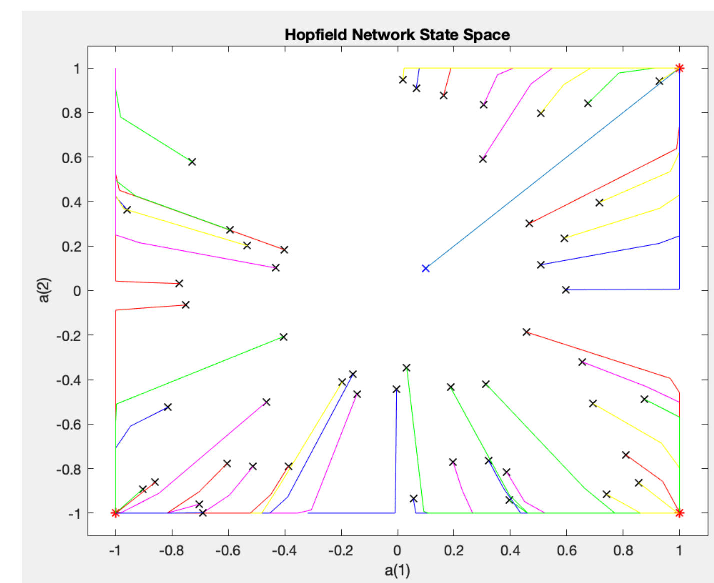

fig4

To reduce new created attractors, one can increase
the number of epochs (fig 3,4)
decrease number of datapoints (fig 1,2)
choose nearer to attractors datapoints

Execute script rep2. Modify this script to start from some particular points (e.g. of high symmetry) or to generate other numbers of points.

Are the attractors always those stored in the network at creation?

No, it depends on our data points in space. For instance, we generate number of points in a symmetrical order of e.g swapping x-y coordinates if the points. In this case if we take a point with x-y coordinates 0, an extra attractor is created in addition to the three stored in 'T' in our network.If the data points created are farther away from the attractors stored, then they might converge to another nearer attractor created in addition to 'T', thus these points become the spurious states.

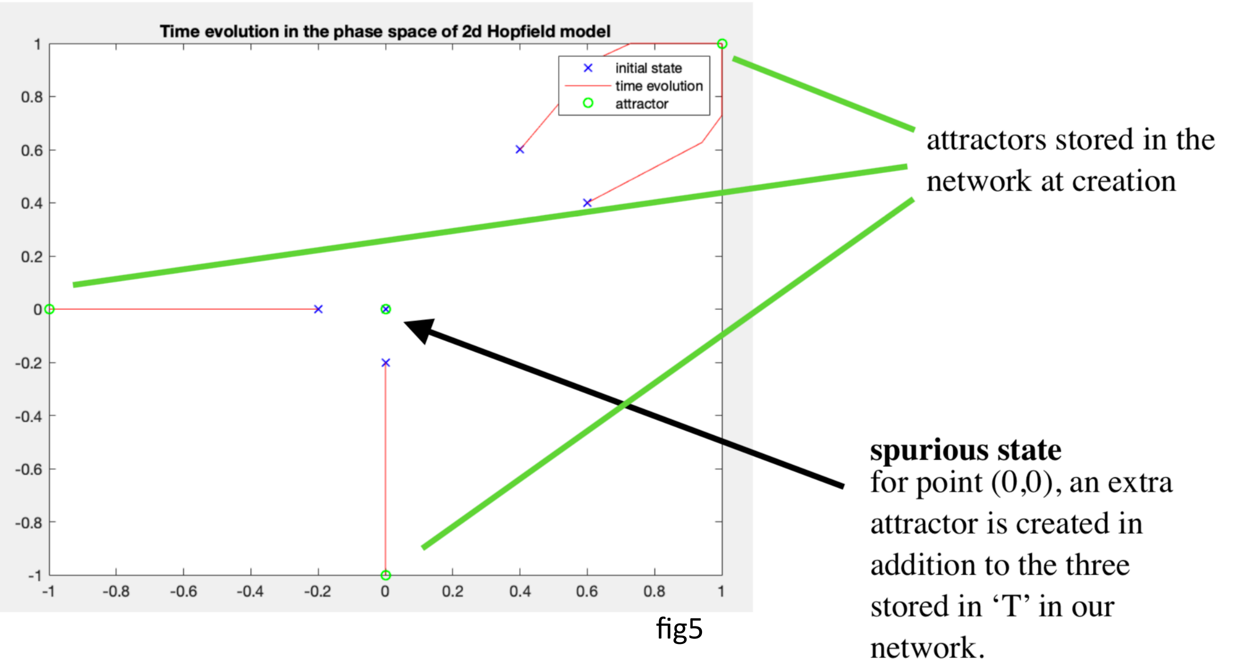

fig5

Do the same for a three neuron Hopfield network. This time use script rep3.

Fig6:

for 3 neuron Hopfield network,

5 initial states datapoints/initial states = [0 0 0 ;0 0 1; 1 1 0; 0.4 0.6 -1; -1 0.6 0.4],

target attractor vector T = [1 1 1; -1 -1 1;1 -1 -1]'—->

we get two additional attractors (black circle) at [X Y Z] = [-0.06 0.06 1; 0.36 -0.36 0.36] .

Therefore two out of the initial 5 datapoints are spurious datapoints (red circle).

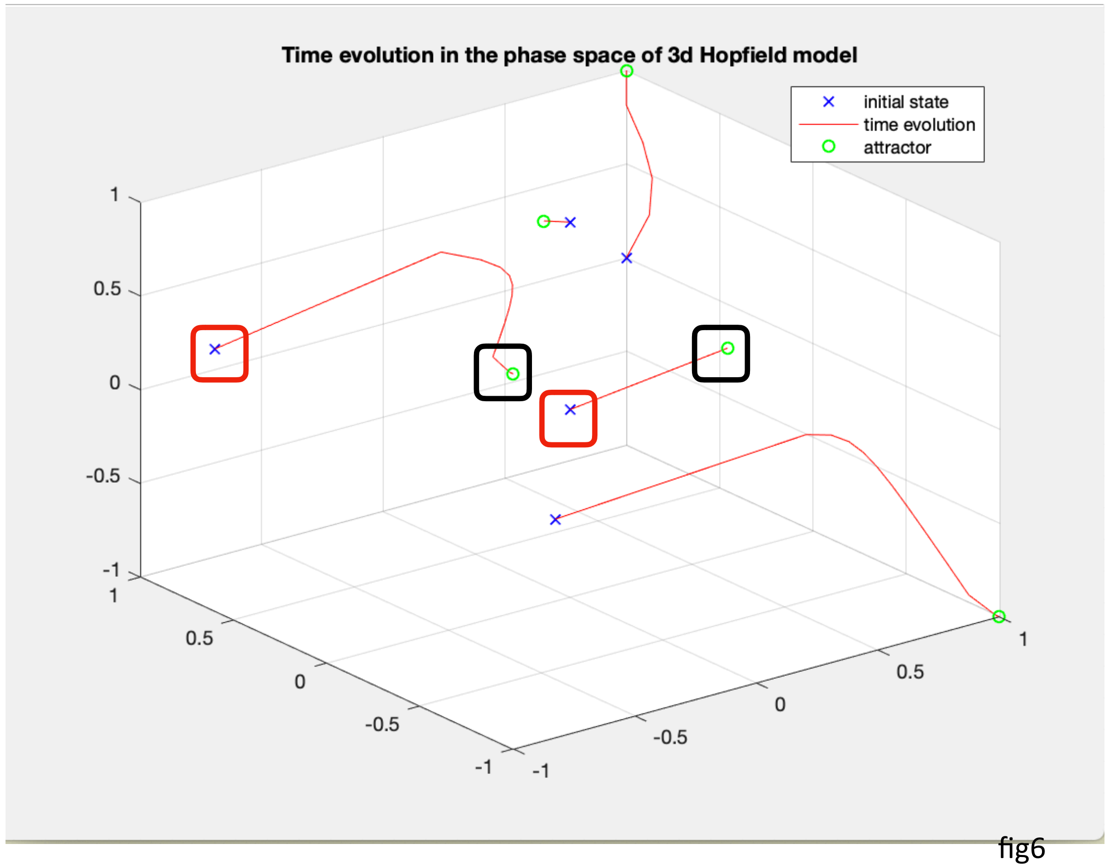

fig6

The function hopdigit creates a Hopfield network which has as attractors the handwritten digits 0; ……. ; 9. Then to test the ability of the network to correctly retrieve these patterns some noisy digits are given to the network.

- Is the Hopfield model always able to reconstruct the noisy digits? If not why?

HN is indeed always able to reconstruct noisy images (fig8) , yet the number of correct reconstructions depends on the amount of noise and the number of iterations. However, adding noisy data also makes the model more generalized.

Figure 7 shows a plot of multiple reconstructions performed with various combinations of noise and the number of iterations. We can see that changes in amount of noise affect the model performance more significantly than changes of number of iterations: the network makes more correct reconstructions at noise-level<15, and performs really worse adding a fewer noise; results for the same amount of noise are instead almost unchanged when choosing different values for the iteration parameter.

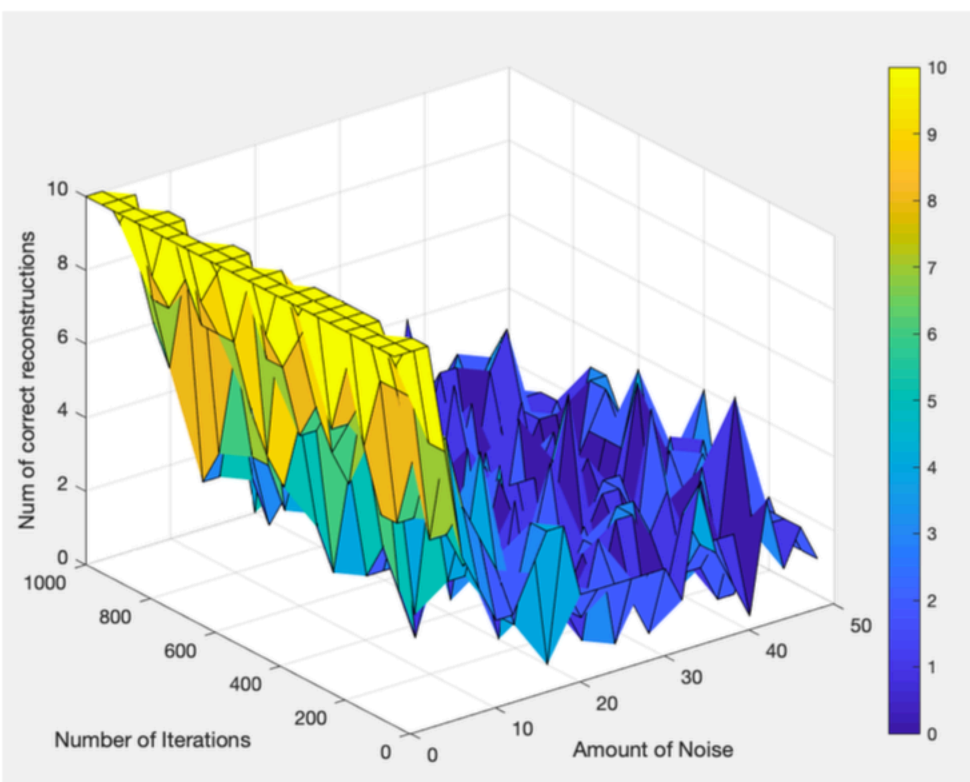

fig7

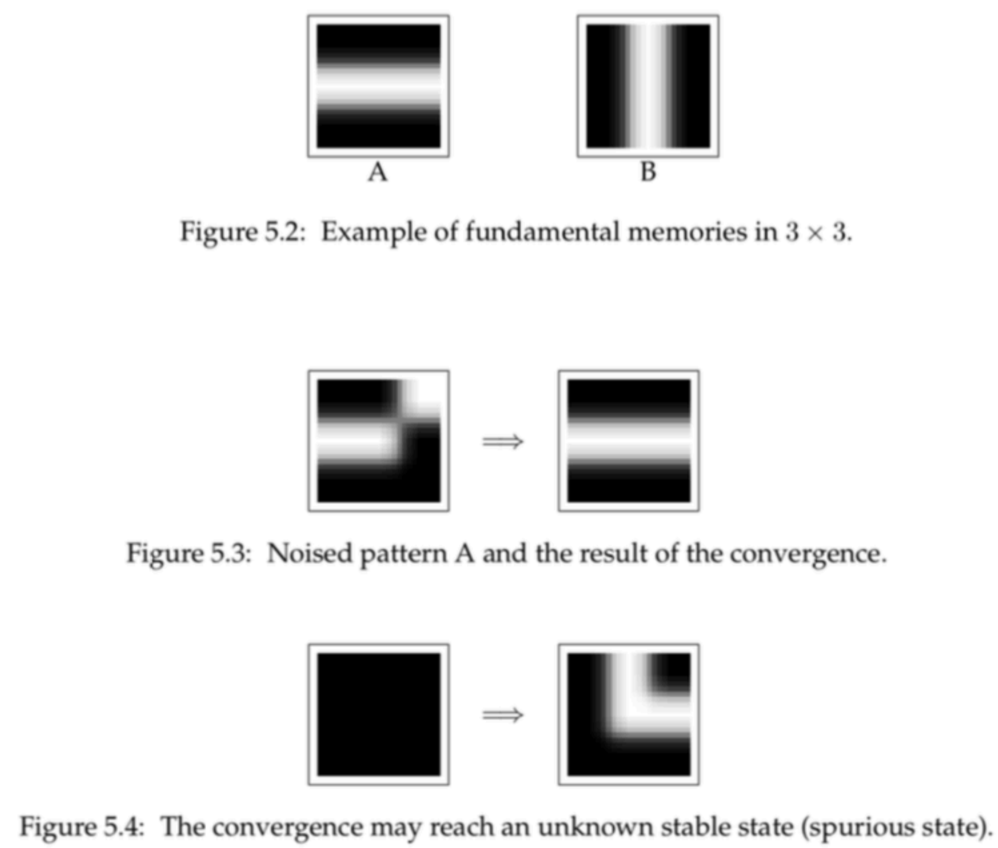

fig8