AT2 – Neuromodeling: Problem set #4 NETWORKS

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
- $f(s) \overset{\text{def}}{=} 50\left(1 + \tanh(s)\right)$ the neuron's activation function, where s is the overall neuron input:

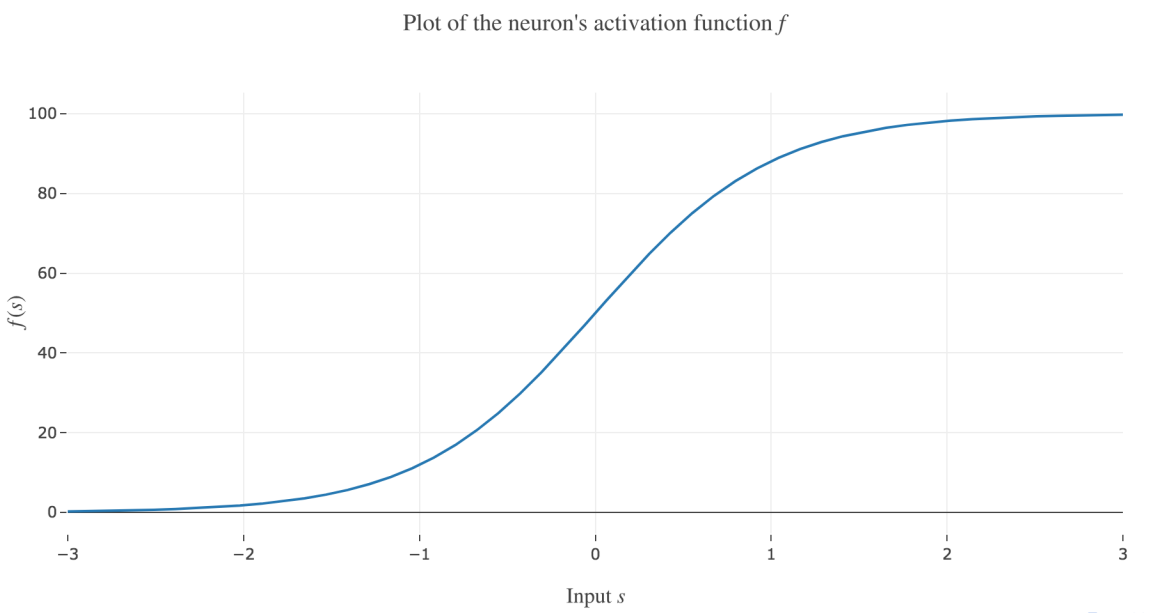


Figure 1.a. - Plot of the neuron's activation function f

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

$$\dot{x}(t) = -x(t) + f\big(wx(t) + I\big)$$

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

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

results in the following 1-dimensional system dynamics:

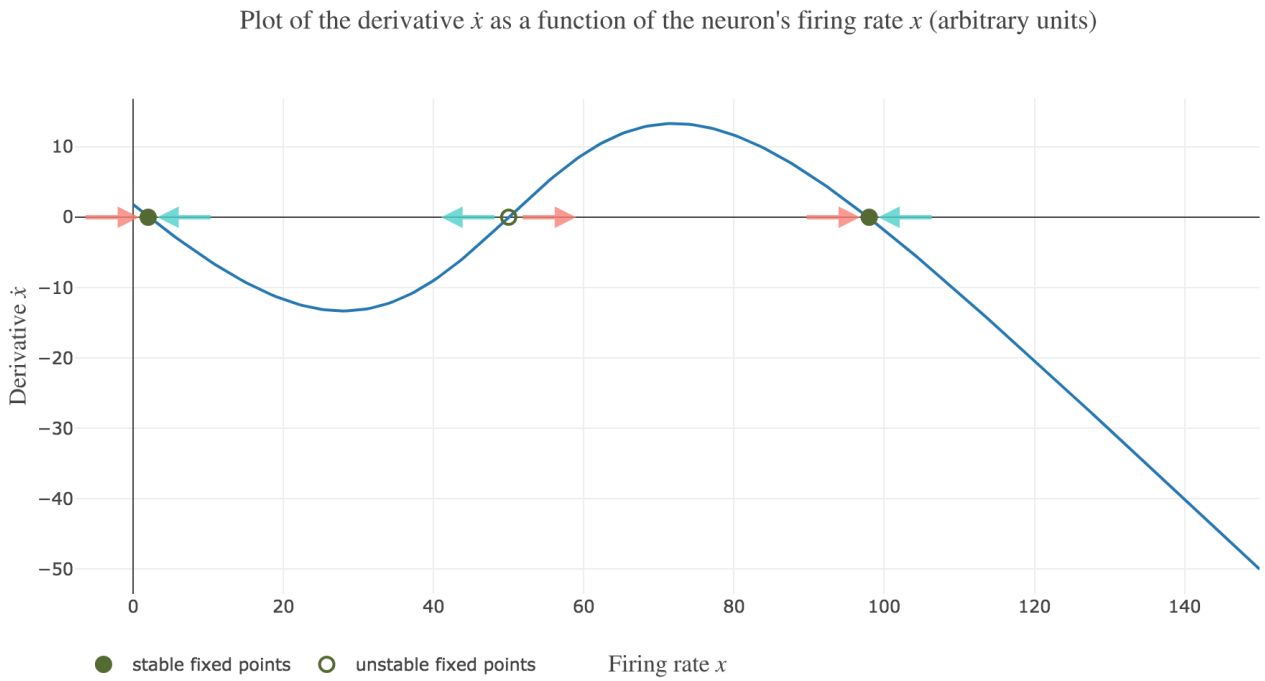


Figure 1.b. - Plot of the derivative as a function of the neuron's firing rate

The zero-crossings of $\dot{x}$ as a function of x correspond to critical points, *i.e.* values of x at which the derivative $t \longmapsto \dot{x}(t)$ equals zero, hence the function $t \longmapsto x(t)$ 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:

$$x(0) \in \left\{ \overbrace{49}^{\text{just below the unstable fixed point}}, \underbrace{50}_{\text{the unstable fixed point}}, \overbrace{51}^{\text{just above the unstable fixed point}} \right\}$$

We set (arbitrary units):

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

so that the Euler method yields:

$$\begin{aligned} & \frac{x(t+\Delta t) - x(t)}{\Delta t} = -x(t) + f\big(wx(t) + I\big) \\ \iff \quad & x(t+\Delta t) - x(t) = -x(t)\Delta t + f\big(wx(t) + I\big)\Delta t \\ \iff \quad & x(t+\Delta t) = (1 - \Delta t)x(t) + f\big(wx(t) + I\big)\Delta t \end{aligned}$$

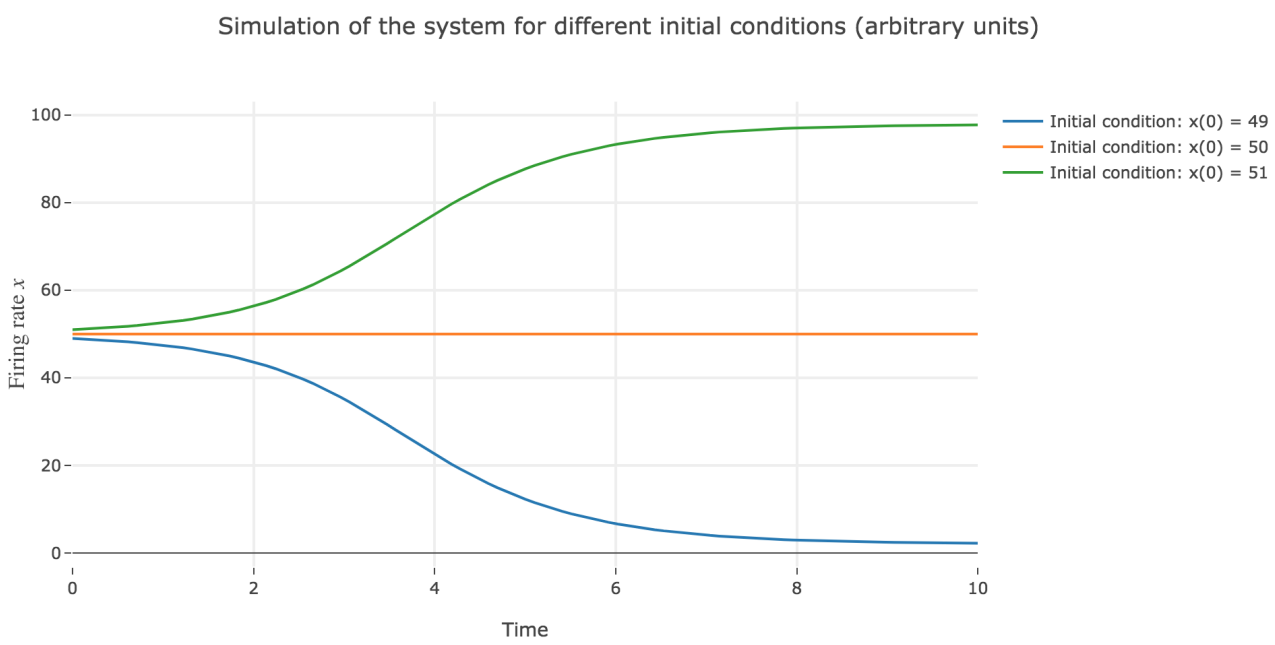


Figure 1.c. - Simulation of the system for different initial conditions

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:

$$\dot{x}(t) = -x(t) + f\big(wx(t) + I\big) + \sigma\eta(t)$$

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

The stochastic differential equation is solved numerically as follows:

$$\begin{aligned} & x(t + \Delta t) - x(t) = -x(t)\Delta t + f\big(wx(t) + I\big)\Delta t + \sigma\eta(t)\sqrt{\Delta t} \\ \iff \quad & x(t + \Delta t) = (1 - \Delta t)x(t) + f\big(wx(t) + I\big)\Delta t + \sigma\eta(t)\sqrt{\Delta t} \end{aligned}$$

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

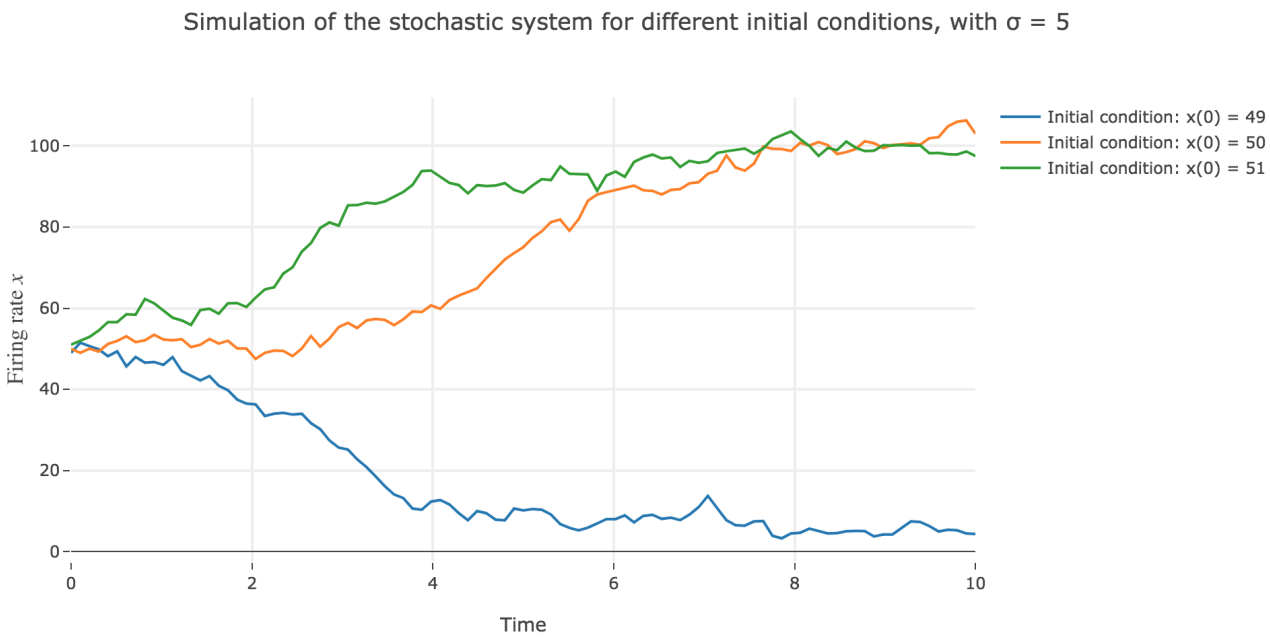


Figure 2.d.1.1 - First seed: Simulation of the stochastic system for different initial conditions, for a noise magnitude $\sigma = 5$

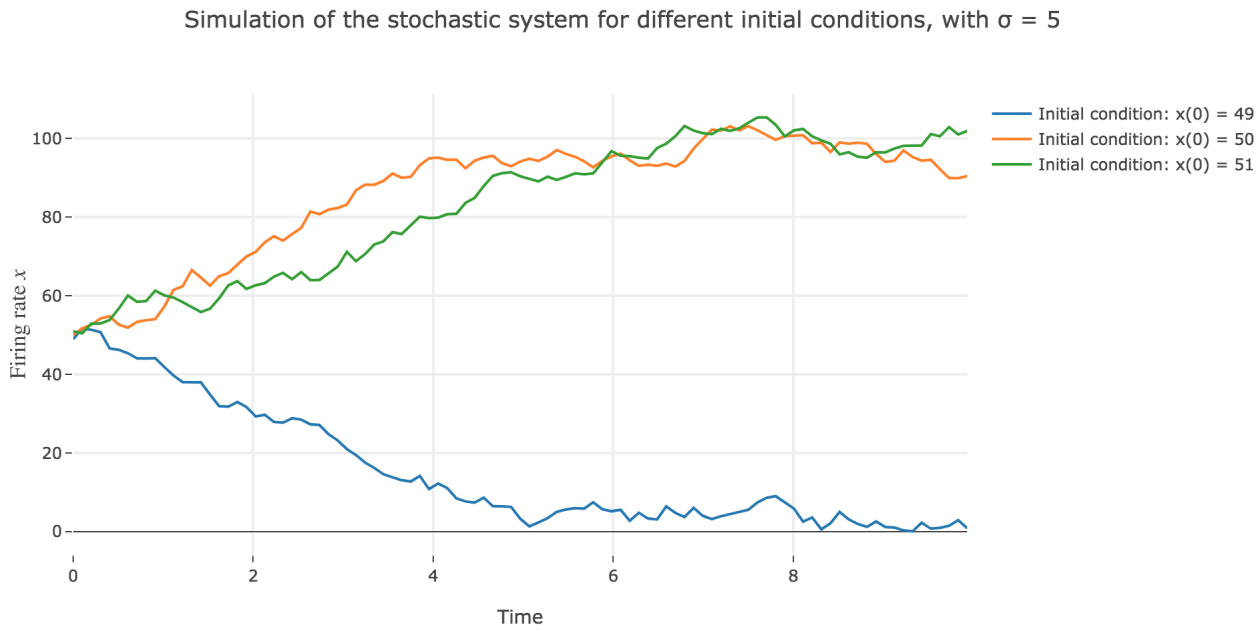


Figure 2.d.1.2 - Second seed: Simulation of the stochastic system for different initial conditions, for a noise magnitude $\sigma = 5$

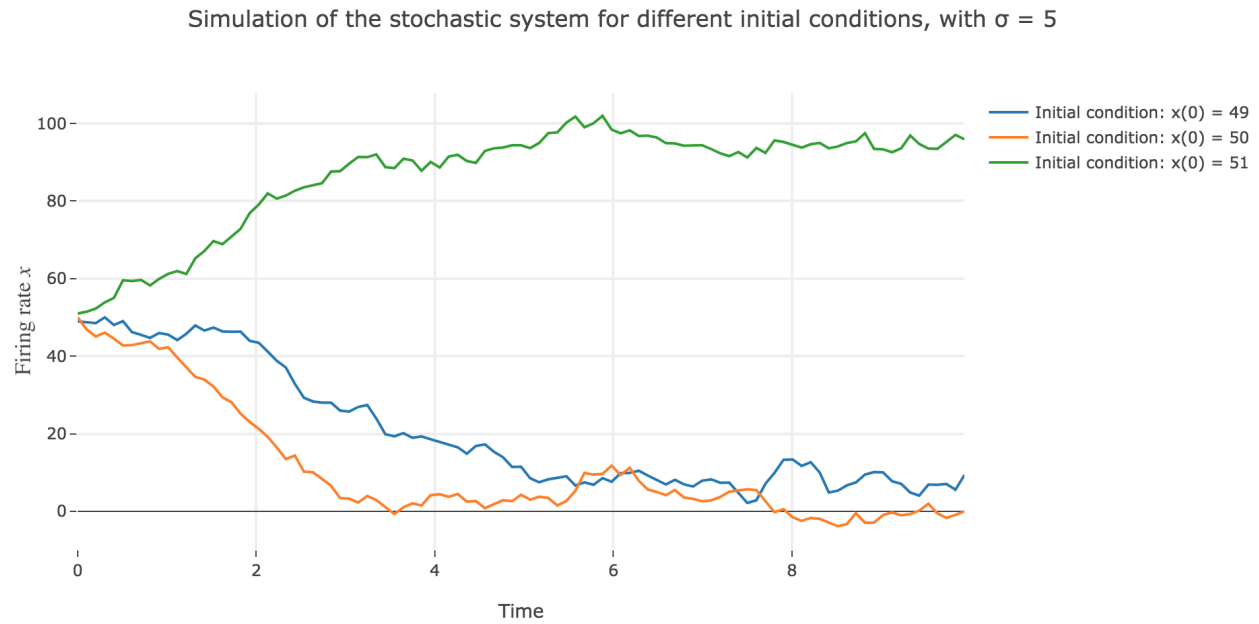


Figure 2.d.1.3 - Third seed: Simulation of the stochastic system for different initial conditions, for a noise magnitude $\sigma = 5$

- 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 $\dot{x}$, 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:

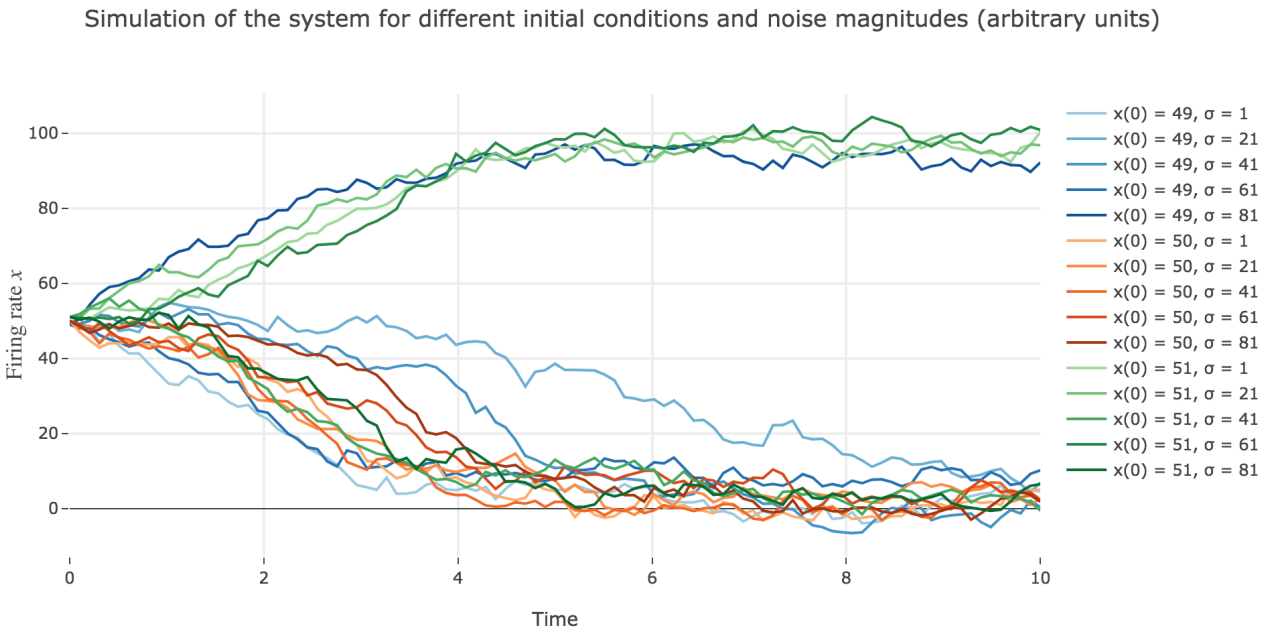


Figure 2.d.2.1 - First seed: Simulation of the stochastic system with $x(0) \in \{49, 50, 51\}$, for different noise magnitudes

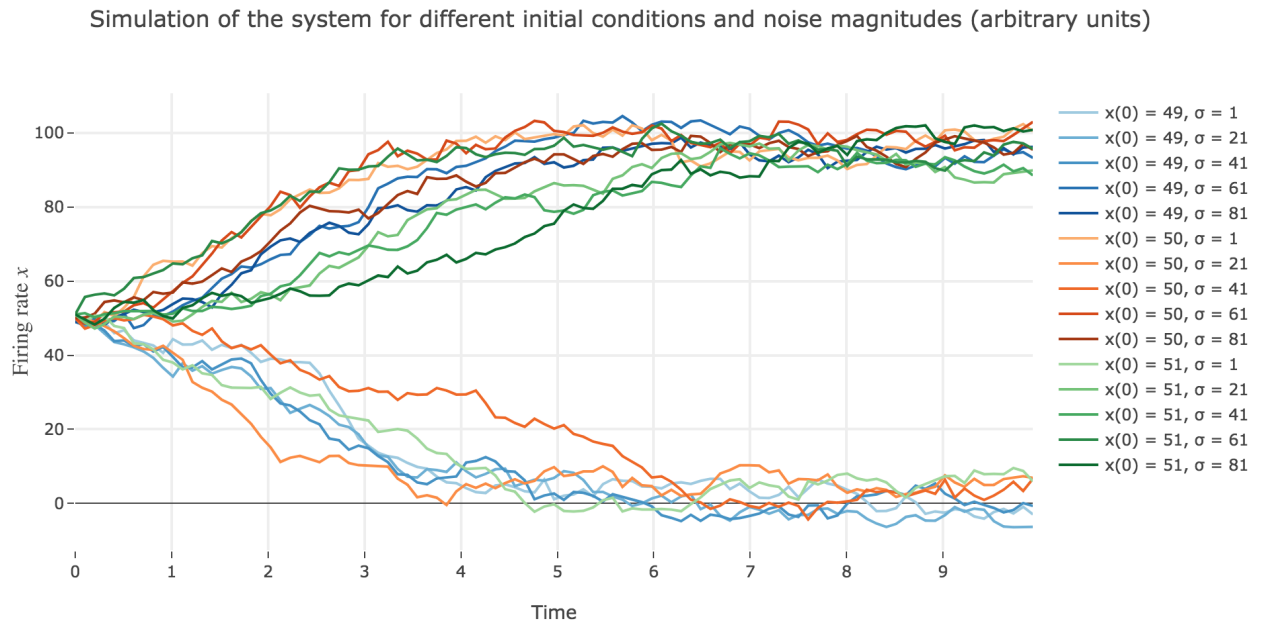


Figure 2.d.2.2 - Second seed: Simulation of the stochastic system with $x(0) \in \{49, 50, 51\}$, for different noise magnitudes

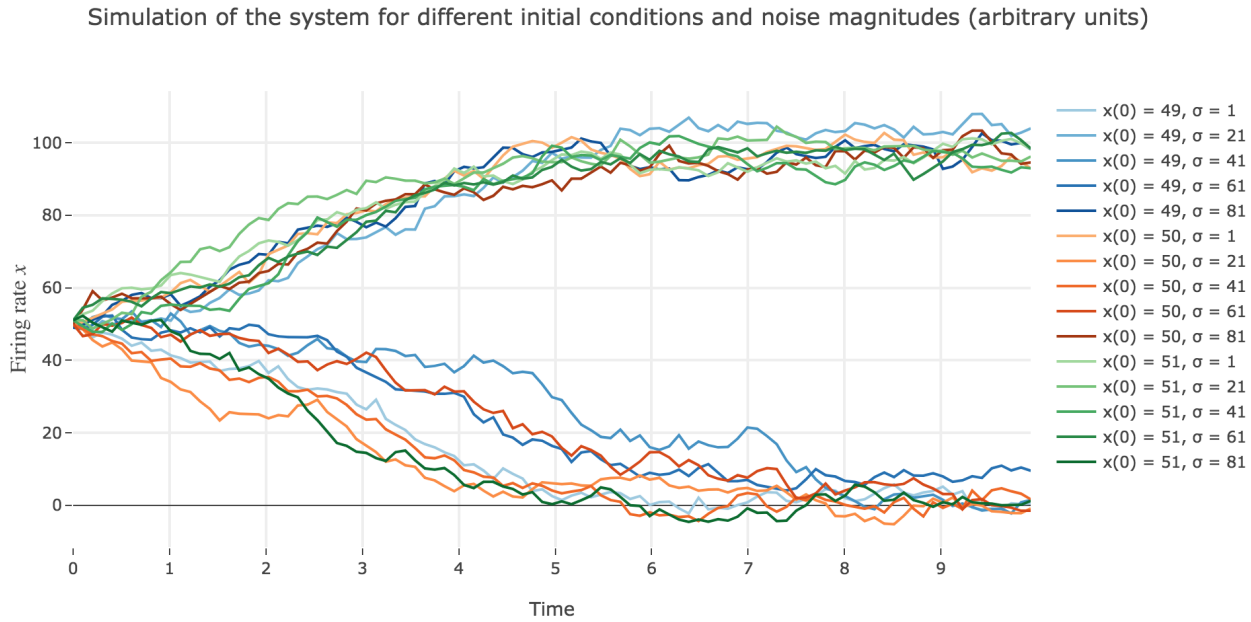


Figure 2.d.2.3 - Third seed: Simulation of the stochastic system with $x(0) \in \{49, 50, 51\}$, for different noise magnitudes

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 ($x(0) \in \{49, 51\}$):

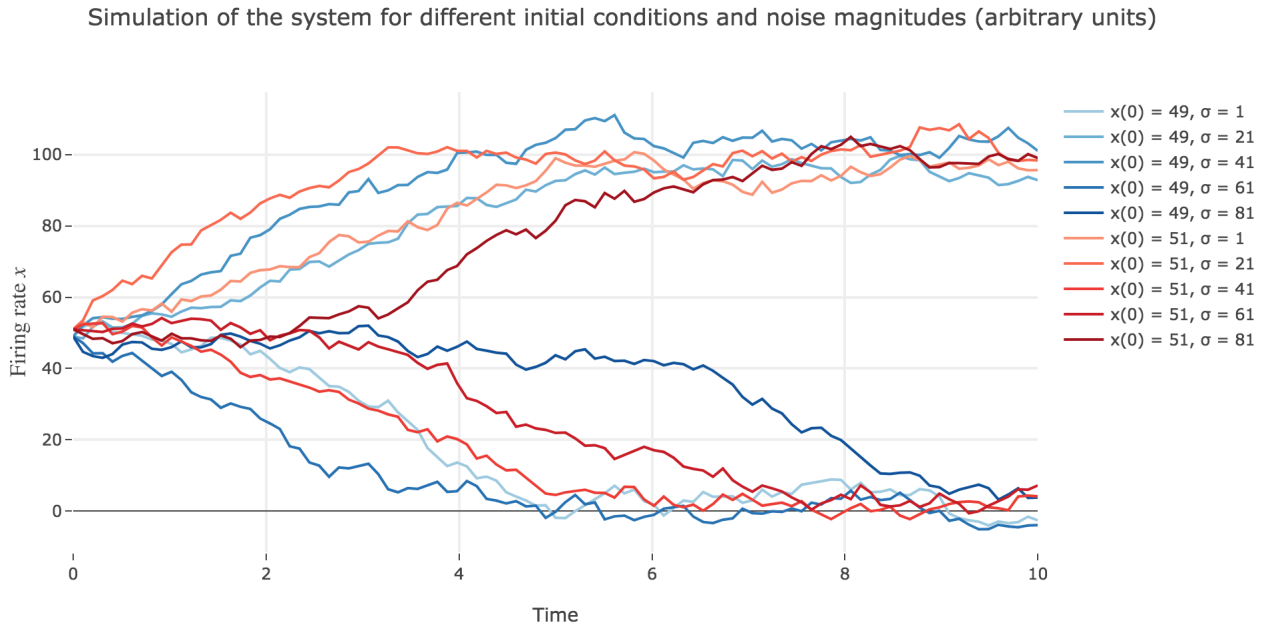


Figure 2.d.3.1 - First seed: Simulation of the stochastic system with $x(0) \in \{49, 51\}$, for different noise magnitudes

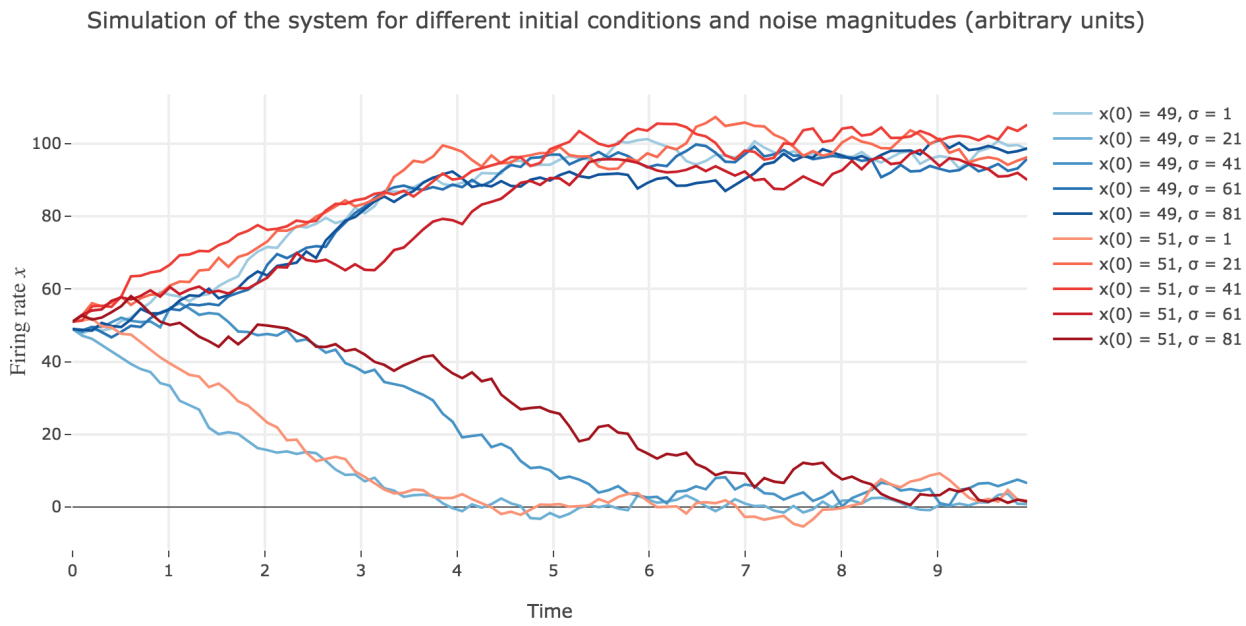


Figure 2.d.3.2 - Second seed: Simulation of the stochastic system with $x(0) \in \{49, 51\}$, for different noise magnitudes

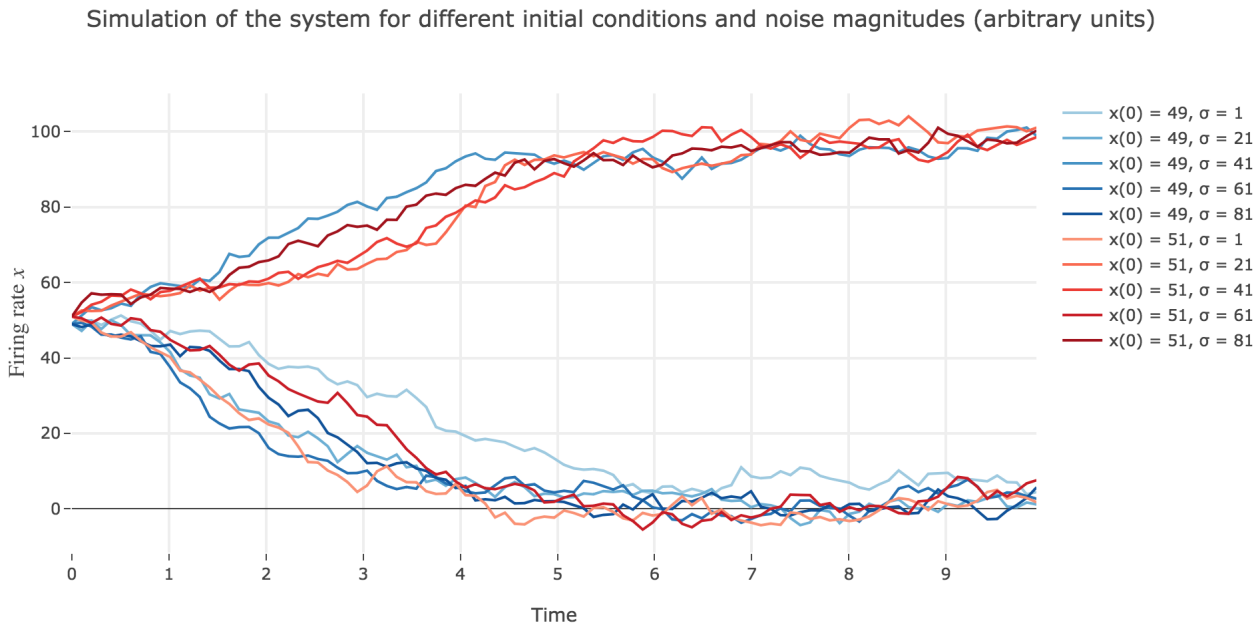


Figure 2.d.3.3 - Third seed: Simulation of the stochastic system with $x(0) \in \{49, 51\}$, for different noise magnitudes

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...:

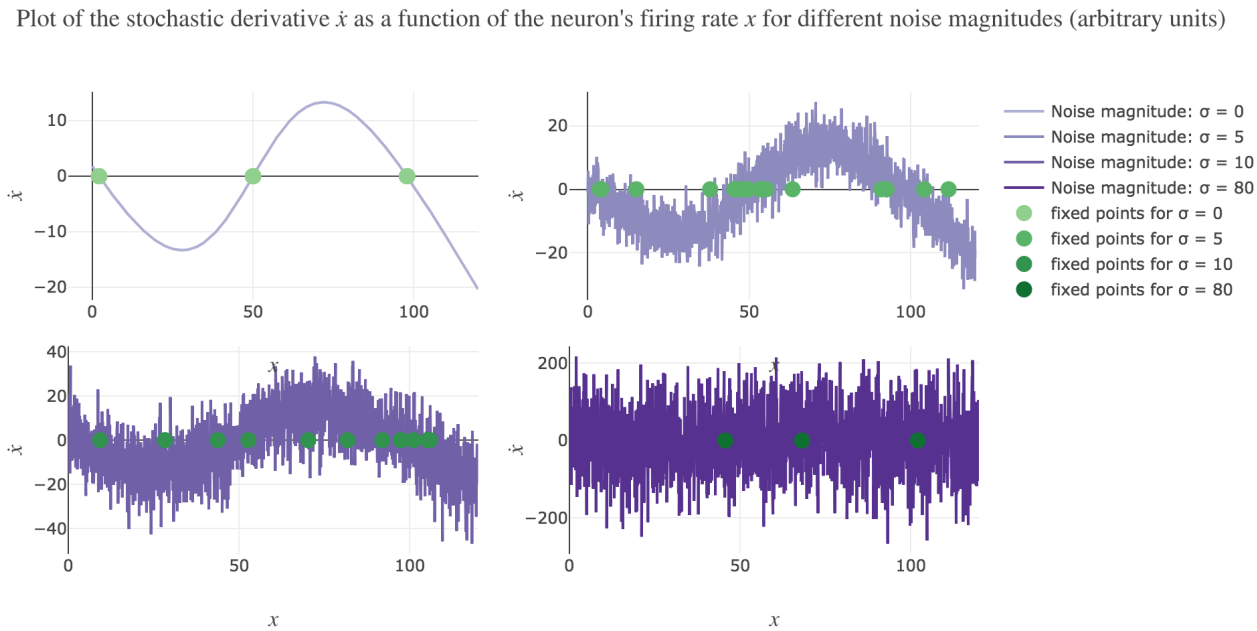


Figure 2.d.4 - Plot of the stochastic derivative as a function of the firing rate (and fixed points of the dynamics), for different noise magnitudes

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

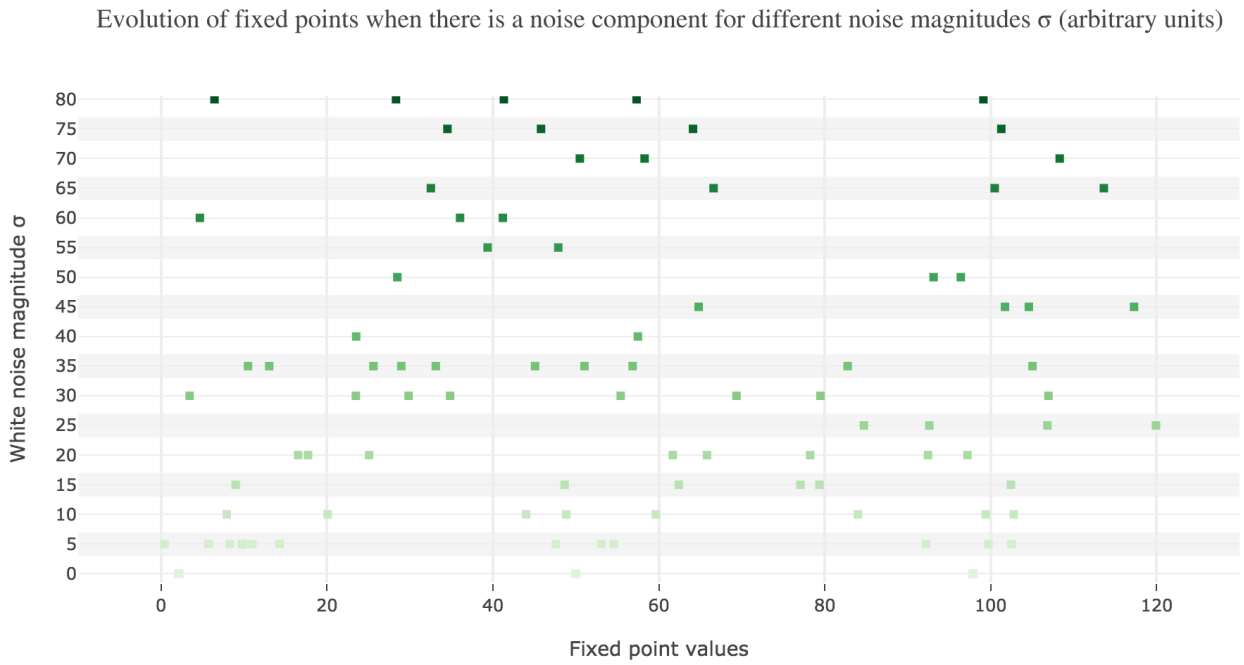


Figure 2.d.5.1 - First seed: Evolution of the stochastic dynamics fixed points for different noise magnitudes

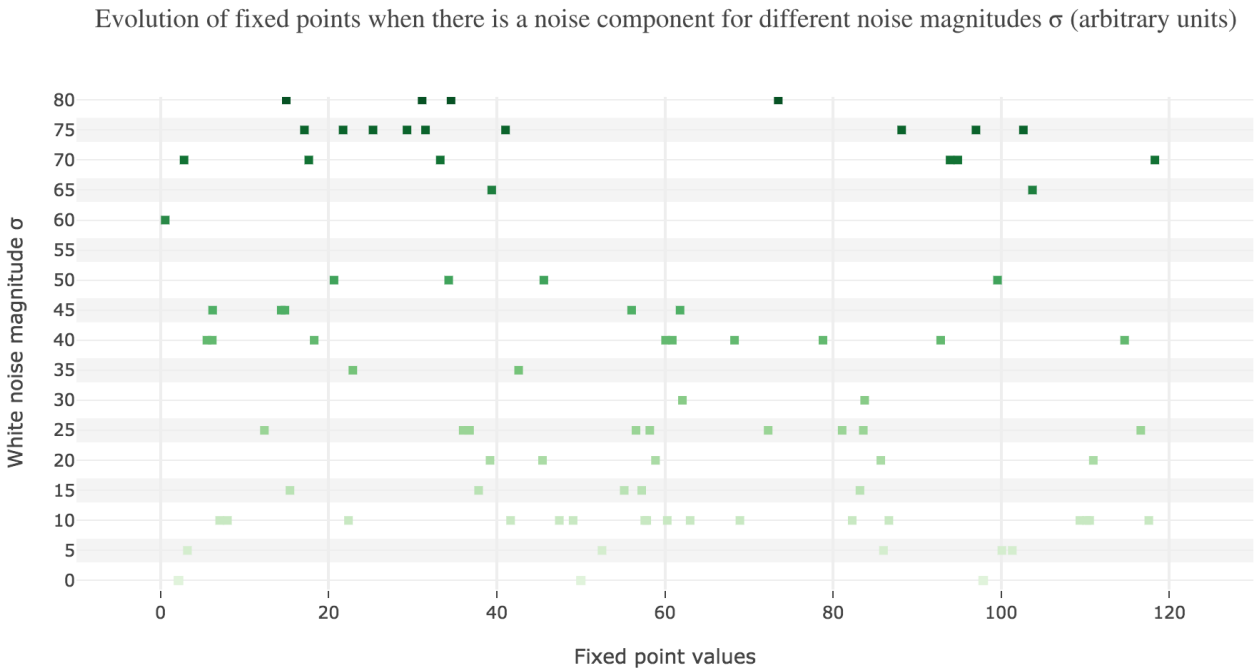


Figure 2.d.5.2 - Second seed: Evolution of the stochastic dynamics fixed points for different noise magnitudes

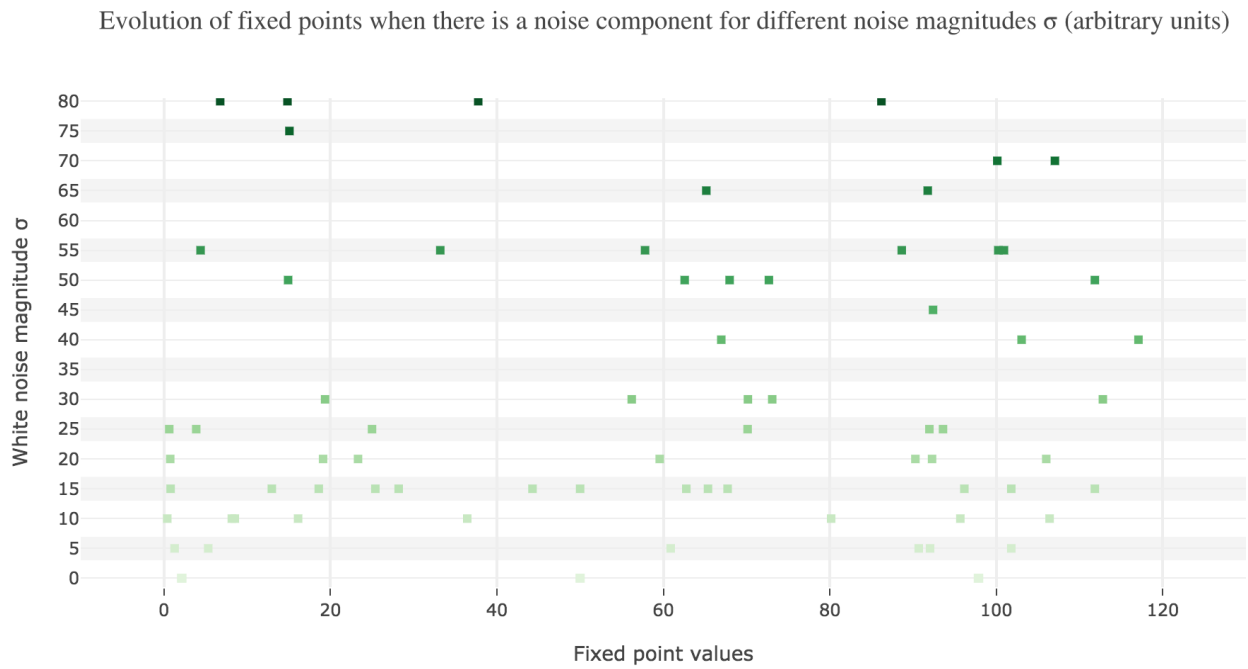


Figure 2.d.5.3 - Third seed: Evolution of the stochastic dynamics fixed points for different noise magnitudes

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

Link of the iPython notebook for the code

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
- $f(s) \stackrel{\text{def}}{=} 50\,(1 + \tanh(s))$ the neurons' activation function as before

1. Separate treatement of each neuron

We assume that we have the following differential equations:

$$\begin{cases} \dot{x}_1(t) = -x_1(t) + f(wx_2(t) + I) \\ \dot{x}_2(t) = -x_2(t) + f(wx_1(t) + I) \end{cases}$$

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

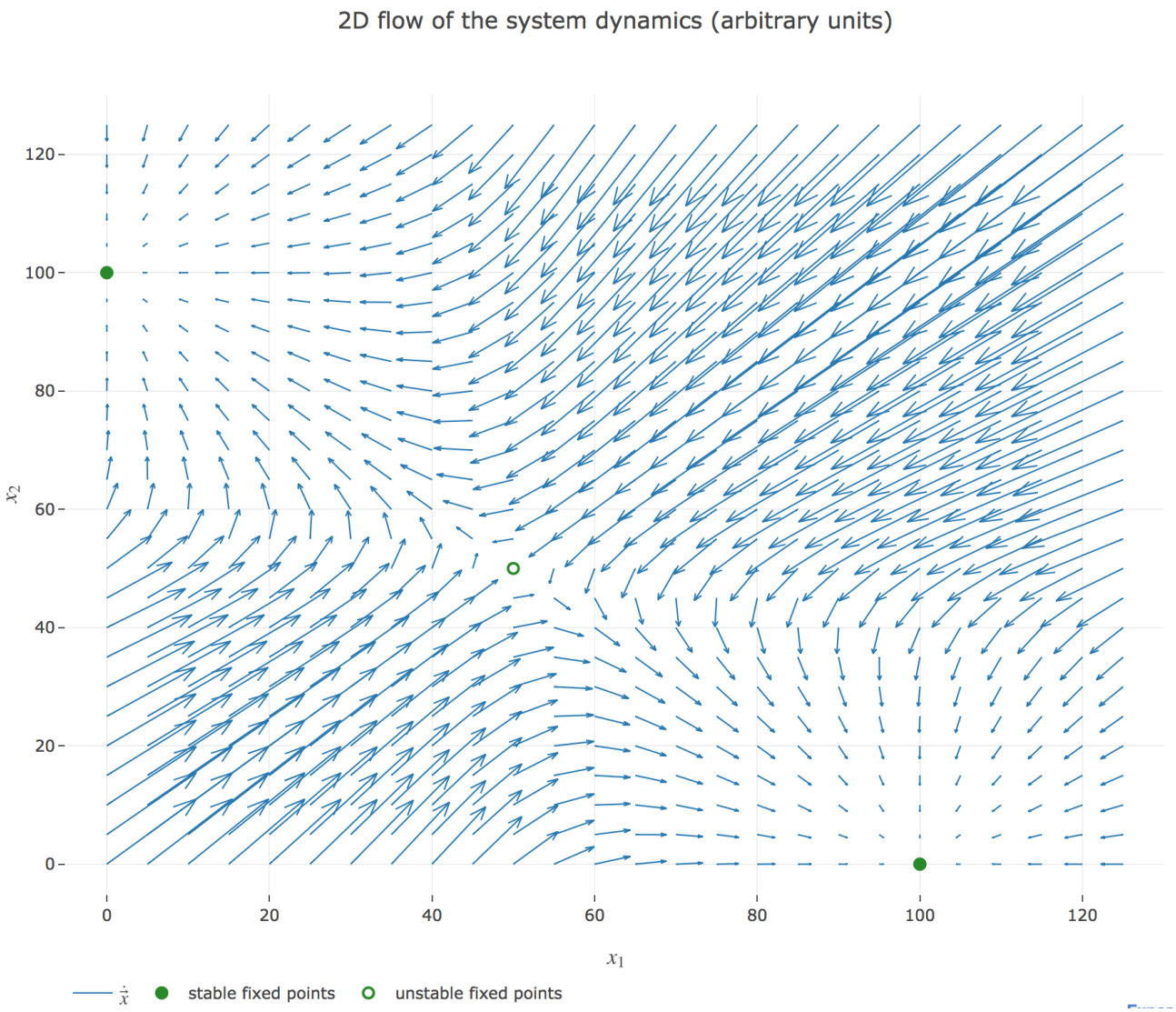


Figure 1.a.1 - 2D flow of the system dynamics

The nullclines of this system are the curves given by:

$$\begin{cases} \dot{x}_1(t) = 0 \\ \dot{x}_2(t) = 0 \end{cases}$$

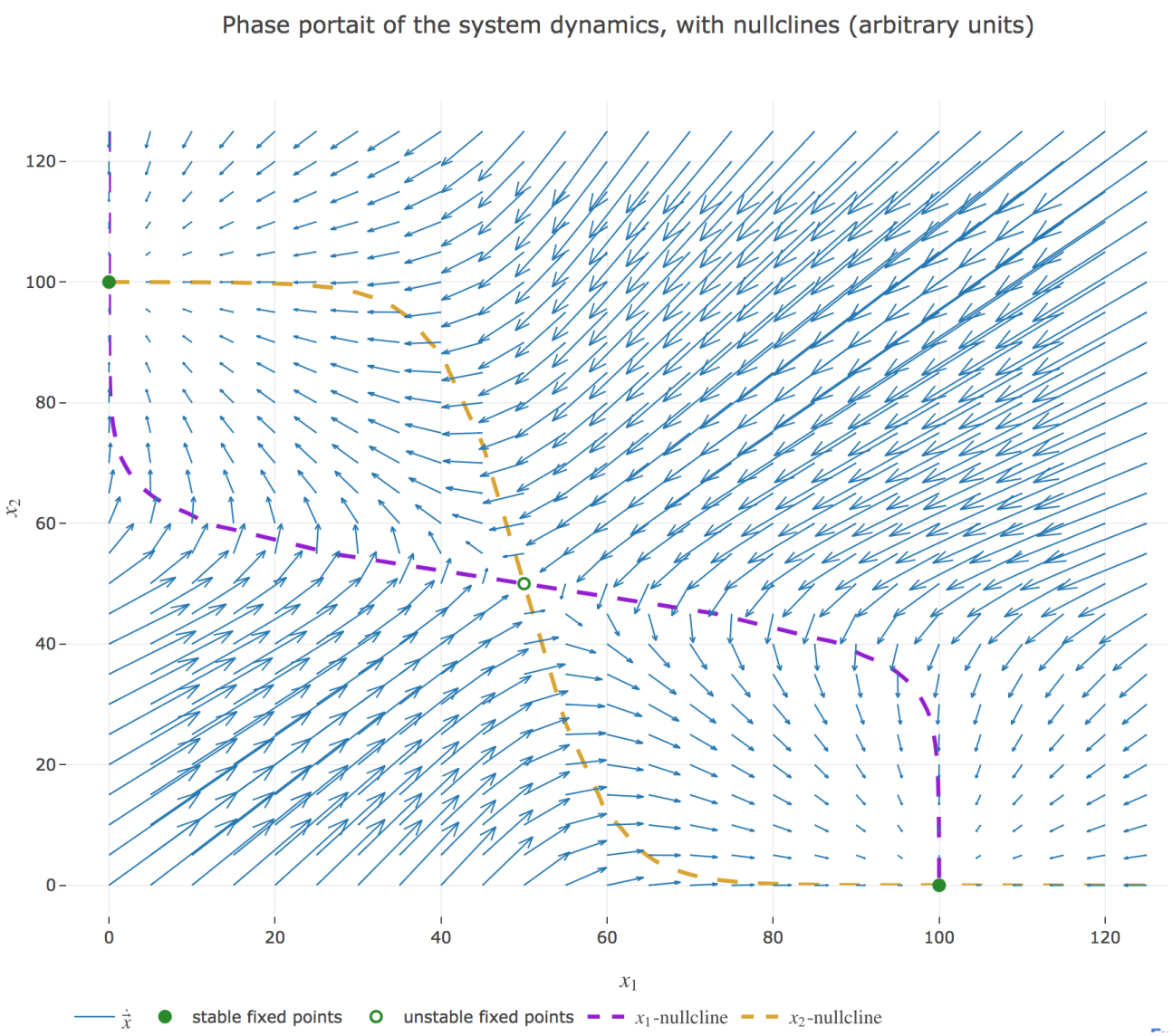


Figure 1.a.2 - Phase portrait of the system dynamics and nullclines

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 $(0, 100)$:

$$\begin{aligned} -0 + f(100w + I) &= f(-5) = 50(1 + \tanh(-5)) \simeq 0 \\ -100 + f(0w + I) &= f(5) - 100 = 50(\underbrace{1 + \tanh(5)}_{\simeq 2}) - 100 \simeq 0 \end{aligned}$$

  and it is symmetric for $(100, 0)$
- For $(50, 50)$:

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

System simulation

We set (arbitrary units):

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

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

$$\begin{aligned} & \frac{x_i(t + \Delta t) - x_i(t)}{\Delta t} = -x_i(t) + f\big(wx_j(t) + I\big) \\ \iff \quad & x_i(t + \Delta t) - x_i(t) = -x_i(t)\Delta t + f\big(wx_j(t) + I\big)\Delta t \\ \iff \quad & x_i(t + \Delta t) = (1 - \Delta t)x_i(t) + f\big(wx_j(t) + I\big)\Delta t \end{aligned}$$

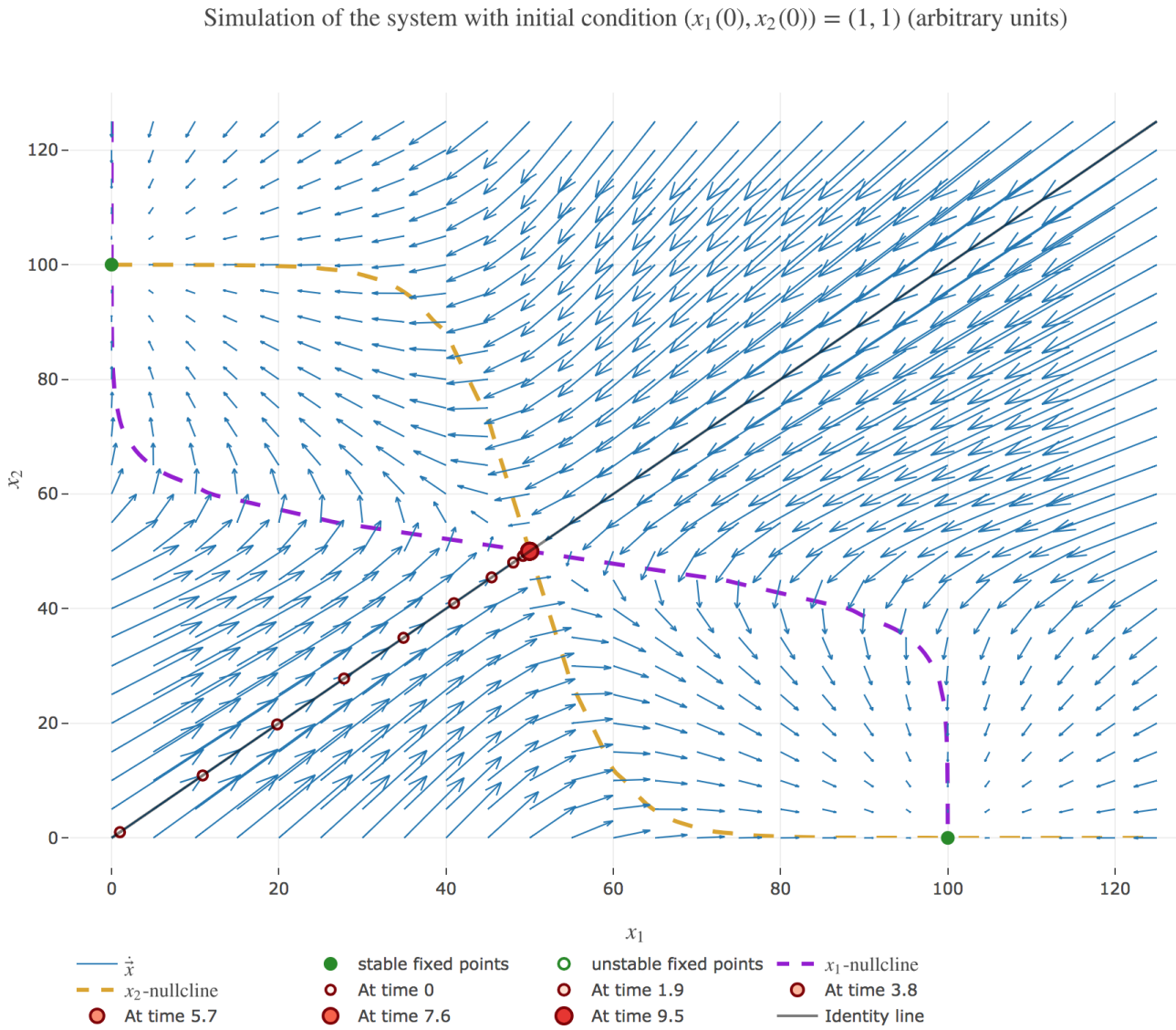


Figure 1.b.1 - Simulation of the system with $(x_1(0), x_2(0)) \overset{\text{def}}{=} (1, 1)$

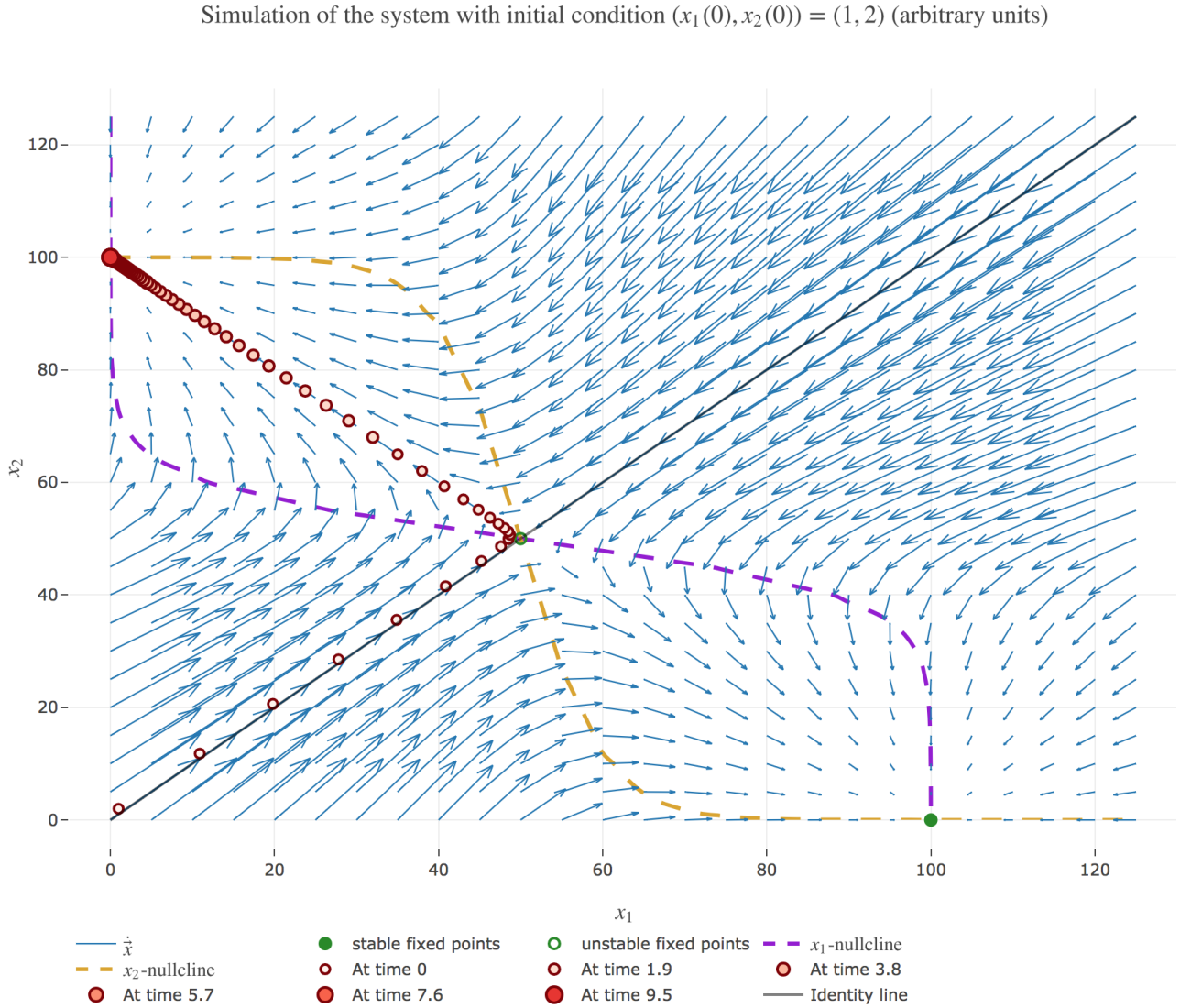


Figure 1.b.2 - Simulation of the system with $(x_1(0), x_2(0)) \stackrel{\text{def}}{=} (1, 2)$

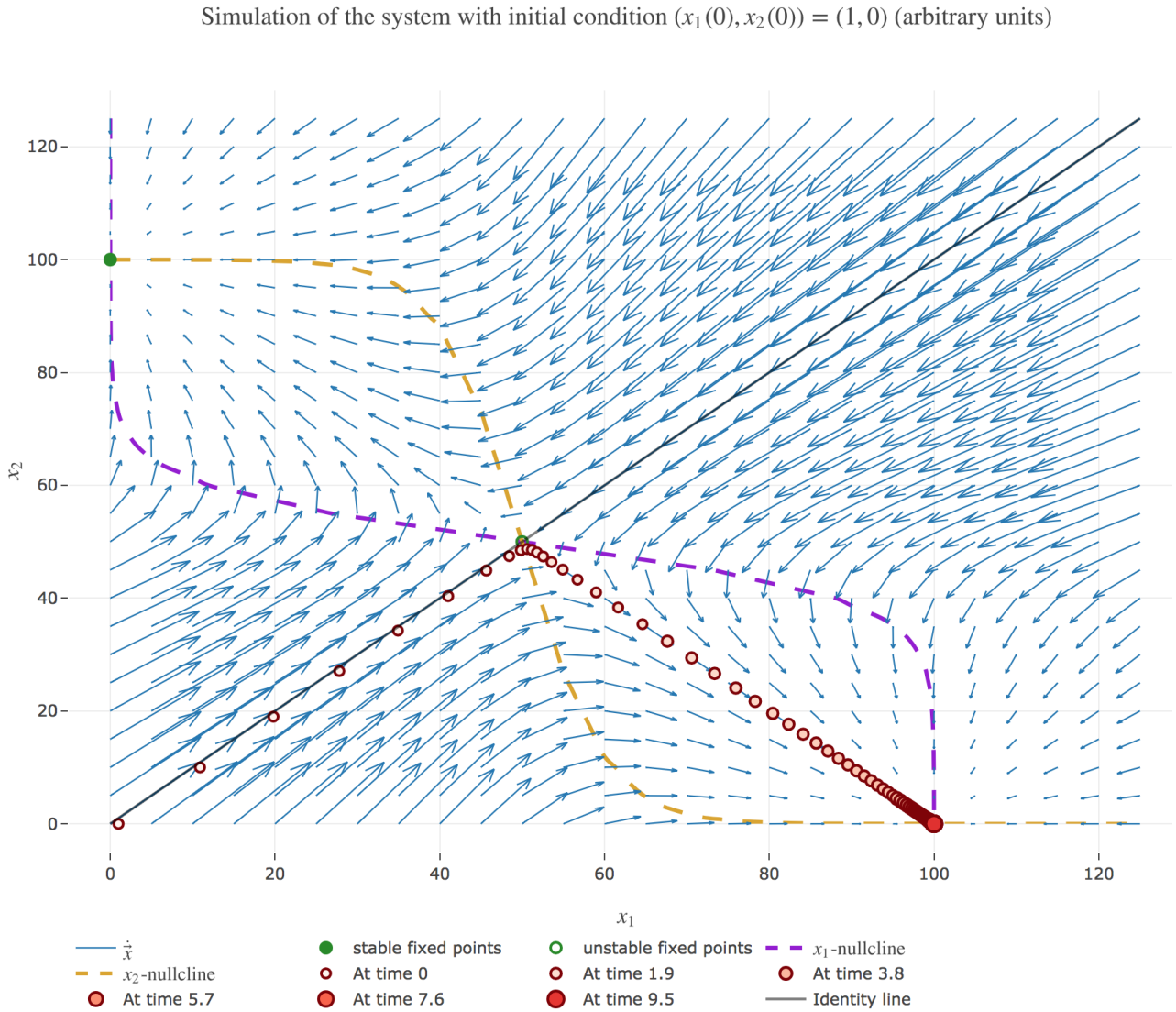


Figure 1.b.3 - Simulation of the system with $(x_1(0), x_2(0)) \overset{\text{def}}{=} (1, 0)$

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:

- $$\mathbf{x}(t) \overset{\text{def}}{=} \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} \in \mathfrak{M}_{2,1}(\mathbb{R})$$
- $$W \overset{\text{def}}{=} \begin{pmatrix} 0 & w \\ w & 0 \end{pmatrix} \in \mathfrak{M}_{2}(\mathbb{R})$$
- $$I \overset{\text{def}}{=} \begin{pmatrix} I \\ I \end{pmatrix} \in \mathfrak{M}_{2,1}(\mathbb{R})$$

as a result of which the Euler method gives:

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

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

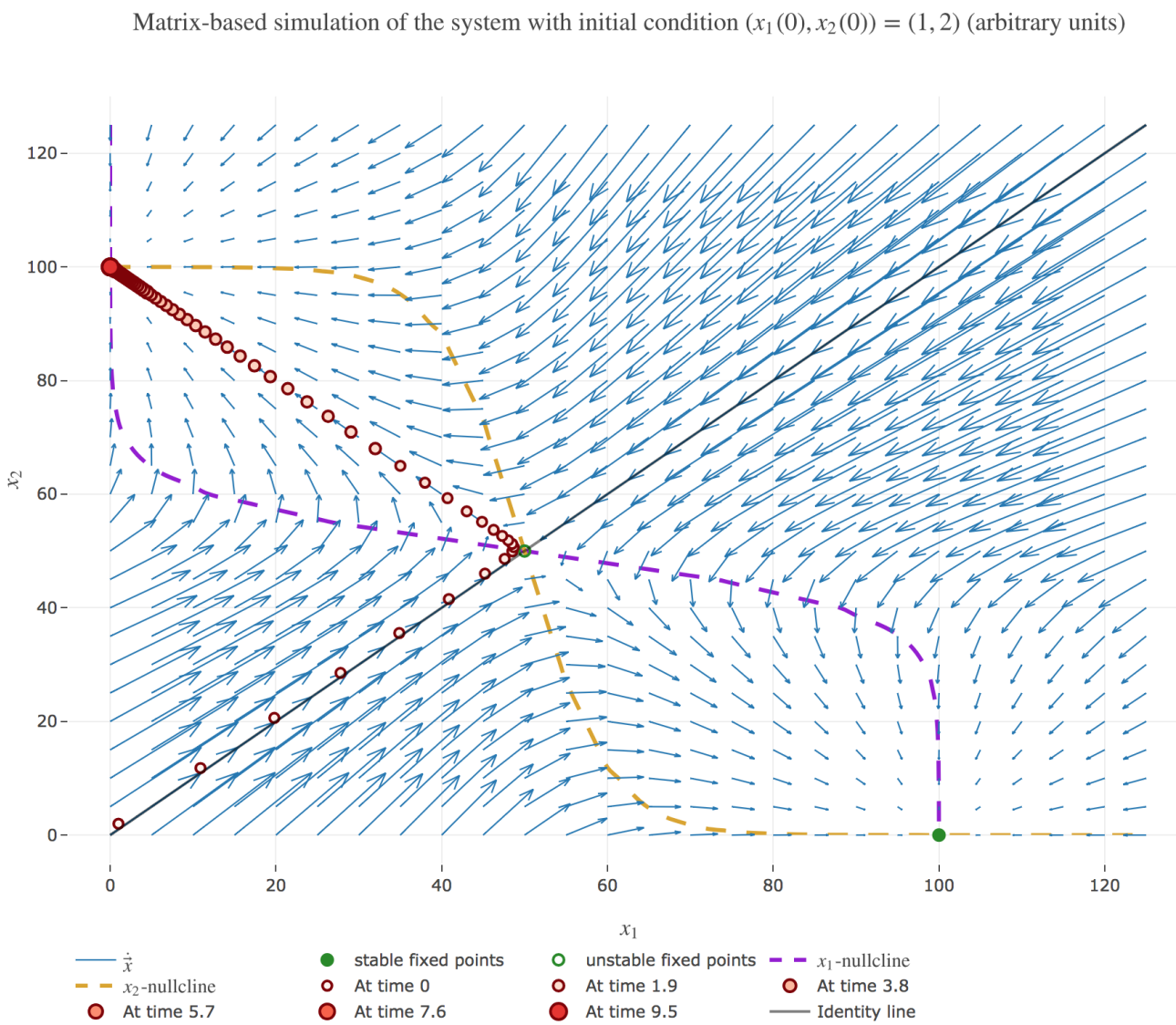


Figure 2.c. - Matrix-based simulation of the system with $(x_1(0), x_2(0)) \overset{\text{def}}{=} (1, 2)$

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

- $\mathbf{x}$ is the firing rates vector
- W is the $N \times N$ weight matrix
- $\eta(t)$ is a Gaussian noise of magnitude $\sigma = 0.1$
- the activation function $f \stackrel{\text{def}}{=} \text{sign}$ 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:

$$W \stackrel{\text{def}}{=} \frac{1}{N}\mathbf{p}\mathbf{p}^\intercal$$

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

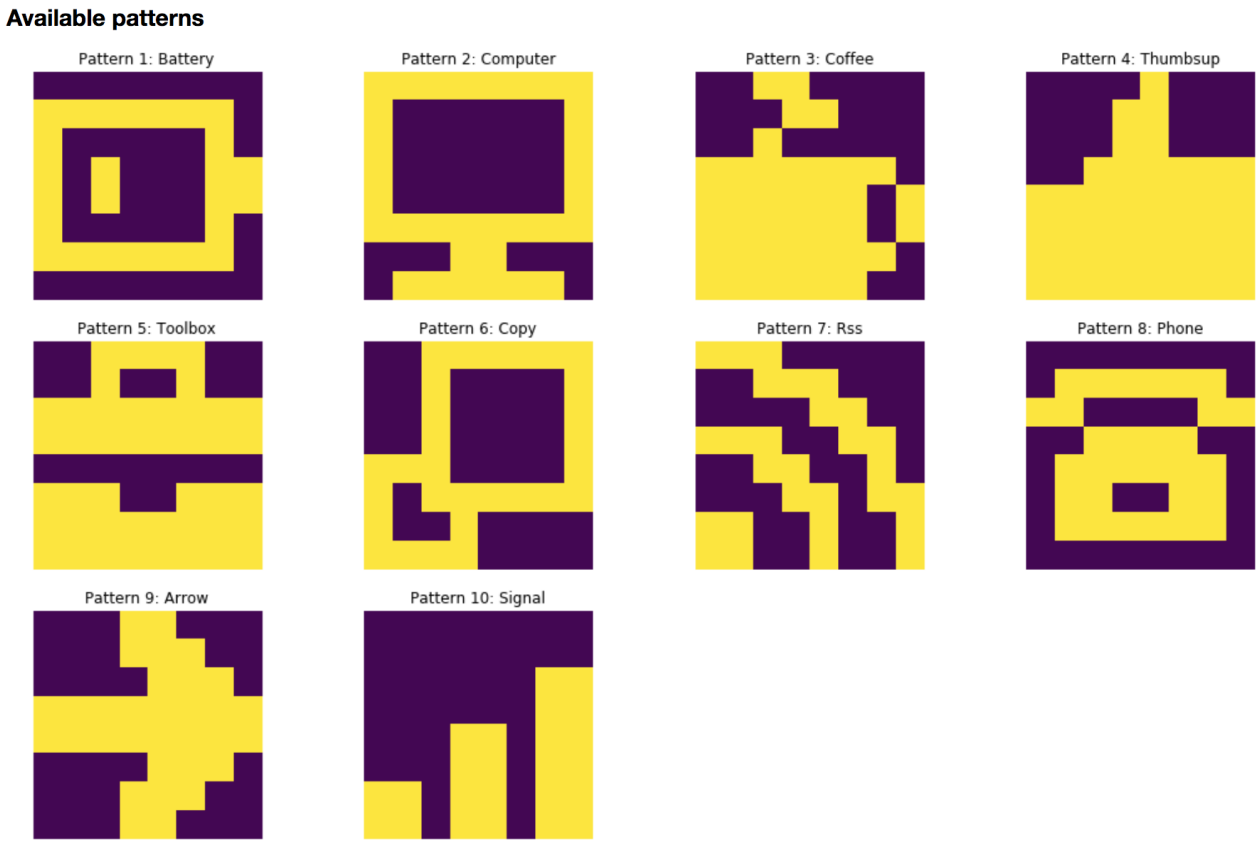


Figure 1.a. - Visualizations of 8×8 patterns

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

Figure 1.b.1 - **One Pattern stored**: Battery

We set

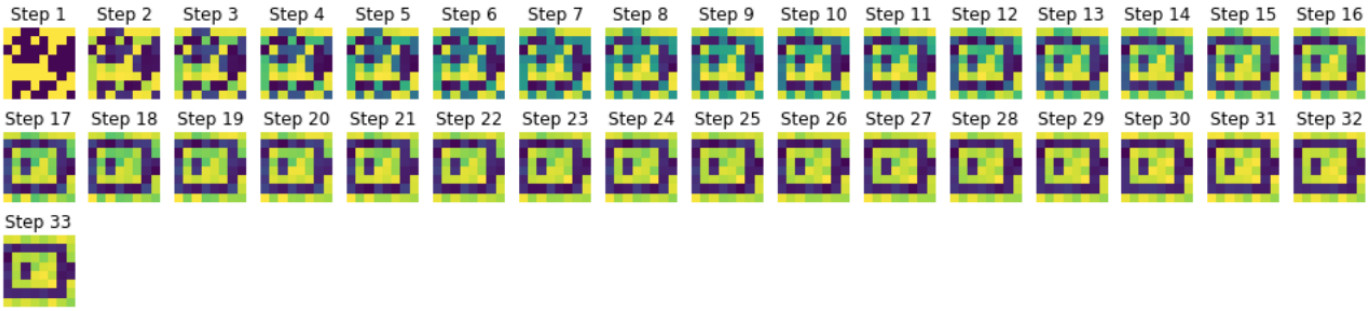
$$W \stackrel{\text{def}}{=} \frac{1}{N} \mathbf{p}_{\text{bat}} \mathbf{p}_{\text{bat}}^{\intercal}$$

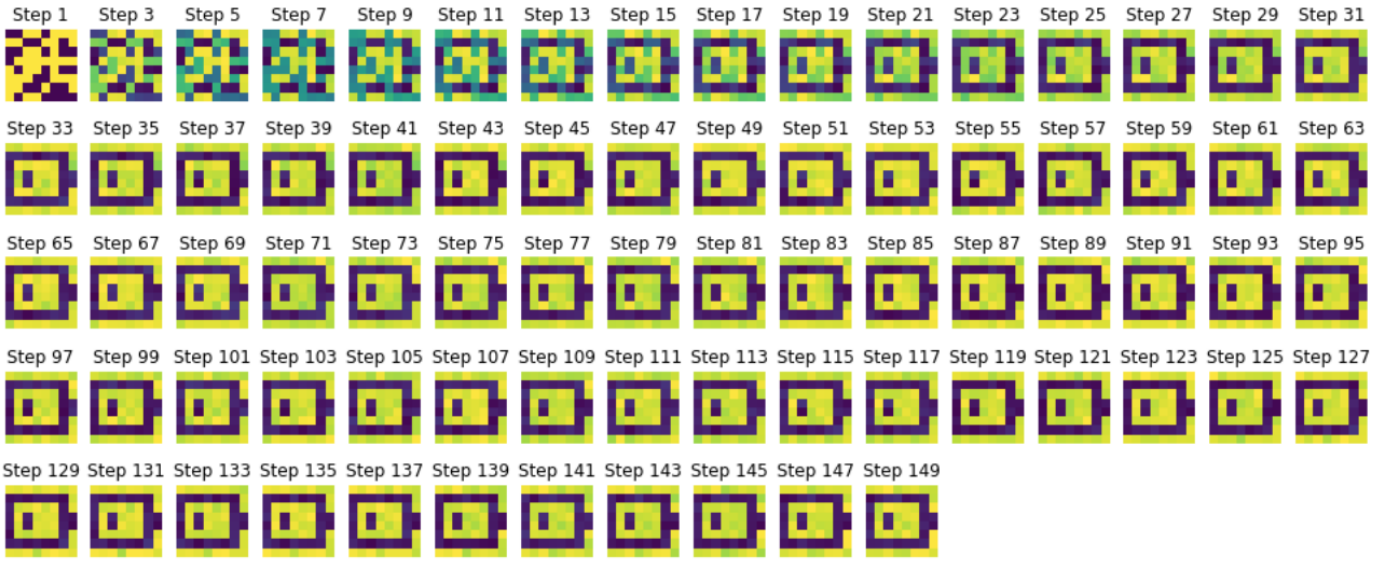
Then, we simulate the network with resort to the Euler method (with time-increment $dt = \Delta t \stackrel{\text{def}}{=} 0.1$ (arbitrary units)):

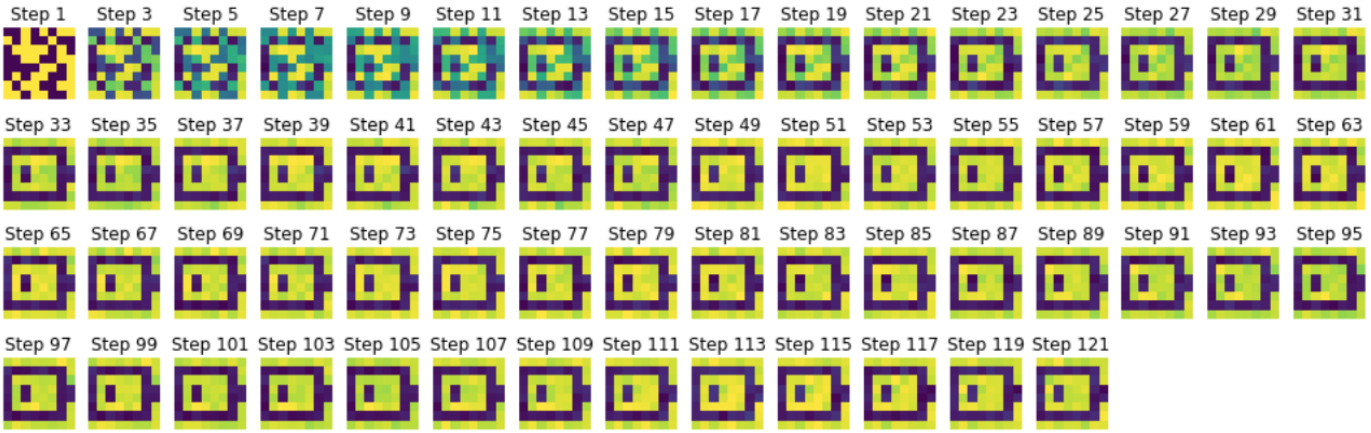
$$\mathbf{x}(t + \Delta t) = (1 - \Delta t)\, \mathbf{x}(t) + \Delta t \, \text{sign}(W \mathbf{x}(t)) + \sqrt{\Delta t}\, \sigma \eta(t)$$

which yields, for random starting patterns:

Network simulations







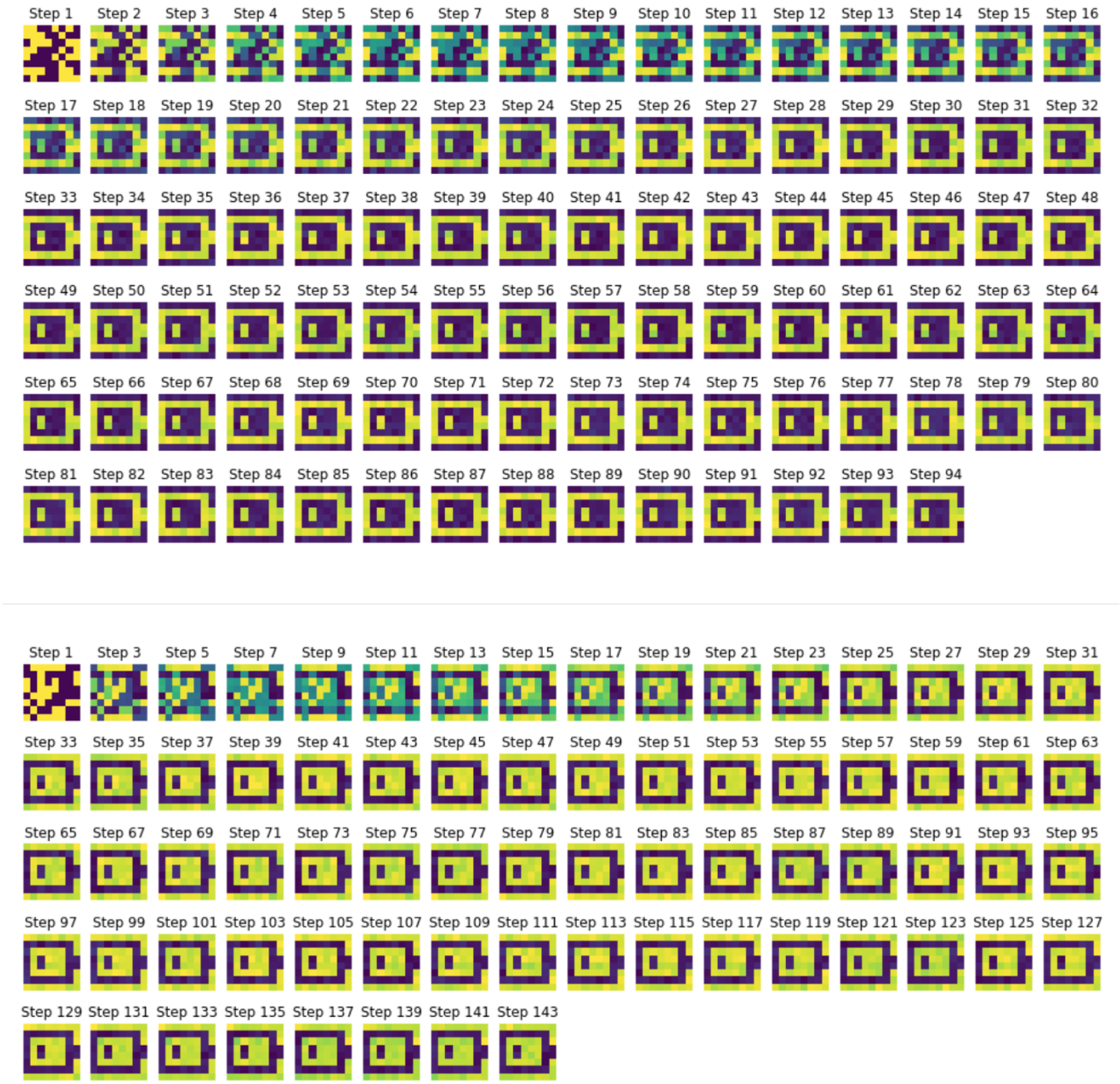


Figure 1.b.2 - **One Pattern (battery) stored**: Network simulation from different starting patterns

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

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

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

$$\mathbf{x} \longmapsto \text{sign}(W\mathbf{x})$$

Indeed:

$$
\begin{aligned}
\operatorname{sign}\left(W\mathbf{p}_{\text{bat}}\right) &= \operatorname{sign}\left(\frac{1}{N}\left(\mathbf{p}_{\text{bat}}\mathbf{p}_{\text{bat}}{}^{\intercal}\right)\mathbf{p}_{\text{bat}}\right) \\
&= \operatorname{sign}\left(\frac{1}{N}\mathbf{p}_{\text{bat}}\left(\mathbf{p}_{\text{bat}}{}^{\intercal}\mathbf{p}_{\text{bat}}\right)\right) && \text{(because the matrix product is associative)} \\
&= \operatorname{sign}\left(\frac{1}{N}\mathbf{p}_{\text{bat}}\overbrace{\left(\sum_{i=1}^{N}\underbrace{\left[\mathbf{p}_{\text{bat}}\right]_i^2}_{=\,1}\right)}^{=\,N}\right) \\
&= \operatorname{sign}\left(\mathbf{p}_{\text{bat}}\right) \\
&= \mathbf{p}_{\text{bat}} && \begin{array}{l}\text{(because } \mathbf{p}_{\text{bat}} \in \mathfrak{M}_{N,1}(\{-1,1\}) \\ \text{so } \operatorname{sign}\left(\left[\mathbf{p}_{\text{bat}}\right]_i\right) = \left[\mathbf{p}_{\text{bat}}\right]_i \quad \forall 1 \leq i \leq N \text{)}\end{array}
\end{aligned}
$$

And:

$$
\begin{aligned}
\operatorname{sign}\left(W(-\mathbf{p}_{\text{bat}})\right) &= \operatorname{sign}\left(-W\mathbf{p}_{\text{bat}}\right) \\
&= -\operatorname{sign}\left(W\mathbf{p}_{\text{bat}}\right) && \text{(because sign is an odd function)} \\
&= -\mathbf{p}_{\text{bat}}
\end{aligned}
$$

Therefore, leaving the noise aside, by setting:

$$
\varphi \overset{\text{def}}{=} \mathbf{x} \longmapsto -\mathbf{x} + \operatorname{sign}(W\mathbf{x})
$$

so that the corresponding dynamics **without noise** is given by:

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

it follows that:

$$
\varphi(\mathbf{p}_{\text{bat}}) = \varphi(-\mathbf{p}_{\text{bat}}) = \mathbf{0}
$$

> *As a result: $\mathbf{p}_{\text{bat}}$ and $-\mathbf{p}_{\text{bat}}$ 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 $\mathbf{p}_{\text{com}}$:

Storing 2 patterns:

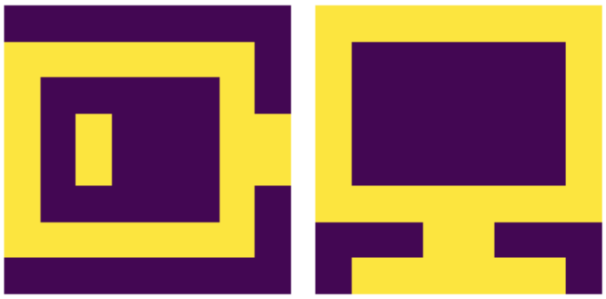


Figure 2.c.1 - **Two Patterns stored**: Battery & Computer

so that

$$W = \frac{1}{N}\left(\mathbf{p}_{\text{bat}}\mathbf{p}_{\text{bat}}^{\intercal} + \mathbf{p}_{\text{com}}\mathbf{p}_{\text{com}}^{\intercal}\right)$$

And we simulate the network on random starting patterns:

Network simulations

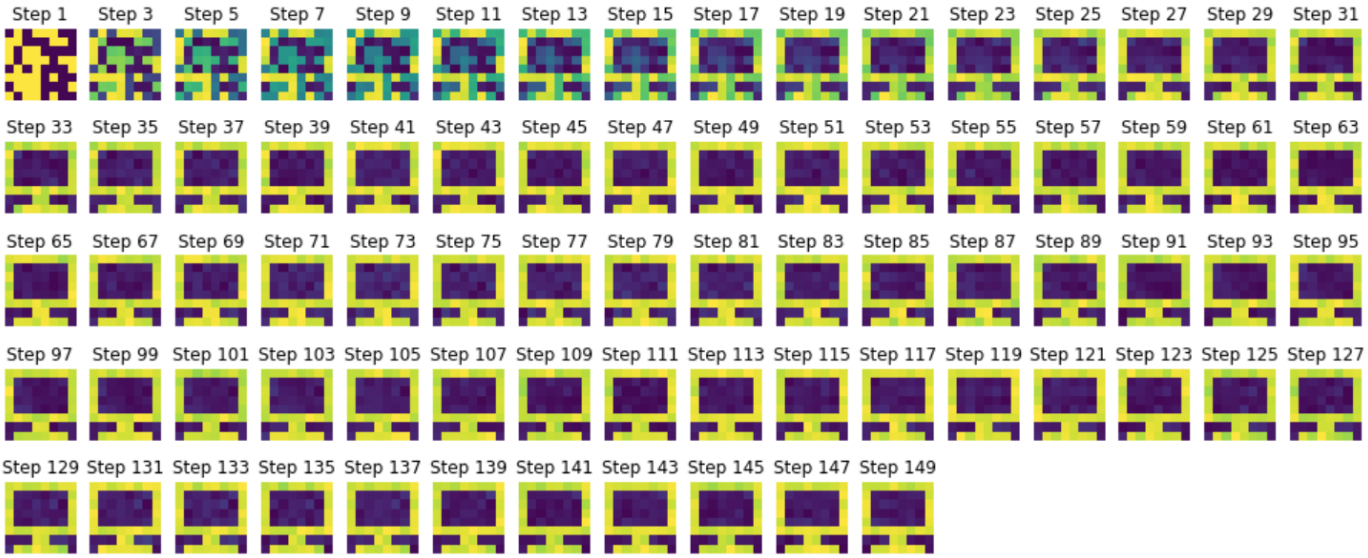


Figure 2.c.2 - **Two Patterns (battery & computer) stored**: Network simulation from different starting patterns

Again, we can show that:

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

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

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

$$\begin{aligned}\operatorname{sign}(W\mathbf{p}) &= \operatorname{sign}\left(\frac{1}{N}\left(\mathbf{p}\mathbf{p}^\intercal + \mathbf{q}\mathbf{q}^\intercal\right)\mathbf{p}\right)\\ &= \operatorname{sign}\Big(\underbrace{\frac{1}{N}\left(\mathbf{p}\mathbf{p}^\intercal\right)\mathbf{p}}_{=\,\mathbf{p}\ \text{(cf. previously)}} + \frac{1}{N}\left(\mathbf{q}\mathbf{q}^\intercal\right)\mathbf{p}\Big)\\ &= \operatorname{sign}\Big(\mathbf{p} + \frac{1}{N}\,\mathbf{q}\underbrace{\left(\mathbf{q}^\intercal\mathbf{p}\right)}_{=\,\langle\mathbf{q},\,\mathbf{p}\rangle}\Big)\end{aligned}$$

But:

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

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

$$\begin{aligned}
& \langle\mathbf{q},\,\mathbf{p}\rangle = \varepsilon N \\
\iff\quad & \sum_{i=1}^{N}[\mathbf{q}]_i[\mathbf{p}]_i = \varepsilon N \\
\iff\quad & \sum_{i=1}^{N}\underbrace{\left([\mathbf{q}]_i[\mathbf{p}]_i - \varepsilon\right)}_{\leq 0 \text{ if } \varepsilon=1 \text{ else } \geq 0} = 0 \\
\iff\quad & \forall 1 \leq i \leq N, \quad [\mathbf{q}]_i[\mathbf{p}]_i - \varepsilon = 0 \\
\iff\quad & \forall 1 \leq i \leq N, \quad [\mathbf{q}]_i[\mathbf{p}]_i = \varepsilon \\
\iff\quad & \forall 1 \leq i \leq N, \quad [\mathbf{q}]_i = \varepsilon/[\mathbf{p}]_i = \varepsilon[\mathbf{p}]_i \qquad (\text{because } [\mathbf{p}]_i \in \{-1, 1\ )\}
\end{aligned}$$

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

$$|\langle\mathbf{q},\,\mathbf{p}\rangle| \leq \underbrace{\|\mathbf{q}\|_2\,\|\mathbf{p}\|_2}_{=\sqrt{N}\sqrt{N}=N} \quad \text{and} \quad \Big(|\langle\mathbf{q},\,\mathbf{p}\rangle| = N \iff \mathbf{q} = \pm\mathbf{p}\Big)$$

Here, as:

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

it comes that:

$$|\langle\mathbf{q},\,\mathbf{p}\rangle| < N$$

Hence:

$$
\begin{aligned}
\operatorname{sign}(W\mathbf{p}) &= \operatorname{sign}\Big(\mathbf{p} + \underbrace{\overbrace{\langle \mathbf{q},\, \mathbf{p} \rangle}^{\in [-(N-1)\,,\, N-1]}}_{\in\,]-1,1[}\, \mathbf{q}\Big) \\
&\overset{\circledast}{=} \operatorname{sign}(\mathbf{p}) && \left(\text{because each } \frac{\langle \mathbf{q},\, \mathbf{p} \rangle}{N}\, [\mathbf{q}]_i \in \left[-1 + \frac{1}{N},\, 1 - \frac{1}{N}\right]\right. \\
& && \left.\text{: it can't change the sign of } [\mathbf{p}]_i \in \{-1, 1\}\ \right) \\
&= \mathbf{p}
\end{aligned}
$$

And likewise, by symmetry:

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

Once we know that $\mathbf{p}$ and $\mathbf{q}$ 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 $\mathbf{q}$ being different from $\pm\mathbf{p}$ (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 third-to-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:

$$W \overset{\text{def}}{=} \frac{1}{N} \sum_{i=1}^{M} \mathbf{p}_i \mathbf{p}_i^\intercal$$

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

3 patterns: linear combinations stored

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

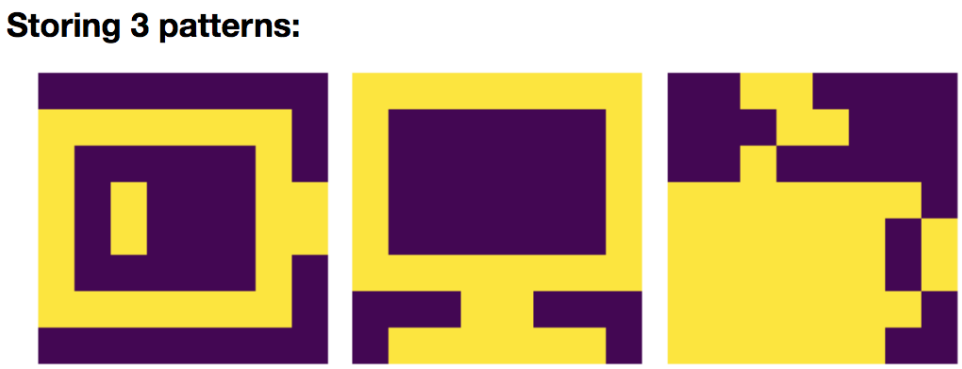


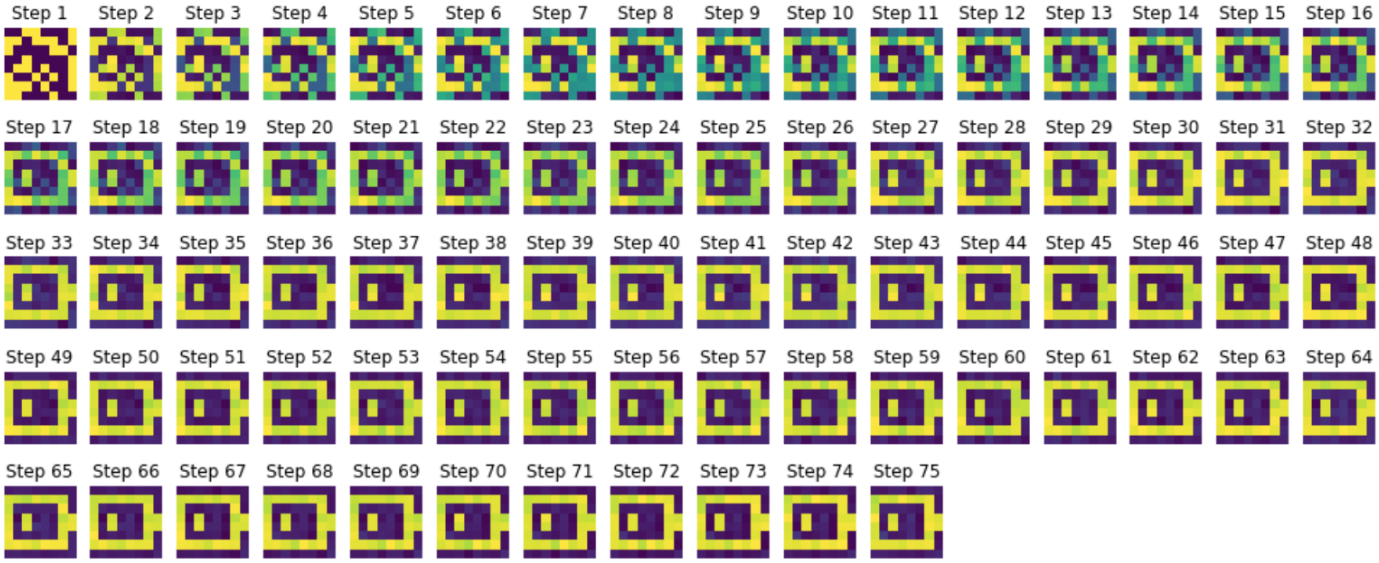
Figure 3.d.1 - **Three Patterns stored**: Battery, Computer & Cup of coffee

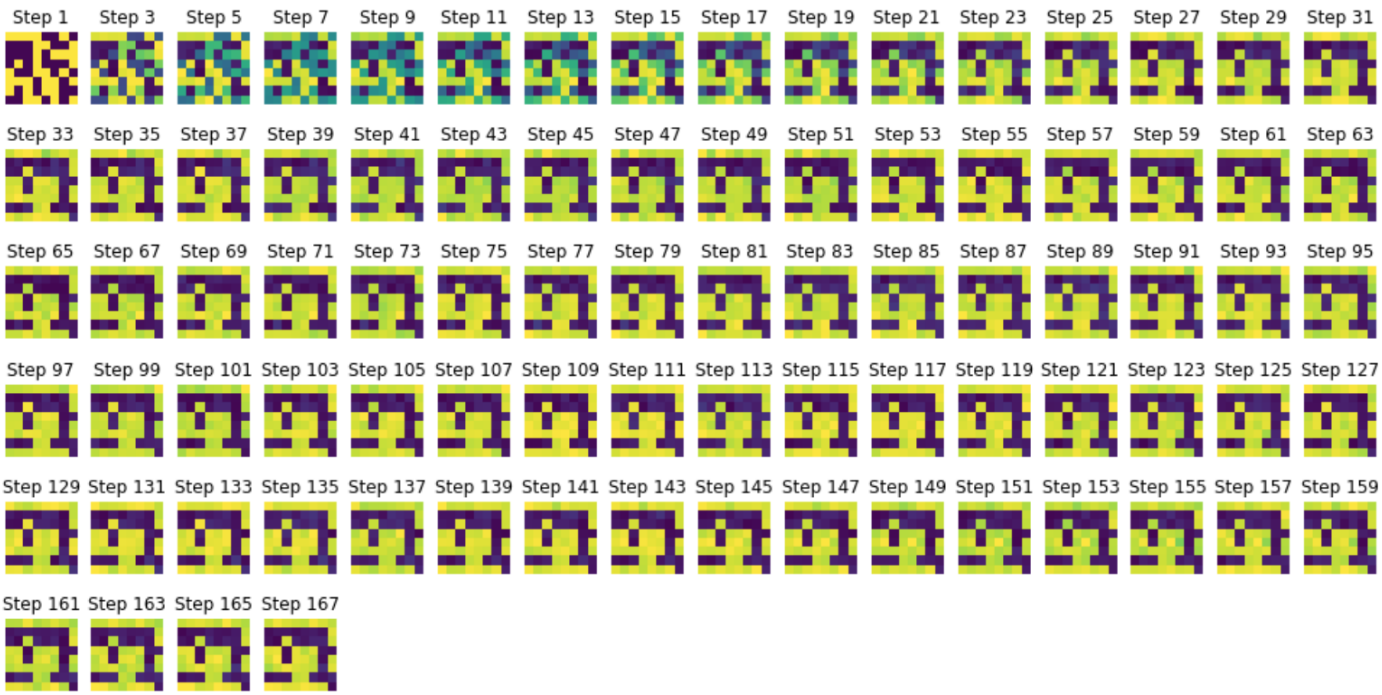
so that, in compliance with the general rule:

$$W = \frac{1}{N}\left(\mathbf{p}_{\text{bat}}\mathbf{p}_{\text{bat}}^{\intercal} + \mathbf{p}_{\text{com}}\mathbf{p}_{\text{com}}^{\intercal} + \mathbf{p}_{\text{cof}}\mathbf{p}_{\text{cof}}^{\intercal}\right)$$

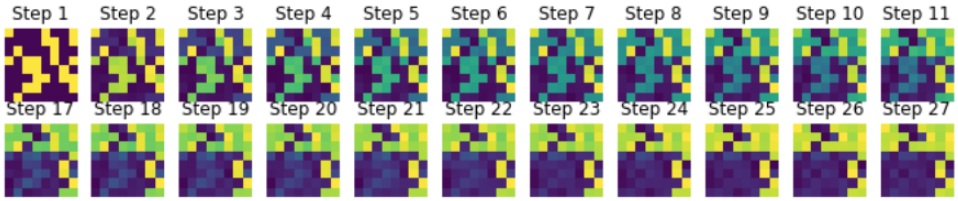
Then, we simulate the network with random initial conditions:

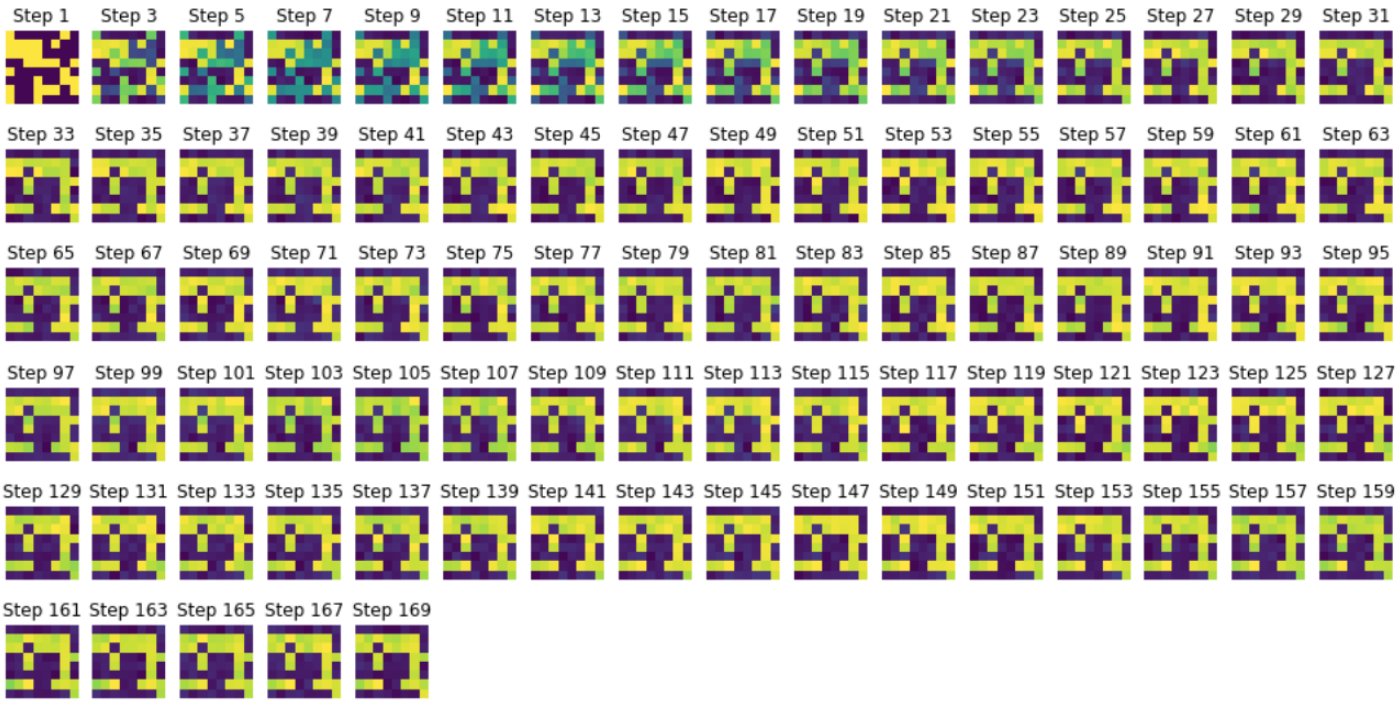
Network simulations






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 151 Step 153 Step 155


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
Step 17 Step 18 Step 19 Step 20 Step 21 Step 22 Step 23 Step 24 Step 25 Step 26 Step 27


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 151 Step 153 Step 155 Step 157 Step 159
Step 161 Step 163 Step 165 Step 167 Step 169

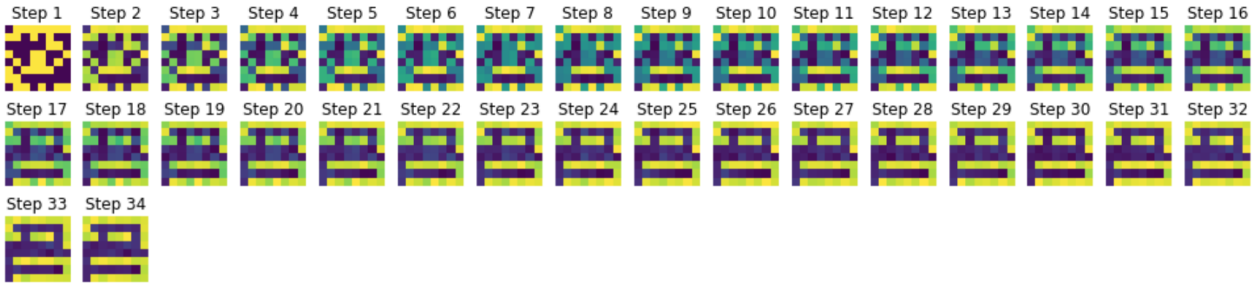


Figure 3.d.2 - **Three Patterns (battery, computer & coffee) stored**: Network simulation from different starting patterns

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

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

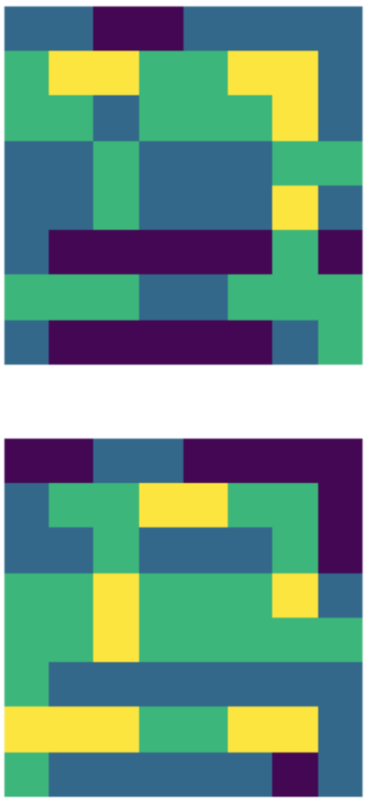


Figure 3.d.3 - **Three Patterns (battery, computer & coffee) stored**: Linear combinations $\mathbf{p}_{\text{bat}} - \mathbf{p}_{\text{com}} - \mathbf{p}_{\text{cof}}$ and $\mathbf{p}_{\text{bat}} - \mathbf{p}_{\text{com}} + \mathbf{p}_{\text{cof}}$ (respectively) that are also stored by the network

> *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

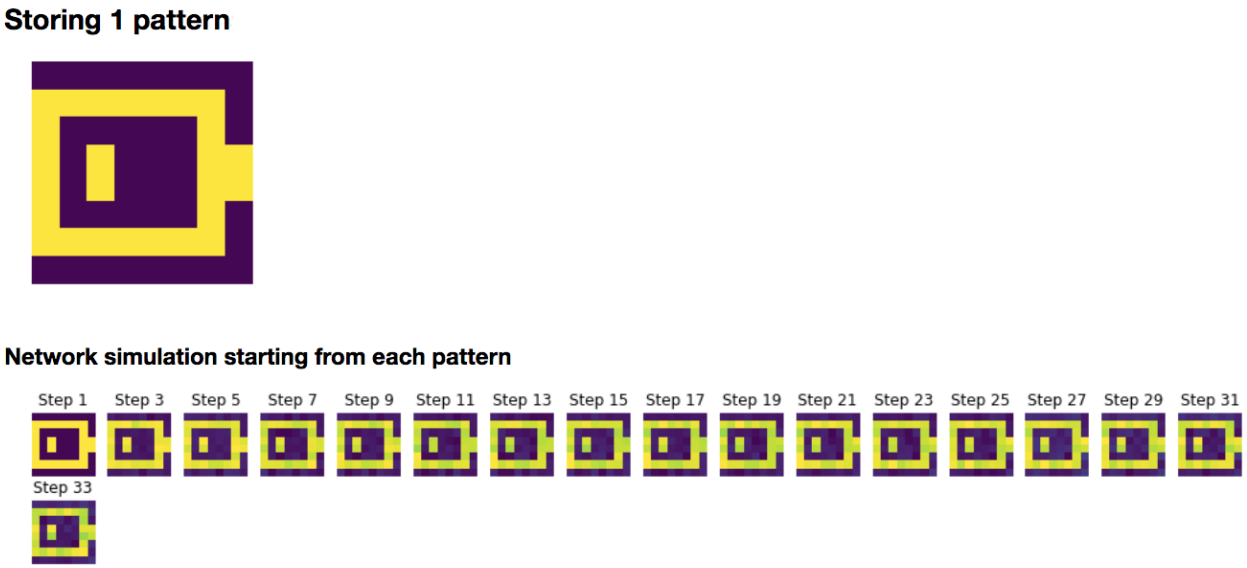


Figure 4.d.1 - **Weight matrix constructed out of one pattern (battery)**: network simulation from this pattern (to check that it is stored)

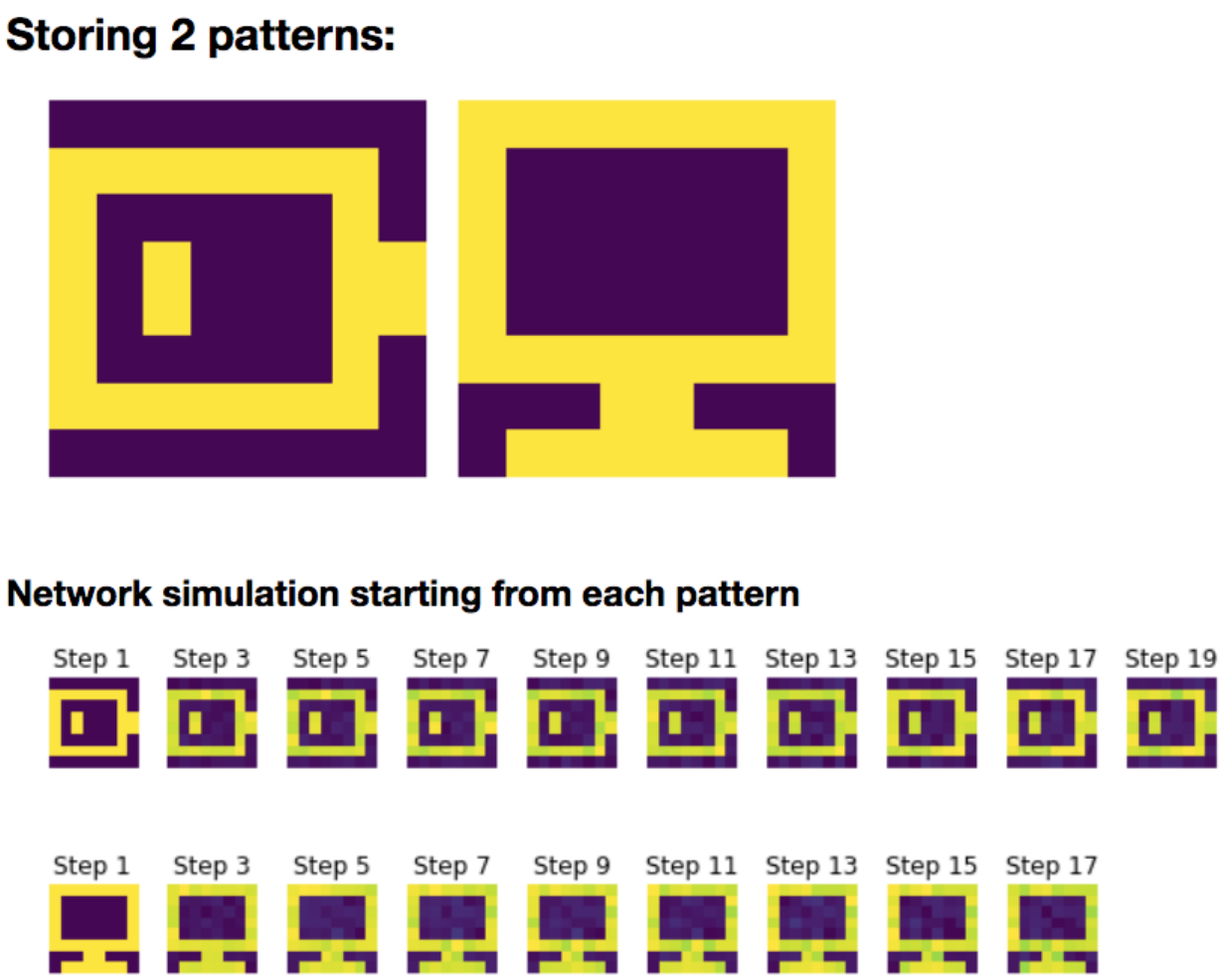


Figure 4.d.2. - **Weight matrix constructed out of two pattern (battery & computer)**: network simulation from these patterns (to check that they are stored)

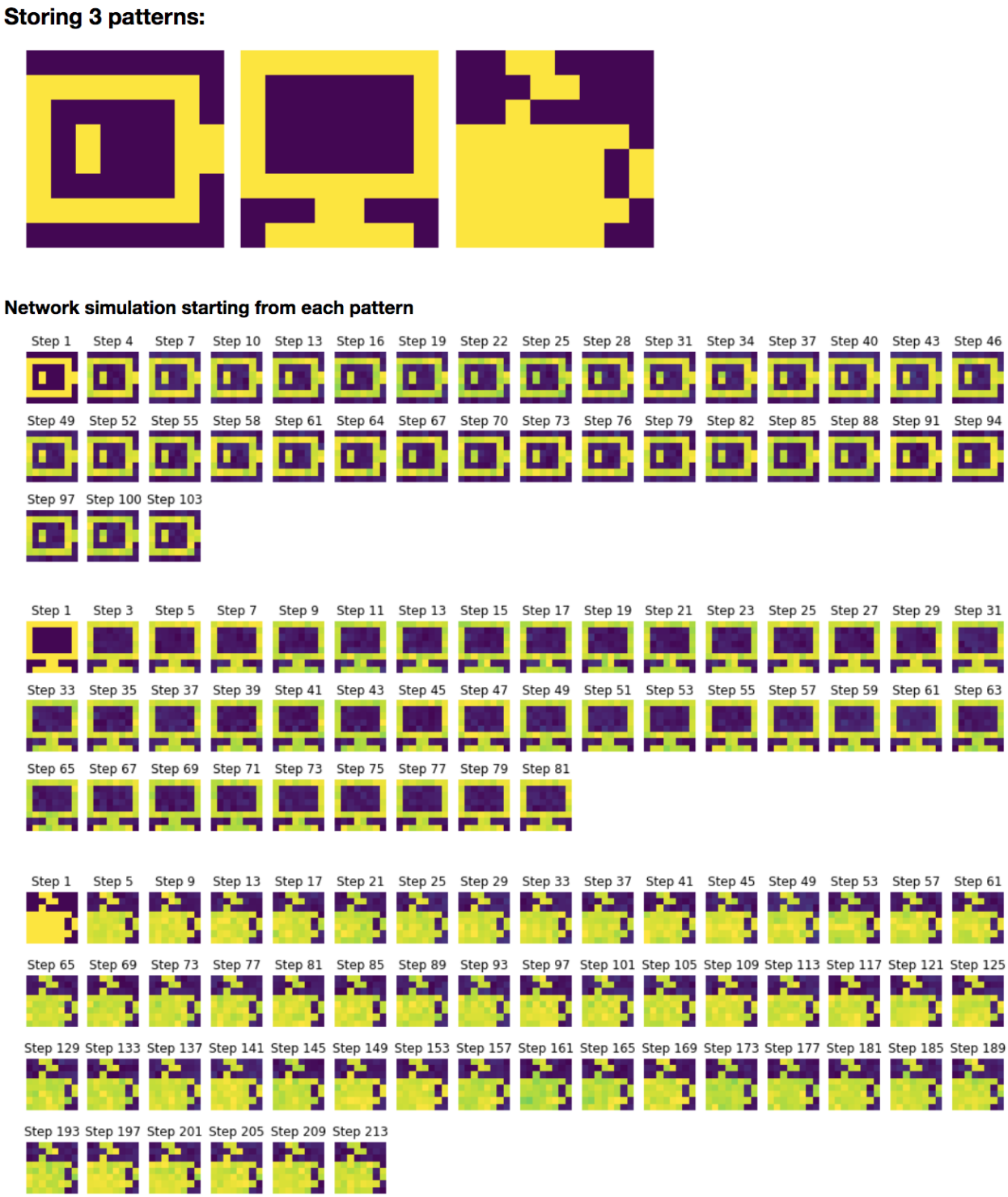


Figure 4.d.3 - **Weight matrix constructed out of three patterns (battery, computer & coffee)**: network simulation from these patterns (to check that they are stored)

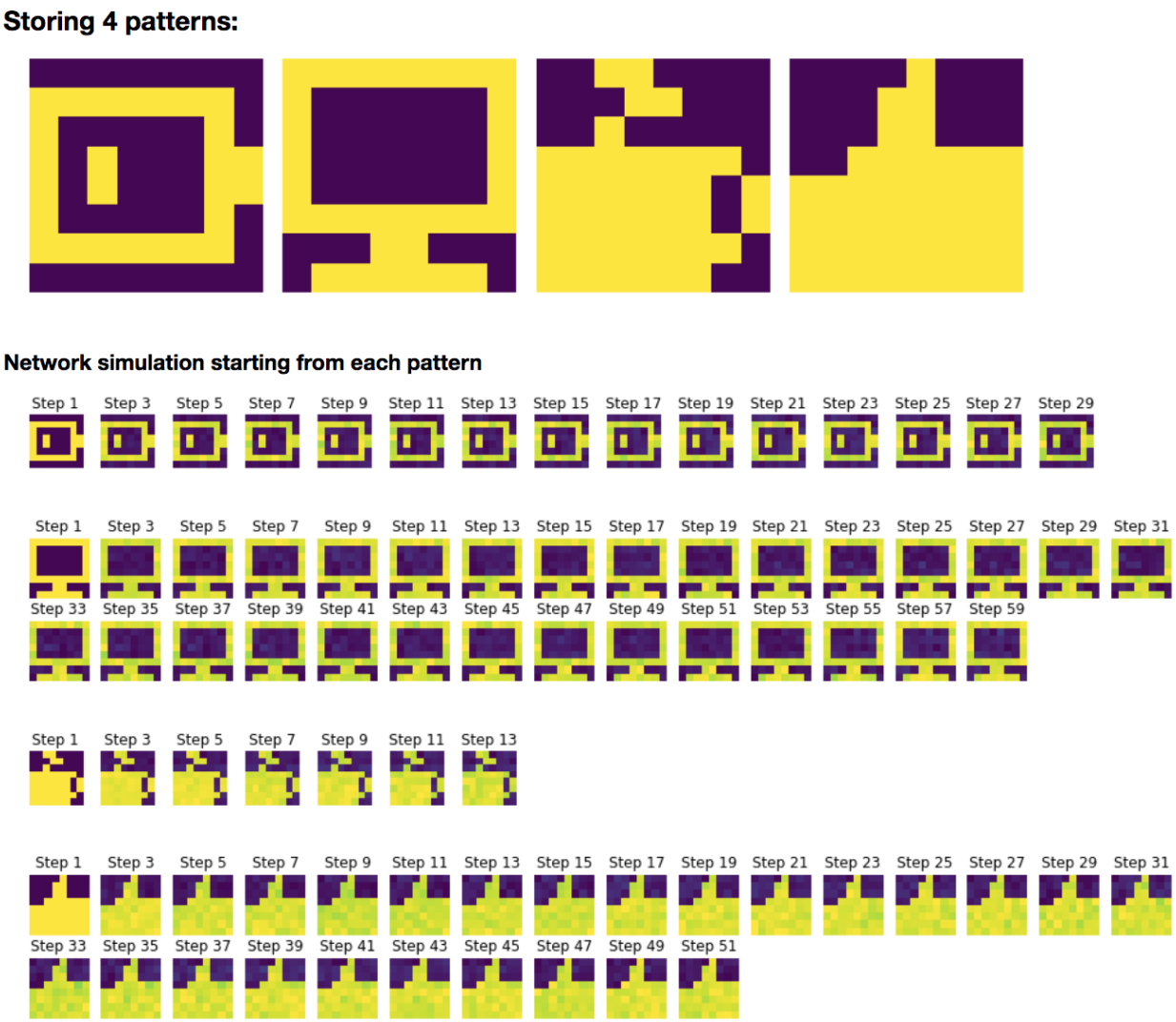


Figure 4.d.4 - **Weight matrix constructed out of four patterns (battery, computer, coffee & thumbs up)**: network simulation from these patterns (to check that they are stored)

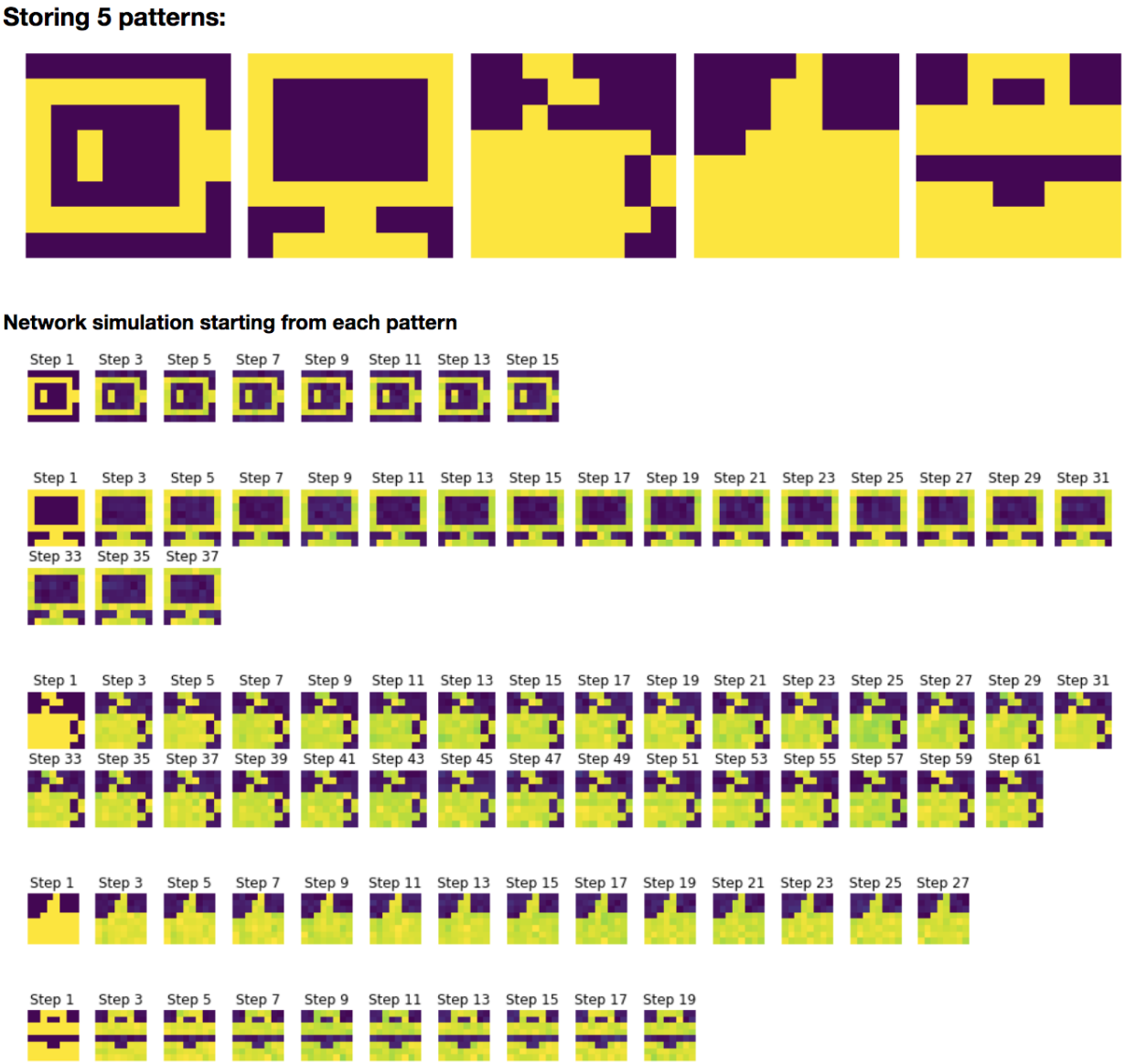


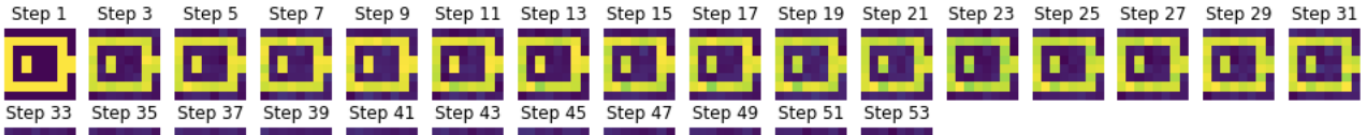
Figure 4.d.5 - **Weight matrix constructed out of five patterns (battery, computer, coffee, thumbs up & toolbox)**: network simulation from these patterns (to check that they are stored)

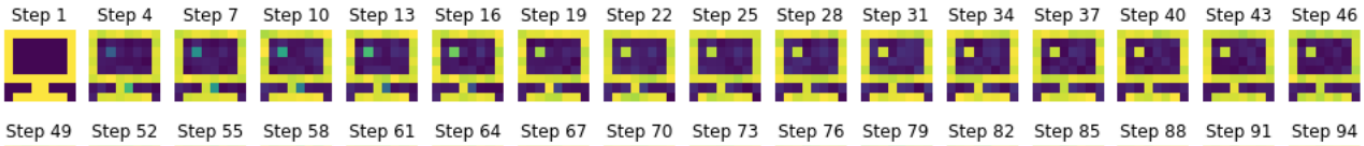
Figure 4.d.6 - **Weight matrix constructed out of six patterns**: network simulation from these patterns (to check that they are stored)

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





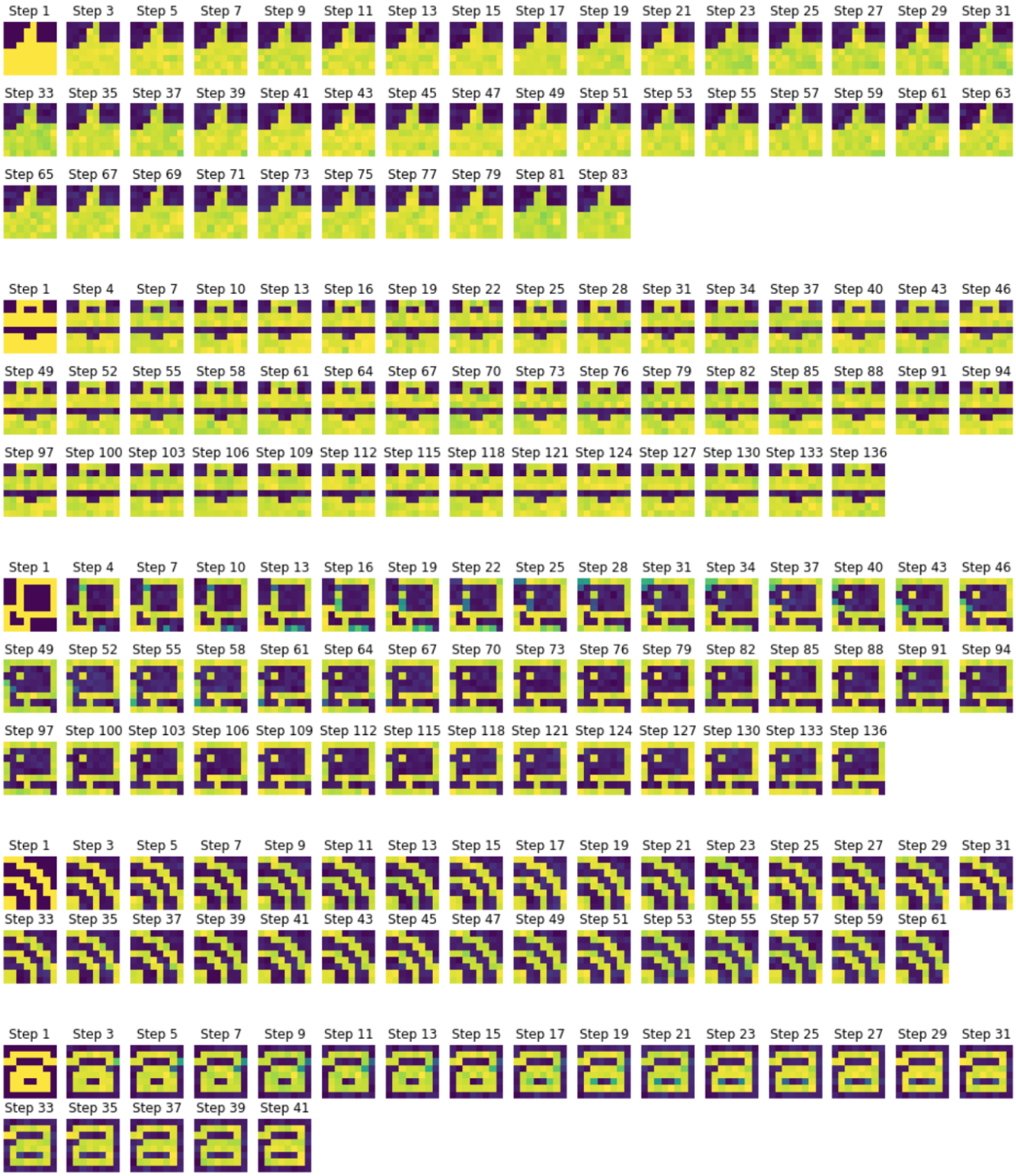
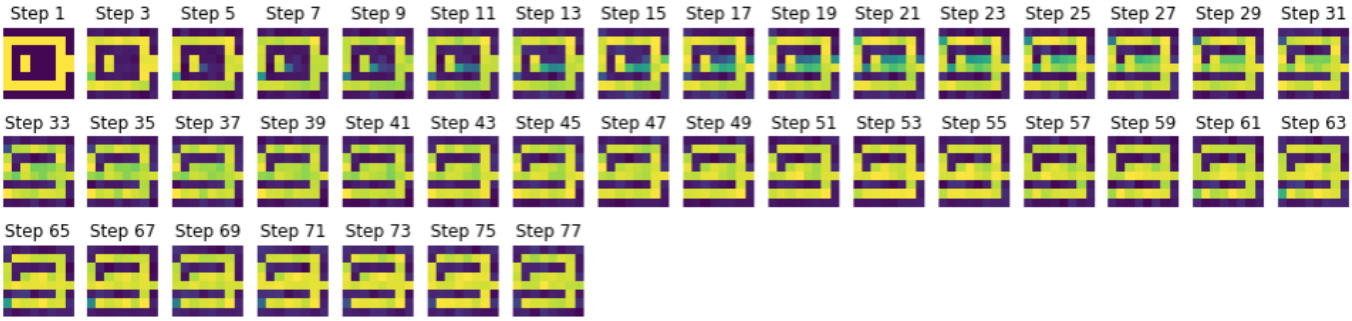
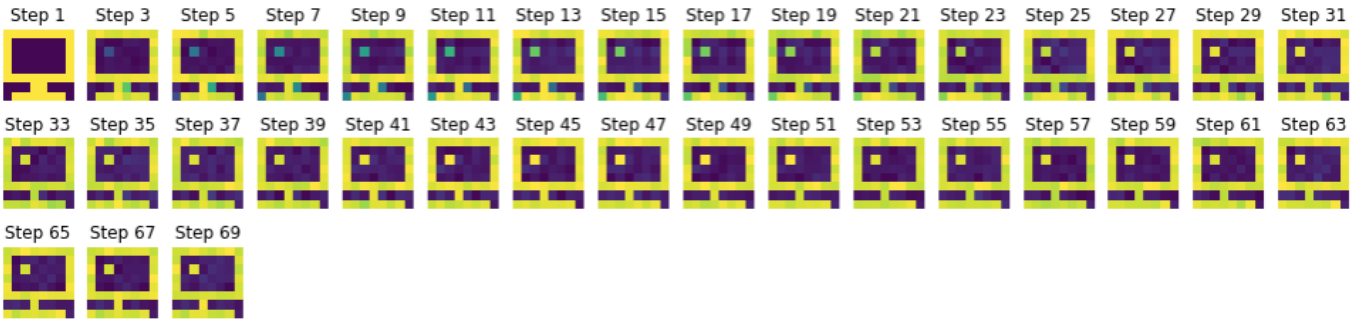


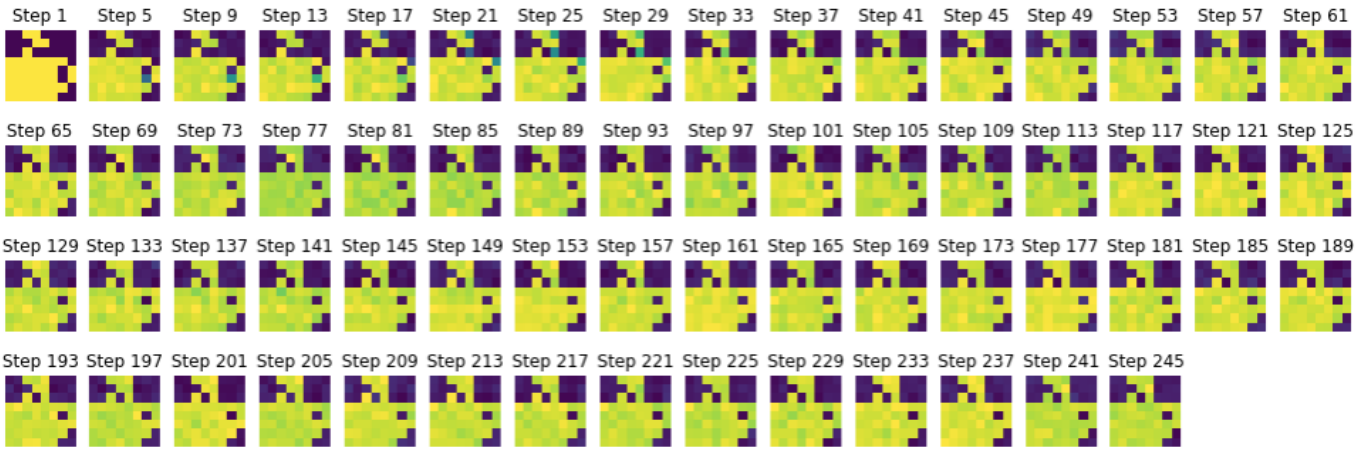
Figure 4.d.8 - **Weight matrix constructed out of eight patterns**: network simulation from these patterns: not all of them are stored!

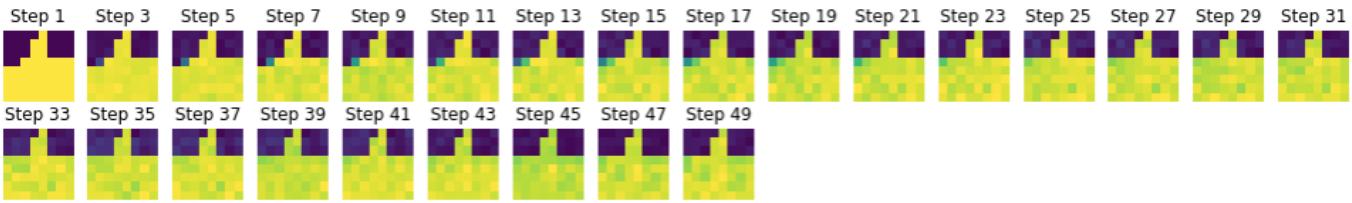
Storing 9 patterns:

Network simulation starting from each pattern









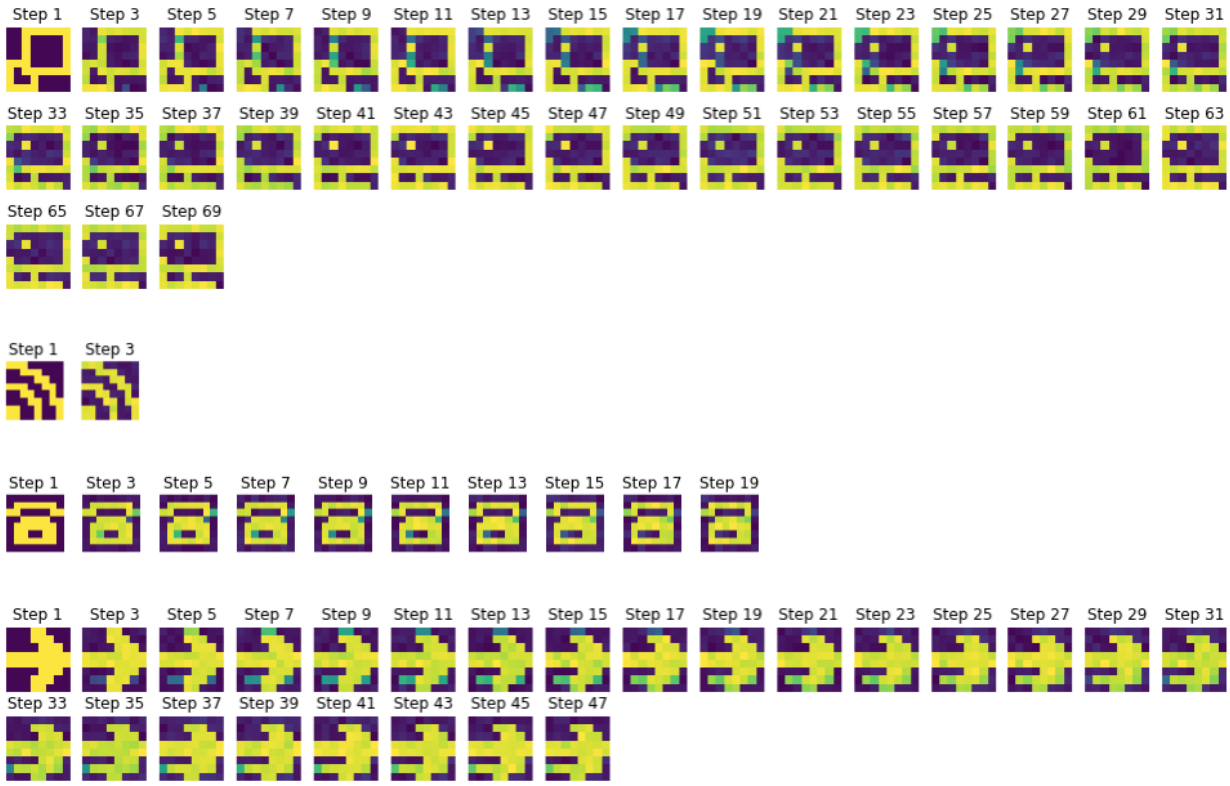


Figure 4.d.9 - **Weight matrix constructed out of nine patterns**: network simulation from these patterns to check that not all of them are stored

---

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).*