Younesse Kaddar

PROBLEM 1: Neuron with Autapse

Link of the iPython notebook for the code

1. One-dimensional deterministic dynamical system

Let us consider a neuron whose axons connects to its own dendrites (the resulting synapse is called an *autapse*): we will denote by

- *x* its firing rate
- w = 0.04 the strength of the synaptic connection
- I=-2 an external inhibitory input
- $f(s) \stackrel{\text{\tiny def}}{=} 50 \left(1 + \tanh(s)\right)$ the neuron's activation function, where s is the overall neuron input:



Plot of the neuron's activation function f

Figure 1.a. - Plot of the neuron's activation function f

(in arbitrary units) so that the firing rate is described by the following differential equation:

$$\dot{x}(t) = -x(t) + fig(wx(t)+Iig)$$

Plotting the derivative \dot{x} as a function of the neuron's firing rate x:

$$x\longmapsto -x+f(wx+I)$$
 .

results in the following 1-dimensional system dynamics:



Plot of the derivative \dot{x} as a function of the neuron's firing rate x (arbitrary units)

Figure 1.b. - Plot of the derivative as a function of the neuron's firing rate

The zero-crossings of \dot{x} as a function of x correspond to critical points, *i.e.* values of x at which the derivative $t \mapsto \dot{x}(t)$ equals zero, hence the function $t \mapsto x(t)$ being constant with respect to t. As a consequence, such values of x at which \dot{x} (as a function of x) vanishes are called **fixed point of the dynamics**, as x is left unchanged over time.

Dynamics fixed points are twofold:

- **stable fixed points** (attractors) are fixed points around which a *slight* change of x leads to x moving back to the stable fixed point value: these correspond to points x^* such that the derivative \dot{x} is:
 - positive at $x^* \varepsilon$ (*x* increases back to x^*)
 - negative at $x^* + \varepsilon$ (x decreases back to x^*)

for $\varepsilon > 0$ sufficiently small. In other words, the zero-crossing goes from positive values to negative ones.

On *Figure 1.b.*, the two stable fixed points at x = 2 and x = 98 are indicated by a filled up green point.

• **unstable fixed points** (repellers) are fixed points around which a *slight* change of x leads to x moving away from the unstable fixed point value: these correspond to points x^* such that the derivative \dot{x} is:

- negative at $x^* \varepsilon$ (x keeps decreasing away from x^*)
- positive at $x^* + \varepsilon$ (x keeps increasing away from x^*)

for $\varepsilon > 0$ sufficiently small. In other words, the zero-crossing goes from negative values to positive ones.

On *Figure 1.b.*, the only unstable fixed point at x = 50 is indicated by a hollow green circle.

Now, let us simulate the system for various initial conditions:



We set (arbitrary units):

- the time-increment to integrate the differential equation with the Euler method: $dt = \Delta t ext{ def} 0.1$
- The total duration $T=10=100\Delta t$

so that the Euler method yields:

$$egin{aligned} &rac{x(t+\Delta t)-x(t)}{\Delta t}=-x(t)+fig(wx(t)+Iig)\ &\iff x(t+\Delta t)-x(t)=-x(t)\Delta t+fig(wx(t)+Iig)\Delta t\ &\iff x(t+\Delta t)=(1-\Delta t)x(t)+fig(wx(t)+Iig)\Delta t \end{aligned}$$

Simulation of the system for different initial conditions (arbitrary units)



Figure 1.c. - Simulation of the system for different initial conditions

As expected, since x = 50 is an **unstable fixed point** of the dynamics: when

• x(0) = 50: x(t) remains constantly equal to 50 over time (as 50 is a **fixed point**)

- x(0) = 51 > 50: x(t) keeps increasing, converging to the (stable) fixed point 98, since the time derivative \dot{x} is **strictly positive** at 98 > x > 50 (cf. *Figure 1.b.*)
- x(0) = 49 < 50: x(t) keeps decreasing, converging to the (stable) fixed point 2, since the time derivative \dot{x} is **strictly negative** at 2 < x < 50 (cf. *Figure 1.b.*)

2. One-dimensional stochastic dynamical system

From now on, for the sake of realism, we add a white noise component to the system:

$$\dot{x}(t) = -x(t) + fig(wx(t)+Iig) + \sigma\eta(t)$$

where $\eta(t)$ is a Gaussian noise, σ is the noise magnitude.

The stochastic differential equation is solved numerically as follows:

$$egin{aligned} x(t+\Delta t)-x(t)&=-x(t)\Delta t+fig(wx(t)+Iig)\Delta t+\sigma\eta(t)\sqrt{\Delta t}\ \iff &x(t+\Delta t)=(1-\Delta t)x(t)+fig(wx(t)+Iig)\Delta t+\sigma\eta(t)\sqrt{\Delta t} \end{aligned}$$

which leads to (for $\sigma = 5$ and several random seeds):



Simulation of the stochastic system for different initial conditions, with σ = 5

Figure 2.d.1.1 - First seed: Simulation of the stochastic system for different initial conditions, for a noise magnitude $\sigma = 5$



Figure 2.d.1.2 - Second seed: Simulation of the stochastic system for different initial conditions, for a noise magnitude $\sigma = 5$

Simulation of the stochastic system for different initial conditions, with $\sigma = 5$



Figure 2.d.1.3 - Third seed: Simulation of the stochastic system for different initial conditions, for a noise magnitude $\sigma=5$

- the x(0) = 50 initial condition starts from an unstable fixed point of the dynamics (as seen before) when there is no noise: any slight variation of x relative to this unstable fixed point leads to x(t) moving away from this value. As it happens, here, this variation stems from the noise added to the time derivative x, which results in the unstable fixed point being slighly moved aside, leading to x(t) ending up above or below this unstable fixed point, and thus converging toward one the two stable fixed points (2 and 98), depending on whether the noise variation has drawn the unstable fixed point:
 - above *x* (leading *x* to decrease toward 2), as in figure 2.d.1.3
 - or below x (leading x to increase toward 98), as in figures 2.d.1.1 and 2.d.1.2

• the x(0) = 49 (resp. x(0) = 51) initial conditions causes x(t) to decrease (resp. increase) toward the **stable fixed point** 2 (resp. 98) when there is no noise (as seen before): any slight variation of x relative to these stable fixed points bring x(t) back to them. The only possibility for x not to converge toward 2 (resp. 98) instead would be that the noise added to derivative has such a magnitude that it makes it positive (resp. negative) at x, which doesn't happen in the three figures 2.d.1.1, 2.d.1.2, and 2.d.1.3*: the added noise magnitude is too small: x still converges toward the expected fixed point.

Now for various noise magnitudes:



Simulation of the system for different initial conditions and noise magnitudes (arbitrary units)

Figure 2.d.2.1 - First seed: Simulation of the stochastic system with $x(0) \in \{49, 50, 51\}$, for different noise magnitudes



Simulation of the system for different initial conditions and noise magnitudes (arbitrary units)

Figure 2.d.2.2 - Second seed: Simulation of the stochastic system with $x(0) \in \{49, 50, 51\}$, for different noise magnitudes



Figure 2.d.2.3 - Third seed: Simulation of the stochastic system with $x(0) \in \{49, 50, 51\}$, for different noise magnitudes

As the initial condition x(0) = 50 starts from an unstable fixed point of the deterministic dynamics, x may equiprobably converge to one the two stable fixed points when adding the noise, which makes this case of little interest (it clutters the graphs uselessly). We may as well only plot the remaining relevant cases ($x(0) \in \{49, 51\}$):



Simulation of the system for different initial conditions and noise magnitudes (arbitrary units)

Figure 2.d.3.1 - First seed: Simulation of the stochastic system with $x(0) \in \{49, 51\}$, for different noise magnitudes



Figure 2.d.3.2 - Second seed: Simulation of the stochastic system with $x(0) \in \{49, 51\}$, for different noise magnitudes



Simulation of the system for different initial conditions and noise magnitudes (arbitrary units)

Figure 2.d.3.3 - Third seed: Simulation of the stochastic system with $x(0) \in \{49, 51\}$, for different noise magnitudes

What appears in the above figures is that the bigger the noise magnitude σ is, the more x(t) is likely not to converge toward the *expected* stable fixed point (*i.e.* 98 for x > 50, 2 for x < 50).

This can be accounted for by the fact that the fixed points of dynamics change all the more that the noise magnitude is larger and larger: indeed, the larger the noise magnitude, the more the derivative \dot{x} is modified (as the noise is added to it) and the more it moves away from its original value: new zero-crossings (*i.e.* fixed points) appear, other disappear, etc...:



Figure 2.d.4 - Plot of the stochastic derivative as a function of the firing rate (and fixed points of the dynamics), for different noise magnitudes

To back this up: here is the evolution of the fixed points of the dynamics for different noise magnitudes (and different random seeds):



Evolution of fixed points when there is a noise component for different noise magnitudes σ (arbitrary units)

Figure 2.d.5.1 - First seed: Evolution of the stochastic dynamics fixed points for different noise magnitudes



Figure 2.d.5.2 - Second seed: Evolution of the stochastic dynamics fixed points for different noise magnitudes





Figure 2.d.5.3 - Third seed: Evolution of the stochastic dynamics fixed points for different noise magnitudes

So the larger the noise magnitude, the "more" the set of fixed points is modified, resulting in x(t) being less and less likely to converge toward the *expected* stable fixed point (98 for x > 50, 2 for x < 50).

PROBLEM 2: Circuit with mutual inhibition

Now, let us consider a two-neurons circuit in which the neurons mutually inhibit one another. We will denote by

- x_1 and x_2 the firing rates of the two neurons
- w=-0.1 the inhibitory synaptic weight
- I = 5 an excitatory external input
- $f(s) \stackrel{\text{\tiny def}}{=} 50 \left(1 + anh(s)
 ight)$ the neurons' activation function as before

1. Separate treatement of each neuron

We assume that we have the following differential equations:

$$\left\{ egin{array}{l} \dot{x}_1(t) = -x_1(t) + f(wx_2(t)+I) \ \dot{x}_2(t) = -x_2(t) + f(wx_1(t)+I) \end{array}
ight.$$

which results in the following two-dimensional system dynamics flow:



Figure 1.a.1 - 2D flow of the system dynamics

The nullclines of this system are the curves given by:



$$\left\{egin{array}{l} \dot{x}_1(t)=0\ \dot{x}_2(t)=0 \end{array}
ight.$$

Phase portait of the system dynamics, with nullclines (arbitrary units)



Figure 1.a.2 - Phase portrait of the system dynamics and nullclines

Their crossing points are the points at which both the x_1 derivative and the x_2 one vanish, *i.e.* the **fixed points of the 2D-system dynamics**. As before (cf. the previous problem), there are:

- stable fixed points: here, the points (0, 100) and (100, 0)
- **unstable fixed points**: here, the point (50, 50)

We can easily check these are indeed fixed points of the dynamics:

• For (0, 100):

$$-0+f(100w+I)=f(-5)=50(1+ anh(-5))\simeq 0 \ -100+f(0w+I)=f(5)-100=50(\underbrace{1+ anh(5)}_{\sim 2})-100\simeq 0$$

and it is symmetric for (100, 0)

• For (50, 50):

$$-50 + f(50w + I) = f(0) - 50 = 50(1 + \underbrace{ anh(0)}_{=0}) - 50 = 0$$

System simulation

We set (arbitrary units):

- the time-increment to integrate the differential equation with the Euler method: $dt = \Delta t \stackrel{ ext{def}}{=} 0.1$
- The total duration $T=10=100\Delta t$

so that the Euler method yields, for $i
eq j \in \{1,2\}$:

$$egin{aligned} &rac{x_i(t+\Delta t)-x_i(t)}{\Delta t}=-x_i(t)+fig(wx_j(t)+Iig)\ &\iff x_i(t+\Delta t)-x_i(t)=-x_i(t)\Delta t+fig(wx_j(t)+Iig)\Delta t\ &\iff x_i(t+\Delta t)=(1-\Delta t)x_i(t)+fig(wx_j(t)+Iig)\Delta t \end{aligned}$$

Simulation of the system with initial condition $(x_1(0), x_2(0)) = (1, 1)$ (arbitrary units)



Figure 1.b.1 - Simulation of the system with $(x_1(0),x_2(0)) riangleq (1,1)$

Simulation of the system with initial condition $(x_1(0), x_2(0)) = (1, 2)$ (arbitrary units)



Figure 1.b.2 - Simulation of the system with $(x_1(0),x_2(0)) riangleq (1,2)$



Figure 1.b.3 - Simulation of the system with $(x_1(0), x_2(0)) \stackrel{\text{\tiny def}}{=} (1, 0)$

It appears that:

- if $(x_1(0), x_2(0))$ is on the identity line *i.e.* if $x_1(0) = x_2(0)$) : $(x_1(t), x_2(t))$ converges toward the (50, 50) unstable fixed point, as exemplified by figure 1.b.1.
- if $(x_1(0), x_2(0))$ is strictly above the identity line *i.e.* if $x_1(0) < x_2(0)$) : $(x_1(t), x_2(t))$ converges toward the (0, 100) stable fixed point, as exemplified by figure 1.b.2.
- if $(x_1(0), x_2(0))$ is strictly below the identity line *i.e.* if $x_1(0) > x_2(0)$) : $(x_1(t), x_2(t))$ converges toward the (100, 0) stable fixed point, as exemplified by figure 1.b.3.

2. Vectorized system dynamics

We have hitherto treated each neuron separately, but there is a way to reduce the two differential equations to a single vectorized one:

$$\dot{\mathbf{x}}(t) = -\mathbf{x}(t) + f(W\mathbf{x}(t) + \mathbf{I})$$

by setting:

$$egin{aligned} \mathbf{x}(t) & \stackrel{ ext{def}}{=} egin{pmatrix} x_1(t) \ x_2(t) \end{pmatrix} \in \mathfrak{M}_{2,1}(\mathbb{R}) \ & W & \stackrel{ ext{def}}{=} egin{pmatrix} 0 & w \ w & 0 \end{pmatrix} \in \mathfrak{M}_2(\mathbb{R}) \ & I & \stackrel{ ext{def}}{=} egin{pmatrix} I \ I \end{pmatrix} \in \mathfrak{M}_{2,1}(\mathbb{R}) \end{aligned}$$

as a result of which the Euler method gives:

$$egin{aligned} &rac{1}{\Delta t} \Big(\mathbf{x}(t+\Delta t) - \mathbf{x}(t) \Big) = -\mathbf{x}(t) + f(W\mathbf{x}(t) + \mathbf{I}) \ & \iff & \mathbf{x}(t+\Delta t) = (1-\Delta t)\,\mathbf{x}(t) + \Delta t\,f(W\mathbf{x}(t) + \mathbf{I}) \end{aligned}$$

And we get the same simulations as before, for example with the initial condition $(x_1(0), x_2(0)) \stackrel{\text{\tiny def}}{=} (1, 2)$:

Matrix-based simulation of the system with initial condition $(x_1(0), x_2(0)) = (1, 2)$ (arbitrary units)



Figure 2.c. - Matrix-based simulation of the system with $(x_1(0), x_2(0)) \stackrel{\text{\tiny def}}{=} (1,2)$

PROBLEM 3: Hopfield Model

Link of the iPython notebook for the code

We will study the properties of a Hopfield network with N = 64 neurons, whose dynamics obeys the following differential equation:

 $\dot{\mathbf{x}} = -\mathbf{x} + f(W\mathbf{x}) + \sigma\eta(t)$

where

- **x** is the firing rates vector
- W is the N imes N weight matrix
- $\eta(t)$ is a Gaussian noise of magnitude $\sigma = 0.1$
- the activation function $f \stackrel{\scriptscriptstyle{ ext{def}}}{=} \operatorname{sign}$ for the sake of simplicity

1. Storing a pattern

A pattern $\mathbf{p} \in \mathfrak{M}_{N,1}(\{-1,1\})$ is stored in the Hopfield network by setting:

$$W \stackrel{\text{\tiny def}}{=} \frac{1}{N} \mathbf{p} \mathbf{p}^{\mathsf{T}}$$

Here are some examples of patterns, visualized as 8×8 images:

Available patterns



Figure 1.a. - Visualizations of 8×8 patterns

To begin with, let us focus on storing the battery pattern $\mathbf{p}_{\mathrm{bat}}$:







We set

$$W \stackrel{\text{\tiny def}}{=} rac{1}{N} \, \mathbf{p}_{ ext{bat}} \mathbf{p}_{ ext{bat}}^{\mathsf{T}}$$

Then, we simulate the network with resort to the Euler method (with time-increment $dt = \Delta t \stackrel{\text{\tiny def}}{=} 0.1$ (arbitrary units)):

$$\mathbf{x}(t + \Delta t) = (1 - \Delta t) \, \mathbf{x}(t) + \Delta t \operatorname{sign}(W \mathbf{x}(t)) + \sqrt{\Delta t} \, \sigma \eta(t)$$

which yields, for random starting patterns:

Network simulations

Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Step 7	Step 8	Step 9	Step 10	Step 11	Step 12	Step 13	Step 14	Step 15	Step 16
69.a	60 A	60 A	60 A 4	60 A 4	60 A.	60 A 4	60 A 4	C 24	C 14	C 14	C 18	C 14	C	C	C 14
1.1	100	100													2.2
Step 17	Step 18	Step 19	Step 20	Step 21	Step 22	Step 23	Step 24	Step 25	Step 26	Step 27	Step 28	Step 29	Step 30	Step 31	Step 32
171	177	177		177	177			17 B							
1 A A	1	1.0	1	1	1										
Step 33															

Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13	Step 15	Step 17	Step 19	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31
Step 33	Step 35	Step 37	Step 39	Step 41	Step 43	Step 45	Step 47	Step 49	Step 51	Step 53	Step 55	Step 57	Step 59	Step 61	Step 63
Step 65	Step 67	Step 69	Step 71	Step 73	Step 75	Step 77	Step 79	Step 81	Step 83	Step 85	Step 87	Step 89	Step 91	Step 93	Step 95
Step 97	Step 99	Step 101	Step 103	Step 105	Step 107	Step 109	Step 111	Step 113	Step 115	Step 117	Step 119	Step 121	Step 123	Step 125	Step 127
Step 129	Step 131	Step 133	Step 135	Step 137	Step 139	Step 141	Step 143	Step 145	Step 147	Step 149					

Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13	Step 15	Step 17	Step 19	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31
Step 33	Step 35	Step 37	Step 39	Step 41	Step 43	Step 45	Step 47	Step 49	Step 51	Step 53	Step 55	Step 57	Step 59	Step 61	Step 63
Step 65	Step 67	Step 69	Step 71	Step 73	Step 75	Step 77	Step 79	Step 81	Step 83	Step 85	Step 87	Step 89	Step 91	Step 93	Step 95
Step 97	Step 99	Step 101	Step 103	Step 105	Step 107	Step 109	Step 111	Step 113	Step 115	Step 117	Step 119	Step 121			

Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Step 7	Step 8	Step 9	Step 10	Step 11	Step 12	Step 13	Step 14	Step 15	Step 16
ž,	ě,	4	2	4	4	4	4	4	6	6	3	3	6	6	3
Step 17	Step 18	Step 19	Step 20	Step 21	Step 22	Step 23	Step 24	Step 25	Step 26	Step 27	Step 28	Step 29	Step 30	Step 31	Step 32
3	6	18													
Step 33	Step 34	Step 35	Step 36	Step 37	Step 38	Step 39	Step 40	Step 41	Step 42	Step 43	Step 44	Step 45	Step 46	Step 47	Step 48
													•		
Step 49	Step 50	Step 51	Step 52	Step 53	Step 54	Step 55	Step 56	Step 57	Step 58	Step 59	Step 60	Step 61	Step 62	Step 63	Step 64
			•			•	•	•	•	•	£.,		1		
Step 65	Step 66	Step 67	Step 68	Step 69	Step 70	Step 71	Step 72	Step 73	Step 74	Step 75	Step 76	Step 77	Step 78	Step 79	Step 80
Step 81	Step 82	Step 83	Step 84	Step 85	Step 86	Step 87	Step 88	Step 89	Step 90	Step 91	Step 92	Step 93	Step 94		
				•	•	0	•	•		•		•	•		
Step 1	Stan 3	Stop 5	Stop 7	Stop 9	Step 11	Step 13	Step 15	Step 17	Step 19	Stop 21	Stop 23	Stop 25	Stop 27	Stop 20	Stop 31
Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13	Step 15	Step 17	Step 19	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31
Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13	Step 15	Step 17	Step 19	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31
Step 1 Step 33	Step 3	Step 5	Step 7 Step 39	Step 9 Step 41	Step 11 Step 43	Step 13	Step 15 Step 47	Step 17 Step 49	Step 19 Step 51	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31
Step 1	Step 3 Step 35	Step 5 Step 37	Step 7 Step 39	Step 9 Step 41	Step 11 Step 43	Step 13 Step 45	Step 15 Step 47	Step 17 Step 49	Step 19 Step 51	Step 21 Step 53	Step 23 Step 55	Step 25 Step 57	Step 27	Step 29 Step 61	Step 31 Step 63
Step 1 Step 33 Step 33 Step 65	Step 3 Step 35 Step 35 Step 67	Step 5 Step 37 Step 37 Step 69	Step 7 Step 39 Step 71	Step 9 Step 41 Step 73	Step 11 Step 43 Step 75	Step 13 Step 13 Step 45 Step 77	Step 15 Step 17 Step 47 Step 79	Step 17 Step 19 Step 49 Step 81	Step 19 Step 51 Step 83	Step 21	Step 23 Step 55 Step 55 Step 87	Step 25 Step 57 Step 89	Step 27	Step 29 Step 61 Step 93	Step 31 Step 63 Step 95
Step 1 Step 33 Step 65	Step 3 Step 35 Step 67	Step 5 Step 37 Step 69	Step 7 Step 39 Step 71	Step 9 Step 41 Step 73	Step 11 Step 43 Step 75	Step 13 Step 45 Step 77	Step 15 Step 47 Step 79 Step 79	Step 17 Step 49 Step 81	Step 19 Step 51 Step 83 Step 83	Step 21 Step 53 Step 85 Step 85	Step 23 Step 55 Step 87 Step 87	Step 25 Step 57 Step 89	Step 27 Step 59 Step 91 Step 91	Step 29 Step 61 Step 93 Step 93	Step 31 Step 63 Step 95 Step 95
Step 1 Step 33 Step 65 Step 65 Step 97	Step 3 Step 35 Step 67 Step 67 Step 99	Step 5 Step 37 Step 69 Step 69 Step 101	Step 7 Step 39 Step 71 Step 71 Step 103	Step 9 Step 41 Step 73 Step 73 Step 105	Step 11 Step 43 Step 75 Step 75 Step 107	Step 13 Step 45 Step 77 Step 77 Step 109	Step 15 Step 47 Step 79 Step 79 Step 111	Step 17 Step 49 Step 81 Step 81 Step 113	Step 19 Step 51 Step 83 Step 83 Step 115	Step 21 Step 53 Step 85 Step 85 Step 117	Step 23 Step 55 Step 87 Step 87 Step 119	Step 25 Step 57 Step 89 Step 89 Step 121	Step 27 Step 59 Step 91 Step 123	Step 29 Step 61 Step 93 Step 125	Step 31 Step 63 Step 95 Step 95 Step 127
Step 1 Step 33 Step 33 Step 65 Step 97	Step 3 Step 35 Step 67 Step 99	Step 5 Step 37 Step 69 Step 101	Step 7 Step 39 Step 71 Step 71 Step 103	Step 9 Step 41 Step 73 Step 105 Step 105	Step 11 Step 43 Step 75 Step 75 Step 107	Step 13 Step 13 Step 45 Step 77 Step 109	Step 15 Step 47 Step 79 Step 111	Step 17 Step 49 Step 81 Step 113 Step 113	Step 19 Step 51 Step 83 Step 115 Step 115	Step 21 Step 53 Step 53 Step 85 Step 117	Step 23 Step 55 Step 87 Step 87 Step 119	Step 25 Step 57 Step 89 Step 121	Step 27 Step 59 Step 91 Step 123 Step 123	Step 29 Step 61 Step 93 Step 125	Step 31 Step 63 Step 95 Step 95 Step 127
Step 1 Step 33 Step 33 Step 65 Step 97 Step 97 Step 129	Step 3 Step 35 Step 67 Step 99 Step 99 Step 131	Step 5 Step 37 Step 69 Step 101 Step 101 Step 133	Step 7 Step 39 Step 71 Step 71 Step 103 Step 103 Step 135	Step 9 Step 41 Step 73 Step 105 Step 105 Step 137	Step 11 Step 43 Step 43 Step 75 Step 107 Step 107 Step 139	Step 13 Step 13 Step 45 Step 77 Step 77 Step 109 Step 109 Step 141	Step 15 Step 47 Step 79 Step 111 Step 111 Step 143	Step 17 Step 49 Step 81 Step 113 Step 113	Step 19 Step 51 Step 83 Step 115 Step 115	Step 21 Step 53 Step 53 Step 85 Step 117	Step 23 Step 55 Step 87 Step 87 Step 119	Step 25 Step 57 Step 89 Step 121	Step 27 Step 59 Step 91 Step 123 Step 123	Step 29 Step 61 Step 93 Step 125	Step 31 Step 63 Step 95 Step 127
Step 1 Step 33 Step 33 Step 65 Step 97 Step 129	Step 3 Step 35 Step 67 Step 99 Step 99 Step 131	Step 5 Step 37 Step 69 Step 101 Step 101 Step 133	Step 7 Step 39 Step 71 Step 71 Step 103 Step 103 Step 135	Step 9 Step 41 Step 73 Step 105 Step 105 Step 137	Step 11 Step 43 Step 75 Step 107 Step 107 Step 139	Step 13 Step 13 Step 45 Step 77 Step 109 Step 109 Step 141	Step 15 Step 47 Step 79 Step 111 Step 111 Step 143	Step 17 Step 49 Step 81 Step 113 Step 113	Step 19 Step 51 Step 83 Step 115 Step 115	Step 21 Step 53 Step 85 Step 85 Step 117	Step 23 Step 55 Step 87 Step 119	Step 25 Step 57 Step 89 Step 121	Step 27	Step 29 Step 61 Step 93 Step 125	Step 31 Step 63 Step 95 Step 127

Figure 1.b.2 - **One Pattern (battery) stored**: Network simulation from different starting patterns

It can be noticed that the network relaxes to two fixed points, in our examples:

- the battery pattern: $p_{\rm bat}$
- and its opposite pattern: $-\mathbf{p}_{\mathrm{bat}}$

From a theoretical standpoint, this can be explained by the fact that both \mathbf{p}_{bat} and $-\mathbf{p}_{bat}$ are fixed points of the function

$$\mathbf{x} \mapsto \operatorname{sign}(W\mathbf{x})$$

Indeed:

$$\begin{split} \operatorname{sign}\left(W\mathbf{p}_{\mathrm{bat}}\right) &= \operatorname{sign}\left(\frac{1}{N} \left(\mathbf{p}_{\mathrm{bat}} \mathbf{p}_{\mathrm{bat}}^{\mathsf{T}}\right) \mathbf{p}_{\mathrm{bat}}\right) \\ &= \operatorname{sign}\left(\frac{1}{N} \mathbf{p}_{\mathrm{bat}} \left(\mathbf{p}_{\mathrm{bat}}^{\mathsf{T}} \mathbf{p}_{\mathrm{bat}}\right)\right) \\ &= \operatorname{sign}\left(\underbrace{\frac{1}{N} \mathbf{p}_{\mathrm{bat}}}_{i=1} \left(\underbrace{\sum_{i=1}^{N} \underbrace{\left[\mathbf{p}_{\mathrm{bat}}\right]_{i}^{2}}_{i=1}\right)}_{i=1}\right) \\ &= \operatorname{sign}\left(\mathbf{p}_{\mathrm{bat}}\right) \\ &= \mathbf{p}_{\mathrm{bat}} \end{split}$$

(because the matrix product is associative)

$$egin{aligned} & ext{(because } \mathbf{p}_{ ext{bat}} \in \mathfrak{M}_{N,1}(\{-1,1\}) \ & ext{ so sign}\left([\mathbf{p}_{ ext{bat}}]_i
ight) = \left[\mathbf{p}_{ ext{bat}}
ight]_i \quad orall 1 \leq i \leq N \) \end{aligned}$$

And:

$$egin{aligned} \operatorname{sign}\left(W(-\mathbf{p}_{ ext{bat}})
ight) &= \operatorname{sign}\left(-W\mathbf{p}_{ ext{bat}}
ight) \ &= -\operatorname{sign}\left(W\mathbf{p}_{ ext{bat}}
ight) \ &= -\mathbf{p}_{ ext{bat}} \end{aligned}$$

(because sign is an odd function)

Therefore, leaving the noise aside, by setting:

 $arphi \stackrel{ ext{def}}{=} \mathbf{x} \longmapsto -\mathbf{x} + ext{sign}(W\mathbf{x})$

so that the corresponding dynamics without noise is given by:

$$\dot{\mathbf{x}} = \varphi(\mathbf{x})$$

it follows that:

$$arphi(\mathbf{p}_{ ext{bat}}) = arphi(-\mathbf{p}_{ ext{bat}}) = \mathbf{0}$$

As a result: \mathbf{p}_{bat} and $-\mathbf{p}_{bat}$ are fixed points of the corresponding deterministic (i.e. without noise) dynamics, as a result of which the stochastic dynamics relaxes to them.

2. Storing two patterns

Now, let us add a new pattern $\mathbf{p}_{\rm com}$:

Storing 2 patterns:



Figure 2.c.1 - **Two Patterns stored**: Battery & Computer

so that

$$W = rac{1}{N} \Big(\mathbf{p}_{ ext{bat}} \mathbf{p}_{ ext{bat}}^\intercal + \mathbf{p}_{ ext{com}} \mathbf{p}_{ ext{com}}^\intercal \Big) \; ,$$

And we simulate the network on random starting patterns:

Network simulations

Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Step 7	Step 8	Step 9	Step 10	Step 11	Step 12	Step 13	Step 14	Step 15	Step 16
3F 1	35	350			3 B B	3 B.	1								
Step 17	Step 18	Step 19	Step 20	Step 21	Step 22	Step 23	Step 24	Step 25	Step 26	Step 27	Step 28	Step 29	Step 30	Step 31	Step 32
						•	•								
Step 33	Step 34	Step 35	Step 36	Step 37	Step 38	Step 39	Step 40	Step 41	Step 42	Step 43	Step 44	Step 45	Step 46	Step 47	Step 48
Step 49	Step 50	Step 51	Step 52	Step 53	Step 54	Step 55	Step 56	Step 57	Step 58	Step 59	Step 60	Step 61	Step 62	Step 63	Step 64
Step 65	Step 66	Step 67	Step 68	Step 69	Step 70	Step 71	Step 72	Step 73	Step 74	Step 75	Step 76	Step 77	Step 78	Step 79	Step 80
Step 81	Step 82	Step 83	Step 84	Step 85	Step 86	Step 87									



Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13	Step 15	Step 17	Step 19	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31
Set E	60 E	- CT 1	100	0.00	000	1000	1000	1000	1000	1000	1000	1000			
÷С.,	ен на 1860 и на 1860	- C.	- C.	1000	100	1000									
Step 33	Step 35	Step 37	Step 39	Step 41	Step 43	Step 45	Step 47	Step 49	Step 51	Step 53	Step 55	Step 57	Step 59	Step 61	Step 63
100			100	100		100		100		100	100	100			
Step 65	Step 67	Step 69	Step 71	Step 73	Step 75	Step 77	Step 79	Step 81	Step 83	Step 85	Step 87	Step 89	Step 91	Step 93	Step 95
						100	1000	1000	100						
Step 97	Step 99	Step 101	Step 103												

Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13	Step 15	Step 17	Step 19	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31
x_{ℓ}	\mathbf{a}_{i}	\mathbf{a}_{i}	11 .1	11.1											
Step 33	Step 35	Step 37	Step 39	Step 41	Step 43	Step 45	Step 47	Step 49	Step 51	Step 53	Step 55	Step 57	Step 59	Step 61	Step 63
			1.1	100	1.25										100
Step 65	Step 67	Step 69	Step 71	Step 73	Step 75	Step 77	Step 79	Step 81	Step 83	Step 85	Step 87	Step 89	Step 91	Step 93	Step 95
Step 97	Step 99	Step 101	Step 103	Step 105	Step 107	Step 109	Step 111	Step 113	Step 115						
					1.0										



Figure 2.c.2 - **Two Patterns (battery & computer) stored**: Network simulation from different starting patterns

Again, we can show that:

- the battery pattern \mathbf{p}_{bat} and its opposite pattern: $-\mathbf{p}_{bat}$
- the computer pattern \mathbf{p}_{com} and its opposite pattern: $-\mathbf{p}_{com}$

are stored, owing to the fact that they are fixed points of the corresponding deterministic dynamics.

Indeed (for the sake of convenience, we set $\mathbf{p} \stackrel{\text{\tiny def}}{=} \mathbf{p}_{bat}$ and $\mathbf{q} \stackrel{\text{\tiny def}}{=} \mathbf{p}_{com}$):

$$\begin{split} \operatorname{sign}\left(W\mathbf{p}\right) &= \operatorname{sign}\left(\frac{1}{N} \left(\mathbf{p}\mathbf{p}^{\mathsf{T}} + \mathbf{q}\mathbf{q}^{\mathsf{T}}\right) \mathbf{p}\right) \\ &= \operatorname{sign}\left(\underbrace{\frac{1}{N} \left(\mathbf{p}\mathbf{p}^{\mathsf{T}}\right) \mathbf{p}}_{= \mathbf{p} \; (\operatorname{cf. previously})} + \frac{1}{N} \left(\mathbf{q}\mathbf{q}^{\mathsf{T}}\right) \mathbf{p}\right) \\ &= \operatorname{sign}\left(\mathbf{p} + \frac{1}{N} \mathbf{q} \underbrace{\left(\mathbf{q}^{\mathsf{T}}\mathbf{p}\right)}_{= \langle \mathbf{q}, \mathbf{p} \rangle}\right) \end{split}$$

But:

$$|\langle \mathbf{q},\,\mathbf{p}
angle|\,=\, ig|\sum_{i=1}^N \underbrace{[\mathbf{q}]_i[\mathbf{p}]_i}_{\in\{-1,1\}}ig|\,\leq\,N$$

And for $arepsilon\in\{-1,1\}$:

$$egin{aligned} &\langle \mathbf{q}, \mathbf{p}
angle = arepsilon N \ &\Leftrightarrow & \sum_{i=1}^{N} [\mathbf{q}]_{i} [\mathbf{p}]_{i} = arepsilon N \ &\Leftrightarrow & \sum_{i=1}^{N} \underbrace{([\mathbf{q}]_{i} [\mathbf{p}]_{i} - arepsilon)}_{\leq 0 ext{ if } arepsilon = 0} = 0 \ &\Leftrightarrow & orall 1 \leq i \leq N, \quad [\mathbf{q}]_{i} [\mathbf{p}]_{i} - arepsilon = 0 \ &\Leftrightarrow & orall 1 \leq i \leq N, \quad [\mathbf{q}]_{i} [\mathbf{p}]_{i} = arepsilon \\ &\Leftrightarrow & orall 1 \leq i \leq N, \quad [\mathbf{q}]_{i} [\mathbf{p}]_{i} = arepsilon \\ &\Leftrightarrow & orall 1 \leq i \leq N, \quad [\mathbf{q}]_{i} [\mathbf{p}]_{i} = arepsilon \\ &\Leftrightarrow & orall 1 \leq i \leq N, \quad [\mathbf{q}]_{i} = arepsilon/[\mathbf{p}]_{i} = arepsilon \end{bmatrix}_{i} \quad (ext{because } [\mathbf{p}]_{i} \in \{-1, 1\,) \} \end{aligned}$$

NB: alternatively, we could have used the Cauchy-Schwarz inequality:

$$|\langle \mathbf{q},\,\mathbf{p}
angle|\leq \underbrace{\|\mathbf{q}\|_2\,\|\mathbf{p}\|_2}_{=\sqrt{N}\sqrt{N}=N} \quad ext{ and } \quad \Big(|\langle \mathbf{q},\,\mathbf{p}
angle|=N \Longleftrightarrow \mathbf{q}=\pm \mathbf{p}\Big)$$

Here, as:

$$\mathbf{q} \neq \mathbf{p} \quad ext{and} \quad \mathbf{q} \neq -\mathbf{p}$$

it comes that:

 $|\langle {f q},\,{f p}
angle| < N$

Hence:

$$egin{aligned} \operatorname{sign}\left(W\mathbf{p}
ight) &= \operatorname{sign}\left(\mathbf{p} + \overbrace{\underbrace{\langle \mathbf{q}, \mathbf{p}
ight\rangle}{N}}^{\in [-(N-1)\,,\,N-1]} \mathbf{q}
ight) \ &\stackrel{\circledast}{=} \operatorname{sign}\left(\mathbf{p}
ight) & \left(\operatorname{because each} rac{\langle \mathbf{q}, \mathbf{p}
angle}{N} \left[\mathbf{q}
ight]_i \in \left[-1 + rac{1}{N}, \, 1 - rac{1}{N}
ight] \ &: \operatorname{it can't change the sign of} \left[\mathbf{p}
ight]_i \in \{-1,1\}\) \ &= \mathbf{p} \end{aligned}$$

And likewise, by symmetry:

 $\operatorname{sign}\left(W\mathbf{q}\right) = \mathbf{q}$

Once we know that **p** and **q** are fixed points, the same holds true for their opposites, as sign is an odd function (the proof is the same as previously).

So $\pm \mathbf{p}$ and $\pm \mathbf{q}$ are all fixed points of the corresponding deterministic dynamics, as a result of which the stochastic dynamics relaxes to them.

NB: in the computations, the key point is the equality \circledast , which stems from **q** being different from \pm **p** (by Cauchy-Schwarz).

On top of that, it can be noted that the network relaxes to the fixed-point pattern which is the *closest* to the starting pattern! For example, consider the second-to-last and the third-to-last examples in figure *2.c.2*:

- for both of these examples, in the starting pattern, we can distinguish what looks like the base of a (future) screen
- but the the second-to-last example converges to a bright-screen computer, whereas the thirdto-last converges to a dark-screen one. This can be accounted for by the fact that there are more dark pixels in the third-to-last starting pattern than in the second-to-last one!

Therefore, we begin to see why Hopfield networks can be thought of as modeling an associative memory-like process: the starting patterns are similar to "visual clues", to which the network associates the best (i.e. the closest) matching stored pattern.

3. The general rule

The general rule defining the weight matrix is:

$$W \stackrel{\text{\tiny def}}{=} rac{1}{N} \sum_{i=1}^M \mathbf{p}_i \mathbf{p}_i^{\mathsf{T}}$$

where the \mathbf{p}_i are the patterns to be stored.

3 patterns: linear combinations stored

Let us add yet another pattern: $\mathbf{p}_{\mathrm{cof}}$:



Figure 3.d.1 - Three Patterns stored: Battery, Computer & Cup of coffee

so that, in compliance with the general rule:

$$W = rac{1}{N} \Big(\mathbf{p}_{ ext{bat}} \mathbf{p}_{ ext{bat}}^\intercal + \mathbf{p}_{ ext{com}} \mathbf{p}_{ ext{com}}^\intercal + \mathbf{p}_{ ext{cof}} \mathbf{p}_{ ext{cof}}^\intercal \Big)$$

Then, we simulate the network with random initial conditions:

Network simulations

Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Step 7	Step 8	Step 9	Step 10	Step 11	Step 12	Step 13	Step 14	Step 15	Step 16
<u> </u>	2		2	2	2	2	9	9	9	9	9	0 81	08	00	0
Step 17	Step 18	Step 19	Step 20	Step 21	Step 22	Step 23	Step 24	Step 25	Step 26	Step 27	Step 28	Step 29	Step 30	Step 31	Step 32
D 8	0	08	0	08	0	8									
Step 33	Step 34	Step 35	Step 36	Step 37	Step 38	Step 39	Step 40	Step 41	Step 42	Step 43	Step 44	Step 45	Step 46	Step 47	Step 48
α,		•						8							
Step 49	Step 50	Step 51	Step 52	Step 53	Step 54	Step 55	Step 56	Step 57	Step 58	Step 59	Step 60	Step 61	Step 62	Step 63	Step 64
							•	•		•					
Step 65	Step 66	Step 67	Step 68	Step 69	Step 70	Step 71	Step 72	Step 73	Step 74	Step 75					
					•	•	•		•	•					

Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13	Step 15	Step 17	Step 19	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31
18 CC	2 C -	50	10 C.	6.0											
10									.			- A	- L	- L	- 5
Chan 22	Chan 35	Chan 37	Chan 20	Shan 41	Chan 43	Chan 45	Shan 47	Shan 40	Chan 51	Chan 53	Chan EE	Chan 57	Stan 50	Shan 61	Chan 63
Step 33	Step 35	Step 37	Step 39	Step 41	Step 43	Step 45	Step 47	Step 49	Step 51	Step 53	Step 55	Step 57	Step 59	Step 61	Step 63
	-					-		10 A							
Step 65	Step 67	Step 69	Step 71	Step 73	Step 75	Step 77	Step 79	Step 81	Step 83	Step 85	Step 87	Step 89	Step 91	Step 93	Step 95
		100													
- L	- A		- L		2.5	- L	- L	- I.	<u></u>	- L	2.5	- L	- L	2.5	- L
								~						-	
Step 97	Step 99	Step 101	Step 103	Step 105	Step 107	Step 109	Step 111	Step 113	Step 115	Step 117	Step 119	Step 121	Step 123	Step 125	Step 127
1 11			1 11	1	11	11	2.2	22	2.2	2.2	2.2	1	11		
			-		-		-		-				-		
Step 129	Step 131	Step 133	Step 135	Step 137	Step 139	Step 141	Step 143	Step 145	Step 147	Step 149	Step 151	Step 153	Step 155	Step 157	Step 159
	2.2			2.1	2.1										
Step 161	Step 163	Step 165	Step 167												
10 B															
			- A												

·····································	p 59 Step 61 Step 63
	p 59 Step 61 Step 63
Step 33 Step 35 Step 37 Step 39 Step 41 Step 43 Step 45 Step 47 Step 49 Step 51 Step 53 Step 55 Step 57 Step	
	c 52 52
Step 65 Step 67 Step 69 Step 71 Step 73 Step 75 Step 77 Step 79 Step 81 Step 83 Step 85 Step 87 Step 89 Step	p 91 Step 93 Step 95
Step 97 Step 99 Step 101 Step 103 Step 105 Step 107 Step 109 Step 111 Step 113 Step 115 Step 117 Step 119 Step 121 Step 1	123 Step 125 Step 127
Step 129 Step 131 Step 133 Step 135 Step 137 Step 139 Step 141 Step 143 Step 145 Step 147 Step 149 Step 151 Step 153 Step	155
프 프 프 프 프 프 프 프 프 프 프 프	



Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13	Step 15	Step 17	Step 19	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31
÷,			1	1	1	2.		23	23	23	33	\mathbf{n}	\mathbf{n}	$\mathbf{r}_{\mathbf{r}}$	\mathbf{n}
Step 33	Step 35	Step 37	Step 39	Step 41	Step 43	Step 45	Step 47	Step 49	Step 51	Step 53	Step 55	Step 57	Step 59	Step 61	Step 63
\mathbf{n}	\mathcal{D}	\mathbf{n}	\mathbf{n}	\mathbf{n}	\mathbf{n}	\mathbf{n}	$\mathbf{r}_{\mathbf{i}}$	\mathbf{n}	\mathbf{n}	2	\mathbf{n}	22	2	$\mathbf{T}_{\mathbf{r}}$	21
Step 65	Step 67	Step 69	Step 71	Step 73	Step 75	Step 77	Step 79	Step 81	Step 83	Step 85	Step 87	Step 89	Step 91	Step 93	Step 95
2	91	2_{1}	2_{1}	\mathbf{n}	\mathbf{n}	21	\mathbf{n}	\mathbf{r}_{1}	<u>ц</u>	9	22	51	51	51	51
Step 97	Step 99	Step 101	Step 103	Step 105	Step 107	Step 109	Step 111	Step 113	Step 115	Step 117	Step 119	Step 121	Step 123	Step 125	Step 127
91	4	$\mathbf{r}_{\mathbf{i}}$	91	31	\mathbf{n}	21	\mathbf{n}	\mathbf{n}	22	20	21	\mathbf{n}	\mathbf{n}	\mathbf{n}	\mathbf{n}
Step 129	Step 131	Step 133	Step 135	Step 137	Step 139	Step 141	Step 143	Step 145	Step 147	Step 149	Step 151	Step 153	Step 155	Step 157	Step 159
33	51	\mathbf{n}	2	\mathbf{n}	\mathbf{x}_{1}	\mathbf{r}_{1}	\mathbf{n}	\mathbf{n}	\mathbf{x}_{1}	\mathbf{n}	\mathbf{n}	1	\mathbf{n}	\mathbf{n}	\mathbf{n}
Step 161	Step 163	Step 165	Step 167	Step 169											
22	11	22	21	21											



Figure 3.d.2 - **Three Patterns (battery, computer & coffee) stored**: Network simulation from different starting patterns

It appears that the networks relaxes to the following linear combinations of $\{\mathbf{p}_{bat}, \mathbf{p}_{con}, \mathbf{p}_{cof}\}$ too:

- $\mathbf{p}_{\text{bat}} \mathbf{p}_{\text{com}} \mathbf{p}_{\text{cof}}$
- $\mathbf{p}_{\text{bat}} \mathbf{p}_{\text{com}} + \mathbf{p}_{\text{cof}}$



Figure 3.d.3 - **Three Patterns (battery, computer & coffee) stored**: Linear combinations $\mathbf{p}_{\text{bat}} - \mathbf{p}_{\text{com}} - \mathbf{p}_{\text{cof}}$ and $\mathbf{p}_{\text{bat}} - \mathbf{p}_{\text{cof}}$ (respectively) that are also stored by the network

So: on top of the patterns from which the weight matrix is constructed and their opposites, linear combinations thereof may also be stored.

Storing capacity

Finally, we will investigate how many patterns we can store in this N=64 neurons network by:

• adding new patterns, one after another

• and then simulating the network with each pattern as initial condition to see if the pattern is stored or not

Storing 1 pattern



Network simulation starting from each pattern



Figure 4.d.1 - **Weight matrix constructed out of one pattern (battery)**: network simulation from this pattern (to check that it is stored)

Storing 2 patterns:



Network simulation starting from each pattern

Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13	Step 15	Step 17	Step 19
Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13	Step 15	Step 17	

Figure 4.d.2. - Weight matrix constructed out of two pattern (battery & computer): network simulation from these patterns (to check that they are stored)

Storing 3 patterns:



Network simulation starting from each pattern

Step 1	Step 4	Step 7	Step 10	Step 13	Step 16	Step 19	Step 22	Step 25	Step 28	Step 31	Step 34	Step 37	Step 40	Step 43	Step 46
•					•		•								
Step 49	Step 52	Step 55	Step 58	Step 61	Step 64	Step 67	Step 70	Step 73	Step 76	Step 79	Step 82	Step 85	Step 88	Step 91	Step 94
Step 97	Step 100	Step 103													
Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13	Step 15	Step 17	Step 19	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31
_															
Step 33	Step 35	Step 37	Step 39	Step 41	Step 43	Step 45	Step 47	Step 49	Step 51	Step 53	Step 55	Step 57	Step 59	Step 61	Step 63
Step 65	Step 67	Step 69	Step 71	Step 73	Step 75	Step 77	Step 79	Step 81							
Step 05	Step 07	Step 05		Step 75	Step 75	Step 77	Step 75	Step 01							
Step 1	Step 5	Step 9	Step 13	Step 17	Step 21	Step 25	Step 29	Step 33	Step 37	Step 41	Step 45	Step 49	Step 53	Step 57	Step 61
Ť3	Ĩ3	Č,	<u> </u>	Č3	3	Тз	13	÷3	1	1	1	Č,	÷3	1	1
Step 65	Step 69	Step 73	Step 77	Step 81	Step 85	Step 89	Step 93	Step 97	Step 101	Step 105	Step 109	Step 113	Step 117	Step 121	Step 125
Č3	Č3	Č,	Č,	Č,	Ť,	Č,	Ť,	Č,	Č,	Ť3	Ť,	Č,	Ť,	Ť3	Č,
Step 129	Step 133	Step 137	Step 141	Step 145	Step 149	Step 153	Step 157	Step 161	Step 165	Step 169	Step 173	Step 177	Step 181	Step 185	Step 189
Ť,	1	Ť,	Č,	Č,	Č,	Č,	Č,	Ť,	Ť,	Č,	Ť,	Ĩ,	Ĩ,	13	Ť,
Step 193	Step 197	Step 201	Step 205	Step 209	Step 213										

Figure 4.d.3 - **Weight matrix constructed out of three patterns (battery, computer & coffee)**: network simulation from these patterns (to check that they are stored)

Storing 4 patterns:



Network simulation starting from each pattern

Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13	Step 15	Step 17	Step 19	Step 21	Step 23	Step 25	Step 27	Step 29	
	•								ш,						
Stop 1	Stop 3	Ston 5	Stop 7	Stop 0	Stop 11	Stop 13	Stop 15	Stop 17	Stop 10	Stop 21	Ston 23	Ston 25	Stop 27	Stop 20	Stop 31
Step 1	Step 5	Step 5	Step 7	Step 9	Step 11	Step 15	Step 15	Step 17	Step 19	Step 21	Step 25	Step 25	Step 27	Step 23	Step 51
Step 33	Step 35	Step 37	Step 39	Step 41	Step 43	Step 45	Step 47	Step 49	Step 51	Step 53	Step 55	Step 57	Step 59		
												-	-		
Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13									
Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13									
Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13									
Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13									
Step 1	Step 3	Step 5 Step 5	Step 7	Step 9 Step 9	Step 11	Step 13	Step 15	Step 17	Step 19	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31
Step 1	Step 3	Step 5 Step 5	Step 7	Step 9 Step 9	Step 11 Step 11	Step 13 Step 13	Step 15	Step 17	Step 19	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31
Step 1 Step 1 Step 1	Step 3 Step 3 Step 3 Step 35	Step 5 Step 5 Step 37	Step 7 Step 7 Step 7 Step 39	Step 9 Step 9 Step 9 Step 41	Step 11 Step 11 Step 11 Step 43	Step 13 Step 13 Step 13 Step 45	Step 15	Step 17 Step 49	Step 19 Step 51	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31
Step 1 Step 1 Step 33	Step 3 Step 3 Step 35	Step 5 Step 5 Step 37	Step 7 Step 7 Step 39	Step 9 Step 9 Step 41	Step 11 Step 11 Step 43	Step 13 Step 13 Step 45	Step 15 Step 47	Step 17 Step 49	Step 19 Step 51	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31

Figure 4.d.4 - Weight matrix constructed out of four patterns (battery, computer, coffee & thumbs up): network simulation from these patterns (to check that they are stored)

Storing 5 patterns:



Figure 4.d.5 - Weight matrix constructed out of five patterns (battery, computer, coffee, thumbs up & toolbox): network simulation from these patterns (to check that they are stored)

Storing 6 patterns:



Network simulation starting from each pattern



Figure 4.d.6 - **Weight matrix constructed out of six patterns**: network simulation from these patterns (to check that they are stored)

Storing 7 patterns:



Network simulation starting from each pattern



Figure 4.d.7 - **Weight matrix constructed out of seven patterns**: network simulation from these patterns (to check that they are stored)

Storing 8 patterns:



Network simulation starting from each pattern

Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13	Step 15	Step 17	Step 19	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31
Step 33	Step 35	Step 37	Step 39	Step 41	Step 43	Step 45	Step 47	Step 49	Step 51	Step 53					
Step 1	Step 4	Step 7	Step 10	Step 13	Step 16	Step 19	Step 22	Step 25	Step 28	Step 31	Step 34	Step 37	Step 40	Step 43	Step 46
_		-	-	-	-	•	•	•	•	-	-	2	-	•	•
Step 49	Step 52	Step 55	Step 58	Step 61	Step 64	Step 67	Step 70	Step 73	Step 76	Step 79	Step 82	Step 85	Step 88	Step 91	Step 94
		-	•	-			-								
Step 97	Step 100	Step 103	Step 106	Step 109	Step 112	Step 115	Step 118	Step 121	Step 124	Step 127	Step 130	Step 133	Step 136	Step 139	Step 142
Step 145	Step 148	Step 151	Step 154	Step 157	Step 160	Step 163	Step 166	Step 169	Step 172						
•	-					9	9	2	9						
Step 1	Step 5	Step 9	Step 13	Step 17	Step 21	Step 25	Step 29	Step 33	Step 37	Step 41	Step 45	Step 49	Step 53	Step 57	Step 61
Ĩ3	3	1	÷3	Ő.	Č3	13	13	13	С3	<u> </u>	Сэ	1	Сş	Ű9	Ű3
Step 65	Step 69	Step 73	Step 77	Step 81	Step 85	Step 89	Step 93	Step 97	Step 101	Step 105	Step 109	Step 113	Step 117	Step 121	Step 125
÷.	é,	ě,	Č9	Č.	Ô,	Č,	÷,	Č.	÷,	÷,	÷.	Č,	ě,	Č,	Č.
Step 129	Step 133	Step 137	Step 141	Step 145	Step 149	Step 153	Step 157	Step 161	Step 165	Step 169	Step 173	Step 177	Step 181	Step 185	Step 189
ě,	è	Č,	è,	è,	1	÷,	÷.	Č,	Č,	ò,	ě,	ě,	ě,	Č.	ě,
Step 193	Step 197	Step 201	Step 205	Step 209	Step 213	Step 217	Step 221	Step 225	Step 229	Step 233	Step 237	Step 241	Step 245	Step 249	Step 253
- S	1.12	- 2	- 2	- A	1.5	1.2	1.5	1.5	1.2	- 2	1.12	12	1.2	1.2	- C
Step 257	Step 261	Step 265	Step 269	Step 273											

Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13	Step 15	Step 17	Step 19	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31
Step 33	Step 35	Step 37	Step 39	Step 41	Step 43	Step 45	Step 47	Step 49	Step 51	Step 53	Step 55	Step 57	Step 59	Step 61	Step 63
4											1		6		
Step 65	Step 67	Step 69	Step 71	Step 73	Step 75	Step 77	Step 79	Step 81	Step 83						
	62	62													
Step 1	Step 4	Step 7	Step 10	Step 13	Step 16	Step 19	Step 22	Step 25	Step 28	Step 31	Step 34	Step 37	Step 40	Step 43	Step 46
Step 49	Step 52	Step 55	Step 58	Step 61	Step 64	Step 67	Step 70	Step 73	Step 76	Step 79	Step 82	Step 85	Step 88	Step 91	Step 94
										1		-			
Step 97	Step 100	Step 103	Step 106	Step 109	Step 112	Step 115	Step 118	Step 121	Step 124	Step 127	Step 130	Step 133	Step 136		
Step 1	Step 4	Step 7	Step 10	Step 13	Step 16	Step 19	Step 22	Step 25	Step 28	Step 31	Step 34	Step 37	Step 40	Step 43	Step 46
6-1	<u>.</u>	4	<u>.</u>	4-	4	4	<u>.</u>	2	24	÷.,	÷.,	24	24	23	2
Step 49	Step 52	Step 55	Step 58	Step 61	Step 64	Step 67	Step 70	Step 73	Step 76	Step 79	Step 82	Step 85	Step 88	Step 91	Step 94
	22									22,	23	22			
Step 97	Step 100	Step 103	Step 106	Step 109	Step 112	Step 115	Step 118	Step 121	Step 124	Step 127	Step 130	Step 133	Step 136		
	12	1	1		1							1			
Stop 1	Stop 3	Stop 5	Stop 7	Stop 0	Stop 11	Stop 13	Stop 15	Stop 17	Stop 10	Stop 21	Stop 23	Stop 25	Stop 27	Stop 20	Stop 31
Step 1	Step 5	Step 5	Step 7	Step 9	Step 11	Step 15	Step 15	Step 17	Step 19	Step 21	Step 25	Step 25	Step 27	Step 23	Step 51
Step 33	Step 35	Step 37	Step 39	Step 41	Step 43	Step 45	Step 47	Step 49	Step 51	Step 53	Step 55	Step 57	Step 59	Step 61	
33		33	33	33	33	3	33	3	3	33	33	33	33	33	
Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13	Step 15	Step 17	Step 19	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31
											а	Ξ	а	а	а
Step 33	Step 35	Step 37	Step 39	Step 41											

Figure 4.d.8 - **Weight matrix constructed out of eight patterns**: network simulation from these patterns: not all of them are stored!

Storing 9 patterns:



Network simulation starting from each pattern

Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13	Step 15	Step 17	Step 19	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31
Step 33	Step 35	Step 37	Step 39	Step 41	Step 43	Step 45	Step 47	Step 49	Step 51	Step 53	Step 55	Step 57	Step 59	Step 61	Step 63
	-														
Step 65	Step 67	Step 69	Step 71	Step 73	Step 75	Step 77									
Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13	Step 15	Step 17	Step 19	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31
			•	•	•			•	•	•					
Step 33	Step 35	Step 37	Step 39	Step 41	Step 43	Step 45	Step 47	Step 49	Step 51	Step 53	Step 55	Step 57	Step 59	Step 61	Step 63
			-						-						
Step 65	Step 67	Step 69													
Step 1	Step 5	Step 9	Step 13	Step 17	Step 21	Step 25	Step 29	Step 33	Step 37	Step 41	Step 45	Step 49	Step 53	Step 57	Step 61
2 A S	22	2.0	2 A	1 A A	1 A A	- A.	- A.	- A.	1.1	1.	1 A A				
2															
Step 65	Step 69	Step 73	Step 77	Step 81	Step 85	Step 89	Step 93	Step 97	Step 101	Step 105	Step 109	Step 113	Step 117	Step 121	Step 125
			12.5	2.5	5.2	2.2			- 12	12.5					
Step 129	Step 133	Step 137	Step 141	Step 145	Step 149	Step 153	Step 157	Step 161	Step 165	Step 169	Step 173	Step 177	Step 181	Step 185	Step 189
				10 A	1.1	1.1	1 A A	1.1	1.	1 A A	1.1	1 A 4	1 A A		1 A A
Step 193	Step 197	Step 201	Step 205	Step 209	Step 213	Step 217	Step 221	Step 225	Step 229	Step 233	Step 237	Step 241	Step 245		
Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13	Step 15	Step 17	Step 19	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31
Step 33	Step 35	Step 37	Step 39	Step 41	Step 43	Step 45	Step 47	Step 49							
100.0	100.00	10.0	1.04	1.1											
Step 1	Step 3	Step 5	Step 7	Step 9	Step 11	Step 13	Step 15	Step 17	Step 19	Step 21	Step 23	Step 25	Step 27	Step 29	Step 31
Stor 33	Stor 35	Stor 37	Stor 30	Stor 42	Stor 42										
Step 33	Step 35	Step 37	Step 39	Step 41	Step 43										
		-	-												



Figure 4.d.9 - **Weight matrix constructed out of nine patterns**: network simulation from these patterns to check that not all of them are stored

As a consequence:

- up until 6 (included) patterns: all the patterns are successfully stored
- from 7 pattern on, some are not properly stored, and it gets worse and worse as the number of patterns increases:
 - with 7 patterns: the computer screen is "prettified by bright pixels" (looking like light reflection!), but not stored unaltered
 - with 8 patterns: on top of the light reflection on the computer screen, the copy icon is not stored at all: the network converges to an out-of-shape computer screen. In the same vein, the phone pattern ends up being deformed.
 - with 9 patterns: the battery pattern is not stored anymore, the computer base is shifted, the cup of coffee, the thumbs up, the phone and the arrow are distorted, and the copy icon keeps converging toward the computer (they must look too similar for the full network's pattern memory).

On the whole: the 64-neurons Hopfield network can perfectly store 6 patterns (and almost 7 patterns, if it were not for the computer pattern ending up with a few bright pixels on the screen).