technische universität
dortmund

Computational Intelligence

Winter Term 2019/20

Prof. Dr. Günter Rudolph

Lehrstuhl für Algorithm Engineering (LS 11)

Fakultät für Informatik

TU Dortmund

Plan for Today

- Radial Basis Function Nets (RBF Nets)
    - Model
    - Training
- Hopfield Networks
    - Model
    - Optimization

Radial Basis Function Nets (RBF Nets)

Definition:

A function $\phi : \mathbb{R}^n \to \mathbb{R}$ is termed **radial basis function**

iff $\exists \varphi : \mathbb{R} \to \mathbb{R} : \forall x \in \mathbb{R}^n : \phi(x; c) = \varphi(\, \|x - c\| \,)$. □

Definition:

RBF **local** iff

φ(r) → 0 as r → ∞ □

typically, || x || denotes Euclidean norm of vector x

examples:

$$\varphi(r) = \exp\left(-\frac{r^2}{\sigma^2}\right)$$ Gaussian, unbounded

$$\varphi(r) = \frac{3}{4}(1 - r^2) \cdot 1_{\{r \le 1\}}$$ Epanechnikov, bounded

$$\varphi(r) = \frac{\pi}{4} \cos\left(\frac{\pi}{2} r\right) \cdot 1_{\{r \le 1\}}$$ Cosine, bounded

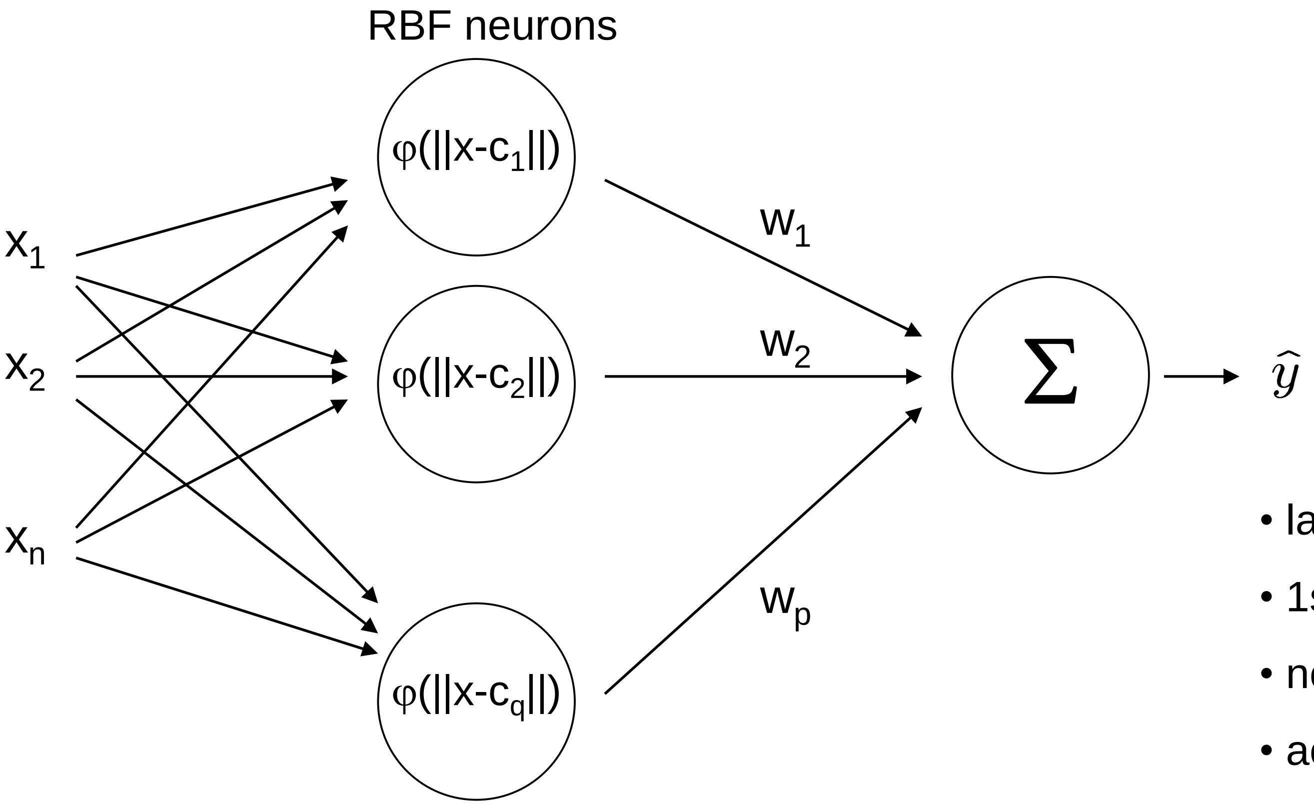
(Gaussian, Epanechnikov, Cosine: local)

Radial Basis Function Nets (RBF Nets)

Definition:

A function f: $\mathbb{R}^n \to \mathbb{R}$ is termed **radial basis function net (RBF net)**

iff $f(x) = w_1\, \varphi(\|\, x - c_1 \,\|) + w_2\, \varphi(\|\, x - c_2 \,\|) + \ldots + w_p\, \varphi(\|\, x - c_q \,\|)$ □



- layered net
- 1st layer fully connected
- no weights in 1st layer
- activation functions differ

Radial Basis Function Nets (RBF Nets)

given : N training patterns (x_i, y_i) and q RBF neurons

find : weights $w_1, \ldots, w_q$ with minimal error

solution:

we know that $f(x_i) = y_i$ for i = 1, ..., N and therefore we insist that

$$\sum_{k=1}^{q} \underset{\substack{\downarrow\\ \text{unknown}}}{w_k} \cdot \underbrace{\varphi(\|x_i - c_k\|)}_{\substack{p_{ik}\\ \text{known value}}} = \underset{\substack{\downarrow\\ \text{known value}}}{y_i}$$

$$\Rightarrow \quad \sum_{k=1}^{q} w_k \cdot p_{ik} = y_i$$

⇒ N linear equations with q unknowns

Radial Basis Function Nets (RBF Nets)

in matrix form: $P\,w = y$ with $P = (p_{ik})$ and P: $N \times q$, y: $N \times 1$, w: $q \times 1$,

case $N = q$: $w = P^{-1}\,y$ if P has full rank

case $N < q$: many solutions but of no practical relevance

case $N > q$: $w = P^{+}\,y$ where P^{+} is Moore-Penrose pseudo inverse

$P\,w = y$ | · P‘ from left hand side (P‘ is transpose of P)

$P'P\,w = P'\,y$ | · $(P'P)^{-1}$ from left hand side

$\underbrace{(P'P)^{-1}\,P'P}_{\text{unit matrix}}\,w = \underbrace{(P'P)^{-1}\,P'}_{P^{+}}\,y$ | simplify

- existence of $(P'P)^{-1}$?
- numerical stability ?

Tikhonov Regularization (1963)

idea:
choose $(P'P + h\, I_q)^{-1}$ instead of $(P'P)^{-1}$ ($h > 0$, I_q is q-dim. unit matrix)

excursion to linear algebra:

Def : matrix A positive semidefinite (p.s.d) iff $\forall x \in \mathbb{R}^n : x'Ax \geq 0$
Def : matrix A positive definite (p.d.) iff $\forall x \in \mathbb{R}^n \setminus \{0\} : x'Ax > 0$
Thm : matrix $A : n \times n$ regular $\Leftrightarrow$ rank$(A) = n \Leftrightarrow A^{-1}$ exists $\Leftarrow A$ is p.d.

Lemma : $a, b > 0,\ A, B : n \times n,\ A$ p.d. and B p.s.d. $\Rightarrow a \cdot A + b \cdot B$ p.d.

Proof : $\forall x \in \mathbb{R}^n \setminus \{0\} : x'(a \cdot A + b \cdot B)x = \underbrace{a}_{>0} \cdot \underbrace{x'Ax}_{>0} + \underbrace{b}_{>0} \cdot \underbrace{x'Bx}_{\geq 0} > 0$ q.e.d.

Lemma : $P : n \times q \Rightarrow P'P$ p.s.d.

Proof : $\forall x \in \mathbb{R}^n : x'(P'P)x = (x'P') \cdot (Px) = (Px)'(Px) = \|Px\|_2^2 \geq 0$ q.e.d.

Tikhonov Regularization (1963)

$\Rightarrow (P'P + h\,I_q)$ is p.d. $\Rightarrow (P'P + h\,I_q)^{-1}$ exists

question: how to justify this particular choice?

$$\|Pw - y\|^2 + h \cdot \|w\|^2 \quad \to \min_w!$$

interpretation: minimize TSSE and prefer solutions with small values!

avoid overfitting

$$\frac{d}{dw}\left[\,(Pw-y)'(Pw-y) + h \cdot w'w\,\right] =$$

$$\frac{d}{dw}\left[\,(w'P'Pw - w'P'y - y'Pw + y'y + h \cdot w'w\,\right] =$$

$$2\,P'Pw - 2\,P'y + 2h\,w = 2\,(P'P + h\,I_q)\,w - 2\,P'y \overset{!}{=} 0$$

$$\Rightarrow \boxed{w^* = (P'P + h\,I_q)^{-1}P'y}$$

$\frac{d}{dw}\left[2\,(P'P + h\,I_q)\,w - 2\,P'y\right] = 2\,(P'P + h\,I_q)$ is p.d. $\quad\Rightarrow$ minimum

Radial Basis Function Nets (RBF Nets)

Tikhonov Regularization (1963)

question: how to find appropriate $h > 0$ in $(P'P + h\,I_q)$?

let $\text{PERF}(h; T)$ with $\text{PERF}: \mathbb{R}^+ \to \mathbb{R}^+$ measure the performance of RBF net for positive h and given training set T

find h^* such that $\text{PERF}(h^*; T) = \max\{\text{PERF}(h; T) : h \in \mathbb{R}^+\}$

→ several approaches in use

→ here: **grid search** and **crossvalidation**

```
choose $n \in \mathbb{N}$ and $h_1, \ldots, h_n \in (0, H] \subset \mathbb{R}^+$; set $p^* = 0$
for $i = 1$ to $n$
    $p_i = \text{PERF}(h_i; T)$
    if $p_i > p^*$
        $p^* = p_i; k = i;$
    endif
endfor
return $h_k$
```

grid search

Crossvalidation

choose $k \in \mathbb{N}$ with $k < |T|$

let $T_1, \ldots, T_k$ be partition of training set T

$T_1 \cup \ldots \cup T_k = T$

$T_i \cap T_j = \emptyset$ for $i \neq j$

```
PERF$(h; T)$ =
(1) set $err = 0$
(2) for $i = 1$ to $k$
(3)     build matrix $P$ and vector $y$ from $T \setminus T_i$
(4)     get weights $w = (P'P + h\,I)^{-1} P'y$
(5)     build matrix $P$ and vector $y$ from $T_i$
(6)     get error $e = (Pw - y)'(Pw - y)$
(7)     $err = err + e$
(8) endfor
(9) return $1/err$
```

Radial Basis Function Nets (RBF Nets)

complexity (naive)

$w = (P‘P)^{-1} P‘ y$

$$\underbrace{P‘P: N^2 q \qquad \text{inversion: } q^3 \qquad P‘y: qN \qquad \text{multiplication: } q^2}_{\boxed{O(N^2 q) \text{ elementary operations}}}$$

remark: if N large then inaccuracies for P‘P likely

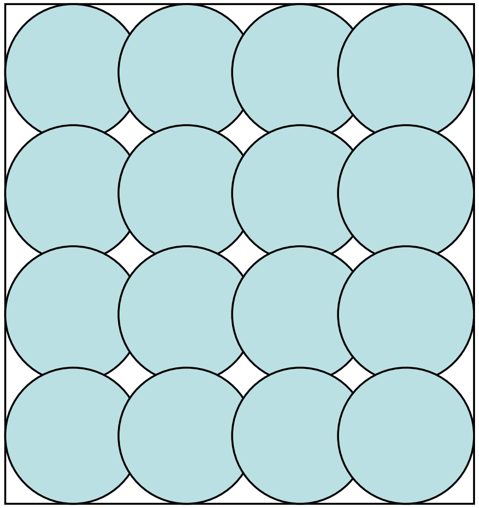
$\Rightarrow$ first analytic solution, then $\underbrace{\text{gradient descent}}_{\substack{\text{requires}\\ \text{differentiable}\\ \text{basis functions!}}}$ starting from this solution

Radial Basis Function Nets (RBF Nets)

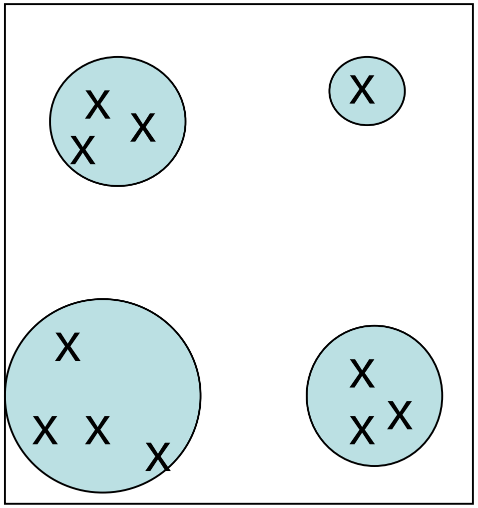
so far: tacitly assumed that RBF neurons are given

⇒ center c_k and radii σ considered given and known

how to choose c_k and σ ?



uniform covering



if training patterns inhomogenously distributed then first cluster analysis

choose center of basis function from each cluster, use cluster size for setting σ

Radial Basis Function Nets (RBF Nets)

advantages:

- additional training patterns → only local adjustment of weights
- optimal weights determinable in polynomial time
- regions not supported by RBF net can be identified by zero outputs

  (if output close to zero, verify that output of each basis function is close to zero)

disadvantages:

- number of neurons increases exponentially with input dimension
- unable to extrapolate (since there are no centers and RBFs are local)

Radial Basis Function Nets (RBF Nets)

Example: XOR via RBF

training data: (0,0), (1,1) with value –1
(0,1), (1,0) with value +1

$$\varphi(r) = \exp\left(-\frac{1}{\sigma^2}\, r^2\right)$$

choose Gaussian kernel; set σ = 1; set centers c_i to training points

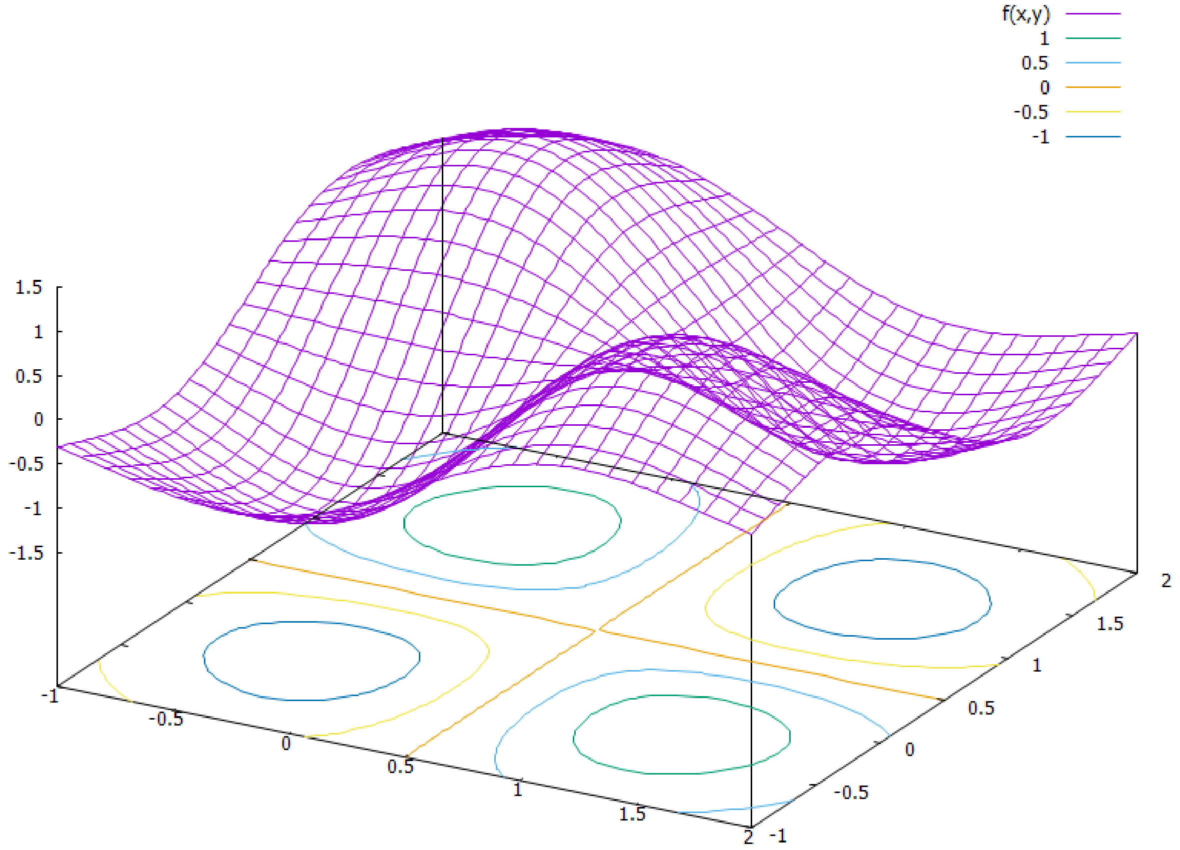
$$\hat{f}(x) = w_1\, \varphi(\|x - c_1\|) + w_2\, \varphi(\|x - c_2\|) + w_3\, \varphi(\|x - c_3\|) + w_4\, \varphi(\|x - c_4\|)$$

$$\begin{array}{lccccccccc}
\hat{f}(0,0) & = & w_1 & + & e^{-1}\cdot w_2 & + & e^{-1}\cdot w_3 & + & e^{-2}\cdot w_4 & \overset{!}{=} & -1 \\
\hat{f}(0,1) & = & e^{-1}\cdot w_1 & + & w_2 & + & e^{-2}\cdot w_3 & + & e^{-1}\cdot w_4 & \overset{!}{=} & 1 \\
\hat{f}(1,0) & = & e^{-1}\cdot w_1 & + & e^{-2}\cdot w_2 & + & w_3 & + & e^{-1}\cdot w_4 & \overset{!}{=} & 1 \\
\hat{f}(1,1) & = & e^{-2}\cdot w_1 & + & e^{-1}\cdot w_2 & + & e^{-1}\cdot w_3 & + & w_4 & \overset{!}{=} & -1
\end{array}$$

$$P = \begin{pmatrix} 1 & e^{-1} & e & e^{-2} \\ e^{-1} & 1 & e^{-2} & e^{-1} \\ e^{-1} & e^{-2} & 1 & e^{-1} \\ e^{-2} & e^{-1} & e^{-1} & 1 \end{pmatrix} \quad y = \begin{pmatrix} -1 \\ 1 \\ 1 \\ -1 \end{pmatrix} \quad w^* = P^{-1}y = \frac{e^2}{(e-1)^2}\begin{pmatrix} -1 \\ 1 \\ 1 \\ -1 \end{pmatrix}$$

Example: XOR via RBF

$$\hat{f}(x) = \frac{e^2}{(e-1)^2} \cdot \left[-e^{-x_1^2-x_2^2} + e^{-x_1^2-(x_2-1)^2} + e^{-(x_1-1)^2-x_2^2} - e^{-(x_1-1)^2-(x_2-1)^2} \right]$$

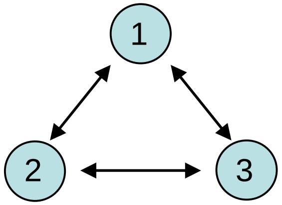


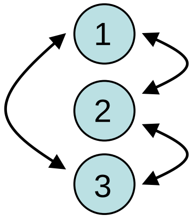
Hopfield Network

proposed 1982

characterization:

- neurons preserve state until selected at random for update
- bipolar states: x ∈ { -1, +1 }^{n}
- n neurons fully connected
- symmetric weight matrix
- no self-loops (→ zero main diagonal entries)
- thresholds θ , neuron i fires if excitations larger than θ_i





transition: select index k at random, new state is $\tilde{x} = \text{sgn}(\, x\, W - \theta\,)$

where $\tilde{x} = (x_1, \ldots, x_{k-1}, \tilde{x}_k, x_{k+1}, \ldots, x_n)$

energy of state x is $E(x) = -\frac{1}{2}\, xWx' + \theta\, x'$

Fixed Points

Definition

x is ***fixed point*** of a Hopfield network iff $x = \mathrm{sgn}(x' W - \theta)$. □

Example:

Set $W = x x'$ and choose θ with $| \theta_i | < n$, where $x \in \{-1, +1\}^n$.

$\rightarrow$ $\mathrm{sgn}(x' W - \theta) = \mathrm{sgn}(x' (x x')) = \mathrm{sgn}((x'x) x' - \theta) = \mathrm{sgn}(|| x ||^2 x' - \theta)$

Note that $|| x ||^2 = n$ for all $x \in \{-1, +1\}^n$.

$\rightarrow$ $x_i = +1$: $\mathrm{sgn}(n \cdot (+1) - \theta_i) = +1$ iff $+n - \theta_i \geq 0 \Leftrightarrow \theta_i \leq +n$

$\rightarrow$ $x_i = -1$: $\mathrm{sgn}(n \cdot (-1) - \theta_i) = -1$ iff $-n - \theta_i < 0 \Leftrightarrow \theta_i > -n$

Theorem:

If $W = x x'$ and $| \theta_i | < n$ then x is fixed point of a Hopfield network. □

Hopfield Network (HN)

Concept of Energy Function

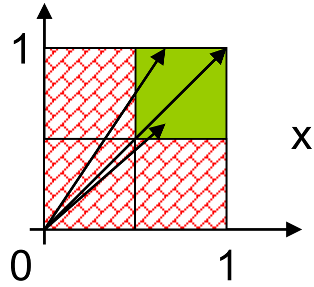
<u>given:</u> HN with W = x x' $\Rightarrow$ x is stable state of HN

starting point $x^{(0)}$ $\Rightarrow x^{(1)} = \text{sgn}(x^{(0)'} W - \theta)$

$\Rightarrow$ excitation $e = W x^{(1)} - \theta$

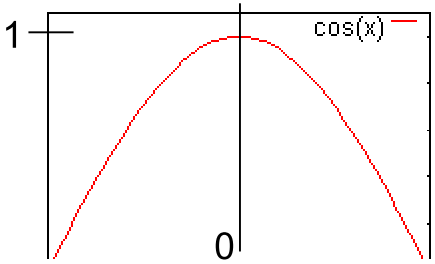
$\Rightarrow \underbrace{\text{if sign}(e) = x^{(0)}}_{\text{true if } e' \textit{ close} \text{ to } x^{(0)}}$ then $x^{(0)}$ stable state

small angle between e' and $x^{(0)}$ $\Leftarrow$



$x^{(0)} = (1, 1)$

recall: $$\frac{ab'}{\|a\| \cdot \|b\|} = \cos \angle(a, b)$$



small angle $\alpha \Rightarrow$ large cos(α)

Concept of Energy Function

required:

small angle between $e = W x^{(0)} - \theta$ and $x^{(0)}$

$\Rightarrow$ larger cosine of angle indicates greater similarity of vectors

$\Rightarrow$ $\forall e'$ of equal size: try to maximize $x^{(0)} e' = \underbrace{\| x^{(0)} \|}_{\text{fixed}} \cdot \underbrace{\| e \|}_{\text{fixed}} \cdot \underbrace{\cos \angle (x^{(0)}, e)}_{\to \text{max!}}$

$\Rightarrow$ maximize $x^{(0)'} e = x^{(0)'} (W x^{(0)} - \theta) = x^{(0)'} W x^{(0)} - \theta' x^{(0)}$

$\Rightarrow$ identical to minimize $-x^{(0)'} W x^{(0)} + \theta' x^{(0)}$

Definition

Energy function of HN at iteration t is $E(x^{(t)}) = -\frac{1}{2} x^{(t)'} W x^{(t)} + \theta' x^{(0)}$ □

Hopfield Network

Theorem:

Hopfield network converges to local minimum of energy function after a finite number of updates. □

Proof: assume that x$_k$ has been updated $\tilde{x}_k = -x_k$ and $\tilde{x}_i = x_i$ for $i \neq k$

$$E(x) - E(\tilde{x}) \;=\; -\tfrac{1}{2}\, x W x' + \theta\, x' + \tfrac{1}{2}\, \tilde{x} W \tilde{x}' - \theta\, \tilde{x}'$$

$$= -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij}\, x_i\, x_j + \sum_{i=1}^{n} \theta_i\, x_i + \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij}\, \tilde{x}_i\, \tilde{x}_j - \sum_{i=1}^{n} \theta_i\, \tilde{x}_i$$

$$= -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij}\, (x_i\, x_j - \tilde{x}_i\, \tilde{x}_j) + \sum_{i=1}^{n} \theta_i\, \underbrace{(x_i - \tilde{x}_i)}_{= 0 \text{ if } i \neq k}$$

$$= -\frac{1}{2} \sum_{\substack{i=1 \\ i \neq k}}^{n} \sum_{j=1}^{n} w_{ij}\, (x_i\, x_j - \underset{\substack{\| \\ x_i}}{\tilde{x}_i}\, \tilde{x}_j) - \frac{1}{2} \sum_{j=1}^{n} \underset{\substack{\| \\ 0 \text{ if } j = k}}{w_{kj}}\, (x_k\, x_j - \tilde{x}_k\, \underset{\substack{\| \\ x_j \text{ if } j \neq k}}{\tilde{x}_j}) + \theta_k\, (x_k - \tilde{x}_k)$$

$$= -\frac{1}{2}\sum_{\substack{i=1\\i\neq k}}^{n}\sum_{j=1}^{n} w_{ij}\,x_i\,\underbrace{(x_j-\tilde{x}_j)}_{=0 \text{ if } j \neq k} - \frac{1}{2}\sum_{\substack{j=1\\j\neq k}}^{n} w_{kj}\,x_j\,(x_k-\tilde{x}_k) + \theta_k\,(x_k-\tilde{x}_k)$$

$$= -\frac{1}{2}\sum_{\substack{i=1\\i\neq k}}^{n} w_{ik}\,x_i\,(x_k-\tilde{x}_k) - \frac{1}{2}\sum_{\substack{j=1\\j\neq k}}^{n} w_{kj}\,x_j\,(x_k-\tilde{x}_k) + \theta_k\,(x_k-\tilde{x}_k)$$

(rename j to i, recall W = W‘, $w_{kk} = 0$)

$$= -\sum_{i=1}^{n} w_{ik}\,x_i\,(x_k-\tilde{x}_k) + \theta_k\,(x_k-\tilde{x}_k)$$

$$\underbrace{= -(x_k-\tilde{x}_k)\left[\underbrace{\sum_{i=1}^{n} w_{ik}\,x_i}_{\text{excitation } e_k} - \theta_k\right]}_{> 0 \text{ if } x_k < 0 \text{ and vice versa}} > 0$$

since:

x_k	$x_k - \tilde{x}_k$	$e_k - \theta_k$	ΔE
$+1$	> 0	< 0	> 0
-1	< 0	> 0	> 0

Hopfield Network

⇒ every update (change of state) decreases energy function

⇒ since number of different bipolar vectors is finite
update stops after finite #updates

remark: dynamics of HN get stable in local minimum of energy function!

q.e.d.

⇒ Hopfield network can be used to optimize combinatorial optimization problems!

Application to Combinatorial Optimization

Idea:

- transform combinatorial optimization problem as objective function with $x \in \{-1,+1\}^n$
- rearrange objective function to look like a Hopfield energy function
- extract weights W and thresholds θ from this energy function
- initialize a Hopfield net with these parameters W and θ
- run the Hopfield net until reaching stable state (= local minimizer of energy function)
- stable state is local minimizer of combinatorial optimization problem

Example I: Linear Functions

$$f(x) = \sum_{i=1}^{n} c_i \, x_i \quad \to \min! \qquad (\, x_i \in \{-1, +1\} \,)$$

Evidently: $E(x) = f(x)$ with $W = 0$ and $\theta = c$

$\Downarrow$

```
choose $x^{(0)} \in \{-1, +1\}^n$
set iteration counter $t = 0$
repeat
    choose index $k$ at random
    $x_k^{(t+1)} = \mathrm{sgn}(x^{(t)} \cdot W_{\cdot,k} - \theta_k) = \mathrm{sgn}(x^{(t)} \cdot 0 - c_k) = -\mathrm{sgn}(c_k) = \begin{cases} -1 & \text{if } c_k > 0 \\ +1 & \text{if } c_k < 0 \end{cases}$
    increment $t$
until reaching fixed point
```

⇒ fixed point reached after Θ(n log n) iterations on average

[proof: → black board]

Example II: MAXCUT

given: graph with n nodes and symmetric weights $\omega_{ij} = \omega_{ji}$, $\omega_{ii} = 0$, on edges

task: find a partition $V = (V_0, V_1)$ of the nodes such that the weighted sum of edges with one endpoint in V_0 and one endpoint in V_1 becomes maximal

encoding: $\forall$ i=1,...,n: $y_i = 0$, node i in set V_0; $y_i = 1$, node i in set V_1

objective function: $$f(y) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \omega_{ij} \left[y_i \, (1-y_j) + y_j \, (1-y_i) \right] \quad \to \max!$$

preparations for applying Hopfield network

step 1: conversion to minimization problem

step 2: transformation of variables

step 3: transformation to “Hopfield normal form“

step 4: extract coefficients as weights and thresholds of Hopfield net

Example II: MAXCUT (continued)

step 1: conversion to minimization problem

⇒ multiply function with -1 ⇒ E(y) = -f(y) → min!

step 2: transformation of variables

⇒ $y_i = (x_i+1) / 2$

$$\Rightarrow f(x) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \omega_{ij} \left[\frac{x_i+1}{2} \left(1 - \frac{x_j+1}{2} \right) + \frac{x_j+1}{2} \left(1 - \frac{x_i+1}{2} \right) \right]$$

$$= \frac{1}{2} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \omega_{ij} \left[1 - x_i \, x_j \right]$$

$$= \underbrace{\xcancel{\frac{1}{2} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n}} \omega_{ij}} - \frac{1}{2} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \omega_{ij} \, x_i \, x_j$$

constant value (does not affect location of optimal solution)

Example II: MAXCUT (continued)

step 3: transformation to “Hopfield normal form“

$$E(x) = \frac{1}{2}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} \omega_{ij}\, x_i\, x_j \quad = -\frac{1}{2}\underset{i \neq j}{\sum_{i=1}^{n}\sum_{j=1}^{n}} \underbrace{\left(-\frac{1}{2}\omega_{ij}\right)}_{\text{w}_{\text{ij}}} x_i\, x_j$$

$$= -\frac{1}{2}x'Wx + \underset{\substack{\downarrow \\ 0'}}{\theta'} x$$

step 4: extract coefficients as weights and thresholds of Hopfield net

$$w_{ij} = -\frac{\omega_{ij}}{2} \text{ for } i \neq j, \quad w_{ii} = 0, \quad \theta_i = 0$$

remark: ω_{ij} : weights in graph — w_{ij} : weights in Hopfield net