ARX neliniar

Zigler Alexandru

December 2021

Cuprins

Introducere: Metoda ARX

ARX liniar:

$$y(k) = \left[-y(k-1), -y(k-2), \ldots, -y(k-na), u(k-1), u(k-2), \ldots, u(k-nb)\right] \cdot \theta + e(k)$$

Ieșirea la pasul k depinde liniar de ieșirile și intrările precedente. θ este vectorul de parametri, iar $e(k)$ reprezintă zgomotul.

ARX neliniar:

$$y(k) = g(y(k-1), y(k-2), \ldots, y(k-na), u(k-1), u(k-2), \ldots, u(k-nb)) + e(k)$$

Polinomul g are $na + nb$ variabile - ieșirile și intrările anterioare, iar gradul polinomului este m. Coeficienții lui g sunt parametrii θ.

Descrierea problemei

Avem la dispoziție un set de date de identificare - *Fig. 1a*. Cu ajutorul lui putem determina modele NARX, în funcție de valorile alese pentru na, nb și m (opțional nk).

Modelul obținut se validează utilizând al doilea set de date - *Fig. 1b*. Atât la pasul de identificare, cât și la cel de validare, modelul se folosește pentru predicție și pentru simulare.

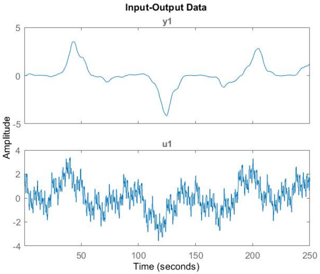


Figure: 1a Identificare

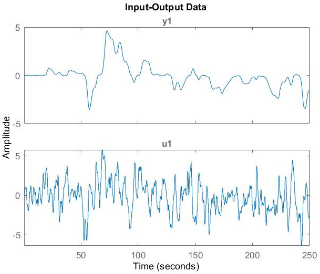


Figure: 1b Validare

Obiective

- proiectarea unui cod cu na, nb, m - configurabile;
- realizarea graficelor de predicție și simulare și compararea acestora cu seturile de date inițiale (id & val);
- calcularea erorilor medii pătratice ;
- pentru na,nb și m în anumite; intervale[1], comparăm performanțele de aproximare;
- realizarea unor grafice cu evoluția erorilor în funcție de na, nb, m;
- alegem cel mai bun model dintre cele considerate mai sus, minimizând eroarea medie pătratică;

[1] $na \leq 5, nb \leq 5, m \leq 5$ și $na = nb$

Structură: Formă polinomială

$$x(k) = \left[y(k-1), y(k-2), \dots, y(k-na), u(k-1), u(k-2), \dots, u(k-nb)\right]$$

Rescriem

$$x(k) = \left[x_1(k), x_2(k), \dots, x_{na}(k), x_{na+1}(k), x_{na+2}(k), \dots, x_N(k)\right], N = na+nb$$

$$g(x(k)) = \theta_1 \cdot 1 + \theta_2 \cdot x_1 + \theta_3 \cdot x_2 + \dots + \theta_{N+1} \cdot x_N +$$
$$+\theta_{N+2} \cdot x_1^2 + \theta_{N+3} \cdot x_2^2 + \dots + \theta_{2N+1} \cdot x_N^2 + \theta_{2N+2} \cdot x_1 x_2 + \theta_{2N+3} \cdot x_1 x_3 + \dots +$$
$$+\theta_{3N} x_1 x_N + \dots + \theta_? \cdot x_{N-1} x_N + \dots$$

Formă matriceală

$$\begin{bmatrix} 1 & x_1(1) & x_2(1) & \dots & x_N(1) & \dots \\ 1 & x_1(2) & x_2(2) & \dots & x_N(2) & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \dots \\ 1 & x_1(n) & x_2(n) & \dots & x_N(n) & \dots \end{bmatrix} \cdot \theta = \begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(n) \end{bmatrix}$$

$\Phi \cdot \theta = Y$,unde n - lungimea setului de identificare
Pentru a construi matricea ϕ, generăm toate combinațiile
$\{p_1(j), p_2(j), \dots, p_N(j)\}, \quad \sum_{i=1}^{N} p_i \le m, \quad 0 \le p_i \le m,$
unde j - indexul combinației

Notăm linia k din Φ cu $\left[\varphi_1(k), \varphi_2(k), \dots\right]$.

Atunci $\varphi_j(k) = \prod_{i=1}^{N} x_i(k)^{p_i(j)}$.

Implementare

- "proiect2arx.m"
- "generate_generalized_v1.m"
- "generate_generalized_v2.m"
- "mlen.m"
- "thesearch.m"

Predicție și simulare

Construim matricea X cu vectorii $x(k)$, $k = \overline{1, n}$

$$x(k) = [y(k-1), y(k-2), \ldots, y(k-na), u(k-1), u(k-2), \ldots, u(k-nb)]$$

Se construiește matricea Φ linie cu line în funcție de linia corespunzătoare din X. Se folosește una din funcțiile de generare.
Se determină parametrii θ prin regresie liniară.
Se calculează matricele X și Φ pentru setul de validare.
Se validează modelul cu θ identificat inițial.

Pentru simulare, matricea X se va construi linie cu linie. Se folosesc ieșirile simulate la pașii anteriori în locul celor reale. Inițial,vectorul y este nul (condiții inițiale 0). El se actualizează la fiecare linie din X.

$$\hat{y}_{sim}(k) = \varphi(k) \cdot \theta$$

unde $\varphi(k)$ reprezintă regresorii corespunzători liniei k din X.

Numărul regresorilor

Combinări cu repetiție

$$\left(\!\binom{n}{k}\!\right) = C_{n+k-1}^{k} = \binom{n+k-1}{k} = \frac{(n+k-1)!}{k!(n-1)!}$$

Exemplu Avem 2 seturi de obiecte: de tip A și de tip B. Vrem să alegem 3 obiecte. În câte moduri se pot alege?

3 x obiecte A + 0 x obiecte B
2 x obiecte A + 1 x obiect B
1 x obiect A + 2 x obiecte B
0 x obiecte A + 3 x obiecte B

$$\left(\!\binom{2}{3}\!\right) = \binom{3+2-1}{3} = \frac{4!}{3!\cdot 1!} = 4$$

Funcția "mlen.m" parcurge numerele de la 0 la m (fixat) și calculează numărul total de regresori aplicând această formulă la fiecare iterație.

Generare regresori metoda 1

$f : \{1, 2, \dots, N\} \longrightarrow \{0, 1, \dots, m\}$

Există $(m+1)^N$ astfel de funcții. Deci fiecărui număr natural de la 1 la $(m+1)^N$ îi corespunde o funcție f.

x=239		
	x%10	9
	x/10%10	3
	x/10^2%10	2

Figure: Extragerea cifrelor dintr-un număr în baza 10

Pornim de la conceptul de extragere a cifrelor dintr-un număr în baza 10 și îl adaptăm pentru baza m+1.

$f_k(i) = (k-1)/(m+1)^{(N-i)}\%(m+1)$

$k = \overline{1, (\mathrm{m}+1)^{\mathrm{N}}}$ $\qquad N - i = \overline{0, \mathrm{N}-1}$

Reținem într-o matrice doar funcțiile f_k care au suma valorilor mai mică sau egală cu m.

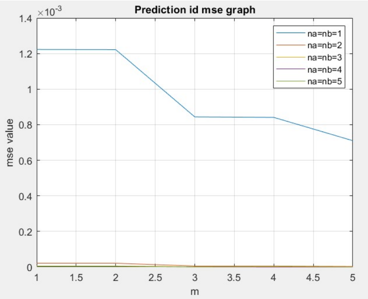
Generare regresori metoda 2

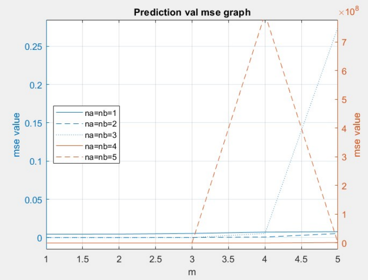
Considerăm 2 vectori de lungime N.
-vector1:inițializat cu 0, în el se generează linia curentă cu puteri
-vector2: vector boolean, inițializat cu 0; o valoare din vector2 devine 1 în momentul în care valoarea corespunzătoare din vector1 este egală cu m
La fiecare iterație, se incrementează ultima poziție din vector1. Când avem valoarea m pe ultima poziție, valoarea corespunzătoare din vector2 devine 1. Se resetează ultima poziție și se incrementează poziția din stânga.
O poziție k se incrementează în momentul în care toate pozițiile din dreapta au valoarea 1 în vector2.
Se rețin doar vectorii "vector1" care au suma elementelor mai mică sau egală cu m.
Algoritmul se oprește când toate valorile din vector2 sunt 1.

Observație: Metoda 1 este mai eficientă.

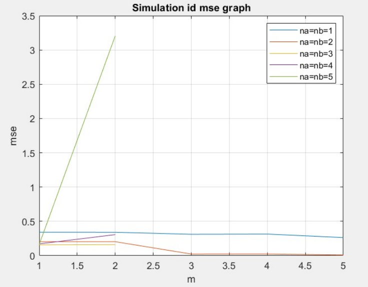
Rezultate de acordare

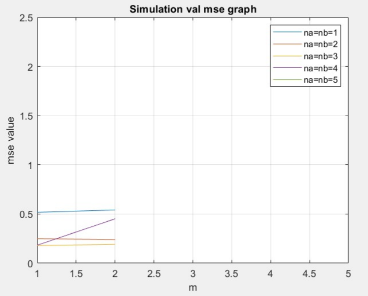
Impunem na,nb, m în intervale de la 1 la 5 pentru a obține modelul cel mai bun (mse minim). Păstrăm valorile mse în 4 matrici (id predicție, val predicție, id simulare, val simulare). Reprezentăm grafic evoluția mse în funcție de m,na,nb.





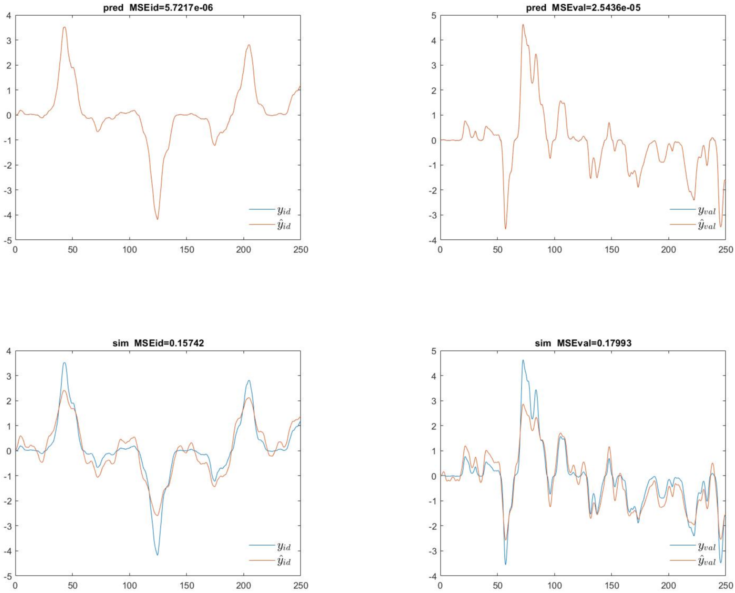
Rezultate de acordare





Observăm că pentru graficele de simulare avem valori NaN. Ele apar deoarece valoarea ieșirii simulate ajunge la valori foarte mari.

Modelul optim: na=nb=3,m=1



Concluzii

- în cazul predicției, erorile de identificare scad pe măsură ce mărim variabilele m,na,nb, rezultat la care ne așteptam;
- erorile de simulare sunt inițial mici(< 1); când eroarea depășește valoarea 1, urmează o creștere exponențială; după câteva eșantioane se ajunge la NaN; acest fenomen se întâmplă pentru m suficient de mare;
- modelul optim are eroarea medie pătratică la predicție $< 10^{-5}$; în cazul simulării eroarea este < 0.2 și observăm că semnalele $\hat{y}_{id,val}$ au formă asemănătoare cu $y_{id,val}$;

În concluzie, am reușit să îndeplinim toate obiectivele propuse. Am găsit un model optim, cu performanțe bune atât la predicție cât și la simulare.