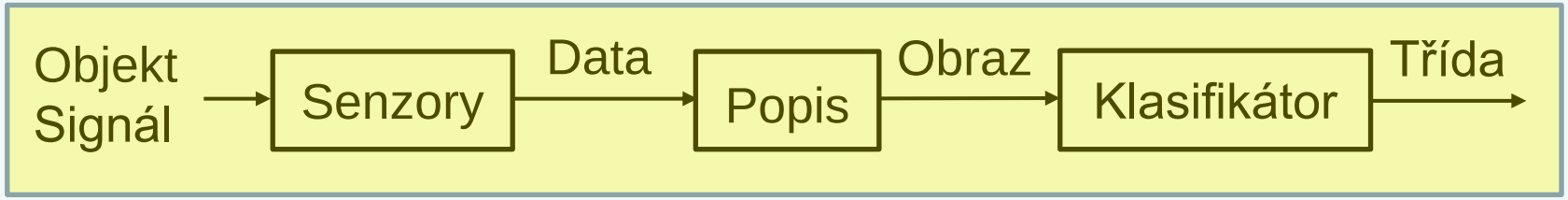
# 9. Rozpoznávání.

# Rozpoznávání obrazů (Pattern Recognition)

Rozpoznáváním se rozumí klasifikace objektů/signálů do tříd na základě jejich popisů (obrazů):



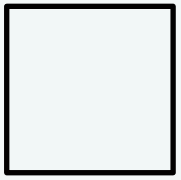
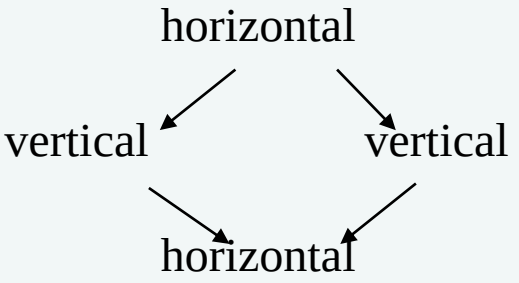
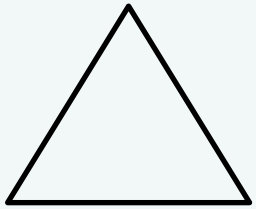
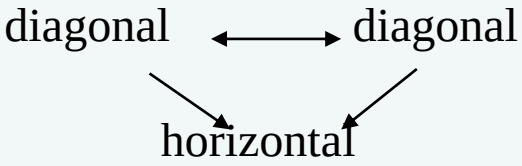

Existují dva základní typy popisů rozpoznávaných objektů/signálů:

- Příznakový: popis vektory číselných příznaků
- Strukturální/syntaktický: popis strukturálními prvky, tzv. primitivy

a také dva rozdílné základní přístupy k vlastní klasifikaci:

- Klasické statistické a syntaktické metody
- Neuronové sítě (předmět Soft-Computing)

# Příklad příznakového a strukturálního popisu stejných objektů:

| příznakový popis | | objekt | strukturální popis |
|---|---|---|---|
| počet segmentů | 4 |  |  |
| počet horizontálních segmentů | 2 | | |
| počet vertikálních segmentů | 2 | | |
| počet diagonálních segmentů | 0 | | |
| vektor příznaků | (4,2,2,0) | | |
| počet segmentů | 3 |  |  |
| počet horizontálních segmentů | 1 | | |
| počet vertikálních segmentů | 0 | | |
| počet diagonálních segmentů | 2 | | |
| vektor příznaků | (3,1,0,2) | | |

# Příznakové rozpoznávání

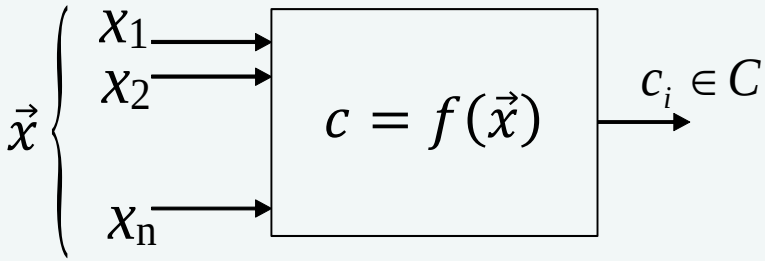
Vektor číselných příznaků $\vec{x} = (x_1, x_2, \ldots, x_n)$

Množina tříd $C = \{c_1, c_2, \ldots, c_R\}$

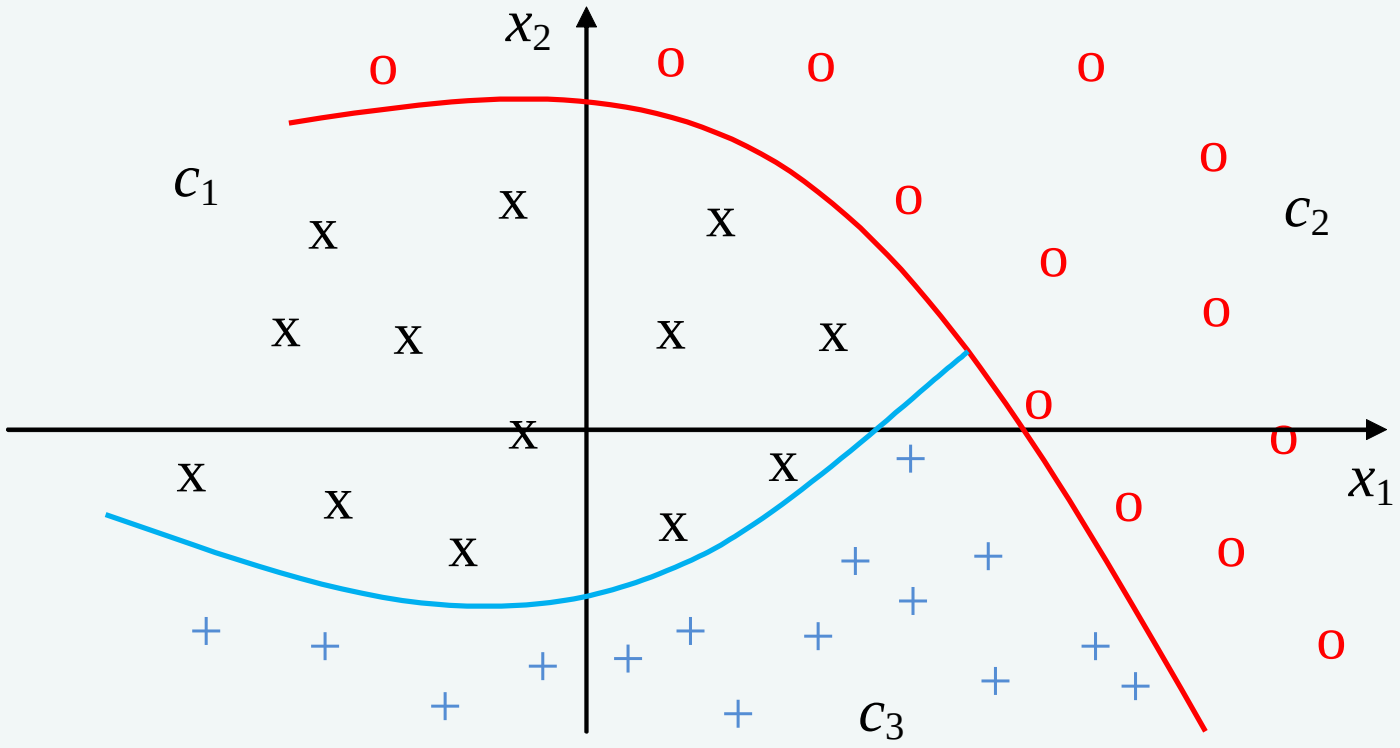
Cílem příznakového rozpoznávání je určit třídu $c_i$, do které vektor příznaků $\vec{x}$ patří:

$$\vec{x} \rightarrow c_i\,, \;\; c_i \in C$$

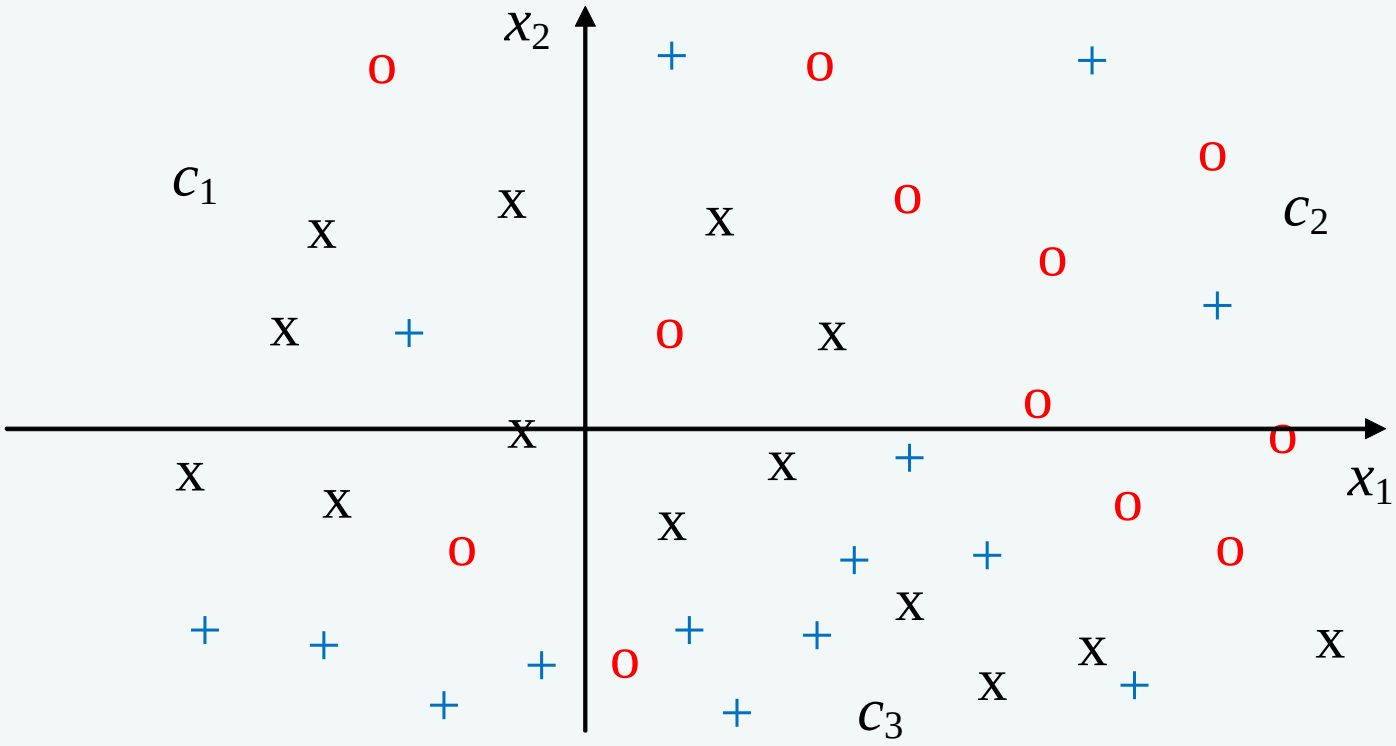
Příznakový klasifikátor:



Třídy obrazů vzájemně od sebe oddělitelné spojitými hyper-plochami (separovatelné obrazy):



Třídy obrazů vzájemně od sebe neoddělitelné spojitými hyper-plochami (neseparovatelné obrazy):



## Klasifikace separovatelných obrazů

*Klasifikace pomocí diskriminačních funkcí*

Diskriminační (rozlišující) funkce jsou takové funkce $g_i$ ($i = 1 \ldots R$, kde $R$ je počet tříd), které pro obraz (vektor příznaků) $\vec{x}$ patřící do třídy $r$ splňují podmínku $g_r(\vec{x}) > g_s(\vec{x}), \quad r, s \in C, \; r \neq s$.

Pro nadplochy rozdělující třídy $r$ a $s$ musí zřejmě platit vztah

$$g_r(\vec{x}) = g_s(\vec{x}), \quad r, s \in C, \; r \neq s$$

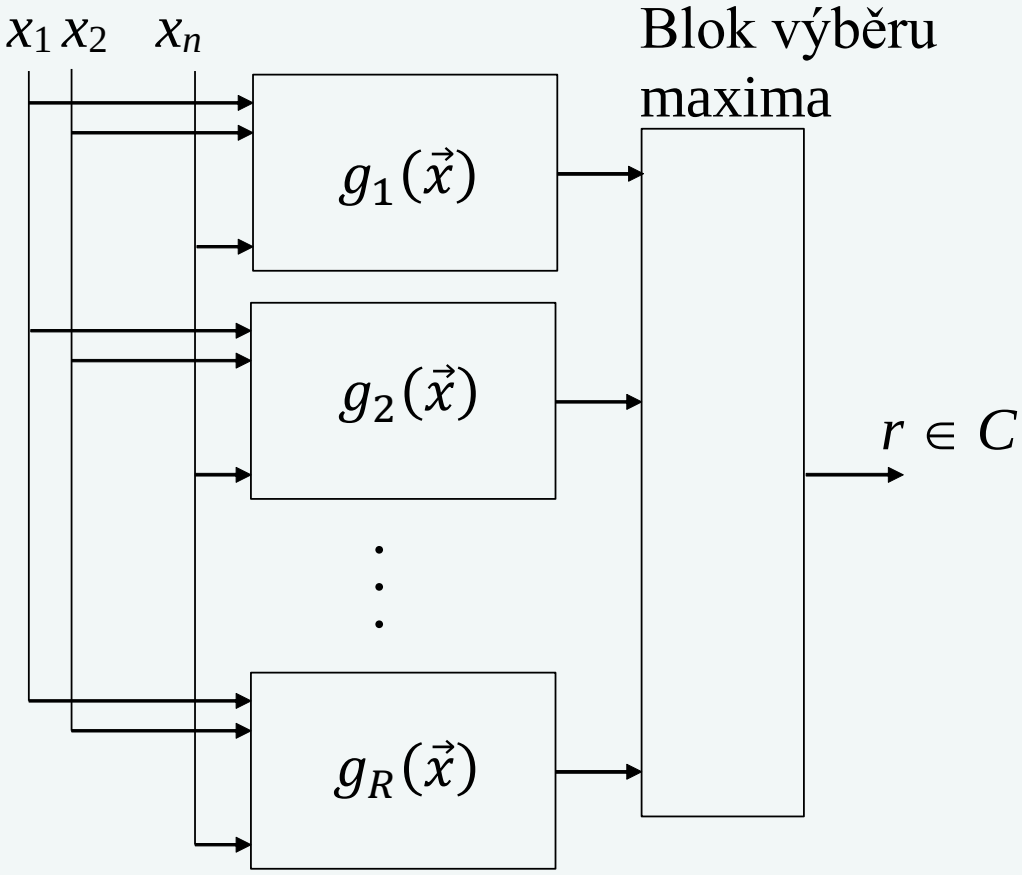
Pro lineárně separovatelné obrazy se používají lineární diskriminační funkce:

$$g_r(\vec{x}) = q_{r,0} + q_{r,1}x_1 + q_{r,2}x_2 + \cdots + q_{r,n}x_n =$$

$$= q_{r,0} + \sum_{i=1}^{n} q_{r,i}x_i \; = q_{r,0} + \vec{q}_r\vec{x},$$

$$\vec{q}_r = \left(q_{r,1}, q_{r,2}, \ldots, q_{r,n}\right)$$

Příznakový klasifikátor založený na diskriminačních funkcích:



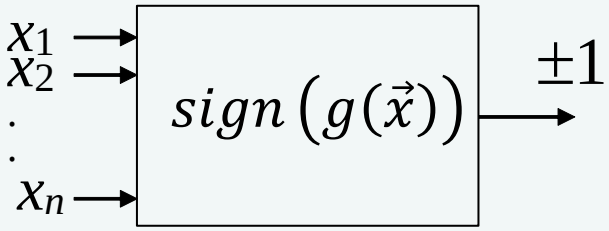
Dichotomie – klasifikace do dvou tříd

$$g(\vec{x}) = g_1(\vec{x}) - g_2(\vec{x})$$

$g(\vec{x}) > 0 \qquad pro\ \vec{x}\ t\check{r}\acute{i}dy\ 1$

$g(\vec{x}) < 0 \qquad pro\ \vec{x}\ t\check{r}\acute{i}dy\ 2$

Klasifikátor pro dichotomii:



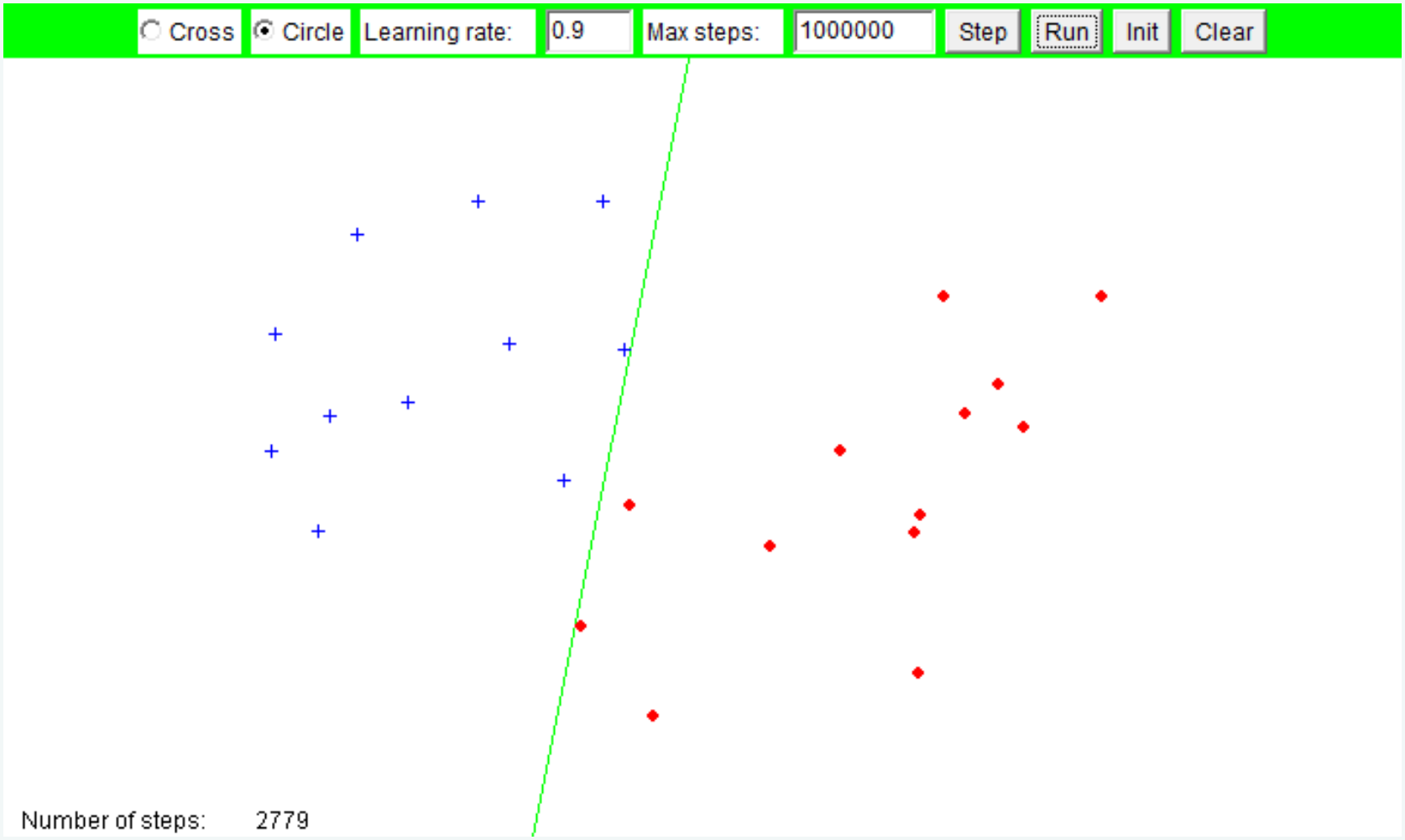
## Algoritmus učení lineárního klasifikátoru pro dichotomii

Nechť $T = \{(\vec{x}_1, c_1), (\vec{x}_2, c_2), \ldots, (\vec{x}_P, c_P)\}$ je trénovací množina, ($c_i \in$ <-1, 1>, $\vec{x} = (x_0, x_1, x_2, \ldots, x_n)$, $x_0 = 1$) a $\mu$ je konstanta:

1. Vynulujte vektor vah.
2. Nastavte indikátor změny *modif* = false.
3. Pro každý vektor $\overrightarrow{x_i}$ z trénovací množiny ($i$ = 1 ... $P$), který není správně klasifikován ($sign\left(g(\vec{x}_i)\right) \neq c_i$) upravte vektor vah podle vztahu

$$\vec{q} := \vec{q} + \frac{\mu\, c_i \vec{x}_i}{\|\vec{x}_i\|} \qquad \|\vec{x}\| = \sqrt{x_0^2 + x_1^2 + \ldots + x_n^2}$$

   a nastavte indikátor změny *modif* = true.
4. Pokud došlo k úpravě vektoru vah (*modif* = = true), vraťte se na bod 2.


Cross
Circle
Learning rate:
0.9
Max steps:
1000000
Step
Run
Init
Clear
Number of steps:
2779

*Klasifikace podle minima vzdáleností* (etalony tříd: $\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_R$)

Pro klasifikaci vektoru $\vec{x}$ do třídy $r$ musí platit:

$$\|\vec{v}_r - \vec{x}\| = \min_{i=1\ldots R}(\|\vec{v}_i - \vec{x}\|) = \min_{i=1\ldots R}\left(\sqrt{(\vec{v}_i - \vec{x})(\vec{v}_i - \vec{x})}\right)$$

a tedy i

$$\|\vec{v}_r - \vec{x}\|^2 = \min_{i=1\ldots R}(\|\vec{v}_i - \vec{x}\|^2) = \min_{i=1\ldots R}\left((\vec{v}_i - \vec{x})(\vec{v}_i - \vec{x})\right) =$$

$$= \min_{i=1\ldots R}(\vec{v}_i\vec{v}_i - 2\vec{v}_i\vec{x} + \vec{x}\vec{x}) \Rightarrow$$

$$\min_{i=1\ldots R}(\vec{v}_i\vec{v}_i - 2\vec{v}_i\vec{x}) = \max_{i=1\ldots R}(-\vec{v}_i\vec{v}_i + 2\vec{v}_i\vec{x}) \Rightarrow$$

$$\max_{i=1\ldots R}\left(-\frac{1}{2}\vec{v}_i\vec{v}_i + \vec{v}_i\vec{x}\right) = g_r(\vec{x}) \Rightarrow$$

$$g_r(\vec{x}) = q_{r,0} + \vec{q}_r\vec{x} \qquad q_{r,0} = -\frac{1}{2}\vec{v}_r\vec{v}_r, \quad \vec{q}_r = \vec{v}_r$$

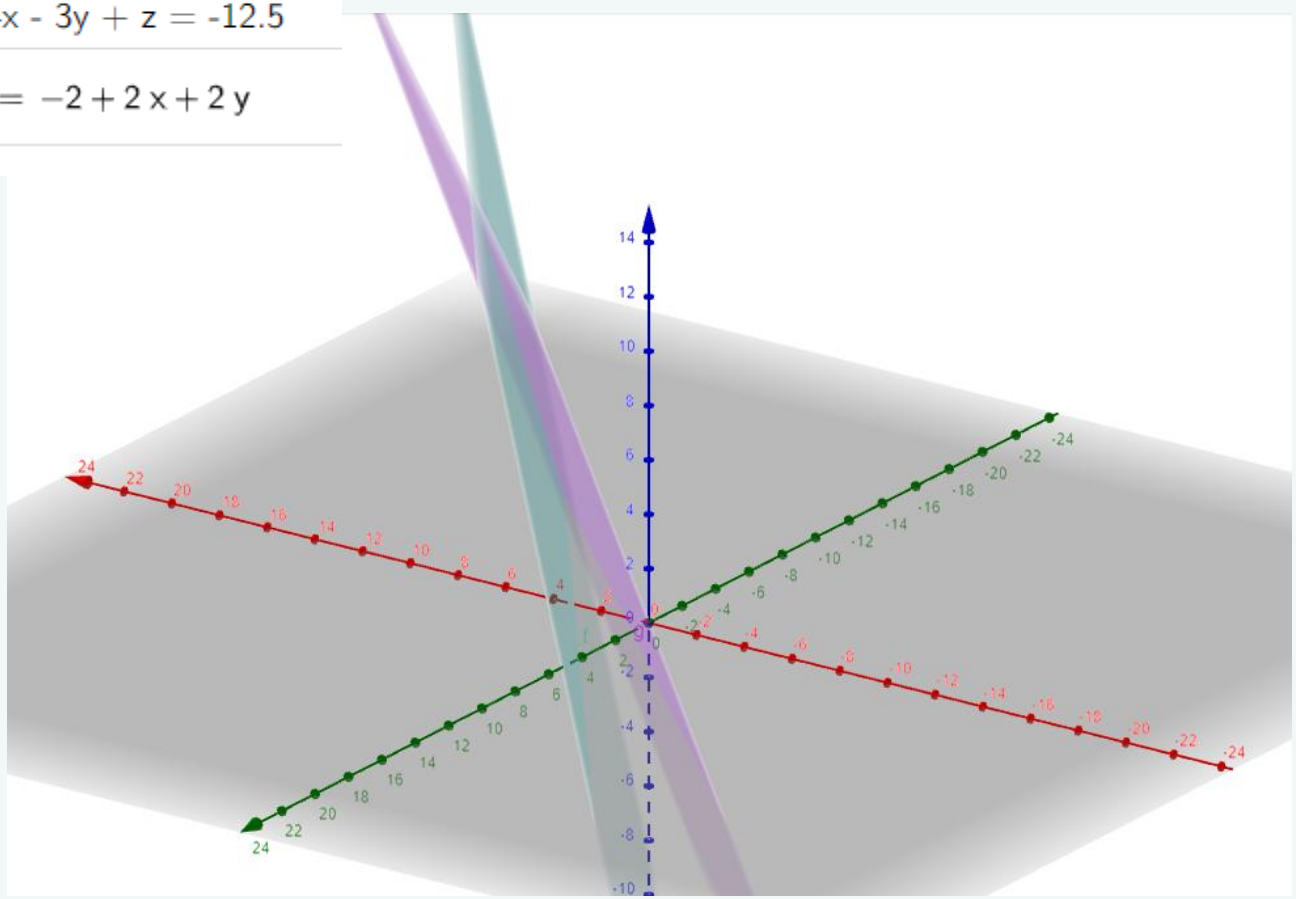
Rozdělující nadroviny jsou kolmé k úsečkám $\vec{v}_r - \vec{v}_s$ $(r, s \in C,\ r \neq s)$ a půlí je (viz k-means)!

# Rozdělující nadroviny pro [4,3] a [2,2]

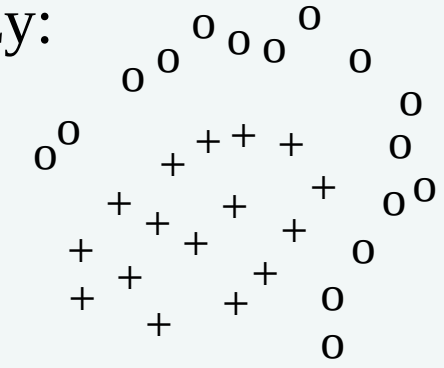
$f : z = -12.5 + 4x + 3y$

→ -4x - 3y + z = -12.5

$g : z = -2 + 2x + 2y$



Lineárně neseparovatelné obrazy:



a) Několik lineárních diskriminačních funkcí pro každou třídu *r*:
$g_r(\vec{x}) = \max_{h=1\ldots H}\left(g_r^h(\vec{x})\right)$

b) Několik etalonů pro každou třídu *r*: $\vec{v}_{11}, \vec{v}_{12}, \ldots, \vec{v}_{1n}, \vec{v}_{21}, \vec{v}_{22}$ ...

c) Obecné diskriminační funkce (prakticky nerealizovatelné):
$g_r(\vec{x}) = a_r + b_r \ln(x_1) + c_r\, tg(d_r x_2) + \ldots$

d) Nelineární převodník transformující původní *n*-dimenzionální obrazový prostor na *m*-dimenzionální obrazový prostor, ve kterém pak lze obrazy separovat lineárními nadrovinami a použít klasifikátor s lineárními diskriminačními funkcemi (například neuronové sítě).

## Klasifikace neseparovatelných obrazů

Nechť $\lambda(r,s)$ je ztráta uživatele, klasifikuje-li obraz třídy $s$ nesprávně do třídy $r$. Obecná ztrátová matice má tvar

$$\lambda = \begin{bmatrix} \lambda(1,1) & \lambda(1,2) & \cdots & \lambda(1,R) \\ \lambda(2,1) & \lambda(2,2) & \cdots & \lambda(2,R) \\ \vdots & \vdots & \vdots & \vdots \\ \lambda(R,1) & \lambda(R,2) & \cdots & \lambda(R,R) \end{bmatrix}$$

Předpokládejme, že $\lambda(i,j) = 0$ pro $i = j$ a $\lambda(i,j) = 1$ pro $i \# j$ . Pak se tato matice zjednoduší

$$\lambda = \begin{bmatrix} 0 & 1 & \cdots & 1 \\ 1 & 0 & \cdots & 1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & \cdots & 0 \end{bmatrix}$$

Dále nechť $p(\vec{x})$ je pravděpodobnost výskytu obrazu $\vec{x}$ a $p(s|\vec{x})$ podmíněná pravděpodobnost toho, že obraz $\vec{x}$ patří do třídy *s*.

Příspěvek ztráty $\lambda(r,s)$ k celkové střední ztrátě způsobené zařazením obrazu do třídy *r* je zřejmě dán vztahem

$$\lambda(r,s)p(s|\vec{x})p(\vec{x})$$

a pro celkovou střední ztrátu při klasifikaci obrazu $\vec{x}$ do třídy *r* pak lze psát

$$\begin{aligned} L_{\vec{x}}(r) &= \sum_{s=1}^{R} \lambda(r,s)p(s|\vec{x})p(\vec{x}) = \\ &= \sum_{s=1}^{R} \lambda(r,s)p(\vec{x}|s)p(s) = \\ &= \sum_{s=1}^{R} p(\vec{x}|s)p(s) - p(\vec{x}|r)p(r) = \\ &= p(\vec{x}) - p(\vec{x}|r)p(r) \end{aligned}$$

Z požadované minimalizace celkové střední ztráty

$$\min\left(L_{\vec{x}}(r)\right) = \max\left(p(\vec{x}|r)p(r)\right)$$

přímo vyplývá vztah pro diskriminační funkci

$$g_r(\vec{x}) = p(\vec{x}|r)p(r)$$

a s uvažováním skutečnosti, že logaritmus většího kladného čísla má vždy větší hodnotu než logaritmus menšího kladného čísla, pak často používaný vztah

$$g_r(\vec{x}) = \ln\left(p(\vec{x}|r)p(r)\right) = \ln\left(p(\vec{x}|r)\right) + \ln\left(p(r)\right)$$

Pro vícerozměrné normální rozložení hustoty pravděpodobnosti

$$p(\vec{x}|r) = \frac{1}{(2\pi)^{\frac{n}{2}}\sqrt{|\bar{\sigma}_r|}} e^{-\frac{1}{2}(\vec{x}-\vec{\mu}_r)^T \bar{\sigma}_r^{-1}(\vec{x}-\vec{\mu}_r)}$$

má diskriminační funkce tvar

$$g_r(\vec{x}) = \ln\big(p(\vec{x}|r)\big) + \ln\big(p(r)\big) =$$
$$= -\frac{1}{2}(\vec{x}-\vec{\mu}_r)^T\, \bar{\sigma}_r^{-1}(\vec{x}-\vec{\mu}_r) - \frac{1}{2}\ln(|\bar{\sigma}_r|) + \ln\big(p(r)\big)$$

Pozn.: Logaritmus konstanty $1/(2\pi)^{\frac{n}{2}}$ je možné vypustit, protože jeho hodnota je stejná pro všechny obrazy i třídy, a na porovnání proto nemá žádný vliv.

Algoritmus učení klasifikátoru metodou odhadu pro normální rozložení hustoty pravděpodobnosti (nechť $R$ je počet tříd a nechť $T = \{(\vec{x}_1, c_1), (\vec{x}_2, c_2), \ldots, (\vec{x}_P, c_P)\}$ je trénovací množina):

for ( $r = 1; r <= R; r{+}{+}$) $\{\vec{\mu}_r = \vec{0}\,; \quad \bar{\sigma}_r = \bar{0}\,; \quad P_r = 0\,;\}$

for $(i = 1; i <= P; i{+}{+})$ {

$\vec{\mu} = \vec{\mu}_{c_i}; \quad \vec{\mu}_{c_i} = \left(P_{c_i}\vec{\mu}_{c_i} + \vec{x}_i\right) / \left(P_{c_i} + 1\right);$

if $(P_{c_i} > 0)$ {

$\vec{v} = \left(\vec{x}_i - \vec{\mu}_{c_i}\right);\ \vec{w} = \left(\vec{\mu} - \vec{\mu}_{c_i}\right);$

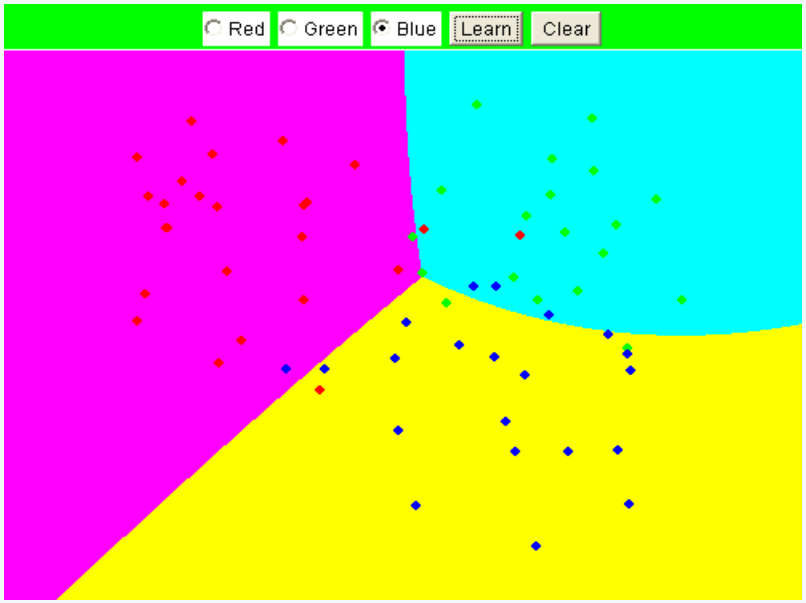
$\bar{\sigma}_{c_i} = \left(\left(P_{c_i} - 1\right)\bar{\sigma}_{c_i} + \vec{v}^T\vec{v} + P_{c_i}\vec{w}^T\vec{w}\right) / P_{c_i};$

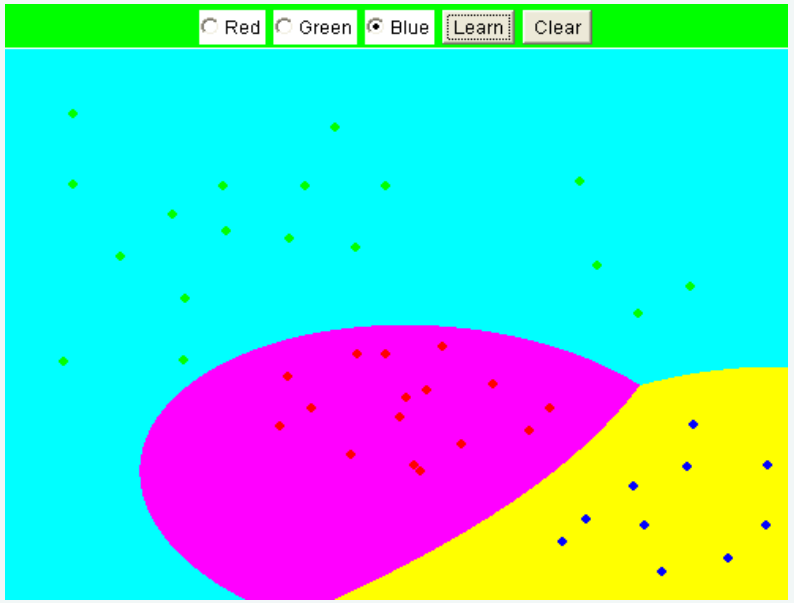
}

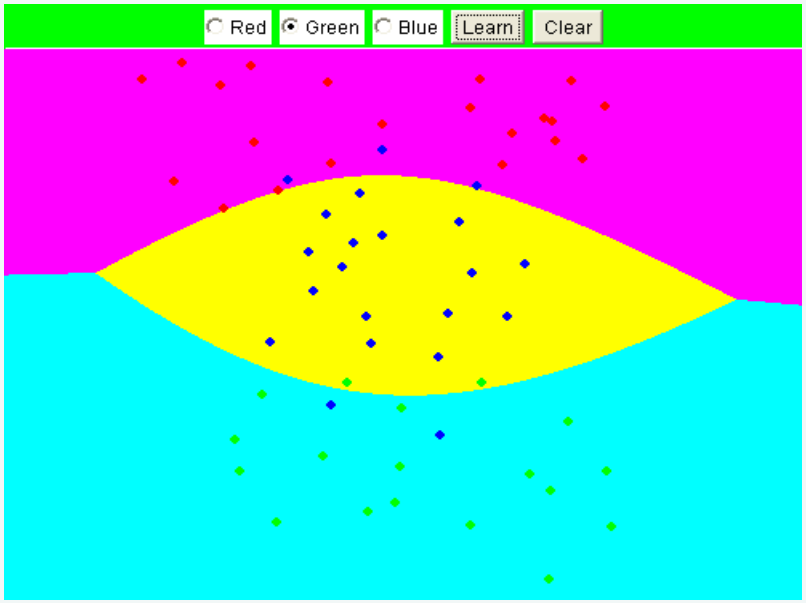
$P_{c_i}{+}{+};$

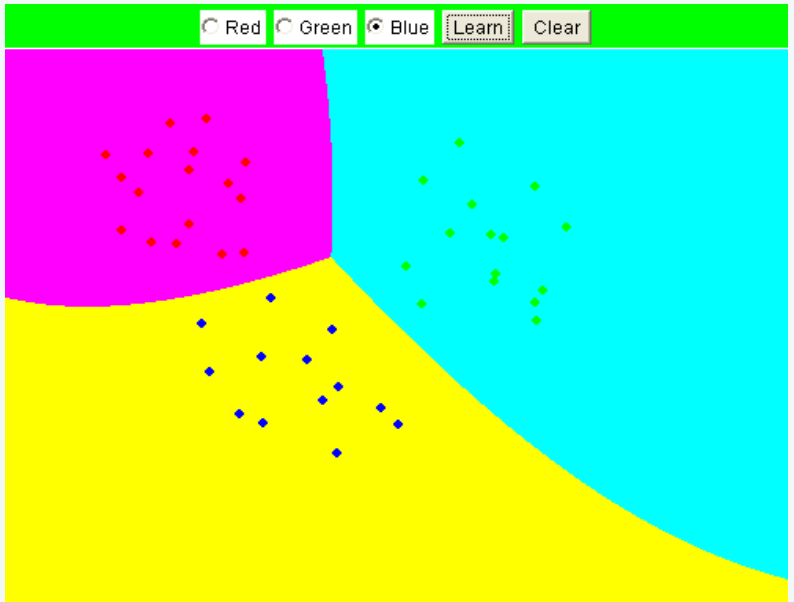
}

for $(r = 1; r <= R; r{+}{+})$ { $p(r) = P_r/P$ ; }


Red
Green
Blue
Learn
Clear


Red
Green
Blue
Learn
Clear

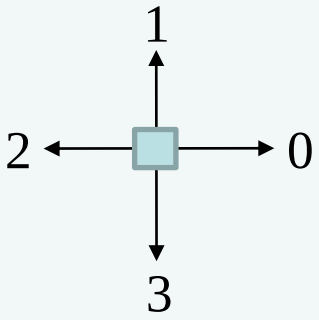

Red
Green
Blue
Learn
Clear


Red
Green
Blue
Learn
Clear

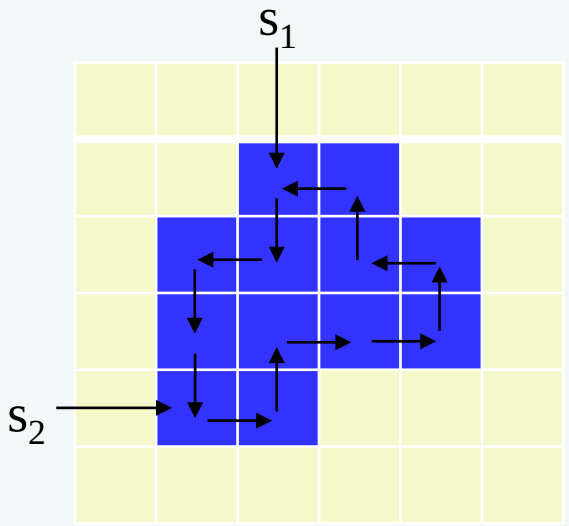
Strukturální popis obrysu objektu (příklad):

Freemanův kód (definice primitiv označených znaky číslic):

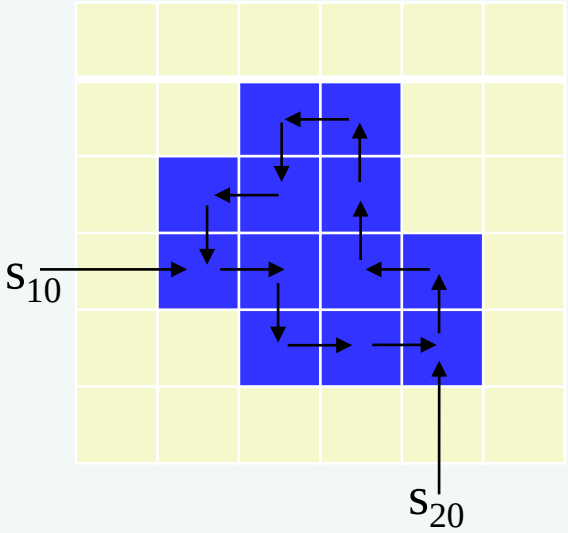
4-sousedé




Objekt:




Objekt otočený o 90º:




Absolutní řetězcový kód (postupujeme například proti směru hodinových ručiček), startovací bod:

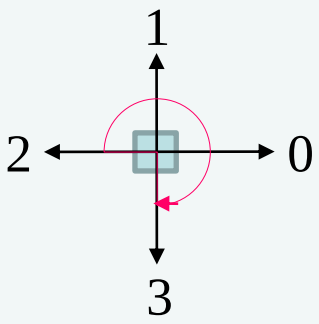
$s_1$: $[3\ 2\ 3\ 3\ 0\ 1\ 0\ 0\ 1\ 2\ 1\ 2]_{abs}$

$s_2$: $[0\ 1\ 0\ 0\ 1\ 2\ 1\ 2\ 3\ 2\ 3\ 3]_{abs}$

$s_{10}$: $[0\ 3\ 0\ 0\ 1\ 2\ 1\ 1\ 2\ 3\ 2\ 3]_{abs}$

$s_{20}$: $[1\ 2\ 1\ 1\ 2\ 3\ 2\ 3\ 0\ 3\ 0\ 0]_{abs}$

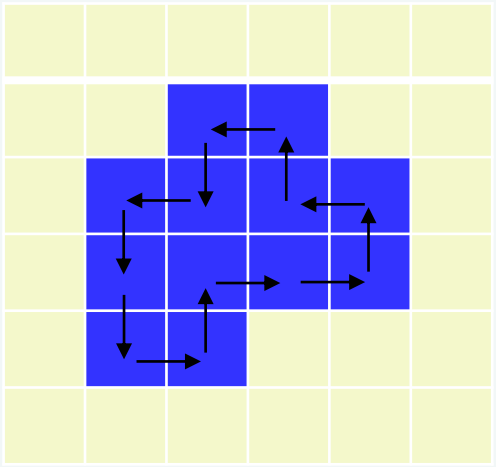
Absolutní řetězcový kód je invariantní vůči posuvu, je však závislý jak na natočení objektu, tak na startovacím bodu popisu, a pro rozpoznání je tak prakticky nepoužitelný. Problém natočení lze vyřešit pomocí diferenciálního (relativního) kódu, který místo primitiv používá rozdílů/diferencí natočení mezi dvěma sousedními primitivy (následující - aktuální), přičemž pro určení těchto rozdílů se postupuje v opačném směru (tj. po směru hodinových ručiček); pro první znak řetězce je předcházejícím znakem znak poslední:




2 – 3 → 3

$s_1$: [3 2 3 3 0 1 0 0 1 2 1 2]$_{abs}$
$s_1$: [3 1 0 1 1 3 0 1 1 3 1 1]$_{dif}$
$s_2$: [0 1 0 0 1 2 1 2 3 2 3 3]$_{abs}$
$s_2$: [1 3 0 1 1 3 1 1 3 1 0 1]$_{dif}$
$s_{10}$: [0 3 0 0 1 2 1 1 2 3 2 3]$_{abs}$
$s_{10}$: [3 1 0 1 1 3 0 1 1 3 1 1]$_{dif}$
$s_{20}$: [1 2 1 1 2 3 2 3 0 3 0 0]$_{abs}$
$s_{20}$: [1 3 0 1 1 3 1 1 3 1 0 1]$_{dif}$

Diferenciální kód vyřešil pouze problém invariantnosti vůči natočení. Problém invariantnosti vůči startovacímu bodu lze vyřešit rotací řetězce tak, aby výsledkem bylo největší číslo, tzv. číslo tvaru (shape number):



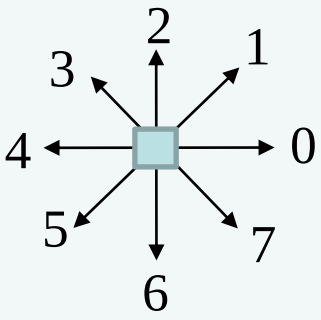
$s_1 = s_{10}$ : $[3\ 1\ 0\ 1\ 1\ 3\ 0\ 1\ 1\ 3\ 1\ 1]_{dif}$

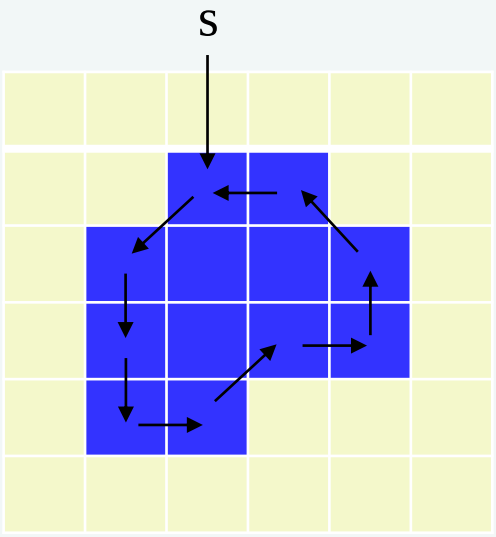
$s_2 = s_{20}$ : $[1\ 3\ 0\ 1\ 1\ 3\ 1\ 1\ 3\ 1\ 0\ 1]_{dif}$

shape number: [3 1 1 3 1 0 1 1 3 0 1 1]

Číslo tvaru je invariantní vůči posunutí, natočení objektu (pro 4-sousedy po 90º) a startovacímu bodu popisu, je však závislé na velikosti objektu.

Freemanův kód pro 8-sousedů:



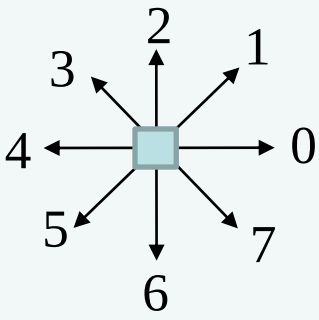

Absolutní řetězcový kód pro 8-sousedů (postupujeme opět proti směru hodinových ručiček), startovací bod s:

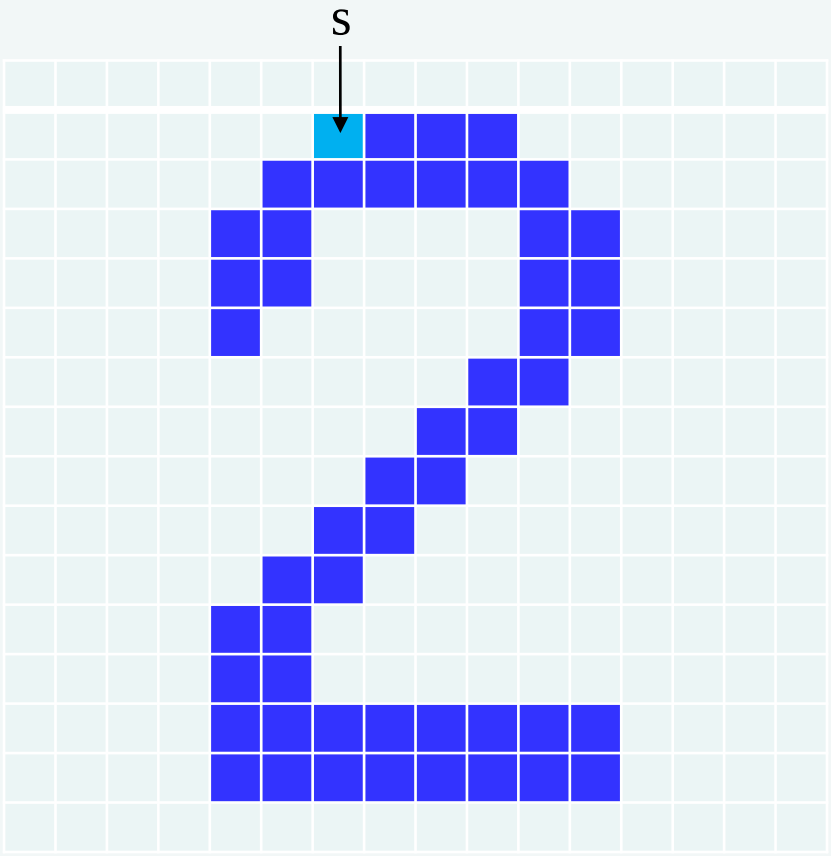
$[5\ 6\ 6\ 0\ 1\ 0\ 2\ 3\ 4]_{abs}$

$[1\ 0\ 2\ 1\ 7\ 2\ 1\ 1\ 1]_{dif}$

shape number: [7 2 1 1 1 1 0 2 1]

Reálný příklad (8-sousedů):




Řetězcový kód pro 8-sousedů:

$[5566121000066655555566600000002444432111111223344 4]_{abs}$ =
$[0103177000600700000100200000022000077700000101010 01]_{dif}$

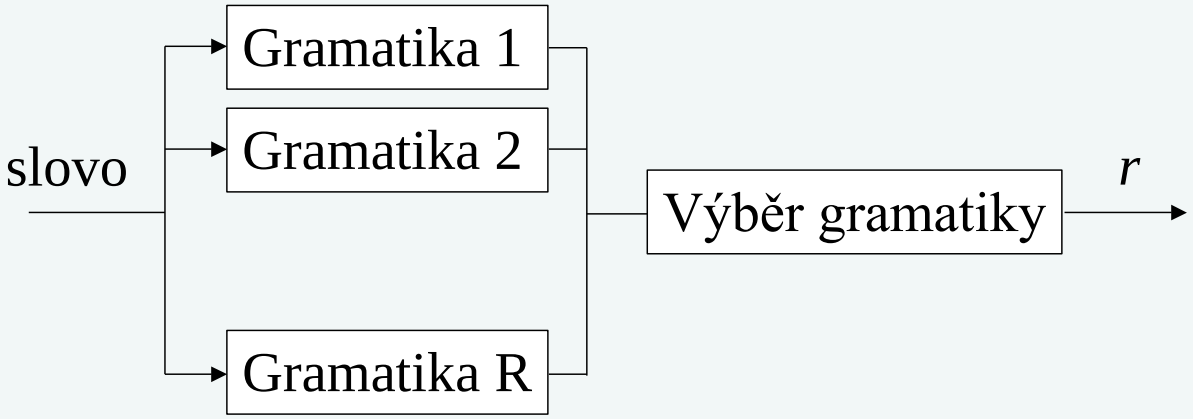
Číslo tvaru:
[77700000101010010103177000600700000100200000220000]

# Syntaktické rozpoznávání

Syntaktické rozpoznávání je založeno na podobnosti mezi strukturálními popisy obrazů a syntaxí jazyků

| | | |
|---|---|---|
| primitiva | – | terminální symboly |
| obrazy | – | slova |
| třídy | – | (regulární) jazyky |
| klasifikátory | – | gramatiky |

a spočívá v nalezení gramatiky, která akceptuje rozpoznávaný strukturální popis (slovo):



Číslo tvaru ze snímku 24 (tj. číslo [7 2 1 1 1 1 0 2 1]) pak může být akceptováno například těmito gramatikami (0, 1, 2, 7 jsou terminální symboly, OBJEKT, OA, …, OI jsou neterminální symboly a *e* je prázdný symbol) :

| | | |
|---|---|---|
| OBJEKT | ⇒ | 7 OA |
| OA | ⇒ | 2 OB |
| OB | ⇒ | 1 OC |
| OC | ⇒ | 1 OD |
| OD | ⇒ | 1 OE |
| OE | ⇒ | 1 OF |
| OF | ⇒ | 0 OG |
| OG | ⇒ | 2 OH |
| OH | ⇒ | 1 OI |
| OI | ⇒ | *e* |

Akceptuje jen a jen dané číslo tvaru

| | | |
|---|---|---|
| OBJEKT | ⇒ | 7 OA |
| OA | ⇒ | 2 OA |
| OA | ⇒ | 1 OB |
| OB | ⇒ | 1 OB |
| OB | ⇒ | 0 OC |
| OC | ⇒ | 2 OD |
| OD | ⇒ | 1 OE |
| OE | ⇒ | *e* |

Akceptuje dané číslo tvaru, ale navíc i čísla [7 2 2 1 1 0 2 1], atp.

## Inference gramatik

| | |
|---|---|
| Gramatika | $G = (V_T, V_N, P, S)$ |
| Množina terminálních symbolů | $V_T$ |
| Množina neterminálních symbolů | $V_N$ |
| Množina přepisovacích pravidel | $P$ |
| Počáteční symbol | $S \in V_N$ |
| Abeceda | $V = V_T \cup V_N$ |
| Množina všech slov nad abecedou V | $V^*$ |
| Jazyk | $L \subseteq V^*$ |
| Prázdný symbol | $e$ |
| | $V^+ = V^* - e$ |
| Přepisovací pravidlo $p \in P$ | $\alpha \rightarrow \beta$ |
| | $\alpha \in V^+, \beta \in V^*$ |

Jazyk generovaný gramatikou G $L(G) = \{x \mid x \in V^*, S \rightarrow x\}$

Trénovací množina $T = \{S^+, S^-\}$

$S^+ = \{x_i \mid x_i \in L(G)\}$

$S^- = \{x_j \mid x_j \notin L(G)\}$

Inferenční procedura:

Nechť $G^{(0)}$ je počáteční odhad gramatiky, $k = 0$.

1. Nastavte $i = 1$ a *modif* = false.
2. Vyberte z trénovací množiny slovo $x_i$ .
3. Jestliže $x_i \in S^+$ a gramatika $G^{(k)}$ negeneruje $x_i$ tak:
   - modifikujte $G^{(k)} \rightarrow G^{(k+1)}$ tak, aby gramatika $G^{(k+1)}$ generovala $x_i$ ,
   - *modif* = true,
   - $k = k+1$.
3. Jestliže $x_i \in S^-$ a gramatika $G^{(k)}$ generuje $x_i$ tak:
   - modifikujte $G^{(k)} \rightarrow G^{(k+1)}$ tak, aby gramatika $G^{(k+1)}$ negenerovala $x_i$ ,
   - *modif* = true,
   - $k = k+1$.
4. $i = i+1$.
5. Není-li trénovací množina $T$ prázdná, vraťte se na bod 2.
6. Jestliže *modif* = = true, tak se vraťte na bod 1.

Strukturální/syntaktický popis obsahuje z principu více informací o rozpoznávaném objektu/signálu než popis příznakový. V poslední době je však syntaktický přístup k rozpoznávání poněkud v pozadí, a to především z důvodu současných velkých úspěchů (konvolučních) neuronových sítí na tomto poli. Navíc je strukturální popis značně citlivý na šum, problémem někdy bývá i „falešná“ abeceda nonterminálů a inference (učení) gramatik je výrazně složitější, než učení příznakových klasifikátorů nebo/i neuronových sítí.

# Perceptron

Inspirací pro perceptron je základní prvek nervové soustavy živých organizmů, nervová buňka - neuron.

Každý biologický neuron se skládá z těla (somy), ke kterému je připojeno přibližně tisíc (v rozsahu$10^2$ až $10^5$) 2-3 mm dlouhých dendritů (vstupů) a jeden, 1 cm až několik desítek cm dlouhý axon (výstup).

Rozhraní mezi dendritem jednoho a axonem druhého neuronu se nazývá synapsí, jejíž váha představuje vliv jednoho na druhý neuron. Velikost signálu přenášená dendritem k tělu neuronu tedy závisí jak na aktivitě axonu, kterého se dendrit dotýká, tak na příslušné synaptické váze. Tato váha může být buď posilující/excitační nebo potlačující/inhibiční.

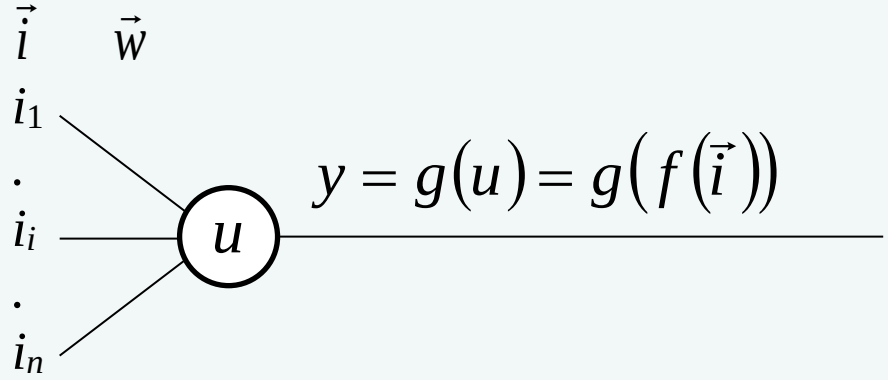
Lidský mozek má přibližně $10^{11}$ neuronů $\Rightarrow$ $10^{14}$ synapsí, které představují jeho paměťové schopnosti. Učení jako takové pak spočívá v nastavování synaptických vah jednotlivých neuronů!

Činnost neuronu:

Hodnoty signálů od jednotlivých dendritů se v těle neuronu sčítají. Přesáhne-li jejich celková hodnota jistý práh, neuron vyšle krátký impuls o velikosti asi 100 mV, který postupuje po jeho axonu rychlostí 0.1 až 10 m/s. Pak se neuron na krátkou dobu stane necitlivým. Pokud je po této době celková hodnota vstupního signálu stále větší než práh, impuls se opakuje. Po axonu se tak šíří pulsy o frekvenci 0.1/s až 100/s.

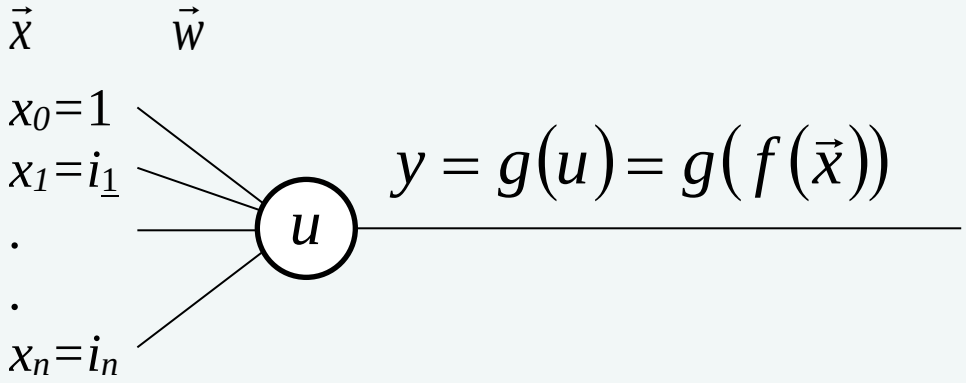
Pro umělý neuron by byla činnost s podobnou výstupní funkcí obtížně realizovatelná. Proto se používají jednodušší modely, jejichž výstupní signály jsou statické.

# Umělý neuron – obecný model




| | |
|---|---|
| $u$ | vnitřní potenciál |
| $f$ | bázová funkce |
| $g$ | aktivační funkce |

Perceptron je nejznámějším umělým neuronem. Pro zjednodušení vztahů pro výpočet odezvy a pro učení doplňuje v obecném modelu vstupní vektor o nultou složku, která má pevnou hodnotu 1 a jejíž váha se nastavuje na hodnotu záporného prahu



Perceptron používá lineární bázovou funkci *f*

$$u = f(\vec{x}) = \vec{w} \cdot \vec{x} = \sum_{i=0}^{n} w_i x_i, \qquad w_0 = -\theta, \quad x_0 = 1 \quad x_i = i_i$$

a nespojitou/skokovou aktivační funkci *g*

$$y = g(u) = sign(u) = \begin{cases} 1 & pro \quad u > 0 \\ -1 & pro \quad u < 0 \\ y^{old} & pro \quad u = 0 \end{cases}$$

Perceptron lze naučit klasifikovat lineárně separabilní číselné vektory do dvou tříd (podobně jako klasický lineární klasifikátor pro dichotonomii). Nechť

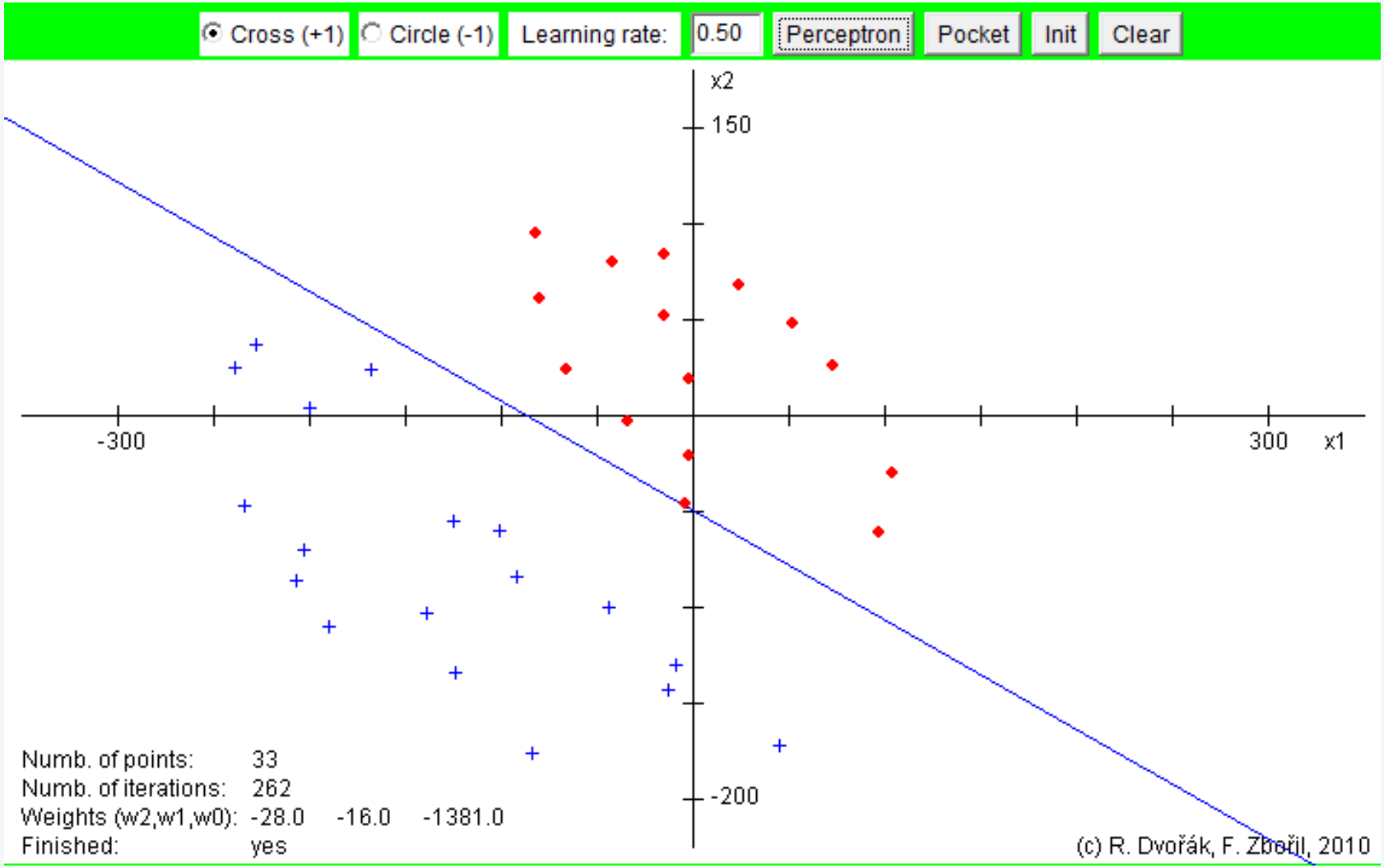
$$\vec{z}(k)=\begin{cases}+\vec{x}(k) & pro \quad d(k)=+1\\ -\vec{x}(k) & pro \quad d(k)=-1\end{cases} \qquad t.j. \quad \vec{z}(k)=d(k)\vec{x}(k)$$

kde $d(k) = \pm 1$ je požadovaná klasifikace (třída) a $k$ je krok učení.

Pak základní pravidlo učení perceptronu má tvar:

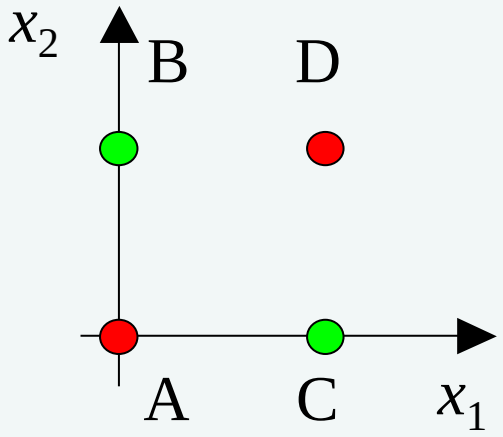
$$\vec{w}(0) = \ libovolné$$
$$\vec{w}(k)=\vec{w}(k-1)+2\mu\,\vec{z}(k) \qquad pro \quad \left(\vec{w}(k-1)\cdot\vec{z}(k)\right)\le 0$$
$$\vec{w}(k)=\vec{w}(k-1) \qquad jinak$$

kde $\mu$ je tzv. koeficient učení. Na jeho velikosti však prakticky nezáleží, a proto se obvykle volí $\mu = 0.5$.

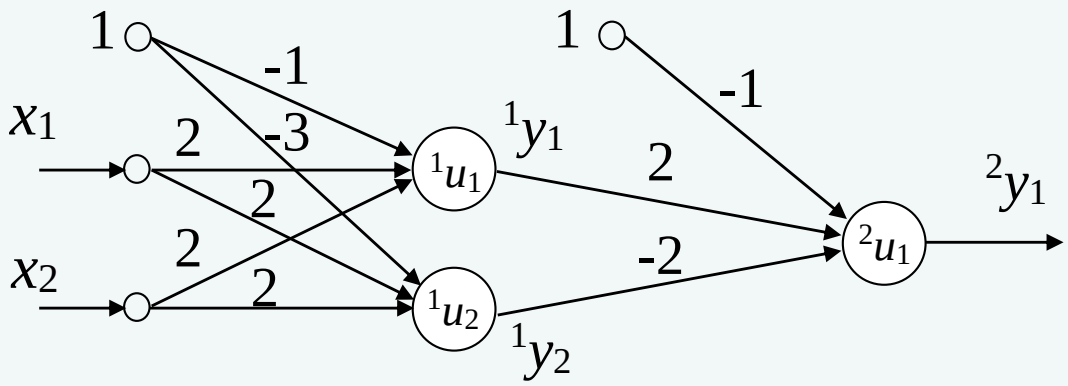

Cross (+1)
Circle (-1)
Learning rate:
0.50
Perceptron
Pocket
Init
Clear
x2
150
-300
300
x1
-200
Numb. of points: 33
Numb. of iterations: 262
Weights (w2,w1,w0): -28.0 -16.0 -1381.0
Finished: yes
(c) R. Dvořák, F. Zbořil, 2010

Lineárně neseparovatelné číselné vektory lze pak relativně snadno klasifikovat pomocí vícevrstvých neuronových sítí (teoreticky stačí dvě vrstvy).

Například vektory (koncové body) přestavující problém XOR ($\vec{x}(A) = (0,0),\ \vec{x}(B) = (0,1),\ \vec{x}(C) = (1,0),\ \vec{x}(D) = (1,1)$):



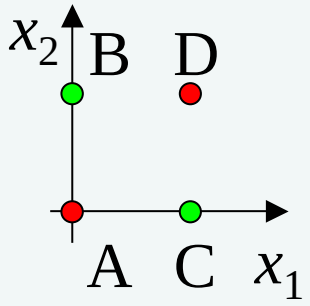

bude úspěšně klasifikovat do dvou tříd síť uvedená na následujícím snímku.

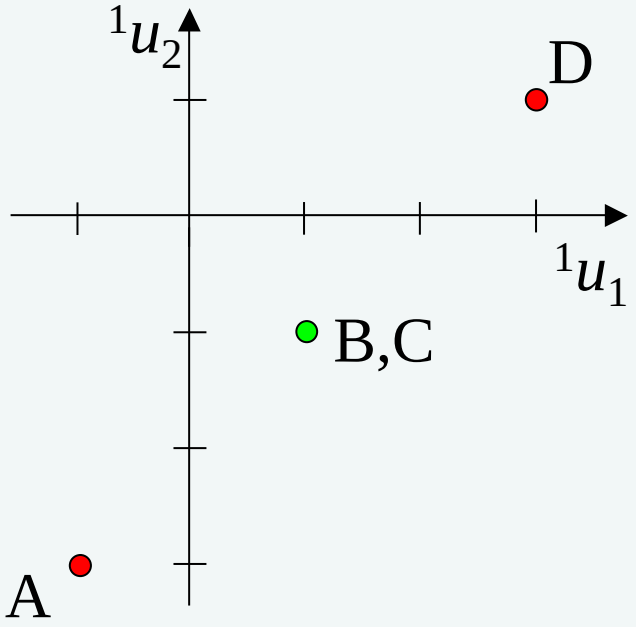


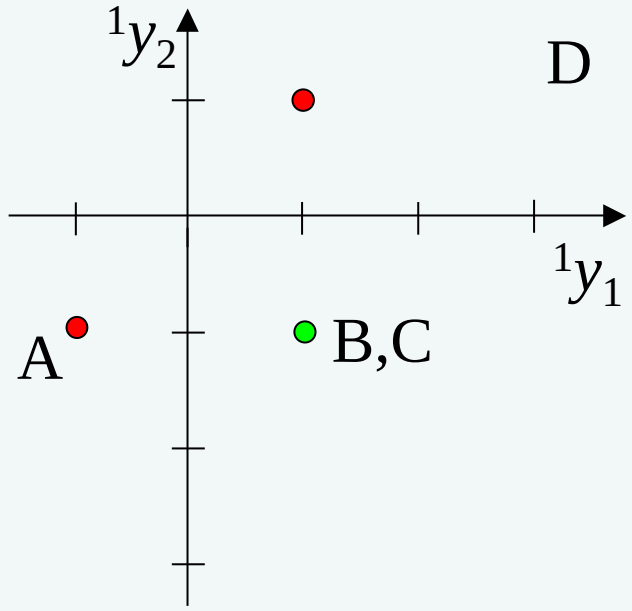

| bod | $x_1$ | $x_2$ | $^1u_1$ | $^1u_2$ | $^1y_1$ | $^1y_2$ | $^2u_1$ | $^2y_1$ |
|---|---|---|---|---|---|---|---|---|
| A | 0 | 0 | -1 | -3 | -1 | -1 | -1 | -1 |
| B | 0 | 1 | 1 | -1 | 1 | -1 | 3 | 1 |
| C | 1 | 0 | 1 | -1 | 1 | -1 | 3 | 1 |
| D | 1 | 1 | 3 | 1 | 1 | 1 | -1 | -1 |

Transformace původního dvourozměrného prostoru na jednodimenzionální (viz snímek 13, bod d)):

| bod | $x_1$ | $x_2$ | ${}^1u_1$ | ${}^1u_2$ | ${}^1y_1$ | ${}^1y_2$ | ${}^2u_1$ | ${}^2y_1$ |
|---|---|---|---|---|---|---|---|---|
| A | 0 | 0 | -1 | -3 | -1 | -1 | -1 | -1 |
| B | 0 | 1 | 1 | -1 | 1 | -1 | 3 | 1 |
| C | 1 | 0 | 1 | -1 | 1 | -1 | 3 | 1 |
| D | 1 | 1 | 3 | 1 | 1 | 1 | -1 | -1 |



Transformace původního dvourozměrného prostoru na jednodimenzionální (pokračování):

| bod | $x_1$ | $x_2$ | ${}^1u_1$ | ${}^1u_2$ | ${}^1y_1$ | ${}^1y_2$ | ${}^2u_1$ | ${}^2y_1$ |
|---|---|---|---|---|---|---|---|---|
| A | 0 | 0 | -1 | -3 | -1 | -1 | -1 | -1 |
| B | 0 | 1 | 1 | -1 | 1 | -1 | 3 | 1 |
| C | 1 | 0 | 1 | -1 | 1 | -1 | 3 | 1 |
| D | 1 | 1 | 3 | 1 | 1 | 1 | -1 | -1 |