Identificarea Sistemelor – Laborator 10
Identificarea ARX recursivă

Organizare

Acest laborator se rezolvă independent de către fiecare student. Doar în situaţia în care există mai mulţi studenţi decât calculatoare, *cu acordul explicit al profesorului de laborator pentru fiecare grup*, studenţii se pot grupa câte doi la un calculator.

Soluţia constă din cod Matlab. Dezvoltaţi acest cod într-un singur script Matlab. Dacă aveţi nevoie de funcţii, acestea pot fi locale în script, vezi funcţii locale în scripturi Matlab.

Regulile generale pentru laboratoare sunt descrise pe site. Pentru fiecare laborator, prezenţa dvs. va fi înregistrată numai dacă aveţi o soluţie originală şi funcţională. Profesorul va verifica **funcţionalitatea** codului dvs. timpul orei de laborator. Doar după aceea, pentru verificarea **originalităţii**, încărcaţi soluţia dvs. aici:

`https://www.dropbox.com/request/ZcBECRE71pJ9nfH6QBNd`

Încărcaţi *o singură data, un singur fişier .m, denumit exact după următorul model*:

`L10_ROgs_NumePrenume.m`

unde g este grupa, s semigrupa, urmate de numele şi prenumele dvs. De exemplu, `L10_RO31_PopAlex.m`. Dacă aţi lucrat în grup cf. procedurii de mai sus, încărcaţi un singur fişier cu ambele nume ale studenţilor din grup. Fişierele duplicate, nonstandard, denumite în mod necorespunzător sau care corespund unor soluţii necontrolate încă de profesorul de laborator nu vor fi luate în considerare. Fişierele vor fi testate automat pentru plagiat, iar orice soluţie care nu trece acest test va fi marcată copiată; doar soluţiile care trec atât testul de funcţionalitate, cât şi pe cel de originalitate, sunt validate definitiv. Prin urmare, chiar dacă sunteţi încurajaţi să discutaţi idei şi algoritmi între colegi, trimiterea şi împrumutarea unor pasaje de cod este strict interzisă.

Descrierea laboratorului

Vom studia în acest laborator varianta recursivă a metodei ARX, vezi cursul *Identificarea recursivă*.

Fiecărui student i se alocă de către profesor un index pentru setul de date. Apoi, studentul descarcă fişierul Matlab ce formează baza laboratorului de pe pagina cursului. Fişierul conţine datele de identificare în variabila `id`, şi separat datele de validare în variabila `val`.

Se ştie în avans că ordinul sistemului este n, dat în variabila `n` din fişierul de date; că sistemul are o structură de tip eroare de ieşire, OE; şi că nu are timp mort. Pentru a compensa nepotrivirea cu structura ARX vom lua ordine mai mari pentru modelele ARX pe care le vom căuta. Recomandarea este să alegeţi $na = nb = 3 \cdot n$.

- Implementaţi algoritmul ARX recursiv într-o funcţie care primeşte la intrare setul de date de identificare, ordinele modelului na şi nb, vectorul iniţial de parametri $\theta(0)$, şi matricea inversă iniţială $P^{-1}(0)$. Funcţia trebuie să producă la ieşire o matrice $\Theta \in \mathbb{R}^{N \times (na+nb)}$ conţinând pe fiecare linie k vectorul de parametri $\theta(k)$: întâi coeficienţii $a_1, \ldots, a_{na}$ ai polinomului A, şi apoi coeficienţii $b_1, \ldots, b_{nb}$ din B (acest format este compatibil cu cel al funcţiei Matlab deja existente, aşadar rezultatele celor două funcţii vor fi mai uşor de comparat). Un pseudocod cu mai multe detalii decât cel prezentat la curs este inclus mai jos.

- Rulaţi identificarea ARX recursivă folosind funcţia dvs., pe datele de identificare, pornind de la o inversă iniţială $P^{-1}(0) = \frac{1}{\delta} I_{na+nb} = 100 I_{na+nb}$ (deci $\delta = 0.01$). Comparaţi *pe datele de validare* calitatea celor două modele: unul cu parametrii finali găsiţi după procesarea întregului set de date; şi altul după 10% din date. Care model este mai bun, şi de ce?

- Repetaţi experimentul, dar de această dată cu $P^{-1}(0) = \frac{1}{\delta} I_{na+nb} = 0.01 I_{na+nb}$ (so $\delta = 100$). Gândiţi-vă la rezultate. Pentru care valoare a lui δ este mai imprecis modelul iniţial, şi de ce?

- Opţional, dacă aveţi timp, repetaţi experimentul iniţial, dar de această dată cu funcţia `rarx` deja existentă în Matlab. Comparaţi rezultatele produse de această funcţie (de ex., modelele de la 10% şi 100% din date) cu cele ale funcţiei dvs., verificând dacă sunt la fel sau similare.

1: iniţializează $\widehat{\theta}(0)$ (vector coloană $na + nb$), $P^{-1}(0)$ (matrice $(na + nb) \times (na + nb)$)
2: **loop** la fiecare pas $k = 1, 2, \ldots$
3: extrage $u(k)$, $y(k)$
4: formează vectorul de regresori ARX:
$$\varphi(k) = [-y(k-1), \cdots, -y(k-na), u(k-1), \cdots, u(k-nb)]^\top$$
5: găseşte eroarea de predicţie $\varepsilon(k) = y(k) - \varphi^\top(k)\widehat{\theta}(k-1)$ (un scalar)
6: actualizează inversa: $P^{-1}(k) = P^{-1}(k-1) - \frac{P^{-1}(k-1)\varphi(k)\varphi^\top(k)P^{-1}(k-1)}{1+\varphi^\top(k)P^{-1}(k-1)\varphi(k)}$
7: calculează ponderile: $W(k) = P^{-1}(k)\varphi(k)$ (vector coloană $(na + nb)$)
8: actualizează parametrii: $\widehat{\theta}(k) = \widehat{\theta}(k-1) + W(k)\varepsilon(k)$
9: **end loop**

Funcţii relevante din toolbox-ul de identificare a sistemelor: `rarx`, `idpoly`, `compare`. Indicii adiţionale:

- După ce aveţi polinoamele A şi B ca vectori de coeficienţi în ordinea crescătoare a puterilor lui q^{-1}, folosiţi `idpoly(A, B, [], [], [], 0, Ts)` pentru a genera modelul ARX, unde `Ts` este perioada de eşantionare. Nu uitaţi că toţi vectorii de coeficienţi din polinoame trebuie să conţină întotdeauna coeficienţii constanţi (puterea 0 a argumentului q^{-1}), care trebuie să fie 1 în A, şi 0 în B. Ţineţi cont că matricea de parametri returnată de algoritm *nu* conţine aceşti coeficienţi constanţi.

- Funcţia existentă `rarx` preia la intrare setul de date de identificare, ordinele modelului na şi nb şi întârzierea nk sub formă de vector, argumentele `'ff', 1` care configurează algoritmul la fel cu cel din curs, vectorul iniţial de parametri $\theta(0)$, şi matricea inversă iniţială $P^{-1}(0)$. Matricea numită P în documentaţia funcţiei Matlab `rarx` este de fapt matricea *inversă* P^{-1} din curs, fiţi aşadar atenţi când o alegeţi. Funcţia produce la ieşire o matrice Θ în acelaşi format cu cel explicat mai sus.