Identificarea Sistemelor

Ingineria Sistemelor, anul 3
Universitatea Tehnică din Cluj-Napoca

Proiect

1/1

Kovács Attila-Levente
Ciordaș Dragoș-Florin

Partea II

ARX Neliniar

Conținut

Introducere

- Se dă: set de date măsurat (date de identificare) de la un sistem dinamic SISO cu ordinul maxim 3, dinamică posibil neliniară. Ieșirea poate fi afectată de zgomot. De asemenea, sunt furnizate și datele de validare pentru a putea valida modelul dezvoltat.
- Cele două seturi de date sunt furnizate într-un fișier MATLAB,de tip iddata, cu variabilele `id` și `val.`
- Se va dezvolta un model **parametric** (polinomial) de tip **cutie neagră,** folosind un model NARX de tip polinomial.
- Va fi prezentată structura NARX, urmat de modelul dezvoltat pe aceste date.

Modelul (Metoda) NARX

- Ieșirea y(k) este calculată din intrările și ieșirile la pași precedenți, cu ajutorul unei funcții neliniare (polinom neliniar) **g(d(k),θ)**
- **Legătură** cu partea I. a proiectului: aproximator polinomial.
- Deci: $\boldsymbol{y}(\boldsymbol{k}) = \boldsymbol{g}(\boldsymbol{d}(\boldsymbol{k}), \boldsymbol{\theta}) + \boldsymbol{e}(\boldsymbol{k})$**,** unde:
  - **g** - un polinom de gradul m în ieșirile și intrările precedente
  - **d(k)** – vectorul de ieșiri și intrări precedente
  - $\boldsymbol{\theta}$ – parametrii polinomului g -> **sistem dinamic neliniar**
  - **e(k)** – zgomot (eroare de predicție $\varepsilon(k)$), de obicei zgomot alb Gaussian, independent de masurari
- Model Neliniar (regresorii conțin neliniarități) AutoRegresiv (y(k) depinde de valorile lui la eșantioane precedente) cu intrare eXogenă (dependentă de u)
- În cazul de față, modelul va fi estimat cu metoda offline pentru a elimina posibilitatea erorilor la estimările de parametri online în timpul unei control real-time.

Clasificarea modelului

Model neliniar DAR Model liniar în parametri

- Nu este valabil principiul suprapunerii efectelor – ci doar pentru modele (sisteme) liniare și se referă la relația dintre variabilele dependente de timp. Descrie procese nestaționare, dinamice.
- Generalizează la orice dependență neliniară
- Modelul se poate liniariza local la un model ARX liniar
- ARX: caz particular al NARX (cazul liniar) (m=1 și termen liber = 0)
- Dacă termenul liber nu este 0, modelul se numește **afin.**

- Un sistem poate fi neliniar dpdv. dinamic și totuși liniar (liniarizabil) în parametri – așa este și NARX
- Model parametric – pornește de la o presupusă descriere matematică a dinamicii procesului cu parametri (coeficienți) - ex.: ecuații diferențiale, funcții de transfer
- Parametrii pot fi găsiți cu ajutorul **regresiei liniare**! (prin rezolvarea ecuațiilor diferențiale)
- Aplicarea metodei Gauss-Newton nu ar fi util – ia prea mult timp computațional și nu este sigur că ajunge la un nivel satisfăcător de precizie.

Metoda NARX (continuare)

- Coeficienții polinomului vor fi stocate în, θ, cărui dimensiuni depind de na+nb și m.
- Regresorii vor fi stocate în matricea $D \in \mathbb{R}^{N\,x\,(na+nb)}$ care are pe fiecare linie câte un vector d, discutat anterior (conține ieșiri și intrări precedente). Aceasta se realizează la fel ca în cazul ARX liniar, doar că vectorii d conțin termeni neliniari. Așadar, avem regresori neliniari.
- **na** și **nb** sunt ordinele polinoamelor A și B din funcția de transfer în z (timp discret) (sau forma explicită)

$$H(z) = \frac{B(q^{-1})}{A(q^{-1})} z^{-nk}$$, **nk** – timpul mort

- Pentru a găsi parametrii, ele se pot separa între ele: parametri liniari și neliniari (liniarizabili) – se poate rezolva în acest caz cu funcția `lsqnonlin` din `MATLAB.`

Problema de identificare

- Problema de identificare constă în modul de a afla parametrii ai modelului dintr-un set de date u(k), y(k).
- Astfel, pentru orice k:

$$y(k) = \varphi^T(k) \cdot \theta + \varepsilon(k)$$

- Pentru seturi mari de date, această formă este impractică pentru a afla coeficienții. S-a menționat în partea de regresie liniară (curs) o formă alternativă:

$$\hat{\theta} = \left[\sum_{k=1}^{N} \varphi(k) \cdot \varphi^T(k)\right]^{-1} \cdot \left[\sum_{k=1}^{N} \varphi(k) \cdot y(k)\right]$$

- Problemă: suma celor N termeni poate fi mare, astfel pentru a evita problemele numerice, normalizăm valorii prin împărțirea fiecărui element cu N.
- Notă: Am încercat această metodă pentru găsirea coeficienților, dar nu a funcționat bine, deși avem set mare de date (500+ eșantioane).

Problema de identificare

- Obținem:

$$\hat{\theta} = \left[\frac{1}{N}\sum_{k=1}^{N}\varphi(k)\cdot\varphi^T(k)\right]^{-1}\cdot\left[\frac{1}{N}\sum_{k=1}^{N}\varphi(k)\cdot y(k)\right]$$

- Împărțirea sumei la N se poate efectua recursiv, fără să fie implicate numere mari. Noi vom face la fiecare pas această împărțire, astfel evităm numerele mari.
- **Notă:** În cazul nostru de identificare, va fi utilizată această metodă fiindcă avem setul de date de identificare foarte mare ($N \cong 7500$).

Obiectivul problemei de identificare

- Minimizarea erorii medii pătratice (MSE)
- Deci, funcția obiectiv este: $V(\theta) = \frac{1}{N}\sum_{k=1}^{N} \varepsilon(k)^2$
- Eroarea la momentul k se poate exprima ca diferența între ieșirea reală și ieșirea prezisă (simulată)
- Deci, obiectivul principal este să găsim acei parametri θ pentru care funcția obiectiv este minimă. (ca și la orice altă metodă de identificare)
- În cod, vom încerca să găsim paramatrii pentru care eroarea este minimă.

Utilizarea modelului

- Ca și ARX, modelul poate fi utilizat în:
  - **Predicție cu un pas înainte:** ieșirea reală a sistemului este cunoscută, astfel putem construi vectorul d(k) din semnalele corecte și putem estima cu precizie mai mare ieșirea:

$$d(k) = [y(k-1) \dots y(k-na)\; u(k-1) \;\dots\; u(k-nb)]^T$$

  - **Simulare:** ieșirea reală nu este disponibilă. Suntem nevoiți să folosim ieșirile prezise anterior pentru a popula vectorul d(k) (momentan doar o estimare / aproximare a sa):

$$\hat{d}(k) = [\hat{y}(k-1) \dots \hat{y}(k-na)\; u(k-1) \;\dots\; u(k-nb)]^T$$

- Ieșirea la orice moment k rămâne egală cu polinomul g.

Exemple – structura aproximatorului

- Pentru ordinele: na=nb=2, nk=0 iar gradul m=1, ieșirea aproximată are forma:

  $y(k) = g(d(k), \theta) = a_1 y(k-1) + a_2 y(k-2) + b_1 y(k-1) + b_2 y(k-2) + c + e(k)$; (forma ARX liniar)

  $\theta = [a_1, a_2, b_1, b_2, c]^T$ sunt parametrii reali aproximatorului

  $d(k) = [y(k − 1), . . . , y(k − na), u(k − 1), . . . , u(k − nb)]^T$ este vectorul de intrări și ieșiri precedente.

- Pentru ordinele na=nb=1 și m=2, ieșirea aproximată va conține neliniarități, de grad maxim 2:

  $y(k) = \theta_1 y(k-1) + \theta_2 u(k-1) + \theta_3 y(k-1) u(k-1) + \theta_4 y(k-1)^2 + \theta_5 u(k-1)^2 + \theta_6$

Explicarea codului realizat

- În realizarea codului au fost utilizate funcții locale în script.
- Ideea de bază: construirea polinoamelor neliniare și rezolvarea sistemului cu regresie liniară.
- Pentru asta ne trebuie toate combinațiile de puteri posibile pentru un anumit m, na, nb. Funcția `combinare_unica(na,nb,m)` generează toate combinațiile de puteri cu suma puterilor mai mică decât m. Aceasta utilizează funcții MATLAB precum: `nchoosek,unique,sum.`
- Generarea vectorului d(k) se realizează la fel cu ARX liniar. Ieșirile anterioare nu trebuie puse cu semnul inversat. Pentru acest scop este utilizată de mai multe ori funcția `generare_PHI`, preluat de la tema de laborator nr.6.

Explicarea codului realizat (continuare)

- Pentru a afla coeficienții polinomului neliniar, va trebui să construim o matrice din aceste vectori d(k). Sunt N vectori d(k), astfel această matrice va avea N linii și nr. De coloane depind de na+nb și m.(nu am reușit să derivăm o formulă concretă, doar pe cazul na=nb=1) N este lungimea setului de date.
- Este evident că vectorii cu indici negativi sau zero vor fi considerați zero.
- Pentru a putea afla coeficienții ulterior, se creează pe fiecare linie combinația cu intrări și ieșiri anterioare cu puterile corespunzătoare.
- De exemplu,pentru na=nb=1 și m=2, linia k din această matrice are forma:
$$phi(k) = [1, u(k-1), y(k-1), u(k-1)\cdot y(k-1)\,, u(k-1)^2, y(k-1)^2]$$
- Acest lucru se realizează cu ajutorul funcției `phi_narx(vector_puteri,d_k,N)`.
- Fiecare linie $phi(k)$ din matrice se realizează prin bucle for și secvența de cod: `PHI(i,k) = PHI(i,k)*d_k(i,j)^vector_puteri(k,j);` - se înțelege mai bine în contextul codului

Explicarea codului realizat (continuare)

- Deci, cu ajutorul acestor funcții vom putea găsi coeficienții pe baza datelor de identificare, cu ajutorul regresiei liniare.
- Funcția `generare_PHI` este utilă și pentru utilizarea modelului în predicție și simulare. Pur și simplu se schimbă parametrul `data` corespunzător.
- Restul se rezolvă la fel ca și în cazul ARX liniar.

Rezultate de reglare (tabele)

Date de identificare

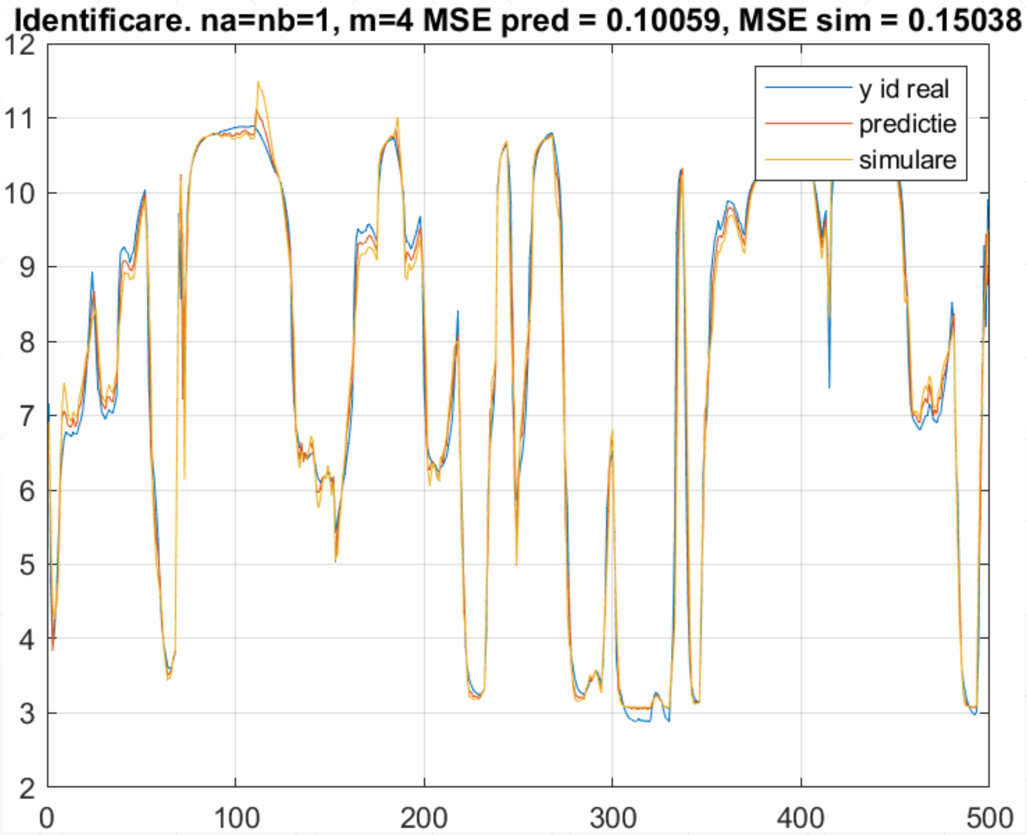
na=nb	m	MSE predicție	MSE simulare
1	1	0.38252	0.62531
1	2	0.23454	0.33966
1	3	0.13817	0.19717
1	4	0.10059	0.15038
1	5	0.07872	0.11891
2	1	0.32346	0.59402
2	2	0.20032	0.31973
2	3	0.07966	0.36996
2	4	0.03972	0.34852
2	5	0.01281	3.64823
3	1	0.13817	0.19717
3	2	0.18890	0.34132
3	3	0.03429	0.65224
3	4	0.00364	1.77875
3	5	0.000006	4.82952

Date de validare

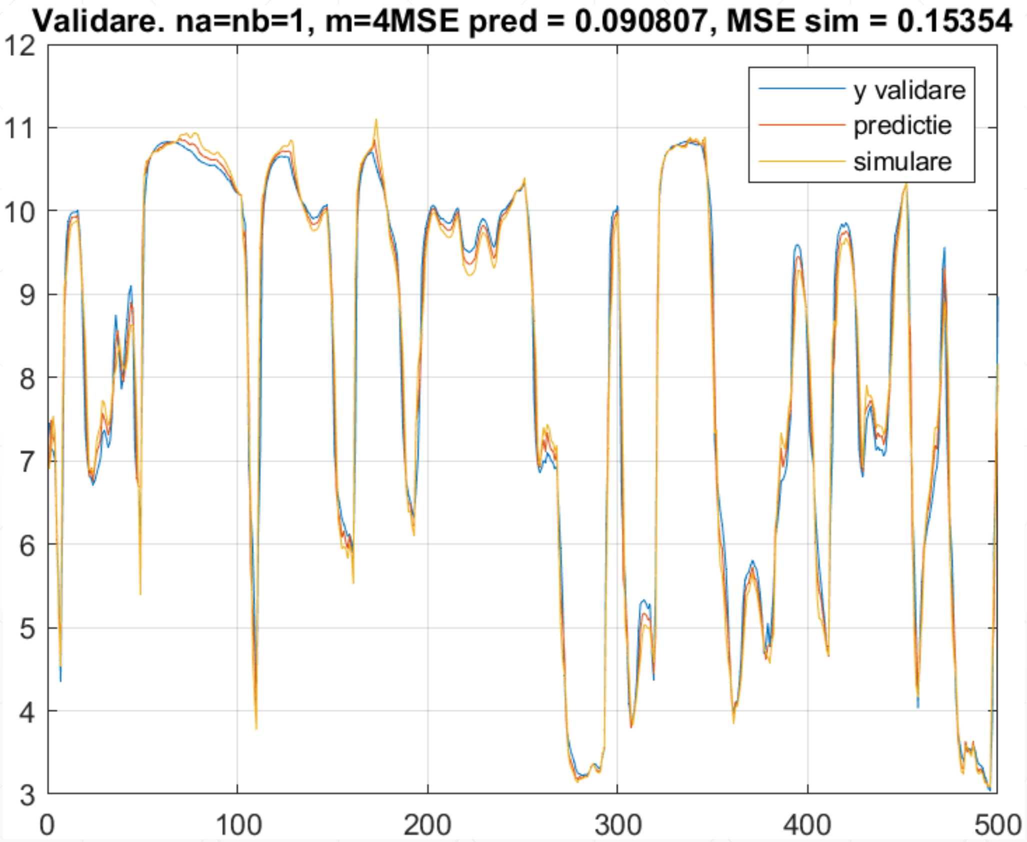
na=nb	m	MSE predicție	MSE simulare
1	1	0.27721	0.59348
1	2	0.19936	0.34035
1	3	0.12094	0.21711
1	4	0.09080	0.15354
1	5	0.14792	0.21588
2	1	0.26161	0.55418
2	2	0.18821	0.32554
2	3	0.58881	1128.48
2	4	936	1.02e19 (code err?)
2	5	1.47*10^3	10^36
3	1	0.25976	0.55949
3	2	0.19508	0.33842
3	3	1.55271	67 312
3	4	27424.31	1.73e25
3	5	93.954e7	1.04e59

Cel mai bun model obținut

Date de identificare

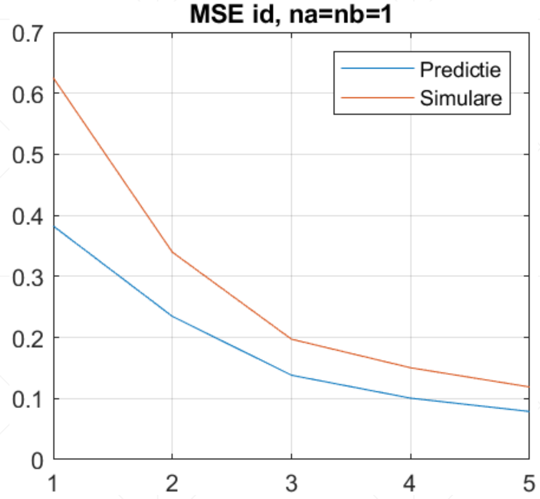


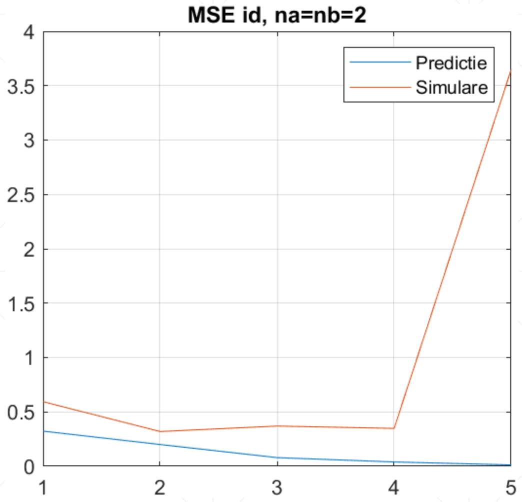
Date de validare

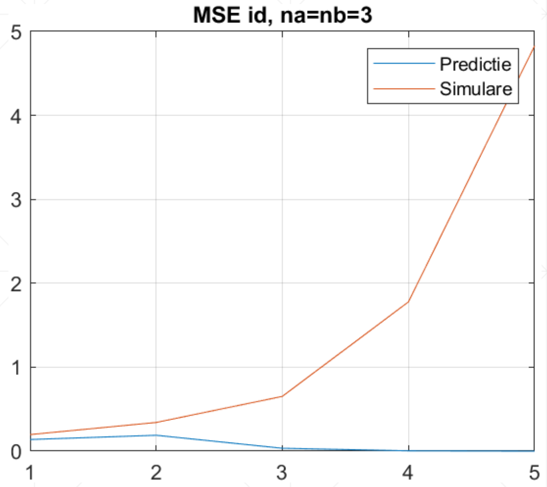


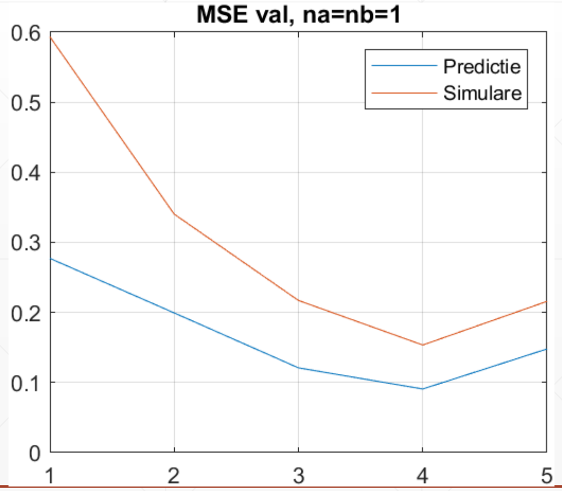
Rezultate de reglare (grafice)

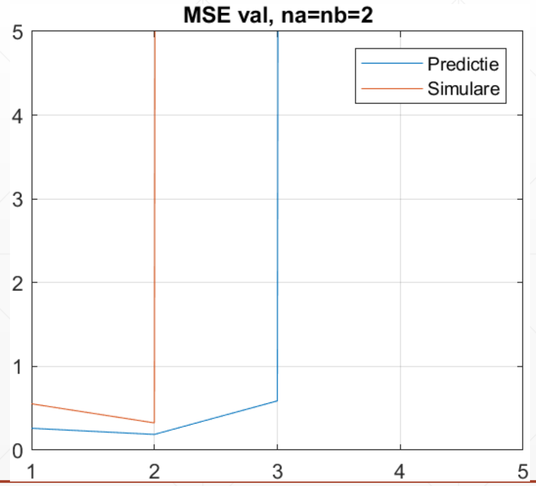
- Afisat separat pentru fiecare na=nb, pe date de id vs. val

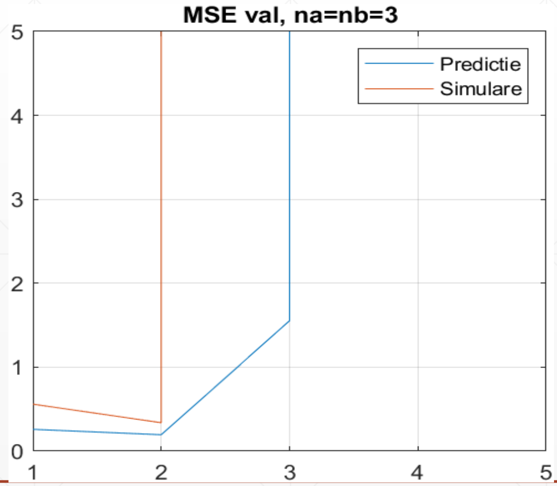












Concluzii

- De pe graficele cu evoluția erorii pe datele de validare, se poate vedea valoarea optimă pentru na=nb=1 și m=4. Acest fapt era marcat și în tabel.
- Se poate observa că sistemul are ordinul 1, fiindcă modelul cu cea mai bună calitate se obține atunci când na=nb=1. Sistemul poate fi aproximat destul de bine cu modelul NARX de ordin m=4.
- Reiese și fenomenul de supraantrenare ca și în cazul aproximatorului polinomial. Eroarea scade pe predicție tot mai mult pe datele de antrenare odată cu creșterea ordinul modelului. Dar între timp, pe datele de validare, eroarea devine tot mai mare. Cum mărim ordinul sistemului, modelul va fi mai particular pentru acele date (modelează și zgomotul). Acest lucru era explicat în Partea I.
- Calitatea simulării este mai slabă față de predicție, fiindcă simularea se face din ieșirea prezisă, care are deja o anumită abatere față de ieșirea reală.

Notă

- Codul creat este disponibil în forma unei anexe.
- Pe alte date de antrenare și validare (alt sistem) am obținut rezultate mult mai calitative (erori de ordinul 10^{-5}).
- Este posibil ca în cod să mai adăugăm ajustări.
- Pe rezultatele de reglare cazul ARX liniar (impunem termenul liber = 0) a fost marcat cu roșu.

Bibliografie

- Ideile din această prezentare au fost preluate din următoarele surse:

  - Lucian Bușoniu: Identificarea Sistemelor, Curs Anul 3 Automatică, UTCN
  - Torsten Söderström, Petre Stoica: System Identification, Prentice Hall International, http://user.it.uu.se/~ts/sysidbook.pdf
  - https://busoniu.net/teaching/sysid2021/index_ro.html
  - H. Peng et al., RBF-ARX model-based nonlinear system modeling and predictive control with application to a NOx decomposition process, Control Engineering Practice 12, paginile 191–203, 2007

Mulțumiri autorilor!