Teoria Sistemelor

Laboratorul 5: Sisteme dinamice pe spaţiul stărilor. Dinamică şi stabilitate

1 Scopul laboratorului

Laboratorul are ca scop familiarizarea studenţilor cu sistemele descrise pe spaţiul stărilor. Mai precis, oricărui sistem liniar şi invariant în timp îi asociem un model de stare, $S = (A, B, C, D)$. Pentru un astfel de model sistemic vom studia probleme ce vizează stabilitatea, raspunsul dinamic şi evoluţia stărilor, echivalenţa între modele. Pentru o mai bună înţelegere a acestei noi modalităţi de reprezentare sistemică vom studia diverse metode de conversie între modelele de stare şi modelele studiate în detaliu la cursul de **Semnale şi Sisteme**, anume funcţiile de transfer.

2 Breviar teoretic

Modele pe spaţiul stărilor

La cursul de **Semnale şi Sisteme** am văzut că sistemele dinamice pot fi reprezentate prin modele de tip *funcţie de transfer* (reprezentări de tip *intrare/ieşire*). Un alt mod de a reprezenta un sistem dinamic îl constituie *ecuaţiile de stare*

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) & x(t_0) = x_0 \\ y(t) = Cx(t) + Du(t), \end{cases} \tag{1}$$

unde ieşirea $y(t) \in \mathbb{R}^p$ depinde de intrarea $u(t) \in \mathbb{R}^m$ prin semnalul $x(t) \in \mathbb{R}^n$, numit *variabila de stare* a sistemului. $A \in \mathbb{R}^{n \times n}$ se numeşte *matrice de stare*, $B \in \mathbb{R}^{n \times m}$ este matricea care ponderează intrările, $C \in \mathbb{R}^{p \times n}$ este matricea care ponderează ieşirile, iar $D \in \mathbb{R}^{p \times m}$ este matricea de *transfer direct.*

Modelul S descris de (1) este invariant la o transformare liniară de stare

$$\tilde{x} = Tx, \tag{2}$$

unde $T \in \mathbb{R}^{n \times n}$ este o matrice nesingulară arbitrară , în sensul că noul vector de stare $\tilde{x}$ satisface ecuaţii de tipul (1), în care matricele corespunzătoare sunt

$$\begin{aligned} \tilde{A} = TAT^{-1} \quad \tilde{B} = TB \\ \tilde{C} = CT^{-1} \quad \tilde{D} = D. \end{aligned} \tag{3}$$

Modelele S şi $\tilde{S} = (\tilde{A}, \tilde{B}, \tilde{C}, \tilde{D})$ se numesc *echivalente pe stare* şi au aceleaşi proprietăţi intrare-ieşire.

Sisteme MIMO

Ştim că pentru sistemele liniare şi invariante în timp avem $y(t) = (h * u)(t)$ sau, în Laplace, $Y(s) = H(s)U(s)$. Considerăm acum $u(t)$ şi $y(t)$ ca fiind vectori de semnale (intrare, respectiv ieşire), prin urmare avem

$$\begin{bmatrix} Y_1(s) \\ Y_2(s) \\ \vdots \\ Y_p(s) \end{bmatrix} = \begin{bmatrix} H_{11}(s) & H_{12}(s) & \dots & H_{1m}(s) \\ H_{21}(s) & H_{22}(s) & \dots & H_{2m}(s) \\ \vdots & \vdots & \ddots & \vdots \\ H_{p1}(s) & H_{p2}(s) & \dots & H_{pm}(s) \end{bmatrix} \begin{bmatrix} U_1(s) \\ U_2(s) \\ \vdots \\ U_m(s) \end{bmatrix}, \tag{4}$$

unde $U_j(s)$ sunt transformatele Laplace ale semnalelor $u_j(t), i = 1 : m$, iar $Y_i(s)$ sunt transformatele Laplace ale semnalelor $y_i(t), j = 1 : p$. Matricea $H \in \mathbb{R}^{p \times m}(s)$ din relatia (4) se numeşte matrice de transfer.

Realizări de stare

Un cvadruplet $S = (A, B, C, D)$ care satisface $H(s) = C(sI - A)^{-1}B + D$ se numeşte realizare de stare a sistemului $H(s)$. Dimensiunea matricei A se numeşte ordinul realizării de stare. Numărul de coloane ale matricei B coincide cu numărul de coloane ale matricei $H(s)$ şi constituie numărul de intrari ale sistemului. Numărul de linii ale matricei C coincide cu numărul de linii ale matricei $H(s)$ şi constituie numarul de ieşiri ale sistemului.

În multe aplicaţii practice sistemul pe care îl avem de procesat (analiză, stabilizare, reglare) este furnizat (sau identificat) sub formă unei matrice de transfer. Cu toate acestea, majoritatea tehnicilor implementabile numeric (algoritmi) sunt formulate în termenii realizărilor de stare, întrucat spaţiile vectoriale liniare beneficiază de proprietăţi de robustţe numerică net superioare spaţiilor polinomiale. Prin urmare, studiem modalităţi de conversie între modele de tip *matrice de transfer* şi *modele de stare*.

Realizări standard

Fiind dată o matrice de transfer $H(s)$ din $\mathbb{R}^{p\times m}$ ca în (4), atunci, la fel ca la funcţiile de transfer, primul pas constă în extragerea părţii întregi, $D = H(\infty)$. Odată obţinută $\tilde{H}(s)$, matrice de transfer **strict** proprie matricele A, B şi C se obţin prin inspecţie după formulele

$$A = \begin{bmatrix} 0_m & I_m & 0_m & \dots & 0_m \\ 0_m & 0_m & I_m & \dots & 0_m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0_m & 0_m & 0_m & \dots & I_m \\ -a_0 I_m & -a_1 I_m & -a_2 I_m & \dots & -a_{n-1} I_m \end{bmatrix} \qquad B = \begin{bmatrix} 0_m \\ 0_m \\ \vdots \\ 0_m \\ I_m \end{bmatrix} \tag{5}$$
$$C = \begin{bmatrix} K_0 & K_1 & K_2 & \dots & K_{n-1} \end{bmatrix} \qquad D = H(\infty)$$

sau după formulele

$$A = \begin{bmatrix} 0_p & 0_p & \dots & 0_p & -a_0 I_p \\ I_p & 0_p & \dots & 0_p & -a_1 I_p \\ 0_p & I_p & \dots & 0_p & -a_2 I_p \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0_p & 0_p & \dots & I_p & -a_{n-1} I_p \end{bmatrix} \qquad B = \begin{bmatrix} K_0 \\ K_1 \\ \vdots \\ K_{n-2} \\ K_{n-1} \end{bmatrix} \tag{6}$$
$$C = \begin{bmatrix} 0_p & 0_p & \dots & 0_p & I_p \end{bmatrix} \qquad D = H(\infty),$$

Spunem că două sisteme, $S = (A, B, C, D)$ şi $\tilde{S} = (\tilde{A}, \tilde{B}, \tilde{C}, \tilde{D})$ sunt *echivalente intrare ieşire* dacă le corespunde aceeaşi matrice de transfer, i.e.

$$C(sI - A)^{-1}B + D = H(s) = \tilde{C}(sI - \tilde{A})^{-1}\tilde{B} + \tilde{D}.$$

Dacă două sisteme sunt echivalente pe stare atunci ele sunt echivalente şi I/O. Reciproca nu este valabilă decât în cazul realizărilor minimale, aspect care va deveni clar pe parcursul laboratorului următor.

Dacă avem o funcţie de transfer $H(s) = \Sigma_{i=1}^{n} \frac{r_i}{s-\lambda_i}$, i.e. cu poli *simpli* atunci o metodă alternativă de a scrie o realizare de stare (numită şi realizarea reziduuri-poli) pentru $H(s)$ este

$$A = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_n \end{bmatrix} \quad b = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \quad c^T = \begin{bmatrix} r_1 & r_2 & \dots & r_n \end{bmatrix} \quad d = H(\infty) \tag{7}$$

Calculul funcţiei de transfer

Conversia inversă, respectiv calculul matricei de transfer asociate unui model de stare dat, se face separat pentru fiecare pereche (u_j, y_i) de intrări şi ieşiri, adică în esenţă se referă la un sistem simplu (SISO), de aceea este suficient să considerăm acest caz.

Avem un sistem SISO $S = (A, B, C, D)$. Pentru a calcula funcţia de transfer, adică două polinoame $p(s)$, $N(s)$ astfel încât

$$\frac{N(s)}{p(s)} = C(sI - A)^{-1}B + D$$

constatăm că funcţia de transfer a unui sistem cu reacţie unitară este

$$H_0(s) = \frac{H(s)}{1+H(s)} = \frac{N(s)}{p(s)+N(s)} = \frac{N(s)}{p_0(s)},$$

deci avem că $N(s) = p_0(s) - p(s)$. Astfel, problema se reduce la a calcula $p(s)$ şi $p_0(s)$. Ştim că $p(s)$ este polinomul caracteristic al matricei A, adică

$$p(s) = \prod_{i=1}^{n}(s - \lambda_i),$$

unde λ_i sunt valorile proprii ale matricei A. Analog $p_0\,(s)$ este polinomul caracteristic al matricei A_0 a sistemului în circuit închis, deci $A_0 = A - BC$.

Ecuaţii matriceale

Ecuaţiile matriceale liniare cele mai întâlnite în aplicaţiile din domeniul sistemelor liniare sunt de forma

$$A_1XB_1 + A_2XB_2 = C,$$

unde $A_i \in \mathbb{R}^{m\times m}$ şi $B_i \in \mathbb{R}^{n\times n}$, iar $C, X \in \mathbb{R}^{m\times n}$. Particularizările de forma

$$\begin{aligned} AX + XB &= C, \\ AXB + X &= C \end{aligned} \tag{8}$$

sunt cunoscute sub numele de ecuaţii matriceale *Sylvester* continuă , respectiv discretă. Ecuaţia matriceală Sylvester continuă admite o soluţie $X \in \mathbb{R}^{m\times n}$ unică dacă şi numai dacă

$$\lambda_i + \mu_j \neq 0, \quad \forall\lambda_i \in \lambda(A), \forall\mu_j \in \lambda(B) \tag{9}$$

sau, altfel spus, dacă şi numai dacă $\lambda(A) \cap \lambda(-B) = \phi$. Ecuaţia matriceală Sylvester discretă admite o soluţie $X \in \mathbb{R}^{m\times n}$ unică dacă şi numai dacă

$$\lambda_i\mu_j \neq -1, \quad \forall\lambda_i \in \lambda(A), \forall\mu_j \in \lambda(B).$$

Particularizând şi mai mult, obţinem ecuaţiile matriceale liniare cunoscute sub denumirile de ecuaţie *Lyapunov* continuă pentru

$$A^TX + XA = C, \tag{10}$$

respectiv ecuaţie *Lyapunov* discretă pentru

$$A^TXA - X = C.$$

Ecuaţia matriceală Lyapunov continuă admite o soluţie $X \in \mathbb{R}^{n\times n}$ unică dacă şi numai dacă matricea A nu are nicio pereche de valori proprii opuse, i.e.

$$\lambda_i + \lambda_j \neq 0, \quad \forall\lambda_i, \lambda_j \in \lambda(A).$$

Ecuaţia matriceală Lyapunov discretă admite o soluţie $X \in \mathbb{R}^{n\times n}$ unică dacă şi numai dacă matricea A nu are nicio pereche de valori proprii inverse, i.e.

$$\lambda_i\lambda_j \neq 1, \quad \forall\lambda_i, \lambda_j \in \lambda(A).$$

În continuare vom prezenta o schemă de rezolvare (concretizată în algoritm numeric) a ecuaţiei Sylvester continuă. Algoritmii pentru rezolvarea ecuacţiilor Sylvester discretă şi Lyapunov sunt propuse studenţilor ca exerciţii.

Să observăm întâi că ambele matrice coeficient pot fi reduse la formele lor Schur complexe (superior triunghiulare) folosind transformări de asemăre unitare.

$$\begin{aligned} U^*AU &= S \\ V^*BV &= T. \end{aligned}$$

Aplicând transformările U^* şi V la stânga, respectiv dreapta primei ecuaţii din (8) obţinem

$$SY + YT = D, \tag{11}$$

unde $Y = U^*XV$ şi $D = U^*CV$. Scriem acum (11) pe coloane

$$S \begin{bmatrix} y_1 & y_2 & \dots & y_n \end{bmatrix} + \begin{bmatrix} y_1 & y_2 & \dots & y_n \end{bmatrix} \begin{bmatrix} t_{11} & t_{12} & \dots & t_{1n} \\ 0 & t_{22} & \dots & t_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & t_{nn} \end{bmatrix} = \begin{bmatrix} d_1 & d_2 & \dots & d_n \end{bmatrix}.$$

Pentru prima coloană a lui Y obţinem sistemul de ecuaţii liniare

$$(S + t_{11} I_m) y_1 = d_1,$$

care este un sistem triunghiular cu soluţie unică (dacă este satisfacută condiţia (9)). Pentru celelalte coloane ale lui Y se obţin relaţiile

$$(S + t_{jj} I_m) y_j = dj - \sum_{k=1}^{j-1} t_{kj} y_k, \quad j = 1 : n. \tag{12}$$

Sistemele (12) sunt triunghiulare, admit soluţie unică şi se rezolvă în ordinea $1 : n$. Pentru implementarea schemei de rezolvare se urmează paşii din algoritmul 1.

Dinamică. Stabilitate

Evoluţia stării. Răspunsul dinamic

Soluţia primei ecuaţii din (1) se numeşte evoluţia stărilor sistemului şi are forma

$$x(t) = e^{At} x(0) + \int_0^t e^{(t-\tau)A} B u(\tau) d\tau, \tag{13}$$

iar matricea $\Phi(t) := e^{At}$ se numeste *matricea de tranzitie a starilor*. Datorită invarianţei în timp a sistemului am putut alege convenabil $x(t_0) = x(0)$. Evoluţia stării are două componente: evoluţia liberă , $x_l(t) = e^{At} x(0)$, care depinde doar de starea iniţială a sistemului şi evoluţia forţată $x_f(t) = \int_0^t e^{(t-\tau)A} B u(\tau) d\tau$, care depinde doar de intrarea sistemului.

Răspunsul dinamic al sistemului se obţine înlocuind expresia lui $x(t)$ în a doua ecuaţie din (1)

$$y(t) = C e^{At} x(0) + C \int_0^t e^{(t-\tau)A} B u(\tau) d\tau + D u(t), \tag{14}$$

unde $y_l(t) = C e^{At} x(0)$ constituie partea de răspuns liber, iar $y_f(t) = C \int_0^t e^{(t-\tau)A} B u(\tau) d\tau + D u(t)$ constituie partea de răspuns forţat.

Stabilitate

Spunem despre un sistem descris de ecuaţiile (1) că este stabil dacă evoluţia liberă tinde la zero (când t tinde la infinit) pentru orice stare iniţială $x(0)$. Evident, acest lucru se intamplă când $\lim_{t \to \infty} e^{At} = 0$, mai exact când toate valorile proprii ale matricei A sunt în $\mathbb{C}_-$. Deoarece proprietatea de stabilitate depinde exclusiv de plasarea spectrului matricei A în planul complex, deducem că o testare numerică eficientă a stabilităţii se poate realiza calculând valorile proprii (cu algoritmul QR fără acumlarea transformărilor).

Pentru testarea numerică a stabilităţii propunem un algoritm bazat pe calculul explicit al valorilor proprii (vezi algoritmul 2 din anexă).

Teoria Lyapunov arată că un sistem liniar şi invariant în timp este stabil dacă şi numai dacă , pentru orice matrice Q simetrică şi pozitiv definită , soluţia ecuaţiei

$$A^T X + X A + Q = 0$$

este pozitiv definită.

Calculul regimului permanent

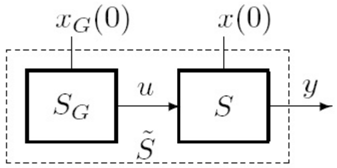
Analiza comportării dinamice a unui sistem liniar presupune, în esenţă , simularea răspunsului în timp al acestuia. În orice mediu de simulare numerică semnalele sunt memorate ca vectori ce au drept componente evaluări ale semnalelor la diverse momente de timp (eşantioane). Orice simulare presupune alegerea unui *interval de simulare* şi a unui *pas de simulare*. Observăm că , pentru calculul răspunsului liber, trebuie practic discretizată ecuaţia diferenţiala liniara omogena $\dot{x}(t) = Ax(t)$. Dificultatea apare la calculul raspunsului forţat, întrucât acest lucru ar presupune evaluarea produsului de convoluţie între *matricea pondere* $T(t) = Ce^{At}B1(t) + D\delta(t)$ şi funcţia de intrare $u(t)$ prin metode de integrare numerică .

O metodă de calcul a răspunsului forţat evitând calculul explicit al produsului de convoluţie este să considerăm semnalul de intrare $u(t)$ ca fiind răspunsul *liber* al unui filtru generator

$$\begin{cases} \dot{x}_G(t) = A_G x_G(t) & x_G(0) = b_G, \\ u(t) = C_G x_G(t) \end{cases} \tag{15}$$

unde $u(s) = C_G(sI - A_G)^{-1} b_G$.

Considerăm acum sistemul format din conexiunea serie a sistemelor (1) şi (15), ca în figura de mai jos



Noul sistem, $\tilde{S}$ are ecuaţiile

$$\begin{cases} \begin{bmatrix} \dot{x}(t) \\ \dot{x}_G(t) \end{bmatrix} = \begin{bmatrix} A & BC_G \\ 0 & A_G \end{bmatrix} \begin{bmatrix} x(t) \\ x_G(t) \end{bmatrix} & \begin{bmatrix} x(0) \\ x_G(0) \end{bmatrix} = \begin{bmatrix} x_0 \\ b_G \end{bmatrix} \\ \\ y(t) = \begin{bmatrix} C & DC_G \end{bmatrix} \begin{bmatrix} x(t) \\ x_G(t) \end{bmatrix} \end{cases} \tag{16}$$

Prin urmare, răspunsul forţat al sistemului S se poate calcula ca fiind răspunsul liber al sistemului $\tilde{S}$, cu starea initială precizată în ecuaţiile (16).

Dacă presupunem sistemul S stabil, putem vorbi despre regim permanent şi tranzitoriu. Ştim că regimul tranzitoriu este generat de matricea pondere a sistemului, iar cel permanent de semnalul de la intrare. Prin urmare considerăm o separare a evoluţiei stărilor $x(t) = \xi(t) + Vx_G(t)$, unde $\xi(t)$ este componenta tranzitorie a traiectoriei stărilor, iar componenta permanentă este o transformare liniară a traiectoriei de stare a filtrului generator. Scriem ieşirea

$$y(t) = C(\xi(t) + Vx_G(t)) + DC_G x_G(t) \Rightarrow y(t) = \eta(t) + (CV + DC_G)x_G(t),$$

unde prin $\eta(t)$ am notat răspunsul tranzitoriu. Prin urmare răspunsul permanent al sistemului S este ieşirea liberă a sistemului

$$\begin{aligned} \dot{x}_G(t) &= A_G x_G(t) \\ y_p(t) &= (CV + DC_G)x_G(t) \end{aligned} \tag{17}$$

unde $x_G(0) = b_G$, iar matricea V este soluţia ecuaţiei matriceale Sylvester continue

$$AV - VA_G + BC_G = 0. \tag{18}$$

Semnalele de intrare $u(t)$ se presupun persistente, deci sistemul S_G rezultă antistabil. Din presupunerea ca S este stabil rezultă că spectrul matricei A este situat în semiplanul stâng deschis. Avem deci că spectrele matricelor A şi A_G sunt disjuncte, prin urmare ecuaţia Sylvester (18) are soluţie şi aceasta este unică .

Rutine Matlab®

a Pentru verificarea consistenţei dimensionale a matricelor dintr-un model de stare se utilizează rutina **abcdchk**, iar pentru validarea unei matrice de transfer rutina **tfchk**.

b Rutina **ss** apelată cu un parametru de tip matrice de transfer returnează un model de stare al acesteia. Rutina **tf** apelată cu un parametru de tip model de stare returnează matricea de transfer asociată acestuia. Pentru transformări de echivalenţă pe spaţiul stărilor se foloseşte rutina **ss2ss**.

c Un test simplu de stabilitate al unui model de stare îl constituie evaluarea valorilor proprii, pentru care se foloseşte rutina **eig**.

d Pentru simularea răspunsului liber (sau a evoluţiei libere) se foloseşte rutina **initial**. Pentru simularea răspunsului forţat se folosesc aceleaşi rutine ca la sistemele SISO scrise ca funcţii de transfer, anume **impulse**, **step**, **lsim**.

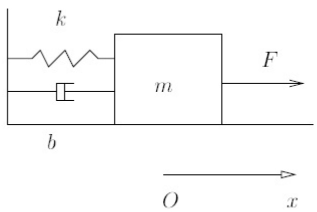
e Pentru rezolvarea ecuaţiilor matriceale Sylvester se foloseşte rutina **sylv**.

f Pentru simularea 3D a unei traiectorii de stare se va folosi rutina

```
function evolutie(sys,x0)
 [y,t,x]=initial(sys,x0);
 N=length(t);
 grid on
 hold on
 plot3(x0(1),x0(2),x0(3),'or')
 plot3(x(:,1),x(:,2),x(:,3))
 plot3(x(N,1),x(N,2),x(N,3),'og')
 hold off
```

3 Exerciţii rezolvate

Exerciţiul 1. Se dă sistemul din figura de mai jos



unde m este masa corpului, b este constanta conexiunii rigide, k este constanta elastica a resortului, iar $F(t)$ este o forta de tracţiune, variabilă în timp.

Se cere:

a Scrieţi un model de stare al acestui sistem, astfel încât semnalul de ieşire să fie viteza corpului

b Modificaţi modelul de stare astfel încât la ieşire să avem forţa elastică şi forţa de inerţie

c Pentru modelul găsit la punctul anterior analizaţi stabilitatea sistemului (valori numerice: $m = 1$, $k = 2$, $b = 3$)

d Pentru acelaşi model de stare simulaţi

(a) răspunsul liber pentru $x(0) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

(b) evoluţia forţată a stărilor pentru $u(t) = 1(t)$ şi pentru $u(t) = sin(2t)1(t)$.

Rezolvare. a. Ecuaţia diferenţială liniară ce descrie sistemul este

$$m\ddot{z}(t) + b\dot{z}(t) + kz(t) = F(t),$$

Considerăm vectorul de stare format din poziţie şi viteză

$$x(t) = \begin{bmatrix} z(t) \\ \dot{z}(t) \end{bmatrix},$$

iar ecuaţia de stare este

$$\begin{bmatrix} \dot{z}(t) \\ \ddot{z}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{b}{m} \end{bmatrix} \begin{bmatrix} z(t) \\ \dot{z}(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} F(t).$$

Am obţinut astfel matricele A şi B. Matricele C şD rezultă în funcţie de semnalele pe care le dorim la ieşire. Astfel, dacă ieşirea este viteza corpului, avem

$$y(t) = \dot{z}(t) = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} z(t) \\ \dot{z}(t) \end{bmatrix},$$

deci $C = \begin{bmatrix} 0 & 1 \end{bmatrix}$ şi $D = 0$.

b. Dacă la ieşire dorim forţa elastică şi forţa de inerţie atunci ecuaţia devine

$$y(t) = \begin{bmatrix} F_e(t) \\ F_i(t) \end{bmatrix} = \begin{bmatrix} k & 0 \\ -k & -b \end{bmatrix} \begin{bmatrix} z(t) \\ \dot{z}(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} F(t).$$

c. Matricea A devine

$$A = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix}$$

şi este o formă standard controlabilă, deci polinomul ei caracteristic este $p(s) = s^2 + 3s + 2$, ale carui rădăcini sunt -1 şi -2, prin urmare sistemul este stabil.

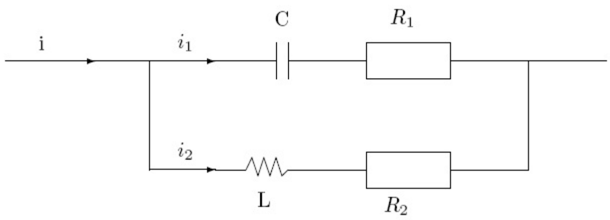
d. Codul Matlab este

```
A=[0 1;-2 -3]; B=[0;1]; C=[2 0;-2 -3]; D=[0;1];
sys=ss(A,B,C,D);
x0=[1;1];
figure(1)
initial(sys,x0)

C=eye(2);
D=[0;0];
sys=ss(A,B,C,D);
t=0:0.01:15;
u=ones(1,length(t));
figure(2)
lsim(sys,u,t)

u=sin(2*t);
figure(3)
lsim(sys,u,t)
```

Am simulat evoluţia forţată a stărilor ca fiind răspunsul forţat pentru $C = I_2$ şi $D = 0$.

□

Exerciţiul 2. Consideram circuitul din figura de mai jos.



Consideram ca semnalul de intrare este $u(t) := i$. Scrieţi un model de stare în care ieşirea să fie, pe rând

a curentul i_2

b curentul i_1

c sarcina q_2

d sarcina q_1 şi tensiunea u_L

Rezolvare. Vom rezolva doar punctul a.

Scriem legile lui Kirchhoff şi obţinem

$$L\ddot{q}_1 + (R_1 + R_2)\dot{q}_1 + \frac{1}{C}q_1 = \frac{1}{C}u + R_1\dot{u}$$

Luăm variabilele de stare

$$\begin{aligned} x_1 &= q_1; \\ \dot{x}_2 &= -\frac{1}{C}x_1 + R_2 u \\ x_2 &= L\dot{x}_1 + (R_1 + R_2)x_1 - Lu \Rightarrow \\ \dot{x}_1 &= -\frac{R_1 + R_2}{L}x_1 + \frac{1}{L}x_2 + u \end{aligned}$$

Ecuaţiile de stare astfel obţinute sunt

$$\begin{aligned}\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} &= \begin{bmatrix} -\frac{R_1+R_2}{L} & \frac{1}{L} \\ -\frac{1}{C} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} -L \\ 1 \end{bmatrix} u \\ y &= \begin{bmatrix} \frac{R_1+R_2}{L} & -\frac{1}{L} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + u. \end{aligned} \tag{19}$$

□

4 Exerciţii propuse

Exerciţiul 3. Pentru circuitul modelat de ecuaţiile (19) consideraţi valorile numerice $R_1 = 1\Omega$, $R_2 = 3\Omega$, $L = 3mH$ şi $C = 6\mu F$. Introduceţi modelul in Matlab, astfel încât la ieşire să apară sarcina q_1 şi tensiunea u_L.

a) Testaţi stabilitatea sistemului.

b) Simulaţi răspunsurile la impuls, treaptă şi rampă. Ţineţi cont de faptul că răspunsul în timp este suma dintre răspunsul liber şi cel forţat. Simulaţi pe rand:

- răspunsul liber, considerând orice stare iniţialănenulă;
- răspunsul forţat la semnalele menţionate;
- răspunsul total.

c) Calculaţi matricea de transfer asociată.

Exerciţiul 4. Introduceţi în Matlab un sistem fără intrare $S = (A, 0, C, 0)$ cu 3 stâri şi o ieşire.

a) Simulaţi evoluţia liberă a stărilor în $\mathbb{R}^3$ (cu $x(0)$ nenul). Se va folosi rutina **evolutie.m**, prezentată în secţiunea 2. Ce puteţi spune despre stabilitatea sistemului?

b) Modificaţi matricea A astfel încât sistemul să rezulte stabil şi simulati din nou evoluţia stărilor.

c) Figuraţi, pe acelaşi grafic, evoluţia liberă a stărilor, variind starea initiala $x(0)$.

d) Modificaţi matricea A astfel încât sistemul să aibă o pereche de poli complex-conjugaţi şi figuaţi, pe acelaşi grafic, evoluţia stărilor, variind partea imaginară a polilor. Variaţi apoi partea reală a polilor, astfel încât aceştia să se apropie de axa imaginară.

Exerciţiul 5. Introduceţi în Matlab un sistem stabil $S = (A, B, C, D)$ cu 3 stări, 3 intrări şi 2 ieşiri.

a) Simulaţi evoluţia forţată a stărilor în $\mathbb{R}^3$, când intrarea este $u(t) = [1(t) \quad 0.5 \cdot 1(t) \quad cos(3t)1(t)]^T$. *Indicaţie*: modificaţi rutina **evolutie.m**.

b) Calculaţi matricea de transfer asociată.

Exerciţiul 6. Considerăm următoarea matrice de transfer

$$H(s) = \begin{bmatrix} \frac{1}{s+11} & \frac{1}{s+12} & \frac{s}{s+13} \\ \frac{s}{s+21} & \frac{1}{s+22} & \frac{1}{s+23} \end{bmatrix}$$

a) Introduceţi sistemul în Matlab. *Indicaţie*: sistemul fiind MIMO, rutina **tf** primeşte drept argumente matricea numărătorilor şi matricea numitorilor:

```
H=tf({1 1 [1 0] ;[1 0] 1 1},{[1 11] [1 12] [1 13];[1 21] [1 22] [1 23]})
```

b) Găsiţi o realizare de stare. Testaţi stabilitatea sistemului pe baza acestei realizări.

c) Figuraţi răspunsul forţat al sistemului la $u(t) = [5 \cdot 1(t) \quad 1(t) \quad cos(3t)1(t)]^T$.

d) Calculaţi răspunsul (liber şi forţat) al sistemului la aceeaşi intrare, apoi figuraţi pe acelaşi grafic semnalele $y_l(t)$ si $y(t) - y_f(t)$. Starea iniţială se consideră $x(0) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}^T$.

e) Calculaţi şi simulaţi răspunsul **permanent** la intrarea $u(t) = [5 \cdot 1(t) \quad 1(t) \quad cos(3t)1(t)]^T$, folosind formulele (17). *Indicaţie*: sistemul S_G va rezulta din realizarea de stare a transformatei Laplace a intrării.

Exerciţiul 7. Fie matricea de transfer

$$H(s) = \begin{bmatrix} \frac{3s+1}{s^2+5s+4} & \frac{s+1}{s+2} \\ \frac{1}{s+2} & \frac{s^2-2}{s^2+3s+2} \\ \frac{1}{s+4} & \frac{s}{s^2+6s+8} \end{bmatrix}$$

a Introduceţi matricea de transfer în Matlab şi găsiţi o realizare de stare.

b Găsiţi o realizare de stare folosind apelul **ss(H,'min')**. Ce observaţi?

c Calculaţi şi simulaţi răspunsul la intrarea $u(t) = [3cos(3t) + 4sin(3t) \quad 1(t)]^T$, cu $u(t) = 0$ pentru $t < 0$.

d Calculaţi şi simulaţi răspunsul **permanent** la aceeaşi intrare, folosind formulele (17).

e Aplicaţi o transformare pe spaţiul stărilor astfel încât matricea A a noii realizări să rezulte strict diagonală. Calculaţi matricea de transfer asociată şi comparaţi-o cu matricea de transfer originală , $H(s)$.

f Aceeaşi cerinţă ca la punctul anterior, astfel încât matricea A să rezulte superior triunghiulară , iar transformarea de echivalenţă pe spaţiul stărilor să fie unitară.

Exerciţiul 8. Fie matricea de transfer

$$H(s) = \begin{bmatrix} \frac{2}{s+\epsilon} \\ \frac{s-1}{(s+\epsilon)(s+1)} \end{bmatrix}$$

a Scrieţi o realizare standard controlabilă pentru $H(s)$.

b Calculaţi răspunsul permanent la intrarea $u(t) = 1(t)$ pentru $\epsilon = 10^{-k}$ cu $k = 1, 2, 7, 10$. Studiaţi cu atenţie ce se întâmplă la rezolvarea ecuaţiei Sylvester (18). Explicaţi fenomenul.

Exerciţiul 9. Considerăm două modele de stare, $S_1 = (A_1, B_1, C_1, D_1)$ şi $S_2 = (A_2, B_2, C_2, D_2)$

$$A_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -2 & -3 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -2 \end{bmatrix}, B_1 = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}, C_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, D_1 = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix},$$

$$A_2 = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix}, B_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, C_2 = \begin{bmatrix} 0 & 1 \\ -4 & -2 \\ 1 & 1 \end{bmatrix}, D_2 = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}.$$

a) Simulaţi răspunsul la treaptă pentru cele două sisteme.

b) Simulaţi răspunsul liber, considerând starea iniţială $x(0) = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}^T$ pentru S_1 şi $x(0) = \begin{bmatrix} 1 & 1 \end{bmatrix}^T$ pentru S_2.

c) Calculaţi cele două matrice de transfer asociate, $H_1(s)$ şi $H_2(s)$.

Exerciţiul 10. Se dau sistemele $S_1 = (A_1, B_1, C_1, D_1)$ şi $S_2 = (A_2, B_2, C_2, D_2)$

$$A_1 = \begin{bmatrix} 2 & 1 & 3 \\ -1 & 3 & 1 \\ 0 & 0 & 8 \end{bmatrix}, B_1 = \begin{bmatrix} 1 & 2 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, C_1 = \begin{bmatrix} 1 & 2 & -1 \end{bmatrix}$$

$$A_2 = \begin{bmatrix} 2 & 1 & -1 \\ -1 & 3 & 2 \\ 0 & 0 & -5 \end{bmatrix}, B_2 = \begin{bmatrix} 1 & 2 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, C_2 = \begin{bmatrix} 1 & 2 & 7 \end{bmatrix}$$

a) Simulaţi evoluţia forţată la treaptă pentru cele două sisteme.

b) Simulaţi evoluţia liberă, considerând starea iniţială $x(0) = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^T$ pentru ambele sisteme.

c) Calculaţi cele două matrice de transfer asociate, $H_1(s)$ şi $H_2(s)$.

Exerciţiul 11. Fiind dată o funcţie de transfer a unui sistem SISO, scrieţi un algoritm care calculează o realizare de stare cu matricea A diagonală fără să mai fie nevoie de transformări de echivalenţă pe spaţiul stărilor. În ce condiţii se poate obţine o astfel de realizare? *Indicaţie*: se foloseşte *realizarea reziduuri poli.*

Exerciţiul 12. Implementaţi în Matlab un algoritm pentru testarea stabilităţii unui sistem $S = (A, B, C, D)$ bazat pe teorema de stabilitate Lyapunov.

Exerciţiul 13. Fiind dat un model de stare al unui sistem $S = (A, B, C, D)$ cu $m > 1$ intrări şi $p > 1$ ieşiri, impelentaţi în Matlab un algoritm care calculează matricea de transfer asociată modelului S, ţinând cont de faptul că fiecare element din matricea de transfer satisface

$$h_{ij}(s) = c_i^T (sI - A)^{-1} b_j + d_{ij}.$$

Folosiţi algoritmul 4 din anexă.

Exerciţiul 14. Scrieţi un algoritm care primeşte drept parametru de intrare o matrice de transfer (cu $m > 1$ şi $p > 1$) şi returnează o realizare standard observabila. Folosiţi algoritmul 3 din anexă. Cel mai mic multiplu comun a două polinoame se calculează ca fiind raportul dintre produsul şi cel mai mare divizor comun al acestora (calculat cu algoritmul lui Euclid). Se vor folosi rutinele Matlab **conv** şi **deconv** pentru înmulţirea, respectiv împărţirea a două polinoame.

Exerciţiul 15. Implementaţi în Matlab algoritmul 1 din anexă, care rezolvă ecuaţia Sylvester continuă. Adaptaţi algoritmul astfel încât acesta să rezolve o ecuaţie Lyapunov continuă.

Exerciţiul 16. Se dă sistemul $S = (A, B, C, D)$ ale cărui matrice au structura

$$A = \begin{bmatrix} A_1 & 0 \\ A_{21} & A_2 \end{bmatrix}, \quad B = \begin{bmatrix} B_1 \\ G_2 \end{bmatrix}$$
$$C = \begin{bmatrix} C_1 & C_2 \end{bmatrix}, \quad D = \begin{bmatrix} D \end{bmatrix}.$$

Scrieţi o procedură de reprezentare a lui S sub forma unei conexiuni

a paralel;

b serie;

5 Anexa - Laborator 5

Algorithm 1: Algoritm pentru rezolvarea ecuaţiei matriceale Sylvester continuă

Data: Matricele $A \in \mathbb{R}^{m\times m}$, $B \in \mathbb{R}^{n\times n}$ şi $C \in \mathbb{R}^{m\times n}$
Result: Matricea $X \in \mathbb{R}^{m\times n}$ ce verifică ecuaţia $AX + XB = C$

$[U, S] = fsc(A)$% Calculează forma Schur complexă a lui A şi acumularea transformărilor
$[V, T] = fsc(B)$% Calculează forma Schur complexă a lui B şi acumularea transformărilor
$D \leftarrow U^* CV$
for *j=1:n* **do**
  if $j > 1$ **then**
    $d_j = d_j - \sum_{k=1}^{j-1} t_{kj} y_k$
  else
    Rezolvă sistemul superior triunghiular
    $(S + t_{jj} I_m) y_j = d_j$
  end
end
$X = UYV^*$

Algorithm 2: Algoritm pentru testarea numerică a stabilităţii unui sistem continuu

Data: Modelul de stare (A, B, C, D)

L=eig(A) % Calculează valorile proprii ale matricei A, folosind algoritmul QR
α=max(Re(L)) % Calculează valoarea proprie cu partea reală maximă
if $\alpha < 0$ **then**
  afişează "Sistemul continuu este stabil"
else
  afişează "Sistemul continuu este instabil"
end

Algorithm 3: Algoritm pentru scrierea realizărilor de stare pentru sisteme MIMO

Data: Matricea de transfer $H(s)$
Result: Modelul de stare (A, B, C, D)

Se calculează $D = H(\infty)$ şi $\tilde{H}(s) = H(s) - D$
Se calculează $p(s) = s^n + a_{n-1}s^{n-1} + \ldots + a_0$, numitorul comun al tuturor funcţiilor de transfer din $\tilde{H}(s)$, pe care îl scriem acum $\tilde{H}(s) = N(s)\frac{1}{p(s)}$, unde $N(s) = K_{n-1}s^{n-1} + K_{n-2}s^{n-2} + \ldots + K_0$ este "numărătorul" matriceal rămas după extragerea numitorului comun;
Se scriu cele 4 matrice în acord cu (5) sau în acord cu (6)

Algorithm 4: Algoritm pentru conversia unui model stare SISO într-o funcţie de transfer

Data: Matricele A, B, C, D

Result: Polinoamele N, p, astfel încât $T(s) = \frac{N(s)}{p(s)}$

Se calculează valorile proprii λ_i ale matricei A utilizând algoritmul QR

Se formează $A_0 = A - BC$

Se calculează valorile proprii λ_{0i} ale matricei A_0 utilizând algoritmul QR

Se calculează $p\left(s\right) = \prod\limits_{i=1}^{n}\left(s - \lambda_i\right)$

Se calculează $p_0\left(s\right) = \prod\limits_{i=1}^{n}\left(s - \lambda_{0i}\right)$

Se calculează $N\left(s\right) = p_0\left(s\right) - p\left(s\right)$