Obsah

- Kompetičné učenie
- Kohonenove samo-organizujúce sa mapy

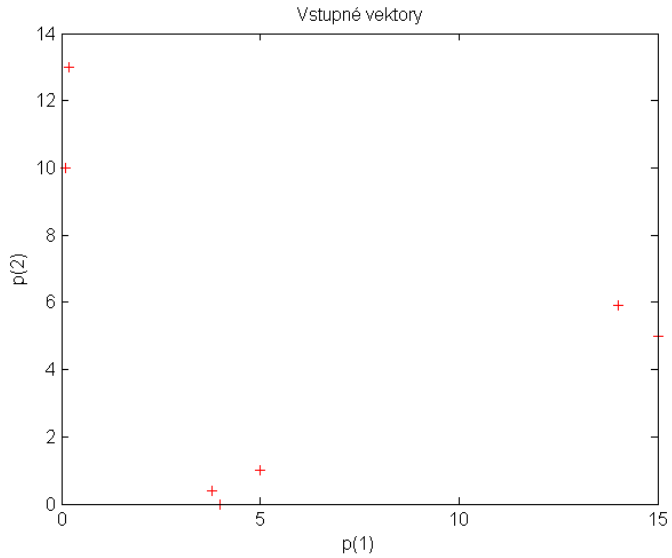
Kompetičné učenie

Máme množinu dvojrozmerných vektorov `p` a chceme ju rozdeliť na skupiny podobných vektorov - nájsť zhluky.

```
p=[  0.1  4  5   0.2  15  3.8  14;
    10    0  1  13     5  0.4   5.9];
```

`p` nakreslíme

```
plot(p(1,:),p(2,:),'+r');
title('Vstupné vektory');
xlabel('p(1)');
ylabel('p(2)');
```



Neuróny kompetitívnej vrstvy sa majú naučiť reprezentovať regióny vo vstupnom priestore vzorov. Vytvoríme kompetičnú sieť s 2 vstupnými a 3 neurónmi v kompetičnej vrstve:

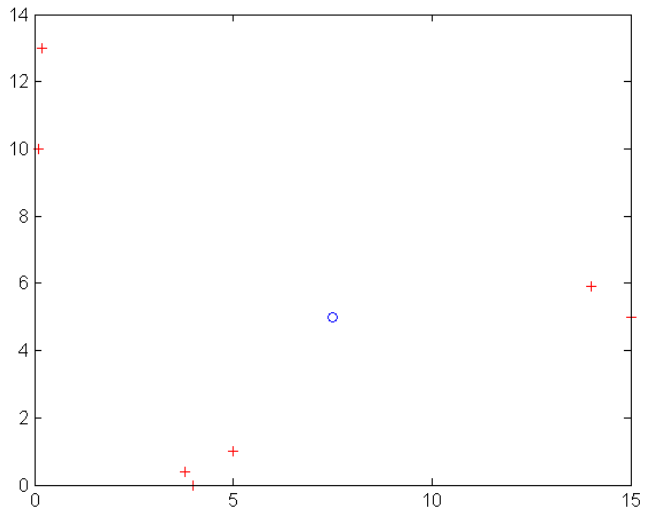
```
net=newc([0 15; 0 10],3);
```

Prvý parameter funkcie `newc` udáva rozsahy hodnôt jednotlivých zložiek vstupných vektorov, druhý argument udáva počet neurónov v kompetičnej

vrstve. Po inicializácii majú neuróny váhy v strede rozsahu jednotlivých zložiek.

```
w = net.IW{1}
plot(p(1,:),p(2,:),'+r');
hold on;
circles = plot(w(:,1),w(:,2),'ob');
hold off;
```

```
w =
    7.5000    5.0000
    7.5000    5.0000
    7.5000    5.0000
```



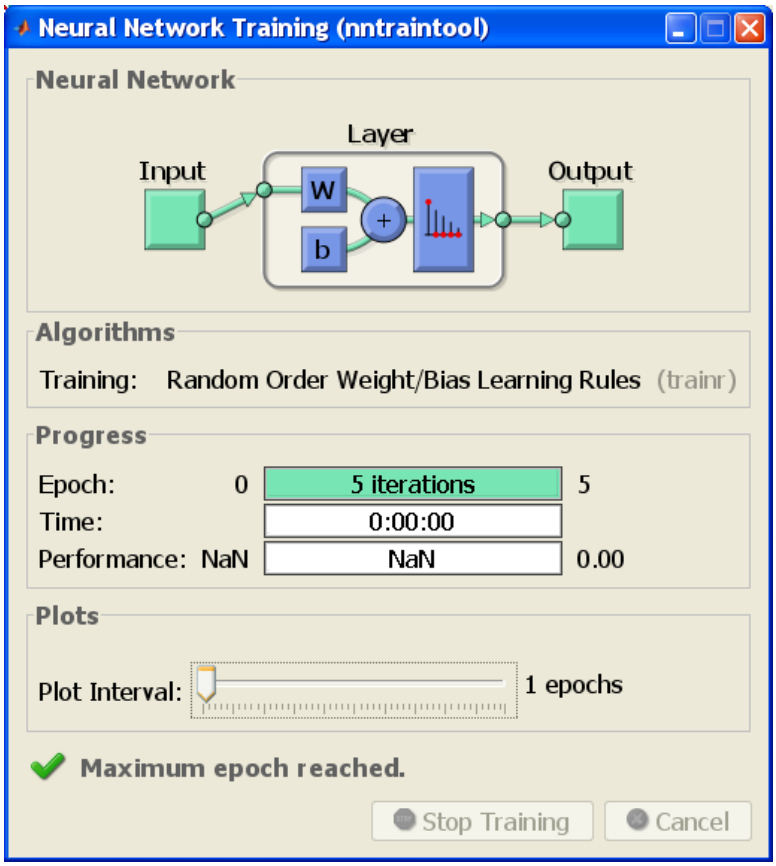
Nastavíme počet iterácií cez vstupné vzory

```
net.trainParam.epochs=5;
```

Spustíme trénovanie siete.

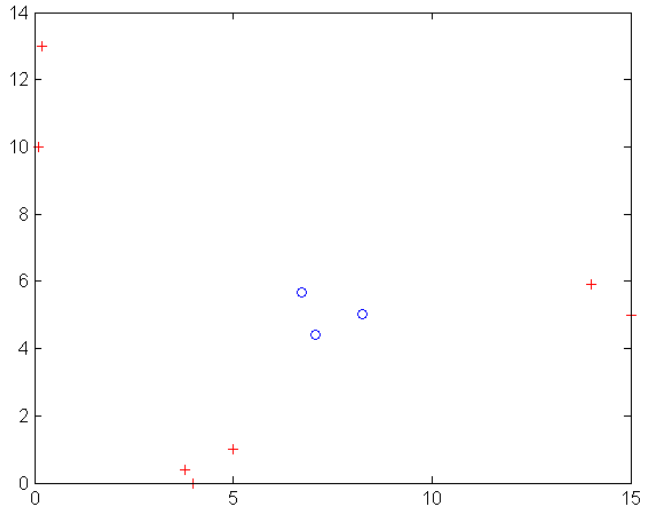
```
net = train(net,p);
```

V priebehu trénovania sa nám zobrazí okno



Po krátkom učení majú neuróny váhy posunuté k jednotlivým zhlukom, ale počet epoch bol príliš malý na naučenie.

```
w = net.IW{1};
plot(p(1,:),p(2,:),'+r');
hold on;
circles = plot(w(:,1),w(:,2),'ob');
hold off
```



Nastavíme vyšší počet iterácií cez vstupné vzory

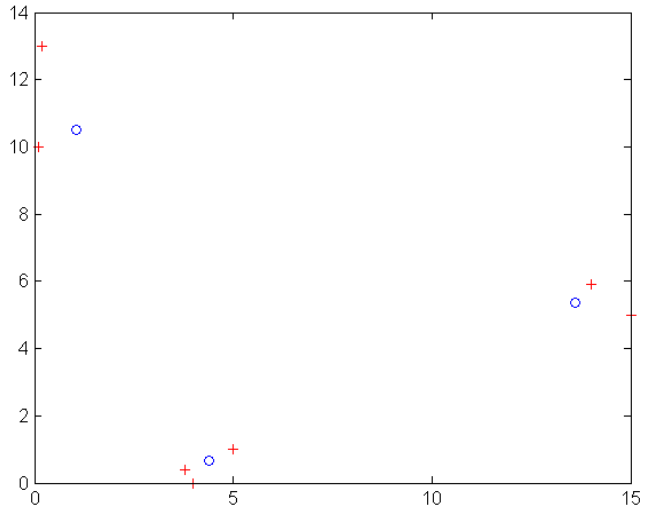
```
net.trainParam.epochs=100;
```

Spustíme znova trénovanie siete.

```
net = train(net,p);
```

Teraz už majú neuróny váhy posunuté k jednotlivým zhlukom.

```
w = net.IW{1};
plot(p(1,:),p(2,:),'+r');
hold on;
circles = plot(w(:,1),w(:,2),'ob');
hold off
```



Natrénovanú sieť necháme spočítať výstupy pre vzory z `p`:

```
a = sim(net,p)
```

```
a =
   (1,1)        1
   (3,2)        1
   (3,3)        1
   (1,4)        1
   (2,5)        1
   (3,6)        1
   (2,7)        1
```

Výsledkom je riedka matica – v každom stĺpci je len jedna hodnota 1. Nasledujúca funkcia vráti riadkové indexy prvkov 1 z matice `a`.

```
ac=vec2ind(a)
```

```
ac =
     1     3     3     1     2     3     2
```

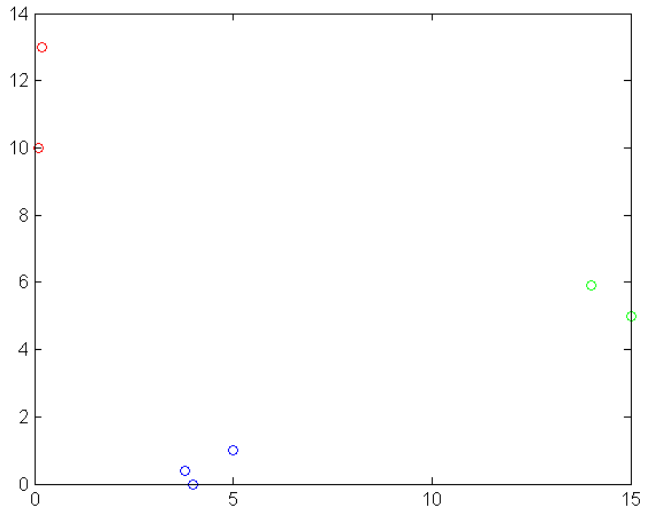
To sú vlastne čísla zhlukov, do ktorých bol zaradený príslušný vstupný vektor. Zhluky môžeme zobraziť farebne rozlíšené. Najprv si uložíme indexy vektorov jednotlivých zhlukov a potom ich zobrazíme rôznymi farbami. Najprv si zistíme indexy vektorov z jednotlivých zhlukov:

```
ind1=find(ac==1)
ind2=find(ac==2)
ind3=find(ac==3)
```

```
ind1 =
     1     4
ind2 =
     5     7
ind3 =
     2     3     6
```

Nasledujúci príkaz `plot` musí byť celý na jedinom riadku!

```
hold off
plot(p(1,ind1),p(2,ind1),'or',p(1,ind2),p(2,ind2),'og',p(1,ind3),p(2,ind3),'ob')
```



Kohonenove samo-organizujúce sa mapy

Tento model je vlastne rozšírením kompetičnej siete. Neuróny majú naviac definované topológiu. Vytvorenie Kohonenovej mapy: `net = newsom(PR,[D1,D2,...],TFCN,DFCN,OLR,OSTEPS,TLR,TND)`

`PR`	matica `R x 2` miním a maxím jednotlivých zložiek `R` vstupných vektorov,
`D` i	rozmery siete; implicitne = [5 8],
`TFCN`	topológia; implicitne ='hextop',
`DFCN`	funkcia vzdialenosti, implicitne = `'linkdist'` ,

`OLR` učiaca konštanta pre fázu usporiadavania, implicitne = 0.9,
`OSTEPS` počet krokov fázy usporiadavania, implicitne = 1000,
`TLR` učiaca konštanta fázy ladenia, implicitne = 0.02,
`TND` veľkosť okolia pre fázu ladenia, implicitne = 1.

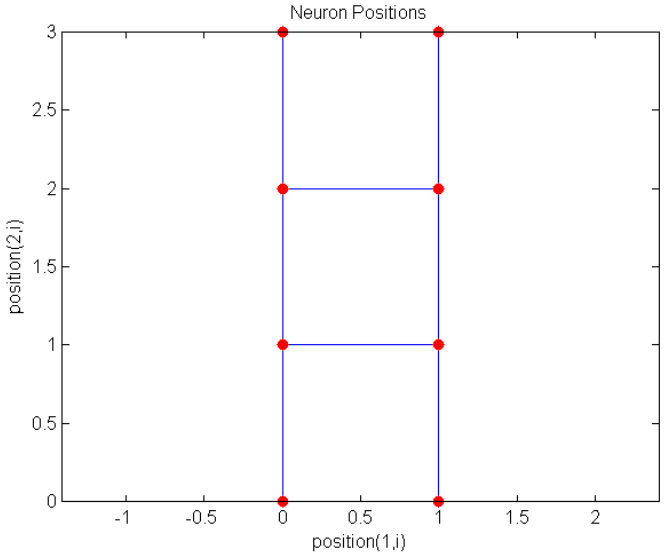
Možné topológie sú `'hextop'`, `'gridtop'` a `'randtop'`. Funkcia vzdialenosti môže byť `'linkdist'`, `'dist'`, `'boxdist'` alebo `'mandist'`, ktoré zodpovedajú počtu prejdených hrán, Euklidovskej vzdialenosti, štvorcovému okoliu a Manhattanskej vzdialenosti.

Odkrokujte si demo program `Demos` > `Self-organizing networks` > `two-dimensional organizing map`. Možné topológie si môžete zobraziť pomocou nasledujúcich funkcií.

- topológia pravouhlej mriežky:

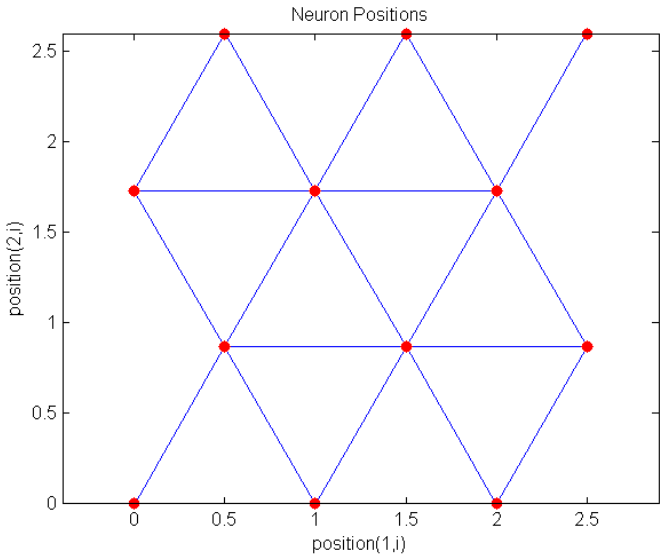
```
pos=gridtop(2,4)
plotsom(pos)
```

```
pos =
     0     1     0     1     0     1     0     1
     0     0     1     1     2     2     3     3
```



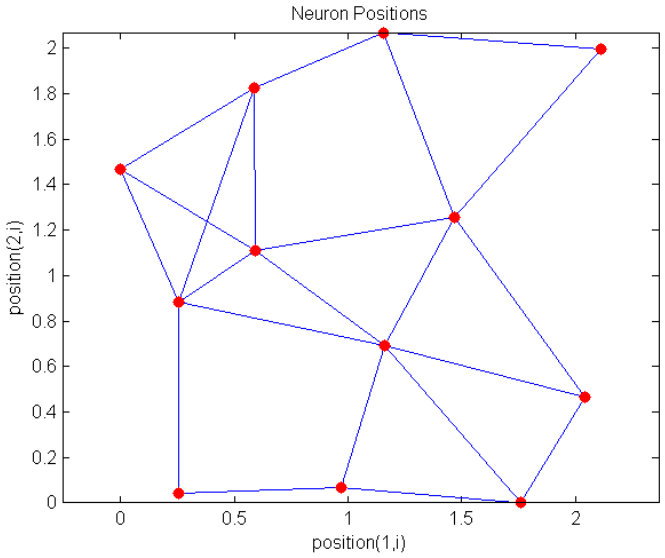
- topológia šesťuholníkovej mriežky:

```
pos=hextop(3,4);
plotsom(pos)
```



- topológia náhodného grafu:

```
pos=randtop(3,4);
plotsom(pos)
```



Učenie v MATLABe má dve fázy. V prvej sa neuróny majú usporiadať podľa topológie a v druhej majú spresniť aproximáciu vstupných vzorov. Typické volanie funkcie `newsom` však vynecháva väčšinu parametrov:

```
net=newsom(minmax(p),[3, 4],'gridtop');
```

Úloha: Natrénujte Kohonenovu mapu s pravouhlou topológiou pre množinu vektorov, ktoré vracia funkcia

```
http://ksvi.mff.cuni.cz/~mraz/datamining/dataset2.m
```

Vyskúšajte rôzne počty neurónov a rôzne topológie siete. Pri učení siete zobrazujte neuróny siete i vstupné vektory vždy po malom počte epoch.