RECURRENT NEURAL NETWORKS
— HOPFIELD NETWORK.

Multilayer (feedforward) NN → relations are Algebraic.

$o = \Gamma(Wx)$

Rewrrent (Feedback) NN → relation are dynamic.

ODE → $\dot{x} = f(x)$

Difference → $x(k+1) = f(x(k))$

Initial condition → input to the system.

How can this be useful? ⇒ asymptotic behaviour.

$x(t) \to$
$x(k) \to$ a bounded set S ($t \to \infty$, $k \to \infty$)

$S \leftrightarrow$ an attractor (attracts solutions)

$S \to$ a point, a periodic solution, a set, chaotic attractors ...

→ information is stored in attractor

$S^1 \rightarrow$ a point $\{x^*\}$

$x(t) \rightarrow x^* \Rightarrow \dot{x} \rightarrow 0$, $\dot{x} = f(x)$

$\Rightarrow \boxed{f(x^*) = 0}$ (Equilibrium point of f. = DC operating point)

$x(k) \rightarrow x^* \Rightarrow x(k+1) \Rightarrow x^*$, $x(k+1) = f(x(k))$

$\Rightarrow \boxed{f(x^*) = x^*}$ Fixed point of f.

Example:

PROBLEM: $\min E(x)$

$\Rightarrow$ Gradient descent (back propagation)

$$\boxed{x(k+1) = x(k) - h \frac{dE(x)}{dx}.}$$

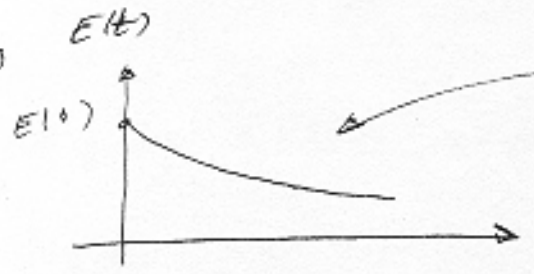
$$\Rightarrow \frac{x(k+1) - x(k)}{h} = -\frac{dE}{dx} \overset{\sim}{\Rightarrow} \boxed{\dot{x} = -\frac{dE}{dx}}$$

Claim: solution $x(t) \rightarrow S$ where $E(x)$ is locally minimum

(Gradient network)

$$\frac{dE}{dt} = \frac{dE}{dx} \cdot \frac{dx}{dt} = \frac{dE}{dx} \cdot \left(-\frac{dE}{dx}\right) = -\left(\frac{dE}{dx}\right)^2 \leq 0$$

$\Rightarrow$



$\Rightarrow$ Along the solutions, $E(t)$ decreases

$\Rightarrow$ under certain conditions

$x(t) \rightarrow S$ where $E(x)$ is locally minimum.

Example:

$$E(x) = \frac{1}{2}(x - x^*)^2 \Rightarrow \min E(x) = 0$$

$$\arg\min E(x) = x^*.$$

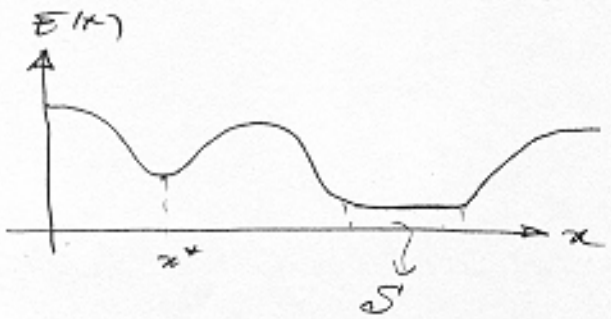
$$\frac{dE}{dx} = (x - x^*) \Rightarrow \boxed{\dot{x} = -(x - x^*)}$$ Gradient eqn.

$$y = x - x^* \Rightarrow \dot{y} = \dot{x} = -y \Rightarrow y(t) = e^{-t} y(0) \to 0$$

$$\Rightarrow \boxed{x(t) \to x^*}$$

Equilibrium pt. $\Rightarrow$ $f(x) = -(x - x^*)$ $\boxed{f(x^*) = 0}$

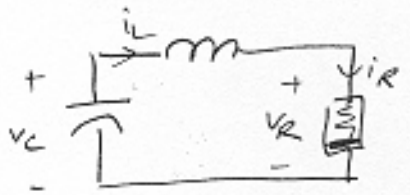
In general:

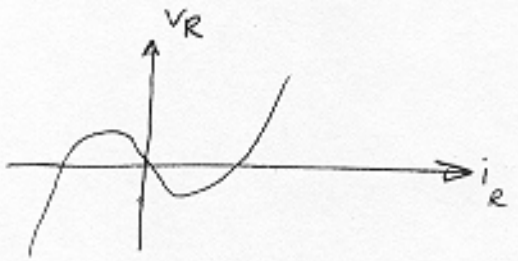


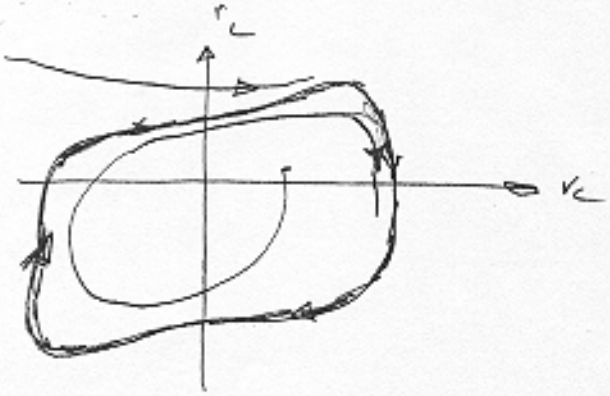
$$\boxed{\dot{x} = -\frac{dE}{dx}}$$

$x(t) \to S$ where $\frac{dE}{dx} = 0$.

Attractor can be a periodic solution: (oscillator)







IN DISCRETE TIME CASE

$$x(k+1) = x(k) - h \cdot \frac{dE}{dx} = x(k) - h\,(x(k) - a)$$

$$= (1-h)\,x(k) + ah$$

For convergence $1-h = \rho$ should satisfy $|\rho| < 1$!

Check convergence.

$y = x - a$

$$\Rightarrow \quad y(k+1) = x(k+1) - a = (1-h)\,x(k) + ah - a = (1-h)\,x(k) - (1-h)\,a$$

$$= (1-h)\,[x(k) - a] = (1-h)\,y(k) = \rho\, y(k)$$

$y(k) = \rho^k\, y(0) \to 0$ if $|\rho| < 1$

$x(k) \to 0$.

Check equilibrium.

$(1-h)\,x^* + ah = x^* \Rightarrow \quad -h\,x^* + ah = 0 \quad \to \quad \boxed{x^* = a}$

Stability of attractor: if $x(0)$ is sufficiently close to the attractor ($d(x(0), S) < \epsilon$), do the solution $x(t)$ ($x(k)$) $\rightarrow S$? If yes, S is stable.

If no $\Rightarrow$ not stable.

In stable case $\Rightarrow$ S has a basin of attraction

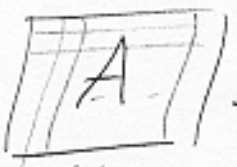
$B = \{ x(0) \mid x(t) \rightarrow S \}$. (All initial conditions such that $x(t) \rightarrow S$)

Pattern storage and recognition

idea: Design a network such that patterns become stable attractors

Recognition: $x(0) \neq S \Rightarrow$ view $x(0)$ as distorted pattern, stability $\Rightarrow$ if $x(0)$ is sufficiently close to S $x(t) \rightarrow S$.

Example.



pattern

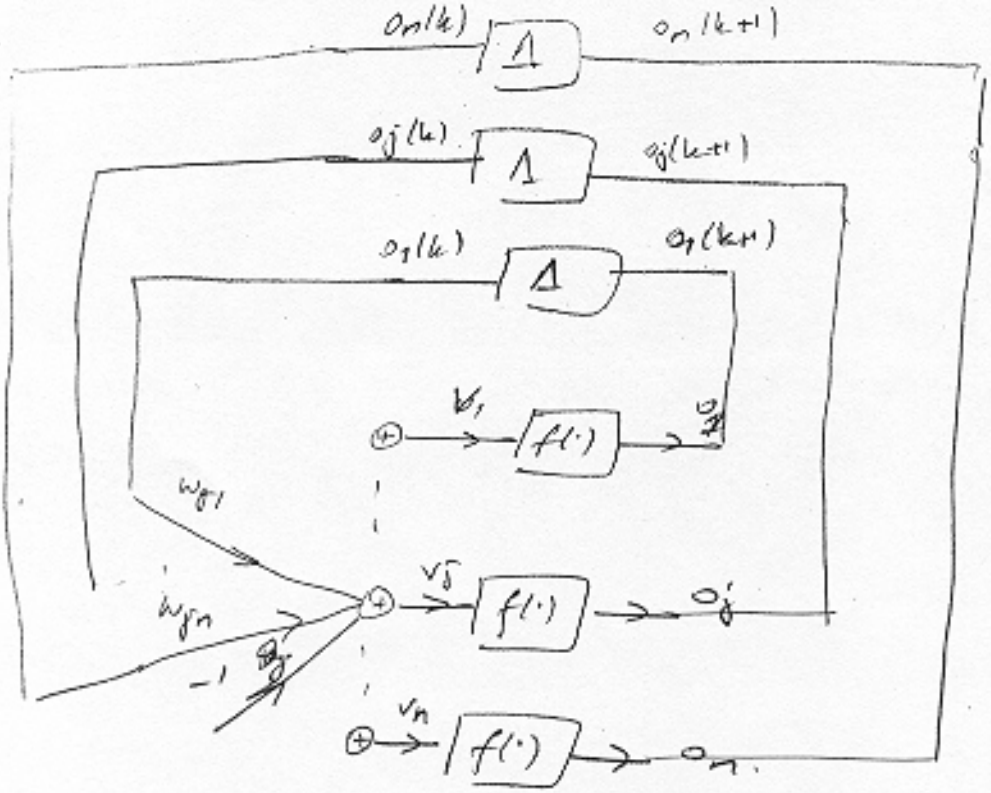
$\Rightarrow$ $x^* = \begin{pmatrix} 0 \\ 1 \\ \vdots \end{pmatrix}$

vector corresponding pattern

$\Rightarrow$ $x(k+1) = f(x(k))$ ($x^0 = f(x)$)

such that x^* is a stable equilibrium (fixed point)

DISCRETE TIME HOPFIELD NETWORK.



Componentwise:

$$v_j(k) = w_{j1}\, o_1(k) + w_{j2}\cdot o_2(k) + \cdots + w_{jn}\, o_n(k) - \theta_j$$

$$o_j(k+1) = f(v_j(k)) \qquad j = 1, 2, \ldots, n$$

$$k = 0, 1, 2, \ldots, \infty$$

Matrix form:

$$v(k) = W\, o(k) - \theta$$

$$o(k+1) = \Gamma(v(k))$$

$$v = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix}, \quad o = \begin{pmatrix} o_1 \\ o_2 \\ \vdots \\ o_n \end{pmatrix}, \quad W = \begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1n} \\ \vdots & & & \\ w_{j1} & w_{j2} & \cdots & w_{jn} \\ \vdots & & & \\ w_{n1} & w_{n2} & \cdots & w_{nn} \end{pmatrix}, \quad \theta = \begin{pmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{pmatrix}$$

for $f(\cdot) \Rightarrow$ sgn. fn.

$$f(v) = \begin{cases} +1 & v \geq 0 \\ -1 & v < 0 \end{cases}$$

1) In Hopfield's original work $\Rightarrow$ $W = W^T$ (symmetric). Not essential for stable patterns, but important in stability.

2) In Hopfield's original work $\Rightarrow$ $w_{ii} = 0$ (no self feedback in ith neuron). Important for asynchronous update stability.

Synchronous update: All neuron outputs are updated at the same time.

Asynchronous update: Only one neuron output is updated at a time. When this update takes place, most recent neuron outputs are used.

→ Suppose that we have $o_1(k), \ldots, o_n(k)$.

→ update o_1: $o_1(k+1) = \text{sgn}(w_{11} o_1(k) + w_{12} o_2(k) + \cdots + w_{1n} o_n(k) - \theta_1)$

→ update o_2: $o_2(k+1) = \text{sgn}(w_{21} o_1(k+1) + w_{22} o_2(k) + \cdots + w_{2n} o_n(k) - \theta_2)$

→ update o_3: $o_3(k+1) = \text{sgn}(w_{31} o_1(k+1) + w_{32} o_2(k+1) + \cdots + w_{3n} o_n(k) - \theta_3)$

⋮

→ update o_n: $o_n(k+1) = \text{sgn}(w_{n1} o_1(k+1) + w_{n2} o_2(k+1) + \cdots + w_{n,n-1} o_{n-1}(k+1) + w_{nn} o_n(k) - \theta_n)$

→ Go to the beginning of a new update

Example-

$$W = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix} \qquad \theta = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$o(0) = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

Synch. update. $\quad o(1) = \Gamma\left(W o(0) - \theta\right) = \Gamma\begin{pmatrix} -1 \\ 1 \end{pmatrix} = \underline{\underline{\begin{pmatrix} -1 \\ 1 \end{pmatrix}}}$

asynch. update. $\quad o_1(1) = \operatorname{sgn}\left(1 \cdot o_1(0) + 2\, o_2(0)\right) = -1$

$o_2(1) = \operatorname{sgn}\left(2 \cdot o_1(1) + 1\, o_2(0)\right) = -1$

ie. $\quad \begin{pmatrix} 1 \\ -1 \end{pmatrix} \rightarrow \begin{pmatrix} -1 \\ 1 \end{pmatrix}$ Synch update.

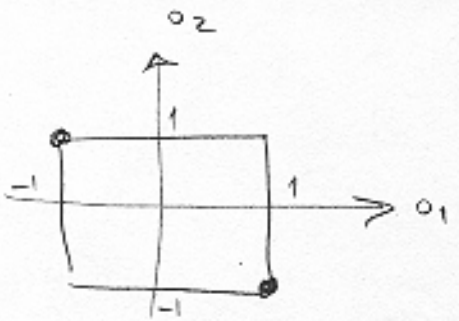
$$\begin{pmatrix} 1 \\ -1 \end{pmatrix} \rightarrow \begin{pmatrix} -1 \\ -1 \end{pmatrix} \rightarrow \begin{pmatrix} -1 \\ -1 \end{pmatrix}$$

first component update

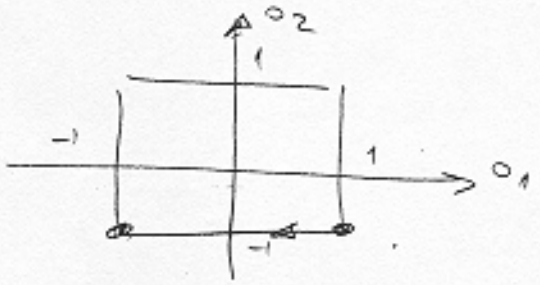
second component update

Synch. update: $o(k+1)$ and $o(k)$ may differ by more than 1 Hamming distance

Asynch. update. At any intermediate step, the iterations differ by one Hamming distance.



Synch.



asynch.

- Relation with Hopfield Network:

* $$\boxed{\begin{aligned} v(k) &= W o(k) - \theta \\ o(k+1) &= \Gamma(v(k)) \end{aligned}}$$

* computational Energy of Hopfield Network:

$$\boxed{E = -\frac{1}{2} o^T W o + \theta^T o}$$ quadratic form.

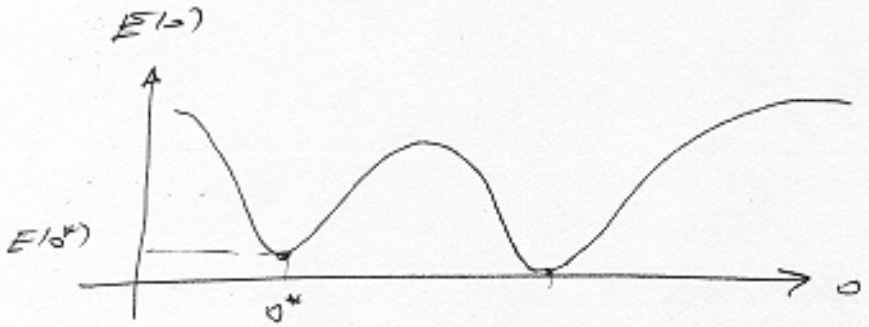
$$\boxed{E = -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ji} o_i o_j + \sum_{i=1}^{n} \theta_i o_i}$$

<u>claim:</u> Hopfield Network solves the following problem:

$$\arg\min_{o} E(o)$$

i.e. start with $o(0)$, iterate; then $o(k) \to o^*$

where $E(o^*)$ is a local minimum of $E(o)$.



Some Linear Algebra

1) Let $A = -A^T$ (skew symmetric matrix).

$\Rightarrow \quad x^T A x = 0, \quad \forall x.$

Proof: $(x^T A x)^T = x^T A^T x = -x^T A x$

$\Rightarrow \quad x^T A x = 0$

2) $A = \underbrace{\frac{1}{2}(A + A^T)}_{V} + \underbrace{\frac{1}{2}(A - A^T)}_{Q} = V + Q$

$V = V^T \quad \left((A + A^T)^T = A^T + A \right)$

$Q = -Q^T \quad \left((A - A^T)^T = A^T - A = -(A - A^T) \right)$

i.e. Any matrix is the sum of a symmetric and a skew symmetric matrices. (This decomposition is unique).

3) $x^T A x = x^T (V + Q) x = x^T V x \qquad (x^T Q x = 0).$

$\Rightarrow$ For quadratic forms, only the symmetric part of the matrix is important.

4) Gradient of a function:

$f(x): \mathbb{R}^n \to \mathbb{R}.$

$f(x + \Delta x) = f(x) + (\Delta x)^T \cdot \nabla f(x) + \text{h.o.t.}$

$\nabla f(x)$: Gradient $= \frac{\partial f}{\partial x} = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{pmatrix}.$