Identificarea Sistemelor

ARX neliniar

Grupa: 30133
Indicii grupului: 8/8
Studenți: Alinei-Poiana Tudor, Ciurezu Gheorghe-Dragos, Kocis Bianca Laura

1 INTRODUCERE

2 METODE DE IMPLEMENTARE

3 REZULTATE DE REGLARE

4 CONCLUZII

INTRODUCERE 1

1.1. Descrierea problemei

Problema abordată în această prezentare constă în identificarea unui model ARX neliniar de tip polinomial pornind de la un set de date de identificare de tip intrare-ieșire.

Modelul obținut îl vom valida pe un al doilea set de date de la același sistem, pentru a asigura corectitudinea modelului dezvoltat.

1.2. Modelul NARX

(Modelul Neliniar AutoRegresiv cu intrare eXogenă)

generalizează la orice dependență neliniară | *y(k) depinde de valorile y precedente* | *dependent de u*

Modelul NARX presupune aflarea unei ieșiri viitoare în funcție de intrările și ieșirile anterioare, cât și de combinațiile neliniare dintre ele.

Notăm: -ordinele: **na, nb**
-întârzierea: **nk**
-grad: **m**

1.2.1. Structura aproximatorului

$$\hat{y}(k)=p(y(k-1),\dots,y(k-na),u(k-nk),u(k-nk-1),\dots,u(k-nk-nb+1);\theta)$$

$$\hat{y}(k)=p\big(d(k)\big)$$

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

unde:

- $\hat{y}(k)$ este aproximarea sistemului
- $d(k)$ este vectorul de ieșiri și intrari întarziate
- p este un polinom de grad m (configurabil) format din variabilele lui $d(k)$

1.2.1. Structura aproximatorului

EXEMPLE:

Pentru na=nb=1, nk=1 : $d(k) = [y(k-1)\ ,\ u(k-1)]$

Pentru simplitate, notăm $y(k-1) = x_1$ $și$ $u(k-1) = x_2$

-pentru m=1 avem: $[1,\ x_1, x_2]$

-pentru m=2 avem: $[1, x_1,\ x_2\ , x_1^2\ , x_2^2,\ x_1 \cdot x_2]$.

Pentru na=nb=2, nk=1 : $d(k) = [y(k-1)\ , y(k-2)\ , u(k-1), u(k-2)]$

Pentru simplitate, notăm $y(k-1) = x_1\ ,\ y(k-2) = x_2,\ u(k-1) = x_3$ $și$ $u(k-2) = x_4$

-pentru m=1 avem: $[1,\ x_1, x_2, x_3, x_4]$

-pentru m=2 avem: $[1, x_1, x_2, x_3, x_4, x_1^2, x_2^2, x_3^2, x_4^2, x_1 \cdot x_2, x_1 \cdot x_3, x_1 \cdot x_4, x_2 \cdot x_3,\ x_2 \cdot x_4,\ x_3 \cdot x_4]$

1.2.2. Structura matricei de regresori

Se va căuta un polinom de forma:

$$P = \sum_i \theta_i \cdot (x_1^{p_{i1}} \cdot x_2^{p_{i2}} \cdot \ldots \cdot x_n^{p_{in}})$$

De exemplu, prima linie din matricea de regresori neliniari va conține elemente de forma:

$X_{11}^{P_1} \cdot X_{12}^{P_2} \cdot \ldots \cdot X_{1(na+nb)}^{P_{na+nb}}$, unde $P_1 + P_2 + \cdots + P_{na+nb} \leq m$, iar $P_1, P_2, \ldots, P_{na+nb} \in \{0,1,\ldots,m\}$.

$$Regresori_{NARX} = \begin{bmatrix} X_{11}^0 \cdot X_{12}^0 \cdot \ldots \cdot X_{1(na+nb)}^0 & X_{11}^1 \cdot X_{12}^0 \cdot \ldots \cdot X_{1(na+nb)}^0 & \cdots & X_{11}^0 \cdot X_{12}^0 \cdot \ldots \cdot X_{1(na+nb)}^m \\ X_{21}^0 \cdot X_{22}^0 \cdot \ldots \cdot X_{2(na+nb)}^0 & X_{21}^1 \cdot X_{22}^0 \cdot \ldots \cdot X_{2(na+nb)}^0 & \cdots & X_{21}^0 \cdot X_{22}^0 \cdot \ldots \cdot X_{2(na+nb)}^m \\ \vdots & \vdots & \ddots & \vdots \\ X_{N1}^0 \cdot X_{N2}^0 \cdot \ldots \cdot X_{N(na+nb)}^0 & X_{N1}^1 \cdot X_{N2}^0 \cdot \ldots \cdot X_{N(na+nb)}^0 & \cdots & X_{N1}^0 \cdot X_{N2}^0 \cdot \ldots \cdot X_{N(na+nb)}^m \end{bmatrix}$$

1.3. Utilizarea modelului

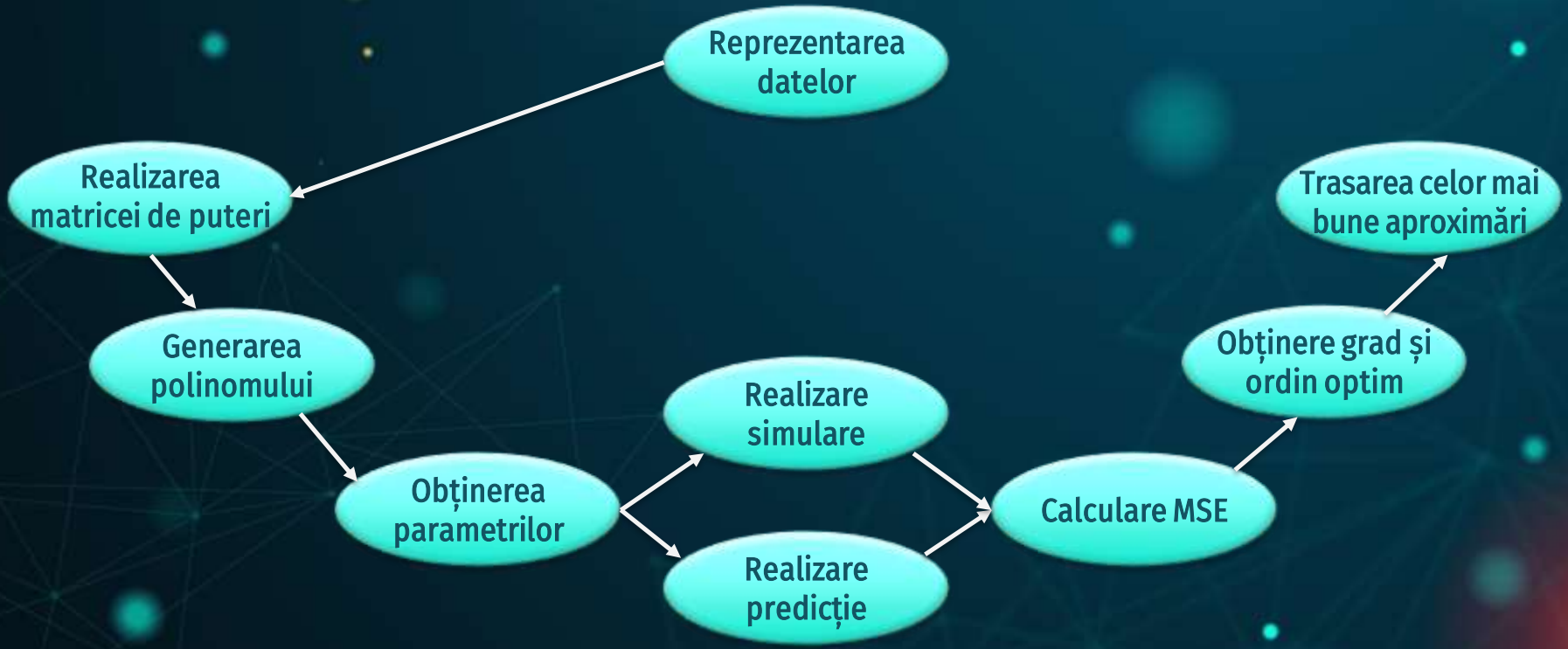
Predicția reprezintă etapa în care știm cum arată ieșirea și intrarea și utilizăm datele respective până la un moment k pentru a anticipa cum va arăta ieșirea la urmatorul pas, k+1.

PREDICȚIA

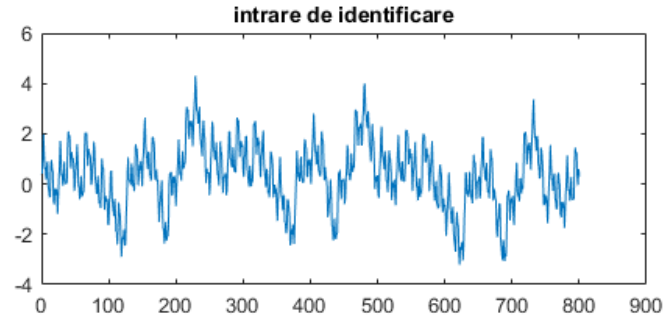
Simularea reprezintă etapa în care construim de la 0 ieșirea, cunoscând doar intrarea și folosind valorile ieșirii simulate anterior (se vor considera nule toate valorile până în momentul 0, inclusiv).

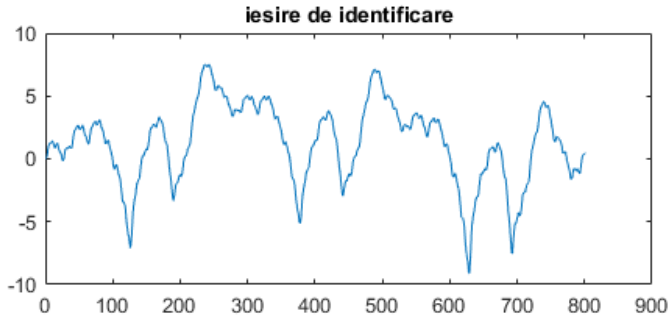
SIMULAREA

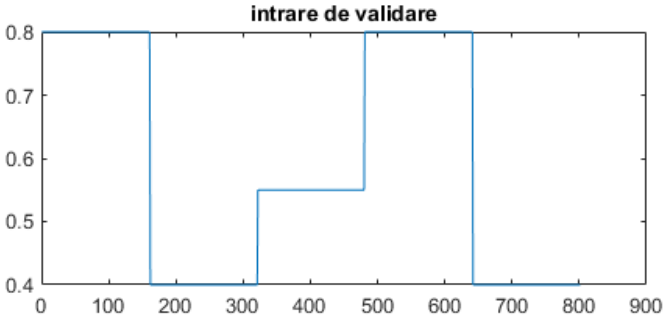
1.4. Algoritm de lucru

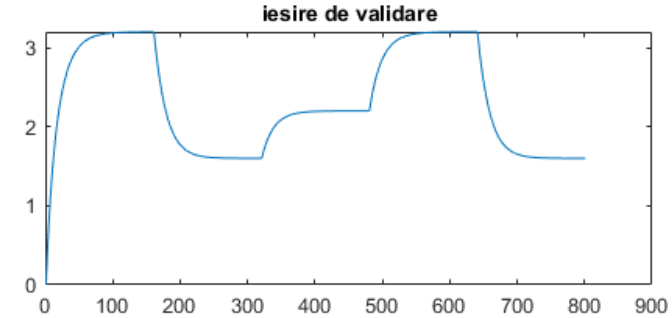


Reprezentarea datelor









METODE DE IMPLEMENTARE 2

2.1. Metoda 1

Pentru determinarea tuturor combinațiilor de tipul $x_1^{p_1} \cdot x_2^{p_2} \cdot \ldots \cdot x_n^{p_n}$ *favorabile unde n=na+nb, ne folosim de operatorii punctuali din Matlab. Astfel, vom ridica fiecare termen din vectorul* $a(k) = [x_1(k), x_2(k), \ldots, x_n(k)]$ *la o putere corespunzatoare din M.*

$$M = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ p_{x1} & p_{x2} & \cdots & p_{xn} \end{bmatrix}, \text{ unde } p_{q1} + p_{q2} + \cdots + p_{qn} \leq m, q = 1, x \Leftrightarrow$$

$$\Leftrightarrow a(k).\hat{}M = \begin{bmatrix} x_1^{p_{11}} & x_2^{p_{12}} & \cdots & x_n^{p_{1n}} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{p_{x1}} & x_2^{p_{x2}} & \cdots & x_n^{xn} \end{bmatrix}$$

Regresorii $x_1^{p_1} \cdot x_2^{p_2} \cdot \ldots \cdot x_n^{p_n}$ cu proprietatea că $p_1 + p_2 + \cdots + p_n \leq m$ se vor obține înmulțind elementele de pe fiecare linie, obținând astfel un vector coloană pe care îl vom transpune și adăuga la matricea de regresori Φ cu ajutorul căreia compunem sistemul $\mathbf{Y = \Phi \cdot \theta}$.

1. Metoda 1

Generarea matricei de puteri

Am pornit de la ideea de a realiza produsul cartezian a n=na+nb mulțimi de forma {0,1,2,. . . ,m} și să îl reținem într-o matrice de dimensiune $m^{m \cdot n}$. Eliminăm din această matrice toate liniile care nu respectă condiția ca suma elementelor să fie ≤m.

1. Metoda 1

Generarea matricei de puteri

Această implementare, deși bună ca idee de bază, creează mari probleme în privința memoriei necesare pentru a stoca matricea ce păstrează produsul cartezian. De asemenea, această implementare ar fi durat mult fiindcă înainte de a filtra rezultatele obținute ar fi trebuit să generam mult mai multe combinații de puteri ce abia mai apoi ar fi fost eliminate, deci nici din punct de vedere al timpului de execuție nu era o metodă fiabilă pentru valori prea mari.

1. Metoda 1

Generarea matricei de puteri

Optimizând ideea de mai sus am ajuns la soluția de a genera produsul cartezian a acelor n mulțimi eliminând treptat din matricea ce reține produsul cartezian liniile ce conțineau elemente a căror sumă depășeau deja valoarea maximă m.

Un alt rezultat important ce a facilitat această implementare a fost **determinarea numarului de elemente ale unui polinom de n variabile si grad m.**

2.1.1. Determinarea numărului de elemente al polinomului de n variabile și grad m

$$P = \sum_i \theta_i \cdot (x_1^{p_{i1}} \cdot x_2^{p_{i2}} \cdot \ldots \cdot x_n^{p_{in}}) \quad 0 \leq p_1 + p_2 + \ldots + p_n \leq m$$

Câți termeni are polinomul $P(x_0, x_1, \dots, x_n)$ de *n+1* variabile și grad *m*?

1. Metoda 1

2.1.1 Determinarea numărului de elemente al polinomului de n variabile și grad m

Pentru început vom determina câte combinații de tipul $x_0^{p_0} \cdot x_1^{p_1} \cdot \ldots \cdot x_n^{p_n}$ cu $p_0 + p_1 + \ldots + p_n = m$ sunt. Fie următoarea schemă: $\underbrace{XXX\ldots X}_{m+n}$, unde m este gradul.

- Alegem oricare n X-uri și le schimbăm în 0: $\underbrace{X0\,0XXX\ldots X0X}_{m \cdot X + n \cdot 0}$
- Citim câte steluțe avem între fiecare cerc, de exemplu:
  $XXX00X0XX0XX0 \rightarrow x_0^3 \cdot x_1^0 \cdot x_2^1 \cdot x_3^2 \cdot x_4^2 \cdot x_5^0$
  $5 \cdot 0 \Rightarrow$ 6 variabile cu grad maxim 8

2. Metoda 1

2.1.1. Determinarea numărului de elemente al polinomului de n variabile și grad m

Revenind la problemă:

Avem n+m X-uri și trebuie să alegem pe poziții diferite n 0-uri. Numărul de X-uri dintre două 0-uri va reprezenta puterea unui anumit $x_i, i \leq n$. Observăm că numărul de X-uri rămase va fi n+m-n=m. Deci avem C_{m+n}^{n} monoame de *n+1* elemente distincte de grad m.

Astfel un polinom de n+1 variabile și grad m va avea:

termeni de n+1 variabile

$C_n^n + C_{n+1}^n + C_{n+2}^n + C_{n+3}^n + \ldots + C_{n+m}^n$ termeni în componența sa. $\Leftrightarrow$

grad 0 grad 1

$$\Leftrightarrow \frac{n!}{n!\cdot 0!} + \frac{n!}{(n+1)!\cdot 1!} + \frac{n!}{(n+2)!\cdot 2!} + \ldots + \frac{n!}{(n+k)!\cdot k!} + \ldots + \frac{n!}{(n+m)!\cdot m!} = \sum_{k=0}^{m} C_{n+m}^{n} = \boldsymbol{C_{n+m+1}^{n+1}}$$ (Identitatea Hockey-Stick)

Pentru n termeni vom avea deci C_{n+m}^{n} termeni în polinomul $p(x_0, \ldots, x_{n-1})$ de grad m.

Demonstrația Identității Hockey-Stick

$$C_n^k = C_{n-1}^{k-1} + C_{n-1}^{n+1}$$

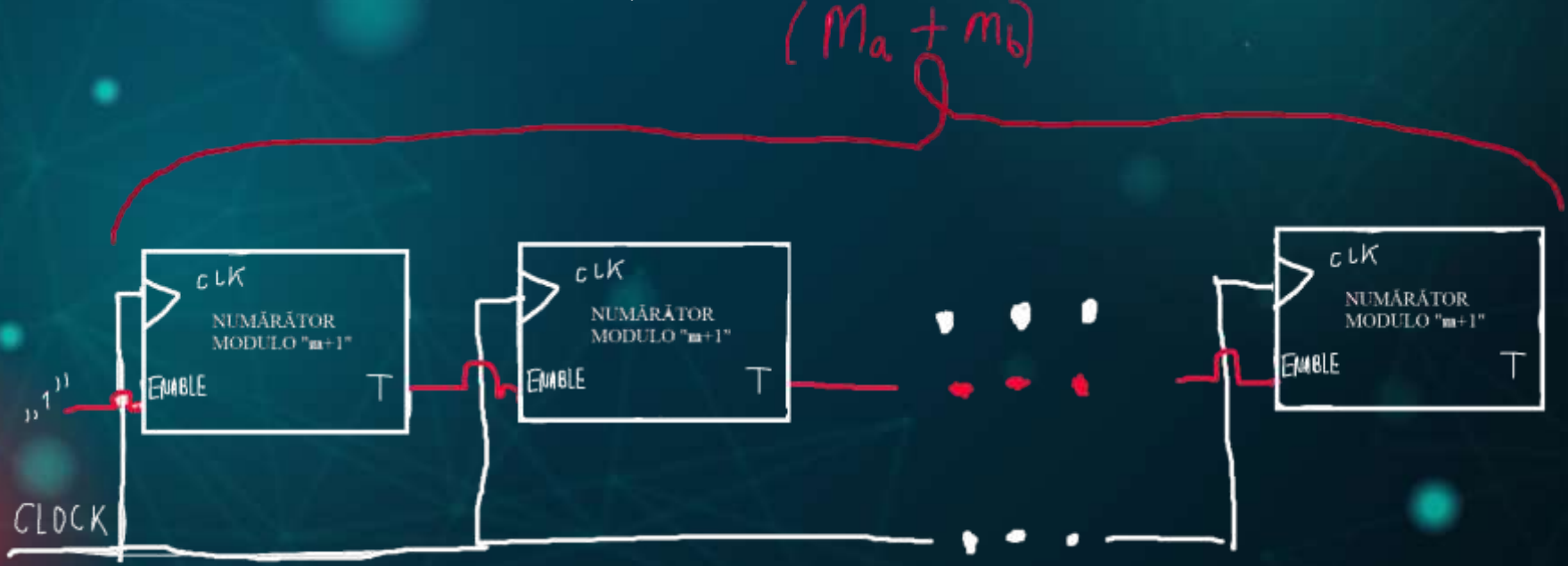
$$\begin{aligned}
\sum_{k=0}^{m} C_{n+k}^{n} &= C_n^n + C_{n+1}^n + C_{n+2}^n + \ldots + C_{n+m}^n \\
&= C_{n+1}^{n+1} + C_{n+1}^n + C_{n+2}^n + \ldots + C_{n+m}^n \\
&= C_{n+2}^{n+1} + C_{n+2}^n + C_{n+3}^n + \ldots + C_{n+m}^n \\
&= C_{n+3}^{n+1} + C_{n+3}^n + \ldots + C_{n+m}^n \\
&= \ldots = C_{n+m}^{n+1} + C_{n+m}^n \\
&= C_{n+m+1}^{n+1}
\end{aligned}$$

Acest rezultat este important din mai multe motive:

1. Se poate verifica dacă generăm numărul corespunzator de linii în matricea de puteri M.
2. Putem prealoca matricea M pentru a scuti matlab-ul de a realoca matricea M pentru fiecare iterație pe care o realizăm, număr de iterații ce poate fi considerabil de mare. Acest fapt facilitează și creșterea vitezei de execuție a algoritmului.
3. Folosind această prealocare nu vor disparea problemele legate de memoria necesară executiei algoritmului, dar aceste erori de executare (runtime error) nu vor apărea atat de repede in program (avem garantia ca vom putea executa programul și pentru valori ale ordinului sistemului mai mari decât 5 și un grad al polinomului de cel puțin 10).

2.2. Metoda 2

Diferențierea față de metoda 1 este reprezentată prin construirea în mod diferit a matricei de puteri M. Implementarea are la bază cascadarea a (na+nb) număratoare modulo (m+1). În practică, fiecare numărator când ar ajunge la valoarea maximă (m), la următorul clock, va da impuls de "enable" următorului numărtor cascadat, ca să îi mărească valoarea.



Metoda 2

Acest cod a fost implementat astfel încât la momentul în care suma valorilor din toate număratoarele va depăși valoarea lui m se va căuta numărătorul de stare cea mai nesemnificativă, ce conține o valoare, se va reseta și va determina următorul numărător să își mărească valoarea.

Algoritmul își va termina executarea în momentul în care ar ajunge să distribuie un impuls numărătorului (na+nb+1), care în realitate nu există. Practic, în algoritm clock-ul numărătorului reprezintă fiecare interație din buclă.

REZULTATE DE REGLARE

3

3.1. Media erorilor pătratice și grafice reprezentative
3.2. Cele mai bune aproximări

3.1. Media erorilor pătratice și grafice reprezentative

Zerourile din tabele următoare se datoreaza preinițializării matricelor cu valori de 0 și nu obținerii unor erori nule.

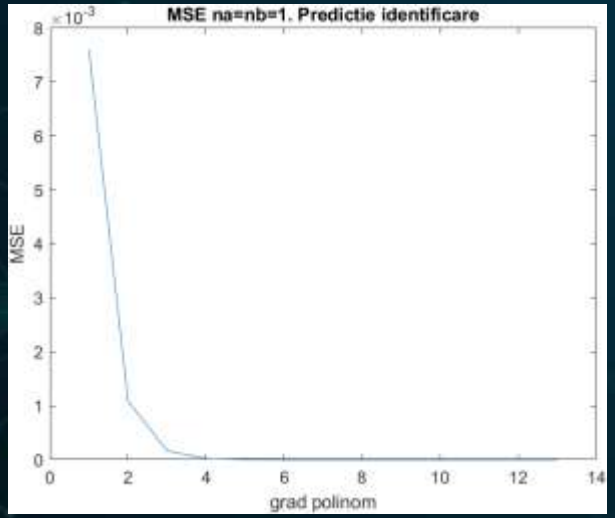
Observație: nu am populat toate locurile din tabel din cauza timpului de execuție foarte mare în principal cauzat de problema MatLab-ului de a lucra cu valori NaN.

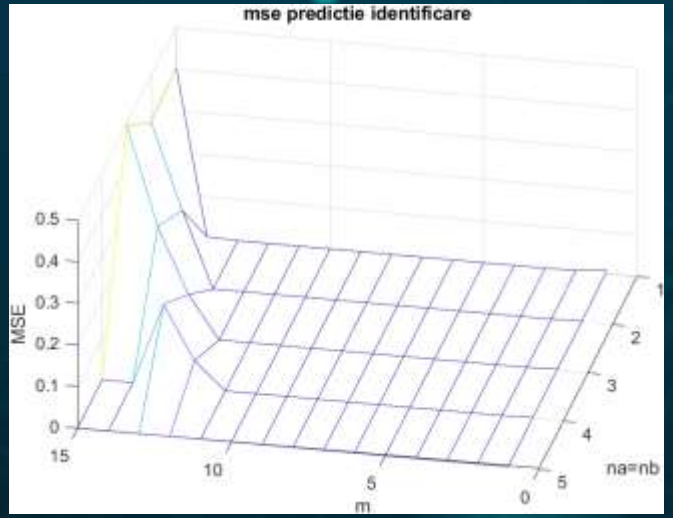
De asemenea, pentru urmărirea ușoară a graficelor, fiecare coloană este corespunzătoare unui grad m din intervalul [1,15], iar liniile sunt corespunzătoare ordinului sistemului na=nb din intervalul [1,5].

3.1. Media erorilor pătratice pentru modelele reprezentative

MSE pentru predicția identificării

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0.0076	0.0011	1.6796e-04	2.8204e-05	5.1450e-06	9.2034e-07	1.7397e-07	3.6793e-08	8.7060e-09	1.9509e-09	4.1050e-10	6.3789e-11	8.7566e-12	0.0032	0.4034
2	0.0030	4.6284e-04	3.6228e-05	1.2064e-06	3.5905e-08	1.1673e-09	2.4990e-11	5.7283e-13	2.4090e-14	4.4777e-16	7.3271e-18	2.1341e-15	1.6951e-06	0.1819	0.3857
3	0.0027	2.9645e-04	1.2256e-05	8.2581e-08	5.4366e-10	1.1407e-12	5.8307e-15	9.7900e-18	3.6190e-20	6.3519e-18	4.9884e-12	1.9700e-06	0.1022	0.2591	0.4949
4	0.0027	2.2320e-04	2.6059e-06	1.6699e-09	8.4633e-14	1.8394e-24	3.5084e-28	1.9313e-26	4.8192e-23	5.6362e-10	2.4523e-07	0.0660	0.1950	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

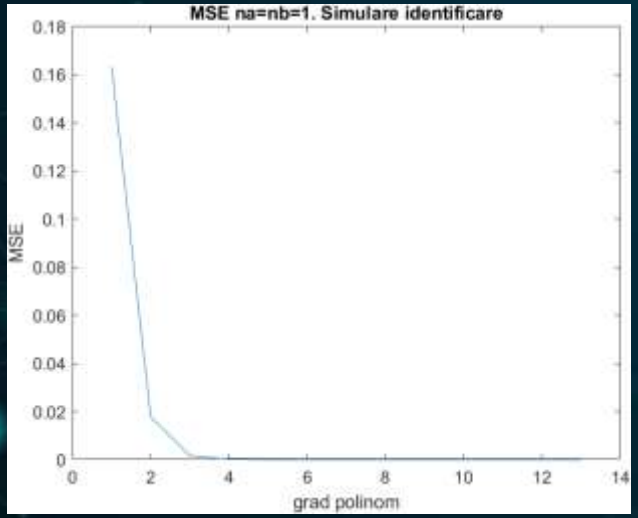


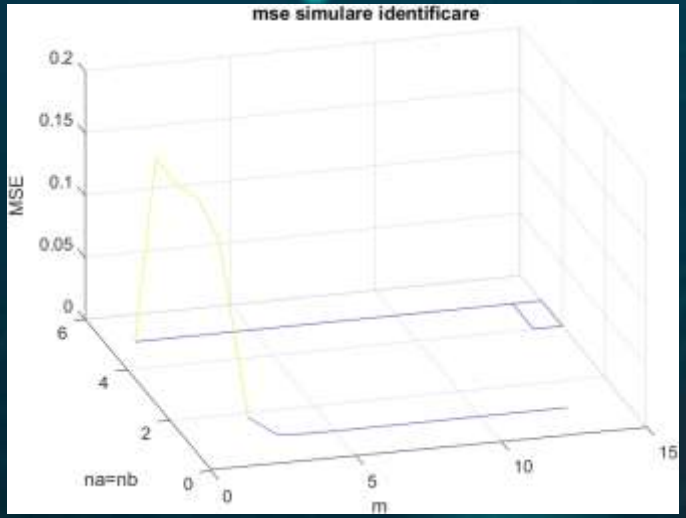


3.1. Media erorilor pătratice ... reprezentative

MSE pentru simularea identificării

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0.1633	0.0178	0.0018	2.0017e-04	2.9758e-05	4.6354e-06	8.3600e-07	1.4166e-07	3.2298e-08	1.0987e-08	2.8811e-09	4.7766e-10	5.8505e-11	NaN	NaN
2	0.1763	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	0.1664	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	0.1684	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

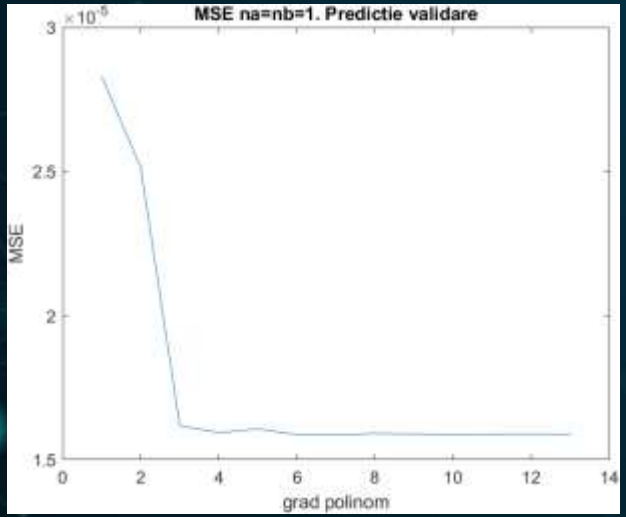


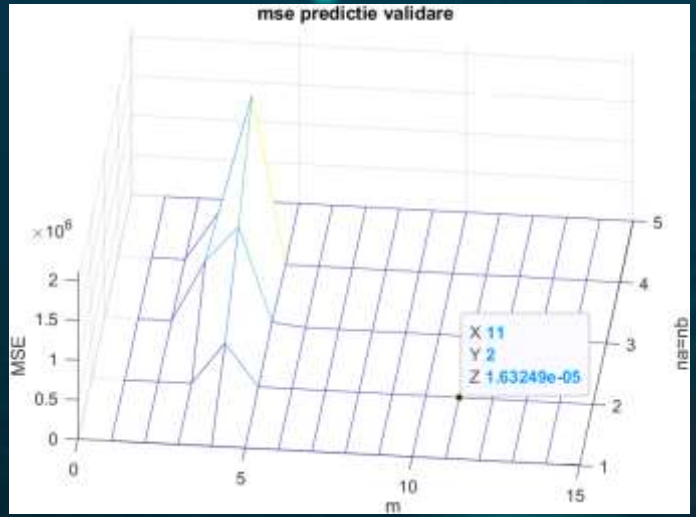


3.1. Media erorilor pătratice ... reprezentative

MSE pentru predicția validării

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	2.8301e-05	2.5132e-05	1.6180e-05	1.5946e-05	1.6071e-05	1.5874e-05	1.5854e-05	1.5916e-05	1.5891e-05	1.5881e-05	1.5889e-05	1.5890e-05	1.5889e-05	2.0594e-04	0.7293
2	3.3492e-05	0.3525	713.2259	5.3109e+05	609.2971	70.5502	0.0075	0.0311	1.8821e-04	1.7626e-05	1.6325e-05	1.5894e-05	0.1139	2.5155e+03	1.8012e+03
3	4.3825e-05	1.0731	7.7085e+05	1.2270e+06	4.5032e+04	0.3305	6.1181e-05	5.9082e-05	1.5889e-05	3.9757e-05	1.0050e-04	18.1608	550.4120	4.2840e+03	63.8909
4	4.5027e-05	0.0497	6.1370e+05	2.1139e+06	339.3165	0.0029	4.8726e-05	5.1263e-05	1.4218e-04	0.4872	4.3573	5.3301e+03	189.1837	281.1897	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

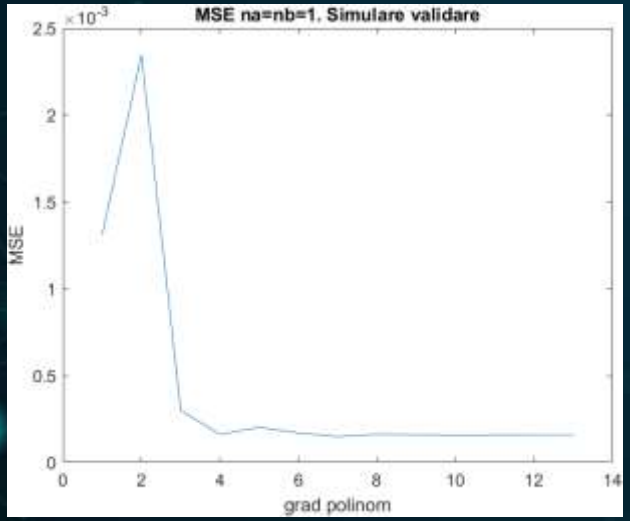


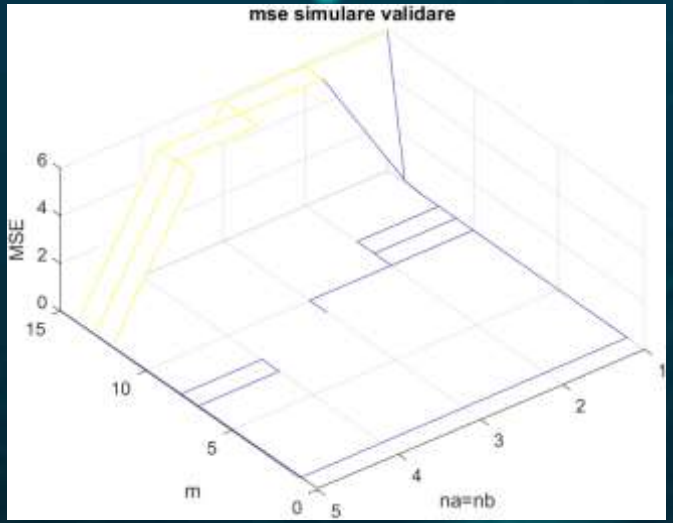


3.1. Media erorilor pătratice pentru diferite reprezentative

MSE pentru simularea validării

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0.0013	0.0023	3.0100e-04	1.6165e-04	2.0297e-04	1.7034e-04	1.4864e-04	1.6303e-04	1.6088e-04	1.5799e-04	1.5876e-04	1.5949e-04	1.5925e-04	0.1001	5.8523
2	0.0043	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1.5922e-04	1.5922e-04	1.5922e-04	NaN	5.8359	5.8559
3	0.0020	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1.5922e-04	1.5921e-04	NaN	NaN	5.8318	5.8561	5.8577
4	0.0024	NaN	NaN	NaN	NaN	NaN	1.5930e-04	1.5935e-04	NaN	NaN	NaN	5.8040	5.8583	5.8577	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0





3.1. Media erorilor pătratice și grafice reprezentative

În urma observării atât a graficelor, cât și a tabelelor cu erori putem face trage următoarele **concluzii**:

- datele de identificare, la predicție, suferă același fenomen de supraantrenare, dar valorile erorilor nu sunt dezastruoase, ba chiar găsim erori de ordinul 10^{-28}cu cât creștem ordinul sistemului, deci comportarea este una bună, aproape ideală;
- asemenea predicției datelor de identificare, și predicția datelor de validare se comportă normal odată cu creșterea ordinului și a gradului, cu diferența că erorile pot avea discrepanțe foarte mari între ele în funcție de ordinul și gradul ales;
- simularea, atât pentru identificare, cât și pentru validare are un comportament mult mai pretențios, ordinul și gradul ales pentru aproximator fiind foarte importante în vederea obținerii unei aproximări bune. Spre deosebire de predicție, în cazul în care încercăm aproximarea unui sistem ce este la bază de ordin 1 cu un sistem de ordin 2, spre exemplu, erorile vor fi din ce în ce mai mari, iar ieșirea simulată poate ajunge sa tindă chiar și la infinit.

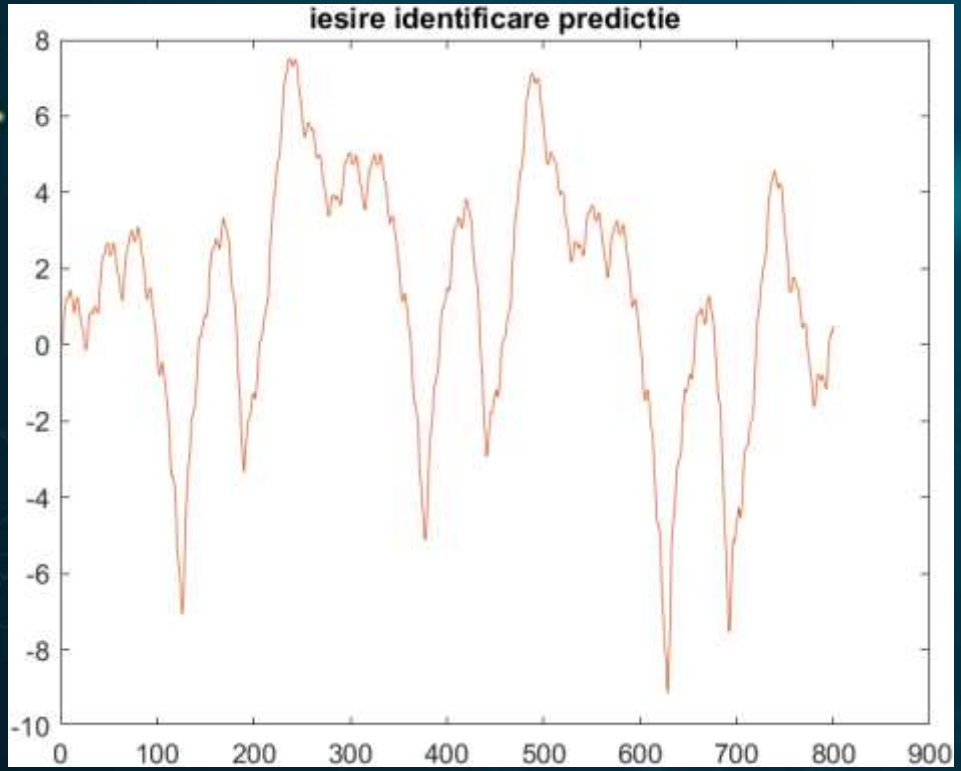
3.1. Media erorilor pătratice și grafice reprezentative

Observație:

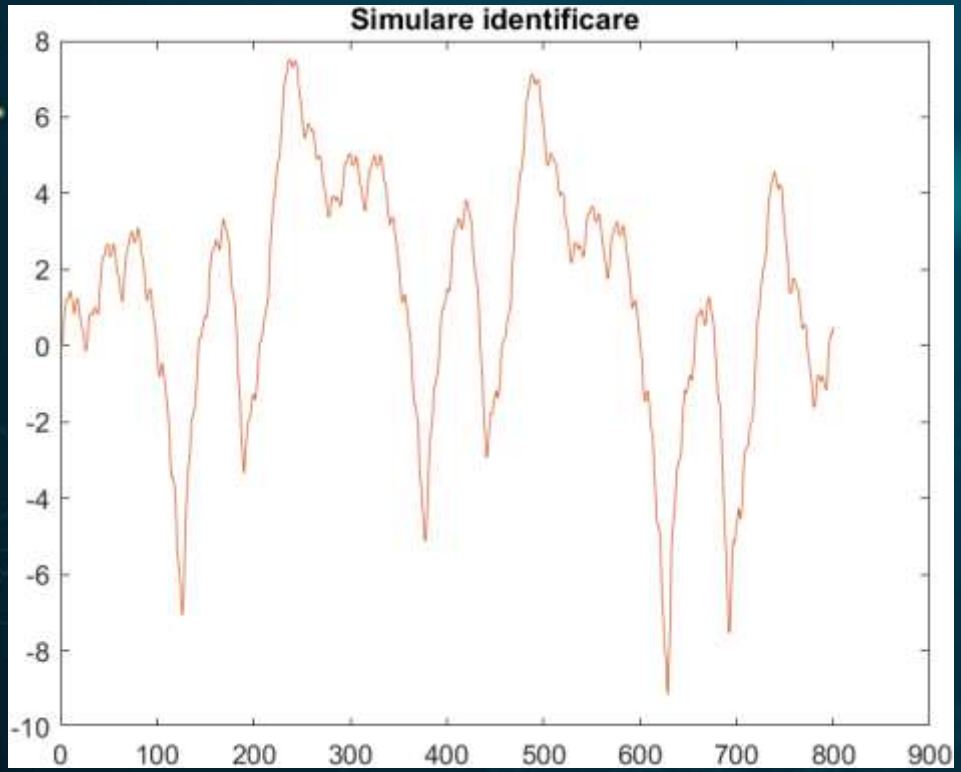
Discontinuitățile apărute in graficele realizate cu funcția mesh() sunt datorate valorilor NaN ale erorilor medii pătratice, valori ce nu sunt reprezentabile pe un grafic.

Considerând cele mai bune aproximări obținute în funcție de eroarea medie pătratică am ajuns la concluzia că pentru ordinul 1 si gradul 13 al aproximatorului vom obține cele mai bune aproximări.Aceste aproximări sunt prezentate mai jos:

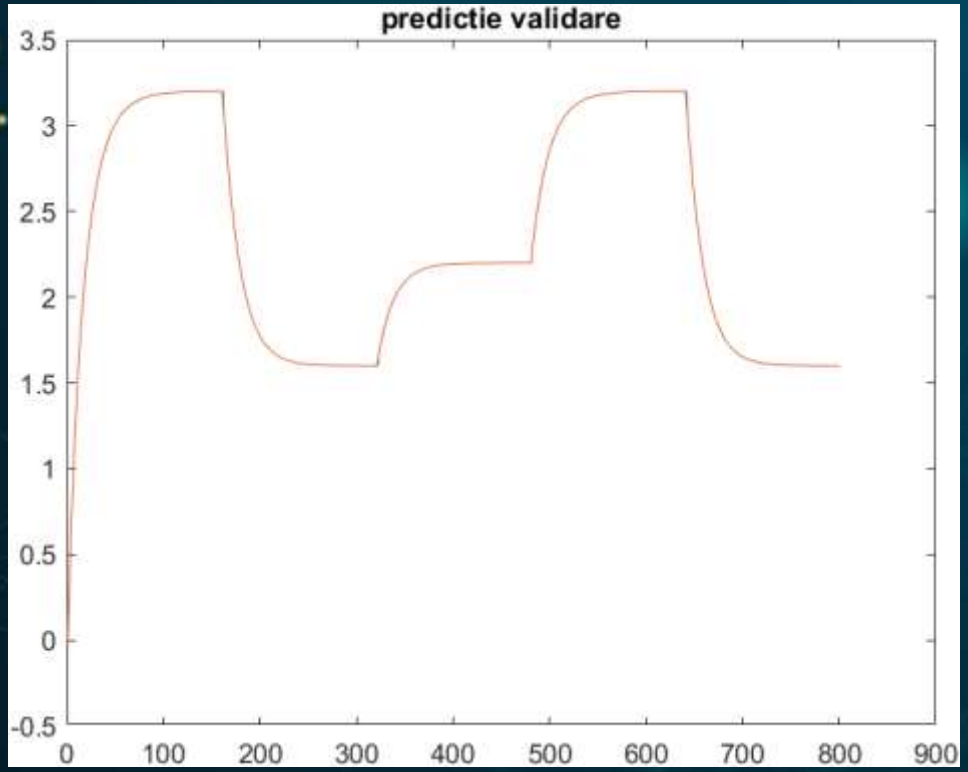
3.2. Cele mai bune aproximări



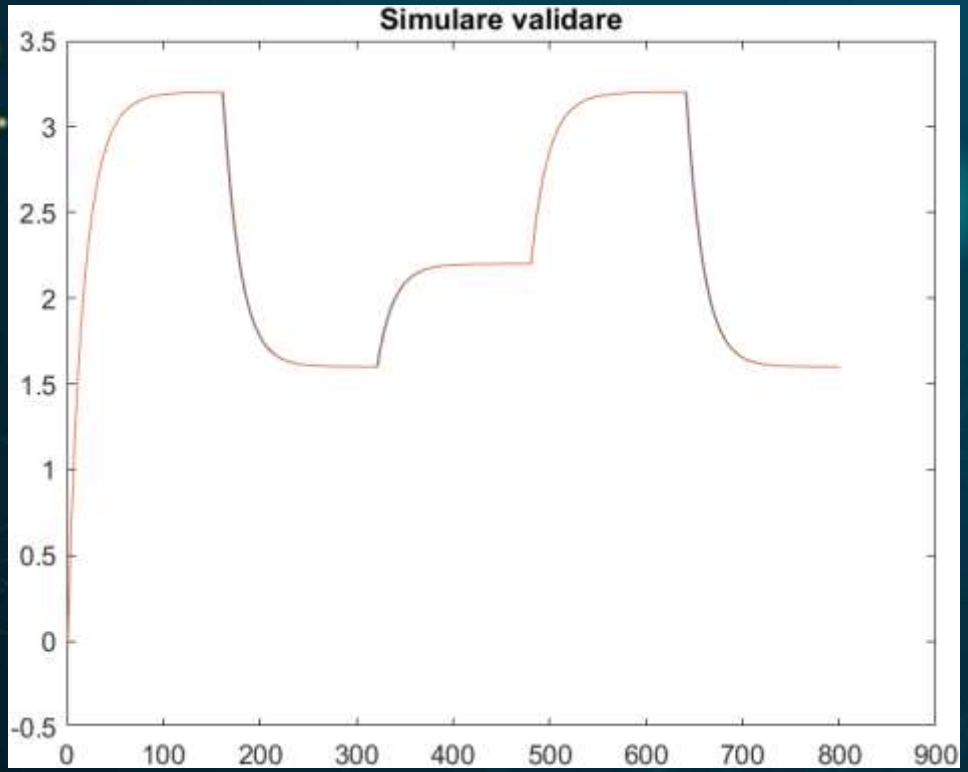
3.2. Cele mai bune aproximări



3.2. Cele mai bune aproximări

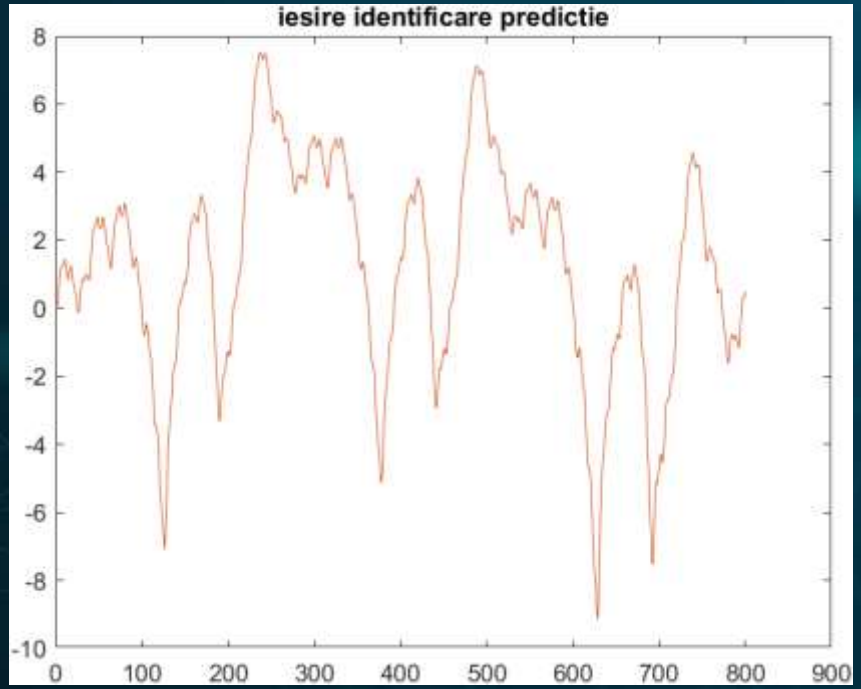


3.2. Cele mai bune aproximări



3.2. Alte aproximări bune

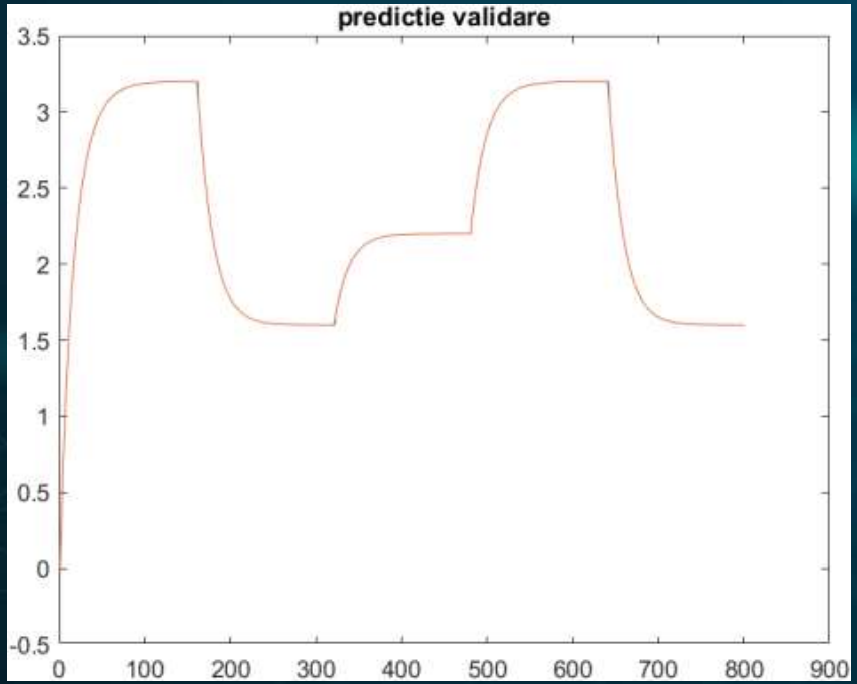
Pentru na=1, nb=2 și m=13:



MSE=1.5939e-12

3.2. Alte aproximări bune

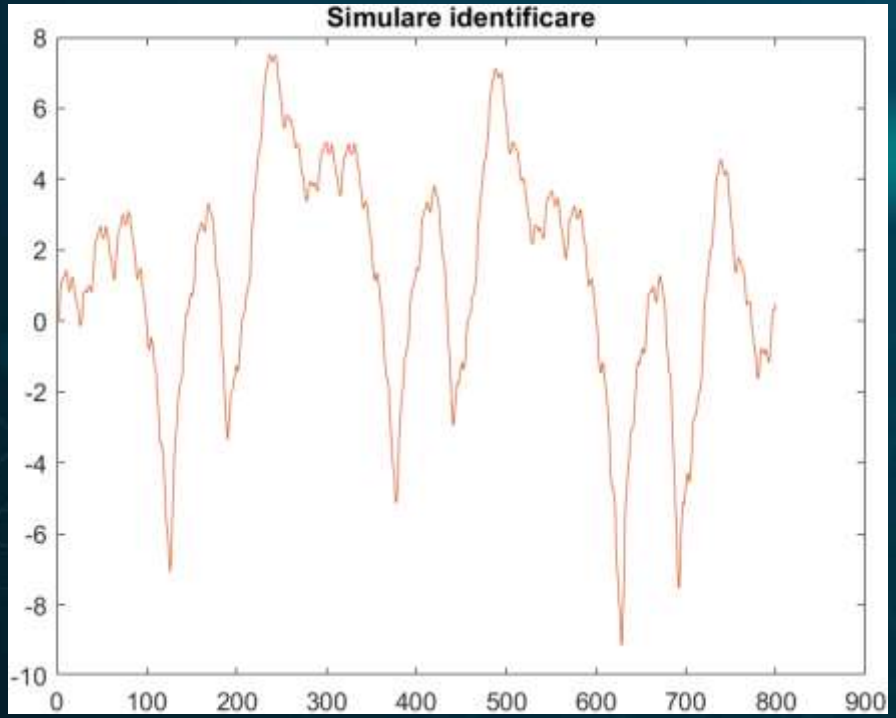
Pentru n...



MSE=1.5889e-5

3.2. Alte aproximări bune

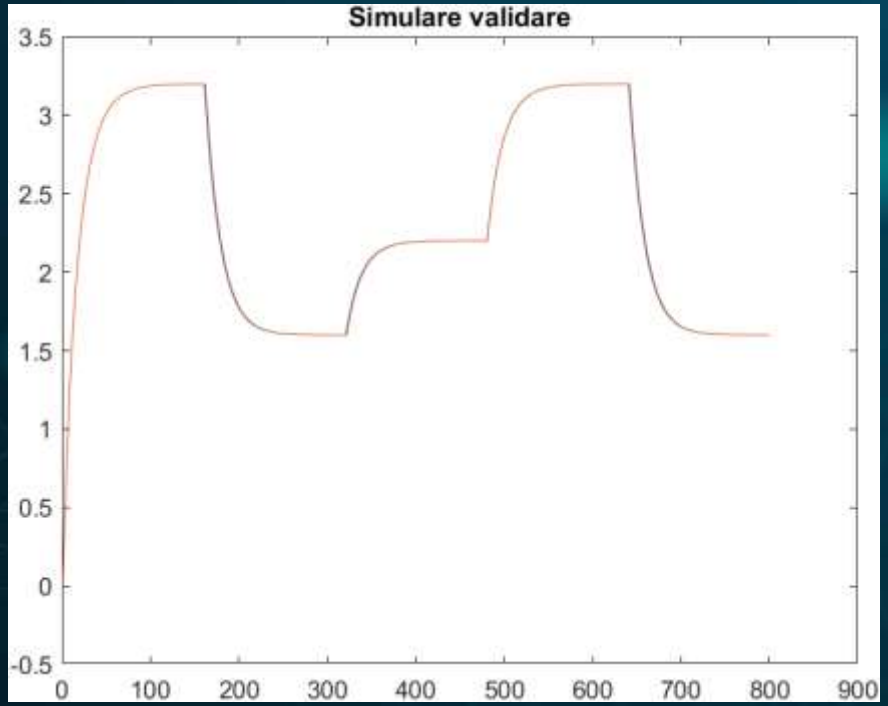
Pentru n...



MSE=1.4944e-11

3.2. Alte aproximări bune

Pentru n...



MSE=1.5923e-4

3.2. Alte aproximări bune

Observații:

Am încercat să considerăm un sistem de ordin mai mare, cele mai bune rezultate pentru acest caz fiind găsite pentru na=1, nb=2 si m=13.

De asemenea, erorile medii pătratice generate de acest model sunt comparabile cu rezultatele obținute pentru na=nb=1 si m=13, însă judecând după memoria și timpul necesar de execuție ai algoritmului, cel mai avantajos de generat model îl vom obține pentru na=nb=1 si m=13.

CONCLUZII 4

4. Concluzii

- Asemenea metodei de regresie liniară, metoda ARX poate produce fenomenul de supraantrenare pentru grade prea mari ale aproximatorului (erori ce pot proveni și din erorile de calcul numeric la care pot fi supuse datele).

- Ordinul sistemului trebuie considerat cu atenție, astfel sunt șanse mari să nu putem obține atât predicții, dar mai ales simulări care se apropie de sistemul real.

- Metoda ARX, spre deosebire de regresie, oferă posibilitatea de a obține ieșirea sistemului, fără a avea acces la ieșirea reală a acestuia, cu ajutorul simulării.

- Deși este o metodă simplă, aceasta are limitările ei deoarece nu poate modela decât sisteme ale cărei intrări au Persistența Excitației suficient de mare (intrări care conțin suficientă informație $\iff$ pot pune sistemul în cât mai multe situații posibile) și în care perturbația este zgomot alb de medie zero.

MULȚUMIM PENTRU ATENȚIE!

Așteptăm întrebările voastre