Control prin învăţare

Master ICAF, An 1 Sem 2

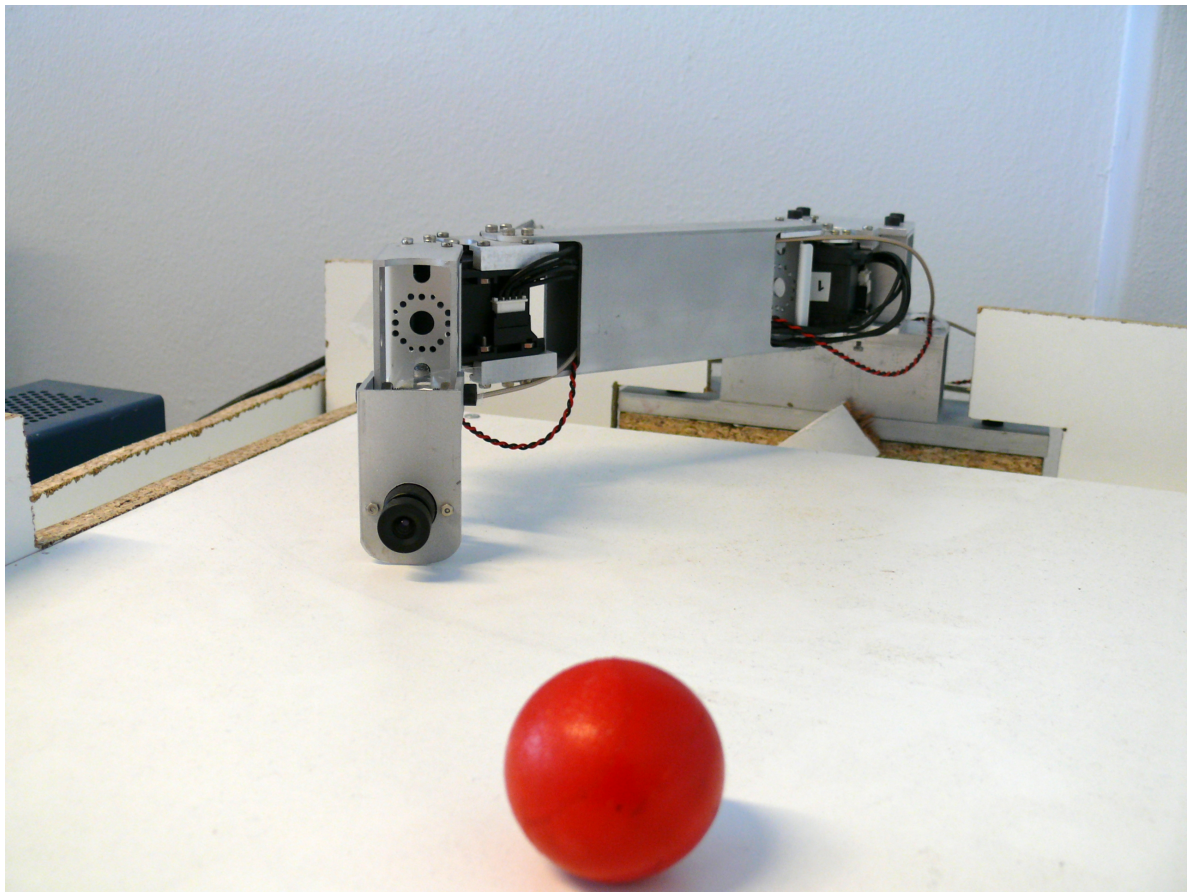
Lucian Buşoniu

Partea IV

Aproximare. Programarea dinamică cu aproximare. Învăţarea prin recompensă offline

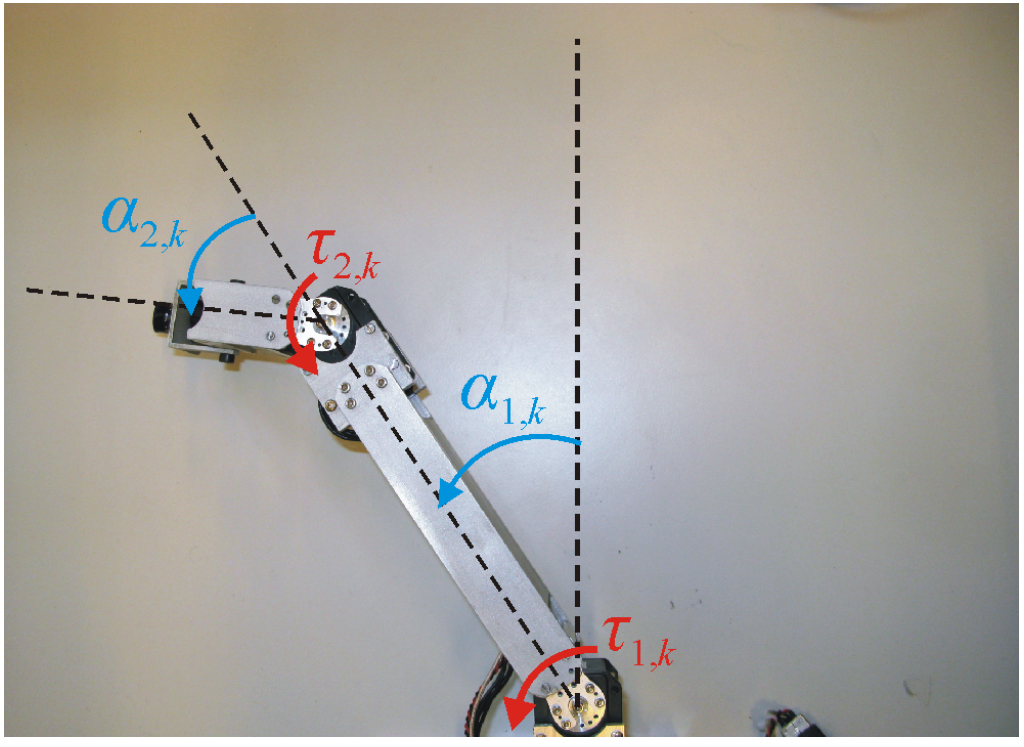
RL pentru un robot-portar (TUDelft)

Învaţă să prindă mingea folosind camera video



Nevoia de aproximare

- RL clasică – reprezentare sub formă de tabel, de ex. $Q(x, u)$ separat pentru toate valorile x şi u
- În aplicaţii reale de control, x, u **continue**!



- **Reprezentarea prin tabel imposibilă**

Nevoia de aproximare (continuare)

In aplicaţii reale de control,
funcţiile de interes trebuie **aproximate**

Partea IV în plan

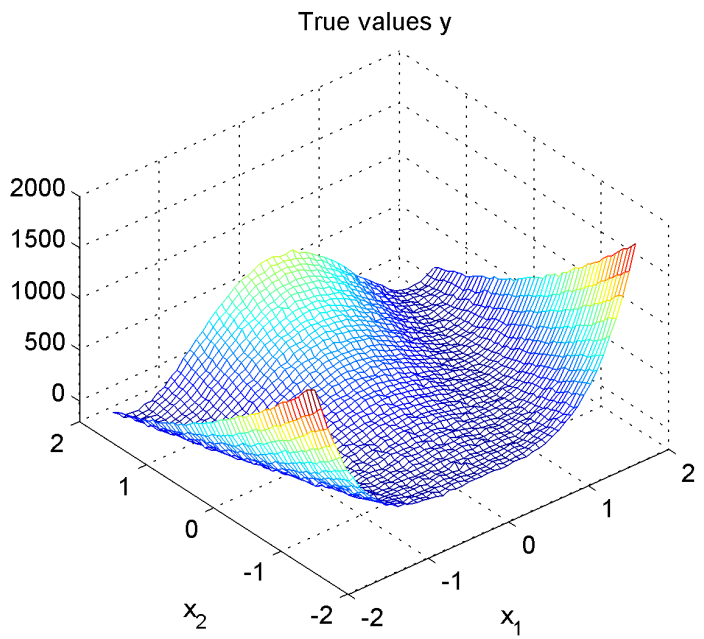
- Problema de învăţare prin recompensă
- Soluţia optimală
- Programarea dinamică exactă
- Învăţarea prin recompensă exactă
- **Tehnici de aproximare**
- **Programarea dinamică cu aproximare (var. continue)**
- Învăţarea prin recompensă cu aproximare (var. continue)

Conţinut partea IV

Aproximare

Aproximare:
funcţie cu un număr infinit de valori
$\longrightarrow$ reprezentare printr-un număr mic de valori

$f(x)$



$\widehat{f}(x)$

?

Aproximare parametrică

Aproximare parametrică: $\widehat{f}(x)$ formă fixată, valoare determinată de vector de **parametri** θ:

$$\widehat{f}(x;\theta)$$

1. Aproximare liniară – combinaţie ponderată de **funcţii de bază** ϕ:

$$\begin{aligned}\widehat{f}(x;\theta) &= \phi_1(x)\theta_1 + \phi_2(x)\theta_2 + \ldots \phi_n(x)\theta_n \\ &= \sum_{i=1}^{n} \phi_i(x)\theta_i = \phi^\top(x)\theta\end{aligned}$$

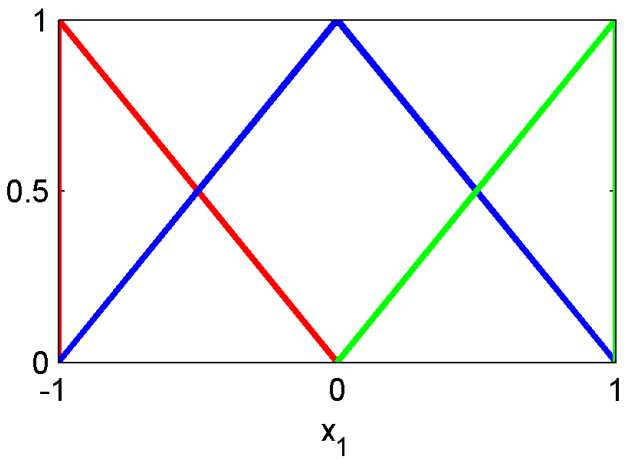
   De notat: liniară în parametri, poate fi neliniară în x

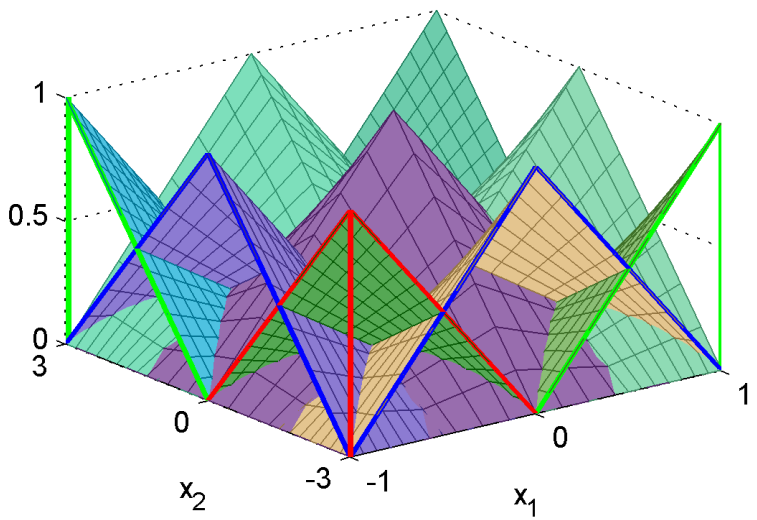
2. Aproximare neliniară: rămâne în forma generală

Aproximare parametrică liniară: Interpolare

Interpolare:

- Grilă D-dimensională de puncte
- Interpolare multiniară între puncte
- Echivalent cu funcţii de bază **piramidale**





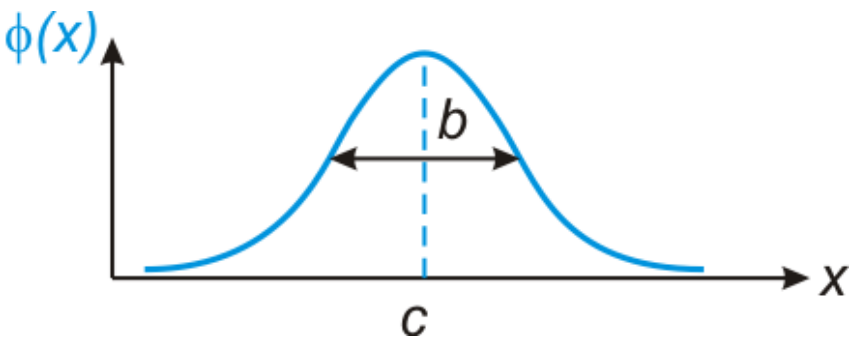
Aproximare parametrică liniară: RBF

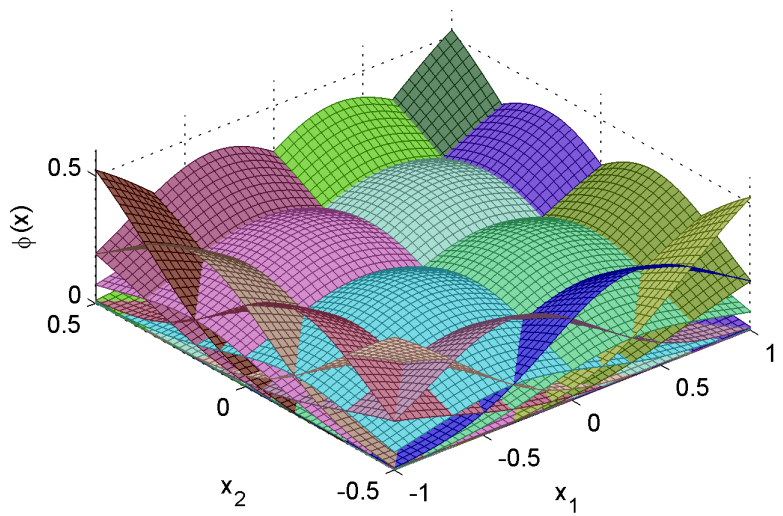
Funcţie de bază radială (Gaussiană):

$$\phi(x) = \exp\left[-\frac{(x-c)^2}{b^2}\right] \qquad \text{(1-dim);}$$

$$= \exp\left[-\sum_{d=1}^{D}\frac{(x_d - c_d)^2}{b_d^2}\right] \qquad \text{(D-dim)}$$

Eventual, normalizare: $\tilde{\phi}_i(x) = \frac{\phi_i(x)}{\sum_{i' \neq i}\phi_{i'}(x)}$





Antrenarea aproximatoarelor liniare: CMMP

- n_s puncte $(x_j, f(x_j))$, obiectivul descris ca un sistem de ecuaţii:

$$\widehat{f}(x_1;\theta) = \phi_1(x_1)\theta_1 + \phi_2(x_1)\theta_2 + \ldots \phi_n(x_1)\theta_n = f(x_1)$$
$$\ldots$$
$$\widehat{f}(x_{n_s};\theta) = \phi_1(x_{n_s})\theta_1 + \phi_2(x_{n_s})\theta_2 + \ldots \phi_n(x_{n_s})\theta_n = f(x_{n_s})$$

- Formă matriceală:

$$\begin{bmatrix} \phi_1(x_1) & \phi_2(x_1) & \ldots & \phi_n(x_1) \\ \ldots & \ldots & \ldots & \ldots \\ \phi_1(x_{n_s}) & \phi_2(x_1) & \ldots & \phi_n(x_{n_s}) \end{bmatrix} \cdot \theta = \begin{bmatrix} f(x_1) \\ \ldots \\ f(x_{n_s}) \end{bmatrix} \qquad A\theta = b$$

- Regresie liniară, vezi SysID

CMMP (continuare)

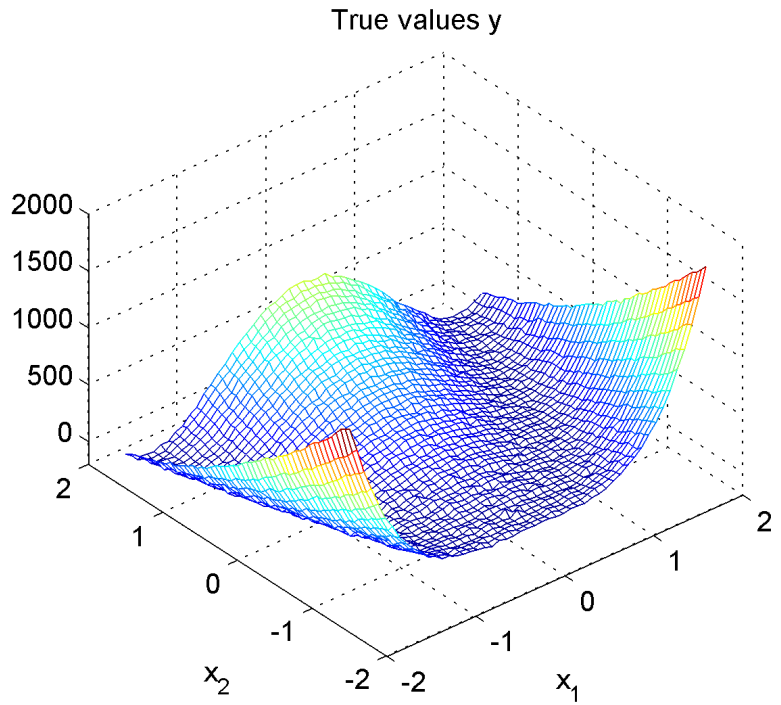
- Sistemul este supradeterminat ($n_s > n$), ecuaţiile nu vor fi toate satisfăcute cu egalitate.
  $\Rightarrow$ Rezolvare în sensul celor mai mici pătrate:

$$\min_{\theta} \sum_{j=1}^{n_s} \left| f(x_j) - \widehat{f}(x_j; \theta) \right|^2$$

  ...algebră şi analiză liniară...

- $\theta = (A^\top A)^{-1} A^\top b$ (Intuiţie: $(A^\top A)\theta = A^\top b$)

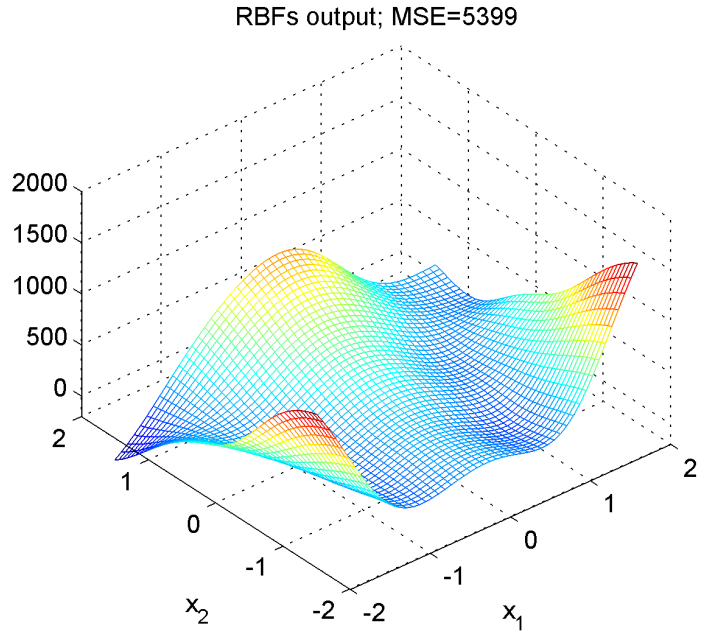
Exemplu: Funcţia “banană” Rosenbrock



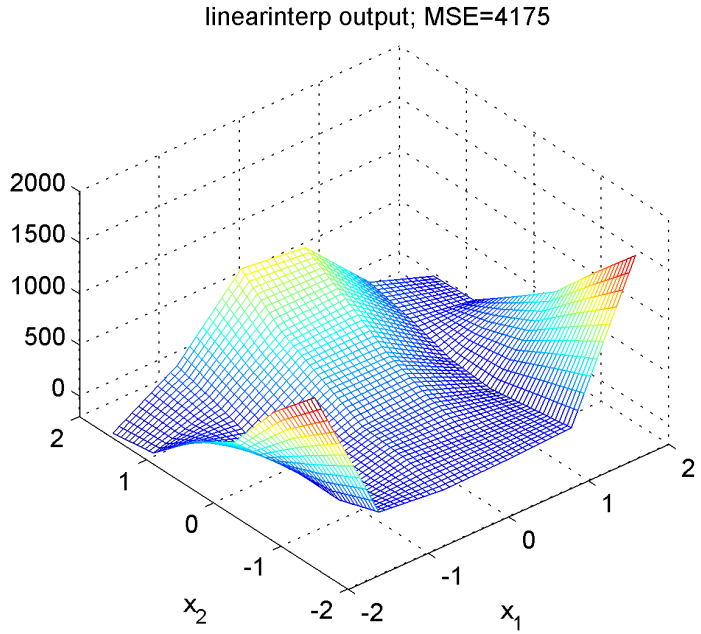
- $f(x) = (1 - x_1)^2 + 100[(x_2 + 1.5) - x_1^2]^2, \qquad x = [x_1, x_2]^\top$
- Antrenare: 200 puncte distribute aleator
- Validare: grilă de 31×31 puncte

Funcţia Rosenbrock: Rezultat cu aproximatoare liniare

6x6 RBFuri:



Interpolare pe grilă 6x6:

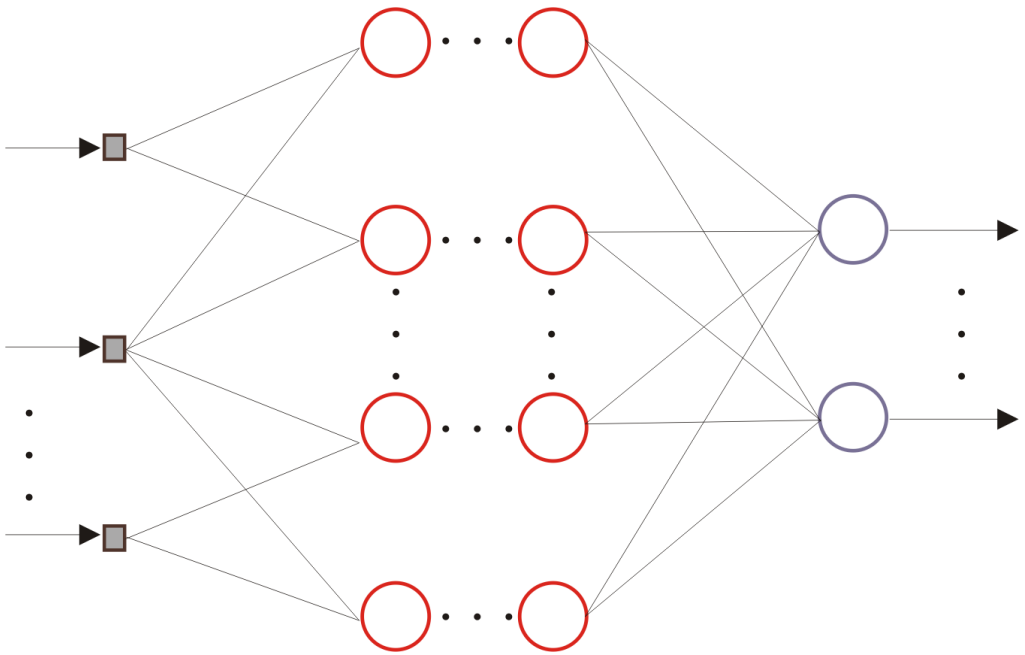


- Aproximarea RBF mai netedă (RBFuri late)
- Interpolarea = colecţie de suprafeţe multiliniare

Aproximare parametrică neliniară: reţea neuronală

Reţea neuronală:

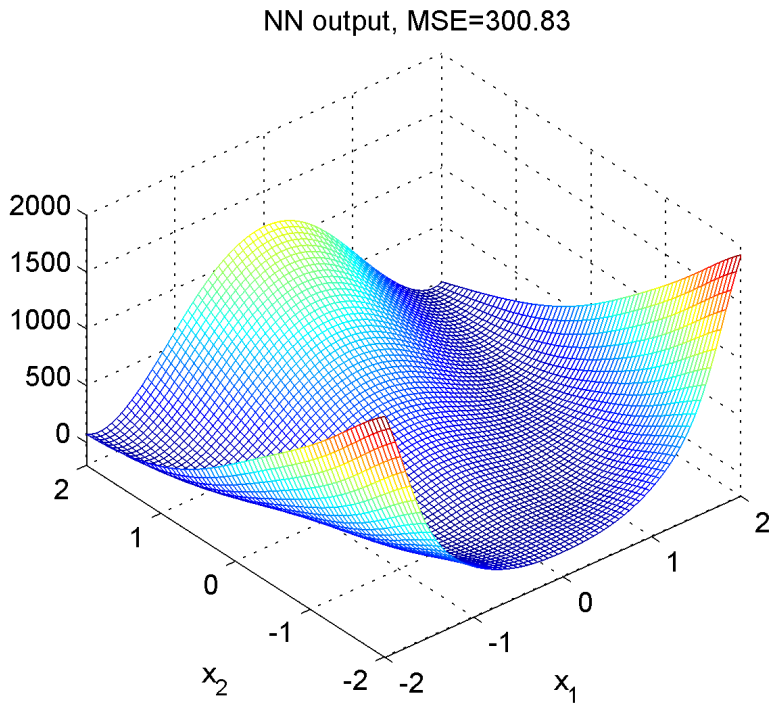
- **Neuroni** cu funcţii de activare (ne)liniare
- **Interconectaţi** pe nivele multiple, prin legături ponderate



Va urma...

Funcţia Rosenbrock: Rezultat cu reţea neuronală

Un nivel ascuns cu 10 neuroni şi funcţii de activare tangent-sigmoidale (liniare pentru nivelul de ieşire). 500 epoci de antrenare.



Datorită flexibilităţii mai bune a reţelei neuronale, rezultatele sunt mai bune decât cu aproximatoarele liniare.

Aproximare neparametrică

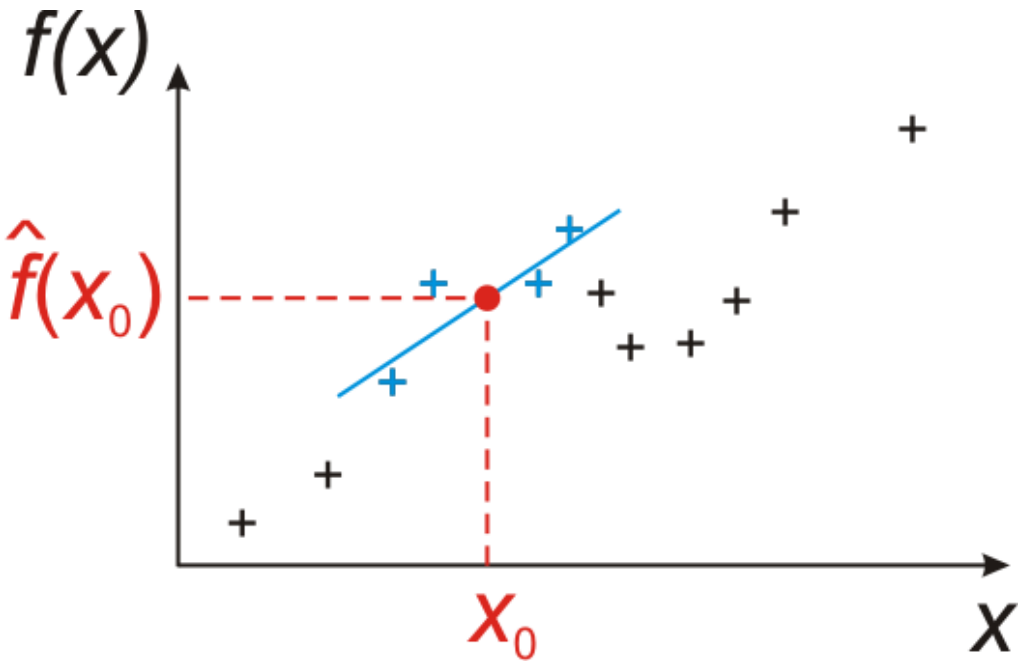
Aproximare parametrică:
formă, număr de parametri fixate

Aproximare neparametrică:
formă, număr de parametri **depind de date**

Aproximare neparametrică: LLR

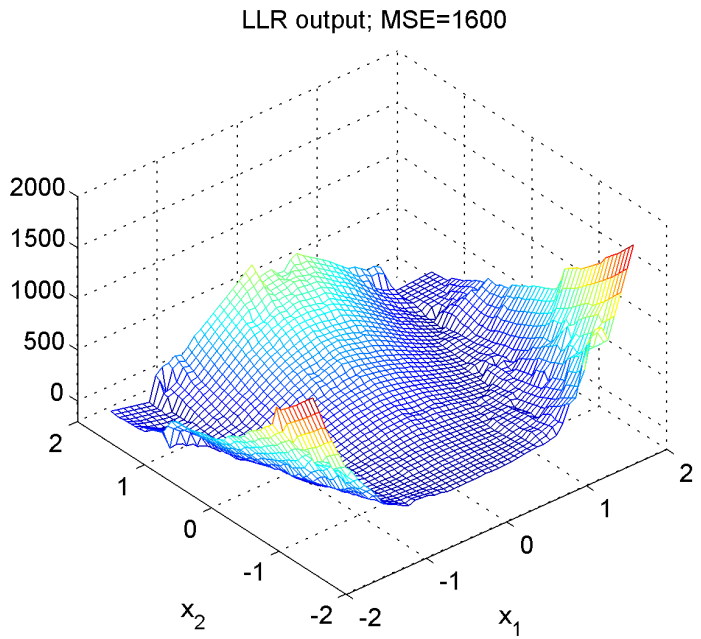
Regresie liniară locală, LLR:

- Bază de date cu puncte de forma $(x, f(x))$
- Pentru x_0 dat, găseşte ***k* cei mai apropiaţi vecini**
- Rezultat calculat cu **regresie liniară** (CMMP) pe vecini



Funcţia Rosenbrock: Rezultat cu LLR

Baza de date = 200 de puncte de antrenament; $k = 5$
Validare: grilă de 31×31 puncte



- Performanţa între aproximatoarele liniare şi reţeaua neuronală

Comparaţie între aproximatoare

În combinaţie cu RL

- liniare mai uşor de tratat teoretic
  decât neliniare
- parametrice mai uşor de tratat teoretic
  decât neparametrice

Flexibilitate

- neliniare mai flexibile decât liniare
- nonparametrice mai flexibile decât parametrice:
  forma aprox. parametrice trebuie adaptată manual
- nonparametrice se adaptează la date:
  complexitatea trebuie controlată când #date creşte

Aproximarea în RL

Probleme de rezolvat:

1. Reprezentare: $Q(x, u)$, $V(x)$, $h(x)$
   Folosind metodele de aproximare discutate

2. Maximizare: ex. $\max_u Q(x, u)$

Maximizare soluţia 1: h implicit

- Legea de control nu este reprezentată explicit
- Acţiuni greedy calculate la cerere din $\widehat{Q}$:

$$h(x) = \arg\max_u \widehat{Q}(x, u)$$

- Problema principală: **aproximarea funcţiei Q**
- Aproximatorul trebuie să garanteze **soluţie eficientă pentru** arg max

Maximizare soluţia 2: h explicit

- Legea de control aproximată explicit: $\widehat{h}(x)$

Avantaje:

- **Acţiuni continue** mai uşor de folosit
- Reprezentarea poate include mai uşor **cunoştinţe a priori**

Focus: Soluţia 1, *h* implicit

În acest curs:

- Legea de control nu este reprezentată explicit
- Problema principală: **aproximarea funcţiei Q**

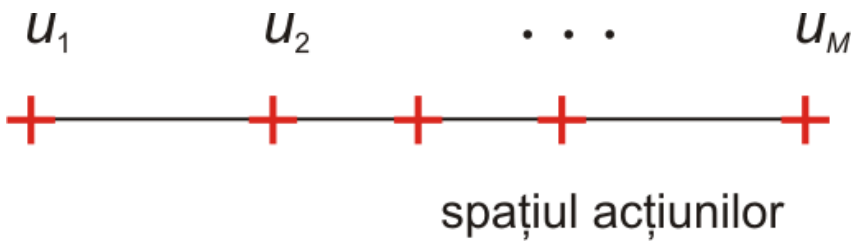
Discretizarea acţiunilor

- Aproximatorul trebuie să garanteze soluţie eficientă pentru arg max

$\Rightarrow$ Tipic: **discretizare acţiuni**

- Alege M acţiuni discrete $u_1, \ldots, u_M \in U$
  Calculează "arg max" folosind enumerare explicită

- Exemplu: discretizare pe o grilă

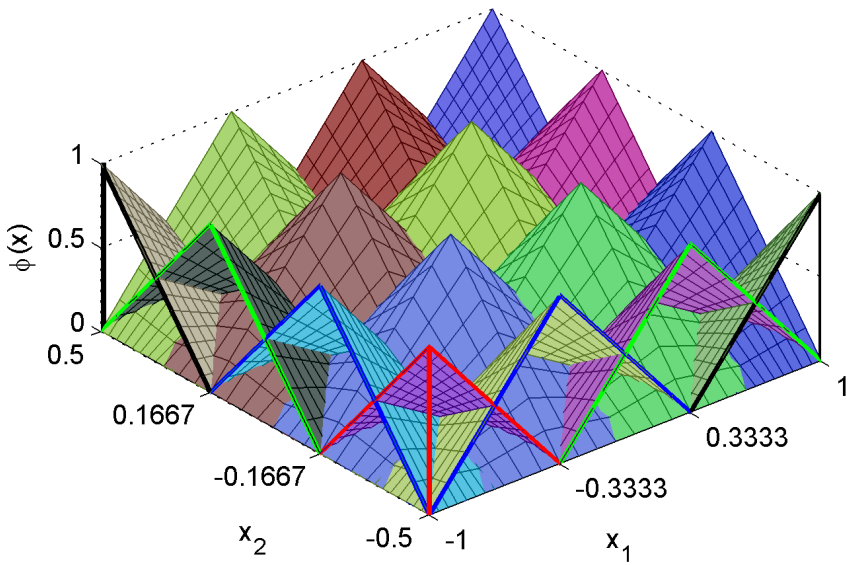


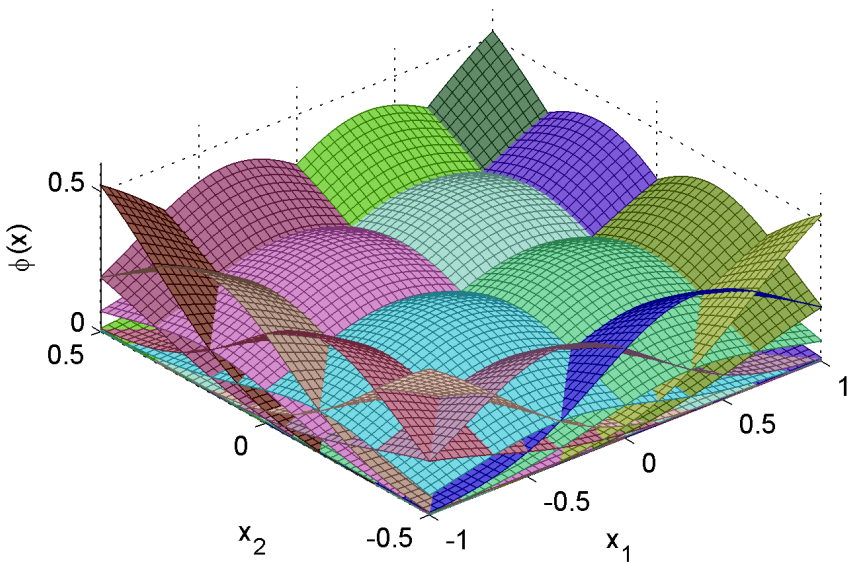
Aproximare în spaţiul stărilor

- Tipic: funcţii de bază

$$\phi_1, \ldots, \phi_N : X \to [0, \infty)$$

- Ex. piramidale, RBF





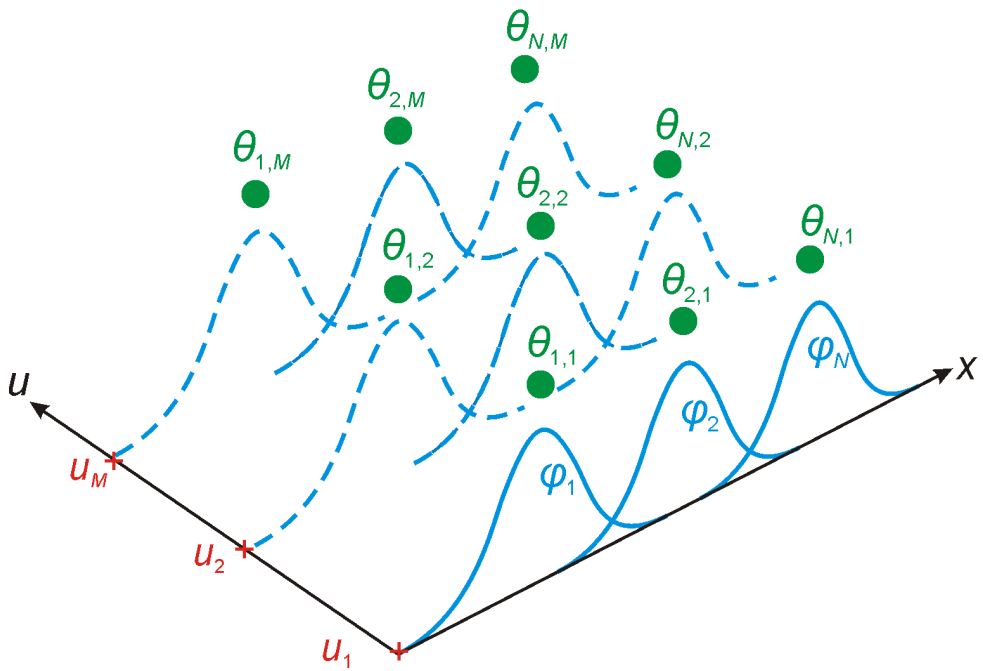
Funcţia Q aproximată cu acţiuni discrete

Date fiind:

1. N funcţii de bază $\phi_1, \ldots, \phi_N$
2. M acţiuni discrete $u_1, \ldots, u_M$

Stochează:

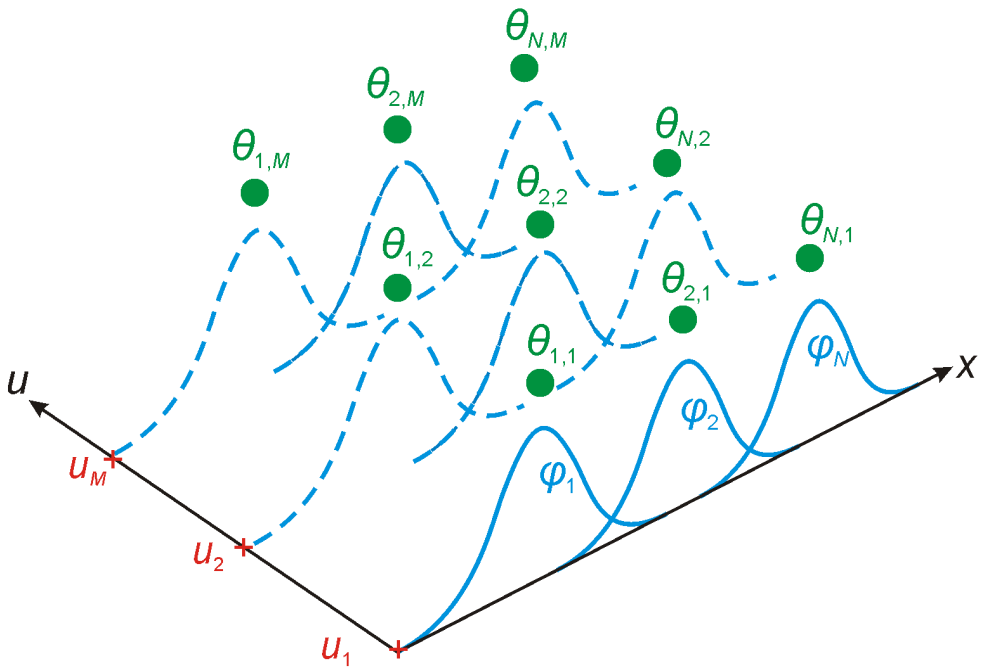
3. $N \cdot M$ parametri θ
   (pentru fiecare pereche funcţie de bază–acţiune discretă)



Funcţia Q aproximată cu acţiuni discrete (continuare)

Funcţia Q aproximată:

$$\widehat{Q}(x, u_j; \theta) = \sum_{i=1}^{N} \phi_i(x)\theta_{i,j} = [\phi_1(x) \dots \phi_N(x)] \begin{bmatrix} \theta_{1,j} \\ \vdots \\ \theta_{N,j} \end{bmatrix}$$



Beneficiul aproximării în RL

Aproximarea permite aplicarea RL în probleme realiste de control

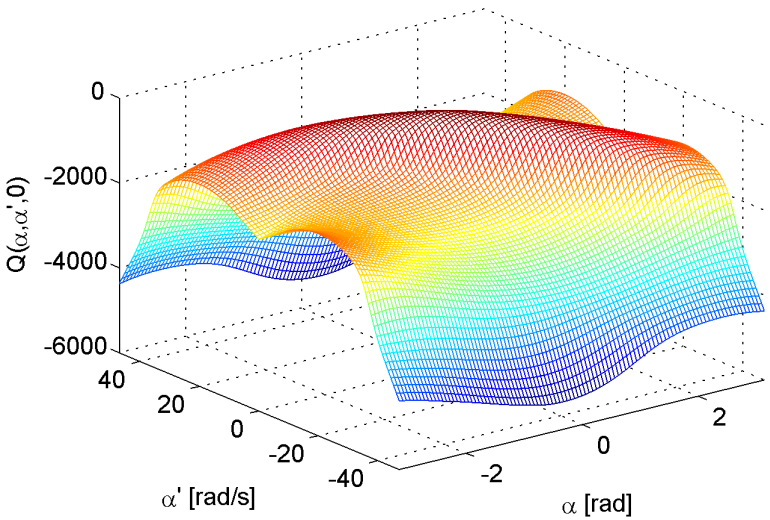
Exemplu: Pendul inversat

- $x = [\text{unghi } \alpha,\ \text{viteză } \dot{\alpha}]^\top$
- u = voltaj
- $\rho(x, u) = -x^\top \begin{bmatrix} 5 & 0 \\ 0 & 0.1 \end{bmatrix} x - u^\top 1 u$
- Factor de discount $\gamma = 0.98$

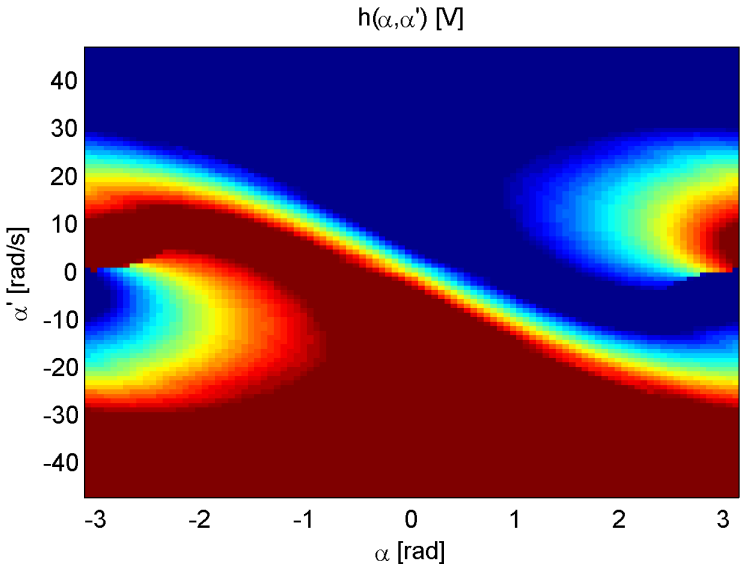
- Obiectiv: stabilizează orientat în sus
- Putere insuficientă $\Rightarrow$ balansează înainte & înapoi

Pendul inversat: Soluţie optimală

Stânga: Funcţia Q pentru $u = 0$



Dreapta: legea de control



▸ Replay

Întrebări ridicate de aproximare

1. **Convergenţă**: rămâne algoritmul convergent?
2. **Calitatea soluţiei**: la o distanţă controlată de optim?
3. **Consistenţă**: pentru un aproximator ideal, de precizie infinită, este soluţia optimală regăsită?

Algoritmii din partea IV în taxonomie

După utilizarea unui model:

- Bazat pe model: f, ρ cunoscute
- Fără model: doar date (învăţarea prin recompensă)

După nivelul de interacţiune:

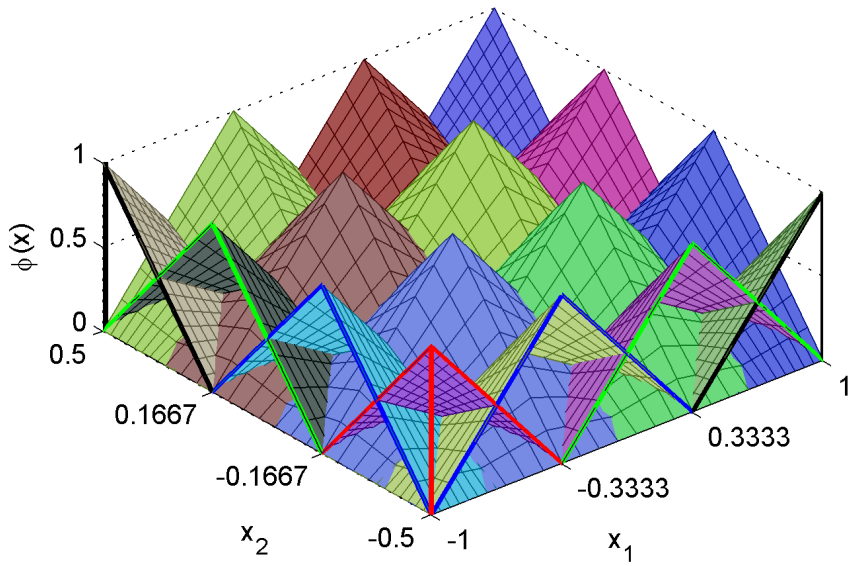
- Offline: algoritmul rulează în avans
- Online: algoritmul controlează direct sistemul

Exact vs. cu aproximare:

- Exact: x, u număr mic de valori discrete
- Cu aproximare: x, u continue (sau multe valori discrete)

Approximator cu interpolare ("fuzzy")

- Interpolare = BF piramidale



- Fiecare BF i are centrul x_i
- $\theta_{i,j}$ poate fi văzut ca $\widehat{Q}(x_i, u_j)$, din motivul: $\phi_i(x_i) = 1, \phi_{i'}(x_i) = 0$ pentru $i' \neq i$

Iteraţia Q cu interpolare (fuzzy Q)

Reamintim iteraţia Q clasică:

repeat la fiecare iteraţie ℓ
  for all x, u **do**
    $Q_{\ell+1}(x,u) \leftarrow \rho(x,u) + \gamma \max_{u'} Q_\ell(f(x,u),u')$
  end for
until convergenţă

Iteraţia fuzzy Q

repeat la fiecare iteraţie ℓ
  for all centrele x_i, acţiunile discrete u_j **do**
    $\theta_{\ell+1,i,j} \leftarrow \rho(x_i,u_j) + \gamma \max_{j'} \widehat{Q}(f(x_i,u_j),u_{j'};\theta_\ell)$
  end for
until convergenţă

Lege de control

- Reamintim legea optimală de control:

$$h^*(x) = \arg\max_u Q^*(x, u)$$

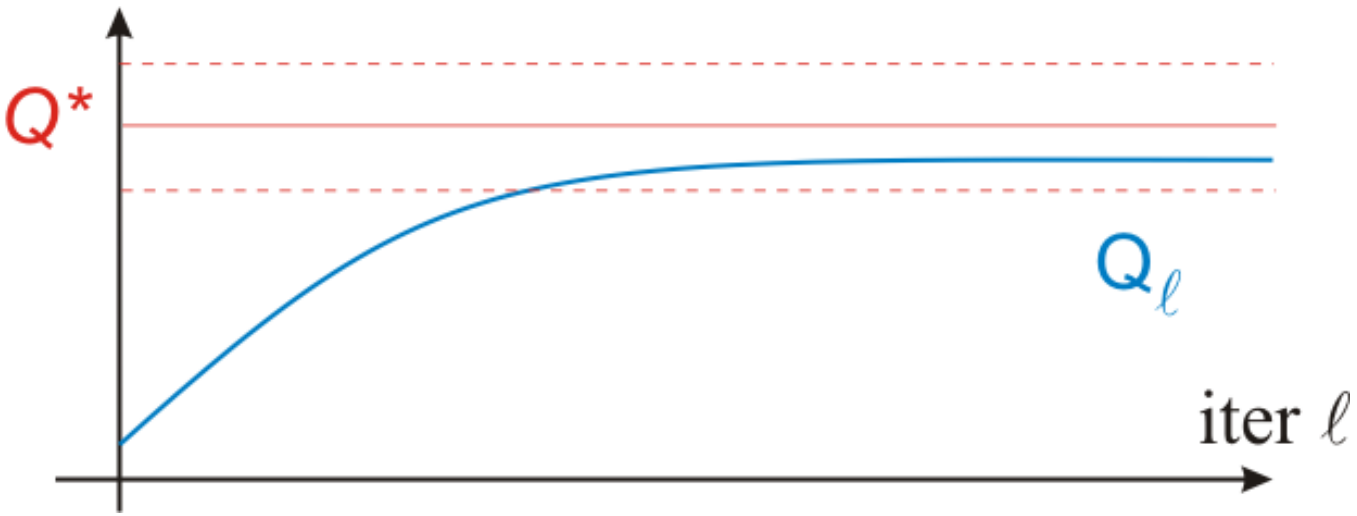
- În iteraţia fuzzy Q:

$$\widehat{h}^*(x) = \arg\max_{u_j,\ j=1,\ldots,M} \widehat{Q}(x, u_j; \theta^*)$$

θ^* = parametrii la convergenţă

Convergenţă în imagini

Convergenţă monotonă la o soluţie **aproape-optimală**



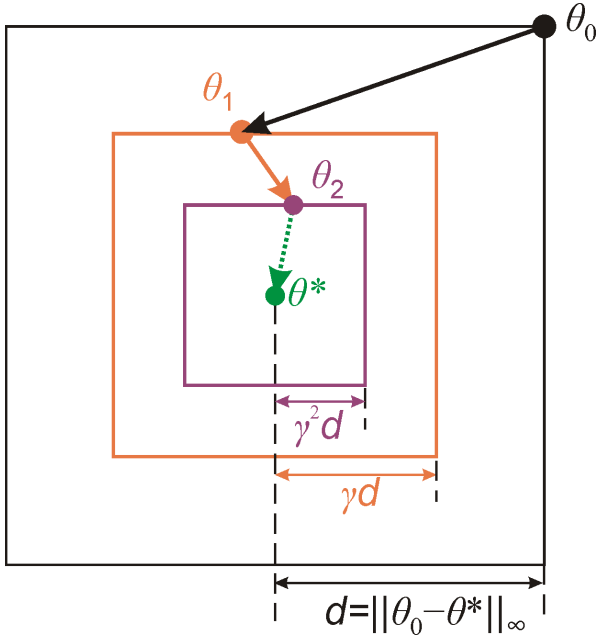
Convergenţă

Similar cu iteraţia Q clasică:

- Fiecare iteraţie este o contracţie cu factor γ:

$$\|\theta_{\ell+1} - \theta^*\|_\infty \leq \gamma \|\theta_\ell - \theta^*\|_\infty$$

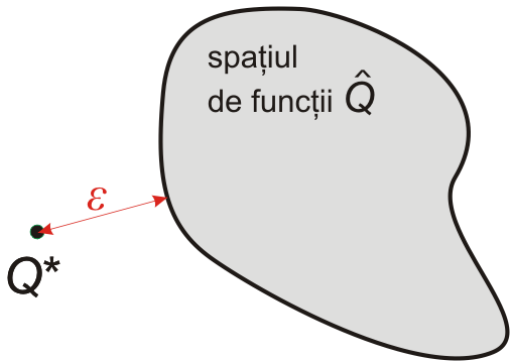
$\Rightarrow$ **Convergenţă monotonă** la θ^*



Calitatea soluţiei

Caracterizăm aproximatorul prin distanţa minimă până la Q^*:

$$\varepsilon = \min_{\theta} \left\| Q^*(x,u) - \widehat{Q}(x,u;\theta) \right\|_\infty$$



Avem:

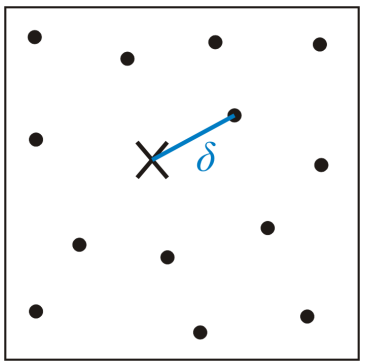
1. Suboptimalitatea rezultatului $\widehat{Q}(x,u;\theta^*)$ mărginită:

$$\left\| Q^*(x,u) - \widehat{Q}(x,u;\theta^*) \right\|_\infty \leq \frac{2\varepsilon}{1-\gamma}$$

2. Suboptimalitatea legii de control $\widehat{h}^*$ mărginită, $\frac{4\varepsilon}{(1-\gamma)^2}$

Consistenţă

- Consistenţă: $\widehat{Q}^{\theta^*} \rightarrow Q^*$ pe măsură ce precizia creşte
- Precizia: $\begin{cases} \delta_x = \max\limits_x \min\limits_i \|x - x_i\|_2 \\ \delta_u = \max\limits_u \min\limits_j \|u - u_j\|_2 \end{cases}$



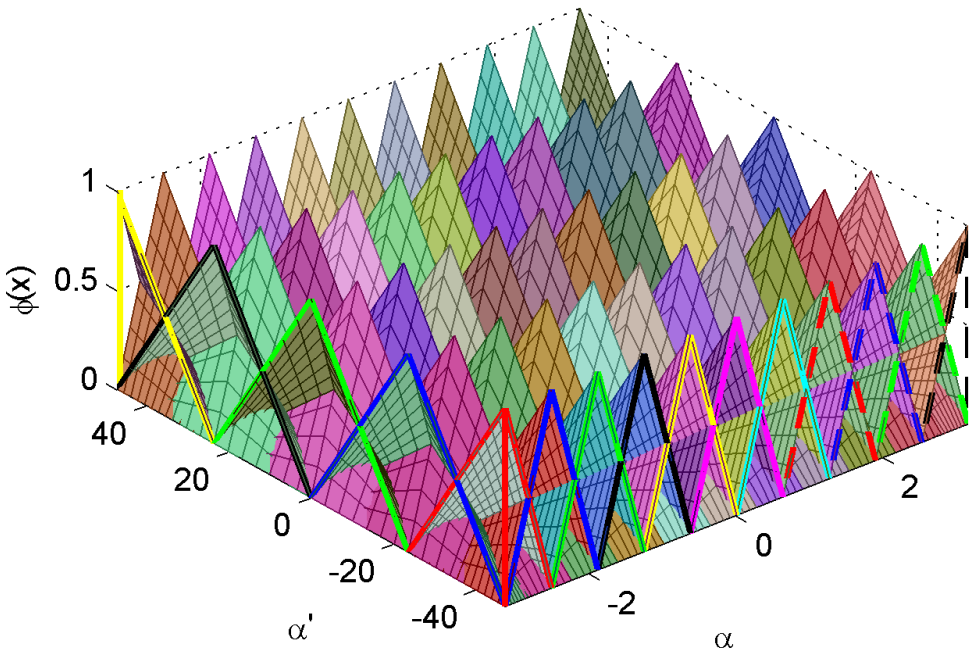
- Date fiind anumite condiţii tehnice,

  $\Rightarrow \lim_{\delta_x \rightarrow 0, \delta_u \rightarrow 0} \widehat{Q}^{\theta^*} = Q^*$ — consistenţă

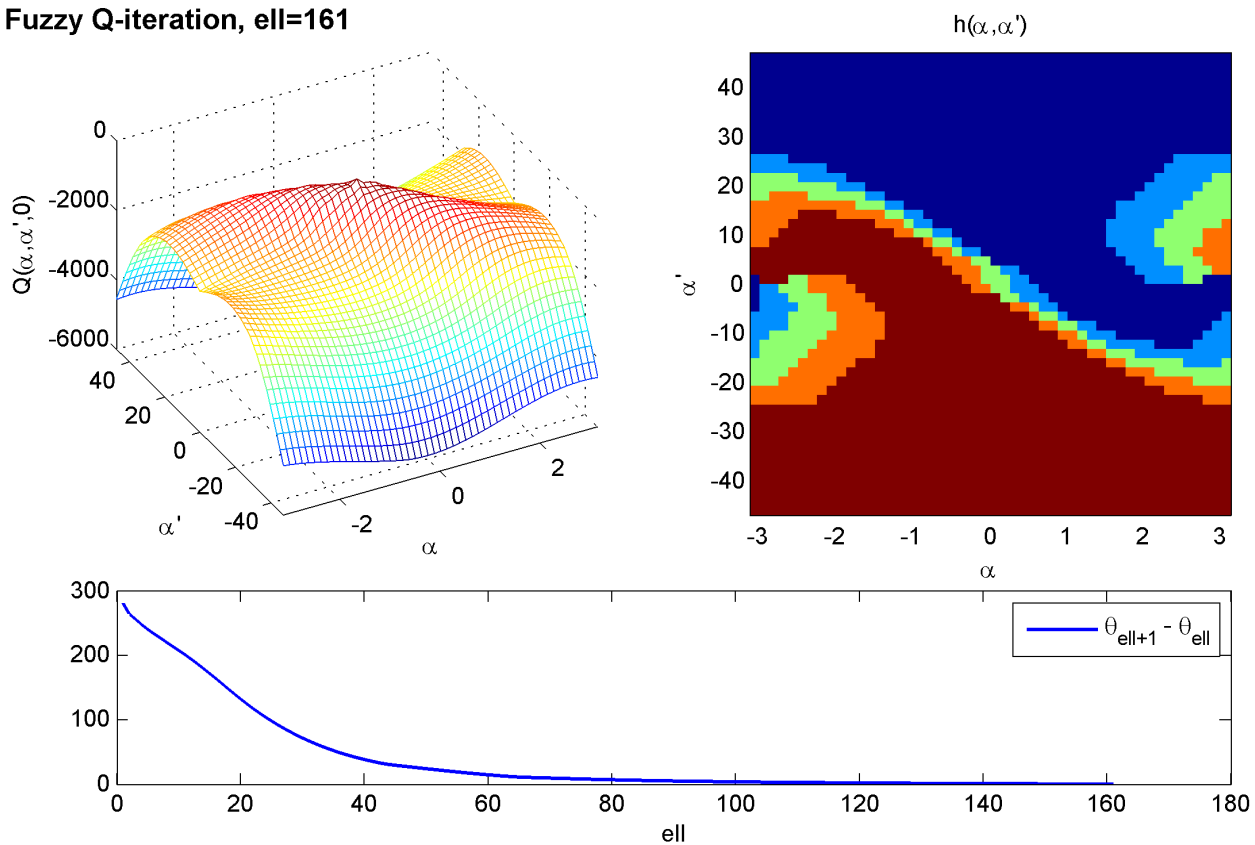
Pendul inversat: Iteraţia fuzzy Q, demo

BF: grilă echidistantă 41×21

Discretizare: 5 acţiuni, distribuite în jurul lui 0



Pendul inversat: Iteraţia fuzzy Q, demo



Iteraţia Q bazată pe date

Pornim de la iteraţia fuzzy Q şi o extindem:

- folositoare cu alte aproximatoare decât interpolare/fuzzy
- **fără model** – RL

Algoritm intermediar bazat pe model

Reamintim iteraţia fuzzy Q:

for all x_i, u_j $\quad \theta_{\ell+1,i,j} \leftarrow \rho(x_i, u_j) + \gamma \max_{j'} \widehat{Q}(f(x_i, u_j), u_{j'}; \theta_\ell)$ **end for**

1. Folosim eşantioane stare-acţiune arbitrare
2. Extindem la aproximator generic
3. Găsim parametrii folosind cele mai mici pătrate

date fiind (x_s, u_s), $s = 1, \ldots, n_s$
repeat la fiecare iteraţie ℓ
  for $s = 1, \ldots, n_s$ **do**
    $q_s \leftarrow \rho(x_s, u_s) + \gamma \max_{u'} \widehat{Q}(f(x_s, u_s), u'; \theta_\ell)$
  end for
  $\theta_{\ell+1} \leftarrow \arg\min \sum_{s=1}^{n_s} \left| q_s - \widehat{Q}(x_s, u_s; \theta) \right|^2$
until terminare

Iteraţia fuzzy Q echivalentă cu algoritmul generalizat dacă eşantioanele sunt toate combinaţiile x_i, u_j

Iteraţia Q bazată pe date: Algoritm

4. Folosim tranziţii în loc de model

Iteraţia Q bazată pe date

date fiind (x_s, u_s, r_s, x'_s), $s = 1, \ldots, n_s$
repeat la fiecare iteraţie ℓ
  for $s = 1, \ldots, n_s$ **do**
    $q_s \leftarrow r_s + \gamma \max_{u'} \widehat{Q}(x'_s, u'; \theta_\ell)$
  end for
  $\theta_{\ell+1} \leftarrow \arg\min \sum_{s=1}^{n_s} \left| q_s - \widehat{Q}(x_s, u_s; \theta) \right|^2$
until terminare

Determinist versus stohastic

- În cazul determinist, $x'_s = f(x_s, u_s)$, $r_s = \rho(x_s, u_s)$
  – înlocuirile sunt exacte
- În cazul stohastic, $x'_s \sim \tilde{f}(x_s, u_s, \cdot)$, $r_s = \tilde{\rho}(x_s, u_s, x'_s)$

$\Rightarrow$ Algoritmul **rămâne valid**; intuiţie:

- Ideal, $Q(x, u) \leftarrow \mathrm{E}_{x'} \left\{ r + \gamma \max_{u'} \widehat{Q}(x', u'; \theta_\ell) \right\}$
- Presupunând n_s eşantioane, toate în $(x_s, u_s) = (x, u)$:

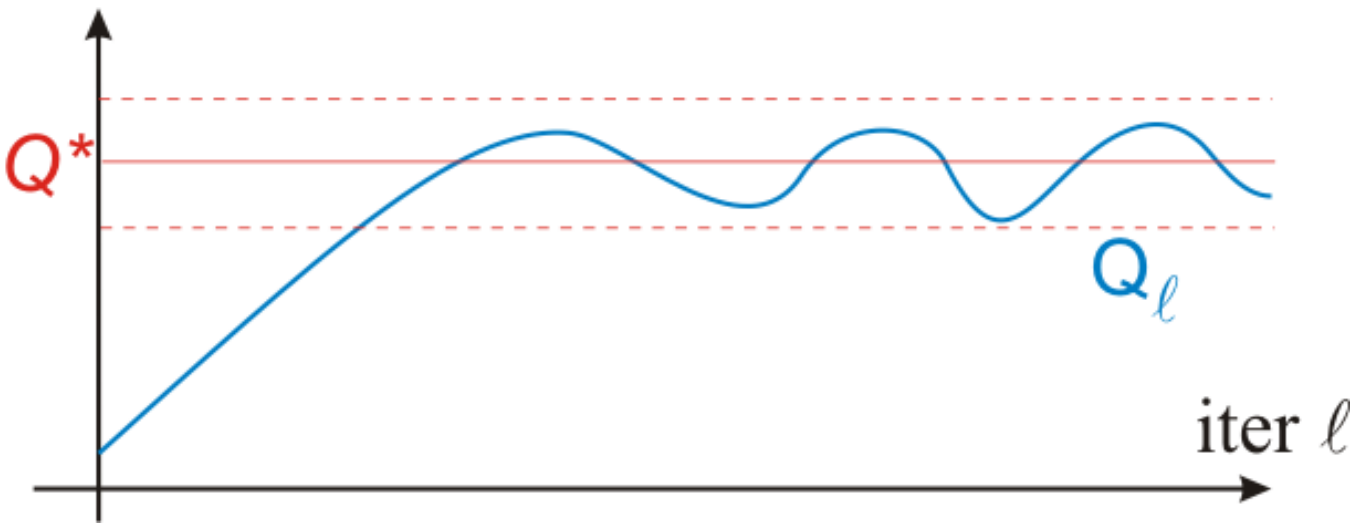
$$\min_\theta \sum_{s=1}^{n_s} \left| r_s + \gamma \max_{u'} \widehat{Q}(x'_s, u'; \theta_\ell) - \widehat{Q}(x, u; \theta) \right|^2$$

duce la $\widehat{Q}(x, u; \theta) \approx \mathrm{E}\{\dots\}$

- Chiar dacă (x_s, u_s) nu se repetă, CMMP aproximează valoarea aşteptată

Iteraţia Q bazată pe date: Convergenţă

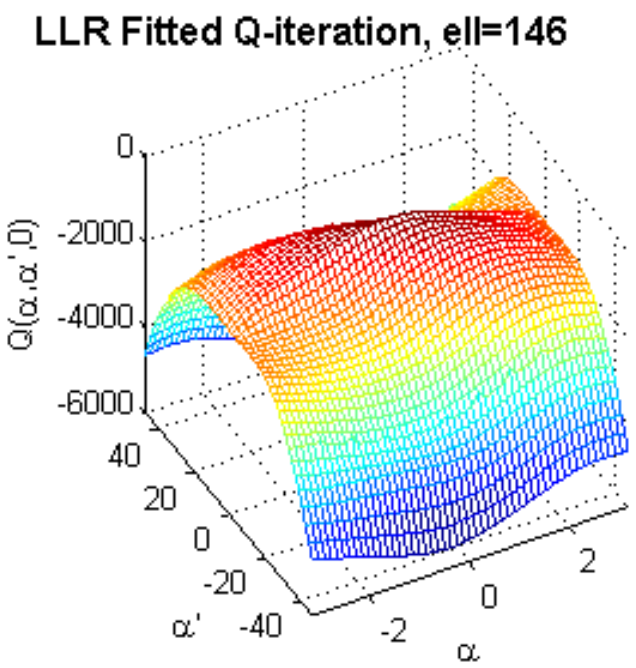
Convergenţă la o **secvenţă** de soluţii,
fiecare din ele **aproape-optimală**

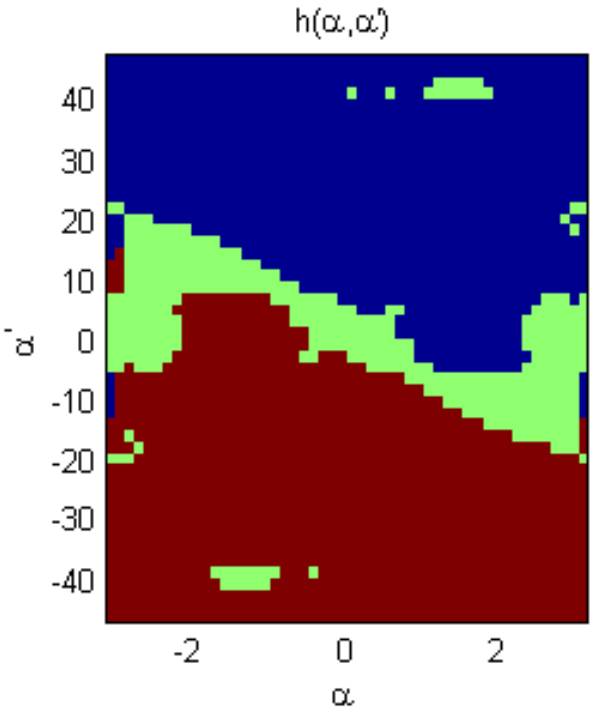


Pendul inversat: Iteraţia Q cu LLR, demo

Discretizare: 3 acţiuni

Baza de date: grilă $31 \times 15 \times 3$; $k = 5$





Terminologie engleză

funcţie de bază	=	basis function
reţea neuronală	=	neural network
regresie liniară locală	=	local linear regression, LLR
iteraţia fuzzy Q	=	fuzzy Q-iteration
iteraţia Q bazată pe date	=	model-free/fitted Q-iteration