Πίνακας 9.1: *Appropriate target (category) values*

reference pattern	target value	category
p_1	- 1	A
p_2	- 1	A
p_3	- 1	A
p_4	- 1	A
p_5	1	B
p_6	1	B
p_7	1	B

Κεφάλαιο 9

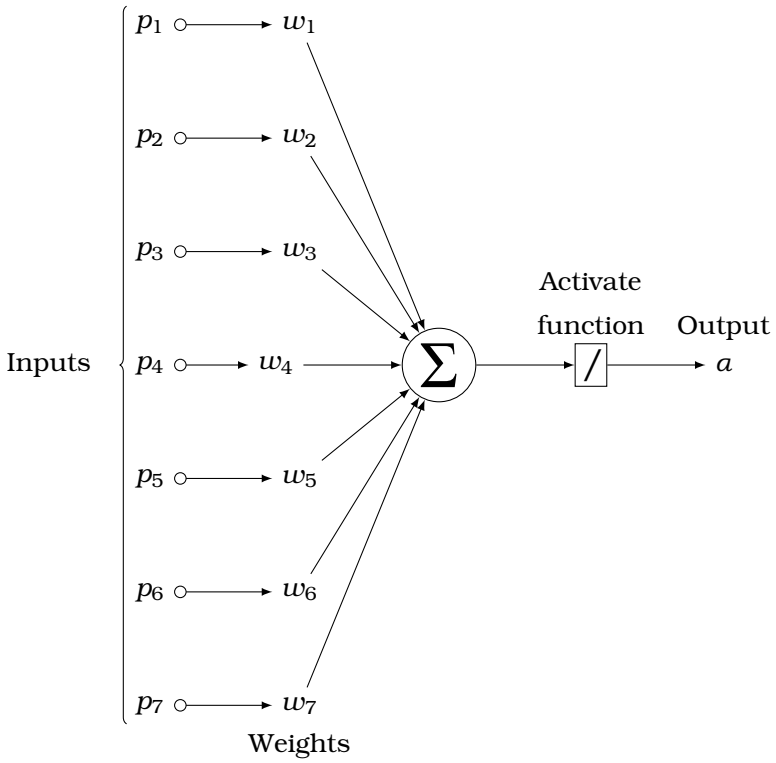
Πρόβλημα 9

9.1 A. Select appropriate target (category) values

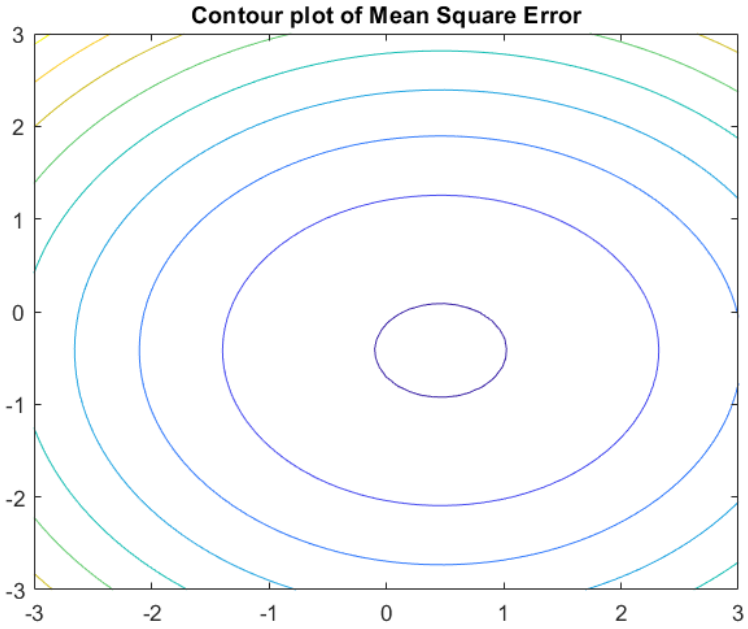
As reference patterns p_1, p_2, p_3 and p_4 belong to category A, I set their target values equal to -1. For the reference patterns belonging to category B, p_5, p_6 and p_7, I set their target values equal to 1 . Selected target values of each reference pattern are shown in table 9.1 .

9.2 B. Draw network diagram for an ADALINE network with no BIAS

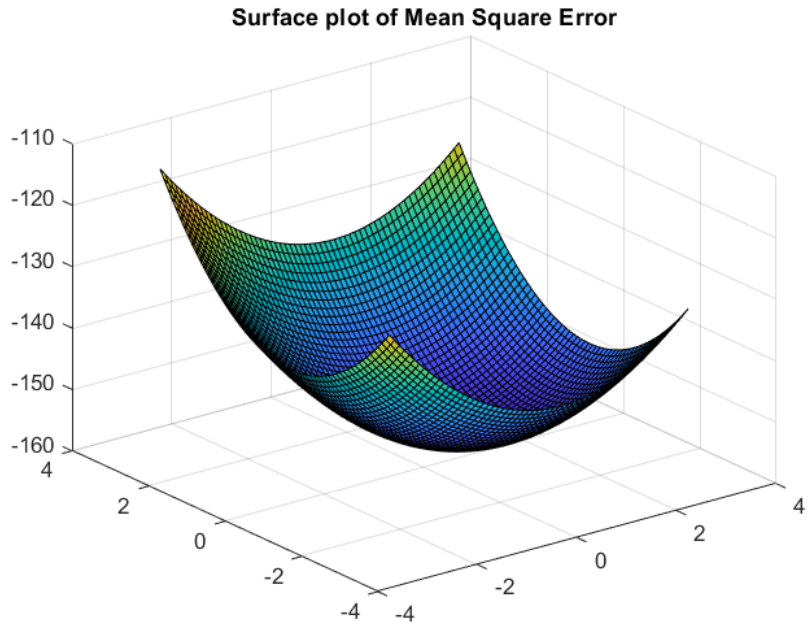
Note that p, w are 2-D vectors, but are shown as 1-D vectors at this diagram . So Input is 7 2-D vectors, weights are also 7 2-D vectors, while output a is a 7 2-D vector . Bias is not shown at the diagram as it is an ADALINE vector with no bias Activate function is purelin .



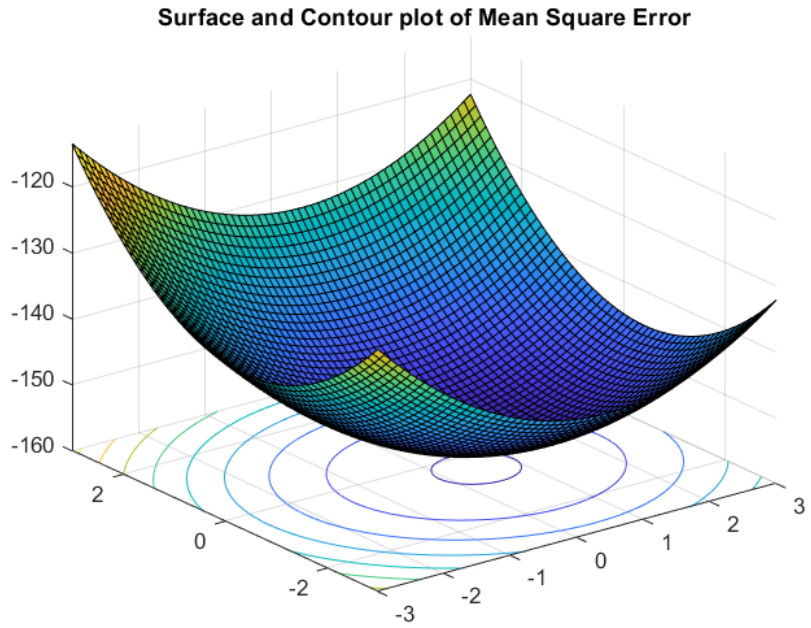
9.3 C. Sketch the contour plot of the mean square error performance index



Εικόνα 9.1: *Contour plot of Mean Square Error Performance Index*



Εικόνα 9.2: *Surface plot of Mean Square Error Performance Index*



Εικόνα 9.3: *Surface and Contour plot of Mean Square Error Performance Index*

9.3.1 Find MSE index

$$F(x) = c - 2x^T h + x^T Rx \tag{9.1}$$

όπου

$$c = E[t^2], h = E[tz], R = E[zz^T] \tag{9.2}$$

$c = E[t^2] = (-1)^2 \cdot 0.1 + (-1)^2 \cdot 0.1 + (-1)^2 \cdot 0.1 + (-1)^2 \cdot 0.1 + 1^2 \cdot 0.2 + 1^2 \cdot 0.2 + 1^2 \cdot 0.2 + 1^2 \cdot 0.2 = 1$

$h = E[tz] = 0.1 \cdot (-1) \cdot \begin{vmatrix} 0 \\ 0 \end{vmatrix} + 0.1 \cdot (-1) \cdot \begin{vmatrix} 0 \\ 1 \end{vmatrix} + 0.1 \cdot (-1) \cdot \begin{vmatrix} 1 \\ 0 \end{vmatrix} + 0.1 \cdot (-1) \cdot \begin{vmatrix} -1 \\ -1 \end{vmatrix} + 0.2 \cdot 1 \cdot \begin{vmatrix} 2.1 \\ 0 \end{vmatrix} + 0.2 \cdot 1 \cdot \begin{vmatrix} 0 \\ -2.5 \end{vmatrix} + 0.2 \cdot 1 \cdot \begin{vmatrix} 1.6 \\ -1.6 \end{vmatrix} = \begin{vmatrix} 0.74 \\ -0.82 \end{vmatrix}$

$R = E[zz^T] = p_1 \cdot p_1^T \cdot Prob_1 + p_2 \cdot p_2^T \cdot Prob_2 + p_3 \cdot p_3^T \cdot Prob_3 + p_4 \cdot p_4^T \cdot Prob_4 + p_5 \cdot p_5^T \cdot Prob_5 + p_6 \cdot p_6^T \cdot Prob_6 + p_7 \cdot p_7^T \cdot Prob_7 = \begin{vmatrix} 0 & 0 \\ 0 & 0 \end{vmatrix} \cdot 0.1 + \begin{vmatrix} 0 & 0 \\ 0 & 1 \end{vmatrix} \cdot 0.1 + \begin{vmatrix} 1 & 0 \\ 0 & 0 \end{vmatrix} \cdot 0.1 + \begin{vmatrix} 1 & 1 \\ 1 & 1 \end{vmatrix} \cdot 0.1 + \begin{vmatrix} 4.41 & 0 \\ 0 & 0 \end{vmatrix} \cdot 0.2 + \begin{vmatrix} 0 & 0 \\ 0 & 6.25 \end{vmatrix} \cdot 0.2 + \begin{vmatrix} 2.56 & -2.56 \\ -2.56 & 2.56 \end{vmatrix} \cdot 0.2 = \begin{vmatrix} 1.594 & -0.412 \\ -0.412 & 1.962 \end{vmatrix}$

Επομένως,

$$F(x) = c - 2x^T h + x^T R x = 1 - 1.48w_1 + 1.64w_2 - 0.824w_1 w_2 + 1.594w_1^2 + 1.962w_2^2 \quad (9.3)$$

9.3.2 Find center

Solve for minimum w*

$$w^* = R^{-1} h = \begin{vmatrix} 1.594 & -0.412 \\ -0.412 & 1.962 \end{vmatrix}^{-} \cdot \begin{vmatrix} 0.74 \\ -0.82 \end{vmatrix} = \begin{vmatrix} 0.66335 & 0.13929 \\ 0.13929 & 0.53893 \end{vmatrix}^{-} \cdot \begin{vmatrix} 0.74 \\ -0.82 \end{vmatrix} = \begin{vmatrix} 0.3766612 \\ -0.338848 \end{vmatrix} \quad (9.4)$$

9.3.3 MATLAB Code

```
% CE418: Neuro-fuzzy Computing
%
%    Evangelos Stamos
%    02338
%    estamos@e-ce.uth.gr

% Problem-09 | C
%
% Sketch the contour plot of the mean square error performance index
%

clear
[W1,W2] = meshgrid(-3 : .1 : 3);
F = 1 - 1.48*W1 + 1.64*W2 - 0.824*W1*W2 + 1.594*W1.^2 + + 1.962*W2.^2;
contour(W1,W2,F)
title('Contour plot of Mean Square Error');
figure;
surf(W1,W2,F)
title('Surface plot of Mean Square Error');
figure;
```

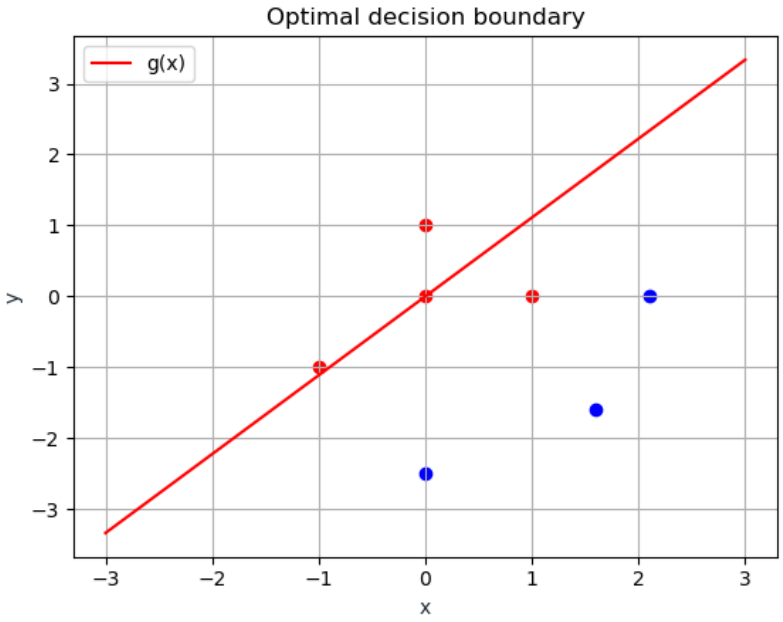
```
21 surfc(W1,W2,F)
22 title('Surface and Contour plot of Mean Square Error');
```

9.4 D. Show the optimal decision boundary

We have already computed w_{minMSE}, and we know that our ADALINE network has no bias so b = 0 . We need to solve :

$$w^T \cdot p = 0 \Longrightarrow 0.3766612p_1 - 0.338848p_2 = 0 \Longrightarrow p_1 = \frac{0.3766612}{0.338848}p_2 \tag{9.5}$$

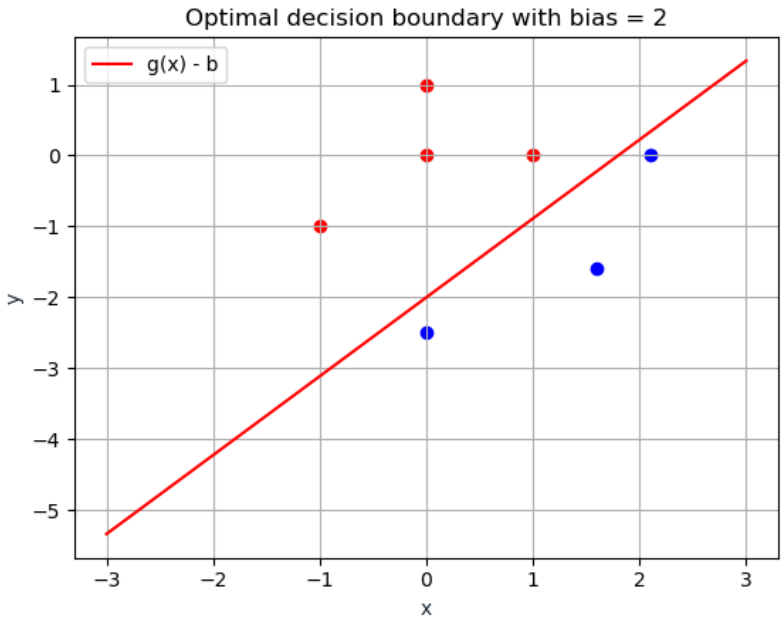
As it is clearly from image 9.4 , the optimal decision boundary (for the weights that minimize mean square error) does not separate the patterns into the appropriate categories, as it misclassifies p_3 into category B . That happens cause we have an ADALINE network with no bias trained on these patterns, with the addition of a bias eg *bias* = 2 as we see at image 9.5 it separates all patterns into the appropriate categories .



Εικόνα 9.4: *Optimal decision boundary*

9.4.1 Python Code

```
1 # CE418: Neuro-fuzzy Computing
2 #
3 #    Evangelos Stamos
4 #    02338
5 #    estamos@e-ce.uth.gr
6
7 # Problem-09 | D
8 #
```



Εικόνα 9.5: *Optimal decision boundary biased*

```
9  # suppose : linearly seperable classes, M = 2, labeled input training
       set data
10 #
11 # Show the optimal decision boundary
12 #
13 # weights that minimize mean square error :
14 #
15 # w1 = x = 0.3766612
16 # w2 = y =   0  .338848
17 #
18 # verify that it separates the patterns into the appropriate categories.
19 # It not separates the patterns into the appropriate categories | needed
       bias
20
21 # defines libraries
22 import numpy as np
23
24 import matplotlib.pyplot as plt
25
26 # category A
27 plt.scatter(0, 0,color='red')
28 plt.scatter(0, 1, color='red')
29 plt.scatter(1, 0, color='red')
30 plt.scatter(-1, -1, color='red')
```

```
31
32 # category B
33 plt.scatter(2.1, 0, color='blue')
34 plt.scatter(0, -2.5, color='blue')
35 plt.scatter(1.6, -1.6, color='blue')
36
37 x = np.linspace(-3,3,10)
38
39 y = 0.3766612*x/0.338848
40
41 plt.plot(x, y, '-r', label='g(x)')
42 plt.title('Optimal decision boundary')
43 plt.xlabel('x', color='#1C2833')
44 plt.ylabel('y', color='#1C2833')
45 plt.legend(loc='upper left')
46 plt.grid()
47 plt.show()
```

9.5 E. Find the maximum stable learning rate for the LMS algorithm

In order to find the maximum stable learning rate for the LMS algorithm, we need to compute the eigenvalues of R. For $R = \begin{vmatrix} 1.594 & -0.412 \\ -0.412 & 1.962 \end{vmatrix}$, eigenvalues are $\lambda_1 = 2.6536$ and $\lambda_2 = 4.4584$. So $\lambda_{max} = 4.4584$

The maximum stable learning rate should satisfy :

$$a < \frac{2}{\lambda_{max}} = \frac{2}{4.4584} = 0.448591423 \tag{9.6}$$

So maximum stable learning rate is $a_{max} = 0.448591423$.

9.5.1 MATLAB Code

```
1 R=2*[1.594 -0.412;-0.412 1.962];
2 eig (R)
```

9.5.2 Change the target values to opposite values, and see how this change affected the maximum stable learning rate

Changing the target values to opposite values as showing at table 9.2 will not affect the maximum stable learning rate, since R would not change . Although, c and R will not change, h will change to

$$h_{new} = \mathrm{E}[tz] = 0.1 \cdot 1 \cdot \begin{vmatrix} 0 \\ 0 \end{vmatrix} + 0.1 \cdot 1 \cdot \begin{vmatrix} 0 \\ 1 \end{vmatrix} + 0.1 \cdot 1 \cdot \begin{vmatrix} 1 \\ 0 \end{vmatrix} + 0.1 \cdot 1 \cdot \begin{vmatrix} -1 \\ -1 \end{vmatrix} + 0.2 \cdot (-1) \cdot \begin{vmatrix} 2.1 \\ 0 \end{vmatrix} + 0.2 \cdot$$

Πίνακας 9.2: *Opposite target (category) values*

reference pattern	new target value	category
p_1	1	A
p_2	1	A
p_3	1	A
p_4	1	A
p_5	- 1	B
p_6	- 1	B
p_7	- 1	B

$$(-1)\cdot\begin{vmatrix}0\\-2.5\end{vmatrix}+0.2\cdot(-1)\cdot\begin{vmatrix}1.6\\-1.6\end{vmatrix}=\begin{vmatrix}-0.74\\0.82\end{vmatrix}$$

9.6 F. One iteration of the LMS algorithm

Starting with all weights equal to zero, and presenting input vector p_1. Using a learning rate of = 0.5

$$output = W\cdot p = 0\cdot p = 0 \tag{9.7}$$

$$error = target - output = -1 - 0 = -1 \tag{9.8}$$

LMS algorithm

$$w_{k+1} = w_k + 2a\cdot e_k\cdot p_k^T = \begin{vmatrix}0\\0\end{vmatrix} + 2\cdot 0.5\cdot -1\cdot\begin{vmatrix}0\\0\end{vmatrix} = \begin{vmatrix}0\\0\end{vmatrix} \tag{9.9}$$

Note that we have no bias .

$$b_{k+1} = b_k + 2a\cdot e_k = 0 + 2\cdot 0.5\cdot -1 = -1 \tag{9.10}$$

If we set bias = 2 :

$$output = W\cdot p + b = 0\cdot p + 2 = 2 \tag{9.11}$$

$$error = target - output = -1 - 2 = -3 \tag{9.12}$$

LMS algorithm

$$w_{k+1} = w_k + 2a\cdot e_k\cdot p_k^T = \begin{vmatrix}0\\0\end{vmatrix} + 2\cdot 0.5\cdot -3\cdot\begin{vmatrix}0\\0\end{vmatrix} = \begin{vmatrix}0\\0\end{vmatrix} \tag{9.13}$$

Note that we bias = 2 .

$$b_{k+1} = b_k + 2a\cdot e_k = 2 + 2\cdot 0.5\cdot -3 = -1 \tag{9.14}$$