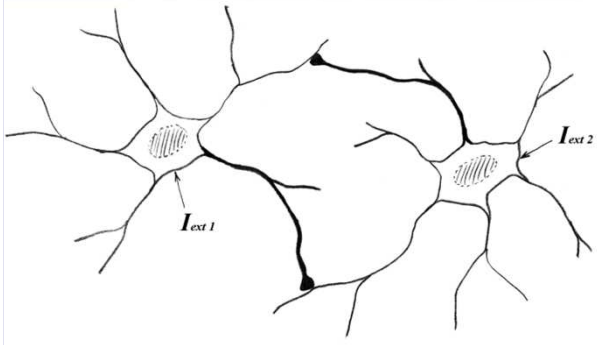
Neuronové sítě



V prezentaci jsou použity podklady z řady zdrojů (Marcel Jiřina, Dan Novák, Jean-Christophe Prévotet, Petr Berka, Jana Tučková a další)

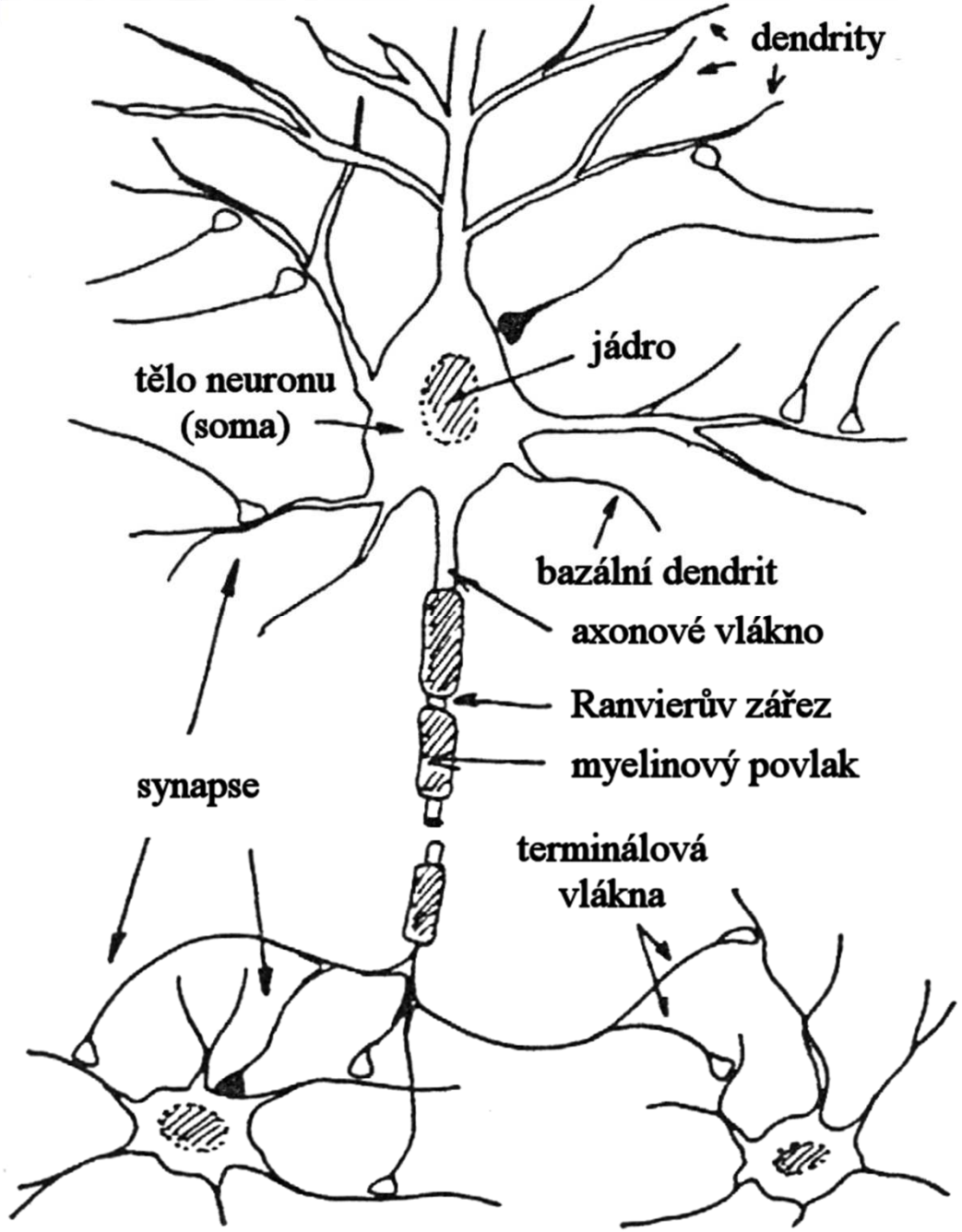
Neuronové sítě

Jsou inspirovány poznatky o neuronech a nervových sítích živých organizmů a jejich schopnostmi:

- extrahovat a reprezentovat závislosti v datech, které nejsou zřejmé
- řešit silně nelineární úlohy
- učit se
- zevšeobecňovat

Využívají se pro klasifikaci, regresi a predikci časových řad

Biologická inspirace

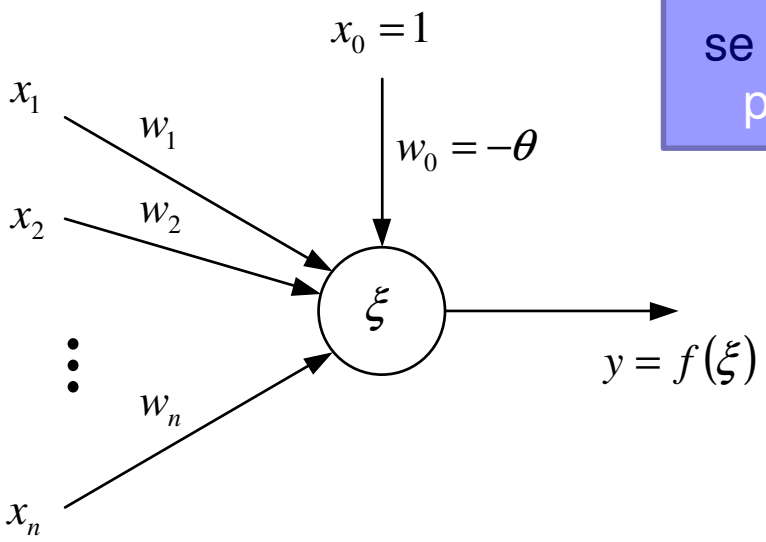


Perceptron - model neuronu

jako základní výpočetní jednotky neuronových sítí.

Perceptron využívá prostý lineární model!

x_0
se používá pro označení
požadované hodnoty



$$\xi = \sum_{i=1}^{n} w_i x_i - \theta = \sum_{i=0}^{n} w_i x_i$$

$$f(\xi) = \frac{1}{1 + e^{-\lambda \xi}}$$

Principy použité při modelování neuronu

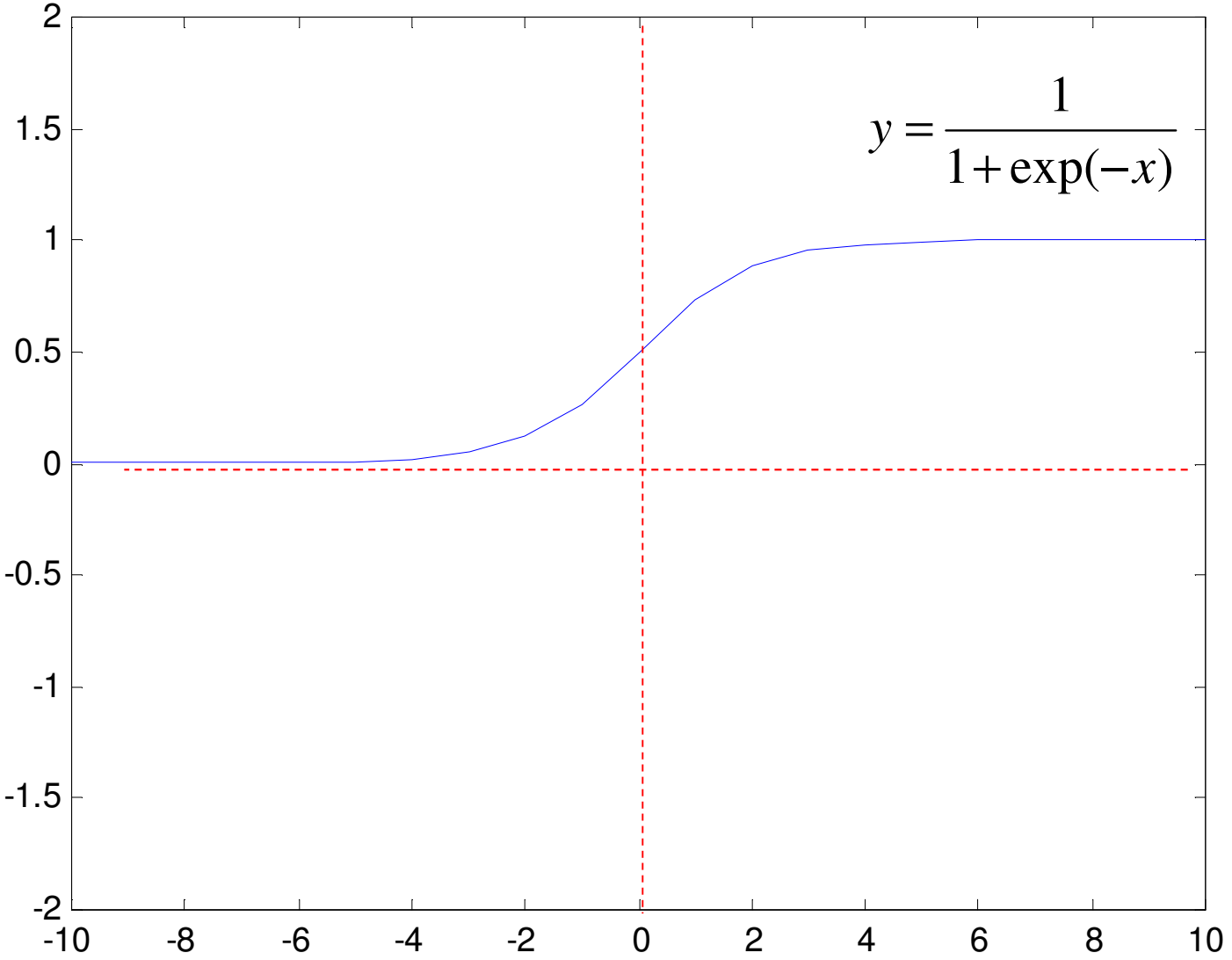
- obsahuje několik vstupů, které jsou ohodnoceny vahami a jeden výstup
- v neuronu pobíhají dva procesy:
  - výpočet (post-synaptického) **potenciálu**

$$\xi = \sum_{i=1}^{n} w_i x_i - \theta = \sum_{i=0}^{n} w_i x_i$$

  - výpočet **hodnoty výstupu** pomocí (přenosové) aktivační funkce, nejčastěji tzv. **sigmoidy**

$$f(\xi) = \frac{1}{1 + e^{-\lambda \xi}}$$

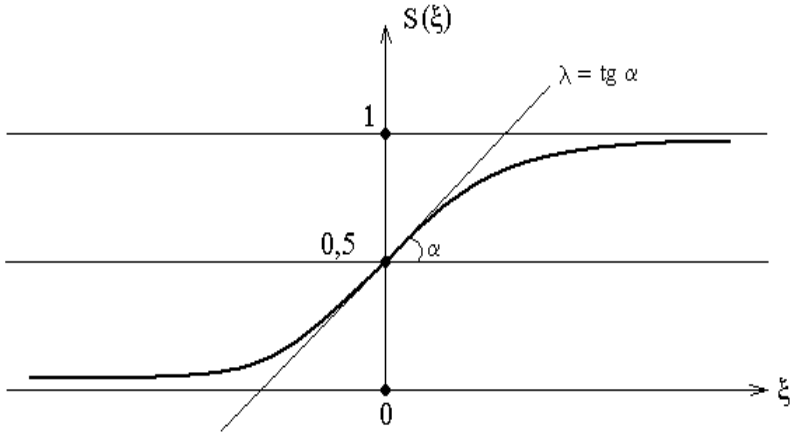
Logistická (sigmoida)



Sigmoidní **přenosová funkce**:

$$S(\xi) = \frac{1}{1 + \exp(-\lambda\xi)}$$

$$z = S\left(\sum_{i=1}^{n} w_i x_i - \Theta\right)$$



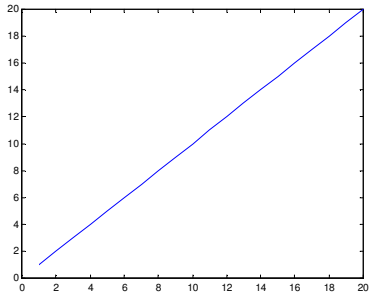
Přenos neuronové sítě je určen:

topologií sítě – počet vrstev a jejich neuronů
parametry sítě

Parametry neuronové sítě:

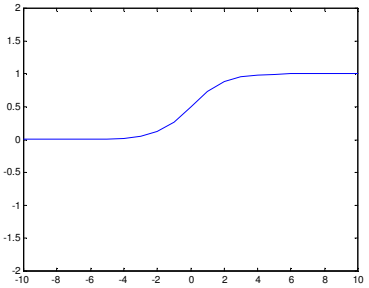
- váhové koeficienty vazeb neuronů $w_{jk} \in < 0, 1 >$
- prahové hodnoty Θ
- *volba a* parametry přenosové funkce S, λ

Příklady aktivačních funkcí



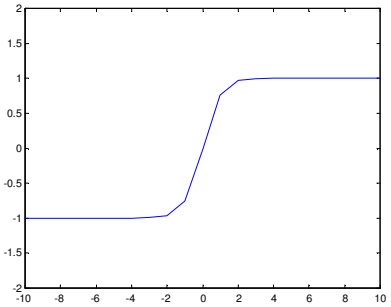
Lineární

$$y = x$$



Logistická (sigmoida)

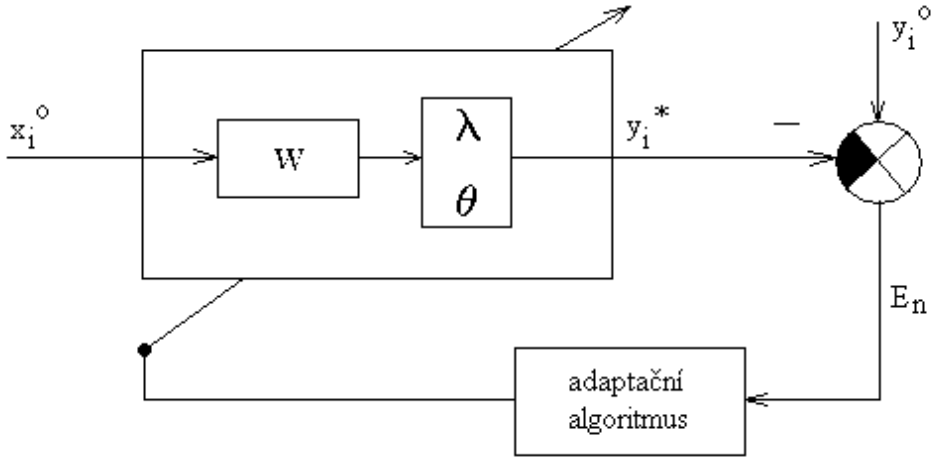
$$y = \frac{1}{1+\exp(-x)}$$



Hyperbolický tangens

$$y = \frac{\exp(x)-\exp(-x)}{\exp(x)+\exp(-x)}$$

Procedura učení – optimalizační gradientní algoritmus Back-Propagation (BP)



Strategie optimalizace
pro ***K*** trénovacích příkladů
[$\boldsymbol{x}_j$, y_j] :

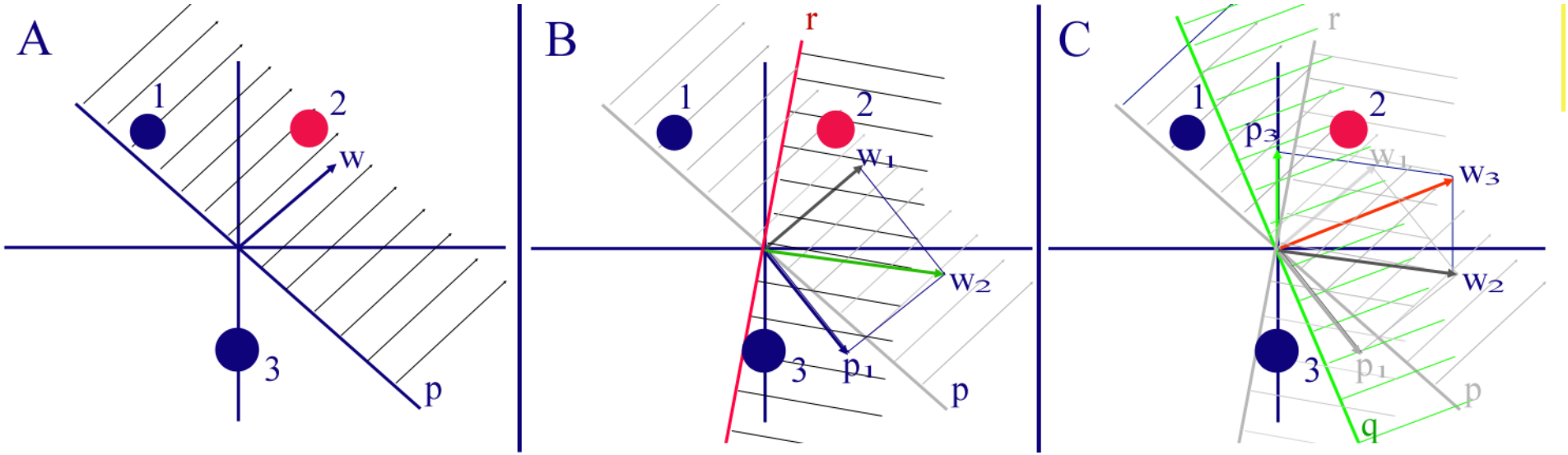
$$E_n = \frac{1}{2}\sum_{j=1}^{K}\left(y_j^0 - y_j^*\right)^2 \rightarrow \min$$

kde y_j^* je výstup perceptr.

Adaptace váhy $$w(t+1) = \mathrm{w(t)} + \mu\frac{\partial \mathrm{E_n}}{\partial \mathrm{w(t)}} \qquad \mu \in \langle 0,1 \rangle$$

Pro perceptron je $y^* = \boldsymbol{w} \,.\, \boldsymbol{x}$, a tedy $$\frac{\partial \mathrm{E_n}}{\partial \mathrm{w(t)}} = \sum_{j=1}^{K}\left(y_j^0 - y_j^*\right)(-x_j)$$

Při inkrementální (stochastické) aproximaci $$w(t+1) = \mathrm{w(t)} + \mu.(y_i - y_i^*).x_i$$

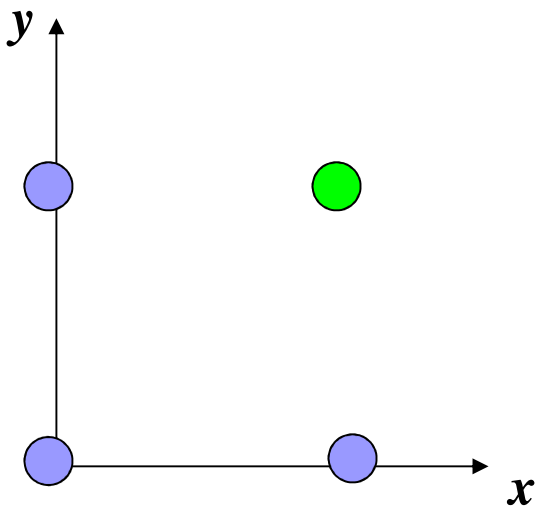


3 fáze učení jednoduchého perceptronu - **A**: je zvolen náhodně vektor vah $\mathbf{w_1}$ a k němu určena kolmá rozhodovací hranice (**p**), zjišťujeme výstup pro bod 1 - leží v oblasti s výstupem 1, i když má ležet v oblasti s výstupem 0 → **B**: odečteme vektor bodu 1 od vektoru vah $\mathbf{w_1}$ a získáme nový vektor vah $\mathbf{w_2}$ a k němu příslušnou rozhodovací hranici znázorněnou přímkou **r**, bod 2 je umístěn správně, bodu 3 přiřadí síť hodnotu 1 i když má dostat výstup 0 → **C**: odečteme vektor bodu 3 od vektoru vah $\mathbf{w_2}$ a získáme nový vektor vah $\mathbf{w_3}$ → nyní je již problém vyřešen - všem bodům je přiřazena odpovídající hodnota výstupu, řešením problému je tedy vektor vah $\mathbf{w_3}$ s příslušnou rozhodovací hranicí **q**.

Perceptron a jeho omezení

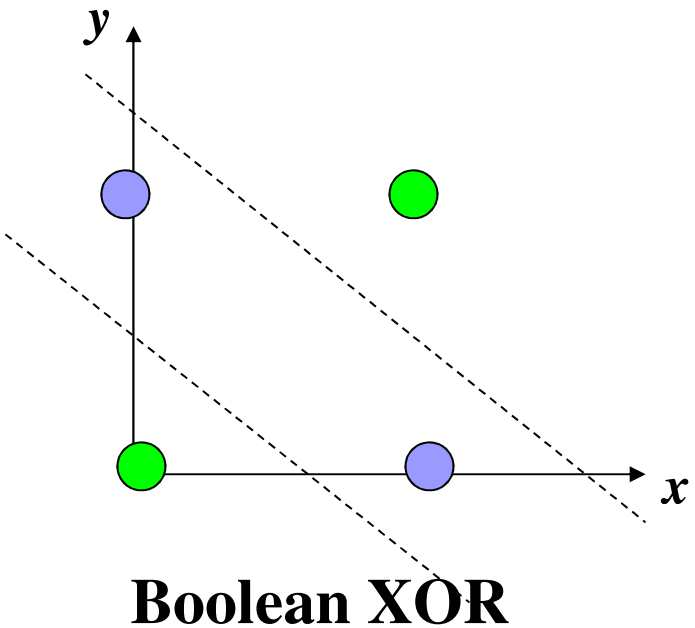
Jednovrstvý perceptron se může naučit řešit jen problémy, které jsou lineárně separabilní.

- Booleovské funkce AND i OR jsou lineárně separabilní,
- avšak booleovská funkce XOR (a obecně problém parity) tuto vlastnost nemá.



Boolean AND

x AND *y*



Boolean XOR

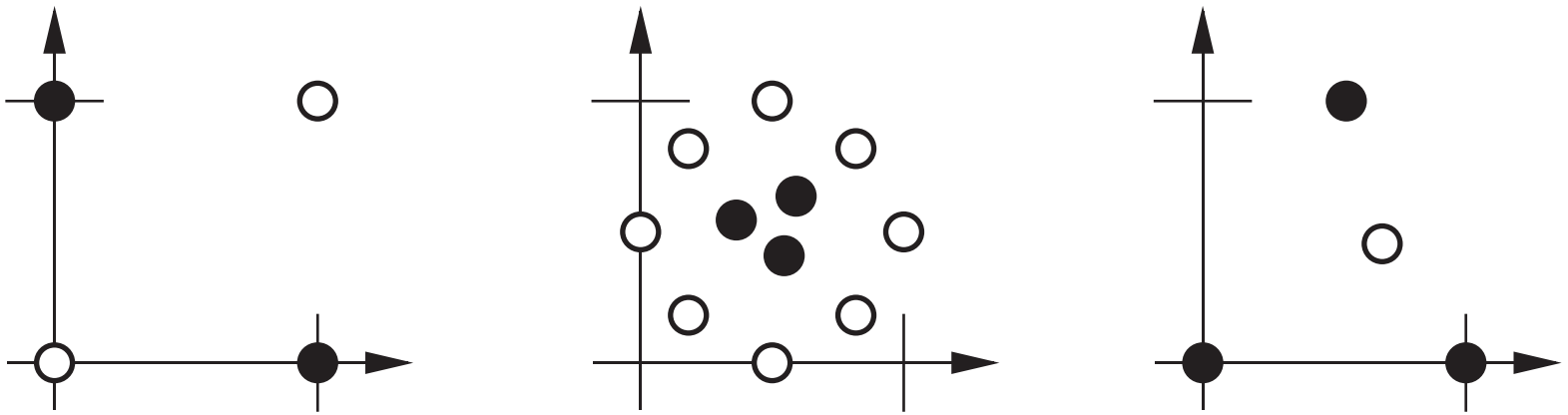
x XOR *y*

Omezení 1 perceptronu

Perceptron pracuje s lineární hranicí

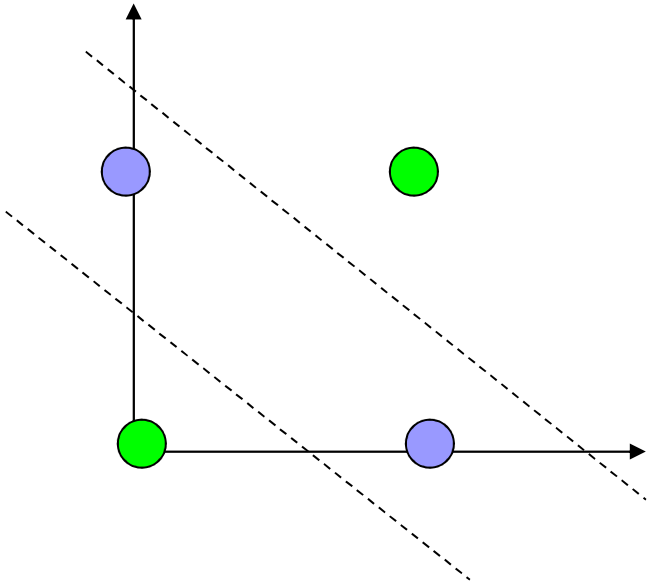
$$_1\mathbf{w}^T\mathbf{p} + b = 0$$

Příklady problémů, které nejsou lineárně separabilní



Jak překonat lin.omezení perceptronu?

- XOR problém by bylo možné řešit s použitím 2 lineárních hranic: Je řešení v použití více vrstev neuronů ?
- Nechť každý neuron v jedné vrstvě implementuje svou lineární hranici a nechť další vrstva obě rozhodnutí kombinuje.

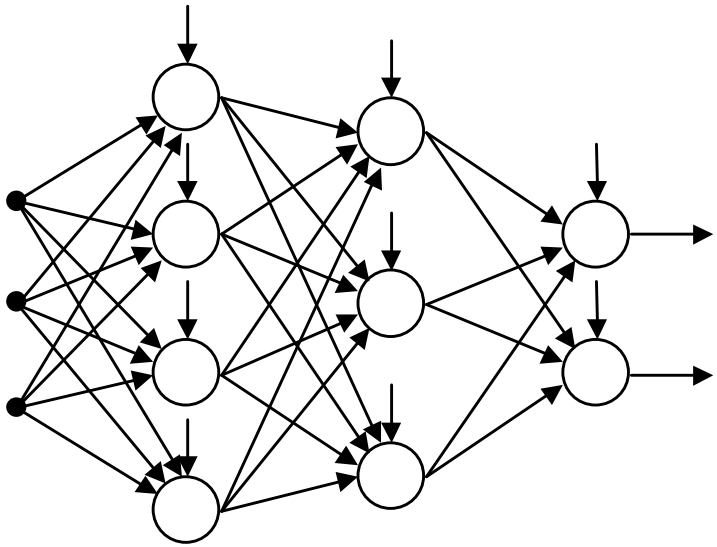


Jak má v takovém případě probíhat učení pro jednotlivé neurony?

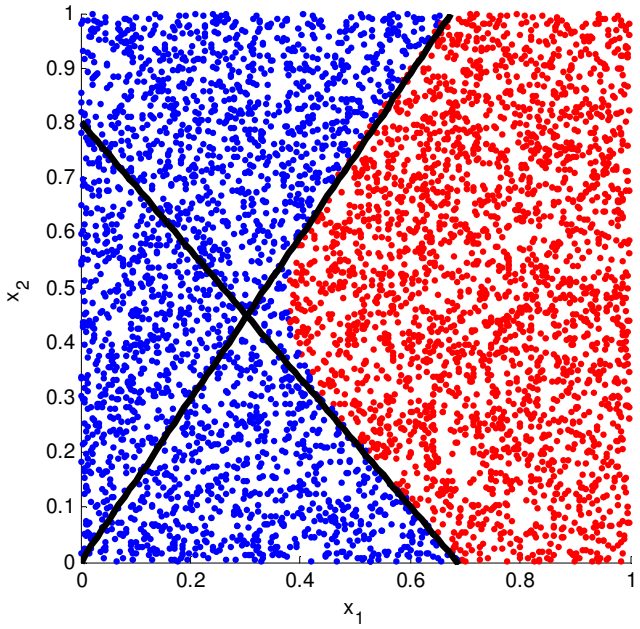
Vícevrstvé sítě používají pro své učení algoritmus zpětné propagace chyby **backpropagation learning algorithm**

Struktura sítě

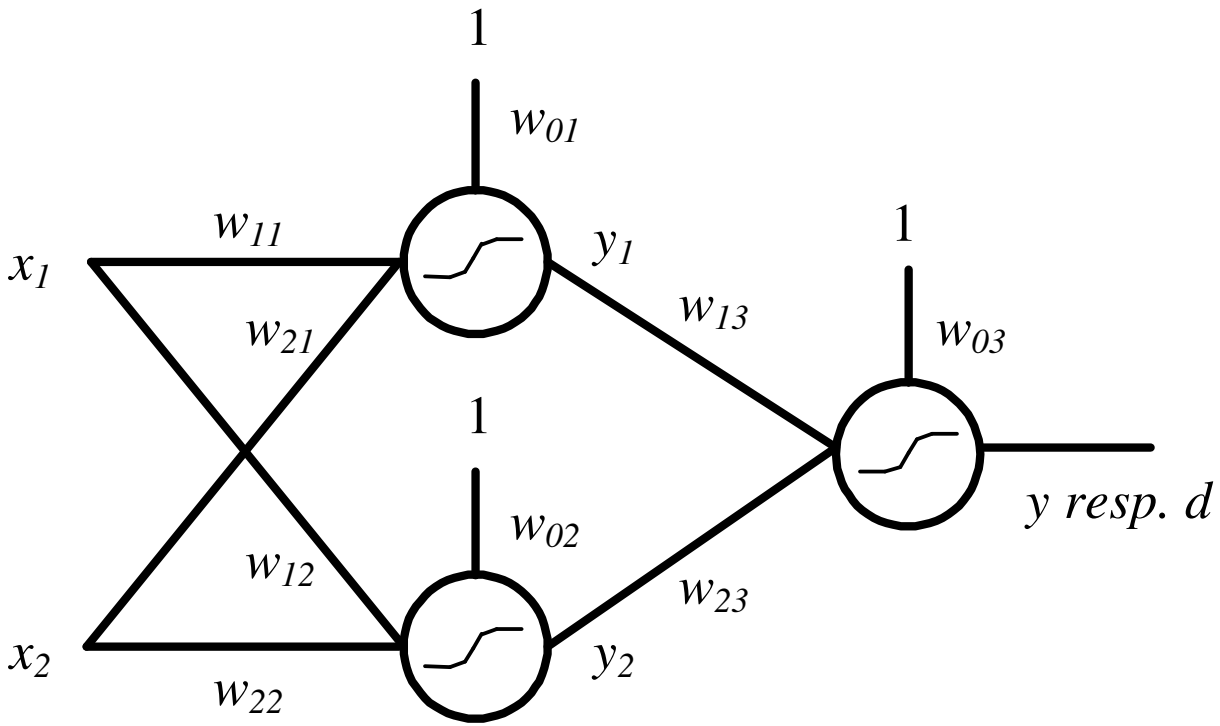
- pro větší výpočetní sílu se neurony uspořádávají do sítí neuronů



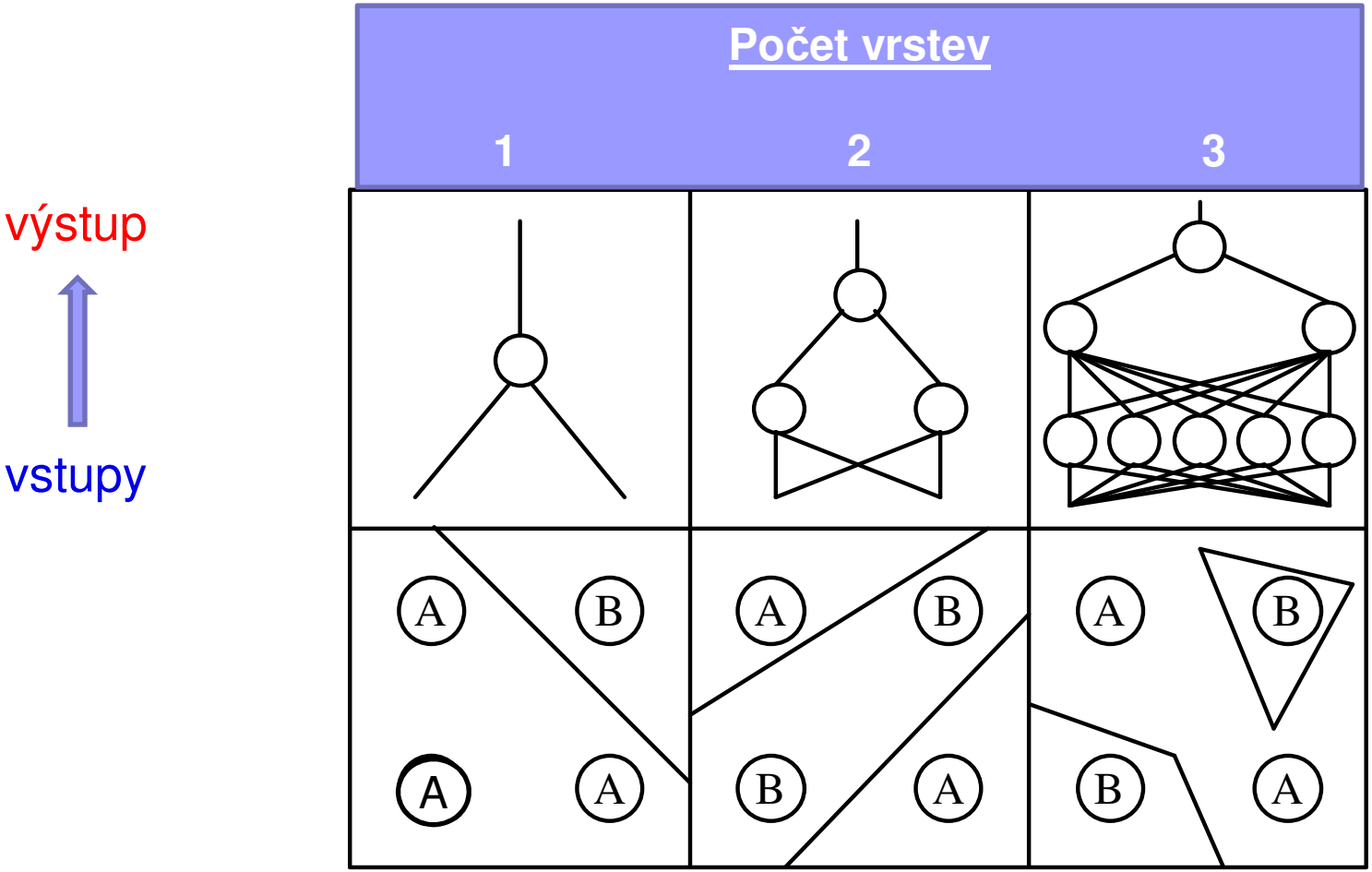
Vícevrstvá perceptonová síť



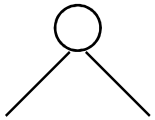
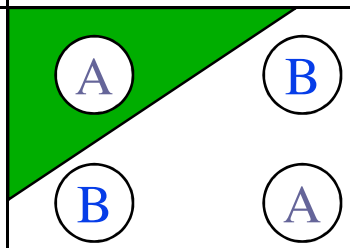
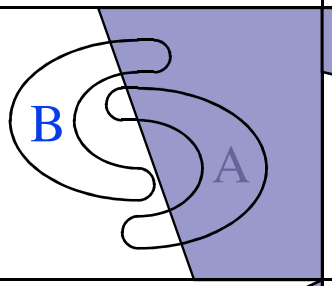
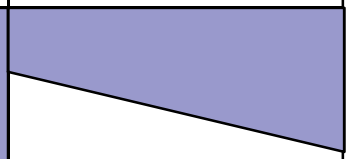
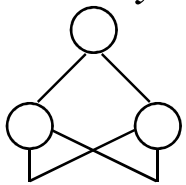
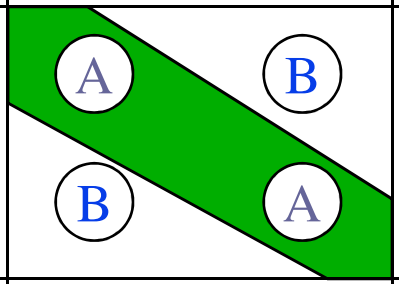
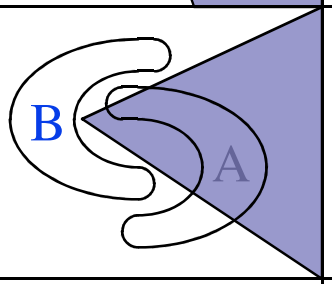
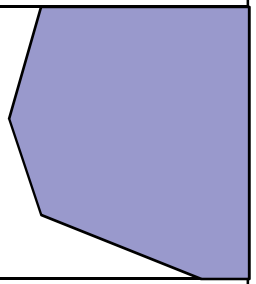
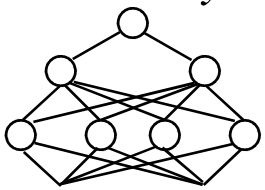
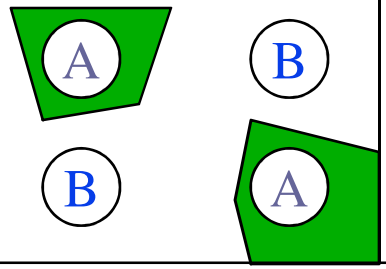
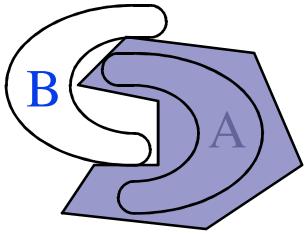
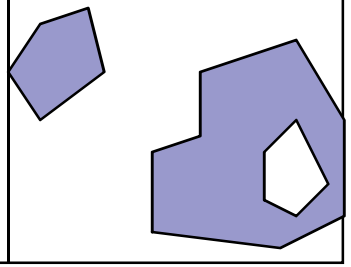
Vícevrstvá perceptonová síť



Vícevrstvá perceptonová síť



Příklady různých problémů, které nejsou lineárně separabilní

Structure	*Types of Decision Regions*	*Exclusive-OR Problem*	*Classes with Meshed regions*	*Most General Region Shapes*
Single-Layer 	*Half Plane Bounded By Hyperplane*			
Two-Layer 	*Convex Open Or Closed Regions*			
Three-Layer 	Abitrary (Complexity Limited by No. of Nodes)			

Vícevrstvá perceptonová síť

- **Věta** (Kolmogorov, 1957): Nechť $n > 1$ je přirozené číslo a f je spojitá reálná funkce. Potom lze tuto funkci reprezentovat vztahem

$$f(x_1, x_2, \ldots, x_n) = \sum_{j=1}^{2n+1} \varphi_j \left(\sum_{i=1}^{n} \psi_{ij}(x_i) \right)$$

  kde φ_j a ψ_{ij} jsou spojité funkce jedné proměnné.

- TEDY libovolnou rozumnou, tj. spojitou, funkci lze zapsat (reprezentovat) s pomocí do sebe vnořených funkcí jediné proměnné.

Kolmogorovův teorém

- Přitom pro danou funkci f jsou specifické pouze funkce Ψ_i, zatímco funkce φ_i jsou pro dané n a f nezávislé. Hecht-Nielsen ukázal, že tuto univerzální vlastnost funkcí φ_{pg} lze využít též pro reprezentaci funkcí s hodnotami v prostoru vyšších rozměrů.
- Kolmogorovův teorém doplňovali v pozdějších letech někteří další autoři, např.
  - Lorenz, který dokázal, že je možno vystačit pouze s jedinou funkcí Ψ
  - D.A. Sprecher, který našel podmínky pro to, aby funkce Ψ_q měly tvar $\lambda_p \varphi_p$, kde λ_p jsou konstanty.
- Důsledek aplikace Kolmogorovovy věty na problematiku neuronových sítí: K tomu, aby bylo transformační funkcí T neuronové sítě možno aproximovat libovolnou funkci f, postačí, aby příslušná **neuronová síť měla alespoň tři vrstvy o odpovídajících počtech neuronů** (výkonných prvků) **v jednotlivých vrstvách**.
- Funkci T lze tedy implementovat jako transformační funkci neuronové sítě, která má
  - nejméně tři vrstvy s dopřednou vzájemnou vazbou
  - + vstupní distribuční vrstvu.

Typy neuronových sítí

- Existuje celá řada neuronových sítí, které se liší architekturou a použitými stavebními prvky (perceptron, neuron s aktivační funkcí typu „radial base“, ..),např.
  - Vícevrstvá perceptonová síť (MLP)
  - Hopfieldova síť
  - Kohonenovy samoorganizující se mapy (SOM)
  - Síť RBF (radial bases functions)
  - ...
- Každý typ se hodí pro jinou třídu úloh
- Základními úlohami neuronových sítí jsou **klasifikace** a **regrese** (aproximace)
- Podle přítomnosti „učitele“ můžeme neuronové sítě dělit na sítě s učitelem a bez učitele

Proces učení neuronových sítí

- Pro učení (trénování NS) je třeba mít dostatek reprezentativních příkladů
- Trénovací, výběrová, testovací množina
- Na začátku učení bývají váhy nejčastěji nastaveny na náhodná čísla
- **Proces učení** se snaží minimalizovat odchylku (chybu) mezi skutečným (aktuálním) a požadovaným výstupem
- Každá neuronová síť má jiný algoritmus učení, vesměs jsou to ale **iterační procesy**

Backpropagation algoritmus

- Inicializuj váhy sítě malými náhod. čísly (např. z intervalu (-0,05 , 0,05))
- Cyklus opakovaný až do splnění kritéria pro zastavení

  Pro každý příklad [***x***,*y*] z trénovacích dat

  1. Spočítej výstup out_u pro každý neuron *u* v síti
  2. Pro každý neuron *v* ve výstupní vrstvě spočítej chybu

     $Err_v = out_v \,.\, (1 - out_v) \,.(y_v - out_v)$
  3. Pro každý neuron *s* ve skryté vrstvě spočítej chybu

     $Err_s = out_s \,.\, (1 - out_s) . \sum_{v \in vystup} (w_{s,v} . Err_v)$
  4. Pro každou vazbu vedoucí od neuronu *j* do neuronu *k* modifikuj váhu vazby

     $w_{j,k} = w_{j,k} + \Delta w_{j,k}$, kde $\Delta w_{j,k} = \eta \; Err_k \, x_{j,k}$

Obvyklé kritérium pro zastavení: Chyba sítě na validačních datech je menší než požadovaná hodnota.

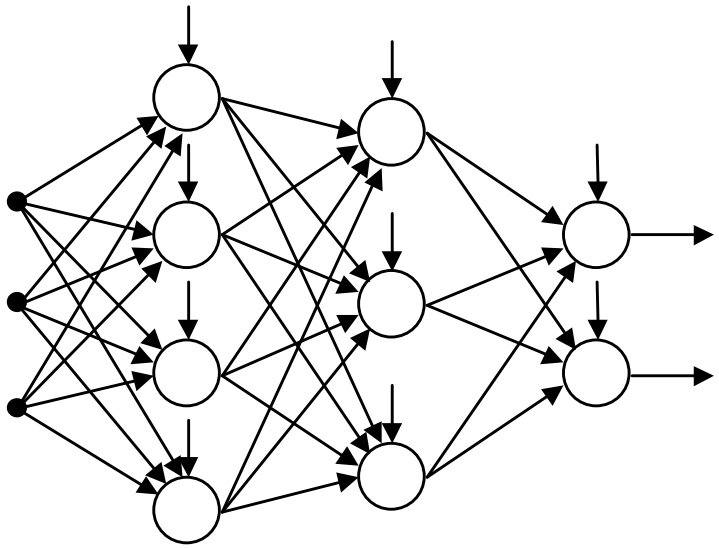
Návrh neuronové sítě

- Pro řešení každé úlohy musí být navržena jedinečná neuronová síť
- Otázka vhodného výběru sítě
- Výběr struktury sítě, tj. počet vstupů, výstupů, vrstev, skrytých neuronů, typ aktivačních funkcí, atd.
- Výběr trénovacího algoritmu
- Typické problémy *over-sizing*, *over-learning (over-fitting)*

Vícevrstvá perceptonová síť

- Nejrozšířenější a nejpoužívanější síť
- Jak pro klasifikaci tak pro regresi (a tedy i predikci spojitých funkcí, např. časových řad)
- Síť s učitelem
- Aktivační funkcí je nejčastěji **sigmoida**
- Otázka výběru počtu vrstev a počtu neuronů
- Kromě základního algoritmu *backpropagation* existuje řada sofistikovaných metod učení, např. metoda sdružených gradientů, Levenbergova-Marquardtova metoda atd.

Vícevrstvá perceptonová síť



- K nevýhodám sítě patří obtížné řešení problému lokálních minim a poměrně dlouhá doba učení
- Pro zlepšení práce se používá řada metod, např. změna architektury (doplnění dalších neuronů, ..), využití momentu, šumu, …

Síť RBF *

- Radial Basis Function (RBF), síť radiálních jednotek
- Pevný počet vrstev
- Dva typy neuronů: radiální a perceptronového typu (nejčastěji lineární)
- Váhy v první vrstvě jsou nastavovány pevně na začátku učení, ve druhé vrstvě se postupuje podobně jako u vícevrstvé perceptronové sítě nebo přímo regresí

Síť RBF *

- Postsynaptický potenciál

$$\xi = \frac{\left\| \vec{x} - \vec{c} \right\|}{b}$$

- Aktivační funkce

$$\Phi(\xi) = e^{-\xi^2}$$

- Adaptace vah

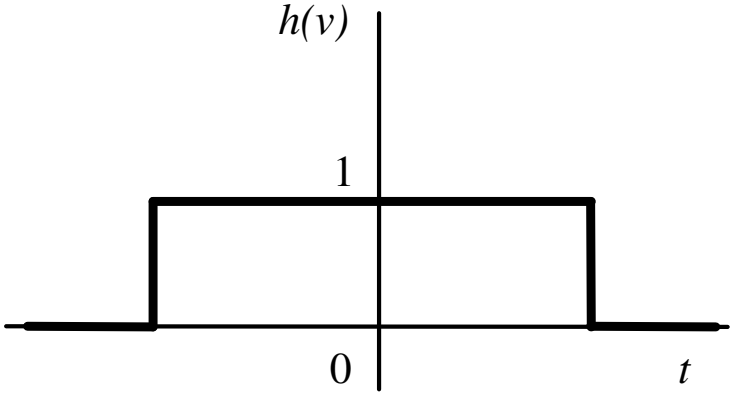
$$y_k = w_{0k} + \sum_{j=1}^{h} w_{jk} \Phi_j(\xi) = \sum_{j=0}^{h} w_{jk} \Phi_j(\xi)$$

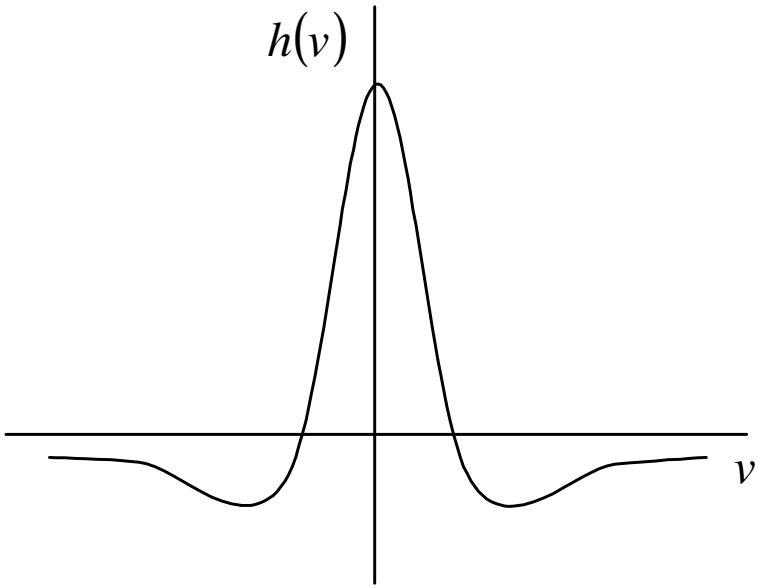
Kohonenova síť *

- Kohonenova síť (SOM – self organizing map, SOFM)
- Bez učitele, provádí proto pouze analýzu vstupních dat, přesněji druh shlukové analýzy
- Obsahuje jedinou vrstvu radiálních neuronů, které mohou být uspořádaný to tzv. mřížky
- Síť je možné rozšířit tak, aby byla schopna klasifikace (Learning Vector Quantization – LVQ)

Kohonenova síť *

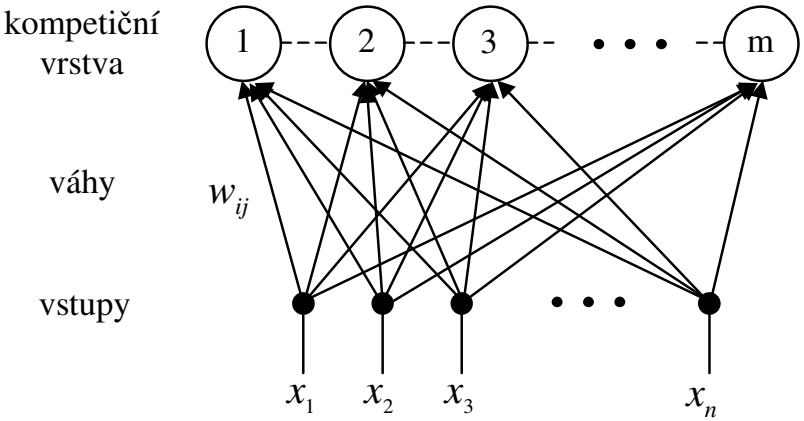
- Adaptační funkce

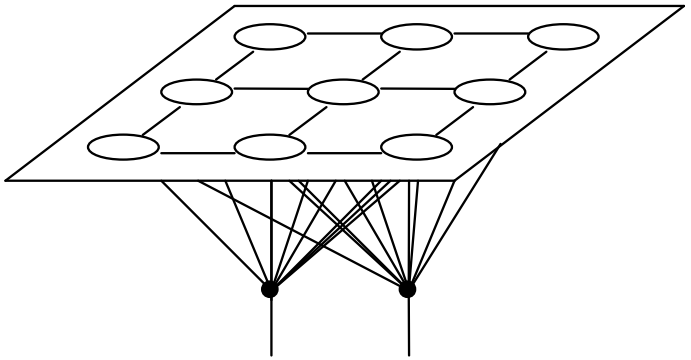




Kohonenova síť *

- Struktura Kohonenovy sítě

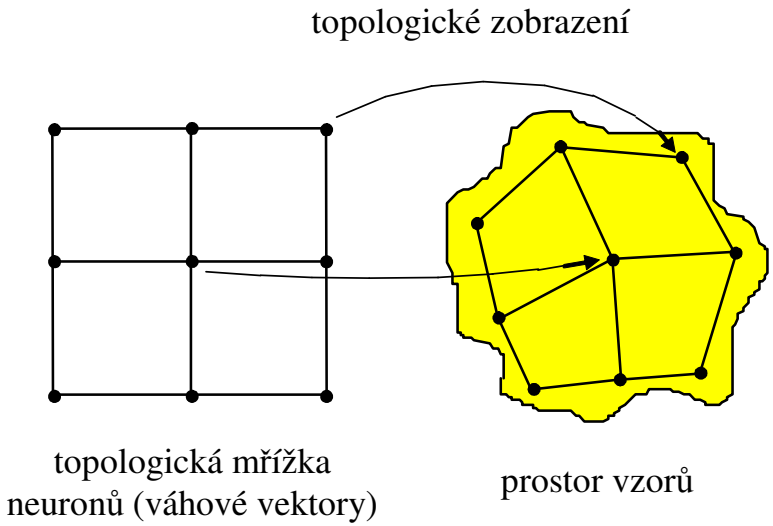




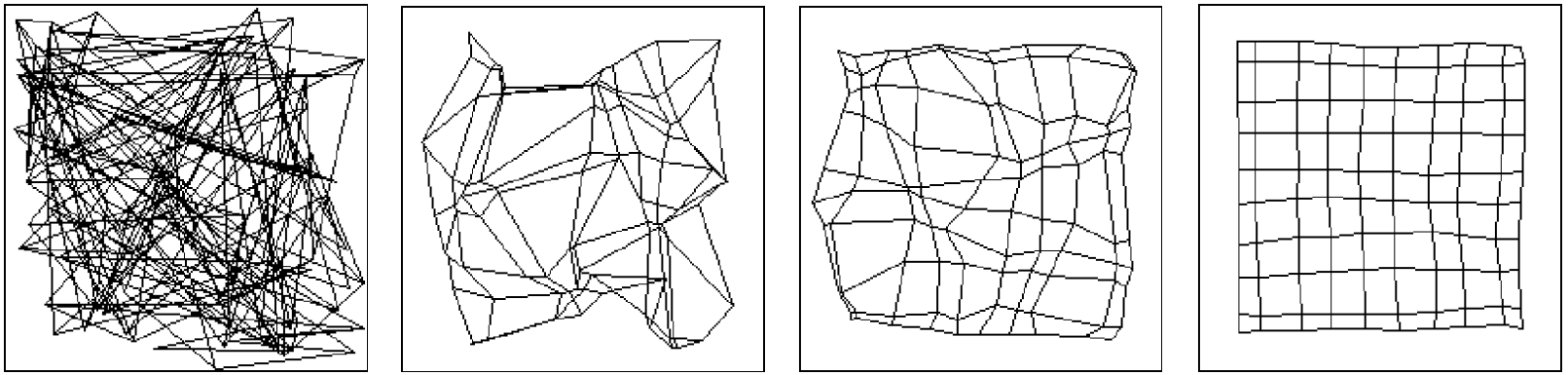
$$d_j = \sum_{i=1}^{m} \left(x_i - w_{ij} \right)^2$$

$$d_{j^*} = \min_j \left(d_j \right)$$

Kohonenova síť *



- Proces učení (adaptace vah)



Kohonenova síť *

- Vzdálenosti vzoru k neuronům

$$d_j = \sum_{i=1}^{m} \left(x_i - w_{ij} \right)^2$$

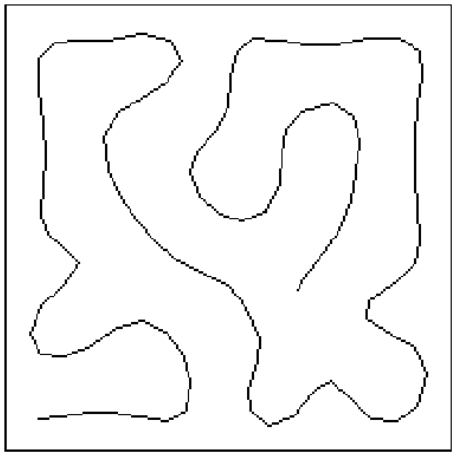
- Výběr nejbližšího neuronu

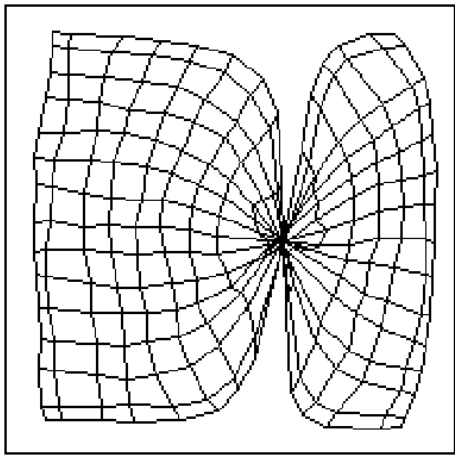
$$d_{j^*} = \min_j \left(d_j \right)$$

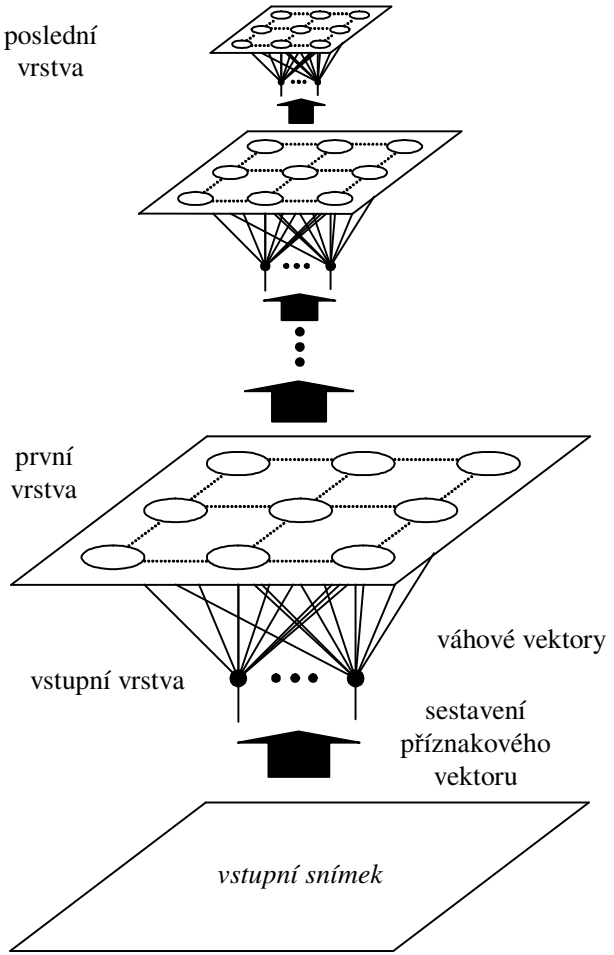
- Adaptace vah

$$w_{ij}(t+1) = w_{ij}(t) + \eta(t) h(v,t) \left(x_i(t) - w_{ij}(t) \right)$$

Kohonenova síť *





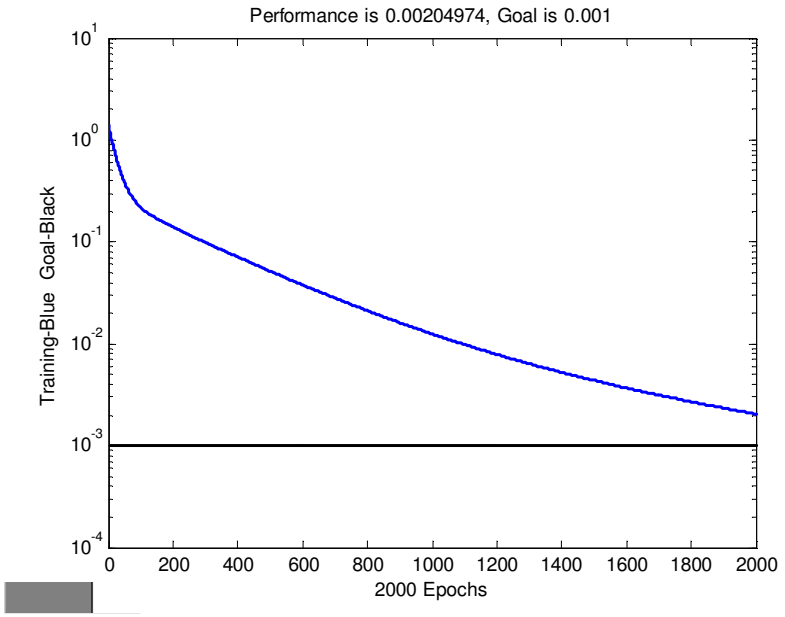


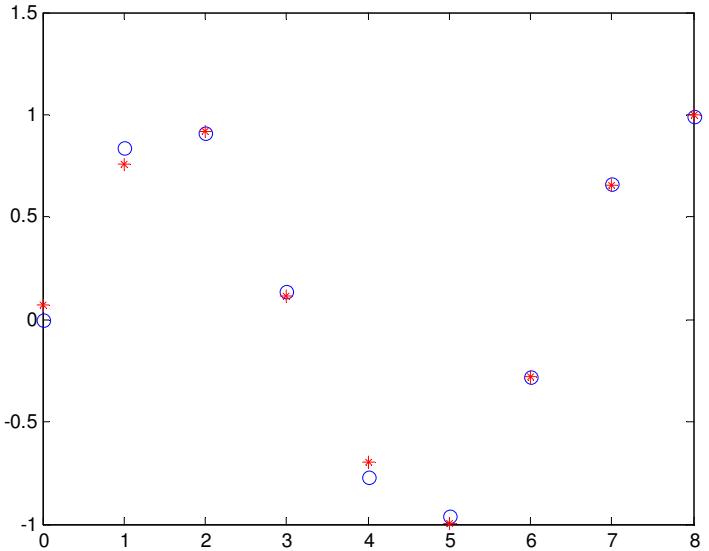
Hopfieldova síť *

- Navržena J. Hopfieldem v roce 1982
- Autoasociativní paměť
- Pracuje s bipolárními (binárními) hodotami vstupů/výstupů
- Spojitá varianta Hopfieldovy sítě se používá pro řešení optimalizačních problémů

Softwarové prostředky pro NS

- Matlab Neural Network Toolbox
- Statistica Neural Networks





History of Artificial Neural Networks (ANNs)

- Pre-1940: von Hemholtz, Mach, Pavlov, etc.
  - General theories of learning, vision, conditioning
  - No specific mathematical models of neuron operation
- **1940s: Hebb, McCulloch and Pitts**
  - Hebb: Explained mechanism for learning in biological neurons
  - McCulloch and Pitts: First neural model
- **1950s: Rosenblatt, Widrow and Hoff**
  - First practical networks (Perceptron and Adaline) and corresponding learning rules
- **1960s: Minsky and Papert**
  - Demonstrated limitations of existing neural networks
  - New learning algorithms not forthcoming, **most research suspended**
- 1970s: Amari, Anderson, Fukushima, Grossberg, Kohonen
  - Progress continues, although at a slower pace
- 1980s: **Grossberg, Hopfield, Kohonen, Rumelhart, etc.**
  - Important new developments cause a **resurgence in the field** (Backpropagation algorithm)

Literatura

- *české učebnice*
- Mařík V., Štěpánková O., Lažanský J. a kol.: *Umělá inteligence 4*. Academia, Praha, 2003.
- Šíma J., Neruda R.: *Teoretické otázky neuronových sítí*. Matfyzpress, Praha, 1996.
- Tučková J.: *Vybrané aplikace neuronových sítí při zpracování signálů*, Nakladatelství ČVUT, Praha 2009

- *zahraniční učebnice*
- Bishop C. M.: *Neural Networks for Pattern Recognition*. Oxford University Press, NewYork, 1995.
- Fausett, L.: *Fundamentals of Neural Networks*. Prentice Hall, New York, 1994.
- Hassoun M. H.: *Fundamentals of Artificial Neural Networks*. The MIT Press, Cambridge, Massachusetts, London, 1995.
- Haykin, S.: *Neural Networks: A Comprehensive Foundation*. Macmillan Publishing, New York, 1994.
- Rojas R.: *Neural Networks: A Systematic Introduction*. Springer-Verlag, Berlín, Heidelberg, New York, 1996.