آرین تسکر - تکلیف شماره هفت درس شبکه های عصبی - ۴۰۰۲۳۴۹۴

E 15.1. Hebb Rule with decay: $w_t = (1-\gamma)w_{t-1} + \alpha a_t p_t^T$

i. first we determine the conditions for which the test set is responded to:

$$\text{hardlim}(w^0 p^0 + w p + b) = 1 \xrightarrow[\substack{b=-0.8 \\ p^0=0 \\ p=1}]{w^0=1} \text{hardlim}(0 + w - 0.8) = 1 \rightarrow \boxed{w \geq 0.8} \; \text{I}$$

obviously, from I we know that w_0 cannot pass the test so we will begin training:

Iteration One:

$a_0 = \text{hardlim}(1\times1 + 0\times1 - 0.8) = 1$

$w_1 = (0.9)\times 0 + 0.3\times1\times1 = 0.3 < 0.8 \longrightarrow$ still fails

Iteration Two:

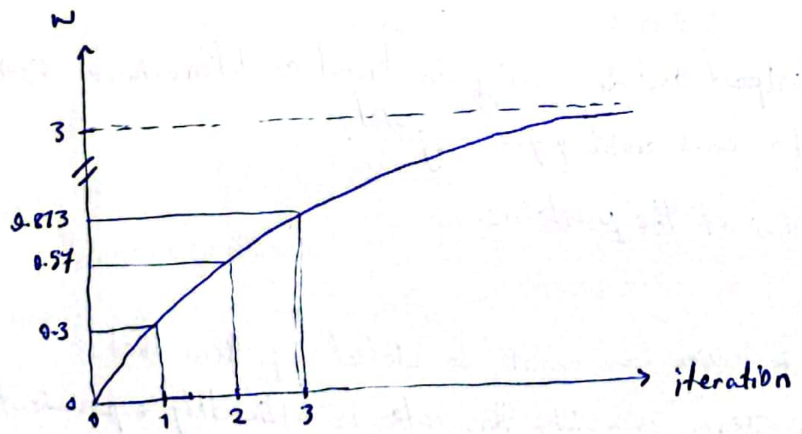
$a_1 = \text{hardlim}(1\times1 + 0.3\times1 - 0.8) = 1$

$w_2 = (0.9)\times 0.3 + 0.3\times1\times1 = 0.57 < 0.8 \longrightarrow$ still fails

Iteration Three:

$a_2 : \text{hardlim}(1\times1 + 0.57\times1 - 0.8) = 1$

$w_3 = (0.9)\times 0.57 + 0.3\times1\times1 = 0.813 > 0.8 \longrightarrow$ passes the test set ✓



$$w_{max} = \frac{\alpha}{\gamma} = 3$$

ii. the condition on w is the same as the previous part (I). So at iteration zero we pass the test.

Iteration One:

$a_0 = \text{hardlim}(1\times0 + 1\times0 - 0.8) = 0$,

$w_1 = (0.9)\times1 + 0.3\times0\times0 = 0.9 > 0.8 \longrightarrow$ still passes.

Iteration Two:

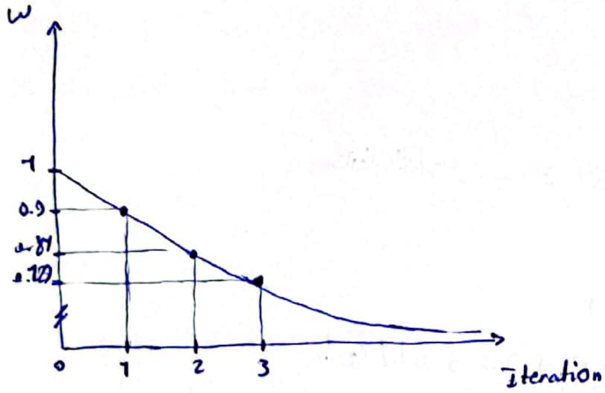
$$a_2 = \text{hardlim}(1\times 0 + 0.9\times 0 - 0.8) = 0$$

$$w_2 = (0.9)\times 0.9 + 0.3\times^{0}\times 0 = 0.81 > 0.8 \longrightarrow \text{still passes}$$

Iteration Three:

$$a_2 = \text{hardlim}(1\times 0 + 0.81\times 0 - 0.8) = 0$$

$$w_3 = (0.9)\times 0.81 + 0.3\times a\times 0 = 0.729 < 0.8 \longrightarrow \text{fails the test set}$$



E 15.4.

i. Δw_{ij} is nonzero if: $\begin{cases} \alpha \neq 0 \\ a_i \neq 0 \\ p_j \neq -w_{ij}^{old} \end{cases}$

ii. from the conditions derived in the first part and discounting the trivial condition where $\alpha = 0$ the network only learns when $a_i \neq 0$ and until $p_j = -w_{ij}^{old}$.

→ the weights approach the negative of the patterns.

iii. One possible use of this rule is for when we want to detect a pattern that is the furthest away from a given pattern. In reality this rule is ultimately equivalent to the instar rule if we always flip the output of the network after it's been trained.

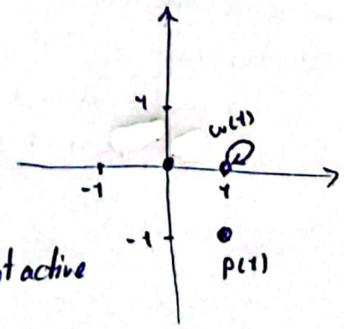
E 15.6.

i. & ii. Instar Rule: ${}_iw(q) = {}_iw(q-1) + \alpha a_i(q)(p(q) - {}_iw(q-1))$

Iteration One: $a(1) = \text{hardlim}(w^0 p^0 + Wp + b) =$
$\text{hardlim}\left(0 + [1\ 0]\begin{bmatrix}1\\-1\end{bmatrix} - 2\right) = 0$

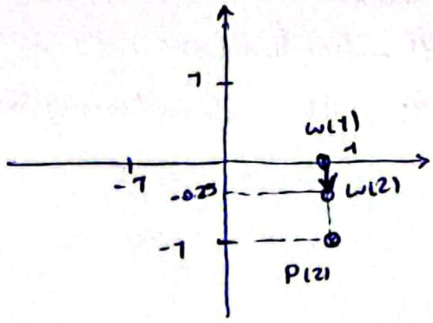
$w(1) = [1\ 0] + 0.25\times 0\times([1\ -1] - [1\ 0])$
$= [1\ 0]$

→ network does not learn because output is not active



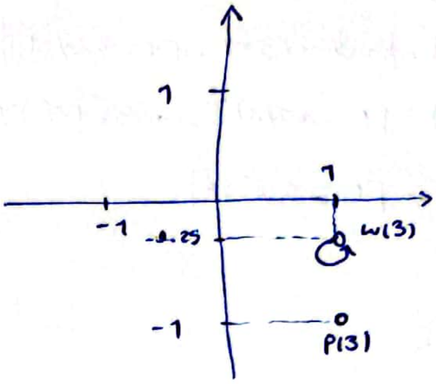
Iteration Two :

$$a(2) = \text{hardlim}(w^0 p^0 + wp + b)$$
$$= \text{hardlim}\left(3\times 1 + [1\ 0]\begin{bmatrix}1\\-1\end{bmatrix} - 2\right) = 1$$
$$w(2) = [1\ 0] + 0.25\times 1\times([1\ -1] - [1\ 0])$$
$$= [1\ 0] + [0\ -0.25] = [1\ -0.25]$$



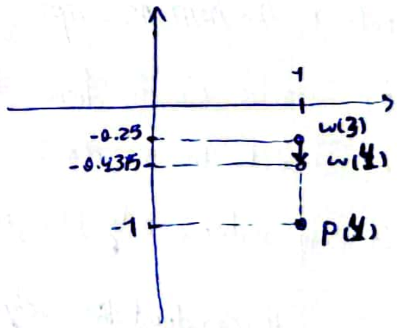
Iteration Three :

$$a(3) = \text{hardlim}(w^0 p^0 + wp + b)$$
$$= \text{hardlim}\left(3\times 0 + [1, 0.25]\begin{bmatrix}1\\-1\end{bmatrix} - 2\right) = 0$$
$$w(3) = w(2) \longrightarrow \text{no learning because } a=0$$



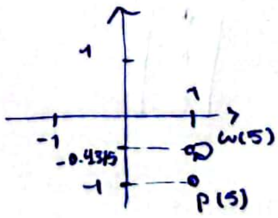
Iteration Four :

$$a(4) = \text{hardlim}(w^0 p^0 + wp + b)$$
$$= \text{hardlim}\left(3\times 1 + [1\ -0.25]\begin{bmatrix}1\\-1\end{bmatrix} - 2\right) = 1$$
$$w(4) = [1\ -0.25] + 0.25\times 1\times([1\ -1] - [1\ -0.25])$$
$$= [1\ -0.25] + [0\ -0.1875] = [1\ -0.4375]$$



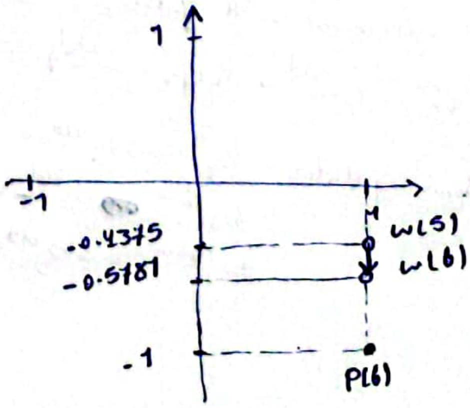
Iteration Five :

$$a(5) = \text{hardlim}(w^0 p^0 + wp + b)$$
$$= \text{hardlim}\left(3\times 0 + [1\ -0.4375]\begin{bmatrix}1\\-1\end{bmatrix} - 2\right) = 0$$
$$w(5) = w(4) \longrightarrow \text{no learning because } a=0$$



Iteration Six :

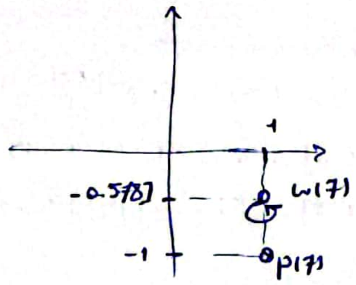
$$a(6) = \text{hardlim}\left(3\times 1 + [1\ -0.4375]\begin{bmatrix}1\\-1\end{bmatrix} - 2\right) = 1$$
$$w(6) = [1\ -0.4375] + 0.25\times 1\times([1\ -1] - [1\ -0.4375]) \approx [1\ -0.5781]$$



Iteration Seven:

$a(7) = \text{hardlim}(3\times 0 + [1 \ \ -0.5787]\begin{bmatrix}1\\-1\end{bmatrix} - 2) = 0$

$w(7) = w(6)$ → no learning because $a=0$

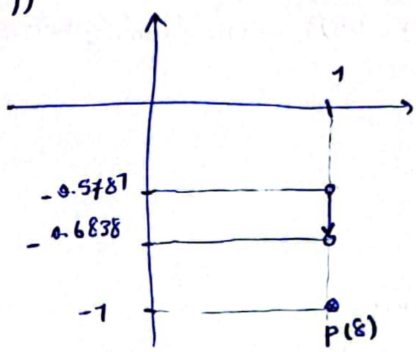


Iteration Eight:

$a(8) = \text{hardlim}(3\times 1 + [1 \ -0.5787]\begin{bmatrix}1\\-1\end{bmatrix} - 2) = 1$

$w(8) = [1 \ \ -0.5787] + 0.25\times 1\times([1 \ \ -1] - [1 \ \ -0.5787])$

$\approx [1 \ \ -0.6838]$



E15.7 → Refer to the MATLAB script.

E16.4. i. as we can see from the figure, p_1 is closest to neuron 2, p_2 is closest to neuron 3 and p_3 is closest to neuron 3 as well. This means that if we present p_1, p_2, p_3 to the network in any order, only ${}_2w, {}_3w, {}_3w$ are getting updated and thus ${}_1w$ will be a dead neuron.

ii. first we will construct the weight matrix and bias vector:

$$W = \begin{bmatrix} 0 & -1 \\ -\frac{2}{\sqrt{5}} & -\frac{1}{\sqrt{5}} \\ -\frac{1}{\sqrt{5}} & -\frac{2}{\sqrt{5}} \end{bmatrix}, \quad b = \begin{bmatrix} 0\\0\\0 \end{bmatrix}$$

(note that ws in this problem are different from those described in Figure E16.3)

Training:

①. Presenting P_1: $a_1 = \text{compet}\left(\begin{bmatrix} 0 & -1 \\ -2/\sqrt{5} & -1/\sqrt{5} \\ -\frac{1}{\sqrt{5}} & \frac{-2}{\sqrt{5}} \end{bmatrix}\begin{bmatrix}-1\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\end{bmatrix}\right) = \text{compet}\left(\begin{bmatrix}0\\2/\sqrt{5}\\1/\sqrt{5}\end{bmatrix}\right) = \begin{bmatrix}0\\1\\0\end{bmatrix}$

neuron 2 wins so we will update ${}_2w$: (learning rate assumed 0.5)

$${}_2w^{new} = {}_2w^{old} + \alpha(p_1 - {}_2w^{old}) = \begin{bmatrix}-2/\sqrt{5}\\-1/\sqrt{5}\end{bmatrix} + 0.5\left(\begin{bmatrix}-1\\0\end{bmatrix} - \begin{bmatrix}-2/\sqrt{5}\\-1/\sqrt{5}\end{bmatrix}\right) = \begin{bmatrix}\frac{-2-\sqrt{5}}{2\sqrt{5}}\\ -\frac{1}{2\sqrt{5}}\end{bmatrix}$$

bias updates: $b_1^{new} = 0.9\, b_1^{old} = 0$

$b_2^{new} = b_2^{old} - 0.2 = -0.2$

$b_3^{new} = 0.9\, b_3^{old} = 0$

$$\rightarrow W_{new} = \begin{bmatrix} 0 & -1 \\ \frac{-2-\sqrt{5}}{2\sqrt{5}} & \frac{-1}{2\sqrt{5}} \\ \frac{-1}{\sqrt{5}} & -\frac{2}{\sqrt{5}} \end{bmatrix}, \quad b_{new} = \begin{bmatrix}0\\-0.2\\0\end{bmatrix}$$

② presenting P_2: $a_2 = \text{compet}(Wp_2+b) \approx \text{compet}\left(\begin{bmatrix}-1.0\\-0.4236\\-0.8944\end{bmatrix}\right) = \begin{bmatrix}0\\1\\0\end{bmatrix}$

for the sake of brevity will convert the symbolic notations to decimal values:

$$W \approx \begin{bmatrix}0 & -1\\ -0.9472 & -0.2236\\ -0.4472 & -0.8944\end{bmatrix}$$

→ neuron 2 wins and gets updated:

$${}_2w^{new} = {}_2w^{old} + \alpha(P_2 - {}_2w^{old})$$

$$= \begin{bmatrix}-0.4736\\0.3882\end{bmatrix}$$

bias updates: $b_1^{new} = 0.9\,b_1^{old} = 0$

$b_2^{new} = b_2^{old} - 0.2 = -0.4$

$b_3^{new} = 0.9\,b_3^{old} = 0$

③. presenting P_3: $a_3 = \text{compet}(Wp_3+b) \approx \text{compet}\left(\begin{bmatrix}-0.7071\\-0.4604\\-0.9487\end{bmatrix}\right) = \begin{bmatrix}0\\1\\0\end{bmatrix}$

→ neuron 2 wins and gets updated:

${}_2w^{new} = \begin{bmatrix}0.1168\\0.5477\end{bmatrix}$, bias updated: $b_1^{new} = 0$, $b_2^{new} = -0.6$, $b_3^{new} = 0$

④. presenting P_1: $a_4 = \text{compet}(Wp_1+b) = \text{compet}\left(\begin{bmatrix}0\\-0.7168\\0.4472\end{bmatrix}\right) = \begin{bmatrix}0\\0\\1\end{bmatrix}$

→ neuron 3 wins and gets updated:

${}_3w^{new} = \begin{bmatrix}-0.7236\\-0.4472\end{bmatrix}$, bias updates: $b_1^{new} = 0$, $b_2^{new} = -0.54$, $b_3^{new} = -0.2$

⑤. presenting P_2: $a_5 = \text{compet}(Wp_2+b) = \text{compet}\left(\begin{bmatrix}-1\\0.0077\\-0.6472\end{bmatrix}\right) = \begin{bmatrix}0\\1\\0\end{bmatrix}$ ${}_2w$ wins

→ ${}_2w^{new} = \begin{bmatrix}0.0584\\0.7738\end{bmatrix}$, bias updates: $b_1^{new} = 0$, $b_2^{new} = -0.74$, $b_3^{new} = -0.18$

⑥. presenting P_3: $a_6 = \text{compet}(Wp_3+b) = \text{compet}\left(\begin{bmatrix}-0.7071\\-0.1515\\-1.0079\end{bmatrix}\right) = \begin{bmatrix}0\\1\\0\end{bmatrix}$ → w_2 wins

→ ${}_2w^{new} = \begin{bmatrix}0.3827\\0.7905\end{bmatrix}$, bias updates: $b_1^{new} = 0$, $b_2^{new} = -0.94$, $b_3^{new} = -0.162$

⑦. presenting P_1: $a_7 = \text{compet}(Wp_1+b) = \text{compet}\left(\begin{bmatrix}0\\-1.3227\\0.5616\end{bmatrix}\right) = \begin{bmatrix}0\\0\\1\end{bmatrix}$ → w_3 wins

${}_2w^{new} = \begin{bmatrix}-0.8618\\-0.2236\end{bmatrix}$, bias updates: $\begin{bmatrix}0\\-0.846\\0.362\end{bmatrix}$

8. presenting P_2 : $a_8 = \text{compet}(Wp_2+b) = \text{compet}\left(\begin{bmatrix}-1\\-0.1055\\-0.5856\end{bmatrix}\right) = \begin{bmatrix}0\\1\\0\end{bmatrix} \to {}_2w$ wins

$${}_2w^{new} = \begin{bmatrix}0.1974\\0.8702\end{bmatrix}, \quad bias^{new} = \begin{bmatrix}0\\-1.0460\\-0.3258\end{bmatrix}$$

9. presenting P_3: $a_9 = \text{compet}(Wp_3+b) = \text{compet}\left(\begin{bmatrix}-0.7071\\-0.2953\\-1.0933\end{bmatrix}\right) = \begin{bmatrix}0\\1\\0\end{bmatrix} \to {}_2w$ wins

$${}_2w^{new} = \begin{bmatrix}0.4492\\0.7887\end{bmatrix}, \quad bias^{new} = \begin{bmatrix}0\\-1.2460\\-0.2932\end{bmatrix}$$

10. presenting P_1, $a_{10} = \text{compet}(Wp_1+b) = \text{compet}\left(\begin{bmatrix}0\\-1.6952\\0.5686\end{bmatrix}\right) = \begin{bmatrix}0\\0\\1\end{bmatrix} \to {}_3w$ wins

$${}_3w^{new} = \begin{bmatrix}-0.8648\\-0.2236\end{bmatrix}, \quad bias^{new} = \begin{bmatrix}0\\-1.7214\\-0.4932\end{bmatrix}$$

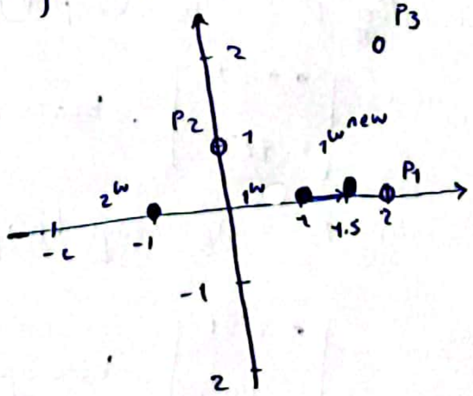
11. Presenting P_2: $a_{11} = \text{compet}(Wp_2+b) = \text{compet}\left(\begin{bmatrix}-0.1118\\-0.3327\\-0.7168\end{bmatrix}\right) = \begin{bmatrix}1\\0\\0\end{bmatrix} \to {}_1w$ wins

iii. $\longrightarrow$ as we can see above 11 presentations occured before ${}_1w$ won.
(note this problem statement is likely wrong (${}_1w$, ${}_2w$, ${}_3w$) are incorrectly initialized according to the figure in the problem statement)

E16.6. $W = \begin{bmatrix}{}_1w^T\\{}_2w^T\end{bmatrix} = \begin{bmatrix}1 & 0\\-1 & 0\end{bmatrix}$

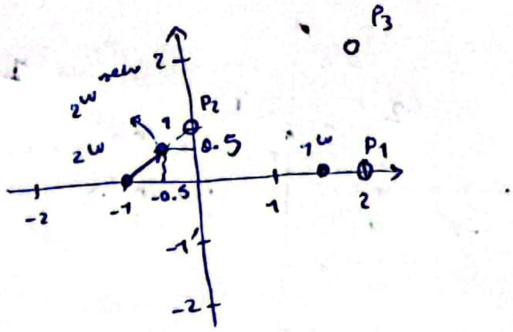
1. presenting P_1: $a_1 = \text{compet}\left(-\| W - [1\ 1]^T \times P_1^T \|_F\right) = \text{compet}\left(\begin{bmatrix}-1\\-3\end{bmatrix}\right) = \begin{bmatrix}1\\0\end{bmatrix}$

$\to {}_1w$ wins $\to$ updating: ${}_1w^{new} = \begin{bmatrix}1\\0\end{bmatrix} + 0.5\left(\begin{bmatrix}2\\0\end{bmatrix} - \begin{bmatrix}1\\0\end{bmatrix}\right) = \begin{bmatrix}1.5\\0\end{bmatrix}$



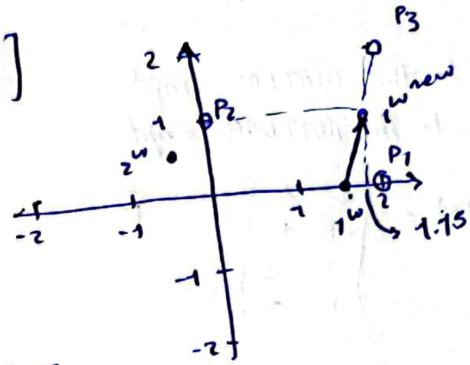
②. presenting P_2: $a_2 = \text{compet}\left(-\| w - [1\ 1]^T \times P_2^T \|_F\right) = \text{compet}\left(\begin{bmatrix} -1.8028 \\ -1.4142 \end{bmatrix}\right) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

$\rightarrow {}_2w$ wins $\longrightarrow {}_2w^{new} = \begin{bmatrix} -0.5 \\ 0.5 \end{bmatrix}$



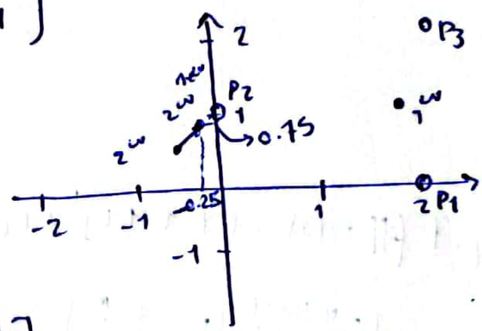
③. presenting P_3: $a_3 = \text{compet}\left(\begin{bmatrix} -2.0616 \\ -2.9155 \end{bmatrix}\right) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

$\rightarrow {}_1w$ wins $\longrightarrow {}_1w^{new} = \begin{bmatrix} 1.75 \\ 1 \end{bmatrix}$



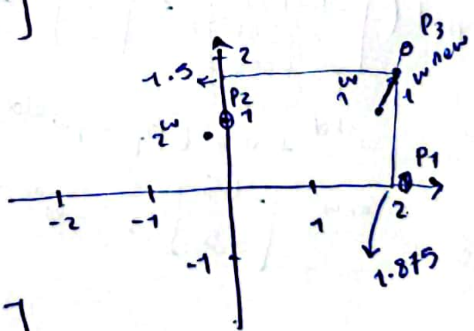
④. presenting P_2: $a_4 = \text{compet}\left(\begin{bmatrix} -1.75 \\ -0.7071 \end{bmatrix}\right) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

$\rightarrow {}_2w$ wins $\longrightarrow {}_2w^{new} = \begin{bmatrix} -0.25 \\ 0.75 \end{bmatrix}$



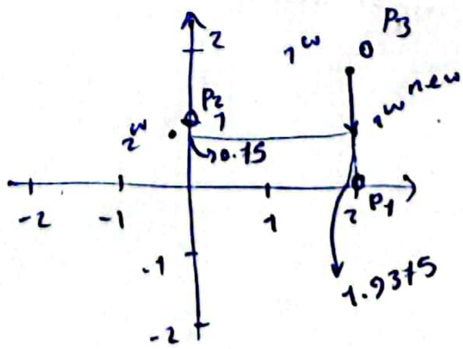
⑤. presenting P_3: $a_5 = \text{compet}\left(\begin{bmatrix} -1.0308 \\ -2.5739 \end{bmatrix}\right) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

$\rightarrow {}_1w$ wins $\longrightarrow {}_1w^{new} = \begin{bmatrix} 1.875 \\ 1.5 \end{bmatrix}$

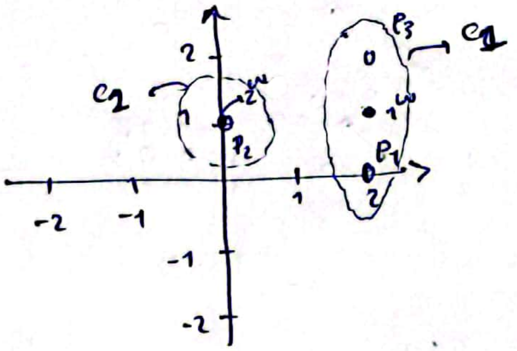


⑥. presenting P_1: $a_6 = \text{compet}\left(\begin{bmatrix} -1.5052 \\ -2.3717 \end{bmatrix}\right) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

$\rightarrow {}_1w$ wins $\longrightarrow {}_1w^{new} = \begin{bmatrix} 1.9375 \\ 0.75 \end{bmatrix}$



Eventually if the network is trained long enough the following clusters will form:

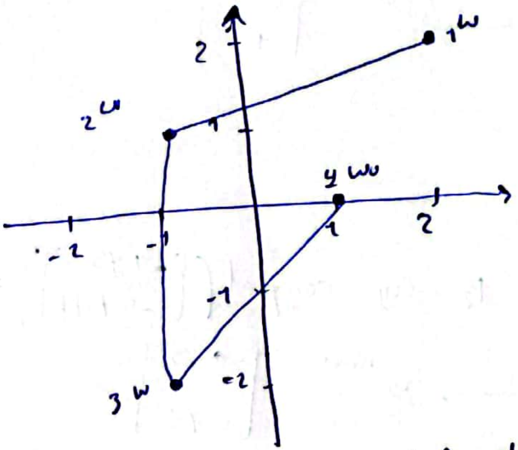


E 16.9 → Refer to the MATLAB script

E 16.10 → Refer to the MATLAB script

E 16.11 $\quad w(0) = \begin{bmatrix} 2 & 2 \\ -1 & 1 \\ -1 & -2 \\ 1 & 0 \end{bmatrix}$

i.



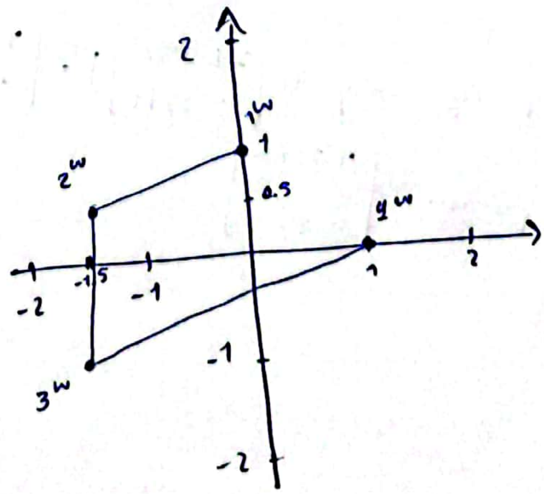
ii. $a_1 = \text{compet}\left(-\| w(0) - [1\ 1\ 1\ 1]^T \times p_1^T \|_F\right)$

$= \text{compet}\left(\begin{bmatrix} -4.4721 \\ -1.4142 \\ -2.2361 \\ -3.0000 \end{bmatrix}\right) = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \longrightarrow {}_2w$ wins $\xrightarrow{\text{neighborhood of radius one}} {}_1w$, ${}_2w$ and ${}_3w$ get updated.

${}_1w^{new} = {}_1w^{old} + 0.5 \cdot (p_1 - {}_1w^{old}) = \begin{bmatrix} 2 \\ 2 \end{bmatrix} + 0.5\left(\begin{bmatrix} -2 \\ 0 \end{bmatrix} - \begin{bmatrix} 2 \\ 2 \end{bmatrix}\right) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

Similarly ${}_2w^{new} = \begin{bmatrix} -1.5 \\ 0.5 \end{bmatrix}$, ${}_3w^{new} = \begin{bmatrix} -1 \\ -2 \end{bmatrix}$

iii. new weight vectors:



E16.14. i. we have 3 classes therefore we have 3 neurons in the second layer.

assuming one cluster is necessary for each input we will have 7 subclasses defined by the first layer so that is another 7 neurons in the first layer.

ii. the first layer determines the subclasses. since we have as many clusters as we have input samples, we can easily determine the weight of each neuron by simply setting to be the same as the pattern it will need to recognize:

$$\rightarrow w^1 = \begin{bmatrix} -1 & 1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & -1 & 1 & -1 & 1 \\ -1 & -1 & 1 & 1 & -1 & -1 & 1 \end{bmatrix}^T$$

ii. the second layer determines that which subclasses compose each class.

we know that the first two subclasses belong to class 1,

the subclasses 3 through 5 belong to class 2,

and the last 2 subclasses belong to class 3.

$$\rightarrow w^2 = \begin{bmatrix} \underbrace{\begin{matrix} 1 & 1 \\ 0 & 0 \\ 0 & 0 \end{matrix}}_{\text{first two subclasses}} & \underbrace{\begin{matrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{matrix}}_{\text{3 throughs 5}} & \underbrace{\begin{matrix} 0 & 0 \\ 0 & 0 \\ 1 & 1 \end{matrix}}_{\text{last 2 subclasses}} \end{bmatrix}$$

iii. we will test the network with the first vector of each class:

class 1: $\begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix} \longrightarrow a^1 = \text{compet}\left(\| w^1 . - \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix}^T \|_F \right) = \text{compet}\left(\begin{bmatrix} 0 \\ -2.8284 \\ -2.8284 \\ -3.4641 \\ -2 \\ -2 \\ -2 \end{bmatrix} \right)$

$= \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \longrightarrow$ vector belongs to subclass 1.

$\longrightarrow a^2 = w^2 a^1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \longrightarrow$ vector belongs to class 1 class 2: $\begin{bmatrix} -1 \\ -1 \\ 1 \end{bmatrix} \rightarrow a^1 = \text{compet}([-2.8284 \ \ -2.8284 \ \ 0 \ \ -2 \ \ -3.4641 \ \ -2 \ \ -2]^T)$

$= [0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0]^T \longrightarrow$ vector belongs to subclass 3.

$\rightarrow a^2 = w^2 a^1 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \longrightarrow$ vector belongs to class 2.

class 3: $\begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \rightarrow a^1 = \text{compet}([-2 \ \ -2 \ \ -2 \ \ -2.8284 \ \ -2.8284 \ \ 0 \ \ -2.8284])$

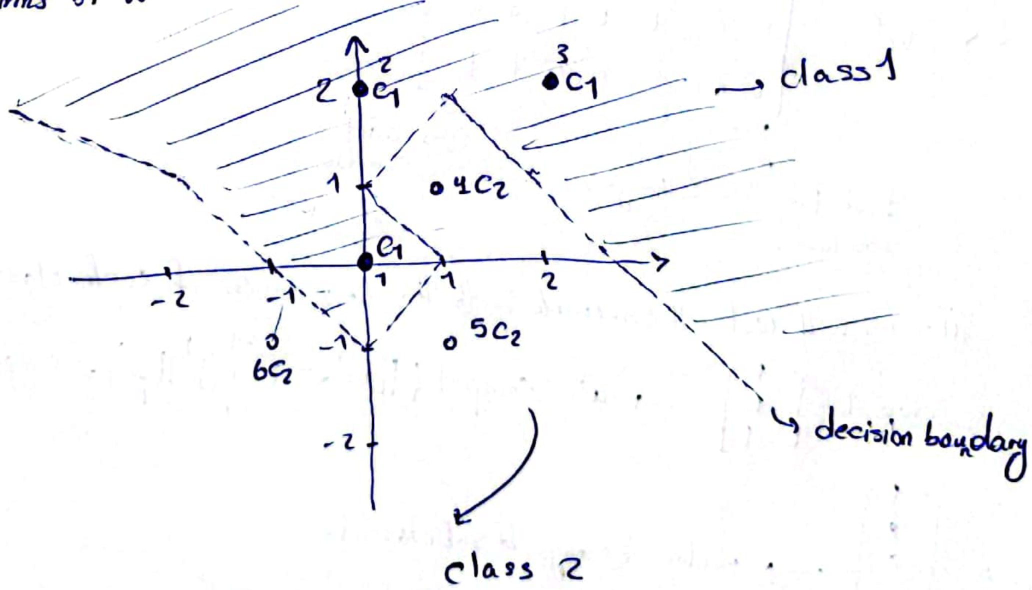
$= [0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0]^T \longrightarrow$ vector belongs to subclass 6

$\rightarrow a^2 = w^2 a^1 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \longrightarrow$ vector belongs to class 3.

E 16.18: $w^1 = \begin{bmatrix} 0 & 0 & 2 & 1 & 1 & -1 \\ 0 & 2 & 2 & 1 & -1 & -1 \end{bmatrix}^T$, $w^2 = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$

i. This network has 2 classes, judging from the number of the rows of w^2.
It also has 6 subclasses which we know from observing the number of the columns of w^2.

ii. and iii.



iv.
first we will present the input to the network.

$a_1 = \text{compet}([-2.2361 \ \ -4.1231 \ \ -5 \ \ -3.6056 \ \ -2.2361 \ \ -1])$

$= [0 \ 0 \ 0 \ 0 \ 0 \ 1] \longrightarrow$ vector belongs to subclass 6

$a_2 = w^2 a^1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, $t = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \rightarrow t_1 \neq a_{2_1} \rightarrow$ bad classification $\rightarrow$ respective neuron is moved away from pattern

${}_6w^1_{new} = {}_6w^1_{old} - 0.5(p_1 - {}_6w^1_{old}) \Rightarrow {}_6w^1_{new} = \begin{bmatrix} -1 \\ -1 \end{bmatrix} - 0.5\left(\begin{bmatrix} -1 \\ -2 \end{bmatrix} - \begin{bmatrix} -1 \\ -1 \end{bmatrix}\right) = \begin{bmatrix} -1 \\ -0.5 \end{bmatrix}$