ADALINE & MADALINE

BY

DR. ANUPAM GHOSH

ADALINE

The ADALINE (Adaptive Linear Neuron), introduced by Widrow and Hoff in 1960, is a single output unit neural net with several input units. One of the input units acts as the bias and is permanently fixed at 1. An ADALINE is trained with the help of the delta, or LMS (Least Mean Square), or Widrow-Hoff learning rule.

The salient features of *ADALINE* are

- Both the inputs and the outputs are presented in bipolar form.
- The net is trained through the delta, or LMS, or Widrow-Hoff rule. This rule tries to minimize the mean-squared error between activation and the target value.
- *ADALINE* employs the identity activation function at the output unit *during training*. This implies that during training *y_out* = *y_in*.
- During application the following bipolar step function is used for activation.

$$y = \begin{cases} 1, & \text{if } y\_in \geq 0 \\ -1, & \text{if } y\_in < 0 \end{cases}$$

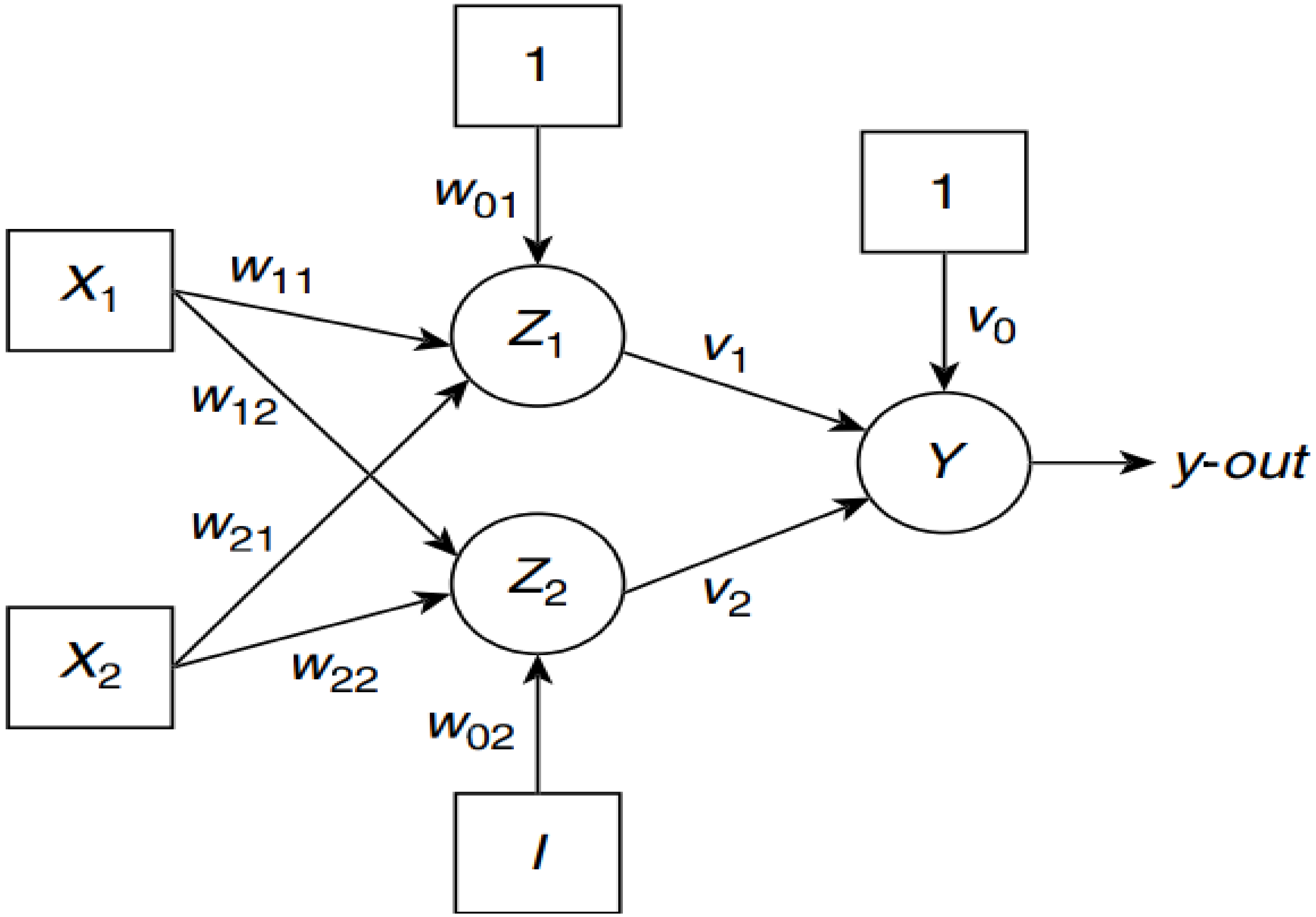
- Learning by an *ADALINE* net is sensitive to the value of the learning rate. A too large learning rate may prevent the learning process to converge. On the other hand, if the learning rate is too low, the learning process is extremely slow. Usually, the learning rate is set on the basis of the inequality $.1 \leq m \times \eta \leq 1.0$, where m is the number of input units.

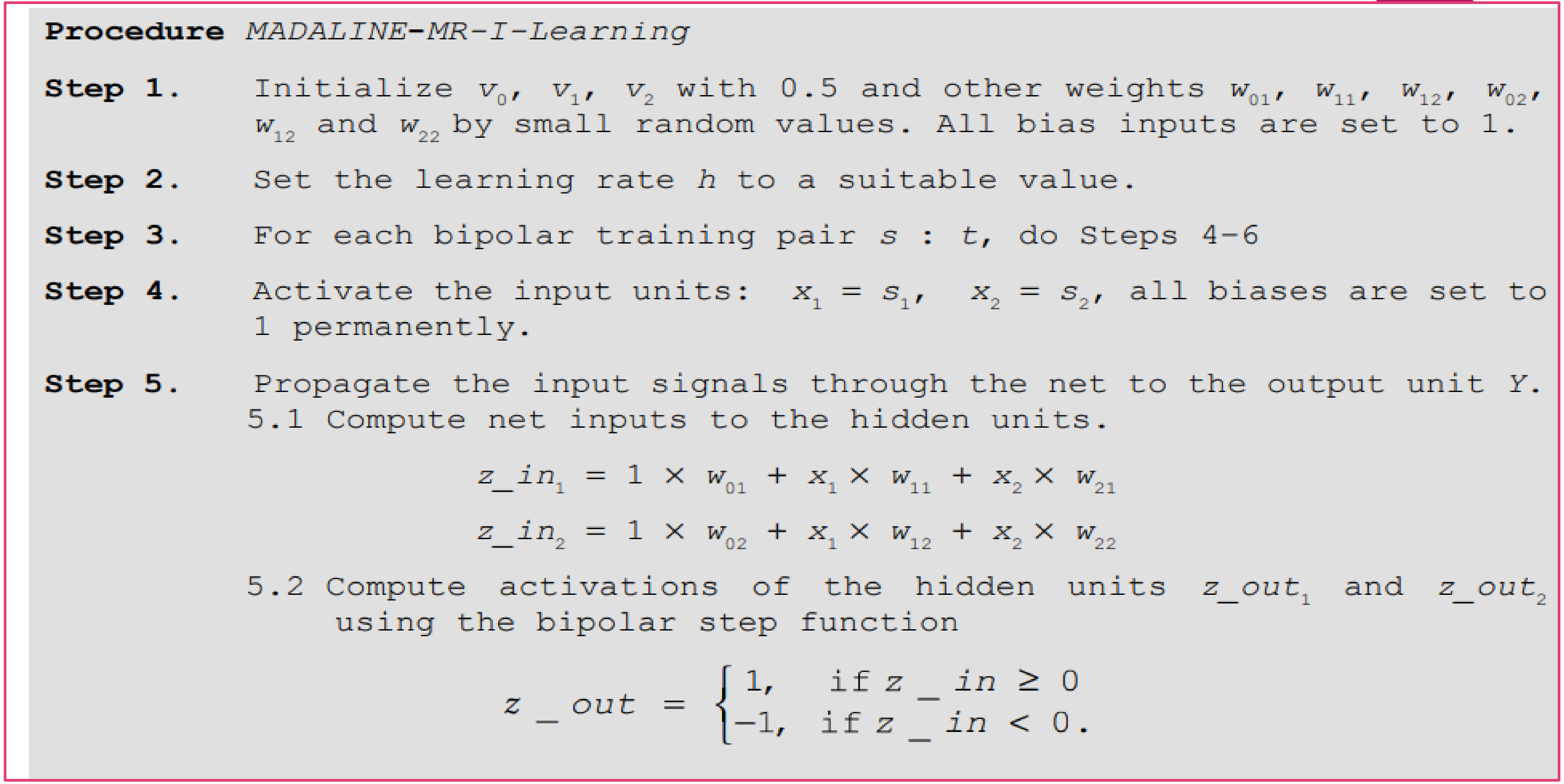
MADALINE

- Several *ADALINE*s arranged in a multilayer net is known as *Many ADALINES*, or *Many Adaptive Linear Neurons*, or *MADALINE* in short. The architecture of a two input, one output, one hidden layer consisting of two hidden *MADALINE* is shown in Figure. *MADALINE* is computationally more powerful than *ADALINE*. The enhanced computational power of the *MADALINE* is due to the hidden *ADALINE* units. Salient features of *MADALINE* are mentioned below.
- All units, except the inputs, employ the same activation function as in *ADALINE*, *i.e.*,

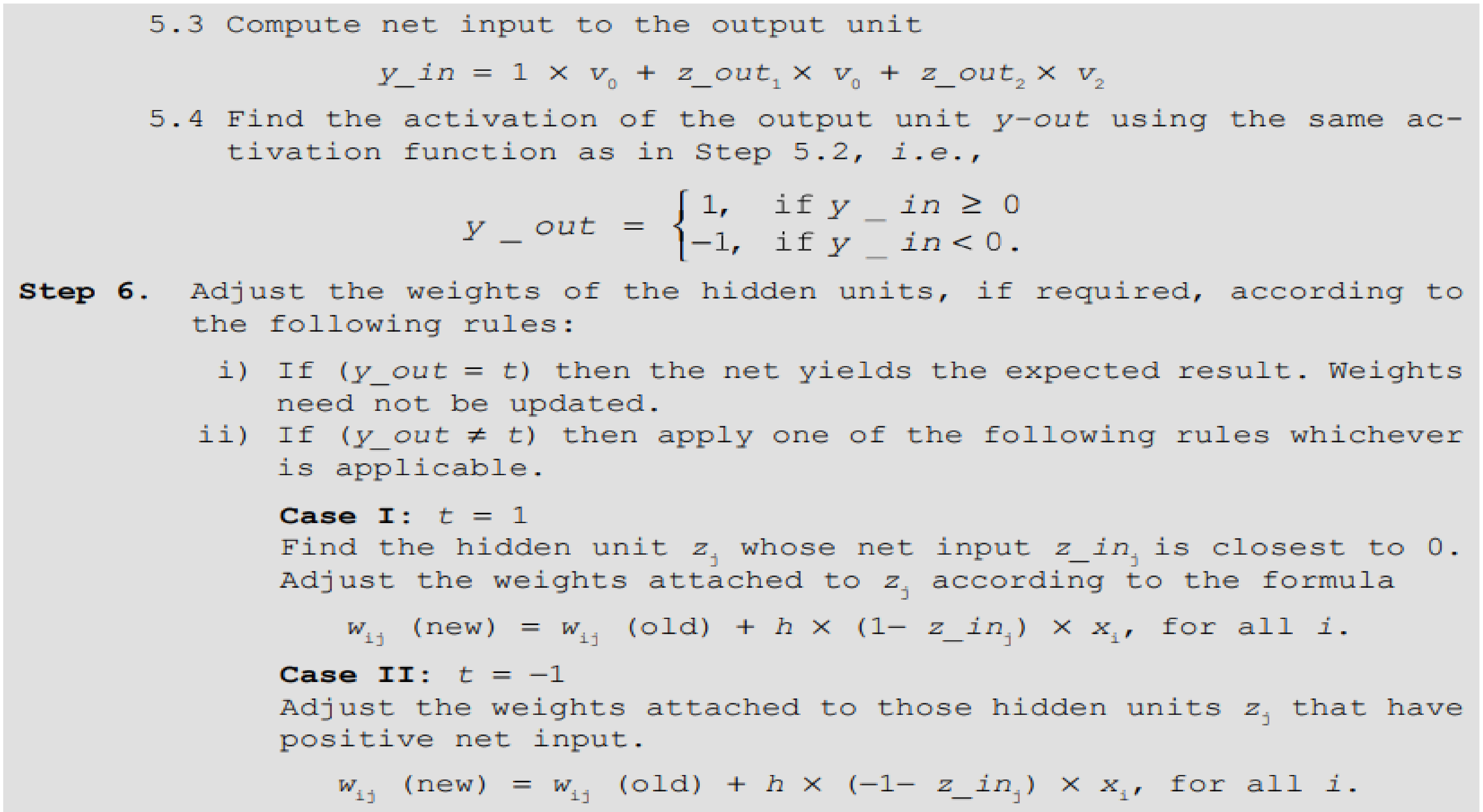
$$f(x) = \begin{cases} 1, & \text{if } x \geq 0, \\ -1, & \text{if } x < 0. \end{cases}$$

- As mentioned earlier, the enhanced computational power of the *MADALINE* is due to the hidden *ADALINE* units. However, existence of the hidden units makes the training process more complicated.
- There are two training algorithms for *MADALINE*, *viz.*, *MR-I* and *MR-II*.


1
w_{01}
1
X_1
w_{11}
Z_1
v_1
v_0
w_{12}
Y
y-out
w_{21}
Z_2
v_2
X_2
w_{22}
w_{02}
I



```
Procedure MADALINE-MR-I-Learning

Step 1.   Initialize $v_0$, $v_1$, $v_2$ with 0.5 and other weights $w_{01}$, $w_{11}$, $w_{12}$, $w_{02}$,
          $w_{12}$ and $w_{22}$ by small random values. All bias inputs are set to 1.

Step 2.   Set the learning rate $h$ to a suitable value.

Step 3.   For each bipolar training pair $s : t$, do Steps 4-6

Step 4.   Activate the input units: $x_1 = s_1$, $x_2 = s_2$, all biases are set to
          1 permanently.

Step 5.   Propagate the input signals through the net to the output unit $Y$.
          5.1 Compute net inputs to the hidden units.

                  $z\_in_1 = 1 \times w_{01} + x_1 \times w_{11} + x_2 \times w_{21}$

                  $z\_in_2 = 1 \times w_{02} + x_1 \times w_{12} + x_2 \times w_{22}$

          5.2 Compute activations of the hidden units $z\_out_1$ and $z\_out_2$
              using the bipolar step function

                  $z\_out = \begin{cases} 1, & \text{if } z\_in \geq 0 \\ -1, & \text{if } z\_in < 0. \end{cases}$
```



5.3 Compute net input to the output unit

$$y\_in = 1 \times v_0 + z\_out_1 \times v_0 + z\_out_2 \times v_2$$

5.4 Find the activation of the output unit *y-out* using the same activation function as in Step 5.2, *i.e.*,

$$y\_out = \begin{cases} 1, & \text{if } y\_in \geq 0 \\ -1, & \text{if } y\_in < 0. \end{cases}$$

Step 6. Adjust the weights of the hidden units, if required, according to the following rules:

i) If ($y\_out = t$) then the net yields the expected result. Weights need not be updated.

ii) If ($y\_out \neq t$) then apply one of the following rules whichever is applicable.

Case I: $t = 1$

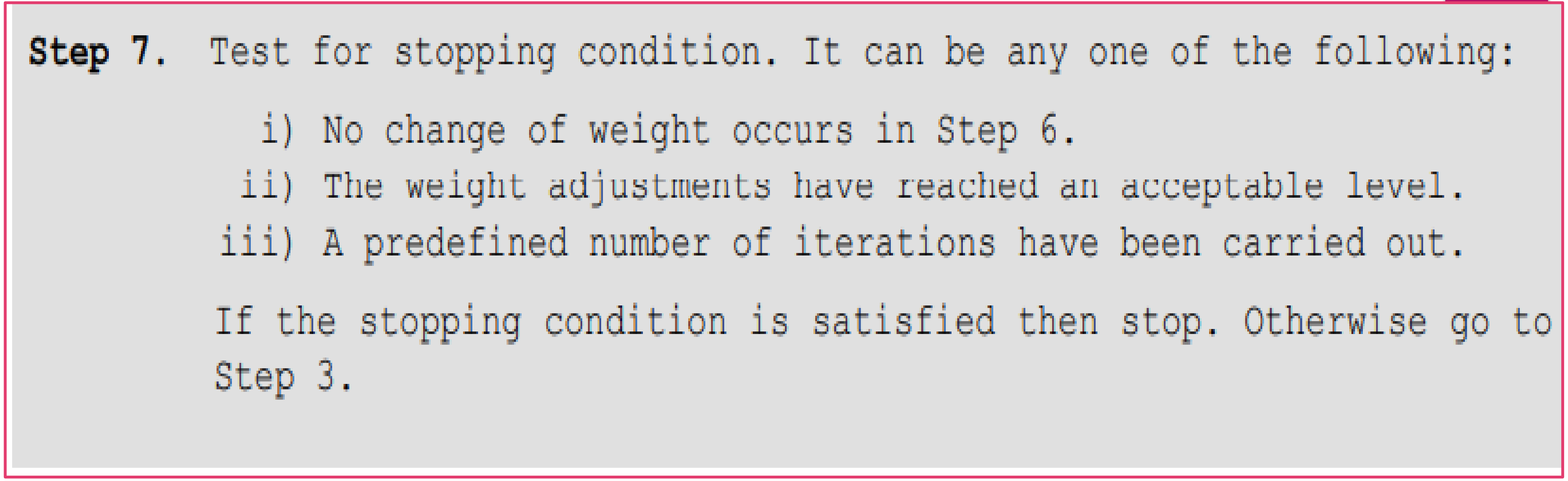
Find the hidden unit z_j whose net input z_in_j is closest to 0. Adjust the weights attached to z_j according to the formula

$$w_{ij} \text{ (new)} = w_{ij} \text{ (old)} + h \times (1 - z\_in_j) \times x_i, \text{ for all } i.$$

Case II: $t = -1$

Adjust the weights attached to those hidden units z_j that have positive net input.

$$w_{ij} \text{ (new)} = w_{ij} \text{ (old)} + h \times (-1 - z\_in_j) \times x_i, \text{ for all } i.$$


Step 7. Test for stopping condition. It can be any one of the following:

i) No change of weight occurs in Step 6.
ii) The weight adjustments have reached an acceptable level.
iii) A predefined number of iterations have been carried out.

If the stopping condition is satisfied then stop. Otherwise go to Step 3.

Sample Run

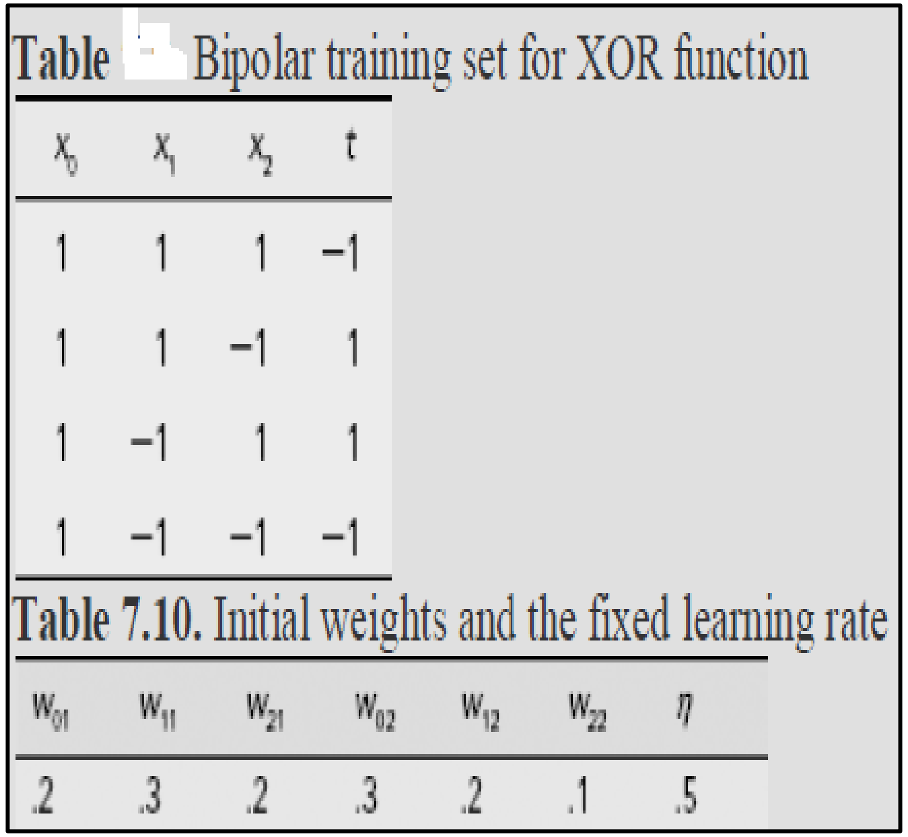
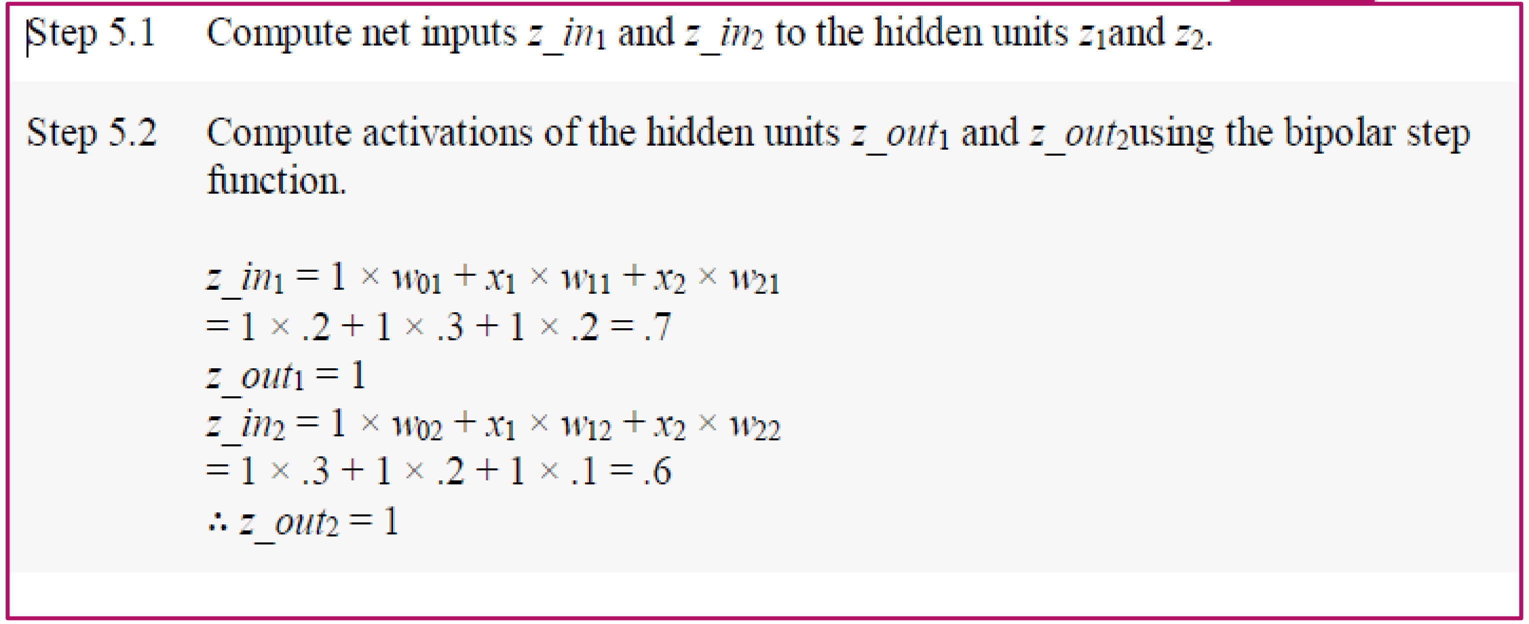


Table Bipolar training set for XOR function

x_0	x_1	x_2	t
1	1	1	−1
1	1	−1	1
1	−1	1	1
1	−1	−1	−1

Table 7.10. Initial weights and the fixed learning rate

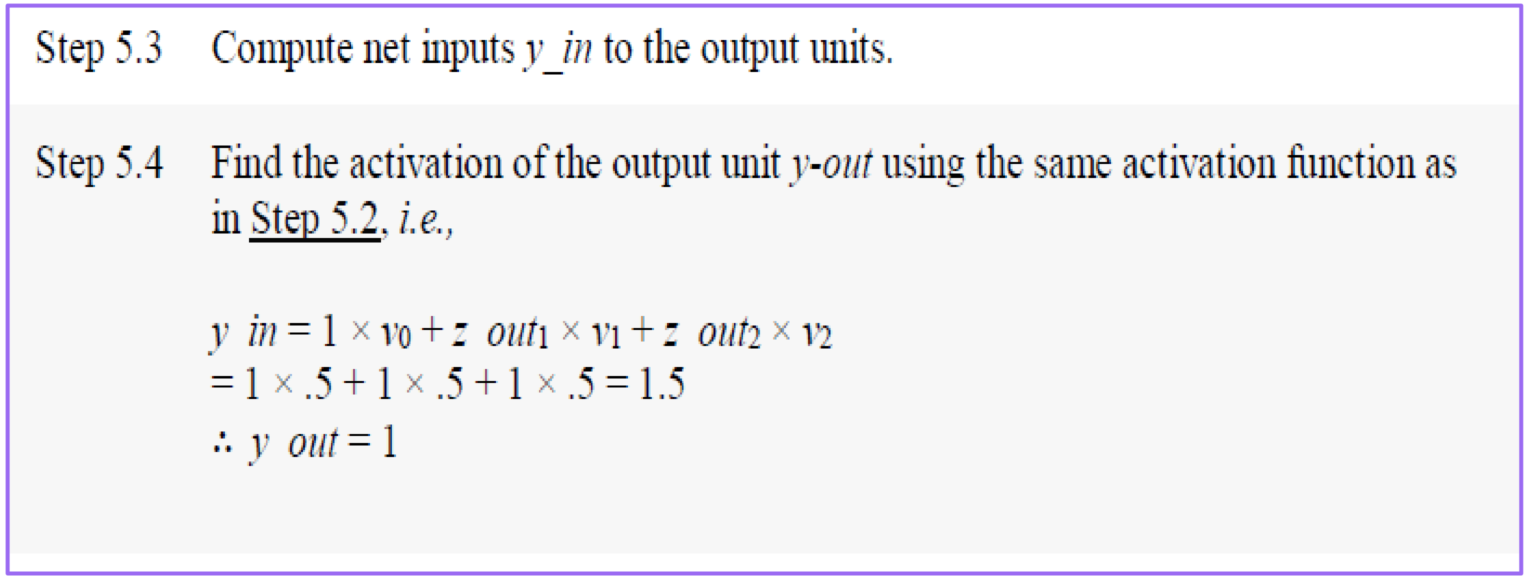
w_{01}	w_{11}	w_{21}	w_{02}	w_{12}	w_{22}	η
.2	.3	.2	.3	.2	.1	.5



Step 5.1 Compute net inputs z_in_1 and z_in_2 to the hidden units $z_1$and z_2.

Step 5.2 Compute activations of the hidden units z_out_1 and $z_out_2$using the bipolar step function.

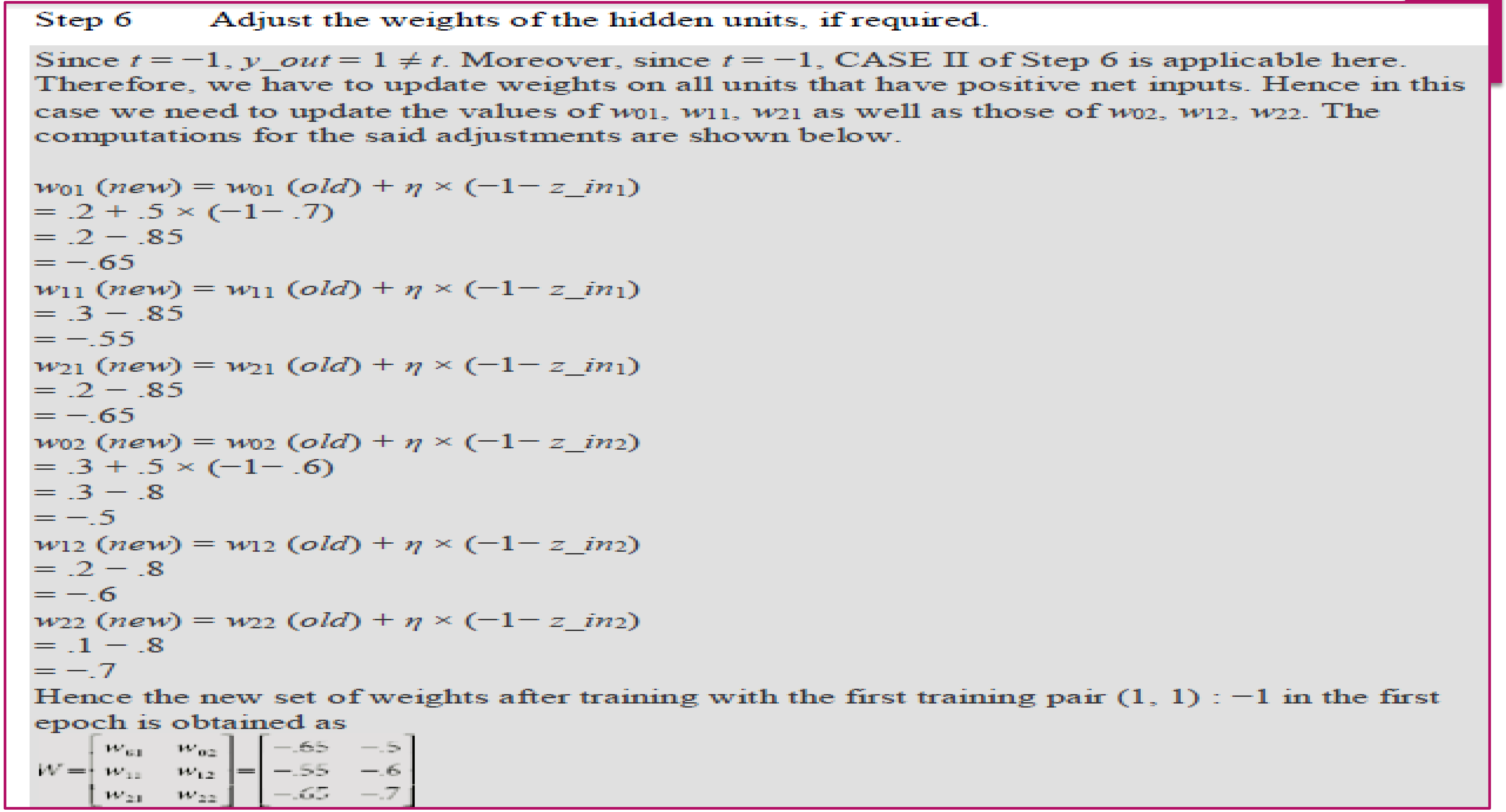
$$z\_in_1 = 1 \times w_{01} + x_1 \times w_{11} + x_2 \times w_{21}$$
$$= 1 \times .2 + 1 \times .3 + 1 \times .2 = .7$$
$$z\_out_1 = 1$$
$$z\_in_2 = 1 \times w_{02} + x_1 \times w_{12} + x_2 \times w_{22}$$
$$= 1 \times .3 + 1 \times .2 + 1 \times .1 = .6$$
$$\therefore z\_out_2 = 1$$



Step 5.3 Compute net inputs y_in to the output units.

Step 5.4 Find the activation of the output unit *y-out* using the same activation function as in Step 5.2, *i.e.*,

$$y\_in = 1 \times v_0 + z\_out_1 \times v_1 + z\_out_2 \times v_2$$
$$= 1 \times .5 + 1 \times .5 + 1 \times .5 = 1.5$$
$$\therefore y\_out = 1$$



Step 6 Adjust the weights of the hidden units, if required.

Since $t = -1$, $y\_out = 1 \neq t$. Moreover, since $t = -1$, CASE II of Step 6 is applicable here. Therefore, we have to update weights on all units that have positive net inputs. Hence in this case we need to update the values of w_{01}, w_{11}, w_{21} as well as those of w_{02}, w_{12}, w_{22}. The computations for the said adjustments are shown below.

$w_{01}\ (new) = w_{01}\ (old) + \eta \times (-1 - z\_in_1)$
$= .2 + .5 \times (-1 - .7)$
$= .2 - .85$
$= -.65$
$w_{11}\ (new) = w_{11}\ (old) + \eta \times (-1 - z\_in_1)$
$= .3 - .85$
$= -.55$
$w_{21}\ (new) = w_{21}\ (old) + \eta \times (-1 - z\_in_1)$
$= .2 - .85$
$= -.65$
$w_{02}\ (new) = w_{02}\ (old) + \eta \times (-1 - z\_in_2)$
$= .3 + .5 \times (-1 - .6)$
$= .3 - .8$
$= -.5$
$w_{12}\ (new) = w_{12}\ (old) + \eta \times (-1 - z\_in_2)$
$= .2 - .8$
$= -.6$
$w_{22}\ (new) = w_{22}\ (old) + \eta \times (-1 - z\_in_2)$
$= .1 - .8$
$= -.7$

Hence the new set of weights after training with the first training pair (1, 1) : −1 in the first epoch is obtained as

$$W = \begin{bmatrix} w_{01} & w_{02} \\ w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix} = \begin{bmatrix} -.65 & -.5 \\ -.55 & -.6 \\ -.65 & -.7 \end{bmatrix}$$

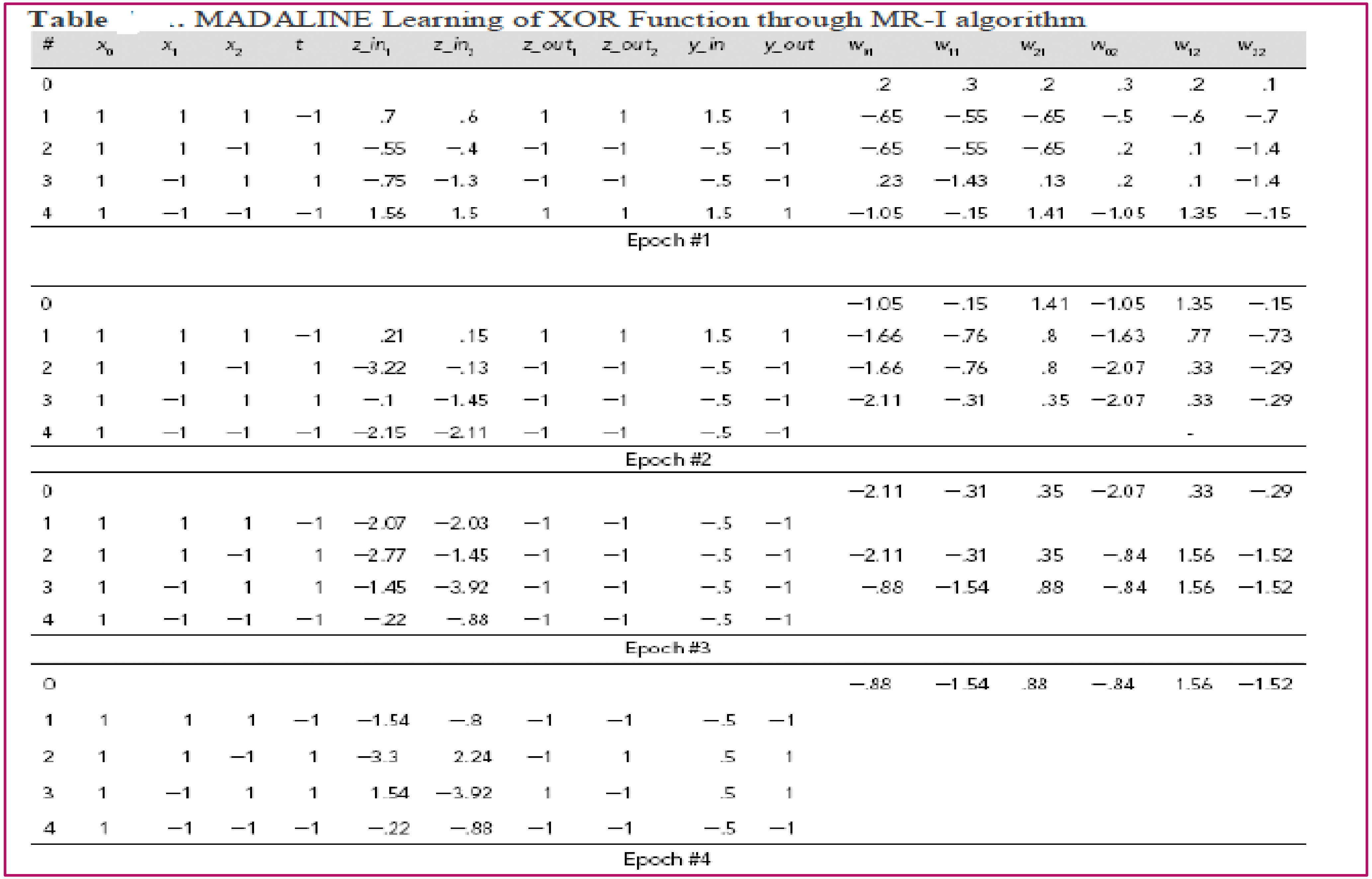


Table .. MADALINE Learning of XOR Function through MR-I algorithm

#	x_0	x_1	x_2	t	z_in_1	z_in_2	z_out_1	z_out_2	y_in	y_out	w_{01}	w_{11}	w_{21}	w_{02}	w_{12}	w_{22}
0											.2	.3	.2	.3	.2	.1
1	1	1	1	−1	.7	.6	1	1	1.5	1	−.65	−.55	−.65	−.5	−.6	−.7
2	1	1	−1	1	−.55	−.4	−1	−1	−.5	−1	−.65	−.55	−.65	.2	.1	−1.4
3	1	−1	1	1	−.75	−1.3	−1	−1	−.5	−1	.23	−1.43	.13	.2	.1	−1.4
4	1	−1	−1	−1	1.56	1.5	1	1	1.5	1	−1.05	−.15	1.41	−1.05	1.35	−.15
								Epoch #1								
0											−1.05	−.15	1.41	−1.05	1.35	−.15
1	1	1	1	−1	.21	.15	1	1	1.5	1	−1.66	−.76	.8	−1.63	.77	−.73
2	1	1	−1	1	−3.22	−.13	−1	−1	−.5	−1	−1.66	−.76	.8	−2.07	.33	−.29
3	1	−1	1	1	−.1	−1.45	−1	−1	−.5	−1	−2.11	−.31	.35	−2.07	.33	−.29
4	1	−1	−1	−1	−2.15	−2.11	−1	−1	−.5	−1					-	
								Epoch #2								
0											−2.11	−.31	.35	−2.07	.33	−.29
1	1	1	1	−1	−2.07	−2.03	−1	−1	−.5	−1						
2	1	1	−1	1	−2.77	−1.45	−1	−1	−.5	−1	−2.11	−.31	.35	−.84	1.56	−1.52
3	1	−1	1	1	−1.45	−3.92	−1	−1	−.5	−1	−.88	−1.54	.88	−.84	1.56	−1.52
4	1	−1	−1	−1	−.22	−.88	−1	−1	−.5	−1						
								Epoch #3								
0											−.88	−1.54	.88	−.84	1.56	−1.52
1	1	1	1	−1	−1.54	−.8	−1	−1	−.5	−1						
2	1	1	−1	1	−3.3	2.24	−1	1	.5	1						
3	1	−1	1	1	1.54	−3.92	1	−1	.5	1						
4	1	−1	−1	−1	−.22	−.88	−1	−1	−.5	−1						
								Epoch #4								

It will be Continued….