

Vasile COMNAC

Simona COMAN
Cristian BOLDIŞOR

TEORIA SISTEMELOR
Îndrumar de laborator

UNIVERSITATEA TRANSILVANIA BRAŞOV
BIBLIOTECA - AULĂ
DEPOZIT

Universitatea Transilvania din Braşov
2009

PREFĂTĂ

În general, o lucrare de laborator nu poate fi altfel privită decât ca pe o aranjare elaborată a multiplelor informații oferite în curs, al cărui urmaș și revelator de drept este. În cazul particular al Îndrumarului, ce poate fi parcurs de orice student de la specializările de Automatică și Informatică Aplicată, de Calculatoare, de Tehnologia Informației, de Electronică Aplicată și de Telecomunicații, cărora le este adresat, autori au încercat să ofere un suport consistent în ceea ce privesc metodele, modurile și mijloacele de a aplica informația și de a putea asimila mai ușor concepțele oferite în diferite cărți de specialitate. Altfel spus, dorința acestora este de a oferi o perspectivă practică asupra cursului de Teoria Sistemelor, printr-un suport oferit de un sistem interactiv, propice atât domeniului ingineresc cât și altor domenii și anume Matlab, dezvoltat de „The MathWorks Inc.”. În încercarea de a atinge echilibrul la care se tinde prin însăși natura sistemelor, interpretabile prin formele unor fenomene fizice, electrice, termice, optice, etc., s-a încercat „contrabalansarea” informațiilor matematice, modelabile prin natura lor, cu o reprezentare simulată a acestora.

Un alt obiectiv al acestui Îndrumar este acela de a face o prezentare atât a limbajului Matlab, cât și a toolbox-urilor asociate Control System Toolbox și Simulink în vederea analizei și proiectării sistemelor.

Bineînțeles, în final, gândul autorilor poate fi întregit de un scop, ce este înlesnit prin organizarea informațiilor prezentate și anume ca studentul să înțeleagă să își poată explica și aplica termenii întâlniți, prezentați într-un mod antetic, da consecutiv. Structural, informația a fost prezentată într-un mod circular, sub forma unor capitulo ce pot fi identificate cu marile categorii de sisteme deja existente tocmai din dorința de a ușura înțelegerea termenilor, iar repetarea explicațiilor uno sau principii nu a avut alt scop decât aprofundarea lor. Orișicum, ca în orice domeniu, acolo unde se lucrează la nivel de detalii, se pot naște paralele ce pot aduna multiple nuanțe între diferenți termeni, care, inițial, nu pot fi considerați în aceleași cote. Doar o viziune corectă ulterioară în privința acestor termeni poate concretiza prin realizarea unor aplicații funktionale și corecte.

Așadar, în speranța că imaginația inginerescă a celor care studiază materialul prezent poate oricând să îmbogățească un bagaj de informații, ater introduse în acest manual, autori doresc să aducă mulțumiri generațiilor de studenți care, prin aportul lor, au contribuit la formula finală, de moment, Îndrumarul fiind continuu perfectibil și au reușit să imprime direcții clare de sinteză și analiză din partea autorilor pentru a putea explica consecvent termenii des întâlniți în domeniul Teoriei Sistemelor.

CUPRINS

PREFĂTĂ.....	7
INTRODUCERE ÎN MATLAB.....	9
1.1. Introducere	9
1.2. Lansarea în execuție	9
1.3. Ferestrele de lucru	10
1.3.1. Fereastra de comenzi	10
1.3.2. Fereastra de reprezentări grafice.....	11
1.4. Programarea în Matlab.....	11
1.4.1. Matrice, vectori și scalari.....	11
1.4.2. Declarații și variabile.....	12
1.4.3. Structura programelor în Matlab	12
1.4.4. Comentariile și help-ul într-un program.....	13
1.5. Instrucțiuni și funcții de control logic	14
1.5.1. Instrucțiunea conditională „if”	14
1.5.2. Instrucțiunea repetitivă „for”	16
1.5.3. Instrucțiunea repetitivă „while”	17
1.5.4. Instrucțiunea „break”	17
1.5.5. Instrucțiunea „return”	17
1.5.6. Instrucțiunea „error”	17
1.5.7. Funcții de control logic.....	17
1.6. Variabile speciale în Matlab.....	18
1.7. Exerciții propuse.....	18
CALCUL NUMERIC ÎN MATLAB.....	21
2.1. Operații aritmetice	21
2.1.1. Operații aritmetice cu scalari.....	22
2.1.2. Operații aritmetice cu vectori	22
2.1.3. Operații aritmetice cu matrice	23
2.2. Generarea matricelor	24
2.3. Generarea vectorilor cu pas liniar și cu pas logaritmic	25
2.4. Calcule cu matrice	25
2.4.1. Manipularea matricelor	25
2.4.2. Analiza matriceală	28
2.5. Calcule numerice cu polinoame	29
2.6. Exerciții propuse	30
REPREZENTĂRI GRAFICE ÎN MATLAB.....	33
3.1. Introducere	33
3.2. Reprezentări grafice bidimensionale (2D)	33

Îndrumar de laborator	Teoria Sistemelor	Îndrumar de laborator	Teoria Sistemelor
3.3. Reprezentări grafice tridimensionale (3D)	36	9.5.3. Conexiunea cu reacție	85
3.4. Exerciții propuse	37	9.6. Exerciții propuse	87
MODELAREA MATEMATICĂ A SISTEMELOR CONTINUE	39	ANALIZA ȘI SIMULAREA ÎN TIMP PENTRU SISTEMELE DISCRETE	89
4.1. Introducere	39	10.1. Introducere	89
4.2. Sistem mecanic resort-masă-amortizare și sistem electric RLC	39	10.2. Răspunsul sistemului discret de ordinul unu	90
4.3. Funcții de transfer	40	10.2.1. Răspunsul la impuls unitar	90
4.4. Modelele diagramei bloc	42	10.2.2. Răspunsul la treaptă unitară	90
4.4.1. Conexiunea serie	42	10.2.3. Răspunsul la rampă unitară	91
4.4.2. Conexiunea paralel	43	10.3. Răspunsul sistemului discret de ordinul doi	92
4.4.3. Conexiunea cu reacție	43	10.4. Performanțele sistemului de ordinul doi	93
4.5. Exerciții propuse	45	10.5. Stabilitatea sistemelor discrete	93
ANALIZA ȘI SIMULAREA ÎN TIMP PENTRU SISTEMELE CONTINUE	49	10.5.1. Criteriul de stabilitate Routh	94
5.1. Introducere	49	10.5.2. Criteriul de stabilitate Schur-Cohn	95
5.2. Răspunsul sistemului de ordinul întâi – T1	49	10.5.3. Criteriul de stabilitate Jury	96
5.3. Răspunsul sistemului de ordinul doi la intrarea treaptă – T2	51	10.6. Exerciții propuse	97
5.3.1. Performanțele sistemului de ordinul doi	53	ANALIZA ȘI SIMULAREA ÎN FRECVENTĂ PENTRU SISTEMELE DISCRETE	99
5.4. Efectele introducerii unor poli și zerouri suplimentare	54	11.1. Răspunsul în frecvență	99
5.5. Stabilitatea sistemelor	54	11.2. Diagrame Bode	100
5.5.1. Criteriul de stabilitate Routh	55	11.3. Criteriul Nyquist de stabilitate	101
5.5.2. Cazuri speciale	56	11.4. Exerciții propuse	101
5.6. Eroarea staționară și tipul sistemului	56	LOCUL RĂDĂCINILOR ȘI DESCRIEREA ÎN SPAȚIUL STĂRILOR PENTRU SISTEMELE DISCRETE	103
5.7. Exerciții propuse	57	12.1. Locul rădăcinilor	103
ANALIZA ȘI SIMULAREA ÎN FRECVENTĂ PENTRU SISTEMELE CONTINUE	61	12.2. Descrierea intrare - stare - ieșire	105
6.1. Răspunsul în frecvență	61	12.3. Exerciții propuse	106
6.2. Diagrame Bode	63	UTILIZAREA PROGRAMULUI SIMULINK	107
6.2.1. Algoritmul de trasare a diagramelor Bode	63	13.1. Introducere	107
6.3. Diagrame Nyquist	65	13.2. Realizarea unei sesiuni în Simulink	107
6.4. Exerciții propuse	65	13.3. Transfer de date între Simulink și Matlab	111
LOCUL RĂDĂCINILOR, UTILITARUL SISOTOOL	69	13.4. Exerciții propuse	113
7.1. Locul rădăcinilor	69	CREAREA DE SUBSISTEME ȘI S-FUNCTIONS	117
7.1.1. Rezumatul pașilor algoritmului de trasare a locului rădăcinilor	69	14.1. Crearea de subsisteme	117
7.2. Utilitarul Sisotool	71	14.1.1. Crearea de măști	119
7.3. Exerciții propuse	73	14.2. S-Functions	121
DESCRIEREA ÎN SPAȚIUL STĂRILOR A SISTEMELOR CONTINUE	75	14.2.1. Conversia unei S-Function într-un bloc	123
8.1. Descrierea intrare - stare - ieșire	75	14.3. Exerciții propuse	123
8.1.1. Trecerea de la funcția de transfer la spațiul stărilor	76	BIBLIOGRAFIE	125
8.1.2. Trecerea de la spațiul stărilor la funcția de transfer	76		
8.1.3. Controlabilitate și observabilitate	77		
8.1.4. Reducerea schemelor bloc în spațiul stărilor	78		
8.2. Exerciții propuse	78		
SISTEME DISCRETE	81		
9.1. Introducere	81		
9.2. Transformata Z	81		
9.3. Reprezentarea polilor și zerourilor în planul „z”	83		
9.4. Transformări între sisteme	83		
9.5. Funcții de transfer Z echivalente	84		
9.5.1. Conexiune în serie (cascadă)	84		
9.5.2. Proprietățile de bază ale operatorului de eșantionare	85		

INTRODUCERE ÎN MATLAB

OBIECTIVE

- Descrierea programului Matlab.
- Lucrul cu ferestre.
- Meniurile programului Matlab.
- Programarea în Matlab.
- Crearea fișierelor script și function.
- Instrucțiuni și funcții de control logic.
- Variabile speciale.

1.1. Introducere

Matlab este un pachet de programe de înaltă performanță, ce integrează calculul numeric și reprezentările grafice în domeniile științei și ingineriei. Matlab oferă, pe baza unor lucrări matematice, o analiză matriceală, sinteza și identificarea sistemelor și programe grafice ingineresci atât 2D cât și 3D. Matlab-ul integrează toate acestea într-un mediu ușor de învățat și folosit, în care enunțurile problemelor și rezolvările acestora sunt exprimate în modul cel mai natural posibil, așa cum sunt scrise matematic, fără a fi necesară programarea tradițională.

O caracteristică importantă a acestui limbaj de programare este ușurința cu care acesta poate fi extins. Astfel, orice utilizator poate adăuga propriile programe scrise în Matlab la fișierele originale, dezvoltând aplicații specifice domeniului în care lucrează. Structural, Matlab-ul este realizat sub forma unui nucleu de bază, cu interpretor propriu, în jurul căruia sunt construite toolbox-urile. Toolbox-urile sunt colecții extinse de funcții Matlab care dezvoltă mediul de programare de la o versiune la alta, pentru a rezolva probleme specifice anumitor domenii.

Firma The MathWorks Inc. (care a realizat acest limbaj) a pus în circulație o serie de toolbox-uri cum ar fi: Signal Processing, Image Processing, Symbolic Math, Neural Network, Control System Design, Robust Control, System Identification, Optimisation, Spline, Statistics, Wavelet, Simulink, Analysis and Synthesis, Finance, Fuzzy, etc.

1.2. Lansarea în execuție

Programul se lansează în execuție din mediul Windows, prin selecția pictogramei Matlab.

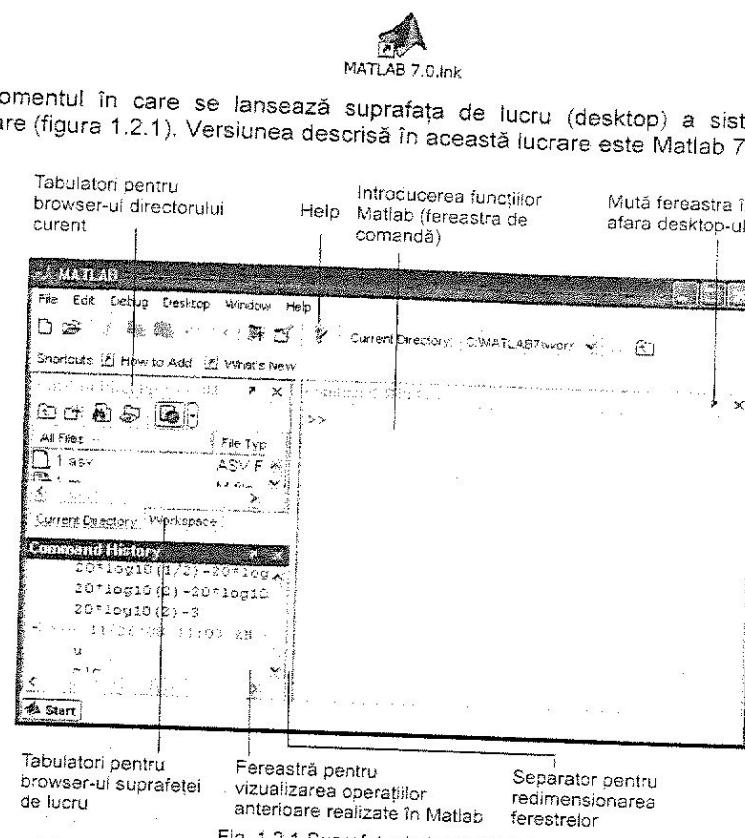


Fig. 1.2.1 Suprafața de lucru Matlab

Din afara mediului Windows (de exemplu DOS) programul poate fi lansat în execuție direct, cu o comandă de forma:

```
win c:\MATLAB\BIN\matlab
```

dacă fișierul „matlab.exe” se găsește în calea „c:\MATLAB\BIN”.

1.3. Ferestrele de lucru

Programul Matlab lucrează cu două tipuri de ferestre: o fereastră de comenzi și o fereastră pentru reprezentări grafice.

1.3.1. Fereastra de comenzi

Este utilizată pentru scrierea, modificarea și depanarea funcțiilor programelor (fișierelor de tip M). Meniul din bara superioară este accesibil prin tastarea simultană a tastei [Alt] și a literelor subliniate a comenziilor dorite sau prin selecția directă cu mouse-ul.

Fiecare comandă din meniul principal furnizează un meniu specific, selecția comenziilor dorite făcându-se prin deplasarea zonei active cu ajutorul săgeților sau prin

selecția directă cu ajutorul mouse-lui.

Meniul **File** conține o serie de comenzi legate de fișiere:

- **New** – este folosit pentru deschiderea unui fișier nou de tip M (M - file), a unei noi ferestre grafice (**Figure**), a unui model Simulink nou (**Model**) sau a unei interfețe grafice de tip Graphical User Interface (**GUI**);
- **Open** – permite deschiderea unor fișiere deja existente pentru editare și lansare;
- **Close Command Window** – închide fereastra de comandă Matlab;
- **Import Data** – opțiune destinată importului de date din fișiere;
- **Save Workspace As** – salvează toate variabilele din spațiul de lucru împreună cu valorile lor atribuite într-un fișier desemnat;
- **Set Path** – permite setarea căii fișierelor Matlab;
- **Preference** – accesează o casetă de dialog în care se pot seta o serie de caracteristici ale programului Matlab;
- **Print și Print Selection** – tipărește conținutul ferestrei de comandă, respectiv a părții selectate;
- **Exit MATLAB** – opțiune folosită pentru părăsirea mediului Matlab.

Meniul **Edit** conține o serie de comenzi cunoscute din Windows destinate editării: **Undo**, **Redo**, **Cut**, **Copy**, **Paste**, **Paste Special**, **Select All**, **Delete**, precum și trei comenzi specifice pentru ștergerea conținutului ferestrei de comandă, ferestrei istoric și suprafeței de lucru: **Clear Command Window**, **Clear Command History** și **Clear Workspace**.

În meniul **View** se regăsesc comenzi destinate manipulării diferitelor ferestre aferente instrumentelor utilizate în Matlab. Comenzile din meniul **Web** pot accesa resursele din Internet ale companiei producătoare (pagina de web, suportul tehnic, etc.).

În meniul **Window** sunt afișate toate ferestrele deschise în timpul sesiunii de lucru curente, putându-se afișa fereastra activă. Ultimul meniu este **Help**, în care, prin intermediul comenziilor existente, se poate accesa documentația atașată programului.

1.3.2. Fereastra de reprezentări grafice

Este o formă elevată de reprezentare a graficelor. Pot exista mai multe ferestre grafice deschise simultan, dar numai o singură fereastră de comenzi.

1.4. Programarea în Matlab

1.4.1. Matrice, vectori și scalari

Matlab-ul lucrează numai cu un singur tip de obiecte, matrice numerice rectangulare, cu elemente reale sau complexe. În acest sens, scalarii sunt asimilați matricelor cu o linie și o coloană (1×1), iar vectorii sunt asimilați matricelor cu o linie ($1 \times N$) sau cu o coloană ($N \times 1$).

Elementele unei matrice pot fi identificate prin una dintre notățiile: A_{ij} , $A[i, j]$, $A(i, j)$, etc. și semnifica elementul de la intersecția liniei i cu coloana j . Notația adoptată în Matlab este ultima dintre cele trei prezentate și anume $A(i, j)$.

Dimensiunea unei matrice este precizată de o pereche de numere care arată numărul de linii și coloane ale matricei respective.

Pentru a face referire la un element $A(i,j)$ al unei matrice A , sunt necesari doi indici, indicele de linie și indicele de coloană, în această ordine. Referirea unui element al unui vector poate fi făcută numai cu un singur indice.

Definirea matricelor se poate face prin una din următoarele metode:

- introducerea explicită a listei de elemente;
- generarea prin instrucțiuni și funcții;
- crearea de fișiere M¹;
- încărcarea din fișiere de date externe.

Matlab-ul nu conține instrucțiuni de dimensionare și declaratii de tip, iar memoria este alocată în mod automat până la valoarea maxim disponibilă. Cea mai simplă metodă de definire a matricelor mici constă în utilizarea unei liste explicate. La introducerea unei astfel de liste trebuie respectate următoarele reguli:

- elementele unei linii trebuie separate prin spațiu liber sau virgulă;
- linile se separă prin semnul punct-virgulă ":",
- elementele unei matrici sunt cuprinse între paranteze drepte "[]".

1.4.2. Declarații și variabile

Matlab-ul este un limbaj de expresii. Expresiile tipărite de utilizator sunt interpretate și evaluate. Instrucțiunile Matlab sunt, de cele mai multe ori, de forma:

variabilă=expresie,

sau, mai simplu:

expresie.

Expresiile sunt compuse din operatori sau alte caractere speciale, din funcții și nume de variabile. Evaluarea expresiei produce o matrice care este afișată pe ecran și atribuită unei variabile. Dacă numele variabilei și semnul egal („variabilă =”) sunt omise, Matlab-ul crează automat o variabilă cu numele „ans”.

Orice instrucțiune este în mod normal terminată cu „Enter”. Dacă ultimul caracter al unei instrucțiuni este punct-virgulă „：“, instrucțiunea este executată, dar tipărirea este suprimată. Utilizarea acestui caracter la sfârșitul unei instrucțiuni în fișiere M este necesară în situațiile în care nu se dorește afișarea datelor intermediare. Numele de variabile și de funcții au ca prim caracter o literă, urmată de litere, cifre sau caracterul special „liniuță de subliniere” (adică „_”). Matlab-ul face deosebirea între litere mari și mici. Funcția „casesen” permite trecerea Matlab-ului în modul senzitiv/nesenzitiv, în vederea separării literelor mari de cele mici. La lansare, Matlab-ul este în modul senzitiv, iar cu comanda casesen off se trece în modul nesenzitiv. Revenirea se face cu comanda casesen on. Numele de funcții este obligatoriu să fie redate cu litere mici.

1.4.3. Structura programelor în Matlab

Matlab-ul lucrează fie în modul linie de comandă, caz în care fiecare linie este prelucrată imediat și rezultatele sunt afișate, fie cu programe scrise în fișiere. Aceste

¹ Fișierele de tip M sunt fișiere ce conțin instrucțiuni Matlab și se numesc așa deoarece au extensia „m”. Acestea sunt de fapt programe Matlab.

două moduri de lucru formează împreună un mediu de programare ușor de învățat și de utilizat. Fișierele ce conțin instrucțiuni Matlab se numesc fișiere M (deoarece au extensia „m”) și sunt programe Matlab. Un fișier M constă dintr-o succesiune de instrucțiuni Matlab, cu posibilitatea apelării altor fișiere M precum și a apelării recursive.

Un program Matlab poate fi scris sub formă unor fișiere „script” sau sub formă de fișiere „function”. Ambele tipuri de fișiere sunt scrise în format ASCII.

1.4.3.1. Fișierele script

Acstea fișiere sunt fișiere externe ce conțin secvențe de instrucțiuni Matlab. Prin apelarea numelui fișierului, se execută secvența de instrucțiuni Matlab conținută de acesta. După execuția completă, variabilele cu care a operat fișierul rămân în zona de memorie a aplicației. Fișierele script nu permit integrarea în programe mari realizate pe principiul modularizării, ele fiind folosite pentru rezolvarea unor probleme care cer comenzi succesive lungi, ce îngreunează lucrul în modul linie de comandă.

1.4.3.2. Fișierele funcție (function)

Dacă prima linie a fișierului conține cuvântul cheie „function”, fișierul respectiv este declarat ca fișier funcție. Spre deosebire de un fișier script, un fișier funcție poate lucra cu argumente. Variabilele definite și utilizate în interiorul fișierului funcție sunt localizate la nivelul acestuia. Deci, la terminarea execuției unei funcții, în memorie rămân doar variabilele de ieșire ale funcției.

Forma generală a primei linii a unui fișier funcție este următoarea:

function[param_ieșire]=nume_funcție(param_intrare),

unde:

function - este cuvântul cheie care declară fișierul ca fișier funcție; prezența acestui cuvânt cheie este obligatorie;
nume_funcție - este numele funcției, adică numele sub care se salvează fișierul, fără extensie. Nu poate fi identic cu cel al unui fișier M preexistent.

param_ieșire - parametrii de ieșire trebuie separați cu virgulă și cuprinși între paranteze drepte (dacă funcția nu are parametri de ieșire, parantezele drepte și semnul egal nu mai au sens, pot să lipsească).

param_intrare - parametrii de intrare trebuie separați cu virgulă și cuprinși între paranteze rotunde (dacă funcția nu are parametri de intrare, parantezele rotunde nu mai au sens, pot să lipsească).

1.4.4. Comentariile și help-ul într-un program

Într-un program Matlab, un comentariu se introduce prin caracterul procent "%" plasat la începutul liniei. În cazul în care caracterul procent apare pe prima poziție într-o linie, aceasta este omisă de compilator, iar dacă apare într-o linie de program, partea de linie care urmează după semnul procent va fi omisă.

Într-un program, un comentariu poate apărea în orice poziție, dar este recomandată prezența unui comentariu după prima linie care declară o funcție. În acest caz, comentariul, care apare imediat după prima linie de declarare a funcției, constituie help-ul fișierului respectiv. Dacă se apelează în fereastra de comandă:

help nume_funcție,

se va afișa help-ul fișierului funcției respective.

1.5. Instrucțiuni și funcții de control logic

Instrucțiunile de control logic în Matlab sunt: **if**, **else**, **elseif**, **for**, **while**, **break**, **return**, **end**, **error**.

1.5.1. Instrucțiunea condițională „if”

Această instrucțiune este folosită în cazul în care este necesară o selecție a grupului de instrucțiuni ce urmează a fi executat, selecție condițională de valoarea de adevăr a unei expresii. Instrucțiunile condiționale utilizează operatorii relaționali și operatorii logici.

Matlab-ul are șase operatori relaționali, care sunt utilizati pentru a compara două matrice de dimensiuni egale. Lista acestor operatori este prezentată în tabelul 1.5.1. Operatorii relaționali compară două matrice sau două expresii matriceale, element cu element. Rezultatul este o matrice de aceeași dimensiune cu a matricelor care se compară, cu elemente 1 când relația este ADEVĂRATĂ și cu elemente 0 când relația este FALSE. Primii patru operatori compară numai partea reală a operatorilor, partea imaginară fiind ignorată, iar ultimii doi operatori tratează atât partea reală cât și partea imaginară. Dacă unul dintre operanzi este un scalar și celălalt este o matrice, scalarul se „extinde” până la dimensiunea matricei.

Forma generală de utilizare a acestor operatori este:

```
rezultat=expresie_1 operator_relațional expresie_2.
```

Tab. 1.5.1 Operatorii relaționali

Operatorii relaționali	Semnificația
<	mai mic
<=	mai mic sau egal
>	mai mare
>=	mai mare sau egal
= =	identic
~ =	diferit

Pentru combinarea a două sau mai multe expresii logice se utilizează operatorii logici prezentați în tabelul 1.5.2. Operatorii logici și, sau sau compară doi scalari sau două matrice de dimensiuni egale.

Tab. 1.5.2 Operatorii logici

Operatori logici	Simbol Matlab	Prioritatea
NU (not)	~	1
ȘI (and)	&	2
SAU (or)		3

Pentru cazul în care cei doi operanzi sunt matrice, se operează element cu element. Operatorul logic NU (sau complementul logic) este operator unar. Acești operatori au prioritate mai mică decât operatorii relaționali și aritmétici.

Operatorii logici au prioritate mai mică decât operatorii relaționali sau cei aritmétici. În tabelul 1.5.3. este dată tabela de adevăr a operatorilor logici.

Tab. 1.5.3 Tabelul de adevăr al operatorilor logici

A	B	~A	A B	A&B
FALS	FALS	ADEV.	FALS	FALS
FALS	ADEV.	ADEV.	ADEV.	FALS
ADEV.	FALS	FALS	ADEV.	FALS
ADEV.	ADEV.	FALS	ADEV.	ADEV.

Instrucțiunea if poate fi implementată ca instrucțiune if simplă, sau poate include clauzele else sau elseif. Forma generală a unei instrucțiuni if simplă este următoarea:

```
if expresie logică
    grup de instrucțiuni
end
```

Dacă expresia logică este adevărată, se execută grupul de instrucțiuni cuprins între instrucțiunea if și instrucțiunea end. În caz contrar, se trece la prima instrucțiune care urmează după instrucțiunea end.

Clausa else este folosită pentru a executa un set de instrucțiuni (grupul de instrucțiuni A) dacă expresia logică este adevărată și un alt set de instrucțiuni (grupul de instrucțiuni B) dacă expresia logică este falsă.

Forma generală este:

```
if expresie logică
    grup de instrucțiuni A
else
    grup de instrucțiuni B
end
```

Dacă funcția de calculat are mai multe nivele de instrucțiuni if-else este recomandată utilizarea clauzei elseif. Aceasta are următoarea formă generală:

```
if expresie logică 1
    grup de instrucțiuni A
elseif expresie logică 2
    grup de instrucțiuni B
elseif expresie logică 3
    grup de instrucțiuni C
end
```

Această instrucțiune este evaluată în modul următor:

- dacă expresia logică 1 este adevărată, se execută numai grupul de instrucțiuni A;

- dacă expresia logică 1 este falsă și expresia logică 2 este adevărată, se execută numai grupul de instrucțiuni B;
- dacă expresiile logice 1 și 2 sunt false și expresia logică 3 este adevărată, se execută numai grupul de instrucțiuni C;
- dacă mai multe expresii logice sunt adevărate, prima expresie logică adevărată determină care grup de instrucțiuni se execută prima dată;
- dacă nicio expresie logică nu este adevărată, nu se execută niciun grup de instrucțiuni din structura if.

Clausa `elseif` poate fi combinată cu clauza `else` într-o structură generală de forma:

```
if expresie logică 1
    grup de instrucțiuni A
elseif expresie logică 2
    grup de instrucțiuni B
elseif expresie logică 3
    grup de instrucțiuni C
else
    grup de instrucțiuni D
end
```

În acest caz, dacă nicio expresie logică nu este adevărată, se execută grupul de instrucțiuni D.

1.5.2. Instrucțiunea repetitivă „for”

Instrucțiunea `for` permite repetarea unui grup de instrucțiuni din corpul buclei, de un anumit număr de ori. Structura generală a acestei instrucțiuni este următoarea:

```
for index = expresie
    grup de instrucțiuni
end
```

unde:

`index` - este numele contorului;

`expresie` - este o matrice, un vector sau un scalar;

`grup de instrucțiuni` - este orice expresie Matlab.

În aplicații, „`index = expresie`” are de cele mai multe ori următoarea formă:

```
k = inițial:pas:final
```

unde:

`inițial` - este prima valoare a variabilei k;

`pas` - este pasul, dacă acesta este omis, este considerat implicit 1;

`final` - este cea mai mare valoare pe care o poate lua variabila k.

La fiecare pas de calcul, „`index`” are valoarea unuia dintre elementele expresiei. Dacă expresia este o matrice, ciclarea se face pe coloane. Pentru un ciclu `for` cu pasul negativ sau neîntreg se generează mai întâi un vector cu pasul și limitele dorite și apoi se citesc valorile acestuia în cadrul buclei `for`. La folosirea buclei `for` trebuie respectate următoarele reguli:

- indexul buclei `for` trebuie să fie o variabilă;
- dacă expresia este o matrice goală, bucla nu se execută. Se va trece la următoarea instrucțiune după `end`;

- dacă expresia este un scalar, bucla se execută o singură dată cu indexul dat de valoarea scalarului;
- dacă expresia este un vector linie, bucla se execută de atâtea ori câte elemente are vectorul, de fiecare dată indexul având valoarea egală cu următorul element din vector;
- dacă expresia este o matrice, indexul va avea la fiecare iterare valorile continute în următoarea coloană a matricei;
- la terminarea ciclului `for`, indexul are ultima valoare utilizată;
- dacă se utilizează operatorul două puncte „`:`” pentru a defini expresia, bucla se execută de $n = \left\lceil \frac{\text{final} - \text{initial}}{\text{pas}} \right\rceil + 1$ ori, dacă `n` este pozitiv, și nu se execută dacă `n` este negativ. Prin `[]` s-a notat valoarea întreagă a numărului.

1.5.3. Instrucțiunea repetitivă „while”

Instrucțiunea repetitivă `while` este o structură care se utilizează pentru repetarea unui set de instrucțiuni, atât timp cât o condiție specificată este adevărată. Formatul general al acestei instrucțiuni este următorul:

```
while expresie
    grup de instrucțiuni
end
```

Grupul de instrucțiuni este executat cât timp expresia este adevărată. Dacă expresia nu este adevărată, se trece la executarea primei instrucțiuni care urmează după `end`. Dacă expresia este întotdeauna adevărată logic, bucla devine infinită.

Notă. Într-o buclă infinită se ieșe forțat prin apăsarea concomitentă a tastelor `[Ctrl]+[C]` !

1.5.4. Instrucțiunea „break”

Instrucțiunea `break` se utilizează pentru a ieși dintr-o buclă înainte ca aceasta să se fi terminat. Se recomandă utilizarea ei dacă o condiție de eroare este detectată în interiorul unei bucle. Această instrucțiune încetează execuția ciclurilor `for` și `while`. În cazul unor cicluri imbricate, `break` comandă ieșirea din ciclul cel mai interior. Se apelează cu sintaxa: `break`.

1.5.5. Instrucțiunea „return”

Instrucțiunea `return` comandă o ieșire normală din fișierul M către funcția care l-a apelat sau către tastatură. Se apelează cu sintaxa: `return`.

1.5.6. Instrucțiunea „error”

Instrucțiunea `error` permite afișarea unor mesaje la întâlnirea unei erori. Se apelează cu sintaxa: `error('mesaj')`.

După afișarea textului „`mesaj`” controlul este redat tastaturii.

1.5.7. Funcții de control logic

În continuare, se vor prezenta pe scurt funcțiile de control logic. În Matlab există următoarele funcții de control logic:

exist - verifică dacă variabilele sau funcțiile argument sunt definite;
any - testează dacă cel puțin un element al unei matrice verifică o condiție logică dată (testează dacă un vector are cel puțin un element diferit de zero; returnează 1 dacă condiția este verificată și 0 în caz contrar);
all - testează dacă toate elementele unei matrice verifică o condiție logică dată (testează dacă un vector are toate elementele diferite de zero; returnează 1 dacă condiția este adevarată și 0 în caz contrar);
find - returnează indicii elementelor diferite de zero;
isnan - testează dacă elementele unei matrice sunt NaN²;
isinf - testează dacă elementele unei matrice sunt infinite;
finite - testează dacă elementele unei matrice sunt finite.
 Aceste funcții sunt des asociate cu instrucțiunea **if**.

1.6. Variabile speciale în Matlab

eps - variabilă în care este memorată variabila relativă, pentru calcule efectuate în virgulă mobilă, valoarea implicită fiind 2.2204e-016;
pi - variabilă permanentă ce are asociată valoarea 3.14159265358;
 $i = j = \sqrt{-1}$ - variabilă folosită pentru reprezentarea unității imaginare;
inf - variabilă folosită pentru reprezentarea lui plus infinit în aritmetică IEEE;
nargin - variabilă permanentă folosită pentru testarea numărului argumentelor de intrare ce trebuie introduse pentru apelarea unei funcții;
nargout - variabilă permanentă folosită pentru testarea numărului argumentelor de ieșire ale unei funcții;
flops - returnează numărul de operații în virgulă mobilă efectuate de către calculator;
computer - variabilă folosită pentru obținerea informațiilor referitoare la tipul calculatorului și numărul maxim de elemente pe care le poate gestiona versiunea respectivă de Matlab;
realmax - reprezintă cea mai mare valoare pozitivă în virgulă mobilă care poate fi folosită în calcule, respectiv 1.7977e+308;
realmin - reprezintă cea mai mică valoare pozitivă în virgulă mobilă care poate fi folosită în calcule, respectiv 2.2251e-308;
isieee - funcție ce returnează 1 dacă calculatorul este compatibil cu aritmetică IEEE și respectiv 0 în caz contrar;
version, ver - funcții pentru determinarea versiunii Matlab și a toolbox-urilor instalate pe calculator.

1.7. Exerciții propuse

Exercițiu 1.7.1

Să se introducă și să se afișeze următoarele matrice:

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}; \quad B = [1 \ 3 \ 5 \ 7]; \quad C = \begin{bmatrix} 1 \\ 4 \\ 9 \end{bmatrix};$$

² NaN – Not a Number, nu este un număr

Să se afișeze următoarele elemente:

- elementul de pe linia 2 și coloana 1 din matrice;
- elementul al patrulea din vectorul linie;
- elementul al treilea din vectorul coloană.

Exercițiu 1.7.2

Să se scrie un fișier funcție, denumit „funcție1”, în care să se calculeze următoarea funcție:

$$f(x,y) = \begin{cases} x^3 + y^3, & \text{dacă } 0 \leq x - y \leq 10 \\ x^2 + y^2, & \text{dacă } x - y < 0 \text{ și } y \geq 0 \\ (x - y)^2, & \text{în restul cazurilor} \end{cases}$$

Această funcție se va implementa în două moduri:

- Folosind instrucțiunea **if-else**;
- Folosind instrucțiunea **if-elseif**.

Să se calculeze următoarele valori: f(1,3), f(1,-3), f(-3,1), f(-4,-1).

Exercițiu 1.7.3

$$\text{Fie } X = \begin{bmatrix} 1 & 0 & 2 \\ 0 & 0 & 1 \\ 0 & 0 & 4 \end{bmatrix}$$

Să se determine:

- coloanele în care matricea are cel puțin un element diferit de zero;
- coloanele în care matricea are toate elementele diferite de zero;
- coloanele în care matricea are cel puțin un element mai mare ca -1.

Exercițiu 1.7.4

Să se calculeze suma primelor 100 de numere naturale. Se va calcula în două moduri: folosind instrucțiunea repetitivă **for** și folosind instrucțiunea repetitivă **while**. Pentru rezolvare, se vor deschide două fișiere de tip M, în care se vor scrie cele două programe.

Exercițiu 1.7.5

Să se scrie un program care calculează suma elementelor vectorului $X = [5 \ 2 \ -9 \ 10 \ -1 \ 9 \ 1]$ până când întâlnește un număr mai mare de 8.

CALCUL NUMERIC ÎN MATLAB

OBIECTIVE

- Introducerea și manipularea matricelor și vectorilor.
- Operații și funcții cu matrice.
- Introducerea și manipularea polinoamelor.

2.1. Operații aritmetice

În Matlab, calculele aritmetice asupra tablourilor de date pot fi operații după regulile calculului matriceal (operații cu matrice) și operații după regulile calculului scalar (operații cu vectori).

În tabelul 2.1.1 sunt prezentate operații aritmetice din Matlab.

Tab. 2.1.1 Operatorii aritmetici Matlab

Operația	Scalari	Matrice	Vectori
Adunarea	+	+	+
Scăderea	-	-	-
Înmulțirea	*	*	*
Împărțirea la stânga	\	\	\
Împărțirea la dreapta	/	/	/
Ridicarea la putere	^	^	^
Transpunerea	'	'	'

Ordinea operațiilor aritmetice este aceeași cu cea cunoscută în matematică elementară, a operațiilor aritmetice standard (tabelul 2.1.2). Deși variabilele memorate au un interval foarte mare, cel mai adesea sunt în limitele 10^{-308} și 10^{308} (vezi funcțiile Matlab **realmax** și **realmin**). Dacă rezultatul unui calcul este mai mare de 10^{308} (uneori, este posibil ca rezultatul unei expresii să depășească limitele menționate anterior), Matlab-ul înregistrează ∞ (**Inf**). Dacă rezultatul unui calcul este mai mic decât 10^{-308} , calculatorul înregistrează valoarea zero. În Matlab, rezultatul

împărțirii cu zero este ∞ . În acest caz este afișat mesajul de atenționare „Warning: Divide by zero”, dar calculele sunt continue cu operandul ∞ .

```
>> x=2.5*10^200;           >> x=2.5*10^200;
>> y=10^200;              >> y=0;
>> z=x*y                  >> z=x/y
z =                                Warning: Divide by zero.
Inf                               z =Inf
```

Tab. 2.1.2 Ordinea operațiilor aritmetice

Ordinea	Operația
1	parantezele
2	ridicarea la putere
3	înmulțirea și împărțirea
4	adunarea și scăderea

2.1.1. Operații aritmetice cu scalari

În tabelul 2.1.3 sunt prezentate operațiile aritmetice între doi scalari, fiind prezentată atât forma algebraică cât și forma Matlab.

Tab. 2.1.3 Forma Matlab a operatorilor scalari

Operația	Forma algebraică	Forma Matlab
Adunarea	$a+b$	$a+b$
Scăderea	$a-b$	$a-b$
Înmulțirea	$a \times b$	$a \cdot b$
Împărțirea la dreapta	$a:b$	a/b
Împărțirea la stânga	$b:a$	a/b
Ridicarea la putere	a^b	a^b

Expresiile aritmetice pot fi evaluate și rezultatul memorat în variabila specificată. O valoare introdusă fără nominalizare este asignată variabilei ans (answer). În variabila ans este memorată, fără excepție, valoarea ultimei variabile căreia nu i s-a atribuit un nume.

2.1.2. Operații aritmetice cu vectori

Operațiile cu vectori sunt operații aritmetice între elementele situate în aceeași poziție a vectorilor, cunoscute sub numele de operații element cu element. Pentru a

preciza că o operație se efectuează element cu element între componentele a două matrice de aceeași dimensiune, se utilizează operatorul corespunzător operației precedat de punct (vezi tabelul 2.1.1). Dacă unul dintre operanzi este un scalar, acesta operează cu fiecare element al vectorului.

2.1.3. Operații aritmetice cu matrice

Operațiile uzuale de algebră liniară cu matrice sunt simbolizate cu semnele grafice prezentate în tabelul 2.1.1 și se efectuează după regulile cunoscute din calculul matriceal.

Operația de adunare a două matrice este simbolizată cu operatorul plus. Instrucțiunea:

$$Z=X+Y,$$

reprezintă adunarea matricelor X și Y, rezultând elementele:

$$Z(i,j) = X(i,j) + Y(i,j)$$

Matricele X și Y trebuie să aibă aceeași dimensiune, în afara cazului când una dintre matrici, X sau Y, este un scalar. Un scalar poate fi adunat cu orice matrice.

Operația de scădere a două matrice este simbolizată cu operatorul minus. Instrucțiunea:

$$Z=X-Y,$$

reprezintă scăderea matricelor X și Y, rezultând elementele:

$$Z(i,j) = X(i,j) - Y(i,j)$$

Matricele X și Y trebuie să aibă aceeași dimensiune, în afara cazului când una dintre ele este un scalar.

Operația de înmulțire a două matrice este simbolizată cu operatorul stelută. Instrucțiunea:

$$Z=X \cdot Y,$$

reprezintă matricea produs având elementele:

$$Z(i,j) = \sum_k X(i,k)Y(k,j)$$

Produsul matriceal este posibil numai dacă numărul de coloane al matricei X este egal cu numărul de linii al matricei Y. Produsul matrice-vector este un caz particular al cazului general al produsului matrice-matrice. De asemenea, un scalar poate fi înmulțit cu orice matrice.

Pentru operația de împărțire a matricelor, în Matlab există două cazuri: împărțirea la dreapta și împărțirea la stânga. Operația de împărțire la dreapta a două matrice este simbolizată cu operatorul slash. Instrucțiunea:

$$Z=X/Y,$$

reprezintă împărțirea la dreapta a matricelor X și Y, și este identică cu:

$$Z = X \cdot Y^{-1}$$

Operația de împărțire la stânga a două matrice este simbolizată cu operatorul backslash. Instrucțiunea:

$$Z=X\backslash Y,$$

reprezintă împărțirea la stânga a matricelor X și Y și este identică cu:

$$Z = X^{-1} * Y$$

Dacă unul dintre operanzi este un scalar, operația nu este posibilă. Operația de ridicare la putere a unei matrice este simbolizată cu operatorul \wedge . Următoarea instrucție:

$$Z = X^p,$$

reprezintă ridicarea la puterea p a matricei X. Expresia are sens doar pentru X matrice pătrată și pentru p scalar. Dacă p este un întreg pozitiv, ridicarea la putere este obținută prin înmulțiri repetitive, iar dacă p este un întreg negativ, matricea X este mai întâi inversată și apoi se înmulțesc inversele de p ori.

Operația de transpunere a unei matrice este simbolizată cu operatorul apostrof. Cu instrucțunea:

$$Z=Y',$$

liniile matricei Y devin coloanele matricei transpușe Z. Dacă elementele matricei Y sunt numere complexe, operația de transpunere returnează conjugata transpușei, adică:

$$Z(i,j) = \text{conj}(Y(i,j)) = \text{real}(Y(i,j)) - i * \text{imag}(Y(i,j))$$

2.2. Generarea matricelor

Deși în Matlab nu există instrucții pentru declararea tipurilor de variabile, matricele se autodimensionează în timpul utilizării. Pentru a crește viteza de lucru se procedea că la crearea unei matrice goale. Acest lucru se face la începutul sesiunii de lucru sau la apelarea unui program. Declararea unei matrice goale se face cu instrucție:

$$X = [],$$

care asignează lui X matricea de dimensiuni 0x0. Orice matrice goală trebuie să aibă cel puțin una din dimensiuni zero.

Pentru a afla dacă o matrice este goală, se folosește funcția Matlab isempty care se apelează cu sintaxa:

$$r = \text{isempty}(X).$$

Funcția întoarce rezultatul r=1 dacă matricea este goală și r=0 în caz contrar.

Matricea unitate este o matrice care are toate elementele egale cu 1 și poate fi generată cu funcția ones ce se apelează cu sintaxa:

$$U = \text{ones}(m, n),$$

unde m și n sunt scalari ce definesc dimensiunea matricei.

Pentru a genera matricea zero se folosește funcția zeros ce se apelează cu sintaxa:

$$O = \text{zeros}(m, n)$$

Matricea identitate (are elementele de pe diagonala principală 1) se generează cu funcția eye care este apelată cu sintaxa:

$$I = \text{eye}(m, n).$$

2.3. Generarea vectorilor cu pas liniar și cu pas logaritmic

În Matlab se pot genera vectori cu pas liniar sau vectori cu pas logaritmic. Generarea vectorilor cu pas liniar implică cunoașterea limitelor intervalului (a_{\min}, a_{\max}) și a pasului dintre două elemente sau a numărului de elemente ale vectorului (N). Dacă se cunosc limitele intervalului și pasul dintre două elemente, vectorul se generează cu instrucție:

$$x = a_{\min} : \text{pas} : a_{\max}$$

Numărul de elemente al vectorului x este: $N = \left\lceil \frac{a_{\max} - a_{\min}}{\text{pas}} \right\rceil + 1$,

unde $\lceil \cdot \rceil$ semnifică partea întreagă a numărului. Dacă pasul este mai mare ca zero, este necesar ca $a_{\min} < a_{\max}$, iar dacă pasul este mai mic decât zero este necesar ca $a_{\min} > a_{\max}$.

Dacă se cunosc limitele intervalului și numărul de elemente ale vectorului, acesta se generează cu ajutorul funcției linspace, care este apelată cu următoarea sintaxă:

$$x = \text{linspace}(a_{\min}, a_{\max}, N),$$

rezultând un vector cu N elemente (dacă valoarea N nu este precizată, atunci aceasta este considerată implicit egală cu 100), pasul dintre două elemente fiind: $\text{pas} = \frac{a_{\max} - a_{\min}}{N - 1}$.

Pentru generarea vectorilor cu pas logaritmic se utilizează funcția logspace ce se apelează cu sintaxa:

$$x = \text{logspace}(a_{\min}, a_{\max}, N).$$

Vectorul x conține N elemente distribuite logaritmice între decadele $[10^{a_{\min}}, 10^{a_{\max}}]$. Dacă N nu este precizat, este luat implicit 50.

2.4. Calcule cu matrice

Formularea matricei a problemelor a condus la simplificarea metodelor de rezolvare și a făcut posibilă extinderea unor soluții deja cunoscute la domeniul nostru. În continuare se vor prezenta funcții Matlab pentru manipularea matricelor și pentru analiza matricială. Cunoașterea acestora este necesară pentru rezolvarea cu ajutorul Matlab-ului a diverselor probleme de automatizări, teoria sistemelor și identificarea sistemelor.

2.4.1. Manipularea matricelor

Elementele individuale ale unei matrice pot fi apelate cu numele acestora urmat de doi indici, cuprinși între paranteze rotunde și separați prin virgulă. Primul indice semnifică linia, iar al doilea coloana în care se găsește elementul apelat. Indicii pot fi scalari sau vectori. Indicii vectori permit definirea unor submatrice prin care se pot referi părți disparate dintr-o matrice. Utilizarea semnului două puncte „:”, în locul indicilor pentru linii sau pentru coloane, presupune considerarea tuturor elementelor pe linii sau respectiv pe coloane.

Crearea matricelor mari, precum și manipularea acestora, se face cu multă flexibilitate dacă se utilizează indici vectori.

Inversarea coloanelor unei matrice A se face cu instrucțiunea:

$A=A(:,n:-1:1)$,

iar inversarea liniilor se face cu instrucțiunea:

$A=A(n:-1:1,:)$.

>> A=[1 2;3 4]

A =

1	2
3	4

>> a=A(:,2:-1:1)

a =

2	1
4	3

>> A=[1 2;3 4]

A =

1	2
3	4

>> a=A(2:-1:1,:)

a =

3	4
1	2

De o mare importanță este utilizarea fără indici, care are ca efect transformarea matricei într-un vector coloană, citind coloanele una după alta. Astfel, secvența următoare:

A=[1 2;3 4]
b=A(:),

are ca rezultat transformarea matricei A într-un vector coloană b.

>> A=[1 2;3 4]

A =

1	2
3	4

>> b=A(:)

b =
1
3
2
4

Dacă în partea stângă a expresiei este asignată o instrucțiune de forma A(:), matricea A trebuie să existe dintr-o utilizare anterioară. Spre exemplu, secvența:

A=[1 2;3 4]
A(:)=11:14

returnează matricea:

A = $\begin{bmatrix} 11 & 13 \\ 12 & 14 \end{bmatrix}$

Pentru extragerea vectorilor cu elemente decupate din alți vectori:

- $j:k$ - selectează elementele $[j, j+1, j+2, \dots, k]$ ale unui vector; dacă $j > k$ vectorul rezultat este gol;
- $j:i:k$ - selectează elementele $[j, j+i, j+2i, \dots, k]$ ale unui vector; vectorul rezultat este gol dacă $i > 0$ și $j > k$ sau dacă $i < 0$ și $j < k$.

În cazul selectării liniilor și coloanelor matricelor, se pot utiliza opțiunile:

- $A(:,j)$ - selectează coloana j a matricei A;
- $A(i,:)$ - selectează linia i a matricei A;
- $A(:,:,)$ - selectează toată matricea A;
- $A(j,k)$ - selectează elementele $A(j), A(j+1), \dots, A(k)$ ale matricei A;
- $A(:,j,k)$ - selectează toate liniile și coloanele de la j la A(:,j), A(:,j+1), ..., A(:,k), ale matricei A;
- $A(:)$ - selectează toate elementele matricei A, privite ca o singură coloană (începând cu prima).

Alt caz interesant este extragerea submatricelor prin vectori cu elemente 0 și 1. Vectorii cu elemente 0 și 1, creați de operatorii logici sau relaționali, au proprietăți deosebite ce pot conduce la scrierea unor programe foarte compacte.

Dacă A este o matrice de dimensiunea $m \times n$ și L este un vector de lungime cu elemente 0 și 1, instrucțiunea:

B=A(L,:)

returnează în matricea B toate elementele din liniile matricei A pentru care elemenții corespunzător ca poziție din vectorul L este 1.

O matrice poate fi definită pe baza altor matrice de dimensiuni inferioare, ca exemplul de mai jos:

C=[A eye(4); ones(A) A^2].

Matricea C este compusă din matricea A (de dimensiune 4×4), matricea identitate (de dimensiune 4×4), matricea unitate (de dimensiune egală cu matricea A) și matricea A ridicată la pătrat. Matricele folosite în asamblare trebuie să fie consistente în dimensiuni, adică să determine blocuri care se integrează compact în tabloul matricei rezultat, în caz contrar vor rezulta mesaje de eroare.

Redimensionarea unei matrice se poate face cu ajutorul funcției reshape. Această funcție redimensionează o matrice A cu dimensiunea $1 \times p$ într-o altă matrice B, dimensiunea $m \times n$. Pentru ca această operație să fie posibilă este necesar ca matricea A să aibă același număr de elemente cu matricea B, deci $1 \times p = m \times n$. Sintagma funcției este:

B=reshape(A,m,n).

Elementele matricei B (în ordinea succesivă a coloanelor și pe fiecare coloană de sus în jos) sunt elementele matricei argument A (citite de sus în jos și de la stânga la dreapta).

Crearea unei matrice diagonale se face cu funcția diag, folosind sintaxa:

Y=diag(X,k).

unde x este vectorul sau matricea asupra căreia se operează, iar argumentul optional k indică diagonala acesteia, cu următoarea semnificație:

- $k=0$ - diagonala principală;
- $k>0$ - indică diagonala k de deasupra celei principale;
- $k<0$ - indică diagonala k de sub cea principală.

Dacă x este un vector cu n componente, funcția $\text{diag}(x, k)$ generează o matrice pătrată de ordinul $n + \text{abs}(k)$, cu elementele lui x pe diagonala k . Dacă x este o matrice, funcția $\text{diag}(x, k)$ extrage un vector coloană format din elementele diagonalei k a matricei x .

Pentru a crea o matrice superior și inferior triunghiulară, se folosesc funcțiile Matlab **tril** și **triu**, apelate cu sintaxele:

```
Y=tril(X,k),
Y=triu(X,k),
```

unde argumentele au aceeași semnificație ca mai sus.

Funcția **triu**(X, k) înlocuiește cu zero toate elementele matricei X de sub diagonala k . Funcția **tril**(X, k) înlocuiește cu zero toate elementele matricei X de deasupra diagonalei k .

2.4.2. Analiza matriceală

În acest paragraf se vor prezenta funcțiile și modurile de calcul ale determinantului unei matrice, ale inversei unei matrice, ale normei și rangului unei matrice. Calculul determinantului unei matrice se face cu funcția **det** ce are următoarea sintaxă:

```
D=det(X).
```

Prin definiție, inversa unei matrice pătrate A este matricea A^{-1} , care satisface relația:

```
AA^{-1} = A^{-1}A = I,
```

unde I este matricea unitate. Se știe că o matrice este inversabilă numai dacă determinantul acesteia este diferit de zero, adică matricea este nesingulară. Inversa se calculează cu funcția **inv**, ce este apelată cu sintaxa:

```
Y=inv(X).
```

Rangul unei matrice reprezintă numărul de linii sau coloane liniar independente ale acesteia și se determină cu funcția **rank**. Această funcție este apelată cu sintaxa:

```
r=rank(X,tol).
```

Ea returnează numărul de valori singulare ale lui x , mai mari decât parametrul optional **tol**.

Norma unei matrice (sau a unui vector) este un scalar care dă o măsură a mărimii elementelor matricei (sau vectorului). Normele vectorilor sau matricelor se calculează cu funcția **norm**:

```
n=norm(X).
```

Această funcție calculează mai multe tipuri de norme pentru matrice sau vectori, în funcție de sintaxă. Detalii se pot afla consultând help-ul funcției.

2.5. Calcule numerice cu polinoame

În continuare se prezintă câteva funcții Matlab pentru evaluarea polinoamelor, calculul derivatei unui polinom, generarea unui polinom când se cunosc rădăcinile acestuia precum și pentru descompunerea în fracții simple. Polinomul este o funcție de o singură variabilă, care are următoarea formă generală:

$$f(x) = a_1x^N + a_2x^{N-1} + \dots + a_{N-1}x^2 + a_Nx + a_{N+1}$$

unde x este variabila, iar a_i ($i = 1 \dots N+1$) sunt coeficienții polinomului. În sintaxa Matlab, polinoamele sunt reprezentate cu un vector linie care conține coeficienții în ordinea descrescătoare a puterilor variabilei. Matlab-ul oferă mai multe posibilități de evaluare a polinoamelor, cea mai simplă metodă fiind evaluarea cu scalari, adică pentru o singură valoare a variabilei. A doua metodă constă în evaluarea polinomului în mai multe puncte. În acest caz se efectuează operații cu vectori. Dimensiunea matricei în care se returnează valorile polinomului trebuie să fie identică cu cea a matricei care conține punctele în care se face evaluarea. A treia metodă de evaluare constă în utilizarea funcției **polyval**, care se apelează cu sintaxa:

```
f=polyval(p,s).
```

Funcția evaluatează polinomul definit de vectorul p în punctul s . Dacă s este un vector sau o matrice, polinomul este evaluat pentru fiecare dintre elementele acestuia.

Operațiile aritmetice de adunare și scădere ale polinoamelor presupun adunarea și scăderea coeficienților de același ordin. Deoarece în Matlab aceste operații necesită vectori de aceeași dimensiune, lungimea vectorilor coeficienților este dată de polinomul cu puterea cea mai mare. Odată stabilită această dimensiune, spre exemplu M , vectorii coeficienților polinoamelor care au puterea maximă N , mai mică decât M , vor fi completăți la stânga cu coeficienții zero până la dimensiunea M .

Înmulțirea a două polinoame este echivalentă unei operații de convoluție și se realizează cu funcția Matlab **conv**:

```
c=conv(a,b),
```

unde a și b sunt vectorii polinoamelor care se înmulțesc, iar c este vectorul coeficienților polinomului produs. Împărțirea a două polinoame este echivalentă unei operații de deconvoluție și se realizează cu funcția Matlab **deconv**:

```
[d,r]=deconv(a,b),
```

unde a și b sunt vectorii polinoamelor deîmpărțit și împărțitor, d este vectorul coeficienților polinomului cît, iar r este vectorul polinomului rest.

Descompunerea în fracții simple reprezintă scrierea raportului a două polinoame ca sumă de fracții cu polinoame de ordinul unu.

$$\frac{B(x)}{A(x)} = \frac{r(1)}{x-p(1)} + \frac{r(2)}{x-p(2)} + \dots + \frac{r(n)}{x-p(n)} + k(x)$$

Dacă polul $p(j)$ (reprezintă una din rădăcinile numitorului) este de ordinul m , descompunerea conține termeni de forma:

Îndrumar de laborator

Teoria Sistemelor

$$\frac{r(j)}{x-p(j)} + \frac{r(j+1)}{(x-p(j))^2} + \dots + \frac{r(j+m-1)}{(x-p(j))^m}$$

Descompunerea în fracții simple se realizează cu funcția Matlab **residue**, ce se apelează cu sintaxele:

`[r,p,k]=residue(B,A)`,
`[B,A]=residue(r,p,k)`,

unde: A și B sunt vectorii coeficienților polinoamelor de la numitor și numărător, x este vectorul coloană al resturilor, p este vectorul coloană al polilor, iar k este vectorul linie al termenilor liberi.

Pentru a calcula derivata unui polinom în Matlab se utilizează funcția **polyder**, care se apelează cu sintaxele:

`D=polyder(C)`

$D(x)=C'(x)$

`D=polyder(A,B)`

$D(x)=(A(x) \cdot B(x))' = A'(x) \cdot B(x) + A(x) \cdot B'(x)$

$$[M,N]=\text{polyder}(A,B) \quad \frac{M(x)}{N(x)} = \left(\frac{A(x)}{B(x)} \right)' = \frac{A'(x) \cdot B(x) - A(x) \cdot B'(x)}{(B(x))^2}$$

Dacă funcția $f(x)$ este un polinom de gradul N, atunci $f(x)=0$ are N rădăcini, care pot fi reale sau complexe. În cazul în care coeficienții polinomului sunt numere reale, rădăcinile complexe sunt perechi complex conjugate. Funcția Matlab **roots** determină rădăcinile polinoamelor și se apelează cu sintaxele:

`r=roots(p)`.

Funcția Matlab **poly** ce determină coeficienții unui polinom al cărui rădăcini sunt cunoscute, se apelează cu sintaxa:

`p=poly(r)`.

În ambele cazuri, p este un vector linie care conține coeficienții polinomului, iar x este un vector coloană care conține rădăcinile polinomului.

2.6. Exerciții propuse

Exercițiu 2.6.1

Fie următorul sistem de trei ecuații cu trei necunoscute:

$$\begin{cases} 3x + 2y - z = 10 \\ -x + 3y + 2z = 5 \\ x - y - z = -1 \end{cases}$$

Să se rezolve sistemul prin împărțirea matricelor. Descrierea matriceală a sistemului este următoarea:

$$A \cdot X = B,$$

unde: A este matricea coeficienților necunoscute, de dimensiune 3×3 . Coeficienții aceleiași necunoscute sunt pe aceeași coloană. X este matricea necunoscute, de dimensiune (3×1) , de dimensiune. B este matricea termenilor libri, de dimensiune 3×1 .

Îndrumar de laborator

Teoria Sistemelor

Exercițiu 2.6.2

Să se rezolve sistemul de la exercițiul 2.6.1 folosind matricea inversă.

Exercițiu 2.6.3

Fie polinoamele:

$$f(x) = 4x^6 - 2x^4 + 3x^2 - x + 4$$

$$g(x) = 2x^2 + 5x - 16$$

Să se calculeze:

- $s(x) = f(x) + g(x)$ și $d(x) = f(x) - g(x)$.
- $a = f(3) - g(7)$; $f(b)$ și $g(b)$ unde $b = \{1,3,4,9\}$.
- $f(x) * g(x)$ și $f(x)/g(x)$.
- $(f(x) * g(x))'$ și $(f(x)/g(x))'$.
- Să se rezolve: $f(x) = 0$ și $g(x) = 0$.

Exercițiu 2.6.4

Să se descompună în fracții simple expresia:

$$\frac{B(x)}{A(x)} = \frac{x^3 + 2x^2 - 2}{x^2 + 1}$$

Exercițiu 2.6.5

Să se determine o matrice H compusă din matricea $G = \begin{bmatrix} 11 & 12 \\ 4 & 6 \end{bmatrix}$

dimensiune 2×2 , matricea identitate 2×2 , matricea $2 \cdot G$ și matricea unitate dimensiunile matricei G.

Exercițiu 2.6.6

$$\text{Se consideră matricea: } A = \begin{bmatrix} 11 & 12 & 13 & 14 \\ 2 & 3 & 4 & 5 \\ 21 & 22 & 23 & 24 \\ 31 & 32 & 33 & 34 \end{bmatrix}.$$

- Să se determine un vector B , care are ca elemente, al doilea, al treilei și al patrulea termen de pe linia a doua a matricei A.
- Să se determine un vector C , care are ca elemente linia a doua a matricei A.
- Să se determine o matrice D , care are ca elemente prima și a treia linie din matricea A..
- Să se determine o matrice F , ce conține elementele matricei A, în care s-a schimbat coloana a treia cu coloana a doua.
- Să se șteargă a doua coloană a matricei A.
- Să se determine un vector E , format din toate elementele matricei A.

REPREZENTĂRI GRAFICE ÎN MATLAB

OBJECTIVE

- Reprezentări grafice 2D.
- Reprezentări grafice 3D.

3.1. Introducere

Matlab-ul oferă o varietate relativ largă de funcții pentru realizarea reprezentărilor grafice. Cu ajutorul funcțiilor oferite de Matlab se pot reprezenta grafice bidimensionale (2D) și tridimensionale (3D), reprezentări grafice în coordonate liniare, în coordonate logaritmice sau în coordonate polare. De asemenea, graficele realizate în Matlab pot fi marcate (se poate preciza titlul graficului, se pot plasa etichete pe axe, se poate nota un text pe grafic la o poziție impusă, sau la o poziție selectabilă cu mouse-ul, sau se poate trasa o rețea de linii ajutătoare pe grafic).

3.2. Reprezentări grafice bidimensionale (2D)

Reprezentarea graficelor bidimensionale se poate face în coordonate liniare, logaritmice, semilogaritmice sau polare. Studiul reprezentărilor grafice este aplicabil în cazul diferitelor tipuri de funcții de sisteme ce reprezintă esența și finalitatea studiului propus de acest laborator. În plus, ca și remarcă, toate sistemele cunoscute și studiate nu pot fi susținute acestei reguli. Așadar, toate aceste reprezentări grafice vor fi detaliate în continuare.

Pentru reprezentarea grafică în coordonate liniare se utilizează funcția `plot`, care poate fi apelată cu una dintre sintaxele:

<code>plot(y),</code>	<code>plot(x,y),</code>
<code>plot(x,y,'linie_tip'),</code>	<code>plot(x1,y1,x2,y2,x3,y3,...),</code>

unde:

`plot(y)` - reprezintă grafic argumentul `y` în funcție de indici, cu următoarele precizări:

- dacă argumentul `y` este complex, `plot(y)` este echivalent cu `plot(real(y),imag(y))`;
- dacă `y` este un vector (linie sau coloană), funcția `plot` trasează graficul $y = y(i)$, unde $i = 1, 2, 3, \dots, n$ este numărul de ordine al elementului `y`;

- dacă y este o matrice de dimensiune $m \times n$, funcția **plot** trasează graficele $y_j = y_j(i)$, unde $i = 1, 2, 3, \dots, n$ este numărul de ordine al elementului de pe coloana j .

plot(x, y) - reprezintă grafic vectorul y funcție de vectorul x , cu următoarele precizări:

- dacă x este un vector, iar y este o matrice, atunci coloanele lui y sunt trasate în funcție de vectorul x ;
- dacă x și y sunt matrice de aceeași dimensiune, se reprezintă coloanele lui y în funcție de coloanele lui x .

plot(x1, y1, x2, y2) - reprezintă simultan mai multe grafice în același sistem de coordinate.

Pentru reprezentările grafice, se asociază fiecărei caracteristici un sir de 1 până la 3 caractere care definesc tipul liniei, tipul indicatorului și culoarea graficului. Aceste sururi de caractere trebuie cuprinse între apostroafe și menționate în combinația culoare-indicator sau culoare-linie-tip. Opțiunile pentru culori și tipuri de linii sau indicatori sunt prezentate în tabelul 3.2.1:

Tab. 3.2.1 Tipurile de culori și linii

Culorile	Tipurile de linii, indicatori
y galben	. punct
m mov (magenta)	o cerc
c albastru - deschis (cyan)	x semnul x
r roșu	+ semnul plus
g verde	- continuu
b albastru	* stea
w alb	: puncte
k negru	-. linie, punct
	-- linie întreruptă

Titlul unui grafic se poate preciza cu funcția:

```
title('text');
```

Axele unui grafic de asemenea se pot eticheta cu ajutorul funcțiilor **xlabel('text')** și **ylabel('text')**, iar un text se poate plasa pe grafic prin funcția:

```
text(x,y,'string');
```

unde:

x, y - reprezintă coordonatele unde se va plasa textul precizat în string.

Rețea de linii „grid”, se poate face setând pe **on** funcția **grid**; **grid off**. Stergerea rețelei de linii se face cu ajutorul comenzi **grid off**.

În cazul mai multor caracteristici într-o singură figură, acestea se pot indexa funcția **legend(string1, string2, string3, ...)**. În acest mod, se atașă caracteristicilor desenate un text explicativ (string1, string2, ...). Indexarea se poate face astfel încât să nu acopere reprezentarea grafică și poată mutată cu ajutorul mouse-ului astfel: se poziționează cursorul pe legendă, ia ajutorul butonului stâng apăsat se trage în poziția dorită. Suprapunerea mai multor reprezentări grafice în aceeași fereastră se face setând pe **on** funcția **hold**; **hold on**. Revenirea la starea initială se face lansând comanda **hold off**.

Pentru a realiza reprezentări grafice în coordonate logaritmice semilogaritmice se utilizează funcțiile **loglog**, **semilogx** și **semilogy**. Aceste funcții se apelează cu sintaxele:

```
loglog(x,y),
semilogx(x,y),
semilogy(x,y).
```

Funcția **loglog** scalează ambele axe utilizând logaritmul în baza 10, în timp ce funcțiile **semilogx** și **semilogy** scalează logaritmic numai axa x sau axa y, ceea ce înseamnă fiind scalată liniar. Modul de utilizare al acestor funcții este identic cu cel al funcției **plot**.

În Matlab se mai pot realiza și o serie de reprezentări grafice speciale cum fi reprezentarea în coordonate polare, reprezentarea graficelor cu bare, reprezentarea graficelor sub formă discretă (utilă pentru reprezentarea semnalelor discrete) și reprezentarea graficelor în trepte (utilă pentru reprezentarea semnalelor cuantizate). Reprezentarea în coordonate polare se face cu funcția **polar**, care poate fi apelată cu sintaxa:

```
polar(theta,r,'linie_tip').
```

Modul de folosire a opțiunii **linie_tip** este identic cu cel al funcției **plot**.

Reprezentarea graficelor cu bare se face cu funcția **bar**, care poate fi apelată cu una dintre sintaxele:

```
bar(y),
```

trasează un grafic de bare cu elementele vectorului y , adică $y = y$;

```
bar(x,y),
```

trasează un grafic de bare cu elementele vectorului y la locațiile specificate de vectorul x ; valorile lui x trebuie să fie egale depărtate și crescătoare;

```
[xb,yb]=bar(y) și [xb,yb]=bar(x,y),
```

calculează vectorii xb și yb , astfel încât **plot(xb,yb)** să poată trasa graficul de bare.

Reprezentarea discretă a datelor se face cu funcția **stem**, sub forma unor linii terminante cu cerculeți la extremitatea opusă axei. Se apelează cu sintaxa:

```
stem(x,y,'linie_tip').
```

Graficele în trepte sunt utilizate la reprezentarea diagramelor sistemele numerice de eşantionare și prelucrare a datelor.

Reprezentarea grafică în trepte se face cu funcția **stairs**, care se apelează cu una dintre sintaxele:

stairs(y),

trasează graficul în trepte al elementelor vectorului y;

stairs(x,y),

trasează graficul în trepte al elementelor vectorului y la locațiile specificate de vectorul x; valorile lui x trebuie să fie egale depărtate și crescătoare;

[xb,yb]=stairs(y) și **[xb,yb]=stairs(x,y),**

calculează vectorii xb și yb, astfel încât **plot(xb,yb)** să poată trasa graficul în trepte.

3.3. Reprezentări grafice tridimensionale (3D)

Reprezentările grafice 3D se pot realiza în trei moduri: reprezentarea de tip contur, reprezentarea de tip mesh și reprezentarea dreptelor și punctelor în spațiu tridimensional. Reprezentarea dreptelor se realizează folosind funcția **plot3** care este asemănătoare funcției **plot**.

Pentru ca o funcție de două variabile să poată fi reprezentată, ($z=f(x,y)$), este necesară generarea unei rețele de noduri în planul xOy . În nodurile rețelei se calculează valoarea funcției de reprezentat.

În Matlab generarea unei astfel de rețele se face cu funcția **[X,Y]=meshgrid(x,y)**. Această funcție transformă domeniul plan definit de cei doi vectori x și y monoton crescători și cu pas constant în matricele X și Y. Dacă vectorul x este de lungime n, iar vectorul y de lungime m, atunci matricele X și Y vor avea dimensiunea $n \times m$.

Funcțiile de două variabile se pot reprezenta cu ajutorul linijilor de contur, folosind funcțiile **contour** în cazul bidimensional, respectiv **contour3** în cazul tridimensional. Cele două funcții au sintaxe de apelare asemănătoare:

contour(Z),
contour(X,Y,Z).

Cu prima comandă se obține reprezentarea cu linii de contur a matricei Z, ce conține rezultatul evaluării funcției de două variabile reprezentate pe domeniul considerat. Implicit se trasează 10 linii de contur, de culori diferite. În cazul celei de a doua sintaxe, se precizează și matricele X și Y care au definit nodurile în domeniul considerat. În acest caz, autoscalarea celor două axe se va face în funcție de cei doi vectori.

Exemplul 3.1

Să se reprezinte grafic expresia $z=x^3+\left(\frac{y}{2}\right)^3$.

Se va realiza o reprezentare de tip contur și un grafic 3D de tip mesh.

% se generează valori pentru vectorii x și y
x=[-1:1:1];
y=[-2:1:2];

Se introduce comanda **meshgrid** care folosește vectorii x și y și evaluează funcția între un sistem rectangular prin construirea tablourilor X și Y, care pot fi utilizate la evaluarea unei funcții de două variabile, precum funcția z. Linile lui X sunt copii ale vectorului x, iar coloanele lui Y sunt copii ale vectorului y

[X,Y]=meshgrid(x,y);

Se calculează z ca o matrice prin efectuarea operațiilor asupra lui X și Y.

z=X.^3+(Y/2).^3;

Se face o reprezentare a conturului și o reprezentare 3D de tip mesh, ca două subgrafice folosind funcția **subplot** (vezi **help subplot**)

subplot(1,2,1)

contour(z) % figura 3.3.1a

subplot(1,2,2)

mesh(z) % figura 3.3.1b

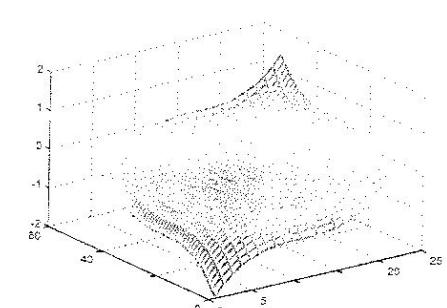
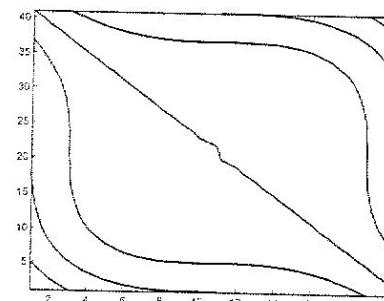


Fig. 3.3.1 a,b

3.4. Exerciții propuse

Exercițiu 3.4.1

Să se reprezinte grafic funcțiile $f(t)=\sin(2\pi 50t)$ cu linie punct de culoare verde și $g(t)=f(t)+0,2$ cu indicator stea de culoare roșie ($t=[0:0.02]$ cu pasul 0.001).

Exercițiu 3.4.2

Să se reprezinte în coordonate semilogaritmice (axa y), funcția $f(x)=10^x$ ($x=[0:10]$ cu pasul 0.1).

Exercițiu 3.4.3

Să se reprezinte în coordonate polare funcția $f(t)=\sin(2t)\cos(2t)$ ($t=[0:2\pi]$ cu pasul 0.01).

Exercițiu 3.4.4

Să se reprezinte funcția discretă sinus:

$$f[n] = \sin\left(\frac{2\pi}{10}n\right), \quad n \in [0, 20]$$

Exercițiu 3.4.5

Să se reprezinte graficul în trepte al funcției $y = \sin(x)$ ($x = [0 : 6]$ cu pasul 0.2).

Exercițiu 3.4.6

Să se reprezinte grafic următoarea funcție, unde $x = [-5 : 5]$ cu pasul 0.1:

$$f(x) = \frac{x|x|}{1+x^2}$$

Exercițiu 3.4.7

Să se reprezinte un grafic 3D unde:

$$x = \cos(2\pi t); \quad y = \sin(2\pi t) \text{ și } t = [-2 : 0.01 : 2].$$

Exercițiu 3.4.8

Să se reprezinte grafic, de tip mesh, funcția:

$$z = x^2 - y^2 \quad (x = y = [-2 : 0.01 : 2]).$$

MODELAREA MATEMATICĂ A SISTEMELOR CONTINUE

OBIECTIVE

- Analiza sistemelor de ordinul doi folosind modele matematice.
- Calculul polilor și zerourilor funcției de transfer.
- Reducerea schemelor bloc.

4.1. Introducere

Analiza și sinteza sistemelor de control se bazează pe modelele matematice ale sistemelor fizice complexe, care se obțin pe baza legilor fizice ale proceselor sunt, în general, ecuații diferențiale neliniare. Pe de altă parte, multe sisteme fizice au proprietatea de liniaritate în jurul anumitor puncte de funcționare, astfel că este posibilă o aproximare liniară a sistemelor fizice. Dezvoltarea în serie Taylor este exemplu pentru acest caz, utilizându-se la liniarizarea proceselor.

Aproximarea liniară a sistemelor fizice este descrisă de o ecuație diferențială cu coeficienți constanți. Transformata Laplace este un mod potrivit de calcul a soluției unei ecuații diferențiale și poate fi folosită pentru a obține descrierea intrare- ieșire a sistemelor liniare, invariante în timp, sub forma funcției de transfer.

Matlab-ul poate fi utilizat pentru studiul sistemelor liniare invariante în timp, descrise prin funcția de transfer sau sub forma intrare-stare- ieșire.

4.2. Sistem mecanic resort-masă-amortizare și sistem electric RLC

Sistemul mecanic resort-masă-amortizare este prezentat în figura 4.2. Deplasarea masei, notată $y(t)$, este descrisă de ecuația diferențială (4.1):

$$My''(t) + By'(t) + ky(t) = f(t) \quad (4.1)$$

Soluția $y(t)$ a ecuației diferențiale (4.1) descrie deplasarea masei în funcție de timp. Funcția de intrare (forță, în acest exemplu) este reprezentată de $f(t)$. Determinarea modelului matematic al acestui sistem mecanic se bazează pe utilizarea unui resort și amortizare ideale, care aproximează elementele reale. Modelul sistemului mecanic resort-masă-amortizare, dat de ecuația (4.1), este o aproximare liniară și invariantă în timp a procesului fizic; el este valabil numai în regimurile unde forța resortului este o funcție liniară de deplasare a masei și un

forța de frecare (amortizarea) este o funcție liniară de viteză. Modelul matematic dat de (4.1) reprezintă un vehicul care absoarbe energie.

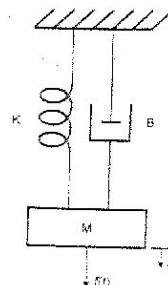


Fig. 4.2.1 Sistem mecanic resort-masă amortizat

Și alte procese fizice sunt descrise de modelele matematice continue (analogice) de tipul (4.1).

Un exemplu de sistem electric tipic este circuitul RLC, redat în figura 4.2.2. În acest caz, modelul matematic este analogic, unde viteza $y(t)$ și tensiunea $u(t)$ sunt variabile analogice. Această noțiune de sistem analogic este importantă în modelarea sistemelor. Analiza sistemelor descrise de modelul (4.1) este importantă nu numai în cazul sistemelor mecanice și electrice, dar și a celor termice și hidraulice.

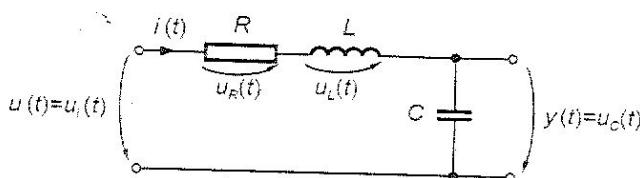


Fig. 4.2.2 Circuitul RLC

Revenind la primul exemplu, răspunsul neforțat ($f(t)=0$) $y(t)$ al sistemului mecanic (figura 4.2.1) este dat de:

$$y(t) = y(0) \frac{e^{-\zeta \omega_n t}}{\sqrt{1-\zeta^2}} \sin(\omega_n \sqrt{1-\zeta^2} t + \varphi), \quad (4.2)$$

unde $\varphi = \arccos(\zeta)$, iar deplasarea inițială este $y(0)$.

Răspunsul tranzitoriu (4.2) este subamortizat dacă $\zeta < 1$, supraamortizat dacă $\zeta > 1$ și amortizat critic dacă $\zeta = 1$.

Se poate utiliza Matlab-ul pentru a vizualiza răspunsul neforțat (tranzitoriu) dat de (4.2) cu condiția inițială $y(0)$. În cazul în care se dorește o simulare a funcționării unui sistem cu reacție, atunci se pot varia intrările și condițiile inițiale ale sistemului, putându-se calcula în Matlab soluțiile numerice și apoi soluția grafică.

4.3. Funcții de transfer

Funcția de transfer reprezintă o descriere intrare-ieșire a sistemelor și se obține aplicând transformata Laplace în condiții inițiale nule ecuației diferențiale (4.1).

În cazul sistemelor liniare continue în timp, funcția de transfer $G(s)$ este dată de un raport de două polinoame de variabilă complexă s , adică $G(s)$ este de forma:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{b_0 s^m + b_1 s^{m-1} + \dots + b_{m-1} s + b_m}{s^n + a_1 s^{n-1} + \dots + a_{n-1} s + a_n} \quad (4.3)$$

unde $m \leq n$. Rădăcinile polinomului de la numărătorul funcției $G(s)$ se numesc zerourile sistemului, iar rădăcinile polinomului de la numitorul funcției $G(s)$ se numesc polii sistemului. În Matlab, pentru crearea unei funcții de transfer, se folosește funcția tf care are următoarea sintaxă:

`sys=tf(num,den)`,

unde

`num, den` - reprezintă numărătorul, respectiv numitorul funcției de transfer, rezultând astfel un sistem continuu; dacă se specifică și perioada de eşantionare atunci va rezulta un sistem discret.

Ecuția obținută prin egalarea cu zero a numitorului lui $G(s)$ se numește ecuația caracteristică a sistemului:

$$s^n + a_1 s^{n-1} + \dots + a_{n-1} s + a_n = 0 \quad (4.4)$$

Răspunsul tranzitoriu al sistemului este determinat de repartiția (localizarea) în planul s a polilor și zerourilor. Se poate utiliza programul Matlab pentru analiza sistemelor descrise de funcții de transfer. Deoarece funcția de transfer este un raport de două polinoame, Matlab-ul poate fi folosit pentru rezolvarea polinoamelor. Polinoamele sunt reprezentate de vectori liniști conținând coeficienții polinomului în ordinea descrescătoare pătrăilor variabilei. În exemplul 4.1 este prezentat modul de calcul al rădăcinilor pentru polinomul:

$$p(s) = s^3 + 3s^2 + 4 \quad (4.5)$$

Exemplul 4.1

```
>> p=[1 3 0 4];
>> r=roots(p)
r =
-3.3553
0.1777 + 1.0773i
0.1777 - 1.0773i
```

Dacă p este un vector linișt ce conține coeficienții lui $p(s)$ în ordinea descrescătoare pătrăilor, atunci $r=roots(p)$ este un vector coloană ce admite rădăcinile polinomului $p(s)$. Învers, dacă r este un vector coloană ce conține rădăcinile unui polinom, atunci $p=poly(r)$ este un vector linișt ce înglobează coeficienții polinomului în ordinea descrescătoare pătrăilor variabilei.

Pentru a înmulți două polinoame se utilizează funcția conv, iar pentru a evalua valoarea polinoamelor pentru o valoare dată a variabilei, se utilizează funcția polyval.

În exemplul 4.2 este exemplificată utilizarea funcțiilor conv și polyval pentru calcularea valorii polinomului $h(s)=(3s^2+2s+1)(s+4)$ în punctul $s = -5$.

Reprezentarea grafică pentru localizarea pol-zero în planul complex se face folosind funcția `pzmap`, astfel: `pzmap(num, den)`. În operatorul pol-zero, zerourile sunt notate cu „o” și polii cu „x”.

Exemplul 4.2

Utilizarea funcțiilor `conv` și `polyval` pentru a multiplica și evalua polinoamele

$$(3s^2 + 2s + 1)(s + 4)$$

```
>> h1=[3 2 1];
>> h2=[1 4];
>> h=conv(h1,h2)
h = 3 14 9 4
>> v=polyval(h,-5)
v = -66
```

4.4. Modelele diagramei bloc

Presupunem că avem modelele matematice în forma funcțiilor de transfer pentru instalația tehnologică (proces) $G(s)$ și pentru regulator $G_R(s)$. Putem avea și conectăm aceste elemente pentru a obține un sistem de control. Ne propunem să funcțiile Matlab pentru transformarea diagramelor bloc descrise în cursul de Teoria Sistemelor.

Procesul controlat este arătat în figura 4.4.1. Un sistem de control în buclă deschisă poate fi obținut prin conectarea în serie a procesului cu regulatorul, așa cum se poate observa în figura 4.4.2.

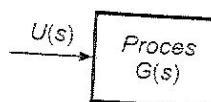


Fig. 4.4.1 Sistem deschis

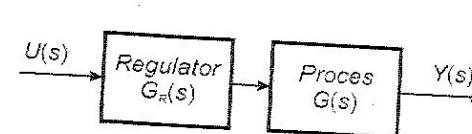


Fig. 4.4.2 Sistem de control deschis

Se poate utiliza Matlab-ul pentru calculul funcției de transfer $\frac{Y(s)}{U(s)}$ așa cum se vede în exemplul 4.3.

4.4.1. Conexiunea serie

Se poate utiliza funcția `series` pentru a lega în cascadă două funcții de transfer $G_1(s)$ și $G_2(s)$, ca în figura 4.4.3.

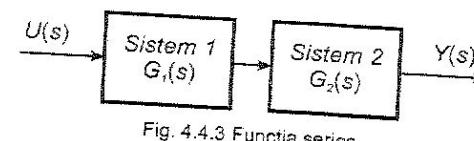


Fig. 4.4.3 Funcția series

$$G(s) = \frac{Y(s)}{U(s)} = \frac{\text{num}}{\text{den}} \quad G_1(s) = \frac{\text{num1}}{\text{den1}} \quad G_2(s) = \frac{\text{num2}}{\text{den2}}$$

`[num, den] = series(num1, den1, num2, den2)`

Exemplul 4.3

Fie procesul (instalația tehnologică) dat de funcția de transfer

$$G(s) = \frac{1}{500s^2}$$

și fie regulatorul, reprezentat de funcția de transfer

$$G_R(s) = \frac{s+1}{s+2}$$

Funcția de transfer $G_R(s)G(s)$ este calculată utilizând funcția `series`.

$$G_R(s)G(s) = \frac{\text{num}}{\text{den}} = \frac{s+1}{500s^3 + 1000s^2}$$

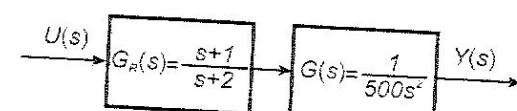


Fig. 4.4.4 Aplicație a funcției series.

`numr=[1 1]; denr=[1 2];`

`num=[0 0 1]; den=[500 0 0];`

`[nums, dens]=series(numr, denr, num, den)`

4.4.2. Conexiunea paralel

Diagrama bloc se obține și pentru legarea în paralel folosind funcția descrisă în figura 4.4.5.

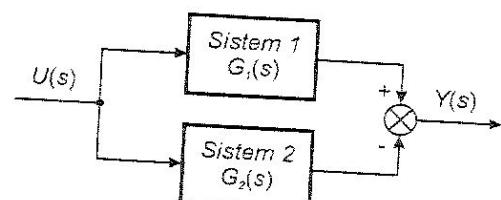


Fig. 4.4.5 Funcția paralel

$$G(s) = \frac{Y(s)}{U(s)} = \frac{\text{num}}{\text{den}} \quad G_1(s) = \frac{\text{num1}}{\text{den1}} \quad G_2(s) = \frac{\text{num2}}{\text{den2}}$$

`[num, den] = parallel(num1, den1, num2, den2)`

4.4.3. Conexiunea cu reacție

Introducerea unui semnal de reacție va genera un sistem în buclă închisă (cu reacție) unitară, așa cum se vede în figura 4.4.6. Semnalul $E(s)$ este semnalul de referință (de intrare), iar $R(s)$ este semnalul de referință (de intrare). În acest sistem de control, reguluatorul este pe calea directă și funcția de transfer a sistemului închis $G_0(s) = \frac{G_R(s)G(s)}{1 \pm G_R(s)G(s)}$. Există două funcții care pot utiliza procesul de reducere

diagramei bloc pentru calculul funcției de transfer a sistemului închis monobucă și multibucă și anume funcțiile **cloop** și **feedback**.

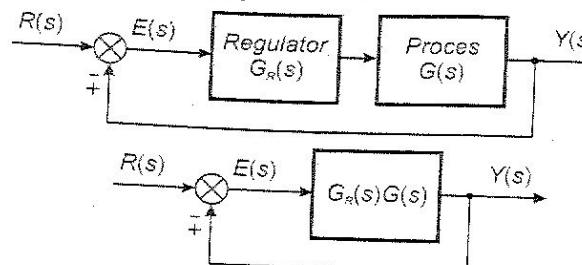


Fig. 4.4.6 Sistem de control cu reacție unitară, funcția cloop

$$G_0(s) = \frac{Y(s)}{R(s)} = \frac{\text{num}}{\text{den}} \quad G_R(s)G(s) = \frac{\text{num1}}{\text{den1}}$$

+1 - reacție pozitivă
-1 - reacție negativă

$[\text{num}, \text{den}] = \text{cloop}(\text{num1}, \text{den1}, \text{sign})$

Funcția **cloop** calculează funcția de transfer a sistemului închis în care apare și configurația sistemului cu reacție unitară. Funcția **feedback** este arătată în figura 4.4.7 în care apare și configurația sistemului care include partea de reacție $G_2(s)$.

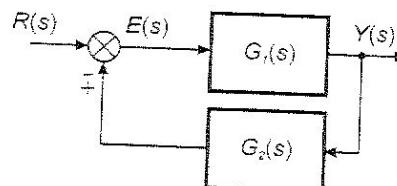


Fig. 4.4.7 Sistem de control cu reacție, funcția feedback

$$G_0(s) = \frac{Y(s)}{R(s)} = \frac{\text{num}}{\text{den}} \quad G_1(s) = \frac{\text{num1}}{\text{den1}}, \quad G_2(s) = \frac{\text{num}}{\text{den2}}$$

+1 - reacție pozitivă
-1 - reacție negativă

$[\text{num}, \text{den}] = \text{feedback}(\text{num1}, \text{den1}, \text{num2}, \text{den2}, \text{sign})$

Pentru ambele funcții, **cloop** și **feedback**, dacă semnul intrării „sign” este omis, atunci reacția negativă este presupusă că există. În exemplul 4.4 este prezentat modul de utilizare al funcției **cloop**, iar în exemplul 4.5, este arătat modul de utilizare al funcției **feedback**.

Exemplul 4.4. Funcția cloop

Fie procesul $G(s)$ și reglatorul $G_R(s)$ din exemplul 4.3 (figura 4.4.8). Se aplică funcția **cloop** utilizând în primul rând funcția **series** pentru calculul lui $G(s)G_R(s)$, urmată de funcția **cloop** pentru bucla închisă.

Funcția de transfer a sistemului închis este:

$$G_0(s) = \frac{G_R(s)G(s)}{1 + G_R(s)G(s)} = \frac{\text{num}}{\text{den}} = \frac{s+1}{500s^3 + 1000s^2 + s + 1}$$

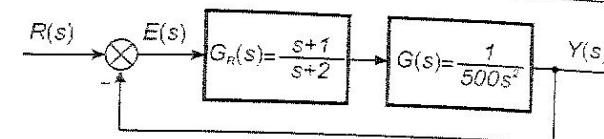


Fig. 4.4.8 Aplicație a funcției cloop

```
numr=[1 1]; denr=[1 2];
num=[0 0 1]; den=[500 0 0];
[nums,dens]=series(numr,denr,num,den);
[num,den]=cloop(nums,dens,-1)
```

Exemplul 4.5. Funcția feedback

Fie încă odată procesul $G(s)$ și reglatorul $G_R(s)$ din exemplul 4.3. Se calculează funcția de transfer a sistemului închis cu reglatorul pe calea de reacție, utilizând funcția **feedback** (figura 4.4.9). Funcția de transfer a sistemului închis este:

$$G_0(s) = \frac{G(s)}{1 + G_R(s)G(s)} = \frac{\text{num}}{\text{den}} = \frac{s+2}{500s^3 + 1000s^2 + s + 1}$$

Sevența de comandă:

```
numr=[1 1]; denr=[1 2];
num=[0 0 1]; den=[500 0 0];
[numf,denf]=feedback(numr,denr,num,den,-1)
```

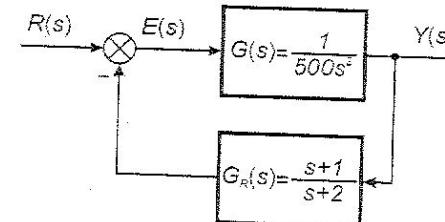


Fig. 4.4.9 Aplicație a funcției feedback

Funcțiile Matlab **series**, **cloop** și **feedback** pot fi utilizate și pentru diagramele multibucă.

4.5. Exerciții propuse

Exercițiul 4.5.1

Să se reprezinte răspunsul neforțat al sistemului mecanic ($t=[0:0.001:7]$, iar relația matematică este prezentată în relația 1.2), pentru:

- Cazul supraamortizat: $y(0) = 0,15m$; $\omega_n = \sqrt{2} \frac{\text{rad}}{\text{s}}$; $\zeta_1 = \frac{3}{2\sqrt{2}}$.
- Cazul subamortizat: $y(0) = 0,15m$; $\omega_n = \sqrt{2} \frac{\text{rad}}{\text{s}}$; $\zeta_2 = \frac{1}{2\sqrt{2}}$.

ANALIZA SI SIMULAREA ÎN TIMP PENTRU SISTEMELE CONTINUE

OBIECTIVE

- Obținerea răspunsului sistemelor de ordinele întâi și doi la intrările treaptă, rampă și impuls.
- Determinarea stabilității sistemelor de reglare.
- Modul de obținere a indicatorilor de calitate în domeniul timpului.
- Calculul erorilor staționare pentru diverse semnale tipice de test aplicate la intrare.

5.1. Introducere

Pentru a defini performanțele unui sistem închis este necesară cunoașterea răspunsului în timp al sistemului la diferite mărimi de intrare standard. O mărime de intrare standard, des folosită, este funcția treaptă. Dacă răspunsul unui sistem la mărimea treaptă este cunoscut, atunci este posibil calculul matematic al răspunsului sistemului la alte mărimi de intrare. În practică, semnalul de intrare al unui sistem de control nu este cunoscut în prealabil.

5.2. Răspunsul sistemului de ordinul întâi – T1

Schema bloc a sistemului de ordinul întâi și funcția de transfer echivalentă sunt prezentate în figura 5.2.1.

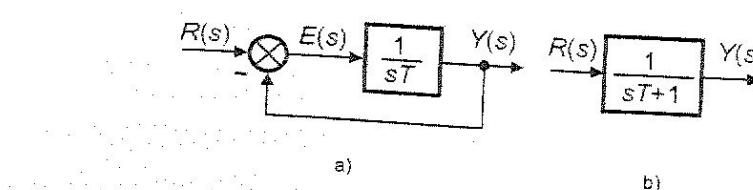


Fig. 5.2.1 Sistem de ordinul întâi
Fizic, un astfel de sistem poate să fie un circuit RC. Funcția de transfer standard a unui element de ordinul întâi este conform relației:

$$\frac{Y(s)}{R(s)} = \frac{1}{sT+1} = \frac{1/T}{s+1/T} = \frac{\sigma}{s+\sigma}, \quad (5.1)$$

Exercițiu 4.5.2

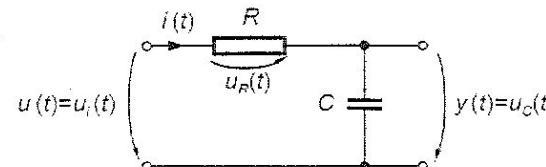
Considerând funcțiile de transfer:

$$G(s) = \frac{6s^2 + 1}{s^3 + 3s^2 + 3s + 1} \quad \text{și} \quad H(s) = \frac{(s+1)(s+2)}{(s+2i)(s-2i)(s+3)},$$

să se calculeze și reprezinte polii și zerourile funcțiilor de transfer $G(s)$ și $H(s)$, precum și rezultatul împărțirii lui $G(s)$ la $H(s)$ (numărul de poli este mai mare sau cel mult egal cu numărul de zerouri).

Exercițiu 4.5.3

Să se deseneze diagrama bloc și să se determine funcția de transfer pentru circuitul RC (rezistor și condensator) cu schema prezentată în figură:



unde: $u_i(t)$ reprezintă mărimea de intrare a sistemului, iar $u_c(t)$ mărimea de ieșire. Condițiile inițiale sunt nule.

Exercițiu 4.5.4

Să se realizeze programul pentru reducerea multibuclă. Rezultatul obținut cu programul scris în Matlab este:

$$G_0(s) = \frac{\text{num}}{\text{den}} = \frac{s^5 + 4s^4 + 6s^3 + 6s^2 + 5s + 2}{12s^6 + 205s^5 + 1066s^4 + 2517s^3 + 3128s^2 + 2196s + 712}$$

Sistemul multibuclă este dat în figura de mai jos și se dorește calcularea funcției de transfer echivalentă: $G(s) = \frac{Y(s)}{R(s)}$. În cadrul sistemului dat, există următoarele funcții de transfer:

$$G_1(s) = \frac{1}{s+10}; \quad G_2(s) = \frac{1}{s+1};$$

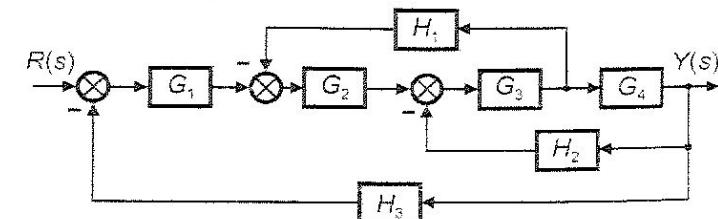
$$G_3(s) = \frac{s^2 + 1}{s^2 + 4s + 4}; \quad G_4(s) = \frac{s+1}{s+6};$$

$$H_1(s) = 2; \quad H_2(s) = \frac{s+1}{s+2} \quad \text{și} \quad H_3(s) = 1.$$

Pentru rezolvarea acestui exemplu trebuie parcursse următoarele etape:

- Etapa 1: Introducerea funcțiilor de transfer ale blocurilor componente în Matlab.
- Etapa 2: Se deplasează blocul $G_4(s)$ în interiorul buclei cu $H_1(s)$ pe reacție.
- Etapa 3: Se elimină bucla $G_3(s)G_4(s)H_2(s)$.

- Etapa 4: Se elimină bucla cu $H_1(s)/G_4(s)$ pe reacție.
- Etapa 5: Se elimină bucla rămasă și se calculează $G(s)$.



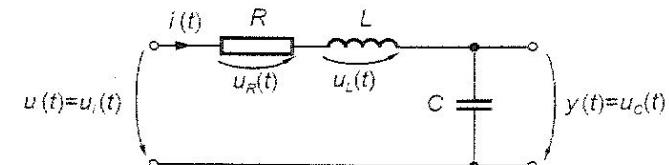
Sistem cu mai multe bucle de reacție.

Dacă se calculează polii și zerourile lui $G(s)$, se constată că se poate da facți comun $(s+1)$ la numărător și la numitor, deci funcția de transfer se simplifică. Această simplificare se poate face cu ajutorul funcției minreal:

`[num1,den1] = minreal(num, den);`

Exercițiu 4.5.5

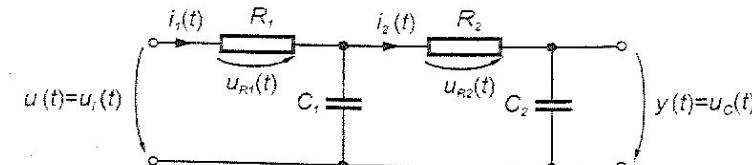
Să se deseneze diagrama bloc și să se determine funcția de transfer pentru circuitul RLC cu schema prezentată în figură:



unde $u_i(t)$ reprezintă mărimea de intrare a sistemului, iar $u_c(t)$ mărimea de ieșire. Condițiile inițiale sunt nule.

Exercițiu 4.5.6

Să se deseneze diagrama bloc și să se determine funcția de transfer pentru circuitul cu schema prezentată în figură:



unde $u_i(t)$ reprezintă mărimea de intrare a sistemului, iar $u_c(t)$ mărimea de ieșire. Condițiile inițiale sunt nule.

ANALIZA ȘI SIMULAREA ÎN TIMP PENTRU SISTEMELE CONTINUE

OBIECTIVE

- Obținerea răspunsului sistemelor de ordinele întâi și doi la intrările treaptă, rampă și impuls.
- Determinarea stabilității sistemelor de reglare.
- Modul de obținere a indicatorilor de calitate în domeniul timpului.
- Calculul erorilor staționare pentru diverse semnale tipice de test aplicate la intrare.

5.1. Introducere

Pentru a defini performantele unui sistem închis este necesară cunoașterea răspunsului în timp al sistemului la diferite mărimi de intrare standard. O mărime de intrare standard, des folosită, este funcția treaptă. Dacă răspunsul unui sistem la mărimea treaptă este cunoscut, atunci este posibil caclculul matematic al răspunsului sistemului la alte mărimi de intrare. În practică, semnalul de intrare al unui sistem de control nu este cunoscut în prealabil.

5.2. Răspunsul sistemului de ordinul întâi – T1

Schema bloc a sistemului de ordinul întâi și funcția de transfer echivalentă sunt prezentate în figura 5.2.1.

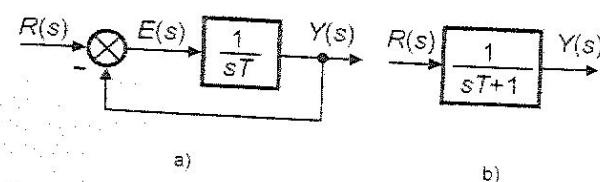


Fig. 5.2.1 Sistem de ordinul întâi

Fizic, un astfel de sistem poate să fie un circuit RC. Funcția de transfer standard a unui element de ordinul întâi este conform relației:

$$\frac{Y(s)}{R(s)} = \frac{1}{sT+1} = \frac{1/T}{s + 1/T} = \frac{\sigma}{s + \sigma}, \quad (5.1)$$

unde $\sigma = 1/T$.

În continuare, se va analiza răspunsul acestui sistem la intrarea treaptă. Condițiile inițiale se presupun nule.

Răspunsul la treaptă unitară se mai numește răspuns indicial. Având în vedere că transformata Laplace a unei trepte unitare este $R(s) = 1/s$, rezultă că transformata Laplace a mărimii de ieșire va fi în acest caz:

$$Y(s) = \frac{1}{(sT+1)s} = \frac{1/T}{s(s+1/T)}, \quad (5.2)$$

cu dezvoltarea în fracții simple:

$$Y(s) = \frac{1}{s} - \frac{1}{s+1/T} = Y_p(s) + Y_t(s). \quad (5.3)$$

Aplicând transformata Laplace inversă relației (5.3), rezultă răspunsul la treaptă unitară al sistemului de ordinul unu, conform expresiei:

$$y(t) = 1 - e^{-\frac{t}{T}} = y_p(t) + y_t(t), \quad t \geq 0, \quad (5.4)$$

Răspunsul la rampă unitară al sistemului de ordinul unu se obține conform expresiei:

$$y(t) = t - T + Te^{-t/T}, \quad t \geq 0, \quad (5.5)$$

cu forma de variație prezentată în figura 5.2.2:

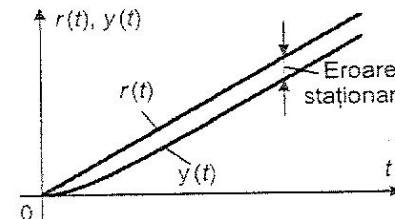


Fig. 5.2.2 Răspunsul la rampă unitară

Eroarea (abaterea) de reglare a sistemului de ordinul întâi pentru mărime de intrare rampă unitară este:

$$e(t) = r(t) - y(t) = t - (t - T + Te^{-t/T}) = T(1 - e^{-t/T}), \quad (5.6)$$

de unde se constată că eroarea staționară obținută când $t \rightarrow \infty$ are valoarea $e_{st} = e(\infty) = T$.

Răspunsul la impuls unitar al sistemului de ordinul unu descris de relația:

$$y(t) = g(t) = \frac{1}{T} \cdot e^{-\frac{t}{T}}, \quad t \geq 0, \quad (5.7)$$

are forma de variație prezentată în figura 5.2.3.

Tangenta la curbă, în momentul initial ($t=0$), intersectează axa absciselui la valoarea $t=T$.

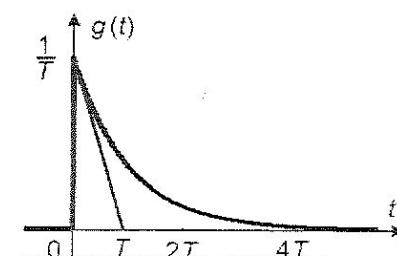


Fig. 5.2.3 Răspunsul la impuls unitar

5.3. Răspunsul sistemului de ordinul doi la intrarea treaptă

Elementul de ordinul doi se poate obține considerând sistemul cu reacție negativă unitară, cu schema bloc din figura 5.3.1a. Calculând funcția de transmisie închis rezultă sistemul din figura 5.3.1b.

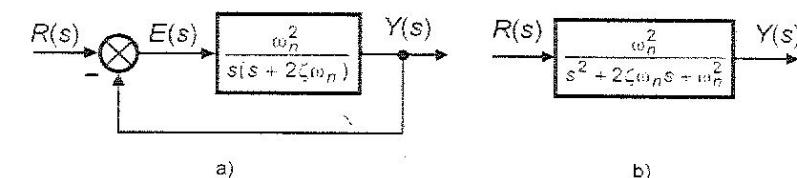


Fig. 5.3.1 Sistem de ordinul doi

Funcția de transfer standard a unui sistem de ordinul doi este:

$$G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2},$$

unde ζ este factorul de amortizare, iar ω_n este pulsăția naturală. Frecvența naturală este frecvența de oscilație dacă amortizările lipsesc, iar factorul de amortizare caracterizează natura răspunsului tranzitoriu al sistemului. Răspunsul sistemului ordinul doi la o mărime treaptă pentru un factor de amortizare $0 < \zeta < 1$ (subamortizat), în cazul condițiilor inițiale zero este dat de:

$$y(t) = 1 - \frac{e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}} \sin(\omega_d t + \phi), \quad \phi = \arctg \frac{\sqrt{1-\zeta^2}}{\zeta}, \quad t \geq 0$$

sau

$$y(t) = L^{-1}\{Y(s)\} = 1 - e^{-\zeta\omega_n t} \left[\cos \omega_d t + \frac{\zeta}{\sqrt{1-\zeta^2}} \sin \omega_d t \right], \quad t \geq 0,$$

unde $\omega_d = \omega_n \sqrt{1-\zeta^2}$.

În cazul particular când $\zeta = 0$, avem $\omega_d = \omega_n$, iar din ecuația (5.9) se obține:

Îndrumar de laborator

Teoria Sistemelor

răspunsul la treaptă al sistemului de ordinul doi, descris de ecuația:

$$y(t) = 1 - \cos \omega_n t, \quad t \geq 0. \quad (5.10)$$

Se observă că dacă $\zeta = 0$, răspunsul devine neamortizat cu oscilații întreținute, care continuă pentru $t \rightarrow \infty$.

Răspunsul la treaptă al sistemului de ordinul doi critic amortizat ($\zeta = 1$) este:

$$y(t) = 1 - \omega_n t e^{-\omega_n t} - e^{-\omega_n t} = 1 - e^{-\omega_n t}(1 + \omega_n t), \quad t \geq 0. \quad (5.11)$$

Acest rezultat se poate obține înlocuind ζ cu 1 în relația (5.9) și folosind limitele:

$$\lim_{\zeta \rightarrow 1} \frac{\zeta}{\sqrt{1-\zeta^2}} \sin \omega_n \sqrt{1-\zeta^2} t = \zeta \omega_n t \lim_{\zeta \rightarrow 1} \frac{\sin \omega_n \sqrt{1-\zeta^2} t}{\omega_n \sqrt{1-\zeta^2} t} = \omega_n t,$$

respectiv faptul că: $\lim_{\zeta \rightarrow 1} \cos \omega_n \sqrt{1-\zeta^2} t = \cos 0 = 1$.

Răspunsul la treaptă al sistemului de ordinul doi supraamortizat (cu $\zeta > 1$) este:

$$y(t) = 1 + \frac{\omega_n}{2\sqrt{\zeta-1}} \left[\frac{e^{-(\zeta+\sqrt{\zeta^2-1})\omega_n t}}{\omega_n(\zeta+\sqrt{\zeta^2-1})} - \frac{e^{-(\zeta-\sqrt{\zeta^2-1})\omega_n t}}{\omega_n(\zeta-\sqrt{\zeta^2-1})} \right], \quad t \geq 0. \quad (5.12)$$

Cei doi poli ai sistemului sunt reali, negativi și distincți, iar elementul de ordinul doi poate fi considerat ca fiind format din două elemente de ordinul unu conectate în serie.

În figura 5.3.2 se prezintă o familie de curbe răspuns la treaptă al elementului de ordinul doi pentru diferite valori ale lui ζ , unde abscisa este variabila adimensională $\omega_n t$. Se poate observa că sistemele de ordinul doi cu aceeași valoare pentru ζ , dar diferită pentru ω_n , vor prezenta aceeași valoare maximă pentru răspuns (același suprareglaj) și același model de oscilație. Despre aceste sisteme se constată că stabilitatea relativă este identică.

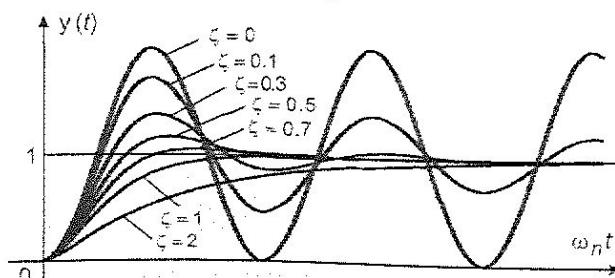


Fig. 5.3.2 Familie de curbe - răspunsul la treaptă unitară

Răspunsul la impuls al sistemului de ordinul doi în cele trei cazuri:

- $0 < \zeta < 1$ - cazul subamortizat:

Îndrumar de laborator

Teoria Sistemelor

$$y(t) = g(t) = \frac{\omega_n}{\sqrt{1-\zeta^2}} e^{-\zeta \omega_n t} \sin \omega_n \sqrt{1-\zeta^2} t, \quad t \geq 0. \quad (5.13)$$

- $\zeta = 0$ - cazul neamortizat:

$$y(t) = g(t) = \omega_n \sin \omega_n t, \quad t \geq 0. \quad (5.14)$$

- $\zeta = 1$ - cazul critic amortizat:

$$y(t) = \omega_n^2 t e^{-\omega_n t}, \quad t \geq 0. \quad (5.15)$$

- $\zeta > 1$ - cazul supraamortizat:

$$y(t) = \frac{\omega_n}{2\sqrt{\zeta^2-1}} [e^{-(\zeta-\sqrt{\zeta^2-1})\omega_n t} - e^{-(\zeta+\sqrt{\zeta^2-1})\omega_n t}], \quad t \geq 0. \quad (5.16)$$

În figura 5.3.3 se prezintă familia de curbe răspuns la impuls pentru elementul de ordinul doi cu diferite valori ale lui ζ .

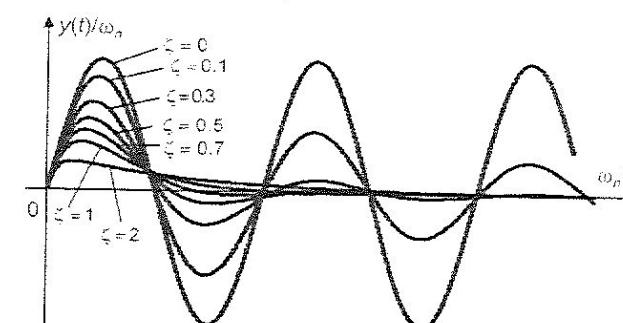


Fig. 5.3.3 Familie de curbe - răspunsul la impuls unitar

Curbele sunt scalate în amplitudine și timp, fiind desenate funcțiile $y(t)/\omega_n$ în raport cu variabila adimensională $\omega_n t$, astfel încât aceste funcții nu vor depinde decât de ζ . Din figură se constată că pentru $\zeta \geq 1$, răspunsul la impuls este tot timpul pozitiv ($y(t) \geq 0$). În cazul subamortizat $y(t)$ oscilează în jurul valorii zero.

5.3.1. Performanțele sistemului de ordinul doi

Criteriile de performanță care se utilizează pentru caracterizarea regimului tranzitoriu la o intrare treaptă unitară sunt:

- suprareglajul (abaterea dinamică maximă)

$$M_v \% = \frac{y_{\max} - y_{st}}{y_{st}} \cdot 100 \%, \quad (5.17)$$

unde y_{\max} - este valoarea maximă a răspunsului y , iar y_{st} - valoarea de regim stationar a răspunsului ($y_{st} = 1$). Valoarea maximă se obține calculând timpul t_v (timpul de vârf = criteriu de performanță) la care se atinge valoarea maximă:

$$\frac{dy(t)}{dt} = 0, \text{ adică } t_v = \frac{\pi}{\omega_d} = \frac{\pi}{\omega_n \sqrt{1-\zeta^2}}. \quad (5.18)$$

Așadar, valoarea maximă se obține introducând (5.18) în (5.9):

$$\begin{aligned} M_V &= y(t_v) - 1 = 1 - e^{-\zeta \omega_n (\pi / \omega_d)} \left(\cos \pi + \frac{\zeta}{\sqrt{1-\zeta^2}} \sin \pi \right) - 1, \\ M_V &= \exp(-\pi \zeta / \omega_d) = \exp(-\pi \zeta / \sqrt{1-\zeta^2}). \end{aligned} \quad (5.19)$$

Suprareglajul în procente:

$$M_V \% = \exp(-\pi \zeta / \sqrt{1-\zeta^2}) 100. \quad (5.20)$$

- timpul de stabilire (durata regimului tranzitoriu) este timpul necesar ca răspunsul sistemului să intre în zona $\pm 0,05 y_{st}$ sau $\pm 0,2 y_{st}$.

O relație acoperitoare de calcul al timpului tranzitoriu se obține astfel: $t_s \approx 5/\sigma = 4/\zeta \omega_n$ pentru bandă de 2%;

$$t_s \approx 4/\sigma = 3/\zeta \omega_n \text{ pentru bandă de 5\%.} \quad (5.21)$$

- timpul de creștere t_c este timpul necesar evoluției răspunsului sistemului de la $(0,1+0,9)y_{st}$.

$$t_c = \frac{\pi - \phi}{\omega_d} = \frac{\pi - \phi}{\omega_n \sqrt{1-\zeta^2}}, \quad \phi = \operatorname{arctg}(\sqrt{1-\zeta^2}/\zeta). \quad (5.22)$$

5.4. Efectele introducerii unor poli și zerouri suplimentare

Introducerea unui zero în funcția de transfer a sistemului de ordinul doi, în cazul în care este mai îndepărtat de origine comparativ cu polii dominantă (polii cei mai apropiati de originea planului s), face ca sistemul să aibă un suprareglaj mai mare și o durată a regimului tranzitoriu mai mică (viteza de răspuns mai mare).

Introducerea unui pol suplimentar are influență directă asupra regimului tranzitoriu, prin componenta tranzitorie introdusă de acest pol. Cu cât polul este situat la o distanță mai mare față de origine în raport cu polii dominantă, cu atât această componentă se poate neglijă în timp. Așadar, urmând aceste condiții, sistemul poate fi aproximat cu un sistem de ordinul doi.

Funcțiile $C=\text{impulse}(\text{num}, \text{den}, t)$ sau $C=\text{impulse}(\text{num}, \text{den})$ și $C=\text{step}(\text{num}, \text{den}, t)$ sau $C=\text{step}(\text{num}, \text{den})$ și $C=\text{lsim}(\text{num}, \text{den}, u, t)$ din Matlab „Control Toolbox” pot fi utilizate pentru simularea regimului tranzitoriu al sistemului.

5.5. Stabilitatea sistemelor

Stabilitatea unui sistem reprezintă proprietatea acestuia de a reveni în regim staționar atunci când a fost scos dintr-un asemenea regim de către o mărime de intrare sau o perturbație. Conform criteriului general de stabilitate, un sistem este stabil dacă toți polii sistemului închis se găsesc în semiplanul stâng al planului

complex s. Deci, o condiție necesară și suficientă pentru ca un sistem să fie stabil este ca toți polii funcției de transfer a sistemului să aibă parte reală negativă.

Stabilitatea unui sistem liniar invariant în timp poate fi realizată utilizând funcția **impulse** pentru a obține răspunsul la impuls al sistemului. Sistemul este stabil dacă răspunsul la impuls al sistemului tinde la zero atunci când timpul tinde la infinit. Un alt mod de a determina stabilitatea sistemului este prin simulare. Funcția **lsim** este utilizată pentru a observa ieșirea pentru intrări tipice. Pentru sisteme nelineare, aceasta se aplică în cazuri particulare. O altă alternativă este funcția **root** controlului clasic există tehnici sigure pentru analiza stabilității sistemului. Una dintre aceste tehnici este criteriul Routh (pentru aplicare se va folosi funcția **routh**). Stabilitatea unui sistem liniar este o proprietate intrinsecă a sistemului care depinde de mărimea de intrare a sistemului.

5.5.1. Criteriul de stabilitate Routh

Criteriul Routh definește o metodă pentru determinarea stabilității ce poate fi aplicată la o ecuație caracteristică de ordinul n de forma:

$$a_0 s^n + a_1 s^{n-1} + \dots + a_{n-1} s + a_n = 0. \quad (5.23)$$

Dacă criteriul se aplică unui sistem de reglare, se pot obține informații despre stabilitate direct din coeficienții ecuației caracteristice.

Criteriul este aplicat folosind tabela Routh definită astfel:

s^n	a_0	a_2	a_4	\dots
s^{n-1}	a_1	a_3	a_5	\dots
s^{n-2}	b_1	b_2	b_3	\dots
s^{n-3}	c_1	c_2	c_3	\dots
\vdots	\vdots	\vdots	\vdots	\vdots

(5.24)

unde a_n, a_{n-1}, \dots, a_0 sunt coeficienții ecuației caracteristice, iar ceilalți coeficienți se determină în modul următor:

$$b_1 = \frac{-|a_0 \quad a_2|}{a_1} = \frac{a_1 a_2 - a_0 a_3}{a_1}, \quad b_2 = \frac{-|a_0 \quad a_4|}{a_1} = \frac{a_1 a_4 - a_0 a_5}{a_1}, \quad (5.25)$$

$$c_1 = \frac{-|a_1 \quad a_3|}{b_1} = \frac{b_1 a_3 - a_1 b_2}{b_1}, \quad c_2 = \frac{-|a_1 \quad a_5|}{b_1} = \frac{b_1 a_5 - a_1 b_3}{b_1},$$

Calculele pentru fiecare linie sunt continuătă până în momentul apariției unui rezultat egal cu zero. Condiția necesară și suficientă este ca toate rădăcinile ecuației (5.23) să se găsească în semiplanul stâng al planului complex s , iar elementele din prima coloană a tabelei Routh să aibă același semn. În cazul schimbării semnelor elementelor din prima coloană, numărul acestor schimbări indică numărul de rădăci cu parte reală pozitivă.

5.5.2. Cazuri speciale

Cazul I. Dacă primul element dintr-o linie este zero, el se înlocuiește cu un număr pozitiv foarte mic ε și se aplică criteriul aşa cum a fost descris în procedura standard.

Cazul II. Dacă toate elementele dintr-o linie sunt zero, sistemul are poli pe axa imaginară, perechea de rădăcini complex conjugate fiind simetrică față de originea planului complex s. În acest caz, se formează un polinom auxiliar pornind de la coeficienții liniei anterioare. Ulterior, toate zerourile din linie se vor înlocui cu coeficienții estimati prin diferențierea polinomului auxiliar.

5.6. Eroarea staționară și tipul sistemului

În regim stationar sistemul trebuie să îndeplinească performanța eroare stationară e_{sf} . Pentru mărimea de intrare treaptă, eroarea staționară trebuie să fie zero (în cazul în care tipul sistemului este 1 sau 2, acesta fiind dat de numărul de poli din origine pe care-i are sistemul), iar pentru mărimea de intrare rampă, eroarea staționară trebuie să fie mai mică decât o valoare impusă.

Se consideră sistemul:

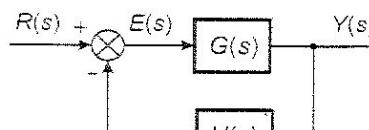


Fig. 5.6.1 Sistem cu reacție

Funcția de transfer este:

$$\frac{Y(s)}{R(s)} = \frac{G(s)}{1 + G(s)H(s)}. \quad (5.26)$$

Transformata Laplace a erorii este:

$$E(s) = R(s) - H(s)Y(s) = \frac{1}{1 + G(s)H(s)}R(s). \quad (5.27)$$

Utilizând teorema valorii finale a transformatei Laplace va rezulta:

$$e_{sf} = \lim_{s \rightarrow 0} \frac{sR(s)}{1 + G(s)H(s)}. \quad (5.28)$$

Dacă intrarea este treaptă unitară, atunci:

$$e_{sf}^1 = \frac{sR(s)}{1 + \lim_{s \rightarrow 0} [G(s)H(s)]} = \frac{1}{1 + k_p}. \quad (5.29)$$

Dacă intrarea este rampă unitară, se va obține:

$$e_{sf}^2 = \frac{1}{\lim_{s \rightarrow 0} [sG(s)H(s)]} = \frac{1}{k_v}. \quad (5.30)$$

Dacă intrarea este parabolică, va rezulta:

$$e_{sf}^3 = \frac{1}{\lim_{s \rightarrow 0} [s^2 G(s)H(s)]} = \frac{1}{k_a}. \quad (5.31)$$

Functiile care calculează eroarea staționară la intrările treaptă unitară, rampă unitară și parabolă unitară sunt:

`errorzp(z,p,k)`,

`errortf(num,den)`.

Funcția `errorzp(z,p,k)` calculează eroarea staționară atunci când sistemul este reprezentat prin poli, zerouri și factor de amplificare (z este un vector care conține zerourile funcției de transfer, p este un vector care conține polii funcției de transfer și k este factorul de amplificare). Dacă gradul numărătorului este mai mic decât gradul numitorului, atunci există n-m zerouri la infinit, iar vectorul z trebuie să conțină (n-m) inf zerouri.

Funcția `errortf(num,den)` calculează eroarea staționară atunci când sistemul este reprezentat ca un raport de două polinoame. În tabelul 5.6.1 se pot urmări relațiile dintre tipul sistemului și mărimele de intrări:

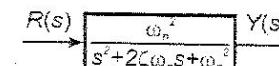
Tab 5.6.1 Erori staționare în funcție de tipul sistemului

Tipul sistemului	Intrarea R(s)		
	1/s	1/s ²	1/s ³
0	1/(1+k _p)	∞	∞
1	0	1/k _v	∞
2	0	0	1/k _a

5.7. Exerciții propuse

Exercițiu 5.7.1

Fie sistemul din figură:



care are următorii parametri: t = [0:2] cu pasul 0.02, ω_n = 5 rad/s, ζ = 0.6. Să se reprezinte răspunsul la intrarea treaptă al sistemului de ordinul 2 și să se determine valoarea suprareglajului.

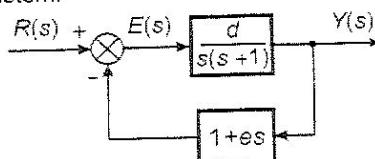
Exercițiu 5.7.2

Să se reprezinte răspunsul la intrarea treaptă pentru sistemul cu funcția de transfer în circuit închis (t are aceleași valori ca în exercițiul anterior) și să se determine valoarea suprareglajului:

$$G_0(s) = \frac{25(1+0.4s)}{(1+0.16s)(s^2 + 6s + 25)}$$

Exercițiu 5.7.3

Se dă următorul sistem:



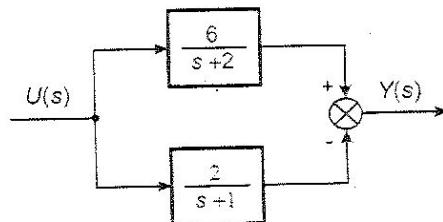
Să se reprezinte răspunsul la intrarea treaptă pentru sistemul anterior. Să se determine valorile lui d și e pentru răspunsul treaptă ($t = [0:4]$, cu pasul 0.02), astfel ca $M_v = 40\%$ și $t_v = 0.8$ s și valoarea regimului tranzitoriu.

Exercițiu 5.7.4

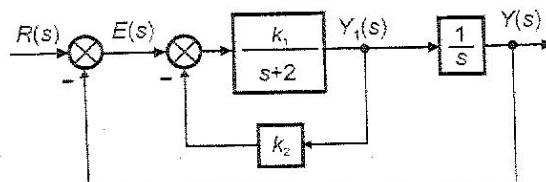
Să se reprezinte grafic suprareglajul sistemului de ordinul doi, știind că $\zeta = [0.001:1]$ cu pasul 0.001.

Exercițiu 5.7.5

Să se reprezinte răspunsul la intrarea treaptă pentru sistemul din figură ($t = [0:10]$, cu pasul 0.1):

**Exercițiu 5.7.6**

Să se reprezinte răspunsul sistemului din figură la intrările treaptă unitară, impuls unitar și rampă unitară, știind că $\zeta = 0.7$ și $\omega_n = 4 \text{ rad/s}$ ($t = [0:0.1:2]$):

**Exercițiu 5.7.7**

Să se aplică criteriul Routh pentru determinarea stabilității sistemului în cazul următoarelor ecuații caracteristice:

$$s^4 + 4s^3 + 7s^2 + 22s + 24 = 0$$

$$s^4 + 4s^3 + 20s + 24 = 0$$

$$s^4 + 10s^3 + 35s^2 + 50s + 24 = 0$$

Exercițiu 5.7.8

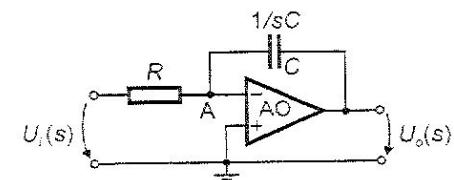
Să se determine eroarea staționară la intrările treaptă, rampă și parabolică pentru următoarele sisteme cu funcția de transfer în circuit deschis:

$$G(s) = \frac{10(s+4)}{s(s+1)(s+2)(s+5)}$$

$$G(s) = \frac{10}{s^2 + 14s + 50}$$

Exercițiu 5.7.9

Să se determine eroarea staționară la intrările treaptă, rampă și parabolică pentru sistemul prezentat în figura de mai jos ($R = 100\text{k}\Omega$, $C = 100\mu\text{F}$):



ANALIZA ȘI SIMULAREA ÎN FRECVENTĂ PENTRU SISTEMELE CONTINUE

OBIECTIVE

- Răspunsul în frecvență la o mărime de intrare sinusoidală.
- Reprezentarea diagramelor Bode și obținerea indicatorilor de calitate.
- Trasarea caracteristicii polare (Nyquist) de frecvență și determinarea stabilității.

6.1. Răspunsul în frecvență

Răspunsul în frecvență este răspunsul sistemului (în circuit deschis) în regim staționar obținut atunci când la intrarea sistemului se aplică o mărime sinusoidală: $r(t) = A \cos \omega t$ (6.1). Așadar, ieșirea sistemului din funcția de transfer $G(s)$ va fi:

$$Y(s) = \frac{sAG(s)}{s^2 + \omega^2} \quad (6.2)$$

Descompunând în sumă de fracții simple rezultă:

$$Y(s) = \frac{k_1}{s - j\omega} + \frac{k_1}{s + j\omega} + \sum \text{termenii dati de polii lui } G(s) \quad (6.3)$$

Polii lui $G(s)$ sunt frecvențele naturale ale sistemului și determină forma componentelor tranzitorii ale răspunsului sistemului. Pentru sistemele liniare, termenii dați de polii lui $G(s)$ nu contribuie la răspunsul staționar $y(t)$. Pe de altă parte, răspunsul staționar este dat de transformata Laplace inversă a primilor doi termeni din (6.3), adică:

$$y(t) = A|G(j\omega)| \cos(\omega t + \theta) \quad (6.4)$$

Din această relație rezultă că ieșirea sistemului are aceeași frecvență ca și intrarea și poate fi obținută prin multiplicarea amplitudinii intrării cu $|G(j\omega)|$, fiind defazată față de intrare cu argumentele lui $G(j\omega)$. Amplitudinea lui $G(j\omega)$ și argumentul lui $G(j\omega)$ pentru toate valorile lui ω , constituie răspunsul în frecvență al sistemului. Corelația dintre răspunsul în frecvență și răspunsul tranzitoriu ale sistemului este indirectă, exceptând cazul sistemului de ordinul doi. În practică, un anumit răspuns în frecvență utilizând diferite criterii de sinteză, va determina răspunsul tranzitoriu dorit.

Răspunsul în frecvență al sistemului închis de ordinul doi
BRASOV

$$G_0(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad (6.5)$$

este dat de:

$$G_0(j\omega) = \frac{1}{\sqrt{\left(1 - \frac{\omega^2}{\omega_n^2}\right)^2 + \left(2\zeta\frac{\omega}{\omega_n}\right)^2}} e^{j\phi(\omega)} \quad (6.6)$$

Deși răspunsul sistemului în circuit închis depinde și de unghiul de fază, acesta se poate neglijă. Astfel, răspunsul în frecvență al sistemului este dat de funcția:

$$M(\omega) = |G_0(j\omega)| = \frac{|Y(j\omega)|}{|R(j\omega)|}, \quad (6.7)$$

sau în decibeli: $M^{dB}(\omega) = 20 \lg |G_0(j\omega)|$ (figura 6.1.1).

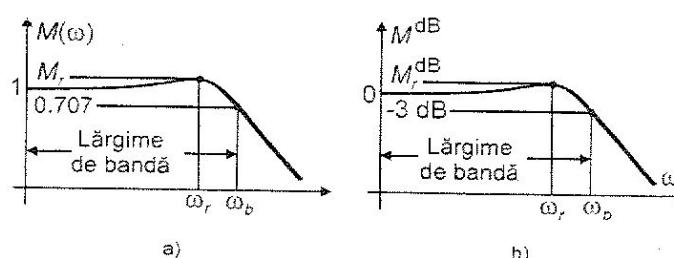


Fig. 6.1.1 Răspunsul în frecvență al sistemului de ordinul doi

Caracteristicile principale ale răspunsului în frecvență al sistemului în circuit închis sunt, de obicei, evaluate cu următoarele indicatori de calitate:

- marginea de amplitudine, m_a ;
- marginea de fază, γ ;
- vârful de rezonanță, M_r ;
- lărgimea (lătimea) de bandă, $0 - \omega_b$.

În legătură cu acești indicatori se poate sublinia faptul că, deși marginile de amplitudine și de fază caracterizează răspunsul în frecvență al sistemului în circuit închis, ei se calculează utilizând funcția de frecvență a sistemului în circuit deschis, $G(j\omega)$.

Frecvența la care se obține valoarea maximă a lui $|G(j\omega)|$ este frecvența de rezonanță. În cazul nostru, pentru $\zeta < 0.707$, frecvența de rezonanță este dată de $\omega_r = \omega_n \sqrt{1 - 2\zeta^2}$ (6.10). Valoarea maximă a amplitudinii răspunsului la o frecvență (vârful de rezonanță) este:

$$M_r = \frac{1}{2\zeta\sqrt{1-\zeta^2}} \quad (6.8)$$

Marginea de amplitudine este definită prin următoarea relație:

$$m_a = \frac{1}{|G(j\omega_{-π})|} \quad (6.9)$$

sau, în decibeli, $m_a^{dB} = 20 \lg m_a = -20 \lg |G(j\omega_{-π})|$ și marginea de fază este definită conform relației:

$$\gamma = 180^\circ + \Phi = 180^\circ + \angle G(j\omega_t) \quad (6.10)$$

unde $\omega_{-π}$ = pulsări de fază, iar ω_t = pulsări de tăiere.

Răspunsul în frecvență cu ajutorul programului Matlab este obținut cu funcția:

`g=freqs(num, den, ω)`,

unde:

`num` = numărătorul funcției de transfer;

`den` = numitorul funcției de transfer;

`ω` = pulsări.

Marginile de amplitudine și de fază se obțin în Matlab cu funcția `margin(num, den)`. Pe de altă parte, în unele versiuni de Matlab, există și funcția `freqspec(ω, mag)`, care determină valorile ω_b , M_r , bazate pe valorile lui ω și mag = amplitudinea.

6.2. Diagrame Bode

Diagramele Bode sunt caracteristici semilogaritmice amplitudine - pulsări și fază - pulsări. Ele reprezintă graficele funcțiilor $A^{dB}(\omega)$ și $\phi(\omega)$, în care abscisa este gradată în scară logaritmică, iar ordonata este gradată în decibeli (dB) pentru $A^{dB}(\omega)$ și respectiv în radiani (grade) pentru $\phi(\omega)$.

6.2.1. Algoritmul de trasare a diagramelor Bode

Se parcurg următoarele etape:

1) Se scrie funcția de transfer a sistemului în circuit deschis în forma standard Bode:

$$G(s)H(s) = \frac{K \prod_i (s\tau_i + 1) \prod_j (s^2\tau_j^2 + 2\zeta_j\tau_j s + 1)}{s^a \prod_k (sT_k + 1) \prod_l (s^2T_k^2 + 2\zeta_l T_k s + 1)}, \quad (6.11)$$

respectiv

$$G(j\omega)H(j\omega) = \frac{K \prod_i (j\omega\tau_i + 1) \prod_j (-\omega^2\tau_j^2 + 2\zeta_j\tau_j j\omega + 1)}{(j\omega)^a \prod_k (j\omega T_k + 1) \prod_l (-\omega^2 T_k^2 + 2\zeta_l T_k j\omega + 1)} \quad (6.12)$$

2) Pe axa pulsărilor, gradată logaritmic, se dispun în ordine crescătoare pulsăriile de frângere $\omega_f = 1/\tau_1$, $\omega_f = 1/\tau_2$, $\omega_f = 1/T_k$, $\omega_f = 1/T_1$.

6.2.1.1. Caracteristica logaritmica a modulului

3a) Se trasează caracteristicile logaritmice ale modulului pentru fiecare factor individual din (6.12).

3b) Se calculează factorul de amplificare în decibeli, $K^{dB} = 20\lg K$ și se notează pe grafic punctul $(1, K^{dB})$. Se trasează prin acest punct asimptota de joasă frecvență, care are panta $-20\alpha dB/dec$. La intersecția asimptotei de joasă frecvență cu linia verticală de la prima frecvență de frângere se modifică panta acesteia cu $\pm 20dB/dec$ sau $\pm 40dB/dec$, după cum verticala corespunde unei pulsări de frângere de ordinul unu sau doi de la numărător sau numitor. Pasul se repetă până la ultima pulsărie de frângere. Se corectează caracteristica asymptotică în zona pulsărilor de frângere cu $\pm 3dB/dec$ pentru un element de ordinul unu de la numărător, respectiv numitor și cu $20\lg(1/2\zeta)$ pentru un element de ordinul doi.

6.2.1.2. Caracteristica logaritmica a fazei

4) Se desenează pe abscisa unui grid (rețea de linii ajutătoare) semilogaritmic (fază-pulsărie), punctele corespunzătoare pulsărilor $\omega_f/10$ și $10\omega_f$ pentru toți factorii din funcția de transfer. Pentru fiecare factor individual din (6.12) se trasează caracteristicile aproximative ale fazelor.

5) Se trasează asimptota de joasă frecvență a caracteristicii fazelor, care este o dreaptă orizontală la -90° . În continuare, se adună grafic caracteristicile aproximative de fază ale factorilor individuali din funcția de frecvență.

Funcția **bode** din Matlab este folosită pentru trasarea caracteristicilor semilogaritmice ale unei funcții de transfer.

Funcția **bode** se apelează astfel:

`bode(num, den)` sau `bode(num, den, ω)`.

Pe de altă parte, dacă se cunoaște funcția de transfer și pulsăria, se pot determina amplitudinea, respectiv faza, astfel:

`[mag, phase] = bode(num, den, ω)`.

O ultimă variantă de apelare a funcției **bode** este:

`[mag, phase, ω] = bode(num, den)`,

cu ajutorul căreia se determină amplitudinea, faza și pulsăria.

Observație

`num, den` reprezintă numărătorul, respectiv numitorul funcției de transfer, `mag=magnitudinea (amplitudinea), phase=faza, iar ω=pulsăria.`

În spațiul stărilor, funcția **bode** se apelează cu una din sintaxele:

`bode(A, B, C, D, u, ω)`,

`bode(A, B, C, D, u)`,

`bode(A, B, C, D)`,

`[mag, phase] = bode(A, B, C, D, u, ω)`,

`[mag, phase, ω] = bode(A, B, C, D, u)`,

unde `A, B, C, D` sunt matricele din spațiul stărilor, iar `u` intrarea sistemului.

6.3. Diagrame Nyquist

Un alt criteriu important de studiu al stabilității sistemelor continue (din domeniul frecvenței) este criteriul lui Nyquist. Acesta este un criteriu frecvențial și are avantajul că folosește caracteristicile amplitudine-pulsărie și fază-pulsărie (caracteristicile Bode). Pentru ca un sistem liniar și continuu stabil în stare deschisă să fie stabil și în stare închisă, este necesar și suficient ca punctul $(-1, j0)$ să nu se afle în interiorul caracteristicii amplitudine-pulsărie al sistemului deschis.

Funcția **nyquist** din Matlab calculează răspunsul în frecvență pentru un sistem liniar. Reprezentarea grafică a acestui răspuns poartă numele de loc Nyquist sau loc de transfer sau hodograf. Locul Nyquist se utilizează în analiza și sinteza sistemelor de reglare automată.

Dacă sistemul liniar este reprezentat prin funcția de transfer:

$$G(s)H(s) = \frac{\text{num}(s)}{\text{den}(s)} \quad (6.13)$$

atunci răspunsul Nyquist se obține cu una din sintaxele:

```
[re, im] = nyquist(num, den, ω),
[re, im, ω] = nyquist(num, den),
nyquist(num, den),
nyquist(num, den, ω).
```

Dacă sistemul liniar este reprezentat în spațiul stărilor prin ecuațiile:

$$\dot{x}(t) = Ax(t) + Bu(t), \quad y(t) = Cx(t) + Du(t), \quad (6.14)$$

atunci:

```
[re, im] = nyquist(A, B, C, D, u, ω),
[re, im, ω] = nyquist(A, B, C, D, u),
nyquist(A, B, C, D, u),
nyquist(A, B, C, D, u, ω),
```

calculează răspunsul în frecvență corespunzător componentei `u` a intrării. Vectorul `ω` (se poate omite) specifică pulsăriile pentru care este evaluat răspunsul Nyquist.

Funcția **nyquist** determină răspunsul în frecvență sub formă a două matrice, `re, im`, care au tot atâtea coloane câte componente are vectorul de ieșire `y` și același număr de linii.

Observație:

Atât caracteristicile Bode cât și Nyquist, pot fi folosite pentru determinarea răspunsului în frecvență al unui sistem. Dacă sistemul liniar are poli situați pe axa imaginară $j\omega$, iar vectorul `ω` conține frecvențele corespunzătoare acestor puncte, atunci matricea $(j\omega I - A)$ este singulară și funcția **nyquist** furnizează mesajul: „Matrix is singular to working precision”.

6.4. Exerciții propuse

Exercițiu 6.4.1

Funcția de transfer a sistemului în circuit închis este:

$$G_0(s) = \frac{4}{s^2 + 2s + 4}$$

Să se determine răspunsul în frecvență al sistemului (pulsăția are următoarele valori $\omega = [0:3]$ cu pasul 0,01).

Exercițiul 6.4.2

Pornind de la exemplul anterior se introduce un pol suplimentar, rezultând astfel următoarea funcție de transfer:

$$G_0(s) = \frac{10}{(s + 2.5)(s^2 + 2s + 4)}$$

Să se reprezinte răspunsul sistemului la intrarea treaptă și răspunsul în frecvență al sistemului (în aceeași fereastră vor apărea 2 grafice). Se va folosi funcția **subplot** ($t = [0:4]$, pasul 0,02 și $\omega = [0:3]$, pasul 0,01).

Exercițiul 6.4.3

Se dă un sistem închis de ordinul trei în circuit închis:

$$G_0(s) = \frac{750}{s^3 + 36s^2 + 205s + 750}$$

Scopuri:

- Determinarea polilor sistemului.
- Obținerea unei reduceri a ordinului modelului.
- Reprezentarea răspunsului în frecvență, la intrarea treaptă a sistemului de ordinul 3 și a sistemului redus (în aceeași fereastră vor apărea patru grafice, două reprezentând răspunsurile în treaptă și în frecvență pentru sistemul de ordinul 3, iar celelalte două fiind răspunsurile în treaptă și în frecvență pentru sistemul redus).

Date $t = [0:2]$, pasul 0,2; $\omega = [0:8]$, pasul 0,2.

Exercițiul 6.4.4

Pentru sistemele liniare cu funcțiile de transfer în circuit deschis:

$$G(s) = \frac{150(s + 0.2)(s + 1)}{s(s + 3)(0.01s^2 + 0.1s + 1)},$$

$$G(s) = \frac{20(s^2 + s + 1)}{s(s + 2)(0.01s^2 + 0.1s + 1)}$$

- Să se traseze caracteristicile de frecvență amplitudine-pulsăție și fază-pulsăție folosind funcția **bode**.
- Să se traseze diagramele bode folosind funcția **plot**; pentru calcularea pulsăției se va folosi funcția **logspace**, care se va aplica, pentru primul sistem, pe intervalul [-3,3], respectiv [-2,2] pentru cel de-al doilea sistem. Se vor reprezenta magnitudinea în funcție de pulsăție, respectiv faza în funcție de pulsăție.

- Să se determine marginile de amplitudine și de fază.

Exercițiul 6.4.5

Fie sistemul caracterizat prin funcția de transfer în circuit deschis:

$$G(s) = \frac{10}{s^3 + 3s^2 + 2s}.$$

Să se traseze locul de transfer al funcției de transfer. Pentru determinarea pulsăției se va folosi funcția **logspace**, care se va aplica pe intervalul [-1,2].

Exercițiul 6.4.6

Se consideră sistemul reprezentat în spațiul stăriilor:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -25 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Să se traseze diagramele **bode** pentru următoarele funcții de transfer $Y_1(j\omega)/U_1(j\omega)$, $Y_2(j\omega)/U_1(j\omega)$ (în aceste două cazuri se consideră $U_2(j\omega) = 0$), respectiv $Y_2(j\omega)/U_1(j\omega)$, $Y_2(j\omega)/U_2(j\omega)$ (se consideră $U_1(j\omega) = 0$).

Exercițiul 6.4.7

Se consideră sistemul reprezentat în spațiul stăriilor:

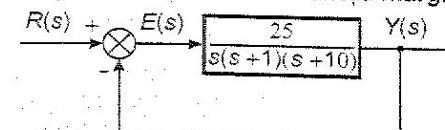
$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -1 & -1 \\ 6.5 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Să se traseze diagramele **nyquist** referitoare la intrarea u_1 , respectiv la intrarea u_2 , în diagrame diferite.

Exercițiul 6.4.8

Să se traseze diagramele **bode** pentru sistemul din figură și să se determine marginile de amplitudine și de fază (se va folosi funcția **margin**).



Exercițiul 6.4.9

Se consideră următoarele două sisteme cu funcțiile de transfer în circuit închis:

$$G_{01}(s) = \frac{1}{s + 1},$$

$$G_{02}(s) = \frac{1}{3s + 1}.$$

Să se traseze răspunsul în frecvență (în coordonate logaritmice), răspunsul în timp la intrarea treaptă unitară și răspunsul în timp la intrarea rampă unitară (în același fereastră se vor trasa trei grafice, în fiecare fereastră reprezentându-se răspunsurile pentru ambele sisteme). Să se compare cele două sisteme și să se determine care este cel mai rapid. Date $\omega = [0 : 20]$, pasul 0,1; $t = [0 : 8]$, pasul 0,01.

LOCUL RĂDĂCINILOR. UTILITARUL SISOTOOL

OBIECTIVE

- Trasarea locului rădăcinilor pentru o serie de sisteme.
- Prezentarea utilitarului Sisotool; folosirea acestuia în analiza și simularea sistemelor.

7.1. Locul rădăcinilor

Metoda locului rădăcinilor elaborată de W. R. Evans¹ constă în trasarea locului rădăcinilor ecuației caracteristice a sistemului închis în funcție de variația unui parametru din această ecuație, parametru care, în majoritatea cazurilor, este factorul de amplificare al sistemului deschis (multe performanțe ale sistemului depind direct de factorul de amplificare). Această metodă este utilă în proiectarea sistemelor liniare de reglare.

7.1.1. Rezumatul pașilor algoritmului de trasare a locului rădăcinilor

- 1) Numărul ramurilor locului rădăcinilor este egal cu n (numărul polilor funcției de transfer a sistemului deschis). Se plasează, în planul complex, polii și zerourile funcției $G(s)H(s)$. Ramurile locului rădăcinilor încep din polii sistemului deschis și se termină în zerouri (cu valoarea finită) sau la infinit.
- 2) La stânga unui număr impar de poli reali plus zerouri reale, se desenează locul rădăcinilor pe axa reală.
- 3) Se trasează asimptotele ramurilor locului rădăcinilor:
 - numărul asimptotelor este egal cu $n - m$;
 - unghiurile asimptotelor $= \phi_k = \frac{\pm 180^\circ (2k+1)}{n-m}$, ($k = 0, 1, 2, \dots$);
 - abscisa punctului de intersecție a asimptotelor $\sigma_a = \frac{\sum p_i - \sum z_j}{n-m}$.
- 4) Se determină unghiul de plecare (unghiul de sosire) al locului rădăcinilor dintr-un pol complex (într-un zero complex).

¹ Walter R. Evans - recunoscut inginer, care, în anul 1948 a deschis drumul proiectării servomecanismelor prin metoda locului rădăcinilor.

- unghiul de plecare dintr-un pol complex = $\varphi_p = 180^\circ - \sum \theta_i + \sum \Phi_j$;
- unghiul de sosire într-un zero complex = $\varphi_s = 180^\circ - \sum \Phi_i + \sum \theta_j$,

în care, Φ sunt unghiiurile ale vectorilor care încep în zerouri, iar θ reprezintă unghiiurile vectorilor complecși care pleacă din poli.

5) Se găsesc punctele de ramificații (de sosire și de plecare).

6) Se determină punctele în care locul rădăcinilor intersectează axa imaginäră. Aceste puncte pot fi găsite folosind criteriul de stabilitate Routh, sau rezolvând pentru ω și K ecuația complexă:

$$1 + \frac{KB(j\omega)}{A(j\omega)} = 1 + \frac{K(j\omega + z_1)(j\omega + z_2)\dots(j\omega + z_m)}{(j\omega + p_1)(j\omega + p_2)\dots(j\omega + p_n)} = 0$$

7) Luându-se o serie de puncte suficient de îndepărtate de originea planului complex, se schițează locul rădăcinilor. Se localizează polii sistemului în circuit închis și se determină, folosindu-se condiția modulului, valorile corespunzătoare lui K , sau se determină poziția polilor sistemului închis pentru o valoare dată a amplificării K .

Funcțiile de mai jos, permit trasarea locului rădăcinilor:

`rlocus (num, den, K)` sau
`rlocus (num, den)`,

unde:

`num` = numărătorul funcției de transfer;

`den` = numitorul funcției de transfer;

K = factorul de amplificare al sistemului deschis din Control System Toolbox.

În primul caz, locul rădăcinilor este gradat după valorile lui K , iar în al doilea caz, este determinat automat. Aceste funcții permit și trasarea locului rădăcinilor după introducerea unor poli și zerouri suplimentare, cu scopul de a corespunde unor performanțe impuse.

În cazul în care sistemul este reprezentat în spațiul stărilor, funcția `rlocus` poate fi apelată prin una din următoarele sintaxe:

`rlocus (A, B, C, D, K)`,
`rlocus (A, B, C, D)`,

unde: A, B, C, D reprezintă matricele ce definesc spațiul stărilor.

Apelând funcția `rlocus` prin una din sintaxele:

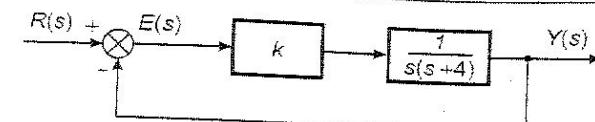
`[r, k] = rlocus (num, den, K)`,
`[r, k] = rlocus (num, den)`,
`[r, k] = rlocus (A, B, C, D, K)`,
`[r, k] = rlocus (A, B, C, D)`,

se pot obține matricea r și vectorul k , ce conțin toate valorile corespunzătoare factorului de amplificare. Fiecare linie din matricea r corespunde unui factor de amplificare din vectorul k . Locul rădăcinilor în această situație se poate trasa cu ajutorul funcției `plot`, astfel:

`plot(r)`.

Exemplu:

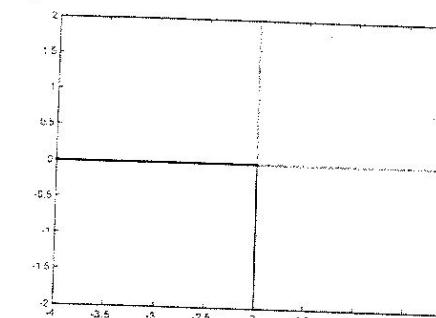
Fie sistemul din figură:



Pentru trasarea locului rădăcinilor sistemului deschis, în Matlab s-a realizat următorul program:

```
k=[0:0.5:12]; num=[1]; den=[1 4 0];
rlocus(num, den, k)
```

S-a obținut următorul grafic:



Se observă în figura de mai sus că locul rădăcinilor pe axa reală este segmentul $[-4, 0]$; $n - m = 2$, deci două ramuri se termină la infinit; asymptotele au unghiiurile cu axa reală $\theta = \pm 90^\circ$; intersecția asymptotelor pe axa reală se face în: $\sigma_a = (-4 - 0)/2 = -2$.

7.2. Utilitarul Sisotool

Sisotool este o interfață grafică, ce permite simularea unor sisteme cu o singură intrare și o singură ieșire folosind diverse metode (diagrame Bode, locul rădăcinilor și diagrame Nichols). În fereastra de comandă din Matlab se va tipări „sisotool” care va lansa în execuție interfața grafică (figura 7.2.1).

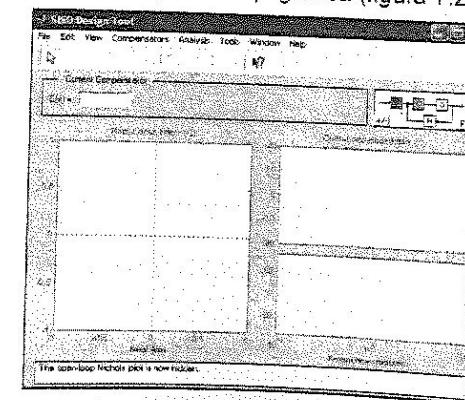


Fig. 7.2.1 Interfața Sisotool

Reprezentarea grafică din stânga corespunde locului rădăcinilor, iar celelalte două, din partea dreaptă, reprezintă amplitudinea și faza caracteristicii Bode.

Pentru a analiza un sistem, mai întâi acesta trebuie lansat în execuție. Acest lucru se realizează alegând opțiunea **import** din meniu File, moment în care pe ecran va apărea o nouă interfață (figura 7.2.2). Inițial, toate blocurile G , H , F , C sunt setate la valoarea 1. Sistemul poate fi încărcat în mai multe moduri:

- din fereastra de comandă, unde sistemul este reprezentat sub forma unei funcții de transfer cu una din funcțiile: **tf**, **ss**, etc.;
- dintr-un fișier cu extensia **.mat**;
- din programul Simulink.

Se poate observa că în figura 7.2.2 s-a creat deja un sistem în fereastra de comandă cu funcția **tf** și încărcat în blocul G . Cu ajutorul butoanelor cu săgeți din mijlocul interfeței, se încarcă sistemele dorite blocurilor de interes G , H , F , C . Ca și exemplu, se consideră funcția de transfer a unui sistem de următoarea formă:

$$G(s) = \frac{s+2}{s^2 + 2s + 4}$$

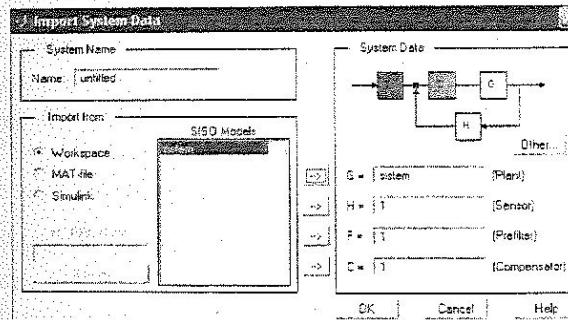


Fig. 7.2.2 Încărcarea sistemului

În fereastra de comandă din Matlab, funcția de transfer a fost creată cu funcția **tf**, astfel:

```
sistem=tf([1 2],[1 2 4])
```

și asociată blocului G . După apăsarea butonului OK, se vor trasa caracteristicile Bode și locul rădăcinilor (figura 7.2.3). Modificând factorul de amplificare pe graficul locului rădăcinilor (cu ajutorul pătratelor de culoare roșie), caracteristicile Bode se vor modifica în funcție de acesta. Grafic, se pot adăuga poli și zerouri (reali sau complecsi); de asemenea, se pot șterge zerouri sau poli, dar numai aceia care nu au fost introdusi de către funcția de transfer din fereastra de comandă.

De obicei, funcția de transfer a sistemului, care se introduce din fereastra de comandă sau din Simulink, se asociază blocului G , iar factorul de amplificare și polii sau zerourile adiționale, blocului C .

Cu ajutorul butonului **+/-** de pe interfață grafică se poate schimba semnul reactiei (dacă reacția este unitară, atunci blocul $H=1$), iar cu ajutorul butonului **FS** modul de interconectare al blocurilor.

Cu ajutorul utilitarului Sisotool se mai pot realiza:

- răspunsul sistemului închis la o intrare treaptă unitară (Response to Step Command - meniu Analysis);
- conversia sistem continuu - sistem discret (Continuous / Discret Conversions - meniu Tools);
- vizualizarea funcției de transfer a sistemului ce a fost importat și a proprietăților acestuia (System Data - meniu View);
- trăsarea schemei bloc în Simulink (Draw Simulink Diagram).

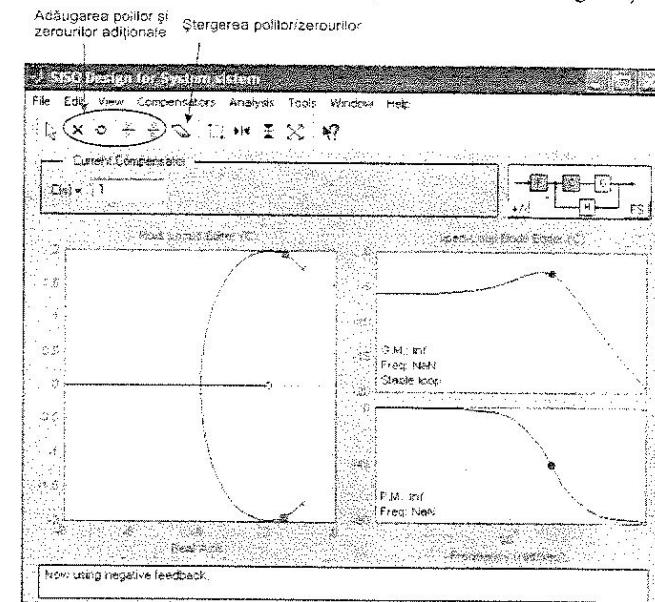


Fig. 7.2.3 Trasarea caracteristicilor Bode și a locului rădăcinilor

7.3. Exerciții propuse

Exercițiul 7.3.1

Să se traseze locul rădăcinilor pentru sistemele reprezentate de următoarele funcții de transfer ($k = [0:12]$ cu pasul 0,5).

$$G(s)H(s) = \frac{k(s+5)}{s^2(s+3,6)}$$

$$G(s)H(s) = \frac{k(s+2)(s+4)}{s^2(s+3,6)}$$

$$G(s)H(s) = \frac{k}{s^3 + 8s^2 + 19s + 12}$$

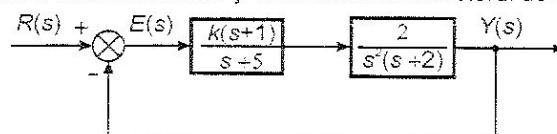
$$G(s)H(s) = \frac{k}{(s+1)(s+3-j)(s+3+j)}$$

Să se specifice:

- Poziunea de pe axa reală care aparține locului rădăcinilor.
- Câte ramuri există la infinit?
- Unghiul asymptotelor cu axa reală.
- Intersecția asymptotelor pe axa reală.

Exercițiu 7.3.2

Să se traseze locul rădăcinilor și să se determine factorul de amplificare k



Exercițiu 7.3.3

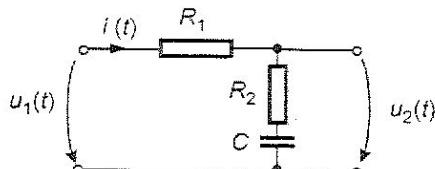
Să se determine și să se analizeze locul rădăcinilor și caracteristicile Bode pentru sistemul cu următoarea funcție de transfer în circuit deschis:

$$G(s)H(s) = \frac{1}{s(s+2)}$$

Odată cu introducerea unui pol adițional ($-p_3 = -4$), să se analizeze modul cum se modifică locul rădăcinilor și caracteristicile Bode și să se determine stabilitatea sistemului.

Exercițiu 7.3.4

Să se determine și să se analizeze locul rădăcinilor, caracteristicile Bode și curba polară, pentru sistemul din figură ($R_1 = 1k\Omega$, $R_2 = 2k\Omega$, $C = 2mF$):



DESCRIEREA ÎN SPAȚIUL STĂRILOR A SISTEMELOR CONTINUIE

OBIECTIVE

- Conversia din spațiu stării în funcții de transfer și invers.
- Controlabilitatea și observabilitatea sistemelor.
- Reducerea schemelor bloc în spațiu stării.

8.1. Descrierea intrare - stare - ieșire

Sistemul liniar continuu este reprezentat de ecuația matriceală a stării:

$$\begin{bmatrix} \dot{x}_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 \\ -a_n & -a_{n-1} & -a_{n-2} & \dots & -a_1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} u(t), \quad (8.1)$$

unde:

$x(t)$ - matricea (vector) stării; $u(t)$ - matricea intrărilor;

A - matricea sistemului (constantă); B - matricea de intrare (constantă).

Această descriere are avantajul că soluția sistemului poate fi obținută destul de ușor, atât prin metode analogice, cât și numerice. De asemenea, metoda poate fi extinsă și la sistemele neliniare.

Pentru exemplificarea unui mod de alegere a variabilelor de stare, se consideră un sistem descris de ecuația diferențială de ordinul n cu o singură intrare:

$$a_0 \frac{d^n y(t)}{dt^n} + a_1 \frac{d^{n-1} y(t)}{dt^{n-1}} + \dots + a_{n-1} \frac{dy(t)}{dt} + a_n y(t) = b_0 u(t), \quad (8.2)$$

unde $y(t)$ este mărimea de ieșire (răspunsul sistemului) și $u(t)$ este mărimea de intrare. Modelul de stare pentru acest sistem nu este unic, ci depinde de modul cum se aleg variabilele de stare. Un set de variabile de stare, întâlnit foarte des, este setul variabilelor de fază în care se alege prima variabilă de stare x_1 (care poate fi mărimea de ieșire, eroarea etc.), iar celelalte $(n-1)$ mărimi de stare, ce completează setul, sunt derivele succesive pînă la $(n-1)$ ale mărimii x_1 , adică:

$$x_1 = y, x_2 = \dot{y}, x_3 = \ddot{y}, \dots, x_n = y^{n-1} \quad (8.3)$$

Cu această alegere, ecuația (8.1) se poate scrie sub forma (8.3):

$$\begin{aligned} \dot{x}_1 &= x_2 \\ x_2 &= x_3 \\ &\vdots \\ x_{n-1} &= x_n \\ x_n &= -a_n x_1 - a_{n-1} x_2 - \dots - a_1 x_n + u(t) \end{aligned} \quad (8.4)$$

Sistemul (8.4) poate fi scris sub forma matriceală:

$$\begin{bmatrix} \dot{x}_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 \\ -a_n & -a_{n-1} & -a_{n-2} & \dots & -a_1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ u(t) \end{bmatrix}, \quad (8.5)$$

iar ecuația ieșirii este:

$$y = [1 \ 0 \ 0 \ \dots \ 0] \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad (8.6)$$

8.1.1. Trecerea de la funcția de transfer la spațiul stărilor

Matlab conține un set de funcții care fac trecerea de la funcția de transfer la matricele A, B, C, D , care se realizează cu funcția:

`[A, B, C, D] = tf2ss(num, den),`

unde A, B, C și D sunt matrice de dimensiuni $n \times n, n \times m, r \times n$ și respectiv $r \times m$.

8.1.2. Trecerea de la spațiul stărilor la funcția de transfer

Fiind date ecuațiile matriceale ale stării și ieșirii:

$$\begin{aligned} x(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned} \quad (8.7)$$

și aplicând transformata Laplace (în condiții inițiale nule) la aceste ecuații, rezultă:

$$Y(s) = C(sI - A)^{-1}BU(s) + DU(s)$$

sau

$$G(s) = \frac{Y(s)}{U(s)} = C(sI - A)^{-1}B + D = C\Phi(s)B + D, \quad (8.8)$$

unde I este matricea unitate, iar $\Phi(s) = (sI - A)^{-1} = \frac{\text{adj}(sI - A)}{\det(sI - A)}$ se numește matricea fundamentală a sistemului modelat la stare. Matricea de tranziție a stărilor, $\phi(t)$, se obține aplicând transformata Laplace inversă matricei fundamentale.

În Matlab transferul este făcut cu funcția `ss2tf`, cu următoarea sintaxă:

`ss2tf(A, B, C, D, u),`

ce convertește ecuația stării în funcția de transfer pentru intrarea u .

Funcția de transfer se poate obține și sub formă vectorială a zerourilor și polilor, folosindu-se funcția:

`ss2zp(A, B, C, D, u),`

unde A, B, C, D reprezintă matricele din spațiul stărilor, iar u reprezintă intrarea. Matricea de tranziție a stărilor se determină cu funcția:

`expm(At),`

unde A este o matrice pătratică, iar t reprezintă timpul. Pentru crearea unor obiecte simbol se va folosi funcția `syms`.

Soluția numerică a ecuației de stare permite simularea numerică a răspunsului unui sistem în reprezentarea variabilei de stare. Pentru sistemele liniare continue funcția este:

`[y, x] = lsim(A, B, C, D, u, t)`

și simulează răspunsul sistemului la o intrare oarecare.

În cazul intrărilor impuls Dirac și treptă, funcțiile sunt:

`[y, x] = impulse(A, B, C, D, u, t)`

și respectiv

`[y, x] = step(A, B, C, D, u, t).`

8.1.3. Controlabilitate și observabilitate

Controlabilitatea unui sistem caracterizează capacitatea de modificare (de control) a întregului set de variabile de stare de către mărimea de intrare. Un sistem descris la stare de către matricele (A, B) se spune că este controlabil dacă există o mărime de intrare u care poate transfera orice stare initială $x(0)$ în orice altă stare $x(t)$. Proprietatea de controlabilitate se poate verifica analitic cu ajutorul matricei de controlabilitate: $P = [B \ AB \ A^2B \ \dots \ A^{n-1}B]$. Matematic, sistemul considerat este controlabil dacă $\text{rang } P = n$.

Condiția de mai sus poate fi îndeplinită dacă determinantul matricei de controlabilitate P este diferit de zero. Observabilitatea unui sistem caracterizează capacitatea de estimare a unei variabile de stare din valorile mărimii de ieșire. Se spune că un sistem este observabil, dacă mărimea de ieșire are componente determinante (este influențată) de toate variabilele de stare. Un sistem este observabil dacă și numai dacă, o stare initială $x(0)$ poate fi determinată prin observare, pe un interval de timp finit T , al mărimii de ieșire $y(t)$ și al mărimii de intrare $u(t)$. Sistemul este observabil dacă determinantul matricei de observabilitate:

$$Q = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix}$$

este diferit de zero.

Un sistem care este descris în forma controlabilă cu variabile de fază este întotdeauna și observabil. În Matlab controlabilitatea și observabilitatea unui sistem se determină cu funcțiile:

`ctrb(A, B)` și `obsv(A, C)`,

unde A , B și C reprezintă matricele din spațiul stăriilor.

8.1.4. Reducerea schemelor bloc în spațiul stăriilor

Limbajul Matlab, în cadrul funcțiilor grupate în "Control Toolbox", realizează funcțiile **blkbuild** și **connect**, care convertează diagramele bloc în modele în spațiul stăriilor. Blocurile funcțiilor de transfer sunt numerotate de la 1 la valoarea numărului de blocuri **nblocks** (numărul total de blocuri) și convertește fiecare bloc la o reprezentare în spațiul stăriilor.

Funcția:

`[A, B, C, D] = connect(a, b, c, d, q, iu, iy);`

conectează blocurile în concordanță cu o matrice predefinită q care specifică interconectarea blocurilor. Primul element al fiecărei linii a matricei q este numărul blocului. Celelalte elemente indică sursa intrărilor blocurilor de însumare. Când intrarea la sumator este legată negativ, numărul blocului apare cu semnul minus. Vectorii linie iu și iy indică blocurile de intrare și ieșire. În final se obține funcția de transfer echivalentă pentru intrarea iu :

`[num, den] = ss2tf(A, B, C, D, iu).`

8.2. Exerciții propuse

Exercițiu 8.2.1

Fie un sistem reprezentat prin următoarea funcție de transfer:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{s^2 + 7s + 2}{s^3 + 9s^2 + 26s + 24}.$$

Să se determine matricele stăriilor pentru sistemul reprezentat prin funcția de transfer.

Exercițiu 8.2.2

Fie un sistem reprezentat în spațiul stăriilor de următoarele matrice:

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 1 \\ -1 & -2 & -3 \end{bmatrix}, B = \begin{bmatrix} 10 \\ 0 \\ 0 \end{bmatrix}, C = [1 \ 0 \ 0], D = [0]$$

Să se determine:

- funcția de transfer a sistemului;
- matricea fundamentală;
- matricea de tranziție a stăriilor;
- dacă sistemul este controlabil și observabil.

Exercițiu 8.2.3

Fie sistemul reprezentat în spațiul stăriilor ($t = [0:10]$ cu pasul 0,05):

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -1 & -0.5 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0.5 \\ 0 \end{bmatrix} u \text{ și } y = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

- Să se simuleze răspunsul sistemului la intrarea treaptă.

- Să se simuleze răspunsul sistemului la intrarea impuls unitar.

Exercițiu 8.2.4

Se dă sistemul modelat la stare:

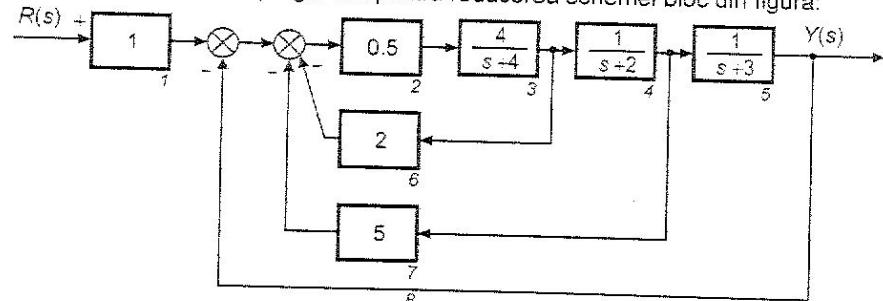
$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -6 & -11 & -6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ u(t) \end{bmatrix} \text{ și } y = \begin{bmatrix} 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Se impun condițiile inițiale $x(0) = \begin{bmatrix} 1 \\ 0.5 \\ -0.5 \end{bmatrix}$, $t = [0:4]$ cu pasul 0,05.

- Să se simuleze răspunsul sistemului la intrarea treaptă.
- Să se simuleze răspunsul sistemului la intrarea $\sin(2\pi t)$.

Exercițiu 8.2.5

Să se realizeze programul pentru reducerea schemei bloc din figură:



Exercițiu 8.2.6

Se consideră sistemul definit la stare sub următoarea formă:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -25 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

Sistemul are două intrări și două ieșiri. Să se determine cele patru funcții de transfer $Y_1(s)/U_1(s)$, $Y_1(s)/U_2(s)$, $Y_2(s)/U_1(s)$, $Y_2(s)/U_2(s)$ (când sistemulul își se aplică intrarea u_1 se presupune că intrarea u_2 este zero și vice-versa).

SISTEME DISCRETE

OBIECTIVE

- Schițarea polilor și zerourilor în planul „z”.
- Trecerea de la un sistem continuu la un sistem discret și invers.
- Reducerea schemelor bloc.

9.1. Introducere

Apariția calculatoarelor numerice a făcut posibilă introducerea acestora în structura unui sistem automat. Calculatorul numeric joacă rolul regulatorului din sistemele continue. Având în vedere că, după cum se știe, calculatorul numeric operează cu numere, este necesar ca semnalele continue să fie convertite în numere, iar după procesare, aceste numere să fie reconvertește în semnale continue care pot fi aplicate instalației tehnologice. Rezultă că este necesar ca un sistem discret, pe lângă calculatorul numeric, să conțină două blocuri și anume un convertor analog-numeric (A/N) și un convertor numeric-analog (N/A).

Converteoarele A/N și N/A operează periodic, transformând semnalul de ieșire continuu în semnal numeric care este prelucrat de calculatorul numeric și, respectiv, semnalul numeric de la ieșirea calculatorului în semnal continuu care, amplificat, se aplică la servomotor.

În contrast cu sistemele continue care sunt descrise cu ajutorul ecuațiilor diferențiale, sistemele discrete sunt descrise cu ajutorul ecuațiilor cu diferențe (recursive).

Metoda transformării Laplace este folosită în analiza sistemelor liniare continue și, în mod similar, metoda transformării Z este folosită în analiza sistemelor discrete liniare și invariante în timp.

Funcția de reglare este determinată de calculatorul numeric. Interfața intrării în calculator este un convertor analog-numeric (A/N) care convertește semnalul continuu de eroare într-o formă ce poate fi procesată de către calculatorul numeric. Ieșirea numerică din calculator este convertită de către convertorul numeric-analog (N/A) într-o formă ce poate fi aplicată instalației (procesului) care se automatizează. În plus, în cadrul procesului de conversie, se realizează și o filtrare cu un filtru trece-jos a semnalelor discretizate în timp și amplitudine.

9.2. Transformata Z

Într-o manieră similară trecerii de la ecuații diferențiale cu ajutorul transformației Laplace, se va trece de la ecuațiile cu diferențe cu ajutorul

transformatei Z. Transformata Z este definită pentru un sir de numere. Funcția $F(z)$, care este transformata Z a lui $f(kT_e)$ (sir de numere), este o serie de puteri în z^{-k} ai cărei coeficienți sunt egali cu valorile șirului de numere, adică:

$$F(z) = F(s) \Big|_{s=(1/T_e)\ln z} = f(0) + f(T_e)z^{-1} + \dots + f(kT_e)z^{-k} + \dots = \sum_{k=0}^{+\infty} f(kT_e)z^{-k} \quad (9.1)$$

La definirea transformatei Z se poate scrie suma de la $k = -\infty$ la $k = +\infty$, adică:

$$F(z) = \sum_{k=-\infty}^{+\infty} f(kT_e)z^{-k} \quad (9.2)$$

O altă variantă de definire a transformatei Z are la bază descrierea specifică unui semnal discret. În acest caz, transformata Z se definește cu relația:

$$F(z) = \sum_{k=-\infty}^{+\infty} f[k]z^{-k} \quad (9.3)$$

Transformatele Laplace și Z ale funcțiilor continue simple sunt prezentate în tabelul 9.2.1:

Tab. 9.2.1 Transformatele Laplace și Z ale funcțiilor continue simple

Nr. crt.	Funcția de timp $f(t), t \geq 0$	Transformata Laplace	Transformata Z
1	Impuls Dirac, $\delta(t)$	1	1
2	Impuls unitar, $\delta_1(t)$	1	1
3	Treaptă unitară, $1_+(t)$	$\frac{1}{s}$	$\frac{z}{z-1}$
4	t	$\frac{1}{s^2}$	$\frac{zT_e}{(z-1)^2}$
5	$\frac{t^2}{2}$	$\frac{1}{s^3}$	$\frac{zT_e^2(z+1)}{2(z-1)^3}$
6	e^{-at}	$\frac{1}{s+a}$	$\frac{z}{z-e^{-aT_e}}$
7	te^{-at}	$\frac{1}{(s+a)^2}$	$\frac{zT_e e^{-aT_e}}{(z-e^{-aT_e})^2}$

Nr. crt.	Funcția de timp $f(t), t \geq 0$	Transformata Laplace	Transformata Z
8	$\sin \omega t$	$\frac{\omega}{s^2 + \omega^2}$	$\frac{z \sin \omega T_e}{z^2 - 2z \cos \omega T_e + 1}$
9	$\cos \omega t$	$\frac{s}{s^2 + \omega^2}$	$\frac{z(z - \cos \omega T_e)}{z^2 - 2z \cos \omega T_e + 1}$
10	$e^{-at} \sin \omega t$	$\frac{\omega}{(s+a)^2 + \omega^2}$	$\frac{ze^{-aT_e} \sin \omega T_e}{z^2 - 2ze^{-aT_e} \cos \omega T_e + e^{-2aT_e}}$
11	$e^{-at} \cos \omega t$	$\frac{s+a}{(s+a)^2 + \omega^2}$	$\frac{z^2 - ze^{-aT_e} \cos \omega T_e}{z^2 - 2ze^{-aT_e} \cos \omega T_e + e^{-2aT_e}}$

9.3. Reprezentarea polilor și zerourilor în planul „z”

În cazul sistemelor discrete, localizarea pol-zero se face folosind funcția **pzmap** (în plus, pentru vizualizarea polilor și zerourilor în planul „z” se va folosi funcția **zgrid**) sau funcția **zplane** care are următoarea sintaxă:

zplane(z, p),

unde: z și p sunt zerourile, respectiv polii funcției de transfer a sistemului discret.

9.4. Transformări între sisteme

Trecerea de la un sistem liniar la un sistem discret se face cu ajutorul funcției **c2dm** (sistemul liniar poate fi reprezentat în spațiul stărilor printr-o funcție de transfer). Se mai poate utiliza și funcția **c2d** care are o sintaxă asemănătoare. Invers, trecerea de la un sistem discret la un sistem liniar se face folosind funcțiile **d2cm** sau **d2c**, care au sintaxă asemănătoare funcției **c2dm**.

[numDz, denDz] = c2dm(num, den, Ts, 'method'),
[F, G, H, J] = c2dm(A, B, C, D, Ts, 'method'),

unde: 'method' reprezintă metoda care se folosește, care poate fi:

- „zoh” - convertește în discret cu un element de reținere de ordin zero;
- „foh” - convertește în discret cu un element de reținere de ordin unu;
- „tustin” - convertește în discret prin folosirea aproximării biliniare (tustin); se face schimbarea de variabilă $s = (2/T_e)[(z-1)/(z+1)]$;
- „prewarp” - convertește în discret prin folosirea aproximării biliniare cu întreținerea frecvenței; se specifică frecvența critică printr-un argument adițional;
- „matched” - convertește sistemul SISO în discret folosind metoda „matched pole-zero”, metodă ce conservă repartitia poli-zerouri a sistemului continuu;

T_s este perioada de eşantionare;
numărătorul funcției de transfer;
denomina numitorul funcției de transfer.

9.5. Funcții de transfer Z echivalente

Trebuie remarcat, de la început, că nu există reguli general valabile, ca la sistemele continue, care să permită reducerea rapidă a schemelor bloc complexe cu funcții de transfer z.

9.5.1. Conexiune în serie (cascadă)

Există două situații posibile de conectare a elementelor continue, cazuri evidențiate de schemele bloc din figurile 9.5.1a și 9.5.1b.

9.5.1.1. Elementele continue conectate în serie cu eşantionor intercalat

Se menționează că toate eşantionoarele au aceeași frecvență de comutare, $f_e = 1/T_e$.

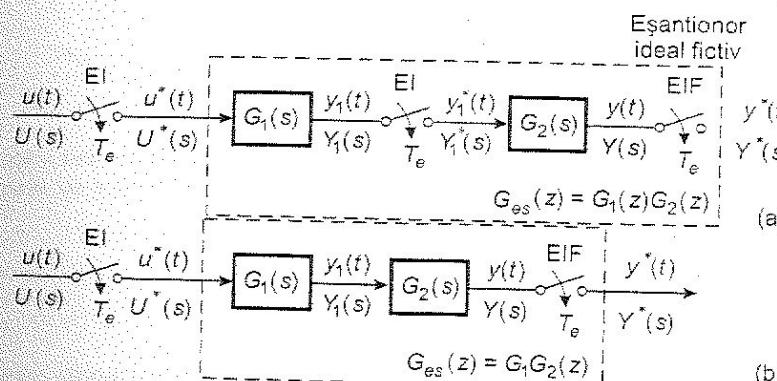


Fig. 9.5.1 Conexiunea serie

Într-o primă etapă, relațiile algebrice dintre semnale sunt:

$$Y(s) = G_2(s)Y_1^*(s) \text{ și } Y_1(s) = G_1(s)U^*(s) \quad (9.4)$$

Aplicând operatorul de eşantionare acestor relații se va obține în continuare:

$$Y^*(s) = G_2^*(s)Y_1^*(s) \text{ și } Y_1^*(s) = G_1^*(s)U^*(s), \quad (9.5)$$

respectiv:

$$Y^*(s) = G_2^*(s)G_1^*(s)U^*(s), \quad (9.6)$$

de unde, având în vedere definiția transformatei Z, rezultă:

$$G_{es}(z) = \left. \frac{Y^*(s)}{U^*(s)} \right|_{s=\frac{1}{T_e} \ln z} = G_1^*(s)G_2^*(s) \Big|_{s=\frac{1}{T_e} \ln z} = G_1(z)G_2(z).$$

În concluzie, funcția de transfer z echivalentă a două elemente continue conectate în serie cu eşantionor ideal între ele este:

$$G_{es}(z) = G_1(z)G_2(z), \quad (9.7)$$

unde:

$$G_1(z) = Z\{G_1(s)\} \text{ și } G_2(z) = Z\{G_2(s)\}.$$

9.5.1.2. Elemente continue conectate direct în serie

Din schema bloc prezentată în figura 9.5.1b se poate scrie relația:

$$Y(s) = G_1(s)G_2(s)U^*(s), \quad (9.8)$$

care, după aplicarea operatorului de eşantionare, devine:

$$Y^*(s) = [G_1(s)G_2(s)]^* U^*(s). \quad (9.9)$$

În final, având în vedere definiția transformatei Z, rezultă:

$$G_{es}(z) = G_1G_2(z), \quad (9.10)$$

unde:

$$G_1G_2(z) = Z\{G_1(s)G_2(s)\}. \quad (9.11)$$

Se poate face următoarea remarcă importantă:

$$G_1(z)G_2(z) \neq G_1G_2(z), \quad (9.12)$$

respectiv:

$$Z\{G_1(s)\}Z\{G_2(s)\} \neq Z\{G_1(s)G_2(s)\}. \quad (9.13)$$

9.5.2. Proprietățile de bază ale operatorului de eşantionare

$$1) [U_1(s) + U_2(s)]^* = U_1^*(s) + U_2^*(s), \quad (9.14)$$

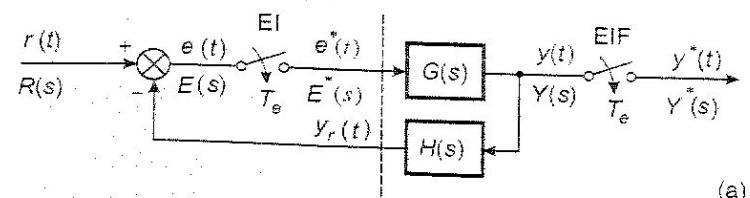
$$2) (U^*(s))^* = U^*(s), \quad (9.15)$$

$$3) [G(s)U^*(s)]^* = G^*(s)U^*(s), \quad (9.16)$$

$$4) [G(s)H(s)]^* \neq G^*(s)H^*(s). \quad (9.17)$$

9.5.3. Conexiunea cu reacție

Se consideră sistemul discret în circuit închis cu eşantionarea erorii, prezentat în figura 9.5.2a. Se poate remarcă faptul că schema cu eşantionare a erorii din figura 9.5.2a este echivalentă funcțională în zona sumatorului cu schema cu două eşantionare ideale plasate ca în figura 9.5.2b.



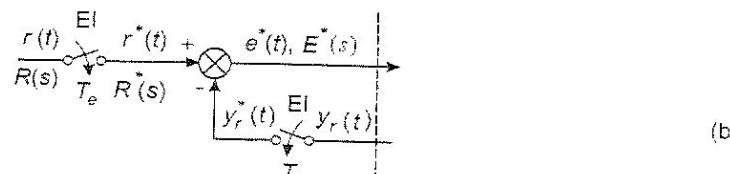


Fig. 9.5.2 Conexiunea cu reacție

Pentru deducerea funcției de transfer echivalente a conexiunii cu reacție, se scriu mai întâi relațiile algebrice care coreleză semnalele din schema bloc:

$$Y(s) = G(s)E^*(s), \quad (9.18)$$

$$E(s) = R(s) - H(s)Y(s). \quad (9.19)$$

Substituind $Y(s)$ din relația (9.18) în expresia (9.19) se obține:

$$E(s) = R(s) - G(s)H(s)E^*(s), \quad (9.20)$$

care, după aplicarea operatorului de eșantionare, devine:

$$E^*(s) = R^*(s) - [G(s)H(s)E^*(s)]^*, \quad (9.21)$$

respectiv:

$$E^*(s) = R^*(s) - [G(s)H(s)]^*E^*(s) = R^*(s) - GH^*(s)E^*(s), \quad (9.22)$$

de unde se obține transformata Laplace eșantionată a erorii:

$$E^*(s) = \frac{R^*(s)}{1 + GH^*(s)}. \quad (9.23)$$

Substituția expresiei (9.23) în relația (9.18) furnizează transformata Laplace a semnalului de ieșire în raport cu semnalul eșantionat de intrare:

$$Y(s) = \frac{G(s)}{1 + GH^*(s)}R^*(s), \quad (9.24)$$

din care, după aplicarea operatorului de eșantionare, rezultă:

$$Y^*(s) = \frac{G^*(s)}{1 + GH^*(s)}R^*(s), \quad (9.25)$$

de unde, cu schimbarea de variabilă $s = \frac{1}{T_e} \ln z$, se obține:

$$Y(z) = \frac{G(z)}{1 + GH(z)}R(z). \quad (9.26)$$

În final, rezultă funcția de transfer z a sistemului cu reacție negativă neunitară, cu eșantionarea erorii, exprimată conform relației:

$$G_{er}(z) = \frac{Y(z)}{U(z)} = \frac{G(z)}{1 + GH(z)}. \quad (9.27)$$

În Matlab nu este posibilă determinarea într-un mod direct a funcției de transfer z pentru un sistem eșantionat fără ER0. Pentru rezolvarea problemei folosește un artificiu care va fi prezentat în continuare.

Se cunoaște că: $Z\{G_{ER0}G(s)\} = Z\left\{\frac{1-e^{-sT_e}}{s}G(s)\right\} = (1-z^{-1})Z\left\{\frac{G(s)}{s}\right\}$, și având vedere $Z\{G(s)\} = \frac{Z\{G_{ER0}(s)[sG(s)]\}}{1-z^{-1}}$, rezultă că funcția de transfer z a sistemului eșantionat deschis fără ER0 se poate calcula aplicând funcția `c2d` cu opțiunea „z” a funcției de transfer $sG(s)$ și apoi înmulțind rezultatul cu $z/(z-1)$.

9.6. Exerciții propuse

Exercițiu 9.6.1

Fie următoarea funcție de transfer:

$$G(z) = \frac{1}{z^2 + 0.25}.$$

Să se reprezinte polii și zerourile în planul „z”.

Exercițiu 9.6.2

Fie funcția de transfer de mai jos. Să se obțină funcția de transfer a sistemului discret ($T_e = 1/50$) și să se afișeze. Conversia în discret se va face cu elementul de reținere de ordin zero.

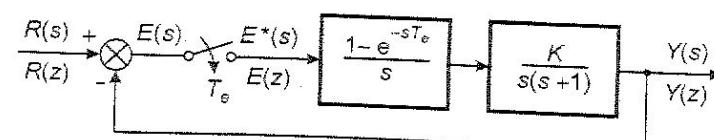
$$G(s) = \frac{k\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2},$$

unde: $k = 2$, $\omega_n = 5$, $\xi = 0.7$.

Pentru afișarea funcției de transfer a sistemului discret se va folosi funcția `printsys`.

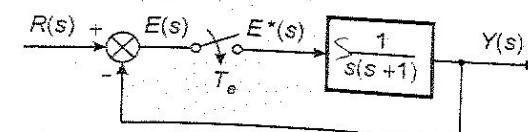
Exercițiu 9.6.3

Să se determine funcția de transfer echivalentă a sistemului din figură. Se presupune că $T_e = 0,3s$ și $K = 1$.



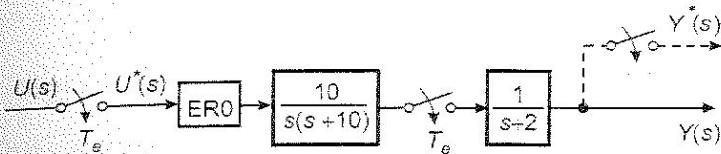
Exercițiu 9.6.4

Să se determine funcția de transfer echivalentă a sistemului din figura de mai jos. Se presupune că $T_e = 0,3s$.



Exercițiul 9.6.5

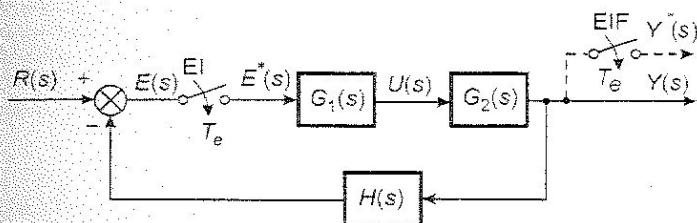
Să se determine funcția de transfer $G(z) = Y(z)/U(z)$ pentru sistemul deschis cu schema bloc din figură. Se presupune că $T_e = 1$ s.



Exercițiul 9.6.6

Să se determine funcția de transfer $G(z) = Y(z)/U(z)$ pentru sistemul deschis cu schema bloc din figură. Se presupune că: $T_e = 0,1$ s, $G_1(s) = \frac{1 - e^{-sT_e}}{s}$.

$$G_2(s) = \frac{1}{s+1}, H(s) = \frac{s}{s+2}.$$



ANALIZA ȘI SIMULAREA ÎN TIMP PENTRU SISTEMELE DISCRETE

OBIECTIVE

- Obținerea răspunsului sistemelor discrete de ordinele întâi și doi la intrările treaptă, rampă și impuls și a indicatorilor de calitate.
- Determinarea stabilității sistemelor cu ajutorul criteriilor Routh, Jury și Schur-Cohn.

10.1. Introducere

Se consideră un sistem discret general, descris cu ajutorul ecuației cu diferențe cu coeficienți constanti:

$$y[k] + a_1 y[k-1] + \dots + a_n y[k-n] = b_0 u[k] + b_1 u[k-1] + \dots + b_m u[k-m] \quad (10.1)$$

unde $k \in \mathbb{Z}$ sau \mathbb{N} , $n, m \in \mathbb{N}$ și $a_i, b_j \in \mathbb{R}$.

Ordinul maxim al eșantionului anterior care apare în ecuația cu diferențe este determinat de ordinul sistemului n , fiind $k-n$. Dacă sistemul liniar conține un timp mort pur (multiplul întreg al perioadei de eșantionare), de forma $\tau = \lambda T_e$, cu $\lambda \in \mathbb{N}$, ecuația cu diferențe se va scrie conform relației:

$$y[k] + a_1 y[k-1] + \dots + a_n y[k-n] = b_0 u[k-\lambda] + b_1 u[k-\lambda-1] + \dots + b_m u[k-\lambda-m] \quad (10.2)$$

Un alt tip de model pentru sistemele discrete se poate introduce, pornind de la ecuația cu diferențe, pe baza operatorului de întârziere, cu un pas q^{-1} . Funcția ratională de argument q :

$$G(q) = \frac{B(q)}{A(q)} = \frac{q^{-\lambda}(b_0 + b_1 q^{-1} + \dots + b_m q^{-m})}{1 + a_1 q^{-1} + \dots + a_n q^{-n}}, \quad (10.3)$$

se numește funcție de transfer operațională. Se remarcă faptul că argumentul q care apare în această funcție de transfer nu are caracterul unei variabile complexe, fiind doar o notație formală. Funcția de transfer z se poate deduce (metoda este folosită mai rar) din ecuația cu diferențe. Având în vedere teorema de deplasare în timp, dacă se aplică transformata Z ecuației cu diferențe (10.1), se obține:

$$(1 + a_1 z^{-1} + \dots + a_n z^{-n})Y(z) = z^{-\lambda}(b_0 + b_1 z^{-1} + \dots + b_m z^{-m})U(z), \quad (10.4)$$

de unde rezultă:

$$G(z) = \frac{Y(z)}{U(z)} = \frac{z^{-\lambda} (b_0 + b_1 z^{-1} + \dots + b_m z^{-m})}{1 + a_1 z^{-1} + \dots + a_n z^{-n}}. \quad (10.5)$$

10.2. Răspunsul sistemului discret de ordinul unu

Se consideră sistemul de ordinul unu continuu care are funcția de transfer standard: $G(s) = \frac{1}{sT + 1}$. Funcția de transfer z a ansamblului format dintr-un sistem de ordinul unu și un element de reținere de ordinul zero, este: $G(z) = \frac{b_0}{z - p_d}$, unde: $p_d = e^{-T_e/T}$ și $b_0 = 1 - p_d = 1 - e^{-T_e/T}$.

10.2.1. Răspunsul la impuls unitar

Răspunsul la impuls unitar se calculează cu relația:

$$y[k] = g[k] = Z^{-1} \left\{ \frac{b_0}{z - p_d} \right\} = r, \text{ unde reziduul } r \text{ se determină cu expresia:}$$

$$r = (z - p_d) \left. \frac{b_0 z^{k-1}}{z - p_d} \right|_{z=p_d} = b_0 p_d^{k-1}. \text{ Astfel, se obține:}$$

$$g[k] = g(kT_e) = b_0 p_d^{k-1} = (1 - p_d) p_d^{k-1}, k \geq 1. \quad (10.6)$$

Observație: Dacă $\text{grad } B(z) < \text{grad } A(z)$, unde $G(z) = \frac{B(z)}{A(z)}$, răspunsul sistemelor discrete este egal cu zero în decursul primei perioade de eşantionare (decî pentru $k = 0$). Dacă $\text{grad } B(z) = \text{grad } A(z)$ atunci este posibil ca răspunsul în timp al sistemului discret, pentru $k = 0$, să fie diferit de zero.

10.2.2. Răspunsul la treaptă unitară

Având în vedere că transformata Z a semnalului treaptă este:

$$R(z) = z / (z - 1),$$

rezultă că transformata Z a mărimi de ieșire va fi în acest caz:

$$Y(z) = G(z)R(z) = \frac{b_0 z}{(z - 1)(z - p_d)}.$$

Răspunsul indicial se determină cu metoda reziduurilor, acesta fiind:

$$y[k] = g_1[k] = Z^{-1} \left\{ \frac{b_0 z \cdot z^{k-1}}{(z - 1)(z - p_d)} \right\} = r_1 + r_2,$$

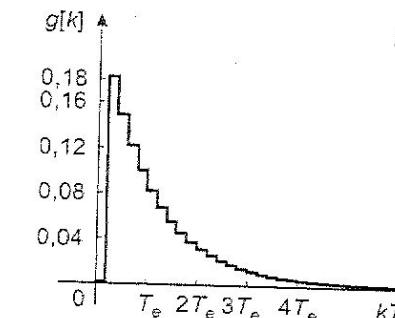
$$\text{unde: } r_1 = (z - 1)Y(z) \Big|_{z=1} = \frac{b_0 z^k}{z - p_d} \Big|_{z=1} = \frac{b_0 \cdot 1}{1 - p_d} = \frac{1 - p_d}{1 - p_d} = 1,$$

$$r_2 = \frac{b_0 z^k}{z - 1} \Big|_{z=p_d} = \frac{(1 - p_d)p_d^k}{p_d - 1} = -p_d^k,$$

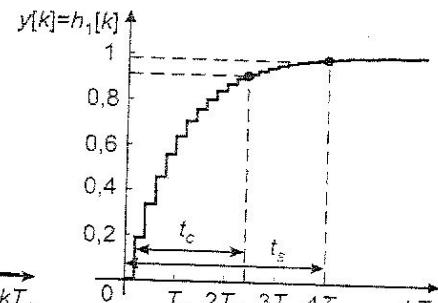
deci:

$$y[k] = g_1[k] = 1 - p_d^k = 1 - e^{-kT_e/T}, \quad (10.7)$$

în cadrul răspunsului (v. figura 10.2.1b) se pot pune în evidență cele două componente, staționară $y_p[k] = 1$, determinată de mărimea de intrare (treaptă unitară) și tranzitorie $y_t[k] = -p_d^k = -\exp[kT_e/T]$, determinată de sistemul, respectiv de polul funcției de transfer z . Din relația răspunsului se constată că $y[0] = 0$; $y[k] = 1$, pentru $k \rightarrow \infty$.



a)



b)

Fig. 10.2.1 Răspunsul la treaptă unitară

10.2.3. Răspunsul la rampă unitară

Transformata Z a semnalului rampă unitară este:

$$R(z) = \frac{T_e z}{(z - 1)^2}.$$

Transformata Z a ieșirii va fi în acest caz:

$$Y(z) = \frac{b_0 T_e z}{(z - 1)^2 (z - p_d)}.$$

Pentru calculul răspunsului în timp se va utiliza, de această dată, dezvoltarea în fracții simple a funcției complexe:

$$\frac{Y(z)}{z} = \frac{b_0 T_e}{(z - 1)^2 (z - p_d)} = \frac{c_1}{(z - 1)^2} + \frac{c_2}{z - 1} + \frac{c_3}{z - p_d}.$$

Coefficienții dezvoltării în fracții simple sunt:

$$c_1 = (z - 1)^2 \left. \frac{b_0 T_e}{(z - 1)^2 (z - p_d)} \right|_{z=1} = \frac{b_0 T_e}{1 - p_d},$$

$$c_2 = \frac{d}{dz} \left[\frac{b_0 T_e}{z - p_d} \right]_{z=1} = b_0 T_e \frac{-1}{(z - p_d)^2} \Big|_{z=1} = -\frac{b_0 T_e}{(1 - p_d)^2},$$

$$c_3 = (z - p_d) \frac{b_0 T_e}{(z - 1)^2(z - p_d)} \Big|_{z=p_d} = \frac{b_0 T_e}{(p_d - 1)^2},$$

deci dezvoltarea în fracții simple va fi:

$$Y(z) = \frac{1 - p_d}{1 - p_d} \left[\frac{z T_e}{(z - 1)^2} - \frac{1}{1 - p_d} \frac{z T_e}{z - 1} - \frac{1}{p_d - 1} \frac{z T_e}{z - p_d} \right],$$

de unde, după aplicarea transformatei Z inverse, rezultă:

$$y[k] = y(kT_e) = kT_e - \frac{T_e}{1 - p_d} + \frac{T_e}{1 - p_d} p_d^k = kT_e - \frac{T_e}{1 - p_d} (1 - e^{-kT_e/T_e}). \quad (10.8)$$

10.3. Răspunsul sistemului discret de ordinul doi

Pentru calculul analitic al răspunsului sistemului discret de ordinul doi se va folosi funcția de transfer z cu expresia:

$$G(z) = \frac{K(z - z_1)}{(z - p_1)(z - p_2)}, \quad (10.9)$$

unde $p_{1,2}$ sunt numere complexe conjugate. Polii discrete complex conjugati sunt corelați cu polii sistemului continuu de ordinul doi conform relației:

$$p_{1,2} = e^{-\zeta \omega_n T_e \pm j \omega_n \sqrt{1 - \zeta^2} T_e}, \quad (10.10)$$

unde ζ și ω_n au semnificații cunoscute, iar T_e este perioada de eşantionare.

Pentru a avea eroare stationară nulă la referință treaptă unitară și totodată pentru simplificarea relațiilor, factorul de amplificare se adoptă astfel încât $G(1) = 1$, de unde rezultă: $K = \frac{(1 - p_1)(1 - p_2)}{(1 - z_1)}$. Această funcție de transfer caracterizează un sistem de ordinul doi cu $0 < \zeta < 1$, cu element de reținere de ordin zero, când $z_1 \neq 0$, respectiv fără element de reținere de ordin zero când $z_1 = 0$. Răspunsul în timp se determină aplicând metoda reziduurilor, rezultând astfel:

$$y[k] = 1 + 2 \left| \frac{(p_1 - z_1)(p_1 - 1)}{(1 - z_1)(p_1 - p_2)} \right| |p_1|^k \cos(k\Phi + \theta), \quad (10.11)$$

unde $\Phi = \arg p_1 = \omega_n \sqrt{1 - \zeta^2} T_e$ și $\theta = \arg(p_1 - z_1) - \arg(p_1 - 1) - \frac{\pi}{2}$.

Expresia răspunsului în timp poate fi simplificată, înlocuind modulul și argumentul polului discret p_1 , obținându-se expresia:

$$y[kT_e] = 1 + e^{-\zeta \omega_n kT_e} |\sec \alpha| \cos(\omega_n \sqrt{1 - \zeta^2} kT_e + \alpha - \pi), \quad (10.12)$$

unde α poate fi pozitiv sau negativ, $\sec \alpha = 1/\cos \alpha$ și $\cos \alpha = \frac{1}{2} \frac{(1 - z_1)(p_1 - p_2)}{(p_1 - z_1)(1 - p_1)}$.

10.4. Performanțele sistemului de ordinul doi

Pentru evaluarea suprareglajului, în relația (10.9) se trece de la timpul discretizat la timpul continuu t . Astfel, se obține relația:

$$y^*(t) = 1 + e^{-\zeta \omega_n t} |\sec \alpha| \cos(\omega_n \sqrt{1 - \zeta^2} t + \alpha - \pi), \quad (10.13)$$

unde cu $y^*(t)$ s-a notat semnalul continuu care are valorile $y(kT_e)$ în momentele de eşantionare. Derivând expresia (10.13) în raport cu timpul și rezolvând ecuația obținută, se calculează timpul de vârf.

$$t_v^* = \frac{\frac{\pi}{2} - \alpha + \varphi}{\omega_n \sqrt{1 - \zeta^2}}, \text{ cu } \varphi = \arctg \frac{\sqrt{1 - \zeta^2}}{\zeta} \quad (10.14)$$

Înlocuindu-se t_v^* în relația (10.6) și avându-se în vedere că $y_{st}^* = 1$ și $\cos(\varphi - \frac{\pi}{2}) = \sin \varphi = \sqrt{1 - \zeta^2}$, va rezulta:

$$M_v^* = \sqrt{1 - \zeta^2} |\sec \alpha| \exp \left[-\frac{\zeta}{\sqrt{1 - \zeta^2}} \left(\frac{\pi}{2} - \alpha + \varphi \right) \right]. \quad (10.15)$$

Timpul de stabilire t_s^* pentru elementul de ordinul doi se determină analitic pornind de la relația de definiție a acestuia: $|y^*(t_s^*) - y_{st}^*| \leq 0.05 y_{st}^*$, în care se înlocuiește $y_{st}^* = 1$ și expresia răspunsului (10.10). În final rezultă expresia timpului de stabilire normalizat $\omega_n t_s^* = -\frac{\ln(0.05 \cdot |\sec \alpha|)}{\zeta}$. Timpul de stabilire poate fi aproximat acoperitor, în cazul unei benzi de toleranță de 5%, cu o relație similară cu cea folosită la sistemul continuu de ordinul doi, adică:

$$\omega_n t_s^* \approx \frac{4}{\zeta}. \quad (10.16)$$

În Matlab, funcțiile `dimpulse`, `dstep` și `dlsim` determină răspunsul unui sistem liniar discret la intrările impuls unitar, treaptă unitară, respectiv pentru o intrare oarecare. Dacă sistemul discret este prezentat printr-o funcție de transfer, atunci, $y=dimpulse(num, den, t)$, $y=dstep(num, den, t)$, $y=dlsim(num, den, u)$. Intervalul de timp este optional.

10.5. Stabilitatea sistemelor discrete

Stabilitatea sistemelor discrete cu reacție negativă poate fi evaluată, ca și la sistemele continue, pe baza poziției, în planul z, a polilor funcției de transfer a sistemului în circuit închis.

Funcția de transfer z a sistemului în circuit închis este :

$$G_0(z) = \frac{Y(z)}{R(z)} = \frac{G(z)}{1+GH(z)} = \frac{B(z)}{A(z)},$$

unde polinomul de la numitor are, într-un caz general, expresia:

$$A(z) = a_0 z^n + a_1 z^{n-1} + \dots + a_k z^{n-k} + \dots + a_{n-1} z + a_n. \quad (10.17)$$

De regulă $a_0 = 1$. Notând cu p_i rădăcinile polinomului $A(z)$, următoarea condiție de stabilitate a sistemului discret este valabilă: dacă toți polii p_i ai funcției de transfer $G_0(z)$ sunt amplasati în planul z în interiorul cercului de rază unitară, atunci toate componentele răspunsului tranzistorului ajung la starea de echilibru și prin urmare sistemul discret este stabil.

10.5.1. Criteriul de stabilitate Routh

Aplicarea directă a criteriului Routh funcțiilor de transfer z, în forma prezentată la sistemele continue, nu este posibilă, deoarece domeniul de stabilitate este, în cazul discret, interiorul cercului de rază unitară. Criteriul se poate totuși aplica, dacă se transformă, printr-o transformare conformă, interiorul cercului de rază unitară în semiplanul complex stâng.

Există două tipuri de transformări conforme (omografice):

- transformarea w

$$z = \frac{1+w}{1-w}, \quad (10.18)$$

- transformarea r

$$z = \frac{r+1}{r-1}. \quad (10.19)$$

În figura 10.5.1 se prezintă modul în care cele două schimbări de variabile transformă interiorul cercului de rază unitară din planul z în semiplanul complex stâng w, respectiv r.

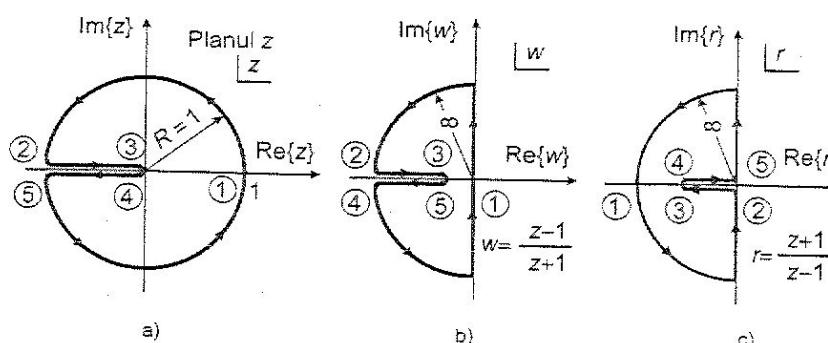


Fig. 10.5.1 Planurile z, w și r

Ambele transformări vor schimba ecuația caracteristică inițială, care este o funcție de z, într-un raport a două polinoame în r sau w de același ordin. Evident, noua ecuație caracteristică este polinomul de la numărător. Niciuna dintre cele două transformări nu are un avantaj respectiv dezavantaj față de celaltă.

Observație:

Înainte de aplicarea efectivă a criteriului Routh se verifică dacă toți coeficienții ecuației caracteristice în w sau r sunt pozitivi. Ca și în cazul sistemelor continue, dacă această condiție nu este îndeplinită, sistemul nu este stabil și nu mai este necesară verificarea cu criteriul Routh.

În Matlab, stabilitatea se determină tot cu funcția `routh` prezentată la sistemele continue, dar numai după ce s-a realizat schimbarea de variabilă.

10.5.2. Criteriul de stabilitate Schur-Cohn

Calculul analitic al rădăcinilor polinomului $A(z)$ este dificil pentru $n \geq 3$. Evaluarea stabilității în aceste situații se poate face cu anumite metode algebrice, care folosesc coeficienții polinomului caracteristic.

Pentru introducerea formei generale a criteriului Schur-Cohn, se consideră polinomul caracteristic al funcției de transfer a sistemului închis, scris conform expresiei (10.17), unde $a_0, a_1, a_2, \dots, a_n$ sunt coeficienți reali sau complecsi. Criteriul Schur-Cohn poate fi formulat astfel: un sistem discret este stabil dacă și numai dacă succesiunea „determinanților Schur-Cohn” $1, \Delta_1, \Delta_2, \dots, \Delta_n$ sunt năvăliri ale seminșului.

Determinantul Schur-Cohn k este definit prin:

$$\Delta_k = \begin{vmatrix} a_n & 0 & \dots & 0 & 0 & a_0 & a_1 & \dots & a_{k-1} \\ a_{n-1} & a_n & \dots & 0 & 0 & 0 & a_0 & \dots & a_{k-2} \\ \dots & \dots \\ a_{n-k+1} & a_{n-k+2} & \dots & a_{n-1} & a_n & 0 & 0 & \dots & a_0 \\ a_0 & 0 & \dots & 0 & 0 & a_n & a_{n-1} & \dots & a_{n-k+1} \\ a_1 & a_0 & \dots & 0 & 0 & a_n & \dots & a_{n-k+2} \\ \dots & \dots \\ a_{k-1} & a_{k-2} & \dots & a_1 & a_0 & 0 & 0 & \dots & a_n \end{vmatrix} \quad (10.20)$$

unde $k = 1, 2, 3, \dots, n$. Când toți coeficienții polinomului sunt reali, determinantul este simetric. Criteriul de stabilitate poate fi reformulat în modul următor: un sistem cu eșantionare este stabil atunci când toate rădăcinile ecuației lui caracteristice sunt situate în interiorul cercului unitate al planului z, iar coeficienții ecuației caracteristice satisfac următoarele condiții:

$$\Delta_k < 0 \text{ pentru } k \text{ impar}$$

$$\Delta_k > 0 \text{ pentru } k \text{ par}$$

Dacă nu sunt satisfăcute condițiile specificate în relația anterioară, atunci ecuația caracteristică are cel puțin o rădăcină situată în afara cercului unitate. Spre deosebire de criteriul Routh-Hurwitz, criteriul Schur-Cohn nu indică explicit că rădăcini sunt în afara cercului unitate și nici nu indică dacă vreuna din rădăcini este situată pe cercul unitate.

În Matlab, pentru determinarea stabilității unui sistem cu ajutorul criteriului Schur-Cohn se va folosi funcția `sschurcohn(p)`, care construiește matricea Schur-

Cohn a unui polinom discret p cu coeficienți reali, sau funcția `sschurcohn(p, q)`, care construiește matricea Schur-Cohn a unui polinom discret p cu coeficienți simbolici. Al doilea argument, q , indică variabila în care este specificat polinomul p . În acest caz, funcția `sschurcohn` este precedată de funcția `syms` care construiește obiecte simbolice.

Exemplu:

```
syms z a0 a1 a2 a3 a4 a5; sau syms a0 a1 a2 a3 a4 a5 z;
a0=1;a1=1;a2=1;a3=1;a4=1;a5=1;
a = a5 + a4*z + a3*z^2 + a2*z^3 + a1*z^4 + a0*z^5.
```

Rezultatul este următorul:

$$a = 1 + z + z^2 + z^3 + z^4 + z^5$$

10.5.3. Criteriul de stabilitate Jury

Criteriul de stabilitate Jury este mai simplu de aplicat decât criteriul Schur, fiind asemănător criteriului Hurwitz de la sistemele continue. Pentru introducerea criteriului se consideră polinomul caracteristic al funcției de transfer a sistemului închis, scris conform expresiei (10.17), cu $a_k \in \mathbb{R}$.

Cu ajutorul coeficienților se formează recurrent tabloul Jury conform schemei:

$$\begin{array}{cccccc} z^n & z^{n-1} & \dots & z^0 \\ a_0 & a_1 & \dots & a_n \\ a_n & a_{n-1} & \dots & a_0 \\ b_0 & b_1 & \dots & \\ b_{n-1} & b_{n-2} & \dots & \\ c_0 & c_1 & \dots & \\ c_{n-2} & c_{n-3} & \dots & \end{array}$$

unde:

(10.21)

$$\begin{aligned} b_0 &= a_0 - \frac{a_n}{a_0} a_n, \quad b_1 = a_1 - \frac{a_n}{a_0} a_{n-1}, \dots, \quad b_k = a_k - \frac{a_n}{a_0} a_{n-k}, \\ c_0 &= b_0 - \frac{b_{n-1}}{b_0} b_{n-1}, \dots, \quad c_k = b_k - \frac{b_{n-1}}{b_0} b_{n-1-k}. \end{aligned} \quad (10.22)$$

Sistemul cu polinomul caracteristic $A(z)$ este stabil dacă:

1) sunt verificate condițiile preliminare:
 $A(1) > 0$ și $A(-1) > 0$ pentru n par, respectiv $A(-1) < 0$ pentru n impar
 și

2) toți termenii din prima coloană, rândurile impare, sunt pozitivi:

$$a_0 > 0, \quad b_0 > 0, \quad c_0 > 0 \dots$$

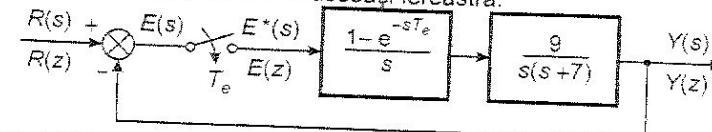
(10.24)

În Matlab, determinarea stabilității unui sistem cu ajutorul criteriului Jury se face folosind funcția `jury` cu următoarea sintaxă:
`jury(coef)`,
 unde `coef` reprezintă coeficienții polinomului caracteristic.

10.6. Exerciții propuse

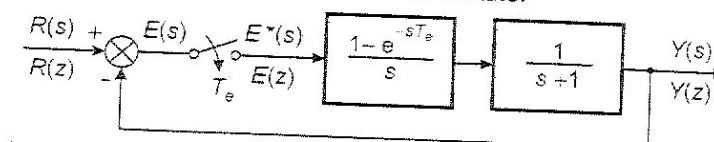
Exercițiu 10.6.1

Să se reprezinte răspunsul la mărimea treaptă unitară ale sistemelor continuu închis, discret cu ERO închis și discret fără ERO închis. Se consideră $T_e = 0,1s$. Cele trei răspunsuri se vor prezenta în aceeași fereastră.



Exercițiu 10.6.2

Să se reprezinte răspunsul la mărimele treaptă unitară, impuls unitar și rampă unitară ale sistemului continuu închis discret cu ERO (element de reținere de ordin zero). Se consideră $T_e = 0,3s$. Cele trei răspunsuri se vor prezenta în aceeași fereastră. Ulterior, să se determine indicatorii de calitate.



Exercițiu 10.6.3

Se consideră următoarea ecuație caracteristică a unui sistem discret:
 $z^3 + 3,5z^2 + 3,5z + 1 = 0$.

Cu ajutorul criteriului Routh-Hurwitz, iar apoi cu criteriul Schur-Cohn, să se determine stabilitatea sistemului discret.

Exercițiu 10.6.4

Cu ajutorul criteriului Schur-Cohn să se determine stabilitatea sistemului discret dat de următoarea ecuație caracteristică:

$$2z^2 + 3z - 2 = 0.$$

Exercițiu 10.6.5

Se consideră următoarele ecuații caracteristice ale unui sistem discret:

$$z^3 - 2,1z^2 + 1,6z - 0,4 = 0$$

$$z^3 - 1,1z^2 + 0,01z + 0,405 = 0$$

Cu ajutorul criteriului Jury, să se determine stabilitatea acestor sisteme.

ANALIZA ȘI SIMULAREA ÎN FRECVENTĂ PENTRU SISTEMELE DISCRETE

OBIECTIVE

- Răspunsul în frecvență la o mărime de intrare sinusoidală.
- Reprezentarea diagramelor Bode.
- Trasarea caracteristicii polare (Nyquist) de frecvență și determinarea stabilității.

11.1. Răspunsul în frecvență

Se consideră sistemul discret descris de: $Y(z) = G(z)U(z)$ și se presupune că la intrare se aplică o sinusoidă:

$$U(z) = Z[\sin(\omega t)] = \frac{z \sin \omega T_e}{z^2 - 2z \cos \omega T_e + 1},$$

$$Y(z) = \frac{G(z)z \sin \omega T_e}{(z - e^{j\omega T_e})(z - e^{-j\omega T_e})} = \frac{k_1 z}{z - e^{j\omega T_e}} + \frac{k_2 z}{z - e^{-j\omega T_e}} + G_g(t). \quad (11.1)$$

unde $G_g(z)$ sunt toate componentele lui $Y(z)$ care provin din polii lui $G(z)$. Dacă sistemul este stabil, aceste componente ale lui $y(kT_e)$ tind la zero atunci când timpul crește, iar răspunsul staționar al sistemului va fi:

$$Y_{st}(z) = \frac{k_1 z}{z - e^{j\omega T_e}} + \frac{k_2 z}{z - e^{-j\omega T_e}}. \quad (11.2)$$

Din (11.2) rezultă:

$$k_1 = \frac{G(z)(e^{j\omega T_e}) \sin \omega T_e}{e^{j\omega T_e} - e^{-j\omega T_e}} = \frac{G(e^{j\omega T_e})}{2j}. \quad (11.3)$$

Exprimând $G(e^{j\omega T_e})$ ca $G(e^{j\omega T_e}) = |G(e^{j\omega T_e})|e^{j\theta}$ se observă că (11.3) devine:

$$k_1 = \frac{|G(e^{j\omega T_e})|e^{j\theta}}{2j}. \quad (11.4)$$

Deoarece k_2 este conjugatul lui k_1 , atunci:

$$k_2 = -\frac{|G(e^{j\omega T_e})|e^{-j\theta}}{2j} \quad (11.5)$$

Așadar, din (11.3), (11.4) și (11.5) rezultă:

$$\begin{aligned} y_{st}(kT_e) &= k_1(e^{j\omega T_e})^k + k_2(e^{-j\omega T_e})^k = |G(e^{j\omega T_e})| \left[\frac{e^{j(\omega kT_e + \theta)} - e^{-j(\omega kT_e + \theta)}}{2j} \right] = \\ &= |G(e^{j\omega T_e})| \sin(\omega kT_e + \theta) \end{aligned} \quad (11.6)$$

Se vede că, dacă la intrarea sistemului discret se aplică o sinusoidă de pulsată ω , răspunsul în regim staționar este, de asemenea, sinusoidal și de aceeași pulsată. Amplitudinea răspunsului este egală cu amplitudinea intrării multiplicată cu $|G(e^{j\omega T_e})|$, iar faza răspunsului este egală cu faza intrării la care se adaugă $\arg G(e^{j\omega T_e})$. În concluzie, $G(e^{j\omega T_e})$ este răspunsul în frecvență la valorile de eșantionare.

Pentru reprezentarea grafică în Matlab a răspunsului în frecvență în domeniul discret se va folosi funcția freqz cu următoarea sintaxă:

$[h, \omega] = \text{freqz}(\text{num}, \text{den}, N)$ sau
 $h = \text{freqz}(\text{num}, \text{den}, \omega)$,

în cazul în care se cunoaște valoarea pulsăției). În relația anterioară:
 N este un număr întreg (dacă nu se specifică, are valoarea 512);
 num , den reprezintă numărătorul respectiv numitorul funcției de transfer;
 h este un vector care conține răspunsul în frecvență al punctului complex N ;
 ω reprezintă pulsata.

11.2. Diagrame Bode

Se păstrează aceleași reguli de trasare ca și la sistemele continue. În Matlab, pentru trasarea diagramelor bode în domeniul discret se vor folosi funcțiile:

$\text{dbode}(\text{numz}, \text{denz}, T_e)$, sau $\text{dbode}(\text{numz}, \text{denz}, T_e, \omega)$.

Dacă sistemul este reprezentat în spațiul stărilor (spațiul stărilor fiind reprezentat prin matricile Φ, Γ, C, D), atunci sintaxa este:

$\text{dbode}(\Phi, \Gamma, C, D, T_e, u)$ sau $\text{dbode}(\Phi, \Gamma, C, D, T_e, u, \omega)$.

Se vor trasa diagramele bode pentru o singură intare u ; dacă pulsata este omisă, va fi aleasă automat.

Cu ajutorul acestei funcții se pot determina amplitudinea, faza și pulsata, așa cum este prezentat în continuare:

$[\text{mag}, \text{phase}, \omega] = \text{dbode}(\text{numz}, \text{denz}, T_e)$,
sau

$[\text{mag}, \text{phase}] = \text{dbode}(\text{numz}, \text{denz}, T_e, \omega)$,
respectiv.

$[\text{mag}, \text{phase}, \omega] = \text{dbode}(\Phi, \Gamma, C, D, T_e, u)$,
sau

$[\text{mag}, \text{phase}] = \text{dbode}(\Phi, \Gamma, C, D, T_e, u, \omega)$.

11.3. Criteriul Nyquist de stabilitate

În Matlab, diagrama nyquist în planul z se trasează cu funcția:
 $\text{dnyquist}(\text{numz}, \text{denz}, T_e, \omega)$,

unde:

numz și denz reprezintă numărătorul, respectiv numitorul funcției de transfer;
 T_e este perioada de eșantionare;
 ω reprezintă pulsata.

Dacă sistemul este reprezentat în spațiul stărilor (spațiul stărilor fiind reprezentat prin matricile Φ, Γ, C, D), atunci sintaxa este:

$\text{dnyquist}(\Phi, \Gamma, C, D, T_e, u, \omega)$.

Se va trasa diagrama nyquist pentru o singură intare u . Pulsata se poate omite în ambele cazuri, fiind aleasă automat.

Funcția dnyquist poate determina răspunsul în frecvență sub forma a două matrice, re și im , care au tot atâta coloane câte componente are vectorul de ieșire y și același număr de linii:

$[re, im, \omega] = \text{dnyquist}(\text{numz}, \text{denz}, T_e)$,

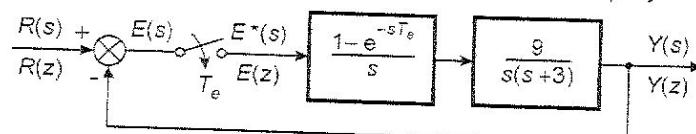
sau

$[re, im, \omega] = \text{dnyquist}(\Phi, \Gamma, C, D, T_e, u)$.

11.4. Exerciții propuse

Exercițiu 11.4.1

Să se reprezinte răspunsul în frecvență al sistemului continuu închis (cu $\omega = [0:4]$ și cu pasul 0,01; reprezentarea grafică se va face în coordonatele pulsăției și magnitudinii (amplitudinii)) și răspunsul în frecvență al sistemului discret închis ($T_e = 0,1s$; reprezentarea grafică se va face în coordonatele pulsăției și magnitudinii)).



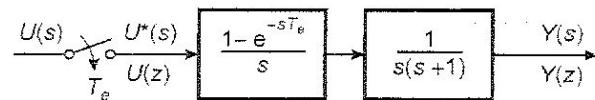
Exercițiu 11.4.2

Să se traseze caracteristicile logaritmice de frecvență ale modulului și fazei atât pentru sistemul continuu cât și pentru sistemul discret cu ERO. Perioada de eșantionare este $T_e = 0,1s$. Caracteristicile logaritmice se vor trasa folosind funcția plot. Pentru determinarea pulsăției se va folosi funcția logspace pe intervalul $[-1:2]$.

$$G(s) = \frac{s+1}{0.1s+1}$$

Exercițiu 11.4.3

Să se determine stabilitatea în domeniul frecvenței atât a sistemului continu cât și a sistemului discret ($T_e = 0,1s$). Pentru determinarea pulsării se va folosi funcția `logspace` pe intervalul $[-1:2]$.



LOCUL RĂDĂCINILOR ȘI DESCRIEREA ÎN SPAȚIUL STĂRILOR PENTRU SISTEMELE DISCRETE

OBIECTIVE

- Trasarea locului rădăcinilor pentru sisteme discrete.
- Răspunsul sistemelor în spațiul stărilor.
- Controlabilitatea și observabilitatea sistemelor discrete.

12.1. Locul rădăcinilor

Pentru sisteme complexe, cu mai multe singularități (poli și zerouri), trasarea locului rădăcinilor este aparent complicată, dar, dacă se aplică sistematic o serie de reguli, construcția grafică a locului rădăcinilor se simplifică mult. Astfel, prin amplasarea punctelor particulare și a asymptotelor, după evaluarea unghiurilor de plecare (sosire) ale ramurilor locului din polii complecși (în zerourile complexe) se poate schița fără dificultate forma de variație a locului rădăcinilor pentru un caz dat. Avantajele cele mai însemnante ale metodei locului rădăcinilor apar în cazul sistemelor de ordin ridicat, pentru care, găsirea polilor funcției de transfer în circuit închis cu alte metode este extrem de laborioasă.

Avgând în vedere că trasarea locului rădăcinilor are același regim de construcție ca și în cazul sistemelor continue, în continuare se vor prezenta, în rezumat, fără prea multe justificări, regulile generale de trasare a locului rădăcinilor pentru un sistem cu eșantionare închis, cu reacție negativă neunitară.

Pasul 1. Se determină ecuația caracteristică

$$1 + GH(z) = 0,$$

care se aranjează astfel încât să se pună în evidență parametrul de interes, ca un factor multiplicativ, conform expresiei:

$$1 + \frac{KB(z)}{A(z)} = 1 + \frac{K(z - z_1)(z - z_2) \cdots (z - z_m)}{(z - p_1)(z - p_2) \cdots (z - p_n)} = 0.$$

În discuția de fată, se va presupune că parametrul considerat este factorul de amplificare K , unde $K > 0$. Dacă se acceptă valori negative ale factorului de amplificare, caz în care reacția este pozitivă, va fi necesar să se modifice condiția fazei (a argumentului).

Pasul 2. Se determină numărul ramurilor locului rădăcinilor care este egal cu n (numărul polilor funcției de transfer a sistemului deschis). Se plasează, în planul

complex, polii și zerourile funcției $GH(z)$. Polii se marchează cu un \times , iar zerourile cu un o . Locul rădăcinilor începe în polii sistemului deschis și se termină în zerouri (cu valoare finită sau infinit).

Pasul 3. La stânga unui număr impar de poli reali plus zerouri reale, se marchează locul rădăcinilor de pe axa reală.

Pasul 4. Se desenează asimptotele ramurilor locului rădăcinilor.

- numărul asimptotelor este egal cu $n - m$;
- unghiurile asimptotelor:

$$\phi_k = \frac{\pm 180^\circ (2k+1)}{n-m}, \quad (k = 0, 1, 2, \dots); \quad (12.1)$$

- abscisa punctului de intersecție a asimptotelor:

$$\sigma_a = \frac{\sum p_i - \sum z_j}{n-m}. \quad (12.2)$$

Pasul 5. Se găsesc punctele de ramificație (de sosire și de plecare). În acest scop se scrie ecuația caracteristică conform expresiei:

$$1 + \frac{KB(z)}{A(z)} = 0, \text{ sau } K = -\frac{A(z)}{B(z)}. \quad (12.3)$$

Punctele de ramificație ale locului rădăcinilor se determină prin rezolvarea ecuației de mai jos și vor fi numai acele rădăcini reale care se află pe porțiunea considerată a locului rădăcinilor de pe axa reală:

$$\frac{dK}{dz} = -\frac{B'(z)A(z) - B(z)A'(z)}{B^2(z)} = 0. \quad (12.4)$$

Pasul 6. Se determină unghiul de plecare (unghiul de sosire) al locului rădăcinilor dintr-un pol complex (intr-un zero complex).

- unghiul de plecare dintr-un pol complex = $\varphi_p = 180^\circ - \sum \theta_i + \sum \Phi_j$,
- unghiul de sosire într-un zero complex = $\varphi_s = 180^\circ - \sum \Phi_j + \sum \theta_i$,

unde Φ sunt unghiurile vectorilor care încep în zerouri, iar θ sunt unghiurile vectorilor complezi care pleacă din poli.

Pasul 7. Se determină punctele în care locul rădăcinilor intersectează cercul de rază unitară. Aceste puncte se determină folosindu-se criteriul de stabilitate Routh-Hurwitz modificat, sau rezolvând pentru ω și K ecuația complexă:

$$1 + \frac{KB(e^{j\omega T_e})}{A(e^{j\omega T_e})} = 1 + \frac{K(e^{j\omega T_e} - z_1)(e^{j\omega T_e} - z_2) \dots (e^{j\omega T_e} - z_m)}{(e^{j\omega T_e} - p_1)(e^{j\omega T_e} - p_2) \dots (e^{j\omega T_e} - p_n)} = 0. \quad (12.5)$$

Pasul 8. Se gradează, dacă este necesar, locul rădăcinilor. Valoarea lui K corespunzătoare fiecărui punct de pe locul rădăcinilor din planul z se determină cu relația:

$$K = \frac{1}{|GH(z_1)|} = \left| \frac{A(z_1)}{B(z_1)} \right|. \quad (12.6)$$

Pentru trasarea locului rădăcinilor se vor folosi funcțiile **rlocus** și **zgrid**.

Cu ajutorul funcției $[k, \text{poli}] = \text{rlocfind}(\text{num}, \text{den})$, se pot determina factorul de amplificare și polii pentru un anumit punct.

12.2. Descrierea intrare - stare - ieșire

Așa cum s-a putut observa la sistemele continue, există și pentru sistemele discrete reprezentarea cu ajutorul variabilelor de stare, așa cum se arată în figura 12.2.1:

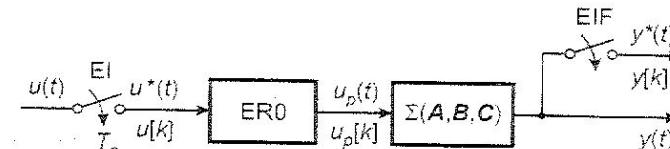


Fig. 12.2.1 Descrierea intrare-stare-iesire

unde EIF constituie eșantionorul ideal, EIF este un eșantionor ideal fictiv, iar ER0 este elementul de reținere de ordinul zero; vectorul de intrare este u , y fiind vectorul ieșirilor și x este vectorul stării.

Mulțimea valorilor vectorului intrărilor u este o funcție discretă în timp, notată cu $u(k)$ și este numită spațiu intrărilor sistemului. În același mod se definesc spațiul stărilor și spațiul ieșirilor sistemului.

Modelul cu variabile de stare al sistemului discret este:

$$\begin{aligned} x[k+1] &= \Phi x[k] + \Gamma u[k], \\ y[k] &= Cx[k]. \end{aligned} \quad (12.7)$$

Observație: În modelul matriceal se presupune, așa cum se întâmplă de obicei în practică, că matricea de transfer direct D este nulă.

Ecuatiile de mai sus sunt ecuațiile matriceale de stare pentru un sistem discret liniar, invariant în timp și se folosesc ca punct de plecare în analiza și sinteza modernă ale sistemelor discrete. Pe de altă parte se poate face, ca și la sistemele continue, o conexiune între descrierea în variabile de stare și metoda funcțiilor de transfer z . Se aplică transformata z în condiții inițiale nule. În final se obține:

$$G(z) = \frac{Y(z)}{U(z)} = C(zI - \Phi)^{-1}\Gamma, \quad (12.8)$$

care este tocmai funcția de transfer z a sistemului discret modelat la stare.

În cazul sistemelor discrete, funcția pentru simularea numerică a răspunsului unui sistem în reprezentarea variabilei la stare este:

$$[y, x] = \text{dsim}(\Phi, \Gamma, C, D, u, x_0),$$

care simulează răspunsul sistemului la o intrare oarecare (x_0 = condiție initială). În cazul intrărilor impuls Dirac și treaptă, funcțiile sunt:

$$[y, x] = \text{dimpulse}(\Phi, \Gamma, C, D, u, t),$$

și respectiv

$$[y, x] = \text{dstep}(\Phi, \Gamma, C, D, u, t).$$

Intervalul de timp este optional.

Controlabilitatea și observabilitatea sistemului discret se determină în Matlab cu aceleași funcții ca și în cazul sistemului continuu.

OBSERVAȚIE: Funcția **dlsim** prezintă răspunsul sistemului discret începând de la $t=1$ și nu din origine.

12.3. Exerciții propuse

Exercițiu 12.3.1

Fie sistemul discret dat prin următoarele matrice:

$$\Phi = \begin{bmatrix} 0.5 & 1 & 2 \\ 0 & 0.5 & 1 \\ 0 & 0 & 0.5 \end{bmatrix}, \Gamma = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Să se reprezinte grafic răspunsul obținut prin aplicarea unui semnal de tip treaptă unitară pe fiecare canal de intrare (2 canale de intrare) cu $t = [0:10]$ și $t_1 = [1:10]$.

Exercițiu 12.3.2

Cunoscând matricele variabile în timp ale unui sistem continuu, să se determine observabilitatea și controlabilitatea sistemului discret. Convertirea în discret se va face cu ER0. Perioada de eşantionare este $T_e = 0.01$ s.

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0.18 & 2.67 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0.45 & 31.18 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 1.8 \\ 0 \\ 4.5 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

Exercițiu 12.3.3

Fie sistemul discret reprezentat în forma intrare - stare - ieșire de următoarele matrice:

$$\Phi = \begin{bmatrix} 0 & -0.25 \\ 1 & 1 \end{bmatrix}, \Gamma = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, C = \begin{bmatrix} 0 & 1 \end{bmatrix}, x_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Să se reprezinte grafic răspunsul sistemului la intrarea treaptă.

Exercițiu 12.3.4

Fie sistemul de reglare cu eşantionare cu reacție negativă unitară având funcția de transfer z în circuit deschis:

$$G(z) = \frac{0.368(z + 0.717)}{(z - 1)(z - 0.368)}$$

Să se traseze locul rădăcinilor și să se determine punctele de părăsire ale axei reale.

UTILIZAREA PROGRAMULUI SIMULINK

OBJECTIVE

- Prezentarea principalelor blocuri din Simulink.
- Crearea schemelor bloc în Simulink, atât pentru sisteme continue, cât și pentru sisteme discrete.
- Transferul de date între Matlab și Simulink.

13.1. Introducere

Programul Simulink este folosit pentru simularea sistemelor dinamice conceput ca o extensie a pachetului de programe Matlab. Utilizarea programului se face în două faze:

- definirea modelului;
- analiza modelului.

Pentru a putea utiliza programul trebuie mai întâi învățat cum se construiesc din punct de vedere al programului Simulink, un model și apoi cum se pot manipula blocurile din biblioteciile programului.

13.2. Realizarea unei sesiuni în Simulink

O sesiune tipică începe prin realizarea modelului unui sistem sau apelarea unui model creat anterior. Pentru a facilita definirea modelului se utilizează o clasă de ferestre numită **diagramă bloc**. În aceste ferestre modelele sunt create și desenate în principal cu ajutorul mouse-ului.

O mare parte din înșurșirea programului constă în familiarizarea cu regulile de manipulare a comportamentelor în cadrul acestor ferestre. Pentru realizarea unui model, blocurile programului sunt copiate în ferestre de lucru și conectate după regulile care vor fi analizate în continuare. Evoluția simulării poate fi urmărită pe parcursul rulării experimentului de simulare, iar rezultatele finale pot fi găsite în spațiul de lucru al Matlab-ului, atunci când simularea este completă.

Pasul 1. Lansarea programului.

Lansarea programului de simulare se face în două moduri:

- prin tastarea în linia de comandă a cuvântului **simulink**;
- prin selectarea Library Browser din Start/Simulink.

Cele două moduri au ca efect deschiderea unei ferestre ce conține bibliotecile cu blocurile de bază reprezentate prin nume și icon. În această fereastră se selectază, din meniu File, comanda New pentru a deschide o fereastră activă goală în care se construiește modelul sistemului ce urmează să fie analizat, sau se selectază Open pentru a deschide o fereastră cu un model editat anterior. Fereastra poate fi redimensionată pentru a realiza o rezoluție convenabilă folosind mouse-ul cu ajutorul căruia se trag marginile în poziția dorită. Se pot salva noile dimensiuni ale ferestrei cu comanda Save din meniu File. La următoarea încărcare a ferestrei, ea va fi deschisă cu noile dimensiuni stabilite.

Pasul 2. Plasarea blocurilor în fereastra activă.

Pentru construirea modelului se vor deschide una sau mai multe biblioteci și blocurile necesare se vor muta cu ajutorul mouse-ului în fereastra activă. Deschiderea unei biblioteci se face prin realizarea unui dublu click pe imaginea acesteia.

Pasul 3. Interconectarea blocurilor.

După ce blocurile au fost plasate, se realizează conectarea lor prin trasarea unor linii de conectare cu ajutorul mouse-ului, mutând pointer-ul acestuia și ținând apăsat butonul din stânga de la ieșirea unor blocuri la intrarea altora.

Pasul 4. Setarea parametrilor blocurilor.

Se deschid pe rând blocurile printr-un dublu click pe imaginea lor și se setează parametrii care apar în caseta de dialog. Parametrii unui bloc pot consta din orice expresie Matlab corectă din punct de vedere sintactic. Un exemplu de casetă de dialog este prezentată în figura 13.2.1

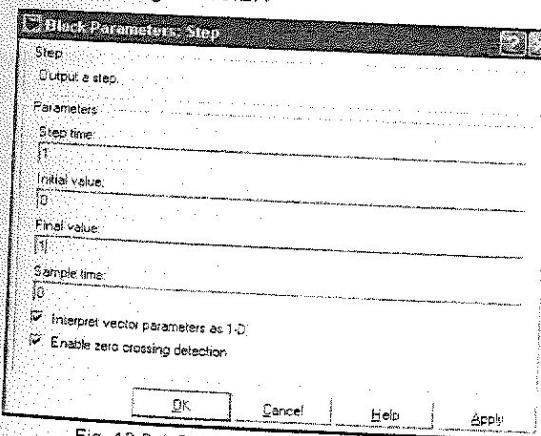


Fig. 13.2.1 Casetă de dialog a blocului step

Această casetă corespunde blocului step (mărime de intrare a unui sistem) și setarea constă în introducerile pasului (reprezintă momentul de timp în secunde când semnalul de ieșire sare de la valoarea inițială la valoarea finală), a valorii inițiale (reprezintă momentul de început al semnalului de ieșire), a valorii finale (reprezintă momentul de sfârșit al semnalului de ieșire, amplitudinea semnalului) și a perioadei de eșantionare (în cazul sistemelor discrete).

Pasul 5. Setarea parametrilor simulării.

Parametrii simulării se setează selectând comanda Configuration Parameters din meniu Simulation. Pentru rularea unei simulări pe PC, este nevoie de o tehnică numerică de rezolvare a ecuațiilor diferențiale (ex ode45). Sistemul poate fi simulațat ca un sistem continuu sau ca un sistem discret, în funcție de blocurile ce formează sistemul. Prin setarea parametrilor se înțeleg: alegerea metodei de integrare, stabilirea limitelor pentru variabila independentă (timpul), stabilirea limitelor pentru funcție și stabilirea pașilor minim și maxim de integrare.

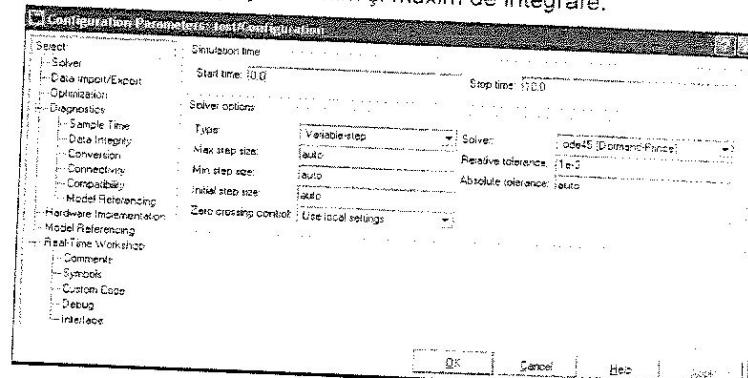


Fig. 13.2.2 Casetă de dialog pentru setarea parametrilor simulării

Pasul 6. Simularea.

Pornirea simulatorului se poate face selectând comanda Start din meniu Simulation. În timpul simulării, în locul comenzii Start apare comanda Stop, prin intermediul căreia se poate opri simularea. Este recomandat ca înainte de simulare să se salveze modelul prin alegerea comenzii Save din meniu File.

Pentru a vizualiza rezultatelor simulării, se utilizează blocul Scope, care poate fi conectat în locurile în care se dorește, prin realizarea unei conexiuni spre intrarea acestuia din punctul dorit. Blocul Scope conține o serie de butoane care au următoarea semnificație:

- tipărire;
- setare parametri;
- zoom pe axe x și y;
- zoom pe axa x;
- zoom pe axa y;
- scalare automată;
- salvare setare axe;
- întoarce vechile setări;
- Scope variabil;
- deblocare setare axe;
- selectare semnal.

Scalarea automată realizează un zooming pe ambele axe. Apăsând butonul setare parametri (parameters) se va deschide fereastra din figura 13.2.3:

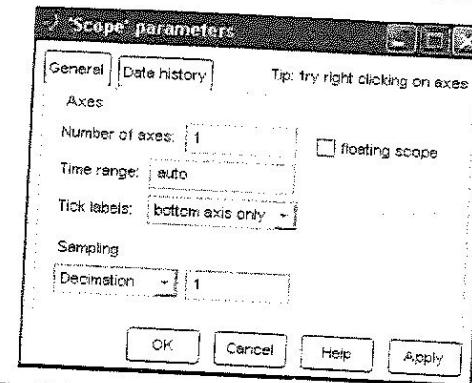


Fig. 13.2.3 Caseta de dialog pentru setarea parametrilor

unde:

- **Number of axes** (numărul de axe) permite selectarea numărului de axe y folosite (de exemplu dacă numărul de axe este 1, atunci pe același grafic se pot reprezenta două sau mai multe semnale, iar dacă numărul de axe este 2 (sau 3, etc), atunci fiecare semnal va fi reprezentat pe un grafic separat), axa x rămânând neschimbată pentru fiecare grafic; dimensiunile axei y pot varia de la un grafic la altul; numărul de axe este egal cu numărul porturilor de intare;
- **Time range** (interval temporal) permite fixarea limitelor axei x prin introducerea unui număr (număr de secunde) sau a cuvântului auto; prin introducerea numărului de secunde, se obțin limitele axei x, iar introducând auto, limitele axei x sunt egale cu durata simulării;
- **Tick labels** permite afișarea axelor graficului, sau omisiunea acestora;
- **Floating Scope** permite transformarea blocului **Scope** într-un osciloscop dual sau, mai simplu, se copiază blocul **Floating Scope** din biblioteca **Sinks** în fereastra activă; blocul **Floating Scope** are parametrul floating selectat automat; pentru a folosi un **Floating Scope** în timpul selectare a semnalului (signal selection); se va deschide o a doua în blocul **Floating Scope**; se pot alege orice semnale, inclusiv cele din cadrul subsistemelor care nu sunt deschise;
- **Sampling** reprezintă un factor de decimare; dacă se dorește o reprezentare sub formă de perioade de eșantionare se va alege opțiunea **sampling time**.

Este posibil controlul „cantității” de informație (numărului de puncte) care să apară în blocul **Scope** prin setarea opțiunii **Data History**. Se poate limita numărul de puncte salvate prin selectarea opțiunii **Limit data points to last**. Datele obținute în urma simulării pot fi salvate automat prin selectarea opțiunii **Save data to workspace**. Se va introduce un nume în câmpul **Variable name**, specificându-se și formatul sub care se salvează datele respective, selectând una din opțiunile din câmpul **Format**.

Pentru selectarea limitelor axei y, se va da click dreapta pe grafic și se va alege opțiunea **Axes properties**. Pe ecran va apărea fereastra din figura 13.2.4:

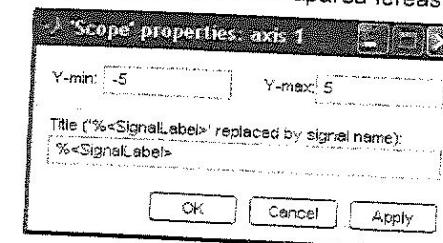


Fig. 13.2.4 Setarea proprietăților axei y

În Y-min se va scrie valoarea minimă pentru axa y, în Y-max se va scrie valoarea maximă pentru axa y, iar în Title se va preciza titlul graficului. Se poate utiliza blocul **To Workspace** pentru a trimite rezultatele în mediul Matlab unde pot fi prelucrate utilizând funcțiile Matlab.

În cazul apariției unor erori, acestea trebuie eliminate, schimbând una sau mai multe conexiuni, stergând blocuri sau schimbând parametri ai unor blocuri. Comenzile pentru stergere sunt comune aplicațiilor Windows. Elementul care urmează să fie sters va fi selectat și apoi, cu tasta **Del**, va putea fi sters.

Se pot utiliza modificări ale traseelor legăturilor cu ajutorul mouse-ului. De asemenea, în cazul în care modelul este prea mare (contine prea multe blocuri), se poate realiza o grupare a acestora într-un singur bloc prin comanda **Block**.

Orientarea unui bloc se poate modifica din meniu **Format**. Dimensiunea unui bloc se poate schimba, ca la orice obiect grafic Windows, prin selectarea cu mouse-ului și apoi redimensionare. Blocurile pot fi mutate în orice loc în fereastră. Orice linie care este conectată la un bloc care se mută rămâne conectată.

Programul **Simulink** realizează adăugarea unui nume pentru fiecare bloc utilizat. Dacă mai multe blocuri sunt de același fel, atunci programul adaugă un număr la numele blocului. Utilizatorul poate schimba numele blocului pentru a realiza o documentare cât mai bună, cu condiția respectării regulii de mai sus.

Ca și exemplu, se va încerca în continuare, reprezentarea funcției sinus. Din fereastra ce conține bibliotecile cu blocurile de bază se vor alege blocurile:

• Sine wave și Scope

În fereastra activă se va construi următorul model:

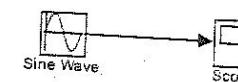


Fig. 13.2.5 Model pentru simularea funcției sinus

După terminarea simulării, prin dublu click pe **Scope** se poate vizualiza rezultatul.

13.3. Transfer de date între Simulink și Matlab

Pentru a realiza transferul de date către programul Matlab, se adaugă la programul din Simulink, două blocuri **To Workspace**, din biblioteca **Sinks**. În

exemplul următor (figura 13.3.1), se va prezenta un sistem cu buclă închisă, realizând transferul de date între Matlab și Simulink. Se vor folosi:

- blocul **Sum** din biblioteca **Math**;
- două blocuri **To Workspace** din biblioteca **Sinks**.

Ieșirea din blocul **Step**, notată cu u , va fi conectată la intrarea unui bloc **To Workspace**. Iar ieșirea din blocul **Scope**, notată cu y , va fi conectată la celălalt bloc **To Workspace**; în cele două blocuri **To Workspace** se vor modifica următoarele câmpuri:

- Variabile **name=Out** se va schimba cu u pentru primul bloc, respectiv cu y pentru celălalt bloc;
- Maximum number of row= inf (numărul real de elemente va fi decis în urma simulației)
- Save format=Structure se modifică în Array.
- Parametrii decimation și sample time pot fi ignorati.

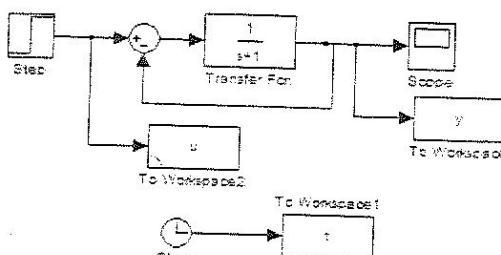


Fig. 13.3.1 Transfer de date către Matlab

Transferul de date către Simulink se realizează introducându-se, de exemplu, număratorul num și numitorul den ale funcției de transfer în mediul Matlab. În linia de comandă Matlab, se vor seta următoarele valori $\text{num}=1$ și $\text{den}=[1 \ 1]$. Se va observa că valorile u și y au fost transferate în Matlab - Workspace (figura 13.3.2).

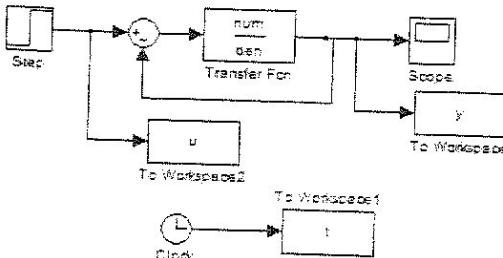


Fig. 13.3.2 Transfer de date către Simulink

Variabila de timp t (timpul de simulare), este transmisă către Matlab folosind blocul **Clock** (ceas) din biblioteca **Source** și blocul **To Workspace**. Utilizând funcția

`plot(t, y)` în linia de comandă a Matlab-ului se va obține o figură asemănătoare cu figura din fereastra grafică Simulink.

13.4. Exerciții propuse

Exercițiul 13.4.1

Să se calculeze valoarea polinomului $f(x)=x^2+2x+3$ în punctul 2. În Simulink, se vor alege:

- din biblioteca **Math Operations**, blocul **Polynomial**;
- din biblioteca **Sources**, blocul **Constant**;
- din biblioteca **Sinks**, blocul **Display** (pentru a vizualiza rezultatul).

Exercițiul 13.4.2

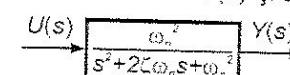
Fie funcțiile: $f(x)=\begin{cases} x^2, & x \geq 1 \\ x^3, & x < 1 \end{cases}$ și $g(x,y)=\begin{cases} (x+y)^2, & 0 \leq (x+y) \leq 2 \\ x^2+y^2, & \text{în restul cazurilor} \end{cases}$

Să se calculeze în Simulink $f(2)$, $f(-2)$, $g(1,2)$. Mai întâi, în mediul Matlab, se vor realiza un fișier **funcțief**, respectiv un fișier **funcțieg**, după care se vor alege din Simulink următoarele blocuri:

- din biblioteca **Sources**, blocul **Constant**;
- din biblioteca **User-Defined Functions**, blocul **Matlab Fcn**;
- pentru vizualizarea rezultatului, din biblioteca **Sinks**, se va alege blocul **Display**.

Exercițiul 13.4.3

Fie sistemul din figură, având o intrare $U(s)$ și o ieșire $Y(s)$.



Acest sistem are următorii parametri: $\omega_n = 5$, $\zeta = 0.6$. Să se reprezinte răspunsul la intrarea treaptă al sistemului de ordinul doi. Din Simulink se vor alege următoarele blocuri:

- din biblioteca **Sources**, blocul **Step**;
- din biblioteca **Continuous**, blocul **Transfer Fcn**;
- din biblioteca **Sinks**, blocul **Scope**.

Exercițiul 13.4.4

Fie sistemul discret reprezentat prin:

$$\Phi = \begin{bmatrix} 0.5 & 1 & 2 \\ 0 & 0.5 & 1 \\ 0 & 0 & 0.5 \end{bmatrix}, \quad \Gamma = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

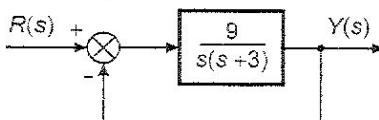
Să se reprezinte grafic răspunsul obținut prin aplicarea unui semnal de tip treaptă unitară pe fiecare canal de intrare ($t = [0:0.1:1]$, respectiv $t_1 = [0:1]$).

Din Simulink se vor alege următoarele blocuri:

- din biblioteca **Sources**, blocul **Step**;
- din biblioteca **Discrete**, blocul **Discrete State-Space**;
- din biblioteca **Sinks**, blocul **Scope**.

Exercițiu 13.4.5

Fie sistemul continuu din figură:



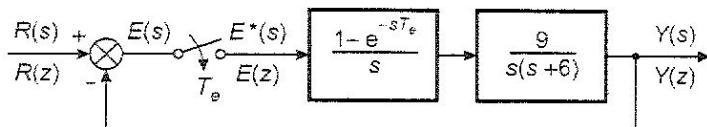
Să se simuleze răspunsul sistemului închis la intrarea treaptă unitară. Din Simulink se vor alege următoarele blocuri:

- din biblioteca **Sources**, blocul **Step**;
- din biblioteca **Continuous**, blocul **Transfer Fcn**;
- din biblioteca **Math Operations**, blocul **Sum**;
- din biblioteca **Sinks**, blocul **Scope**.

Să se transfere ulterior datele din Simulink în Matlab și să se realizeze simularea în Matlab.

Exercițiu 13.4.6

Fie sistemul discret din figură:



Să se simuleze răspunsul sistemului închis la intrarea treaptă unitară. Din Simulink se vor alege următoarele blocuri:

- din biblioteca **Sources**, blocul **Step**;
- din biblioteca **Discrete**, blocul **Discrete Transfer Fcn**;
- din biblioteca **Math Operations**, blocul **Sum**;
- din biblioteca **Sinks**, blocul **Scope**.

Observație:

Trecerea de la sistem continuu la sistem discret se va realiza în Matlab.

Exercițiu 13.4.7

Fie sistemul continuu reprezentat în spațiul stărilor:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u,$$

$$\begin{aligned} y &= [1 \quad 0] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + [0] u, \\ x(0) &= 0. \end{aligned}$$

Să se reprezinte grafic răspunsul sistemului obținut prin aplicarea unui semnal de tip rampă unitară la intrare. Din Simulink se vor alege următoarele blocuri:

- din biblioteca **Sources**, blocul **Ramp**;
- din biblioteca **Continuous**, blocul **State-Space**;
- din biblioteca **Sinks**, blocul **Scope**.

CREAREA DE SUBSISTEME ȘI S-FUNCTIONS

OBIECTIVE

- Crearea de subsisteme și măști.
- Scrierea funcțiilor S.

14.1. Crearea de subsisteme

Subsistemele reprezintă metoda prin care se pot grupa mai multe blocuri între ele, controlându-se astfel complexitatea modelului.

În Simulink există două clase de subsisteme:

- subsistemele virtuale permit o ierarhie grafică în modele (nu se împiedică execuția - Simulink-ul extinde sistemul înaintea execuției); această extindere este similară modului de lucru al macrourilor în C sau C++.
- subsistemele nevirtuale sunt executate ca un singur bloc; blocurile dintr-un subsistem nevirtual se execută numai atunci când toate intrările subsistemelor sunt valide.

De asemenea, în Simulink, s-au definit următoarele tipuri de subsisteme nevirtuale:

- subsistemele atomic reprezintă un sistem din interiorul altui sistem; diferența dintre subsistemele virtuale și subsistemele atomic, constă în faptul că blocurile din subsistemul atomic se execută ca un întreg (unitate); orice bloc Simulink poate fi plasat într-un subsistem atomic, inclusiv blocuri cu diferite rate de execuție; se poate crea un subsistem atomic prin selectarea opțiunii **Treat as atomic unit**;
- subsistemele enabled (subsisteme de activare) sunt subsisteme ce se execută la fiecare pas al simulării, atunci când la intrare există o valoare pozitivă a semnalului (de exemplu, dacă semnalul de intrare este un sinus, atunci subsistemul este alternativ activ sau inactiv); un subsistem enabled are o singură intrare ce poate fi un scalar sau un vector;
- subsistemele triggered sunt subsisteme ce se execută la fiecare eveniment ce apare la intrare; aceste subsisteme au o singură intrare numită „intrare trigger”, care determină execuția subsistemului.

Librăria folosită este **Ports & Subsystems**, iar descrierea blocului este:  Un subsistem se poate crea în două moduri, astfel:

- În fereastra activă, din biblioteca **Ports & Subsystems**, se alege subsistemul ce trebuie creat; cu dublu click pe subsistem se va deschide o

- În fereastra activă se pot introduce blocurile care fac parte din subsistem; după care, se crează subsistemul prin selecția opțiunii **Create Subsystem** din meniu **Edit** (în acest fel se ajunge la un sistem atomic - Simulink-ul va înlocui blocurile existente cu un singur bloc).

Numărul de porturi de intrare/ieșire din cadrul subsistemului este același cu numărul de intrări/ieșiri ale subsistemului. Dacă se dorește vizualizarea parametrilor **SubSystem Parameters**, deschizându-se fereastra din figura 14.1.1:

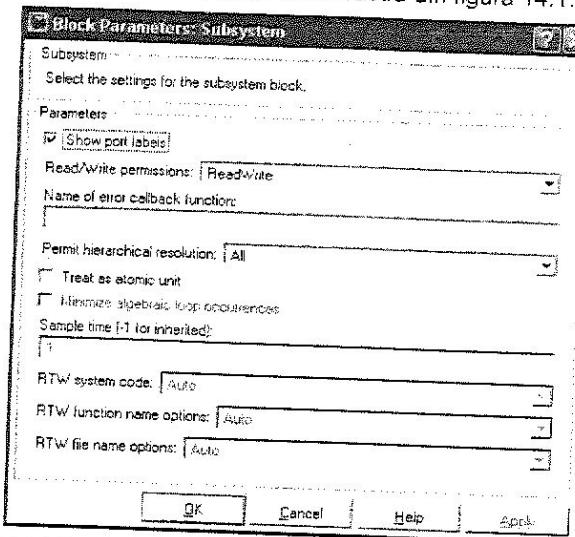


Fig. 14.1.1 Casetă de dialog pentru vizualizarea parametrilor subsistemului

unde:

Show port labels - expune eticheta de subsistem pe iconul subsistemului; **ReadWrite permissions** - utilizatorul poate avea acces la conținutul subsistemului, putându-se selecta următoarele opțiuni: *ReadWrite* (utilizatorul poate deschide subsistem și modifica conținutul acestuia), *ReadOnly* (utilizatorul poate deschide subsistem, dar nu poate modifica conținutul acestuia) sau *NoReadOrWrite*; **Name of error callback function** - se pot da nume erorilor care pot apărea atunci când se execută subsistem.

Parametrii ai căror nume încep cu RTW sunt folosiți de **Real Time Workshop** pentru generarea de cod.

Avantajul utilizării subsistemelor este acela că numărul de blocuri utilizate în fereastra activă este redus.

Exemplu 14.1

Cu ajutorul subsistemelor se vor reprezenta două funcții sinus de frecvențe diferite (1 rad/sec respectiv 2 rad/sec). În fereastra activă se aleg două subsisteme și un bloc **Scope** pentru vizualizarea celor două funcții sinus. Cele două subsisteme sunt legate la blocul **Scope** prin intermediul unui multiplexor.

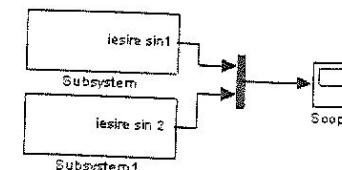


Fig. 14.1.2 Crearea modelului

Subsistemele conțin următoarele blocuri: un bloc sinus și un port de ieșire care coincide cu ieșirea subsistemului. Prin dublu click pe blocul de sinus se vor putea introduce parametrii funcției sinus.



Fig. 14.1.3 Blocurile componente din cadrul unui subsistem

După terminarea simulării, pentru vizualizarea rezultatelor se va da dublu click pe blocul **Scope**.

14.1.1. Crearea de măști

O mască reprezintă o interfață pentru un subsistem, ce ascunde componentele acestuia. Mască apare ca un bloc atomic, având proprii icon și parametri. Se pot crea o mască pentru orice subsistem și propriul icon pentru masca acestuia, icon ce îl va înlocui pe cel standard. Prin parametri măștii, Simulink-ul permite definirea unui set de parametri pentru subsistemul mascat; totodată acesta stochează valoarea parametrului cu numele specificat de către utilizator. Codul de initializare a măștii este un cod Matlab, pe care Simulink-ul îl rulează la începutul simulării pentru initializarea subsistemului mascat și a parametrilor.

Exemplu 14.2

Se consideră un subsistem ce trasează o linie de forma $y=mx+b$ (figura 14.1.4). În continuare se dorește mascarea acestui subsistem. Subsistemu este alcătuit din:

- un bloc **Gain** care conține parametrul m ;
- un bloc **Constant** care conține parametrul b .

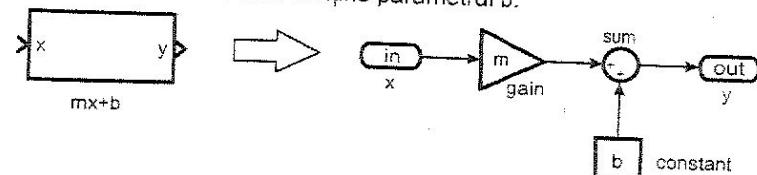


Fig. 14.1.4 Subsistemu mascare_exemplu

Se crează masca subsistemului: **Edit/Mask Subsystem**.

Se vor putea introduce din **Edit/Edit Mask** (vezi figura 14.1.5):

- numele parametrilor măștii: m și b ; valorile acestora se pot seta din **Edit/Mask Parameters** prin dublu click pe subsistem (pentru m se va introduce valoarea 3 iar pentru b , valoarea 2); în fereastra **callback**, se scrie un cod Matlab pe care Simulink-ul îl execută atunci când utilizatorul

- realizează o modificare asupra unui parametru (de exemplu, acest cod se va executa atunci când, pentru pantă, se va introduce o valoare negativă); icon-ul măștii: se pot asigna imagini, texte etc;
- initializarea măștii: în partea stângă vor apărea numele variabilelor asociate parametrilor (m și b), iar în partea dreaptă se introduc comenzi de initializare (comenzi Matlab formate din funcții Matlab, operatori sau variabile definite în mască, ca de exemplu, **dstop if error** (dacă o eroare apare în comenzi de initializare, atunci execuția este opriță pentru examinarea măștii) sau **keyboard** (execuția este opriță, iar controlul este redat tastaturii));
- documentația: permite definirea sau modificarea descrierii, a tipului și a help-ului pentru un bloc mascat (pentru câmpul **mask type** se poate introduce orice nume se dorește (ex. „subsistem”); în câmpul **mask description** se pot scrie date despre scopul și funcțiile blocului (ex. „m și b reprezintă parametrii blocului și se dorește trasarea unei linii”); în câmpul **mask help** se poate indica help-ul blocului, care va apărea atunci când se va apăsa butonul de help din **Block Parameters** (ex. „blocul trasează graficul lui y pentru diferite valori ale lui x”)).

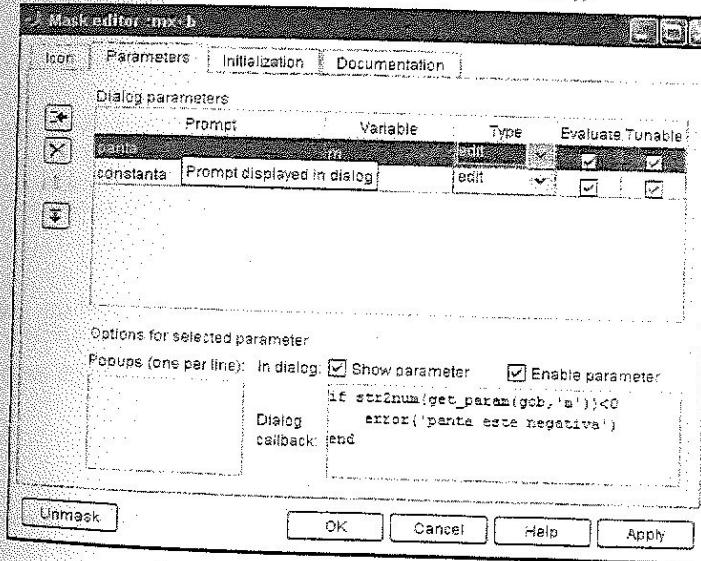
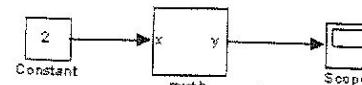


Fig. 14.1.5 Editor pentru masca subsistemului

Pentru reprezentarea grafică a funcției y , se conectează la ieșirea subsistemului mascat blocul **Scope**, iar la intrarea acestuia se conectează blocul **Constant**, atribuindu-se lui x valoarea 2.

Fig. 14.1.6 Graficul funcției $y=mx+b$

14.2. S-Functions

Atunci când se crează o funcție în Simulink, se generează de către sistem o funcție nouă numită S-Function. Această funcție definește dinamica sistemului și este utilizată de rutinele de integrare, liniarizare, etc. O S-function poate fi scrisă în Matlab sau în limbajul C, folosind o sintaxă specială care permite interacționarea cu Simulink-ul și poate fi folosită pentru descrierea sistemelor continue, discrete și hibride. Orice model Simulink poate fi descris ca o S-Function. Fiecare bloc dintr-un model Simulink are următoarele caracteristici: un vector de intrare u , un vector de ieșire y și un vector de stare. Vectorul de stare poate fi format din stări continue, discrete sau o combinație a celor două.

Parametrii de intrare ai unei S-Function sunt:

t, x, u, flag ,

iar parametrii de ieșire:

$[sys, x0, str, Te]$,

unde:

x - vector de stare;

u - vector de intrare;

flag - un parametru care controlează informația returnată;

$x0$ - condițiile inițiale;

str - parametru neimplementat și va fi inițializat cu o matrice goală;

Te - matrice cu două coloane, în prima coloană specificându-se perioada de eșantionare, iar în a doua compensarea făță de o mărime de referință (offset); de exemplu, în cazul în care se dorește ca o S-Function să ruleze la fiecare pas, atunci $Te=[0,0]$ (perioadă de eșantionare continuă); dacă rata cu care rulează o S-Function trebuie să fie identică cu cea a blocului la care este conectată, atunci $Te=[-1,0]$, iar dacă rularea este cerută la fiecare 0.25 secunde (perioadă de eșantionare discretă), începând de la 0.1 secunde de la startul simulării, atunci $Te=[0.25,0.1]$.

Din cele șase opțiuni disponibile ale indicatorului flag , trebuie tratată cel puțin una în cadrul unei S-Function. Acestea sunt prezentate în tabelul 14.2.1:

Tab. 14.2.1 Valorile indicatorului flag

Valoare parametru	Valoarea returnată de S-Function
0	Dimensiunile parametrilor și condițiile inițiale
1	Starea derivatorilor matricei de stare
2	Stările discrete $x[k+1]$
3	Vectorul de ieșire
4	Următorul interval de timp pentru actualizarea discretă
9	Încetarea simulării

Setarea indicatorului flag este importantă pentru modul de scriere a unei S-Function:

- dacă sistemul este strict continuu (lipsit de stări discrete), este necesar să se scrie în S-Function stările derivatelor (flag=1) și ieșirile sistemului (flag=3);
- dacă sistemul este strict discret, este necesar să se scrie în S-Function stările discrete (flag=2) și ieșirile sistemului (flag=3) și (flag=0).

Apelul unei S-Function se face o singură dată în timpul simulării pentru flag=0, nu aceeași condiție fiind valabilă pentru apelurile acesteia cu alte valori pentru indicatorul flag. Revenind în cazul flag=0, se initializează un vector care are următoarele semnificații:

- sys(1) - numărul stărilor continue;
- sys(2) - numărul stărilor discrete;
- sys(3) - numărul ieșirilor;
- sys(4) - numărul intrărilor;
- sys(5) - alocat pentru găsirea rădăcinilor (se adoptă zero atunci când matricea D=0);
- sys(6) - flag-ul de acces direct al mărimii de intrare;
- sys(7) - numărul momentelor de eșantionare.

În general, o S-Function se salvează cu numele modelului folosit.
Exemplul 14.3

Se consideră un sistem continuu reprezentat în spațiul stărilor $\dot{x}(t) = Ax(t) + Bu(t)$. Se încearcă realizarea unei S-Function, pentru anumite valori ale matricelor A,B,C,D=0.

```
function [sys,x0,str,Tel]=sistem_continuu(t,x,u,flag)
A=[0 1;-6 -11];
B=[1 ;1];
C=[1 1];
%Daca flag=1 returneaza derivele
if flag==1
    sys=A*x+B*u;
%Daca flag=3 returneaza iesirile
elseif flag==3
    sys=C*x;
%Daca flag=0 initializari
% sys(1) - nr stărilor continue;
% sys(2) - nr stărilor discrete;
% sys(3) - nr ieșirilor;
% sys(4) - nr intrărilor;
% sys(5) - alocat pentru găsirea rădăcinilor;
% sys(6) - flagul de acces direct al mărimii de
% intrare;
% sys(7) - numărul momentelor de eșantionare.
```

```
elseif flag==0
    sys=[2,0,1,1,0,0,0];
    x0=0;str=[];
    Tel=[];
    %In alte conditii nu returneaza nimic
    else sys=[];
    end
```

14.2.1. Conversia unei S-Function într-un bloc

O S-Function poate fi transformată într-un bloc utilizând blocul **S-Function** sau să se definească noi tipuri de modele cu structuri ierarhice. De exemplu, pă îngloba o S-Function într-un bloc, se va introduce numele său în caseta de care se deschide prin dublu click pe blocul S-Function, eventual și numele parametrii adiționali separați prin virgulă. Numele modelului în care se folosește Function, nu trebuie să fie același cu cel al funcției scrise în Matlab.

14.3. Exerciții propuse

Exercițiu 14.3.1

Fie un regulator PID, unde elementul proporțional P are factorul de amplificare 0.1, iar elementul integrativ respectiv elementul derivativ au constantele de egale cu 1. La intrare se aplică un generator de semnale (se va alege din biblioteca User-Defined Functions). Să se simuleze, creând un subsistem masculajitorul regulatorului PID, forma de undă sin.

Exercițiu 14.3.2

Se consideră un circuit de curent continuu, compus dintr-un rezistor bobină inseriate, căruia î se aplică la momentul t=0 o tensiune continuă. Ecuția circuitului este: $L \frac{di}{dt} + Ri = u$, care, exprimată sub formă unei ecuații de stare în

$$\begin{cases} \frac{di}{dt} = -\frac{R}{L}i + \frac{1}{L}u \\ y = i \end{cases}$$

Se observă că $A=-R/L$; $B=1/L$; $C=1$ și $D=0$ ($R=2$; $L=0.05$). Să se realizeze o Function pentru rezolvarea acestui circuit, să se transforme apoi funcția într-un bloc Simulink și să se simuleze pentru o intrare treaptă unitară.

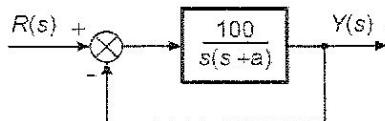
Exercițiu 14.3.3

Se consideră un sistem discret reprezentat în spațiul stării $\begin{cases} x[k+1] = \Phi x[k] + \Gamma u[n] \\ y[k] = Cx[k] + Du[k] \end{cases}$. Condiția inițială este 0.

Matricele au următoarele valori: $\Phi = \begin{bmatrix} 0 & -0.25 \\ 1 & 1 \end{bmatrix}$, $\Gamma = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, $C = [0 \ 1]$. Să se realizeze o S-Function pentru rezolvarea acestui sistem (perioada de eșantionare este $T_e = 0.1$). Să se transforme S-Function într-un bloc în Simulink și să se simuleze pentru o intrare treaptă unitară.

Exercitiul 14.3.4

Să se determine în Simulink răspunsul sistemului de ordinul doi la intrarea treaptă unitară. Pentru funcția de transfer din Simulink se va crea un subsistem mascat. Masca va conține un singur câmp în care se vor introduce pe rând valorile parametrului a ($a = 12,25$).

**BIBLIOGRAFIE**

1. Ogata, K., *Modern Control engineering*, 3rd edition, Prentice Hall 1997.
2. Franklin, G. F., Powell, J. D., Emami-Naeini, A *Feedback Control of Dynamic Systems*, Addison-Wesley Publishing Company, Reading, Massachusetts, 1994.
3. Nise, N. S., *Control system engineering*, 4th edition, John Wiley&Sons Inc. 2004.
4. Cavallo, A., Setola, R., Vasca, F., *Using MATLAB, SIMULINK and Control System Toolbox- A practical approach*, Prentice Hall, Europe, 1996.
5. Dorf, R. S., Bishop, R. H., *Modern Control Systems*, 9th ed., Prentice Hall, Englewood Cliffs, N. J., 2001.
6. Comnac, V., Tollet, I., Lahti, S., *System Theory–Analysis, Modelling and Simulation of Electrical Processes*, Editura Lux Libris, Brașov, 2004.
7. D'Azzo, Houpis, C., Sheldon, S., *Linear Control System Analysis and Design with MATLAB*, 5th edition, M. Dekker Inc., New York, 2003.
8. Dukkipati, R. V., *Analysis and Design of Control System Using MATLAB* New Age, International publishers, 2006.
9. Bishop, R., *Modern Control System Analysis and Design Using MATLAB*, Addison Wesley Publishing Company, 2003.
10. Comnac, V., *Teoria Sistemelor vol I*, Editura Lux Libris, Brașov, 2006.
11. Saadat, H., *Computational Aids in Control Systems using MATLAB*, McGraw-Hill Inc., 1993.



Tipărit la:
Tipografia Universității "Transilvania" din Brașov
Str. Iuliu Maniu, nr. 41A
tel.: 0268 47 60 50; fax: 0268 47 60 51

Referenți științifici: Prof. dr. ing. Florin MOLDOVEANU
Şef lucr. dr. ing. Dan FLOROIAN

UNIVERSITATEA TRANSILVANIA DIN BRAȘOV

Titlul lucrării	: Teoria sistemelor. Îndrumar de laborator
Comanda	: 1 MC/2009
Tehnoredactare:	Autorii
Corectură	Autorii
Coperta	: Cristian Boldișor

Pentru uzul studentilor

8,00 RON

UNIVERSITATEA TRANSILVANIA DIN BRAȘOV
BIBLIOTECĂ-AULĂ

DEPOZIT

Univ. "Transilvania" Brașov



* 777429 *