Development of Complex Curricula for Molecular Bionics and Infobionics Programs within a consortial* framework**

Consortium leader

PETER PAZMANY CATHOLIC UNIVERSITY

Consortium members

SEMMELWEIS UNIVERSITY, DIALOG CAMPUS PUBLISHER

The Project has been realised with the support of the European Union and has been co-financed by the European Social Fund ***

**Molekuláris bionika és Infobionika Szakok tananyagának komplex fejlesztése konzorciumi keretben

***A projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg.

Digital- and Neural Based Signal Processing & Kiloprocessor Arrays

Digitális- neurális-, és kiloprocesszoros architektúrákon alapuló jelfeldolgozás

Hopfield network, Hopfield net as associative memory and combinatorial optimizer

Hopfield hálózat, Hopfield, mint asszociatív memória és kombinatorikus optimalizáló

J. Levendovszky, A. Olah, G. Treplan, D. Tisza

Outline

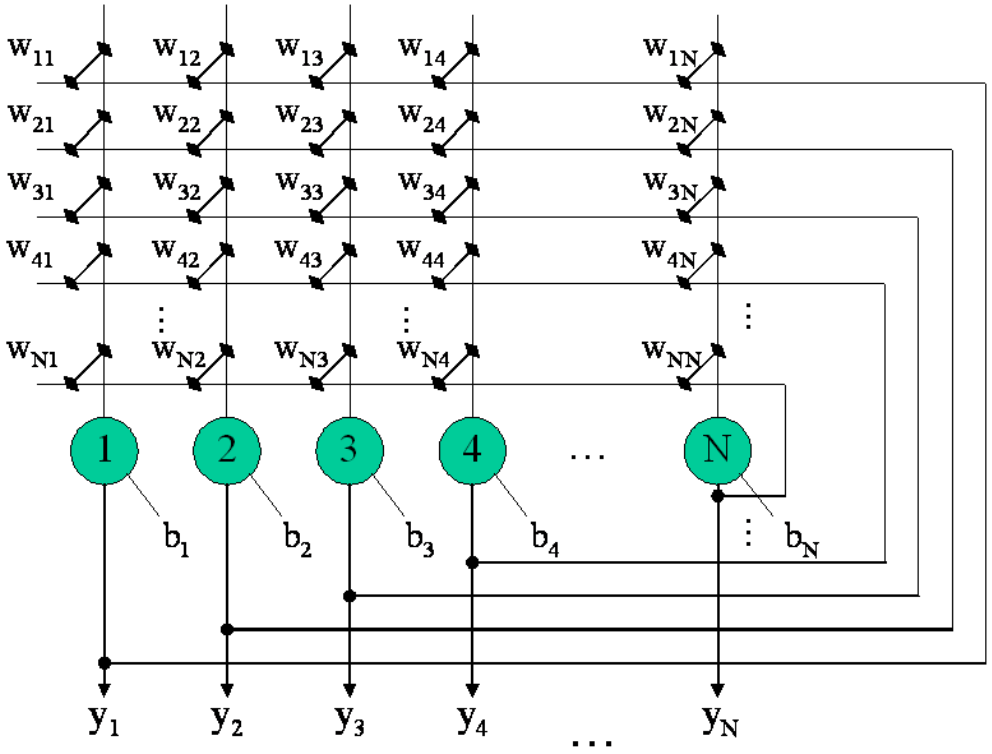
- Introduction
- Hopfield net - Structure and operation
- Hopfield net - Stability and convergence properties
- Hopfield net as an Associative Memory (AM)
- Capacity analysis of the Hopfield net
- Applications of Hopfield net as AM
- Hopfield net as combinatorial optimizers
- The way towards CNN
- Example Problems

Introduction (1)

Hopfield neural network is a

- Recurrent artificial neural network,
- Invented by John Hopfield,
- Serve as an associative memory system
- Or operate as a combinatorial optimizer (quadratic programming)
- A stable dynamic system, guaranteed to converge to a local minimum
- Convergence to one of the stored patterns is not guaranteed.

Introduction (2)



Number of connections:

$$N^2$$

Implementation difficulty!

- Topology of Hopfield Neural Network

Structure and operation (1)

Notations (1)

- Weight matrix contains the W_{ij} synaptic weight strength feedback from neuron i to neuron j:

$$\mathbf{W} = \begin{bmatrix} W_{11} & W_{12} & W_{13} & & W_{1N} \\ W_{21} & W_{22} & W_{23} & \dots & W_{2N} \\ W_{31} & W_{32} & W_{33} & & \dots \\ \dots & & & \dots & \dots \\ W_{N1} & \dots & \dots & \dots & W_{NN} \end{bmatrix},$$

- where

$$W_{ij} = W_{ji},\ i, j = 1, \dots, N.$$

Structure and operation (2)

Notations (2)

- Bias vector contains the threshold values of each neuron:

$$\mathbf{b} = \begin{bmatrix} b_1 & b_2 & \dots & b_N \end{bmatrix}^T .$$

- Let state vector of the system be

$$\mathbf{y} = \begin{bmatrix} y_1 & y_2 & \dots & y_N \end{bmatrix}^T ,$$

- where

$$\mathbf{y} \in \{-1, +1\}^N .$$

Structure and operation (3)

- The discrete Hopfield Neural Network can be regarded as a nonlinear recursion given in the form of

$$y_l(k+1) = \operatorname{sgn}\left\{\sum_{j=1}^{N} W_{lj} y_j(k) - b_l\right\},$$

for every neuron.

- If we reduce our attention only to the sequential updating rule, the neuron selection rule becomes:

$$l = \operatorname{mod}_n k.$$

Structure and operation (4)

- When investigating such a nonlinear recursion as an associative mapping, the following questions can arise:

1. How to construct matrix **W** if one wants to store a set of patterns

$$S = \left\{ \mathbf{s}^{\alpha}, \alpha = 1,\ldots,M \right\}$$

as the fix points of algorithm

$$y_l(k+1) = \operatorname{sgn}\left\{ \sum_{j=1}^{N} W_{lj} y_j(k) - b_l \right\},$$

such as that

$$s_l^{\alpha} = \operatorname{sgn}\left\{ \sum_{j=1}^{n} W_{lj} s_j{}^{\alpha} - b_l \right\} \quad \forall \alpha = 1,\ldots,\mathrm{M}$$

holds.

Structure and operation (5)

2. What is the number of those fix points *M* as a function of the dimension (number of neurons in the Hopfield net) ?

   In other words we want to reveal the *storage capacity* of the Hopfield net as a function of the number of neurons *N*.

$$M = \Upsilon(N)$$

$$\Upsilon(.) = ?$$

Structure and operation (6)

3. Is recursion

$$y_l(k+1) = \operatorname{sgn}\left\{\sum_{j=1}^{N} W_{lj} y_j(k) - b_l\right\}$$

stable and if yes then what are the its convergence properties?

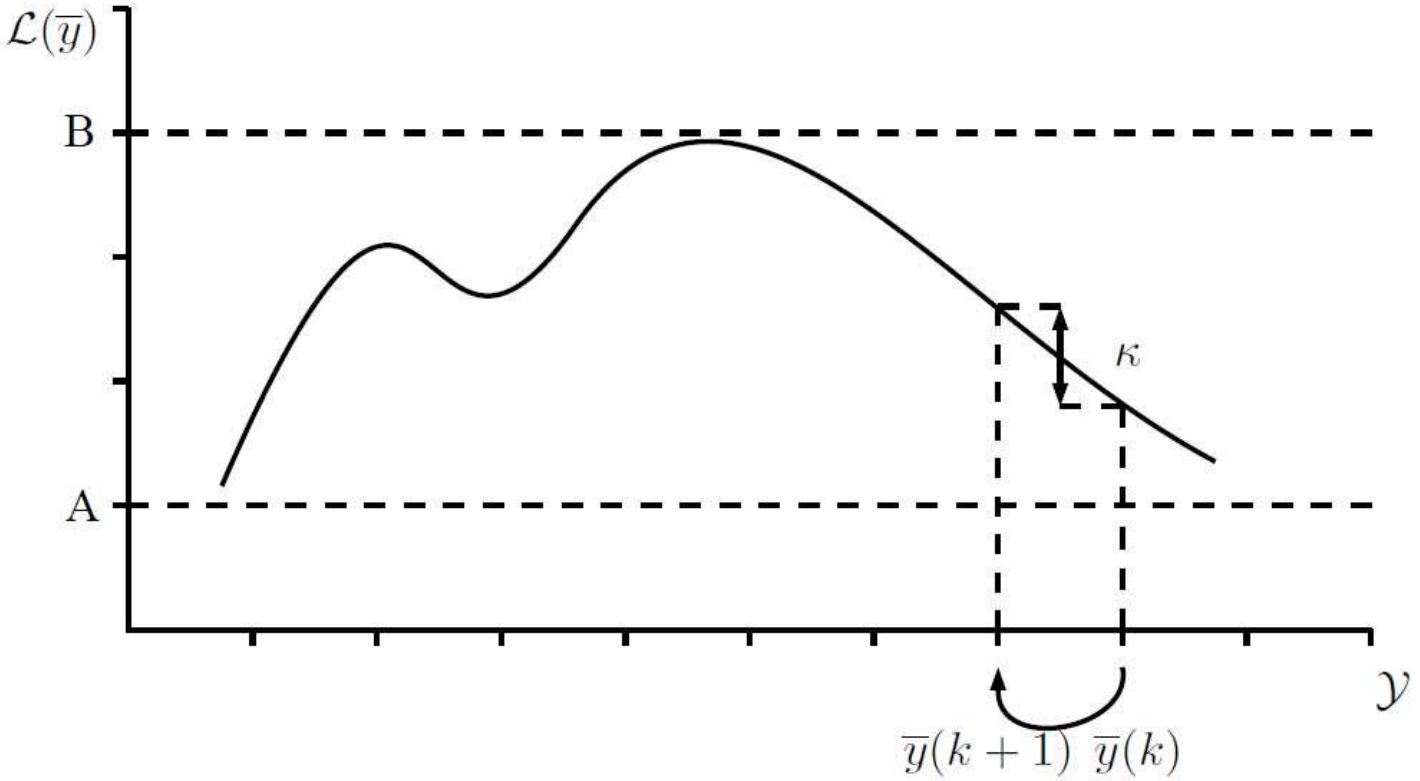
- Next we will thoroughly discuss these questions. Before getting down to a detailed analysis, we need some tools rooted in the classical stability theory called *Lyapunov technique*.

Stability and convergence properties (1)

- Lyapunov[1] functions are widely used in the study of dynamical systems in order to prove the stability of a system. T
- This technique can be used to analyze the stability properties of the Hopfield Neural Networks.

[1] Aleksandr Mikhailovich Lyapunov (June 6, 1857 - November 3, 1928) was a Russian mathematician, mechanician and physicist.

Stability and convergence properties (2)



- One possible Lyapunov function

Stability and convergence properties (3)

Lyapunov's weak theorem (1)

- Let us assume that there is a nonlinear recursion given in the following general form

$$\mathbf{y}(k+1) = \varphi\left(\mathbf{y}(k)\right) \quad \mathbf{y}(k) \in Y.$$

- If one can define a function (the so-called Lyapunov function)

$$L(y) \quad y \in Y$$

- over the state space Y, for which

Stability and convergence properties (4)

Lyapunov's weak theorem (2)

1. $L(\mathbf{y})$ has a global upper bound over the state space:

$$L(\mathbf{y}) \le B \quad \forall \mathbf{y} \in Y;$$

2. the change of $L(\mathbf{y})$ denoted by

$$\Delta L(k) := L\big(y(k+1)\big) - L\big(y(k)\big) > 0$$

in each step of the recursion;

Then the recursion

$$\mathbf{y}(k+1) = \varphi\big(\mathbf{y}(k)\big)$$

is stable and converges one of the local maxima of $L(\mathbf{y})$.

Stability and convergence properties (5)

Lyapunov's weak theorem (3)

- The exact proof, which can be found in numerous books dealing with control and stability theory, is omitted here.
- However, it is easy to see if in each step the $L(\mathbf{y})$ can only increase and at the same time there exist a global lower bound then the recursion cannot go on indefinitely but it will converge to one of the local minima.

Stability and convergence properties (6)

Lyapunov's strong theorem (1)

- Let us assume that there is a nonlinear recursion given in the following general form

$$\mathbf{y}(k+1) = \varphi\left(\mathbf{y}(k)\right) \quad \mathbf{y}(k) \in Y.$$

- If one can define a function (the so-called Lyapunov function)

$$L(y) \quad y \in Y$$

- over the state space Y, for which

Stability and convergence properties (7)

Lyapunov's strong theorem (2)

1. $L(\mathbf{y})$ has a global lower and upper bound over the state space:

$$B \geq L(\mathbf{y}) \geq A \quad \forall \mathbf{y} \in Y;$$

2. the change of $L(\mathbf{y})$ denoted by

$$\Delta L(k) := L\big(y(k+1)\big) - L\big(y(k)\big) \geq \chi$$

in each step of the recursion;

Then the recursion

$$\mathbf{y}(k+1) = \varphi\big(\mathbf{y}(k)\big)$$

is stable and converges one of the local maxima of $L(\mathbf{y})$.

Stability and convergence properties (8)

Lyapunov's strong theorem (3)

and its transient time can be upper bounded as .

$$TR \leq \frac{B - A}{\chi}$$

- Again the proof omitted, however it is easy to interpret the result as follows: in each step $L(\mathbf{y})$ increases at least by κ and in the worst case the maximum number of steps needed to cover the distance $B - A$ is $TR \leq \frac{B - A}{\chi}.$
- With this tools at our disposal we can embark on ascertaining the stability and convergence properties of the Hopfield net.

Stability and convergence properties (9)

- To use the Lyapunov technique we have to assume a Lyapunov function associated to recursion

$$\mathbf{y}(k+1) = \varphi\big(\mathbf{y}(k)\big) \quad \mathbf{y}(k) \in Y.$$

- According to Hopfield, Cohen and Grossberg, we define the corresponding Lyapunov function as follows:

$$L(\mathbf{y}) = \mathbf{y}^{\mathrm{T}}\mathbf{W}\mathbf{y} - \mathbf{2}\mathbf{b}^{\mathrm{T}}\mathbf{y} = \sum_i \sum_j y_i W_{ij} y_j - 2\sum_i b_i y_i.$$

Stability and convergence properties (10)

- Now we want to apply Lyapunov's strong theorem, therefore we have to check the following three conditions:

1. existence of global upper bound;
2. existence of global lower bound;
3. it is true, that $L(\mathbf{y}) \leq \kappa$.

Stability and convergence properties (11)

The existence of global upper bound

- To derive an upper bound we can use the Cauchy-Schwartz inequality as follows:

$$L(\mathbf{y}) = \mathbf{y}^{\mathrm{T}}\mathbf{W}\mathbf{y} - \mathbf{2}\mathbf{b}^{\mathrm{T}}\mathbf{y} \le \|\mathbf{y}\|\|\mathbf{W}\mathbf{y}\| + 2\|\mathbf{b}\|\|\mathbf{y}\| \le$$

$$\le \|\mathbf{y}\|\|\mathbf{W}\mathbf{y}\| + 2\|\mathbf{b}\|\|\mathbf{y}\| = \|\mathbf{W}\|\|\mathbf{y}\|^2 + 2\|\mathbf{b}\|\|\mathbf{y}\|,$$

- taking into account that we are dealing with binary state vectors, elements of $\{-1,1\}^N$ for which

$$\|\mathbf{y}\|^2 = \sum_{i=1}^{N} y_i^2 = N \longrightarrow L(\mathbf{y}) \le N\|\mathbf{W}\| + 2\sqrt{N}\|\mathbf{b}\|.$$

Stability and convergence properties (12)

The existence of global lower bound (1)

- To derive a global lower upper for $L(\mathbf{y})$ let us first broaden the state space from $Y = \{-1,1\}^N$ to the space of N dimensional real numbers and define

$$L(\mathbf{x}), \quad \mathbf{x} \in \Re^N$$

- over this broadened state space. Therefore

$$\min_{y \in Y} L(y) \geq \min_{x \in R^n} L(x).$$

Stability and convergence properties (13)

The existence of global lower bound (2)

- The minimum

$$\min_{x \in R^n} L(x)$$

- can be easily calculated considering that

$$\Re^N$$

- is continuum therefore the gradient of

$$L(\mathbf{x}) = \mathbf{x}^T \mathbf{W} \mathbf{x} - \mathbf{2b}^T \mathbf{x} = \sum_i \sum_j x_i W_{ij} x_j - 2 \sum_i b_i x_i$$

- exists and form the well known results related to quadratic forms the location of this maximum is given as $\mathbf{x}_{opt} = \mathbf{W}^{-1}\mathbf{b}$.

Stability and convergence properties (14)

The existence of global lower bound (3)

- Substituting $\mathbf{x}_{opt}$ into

$$L(\mathbf{x}) = \mathbf{x}^{\mathrm{T}}\mathbf{W}\mathbf{x} - \mathbf{2}\mathbf{b}^{\mathrm{T}}\mathbf{x} = \sum_i \sum_j x_i W_{ij} x_j - 2\sum_i b_i x_i,$$

- one obtains that

$$L(\mathbf{y}) \geq \min_{y \in Y} L(\mathbf{y}) \geq \min_{x \in R^n} L(\mathbf{x}) = -\mathbf{b}^{\mathrm{T}}\mathbf{W}^{-1}\mathbf{b} \geq -\left\|\mathbf{W}^{-1}\right\|\left\|\mathbf{b}^2\right\|.$$

$$L(\mathbf{y}) \geq -\left\|\mathbf{W}^{-1}\right\|\left\|\mathbf{b}^2\right\|$$

Stability and convergence properties (15)

The change of the Lyapunov function (1)

- Let us define the change of the Lyapunov function as follows

$$\Delta L(k) := L(y(k+1)) - L(y(k)) =$$

$$= \sum_i \sum_j y_i(k+1) W_{ij} y_j(k+1) - 2\sum_i b_i y_i(k+1)$$

$$- \sum_i \sum_j y_i(k) W_{ij} y_j(k) \qquad + 2\sum_i b_i y_i(k).$$

Stability and convergence properties (16)

The change of the Lyapunov function (2)

- We apply the sequential update rule, which means that only the component

$$l = \mathrm{mod}_N\, k$$

- in the state vector $\mathbf{y}(k)$ changes, we can write

$$\mathbf{y}(k) = \begin{bmatrix} y_1(k) \\ y_2(k) \\ \vdots \\ y_l(k) \\ \vdots \\ y_N(k) \end{bmatrix} \longrightarrow \mathbf{y}(k+1) = \begin{bmatrix} y_1(k) \\ y_2(k) \\ \vdots \\ y_l(k+1) \\ \vdots \\ y_N(k) \end{bmatrix}.$$

Stability and convergence properties (17)

The change of the Lyapunov function (3)

- Taking this into consideration $\Delta L(k)$ takes the following form

$$\Delta L(k) := W_{ll}\Delta y_l^2(k) + 2\Delta y_l(k)\left(\sum_j W_{lj} y_j(k) - b_l\right),$$

- where

$$\Delta y_l(k) = y_l(k+1) - y_l(k).$$

Stability and convergence properties (18)

The change of the Lyapunov function (4)

- Let us introduce quantity κ as

$$\kappa = \min_{y \in \{-1,1\}^n, i=1,..,n} \left| \sum_j W_{ij} y_j - b_i \right|.$$

- To calculate the values of this expression we can create the following table:

$y_l(k)$	$y_l(k+1)$	$\Delta y_l(k)$	$W_{ll}\Delta y_l^2(k)$	$2\Delta y_l(k)$	$\sum_j W_{lj}\Delta y_j(k)$-$b_l$	$\Delta L(k)$
-1	+1	+2	$4W_{ll}$	+4	$\kappa>0$	$4(W_{ll}+\kappa)>0$
+1	-1	-2	$4W_{ll}$	-4	$-\kappa<0$	$4(W_{ll}+\kappa)>0$

Stability and convergence properties (19)

The change of the Lyapunov function (5)

- From this table it can be seen that whenever a state transition occurs, then

$$\Delta L(k) \geq 4(W_{ll} + \chi)$$

- This means that for each step when a state transition occurs the energy function increases.
- Given a global upper and lower bound it follows that there exist a time step where the algorithm must stop in a local maxima.

Stability and convergence properties (20)

- **Theorem 3.** The Hopfield type of recursion

1. is stable;
2. it converges to the local maxima of the function

$$L(\mathbf{y}) = \mathbf{y}^{\mathbf{T}}\mathbf{W}\mathbf{y} - \mathbf{2}\mathbf{b}^{\mathbf{T}}\mathbf{y} = \sum_i \sum_j y_i W_{ij} y_j - 2\sum_i b_i y_i;$$

3. the transient time can be upper-bounded as

$$TR \le N \frac{N\|\mathbf{W}\| + 2\sqrt{N}\|\mathbf{b}\| + \|\mathbf{W}^{-1}\|\|\mathbf{b}\|^2}{4(W_{ll} + \chi)} \approx O\left(N^2\right).$$

Hopfield Neural Network as an AM (1)

- Analyzing the non-linear state recursion of the Hopfield Neural Network we have come to the conclusion that this is a finite state automata with binary state vectors, which gave rise to a new application of this network: Associative Mapping (AM).

- We start the HNN from an initial state vector **x**, which corresponds to a corrupted version of a stored memory item, we call this vector clue.

Hopfield Neural Network as an AM (2)

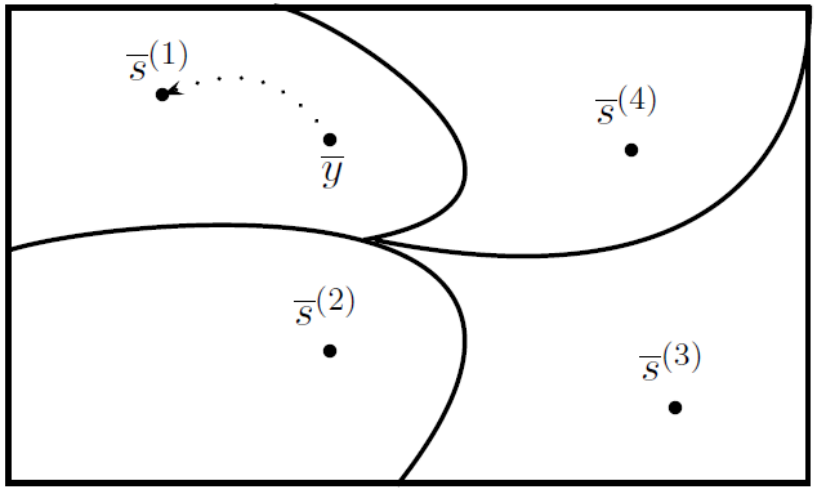
- Then we start the iteration

$$y_l(k+1) = \operatorname{sgn}\left\{\sum_{j=1}^{N} W_{lj} y_j(k) - b_l\right\}$$

- of the network, and if the network gets stuck in one of its steady-states then we call this vector as recalled memory item.
- This mapping is the so-called Associative Mapping.

Hopfield Neural Network as an AM (3)

- This new computational gives rise to the model in Figure where the N dimensional binary vectors are mapped into two dimensions. The box represents the state space $Y \in \{-1, 1\}^N$, there are a couple of fix points of the HNN $\mathbf{s}^{(1)}$, $\mathbf{s}^{(2)}$, $\mathbf{s}^{(3)}$, $\mathbf{s}^{(4)}$.



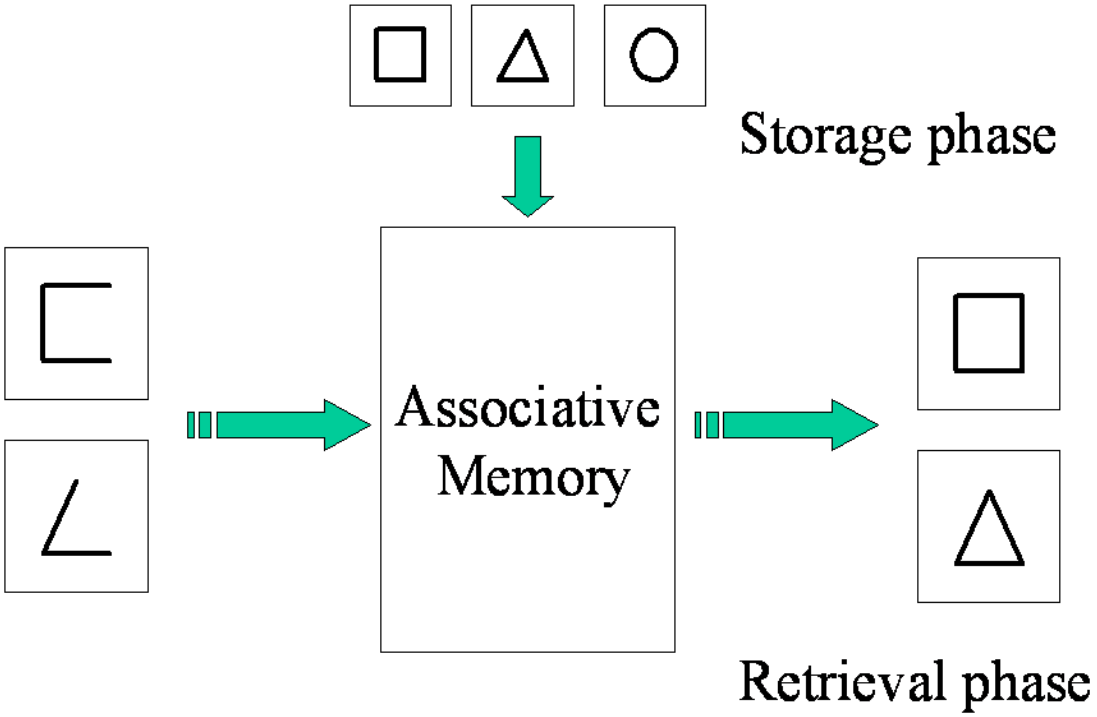
- An Associative Mapping is a partitioning of the space $Y \in \{-1, 1\}^N$.

Hopfield Neural Network as an AM (4)

- There is a separation of this state space, in terms if we start the network from a state **x**, which falls into the basin of convergence of the memory pattern $\mathbf{s}^{(4)}$, then finally the network will stuck in this steady state.
- In general this also holds for the other memory patterns, and their basin of attraction.

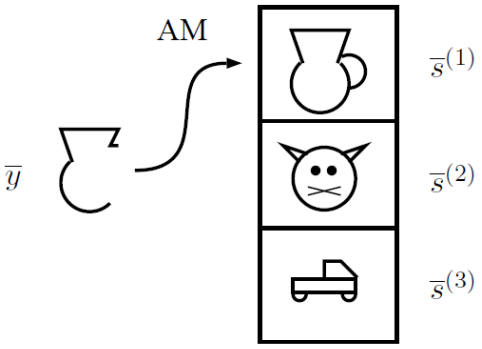
Hopfield Neural Network as an AM (5)

- A pretty general demonstration of the working of an Associative Mapping is depicted in figure.



Hopfield Neural Network as an AM (6)

- Lets assume that there are some stored memory items, for example a picture of a vase, a cat and a lorry, these are the three patterns.
- An Associative Mapping means that if we have a corrupted and incomplete version of one of the memory patterns then it will be mapped to one of the stored items, which is the closest.



A demonstration how Associative Mapping works.

Hopfield Neural Network as an AM (7)

- In order to use the Hopfield Neural Network to implement this new computational paradigm, we have to make sure that the network is stable, meaning the network will converge to a steady state.
- However when we started to investigate the stability, we came up with the Lyapunov concept of stability, and there is a quadratic form

$$L(\mathbf{y}) = \mathbf{y}^{\mathrm{T}}\mathbf{W}\mathbf{y} - \mathbf{2b}^{\mathrm{T}}\mathbf{y} = \sum_i \sum_j y_i W_{ij} y_j - 2\sum_i b_i y_i,$$

- which is the Lyapunov function of the HNN, and we have proven that the Hopfield Network is stable.

Hopfield Neural Network as an AM (8)

- Furthermore we have come to the conclusion that the transient time of this network is $O(N^2)$, which is must faster than exhaustive search $O(2^N)$.
- When one implements an Associative Memory, there is a given set of items, which are to be stored, this is called the set of stored memory items, and noted by S,

$$S = \left\{ \mathbf{s}^{(\alpha)}, \alpha = 1,\ldots,M \right\}$$

- here the number of the stored memory items is M, which items are binary vectors, with dimension N,

$$\dim\left(\mathbf{s}^{(\alpha)} \right) = N, \alpha = 1,\ldots,M.$$

Hopfield Neural Network as an AM (9)

- These binary vectors can refer to images, speech patterns or bank account numbers, depending on the actual application.
- The set X represents the observation space, this can be any possible N dimensional binary vector,

$$X = \{-1,1\}^N .$$

- We can see that $S \subset X$, because $\mathbf{s}^{(\alpha)}$ is also N dimensional binary vector.

Hopfield Neural Network as an AM (10)

- **Definition 1.** An Associative Mapping ψ is defined as a mapping from the observation space X to the set of stored memory items S, in such a way that it maps any specific observation vector $\mathbf{x}$ into a stored memory item $\mathbf{s}^{(\beta)}$ for which it holds, that this stored memory item is the closest to the observation vector, according to a certain distance criterion $d()$.
- Formally *AM is a*

$$\psi : X \to S$$

$$\psi(\mathbf{x}) = s^{\beta}, \text{ for which } d\left(\mathbf{s}^{(\beta)}, \mathbf{x}\right) \leq d\left(\mathbf{s}^{(\alpha)}, \mathbf{x}\right), \ \forall \alpha = 1,...,M.$$

Hopfield Neural Network as an AM (11)

- We can define distance in any arbitrary way, which suits our application. However when we deal with binary vectors it is rather plausible that this distance is the Hamming distance, which measures in how many bits two vectors differ from each other.
- There are two fundamental attributes of an Associative Memory: 1) stability, 2) capacity.
- Where capacity boils down to that what is the size of the stored memory items, in our notation $M = |S|$. When we speak of the analysis of Hopfield Neural Network as an Associative Mapping we are trying to reveal these two properties.

Capacity Analysis (1)

- When we use the Hopfield Neural Network as an Associative Memory the threshold vector b is set to the all zero vector 0, and if we want to store in the network a predefined set of memory items then the components of the weight matrix is as follows:

$$W_{lj} = \begin{cases} \frac{1}{N}\sum_{\alpha=1}^{M} s_l^{(\alpha)} s_j^{(\alpha)} & \text{if} \quad l \neq j \\ 0 & \text{if} \quad l = j \end{cases},$$

- which can be written with vector notations using the outer product operation:

$$\mathbf{W} = \frac{1}{N}\sum_{\alpha=1}^{M} s^{(\alpha)} \left(s^{(\alpha)}\right)^{\mathrm{T}}.$$

Capacity Analysis (2)

- This rule was discovered and fully elaborated on in the work of D. O. Hebb[3], a Canadian born psychologist.
- That's why it is named as **Hebbian Learning Rule**. He described this rule as a psychologist in a textual way and from this description it was mathematically inferred that

$$\mathbf{W} = \frac{1}{N}\sum_{\alpha=1}^{M}\mathbf{s}^{(\alpha)}\mathbf{s}^{(\alpha)T}$$

is the learning rule.

[3]Donald Olding Hebb (July 22, 1904 - August 10, 1985) was a psychologist who was influential in the area of neuropsychology, where he sought to understand how the function of neurons contributed to psychological processes such as learning. He has been described as the father of neuropsychology and neural networks.

Capacity Analysis (3)

- By capacity we mean that how many stored memory items can be recalled from the Hopfield Neural Network.
- In order to do that we start with a very elementary investigation and then are going to penetrate deeper and deeper into the capacity analysis, until we arrive at the stage of

1. Statistical Neurodynamics and
2. the Informational Theoretical Capacity.

Capacity Analysis (4)

Static Capacity (1)

- The first issue is the so called Static Capacity Analysis and Fix point Analysis. Recall that there is a set of stored memory items, which are represented by binary vectors:

$$S = \left\{ \mathbf{s}^{(\alpha)}, \alpha = 1,...,M \right\}$$

- and the W_{lj} element of the weight matrix is set according to the Hebbian Learning Rule

$$W_{lj} = \begin{cases} \frac{1}{N} \sum_{\alpha=1}^{M} s_l^{(\alpha)} s_j^{(\alpha)} & \text{if} \quad l \neq j \\ 0 & \text{if} \quad l = j \end{cases}.$$

Capacity Analysis (5)

Static Capacity (2)

- What we are going to investigate now is that if we pick up any stored memory vector $\mathbf{s}^{(\beta)} \in S$ in order to recall this vector what we have to make sure that this vector is a fix point of the Hopfield Network, which means that the recursion

$$y_l(k+1) = \operatorname{sgn}\left\{\sum_{j=1}^{N} W_{lj} y_j(k) - b_l\right\},$$

- exhibits an equilibrium behavior, in terms of once we have reached $\mathbf{s}^{(\beta)}$ we can not get out of this state. This can be written with vector notations as follows:

$$\mathbf{s}^{(\beta)} = \operatorname{sgn}\left(\mathbf{W}\mathbf{s}^{(\beta)}\right).$$

Capacity Analysis (6)

Static Capacity (3)

- One can see, that what we have spelled out here is the so called fix point of this non-linear recursion, which is a necessary condition for $\mathbf{s}^{(\beta)}$ being a stored memory item.
- The question is whether under what condition we can enforce this equality, meaning that what is the upper limit on the capacity which enforces that equation

$$\mathbf{s}^{(\beta)} = \mathrm{sgn}\left(\mathbf{W}\mathbf{s}^{(\beta)}\right)$$

- holds. Since we investigate this equation with respect to the number of stored memory items M, we are going to draw a condition under which this equation will hold.

Capacity Analysis (7)

Static Capacity (4)

- In order to simplify the analysis let us analyze

$$s_l^{(\beta)} = \operatorname{sgn}\left(\sum_{j=1}^{N} W_{lj} s_j^{(\beta)} \right), \ \forall l = 1,\ldots,N,$$

- which should hold for all components in order to obtain a steady state. We can replace W_{lj} with its definition and write

$$s_l^{(\beta)} = \operatorname{sgn}\left(\sum_{j=1}^{N} \frac{1}{N} \sum_{\alpha=1}^{M} s_l^{(\alpha)} s_j^{(\alpha)} s_j^{(\beta)} \right).$$

Capacity Analysis (8)

Static Capacity (5)

- However these are finite sum, as a result we can exchange the sequence of these summations, and we can rewrite this

$$s_l^{(\beta)} = \operatorname{sgn}\left(\sum_{\alpha=1}^{M} s_l^{(\alpha)} \frac{1}{N} \sum_{j=1}^{N} s_j^{(\alpha)} s_j^{(\beta)} \right).$$

- From the outer summation where α sweeps from 1 to M we can single out one term, where α equals to β, as $\mathbf{s}^{(\beta)}$ is an element of the set S, once α will hit the value of β, as a result we can rewrite the previous expression as follows

$$s_l^{(\beta)} = \operatorname{sgn}\left(s_l^{(\beta)} \frac{1}{N} \sum_{j=1}^{N} \left(s_j^{(\beta)} \right)^2 + \sum_{\alpha=1, \alpha \neq \beta}^{M} s_l^{(\alpha)} \frac{1}{N} \sum_{j=1}^{N} s_j^{(\alpha)} s_j^{(\beta)} \right).$$

Capacity Analysis (9)

Static Capacity (6)

- Where in the first part of the expression we have the square of $s^{(\beta)}_j$ which is always 1, because $\mathbf{s}^{(\beta)} \in \{-1, 1\}^N$, and adding up N times 1 gives N, which is divided by N yields 1. Let us denote the second part of the previous expression by v_l, which gives the following

$$s_l^{(\beta)} = \operatorname{sgn}\left(s_l^{(\beta)} + v_l\right).$$

- Now what we are investigating is that under what condition the above equation holds. We can see that if v_l is zero then indeed this equation will be satisfied. This criterion can be satisfied if the capacity is smaller than N $\quad M \leq N.$

Capacity Analysis (10)

Static Capacity (7)

- If we investigate the expression of v_l

$$v_l = \sum_{\alpha=1,\alpha\neq\beta}^{M} s_l^{(\alpha)} \frac{1}{N} \sum_{j=1}^{N} s_j^{(\alpha)} s_j^{(\beta)} = \sum_{\alpha=1,\alpha\neq\beta}^{M} s_l^{(\alpha)} \frac{1}{N} \left(\mathbf{s}^{(\alpha)}\right)^{\mathrm{T}} \mathbf{s}^{(\beta)},$$

- what we see is the inner product of the memory vectors. If

$$\left(\mathbf{s}^{(\alpha)}\right)^{\mathrm{T}} \mathbf{s}^{(\beta)} = 0, \ \forall \alpha, \beta = 1,\ldots,M, \ \alpha \neq \beta, \ \text{then } v_l = 0.$$

- This entails that the memory items must be orthogonal to each other because their inner product should be 0. However we have N dimensional memory items, and in an N dimensional space the maximum number of orthogonal vectors is N.

Capacity Analysis (11)

Dynamic Capacity (1)

- What we have investigated so far was a fix point analysis, but it does not necessarily entails that the Hopfield Network will converge to this fix point. Now we are going to pursue further investigation into this capacity matter, and the second stage of this investigation is that we are going to evaluate the Dynamic Capacity of the Hopfield Network, where the notion of Dynamic Capacity implies that we are investigating the steady states.

Capacity Analysis (12)

Dynamic Capacity (2)

- **Definition 2 (Steady state).** Steady states are a subset of fix points, into which the Hopfield Network converges.
- The stability of the Hopfield Network was proven by using the Lyapunov concept of stability, where the center point of the concept was that there is a Lyapunov function associated with the Hopfield Network.
- Since in the case of applying the HNN as an Associative Mapping vector **b** is zero what remains is as follows:

$$L(\mathbf{y}) = \mathbf{y}^{\mathrm{T}}\mathbf{W}\mathbf{y} = \sum_i \sum_j y_i W_{ij} y_j .$$

Capacity Analysis (13)

Dynamic Capacity (3)

- We have proven that the Hopfield Network is stable and converges one of the local maxima oft his Lyapunov function.
- As a result if we want to make sure that an $\mathbf{s}^{(\beta)}$ vector, taken out of the set of stored memory items S, is a steady state then it is not enough to have it as a fix point, but we also have to make sure that the Lyapunov function has a local maxima over $\mathbf{s}^{(\beta)}$, meaning

$$L(\mathbf{s}^{(\beta)}) > L(\mathbf{y}),\ \mathbf{y} \notin S.$$

Capacity Analysis (14)

Dynamic Capacity (4)

- First we deal with the Lyapunov function at the place $\mathbf{s}^{(\beta)}$

$$L\left(\mathbf{s}^{(\beta)}\right)=\left(\mathbf{s}^{(\beta)}\right)^{\mathrm{T}}\mathbf{W}\mathbf{s}^{(\beta)},$$

- which can be fully spelt out as follows

$$L\left(\mathbf{s}^{(\beta)}\right)=\sum_{i=1}^{N}\sum_{j=1}^{N}W_{ij}s_i^{(\beta)}s_j^{(\beta)},$$

- We can put instead of W_{ij} its definition,

$$L\left(\mathbf{s}^{(\beta)}\right)=\sum_{i=1}^{N}\sum_{j=1}^{N}s_i^{(\beta)}s_j^{(\beta)}\frac{1}{N}\sum_{\alpha=1}^{M}s_i^{(\alpha)}s_j^{(\alpha)}.$$

Capacity Analysis (15)

Dynamic Capacity (5)

- However these finite summations can be rearranged in the following way, if we collect the terms which depend on index *i* and *j*

$$L\left(\mathbf{s}^{(\beta)}\right)=\frac{1}{N}\sum_{\alpha=1}^{M}\left(\sum_{i=1}^{N}s_i^{(\beta)}s_i^{(\alpha)}\right)\left(\sum_{j=1}^{N}s_j^{(\beta)}s_j^{(\alpha)}\right)$$

- what one has to note that the summation with respect to i is the same as the summation with respect to j, as a result we can write this in a more compact form:

$$L\left(\mathbf{s}^{(\beta)}\right)=\frac{1}{N}\sum_{\alpha=1}^{M}\left(\sum_{i=1}^{N}s_i^{(\alpha)}s_i^{(\beta)}\right)^2.$$

Capacity Analysis (16)

Dynamic Capacity (6)

- It gives rise to the following formula, if we notice that there is the inner product between two memory items

$$L\left(\mathbf{s}^{(\beta)}\right)=\frac{1}{N}\sum_{\alpha=1}^{M}\left(\left(\mathbf{s}^{(\alpha)}\right)^{\mathrm{T}}\mathbf{s}^{(\beta)}\right)^{2}.$$

- However in the very first investigation we pointed out that the stored memory items should be orthogonal to each other. As a result when α sweeps through 1 to M, once α will hit β which can be singled out, giving the following expression

$$L\left(\mathbf{s}^{(\beta)}\right)=\frac{1}{N}\left(\left(\mathbf{s}^{(\beta)}\right)^{\mathrm{T}}\mathbf{s}^{(\beta)}\right)^{2}+\frac{1}{N}\sum_{\alpha=1,\alpha\neq\beta}^{M}\left(\left(\mathbf{s}^{(\alpha)}\right)^{\mathrm{T}}\mathbf{s}^{(\beta)}\right)^{2}.$$

Capacity Analysis (17)

Dynamic Capacity (7)

- Due to the orthogonality the second term is going to be zero, giving

$$L\left(\mathbf{s}^{(\beta)}\right)=\frac{1}{N}\left(\sum_{i=1}^{N}\left(s_i^{(\beta)}\right)^2\right)^2=\frac{1}{N}\left(\sum_{i=1}^{N}1\right)^2=\frac{1}{N}N^2=N.$$

- We have evaluated the left hand side of the inequality

$$L(\mathbf{s}^{(\beta)})>L(\mathbf{y}),\ \mathbf{y}\notin S,$$

- and now we are going to evaluate the right hand side, in a very similar manner:

Capacity Analysis (18)

Dynamic Capacity (8)

$$L(\mathbf{y}) = \mathbf{y}^{\mathrm{T}}\mathbf{W}\mathbf{y} = \sum_i \sum_j y_i y_j W_{ij} =$$

$$= \sum_i \sum_j y_i y_j \frac{1}{N} \sum_{\alpha=1}^{M} s_i^{(\alpha)} s_j^{(\alpha)} =$$

$$= \frac{1}{N} \sum_{\alpha=1}^{M} \left(\sum_i s_i^{(\alpha)} y_i \right) \left(\sum_j s_j^{(\alpha)} y_j \right) =$$

$$= \frac{1}{N} \sum_{\alpha=1}^{M} \left(\sum_i s_i^{(\alpha)} y_i \right)^2 = \frac{1}{N} \sum_{\alpha=1}^{M} \left(\left(\mathbf{s}^{(\alpha)} \right)^{\mathrm{T}} \mathbf{y} \right)^2 .$$

Capacity Analysis (19)

Dynamic Capacity (9)

- However we can say that if we have two binary vectors of dimension N and components -1 and 1 it can be verified that the inner product of two vectors $\mathbf{a}, \mathbf{b} \in \{-1, 1\}^N$ can be expressed by the means of Hamming distance as

$$\mathbf{a}^T\mathbf{b} = N - 2d(\mathbf{a},\mathbf{b}),$$

- where the Hamming distance $d()$ are the number of components in which the two binary vectors differ from each other. Using this equation we can rewrite $L(\mathbf{y})$ as follows

$$L(\mathbf{y}) = \frac{1}{N}\sum_{\alpha=1}^{M}\left(\left(\mathbf{s}^{(\alpha)}\right)^{\mathrm{T}}\mathbf{y}\right)^2 = \frac{1}{N}\sum_{\alpha=1}^{M}\left(N - 2d\left(\mathbf{s}^{(\alpha)},\mathbf{y}\right)\right)^2.$$

Capacity Analysis (20)

Dynamic Capacity (10)

- And now we are going to provide an upper-bound on this expression, taking into account that the minimum Hamming distance in which the state vector can differ from any stored memory items is 1. As a result we can write

$$L(\mathbf{y}) \le \frac{1}{N}\sum_{\alpha=1}^{M}(N-2)^2 = \frac{(N-2)^2}{N}M.$$

- And then we are done with because what we have obtained is the following

$$L(\mathbf{s}^{(\beta)}) > L(\mathbf{y}) \quad \rightarrow \quad N > \frac{(N-2)^2}{N}M \quad \rightarrow \quad M < \frac{(N-2)^2}{N^2}.$$

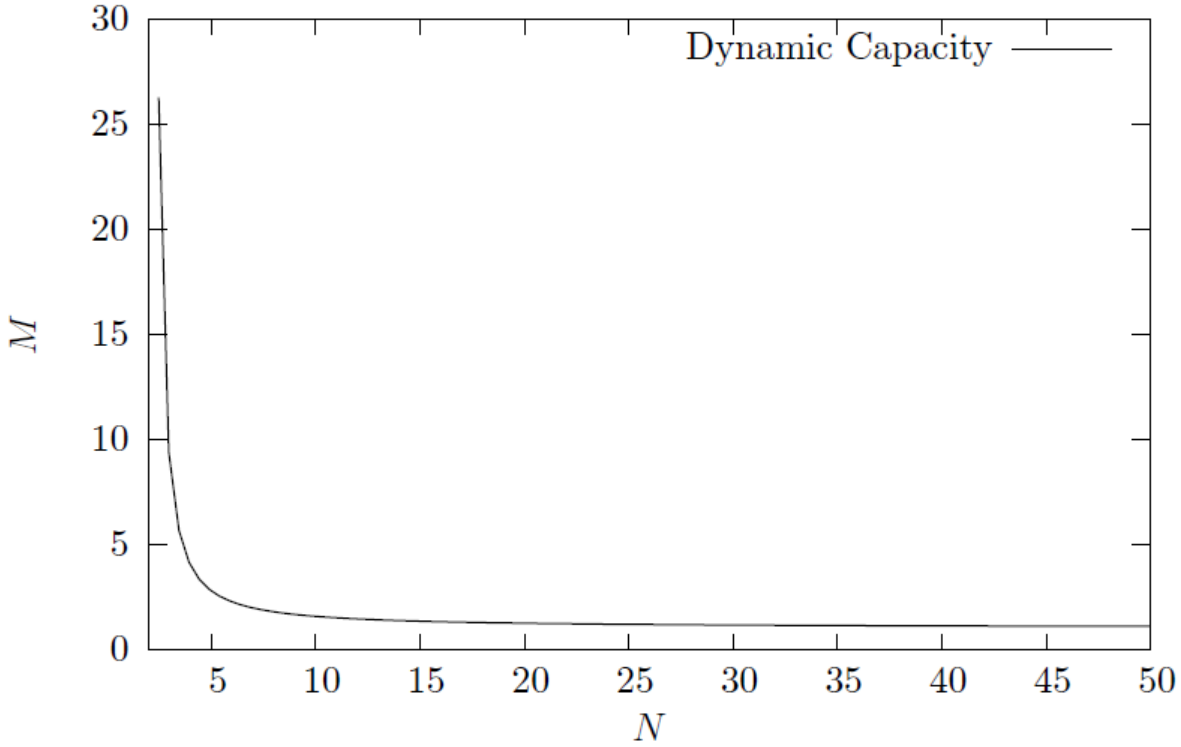
Capacity Analysis (21)

Dynamic Capacity (10)

- If this is fulfilled and indeed the stored memory items are going to be steady states, because this inequality holds and the Hopfield Network will converge to one of the local maxima of the Lyapunov function, and this local maxima is at the place of the stored memory item, as a result this is going to be a steady state.
- However this result is devastating because this capacity is very small, the number of stored memory items is very limited by this expression, asymptotically it converges to 1, giving an unusable associative mapping, capable of storing one memory item.

Capacity Analysis (22)

Dynamic Capacity (11)



- The Dynamic Capacity of the Hopfield Network.

Capacity Analysis (23)

Information Theoretical Capacity (1)

- As we have seen, the Dynamic Capacity of the Hopfield Neural Network tends to be one as N (the dimension of stored patterns) increases to infinity. This strongly discourages us in using these networks as associative memory.
- In this section we describe the solution to get out of this dead-lock. However this result is devastating because this capacity is very small, the number of stored memory items is very limited by this expression, asymptotically it converges to 1, giving an unusable associative mapping, capable of storing one memory item.

Capacity Analysis (24)

Information Theoretical Capacity (2)

- The underlying assumption when we investigate the IT capacity of the HNN is that we choose the stored memory patterns as random variables. As a result

$$s_i^{(\alpha)}, \ \alpha = 1, \ldots, M$$

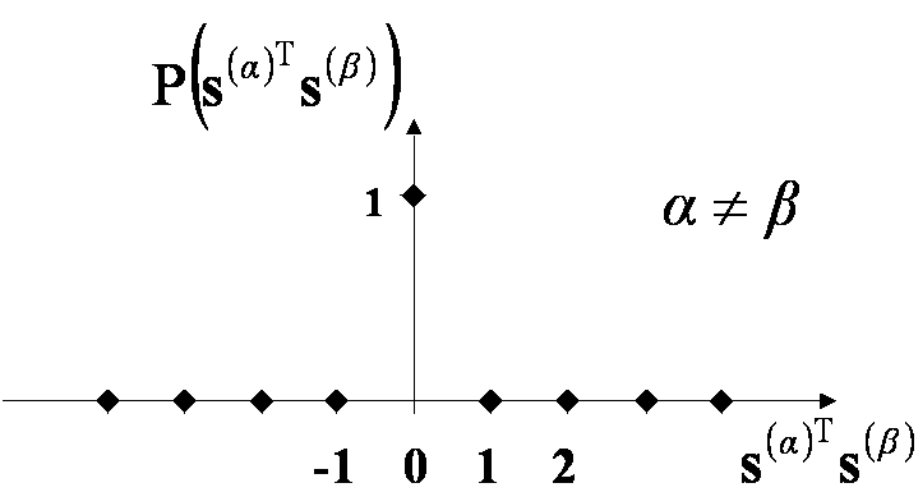
- is independent identically distributed Bernoulli random variable, with properties

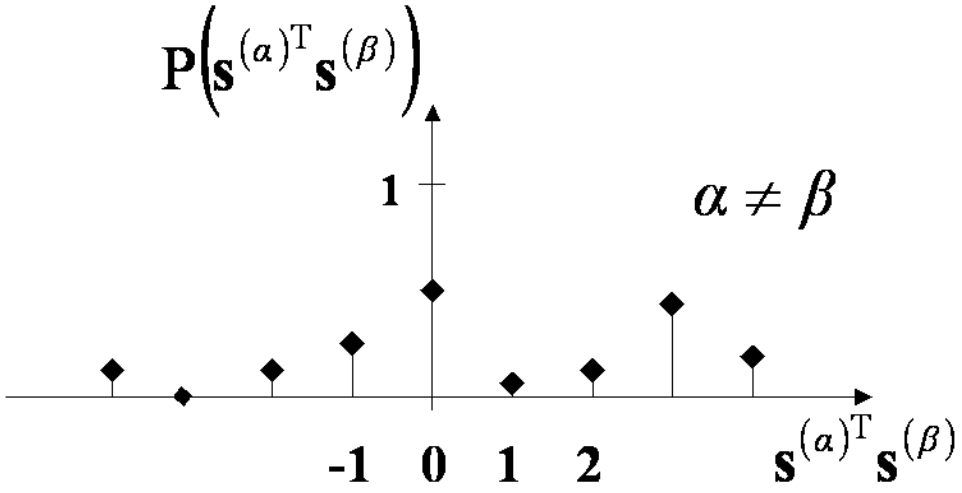
$$P\left(s_i^{(\alpha)} = 1\right) = P\left(s_i^{(\alpha)} = -1\right) = 0.5, \ \forall \alpha = 1, \ldots, M, \ \forall i = 1, \ldots, N.$$

Capacity Analysis (25)

Information Theoretical Capacity (3)

- Which entails that the stored memory patterns are chosen as random series of coin flipping.





- Orthogonal and random patterns.

Capacity Analysis (26)

Information Theoretical Capacity (4)

- What we are going to investigate is that if we take a memory vector $\mathbf{s}^{(\beta)}$ then is it a fix point of the HNN or not. This is a random event, because the components of $\mathbf{s}^{(\beta)}$ are chosen randomly.
- Furthermore applying the Hebbian Learning rule, the weight matrix is a random event as well.
- So we can investigate the probability of $\mathbf{s}^{(\beta)}$ being a fix point:

$$P\left(\mathbf{s}^{(\beta)} = \operatorname{sgn}\left(\mathbf{W}\mathbf{s}^{(\beta)}\right)\right).$$

Capacity Analysis (27)

Information Theoretical Capacity (5)

- **Definition 3.** Information Theoretical Capacity means, that what is the number of possible stored memory items M which guarantee that probability

$$P\left(\mathbf{s}^{(\beta)} = \operatorname{sgn}\left(\mathbf{W}\mathbf{s}^{(\beta)}\right)\right).$$

- asymptotically is going to be one, when N tends to be infinity.:

$$M: \lim_{N\to\infty} P\left(\mathbf{s}^{(\beta)} = \operatorname{sgn}\left(\mathbf{W}\mathbf{s}^{(\beta)}\right)\right) = 1$$

Capacity Analysis (28)

Information Theoretical Capacity (6)

- In order to evaluate this equation we have to perform some calculations with

$$P\left(\mathbf{s}^{(\beta)} = \operatorname{sgn}\left(\mathbf{W}\mathbf{s}^{(\beta)}\right)\right).$$

- which can be rewritten into the following form,

$$P\left(\mathbf{s}^{(\beta)} = \operatorname{sgn}\left(\mathbf{W}\mathbf{s}^{(\beta)}\right)\right) = P\left(\bigcap_{i=1}^{N} s_i^{(\beta)} = \operatorname{sgn}\left(\mathbf{W}\mathbf{s}^{(\beta)}\right)\right),$$

- because being a vector valued equation, and the starting probability holds if the equation holds for every components of the vector.

Capacity Analysis (29)

Information Theoretical Capacity (7)

$$P\left(\mathbf{s}^{(\beta)} = \operatorname{sgn}\left(\mathbf{W}\mathbf{s}^{(\beta)}\right)\right) = \prod_{i=1}^{N} P\left(s_i^{(\beta)} = \operatorname{sgn}\left(\sum_{j=1}^{N}\frac{1}{N}\sum_{\alpha=1}^{M} s_i^{(\alpha)} s_j^{(\alpha)} s_j^{(\beta)}\right)\right),$$

- holds due to the fact, that the components of $\mathbf{s}^{(\beta)}$ are independent random variables, and we have replaced W_{ij} with its definition, which is coming from the Hebbian Learning Rule.
- Restructuring this double summation we get to the following:

$$P\left(\mathbf{s}^{(\beta)} = \operatorname{sgn}\left(\mathbf{W}\mathbf{s}^{(\beta)}\right)\right) = \prod_{i=1}^{N} P\left(s_i^{(\beta)} = \operatorname{sgn}\left(\sum_{\alpha=1}^{M} s_i^{(\alpha)}\frac{1}{N}\sum_{j=1}^{N} s_j^{(\alpha)} s_j^{(\beta)}\right)\right).$$

Capacity Analysis (30)

Information Theoretical Capacity (8)

- Since $\mathbf{s}^{(\beta)}$ is among the stored memory patterns once α is going to hit β and we can single out that case in the summation, which results in

$$P\left(\mathbf{s}^{(\beta)} = \operatorname{sgn}\left(\mathbf{W}\mathbf{s}^{(\beta)}\right)\right) = \prod_{i=1}^{N} P\left(s_i^{(\beta)} = \operatorname{sgn}\left(s_i^{(\beta)} \frac{1}{N}\sum_{j=1}^{N}\left(s_j^{(\beta)}\right)^2 + \sum_{\alpha=1,\alpha\neq\beta}^{M} s_i^{(\alpha)} \frac{1}{N}\sum_{j=1}^{N} s_j^{(\alpha)} s_j^{(\beta)}\right)\right),$$

- where

$$\frac{1}{N}\sum_{j=1}^{N}\left(s_j^{(\beta)}\right)^2 = \frac{1}{N} N = 1.$$

- because we are dealing with binary vectors with values $\{+1,-1\}$, and raising these values to the square we always get $+1$, which added up N times gives N.

Capacity Analysis (31)

Information Theoretical Capacity (9)

- Let us introduce a new variable

$$v_i = \sum_{\alpha=1,\alpha\neq\beta}^{M} s_i^{(\alpha)} \frac{1}{N} \sum_{j=1}^{N} s_j^{(\alpha)} s_j^{(\beta)}.$$

- v_i is a random variable as well, because the multiplication of Bernoulli random variables results in a Bernoulli random variable. What is in this expression is a double summation of Bernoulli random variables.

Capacity Analysis (32)

Information Theoretical Capacity (10)

- Due to the Central Limit Theorem, this approximates a Gaussian random variable

$$v_i \sim N\left(0, \sqrt{\frac{M-1}{N}}\right),$$

- since the mean of each Bernoulli random variable is zero, the mean of the Gaussian random variable is zero as well, and because of the standard deviation of the Bernoulli random variables is one, and we add it up $M-1$ times and normalize it with N the standard deviation of the Gaussian random variable is going to be

$$\sqrt{\frac{M-1}{N}}.$$

Capacity Analysis (33)

Information Theoretical Capacity (11)

- Furthermore since we have delved into an asymptotic investigation when N tends to be infinity if we replace $M-1$ with M it does not make any difference, so finally we get

$$v_i \sim N\left(0, \sqrt{\frac{M}{N}}\right).$$

- What we have arrived at is:

$$P\left(\mathbf{s}^{(\beta)} = \operatorname{sgn}\left(\mathbf{W}\mathbf{s}^{(\beta)}\right)\right) = \prod_{i=1}^{N} P\left(s_i^{(\beta)} = \operatorname{sgn}\left(s_i^{(\beta)} + v_i\right)\right).$$

Capacity Analysis (34)

Information Theoretical Capacity (12)

- It can be expressed by the means of conditional probabilities, namely

$$P\left(\mathbf{s}^{(\beta)} = \operatorname{sgn}\left(\mathbf{W}\mathbf{s}^{(\beta)}\right)\right) = \prod_{i=1}^{N}\left\{P\left(1 = \operatorname{sgn}\left(1+v_i\right) \mid s_i^{(\beta)} = 1\right)P\left(s_i^{(\beta)} = 1\right) + \right.$$

$$\left. + P\left(-1 = \operatorname{sgn}\left(-1+v_i\right) \mid s_i^{(\beta)} = -1\right)P\left(s_i^{(\beta)} = -1\right)\right\},$$

where we know, that

$$P\left(s_i^{(\beta)} = 1\right) = P\left(s_i^{(\beta)} = -1\right) = 0.5.$$

Capacity Analysis (35)

Information Theoretical Capacity (13)

- We can rewrite these probabilities taking into account, that the sgn() function gives +1 for $(1 + v_i)$ if $(1 + v_i) > 0$ holds, which results in

$$P\left(\mathbf{s}^{(\beta)} = \operatorname{sgn}\left(\mathbf{W}\mathbf{s}^{(\beta)}\right)\right) = \prod_{i=1}^{N}\left\{P\left(1+v_i>0\right)+P\left(-1+v_i<0\right)\right\} =$$

$$= \prod_{i=1}^{N} 0.5\cdot\left\{\mathrm{P}\left(v_i<1\right)+\mathrm{P}\left(v_i>-1\right)\right\}.$$

- Now we can take advantage of that fact that v_i is Gaussian:

$$P\left(\mathbf{s}^{(\beta)} = \operatorname{sgn}\left(\mathbf{W}\mathbf{s}^{(\beta)}\right)\right) = \prod_{i=1}^{N} 0.5\left\{1-\Phi\left(\sqrt{\frac{N}{M}}\right)+\Phi\left(\sqrt{\frac{N}{M}}\right)\right\} = \prod_{i=1}^{N}\Phi\left(\sqrt{\frac{N}{M}}\right) = \Phi^{N}\left(\sqrt{\frac{N}{M}}\right).$$

Capacity Analysis (36)

Information Theoretical Capacity (14)

- $\Phi(u)$ is the cumulative distribution function of the random variable v_i, with point symmetry property $1 - \Phi(-u) = \Phi(u)$.

$$\Phi(u) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{u} exp\left(\frac{-u^2}{2}\right).$$

- What we have arrived to is the following

$$P\left(\mathbf{s}^{(\beta)} = \mathrm{sgn}\left(\mathbf{W}\mathbf{s}^{(\beta)}\right)\right) = \Phi^N\left(\sqrt{\frac{N}{M}}\right),$$

- which is a simple formula to investigate this probability.

Capacity Analysis (37)

Information Theoretical Capacity (15)

- In Definition 3. we are investigating that under what condition this probability tends to be one. However this is equivalent with investigating the logarithm of this probability when tends to zero:

$$\ln\left\{P\left(\mathbf{s}^{(\beta)} = \operatorname{sgn}\left(\mathbf{W}\mathbf{s}^{(\beta)}\right)\right)\right\} = 0$$

- which is equivalent with

$$N \ln\left\{\Phi\left(\sqrt{\frac{N}{M}}\right)\right\} = 0.$$

Capacity Analysis (38)

Information Theoretical Capacity (16)

- Remind the following facts from asymptotic analysis:

$$\lim_{u\to\infty}\Phi(u)\approx 1-\frac{1}{u}\exp\left(-\frac{1}{2}u^2\right) \text{ and } \lim_{u\to\infty}\ln(u)\approx 1-u.$$

- Having these approximations at hand we can see that if N tends to be infinity and we reverse the fraction in the argument of Φ(u) then Φ(u) tends to have a very large argument, as a result we can approximate with formula above, which gives

$$N\ln\left\{\Phi\left(\sqrt{\frac{N}{M}}\right)\right\}=N\ln\left\{1-\sqrt{\frac{M}{N}}\exp\left(-\frac{1}{2}\left(\sqrt{\frac{M}{N}}\right)^2\right)\right\}.$$

Capacity Analysis (39)

Information Theoretical Capacity (17)

- Now the argument of the logarithm function tends to zero when N goes to infinity, and because of this we can use

$$\lim_{u\to\infty} \ln(u) \approx 1-u.$$

- to further rewrite this expression

$$N \ln\left\{\Phi\left(\sqrt{\frac{N}{M}}\right)\right\} = N\sqrt{\frac{M}{N}} \exp\left(-\frac{N}{2M}\right) = \exp\left(-\frac{N}{2M} + \ln(N) - \frac{1}{2}\ln\left(\frac{N}{M}\right)\right).$$

Capacity Analysis (40)

Information Theoretical Capacity (18)

- Having this result, we have to find M for which this expression tends to zero, because we have evaluated the logarithm of the initial probability, and this logarithmic probability should tend to zero, if we want the probability to tend to one.
- We can see that we are not in trouble to make this zero, if we choose

$$\frac{N}{2M} = \ln(N) \quad \longrightarrow \quad M = \frac{N}{2\ln(N)}.$$

Capacity Analysis (41)

Statistical Neurodynamics (1)

- When deriving the IT capacity of the HNN we used only a fix-point analysis, we did not pay attention of the dynamics of the network. Still we have to investigate the dynamics of the HNN, whether we really will converge to the stored memory items if they are chosen to be Bernoulli random variables.
- Statistical Neurodynamics relies on the theory of Statistical Physics where we investigate very huge systems, containing a lot of particles (for example the Oxygen atoms in a room).

Capacity Analysis (42)

Statistical Neurodynamics (2)

- Then we can characterize the state of the system by micro-states $\mathbf{y}(k)$, but this description is meaningless, because it contains too much information, does not reveal any important property of the system.
- However in order to really characterize the statistical system the concept of macro-states can be developed, where the macro-state

$$a(k) = \text{average}\left(\varphi\left(\mathbf{y}(k)\right)\right).$$

Capacity Analysis (43)

Statistical Neurodynamics (3)

- And now we can investigate that if we know the state transition rule between the micro-states, Statistical Physics wants to derive the state transition rule for the macro-states.

$$\overline{y}(k+1) = \operatorname{sgn}\left\{\overline{\overline{W}}\,\overline{y}(k)\right\} \quad \Rightarrow \quad a(k+1) = \Psi\left(a(k)\right)$$

- For example if we take a room, we can observe the position of Oxygen atoms in there. A micro-state of this system would be characterized by giving the coordinates of each Oxygen atoms, which would be so much data that we could not retrieve any meaningful information from it.

Capacity Analysis (44)

Statistical Neurodynamics (4)

- However defining macro-states as what is the average distribution of Oxygen atoms, then this is a meaningful information.
- And when we are investigating the state-transmission, then for example if we turn up the heating at one corner, then we know from the basic rules of physics how the micro-states will change and then we are investigating how the macro-state will change, how the temperature will effect the average distribution of the Oxygen atoms.

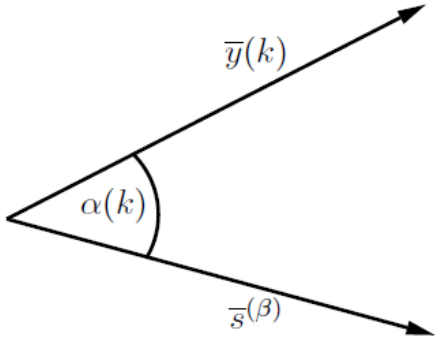
Capacity Analysis (45)

Statistical Neurodynamics (5)

- We define the macro state of the network as

$$a(k) = \frac{1}{N}\overline{s}^{(\beta)}\overline{y}(k) = \frac{1}{N}\sum_{i=1}^{N} s_i^{(\beta)} y_i(k)$$

- The meaning of this macro state can be graphically represented in next Figure.



Representation of the macro state.

Capacity Analysis (46)

Statistical Neurodynamics (6)

- Here the macro state $a(k)$ corresponds to $\cos(\alpha(k))$. If we can prove that for a given (u) $a(k+1) > a(k)$ and $a(k)$ converges to 1, than it means that the cosine of $\alpha(k)$ converges to 1, which means that $\alpha(k)$ converges to 0. Consequently if $\alpha(k)$ tends to 0, this means that the micro state $y(k)$ converges to s().

Capacity Analysis (47)

Statistical Neurodynamics (7)

- Throughout these discussions we restrict ourselves to s() = (1, 1, . . . , 1), for the sake of easier formulas, but these results can be also derived for any arbitrary patterns. In this case if we consider

$$a(k) = \frac{1}{N}\overline{s}^{(\beta)}\overline{y}(k) = \frac{1}{N}\sum_{i=1}^{N} s_i^{(\beta)} y_i(k)$$

- when s() is this special vector, we get

$$a(k) = \frac{1}{N}\sum_{i=1}^{N} y_i(k),$$

Capacity Analysis (48)

Statistical Neurodynamics (8)

- It is an empirical average of the random variable yi(k), which approximates the expected value E{yi(k)} of this random variable, due to the Law of Large Numbers.
- We would like to derive the macro state transition rule, and for this we start by writing down the 0th macro state:

$$a(0) = \frac{1}{N}\sum_{i=1}^{N} y_i(0)$$

- after this we take the 1st macro state, which is as follows:

$$a(1) = \frac{1}{N}\sum_{i=1}^{N} y_i(1) \sim \mathrm{E}y_i(1),$$

Capacity Analysis (49)

Statistical Neurodynamics (9)

- It is an empirical average of random variables, because Wij is chosen according to the Hebbian Learning Rule, and s() i is subject to Bernoulli distribution, that is why yi(1) is a Bernoulli random variable. When we further elaborate on this expected value we can write

$$\mathrm{E}y_i(1) = 1 \cdot \mathrm{P}\left(y_i(1) = 1\right) + (-1) \cdot \mathrm{P}\left(y_i(1) = -1\right) =$$

$$= \mathrm{P}\left(\operatorname{sgn}\left\{\sum_{j=1}^{N} W_{ij} y_j(0)\right\} = 1\right) - \mathrm{P}\left(\operatorname{sgn}\left\{\sum_{j=1}^{N} W_{ij} y_j(1)\right\} = -1\right) =$$

Capacity Analysis (50)

Statistical Neurodynamics (10)

- After this we can substitute the definition of Wij , which comes from the Hebbian Learning Rule we applied

$$= \mathrm{P}\left(\mathrm{sgn}\left\{\sum_{j=1}^{N}\frac{1}{N}\sum_{\alpha=1}^{M}s_i^{(\alpha)}s_j^{(\alpha)}y_j(0)\right\}=1\right) -$$

$$\mathrm{P}\left(\mathrm{sgn}\left\{\sum_{j=1}^{N}\frac{1}{N}\sum_{\alpha=1}^{M}s_i^{(\alpha)}s_j^{(\alpha)}y_j(0)\right\}=-1\right) =$$

- the two summations can be reordered, which yields

$$= \mathrm{P}\left(\mathrm{sgn}\left\{\sum_{\alpha=1}^{M}s_i^{(\alpha)}\frac{1}{N}\sum_{j=1}^{N}s_j^{(\alpha)}y_j(0)\right\}=1\right) -$$

$$\mathrm{P}\left(\mathrm{sgn}\left\{\sum_{\alpha=1}^{M}s_i^{(\alpha)}\frac{1}{N}\sum_{j=1}^{N}s_j^{(\alpha)}y_j(0)\right\}=-1\right) =$$

Capacity Analysis (51)

Statistical Neurodynamics (11)

- The first summation here sweeps through all the possible values of $\mathbf{s}^{(i)}$, which contains $\mathbf{s}^{(\beta)}$as well, and because of this we can single this term out, and write

$$= \mathrm{P}\left(\mathrm{sgn}\left\{s_i^{(\beta)}\frac{1}{N}\sum_{j=1}^{N}s_j^{(\beta)}y_j(0)+\sum_{\alpha=1,\alpha\neq\beta}^{M}s_i^{(\alpha)}\frac{1}{N}\sum_{j=1}^{N}s_j^{(\alpha)}y_j(0)\right\}=1\right)-$$

$$\mathrm{P}\left(\mathrm{sgn}\left\{s_i^{(\beta)}\frac{1}{N}\sum_{j=1}^{N}s_j^{(\beta)}y_j(0)+\sum_{\alpha=1,\alpha\neq\beta}^{M}s_i^{(\alpha)}\frac{1}{N}\sum_{j=1}^{N}s_j^{(\alpha)}y_j(0)\right\}=-1\right)=$$

- and because we have chosen $\mathbf{s}^{(\beta)}$in a special way, namely $\mathbf{s}^{(\beta)}=(1,1,1,\ldots,1)$ we can write

$$s_i^{(\beta)}\frac{1}{N}\sum_{j=1}^{N}s_j^{(\beta)}y_j(0)=1\cdot\frac{1}{N}\cdot\sum_{j=1}^{N}1\cdot y_j(0)=a(0),$$

Capacity Analysis (52)

Statistical Neurodynamics (12)

- It was the value of the macro state at the 0^{th} time instance.
- Elaborating on the second term of the summation we can see that it is a normalized summation of Bernoulli random variables, and due to the Central Limit Theorem this will approximate a Gaussian random variable vi in the following way

$$\sum_{\alpha=1,\alpha\neq\beta}^{M} s_i^{(\alpha)} \frac{1}{N} \sum_{j=1}^{N} s_j^{(\alpha)} y_j(0) = \nu_i \quad \sim \quad \mathcal{N}\left(0, \sqrt{\frac{M-1}{N}}\right).$$

Capacity Analysis (53)

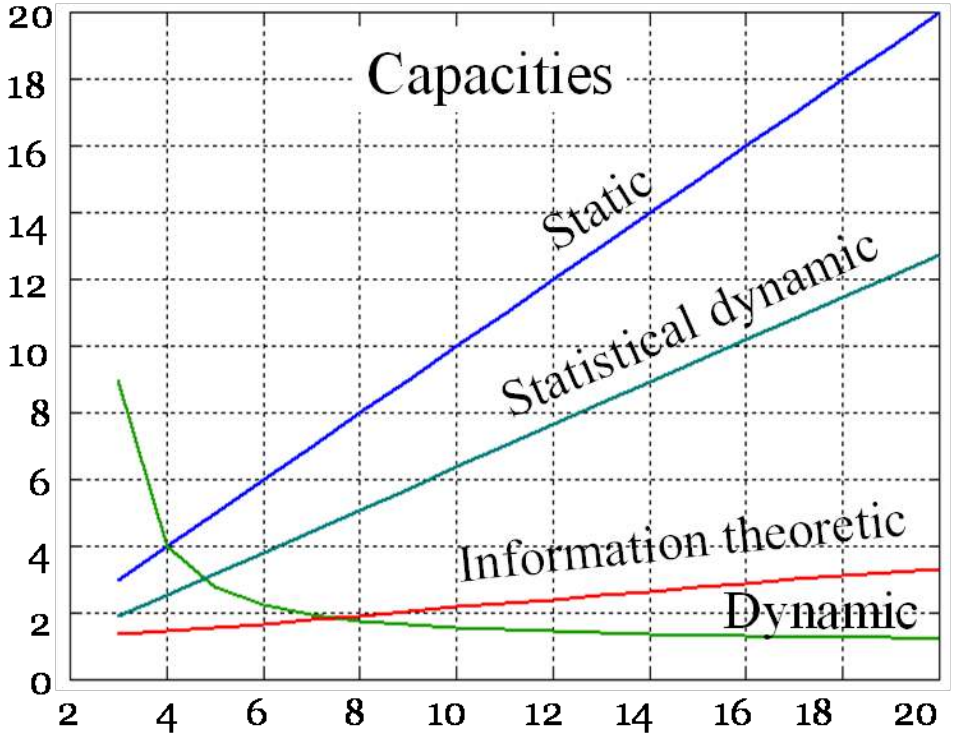
Statistical Neurodynamics (13)

- Considering the fact, that we are interested in the asymptotic behavior of the Hopfield Network, when N tends to infinity, then writing M instead of M −1 does not make any difference, which gives us

$$\nu_i \sim \mathcal{N}\left(0, \sqrt{\frac{M}{N}}\right)$$

Capacity Analysis (54)

Summary of the capacity



Capacity limits of Hopfield net.

$$\mathrm{Cap}^{\mathrm{stat}} = M = N$$

$$\mathrm{Cap}^{\text{statistical dynamic}} = M < \frac{2}{\pi} N$$

$$\mathrm{Cap}^{\mathrm{Inf.theo}} = M = \frac{N}{2\log(N)}$$

$$\mathrm{Cap}^{\mathrm{din}} = M \leq \frac{N^2}{(N-2)^2}$$

Outline of applying the HNN as a combinatorial optimizer

- Defining the problem set of combinatorial optimization
- Binary Quadratic programming as a combinatorial optimization problem
- HNN as a combinatorial optimizer
  - Philosophy
  - Minimization modification
  - Constraint mapping
- Travelling salesman problem
  - Problem formulation
  - Cost function as a quadratic function
  - Constraints as linear combination of penalty functions

Outline of applying the HNN as a combinatorial optimizer

- TSP problem mapped into a quadratic form
- Solution with HNN

- HNN for ISI corrupted signal detection
  - Problem formulation
  - Mapping the problem into a quadratic form
  - Solution with HNN
  - Performance analysis
- Examples
- Summary

Combinatorial optimization task

- A discrete optimization task is an optimization task:

$$\min f(x)$$

$$\text{subject to } g_i(x) \lhd_i 0, i = 1, \ldots, M$$

$$\lhd_i \in \{=, \leq, \geq, <, >\}$$

where the domain of the objective function and the constraints are from a discrete set e.g. $x \in$ discrete points in space or

$x \in$ vertices of a graph or

$x \in$ edges of a graph or

$x \in \mathbb{N}$

etc.

Combinatorial optimization task

- A combinatorial optimization problem COP is a discrete optimization task where the domain of the functions is also a discrete set, but the elements are combinations of simpler elements. E.g.

  $x \in$ group of vertices in a graph or

  $x \in$ group of edges in a graph, like a tree or a cycle

  $x \in \mathbb{N}^n$ vector with poz. integer elements

  etc.

- The trivial solution for a discrete optimization problem is the exhaustive search.
- Usually the combinatorial optimization tasks are NP problems, so for a large size the exhaustive search is intractable.

Hopfield network as a combinatorial optimizer

- We have learned that the HNN minimizes it's energy function which is:

$$L(\mathbf{y}) = \mathbf{y}^T \mathbf{W}\mathbf{y} - 2\mathbf{y}^T \mathbf{b}, \quad \mathbf{y} \in \{-1,1\}^N$$

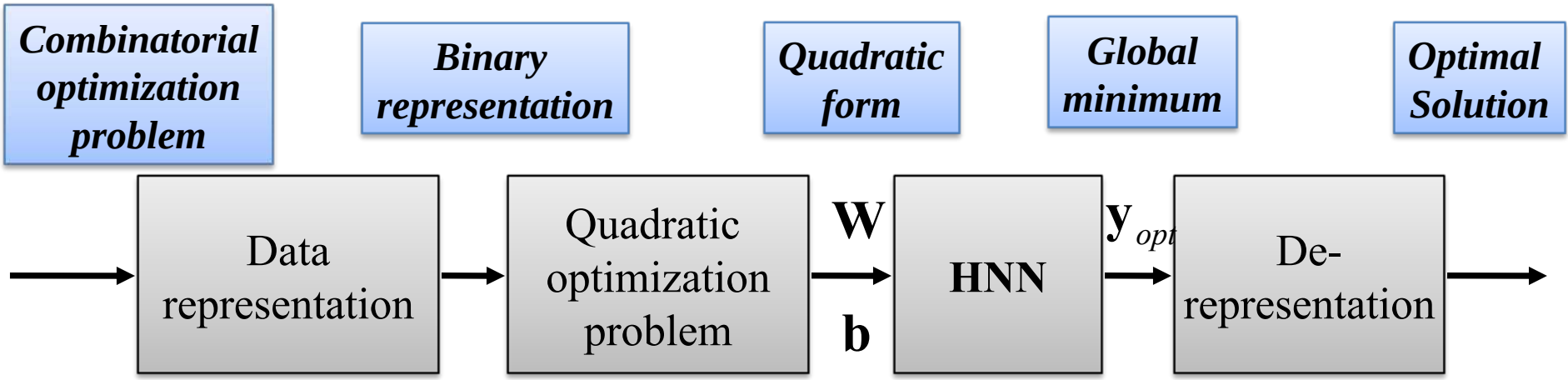
- Note that if we don't assume any constraints or incorporate the constraints in the energy function and choose the objective function to be the energy function, then the HNN can perform the combinatorial optimization.

$$\min_{\mathbf{y} \in \{-1,+1\}^N} L(\mathbf{y})$$

- So if we can address a combinatorial optimization problem as a quadratic function minimization, then a HNN can be used.

HNN as a combinatorial optimizer – philosophy

- We can approximate the solution of a traditionally NP problem



O(2^N) or O(N!)

$$\mathbf{y}_{opt} = \min_{\mathbf{y}\in\{-1,1\}^N} \mathbf{y}^T\mathbf{W}\mathbf{y} - 2\mathbf{b}^T\mathbf{y}$$

O(N²)

HNN as a combinatorial optimizer – minimization

- We have used the HNN for maximizing the energy function, we will prove that with a modification it can be used for minimization.
- Modify the weight matrix as:

$$W_{ij}^{\Diamond} = \begin{cases} W_{ij}, & \text{if } i \neq j \\ 0, & \text{if } i = j \end{cases}$$

- The new energy function is: $\mathbf{y}^{\Diamond}_{opt} = \min_{\mathbf{y} \in \{-1,1\}^N} \mathbf{y}^T \mathbf{W}^{\Diamond} \mathbf{y} - 2\mathbf{b}^T \mathbf{y}$

- It can be proven that: $\mathbf{y}^{\Diamond}_{opt} = \mathbf{y}_{opt}$

HNN as a combinatorial optimizer – minimization

- The two energy functions only differ in a constant term so the minimum is the same location:

$$\mathbf{y}^T \mathbf{W}^{\Diamond} \mathbf{y} - 2\mathbf{b}^T \mathbf{y} + C = \mathbf{y}^T \mathbf{W} \mathbf{y} - 2\mathbf{b}^T \mathbf{y}$$

$$\mathbf{y}^T \mathbf{W}^{\Diamond} \mathbf{y} - 2\mathbf{b}^T \mathbf{y} + C = \mathbf{y}^T \left(\mathbf{W}^{\Diamond} + \mathbf{W} - \mathbf{W}^{\Diamond} \right) \mathbf{y} - 2\mathbf{b}^T \mathbf{y}$$

$$\mathbf{y}^T \mathbf{W}^{\Diamond} \mathbf{y} - 2\mathbf{b}^T \mathbf{y} + C = \mathbf{y}^T \mathbf{W}^{\Diamond} \mathbf{y} - 2\mathbf{b}^T \mathbf{y} + \mathbf{y}^T \left(\mathbf{W} - \mathbf{W}^{\Diamond} \right) \mathbf{y}$$

$$C = \mathbf{y}^T \left(\mathbf{W} - \mathbf{W}^{\Diamond} \right) \mathbf{y} = \sum_{i=1}^{N} \sum_{j=1}^{N} \underbrace{\left(W_{ij} - W_{ij}^{\Diamond} \right) y_j y_i}_{\substack{0 \text{ if } i \neq j \\ W_{ij} \text{ if } i = j}}$$

$$C = \sum_{i=1}^{N} W_{ii} \underbrace{y_i y_i}_{1} = \sum_{i=1}^{N} W_{ii}$$

HNN as a combinatorial optimizer – minimization

- The modified energy function does not alter the place of the minimum, so we can define a state transition rule which minimizes the new energy function:

$$y_i(k+1) = -\operatorname{sgn}\left\{\sum_{j=1}^{N} W_{ij}^{\Diamond} y_j(k) - b_i\right\}, \quad i = \operatorname{mod}_N(k)$$

- The Modified HNN is capable of minimizing the quadratic form and the number of steps needed for this minimization is a polynomial function of the dimension of the network.

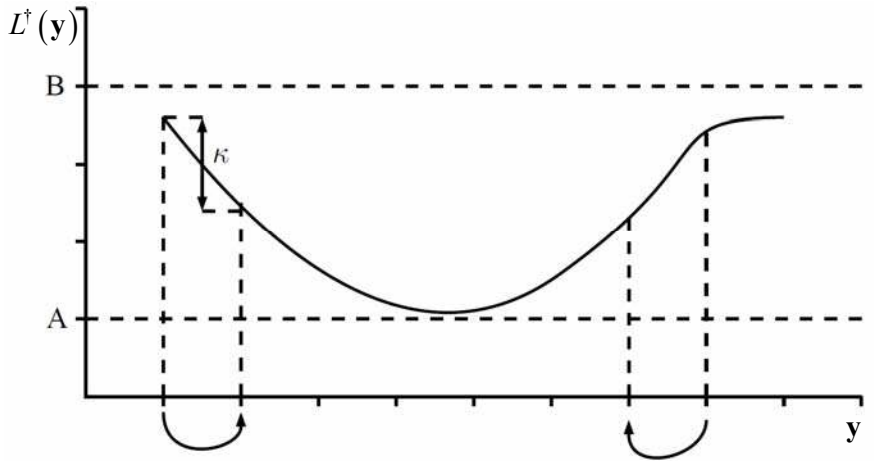
$$O\left(N^2\right) \ll O\left(2^N\right)$$

HNN as a combinatorial optimizer – minimization

- To prove this we have to ensure 3 properties for this structure:

1. there exists a global upper-bound B;
2. there exists a global lower-bound A;
3. the change in the Lyapunov function is always larger than κ

$$\Delta L^{\Diamond}(k) := L^{\Diamond}(\mathbf{y}(k+1)) - L^{\Diamond}(\mathbf{y}(k)) < -\kappa$$



HNN as a combinatorial optimizer – minimization

- If properties shown we can state that the transient time is:

$$\mathrm{TR} \leq \frac{B - A}{\kappa}$$

- Following the methods shown at the maximization the lower and upper bounds can be derived similarly:

$$L^{\Diamond}(\mathbf{y}) \leq N \left\| \mathbf{W}^{\Diamond} \right\| + 2\sqrt{N} \left\| \mathbf{b} \right\| = B$$

$$L^{\Diamond}(\mathbf{y}) \geq -\mathbf{b}^{T} \mathbf{W}^{\Diamond -1} \mathbf{b} = A$$

- We have to elaborate on the change of the energy function to prove property 3.

HNN as a combinatorial optimizer – minimization

- Rewriting the change of the energy function:

$$\Delta L^{\Diamond}(k) = L^{\Diamond}(\mathbf{y}(k+1)) - L^{\Diamond}(\mathbf{y}(k)) = \Delta y_i W_{ii}^{\Diamond} + 2\Delta y_i(k)\left\{\sum_{j=1}^{N} W_{ij}^{\Diamond} y_j(k) - b_i\right\}$$

where $\Delta y_i(k) := y_i(k+1) - y_i(k)$ and $i = \mathrm{mod}_N(k)$

due to $W_{ii}^{\Diamond} = 0$

$$\Delta L^{\Diamond}(k) = 2\Delta y_i(k)\left\{\sum_{j=1}^{N} W_{ij}^{\Diamond} y_j(k) - b_i\right\}$$

- To show that the change is bounded we introduce a table with the possible events:

HNN as a combinatorial optimizer – minimization

- Table of the possible changes in the energy function:

$y_i(k)$	$y_i(k+1)$	$\Delta y_i(k)$	$\sum_{j=1}^{N} W_{ij}^{\Diamond} y_j(k) - b_i$	$\Delta L^{\Diamond}(k)$
-1	1	2	$\leq -\beta < 0$	-4β
1	-1	-2	$\geq \beta > 0$	-4β

- So the energy function always decreases which yields to a minimum point.
- We can give a bound on a transition time:

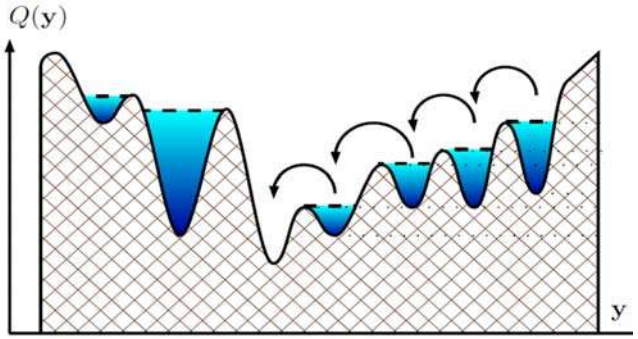
$$\mathrm{TR} \leq \frac{N\left\|\mathbf{W}^{\Diamond}\right\| + 2\sqrt{N}\left\|\mathbf{b}\right\| + \mathbf{b}^{T}\mathbf{W}^{\Diamond^{-1}}\mathbf{b}}{4\beta} N$$

HNN as a combinatorial optimizer – minimization

Local vs. global optima

- The Hopfield network acts along the energy "surface" function and in every step decreases it. However if a valley other than the basin of the desired solution (marked with blue) exist in the energy function, it can stuck in a suboptimal answer if started from a "wrong" basin.
- In the energy function we call the bottom of such valleys local minima. Local minima usually have higher value than the global minimum.

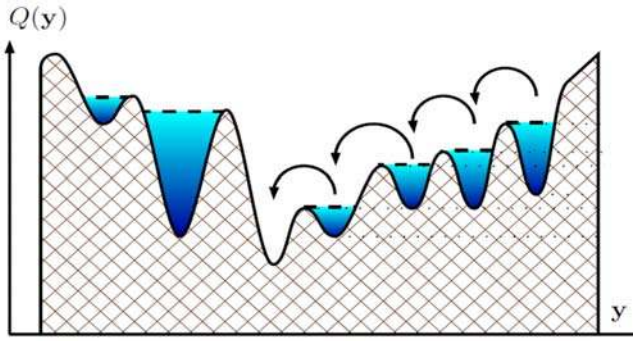
$$\min_{\mathbf{y}\in\{-1,1\}^N} \mathbf{y}^T\mathbf{W}\mathbf{y} - 2\mathbf{b}^T\mathbf{y}$$



HNN as a combinatorial optimizer – minimization

Strategies overcoming staying in local minima

- There are a lot of different strategies to overcome stucking in a local minima. A lot of them are heuristics or combined with other combinatorial optimizer strategies.
- A few examples:
  - If stuck, "shake" ~ add noise to the state, maybe it shakes into a "better" valley – Noisy Hopfield Network
  - Chaotic Hopfield Network
  - Taboo search combined HNN
  - EDA+HNN
  - Hysteretic HNN
  - Multi stage, multi init. HNN, etc.



HNN as a combinatorial optimizer – constraint mapping

- We have seen that the HNN is capable to minimize a quadratic function

$$\min f(x)$$

$$\text{subject to } g_i(x) \lhd_i 0, i = 1, \ldots, M$$

$$\lhd_i \in \{=, \leq, \geq, <, >\}$$

$$f(x) = L(\mathbf{y}) = \mathbf{y}^T \mathbf{W} \mathbf{y} - 2\mathbf{y}^T \mathbf{b}$$

- But we cannot deal with the constraints in a straightforward manner.
- The most common way to deal with the constraints is to incorporate them into the energy function as penalizing terms

HNN as a combinatorial optimizer – constraint mapping

- Constructing an objective function as a linear combination of the original cost function and the penalizing terms

$$\min \alpha_0 f(x) + \sum_{i=1}^{M} \alpha_i p_i(x)$$

$$\text{where } p_i(x) := \begin{cases} > 0, & \text{if } g_i(x) \lhd_i 0 \text{ constraint not met} \\ 0, & \text{if } g_i(x) \lhd_i 0 \text{ satisfied} \end{cases}$$

$$f(x) = L(\mathbf{y}) = \mathbf{y}^T \mathbf{W} \mathbf{y} - 2\mathbf{y}^T \mathbf{b}, \quad \mathbf{y} \in \{-1, +1\}^N$$

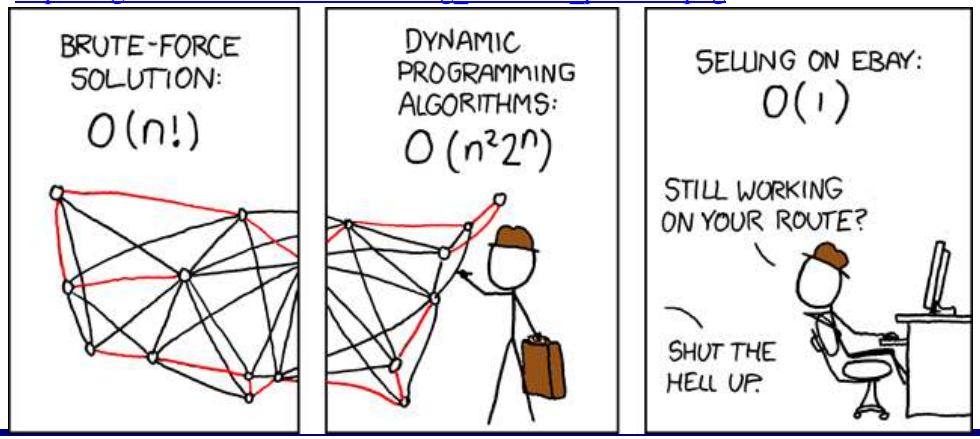
- Although other construction could be used due to the linearity property the linear combination is the most commonly used choice.

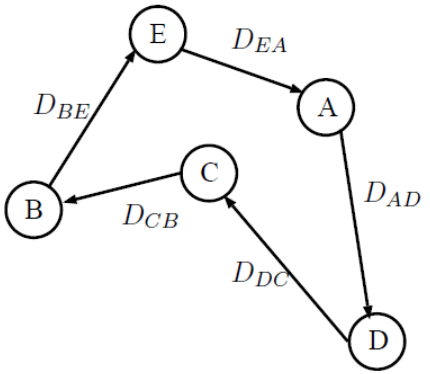
Travelling salesman COP

- One of the most important COPs is the Travelling Salesman Problem (TSP).
- We have *K* cities. The TSP is that we have an agent at city 1 and we want him to visit all the cities exactly only once and arrive back to city 1 while we require him to travel the least (on the possible shortest route)

Travelling salesman COP

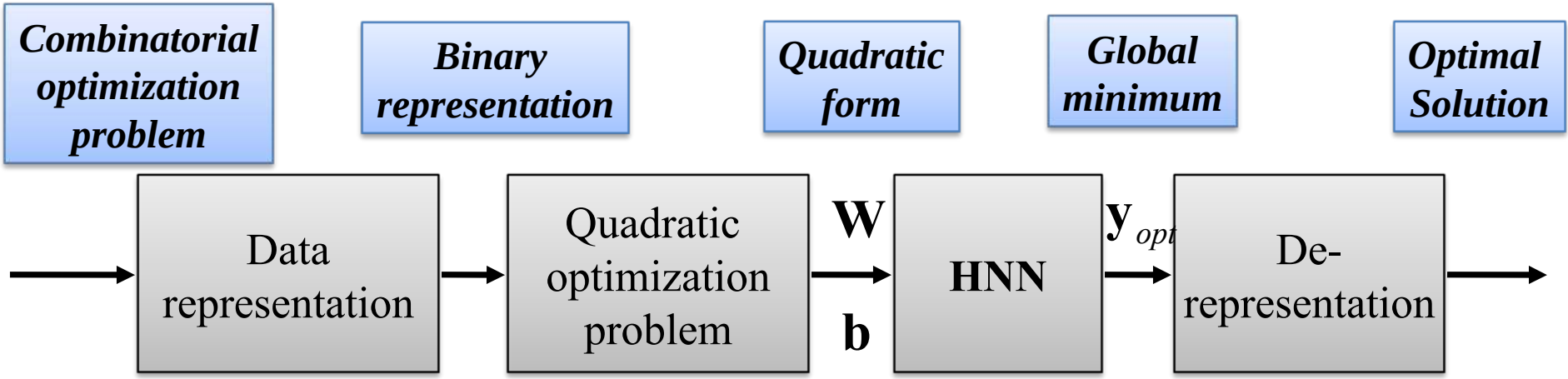
- We have an edge weighted graph G(V,E) where the vertices are representing the cities the edges are the possible travelling connections between the cities and the distance between the cities (travelling costs) are the weights of the edges.
- The TSP problem is the same as finding the shortest Hamiltonian cycle in an edge weighted graph
- http://imgs.xkcd.com/comics/travelling_salesman_problem.png





Travelling salesman COP

- Let us follow the steps of



O(2^N) or O(N!) $\mathbf{y}_{opt} = \min_{\mathbf{y}\in\{-1,1\}^N} \mathbf{y}^T\mathbf{W}\mathbf{y} - 2\mathbf{b}^T\mathbf{y}$ **O(N²)**

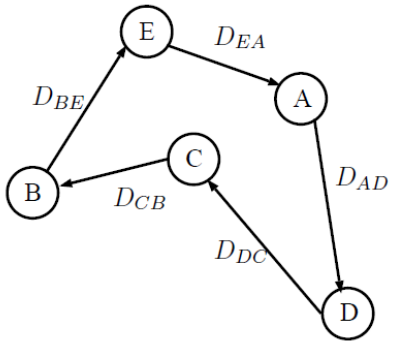
- Let's transform the TSP into a quadratic optimization problem tractable by HNN

Travelling salesman COP

- Given a distance matrix D for the graph we can represent a trip by a matrix V having

$D_{ij} :=$ distance between city i and j

$$V_{ij} := \begin{cases} 1, & \text{if we are at city } j \text{ at stage } i \\ 0, & \text{otherwise} \end{cases}$$



- For example the shown route can be described by the following route matrix:

$$\mathbf{V} = \begin{array}{|ccccc|c|} \hline A & B & C & D & E & \\ \hline 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 0 & 0 & 3 \\ 0 & 1 & 0 & 0 & 0 & 4 \\ 0 & 0 & 0 & 0 & 1 & 5 \\ \hline \end{array}, \qquad \mathbf{V} \in \{0,1\}^{K \times K}$$

Travelling salesman COP

- Note that **V** could be arbitrarily chosen, but for **V** to describe a valid tour for the agent **V** must satisfy several constraints:

a) Each row in **V** must contain exactly one 1-s, because the agent cannot be in two or more cities at the same time

b) Each column in **V** must contain exactly one 1-s because we must visit all the cities and we must visit them only once.

- We can describe mathematically these constraints as

1. Row orthogonality: $\sum_{k=1}^{K} V_{ik} V_{jk} = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{if } i \neq j \end{cases}$
2. Column orthogonality: $\sum_{k=1}^{K} V_{ki} V_{kj} = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{if } i \neq j \end{cases}$
3. Sum of the elements of **V** must be K: $\sum_{i=1}^{K} \sum_{j=1}^{K} V_{ij} = K$

Travelling salesman COP

- For **V** to describe a valid tour (be a feasible solution) **V** has to be a permutation matrix
- Having this notation we can formulate the cost of a route:

$$\sum_{i=1}^{K}\sum_{k=1}^{K}\sum_{j=1}^{K} V_{ij} D_{jk} V_{(i+1)k}$$

- The objective function to be minimized is:

$$\mathbf{V}_{opt} = \min_{\mathbf{V}\in P} \sum_{i=1}^{K}\sum_{k=1}^{K}\sum_{j=1}^{K} V_{ij} D_{jk} V_{(i+1)k}, \quad P = \left\{ \mathbf{A} : \mathbf{A} \text{ is a permutation matrix} \right\}$$

- Note that the HNN works with state vectors of

$$\mathbf{y} \in \{-1, +1\}^N$$

Travelling salesman COP

- We have to transform **V** to the domain of **y**

$$V_{ij} = \frac{y_{(i-1)K+j} + 1}{2}, \quad y_{(i-1)K+j} = 2V_{ij} - 1$$

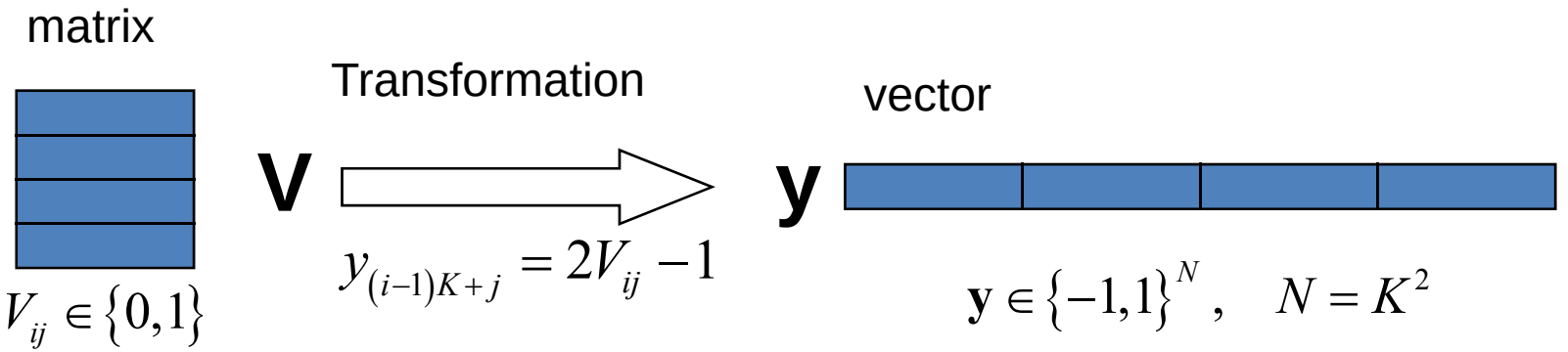
- After this we can write the objective function as:

$$\mathbf{y}_{opt} = \min_{\mathbf{y} \in \{-1,+1\}^{K^2}} \sum_{i=1}^{K} \sum_{k=1}^{K} \sum_{j=1}^{K} \frac{y_{(i-1)K+j} + 1}{2} D_{jk} \frac{y_{(i)K+k} + 1}{2},$$

- is this a quadratic form?

Travelling salesman COP

- This is a quadratic form which we will write in the traditional matrix-vector notation. Thus giving **W** and **b**



$$\mathbf{y}_{\text{opt}} = \min_{\mathbf{y}\in\{-1,1\}^N} \sum_{m=1}^{N}\sum_{j=1}^{N}\sum_{i=1}^{N} \frac{y_{(i-1)K+j}+1}{2} D_{jm} \frac{y_{(i)K+m}+1}{2}$$

W, b
???

$$\mathbf{y}_{\text{opt}} = \min_{\mathbf{y}\in\{-1,1\}^N} \mathbf{y}^T\mathbf{W}\mathbf{y} - 2\mathbf{b}^T\mathbf{y} = \min_{\mathbf{y}\in\{-1,1\}^N} \sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j W_{ij} - 2\sum_{i=1}^{N} b_i y_i$$

Travelling salesman COP

- We perform an index change: $k := (i-1)K + j \qquad h := (i)K + m$

  $i = 1, 2..., K, \quad m = 1, 2..., K, \quad j = 1, 2..., K$

  $k = 1, 2..., N-K, \quad h = 1, 2..., N$

- From the original cost function

$$\mathbf{y}_{\text{opt}} = \min_{y \in \{-1,1\}^N} \sum_{m=1}^{K} \sum_{j=1}^{K} \sum_{i=1}^{K} \frac{y_{(i-1)K+j} + 1}{2} D_{jm} \frac{y_{(i)K+m} + 1}{2}$$

- We come to:

$$\mathbf{y}_{\text{opt}} = \min_{\mathbf{y} \in \{-1,1\}^N} \sum_{h=1}^{N} \sum_{k=1}^{N-K} \frac{y_k + 1}{2} D^*_{kh} \frac{y_h + 1}{2}$$

$$D_{jm} = D^*_{\text{mod}_K h, \text{mod}_K k}$$

Travelling salesman COP

$$\mathbf{y}_{opt} = \min_{\mathbf{y}\in\{-1,1\}^N} \sum_{h=1}^{N}\sum_{k=1}^{(N-K)} \frac{y_h+1}{2} D^*_{hk} \frac{y_k+1}{2} =$$

$$= \min_{\mathbf{y}\in\{-1,1\}^N} \frac{1}{4}\sum_{h=1}^{N}\sum_{k=1}^{(N-K)} y_h y_k D^*_{hk} + \frac{1}{4}\sum_{h=1}^{N}\sum_{k=1}^{(N-K)} y_h D^*_{hk} + \frac{1}{4}\sum_{h=1}^{N}\sum_{k=1}^{(N-K)} D^*_{hk} y_k =$$

$$= \min_{\mathbf{y}\in\{-1,1\}^N} \frac{1}{4}\sum_{h=1}^{N}\sum_{k=1}^{(N-K)} y_h y_k D^*_{hk} + \frac{1}{2}\sum_{h=1}^{N} y_h \sum_{k=1}^{(N-K)} D^*_{hk} = \min_{\mathbf{y}\in\{-1,1\}^N} \mathbf{y}^T\mathbf{W}\mathbf{y} - 2\mathbf{b}^T\mathbf{y}$$

Weight matrix Bias vector

$\mathbf{y}_{opt}$ ➡ De-representation ➡ $\mathbf{V}_{opt}$

Is it a permutation matrix ???

We need a modified goal function to ensure a permutation matrix

Travelling salesman COP

- This energy function does not guarantee us to find a feasible solution (valid tour along the cities), so we have to incorporate the constraints for **V** into the energy function in such a way that the energy function has to be minimal if all the constraints are satisfied and the cost function is minimal.
- We choose a weighted linear combination of the cost function and the constraint terms so that if any of the constraints are not satisfied then the energy function is penalized thus pressing it farther from the minimum.

Travelling salesman COP

- We have the new energy function as

$$L(\mathbf{y}) = \alpha \underbrace{\sum_{i=1}^{K}\sum_{k=1}^{K}\sum_{j=1}^{K} \frac{y_{(i-1)K+j}+1}{2} D_{jk} \frac{y_{(i)K+k}+1}{2}}_{\text{route cost}} +$$

$$+\beta \underbrace{\sum_{k=1}^{K}\sum_{i=1}^{K}\sum_{\substack{j=1 \\ j\neq i}}^{K} V_{ik}V_{jk}}_{\text{row orthogonality}} + \gamma \underbrace{\sum_{k=1}^{K}\sum_{i=1}^{K}\sum_{\substack{j=1 \\ j\neq i}}^{K} V_{ki}V_{kj}}_{\text{column orthogonality}} + \delta \underbrace{\left(\sum_{i=1}^{K}\sum_{j=1}^{K} V_{ij} - K\right)^2}_{K \text{ city visit penalty}}$$

- Substituting

$$V_{ij} = \frac{y_{(i-1)K+j}+1}{2}$$

into all terms, we get

Travelling salesman COP

$$L(\mathbf{y}) = \alpha \sum_{i=1}^{K}\sum_{k=1}^{K}\sum_{j=1}^{K} \frac{y_{(i-1)K+j}+1}{2} D_{jk} \frac{y_{(i)K+k}+1}{2} + \beta \sum_{k=1}^{K}\sum_{i=1}^{K}\sum_{\substack{j=1 \\ j\neq i}}^{K} \frac{y_{(i-1)K+k}+1}{2} \frac{y_{(j-1)K+k}+1}{2}$$

$$+\gamma \sum_{k=1}^{K}\sum_{i=1}^{K}\sum_{\substack{j=1 \\ j\neq i}}^{K} \frac{y_{(k-1)K+i}+1}{2} \frac{y_{(k-1)K+j}+1}{2} + \delta \left(\sum_{i=1}^{K}\sum_{j=1}^{K} \frac{y_{(i-1)K+j}+1}{2} - K \right)^2$$

- This is a quadratic form again which we will write in the traditional matrix-vector notation. Thus giving **W** and **b**.
- Let us first separate the quadratic terms the linear terms and the constant terms the same way as we did for only the cost term.

Travelling salesman COP

$$L(\mathbf{y}) = \frac{\alpha}{4}\sum_{i=1}^{K}\sum_{k=1}^{K}\sum_{j=1}^{K}\left(y_{(i-1)K+j}y_{(i)K+k} + y_{(i-1)K+j} + y_{(i)K+k} + 1\right)D_{jk}$$

$$+\frac{\beta}{4}\sum_{k=1}^{K}\sum_{i=1}^{K}\sum_{\substack{j=1\\ j\neq i}}^{K}\left(y_{(i-1)K+k}y_{(j-1)K+k} + y_{(i-1)K+k} + y_{(j-1)K+k} + 1\right)$$

$$+\frac{\gamma}{4}\sum_{k=1}^{K}\sum_{i=1}^{K}\sum_{\substack{j=1\\ j\neq i}}^{K}\left(y_{(k-1)K+i}y_{(k-1)K+j} + y_{(k-1)K+j} + y_{(k-1)K+i} + 1\right)$$

$$+\frac{\delta}{4}\sum_{i=1}^{K}\sum_{j=1}^{K}\sum_{m=1}^{K}\sum_{n=1}^{K}\left(y_{(i-1)K+j}y_{(m-1)K+n} + y_{(i-1)K+j} + y_{(m-1)K+n} + 1\right)$$

$$-\delta K\sum_{i=1}^{K}\sum_{j=1}^{K}\left(y_{(i-1)K+j} + 1\right) + \delta K^2$$

Travelling salesman COP

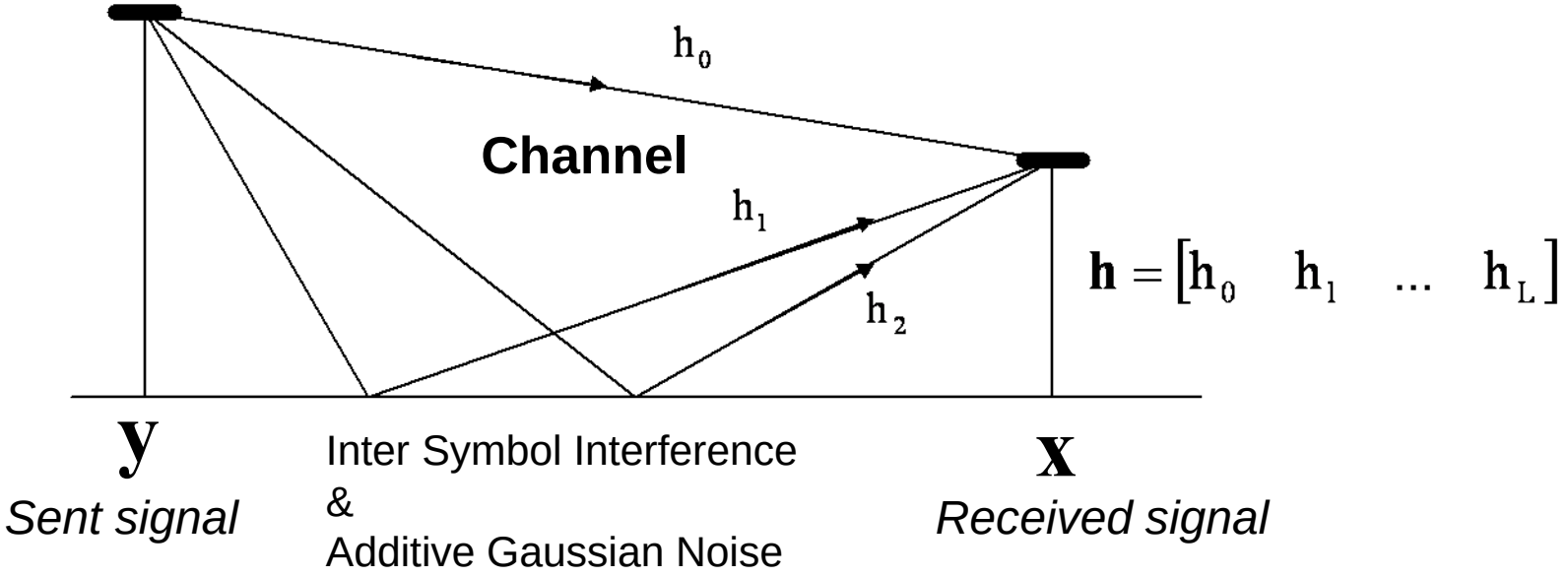
- After substituting and evaluating the parameters we get independent matrices and vectors to form the overall quadratic function:

$$L(\mathbf{y}) = \mathbf{y}^T \cdot \underbrace{\left(\alpha \cdot \mathbf{A} + \beta \cdot \mathbf{B} + \gamma \cdot \mathbf{C} + \delta \cdot \mathbf{D}\right)}_{\mathbf{W}} \cdot \mathbf{y} - 2\mathbf{y}^T \cdot \underbrace{\left(\alpha \cdot \mathbf{a} + \beta \cdot \mathbf{g} + \gamma \cdot \mathbf{c} + \delta \cdot \mathbf{d}\right)}_{\mathbf{b}}$$

- Where the matrices and vectors correspond properties of the row, column orthogonality, the permutation matrix property and the press of the cost term.
- The weights of the linear combinations can be adjusted over the solution process in each stage to emphasize one property over another. Usually heuristics are applied to change them.

Signal detection as combinatorial optimization

- One other example for a COP is a multipath propagated radio wave compensation and detection in communication.
- We send a block of symbols but the receiver gets a noisy linear combination of them due to ISI and additive noise



Signal detection as combinatorial optimization

- We send a block of symbols

$$\mathbf{y} = \begin{bmatrix} y_1 & y_2 & \dots & y_L & \dots & y_N \end{bmatrix}^{\mathrm{T}}$$

ISI

$$\mathbf{x} = \begin{bmatrix} x_1 & x_2 & \dots & x_L & x_N \end{bmatrix}^{\mathrm{T}}$$

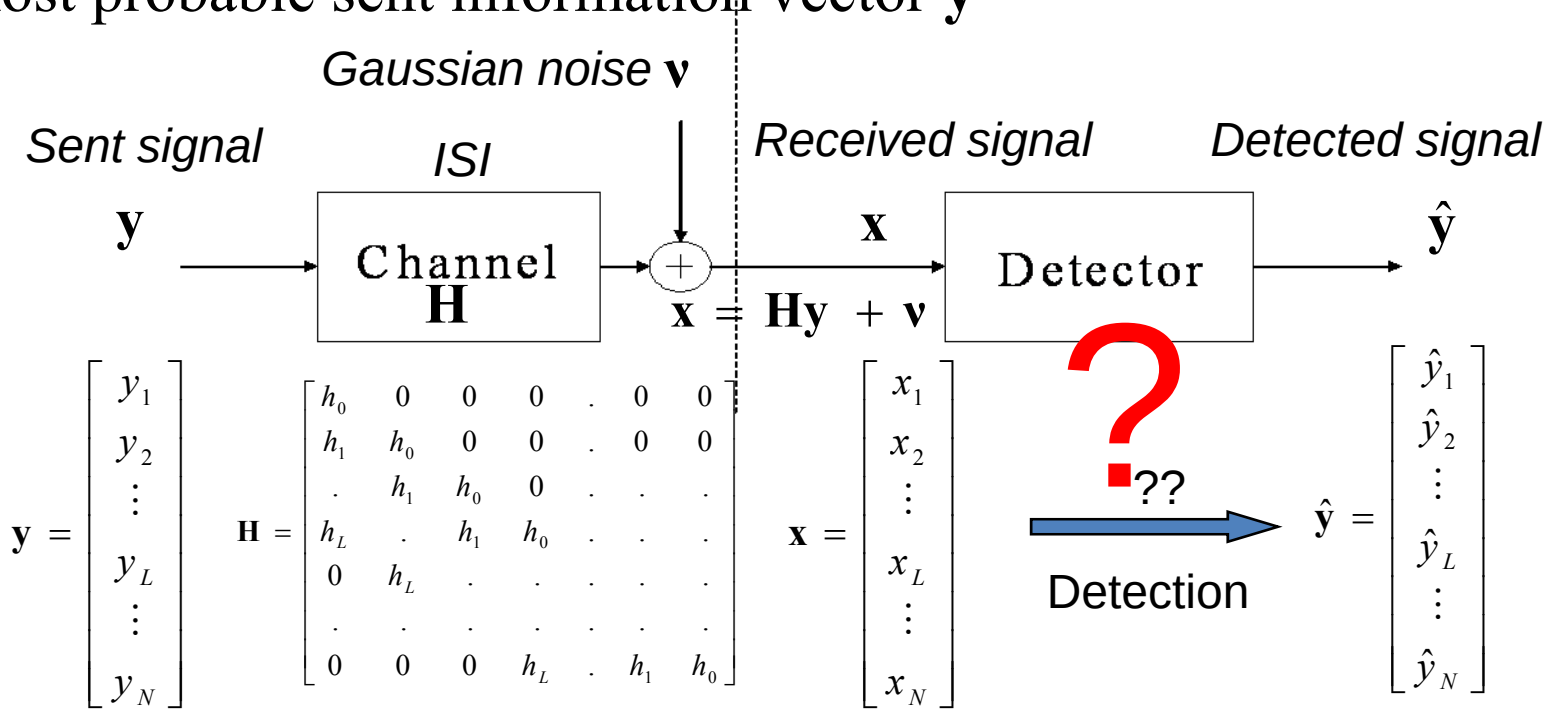
- But due to the channel acts like a linear filter, we receive a noise added convolution of the sent symbols with the channel's impulse response function

$$x_k = h_0 y_k + h_1 y_{k-1} + h_2 y_{k-2} + \dots + h_L y_{k-L} + \nu_k =$$

$$= \sum_{j=0}^{L} h_j y_{k-j} + \nu_k$$

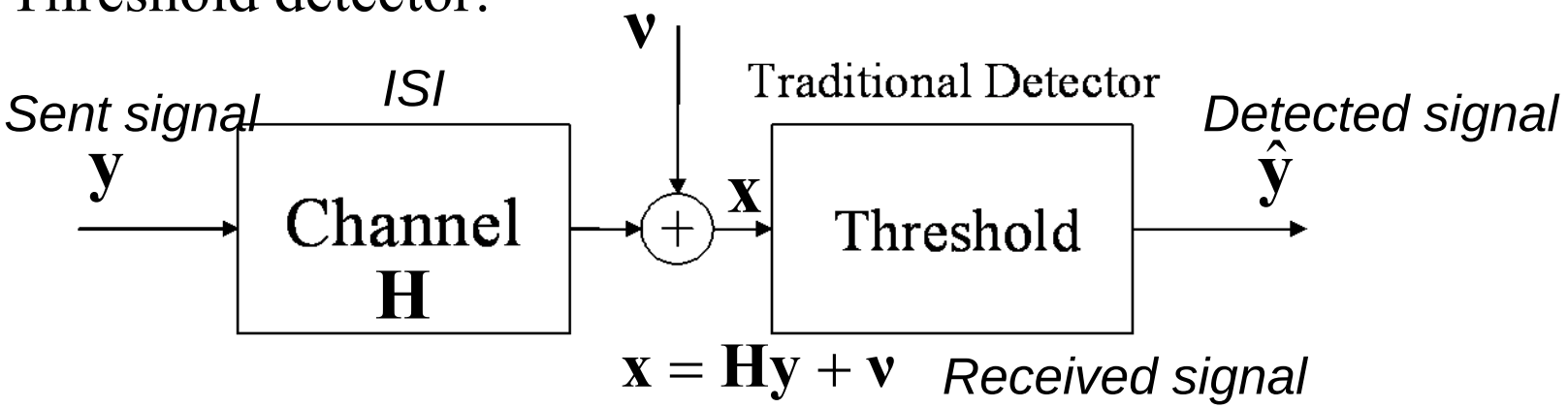
Signal detection as combinatorial optimization

- We want to make a decision based on the knowledge of **H** (the channel matrix) and the received message **x**, what was the most probable sent information vector **y**



Signal detection as combinatorial optimization

- We can use a simple decision rule, taking the sign of the received signal.
- Threshold detector:



$$\hat{\mathbf{y}} = \operatorname{sgn}\{\mathbf{x}\} = \operatorname{sgn}\{\mathbf{H}\mathbf{y} + \boldsymbol{\nu}\}$$

$$\hat{y}_k = \operatorname{sgn}\{x_k\} = \operatorname{sgn}\left\{\sum_{j=0}^{L} h_j y_{k-j} + \nu_k\right\} = \operatorname{sgn}\left\{h_0 y_k + \sum_{j=1}^{L} h_j y_{k-j} + \nu_k\right\}$$

Signal detection as combinatorial optimization

- But we know more information of the underlying phenomenon (we know **H** and that a noise is added). So applying the Bayesian decision rule:

$$\hat{\mathbf{y}}_{\text{opt}} : \max_{\mathbf{y}\in\{-1,1\}^N} p(\mathbf{y}\,|\,\mathbf{x}) = \max_{\mathbf{y}\in\{-1,1\}^N} \frac{p(\mathbf{x}\,|\,\mathbf{y})p(\mathbf{y})}{p(\mathbf{x})} = \max_{\mathbf{y}\in\{-1,1\}^N} p(\mathbf{x}\,|\,\mathbf{y})$$

- We know that the received signal is constructed by the channel as: $\mathbf{x} = \mathbf{H}\mathbf{y} + \mathbf{v}$
- And that the noise is an additive white Gaussian noise
- So the observed signal can be treated as a random variable:

$$\mathbf{v} \sim N(0, \mathbf{K}) \quad \mathbf{x} \sim N(\mathbf{H}\mathbf{y}, \mathbf{K})$$

Signal detection as combinatorial optimization

- We can describe the optimal decision based on the Bayesian rule:

$$\hat{\mathbf{y}}_{opt} : \max_{\mathbf{y}\in\{-1,1\}^N} \frac{1}{\sqrt{(2\pi)^N \det(\mathbf{K})}} e^{-\frac{1}{2}(\mathbf{x}-\mathbf{H}\mathbf{y})^T \mathbf{K}^{-1}(\mathbf{x}-\mathbf{H}\mathbf{y})} =$$

$$= \max_{\mathbf{y}\in\{-1,1\}^N} -(\mathbf{x}-\mathbf{H}\mathbf{y})^T \mathbf{K}^{-1}(\mathbf{x}-\mathbf{H}\mathbf{y}) =$$

$$= \min_{\mathbf{y}\in\{-1,1\}^N} (\mathbf{x}-\mathbf{H}\mathbf{y})^T \mathbf{K}^{-1}(\mathbf{x}-\mathbf{H}\mathbf{y})$$

- We will show that this is a quadratic form indeed.

Signal detection as combinatorial optimization

- Expanding the expression:

$$\hat{\mathbf{y}}_{\text{opt}} : \min_{\mathbf{y}\in\{-1,1\}^N} \left(\mathbf{x}-\mathbf{H}\mathbf{y}\right)^T \mathbf{K}^{-1}\left(\mathbf{x}-\mathbf{H}\mathbf{y}\right) =$$

$$= \min_{\mathbf{y}\in\{-1,1\}^N} \underbrace{\mathbf{x}^T\mathbf{K}^{-1}\mathbf{x}}_{\text{constant respect to } \mathbf{y}} - \mathbf{y}^T\mathbf{H}^T\mathbf{K}^{-1}\mathbf{x} - \mathbf{x}^T\mathbf{K}^{-1}\mathbf{H}\mathbf{y} + \mathbf{y}^T\mathbf{H}^T\mathbf{K}^{-1}\mathbf{H}\mathbf{y} =$$

$$= \min_{\mathbf{y}\in\{-1,1\}^N} \mathbf{y}^T\mathbf{W}\mathbf{y} - 2\mathbf{b}^T\mathbf{y}$$

where (note that $\mathbf{K}=\mathbf{K}^T$ so $\mathbf{K}^{-1}=\mathbf{K}^{-1T}$)

$$\mathbf{W}=\mathbf{H}^T\mathbf{K}^{-1}\mathbf{H} \quad \text{and} \quad \mathbf{b}=\mathbf{H}^T\mathbf{K}^{-1}\mathbf{x}$$

Signal detection as combinatorial optimization

- So we have constructed a quadratic energy function that can be used as the energy function of the HNN.

$$\hat{\mathbf{y}}_{\text{opt}} : \min_{\mathbf{y}\in\{-1,1\}^N} \mathbf{y}^T\mathbf{W}\mathbf{y} - 2\mathbf{b}^T\mathbf{y}$$

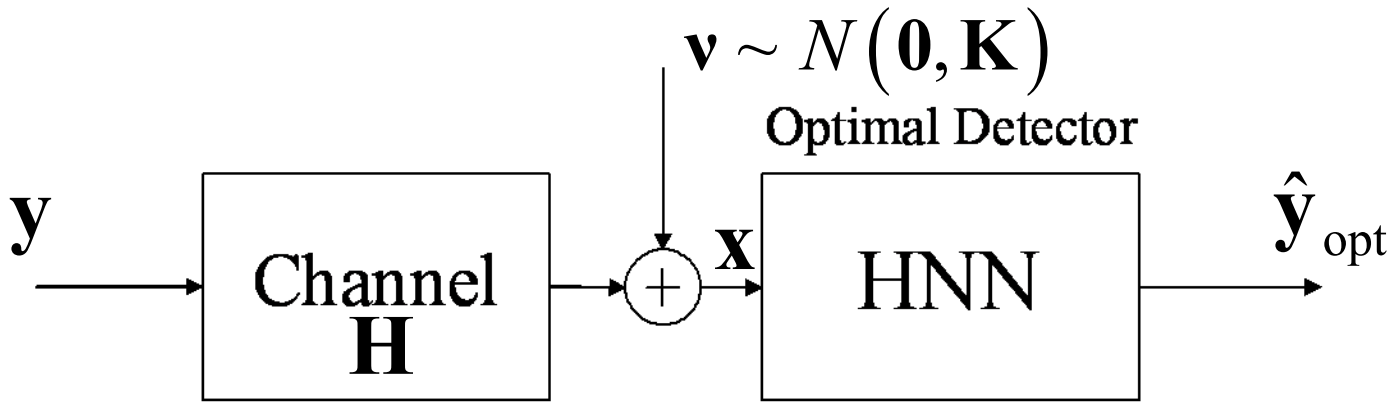
$$\mathbf{W} = \mathbf{H}^T\mathbf{K}^{-1}\mathbf{H} \quad \text{and} \quad \mathbf{b} = \mathbf{H}^T\mathbf{K}^{-1}\mathbf{x}$$

$$y_i(k+1) = -\operatorname{sgn}\left\{\sum_{k=1}^{N} \tilde{W}_{ij} y_j(k) - b_i\right\}$$

$$\tilde{W}_{ij} = \begin{cases} W_{ij}, & \text{if } i \neq j \\ 0, & \text{if } i = j \end{cases}$$

Signal detection as combinatorial optimization

- The HNN with the given parameters can approximate the optimal detection.

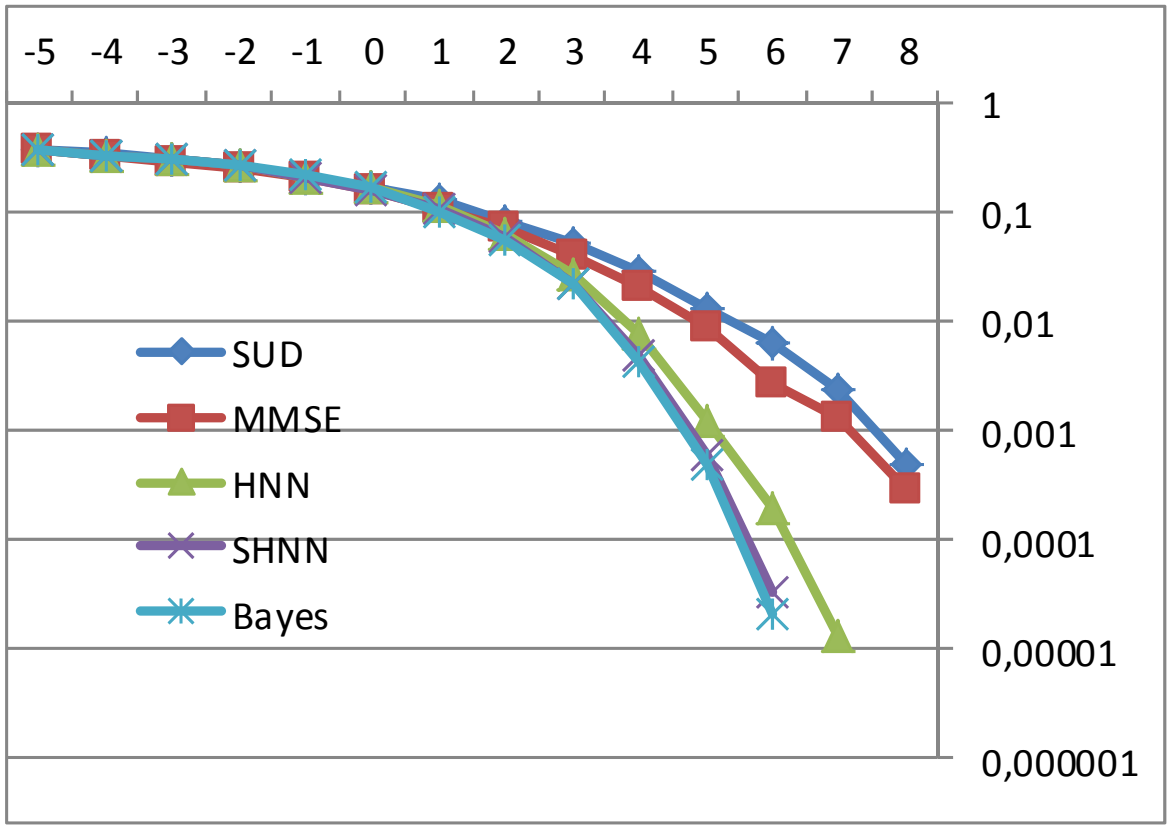


$$\mathbf{W} = \mathbf{H}^T \mathbf{K}^{-1} \mathbf{H} \quad \text{and} \quad \mathbf{b} = \mathbf{H}^T \mathbf{K}^{-1} \mathbf{x}$$

$$y_i(k+1) = -\operatorname{sgn}\left\{ \sum_{k=1}^{N} \tilde{W}_{ij} y_j(k) - b_i \right\}$$

Signal detection as combinatorial optimization

- Performance analysis for an example: $\mathbf{h} = [1 \quad 0.4 \quad 0.1 \quad 0.3 \quad 0.2]^T$



SNR vs. BER

Analog circuit implementation of the HNN

- There are several types of implementation of the HNN. Software like Matlab or Labview contain packages of different neural networks.
- On a DSP one can exploit the fast matrix vector multiplication capabilities.
- The optical implementation gives us a very fast architecture.
- However the available software are very slow in contrast to the hardware implementations, while the DSP and optical implementation is not cut out for large scale. Due to the quadratically growing interconnections between neurons.

Analog circuit implementation of the HNN

- The first step in implementing Hopfield Neural Network as an analog circuit is to analyze the nonlinear state transition rule of the network: $$y_i(k+1) = \operatorname{sgn}\left\{\sum_{j=1}^{N} W_{ij} y_j(k)\right\}$$ where we have set **b** = **0** for the ease of simpler formulas.
- This is a discrete time state transition rule, in terms k=1,2,…
- When we are implementing Hopfield Neural Networks as an analog circuit then this circuit can not handle discrete time but continuous time. This gives rise to the first question, namely how to change this network from discrete to continuous time?

Analog circuit implementation of the HNN

- We need to modify the state transition from a discrete time step to an infinitesimally small time step (difference) and use a differentiable activation function instead of the sign function.

$$y_i(k+1) - y_i(k) = -y_i(k) + \varphi\left\{\sum_{j=1}^{N} W_{ij} y_j(k)\right\}$$

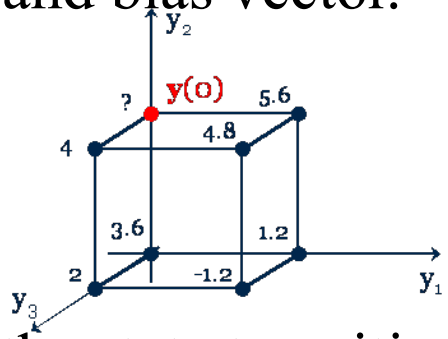
- If we choose an arbitrary small time step

$$\frac{y_i(t+\Delta t) - y_i(t)}{\Delta t} = \dot{y}_i(t) = -y_i(t) + \varphi\left\{\sum_{j=1}^{N} W_{ij} y_j(t)\right\}$$

Exercises – example 1

Given a DHNN by it's weight matrix and bias vector.

$$\mathbf{W} = \begin{bmatrix} 1 & 0.5 & -0.1 \\ 0.5 & 1 & 0.2 \\ -0.1 & 0.2 & 1 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} 0.2 \\ -0.8 \\ 0.3 \end{bmatrix}$$



a) Determine and draw to the figure the state transitions and the stable point using the given values of the Lyapunov function if we use the network for minimization, and the initial state is:

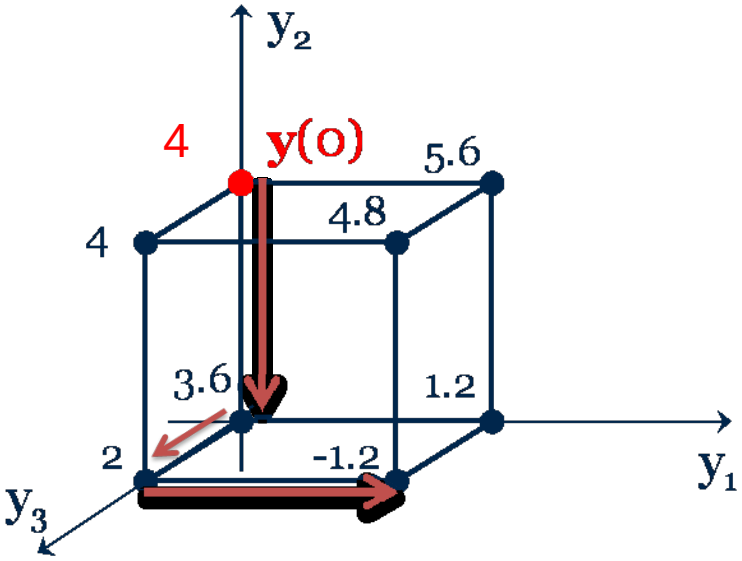
$$\mathbf{y}(0) = \begin{bmatrix} -1 & 1 & -1 \end{bmatrix}^{\mathrm{T}}$$

b) Verify the solution applying and computing the states according to the state transition rule.

Exercises – example 1 solution

a)

$$\mathcal{L}\left(\mathbf{y}^{(3)}\right)=\begin{bmatrix}-1 & 1 & -1\end{bmatrix}\begin{bmatrix}1 & 0.5 & -0.1\\ 0.5 & 1 & 0.2\\ -0.1 & 0.2 & 1\end{bmatrix}\begin{bmatrix}-1\\ 1\\ -1\end{bmatrix}-2\begin{bmatrix}0.2 & -0.8 & 0.3\end{bmatrix}\begin{bmatrix}-1\\ 1\\ -1\end{bmatrix}=4$$



Exercises – example 1 solution

b)

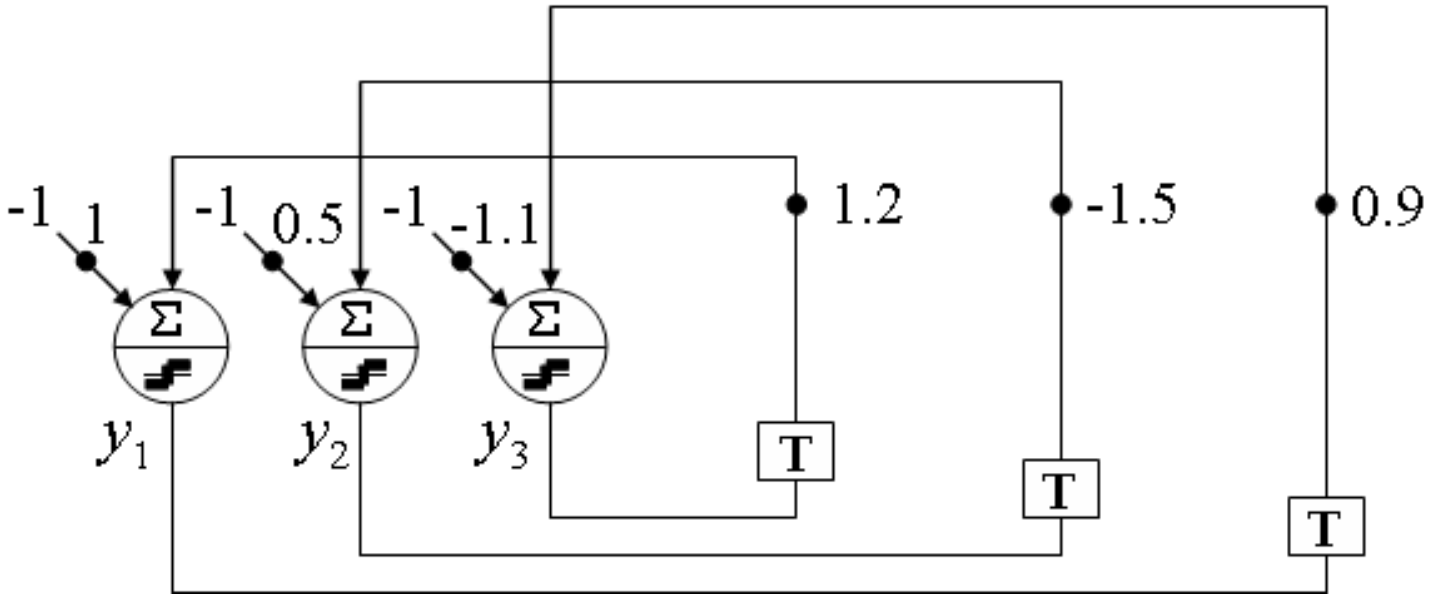
$$y_l(k+1) = -\operatorname{sgn}\left\{\sum_{j=1}^{N} \tilde{W}_{lj} y_j(k) - b_l\right\}, \text{ ahol } \tilde{W}_{ij} = \begin{cases} W_{ij} & i \neq j \\ 0 & i = j \end{cases}, \quad l = \operatorname{mod}_N k$$

$$\begin{matrix} & k=0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{matrix}$$

$$\begin{bmatrix} 0 & 0.5 & -0.1 \\ 0.5 & 0 & 0.2 \\ -0.1 & 0.2 & 0 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix} \begin{bmatrix} \boxed{-1} \\ 1 \\ -1 \end{bmatrix} \begin{bmatrix} -1 \\ \boxed{-1} \\ -1 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ \boxed{1} \end{bmatrix} \begin{bmatrix} \boxed{1} \\ -1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ \boxed{-1} \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ \boxed{1} \end{bmatrix} \begin{bmatrix} \boxed{1} \\ -1 \\ 1 \end{bmatrix}$$

Exercises – example 2

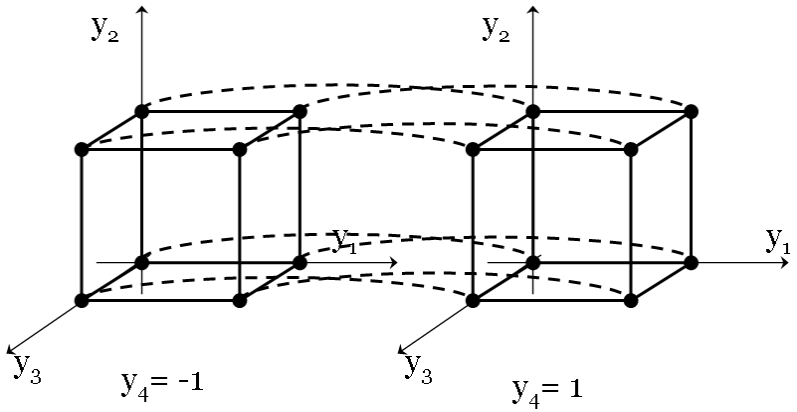
Give the stable point of the following HNN:



Exercises – example 3

We want to store the following samples in a HNN used as an associative memory: $\mathbf{s}_1 = [1 \quad -1 \quad 1 \quad -1]^T \qquad \mathbf{s}_2 = [1 \quad 1 \quad -1 \quad 1]^T$

- Give the weight matrix and the bias vector of the network!
- Are the samples orthogonal?
- Show a stable point beside the stored sample points!
- Mark the states in the figure from where the net converges to the stored samples:



Exercises – example 4

We want to solve the following optimization problem with a Hopfield net:

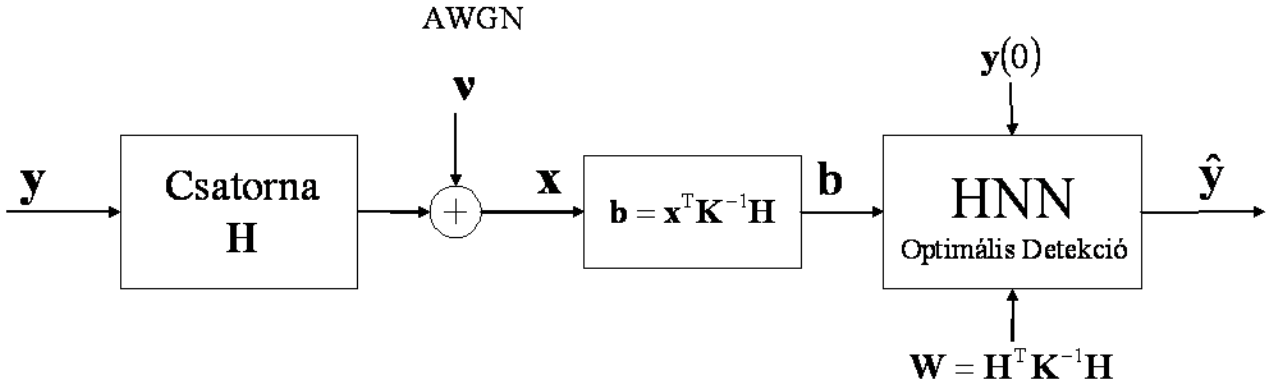
$$\mathbf{y}_{\text{opt}} : \min_{\mathbf{y}\in\{-1,1\}^3} \mathbf{y}^T\mathbf{W}\mathbf{y} - 2\mathbf{b}^T\mathbf{y}, \quad \mathbf{W} = \begin{pmatrix} 4 & -0.5 & 0.1 \\ -0.5 & 4 & 0.2 \\ 0.1 & 0.2 & 4 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 0.6 \\ 1.4 \\ -5.2 \end{pmatrix}$$

- Give the concrete recursive state update formula of this Hopfield net used for minimization!

Exercises – example 5

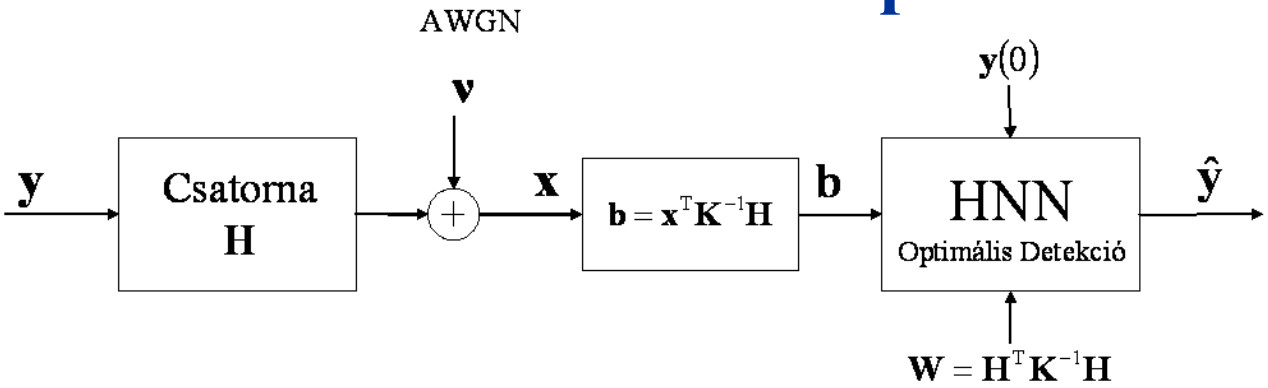
We want to use a Hopfield net in a digital communication system for detection. The state of the linear channel distortion with an AWGN noise is assumed to be known and to be stationer.

The impulse response of the channel is: $\mathbf{h} = [1 \quad 0.5 \quad 0.1]^T$ and the SNR is 0dB. The block representation of the system model is:



- Show that the HNN is the optimal detector for this problem.

Exercises – example 5



- Give the weight matrix and the bias vector of the Hopfield net for the given channel impulse response and noise power if the received signal is: $\mathbf{x} = [2.4435 \quad 1.1490 \quad 0.2232]^T$
- What will be the decoded message ($\hat{\mathbf{y}}$)?
- What would be the decoded message if a threshold detector would have been used instead of the HNN detector?

Note: the initial state of the HNN is random. In the example use $\mathbf{y}(0) = [-1 \quad -1 \quad 1]^T$

Summary

- A method was shown how to use the HNN to minimize a quadratic cost function
- This construction was used for solving combinatorial optimization problems which are traditionally NP problems, but with the HNN polynomial complexity approximation is given for the solution.
- Examples have been shown of mapping combinatorial optimization problems into quadratic programming tasks
- Possible analog circuit implementation was shown for the HNN