Proiect partea 2:

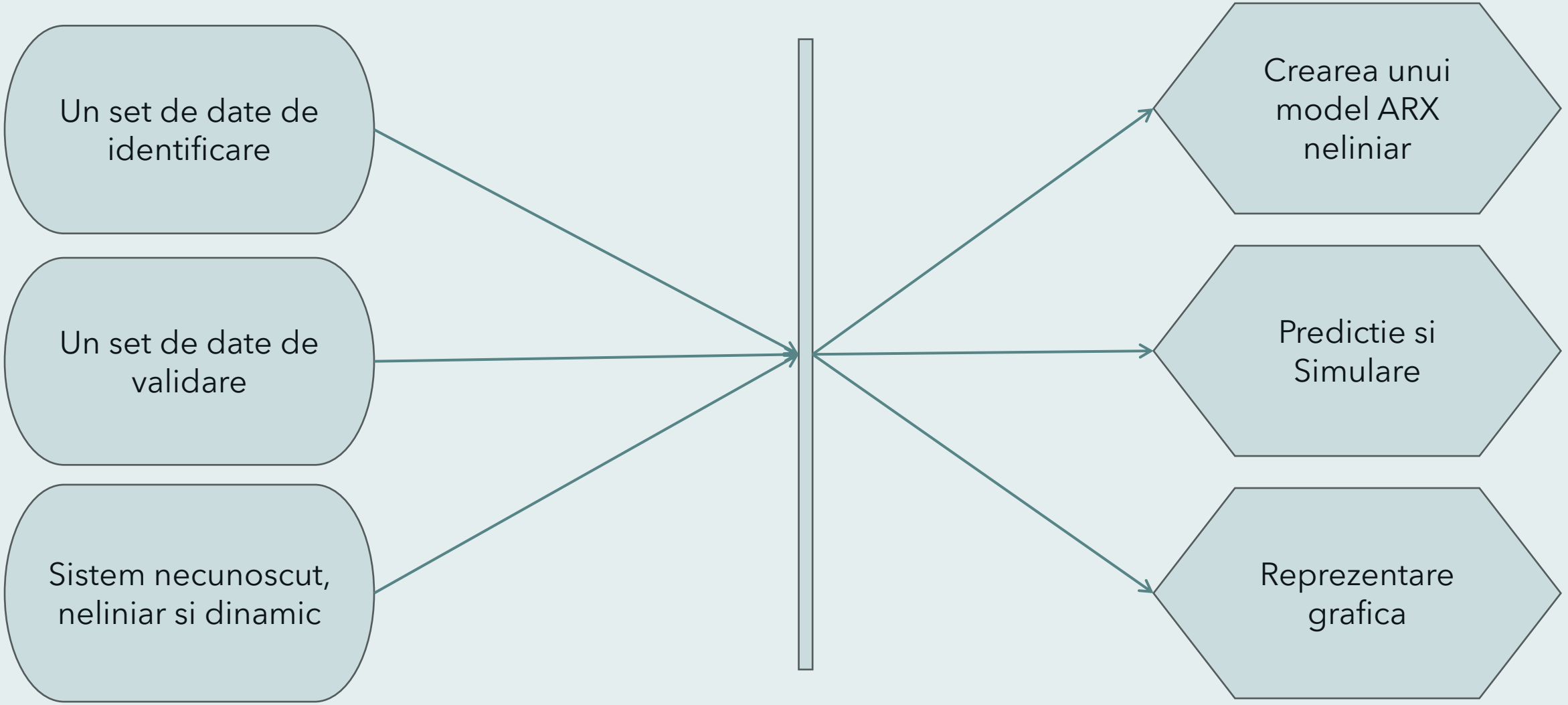
ARX Neliniar

Borzasi Naomi, Butcovan Amalia,
Domide Maria

PidX: **06/16**

Cuprins

Prezentarea problemei



Ce este un model NARX?

Structura unui model NARX:

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

Unde:

- $\hat{y}$ – modelul obtinut
- p – un polinom de grad n
- na, nb – ordinele modelului
- nk – intarzierea
- d(k) – vectorul de iesiri si intrari intarziate

Cum gasim parametrii?

1. Incarcarea datelor si definirea lui na, nb si n;

2. Generarea combinatiilor polinomiale, utilizand functia nchoosek;

3. Construirea matricii de regresori;

$$\textbf{termen = termen} * \textbf{d\_k(combinatii(i, j))}$$

$$\textbf{matrice\_regresori(k, i) = termen}$$

4. Aflarea parametrilor

$$\theta = \textbf{matrice\_regresori \textbackslash y\_identificare}$$

Care este procesul de ajustare (predictie si simulare)?

1. Calcularea iesirii de predictie

$$\hat{y} = \textbf{matrice\_regresori} * \theta$$

   - Matricea de regresori contine valorile reale, intarziate ale iesirii y

2. Calcularea iesirii de simulare

$$\tilde{y} = \textbf{matrice\_regresori} * \theta$$

   - Matricea de regresori contine valorile intarziate ale iesirii $\tilde{y}$

3. Calcul MSE

$$\boldsymbol{MSE} = \frac{\mathbf{1}}{\boldsymbol{N}} \sum_{\boldsymbol{k=1}}^{\boldsymbol{N}} (\boldsymbol{Y} - \hat{\boldsymbol{Y}})^{\mathbf{2}}$$

4. Ajustarea gradului n, si a intarzierilor na si nb

Analiza rezultatelor

FIGURA 1.1 na=nb=1 n=2

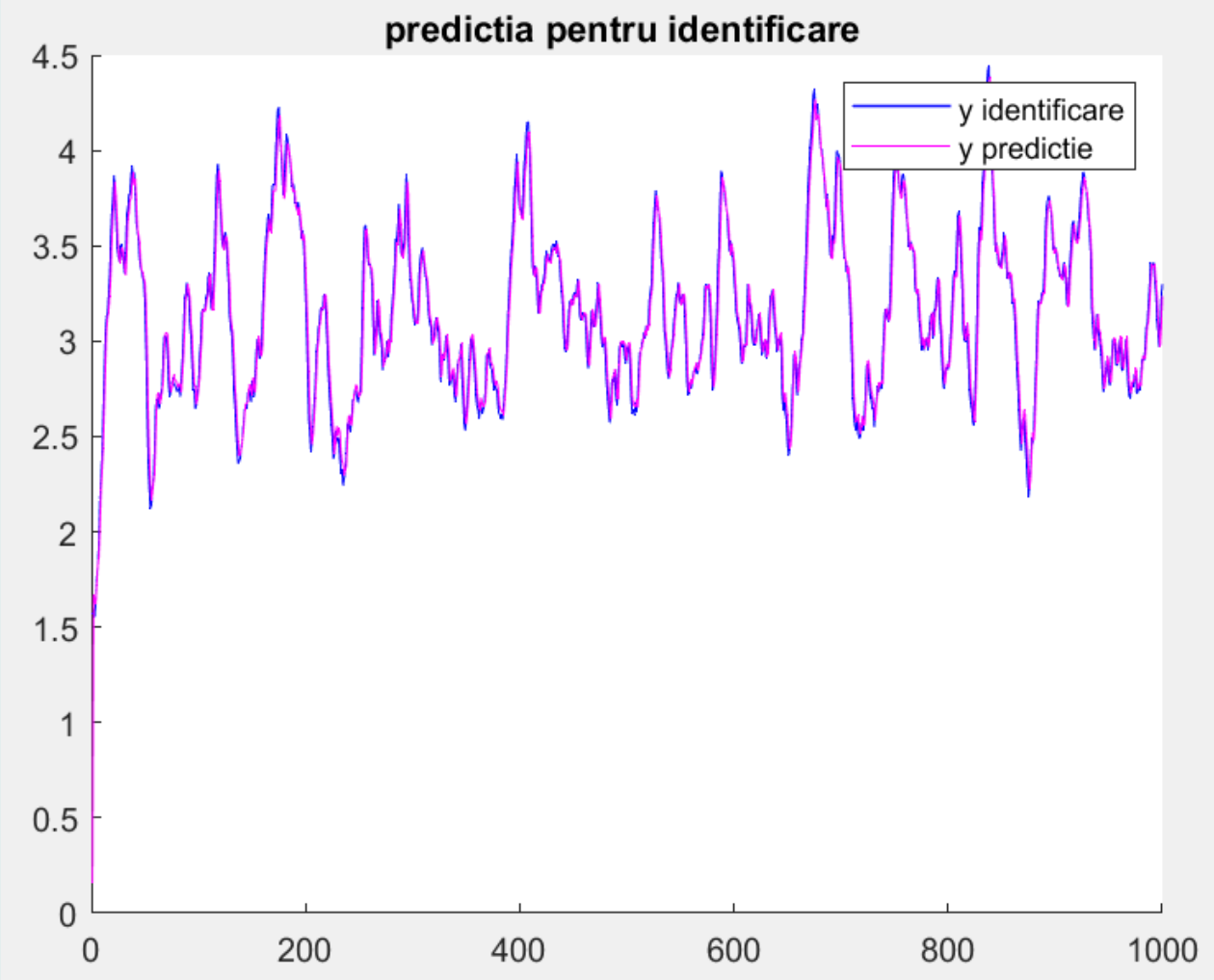


Fig 1.1 a)

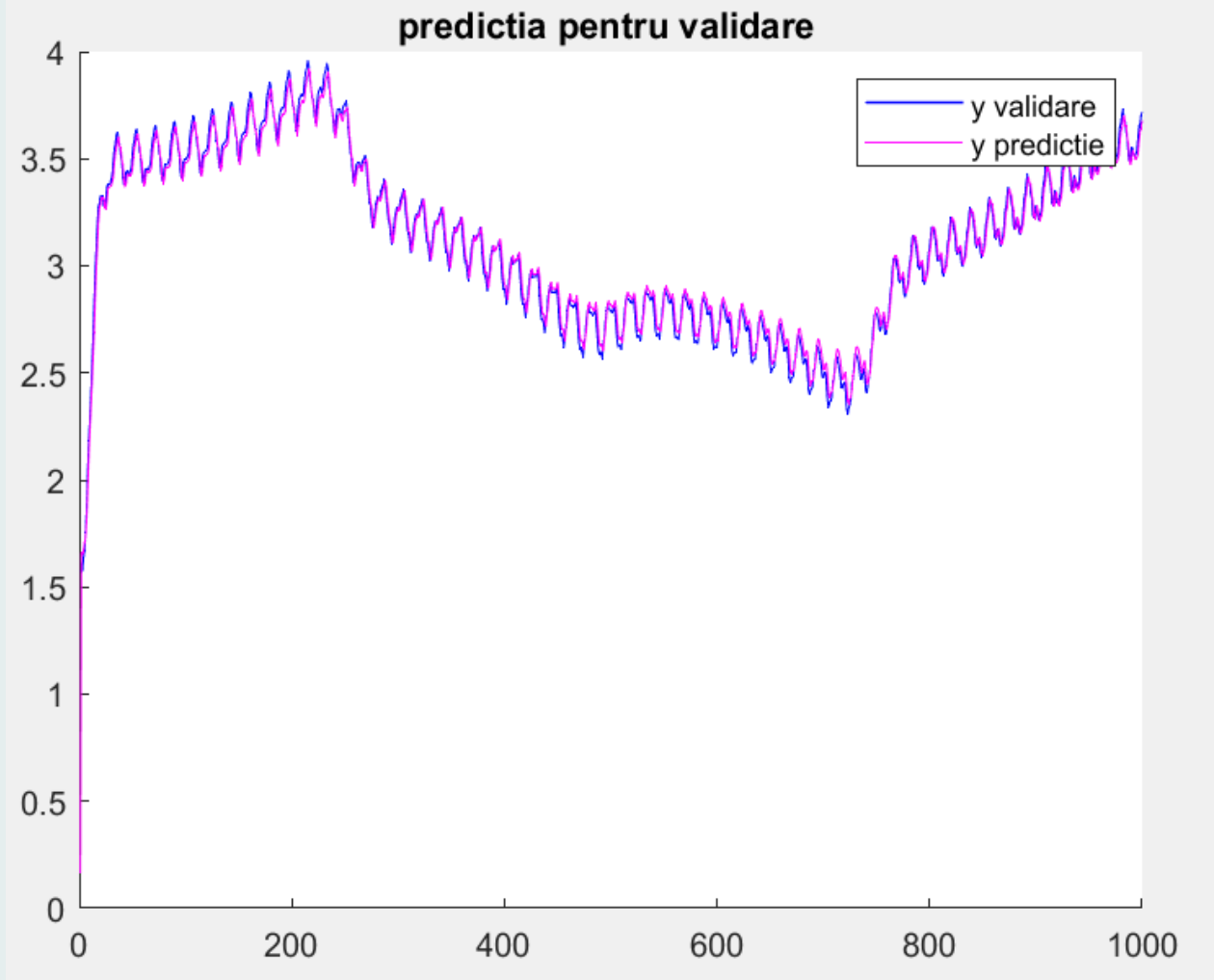


Fig 1.1 b)

FIGURA 1.2 na=nb=1 n=2

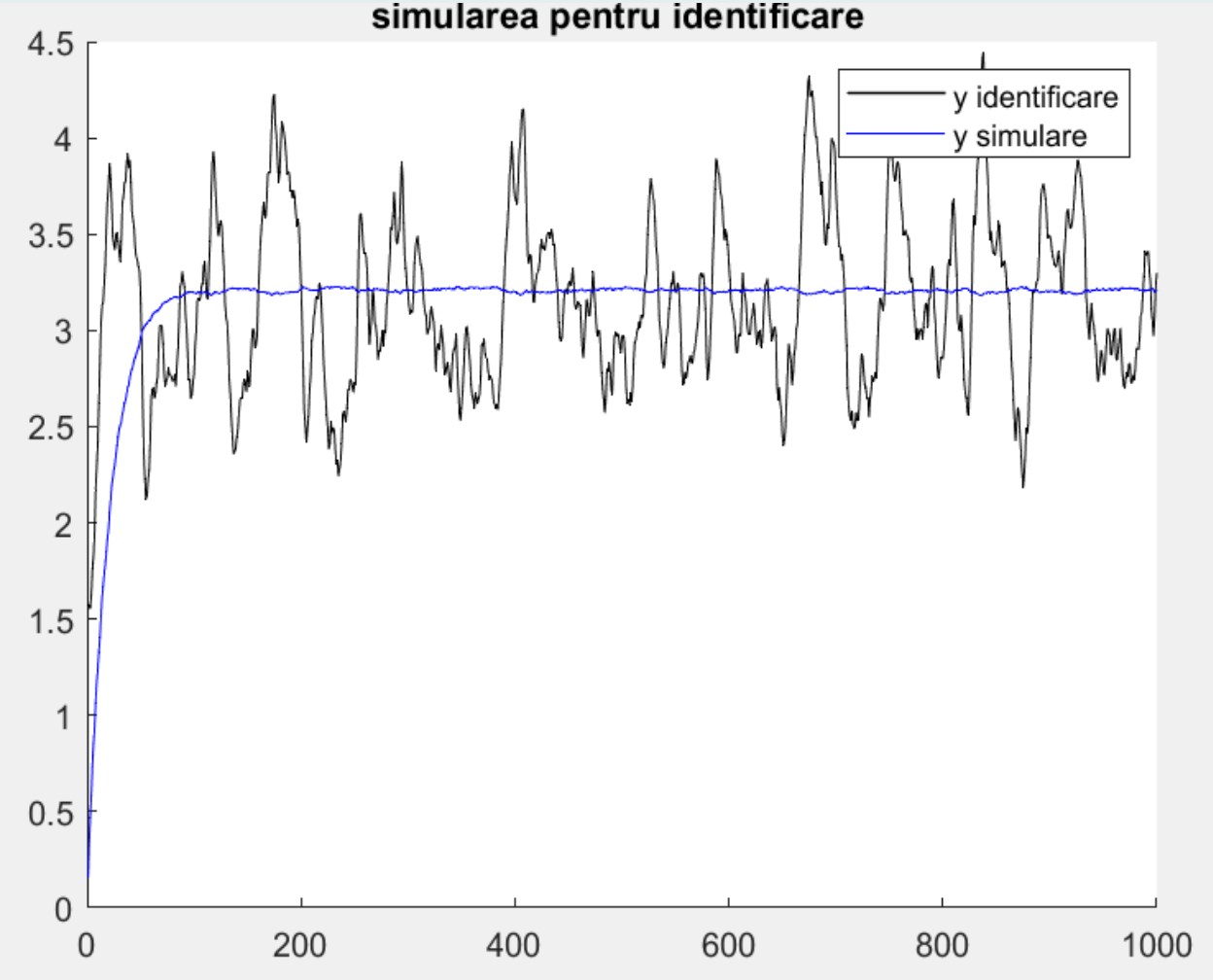


Fig 1.2 a)

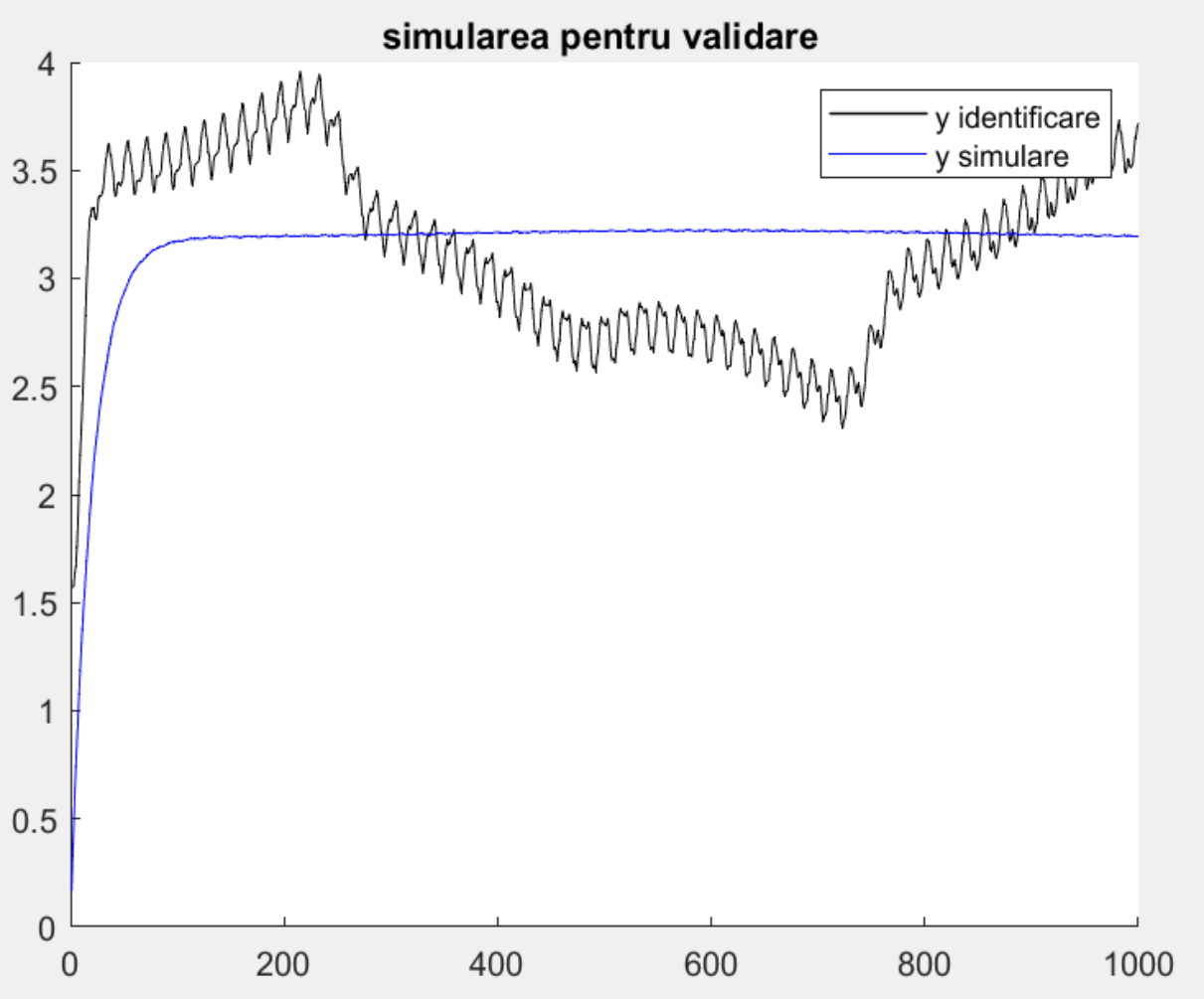


Fig 1.2 b)

FIGURA 2.1 na=2 nb=2 n=2

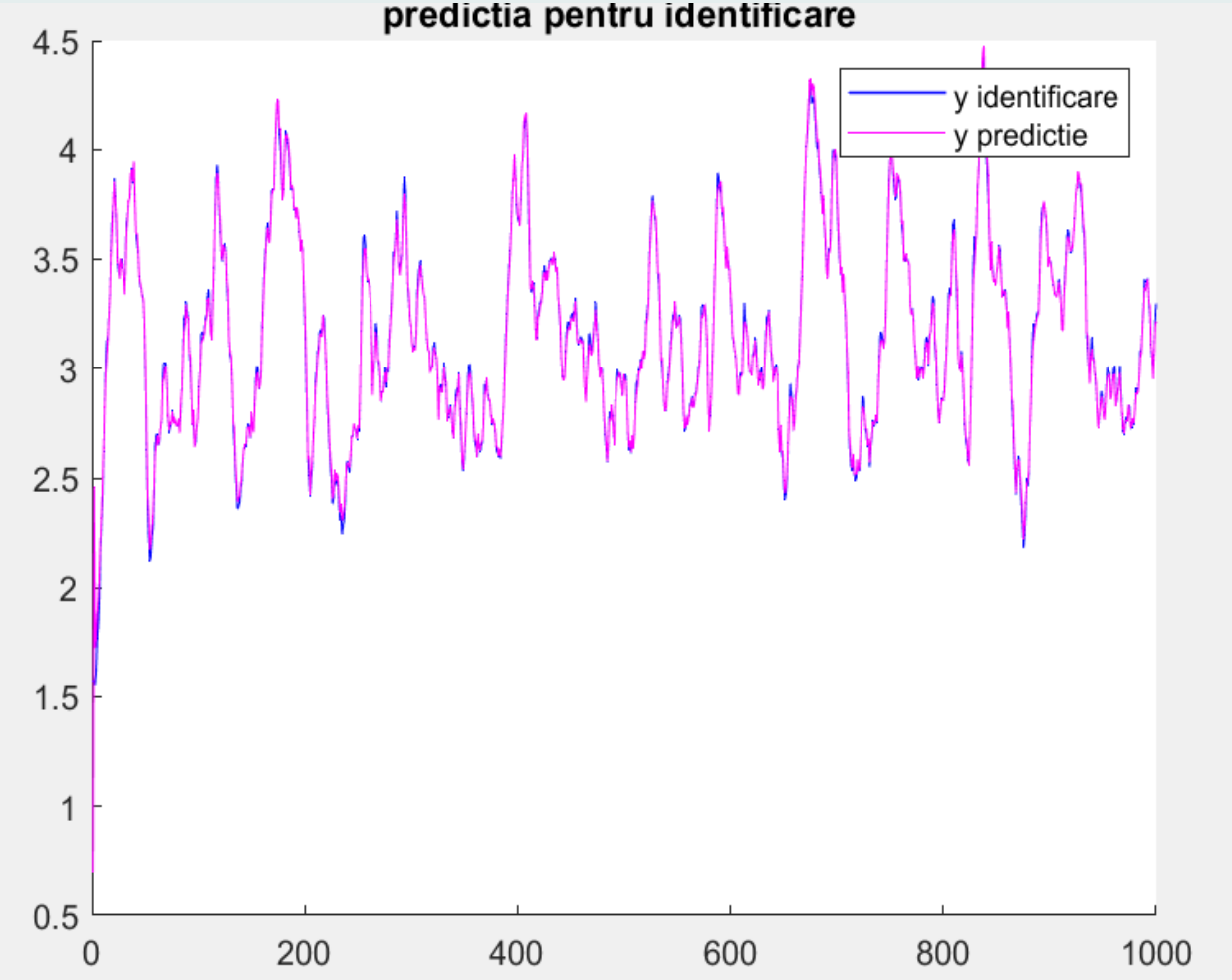


Fig 2.1 a)

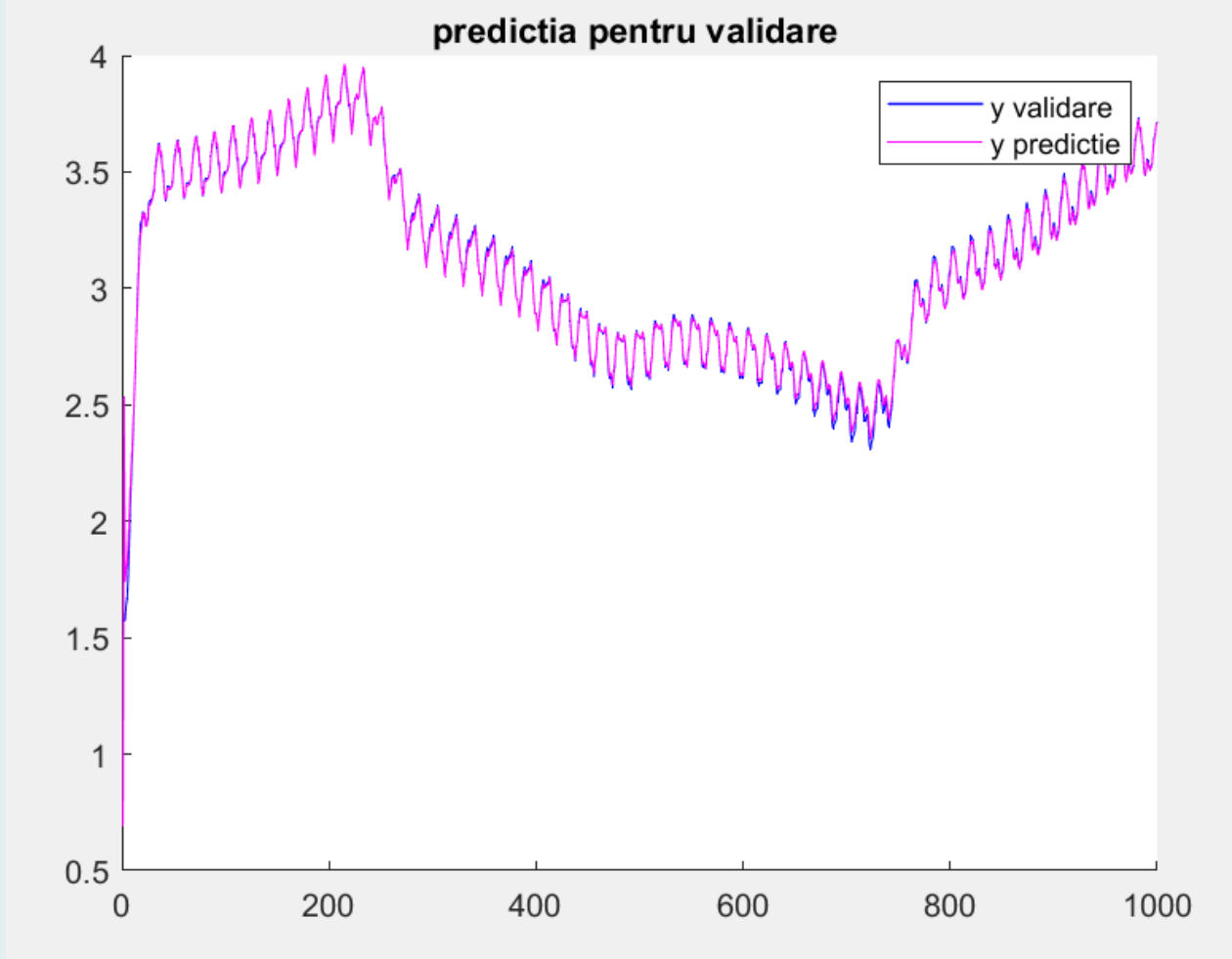


Fig 2.1 b)

FIGURA 2.2 na=2 nb=2 n=2

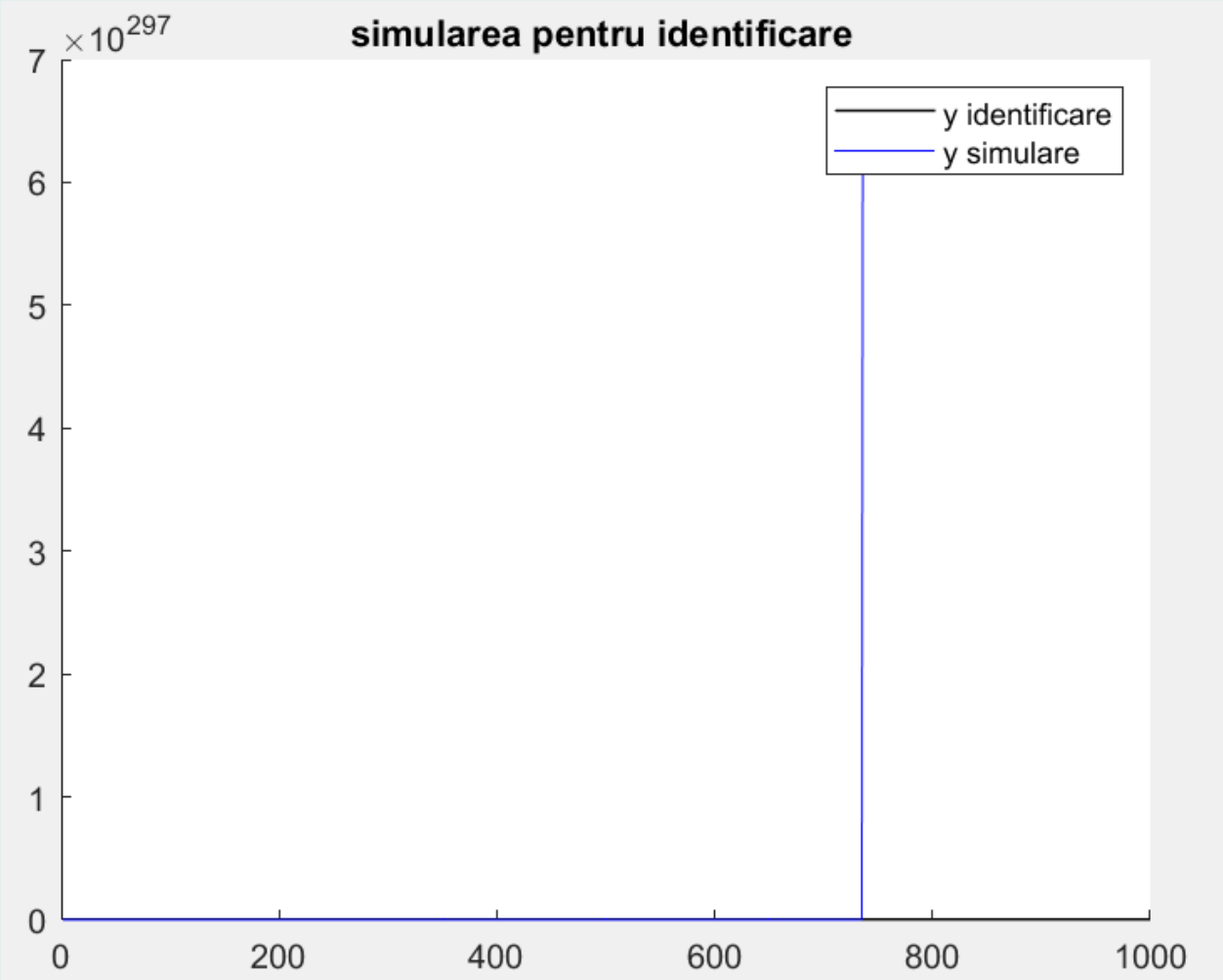


Fig 2.2 a)

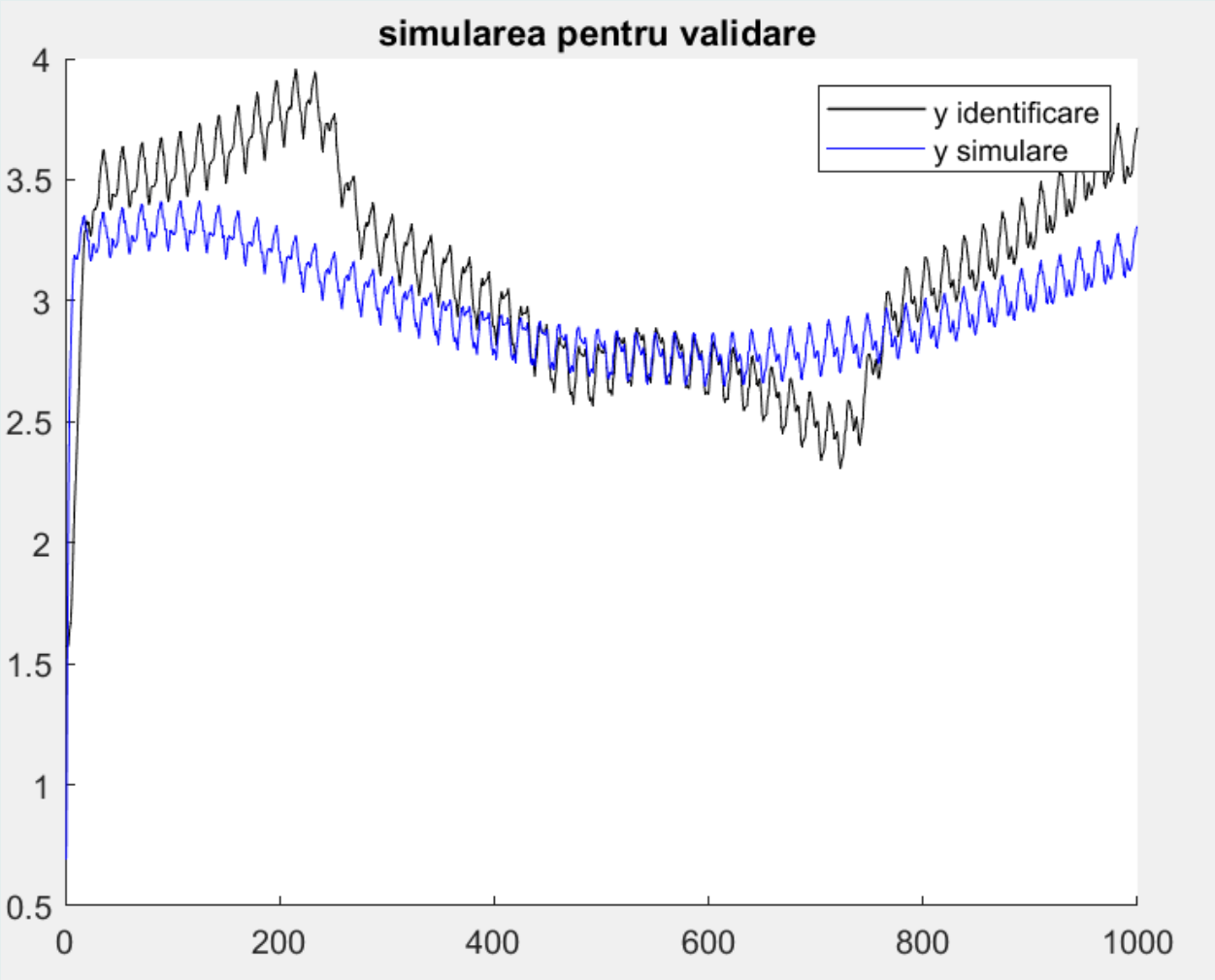


Fig 2.2 b)

FIGURA 3.1 na=1 nb=10 n=2

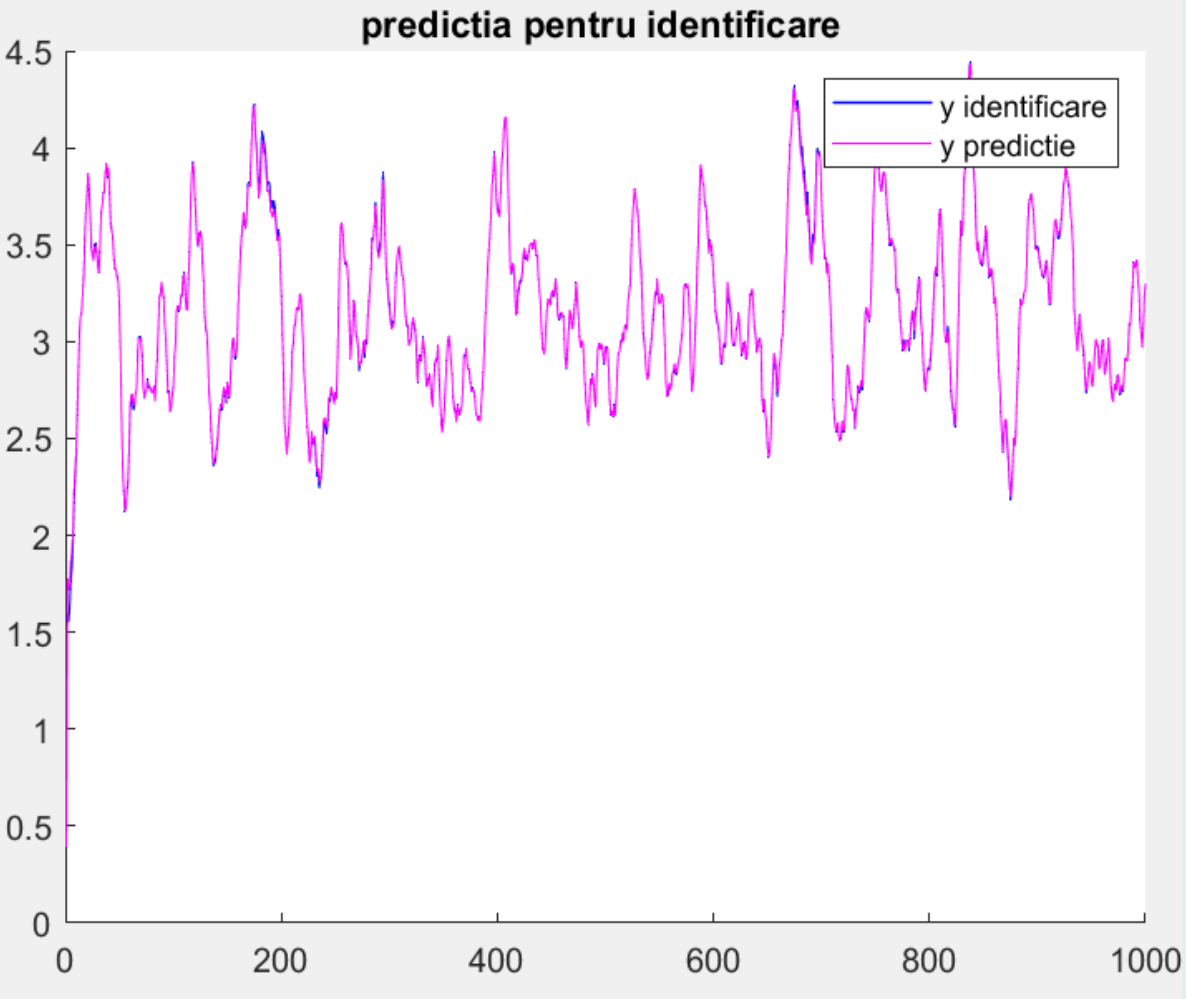


Fig 3.1 a)

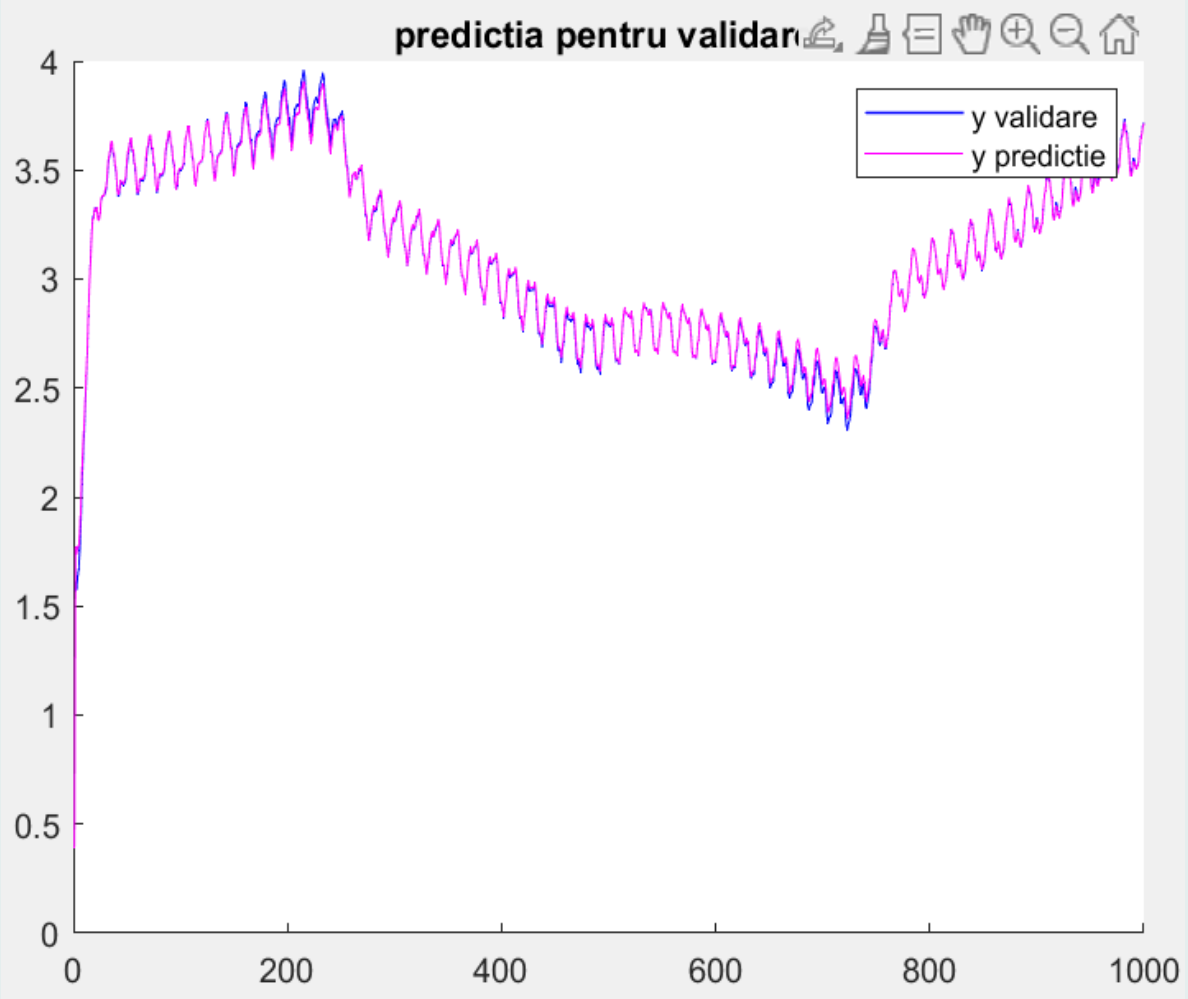


Fig 3.1 b)

FIGURA 3.2 na=1 nb=10 n=2

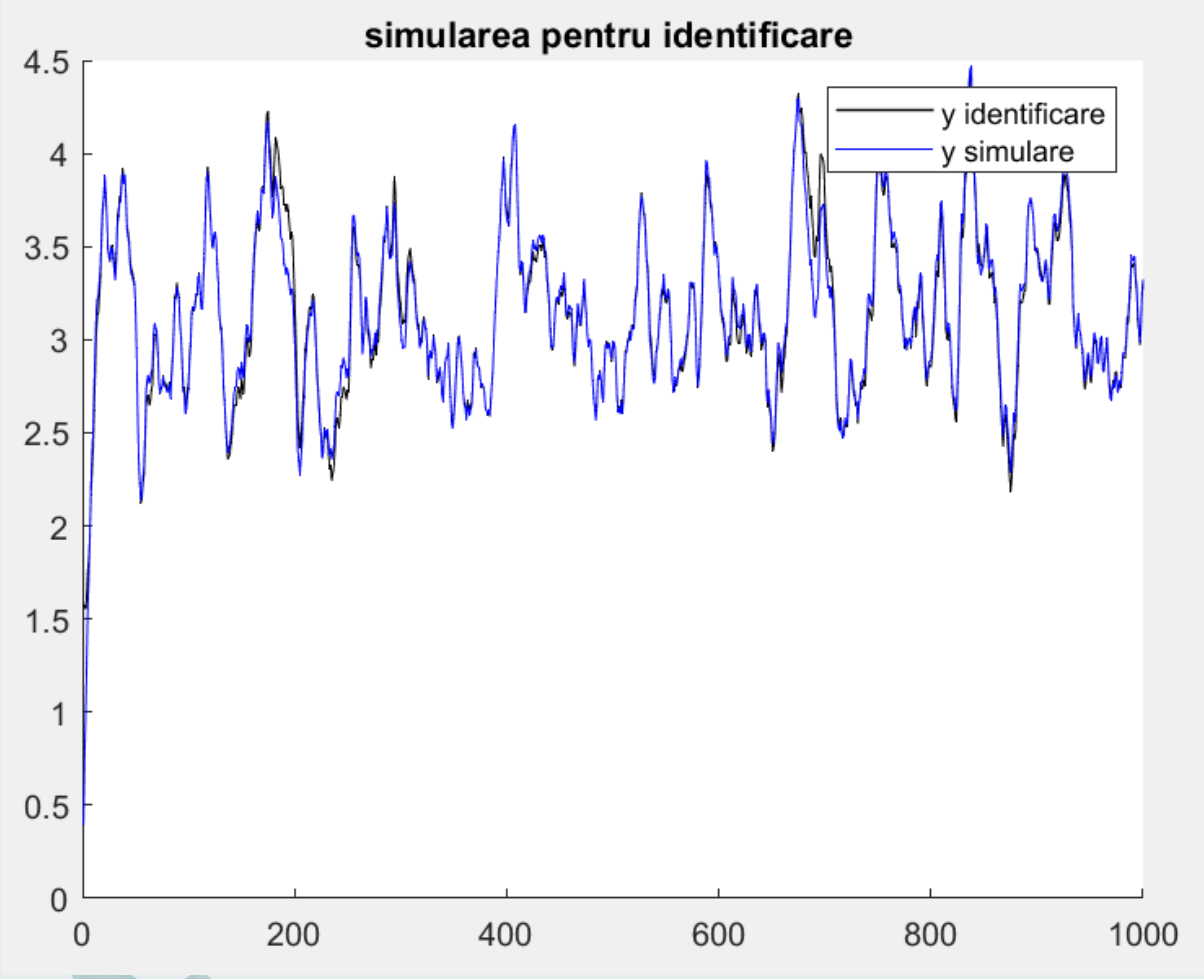


Fig 3.2 a)

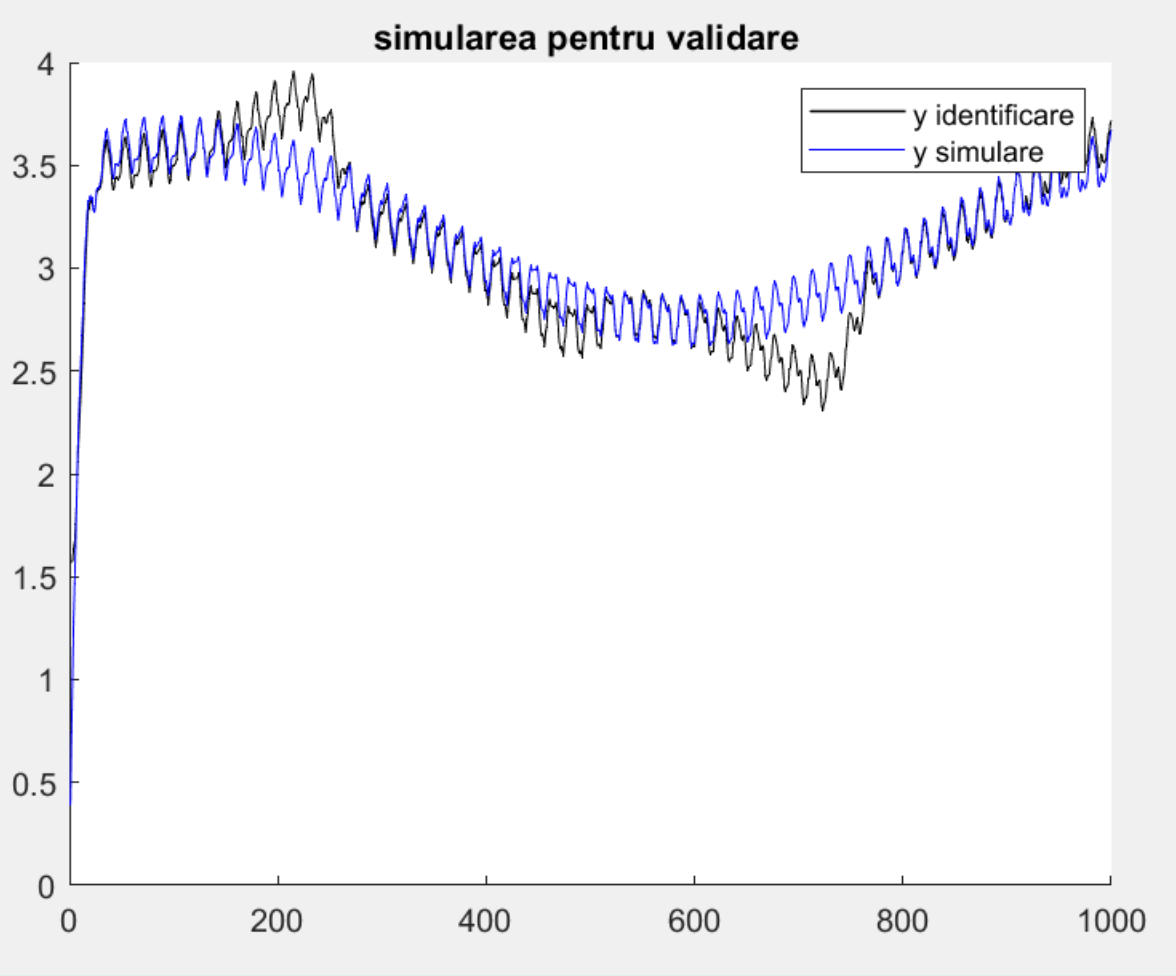


Fig 3.2 b)

Analiza valorilor MSE pentru diferite configuratii ale na nb si n

Grad n	na	nb	MSE predictie identificare	MSE predictie validare	MSE simulare identificare	MSE simulare validare
1	1	1	0.0106	0.0040	0.2467	0.2211
1	1	2	0.0057	0.0028	0.1277	0.0954
1	2	1	0.0081	0.0041	0.1992	0.1727
1	2	2	0.0031	0.0029	0.0333	0.0458
2	1	2	0.0057	0.0028	0.1306	0.0981
2	2	1	0.0073	0.0032	0.2348	0.2021
2	2	2	0.0023	0.0020	NaN	0.0878
2	**1**	**10**	**0.0017**	**0.0019**	**0.0084**	**0.0243**
2	2	8	7.7784e-04	9.0002e-04	0.0146	NaN
3	1	2	0.0057	0.0028	0.1300	0.0973
3	3	3	1.6718e-04	3.1412e-04	0.0207	0.0401
3	4	4	7.1288e-05	0.0232	NaN	NaN
3	1	11	0.0015	0.0020	0.0063	0.0271

Concluzii

Criteriile de alegere a gradului modelului NARX si ale ordinelor na si nb au fost:

- ✓o eroare medie patratica (MSE) cat mai mica
- ✓suprapunerea graficelor sa fie cat mai exacta

Observatii:

- Pentru cazul **na=nb=1 si n=2** prezentat in Fig 1.1 si Fig 1.2 se remarca:
  - ✓ In cazul predictiei o suprapunere adecvata a graficelor
  - ✓ In cazul simularii o discrepanta foarte mare intre grafice
- Pentru cazul **na=2, nb=2 si n=2** prezentat in Fig 2.1 si Fig 2.2 se pot sesiza:
  - ✓ Pentru predictie o suprapunere potrivita
  - ✓ Pentru simularea realizata cu datele de identificare o sensibilitate ridicata =>MSE nedefinit

Modelul **optim** identificat este pentru configuratia **na=1, nb=10 si n =2.** (Fig 3.1 si Fig 3.2)

Anexa: Cod

```
clc
close all

%Incarcare date
load('iddata-16.mat');

%Initializare
na=1; %ordinul iesirii
nb=1; %ordinul intrarii
n=2; %gradul polinomului

%Date de identificare
u_identificare=id.InputData;
y_identificare=id.OutputData;

%Date de validare
u_validare=val.InputData;
y_validare=val.OutputData;
```

```
%Generarea COMBINATIILOR polinomiale
N=na+nb; %N=numarul total de varibile intarziate
combinatii=[]; %initializam o matrice ce va stoca combinatiile
% generate ale varibilelor intarziate
for m=0:n
    numar_total_elemente=N+m-1;
    %genereaza toate combinatiile posibile de m elemente dintr-un
    %numar de elemente din care se aleg combinatiile
    combiniatii_de_indici=nchoosek(1:numar_total_elemente,m);

    %numar de combinatii generate
    numar_de_combinatii=size(combiniatii_de_indici,1);

    %initalizare matrice de combinatii complete
    combinatii_complete=zeros(numar_de_combinatii,N);

    %copiaza combinatiile in primele m coloane ale matricei
    combinatii_complete(:,1:m)=combiniatii_de_indici;

    %adaugam combinatiile complete la matricea gloabala de combinatii
    combinatii=[combinatii;combinatii_complete];
end
%Verificam numarul combinatiilor generate
numar_combinatii_generate=length(combinatii);
disp('Numar total de combinatii generate: ')
disp(numar_combinatii_generate)
```

```
%Construim matricea de regresori pentru DATELE DE IDENTIFICARE
N_identificare=length(u_identificare);
matrice_regresori_identificare=zeros(N_identificare,numar_combinatii_generate);

for k=1:N_identificare

    %construim vectorul d(k) cu valorile de iesire intarziate y si valorile de intrare intarziate u
    d_k=zeros(N,1);%vector coloana

    %termenii de iesire -y(k-1)-y(k-2)-.....-y(k-na)
    for i=1:na
        if (k-i)>0 %verificare daca indexul i este valid
            d_k(i)=y_identificare(k-i);%atribuim valorile de iesire intaziata in vectorul d_k
        end
    end

    %termenii de intrare u(k-1)-u(k-2)+....+u(k-nb)
    for i=1:nb
        if (k-i)>0%verificare daca indexul i este valid
            d_k(na+i)=u_identificare(k-i);%atribuim valorile de intrare intarziata in vectorul d_k
        end
    end
    %calcularea termeniilor polinomiali
    for i=1:numar_combinatii_generate %pacurgem fiecare combinatie posibila

        termen=1;%varibila care va stoca produsul termeniilor pentru fiecare combinatie
        for j=1:N %parcugem fiecare element dintr-o combinatie
            %verificam daca combinatii(i,j) reprezinta un index valid pentru vectorul d_k
            if combinatii(i,j)>0  &&  combinatii(i,j) <=N
                %inmultim valoarea curenta termen cu valoarea din d_k la pozita data de combinatii(i,j)
                termen=termen*d_k(combinatii(i,j));
            end
        end
        %pentru punctul de date k si combinatia i atribuim produsul termenilor calculat
        %pentru combinatia curenta in matricea de regresori la pozitia  corespunzatoare
        matrice_regresori_identificare(k,i)=termen;
    end
end

%Identificarea parametriilor modelului
teta=matrice_regresori_identificare\y_identificare;

%Iesirea de PREDICTIE pentru identificare
y_predictie_identificare=matrice_regresori_identificare*teta;
```

```
%Construim matricea de regresori pentru DATELE DE VALIDARE
N_validare=length(u_validare);
matrice_regresori_validare=zeros(N_validare,numar_combinatii_generate);

for k=1:N_validare

    %construim vectorul d(k) cu valorile de iesire intarziate y si valorile de intrare intarziate u
    d_k=zeros(N,1);%vector coloana

    %termenii de iesire -y(k-1)-y(k-2)-....-y(k-na)
    for i=1:na
        if (k-i)>0 %verificare daca indexul i este valid
            d_k(i)=y_validare(k-i);%atribuim valorile de iesire intaziata in vectorul d_k
        end
    end

    %termenii de intrare u(k-1)-u(k-2)+....+u(k-nb)
    for i=1:nb
        if (k-i)>0 %verificare daca indexul i este valid
            d_k(na+i)=u_validare(k-i);%atribuim valorile de intrare intarziata in vectorul d_k
        end
    end
                                                    %calcularea termeniilor polinomiali
                                                    for i=1:numar_combinatii_generate %pacurgem fiecare combinatie posibila

                                                        termen=1;%variabila care va stoca produsul termeniilor pentru fiecare combinatie
                                                        for j=1:N %parcugem fiecare element dintr-o combinatie

                                                            %verificam daca combinatii(i,j) reprezinta un index valid pentru vectorul d_k
                                                            if combinatii(i,j)>0  &&  combinatii(i,j) <=N
                                                                %inmultim valoarea curenta termen cu valoarea din d_k la pozita data de combinatii(i,j)
                                                                termen=termen*d_k(combinatii(i,j));
                                                            end
                                                        end
                                                        %pentru punctul de date k si combinatia i atribuim produsul termenilor calculat
                                                        %pentru combinatia curenta in matricea de regresori la pozitia corespunzatoare
                                                        matrice_regresori_validare(k,i)=termen;

                                                    end
                                                end

                                                %Iesirea de PREDICTIE pentru validare
                                                y_predictie_validare=matrice_regresori_validare*teta;
```

```
%SIMULAREA pentru DATELE DE IDENTIFICARE
y_simulare_identificare=zeros(N_identificare,1);

for k=1:N_identificare

    %construim vectorul d(k) cu valorile simulate de isire si intrari intarziate
    d_k=zeros(N,1);

    %termenii de iesire -y(k-1)-y(k-2)-....-y(k-na)
    for i=1:na
        if (k-i)>0 %verificare daca indexul i este valid
            d_k(i)=y_simulare_identificare(k-i);%atribuim valorile de iesire simulate in vectorul d_k
        end
    end

     %termenii de intrare u(k-1)-u(k-2)+....+u(k-nb)
     for i=1:nb
        if (k-i)>0%verificare daca indexul i este valid
            d_k(na+i)=u_identificare(k-i);%atribuim valorile de intrare intaziata in vectorul d_k
        end
     end
     matrice_regresori_simulare_id=zeros(1,numar_combinatii_generate);
    %calcularea termeniilor polinomiali
    for i=1:numar_combinatii_generate %pacurgem fiecare combinatie posibila

        termen=1;%varibila care va stoca produsul termeniilor pentru fiecare combinatie
        for j=1:N %parcugem fiecare element dintr-o combinatie
            %verificam daca combinatii(i,j) reprezinta un index valid pentru vectorul d_k
            if combinatii(i,j)>0  &&  combinatii(i,j) <=N
                %inmultim valoarea curenta termen cu valoarea din d_k la pozita data de combinatii(i,j)
                termen=termen*d_k(combinatii(i,j));
            end
        end
        %stocam rezultatul in matricea de regresori
        matrice_regresori_simulare_id(i)=termen;
    end
    %calculam iesirea simulata pentru identificare
    y_simulare_identificare(k)=matrice_regresori_simulare_id*teta;
end
```

```
%SIMULAREA pentru DATELE DE VALIDARE
y_simulare_validare=zeros(N_validare,1);

for k=1:N_validare

    %construim vectorul d(k) cu valorile simulate de isire si intrari intarziate
    d_k=zeros(N,1);

    %termenii de iesire -y(k-1)-y(k-2)-....-y(k-na)
    for i=1:na
        if (k-i)>0 %verificare daca indexul i este valid
            d_k(i)=y_simulare_validare(k-i); %atribuim valorile de iesire simulate in vectorul d_k
        end
    end

    %termenii de intrare u(k-1)-u(k-2)+....+u(k-nb)
    for i=1:nb
        if (k-i)>0 %verificare daca indexul i este valid
            d_k(na+i)=u_validare(k-i); %atribuim valorile de intrare intaziata in vectorul d_k
        end
    end

    matrice_regresori_simulare_val=zeros(1,numar_combinatii_generate);
    %calcularea termeniilor polinomiali
    for i=1:numar_combinatii_generate %pacurgem fiecare combinatie posibila

        termen=1; %varibila care va stoca produsul termeniilor pentru fiecare combinatie
        for j=1:N %parcugem fiecare element dintr-o combinatie
            %verificam daca combinatii(i,j) reprezinta un index valid pentru vectorul d_k
            if combinatii(i,j)>0  &&  combinatii(i,j) <=N
                %inmultim valoarea curenta termen cu valoarea din d_k la pozita data de combinatii(i,j)
                termen=termen*d_k(combinatii(i,j));
            end
        end
        %stocam rezultatul in matricea de regresori
        matrice_regresori_simulare_val(i)=termen;

    end
    %calculam iesirea simulata pentru validare
    y_simulare_validare(k)=matrice_regresori_simulare_val*teta;
end
```

```
%Vizualizarea predictiei
figure
hold on
plot(y_identificare,'blue','LineWidth',0.7)
plot(y_predictie_identificare,'magenta')
title('predictia pentru identificare ')
legend('y identificare' , 'y predictie')


figure
hold on
plot(y_validare,'blue','LineWidth',0.7)
plot(y_predictie_validare,'magenta')
title('predictia pentru validare ')
legend('y validare' , 'y predictie')
```

```
%Eroarea medie patratica de predictie pentru datele de identificare
eroare_id_predictie=0;
for i=1:N_identificare
    eroare_id_predictie=eroare_id_predictie+(y_identificare(i)-y_predictie_identificare(i))^2;
end
eroare_id_predictie=eroare_id_predictie/N_identificare;


%Eroarea medie patratica de predictie pentru datele de validare
eroare_val_predictie=0;
for i=1:N_validare
   eroare_val_predictie= eroare_val_predictie+(y_validare(i)-y_predictie_validare(i))^2;
end
 eroare_val_predictie= eroare_val_predictie/N_validare;
```

```
%Vizualizarea simularii
figure
hold on
plot(y_identificare,'black','LineWidth' ,0.7)
plot(y_simulare_identificare,'blue')
title('simularea pentru identificare ')
legend('y identificare' , 'y simulare')


figure
hold on
plot(y_validare,'black','LineWidth',0.7)
plot(y_simulare_validare,'blue')
title('simularea pentru validare ')
legend('y identificare' , 'y simulare')
```

```
%Eroarea medie patratica de simulare pentru datele de identificare
eroare_id_simulare=0;
for i=1:N_identificare
    eroare_id_simulare=eroare_id_simulare+(y_identificare(i)-y_simulare_identificare(i))^2;
end
eroare_id_simulare=eroare_id_simulare/N_identificare;


%Eroarea medie patratica de simulare pentru datele de identificare
eroare_val_simulare=0;
for i=1:N_validare
    eroare_val_simulare=eroare_val_simulare+(y_validare(i)-y_simulare_validare(i))^2;
end
eroare_val_simulare=eroare_val_simulare/N_validare;
```

```
disp("Eroarea medie de predictie pentru identificare: ")
disp(eroare_id_predictie)
disp("Eroarea medie de predictie pentru validare: ")
disp(eroare_val_predictie)
disp("Eroarea medie de simulare pentru identificare: ")
disp(eroare_id_simulare)
disp("Eroarea medie de simulare pentru validare: ")
disp(eroare_val_simulare)
```

```
%Modelul NARX cu functie pentru comparatie
date_identificare=iddata(y_identificare,u_identificare);
date_validare=iddata(y_validare,u_validare);

model_id=nlarx(date_identificare,[na nb 1]);
model_val=nlarx(date_validare,[na nb 1]);

figure
subplot(2,1,1)
compare(date_identificare,model_id)
subplot(2,1,2)
compare(date_validare,model_val)
```

Va multumim!