Neuro-fuzzy rendszerek alapjai, ANFIS

<u>Előadás vázlat</u>

Összeállította: Harmati István Ph.D., egyetemi docens

Felhasznált irodalom:

Dr. Lantos Béla: Fuzzy systems and genetic algorithms, 2001, Műegyetemi kiadó, Budapest

5. ADAPTÍV NEURO-FUZZY RENDSZEREK

Sok mérnöki probléma vezethető vissza függvény approximációra. Az előadás keretében az adaptív neuro-fuzzy technikákon alapuló módszereket vizsgáljuk:

- Függvény approximáció többrétegű neurális hálózattal.
- Adaptív hálózatok
- ANFIS (Adaptive Neuro-Fuzzy Inference System)
- Nemlineáris rendszerdinamika identifikációja ANFIS-al

5.1. Függvényapproximáció többrétegű neurális hálózattal

Elemi neuron

Ismeretlen: $y = f(x)$

Alkalmazzuk: FFNN (Feedforward Neural Network) (MIMO eset)

Statikus elemi neuron:

$$\hat{y} = \sigma\left(\sum_{i=1}^{n} w_i x_i + t\right)$$

w_i súlyozó faktor (weight factor)

t offszet (threshold)- konstans 1 bemenetet rendelve hozzá, súlynak fogható fel. (Ezt csinaljuk)

$\sigma(\cdot)$ a neuron átmeneti függvénye (transition függvény)

A neuron tanítása ≡ A súlyok megválasztása

Példák átmeneti függvényekre

Függvények és deriváltjaik:

$$\sigma(x) = \frac{1}{1+\exp(-cx)} \in [0,1]$$

logsig

$$\sigma'(x) = \frac{c\exp(-cx)}{[1+\exp(-cx)]^2} = c\sigma(x)[1-\sigma(x)]$$

$$\sigma(x) = \frac{2}{1+\exp(-cx)} - 1 \in [-1,1]$$

bipoláris

$$\sigma'(x) = \frac{2c\exp(-cx)}{[1+\exp(-cx)]^2} = \frac{c}{2}[1+\sigma(x)]\cdot[1-\sigma(x)]$$

$$\sigma(x) = \tanh(cx) = \frac{\exp(cx)-\exp(-cx)}{\exp(cx)+\exp(-cx)} \in [-1,1]$$

tansig

$$\sigma'(x) = c[1-\tanh^2(cx)] = c[1+\sigma(x)]\cdot[1-\sigma(x)]$$

$$\sigma(x) = cx \in [-\infty,\infty]$$

purelin

$$\sigma'(x) = c$$

Egyrétegű neurális hálózat (single layer neural network)

Legyen a bemenetek száma: n_i

a kimenetek száma: n_o

A kimenetek:

$$\hat{y}_p = \sigma_p \left(\sum_{j=1}^{n_i} w_{pj} \, x_j \right), \quad p = 1,2,\ldots,n_o$$

vagy ugyanez gyakoribb jelölési konvenció alapján:

$$\hat{y}_p^* = \sum_{j=1}^{n_i} w_{pj} \, x_j, \quad \hat{y}_p = \sigma_p(\hat{y}_p^*), \quad p = 1,2,\ldots,n_o$$

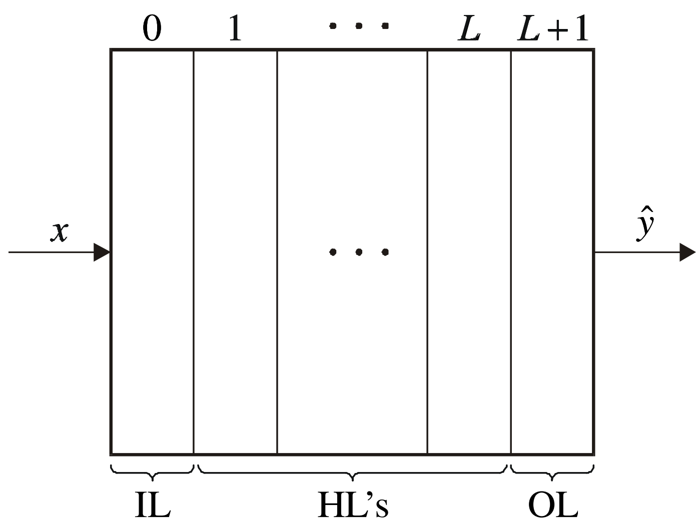
$\hat{y}_p^*$ eredő súlyozott bemenet (net input)

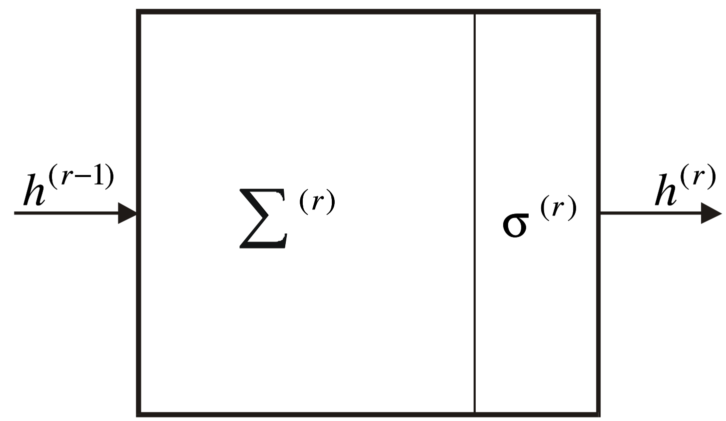
Az átmeneti függvény tipukusan azonos minden neuronra: $\sigma_p(\cdot) := \sigma(\cdot)$

Többrétegű visszacsatolás nélküli neurális hálózat struktúrája

Egy darab bemeneti rétegből (IL - Input Layer), L darab rejtett rétegből (HL – Hidden Layer) és szintén egy kimeneti rétegből (OL – Output Layer) áll.

A 0. réteg a bemeneti réteg, az $L+1$. réteg a kimeneti réteg:





Legyen

$h^{(r-1)}$ az (r)-dik réteg bemenő vektora

$h^{(r)}$ az (r)-dik réteg kimenő vektora

$n_o^{(r)}$ az (r)-dik réteg neuronjainak száma

Akkor

Az eredő súlyozott bemenet és a réteg kimenete:

$$h_p^{*(r)} = \sum_{j=1}^{n_o^{(r-1)}} w_{pj}^{(r)} h_j^{(r-1)}, \qquad h_p^{(r)} = \sigma_p^{(r)}(h_p^{*(r)})$$

Az egyszerűsített jelölésben:

$$\Sigma^{(r)} : h_p^{*(r)} = \sum_j w_{pj}^{(r)} h_j^{(r-1)},\ p = 1,2,\ldots,n_o^{(r)}$$

$$\sigma^{(r)} : \sigma_p^{(r)} = \sigma_p^{(r)}(h_p^{*(r)}), \quad p = 1,2,\ldots,n_o^{(r)}$$

$W^{(r)}$ az (r)-dik réteg súlyainak mátrixa

$h^{(0)} = x$ a neurális háló bemenete

$h^{(L+1)} = \hat{y}$ a neurális háló kimenete

Többrétegű visszacsatolás nélküli neurális hálózat működési vázlata

A neurális hálózat bemenetére adandó jel:

$$x = (x_1, x_2, \dots, x_{n_i})^T$$

A jel *előreterjedése* a rétegekben:

$$x = h^{(0)} \xrightarrow{W^{(1)},\sigma^{(1)}} h^{*(1)}, h^{(1)} \cdots\cdots h^{*(L)}, h^{(L)} \xrightarrow{W^{(L+1)},\sigma^{(L+1)}} h^{*(L+1)}, h^{(L+1)} = \hat{y} \qquad \text{(FP)}$$

A neurális hálózat kimenetén megjelenő jel:

$$\hat{y} = (\hat{y}_1, \hat{y}_2, \dots, \hat{y}_{n_o})^T$$

Cél: Egy előírt viselkedés (függvény) approximációja a tanítási minták alapján.

Tanítási minták: $\{x(k), y(k)\}_{k=1}^{N}$ A neurális hálózat válasza: $\{x(k), \hat{y}(k)\}_{k=1}^{N}$

Feladat a cél eléréséhez: Egy hibakritérium minimaizálása a $w_{pj}^{(r)}$ súlyok függvényében.

Mód: A hibakritérium parciális deriváltjainak meghatározása súlyok szerint
$\rightarrow$ gradiens, konjugált gradiens vagy más gradiens technikákkal optimalizálás.

Többrétegű visszacsatolás nélküli neurális hálózat hibája

Legyen $\Phi : R^1 \to [0,\infty)$ egy konvex, pozitív definit és majdnem mindenütt differenciálható függvény, pl: $\Phi(e) = \frac{1}{2}e^2$. Akkor

A neurális hálózat hibája $x(k)$ bemenetre: $E(k) = \sum_{i=1}^{n_o} \Phi(y_i(k) - \hat{y}_i(k))$

A neurális hálózat hibája az összes bemenet esetén: $E_{\text{total}} = \sum_{k=1}^{N} E(k)$

Tudjuk, hogy: $\dfrac{\partial E_{\text{total}}}{\partial w_{pj}^{(r)}} = \sum_{k=1}^{N} \dfrac{\partial E(k)}{\partial w_{pj}^{(r)}}$

A kulcskérdés: $\dfrac{\partial E(k)}{\partial w_{pj}^{(r)}}$ meghatározása

A k index kiírását az egyszerűség kedvéért elhagyva:	A differenciálás láncszabálya szerint:
$w_{pj}^{(r)} \to \quad h_p^{*(r)} = \sum_j w_{pj}^{(r)} h_j^{(r-1)}$ $h_p^{(r)} = \sigma_p^{(r)}(h_p^{*(r)}) \qquad \Rightarrow$ $h_l^{*(r+1)} = \sum_p w_{lp}^{(r+1)} h_p^{(r)}.$	$\dfrac{\partial E}{\partial w_{pj}^{(r)}} = \dfrac{\partial E}{\partial h_p^{*(r)}} \cdot \dfrac{\partial h_p^{*(r)}}{\partial w_{pj}^{(r)}} = \dfrac{\partial E}{\partial h_p^{*(r)}} h_j^{(r-1)}$ $\underbrace{\dfrac{\partial E}{\partial h_p^{*(r)}}}_{\varepsilon_p^{(r)}} = \sum_l \underbrace{\dfrac{\partial E}{\partial h_l^{*(r+1)}}}_{\varepsilon_l^{(r+1)}} \cdot \underbrace{\dfrac{\partial h_l^{*(r+1)}}{\partial h_p^{(r)}}}_{w_{lp}^{(r+1)}} \cdot \underbrace{\dfrac{\partial h_p^{(r)}}{\partial h_p^{*(r)}}}_{\sigma_p'^{(r)}(h_p^{*(r)})},$ (EBP) $\varepsilon_p^{(r)} = \sigma_p'^{(r)}(h_p^{*(r)}) \sum_l \varepsilon_l^{(r+1)} w_{lp}^{(r+1)}.$

Többrétegű visszacsatolás nélküli neurális hálózat tanítása

Algoritmus [FFNN tanítása error backpropagation algoritmussal]

1. A bemeneti minta ráhelyezése a neurális háló bemenetére és FP kiértékelése az $\hat{y}$ meghatározásához:

$$x = h^{(0)} \xrightarrow{W^{(1)},\sigma^{(1)}} h^{*(1)}, h^{(1)} \cdots\cdots h^{*(L)}, h^{(L)} \xrightarrow{W^{(L+1)},\sigma^{(L+1)}} h^{*(L+1)}, h^{(L+1)} = \hat{y}$$

2. Hibavisszaterjesztés (BP – error backprogataion):

2.1. A hiba meghatározása: $e = y - \hat{y}$

2.2. A súlyok meghatározása a súlyok gradiensének segítségével, a hibaviszaterjesztő (BP - algoritmussal)

Indulás: $r = L + 1$

$$\varepsilon_p^{(L+1)} = \frac{\partial E}{\partial h_p^{*(L+1)}} = -\Phi'(y_p - \hat{y}_p)\sigma_p'^{(L+1)}(h_p^{*(L+1)}),$$

$$\frac{\partial E}{\partial w_{pj}^{(L+1)}} = \varepsilon_p^{(L+1)} h_j^{(L)}.$$

Hátratartó rekurzió: $r = L, L-1, \ldots, 1$

$$\varepsilon_p^{(r)} = \sigma_p'^{(r)}(h_p^{*(r)})\sum_l \varepsilon_l^{(r+1)} w_{lp}^{(r+1)},$$

$$\frac{\partial E}{\partial w_{pj}^{(r)}} = \varepsilon_p^{(r)} h_j^{(r-1)}.$$

Hangolás történhet:

- Szekvenciálisan, azaz on-line minden egyes bemutatott minta után
- Batch módban, azaz off-line, az összes minta bemutatása után
  (ez a jobb E_{total} használata miatt + hatékony [pl konjugált] gradiens technikák is használhatók)

5.2. Adaptív hálózatok (AN)

5.2.1. Az adaptív hálózat felépítése

A neurális hálózatok és a fuzzy rendszerek álalánosításai bizonyos értelemben.

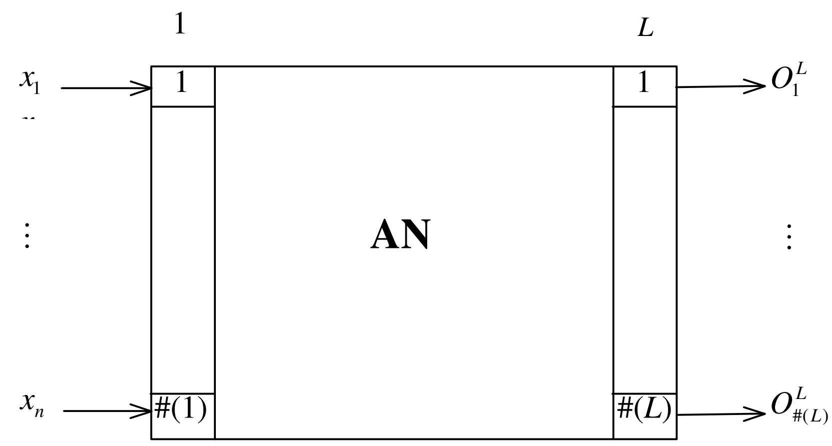
Jellemzők:

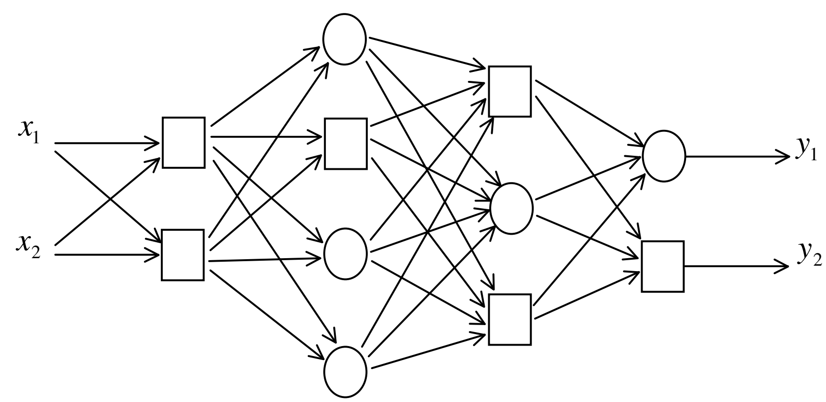
- Csomópontokból áll, amelyek irányított élekkel vannak összekötve.
- A jelek az élek mentén terjednek
- Az AN (L számú) rétegekbe van szervezve
- Általában nem tartalmaz visszacsatolást
- Bemenet: x_i, Kimenet: O_i^L
- A csomópontok száma a k -ik rétegben: $\#(k)$
- **Hasonló FFNN-hez, de nincsenek súlyok az éleken!**
- A csomópontok realizált függvénye (és egyúttal neve is):

  $O_i^k = O_i^k(O_1^{k-1}, \ldots, O_{\#(k-1)}^{k-1}, a, b, c, \ldots)$, ahol

  $O_1^{k-1}, \ldots, O_{\#(k-1)}^{k-1}$ a bemenő jelek,

  $a, b, c, \ldots$ a függvény paraméterek.
- Az AN paramétereinek halmaza: S
- Adaptív csomópont jelölése: négyzet.
- Fix csomópont jelölése: kör.





5.2.3. Az adaptív hálózatok hibája

A bemeneti/kimeneti vektorok célhalmaza: $\{x(k), y(k)\}_{k=1}^{N}$

Az AN kimeneti vektora: $\hat{y} = (O_1^L, \ldots, O_{\#(L)}^L)^T$

Cél: Minimalizálni az $E_{\text{total}} = \sum_{k=1}^{N} E(k)$ hibafüggvényt, ahol

$$E(k) = \frac{1}{2}\sum_{i=1}^{\#(L)}(y_i(k) - \hat{y}_i(k))^2 = \frac{1}{2}\sum_{i=1}^{\#(L)}\left(y_i(k) - O_i^L(k)\right)^2$$

Az AN paramétereinek optimalizálása történhet pl. valamilyen gradiens technikával. Ha $\alpha \in S$ az AN egy paramétere, akkor a hangolásnál:

$$\boxed{\Delta\alpha = -\eta\frac{\partial E_{\text{total}}}{\partial\alpha} = -\frac{\lambda}{\sqrt{\sum_\alpha \left|\frac{\partial E_{\text{total}}}{\partial\alpha}\right|^2}}\frac{\partial E_{\text{total}}}{\partial\alpha}}$$

Ahol

λ a lépésköz

$$\frac{\partial E_{\text{total}}}{\partial\alpha} = \sum_{k=1}^{N}\frac{\partial E(k)}{\partial\alpha}$$

$$\frac{\partial E(k)}{\partial\alpha} = \sum_{O\in O_\alpha^*}\frac{\partial E(k)}{\partial O}\cdot\frac{\partial O}{\partial\alpha}$$

(O_α^* az α-tól függő csomópontok)

$O(\cdot)$ analitikusan ismert $\rightarrow$ $\partial O / \partial\alpha$ számítható.$\Rightarrow$

$\partial E(k)/\partial O$ számítása hátratartó rekurzióval:

$$\frac{\partial E}{\partial O_i^L} = -(y_i - O_i^L) = -(y_i - \hat{y}_i) \quad \text{(utolsó réteg)}$$

$$\frac{\partial E(k)}{\partial O_i^l} = \sum_{m=1}^{\#(l+1)}\frac{\partial E(k)}{\partial O_m^{l+1}}\cdot\frac{\partial O_m^{l+1}}{\partial O_i^l}$$

(mivel O_i^l az $O_m^{l+1} = O_m^{l+1}(O_1^l, \ldots, O_{\#(l)}^l, \ldots)$ bemenete)

5.3. ANFIS

(Adaptive Neuro-Fuzzy Inference System vagy

Adaptíve Network Based Fuzzy Inference System)

Egy kimenetű Sugeno fuzzy rendszerek könnyen konvertálhatók adaptív hálózatokká (AN).

Elsőrendű Sugeno fuzzy rendszer:

$$R_i : \textit{if} \;\; x_1 \;\; \textit{is} \;\; A_1^i \;\; \textit{and} \;\; \ldots \textit{and} \;\; x_n \;\; \textit{is} \;\; A_n^i \;\; \textit{then} \;\; y \;\; \textit{is} \;\; c_{i1}x_1 + \cdots c_{in}x_n + c_{i0},$$
$$i = 1, \ldots, m.$$

(Speciálisan nulladrendű Sugeno fuzzy rendszer esetén: $c_{i1} = \cdots = c_{in} = 0$)

Tétel [Li Xin Wang] Bármely (kompakt halmaz feletti) folytonos függvény tetszőleges $\varepsilon > 0$ előírt pontossággal közelíthető nulladrendű, Gauss tagsági függvényeket használó Sugeno fuzzy rendszerrel.

Megjegyzések:

- Csökkenő ε a relációk növelését igényli.
- Az elsőrendű szabályozás általánosabb, ezért a továbbiakban azt használjuk.

A tagsági függvények alakja:

$$\mu_{A_j^i}(x_j) := \exp\left[-\frac{1}{2}\left(\frac{x_j - \bar{x}_j^i}{\sigma_j^i}\right)^2\right]$$

A $x=(x_1,\dots,x_n)^T$ **bemeneti adathoz tartozó Sugeno közelítés lépései** (szorzás, mint T-norma mellett):

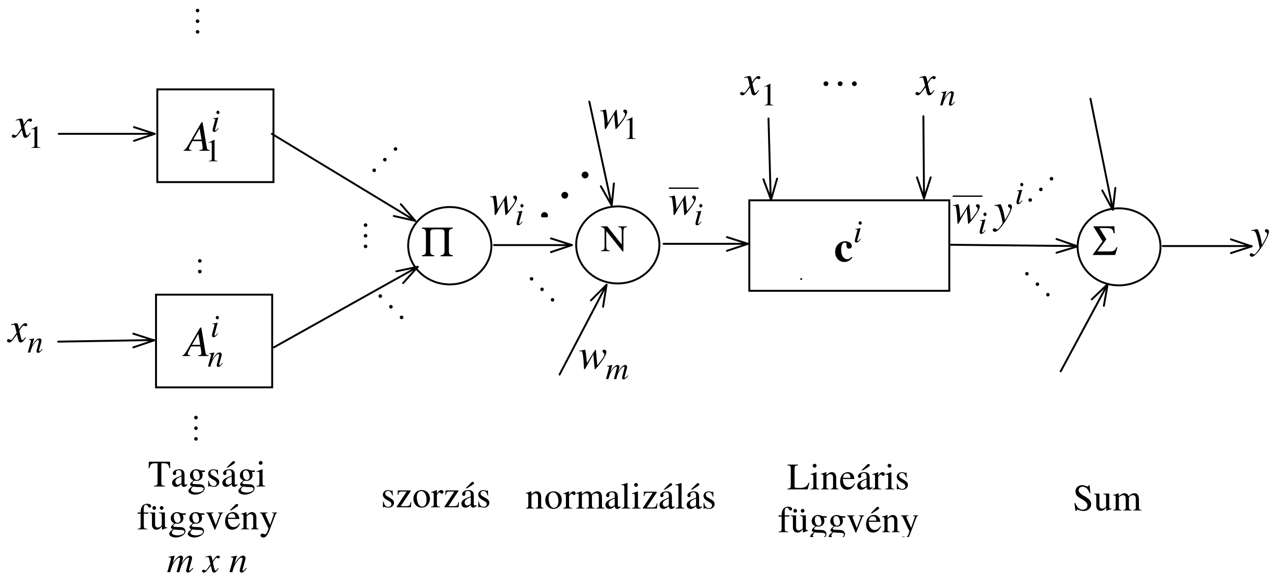
$$w_i := \mu_{A_1^i}(x_1)\cdot\dots\cdot\mu_{A_n^i}(x_n)$$

$$\hat{y}^i(x) := c_{i1}x_1 + \dots + c_{in}x_n + c_{i0}$$

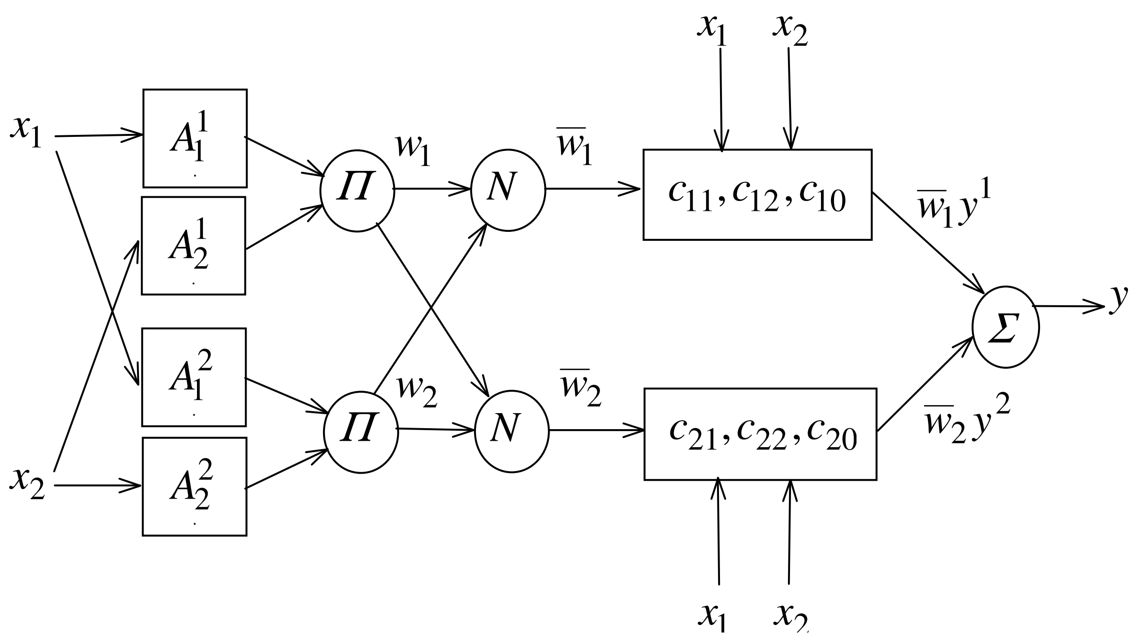
$$\hat{y} = \hat{f}(x) = \frac{\sum_{i=1}^{m} w_i \hat{y}^i(x)}{\sum_{i=1}^{m} w_i} = \sum_{i=1}^{m} \overline{w}_i \hat{y}^i(x)$$

Az elsőrendű Sugeno-rendszer paraméterei: $\bar{x}_j^i, \sigma_j^i, c_{i1},\dots,c_{in}, c_{i0}, \quad j=1,\dots,n; \quad i=1,\dots,m$

A Sugeno-rendszer i-dik relációjával ekvivalens AN részlet:



2 bemenetű, 1 kimenetű, 2 relációval rendelkező (Sugeno fuzzy) rendszer AN alakban:



ANFIS tanítása:

Az előírt (közelítendő) adat: (x, y)

Az elsőrendű Sugeno-rendszer kimenete a közelítéshez: $\hat{y}(x)$ AN totális kimenete, $\hat{y}_i(x)$ az i-edik relációé

Gradiens vektor számítása:

$$E(k) = \frac{1}{2}[y(k) - \hat{y}(x(k))]^2$$

$$E_{\text{total}} = \sum_k E(k)$$

$$\frac{\partial E(k)}{\partial p} = -[y(k) - \hat{y}(x(k)]\frac{\partial \hat{y}(x(k))}{\partial p}$$

$$\frac{\partial E_{\text{total}}}{\partial p} = \sum_k \frac{\partial E(k)}{\partial p}$$

Az általános p paraméterbe behelyettesítve a konkrét paramétereket, a parciális deriváltak:

$$\frac{\partial \hat{y}(x)}{\partial c_{ij}} = \begin{cases} x_j \overline{w}_i, & if \quad j = 1, \ldots, n \\ \overline{w}_i, & if \quad j = 0, \end{cases}$$

$$\frac{\partial \hat{y}(x)}{\partial \overline{x}_j^i} = [\hat{y}^i(x) - \hat{y}(x)]\,\overline{w}_i \frac{x_j - \overline{x}_j^i}{(\sigma_j^i)^2}$$

$$\frac{\partial \hat{y}(x)}{\partial \sigma_j^i} = [\hat{y}^i(x) - \hat{y}(x)]\,\overline{w}_i \frac{(x_j - \overline{x}_j^i)^2}{(\sigma_j^i)^3}$$

Bármely gradiens technikán alapuló optimalizálás alkalmazható.

A következmény (konzekvens) részek paraméterei lineárisak $\hat{y}(x)$-ben:

$$\hat{y}(x) = \varphi^T(x)\vartheta := \left[\overline{w}_1(x^T \;\; 1) \quad \overline{w}_2(x^T \;\; 1)\right]\begin{bmatrix} c^1 \\ c^2 \end{bmatrix} = \begin{pmatrix} \overline{w}_1 x_1 & \overline{w}_1 x_2 & \overline{w}_1 & \overline{w}_2 x_1 & \overline{w}_2 x_2 & \overline{w}_2 \end{pmatrix}\begin{pmatrix} c_{11} \\ c_{12} \\ c_{10} \\ c_{21} \\ c_{22} \\ c_{20} \end{pmatrix}.$$

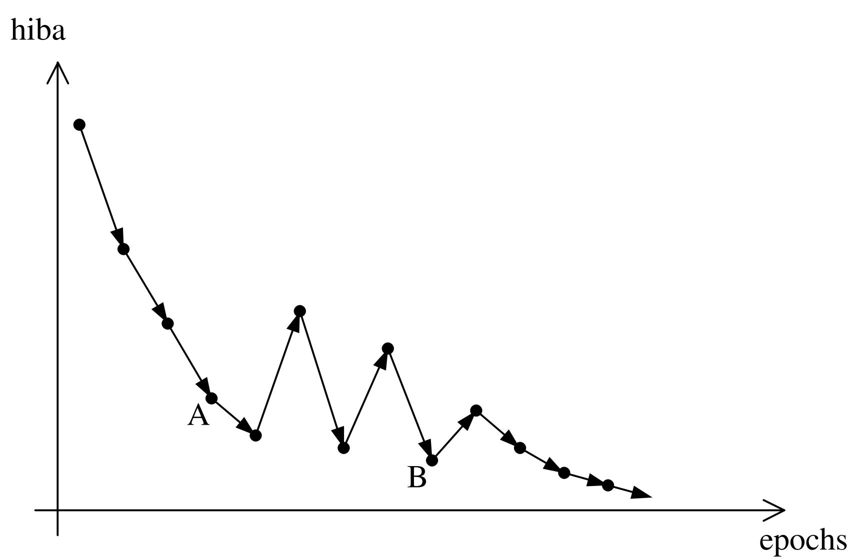
Az elsőrendű Sugeno fuzzy rendszer optimalizációja *hybrid* ANFIS technikával végezhető el.

Minden hangolási lépésben (epoch) a következőket hajtjuk végre:

1. A következmény rész paramétereinek LS hangolása (1 lépéses LS technika, gradiens alkalmazása nélkül – gyors, robusztus).
2. A maradék paraméterek hangolása gradiens technikákkal.

Megjegyzések:

- A következményrész paramétereinek hangolása növeli a konvergencia sebességét
- A lépésköz nagysága a gradiens technikánál adaptívan változtatható. Pl.
    a. A lépésköz növelése 4 egymásutáni ereszkedő fázis után (A)
    b. A lépésköz csökkentése 2 darab "föl-le" ciklus után (B)



Többkimnetű rendszer tanítása

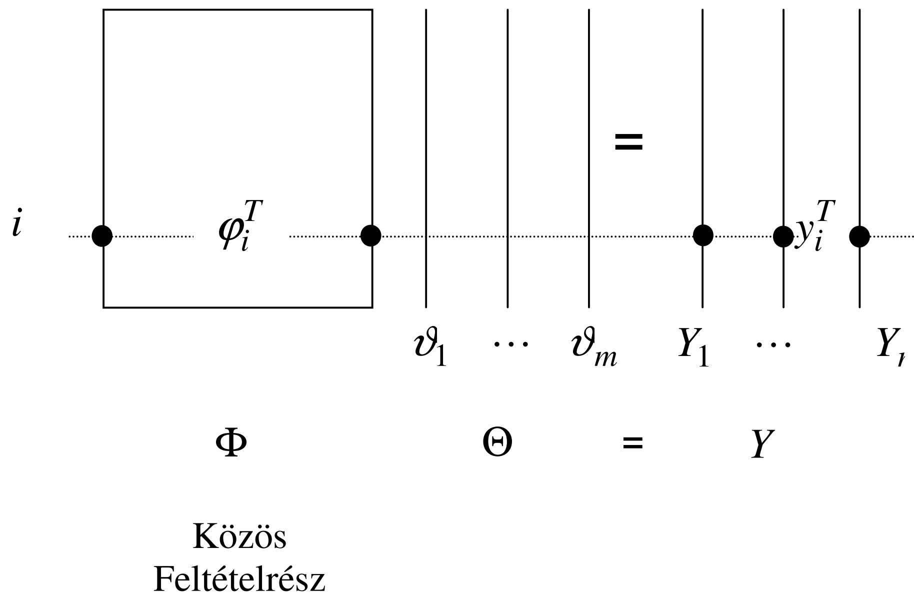
Koncepció: Tanítás minden kimenetre külön.

Egyszerűsítés:
Ha minden kimenetre azonos a feltételrész (R_i azonos minden kimenetre), akkor az LS probléma egyszerűsíthető:

$$Y_j = \Phi \vartheta_j, \quad j = 1, \ldots r, \quad r = \dim y \Rightarrow$$

$$\begin{bmatrix} Y_1 & \cdots & Y_r \end{bmatrix} = \Phi \begin{bmatrix} \vartheta_1 & \cdots & \vartheta_r \end{bmatrix}$$ (Φ közös minden ϑ_i paraméter vektorra)

$$Y = \Phi\Theta,$$



Mivel Y és Θ matrix, az LS módszer képletei általánosíthatók:

Batch mód: $$\hat{\Theta} = (\Phi^T \Phi)^{-1} \Phi^T Y$$

Rekurzív mód: $$P_i = \frac{1}{\lambda}\left\{P_{i-1} - P_{i-1}\varphi_i \frac{1}{\lambda + \varphi_i^T P_i \varphi_i} \varphi_i^T P_{i-1}\right\}$$

$$\hat{\Theta}_i = \hat{\Theta}_{i-1} + P_i \varphi_i (y_i^T - \varphi_i^T \hat{\Theta}_{i-1})$$

Megjegyzés: Előfordulhat olyan struktúra is, ahol a kimeneti paraméterek nemlineárisan jelennek meg a kimenetben. Ekkor a hibrid technika nem alkalmazható, azoban néhány esetben LS problémára vezethető vissza:

Ehhez legyen: $output = F(input, S)$
$\exists H(output)$ és $S = S_1 \oplus S_2$, hogy $H \circ F(input, S)$ már lineáris S_2-ben, így S_2-ben már alkalmazható LS.

Példa: Egy rejtett réteggel rendelkező FFNN, $s(x)$ szigmoid átmeneti függvénnyel :

$$H(s) = \ln \frac{s}{1-s}, \quad s(x) = \frac{1}{1+e^{-x}} \Rightarrow H \circ s(x) = \ln \frac{\frac{1}{1+e^{-x}}}{1 - \frac{1}{1+e^{-x}}} = \ln \frac{1}{e^{-x}} = x$$

$$x = w_1 x_1 + \cdots + w_n x_n + t \cdot 1 \Rightarrow S_2 = (w_1, \ldots, w_n, t)^T$$

5.4. Nemlineáris dinamikus rendszer identifikációja ANFIS használatával

Feladat: $y(t) = f(y(t-1), y(t-2), u(t-1), u(t-2))$ nemlineáris, diszkrétidejű model közelítése

azaz $y = f(x)$ nemlineáris függvény transzformációja, ahol $x = (x_1, x_2, x_3, x_4)^T$ és

$x_1(t) := y(t-1)$, $x_2(t) := y(t-2)$, $x_3(t) := u(t-1)$, $x_4(t) := u(t-2)$

Adott: $\{u(t), y(t)\}_{t=1}^{N}$ bemeneti/kimeneti adatpárok (mérési eredmények) a rendszeren.

Koncepció:

1. Sugeno fuzzy rendszer generálása szubtraktív klaszterezéssel (genfis2 MATLAB függvény).
   A generált fuzzy rendszer relációi:

$$R_i : \textit{if}\ \ x_1\ \ \textit{is}\ \ A_1^i\ \ \textit{and}\ \ \ldots \textit{and}\ \ x_n\ \ \textit{is}\ \ A_n^i\ \ \textit{then}\ \ y\ \ \textit{is}\ \ c_{i1}x_1 + \cdots c_{in}x_n + c_{i0},$$

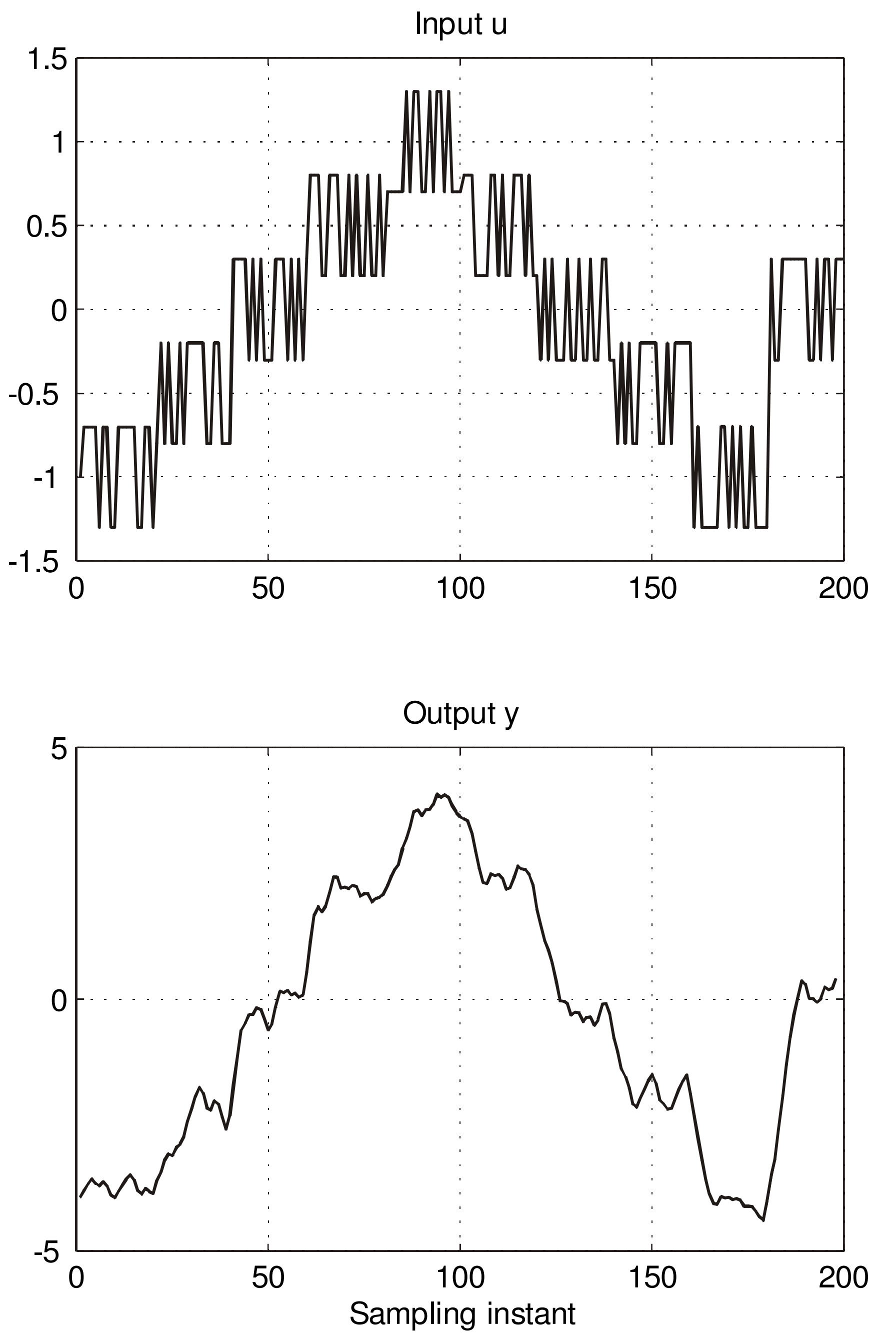
$$i = 1, \ldots, m,$$

(speciálisan $c_{i1} = \cdots = c_{in} = 0$ nulladrendű Sugeno-rendszer esetén)

2. A Sugeno-rendszer tovább javítása ANFIS-al (*anfis* MATLAB függvény)

Példa:

Bemenet és kimenet:



Eredmények:

- A meghívott anfis függvény 3 Gauss-tagsági függvényt generált minden változóra,
- 3 reláció generálása:

$$R_1: if \;\; x_1 \;\; is \; \exp\left(-\frac{(x_1+1.589)^2}{2\cdot 2.981^2}\right) \; and \;\; x_2 \;\; is \; \exp\left(-\frac{(x_2+1.744)^2}{2\cdot 2.983^2}\right)$$

$$and \;\; x_3 \;\; is \; \exp\left(-\frac{(x_3+0.2744)^2}{2\cdot 0.9167^2}\right) \; and \;\; x_4 \;\; is \; \exp\left(-\frac{(x_4+0.2179)}{2\cdot 0.8897^2}\right)$$

$$then \;\; \hat{y}_1 = 1.048x_1 - 0.2336x_2 + 0.4642x_3 + 0.1942x_4 - 0.02663,$$

$$R_2: if \;\; x_1 \;\; is \; \exp\left(-\frac{(x_1-2.435)^2}{2\cdot 2.99^2}\right) \; and \;\; x_2 \;\; is \; \exp\left(-\frac{(x_2-2.261)^2}{2\cdot 2.991^2}\right)$$

$$and \;\; x_3 \;\; is \; \exp\left(-\frac{(x_3-0.7452)^2}{2\cdot 0.872^2}\right) \; and \;\; x_4 \;\; is \; \exp\left(-\frac{(x_4-0.7353)^2}{2\cdot 0.8807^2}\right)$$

$$then \;\; \hat{y}_1 = 1.028x_1 - 0.2044x_2 + 0.3663x_3 + 0.2104x_4 + 0.1375,$$

$$R_3: if \;\; x_1 \;\; is \; \exp\left(-\frac{(x_1+4.001)^2}{2\cdot 2.981^2}\right) \; and \;\; x_2 \;\; is \; \exp\left(-\frac{(x_2+3.963)^2}{2\cdot 2.981^2}\right)$$

$$and \;\; x_3 \;\; is \; \exp\left(-\frac{(x_3+1.218)^2}{2\cdot 1.053^2}\right) \; and \;\; x_4 \;\; is \; \exp\left(-\frac{(x_4+1.258)^2}{2\cdot 1.003^2}\right)$$

$$then \;\; \hat{y}_1 = 1.057x_1 - 0.2306x_2 + 0.1677x_3 + 0.09548x_4 - 0.4582.$$

Az anfis által szolgáltatott eredmény: