Exercise Sheet 5: Perceptrons and Hopfield networks

Younesse Kaddar

- PDF Version / Online Version
- Associated Jupyter Notebook for the code

1. Peceptrons

Consider the following two sets of two-dimensional patterns:

$$\mathscr{D}_{\text{blue}} = \{(1,2),(1,3),(3,4),(2,4)\}$$
$$\mathscr{D}_{\text{red}} = \{(4,0),(6,2),(2,-3),(6,-2)\}$$

a. Are the two sets linearly separable? If they are, use a preceptron rule to find a readout weight vector that allows a binary neuron to classify the two sets in two different categories

The two sets are easily seen to be separable, by the line whose equation is $y = x - 1.5$ for example, as shown in the figure below:

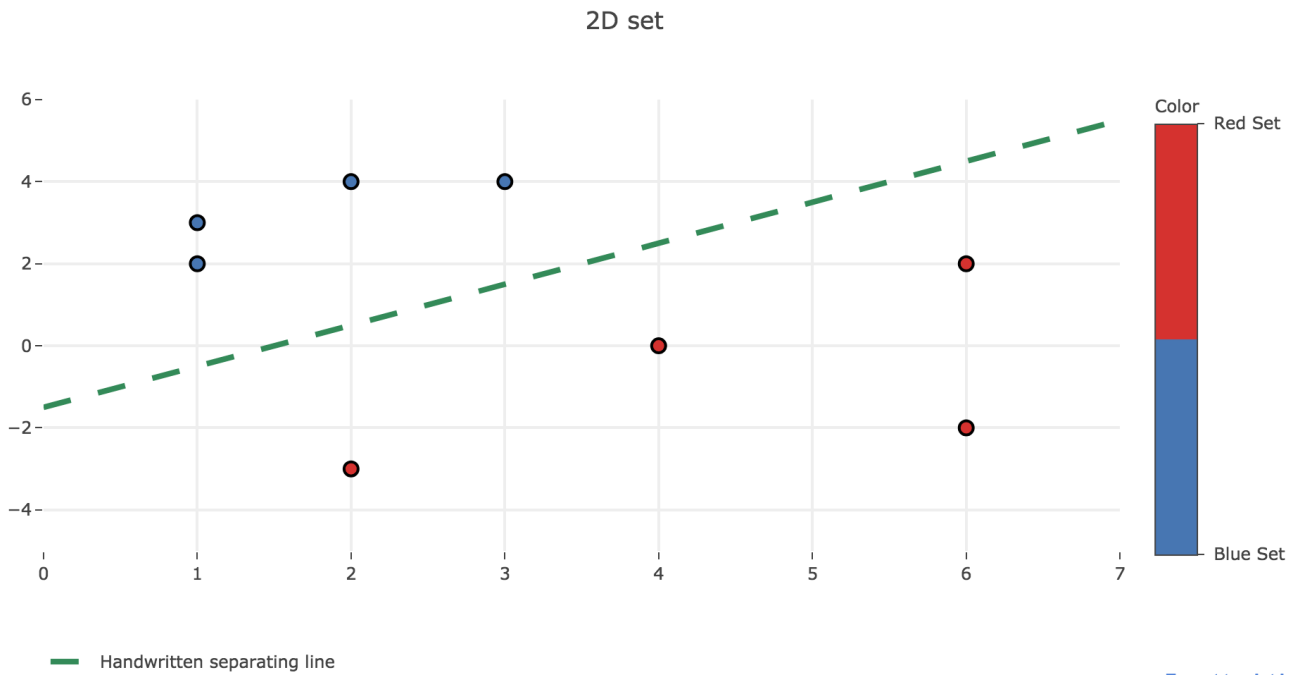

Figure 1.a.1. - Plot of the 2D-data points, along with a handwritten separating line of equation $y = x - 1.5$

The algorithm using the perceptron rule is the following one (in pseudo-code):

```
n_trials = 8

# Color of the points: 0 ≡ blue / 1 ≡ red
Y → [0, 0, 0, 0, 1, 1, 1, 1]

# Data points
X → [[1,2],[1,3],[3,4],[2,4],[4,0],[6,2],[2,−3],[6,−2]]
```

```
# Initialize the weight vector
w → (1, 1)

while True:
    # is the weight still changing? To end the loop
    still_changing → False

    for 1 ≤ k ≤ nb_trials:
        # output = 1 if the scalar product is > 0, else 0
        output → H(⟨w, X[k]⟩)

        w_new → w + (Y[k]−output)∗X[k]

        if w_new != w:
            still_changing = True

        w → w_new

    if not still_changing:
        break

return w
```

Let us run it on our 2D-data set:

STEP	w	k	$x^{(k)}$	$y^{(k)}$	$\langle w, x^{(k)} \rangle$	OUTPUT	y_k - OUTPUT	w_{new}
	$(1,1)$							
1	$(1,1)$	1	$(1,2)$	0	$1 \cdot 1 + 1 \cdot 2 = 3$	1	-1	$(1,1) - (1,2) = (0,-1)$
1	$(0,-1)$	2	$(1,3)$	0	$0 \cdot 1 - 1 \cdot 3 = -3$	0	0	$(0,-1)$
1	$(0,-1)$	3	$(3,4)$	0	$0 \cdot 3 - 1 \cdot 4 = -4$	0	0	$(0,-1)$
1	$(0,-1)$	4	$(2,4)$	0	$0 \cdot 2 - 1 \cdot 4 = -4$	0	0	$(0,-1)$
1	$(0,-1)$	5	$(4,0)$	1	$0 \cdot 4 - 1 \cdot 0 = 0$	1	0	$(0,-1)$
1	$(0,-1)$	6	$(6,2)$	1	$0 \cdot 6 - 1 \cdot 2 = -2$	0	1	$(0,-1) + (6,2) = (6,1)$
1	$(6,1)$	7	$(2,-3)$	1	$6 \cdot 2 - 1 \cdot 3 = 9$	1	0	$(6,1)$
1	$(6,1)$	8	$(6,-2)$	1	$6 \cdot 6 - 1 \cdot 2 = 34$	1	0	$(6,1)$
2	$(6,1)$	9	$(1,2)$	0	$6 \cdot 1 + 1 \cdot 2 = 8$	1	-1	$(5,-1)$
2	$(5,-1)$	10	$(1,3)$	0	$5 \cdot 1 - 1 \cdot 3 = 2$	1	-1	$(5,-1) - (1,3) = (4,-4)$
2	$(4,-4)$	11	$(3,4)$	0	$4 \cdot 3 - 4 \cdot 4 = -4$	0	0	$(4,-4)$
2	$(4,-4)$	12	$(2,4)$	0	$4 \cdot 2 - 4 \cdot 4 = -8$	0	0	$(4,-4)$
2	$(4,-4)$	13	$(4,0)$	1	$4 \cdot 4 - 4 \cdot 0 = 16$	1	0	$(4,-4)$
2	$(4,-4)$	14	$(6,2)$	1	$4 \cdot 6 - 4 \cdot 2 = 16$	1	0	$(4,-4)$
2	$(4,-4)$	15	$(2,-3)$	1	$4 \cdot 2 + 4 \cdot 3 = 20$	1	0	$(4,-4)$
2	$(4,-4)$	16	$(6,-2)$	1	$4 \cdot 6 + 4 \cdot 2 = 32$	1	0	$(4,-4)$

STEP						OUTPUT	- OUTPUT	
3	$(4,-4)$	17	$(1,2)$	0	$4\cdot 1-4\cdot 2$ $=-4$	0	0	$(4,-4)$
3	$(4,-4)$	18	$(1,3)$	0	$4\cdot 1-4\cdot 3$ $=-8$	0	0	$(4,-4)$
3	$(4,-4)$	19	$(3,4)$	0	$4\cdot 3-4\cdot 4$ $=-4$	0	0	$(4,-4)$
3	$(4,-4)$	20	$(2,4)$	0	$4\cdot 2-4\cdot 4$ $=-8$	0	0	$(4,-4)$
3	$(4,-4)$	21	$(4,0)$	1	$4\cdot 4-4\cdot 0$ $=16$	1	0	$(4,-4)$
3	$(4,-4)$	22	$(6,2)$	1	$4\cdot 6-4\cdot 2$ $=16$	1	0	$(4,-4)$
3	$(4,-4)$	23	$(2,-3)$	1	$4\cdot 2+4\cdot 3$ $=20$	1	0	$(4,-4)$
3	$(4,-4)$	24	$(6,-2)$	1	$4\cdot 6+4\cdot 2$ $=32$	1	0	$(4,-4)$

The algorithm stops at the end of the third step because the readout weight vector $w \stackrel{\text{def}}{=} (4,-4)$ hasn't changed throughout this step: all the points have finally been correctly classified.

Therefore: the decision boundary turns out to the line given by the equation:

$$\underbrace{w_0}_{\stackrel{\text{def}}{=}4}\, x + \underbrace{w_1}_{\stackrel{\text{def}}{=}-4}\, y = 0 \iff y = x$$

which can easily be checked on the data points: a point (x,y) is classified as red (resp. blue) if and only if $x > y$ (resp. $x \leq y$).

Here is the resulting separating line (in green), along with the normalized readout weight vector w (in purple):

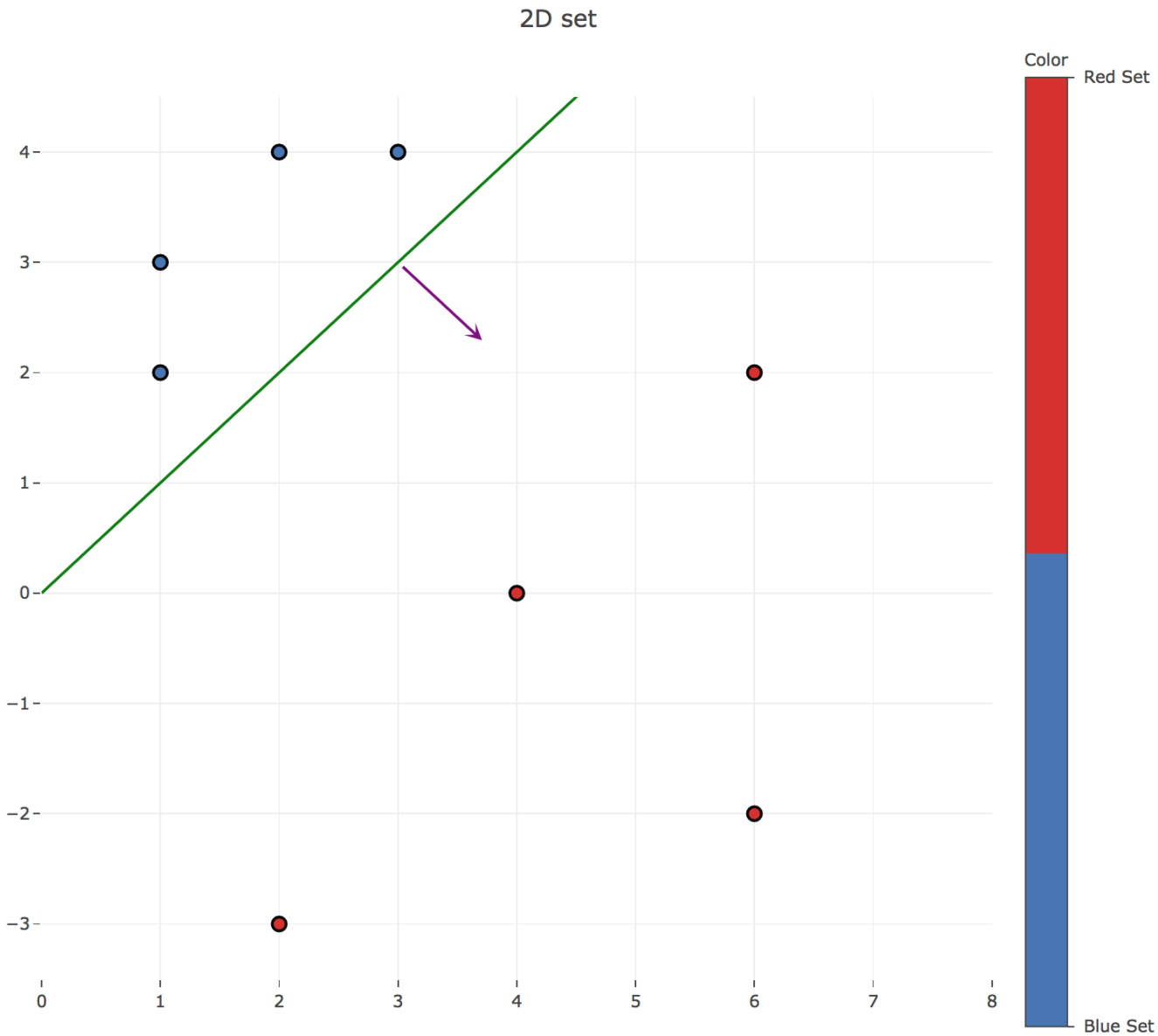

Figure 1.a.2 - Plot of the decision boundary $y = x$ (in green) for the two 2D-data sets, along with the normalized readout vector (in purple)

b. Repeat the exercise with the following two sets of three-dimensional patterns:

$$\mathscr{D}_{\text{blue}} = \{(2,1,2),(3,2,3),(4,2,4),(3,1,2)\}$$
$$\mathscr{D}_{\text{red}} = \{(1,4,-1),(2,3,0),(1,2,0),(0,1,0)\}$$

Again, the data sets are seen to be linearly separable (with the $z = 1$ plane for instance):

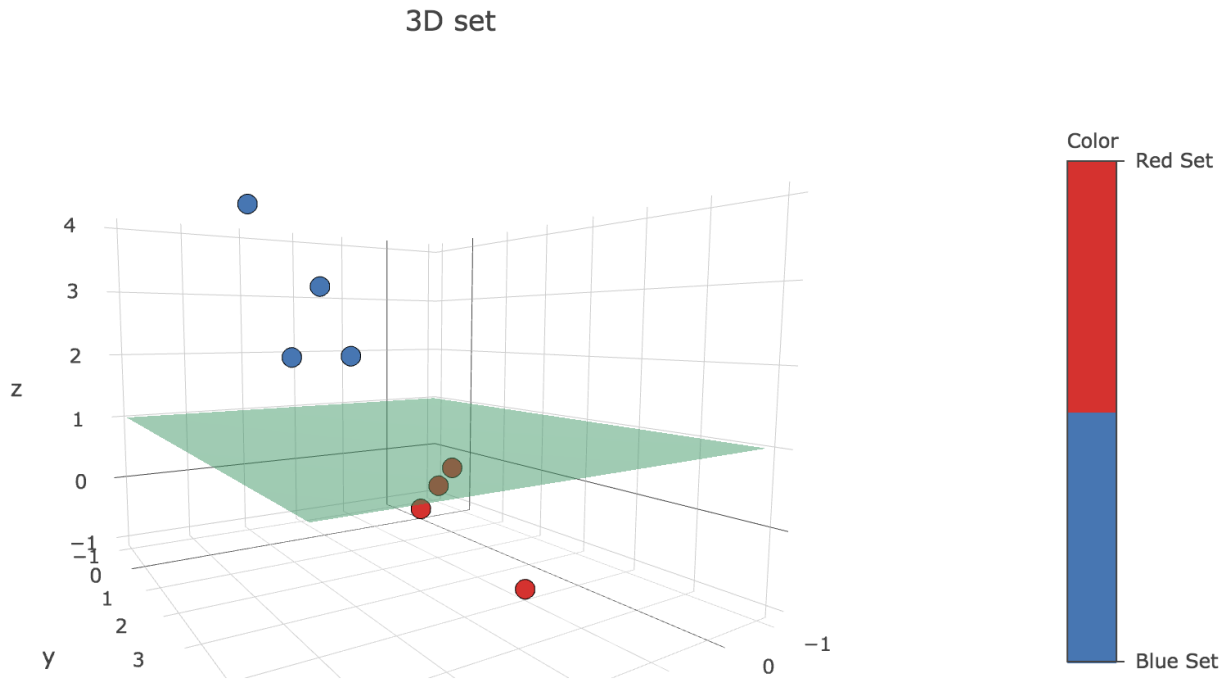

Figure 1.b.1. - Plot of the 3D-data points, along with a handwritten separating plane of equation $z = 1$

This time, the values are initialized as follows:

```
n_trials = 8

# Color of the points: 0 ≡ blue / 1 ≡ red
Y → [0, 0, 0, 0, 1, 1, 1, 1]

# Data points
X → [[2,1,2],[3,2,3],[4,2,4],[3,1,2],[1,4,-1],[2,3,0],[1,2,0],[0,1,0]]

# Initialize the weight vector
w → (1, 1, 1)
```

Let us run the algorithm on this 3D-data set:

STEP	w	k	$x^{(k)}$	$y^{(k)}$	$\langle w, x^{(k)} \rangle$	**OUTPUT**	y_k - **OUTPUT**	w_{new}
	$(1,1,1)$							
1	$(1,1,1)$	1	$(2,1,2)$	0	$1\cdot 2 + 1\cdot 1 + 1\cdot 2 = 5$	1	-1	$(1,1,1) - (2,1,2) = (-1,0,-1)$
1	$(-1,0,-1)$	2	$(3,2,3)$	0	$-1\cdot 3 + 0\cdot 2 - 1\cdot 3 = -6$	0	0	$(-1,0,-1)$
1	$(-1,0,-1)$	3	$(4,2,4)$	0	$-1\cdot 4 + 0\cdot 2 - 1\cdot 4 = -8$	0	0	$(-1,0,-1)$
1	$(-1,0,-1)$	4	$(3,1,2)$	0	$-1\cdot 3 + 0\cdot 1 - 1\cdot 2 = -5$	0	0	$(-1,0,-1)$
1	$(-1,0,-1)$	5	$(1,4,-1)$	1	$-1\cdot 1 + 0\cdot 4 + 1\cdot 1 = 0$	1	0	$(-1,0,-1)$
1	$(-1,0,-1)$	6	$(2,3,0)$	1	$-1\cdot 2 + 0\cdot 3 - 1\cdot 0 = -2$	0	1	$(-1,0,-1) + (2,3,0) = (1,3,-1)$
1	$(1,3,-1)$	7	$(1,2,0)$	1	$1\cdot 1 + 3\cdot 2 - 1\cdot 0 = 7$	1	0	$(1,3,-1)$
1	$(1,3,-1)$	8	$(0,1,0)$	1	$1\cdot 0 + 3\cdot 1 - 1\cdot 0 = 3$	1	0	$(1,3,-1)$

STEP						OUTPUT	- OUTPUT	
2	$(1,3,-1)$	9	$(2,1,2)$	0	$1\cdot 2+3\cdot 1-1\cdot 2$ $=3$	1	-1	$(1,3,-1)-(2,1,2)$ $=(-1,2,-3)$
2	$(-1,2,-3)$	10	$(3,2,3)$	0	$-1\cdot 3+2\cdot 2-3\cdot 3$ $=-8$	0	0	$(-1,2,-3)$
2	$(-1,2,-3)$	11	$(4,2,4)$	0	$-1\cdot 4+2\cdot 2-3\cdot 4$ $=-12$	0	0	$(-1,2,-3)$
2	$(-1,2,-3)$	12	$(3,1,2)$	0	$-1\cdot 3+2\cdot 1-3\cdot 2$ $=-7$	0	0	$(-1,2,-3)$
2	$(-1,2,-3)$	13	$(1,4,-1)$	1	$-1\cdot 1+2\cdot 4+3\cdot 1$ $=10$	1	0	$(-1,2,-3)$
2	$(-1,2,-3)$	14	$(2,3,0)$	1	$-1\cdot 2+2\cdot 3-3\cdot 0$ $=4$	1	0	$(-1,2,-3)$
2	$(-1,2,-3)$	15	$(1,2,0)$	1	$-1\cdot 1+2\cdot 2-3\cdot 0$ $=3$	1	0	$(-1,2,-3)$
2	$(-1,2,-3)$	16	$(0,1,0)$	1	$-1\cdot 0+2\cdot 1-3\cdot 0$ $=2$	1	0	$(-1,2,-3)$
3	$(-1,2,-3)$	17	$(2,1,2)$	0	$-1\cdot 2+2\cdot 1-3\cdot 2$ $=-6$	0	0	$(-1,2,-3)$
3	$(-1,2,-3)$	18	$(3,2,3)$	0	$-1\cdot 3+2\cdot 2-3\cdot 3$ $=-8$	0	0	$(-1,2,-3)$
3	$(-1,2,-3)$	19	$(4,2,4)$	0	$-1\cdot 4+2\cdot 2-3\cdot 4$ $=-12$	0	0	$(-1,2,-3)$
3	$(-1,2,-3)$	20	$(3,1,2)$	0	$-1\cdot 3+2\cdot 1-3\cdot 2$ $=-7$	0	0	$(-1,2,-3)$
3	$(-1,2,-3)$	21	$(1,4,-1)$	1	$-1\cdot 1+2\cdot 4+3\cdot 1$ $=10$	1	0	$(-1,2,-3)$
3	$(-1,2,-3)$	22	$(2,3,0)$	1	$-1\cdot 2+2\cdot 3-3\cdot 0$ $=4$	1	0	$(-1,2,-3)$
3	$(-1,2,-3)$	23	$(1,2,0)$	1	$-1\cdot 1+2\cdot 2-3\cdot 0$ $=3$	1	0	$(-1,2,-3)$
3	$(-1,2,-3)$	24	$(0,1,0)$	1	$-1\cdot 0+2\cdot 1-3\cdot 0$ $=2$	1	0	$(-1,2,-3)$

Again, the algorithm stops at the end of the third step because the readout weight vector $w \stackrel{\text{def}}{=} (-1,2,-3)$ hasn't changed throughout this step: all the points have been correctly classified.

The decision boundary is the plane given by the equation:

$$\underbrace{w_0}_{\stackrel{\text{def}}{=}-1}x+\underbrace{w_1}_{\stackrel{\text{def}}{=}2}y+\underbrace{w_2}_{\stackrel{\text{def}}{=}-3}z=0 \iff z=\frac{2}{3}y-\frac{1}{3}x$$

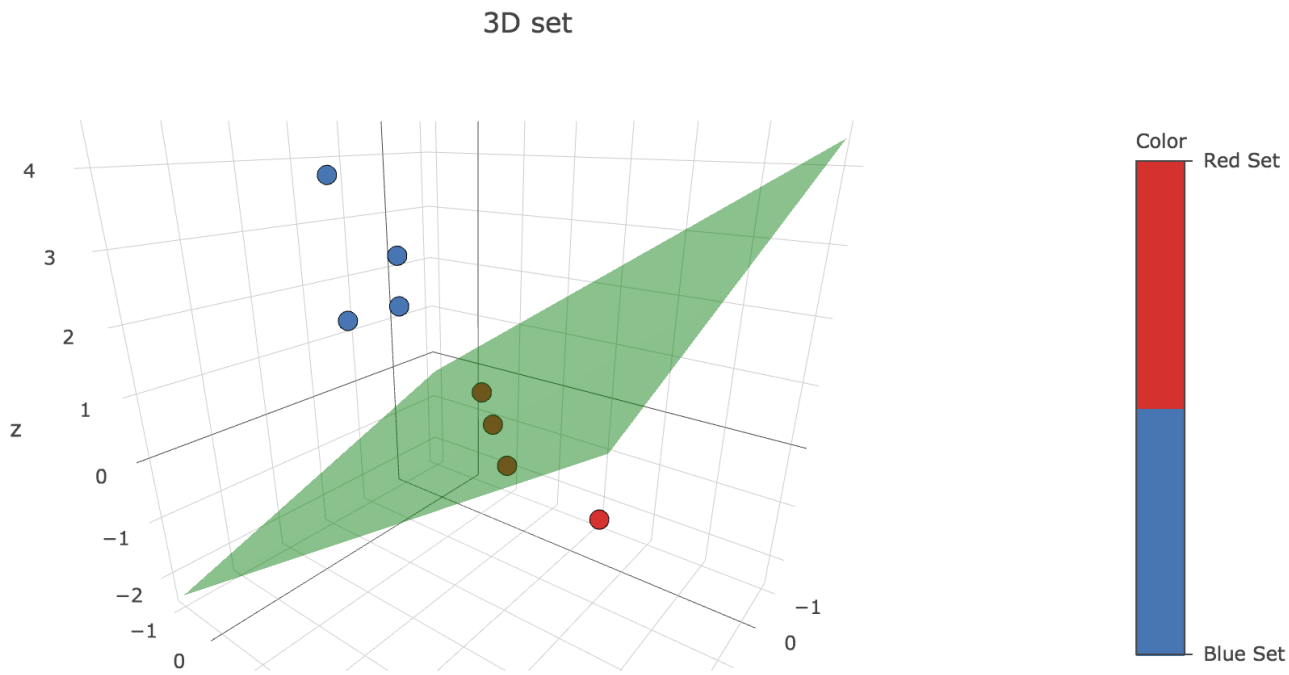

Figure 1.b.2. - Plot of the resulting decision plane (in green) for the two 3D-data sets

c. *Optional*: Code the perceptron rule in a computer, and use the datasets of exercises a) and b) to check the validity of your answers. Please attach your code and the results obtained.

Here is a python implementation:

```
def perceptron_rule(X, Y):
    nb_trials, dimension = X.shape
    w = np.ones(X.shape[1])
    w_values = [np.copy(w)]

    while True:
        # is the readout vector still changing?
        # (to halt the loop, in case it isn't)
        still_changing = False

        for i in range(nb_trials):
            w += (Y[i]-int(w.dot(X[i])>=0))*X[i]

            if np.any(w_values[-1] != w):
                still_changing = True
            w_values.append(np.copy(w))

        if not still_changing:
            break

    # Returning all the readout weight vectors,
    # except those of the last step
    # (throughout which the readout vector didn't change)
    return np.array(w_values[:-nb_trials])
```

Here are the outputs:

- for 2D-data sets:

```
array([[ 1.,  1.],
   [ 0., -1.],
   [ 0., -1.],
   [ 0., -1.],
   [ 0., -1.],
```

```
[ 0., -1.],
[ 6.,  1.],
[ 6.,  1.],
[ 6.,  1.],
[ 5., -1.],
[ 4., -4.],
[ 4., -4.],
[ 4., -4.],
[ 4., -4.],
[ 4., -4.],
[ 4., -4.]])
```

which gives the following successive decision boundaries (the successive normalized readout vectors are plotted in purple) throughout the iterations of the main loop:

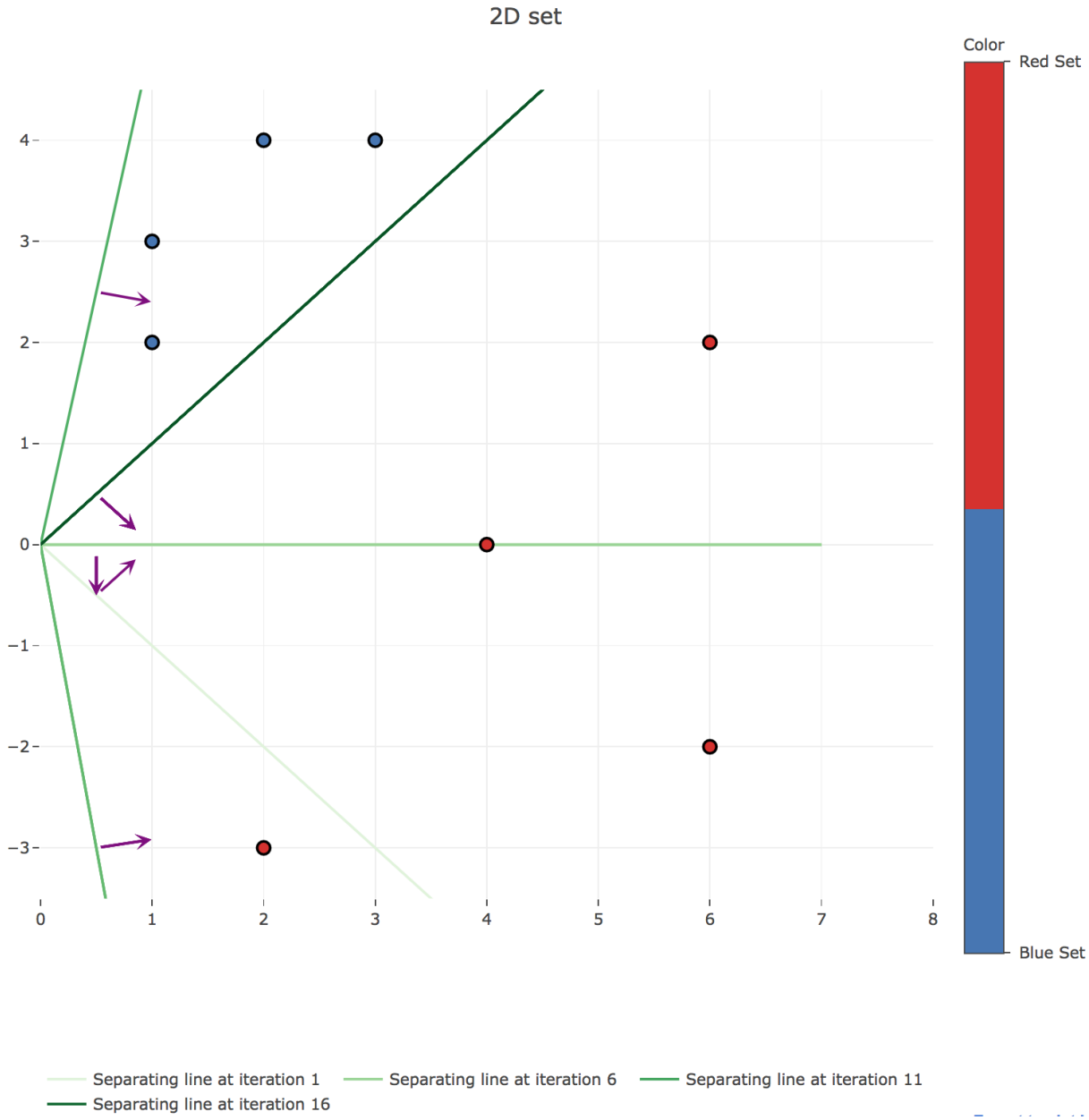

Figure 1.c.1. - Plot of the successive decision boundaries thoughout the algorithm (with their corresponding normalized readout vectors in purple)

- for the 3D-one:

```
array([[ 1.,  1.,  1.],
   [-1.,  0., -1.],
   [-1.,  0., -1.],
   [-1.,  0., -1.],
   [-1.,  0., -1.],
```

```
[-1.,  0., -1.],
[ 1.,  3., -1.],
[ 1.,  3., -1.],
[ 1.,  3., -1.],
[-1.,  2., -3.],
[-1.,  2., -3.],
[-1.,  2., -3.],
[-1.,  2., -3.],
[-1.,  2., -3.],
[-1.,  2., -3.],
[-1.,  2., -3.],
[-1.,  2., -3.]])
```

which gives the following successive decision planes throughout the iterations of the main loop:

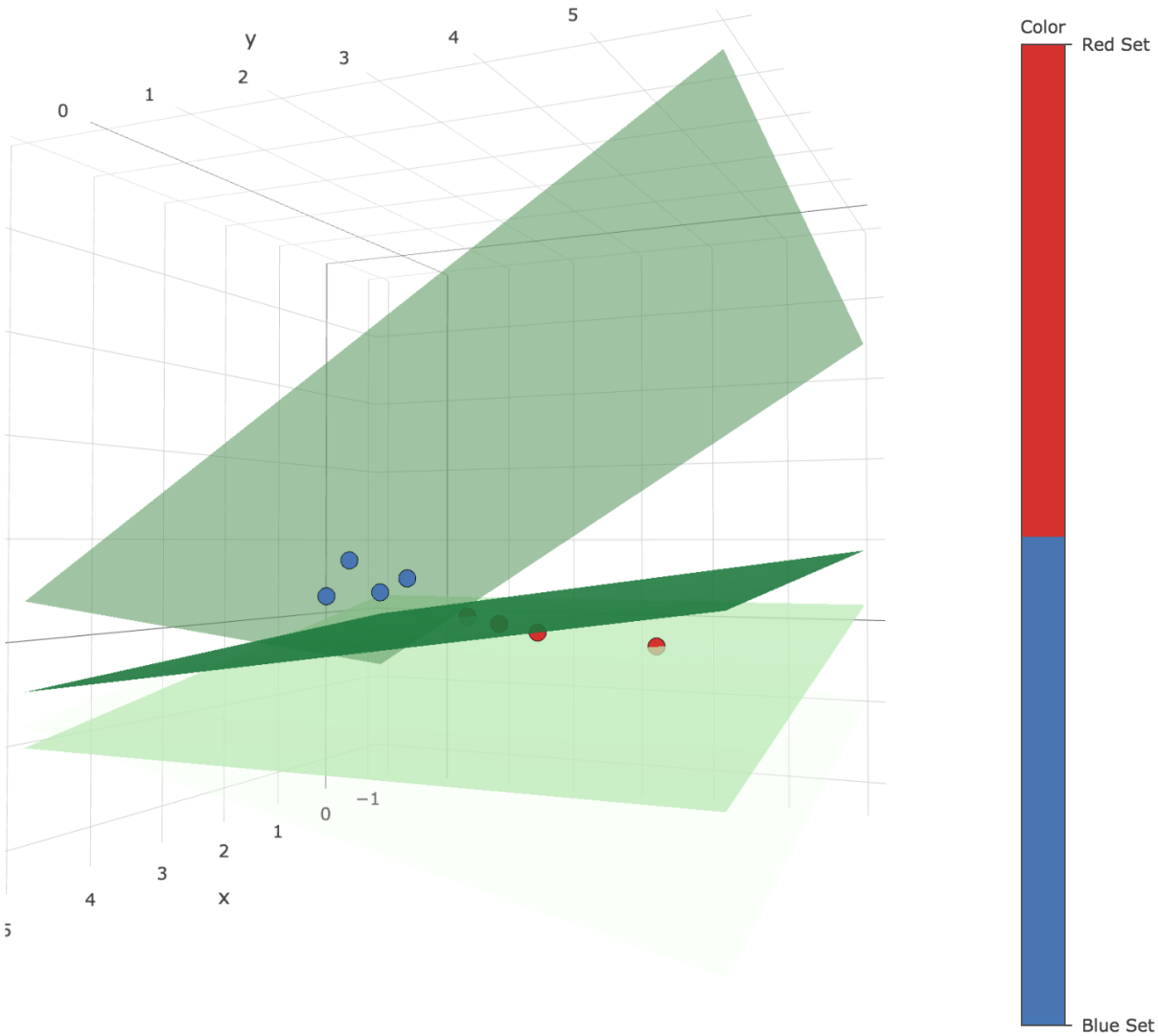

Figure 1.c.1. - Plot of the successive decision planes thoughout the algorithm (the darker, the higher the corresponding iteration number)

which is exactly what we've computed by hand.

2. Hopfield networks

---

a. Consider a recurrent network of five binary neurons. Use the Hopfield rule to determine the synaptic weights of the network so that the pattern $\xi^* = (1, -1, -1, 1, -1) \in \mathfrak{M}_{1,5}(\mathbb{R})$ is memorized. Show explicitly that ξ^* is a fixed point of the dynamics.

According to the Hopfield rule, the weights of the synaptic matrix $W \stackrel{\text{def}}{=} (w_{ij})_{1 \leq i,j \leq 5}$ are to be set to:

$$w_{ij} \stackrel{\text{def}}{=} \frac{1}{5} \xi^*_i \xi^*_j$$

The resulting synaptic matrix is:

$$W \stackrel{\text{def}}{=} \frac{1}{5} {\xi^*}^\intercal \xi^* = \frac{1}{5} \begin{pmatrix} 1 & -1 & -1 & 1 & -1 \\ -1 & 1 & 1 & -1 & 1 \\ -1 & 1 & 1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 \\ -1 & 1 & 1 & -1 & 1 \end{pmatrix}$$

As a matter of fact:

$$\begin{aligned} \operatorname{sign}\left(W {\xi^*}^\intercal\right) &= \operatorname{sign}\left(\frac{1}{5}\left({\xi^*}^\intercal \xi^*\right) {\xi^*}^\intercal\right) \\ &= \operatorname{sign}\left(\frac{1}{5} {\xi^*}^\intercal \left(\xi^* {\xi^*}^\intercal\right)\right) && \text{(because the matrix product is associative)} \\ &= \operatorname{sign}\left(\frac{1}{5} {\xi^*}^\intercal \overbrace{\left(\sum_{i=1}^{5} \underbrace{(\xi^*_i)^2}_{=1}\right)}^{=5}\right) \\ &= \operatorname{sign}\left({\xi^*}^\intercal\right) \\ &= {\xi^*}^\intercal && \text{(because } \xi^* \in \{-1,1\}^5 \text{)} \end{aligned}$$

> *So ${\xi^*}^\intercal$ (ξ^* by abuse of notation) is indeed a fixed point of the dynamics.*

b. What is the output of the network if the dynamics start from the initial state $\xi = (1,-1,-1,1,-1)$?

As shown before, if the dynamics start from the initial state $\xi = (1,-1,-1,1,-1)$, the ouput is the same state $(1,-1,-1,1,-1)$ as it is a fixed point.

Based on the teacher's remark in class, I think the question was rather meant to be about the initial state being $\xi \stackrel{\text{def}}{=} (-1,1,1,-1,1) = -\xi^*$

In this case, retrieving last computation:

$$\begin{aligned} \operatorname{sign}\left(W \xi^\intercal\right) &= \operatorname{sign}\left(\frac{1}{5}\left({\xi^*}^\intercal \xi^*\right) \xi^\intercal\right) \\ &= \operatorname{sign}\left(\frac{1}{5}\left({\xi^*}^\intercal \xi^*\right) (-\xi^*)^\intercal\right) \\ &= \operatorname{sign}\left(-\frac{1}{5}\left({\xi^*}^\intercal \xi^*\right) {\xi^*}^\intercal\right) \\ &= -\operatorname{sign}\left(\frac{1}{5} {\xi^*}^\intercal \underbrace{\left(\xi^* {\xi^*}^\intercal\right)}_{=5}\right) && \text{(because sign is an odd function)} \\ &= -\underbrace{\operatorname{sign}\left({\xi^*}^\intercal\right)}_{={\xi^*}^\intercal \text{ (cf. the previous question)}} \\ &= -{\xi^*}^\intercal \\ &= \xi^\intercal && \text{(by definition of } \xi \text{)} \end{aligned}$$

> *As a result: $\xi^\intercal \stackrel{\text{def}}{=} -{\xi^*}^\intercal$ (ξ by abuse of notation) is also a fixed point of the dynamics.*

c. Using the Hopfield rule, modify the synaptic weights so that $\eta^* = (1,1,-1,-1,-1)$ is also memorized. Verify that both ξ^* and η^* are fixed points of the dynamics.

This, the Hopfield rule yields:

$$\forall i,j \in \{1,\dots,5\}, \quad w_{ij} \stackrel{\text{def}}{=} \frac{1}{5}(\xi^*_i \xi^*_j + \eta^*_i \eta^*_j)$$

The resulting synaptic matrix is:

$$W \stackrel{\text{def}}{=} \frac{1}{5}({\xi^*}^\intercal \xi^* + {\eta^*}^\intercal \eta^*) = \frac{2}{5}\begin{pmatrix} 1 & 0 & -1 & 0 & -1 \\ 0 & 1 & 0 & -1 & 0 \\ -1 & 0 & 1 & 0 & 1 \\ 0 & -1 & 0 & 1 & 0 \\ -1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

As a matter of fact, now:

$$\begin{aligned}
\text{sign}\left(W{\xi^*}^\intercal\right) &= \text{sign}\left(\frac{1}{5}\left({\xi^*}^\intercal \xi^* + {\eta^*}^\intercal \eta^*\right){\xi^*}^\intercal\right) \\
&= \text{sign}\Big(\underbrace{\frac{1}{5}\left({\xi^*}^\intercal \xi^*\right){\xi^*}^\intercal}_{= {\xi^*}^\intercal \text{ (cf. previous questions)}} + \frac{1}{5}\left({\eta^*}^\intercal \eta^*\right){\xi^*}^\intercal\Big) \\
&= \text{sign}\Big({\xi^*}^\intercal + \frac{1}{5}{\eta^*}^\intercal \underbrace{\left(\eta^* {\xi^*}^\intercal\right)}_{= \langle \eta^*, \xi^* \rangle = \sum_{i=1}^5 \eta^*_i \xi^*_i = 1}\Big) \\
&= \text{sign}\Big({\xi^*}^\intercal + \frac{1}{5}{\eta^*}^\intercal\Big) \\
&\stackrel{\circledast}{=} \text{sign}\left({\xi^*}^\intercal\right) \qquad \text{(because } \frac{1}{5}\eta^* \in \left\{-\frac{1}{5}, \frac{1}{5}\right\}^5 \text{: it can't change the sign of } \xi^* \in \{-1,1\}^5 \text{)} \\
&= {\xi^*}^\intercal
\end{aligned}$$

And likewise:

$$\begin{aligned}
\text{sign}\left(W{\eta^*}^\intercal\right) &= \text{sign}\left(\frac{1}{5}\left({\xi^*}^\intercal \xi^* + {\eta^*}^\intercal \eta^*\right){\eta^*}^\intercal\right) \\
&= \text{sign}\Big(\frac{1}{5}\left({\xi^*}^\intercal \xi^*\right){\eta^*}^\intercal + \frac{1}{5}\left({\eta^*}^\intercal \eta^*\right){\eta^*}^\intercal\Big) \\
&= \text{sign}\Big(\frac{1}{5}{\xi^*}^\intercal \underbrace{\left(\xi^* {\eta^*}^\intercal\right)}_{= \langle \xi^*, \eta^* \rangle = 1} + \frac{1}{5}{\eta^*}^\intercal \underbrace{\left(\eta^* {\eta^*}^\intercal\right)}_{= \langle \eta^*, \eta^* \rangle = \sum_{i=1}^5 (\eta^*_i)^2 = 5}\Big) \\
&= \text{sign}\Big(\frac{1}{5}{\xi^*}^\intercal + {\eta^*}^\intercal\Big) \\
&\stackrel{\circledast}{=} \text{sign}\left({\eta^*}^\intercal\right) \qquad \text{(because } \frac{1}{5}\xi^* \in \left\{-\frac{1}{5}, \frac{1}{5}\right\}^5 \text{: it can't change the sign of } \eta^* \in \{-1,1\}^5 \text{)} \\
&= {\eta^*}^\intercal
\end{aligned}$$

> *So ${\xi^*}^\intercal$ and ${\eta^*}^\intercal$ (ξ^* and η^* by abuse of notation) are both fixed points of the dynamics.*

NB: in the computations, we see that the equality $\circledast$ is crucial to have ${\xi^*}^\intercal$ and ${\eta^*}^\intercal$ be fixed points. This stems from the fact that

$$ {\xi^*}^\intercal + \frac{1}{5}{\eta^*}^\intercal \langle \eta^*, \xi^* \rangle \quad \text{(resp. } {\eta^*}^\intercal + \frac{1}{5}{\xi^*}^\intercal \langle \eta^*, \xi^* \rangle \text{)}$$

has the same sign as ${\xi^*}^\intercal$ (resp. ${\eta^*}^\intercal$), which comes itself from the fact that $|\langle \eta^*, \xi^* \rangle| < 5$ (note that $|\langle \eta^*, \xi^* \rangle| \leq 5$ is a consequence of $\eta^*, \xi^* \in \{-1,1\}^5$, but the latter condition is not enough not ensure that $|\langle \eta^*, \xi^* \rangle| < 5$).

> *So a sufficient condition for **any** $\xi, \eta \in \{-1,1\}^N$ to be fixed points of a dynamics for which the synaptic matrix follows the Hopfield rule is:*

> $$|\underbrace{\langle \eta, \xi \rangle}_{\triangleq \sum_{i=1}^{N} \eta_i \xi_i}| < N$$
>
> *i.e. (as $\xi, \eta \in \{-1, 1\}^N$):*
>
> $$|\langle \eta, \xi \rangle| \neq N$$

d. What is the output of the network if the dynamics starts from $\xi = (1, -1, 1, 1, 1)$?

We can check that

$$\text{sign}\left(W\xi^\intercal\right) = -\eta^*$$

which is a fixed point of dynamics, as η^* is a fixed point and inverse patterns are stored as well.

> *So in one step, the fixed point $-\eta^*$ is reached when strating with the state $\xi = (1, -1, 1, 1, 1)$.*

The following code returns the markdown table of the states reached step by step for a starting state `state` :

```
N = 5
xi_ast = np.array([1., -1, -1, 1, -1])
eta_ast = np.array([1., 1, -1, -1, -1])

W = (np.outer(xi_ast, xi_ast) + np.outer(eta_ast, eta_ast))/N

def dynamics(state):
    return np.sign(W.dot(state))

fixed_point_reached = False
state = np.array([1., -1, 1, 1, 1])

table = '''
Step|State|
:--:|:--:|
0|{}
'''.format(state)

step = 1

while not fixed_point_reached:
    new_state = dynamics(state)

    if (new_state == state).all():
        fixed_point_reached = True

    table += "{}|{}\n".format(step, new_state)

    state = new_state
    step += 1

print(table)

# To display the properly formatted table in a Jupyter notebook for instance:
display(Markdown(table))
```

With the starting state `state = np.array([1., -1, 1, 1, 1])` , it gives:

STEP	STATE
0	[1. -1. 1. 1. 1.]

STEP	STATE
1	[-1. -1. 1. 1. 1.]
2	$\underbrace{[-1.\ -1.\ 1.\ 1.\ 1.]}_{=-\eta^*}$