energy function

$$E = -\frac{1}{2} \sum_j \sum_k w_{kj} x_k x_j$$

$$P(x_k \rightarrow -x_k) = \frac{1}{1 + \exp(-\Delta E_k / T)}$$

In clamped condition in which the visible neuron are clamp onto the specific state in which determine by environment.

In free-running condn in which all neuron are allowed to operate freely

$$\Delta w_{kj} = \eta \left(\rho_{kj}^{+} - \rho_{kj}^{-} \right), \; j \neq k.$$

ρ_{kj}^{+} → denotes co-relation b/w neuron k & j with the N/w in its clamped condn

ρ_{kj}^{-} → co-relation b/w neuron k & j with the N/w in its free running condn

2/4/14

* Credit-assignment Problem :-

1. Temporal credit assignment Prob.
2. Structural " " "

* Memory :-

1. short-term memory
2. long-term memory

1. short-term : a compilation knowledge representing the current state of environment is called short-term memory. Any discrepancies b/w knowledge stored in the short-term mem and ... the short-term memory ... state is used to update

Long time memory refered to knowledge stored

... memory :-

... following ch...

2. The memory is distributed.

[illegible]

$$= \sum_{k=1}^{m} (\vec{x}_k^{T} \vec{x}_j) \vec{y}_k$$

$$= (\vec{x}_k^{T} \cdot \vec{x}_j) \vec{y}_j + \sum_{\substack{k=1 \\ k \neq j}}^{m} (\vec{x}_k^{T} \vec{x}_j) \vec{y}_k$$

$$= \vec{y}_j + \vec{v}_j$$

$v_j \rightarrow$ Noise vector.

$$v_j = \sum_{\substack{k=1 \\ k \neq j}}^{m} (\vec{x}_k^{T} \vec{x}_j) \vec{y}_k = \sum_{\substack{k=1 \\ k \neq j}} \cos(\vec{x}_k, \vec{x}_j) \vec{y}_k$$

$$\cos(\vec{x}_k, \vec{x}_j) = \frac{\vec{x}_k^{T} \vec{x}_j}{\| \vec{x}_k \| \, \| \vec{x}_j \|}$$

... for perfect recall

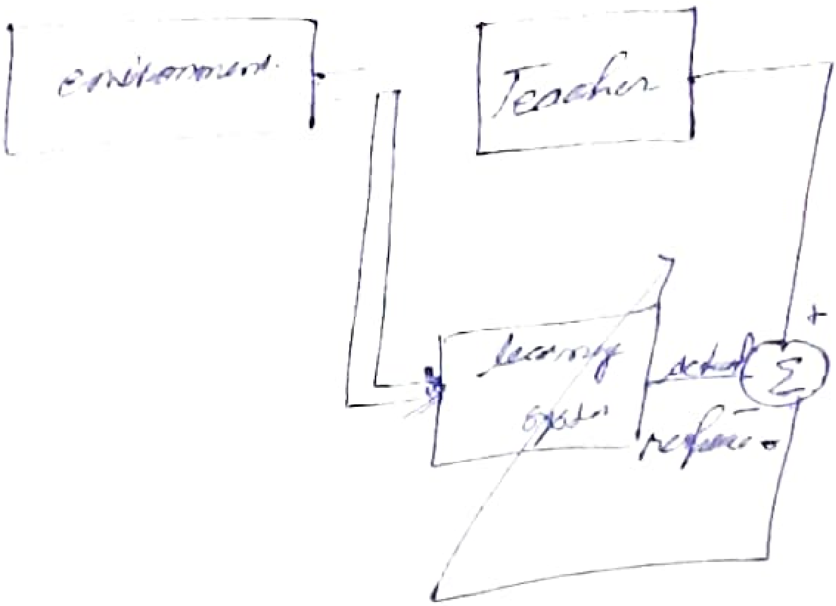
$$\vec{x}_k^{T} \vec{x}_j = \begin{cases} 1 & k = j \\ 0 & k \neq j \end{cases}$$

[illegible] of storage capacity of memory

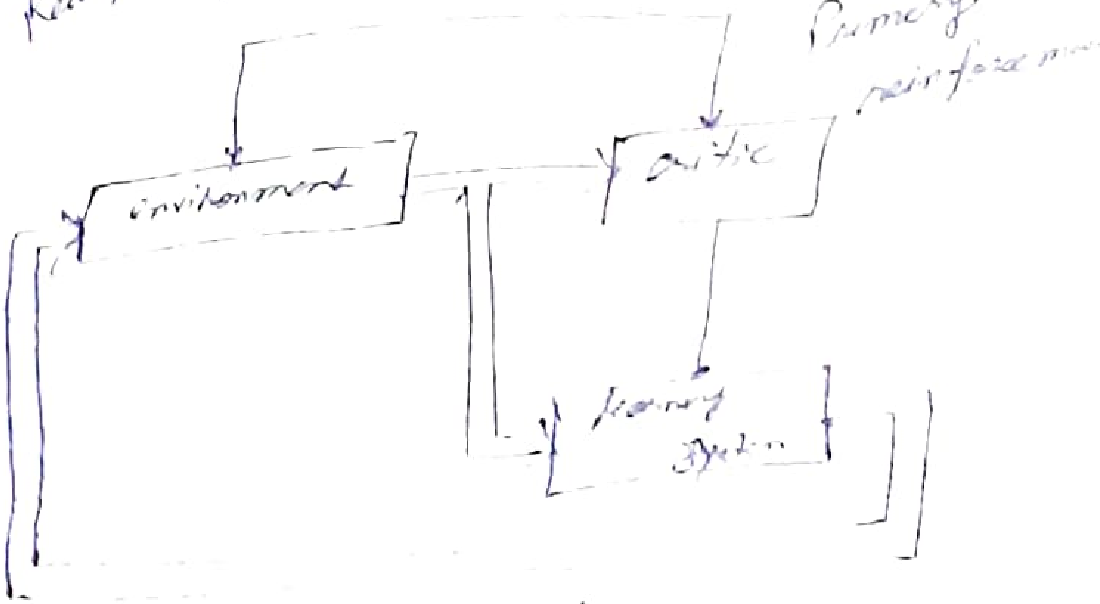
[illegible] rectangular memory matrix $L \times m$

[illegible] m

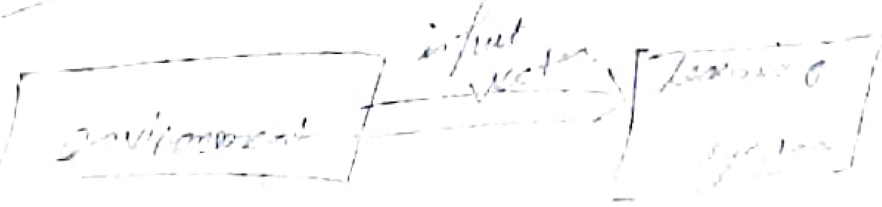
* Learning with teacher (supervised learning)



Reinforcement learning

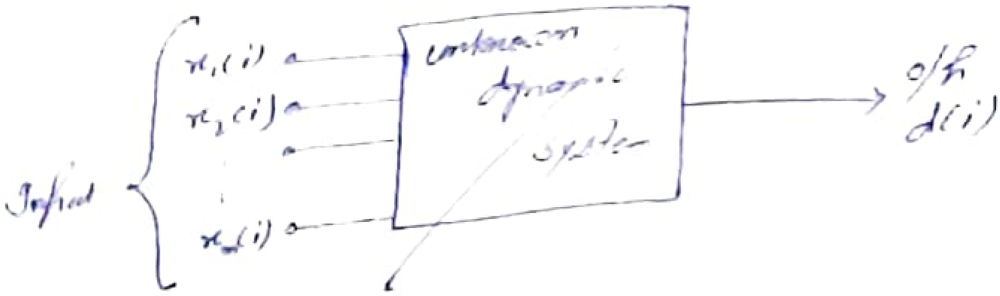


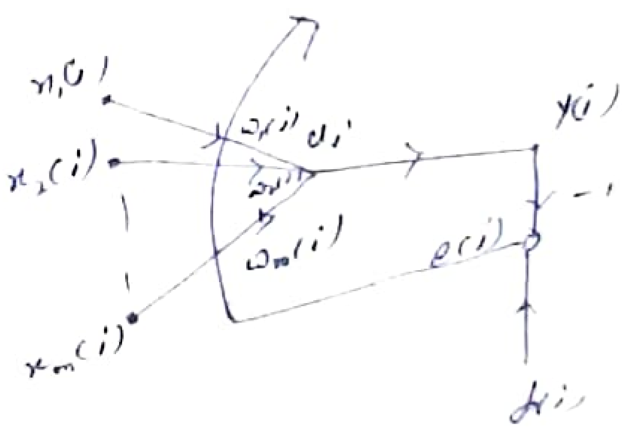
unsupervised



27/1/14

Single Layer Perceptron (SLP)





Signal flow representation

Data!

$$\mathcal{J} : \{\vec{x}(i),\ d(i) : i = 1, 2, 3, \dots m\}$$

$$\vec{x}(i) = [x_1(i)\ \ x_2(i)\ \ \dots\ \ x_m(i)]^T$$

neural model operates under the influence of ~~coeffi~~ algo that controls the adjustment to the coeffi. wt. to the statistical variation in of the N/W. with the following point:

1) The algo^m start from arbitrary setting of neuron synaptic wt.

2) The adjustment to the synaptic wt. in response to statistical variation in system var are made on continuous basis

3) ~~Continue~~ — — wt. are completed inside a time — — that is one sample period long. / ~~neural model~~

neural mode described in fig is call adaptive filter. Its operation consists of two continuous process which involves computation of two signal: and o/p denoted by y(i) that is produced in response to the m element to the stimulus vector x(i) an error signal denoted by e(i) ② adaptive process: which involve adjustment of adaptive wt. of neuron in accordance to adaptive error signals.

$$y(i) = v(i) = \sum_{k=1}^{m} w_k(i)\, x_k(i)$$

$$y(i) = \vec{x}^T(i)\, \vec{w}(i)$$

$$e(i) = d(i) - y(i)$$

- unconstrained optimization technique to find $\vec{w}^*$.

Cost function:

$$\xi(\vec{w})$$

$$\xi(\vec{w}^*) \leq \xi(\vec{w})$$

$\downarrow$

optimized

To minimize the cost function of $\xi(\vec{w})$ with respect to $\vec{w}$. The necessary ~~condition optim~~ optimization cond^n is $\nabla \xi(\vec{w}^*) = 0$

grad.

$$\nabla = \left\{ \frac{\partial}{\partial w_1}, \frac{\partial}{\partial w_2}, \ldots \frac{\partial}{\partial w_m} \right\}$$

$$\xi(\vec{\omega}(n+1)) < \xi(\vec{\omega}(n))$$

$\vec{\omega}(n) \rightarrow$ old value of wt. vect

$\vec{\omega}(n+1) \rightarrow$ updated val of wt. vect

Methods of Steepest Descent :-

$$\vec{g} = \nabla \xi(\vec{\omega})$$

in this method the successive adjustment applied to the wt vector $\vec{\omega}$ are in the dir^n of steepest descent, that is in a direction opposite to the gradient vector $(\vec{g})$.

$$\vec{\omega}(n+1) = \vec{\omega}(n) - \underbrace{\eta}_{\text{true constant value called step size or learning rate parameter}} \vec{g}(n)$$

$$\Delta \vec{\omega}(n) = -\eta \vec{g}(n)$$

first order Taylor series approximation,

$$f(x_0 + h) \simeq f(x_0) + f'(x_0)h$$

$$\xi(\vec{w}(n+1)) \simeq \xi(\vec{w}(n)) + g^T(h)\,\Delta w(n)$$

$$\xi(\vec{w}(n+1)) = \xi(\vec{w}(n)) - \eta\, g^T(n)\, g(n)$$

$$= \xi(\vec{w}(n)) - \eta \,\|\vec{g}(n)\|^2$$

13/1/14

- Newton's method

  - basic idea of this method is to minimize the quadratic ~~function~~ approx of the cost function around $x(n)$.

$$\mathcal{E}(x(n+1)) = \mathcal{E}(x(n)) + \Delta\mathcal{E}(x(n))$$

$$\Delta\mathcal{E}(x(n)) = \mathcal{E}(x(n+1)) - \mathcal{E}(x(n))$$

[illegible]

[illegible]

$$f(x_0+h) \cong f(x_0) + f'(x_0)h + f''(x_0)h^2$$

$$\Delta \xi(w(n)) \cong g'(n)\,\Delta w(n) + \frac{1}{2}\Delta w^{T}(n)\, H(n)\, \Delta w(n)$$

$$H(n) = \begin{bmatrix} \frac{\partial^2 \xi(n)}{\partial w_1^2} & \frac{\partial^2 \xi(n)}{\partial w_1 \partial w_2} & \cdots & \frac{\partial^2 \xi(n)}{\partial w_1 \partial w_m} \\ \frac{\partial^2 \xi(n)}{\partial w_2 \partial w_1} & \frac{\partial^2 \xi(n)}{\partial w_2^2} & \cdots & \frac{\partial^2 \xi(n)}{\partial w_2 \partial w_m} \\ \vdots & & & \\ \frac{\partial^2 \xi(n)}{\partial w_m \partial w_1} & \frac{\partial^2 \xi(n)}{\partial w_m \partial w_2} & \cdots & \frac{\partial^2 \xi(n)}{\partial w_m^2} \end{bmatrix}$$

$$\cong g(n) + H(n)\,\Delta w(n) = 0$$

$$\Delta w(n) = -H^{-1}(n)\, g(n)$$

$$w(n+1) = w(n) - H^{-1} g(n)$$

- **Gauss Newton method:**

$$\xi(w) = \frac{1}{2}\sum_{i=1}^{n} e^2(i)$$

$$1 \le i \le n$$

$$e'(i,w) = e(i) + \left[\frac{\partial e(i)}{\partial w}\right]^T_{w=w(n)} \left[w(n+1) - w(n)\right]$$

$$e'(n,w) = e(n) + J(n)\left[w(n+1) - w(n)\right]$$

$$J(n) = \begin{bmatrix} \frac{\partial e_1}{\partial w_1} & \frac{\partial e_1}{\partial w_2} & \cdots & \frac{\partial e_1}{\partial w_m} \\ \frac{\partial e_2}{\partial w_1} & \frac{\partial e_2}{\partial w_2} & \cdots & \frac{\partial e_2}{\partial w_m} \\ \vdots & & & \\ \frac{\partial e_n}{\partial w_1} & \frac{\partial e_n}{\partial w_2} & \cdots & \frac{\partial e_n}{\partial w_m} \end{bmatrix}$$

Objective

$$w(n+1) = \arg\min_{w} \left\{\frac{1}{2}\|e'(n,w)\|^2\right\}$$

31/1/19

$$\frac{1}{2}\|e'(n,w)\|^2 = \frac{1}{2}\ e'(n,w)\ e'^T(n,w)$$

$$= \frac{1}{2}\left[\vec{e}(n) + \vec{J}(n)\left(w(n+1) - w(n)\right)\right]$$

$$\left[\vec{e}(n) + \vec{J}(n)\left(w(n+1) - w(n)\right)\right]^T$$

$$= \frac{1}{2}\ \|e(n)\|^2 + e^T(n)\,\vec{J}(n)\left(w(n+1) - w(n)\right)$$

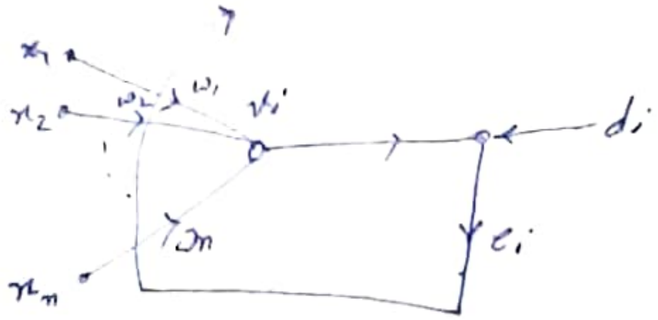
$$+ \frac{1}{2}\left(w(n+1) - w(n)\right)^T \vec{J}^T(n)\ J(n)\left(w(n+1) - w(n)\right)$$

Derivative w.r.t $\Delta w(n)$ and setting it to zero.

$$= J^T e(n) + J^T(n)\ J(n)\left(w(n+1) - w(n)\right)$$

$$w(n+1) = w(n) - \left(J^T(n)\,J(n)\right)^{-1} J^T(n)\ e(n)$$

* Linear least square filter :-



$$\xi(\vec{w}) = f(e(i))$$

$$e(n) = d(n) - \vec{x}(n)\,\vec{w}(n)$$

diff. w.r.t $w(n)$

$$\nabla e(n) = -x^T(n)$$

$$\vec{J}(n) = -x(n)$$

$$w(n+1) = w(n) + (x^T(n)\,x(n))^{-1}\,x^T(n)\,(d(n) - \vec{x}(n)\vec{w}(n))$$

$$= w(n) + (x^T(n)\,x(n))^{-1}\,x^T(n)\,d(n)$$

$$- (x^T(n)\,x(n))^{-1}\,(x^T(n)\,x(n))\,w(n)$$

$$= w(n) + (x^T(n)\,x(n))^{-1}\,x(n)\,d(n) - w(n)$$

$$\boxed{w(n+1) = (x^T(n)\,x(n))^{-1}\,x^T(n)\,d(n)}$$

$$w(n+1) = (X^T(n)\, X(n))^{-1}\, X^T(n)\, d(n)$$

$$X^{+}(n) = (X^T(n)\, X(n))^{-1}\, X^T(n)$$

~~Pseudo~~ Pseudo inverse of data matrix $X(n)$

$$w(n+1) = X^{+}(n)\, d(n)$$

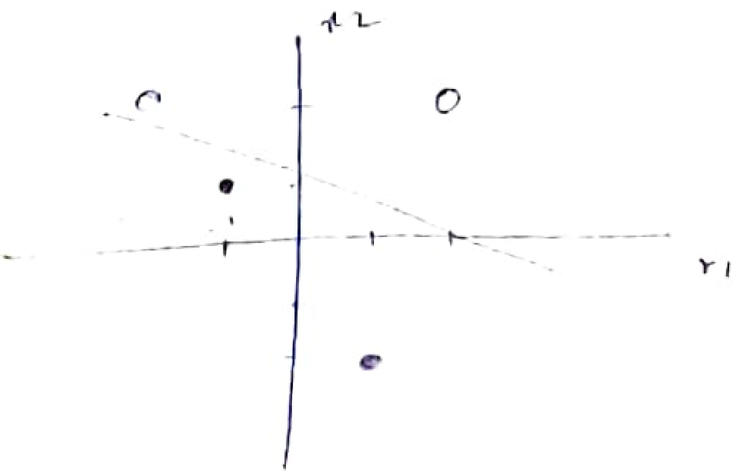
3/2/14

7/2/14

$$p = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\left\{ p_1 = \begin{bmatrix} 2 \\ 2 \end{bmatrix}, t_1 = 0 \right\}, \left\{ p_2 = \begin{bmatrix} 1 \\ -2 \end{bmatrix}, t_1 = 1 \right\}$$

$$\left\{ p_3 = \begin{bmatrix} -2 \\ 2 \end{bmatrix}, t_3 = 0 \right\} \quad \left\{ p_4 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}, t_4 = 1 \right\}$$



Derivation

$$\begin{cases} w = [0 \quad 0] \\ b = 0 \end{cases}$$

iteration - 1

$w_0 = [0 \quad 0]$, $b_0 = 0$

$$w_{n+1} = w_n + \underbrace{\eta}_{\text{assume } 1} e p$$

$$\boxed{\begin{aligned} w_{n+1} &= w_n + e p \\ b_{n+1} &= b_n + e \end{aligned}}$$

$e = t - a$

$=$ ↳ output value found by assumed weight.

$a = \text{hardlim}(w_0 P_1 + b)$

$= \text{hardlim}\left([0 \ 0]\begin{bmatrix}2\\1\end{bmatrix} + 0\right)$

$= \text{hardlim}(0) = 1$

$e = 0 - 1 = 0 - 1$

$b_1 = b_0 + e = -1$

$w_1 = [0 \ 0] - 1 \cdot [2 \ 1] = \ldots$