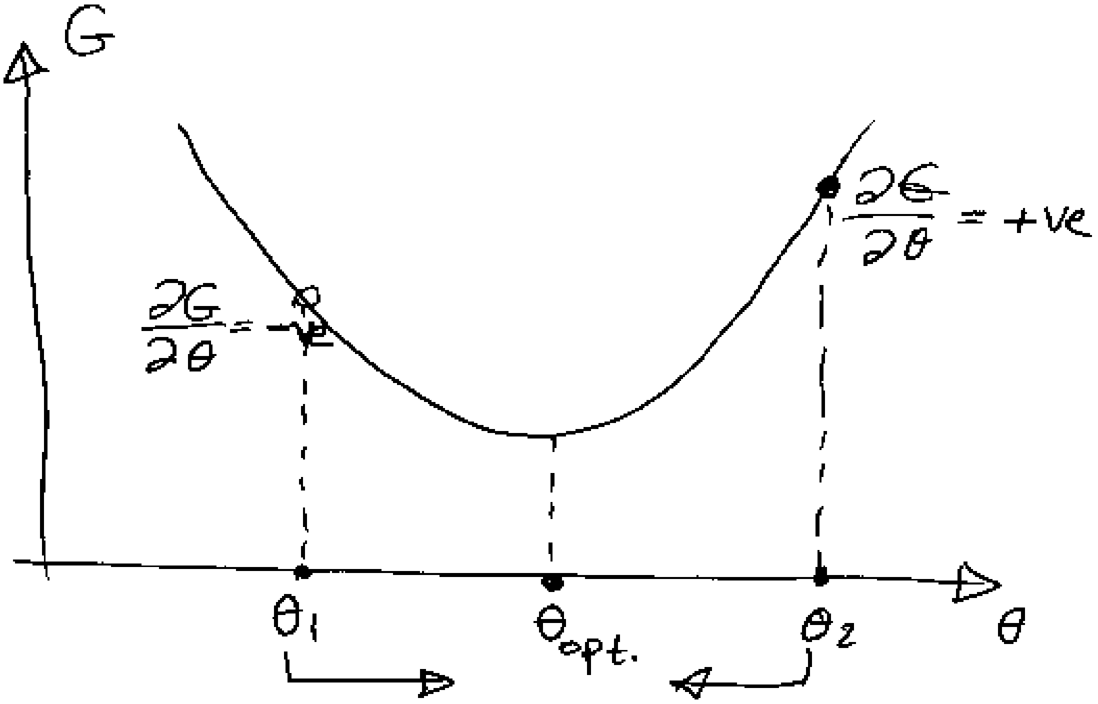
(*) The Gradient learning

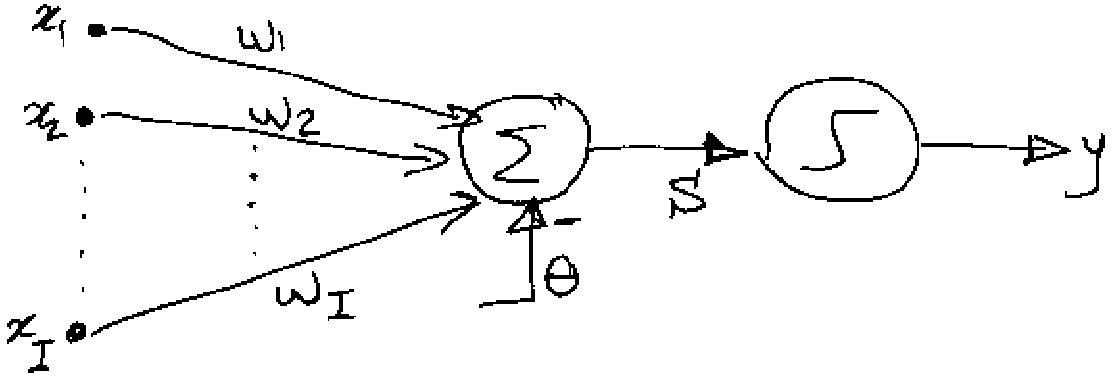
$$\theta_{(t+1)} = \theta_{(t)} - \eta * \frac{\partial G}{\partial \theta}(t)$$



η and $\left|\frac{\partial G}{\partial \theta}\right|$ control the step

learning rate

(*) The Linear perceptron



$$s = \left(\sum_{i=1}^{I} x_i * w_i - \theta \right)$$

$$y = f(s)$$

non-linear activation function (Biologically inspired)

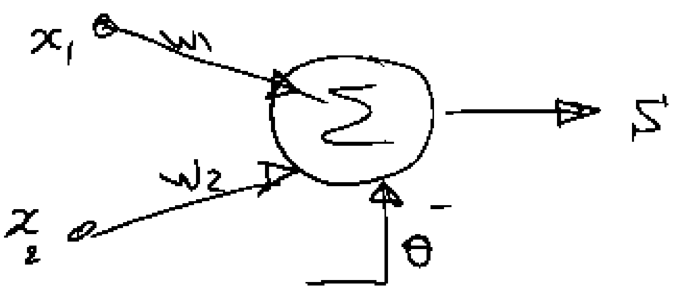
θ = Threshold

(*)

Neuron fires if $s' > \theta$
($y \Rightarrow 1$) (high output)

Neuron gives low output ($y \Rightarrow 0$) if $s' < \theta$

* perceptron capabilities :-



$$S = (x_1 w_1 + x_2 w_2) - \theta$$

Decision boundary at $S = 0$

since $S > 0$ class ① $(y = 1)$

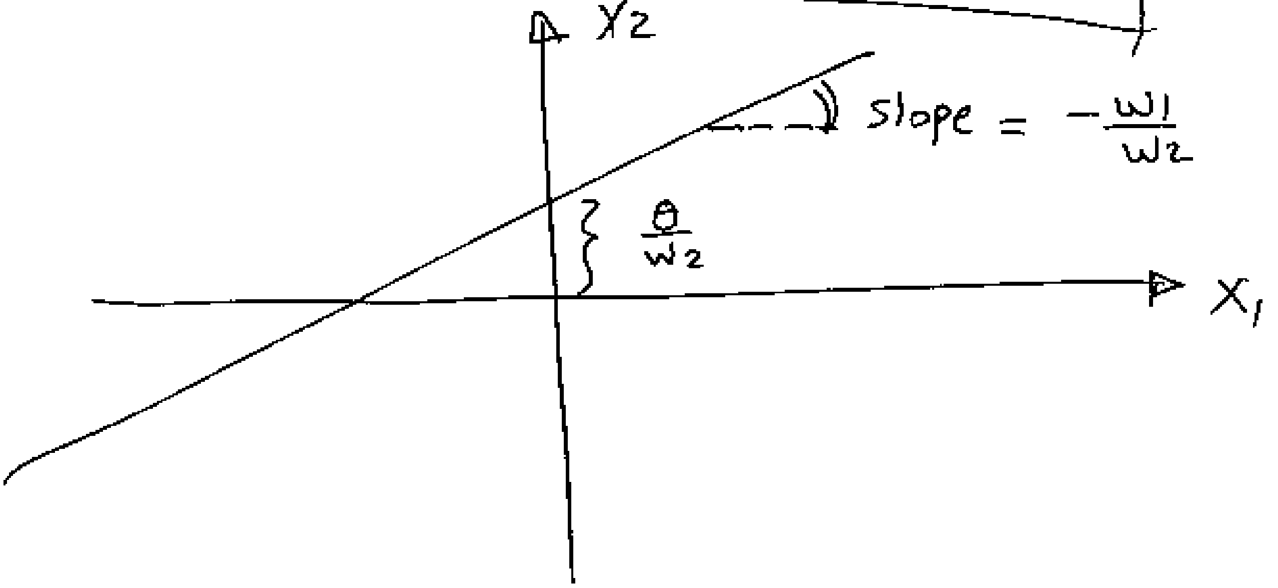
$S < 0$ class ② $(y = 0)$

$\therefore$

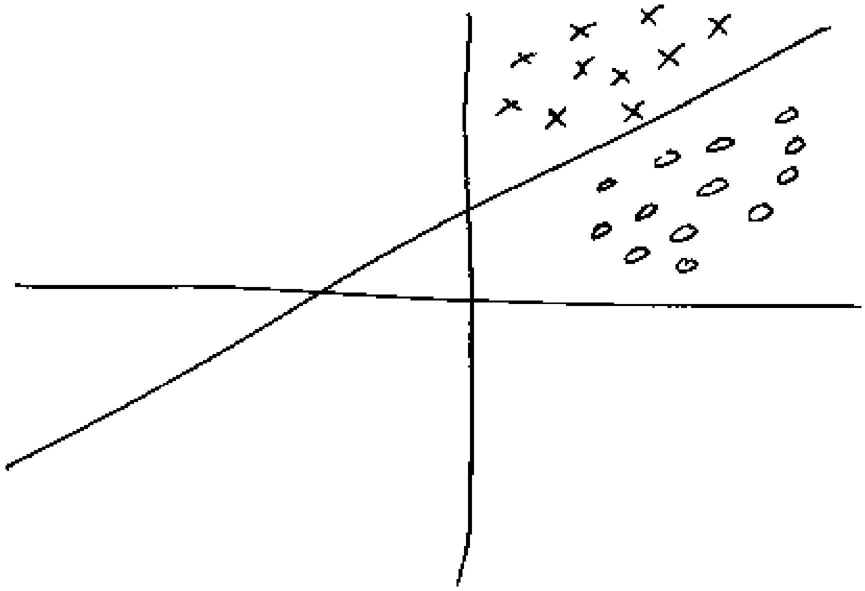
$$x_1 w_1 + x_2 w_2 = \theta$$

or

$$\boxed{x_2 = -\frac{w_1}{w_2} x_1 + \left(\frac{\theta}{w_2}\right)}$$



(*) During learning (w_1 and w_2) (and also θ) may change to adjust the decision boundary to separate the 2-classes.



It works well for linearly-separable classes only.

(*) perceptron is a linear machine, it can only produce linear decision boundary between 2-classes.

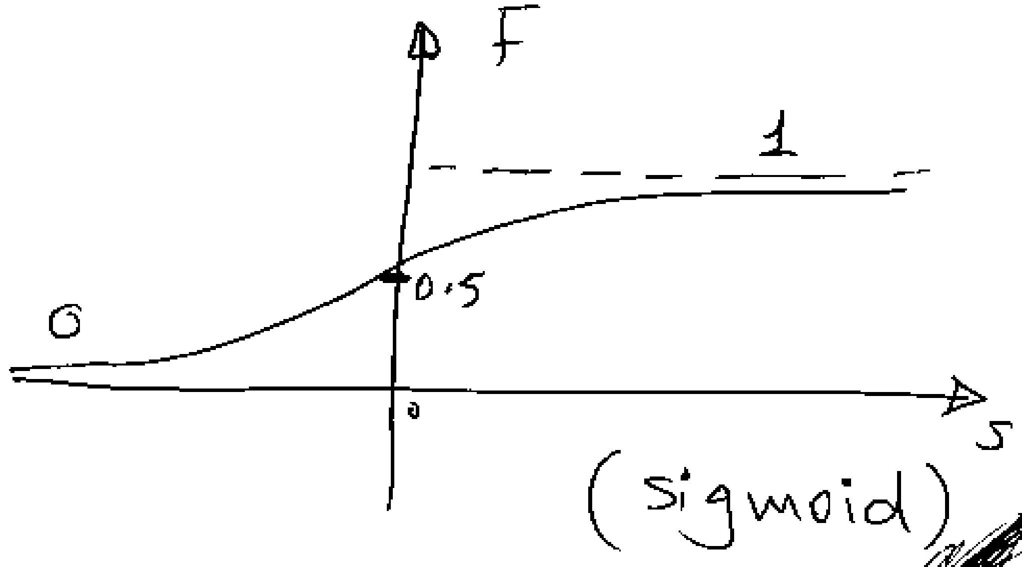
* perceptron learning:
(The Gradient approach):

$$G = MSE = \frac{1}{N_1} \sum_{\substack{\text{class 1} \\ \text{data}}} (y-1)^2 + \frac{1}{N_2} \sum_{\substack{\text{class 2} \\ \text{data}}} (y-0)^2$$

$$y = F\left(\sum_{i=0}^{I} x_i w_i \right)$$

where $i=0$ corresponds to $\theta = w_0$
$x_0 = -1$

In order to apply gradient methods, F has to be smooth, differentiable



(sigmoid)

$$y = \frac{1}{1 + e^{-s}}$$

For C_1 patterns,

$$\frac{\partial G}{\partial w_j} = 2(y-1) * F' * x_j$$

$$F' = y(1-y)$$

$$\therefore \quad w_{j(t+1)} = w_{j(t)} + \eta * y(1-y)^2 * x_{j(t)}$$

For C_2 patterns,

$$\frac{\partial G}{\partial w_j} = 2y * F' * x_j$$

$$w_j(t+1) = w_j(t) - \eta * y^2(1-y) * x_j(t)$$

$$0 < \eta < \frac{1}{\sum x_i^2}$$

to avoid oscillations

Some Approximations :

(Perceptron learning Rule)

Class (1) data :-

If pattern classified correctly (Do Nothing)
$(y \geq \frac{1}{2})$

If Not :

$$w_j(t+1) = w_j(t) + \eta * x_j(t)$$

Class (2) data :

If pattern classified correctly
$(y < \frac{1}{2})$ (Do Nothing)

If Not :

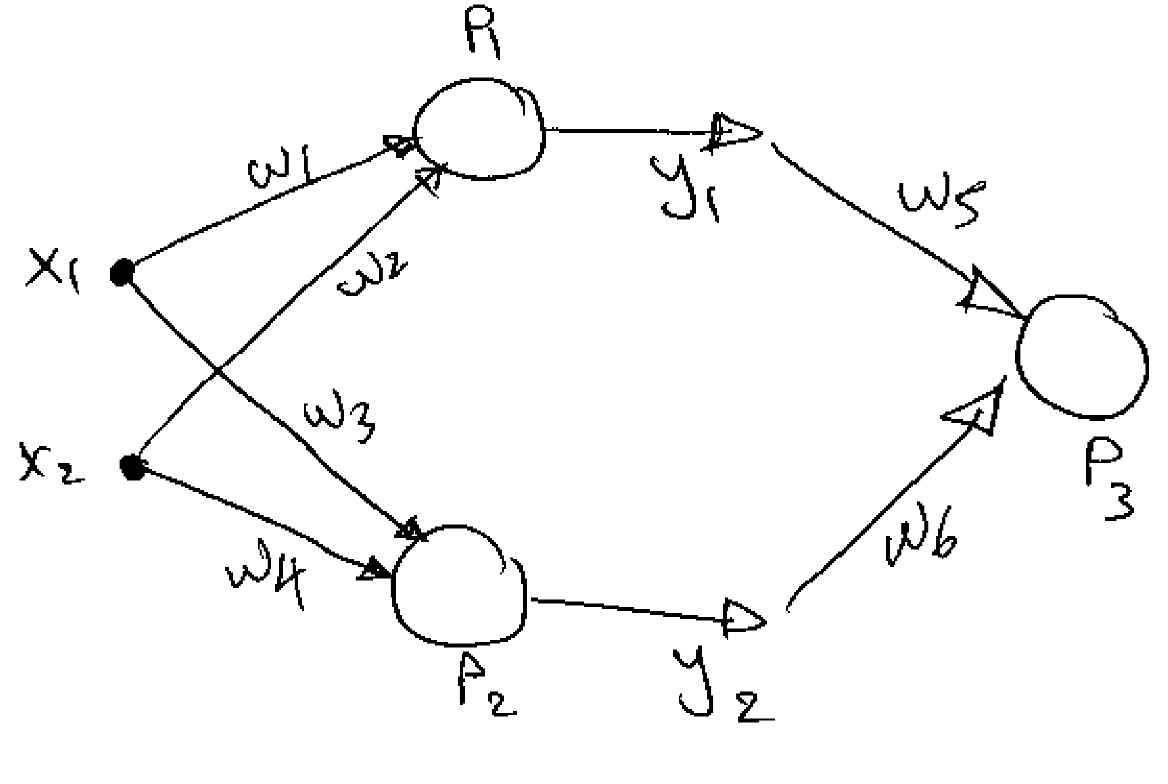
$$w_j(t+1) = w_j(t) - \eta * x_j(t)$$

Perceptron problem :-

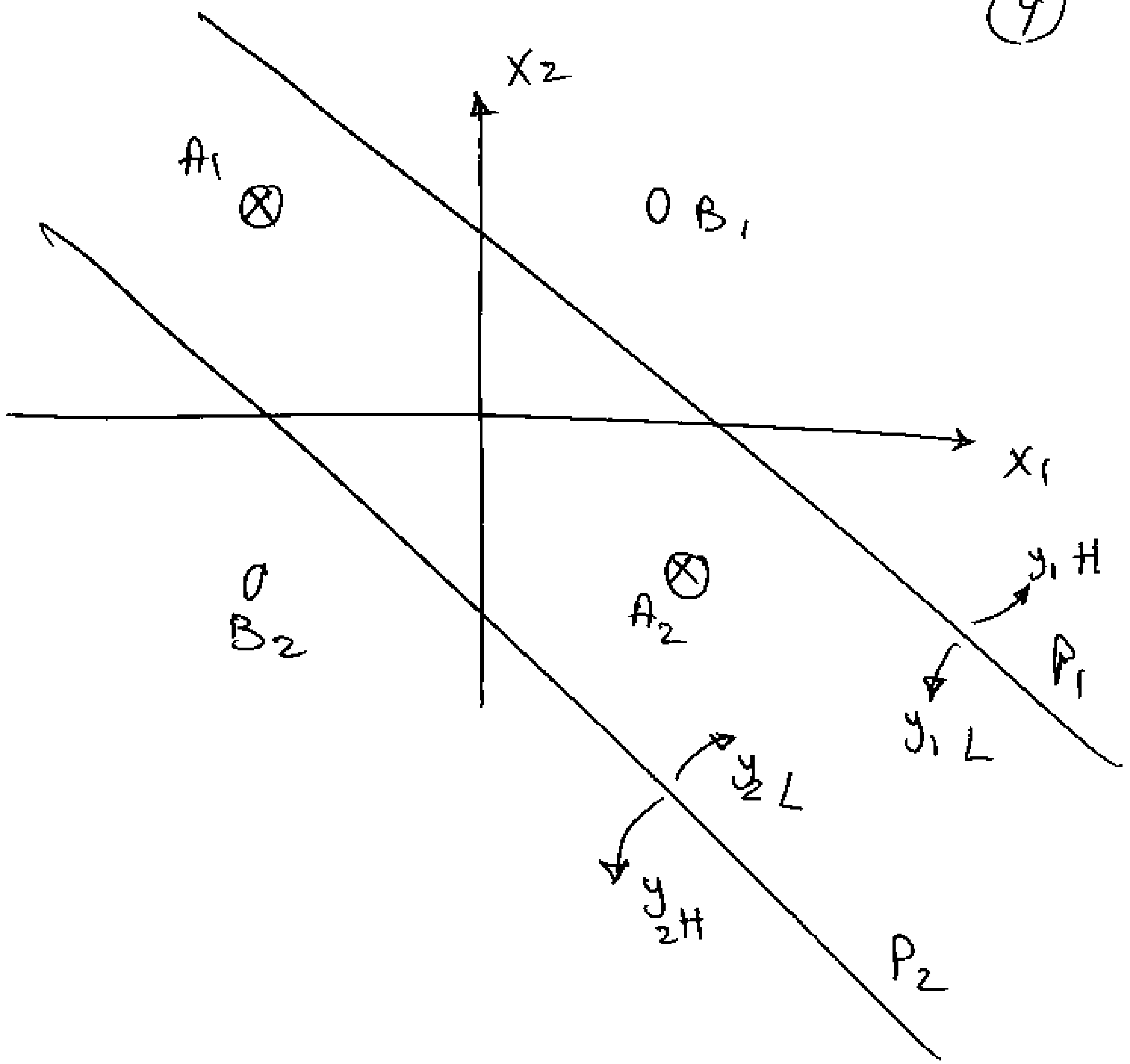
(*) Can only solve linear problems,

(*) No generalization to more than 2-classes

(*) The Many perceptron Idea



(*) P_1 and P_2 give linear decision boundaries :-
(XOR) problem



	y_1	y_2
A_1	L	L
A_2	L	L
B_1	H	L
B_2	L	H

we can have

$$y_3 = 0.7\, y_1 + 0.7\, y_2$$

<u>exercise</u>: Find $(w_1, w_2, w_3, w_4, \theta_1, \theta_2)$ if points are at $(\pm \pm \mp = 1)$

(*) Matlab (newp)
(please see files in)
(Matlab_Neural)

(*) The Many-perceptron idea
cannot be automated for larger
problems, we need something
more theoretically founded ??

↓

The Most famous (NN) ever,
The Multi-layer perceptron
(MLP) (1986-1987)

↓ Rumelhart, Hinton,
etc...

(PDP book)