REGLAREA UNUI SISTEM DINAMIC NELINIAR CU INCERTITUDINI FOLOSIND REŢELE NEURALE

dr. ing. Ovidiu Grigore
dr. ing. Octavian Grigore

Universitatea Politehnica Bucureşti

Rezumat: Reglarea sistemelor dinamice se face destul de complicat prin metodele clasice, în special, când dinamica sistemului este foarte rapidă. O astfel de problemă devine şi mai complicată atunci când sistemul mai cuprinde şi mărimi necunoscute şi el este şi neliniar. Pentru rezolvarea acestor probleme, propunem proiectarea comenzii cu ajutorul unei reţele neurale. Această metodă are avantajul de a elimina rezolvarea unui sistem diferenţial, de dimensiune apreciabilă.

Cuvinte cheie: neuron, reţea neurală feed forward, controler cu reţea neurală.

1. Introducere

Teoria reţelelor neurale cu un singur strat a fost pusă la punct în anii '80. Prin dezvoltarea tehnicii de calcul (viteză şi capacitate) la sfârşitul anului 1988, primele aplicaţii ale reţelelor neurale cu straturi multiple au fost în domeniul recunoaşterii formelor.

Cum sistemele dinamice constituie un domeniu important, începând cu 1992, cercetătorii au încercat aplicarea reţelelor neurale la identificarea şi reglarea lor.

S-a început aplicarea lor la rezolvarea sistemelor neliniare, ştiindu-se faptul că, prin metodele clasice, reglarea acestor sisteme se face destul de greu. În prezentul articol, se aplică reţelele neurale pentru rezolvarea unui sistem dinamic complex cu incertitudini.

Pentru început, vom da noţiunile de bază, referitoare la reţele neurale, după care vom explicita aplicarea lor pentru determinarea unui regulator robust.

2. Reţea neurală

Prin *neuron* înţelegem o aplicaţie $y_i : \mathbf{R}^N \rightarrow \mathbf{R}$ descrisă de [5]:

$$y_i = \Gamma\left(\sum_{j=1}^{N} w_{ji} x_j + \theta_i\right)$$

unde:

$X^T = (x_1\ x_2\ x_3 \ldots\ x_N)$ este vectorul de intrare;

w_{ji} este ponderea legăturii între unitatea de intrare j şi neuronul curent i;

θ_i este termenul de polarizare a unităţii "i" (a neuronului);

$\Gamma : \mathbf{R} \rightarrow (0\ ,\ 1)$ este o funcţie monotonă crescătoare pe care o alegem de tipul:

$$\Gamma(u) = \frac{1}{1+e^{-u}}$$

Un set de neuroni interconectaţi ca în figura 1, formează o reţea neurală (RN). Reţeaua este organizată în $l = 0 \ldots L$ straturi. Dacă neuronul de pe stratul l primeşte semnal doar de la neuronii de pe stratul $l - 1$ (transferul informaţiei este într-un singur sens), atunci reţeaua se numeşte reţea neurală de tip "*feed forward*".

Cybenko în [1] şi Hornik în [2] au demonstrat că orice aplicaţie continuă pe un interval compact $Y = \Phi(X)$ poate fi aproximată suficient de bine de o RN cu un singur strat ascuns ($l = 3$) dacă dimensiunea acestuia este aleasă corespunzător.

Presupunem că avem un set de P perechi intrare- ieşire $\{(X_1, Y_1) \ldots (X_P, Y_P)\}$, care rezultă dintr-o corespondenţă funcţională $Y = \Phi(X)$ cu $X \in \mathbf{R}^N$ şi $Y \in \mathbf{R}^M$.

" A antrena" o reţea înseamnă ca ea să înveţe o aproximaţie $Y' = \Phi'(X)$ a funcţiei dorite $Y = \Phi(X)$, ceea ce practic înseamnă a găsi un set corespunzător de ponderi în stratul ascuns $\{ w^h_{ij} \}$ şi ai setului de ponderi în stratul de ieşire $\{ w^o_{kj} \}$ (figura 1).

Obţinerea valorilor la ieşirea reţelei neuronale se face prin propagarea informaţiei de la intrare către ieşire. Astfel, fie $X_p = (x_{p1} \ldots x_{pN})^T$ vectorul aplicat pe stratul de intrare.

Unităţile de intrare distribuie valorile către stratul ascuns, astfel intrarea în unitatea ascunsă j va fi:

$$\text{net}\, p^h_j = \sum_{i=1}^{N} w^h_{ji} x_{pi} + \theta^h_j ,$$

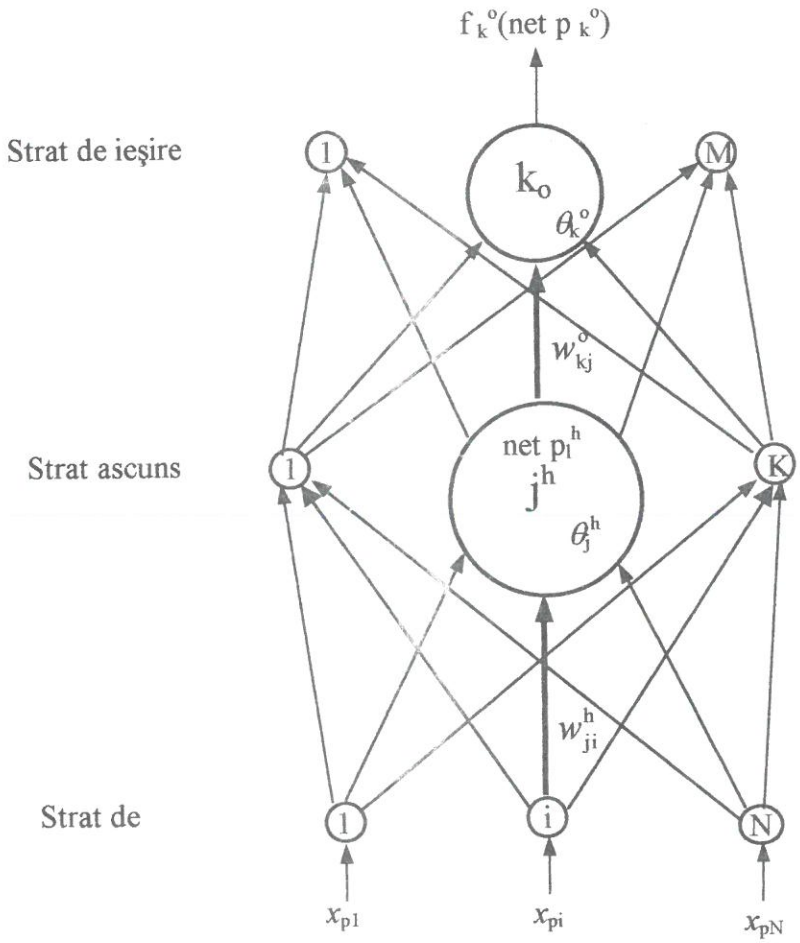

Figura 1. Reţea neurală cu $l = 3$ straturi

unde notaţiile sunt cele de la neuron, iar indicele h arată că este vorba de stratul ascuns.

Ieşirea din acest nod va fi:

$$i_{pj} = f_j^h\left(\text{net } p_j^h\right).$$

În mod corespunzător, ecuaţiile pe stratul de ieşire, vor fi:

$$\text{net } p_k^o = \sum_{j=1}^{N} w_{kj}^o i_{pj} + \theta_k^o, \qquad (1)$$

$$o_{pk} = f_k^o\left(\text{net } p_k^o\right),$$

unde $O_p = (o_{p1} \ldots o_{pM})^T$ este ieşirea reală (curentă) a RN când la intrare este aplicat $X_p = (x_{p1} \ldots x_{pN})^T$.

O RN cu suficiente noduri în stratul ascuns va putea fi antrenată suficient de bine dacă pentru $(\forall)$ $\varepsilon > 0$ (*"legea delta"*, [6]):

$$\|\Phi(X) - O(X)\| < \varepsilon \qquad \text{pentru } (\forall)\, X \in \wp$$

unde: $\wp \in \mathbf{R}^n$ este un compact;

X este intrarea în RN;

$\Phi(X)$ este ieşirea ideală (dorită) a RN;

$O(X)$ este ieşirea reală a RN.

Pentru determinarea ponderilor, se aplică metoda recursivă a gradientului negativ [5] conform căruia:

$$w^{(t+1)} = w^{(t)} - \eta \nabla_w E_p \qquad (2)$$

unde:

$\eta(t)$ este rata de învăţare.

$\nabla_w E_p$ este gradientul erorii E_p în raport cu ponderea calculată w.

Criteriul ce dorim să îl optimizăm este minimizarea erorii pătratice de la ieşirea RN, care este:

$$E_p = \frac{1}{2}\sum_{k=1}^{M} \delta_{pk}^2,$$

unde:

$\delta_{pk} = (y_{pk} - o_{pk})$ este eroarea de ieşire din reţea;

y_{pk} este ieşirea dorită din RN;

o_{pk} este ieşirea actuală din RN.

Particularizând (2) pentru ponderile stratului de ieşire, obţinem:

$$w_{kj}^{o}(t+1) = w_{kj}^{o}(t) - \eta \frac{\partial E_p}{\partial w_{kj}^{o}} \qquad (3)$$

$$E_p = \frac{1}{2}\sum_{k=1}^{M}(y_{pk} - o_{pk})^2 = \frac{1}{2}\sum_{k=1}^{M}\left[y_{pk} - f_k^o(netp_k^o)\right]^2 =$$

$$= \frac{1}{2}\sum_{k=1}^{M}\left[y_{pk} - f_k^o\left(\sum_{j=1}^{K} i_{pj} + \theta_k^o\right)\right]^2$$

$$\frac{\partial E_p}{\partial w_{kj}^{o}} = -(y_{pk} - o_{pk})\frac{\partial f_k^o}{\partial(net\, p_k^o)}\frac{\partial(net\, p_k^o)}{\partial w_{kj}^{o}} \qquad (4)$$

$$\frac{\partial(net\, p_k^o)}{\partial w_{kj}^{o}} = \frac{\partial}{\partial w_{kj}^{o}}\sum_{j=1}^{K}\left(w_{kj}^{o} i_{pj} + \theta_k^o\right) = i_{pj} \qquad (5)$$

Introducând (5) în (4) se obţine:

$$-\frac{\partial E_p}{\partial w_{kj}^{o}} = (y_{pk} - o_{pk})\left[f_k^o(net\, p_k^o)\right]' i_{pj} \qquad (6)$$

Ponderile de pe stratul de ieşire sunt modificate astfel:

$$w_{kj}^{o}(t+1) = w_{kj}^{o}(t) - \Delta_p w_{kj}^{o}(t)$$

unde:

$$\Delta_p w_{kj}^{o}(t) = \eta(y_{pk} - o_{pk})\left[f_k^o(net\, p_k^o)\right]' i_{pj}.$$

Dacă se alege funcţia RN (sau funcţia de activare a neuronului) ca fiind:

$$f_k^o(net\, p_k^o) = \frac{1}{1 + \exp(-net\, p_k^o)}$$

$$\Rightarrow \left[f_k^o(netp_k^o)\right]' = f_k^o(netp_k^o)\left[1 - f_k^o(netp_k^o)\right] = o_{pk}(1 - o_{pk})$$

deci:

$$w_{kj}^{o}(t+1) = w_{kj}^{o}(t) + \eta(y_{pk} - o_{pk})o_{pk}(1 - o_{pk})\, i_{pj}.$$

Dacă se notează cu:

$$\delta_{pk}^{o} = (y_{pk} - o_{pk})o_{pk}(1 - o_{pk})$$

obţinem relaţia de actualizare a ponderilor pe stratul de ieşire ca fiind:

$$w_{kj}^{o}(t+1) = w_{kj}^{o}(t) + \eta\, \delta_{pk}^{o}\, i_{pj}.$$

Aplicând acum procedeul de actualizare şi pe stratul ascuns se obţine:

$$w_{kj}^{o}(t+1) = w_{kj}^{o}(t) + \eta i_{pj}(1 - i_{pj}) x_{pi}(y_{pk} - o_{pk}) o_{pk}(1 - o_{pk}) w_{kj}^{o}(t).$$

Acest mod de antrenare al reţelei, prin reactualizare a ponderilor funcţie de eroarea la ieşire, se numeşte "*antrenare supervizată*".

Algoritmul de antrenare a RN este:

- presupunem cunoscut un lot de perechi intrare- ieşire $\{(X_1, Y_1) \ldots (X_P, Y_P)\}$;
- pentru $p = 1 \ldots P$
  - se aplică $X_p = (x_{p1} \ldots x_{pN})^T$;
  - se calculează ieşirea curentă a RN: $o_{pk} = f(net\, p_k^o)$ prin propagarea informaţiei de la intrare la ieşire;
  - se fac corecţiile pentru w_{kj}^{h} şi w_{kj}^{o} ;
  - se calculează $E_p = \frac{1}{2}\sum_{k=1}^{M}(y_{pk} - o_{pk})^2$;
- eroarea pătratică totală, pentru tot lotul de învăţare, este: $E = \sum_p E_p$
- testul care arată dacă RN a fost suficient de bine antrenată este: $E < \varepsilon$

Dacă acest test nu este satisfăcut, se reia procesul de antrenare a RN.

3. Regulator folosind reţea neurală

Proiectarea unui regulator înseamnă a determina valorile comenzii curente, în funcţie de valorile de intrare, de stare şi a comenzii anterioare. De aceea, în cazul folosirii unei RN drept regulator ieşirea reţelei trebuie să fie comanda la momentul curent, iar intrările sunt reprezentate de stări, intrări si comanda anterioară (vezi figura 2).

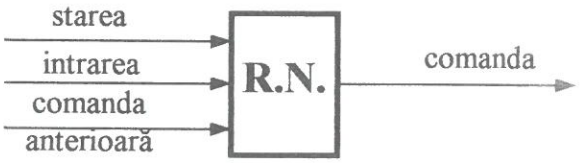

Figura 2. Regulator cu RN

Deoarece RN folosită este de tip *supervizat "cu învăţare"*, s-a pus problema determinării unui set cât mai reprezentativ de perechi intrare - ieşire cu ajutorul căruia să fie învăţată RN. Pentru aceasta, un rol important îl ocupă clasificarea apriorică a

formelor de intrare ("input patterns") într-un număr limitat de clase distincte. Presupunem (vezi figura 1) intrările $x_1 \ldots x_N$ numite şi caracteristici ("features"), reprezentate de un vector $\boldsymbol{X}$ de dimensiune N în spaţiul caracteristicilor Ω_X [2], iar $\omega_1 \ldots \omega_R$ cele R clase corespunzătoare în cazul nostru celor R situaţii de reglaj. O astfel de operaţie se mai poate interpreta ca o partiţionare a spaţiului caracteristicilor k dimensional Ω_X în R regiuni de decizie corespunzătoare celor R clase de reglaj.

În esenţă, algoritmul folosit pentru determinarea celor R clase de comandă [7], constă în gruparea vectorilor de intrare pe baza minimului distanţei euclidiene dintre vectori şi mediile claselor. Astfel, pentru fiecare valoare de intrare se calculează distanţa faţă de centrele claselor existente. Se alege drept clasă de apartenenţă pentru vectorul respectiv, aceea pentru care distanţa este minimă şi, în plus, mai mică decât o valoare de prag aleasă apriori (care caracterizează domeniul clasei). Altfel, el este considerat ca o nouă situaţie de reglaj şi se creează o nouă clasă având în centru pe X, iar mărimea este dată de raza D.

Putem spune că toţi vectorii care aparţin unei clase (deci, distanţa între vectori este mai mică decât D) au aproximativ aceleaşi caracteristici şi, deci, putem aproxima comenzile lor cu comanda corespunzătoare centrului clasei de intrare.

4. Aplicaţie

Fie sistemul de urmărire [3]:

$$(\Sigma_R):\ \begin{cases} \dot{\theta}_R = 5{,}23\alpha_R \\ \dot{\alpha}_R = p_R \\ \dot{p}_R = -5{,}88p_R - 728{,}43\alpha_R - 529.5\delta_R \\ \dot{\Phi} = \dfrac{V_R - V_T \Xi[\Phi - \theta_T]}{R}\Phi - \dfrac{V_R}{R}\theta_R + \dfrac{V_T}{R}\Xi[\Phi - \theta_T]\theta_T \end{cases}$$

$$(\Sigma_T):\ \begin{cases} \dot{\theta}_T = 5{,}23\,\alpha_T \\ \dot{\alpha}_T = p_T \\ \dot{p}_T = -2{,}505\ p_T - 572\ ,5\alpha_T - 300\ .5\delta_T \end{cases}$$

unde:

- (Σ_R) reprezintă ecuaţiile dinamice ale modelului rachetei în planul vertical, iar (Σ_T), ale ţintei;
- θ – unghiul de tangaj al rachetei, respectiv ţintei;
- α – unghiul de incidenţă al rachetei, respectiv ţintei;
- p – viteza de variaţie a unghiul de incidenţă al rachetei, respectiv ţintei;
- Φ – unghiul de vizare al ţintei;
- R – distanţa dintre rachetă şi ţintă;
- V – viteza rachetei, respectiv ţintei;
- δ – comanda rachetei, respectiv ţintei.

Pentru sistemul compus rachetă – ţintă (Σ_R – Σ_T) construim un regulator bazat pe reţea neurală.

Vom considera ca intrări ale reţelei neurale mărimile R, Φ şi θ (care sunt dependente de comanda anterioară), deci RN are 3 noduri de intrare.

În stratul ascuns, s-au ales 7 neuroni, obţinându-se rezultate bune. Alegând un număr mai mare de neuroni în stratul ascuns, rezultatele experimentale se îmbunătăţesc vizibil (eroarea pătratică de învăţare scade), dar cantitatea calculelor şi, respectiv, timpul necesar învăţării cresc considerabil.

Ca ieşire a RN, considerăm comanda δ a rachetei cu valori cuprinse între –0.56rad şi 0.56rad (rezultate din condiţia de suprasarcină admisibilă permisă – n_{adm} – care, de obicei, este cuprinsă între –50g şi +50g). Deoarece RN se antrenează mai greu pentru ieşiri ce variază continuu în timp, vom partiţiona domeniul comenzii în 11 domenii iniţiale astfel încât fiecărui subdomeniu să îi corespundă un neuron de ieşire al reţelei. RN are deci 11 neuroni la ieşire.

Astfel s-au determinat perechi particulare de valori intrare-ieşire prin rezolvarea sistemului de ecuaţii diferenţiale rachetă – ţintă (Σ_R – Σ_T) într-un număr de 20.000 de puncte. Aceste valori particulare se pot obţine şi experimental printr-o simulare reală a scenariului de urmărire, însă necesită cheltuieli considerabile şi un timp destul de mare. Datorită cantităţii mari de date ce trebuie prelucrate, timpul de învăţare al reţelei este foarte mare, de aceea, am redus numărul de vectori intrare-ieşire prin metoda "clustering Isodata", în final obţinându-se un set de 470 vectori (şi, deci, implicit, a crescut viteza de convergenţă a procesului de învăţare).

Rata de învăţare $\eta(t)$, factor folosit în metoda gradientului negativ, poate lua valori între 0 şi 1. Dacă alegem pe η mare (cu valori apropiate de 1), se poate ca metoda gradientului să nu fie convergentă. Dacă η este prea mic algoritmul converge lent şi se poate bloca pe minime locale, nu pe minimul absolut.

5. Concluzii şi rezultate

În figura 3 şi tabelul 1 sunt prezentate rezultatele simulărilor atât pentru un regulator robust, cât şi pentru regulatorul cu reţea neurală.

Ca parametri iniţiali, pentru iniţializarea procesului de urmărire s-au ales: R şi Φ. Manevrele ţintei, obţinute prin comanda ţintei δ_T,

considerate în acest caz perturbaţii, sunt mărginite şi pot fi aplicate de utilizator.

Din punct de vedere al parametrului dinamic calitativ – distanţa de ratare D – racheta cu regulator robust fiind proiectată să îl optimizeze, pe traiectoria finală, aceasta oscilează în jurul valorii zero. În cazul rachetei cu reţea neurală, acest parametru scade în timpul apropierii de ţintă, dar, în momentul lovirii, are valori de zeci de metri. Dezavantajul în acest caz poate fi scăparea ţintei, dacă ea are o dinamică mai rapidă decât a rachetei (fapt rar întâlnit însă). El poate fi contracarat prin mai multe date despre urmărirea pe faza finală a traiectoriei, introducându-se mai multe poziţii de tragere a rachetei pentru o distanţă R ca valori iniţiale pentru învăţarea reţelei neurale.

Din aceste rezultate comparative şi metode grafice, putem trage concluzia că a construi un regulator cu reţea neurală pentru a comanda o rachetă ce urmăreşte o ţintă cu dinamică asemănătoare ei, este un fapt viabil. Practic, pilotul automat, în acest caz, este format dintr-o reţea neurală (un cip dedicat), care înlocuieşte întregul calculator de bord din cazul rachetei cu regulator robust. În acest caz, datorită compactităţii şi a simplităţii constructive (din punct de vedere electronic), acesta este mult mai fiabil şi rezistent, decât cel cu regulator, la toate genurile de bruiaje folosite de eventualii inamici (bruiaj electronic, de exemplu).

De asemenea, metoda dovedind viabilitate în reglarea unor sisteme rapide şi complexe, cum este cazul scenariului de urmărire, realizat în această lucrare, ea poate fi extinsă şi la alte sisteme dinamice mai simple cu rezultate imediate (de exemplu, reglarea şi controlarea vitezei motoarelor pas cu pas, servomotoarelor etc.).

Tabelul 1 Comparaţia între regulatorul clasic şi cel cu reţea neurală

Distanţa iniţială rachetă-ţintă [m]	Unghiul de vizare iniţial [°]	Timp [s]	
		Regulator clasic	Regulator cu reţea
1000	-15	3.36	3.15
1500	25	4.89	4.65
2500	45	7.77	7.45
2500	-50	7.23	6.6
3000	-20	9.78	9.69

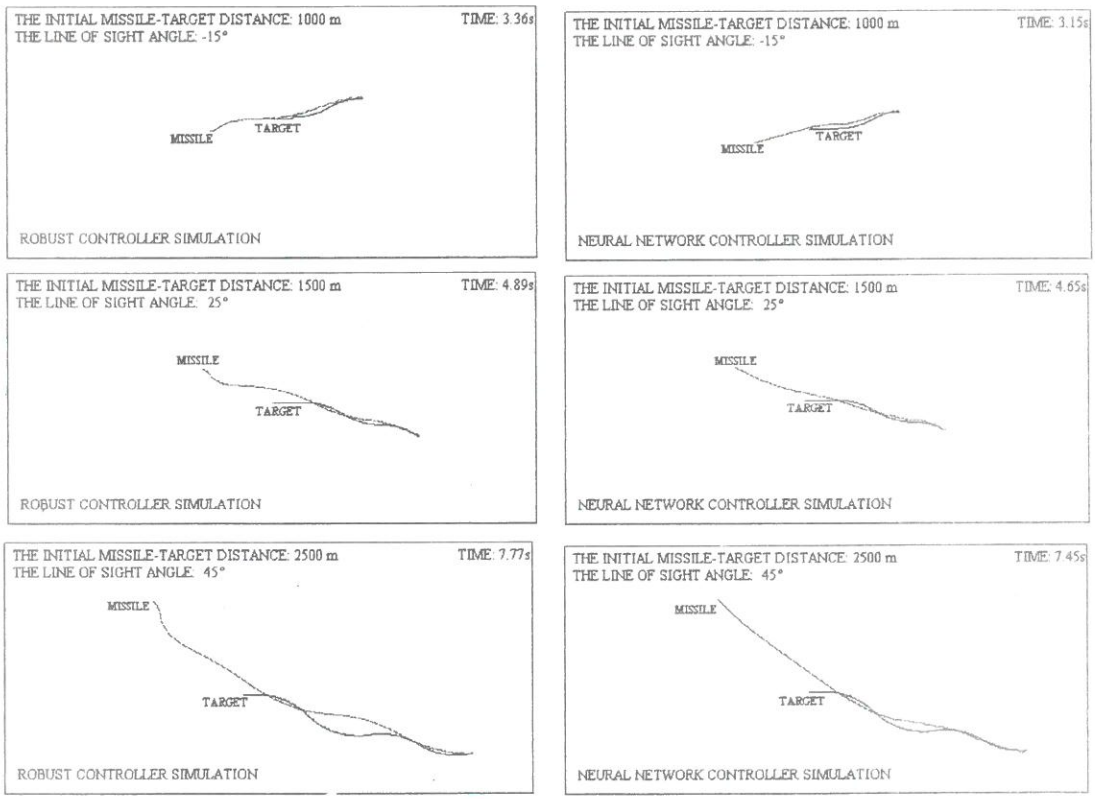

Figura 3. Simulările traiectoriilor în cazul regulatorului robust şi cel cu reţea neurală

Bibliografie

1. **CYBENKO, G.**: Approximation by superposition of sigmoidal function. În: Math. Contr. Sign. and Systems, vol.2, no.4, , 1989, pp. 303-314.
2. **FU, KING-SU**: Learning Control System - Review and Outlook. În: Trans. on Pattern Analysis and Machine Intellig., vol. PAMI-8, no.3, 1986, pp. 327-343.
3. **GRIGORE, O.**: Stabilitatea şi Controlul Sistemelor Dinamice cu Incertitudini, Teză de Doctorat, 1987.
4. **HORNIK, K., STINCHCOMB, M.**: Multilayer feedforward networks are universal approximators. În: I.E.E.E. Neural Networks, vol.2, 1989, pp. 359-366.
5. **LEVIN, A.V., NARENDRA K.S.**: Control of Nonlinear Dynamical Systems using Neural Networks: Controllability and Stabilization. În: I.E.E.E. Trans. Neural Networks, vol. IV, no.2, 1993, pp. 194-195.
6. **SADEGH, N.**: A Perceptron Network for Functional Identification and Control of Nonlinear System. În: I.E.E.E. Neural Networks, vol.4, no.6, 1993, pp.982-988.
7. **TOU, J.T., GONZALES, R.C.**: Pattern Recognition Principles, Addison-Wesley Publishing Company-1974, p.97.