Proiect
Identificarea sistemelor 2017

Organizare

Acest proiect este parte obligatorie a cursului de Identificarea Sistemelor, seria 2, linia de licenţă Ingineria Sistemelor, Universitatea Tehnică din Cluj-Napoca. Nota la proiect are o pondere de 20% în nota finală a disciplinei. Proiectul va fi efectuat în grupuri de câte doi studenţi, şi rezolvarea sa necesită aproximativ 20 ore / student, depinzând şi de experienţa fiecăruia în MATLAB. Fiecare grup va primi seturile sale de date separate. Pentru a le primi, formaţi cât mai repede grupuri şi trimiteţi un email profesorului Lucian Buşoniu, la adresa `lucian@busoniu.net`. Menţionaţi numele celor doi membri ai grupului şi adresele de email ale ambilor membri. Soluţia proiectului va fi descrisă într-un scurt raport, scris în engleză sau în română. Un grup produce un singur raport.

Raportul va fi trimis prin email **în format PDF**, la adresa de mai sus. Nu uitaţi să menţionaţi numele studenţilor din grup pe pagina de titlu. Raportul trebuie să includă listing-uri complete ale codului MATLAB dezvoltat (funcţii şi scripturi). Codul trebuie de asemenea ataşat într-o arhivă ZIP aceluiaşi email în care trimiteţi raportul. Folosiţi o versiune de MATLAB mai nouă sau egală cu R2006b, şi menţionaţi versiunea folosită.

Termenul limită pentru raportul şi codul final este **23 decembrie 2017, cel târziu la 24:00**. În plus, o versiune intermediară conţinând raportul şi codul pentru prima problemă trebuie trimisă până pe **19 noiembrie 2017, 24:00**. În caz că un grup întârzie cu trimiterea soluţiei, fiecare nouă zi de întârziere începută duce la pierderea a două puncte din nota maximă la proiect (de exemplu, dacă raportul este livrat în 25 decembrie la ora 00:10AM, nota maximă devine 6 fiindcă a început deja a doua zi de întârziere). Este strict interzisă preluarea de cod sau de rezultate de la colegi, precum şi preluarea de text pentru raport de la colegi sau din alte surse cum ar fi website-uri etc. Încălcarea acestei reguli constituie plagiarism şi va duce la descalificarea soluţiei dvs. la proiect, şi implicit la pierderea dreptului la participare la examen.

Proiectul constă din două probleme, ambele implicând aproximatoare polinomiale. În prima problemă, polinomul este folosit pentru a modela comportamentul unei funcţii necunoscute, iar a doua problemă tratează identificarea de tip cutie neagră a unui sistem dinamic necunoscut.

Modelarea unei funcţii necunoscute

Se dă un set de date de intrare-ieşire, unde ieşirea este generată de o funcţie necunoscută, neliniară dar statică. Ieşirea este afectată de zgomot, pe care-l vom presupune aditiv, Gaussian, şi de medie zero. Funcţia are două variabile de intrare şi una de ieşire. Va trebui dezvoltat un model pentru această funcţie, folosind un aproximator polinomial. Un al doilea set de date, generat de aceeaşi funcţie, este furnizat pentru validarea modelului dezvoltat. Cele două seturi sunt furnizate într-un fişier de date MATLAB, conţinând câte o structură (tip de date MATLAB) pentru fiecare set. Structura pentru antrenarea modelului este numită `id`, iar cea pentru validare `val`. Fiecare din aceste structuri conţine următoarele câmpuri:

- O colecţie de coordonate `X` pentru intrări, unde `X` este un *cell array* conţinând doi vectori, fiecare vector `X{dim}` conţinând o grilă de n puncte pentru dimensiunea `dim` a intrării.
- O colecţie de ieşiri corespunzătoare `Y`, reprezentată printr-o matrice de dimensiunea $n \times n$, unde `Y(i,j)` este valoarea funcţiei f în punctul `(X{1}(i),X{2}(j))` (afectată de zgomot).

- Aceleaşi date sunt furnizate şi într-un format alternativ "plat": intrările `Xflat`, o matrice lată conţinând cele n^2 puncte de intrare, câte unul pe fiecare coloană, şi ieşirile corespunzătoare într-un vector linie `Yflat`.

Variantele"structurate" `X`, `Y` sunt mai uşor de folosit pentru crearea graficelor, iar cele "plate" sunt mai potrivite pentru antrenarea şi folosirea aproximatorului.

Aproximatorul polinomial g trebuie să aibă gradul m configurabil. De exemplu, pentru primele câteva valori ale lui m, aproximatorul are forma:

$$
\begin{aligned}
&m = 1, \quad \hat{g}(x) = [1, x_1, x_2] \cdot \theta = \theta_1 + \theta_2 x_1 + \theta_3 x_2 \\
&m = 2, \quad \hat{g}(x) = [1, x_1, x_2, x_1^2, x_2^2, x_1 x_2] \cdot \theta = \theta_1 + \theta_2 x_1 + \theta_3 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2 + \theta_6 x_1 x_2 \\
&m = 3, \quad \hat{g}(x) = [1, x_1, x_2, x_1^2, x_2^2, x_1^3, x_2^3, x_1 x_2, x_1^2 x_2, x_1 x_2^2] \cdot \theta
\end{aligned}
$$

unde primele două polinoame au fost scrise şi explicit pentru claritate. Pentru o mai bună stabilitate numerică, puteţi rescala domeniul variabilelor de intrare x_1, x_2 de la intervalele originale $[a_1, b_1]$, $[a_2, b_2]$ la intervalul standard $[-1, 1]$, operaţie pe care codul dvs. ar trebui să o trateze intern. Intervalele originale se pot găsi studiind domeniul datelor din set.

Odată ce gradul m este ales, antrenarea modelului constă din găsirea vectorului optim de parametri θ pentru care polinomul $g(x)$ se apropie cel mai mult de funcţia ţintă $f(x)$ pe setul de date de identificare – matematic, vectorul θ care minimizează suma (sau echivalent media) erorilor pătratice. Antrenarea se poate face cu metoda regresiei liniare, ţinând cont de faptul că g este liniar în parametri (chiar dacă este neliniar în variabilele de intrare x_1, x_2). Detaliile despre această metodă se găsesc în curs, Partea 3: *Baze matematice*, secţiunile despre regresia liniară. Regresorii sunt în cazul nostru poliomial puterile lui x_1 şi x_2, de ex. pentru $m = 2$ ei sunt 1, x_1, x_2, x_1^2, x_2^2, $x_1 x_2$.

Cerinţele pentru prima problemă sunt următoarele. Programaţi un aproximator polinomial cu grad configurabil. Încercaţi să antrenaţi aproximatoare de diverse grade, pentru a-l obţine pe cel mai precis. Validarea trebuie efectuată permanent pe setul diferit de date pentru validare. Raportaţi eroarea medie pătratică pe ambele seturi de date (identificare şi validare), şi reprezentaţi grafic rezultatul dat de aproximator, comparat cu valorile reale ale funcţiei, pe cele două seturi de date. Discutaţi rezultatele, inclusiv alegerea gradului şi erorile pe cele două seturi de date, ţinând cont de discuţia din curs despre alegerea modelului şi supraantrenarea (en. *overfitting*) în regresie.

Chiar dacă MATLAB furnizează funcţii deja implementate pentru regresia liniară cu polinoame, vi se recomandă insistent să programaţi dvs. aproximatorul şi procedura de regresie, fiindcă veţi avea nevoie de experienţa acumulată când veţi rezolva partea a doua a proiectului, care *nu* are o implementare standard în MATLAB. Pentru inspiraţie puteţi examina soluţia dvs. pentru laboratorul 4, *Regresia liniară pentru aproximarea funcţiilor*, şi codul funcţiei `rbfapprox` furnizate la începepului acelui laborator.

Identificarea de tip cutie neagră a unui sistem dinamic

Pentru a efectua această parte din proiect, veţi avea nevoie de bazele despre modele ARX *liniare* din Partea 5 a materialului de curs, *Metode ARX*, predate în cursul 6.

Se dă un set de date măsurat de la un **sistem dinamic** cu o intrare şi o ieşire. Ordinul sistemului nu este mai mare de 3, şi dinamica poate fi neliniară, în timp ce ieşirea poate fi afectată de zgomot. Vom dezvolta un model de tip cutie neagră pentru acest sistem, folosind o structură ARX neliniară de tip polinomial.

Un al doilea set de date, măsurat de la acelaşi sistem, este furnizat pentru validarea modelului dezvoltat. Cele două seturi de date sunt furnizate într-un fişier MATLAB, respectiv în variabilele `id` şi `val`, ambele obiecte de tip `iddata` din toolbox-ul de identificarea sistemelor. Reamintim că intrarea, ieşirea, şi perioada de eşantionare sunt disponibile în câmpurile `u`, `y`, `Ts` ale acestor obiecte. În caz că toolbox-ul nu este instalat, aceleşi seturi de date sunt furnizate şi în format vectorial, `id_array` şi `val_array`, fiecare dintre ele o matrice cu structura: valorile de timp pe prima coloană, intrarea pe a doua, şi ieşirea pe ultima coloană.

Considerăm un model cu ordinele na, nb, şi întârzierea nk, folosind aceeaşi convenţie ca şi funcţia MATLAB `arx`. Notăm vectorul de ieşiri şi intrări întârziate cu $d = [y(k-1), y(k-2), \ldots, y(k-na), u(k-nk), u(k-nk-1), \ldots, u(k-nk-nb+1)]^T$. Modelul ARX neliniar are atunci structura:

$$\begin{aligned}\hat{y}(k) = &- p_1(d)y(k-1) - p_2(d)y(k-2) - \ldots - p_{na}(d)y(k-na) \\ &z_1(d)u(k-nk) + z_2(d)u(k-nk-1) + \ldots + z_{nb}(d)u(k-nk-nb+1)\end{aligned} \tag{1}$$

unde fiecare p_i and z_j este un polinom în variabilele din d. De notat că aceste polinoame înlocuiesc coeficienţii constanţi ai modelului ARX liniar, ducând la un model neliniar în variabilele din d pentru orice grad de polinom mai mare decât zero. Dacă toate polinoamele au gradul zero, modelul se reduce la cel liniar. Parametrii modelului sunt coeficienţii fiecărui polinom. Pentru claritate, vom da un exemplu pentru cazul în care toate polinoamele sunt de gradul $m = 1$, şi $na = nb = nk = 1$. În acest caz, $d = [y(k-1), u(k-1)]^T$ şi modelul conţine doar două polinoame, unul pentru $y(k-1)$ şi celălalt pentru $u(k-1)$:

$$\hat{y}(k) = - p_1(d)y(k-1) + z_1(d)u(k-1)$$

Ambele polinoame sunt de gradul 1, de ex. $p_1(d) = a_1 + a_2 y(k-1) + a_3 u(k-1)$, şi similar pentru $z_1(d)$. Aşadar, modelul poate fi scris explicit după cum urmează:

$$\begin{aligned}\hat{y}(k) &= - [a_1 + a_2 y(k-1) + a_3 u(k-1)]y(k-1) + [b_1 + b_2 y(k-1) + b_3 u(k-1)]u(k-1) \\ &= - a_1 y(k-1) - a_2 y(k-1)^2 - a_3 u(k-1)y(k-1) \\ &\quad + b_1 u(k-1) + b_2 y(k-1)u(k-1) + b_3 u(k-1)^2\end{aligned} \tag{2}$$

Modelul (1) are o proprietate esenţială: este liniar în parametri (în exemplul (2), a_i şi b_j), ceea ce înseamnă că parametrii pot fi găsiţi folosind metoda regresiei liniară. Dacă modelul este însă lăsat în forma iniţială (1), există o problemă: unii dintre regresori sunt liniar dependenţi, fapt care neadresat ar duce la singularităţi în sistemul de ecuaţii liniare. Pentru a ne asigura că modelul este bine condiţionat numeric, termenii de aceeaşi formă trebuiesc combinaţi într-unul singur, cu un singur coeficient/parametru. În exemplul (2), produsul $y(k-1)u(k-1)$ apare de două ori, şi aceşti termeni trebuiesc combinaţi:

$$\hat{y}(k) = -a_1 y(k-1) - a_2 y(k-1)^2 + b_1 u(k-1) + b_3 u(k-1)^2 + c y(k-1)u(k-1) \tag{3}$$

unde $c = -a_3 + b_2$.

Cerinţele sunt descrise în cele ce urmează. Programaţi o funcţie care generează un model ARX neliniar de tip polinomial, pentru ordine na, nb, întârzieri nk, şi grade polinomiale configurabile. Pentru simplitate puteţi lua toate polinoamele de acelaşi grad m. Asiguraţi-vă că grupaţi termenii liniar dependenţi conform explicaţiei de mai sus, pentru a evita problemele numerice. Programaţi de asemenea procedura de regresie pentru identificarea parametrilor, şi utilizarea modelului cu intrări noi. De notat că această utilizare trebuie să se poată face în două moduri:

- Predicţie (cu un pas înainte), care foloseşte valorile reale ale ieşirilor întârziate ale sistemului; în exemplu, la pasul k s-ar aplica ecuaţia (3) folosind variabilele $y(k-1)$ şi $u(k-1)$ în partea dreaptă a egalităţii.

- Simulare, în care ieşirile precedente ale sistemului nu sunt disponibile, şi ca atare nu se pot folosi decât ieşirile anterioare ale modelului însuşi; în exemplu, $y(k-1)$ ar fi înlocuit cu valoarea simulată precedent $\hat{y}(k-1)$ în partea dreaptă a ecuaţiei (3)

Identificaţi un astfel de model ARX neliniar folosind setul de date de identificare, şi validaţi-l pe setul de validare. Alegeţi atent ordinele modelului şi întârzierea, precum şi gradul polinoamelor. Pentru a simplifica procedura de căutare, puteţi lua $na = nb$. Raportaţi eroarea de predicţie şi eroarea de simulare atât pentru setul de identificare, cât şi pentru cel de validare (folosiţi eroarea medie pătratică). Includeţi un grafic reprezentativ al calităţii modelului pentru cele două seturi, în simulare şi pentru predicţie. Discutaţi rezultatele, incluzând calitatea modelului pe cele două seturi de date.

Prezentarea de mai sus este suficientă pentru implementarea metodei, dar pentru detalii tehnice puteţi studia de ex. următoarele articole:

1. 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. Modelul este explicat în secţiunile 2.1-2.2, dar foloseşte în loc de polioname funcţii de bază radiale reglabile în poziţie şi lăţime.

2. L. Ljung, *System Identification*, Wiley Encyclopedia of Electrical and Electronics Engineering, 2007. Disponibil sub formă de raport tehnic cu numărul LiTH-ISY-R-2809. Vezi secţiunea 4 pentru modele neliniare, care din nou folosesc funcţii de bază.

Programare în MATLAB şi remarci adiţionale

Faceţi un efort pentru a produce un cod MATLAB compact şi elegant, evitând folosirea buclelor `for`, `while` şi a instrucţiunilor `if-then-else` pentru operaţii care pot fi efectuate mai simplu folosind vectori sau matrici. Indicii în această direcţie găsiţi căutând conceptul de "vectorizare" (en. *vectorization*) în documentaţia MATLAB. Nu exageraţi însă cu vectorizarea: dacă programul este mai clar sau uşor de înţeles cu bucle sau instrucţiuni if, folosiţi-le.

Dacă sunteţi mai puţin familiari cu programarea în MATLAB, următoarele idei vă pot fi de folos. Scrieţi `doc` în linia de comandă pentru accesarea documentaţiei. Un nod util la început este *Getting Started with* MATLAB . Vor fi de folos de asemenea şi *Matrices and Arrays*, *Programming Basics*, şi *Plotting Basics*.

Este de preferat ca fiecare grup să lucreze la proiect pe laptopul lor, şi ca acest laptop să fie acelaşi pe toată durata proiectului. Se evită astfel incompatibilităţile între versiunile de MATLAB instalate pe calculatoarele dvs. şi cele ale Universităţii. Dacă acest lucru nu este posibil pentru dvs., contactaţi profesorul pentru a discuta situaţia.