

**Dănuț Zahariea**

**MATLAB  
CALCUL NUMERIC ȘI SIMBOLIC**



**Dănuț Zahariea**

**MATLAB**  
**CALCUL NUMERIC ȘI SIMBOLIC**

**Editura PIM**  
**Iași 2014**

Coperta: Dănuț Zahariea  
Tehnoredactare: Dănuț Zahariea

Descrierea CIP a Bibliotecii Naționale a României

ZAHARIEA, DĂNUȚ

MATLAB: calcul numeric și simbolic / Zahariea Dănuț - Iași : PIM  
2014

Bibliogr.

ISBN 978-606-13-2091-2

004.42 MATLAB

MATLAB și Simulink sunt mărci înregistrate MathWorks Inc.

MuPAD este marcă înregistrată SciFace Software GmbH & Co. KG.

Informații suplimentare referitoare la MATLAB și aplicațiile asociate pot fi  
obținute la:

MathWorks Inc.

3 Apple Hill Drive

Natick, MA 01760-2098

Phone: (508) 647-7000; Fax: (508) 647-7001

E-mail: [info@mathworks.com](mailto:info@mathworks.com)

<http://www.mathworks.com>

## Prefață

*Lucrarea prezintă elemente semnificative referitoare la utilizarea limbajului de programare MATLAB (MathWorks Inc.) pentru efectuarea de calcule numerice și simbolice. Lucrarea sintetizează experiența didactică de peste 15 ani a autorului în calitate de titular al cursului MATLAB/Simulink în ingineria fluidelor, curs existent în planul de învățământ al studenților specializării de studii universitare de licență Mașini și Sisteme Hidraulice și Pneumatice de la Facultatea de Construcții de Mașini și Management Industrial, Universitatea Tehnică „Gheorghe Asachi” din Iași.*

*Lucrarea se adresează studenților, doctoranzilor, cercetătorilor și cadrelor didactice care își desfășoară activitatea în domeniul științelor tehnice ingineresti, în particular în domeniul ingineriei mecanice. Lucrarea cuprinde 11 capitole în care sunt prezentate atât aspecte teoretice fundamentale, cât și exemple de utilizare a diferitelor metode de lucru și a instrucțiunilor specifice în peste 100 de probleme, multe dintre ele din domeniul aplicativ al mecanicii fluidelor.*

*Capitolul 1 prezintă a scurtă introducere în domeniul tehnologiei informației, arhitecturii și structurii calculatoarelor. Sunt prezentate, de asemenea, elemente de algebră booleană, sisteme de numerație, metode de reprezentare a numerelor în calculator, precum și erorile specifice calculelor numerice.*

*Prezentarea generală a limbajului de programare MATLAB este realizată în capitolul 2: interfața programului, modul de lucru în linie de comandă, fișiere script, funcții, variabile scalare, numere, operatori, funcții matematice elementare.*

*În capitolul 3 se prezintă elemente fundamentale referitoare la algoritmi și metode de reprezentare, structuri de control și schemele lor logice, instrucțiuni de control MATLAB.*

*Capitolul 4 tratează problema variabilelor vectoriale din punct de vedere al: definirii și extragerii elementelor, al operațiilor cu elementele variabilei vectoriale, al operațiilor între un scalar și o variabilă vectorială, precum și al operațiilor între elementele a două variabile vectoriale.*

*În capitolul 5 sunt prezentate elementele de bază cu privire la realizarea reprezentărilor grafice 2D. Sunt prezentate atât principalele metode de reprezentare grafică 2D a funcțiilor, cât și metodele de formatare a graficelor respective.*

*Capitolul 6 conține principalele aspecte de calcul numeric: calcule numerice cu polinoame, rezolvarea ecuațiilor algebrice și transcendente, rezolvarea sistemelor de ecuații neliniare, calculul punctelor de extrem ale*

unei funcții, derivarea și integrarea numerică a funcțiilor, rezolvarea ecuațiilor și sistemelor de ecuații diferențiale ordinare.

În capitolul 7 sunt prezentate principalele elemente de analiză a datelor experimentale. Sunt prezentate etapele analizei statistice a datelor experimentale, precum și instrucțiunile MATLAB necesare. Sunt prezentate de asemenea, principalele instrucțiuni necesare pentru efectuarea analizei numerice a datelor experimentale. Sunt prezentate și interfețele grafice specializate pentru aproximarea numerică a funcțiilor prin interpolare și regresie: interfața Basic Fitting, interfața polytool și interfața cftool.

Capitolul 8 tratează problema variabilelor matriceale din punct de vedere al: definirii și extragerii elementelor unei matrice, al operațiilor cu elementele variabilei matriceale, al operațiilor între un scalar și o variabilă matriceală, al operațiilor între elementele a două variabile matriceale, precum și al operațiilor specifice analizei matriceale. Sunt prezentate două aplicații ale calculului matriceal: rezolvarea sistemelor de ecuații algebrice liniare și transformările geometrice afine.

În capitolul 9 sunt prezentate elementele de bază cu privire la realizarea reprezentărilor grafice 3D. Sunt prezentate atât principalele metode de reprezentare grafică 3D a funcțiilor, cât și trei aplicații specifice: reprezentarea spectrului hidrodinamic al mișcărilor potențiale, reprezentarea câmpurilor vectoriale 2D și 3D, precum și utilizarea transformărilor geometrice afine la proiectarea rețelei de palete rotorice pentru ventilatoarele axiale.

Capitolul 10 abordează elementele de bază ale limbajului de programare Simulink, necesare pentru modelarea și simularea sistemelor dinamice. Sunt prezentate scheme bloc particulare pentru rezolvarea ecuațiilor diferențiale ordinare de ordinul 1 și 2, precum și pentru rezolvarea sistemelor de ecuații diferențiale ordinare.

Capitolul 11 tratează probleme specifice de calcul simbolic în MATLAB, în particular prin metoda limbajului de programare MuPAD (SciFace Software). Sunt prezentate metodele de definire și tehnicile de manipulare a expresiilor și funcțiilor simbolice, instrucțiunile pentru calculul simbolic al limitelor, derivatelor și integralelor, instrucțiunile specifice pentru realizarea reprezentărilor grafice ale expresiilor simbolice. Sunt prezentate, de asemenea, probleme specifice referitoare la rezolvarea simbolică a ecuațiilor și sistemelor de ecuații algebrice și transcendente, precum și a ecuațiilor și sistemelor de ecuații diferențiale ordinare.

*Prof. univ. dr. ing. Zăbărica Dănuț  
Iași, 2014*

# CAPITOLUL 1

## INTRODUCERE

### 1.1. TEHNOLOGIA INFORMAȚIEI

**Tehnologia informației** (Information Technology, IT) reprezintă domeniul științelor ingineresti care are ca obiect prelucrarea informației (accesarea, reprezentarea, manipularea, stocarea și transmiterea informației) folosind resurse tehnice adecvate, precum și resurse umane cu abilități și competențe corespunzătoare. Termenul IT a fost introdus în anul 1958 de către Leavitt H. J. și Whisler T. L. care au menționat: „Această nouă tehnologie nu are încă un nume. O vom numi tehnologia informației”, [16].

**Tehnologia informației și a telecomunicațiilor** (Information and Communication Technology, ICT) reprezintă o extindere a termenului IT, care evidențiază rolul major al sistemelor de telecomunicație în prelucrarea informației. Termenul ICT a fost introdus în anul 1986 de către Melody W., [17]. În prezent, termenul ICT se referă la sistemele unificate de accesare, stocare, manipulare și transmitere a informației între diferite sisteme de calcul și rețele de calculatoare, rețele de transmitere a informației de tip date-audio-video și sisteme complexe de management a informației utilizând rețelele de telefonie și sistemele de telecomunicație de tip wireless, fiind din ce în ce mai des înlocuit cu termenul computing.

**Informatica** (Computer Science) reprezintă ramura științei care are ca obiect studiul prelucrării informației cu ajutorul sistemelor automate de calcul, [4]. Principalele elemente ale informaticii sunt:

- Arhitectura sistemelor de calcul.
- Rețele de calculatoare și sisteme de comunicații.
- Sisteme de operare.
- Metode și limbaje de programare.
- Studiul teoretic al algoritmilor.
- Elemente de calcul numeric și simbolic.
- Structuri de calcul distribuit și paralel.
- Sisteme de management al bazelor de date.
- Sisteme de protecție a informației.
- Inteligența artificială.
- Structuri de interacțiune om-calculator.

**Calculatorul** (Computer), denumit și sistem de calcul sau computer reprezintă ansamblul elementelor fizice (elemente de tip hardware) care pot fi programate astfel încât, prin folosirea anumitor programe generale și specifice (elemente de tip software), să asigure prelucrarea informației în conformitate cu un algoritm definit de utilizator. Utilizând aceleași elemente hardware, calculatorul poate rezolva mai multe probleme prin modificarea programului. Din punct de vedere al elementelor hardware, calculatoarele programabile au evoluat pornind de la sisteme de calcul bazate doar pe componente mecanice, ulterior relee electromagnetice, tuburi electronice, tranzistori, circuite integrate-microprocesoare.

**Inteligența artificială** (Artificial Intelligence, [11]), reprezintă ramura informaticii care are ca obiect proiectarea și realizarea sistemelor inteligente. În general, sistemele inteligente sunt definite ca sisteme care pe baza unor reguli și algoritmi predefiniți, analizează și interpretează toate elementele logice, funcționale și de obiectiv, fiind capabile să aducă modificări algoritmilor inițiali astfel încât să asigure stabilitatea funcțională și să îmbunătățească performanțele sistemului.

**Unitatea de măsurare** pentru cantitatea de informație este bitul (Binary Digit).

Simbolul unității de măsurare a informației este:

- b, în conformitate cu IEEE 1541 Standard (2002), [5];
- bit (notația recomandată), în conformitate cu ISO/IEC Standard 80000-13 (2008), [6].

Bitul reprezintă o variabilă care poate avea doar două valori posibile.

Aceste două valori posibile pot fi asociate cu:

- numerele 0 sau 1;
- valorile logice adevărat (TRUE) sau fals (FALSE);
- semnele algebrice + sau -;
- stările bipoziționale ON sau OFF.

**Principalul multiplu** al unui bit este octetul, denumit și byte (simbolul unui byte este B, în conformitate cu IEC80000-13, IEEE 1541), care reprezintă 8 bit:

$$1 \text{ byte} = 8 \text{ bit}$$

Definirea multiplilor unităților pentru măsurarea informației trebuie realizată făcând distincție între semnificația clasică, conform Sistemului Internațional de Unități (SI) în care multiplii se definesc în sistem zecimal, ca puteri ale numărului 10 și semnificația specifică unității de măsurare a informației, în care multiplii se definesc în sistem binar, ca puteri ale numărului 2. Astfel, dacă în sistem zecimal o anumită unitate de măsură se exprimă prin  $U_{10}$ , iar în sistemul binar, aceeași unitate de măsură este  $U_2$ , atunci eroare relativă de reprezentare a unității de măsură se calculează cu relația generală:



$$\varepsilon_U = \frac{|U_2 - U_{10}|}{U_2} 100 [\%]$$

De exemplu, pentru multiplul kilo, în sistem zecimal 1 K<sub>10</sub> reprezintă 10<sup>3</sup>=1000, în timp ce în sistem binar 1 K<sub>2</sub> este egal cu 2<sup>10</sup>=1024. Eroarea relativă dintre valoarea exprimată în sistem binar și valoarea corespunzătoare din sistemul zecimal se exprimă prin:

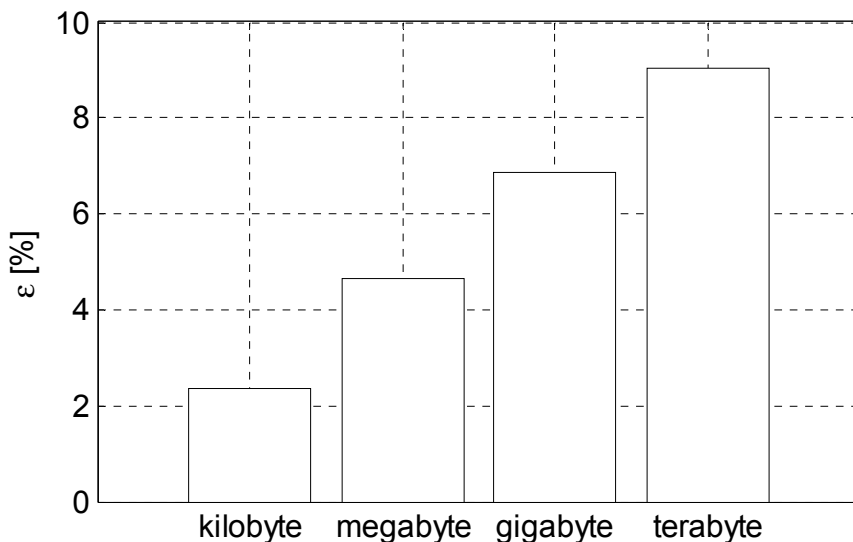
$$\varepsilon_K = \frac{|K_2 - K_{10}|}{K_2} 100 = 2,3438 \%$$

În tabelul 1.1 se prezintă valorile multiplilor kilo, mega, giga și tera exprimate atât în sistem zecimal cât și binar, precum și eroarea relativă dintre aceste valori. Se constată creșterea erorii relative de la 2,3438% pentru multiplul kilo, până la 9,0505% pentru multiplul tera.

**Tabel 1.1.** Eroarea relativă a multiplilor unității de măsurare a informației.

Multiplu		Reprezentare				Eroarea
Nume	Simbol	Sistem zecimal		Sistem binar		$\varepsilon$ [%]
kilo	K	10 <sup>3</sup>	1000 <sup>1</sup>	2 <sup>10</sup>	1024 <sup>1</sup>	2,3438
mega	M	10 <sup>6</sup>	1000 <sup>2</sup>	2 <sup>20</sup>	1024 <sup>2</sup>	4,6326
giga	G	10 <sup>9</sup>	1000 <sup>3</sup>	2 <sup>30</sup>	1024 <sup>3</sup>	6,8677
tera	T	10 <sup>12</sup>	1000 <sup>4</sup>	2 <sup>40</sup>	1024 <sup>4</sup>	9,0505

Pe baza datelor numerice din tabelul 1.1, în figura 1.1 se prezintă variația erorii relative dintre valorile multiplilor unităților de măsurare a informației exprimate în sistemele binar și zecimal.



**Figura 1.1.** Eroarea relativă dintre valorile multiplilor unităților de măsurare a informației.

În vederea evitării acestei confuzii, conform standardului IEC 80000-13:2008, pentru exprimarea multiplilor unităților de măsurare a cantității de informație se recomandă utilizarea unor multipli specifici, prezentați în tabelul 1.2.

**Tabel 1.2.** Multipli recomandați ai unităților pentru măsurarea informației.

Multiplu IEC			Reprezentare			
Nume		Simbol	Sistem binar		Valoare	Sistem zecimal
			Baza 2	Baza 1024		
kibi	kilobinary	Ki	$2^{10}$	$1024^1$	1024	$\cong 1.02 \times 10^3$
mebi	megabinary	Mi	$2^{20}$	$1024^2$	1048576	$\cong 1.05 \times 10^6$
gibi	gigabinary	Gi	$2^{30}$	$1024^3$	1073741824	$\cong 1.07 \times 10^9$
teri	terabinary	Ti	$2^{40}$	$1024^4$	1099511627776	$\cong 1.10 \times 10^{12}$

Din puncte de vedere practic însă, exprimarea multiplilor unităților de măsurare a informației pentru diferitele elemente ale unui calculator se face chiar și în prezent, utilizând atât sistemul binar cât și cel zecimal. Astfel, în general, se poate considera că:

- Pentru module de memorie și unități de tip CD se folosește exprimarea în sistem binar; 512 MB reprezintă  $512 \times 1024^2$  byte.
- Pentru hard-diskuri, unități de stocare de tip flash drive și unități de tip DVD, Blu-ray Disc, HD DVD se folosește exprimarea în sistem zecimal, 1GB reprezintă  $10^9$  byte.
- Toate sistemele de transfer a informației folosesc exprimarea în sistem zecimal.
  - Un modem de 56k are o rată de transfer de 56000 biți pe secundă.
  - O placă de rețea Ethernet de 10 Gbit/s poate transfera  $10 \times 10^9$  biți pe secundă.
  - O placă de rețea InfiniBand QDR poate transfera 32 Gbps, echivalent cu  $32 \times 10^9$  biți pe secundă.
  - O magistrală de comunicație de tip SATA 3.0 are o rată de transfer de 6 Gbps, deci  $6 \times 10^9$  biți pe secundă.
  - Magistrala de comunicație USB 3.1 (Superspeed USB) are o rată de transfer de 10 Gbps, echivalentă cu  $10 \times 10^9$  biți pe secundă.
  - Magistrala de comunicație de tip Thunderbolt are o rată de transfer totală de 20 Gbps, echivalentă cu  $20 \times 10^9$  biți pe secundă.

## 1.2. ARHITECTURA CALCULATOARELOR PERSONALE

Conceptual, arhitectura calculatoarelor personale se referă la modul de interconectare a diferitelor elemente componente, astfel încât să se asigure performanțele generale și specifice impuse, [3, 10, 13].

Principalele elemente componente ale unui calculator personal sunt:

- **Memoria internă (M)** este utilizată pentru stocarea atât a instrucțiunilor, a datelor de intrare, cât și a rezultatelor obținute în urma executării instrucțiunilor. Memoria internă este de două feluri:
  - Memorie de tip ROM (Read Only Memory, memorie nevolatilă), este memoria utilizată doar pentru citire, având un conținut care nu se pierde la oprirea calculatorului. În memoria ROM sunt stocate programe și date necesare în permanență sistemului de calcul, ca de exemplu BIOS-ul sistemului de operare (Basic Input/Output System) care se încarcă în mod automat la pornirea sistemului de calcul.
  - Memorie de tip RAM (Random Access Memory, memorie volatilă), este memoria utilizată atât pentru citire cât și pentru scriere, al cărei conținut se pierde la oprirea calculatorului. În memoria RAM se stochează datele și instrucțiunile necesare, reprezentând memoria principală a calculatorului.
- **Unitatea centrală de procesare (Central Processing Unit-CPU)** este utilizată pentru executarea instrucțiunilor. Unitatea centrală de procesare are două componente principale:
  - Unitatea de comandă și control (UC) primește instrucțiunile de la memorie, le interpretează și transmite apoi comenzi către unitatea aritmetică-logică și către memorie, precum și comenzi de transfer către dispozitivele de intrare/ieșire.
  - Unitatea aritmetică-logică (UAL) are rolul de a executa diferite tipuri de operații aritmetice și logice asupra datelor, precum și de a transfera către memorie rezultatele obținute.
- **Dispozitivele de intrare/ieșire (I/O)** asigură interconectarea dintre CPU și utilizator prin dispozitive periferice de intrare (tastatură, mouse, etc.) și de ieșire (monitor, imprimantă, etc.), respectiv dintre CPU și mediile de stocare (floppy-disc, CD-ROM, DVD-ROM, hard disk, etc.).
- **Magistrala de comunicație** asigură interconectarea internă a tuturor elementelor componente ale calculatorului și are trei componente:
  - Magistrala de date asigură transferul bidirecțional al datelor între unitatea aritmetică-logică și memorie.
  - Magistrala de adrese asigură transferul spre unitatea centrală de procesare atât al adreselor de memorie pentru efectuarea de operațiuni de citire sau scriere cu memoria, cât și al

adreselor dispozitivelor de intrare/ieșire pentru transferul datelor cu dispozitivele periferice.

- Magistrala de control asigură transmiterea semnalelor de comandă și control realizând astfel sincronizarea funcționării tuturor componentelor calculatorului.

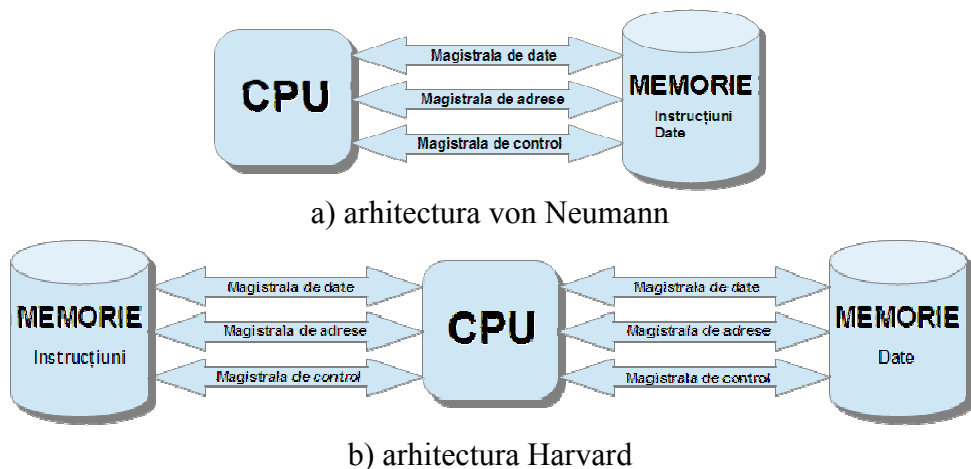
Funcționarea elementelor componente ale unui sistem de calcul presupune parcurgerea mai multor etape:

- Înainte de executarea unui program, instrucțiunile acestuia ca și o serie de date inițiale trebuie să se afle în memoria internă. Ca urmare, prima operațiune este citirea și transferul acestor informații de pe mediul de stocare (hard disk) în memoria internă a calculatorului. Organizarea informațiilor în memorie se realizează cu ajutorul adreselor. Adresele sunt identificatori unici ai locațiilor de memorie care asigură localizarea exactă a tuturor zonelor de memorie permițând astfel furnizarea conținutului unei anumite locații din memorie pe baza adresei acesteia.
- Unitatea de comandă și control transmite pe magistrala de adrese, adresa de memorie care conține instrucțiunea de executat (faza de adresare). Instrucțiunea este apoi citită și transferată pe magistrala de date spre unitatea de comandă și control (faza de citire). Unitatea de comandă și control decodifică instrucțiunea (faza de decodificare) și transmite comenzile necesare executării spre unitatea aritmetică logică care realizează execuția instrucțiunii respective (faza de execuție). Rezultatul obținut în urma executării instrucțiunii este apoi transferat pe magistrala de date înapoi spre memorie.

Existența în memoria internă a calculatorului atât a instrucțiunilor, cât și a datelor a reprezentat un concept inovator introdus în anul 1945 de matematicianul american John von Neumann, recunoscut astăzi drept „părintele calculatoarelor electronice”, [18].

Pe baza acestui concept, von Neumann a elaborat o arhitectură a sistemelor de calcul, denumită arhitectura von Neumann, întâlnită și astăzi în multe din calculatoarele personale, figura 1.2, a). Faptul că atât datele cât și instrucțiunile se transferau între memoria internă și unitatea centrală de procesare pe aceeași magistrală de date a reprezentat o limitare conceptuală importantă a arhitecturii von Neumann (datele și instrucțiunile nu se puteau procesa în același timp, procesorul trebuia să aștepte ca datele necesare executării instrucțiunilor să fie transmise spre și dinspre memoria internă).

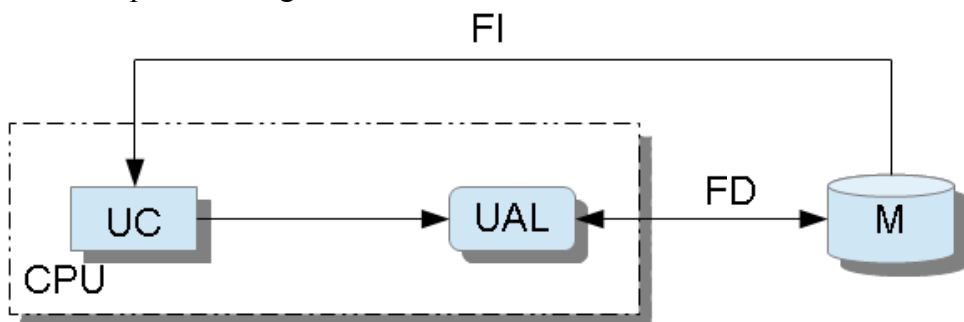
Eliminarea limitării conceptuale a modelului von Neumann s-a realizat prin modelul Harvard, [15] (figura 1.2, b), la care există magistrale de comunicație diferite pentru instrucțiuni și pentru date, astfel încât procesorul primește în permanență informații atât sub forma instrucțiunilor, cât și a datelor.



**Figura 1.2.** Arhitecturi conceptuale ale calculatoarelor.

Din punct de vedere al arhitecturii interne, cercetătorul american Flynn M. J. a propus în anul 1966 o clasificare a sistemelor de calcul în funcție de numărul fluxurilor de date și al fluxurilor de instrucțiuni, [14]:

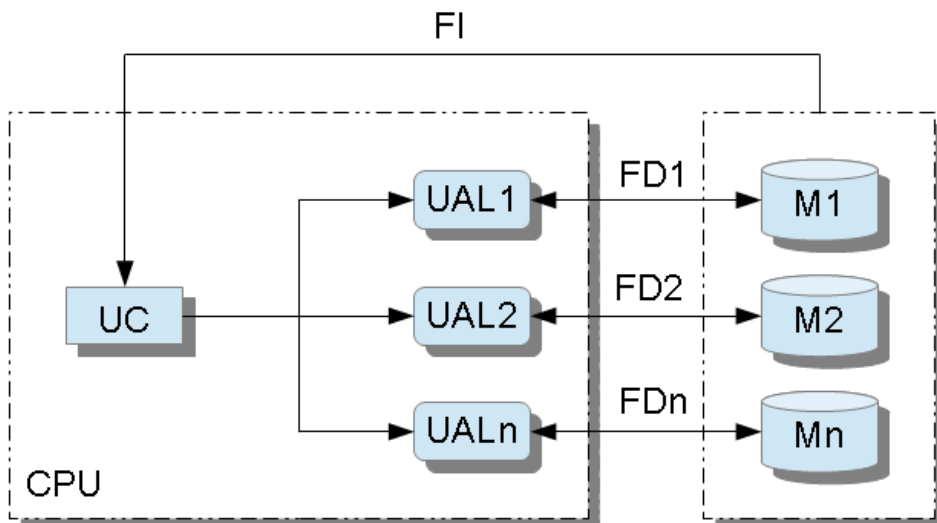
- Sisteme de calcul cu un singur flux de instrucțiuni FI și un singur flux de date FD, SISD (Single Instruction Single Data), figura 1.3. Modelul SISD corespunde arhitecturii von Neumann fiind caracteristic sistemelor de calcul cu un singur microprocesor pe 8, 16, 32 sau 64 de biți, la care un singur flux de instrucțiuni acționează asupra unui singur flux de date. Modelul SISD conține o singură unitate de control UC care coordonează o singură unitate aritmetică logică UAL aflată în conexiune cu o singură unitate de memorie M, fiind astfel capabil să execute în același timp o singură instrucțiune asupra unei singure date.



**Figura 1.3.** Arhitectura Flynn de tip SISD.

- Sisteme de calcul cu un singur flux de instrucțiuni FI și mai multe fluxuri de date FD1, FD2, ..., FDn, SIMD (Single Instruction

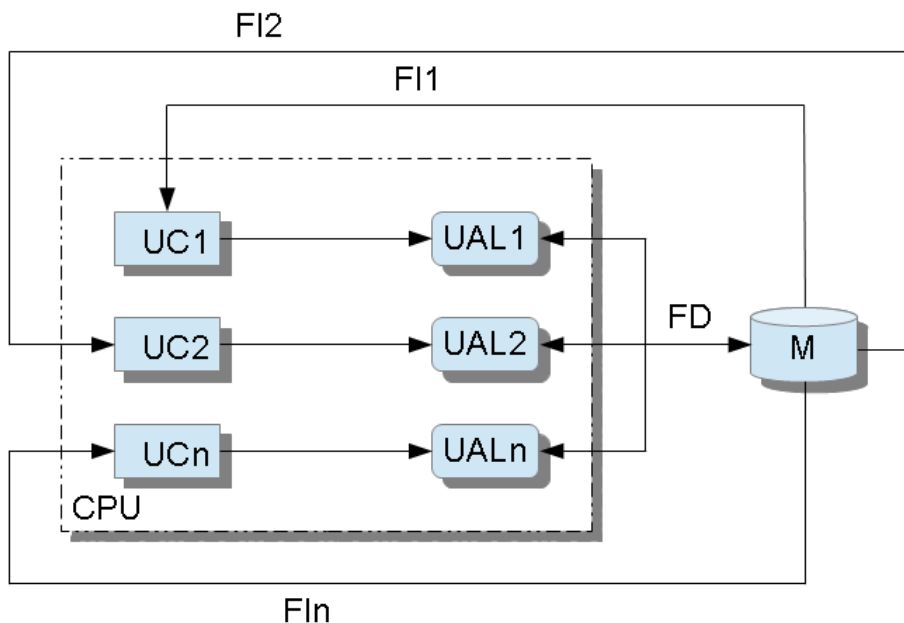
Multiple Data), figura 1.4. Modelul SIMD devine eficient în cazul executării aceleiași instrucțiuni asupra mai multor date. Modelul SIMD conține o singură unitate de control UC care coordonează mai multe unități aritmetice logice UAL1, UAL2,...,UALn aflate în conexiune cu mai multe unități de memorie M1, M2, ..., Mn, fiind astfel capabil să execute în același timp aceeași instrucțiune asupra mai multor date. Acest model corespunde sistemelor de calcul de tip “array processors”.



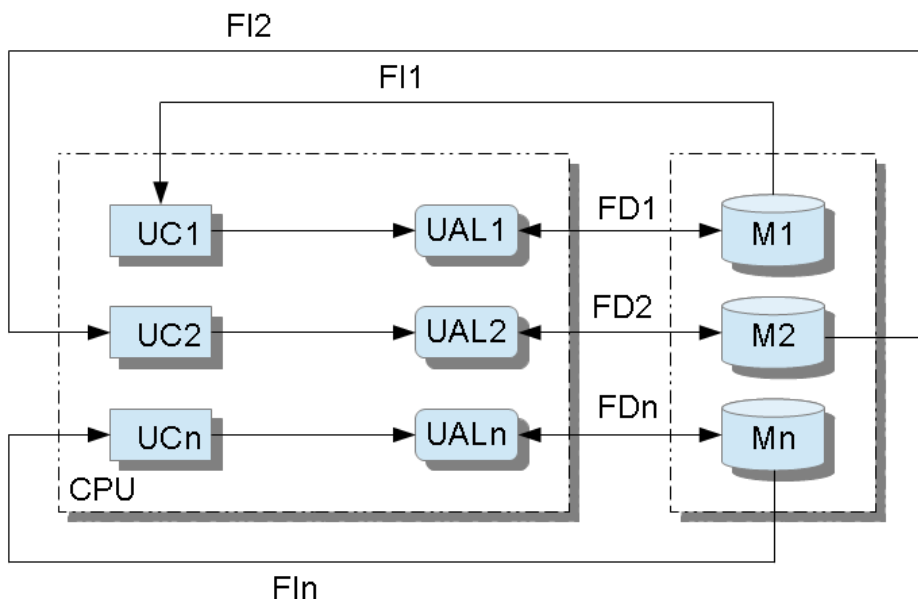
**Figura 1.4.** Arhitectura Flynn de tip SIMD.

- Sisteme de calcul cu mai multe fluxuri de instrucțiuni FI1, FI2, ..., FI<sub>n</sub> și un singur flux de date FD, MISD (Multiple Instruction Single Data), figura 1.5. Modelul MISD conține mai multe unități de control UC1, UC2, ..., UC<sub>n</sub> care coordonează mai multe unități aritmetice logice UAL1, UAL2, ..., UAL<sub>n</sub> aflate în conexiune cu aceeași unitate de memorie M, fiind astfel capabil să execute în același timp mai multe instrucțiuni asupra aceleiași date. Acest model corespunde sistemelor de calcul multiprocesor de tip centralizat (multiprocesor cu acces uniform la memorie-UMA).
- Sistemul de calcul cu mai multe fluxuri de instrucțiuni FI1, FI2, ..., FI<sub>n</sub> și mai multe fluxuri de date FD1, FD2, ..., FD<sub>n</sub>, MIMD (Multiple Instruction Multiple Data), figura 1.6. Modelul MIMD conține mai multe unități de control UC1, UC2, ..., UC<sub>n</sub> care coordonează mai multe unități aritmetice logice UAL1, UAL2, ..., UAL<sub>n</sub> aflate în conexiune cu mai multe unități de memorie M1, M2, ..., Mn, fiind astfel capabil să execute în același timp mai multe instrucțiuni asupra mai multor date. Acest model corespunde

sistemelor de calcul multiprocesor de tip distribuit (multiprocesor cu acces neuniform la memorie-NUMA).



**Figura 1.5.** Arhitectura Flynn de tip MISD.



**Figura 1.6.** Arhitectura Flynn de tip MIMD.

### 1.3. STRUCTURA CALCULATOARELOR PERSONALE

Elementele componente care definesc structura internă a calculatoarelor personale sunt de două tipuri: elemente de tip hardware și elemente de tip software.

#### 1.3.1. Elemente de tip hardware

Principalele elemente de tip hardware sunt:

- **Carcasa.** Conține principalele componente ale calculatorului: sursa de alimentare; placa de bază; hard-disk-ul; unitatea floppy-disk; unitatea CD/DVD; cablurile de interconectare; ventilatoare pentru răcire; etc. Tipul carcasei este în strânsă corelare cu factorul de formă al plăcii de bază, cele mai răspândite fiind ATX, extended ATX și microATX. Din punct de vedere al înălțimii, carcusele se clasifică în: full-tower ( $\cong 560$  mm), mid-tower ( $\cong 460$  mm) și mini-tower ( $\cong 410$  mm). Asigurarea bunei funcționări a sistemului de calcul necesită menținerea temperaturii sistemului între limitele prestabilite prin gruparea judicioasă a cablurilor din interiorul carcasei, prin asigurarea traseelor de circulație a aerului și prin montarea în carcasă a unor ventilatoare de răcire.
- **Placa de bază.** Reprezintă elementul care asigură interconectarea tuturor celorlalte componente ale unui calculator: chipsetul; sochetul microprocesorului; microprocesorul și sistemul de răcire al microprocesorului; sloturile pentru conectarea dispozitivelor externe; sloturile pentru conectarea memoriei; porturile de intrare/ieșire; etc. Unele plăci de bază au încorporate și placa de rețea, placa audio și placa video. Din punct de vedere al numărului de procesoare suportate, în general, plăcile de bază pentru calculatoare personale pot fi mono-procesor și bi-procesor.
- **Microprocesorul.** Reprezintă elementul principal al unui calculator și care este responsabil de procesarea datelor în orice aplicație care rulează pe calculatorul respectiv. Conectarea microprocesorului cu placa de bază se realizează prin intermediul unui socket. Principalele tipuri de socketuri sunt: PGA (Pin Grid Array) care constă într-o rețea de găuri în care se vor introduce pini microprocesorului; LGA (Land Grid Array) care constă într-o rețea de pini care vor intra în contact cu rețeaua corespunzătoare de mici contacte electrice de pe microprocesor. Una din principalele probleme ale microprocesoarelor este degajarea unei importante cantități de căldură. Evacuarea căldurii și menținerea temperaturii microprocesorului în limitele funcționale se realizează cu ajutorul unor coolere. Din punct de vedere al tipului de sistem pentru care sunt destinate, procesoarele pot fi pentru sisteme de tip desktop, de



tip mobile și de tip server. În funcție de setul de instrucțiuni, în prezent, marea majoritate a procesoarelor sunt pe 64 de biți. Principalele caracteristici care fac diferența între microprocesoarele actuale sunt: tehnologia de fabricație, viteza (frecvența ceasului intern), numărul de nuclee, numărul de fire de execuție, cantitatea maximă de memorie suportată, configurația multiprocesor suportată, tehnologiile încorporate (de exemplu, Hyper-Threading, Virtualization, Turbo Boost, etc.), dimensiunea memoriei cache. Memoria cache este o memorie volatilă de tip static, inclusă în procesor, în care se stochează cele mai frecvent utilizate date și instrucțiuni. Datorită timpilor de acces mult mai mici, utilizarea ulterioară a informațiilor din memoria cache este mult mai rapidă decât dacă ar fi accesată memoria internă a sistemului de calcul.

- **Memoria internă.** Este o memorie volatilă de tip RAM (Random Access Memory) în care se transferă în mod temporar instrucțiunile și datele în vederea prelucrării. Memoria RAM este mult mai rapidă decât orice alt tip de mediu de stocare: HDD (Hard Disk), FDD (Floppy Disk), CD ROM (Compact Disk Read Only Memory), DVD ROM (Digital Video Disk), etc. Conectarea memoriilor cu placa de bază se realizează prin intermediul sloturilor speciale de memorie. Principalele tipuri de memorii sunt: DRAM (Dynamic RAM) și SRAM (Static RAM). Memoria de tip SRAM este mai rapidă dar și mai scumpă. Cea mai utilizată memorie de tip DRAM este DDR SDRAM (Double Data Rate Synchronous Dynamic Random-Access Memory). Principalele caracteristici ale plăcilor de memorie sunt: capacitatea (512 MB, 1 GB, 2 GB, 4 GB, 8 GB, 16 GB, 32 GB, 64 GB, 128 GB) și viteza de lucru (DDR1: 200...400 MHz; DDR2: 400...1066 MHz; DDR3: 800...2133 MHz; DDR4: 2133...4266 MHz).
- **Unitatea de stocare hard disk.** Asigură transferul informației în vederea stocării folosind medii interne nevolatile. Pe hard disk se găsesc: sistemul de operare, programele, fișierele, etc. Principalele elemente componente ale hard disk-urilor sunt: discurile magnetice pentru stocarea informației (platane), motorul electric care asigură rotația discurilor, capetele magnetice de scriere/citire, sistemul de poziționare al capetelor magnetice de scriere/citire. Principalele caracteristici ale hard-disk-urilor sunt: capacitatea de stocare (valori uzuale 500 GB, 1 TB, 2 TB, 3 TB, 4 TB), viteza de rotație a discurilor (5400 rpm, 7200 rpm, 10000 rpm, 15000 rpm), interfața de conectare cu placa de bază (IDE-Integrated Drive Electronics, EIDE-Enhanced Integrated Drive Electronics, PATA-Parallel ATA, SATA-Serial ATA, USB-Universal Serial Bus, SCSI-Small

Computer System Interface, SAS-Serial Attached SCSI), viteza de scriere/citire, timpul mediu de acces. În încercarea de a elimina elementele în mișcare din structura hard disk-urilor, s-a ajuns la soluția utilizării unui mediu de stocare nevolatilă de tip flash memory în așa numitul SSD (Solid State Drive) care este mult mai rapid și asigură reducerea considerabilă a consumului de energie. O varianta intermediară între unitatea SSD și clasicul HDD o reprezintă unitatea de stocare de tip hibrid care poate fi de două tipuri: unitate hibridă de tip dual-drive formată din două unități de stocare separate, o unitate SSD și o unitate HDD; unitatea hibridă de tip solid state (SSHD) formată prin încorporarea unei unități SSD în structura unui HDD clasic. Managementul unităților de stocare de tip hibrid se realizează fie de către utilizator prin copierea pe unitatea mai rapidă a datelor mai des utilizate, fie de către sistemul de operare folosind același principiu ca în cazul memoriei de tip cache.

- **Unitățile de stocare CD/DVD/BD.** Reprezintă principala modalitate de transfer a informației în vederea stocării folosind medii externe nevolatile. Capacitatea uzuală de stocare a unităților de tip CD este de 700 MB, în timp ce unitățile de tip DVD single-layer au o capacitate de stocare de 4,7 GB, iar cele de tip DVD dual-layer pot avea o capacitate de stocare de 8,5 GB. Conectarea unității CD/DVD cu placa de bază se realizează prin intermediul unui conector de tip IDE, PATA, SATA, USB. Rescrierea informației este posibilă doar pentru anumite formate ale acestor unități de stocare: CD-R, CD-RW, DVD-R/RW, DVD+R/RW, DVD-RAM, DVD-R, DVD+R, DVD-R DL, DVD+R DL. Unitățile de stocare de tip BD (Blu-ray Disc) au aceeași dimensiune geometrică ca și discurile DVD dar au o capacitate de stocare mult mai mare, 25 GB (single-layer).
- **Placa video.** Realizează transferul informației spre monitor. Unele plăci de bază conțin o placă video încorporată, însă pentru obținerea unor performanțe grafice superioare se utilizează o placă video dedicată. Procesorul grafic inclus pe placa video preia o serie din sarcinile microprocesorului calculatorului îmbunătățind astfel performanța întregului sistem de calcul. Conectarea plăcii video cu placa de bază se realizează pe un port special denumit PCI Express. Conectarea plăcii video cu monitorul se poate realiza prin mai multe metode: conector de tip VGA (Video Graphics Array) conector de tip DVI (Digital Video Interactive) și conector de tip HDMI (High Definition Multimedia Interface). Unele plăci video au mai multe porturi de conectare, permițând astfel conectarea mai multor monitoare. Unele plăci video dispun și de un port de tip S-Video (TV Out) care permite utilizarea ca și monitor a unui televizor.

Principalele caracteristici ale plăcilor video sunt: tipul procesorului grafic, numărul de biți pe care se realizează transmisia semnalului video între procesorul grafic și memoria video (32 biți, 64 biți, 128 biți, 192 biți, 256 biți, 384 biți), tipul de memorie video (DDR, DDR2, DDR3, DDR4, DDR5), cantitatea de memorie video disponibilă (256 MB, 512 MB, 1 GB, 2 GB, 3 GB, 4 GB, 12 GB), numărul de nuclee de calcul paralel CUDA (192, 384, 768, 1536, 2880), tehnologii încorporate (de exemplu OpenGL, Shader Model, DirectX, Optimus, PhysX, Multi GPU, CUDA).

- **Monitorul.** Reprezintă principalul dispozitiv de ieșire al calculatorului. Se conectează cu placa video a calculatorului. Există două tipuri de monitoare: monitoare de tip CRT (Cathode Ray Tube) care se conectează cu placa video prin portul VGA și monitoare de tip LCD (Liquid Crystal Display) care se conectează cu placa video prin portul S-VGA sau DVI. Principalele caracteristici ale monitoarelor actuale sunt: diagonala, factorul de formă, rezoluția nativă, dimensiunea unui pixel, numărul de culori, luminozitatea, contrastul, timpul de răspuns, vizibilitatea, conectori de intrare/ieșire, consumul mediu de energie electrică.
- **Placa de sunet.** Realizează conversia semnalelor electrice în semnale sonore care pot fi apoi auzite prin intermediul unor difuzoare. În prezent majoritatea plăcilor de bază conțin o placă audio încorporată, însă pentru obținerea unei performanțe superioare trebuie să se utilizeze o placă audio dedicată. Conectarea plăcii audio cu placa de bază se realizează prin intermediul unui port de conexiune de tip: PCI (Peripheral Component Interconnect), PCI Express (Peripheral Component Interconnect Express), PCMCIA (Personal Computer Memory Card International Association), USB.
- **Difuzoarele.** Reprezintă elementul final al unui sistem audio permițând transmiterea sunetelor spre utilizator. Din punct de vedere al configurației și în strânsă corelare cu tipul plăcii de sunet, principalele tipuri de difuzoare sunt: mono, stereo, 2.1, 5.1 și 7.1.
- **Microfonul.** Reprezintă un sistem care realizează conversia sunetelor în semnale electrice.
- **Camera Web.** Reprezintă o cameră video miniaturală care permite transferul în timp real al imaginilor către calculator. În general, camerele Web au un microfon încorporat, asigurând astfel transmiterea combinată a semnalelor audio-video.
- **Placa de rețea.** Reprezintă ansamblul componentelor care permit conectarea sistemului de calcul la rețea folosind sisteme de conexiune prin cablu sau wireless. Pentru realizarea diferitelor configurații de rețea, unele plăci de rețea pot avea mai multe porturi.

Pentru creșterea cantității de informație transmise, plăcile de rețea wireless pot avea mai multe antene.

- **Tastatura.** Reprezintă principalul dispozitiv de intrare al calculatoarelor personale și conține taste pentru introducerea literelor, a cifrelor, a caracterelor speciale; taste pentru asigurarea deplasării cursorului și taste pentru funcții speciale. Cele mai răspândite modele sunt: 101 key US și 104 key Windows. Conectarea tastaturii la calculator se realizează prin cablu sau prin sistem de tip wireless.
- **Mouse-ul.** Este un dispozitiv care convertește mișcarea în coordonate xOy în deplasarea cursorului de pe monitorul unui sistem de calcul. Conversia mișcării mouse-ului se poate realiza prin sistem mecanic sau optic. Mouse-ul are în general trei butoane, din care butonul din mijloc are și funcția de scroll. Conectarea mouse-ului cu sistemul de calcul se realizează prin cablu sau prin sistem wireless.
- **Imprimanta.** Este dispozitivul de ieșire care permite transformarea unui document electronic într-un document tipărit utilizând diferite medii fizice (suport de hârtie, folie transparentă, etc.). Principalele tipuri de imprimante sunt: imprimanta matriceală, imprimanta laser, imprimanta cu jet de cerneală, imprimanta termică (cu cerneală solidă).

### 1.3.2. Elemente de tip software

Principalele elemente de tip software sunt:

- **Sistemul de operare.** Realizează interacțiunea dintre utilizator și sistemul de calcul, asigurând în același timp alocarea corespunzătoare a resurselor hardware ale calculatorului pentru a finaliza în condiții optime sarcinile de calcul. Din punct de vedere al resurselor, o importanță deosebită o are alocarea memoriei și gestionarea procesoarelor/nucleelor. În cazul în care, memorie internă fizică a sistemului de calcul nu este suficientă, sistemele de operare alocă spații din memoria mai lentă de pe hard disk pentru simularea unei memorii de tip virtual. Din punct de vedere al numărului de procesoare suportate, sistemele de operare se clasifică în sisteme de operare de tip desktop (maxim două procesoare) și de tip server (mai mult de două procesoare). După numărul de utilizatori care pot lucra la un moment dat, sistemele de operare pot fi monoutilizator și multiutilizator. După numărul de programe care pot fi executate în același timp, sistemele de operare pot fi monotasking și multitasking. Sistemele monotasking nu permit lansarea unui program până ce nu s-a terminat execuția programului anterior. Sistemele multitasking permit, prin diverse metode,

comutarea rapidă a sarcinilor pe procesor realizând astfel execuția mai multor programe cvasi-simultan. Interfața cu utilizatorul a sistemelor de operare poate fi de tip text (Text User Interface), sau de tip grafic (Graphic User Interface).

Principalele sisteme de operare din punct de vedere al cotei de piață (iulie 2014) sunt, [20]:

- Windows 7 (51,22%), Microsoft Corp.
  - Windows XP (24,82%), Microsoft Corp.
  - Windows 8/8.1 (12,48%), Microsoft Corp.
  - OSX (6,64%), Apple Inc.
  - Windows Vista (3,05%), Microsoft Corp.
  - GNU/Linux (1,58%), open source software.
  - Altele (2,65%).
- **Limbaje de programare.** Reprezintă un limbaj utilizat pentru a transmite instrucțiuni unui calculator. Din punct de vedere al modului în care sunt executate instrucțiunile, limbajele de programare se clasifică în:
    - Limbaje de programare de tip compilator. Limbajele de tip compilator transformă programul sursă prin compilare într-un program echivalent în limbaj mașină care este apoi executat în întregime de procesor. Exemple de limbaje de programare de tip compilator: FORTRAN, C, C++.
    - Limbaje de programare de tip interpretor. Limbajul de tip interpretor realizează aceeași transformare dar nu asupra întregului program sursă, ci asupra fiecărei instrucțiuni în parte. Exemple de limbaje de programare de tip interpretor: PASCAL, BASIC, MATLAB.
  - **Software aplicativ de tip office.** Reprezintă o colecție de programe având o interfață comună care furnizează aplicații specifice pentru: editare de text, calcul tabelar, realizarea de prezentări, baze de date, etc. Exemple de produse software de tip office: Microsoft Office, Microsoft Works, IBM Lotus Smartsuite, Apache Open Office.
  - **Software aplicativ de calcul numeric.** Principalele limbaje de programare destinate efectuării de calcule numerice și simbolice sunt: MATLAB (MathWorks), Maple (Maplesoft), Mathcad (Mathsoft, PTC); Mathematica (Wolfram Research).
  - **Software aplicativ de tip CAD/CAE** (Computer Aided Design/Computer Aided Engineering).Principalele aplicații software de tip CAD/CAE sunt: CATIA (Dassault Systemes), ANSYS (ANSYS), Solid Edge (Siemens PLM), Solid Works (Dassault Systemes), AutoCAD și Inventor (Autodesk), Pro/Engineer (Parametric Technology), Microstation (Bentley Systems).

## 1.4. ELEMENTE DE ALGEBRĂ BOOLEANĂ

### 1.4.1. Operații binare și unare

Algebra booleană reprezintă un tip particular de algebră formată dintr-o mulțime de elemente și un set de operații cu ajutorul cărora se pot formula o serie de propoziții, [12]. Propozițiile sunt afirmații care pot fi ori adevărate, ori false, însă nu pot fi și adevărate și false în același timp. Valoarea de adevăr a unei propoziții este notată cu 1, iar valoarea de fals se notează cu 0.

Principalele caracteristici ale algebrei booleene sunt:

- Mulțimea de elemente cu care operează algebra booleană este formată din elementele 1 (adevărat, TRUE) și 0 (fals, FALSE).
- Două operații binare denumite disjuncție logică (sau, OR) și conjuncție logică (și, AND), simbolizate prin simbolurile  $\vee$  și  $\wedge$ .

Disjuncția logică (OR) a două propoziții  $x$  și  $y$  este acea propoziție care este falsă atunci când ambele propoziții  $x$  și  $y$  sunt false, și adevărată în caz contrar. Tabelul de adevăr pentru operația binară de disjuncție logică (OR) este:

$x$	$y$	$x \vee y$
0	0	0
0	1	1
1	0	1
1	1	1

Conjuncția logică (AND) a două propoziții  $x$  și  $y$  este acea propoziție care este adevărată numai atunci când ambele propoziții  $x$  și  $y$  sunt adevărate, și falsă în caz contrar. Tabelul de adevăr pentru operația binară de conjuncție logică (AND) este:

$x$	$y$	$x \wedge y$
0	0	0
0	1	0
1	0	0
1	1	1

- O operație unară denumită negație logică (NOT), simbolizată prin simbolul  $\bar{\phantom{x}}$ .

Negația logică (NOT) a unei propoziții  $x$  este acea propoziție care este adevărată dacă  $x$  este falsă și falsă dacă  $x$  este adevărată. Tabelul de adevăr al operației unare de negație logică (NOT) este:

$x$	$\bar{x}$
1	0
0	1

- Ordinea de prioritate a operațiilor binare este: NOT, AND, OR.

### 1.4.2. Proprietățile algebrei boolene

Principalele proprietăți ale algebrei boolene sunt:

- Tautologia:

$$x \vee x = x; x \wedge x = x$$

	$x \vee x = x$	$x \wedge x = x$
$x$	$x \vee x$	$x \wedge x$
0	0	0
1	1	1

- Comutativitatea:

$$x \vee y = y \vee x; x \wedge y = y \wedge x$$

		$x \vee y = y \vee x$		$x \wedge y = y \wedge x$	
$x$	$y$	$x \vee y$	$y \vee x$	$x \wedge y$	$y \wedge x$
0	0	0	0	0	0
0	1	1	1	0	0
1	0	1	1	0	0
1	1	1	1	1	1

- Asociativitatea:

$$(x \vee y) \vee z = x \vee (y \vee z)$$

$x$	$y$	$z$	$x \vee y$	$y \vee z$	$(x \vee y) \vee z$	$x \vee (y \vee z)$
0	0	0	0	0	0	0
0	0	1	0	1	1	1
0	1	0	1	1	1	1
0	1	1	1	1	1	1
1	0	0	1	0	1	1
1	0	1	1	1	1	1
1	1	0	1	1	1	1
1	1	1	1	1	1	1

$$(x \wedge y) \wedge z = x \wedge (y \wedge z)$$

$x$	$y$	$z$	$x \wedge y$	$y \wedge z$	$(x \wedge y) \wedge z$	$x \wedge (y \wedge z)$
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	1	0	0
1	0	0	0	0	0	0
1	0	1	0	0	0	0
1	1	0	1	0	0	0
1	1	1	1	1	1	1

- Absorbția:

$$x \vee (x \wedge y) = x; x \wedge (x \vee y) = x$$

		$x \vee (x \wedge y) = x$		$x \wedge (x \vee y) = x$	
$x$	$y$	$x \wedge y$	$x \vee (x \wedge y)$	$x \vee y$	$x \wedge (x \vee y)$
0	0	0	0	0	0
0	1	0	0	1	0
1	0	0	1	1	1
1	1	1	1	1	1

- Distributivitatea:

$$x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$$

$x$	$y$	$z$	$y \wedge z$	$x \vee y$	$x \vee z$	$x \vee (y \wedge z)$	$(x \vee y) \wedge (x \vee z)$
0	0	0	0	0	0	0	0
0	0	1	0	0	1	0	0
0	1	0	0	1	0	0	0
0	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1

$$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$$

$x$	$y$	$z$	$y \vee z$	$x \wedge y$	$x \wedge z$	$x \wedge (y \vee z)$	$(x \wedge y) \vee (x \wedge z)$
0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	0	0	0	0
1	0	0	0	0	0	0	0
1	0	1	1	0	1	1	1
1	1	0	1	1	0	1	1
1	1	1	1	1	1	1	1

- Există un element 0 (element neutru pentru operația  $\vee$ ), denumit prim element, cu proprietățile:

$$x \vee 0 = x; x \wedge 0 = 0$$

		$x \vee 0 = x$	$x \wedge 0 = 0$
$x$		$x \vee 0$	$x \wedge 0$
0		0	0
1		1	0



- Există un element 1 (element neutru pentru operația  $\wedge$ ), denumit ultim element, cu proprietățile:

$$x \wedge 1 = x; x \vee 1 = 1$$

	$x \vee 1 = 1$	$x \wedge 1 = x$
$x$	$x \vee 1$	$x \wedge 1$
0	1	0
1	1	1

- Legea dublei negații:

$$\bar{\bar{x}} = x$$

	$\bar{\bar{x}} = x$	
$x$	$\bar{x}$	$\bar{\bar{x}}$
0	1	0
1	0	1

- Legea contradicției:

$$x \wedge \bar{x} = 0$$

	$x \wedge \bar{x} = 0$	
$x$	$\bar{x}$	$x \wedge \bar{x}$
0	1	0
1	0	0

- Legea terțului exclus:

$$x \vee \bar{x} = 1; x \wedge \bar{x} = 0$$

		$x \vee \bar{x} = 1$	$x \wedge \bar{x} = 0$
$x$	$\bar{x}$	$x \vee \bar{x}$	$x \wedge \bar{x}$
0	1	1	0
1	0	1	0

- Legile lui De Morgan:

$$\overline{x \vee y} = \bar{x} \wedge \bar{y}$$

		$\overline{x \vee y} = \bar{x} \wedge \bar{y}$				
$x$	$y$	$x \vee y$	$\overline{x \vee y}$	$\bar{x}$	$\bar{y}$	$\bar{x} \wedge \bar{y}$
0	0	0	1	1	1	1
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	0

$$\overline{x \wedge y} = \bar{x} \vee \bar{y}$$

		$\overline{x \wedge y} = \bar{x} \vee \bar{y}$				
$x$	$y$	$x \wedge y$	$\overline{x \wedge y}$	$\bar{x}$	$\bar{y}$	$\bar{x} \vee \bar{y}$
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	0

## 1.5. SISTEME DE NUMERAȚIE

### 1.5.1. Clasificarea sistemelor de numerație

Sistemul de numerație reprezintă ansamblul regulilor utilizate pentru exprimarea numerelor cu ajutorul unui set unitar de simboluri, [19].

Sistemele de numerație sunt de două feluri:

- **Sisteme de numerație nepoziționale.** Sunt sistemele de numerație pentru care valoarea unei anumite cifre din structura unui număr nu este unic determinată de poziția cifrei în numărul respectiv, ci de contextul în care se găsește cifra. Reprezentativ pentru sistemele de numerație nepoziționale este sistemul de numerație roman. Acest sistem de numerație utilizează următoarele simboluri:

Cifra	I	V	X	L	C	D	M
Valoare	1	5	10	50	100	500	1000

La formarea numerelor în sistemul de numerație roman se au în vedere următoarele reguli:

- Prezența consecutivă a mai multor cifre identice implică efectuarea sumei cifrelor respective.  
De exemplu: II=1+1=2; III=1+1+1=3; XX=10+10=20;  
CC=100+100=200; CCC=100+100+100=300;  
MM=1000+1000=2000.
- Prezența a două cifre diferite consecutive, din care pe prima poziție se găsește cifra mai mare implică efectuarea sumei cifrelor respective.  
De exemplu: VI=5+1=6; VII=5+1+1=7; LV=50+5=55;  
XIII=10+1+1+1=13; XXII=10+10+1+1=22; LX=50+10=60;  
CL=100+50=150.
- Prezența a două cifre diferite consecutive, din care pe prima poziție se găsește cifra mai mică implică efectuarea diferenței cifrelor respective.  
De exemplu: IV=5-1=4; IX=10-1=9; XL=50-10=40;  
CD=500-100=400; XC=100-10=90.

Principalele dezavantaje ale sistemelor de numerație nepoziționale, în particular al sistemului de numerație roman, sunt: scrierea greoaie a numerelor, în special a numerelor mari; posibilitatea reprezentării aceluiași număr prin mai multe succesiuni de simboluri, ceea ce poate crea numeroase confuzii. De exemplu, pentru exprimarea în sistemul de numerație roman a numărului 9 se pot folosi următoarele două reprezentări diferite: IX și VIII.

- **Sisteme de numerație poziționale.** Sunt sistemele de numerație pentru care valoarea unei anumite cifre din structura unui număr depinde în mod unic de poziția ocupată de acea cifră în numărul respectiv. Baza unui sistem de numerație pozițional reprezintă

numărul de elemente distincte cu ajutorul cărora se exprimă numerele în sistemul de numerație respectiv.

Reprezentative pentru sistemele de numerație poziționale sunt sistemul zecimal, sistemul binar, sistemul hexazecimal:

- Sistemul de numerație zecimal (baza 10) utilizează următoarele simboluri:

0	1	2	3	4	5	6	7	8	9
---	---	---	---	---	---	---	---	---	---

- Sistemul de numerație binar (baza 2) utilizează următoarele simboluri:

0	1
---	---

- Sistemul de numerație hexazecimal (baza 16) utilizează următoarele simboluri:

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

În general, se presupune că valorile numerice utilizate sunt exprimate în sistemul de numerație zecimal. În cazul aplicațiilor care operează cu numere exprimate în diferite sisteme de numerație, trebuie menționată în mod clar și baza sistemului de numerație respectiv. Pentru a specifica în care sistem de numerație este reprezentat un anumit număr, se pot folosi mai multe metode:

- La sfârșitul numărului se scrie un simbol special de identificare:

Pentru sistemul zecimal	d
Pentru sistemul binar	b
Pentru sistemul hexazecimal	h

- La sfârșitul numărului se scrie baza în care a fost scris numărul, între paranteze rotunde, sau ca indice inferior.

De exemplu, numărul 1101,101 poate avea diferite semnificații în funcție de sistemul de numerație în care se reprezintă:

- 1101,101d; 1101,101(10); 1101,101<sub>(10)</sub> - pentru cazul sistemului zecimal.
- 1101,1b; 1101,101(2); 1101,101<sub>(2)</sub> - pentru cazul sistemului binar.
- 1101,101h; 1101,101(16); 1101,101<sub>(16)</sub> - pentru cazul sistemului hexazecimal.

### 1.5.2. Exprimarea numerelor în diferite sisteme de numerație poziționale

Se consideră un numărul  $N$ , cu  $n$  cifre întregi și  $m$  cifre fracționare:

$$N = a_{n-1}a_{n-2} \dots a_1a_0, a_{-1}a_{-2} \dots a_{-m+1}a_{-m}$$

$$N = [N] + \{N\}, \text{ cu } 0 \leq \{N\} < 1$$

$$[N] = a_{n-1}a_{n-2} \dots a_1a_0; \{N\} = 0, a_{-1}a_{-2} \dots a_{-m+1}a_{-m}$$

Pentru numărul  $N$  astfel definit, cifra  $a_{n-1}$  reprezintă cifra cea mai semnificativă, iar cifra  $a_{-m}$  reprezintă cifra cea mai puțin semnificativă.

Numărul  $N$  se poate exprima în sistemul de numerație cu baza  $q$  cu ajutorul relațiilor următoare:

$$N_q = a_{n-1}q^{n-1} + a_{n-2}q^{n-2} + \dots + a_1q^1 + a_0q^0 + a_{-1}q^{-1} + a_{-2}q^{-2} + \dots + a_{-m}q^{-m} \Rightarrow N = \sum_{i=-m}^{n-1} a_i q^i$$

în care  $a_{n-1}, a_{n-2}, \dots, a_1, a_0 \in 0 \div q - 1$  reprezintă cele  $n$  cifre ale părții întregi, iar  $a_{-1}, a_{-2}, \dots, a_{-m+1}, a_{-m} \in 0 \div q - 1$  reprezintă cele  $m$  cifre ale părții fracționare pentru sistemul de numerație caracterizat prin baza  $q$ .

### Problema 1.1

Să se exprime numărul 1101,101 în sistemele de numerație cu baza 10, 2 și 16.

- $1101,101(10) = 1 \cdot 10^3 + 1 \cdot 10^2 + 0 \cdot 10^1 + 1 \cdot 10^0 + 1 \cdot 10^{-1} + 0 \cdot 10^{-2} + 1 \cdot 10^{-3}$
- $1101,101(2) = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3}$
- $1101,101(16) = 1 \cdot 16^3 + 1 \cdot 16^2 + 0 \cdot 16^1 + 1 \cdot 16^0 + 1 \cdot 16^{-1} + 0 \cdot 16^{-2} + 1 \cdot 16^{-3}$

### 1.5.3. Conversia unui număr dintr-un sistem de numerație oarecare în sistemul de numerație zecimal

Se consideră un număr oarecare  $N$  cu  $n$  cifre întregi și  $m$  cifre fracționare exprimat în sistemul de numerație caracterizat prin baza  $q$ :

$$N = a_{n-1}a_{n-2} \dots a_1a_0, a_{-1}a_{-2} \dots a_{-m+1}a_{-m}$$

$$N = [N] + \{N\}, \text{ cu } 0 \leq \{N\} < 1$$

$$[N] = a_{n-1}a_{n-2} \dots a_1a_0; \{N\} = 0, a_{-1}a_{-2} \dots a_{-m+1}a_{-m}$$

în care  $a_{n-1}, a_{n-2}, \dots, a_1, a_0 \in 0 \div q - 1$  reprezintă cele  $n$  cifre ale părții întregi, iar  $a_{-1}, a_{-2}, \dots, a_{-m+1}, a_{-m} \in 0 \div q - 1$  reprezintă cele  $m$  cifre ale părții fracționare pentru sistemul de numerație caracterizat prin baza  $q$ .

Exprimarea numărului astfel definit din baza  $q$  în baza 10 se face cu ajutorul relației următoare:

$$N_q = a_{n-1}q^{n-1} + a_{n-2}q^{n-2} + \dots + a_1q^1 + a_0q^0 + a_{-1}q^{-1} + a_{-2}q^{-2} + \dots + a_{-m}q^{-m}$$

### Problema 1.2

Să se convertească numărul 1101,101 din sistemele de numerație cu bazele 10, 2 și 16 în sistemul de numerație cu baza 10.

- $1101,101(10) = 1 \cdot 10^3 + 1 \cdot 10^2 + 0 \cdot 10^1 + 1 \cdot 10^0 + 1 \cdot 10^{-1} + 0 \cdot 10^{-2} + 1 \cdot 10^{-3} = 1101,101(10)$

- $1101,101(2) = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} = 13.625(10)$
- $1101,101(16) = 1 \cdot 16^3 + 1 \cdot 16^2 + 0 \cdot 16^1 + 1 \cdot 16^0 + 1 \cdot 16^{-1} + 0 \cdot 16^{-2} + 1 \cdot 16^{-3} = 4353,062744140625(10)$

#### 1.5.4. Conversia unui număr din sistemul de numerație zecimal într-un sistem de numerație oarecare

Se consideră un număr oarecare  $N$  cu  $n$  cifre întregi și  $m$  cifre fracționare exprimat în sistemul de numerație caracterizat prin baza 10:

$$N = a_{n-1}a_{n-2} \dots a_1a_0, a_{-1}a_{-2} \dots a_{-m+1}a_{-m}$$

$$N = [N] + \{N\}, \text{ cu } 0 \leq \{N\} < 1$$

$$[N] = a_{n-1}a_{n-2} \dots a_1a_0; \{N\} = 0, a_{-1}a_{-2} \dots a_{-m+1}a_{-m}$$

în care  $a_{n-1}, a_{n-2}, \dots, a_1, a_0 \in 0 \div 9$  reprezintă cele  $n$  cifre ale părții întregi, iar  $a_{-1}, a_{-2}, \dots, a_{-m+1}, a_{-m} \in 0 \div 9$  reprezintă cele  $m$  cifre ale părții zecimale corespunzătoare sistemului de numerație cu baza 10.

Conversia numărului  $N$  astfel definit într-un sistem de numerație caracterizat prin baza  $q$  se face în mod separat pentru partea întregă și partea zecimală a numărului:

- Conversia părții întregi a numărului  $N$  din baza 10 în baza  $q$  se realizează printr-un proces de împărțire succesivă pornind de la partea întregă a numărului. Astfel, în prima etapă, se împarte partea întregă a numărului la baza  $q$ , obținându-se un cât și un rest. Primul rest reprezintă cifra cea mai puțin semnificativă a numărului în baza  $q$ . În etapa a doua, se împarte câtul la baza  $q$ , obținându-se un nou cât și un nou rest. Împărțirile succesive continuă până când câtul obținut devine zero. Ultimul rest obținut reprezintă cifra cea mai semnificativă a numărul scris în baza  $q$ . Deci resturile obținute în urma împărțirilor succesive reprezintă cifrele numărului în baza  $q$ .
- Conversia părții zecimale a numărului  $N$  din baza 10 în baza  $q$  se realizează printr-un proces de înmulțire succesivă pornind de la partea zecimală a numărului. Astfel, în prima etapă se înmulțește partea zecimală a numărului cu baza  $q$  obținându-se un număr având, la rândul său o parte întregă și o parte zecimală. Partea întregă a acestui număr reprezintă cifra cea mai semnificativă a părții zecimale a numărului exprimat în baza  $q$ . În etapa a doua, se înmulțește partea zecimală a acestui număr cu baza  $q$ , obținându-se un nou număr. Înmulțirile succesive continuă până se obține partea fracționară nulă, sau un număr suficient de cifre, sau valorile obținute se repetă. Partea întregă a ultimului număr obținut reprezintă cifra cea mai puțin semnificativă a numărului scris în baza  $q$ . Prin urmare, părțile întregi obținute în urma înmulțirilor succesive reprezintă cifrele zecimale ale numărului în baza  $q$ .

### Problema 1.3

Să se transforme numărul  $13,625(10)$  în baza 2.

Se transformă mai întâi partea întreaga a numărului:

$$\begin{array}{rcllcl} & & & \text{cât} & \text{rest} \\ 13 & : & 2 & = & 6 & 1 \uparrow \\ 6 & : & 2 & = & 3 & 0 \\ 3 & : & 2 & = & 1 & 1 \\ 1 & : & 2 & = & \blacksquare 0 & 1 \end{array}$$

Se obține deci conversia părții întregi:  $13(10)=1101(2)$

Se transformă apoi partea zecimală  $0,625$  a numărului:

$$\begin{array}{rcllcl} 0,625 & \times & 2 & = & 1 & , & 25 \\ 0,25 & \times & 2 & = & 0 & , & 5 \\ 0,5 & \times & 2 & = & \downarrow 1 & , & \blacksquare 0 \end{array}$$

Se obține deci conversia părții zecimale:  $0,625(10)=0,101(2)$

Combinând cele două rezultate parțiale se obține conversia:

$$13,625(10)=1101,101(2)$$

#### 1.5.5. Operații aritmetice în sistem binar

Efectuarea operațiilor aritmetice asupra unor numere exprimate în diferite sisteme de numerație este posibilă doar după conversia celor două numere în același sistem de numerație. Între două numere exprimate în același sistem de numerație (de exemplu sistem binar) se pot efectua operațiile aritmetice de bază fără a fi necesară conversia numerelor în sistemul de numerație zecimal.

În cazul în care adunarea a două numere conduce la un rezultat mai mare decât baza de numerație a sistemului respectiv, atunci apare fenomenul de transport spre coloana din stânga, rezultând astfel incrementarea cu o unitate a acestei coloane. De exemplu adunarea în sistem binar a numerelor  $1+1$  conduce la cifra 0 și la transferul cifrei 1 spre stânga, deci în sistem binar se obține:  $1+1=10$ . Înmulțirea a două numere în sistem binar se efectuează ca și înmulțirea cu numere în baza 10 considerând ca regulă suplimentară faptul că înmulțirea cu 1 conduce la rezultatul 1, iar înmulțirea cu zero conduce la rezultatul 0. În tabelul 1.3 se prezintă regulile de efectuare a operațiilor aritmetice de adunare și înmulțire pentru sistemul de numerație binar:

**Tabel 1.3.** Operații aritmetice în sistem binar.

Adunare	0	1
0	0	1
1	1	10

Înmulțire	0	1
0	0	0
1	0	1

## 1.6. REPREZENTAREA NUMERELOR ÎN CALCULATOR

Reprezentarea numerelor în calculator se face folosind sistemul de numerație binar, pe un anumit număr de biți. Cu cât reprezentarea se face pe un număr mai mare de biți, cu atât numerele care se vor putea defini vor fi mai mari. Reprezentarea numerelor în calculator se face în mod diferit în funcție de natura numerelor respective: numere naturale, numere întregi cu sau fără semn, numere zecimale exprimate în virgulă fixă sau în virgulă mobilă.

### 1.6.1. Reprezentarea numerelor naturale

Reprezentarea numerelor naturale  $N \in \{0; 1; 2; 3; \dots\}$  se poate realiza folosind un număr fix de poziții, de obicei 8, 16, 32 sau 64 de poziții. Se definesc astfel reprezentările pe 8, 16, 32 sau 64 de biți, pentru care prezintă un deosebit interes determinarea valorii maxime a numărului care se poate reprezenta:

- Pentru cazul reprezentării pe 8 biți.  
Valoarea maximă a numărului în sistem binar care se poate reprezenta pe 8 biți este:  $N_{max}^8 = 11111111(2)$ .

	Biți							
	8	7	6	5	4	3	2	1
Numărul maxim în baza 2	1	1	1	1	1	1	1	1
Conversia în baza 10	$2^7$	$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$

Conversia acestui număr din baza 2 în baza 10 conduce la următorul rezultat:

$$11111111(2) = \sum_{i=0}^7 1 \cdot 2^i = 255$$

- Pentru cazul reprezentării pe 16 biți.  
Valoarea maximă a numărului în sistem binar care se poate reprezenta pe 16 biți este:  $N_{max}^{16} = 1111111111111111(2)$ .  
Conversia acestui număr din baza 2 în baza 10 conduce la următorul rezultat:

$$1111111111111111(2) = \sum_{i=0}^{15} 1 \cdot 2^i = 65535$$

- Pentru cazul reprezentării pe 32 biți.  
Valoarea maximă a numărului în sistem binar care se poate reprezenta pe 32 biți este:  $N_{max}^{32} = a_{31}a_{30} \dots a_1a_0$ ,  $a_i = 1$ ,  $\forall i = 0 \div 31$ .  
Conversia acestui număr din baza 2 în baza 10 conduce la următorul rezultat:

$$N_{max}^{32} = \sum_{i=0}^{31} 1 \cdot 2^i = 4294967295$$

- Pentru cazul reprezentării pe 64 biți.  
Valoarea maximă a numărului în sistem binar care se poate reprezenta pe 64 biți este:

$$N_{max}^{64} = a_{63}a_{62} \dots a_1a_0, a_i = 1, \forall i = 0 \div 63.$$

Conversia acestui număr din baza 2 în baza 10 conduce la următorul rezultat:

$$N_{max}^{64} = \sum_{i=0}^{63} 1 \cdot 2^i = 18446744073709550000$$

- Pentru cazul general al reprezentării pe  $n$  biți.  
Valoarea maximă a numărului în sistem binar care se poate reprezenta pe  $n$  biți este:

$$N_{max}^n = a_{n-1}a_{n-2} \dots a_1a_0, a_i = 1, \forall i = 0 \div n - 1.$$

Conversia acestui număr din baza 2 în baza 10 conduce la următorul rezultat:

$$N_{max}^n = \sum_{i=0}^{n-1} 1 \cdot 2^i = 2^n - 1$$

Aplicarea acestei relații generale conduce la rezultatele deduse anterior:

n	Relația	$N_{max}^n$
8	$2^n - 1$	255
16		65535
32		4294967295
64		18446744073709550000

### 1.6.2. Reprezentarea numerelor întregi cu semn

Se consideră un număr  $N$  oarecare exprimat în sistemul binar de numerație.

$$N = a_{n-1}a_{n-2} \dots a_1a_0, a_i \in \{0;1\}, \forall i = 0 \div n - 1$$

Pentru reprezentarea în sistemul de numerație binar al numerelor întregi cu semn, primul bit (bitul de pe poziția cea mai semnificativă) se utilizează pentru specificarea semnului. Astfel, pentru numere pozitive ( $N > 0$ ), primul bit are valoarea 0, iar pentru numere negative ( $N < 0$ ), primul bit are valoarea 1.

Reprezentarea numerelor întregi cu semn se realizează prin mai multe metode:

- **Reprezentarea prin mărime și semn (cod binar direct)** constă în reprezentarea semnului pe cel mai semnificativ bit (0 pentru numere pozitive și 1 pentru numere negative), după care urmează cifrele



numărului propriu-zis. De exemplu, reprezentarea numerelor 1011(2) și -1011(2) pe 8 biți se face conform tabelelor:

Numărul: 1011(2)	Semn	Biți						
	8	7	6	5	4	3	2	1
Reprezentare în cod direct	0	0	0	0	1	0	1	1
Conversia în baza 10		$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$

Numărul: -1011(2)	Semn	Biți						
	8	7	6	5	4	3	2	1
Reprezentare în cod direct	1	0	0	0	1	0	1	1
Conversia în baza 10		$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$

Intervalul de valori care se poate reprezenta pe  $n$  biți folosind această metodă este definit prin limitele:

$$N_{max}^n = \sum_{i=0}^{n-2} 1 \cdot 2^i = 2^{n-1} - 1$$

Aplicarea acestei relații generale conduce la următoarele rezultate:

$n$	Relația	$\{N_{min}^n, N_{max}^n\}$
8	$2^n - 1$	$\{-127; 127\}$
16		$\{-32767; 32767\}$
32		$\{-2147483647; 2147483647\}$
64		$\{-9223372036854776000; 9223372036854776000\}$

- **Reprezentarea prin complement față de 1 (cod binar invers)** coincide cu reprezentarea în cod direct pentru numerele pozitive. Pentru numerele negative, reprezentarea în cod invers se obține prin inversarea cifrelor din reprezentarea în cod direct a modulului numărului respectiv. De exemplu, reprezentarea numerelor 1011(2) și -1011(2) pe 8 biți se face conform tabelelor:

Numărul: 1011(2)	Semn	Biți						
	8	7	6	5	4	3	2	1
Reprezentare în cod direct	0	0	0	0	1	0	1	1

Numărul: -1011(2)	Semn	Biți						
	8	7	6	5	4	3	2	1
Reprezentare în cod direct a modulului numărului	0	0	0	0	1	0	1	1
Reprezentare în cod invers	1	1	1	1	0	1	0	0

- **Reprezentarea prin complement față de 2 (cod binar complementar)** coincide cu reprezentarea în cod direct pentru numerele pozitive. Pentru numerele negative, reprezentarea în cod complementar se obține prin adăugarea cifrei 1 la numărul obținut ca reprezentare în cod invers a modulului numărului respectiv. De exemplu, reprezentarea numerelor 1011(2) și -1011(2) pe 8 biți se face conform tabelelor:

Numărul: 1011(2)	Semn	Biți						
	8	7	6	5	4	3	2	1
Reprezentare în cod direct	0	0	0	0	1	0	1	1

Numărul: -1011(2)	Semn	Biți						
	8	7	6	5	4	3	2	1
Reprezentare în cod direct a modulului numărului	0	0	0	0	1	0	1	1
Reprezentare în cod invers	1	1	1	1	0	1	0	0
Reprezentare în cod complementar	1	1	1	1	0	1	0	1

### 1.6.3. Reprezentarea numerelor zecimale în virgulă fixă

Reprezentarea numerelor zecimală în virgulă fixă se realizează ca în cazul numerelor întregi cu semn, cu deosebirea că, în acest caz, există o virgulă virtuală într-o poziție bine determinată. Prin convenție, virgula virtuală poate fi plasată în următoarele poziții particulare (pentru o reprezentare pe 8 biți):

- Dacă virgula este plasată imediat după bitul de semn, atunci se obține cazul particular al reprezentării numerelor fracționare subunitare.

Numărul: 0,1001011(2)	Semn	Biți						
	8	7	6	5	4	3	2	1
Reprezentare în cod direct	0	1	0	0	1	0	1	1
Conversia în baza 10		$2^{-1}$	$2^{-2}$	$2^{-3}$	$2^{-4}$	$2^{-5}$	$2^{-6}$	$2^{-7}$

- Dacă virgula virtuală este plasată imediat în dreapta primului bit (după cifra cea mai puțin reprezentativă), atunci se obține cazul particular al reprezentării numerelor întregi.

Numărul: 1001011(2)	Semn	Biți						
	8	7	6	5	4	3	2	1
Reprezentare în cod direct	0	1	0	0	1	0	1	1
Conversia în baza 10		$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$

- Dacă virgula virtuală este plasată între doi biți oarecare din structura numărului (de exemplu între biții 3 și 4), atunci se obține o reprezentare a numerelor zecimale cu 4 cifre pentru partea întregă și 3 cifre pentru partea fracționară.

Numărul: 1001,011(2)	Semn	Biți						
	8	7	6	5	4	3	2	1
Reprezentare în cod direct	0	1	0	0	1	0	1	1
Conversia în baza 10		$2^3$	$2^2$	$2^1$	$2^0$	$2^{-1}$	$2^{-2}$	$2^{-3}$

### Observația 1

În cazul reprezentării unor numere cu mai puține cifre ale părții întregi sau ale părții zecimale decât numărul posibil stabilit conform convenției de reprezentare (4 cifre pentru partea întregă și 3 cifre pentru partea fracționară, ca în exemplul anterior) atunci se procedează la adăugarea unor zerouri suplimentare la stânga, sau la dreapta numărului inițial până la completarea numărului de biți disponibili. Prin adăugarea de zerouri suplimentare nu se modifică numărul inițial:

- În cazul numărului 101,11(2) trebuie adăugate zerouri suplimentare atât în zona părții întregi (un zero suplimentar la stânga), cât și în zona părții fracționare (un zero suplimentar la dreapta).

Biți semnificativi	8	7	6	5	4	3	2	1
Numărul inițial	Bit de semn		1	0	1	1	1	
Numărul rezultat prin adăugarea zerourilor suplimentare		0	1	0	1	1	1	0
		Zero suplimentar						Zero suplimentar
Conversia în baza 10		$2^3$	$2^2$	$2^1$	$2^0$	$2^{-1}$	$2^{-2}$	$2^{-3}$

## Observația 2

În cazul reprezentării unor numere cu mai multe cifre ale părții întregi sau ale părții zecimale decât numărul posibil stabilit conform convenției de reprezentare (4 cifre pentru partea întregă și 3 cifre pentru partea fracționară, ca în exemplul anterior) atunci se procedează la eliminarea unor cifre din zona părții întregi, sau a părții fracționare, după caz. Dacă pentru partea fracționară, această metodă conduce la eliminarea celor mai puțin semnificative cifre, în cazul părții întregi, metoda conduce la eliminarea celor mai semnificative cifre:

- În cazul reprezentării numărului  $1001,01101(2)$  se observă că trebuie eliminate ultimele două cifre ale părții fracționare (01) din zona cifrelor celor mai puțin semnificative, rezultând astfel doar aproximarea numărului inițial. În sistemul de numerație zecimal, numărul inițial este  $1001,01101(2)=9,40625(10)$ , în timp ce numărul rezultat prin eliminarea celor mai puțin semnificativi biți este  $1001,011(2)=9,375(10)$  (aproximarea numărului inițial).

Biți semnificativi	8	7	6	5	4	3	2	1		
Numărul inițial	Bit de semn	1	0	0	1	0	1	1	0	1
Numărul rezultat prin aproximare		1	0	0	1	0	1	1	Cifre eliminate	
Conversia în baza 10		$2^3$	$2^2$	$2^1$	$2^0$	$2^{-1}$	$2^{-2}$	$2^{-3}$	$2^{-4}$	$2^{-5}$

- În cazul reprezentării numărul  $101001,011(2)$  trebuie eliminate primele două cifre ale părții întregi (10) din zona cifrelor celor mai semnificative, rezultând astfel alterarea numărului inițial. În sistemul de numerație zecimal, numărul inițial este  $101001,011(2)=41,375(10)$ , în timp ce numărul rezultat prin eliminarea celor mai semnificativi biți este  $1001,011(2)=9,375(10)$  (alterarea numărului inițial).

Biți semnificativi	8			7	6	5	4	3	2	1
Numărul inițial	Bit de semn	1	0	1	0	0	1	0	1	1
Numărul rezultat prin aproximare		Cifre eliminate		1	0	0	1	0	1	1
Conversia în baza 10		$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$	$2^{-1}$	$2^{-2}$	$2^{-3}$

#### 1.6.4. Reprezentarea numerelor reale în virgulă mobilă

Metoda de reprezentare a numerelor zecimale în virgulă mobilă permite specificarea numerelor având oricâte cifre ale părții întregi și ale părții fracționare pe un anumit număr de biți (8, 16, 32 sau 64) prin eliminarea doar a cifrelor celor mai puțin semnificative, rezultând astfel doar aproximarea numărului inițial și nu alterarea acestuia.

Reprezentarea numerelor reale în virgulă mobilă se bazează pe faptul că orice număr real poate fi scris într-un sistem de numerație caracterizat prin baza  $q$  (de obicei 10 sau 2) sub următoarea formă:

$$N = (-1)^{sn} \cdot M \cdot q^{(-1)^{se}e}$$

în care:  $sn$  este semnul numărului,  $M$  este mantisa,  $e$  este exponentul, iar  $se$  este semnul exponentului.

#### Problema 1.4

Exprimarea numerelor reale în baza 10 ( $q=10$ ):

- $14,25(10) = (-1)^0 \cdot 1,425 \cdot 10^1$ ;  $sn=0$ ,  $se=0$ ,  $M=1,425$ ,  $e=1$ .
- $-7,15(10) = (-1)^1 \cdot 7,15 \cdot 10^0$ ;  $sn=1$ ,  $se=0$ ,  $M=7,15$ ,  $e=0$ .
- $120,35(10) = (-1)^0 \cdot 1,2035 \cdot 10^2$ ;  $sn=0$ ,  $se=0$ ,  $M=1,2035$ ,  $e=2$ .
- $-6320,75(10) = (-1)^1 \cdot 6,32075 \cdot 10^3$ ;  $sn=-1$ ,  $se=0$ ,  $M=6,32075$ ,  $e=3$ .
- $0,0125(10) = (-1)^0 \cdot 1,25 \cdot 10^{-2}$ ;  $sn=0$ ,  $se=1$ ,  $M=1,25$ ,  $e=-2$ .
- $-0,0025(10) = (-1)^1 \cdot 2,5 \cdot 10^{-3}$ ;  $sn=1$ ,  $se=1$ ,  $M=2,5$ ,  $e=-3$ .

Exprimarea numerelor reale în baza 2 ( $q=2$ ):

- $14,25_{10} = (-1)^0 \cdot 1110,01_2 = (-1)^0 \cdot 1,11001_2 \cdot 2^{3_{10}} = (-1)^0 \cdot 1,11001_2 \cdot 2^{11_2}$ ;  $sn=0$ ,  $se=0$ ,  $M=1,11001$ ,  $e=11$ .
- $-7,15_{10} = (-1)^1 \cdot 111,001001_2 = (-1)^1 \cdot 1,11001001_2 \cdot 2^{2_{10}} = (-1)^1 \cdot 1,11001001_2 \cdot 2^{10_2}$ ;  $sn=1$ ,  $se=0$ ,  $M=1,11001001$ ,  $e=10$ .
- $0,0125_{10} = (-1)^0 \cdot 0,00000011001_2 = (-1)^0 \cdot 1,1001_2 \cdot 2^{-7_{10}} = (-1)^0 \cdot 1,1001_2 \cdot 2^{-11_2}$ ;  $sn=0$ ,  $se=1$ ,  $M=1,1001$ ,  $e=-11$ .

#### Observația 3

Reprezentarea oricărui număr zecimal în virgulă mobilă folosind sistemul de numerație binar pe un număr dat  $n$  de biți, presupune alocarea de biți pentru:

- Reprezentarea semnului numărului ( $sn$ ), se realizează pe primul bit semnificativ.
- Reprezentarea semnului exponentului ( $se$ ), se realizează pe al doilea bit semnificativ, după care urmează reprezentarea exponentului  $e$ .

Pentru evitarea folosirii unui bit special pentru semnul exponentului (câștigând astfel un bit pentru exponent sau mantisă), la valoarea exponentului se adună un număr dependent de numărul de biți pe care se face reprezentarea, obținându-se astfel o valoare totdeauna pozitivă denumită caracteristica exponentului ( $C$ ).

- Reprezentarea mantisei,  $M$ . Pentru a câștiga un bit la reprezentarea mantisei se are în vedere doar partea sa fracționară ( $m$ ), datorită faptului că partea întreagă este totdeauna 1. Convențional, prin mantisă se înțelege doar partea fracționară.

Rezultă așadar, următoarea formă utilizată pentru reprezentarea în sistemul de numerație binar a numerelor zecimale în virgulă mobilă:

$$N_2 = (-1)^{sn} \cdot 1, m \cdot 2^C$$

în care:  $sn$  este semnul numărului,  $m$  este partea fracționară a mantisei (convențional numită mantisă), iar  $C$  este caracteristica exponentului.

Pentru calculul caracteristicii exponentului se utilizează relația:

$$C = e + 2^{N_{be}-1} - 1$$

în care  $N_{be}$  reprezintă numărul de biți alocați pentru reprezentarea exponentului, care depinde de numărul de biți pe care se realizează întreaga reprezentare a numărului dat.

Valoarea maximă a caracteristicii exponentului se determină în funcție de numărul de biți  $n$  alocați pentru exprimarea exponentului prin:

$$C_{max} = 2^n - 1$$

Valoarea maximă a exponentului se calculează apoi cu relația:

$$e_{max} = C_{max} - (2^{N_{be}-1} - 1)$$

În tabelul 1.4 sunt prezentate principalele caracteristici necesare pentru reprezentarea numerelor zecimale în virgulă mobilă în funcție de numărul de biți maxim disponibili pentru reprezentările pe 8, 16, 32 și 64 de biți.

**Tabel 1.4.** Reprezentarea numerelor zecimale în virgulă mobilă.

Număr total de biți	Biți pentru semn	Biți pentru exponent	Biți pentru mantisă	$2^{N_{be}-1} - 1$	$C$	$C_{max}$	$e_{max}$
8	1	3	4	3	e+3	7	4
16	1	5	10	15	e+15	31	16
32	1	8	23	127	e+127	255	128
64	1	11	52	1023	e+1023	2047	1024

### Problema 1.5

Să se reprezinte în virgulă mobilă pe 8, 16 și 32 de biți numărul zecimal  $7,5_{10}$ .

- Folosind reprezentarea pe 8 biți se obține:

$$7,5_{10}=111,1_2=(-1)^0 \cdot 1,111 \cdot 2^2=(-1)^0 \cdot 1,111 \cdot 2^{2+3}=$$
$$=(-1)^0 \cdot 1,111 \cdot 2^5=(-1)^0 \cdot 1,111 \cdot 2^{101}$$

sn	C				m			
0	1	0	1	1	1	1	0	

- Folosind reprezentarea pe 16 biți se obține:

$$7,5_{10}=111,1_2=(-1)^0 \cdot 1,111 \cdot 2^2=(-1)^0 \cdot 1,111 \cdot 2^{2+15}=$$
$$=(-1)^0 \cdot 1,111 \cdot 2^{17}=(-1)^0 \cdot 1,111 \cdot 2^{10001}$$

sn	C						m									
0	1	0	0	0	1	1	1	1	0	0	0	0	0	0	0	

- Folosind reprezentarea pe 32 biți se obține:

$$7,5_{10}=111,1_2=(-1)^0 \cdot 1,111 \cdot 2^2=(-1)^0 \cdot 1,111 \cdot 2^{2+127}=$$
$$=(-1)^0 \cdot 1,111 \cdot 2^{129}=(-1)^0 \cdot 1,111 \cdot 2^{10000001}$$

### Problema 1.6

Să se reprezinte în virgulă mobilă pe 8, 16 și 32 biți numărul - $3,0625_{10}$ .

- Folosind reprezentarea pe 8 biți se obține:

$$-3,0625_{10}=-11,0001_2=(-1)^1 \cdot 1,10001 \cdot 2^1=(-1)^1 \cdot 1,10001 \cdot 2^{1+3}=$$
$$=(-1)^1 \cdot 1,10001 \cdot 2^4=(-1)^1 \cdot 1,10001 \cdot 2^{100}$$

sn	C				m			
0	1	0	0	1	0	0	0	

- Folosind reprezentarea pe 16 biți se obține:

$$-3,0625_{10}=-11,0001_2=(-1)^1 \cdot 1,10001 \cdot 2^1=(-1)^1 \cdot 1,10001 \cdot 2^{1+15}=$$
$$=(-1)^1 \cdot 1,10001 \cdot 2^{16}=(-1)^1 \cdot 1,10001 \cdot 2^{10000}$$

sn	C						m									
0	1	0	0	0	0	1	0	0	0	1	0	0	0	0	0	

- Folosind reprezentarea pe 32 biți se obține:

$$-3,0625_{10}=-11,0001_2=(-1)^1 \cdot 1,10001 \cdot 2^1=(-1)^1 \cdot 1,10001 \cdot 2^{1+127}=$$
$$=(-1)^1 \cdot 1,10001 \cdot 2^{128}=(-1)^1 \cdot 1,10001 \cdot 2^{10000000}$$

## 1.7. ERORI ALE CALCULELOR NUMERICE

Rezolvarea problemelor din domeniul aplicativ al științelor tehnice implică în general utilizarea unor algoritmi care să permită, ca pornind de la modelul sistemului fizic supus analizei, prin utilizarea unor metode de calcul adecvate, să se ajungă la un rezultat calitativ sau cantitativ. Indiferent de calitatea procesului de modelare, de metodele de calcul și de mijloacele tehnice utilizate pentru implementarea acestora, rezultatele obținute (notate convențional cu  $x$ ) vor reprezenta în general doar aproximări ale valorilor adevărate ale mărimilor respective (notate convențional cu  $X$ ). Diferența dintre rezultatele obținute în urma calculelor și valorile adevărate (necunoscute) ale mărimilor respective furnizează informații despre erorile întregului proces.

### 1.7.1. Clasificarea erorilor

Considerând cazul general al unei probleme aplicative care implică atât efectuarea unui proces de măsurare asupra unei mărimi fizice, cât și efectuarea unor analize statistice și calcule numerice, erorile care afectează rezultatele obținute se clasifică în:

- Erori de măsurare, [21]. Aceste erori sunt specifice procesului de măsurare și pot fi:
  - Erori de model, generate de idealizarea modelului fizic supus analizei. Aceste erori apar datorită introducerii unor ipoteze simplificatoare de tipul: sistem fizic incompresibil, nevâscos, omogen, etc.
  - Erori instrumentale, generate de mijlocul de măsurare utilizat. Aceste erori apar datorită utilizării unor mijloace de măsurare neconforme sau datorită operării lor defectuoase.
  - Erori de interacțiune, generate de acțiunea perturbatoare reciprocă dintre mijlocul de măsurare și sistemul fizic supus analizei.
  - Erori generate de factorii de mediu care influențează caracteristicile metrologice ale mijlocului de măsurare dar și parametrii sistemului fizic supus analizei.
- Erori de calcul [1, 2, 7, 8, 9]. Aceste erori sunt specifice prelucrării statistice și numerice a datelor experimentale și pot fi:
  - Erori inerente. Sunt erorile incluse în datele inițiale ale algoritmului de calcul și sunt generate, în general, de erorile de măsurare (în cazul în care datele inițiale reprezintă mărimi măsurate), sau de utilizarea unor coeficienți aproximativi.
  - Erori de metodă (eroari de trunchiere). Aceste erori sunt generate de metoda de calcul utilizată pentru rezolvarea unei anumite probleme, metoda care ar putea conduce la un



rezultat exact doar dacă s-ar efectua un număr infinit de iterații. În realitate însă, procesul iterativ de calcul conține doar un număr finit de etape, fapt care conduce la apariția erorilor de trunchiere și deci la obținerea unui rezultat inexact. Finalizarea calculului iterativ este determinată fie de impunerea unui anumit număr de etape, caz în care rezultă o anumită valoare a erorii de trunchiere, fie de impunerea unei anumite valori a erorii, caz în care rezultă numărul necesar de iterații. În ambele cazuri, eroarea de trunchiere poate fi controlată, singurele limitări fiind determinate de timpul de calcul și de resursele hardware necesare pentru efectuarea calculelor.

- Erori de rotunjire. Aceste erori sunt determinate de modul de reprezentare a numerelor în calculator și apar atunci când numărul de cifre semnificative ale numărului de reprezentat este mai mare decât numărul finit de biți disponibili (8, 16, 32, 64), caz în care se procedează, în mod inevitabil, la eliminarea celor mai puțin semnificative cifre ale numărului, deci la rotunjirea sa.

Din punct de vedere al modului de exprimare matematică, erorile de calcul se clasifică în, [21]:

- Eroarea absolută (expresie dimensională), definită prin diferența dintre valoarea aproximativă  $x$  și valoarea reală  $X$  a mărimii:

$$e_x = x - X$$

În general, valoarea reală  $X$  este necunoscută, prin urmare nici eroarea absolută nu se poate determina. În acest caz se consideră o valoare limită a erorii absolute  $e_x^*$  definită prin inegalitatea:

$$e_x \leq e_x^*$$

Considerând exprimarea erorii absolute ca valoare absolută a diferenței dintre valoarea aproximativă  $x$  și valoarea reală  $X$  a mărimii, se obține:

$$e_x = |x - X| \leq e_x^* \Rightarrow x - e_x^* \leq X \leq x + e_x^*$$

ceea ce reprezintă o estimare a valorii reale  $X$  a mărimii.

- Eroarea relativă (expresie adimensională), definită prin raportul dintre eroarea absolută  $e_x$  și valoarea reală  $X$  a mărimii:

$$\varepsilon_x = \frac{e_x}{X} = \frac{x - X}{X}$$

Considerând exprimarea erorii absolute ca valoare absolută a diferenței dintre valoarea aproximativă  $x$  și valoarea reală  $X$  a mărimii, eroarea relativă se poate exprima în procente prin relația:

$$\varepsilon_x = \frac{|x - X|}{X} \cdot 100 [\%]$$

### 1.7.2. Propagarea erorilor

Se consideră că într-o anumită etapă a unui algoritm se obțin două rezultate numerice:

$$x = e_x + X \text{ și } y = e_y + Y$$

cu erorile relative corespunzătoare  $\varepsilon_x$  și  $\varepsilon_y$ . Dacă în următoarele etape ale algoritmului se impune efectuarea unor operații aritmetice între cele două rezultate numerice  $x$  și  $y$ , atunci erorile individuale absolute și relative ale celor două rezultate vor influența într-o anumită măsură rezultatul final al operației aritmetice. În cazul repetării calculului numeric se observă apariția unui proces de propagare a erorilor de la rezultatele numerice inițiale până la rezultatul numeric final, astfel încât, se poate ajunge în anumite situații la rezultate finale complet eronate.

Analiza propagării erorilor se face pentru fiecare operație aritmetică, atât pentru erorile absolute cât și pentru erorile relative. În cazul operațiilor aritmetice de adunare, scădere, înmulțire și împărțire, analiza propagării erorilor conduce la următoarele concluzii:

- Adunarea.

$$e_{x+y} = (x + y) - (X + Y) = (x - X) + (y - Y) = e_x + e_y$$

Eroarea absolută  $e_{x+y}$  a sumei celor două valori numerice  $x$  și  $y$  este egală cu suma erorilor absolute  $e_x$  și  $e_y$  ale valorilor numerice individuale.

$$\varepsilon_{x+y} = \frac{e_{x+y}}{X+Y} = \frac{e_x + e_y}{X+Y} = \frac{e_x}{X+Y} + \frac{e_y}{X+Y} \Rightarrow$$

$$\varepsilon_{x+y} = \frac{e_x}{X+Y} \varepsilon_x + \frac{e_y}{X+Y} \varepsilon_y$$

Eroarea relativă  $\varepsilon_{x+y}$  a sumei celor două valori numerice  $x$  și  $y$  este egală cu suma ponderată a erorilor relative  $\varepsilon_x$  și  $\varepsilon_y$  ale valorilor numerice individuale.

- Scăderea.

$$e_{x-y} = (x - y) - (X - Y) = (x - X) - (y - Y) = e_x - e_y$$

Eroarea absolută  $e_{x-y}$  a diferenței celor două valori numerice  $x$  și  $y$  este egală cu diferența erorilor absolute  $e_x$  și  $e_y$  ale valorilor numerice individuale.

$$\varepsilon_{x-y} = \frac{e_{x-y}}{X-Y} = \frac{e_x - e_y}{X-Y} = \frac{e_x}{X-Y} - \frac{e_y}{X-Y} \Rightarrow$$

$$\varepsilon_{x-y} = \frac{e_x}{X-Y} \varepsilon_x - \frac{e_y}{X-Y} \varepsilon_y$$

Eroarea relativă  $\varepsilon_{x-y}$  a diferenței celor două valori numerice  $x$  și  $y$  este egală cu diferența ponderată a erorilor relative  $\varepsilon_x$  și  $\varepsilon_y$  ale valorilor numerice individuale.

- Înmulțirea.

$$e_{x \cdot y} = (x \cdot y) - (X \cdot Y) = (e_x + X) \cdot (e_y + Y) - X \cdot Y \cong e_x Y + e_y X$$

în care s-a neglijat produsul erorilor absolute  $e_x e_y \cong 0$ .

Eroarea absolută  $e_{x+y}$  a sumei celor două valori numerice  $x$  și  $y$  este egală cu suma erorilor absolute  $e_x$  și  $e_y$  ale valorilor numerice individuale.

$$\varepsilon_{x \cdot y} = \frac{e_{x \cdot y}}{X \cdot Y} = \frac{e_x Y + e_y X}{X \cdot Y} = \frac{e_x}{X} + \frac{e_y}{Y} = \varepsilon_x + \varepsilon_y$$

Eroarea relativă  $\varepsilon_{x \cdot y}$  a produsului celor două valori numerice  $x$  și  $y$  este egală cu suma erorilor relative  $\varepsilon_x$  și  $\varepsilon_y$  ale valorilor numerice individuale.

- Împărțirea.

$$e_{x/y} = \frac{x}{y} - \frac{X}{Y} = \frac{e_x + X}{e_y + Y} - \frac{X}{Y} = \frac{X \cdot (e_x/X + 1)}{Y \cdot (e_y/Y + 1)} - \frac{X}{Y} = \frac{X}{Y} \cdot \left( \frac{e_x/X + 1}{e_y/Y + 1} - 1 \right)$$

$$= \frac{X}{Y} \cdot \left( \frac{e_x}{X} - \frac{e_y}{Y} \right) \cdot \frac{1}{1 + e_y/Y} \cong \frac{X}{Y} \cdot \left( \frac{e_x}{X} - \frac{e_y}{Y} \right)$$

în care s-au considerat doar primii doi termeni din dezvoltarea în serie Taylor:

$$\frac{1}{1 + e_y/Y} = 1 - \frac{e_y}{Y} + \frac{e_y^2}{Y^2} - \dots$$

și în plus s-au neglijat termenii  $e_x e_y \cong 0$  și  $e_y^2 \cong 0$ .

$$\varepsilon_{x/y} = \frac{e_{x/y}}{X/Y} = \frac{\frac{X}{Y} \cdot \left( \frac{e_x}{X} - \frac{e_y}{Y} \right)}{X/Y} = \frac{e_x}{X} - \frac{e_y}{Y} = \varepsilon_x - \varepsilon_y$$

Eroarea relativă  $\varepsilon_{x/y}$  a câtului celor două valori numerice  $x$  și  $y$  este egală cu diferența erorilor relative  $\varepsilon_x$  și  $\varepsilon_y$  ale valorilor numerice individuale.

Diminuarea influenței erorilor de calcul asupra unui rezultat final se poate obține prin proiectarea unui algoritm adecvat problemei de rezolvat, prin utilizarea unor resurse software și hardware performante, prin alocarea unor timpi de calcul corespunzători, precum și prin utilizarea unor resurse umane cu abilități și competențe specifice în domeniu.

## BIBLIOGRAFIE

1. Berbente C., Mitran S., Zancu S., Metode numerice, Editura Tehnică, București, 1998.
2. Brătianu C., Bostan V., Cojocia L., Negreanu G.P., Metode numerice, Editura Tehnică, București, 1996.
3. Ciureanu S.A., Arhitecture des Ordinateurs, Ed. Printech, București, 2008.
4. DEX Online, (Dicționar explicativ al limbii române online), Informatica, <http://dexonline.ro/definitie/informatica>, accesat la data 22.01.2014.
5. IEEE Standards Association, <http://standards.ieee.org/findstds/standard/1541-2002.html>, accesat la data 22.01.2014.
6. International Organization of Standardization (ISO), [http://www.iso.org/iso/home/store/catalogue\\_tc/catalogue\\_detail.htm?csnumber=31898](http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=31898), accesat la data 22.01.2014.
7. Kilyeni S., Metode numerice. Algoritme. Programe de calcul. Aplicații în energetică, Editura Orizonturi Universitare, Timișoara, 2004.
8. Larionescu D., Metode numerice, Editura Tehnică, București, 1989.
9. Rusu I., Metode numerice- Algoritmi în limbaj C, 2006 (online), [http://derivat.ro/cursuri/electronica/an2/an2\\_derivat.ro\\_metode-numerice\\_Carte%20MN.pdf](http://derivat.ro/cursuri/electronica/an2/an2_derivat.ro_metode-numerice_Carte%20MN.pdf), accesat la data 24.01.2014.
10. Slușanschi E., ș.a., Arhitectura sistemelor de calcul, Ed. Printech, București, 2010.
11. Wikipedia, Artificial Intelligence, [http://en.wikipedia.org/wiki/Artificial\\_intelligence](http://en.wikipedia.org/wiki/Artificial_intelligence), accesat la 25.08.2014.
12. Wikipedia, Boolean Algebra, [http://en.wikipedia.org/wiki/Boolean\\_algebra](http://en.wikipedia.org/wiki/Boolean_algebra), accesat la 25.08.2014.
13. Wikipedia, Computer Architecture, [http://en.wikipedia.org/wiki/Computer\\_architecture](http://en.wikipedia.org/wiki/Computer_architecture), accesat la 25.08.2014.
14. Wikipedia, Flynn's Taxonomy, [http://en.wikipedia.org/wiki/Flynn%27s\\_taxonomy](http://en.wikipedia.org/wiki/Flynn%27s_taxonomy), accesat la data 22.01.2014.
15. Wikipedia, Harvard Architecture, [http://en.wikipedia.org/wiki/Harvard\\_architecture](http://en.wikipedia.org/wiki/Harvard_architecture), accesat la data 22.01.2014.
16. Wikipedia, Information Technology, [http://en.wikipedia.org/wiki/Information\\_technology](http://en.wikipedia.org/wiki/Information_technology), accesat la 22.01.2014.
17. Wikipedia, Information and Communication Technology, [http://en.wikipedia.org/wiki/Information\\_and\\_communications\\_technology](http://en.wikipedia.org/wiki/Information_and_communications_technology), accesat la 22.01.2014.
18. Wikipedia, John von Neumann Architecture, [http://en.wikipedia.org/wiki/John\\_von\\_Neumann](http://en.wikipedia.org/wiki/John_von_Neumann), accesat la data 22.01.2014.
19. Wikipedia, Numeral System, [http://en.wikipedia.org/wiki/Numeral\\_system](http://en.wikipedia.org/wiki/Numeral_system), accesat la 25.08.2014.
20. Wikipedia, Usage Share of Operating Systems, [http://en.wikipedia.org/wiki/Usage\\_share\\_of\\_operating\\_systems](http://en.wikipedia.org/wiki/Usage_share_of_operating_systems), accesat la data 25.08.2014.
21. Zahariea D., Măsurări hidraulice, Rotaprint, Universitatea Tehnică „Gheorghe Asachi”, Iași, 1999.

## CAPITOLUL 2

### MATLAB-PREZENTARE GENERALĂ

#### 2.1. ISTORIC. CARACTERISTICI GENERALE

MATLAB (Matrix Laboratory) este un limbaj de programare dezvoltat inițial de matematicianul programator Cleve Moler [19], de la Universitatea din New Mexico, SUA la sfârșitul anilor '70. În anul 1984, Cleve Moler, împreună cu Jack Little [20], au rescris limbajul de programare MATLAB în C și au înființat compania MathWorks [21], pentru a produce și distribui limbajul de programare MATLAB.

MATLAB este un limbajul de programare orientat pe efectuarea de calcule numerice și simbolice specifice diferitelor domenii ale științelor ingineresti, precum și pe modelarea și simularea sistemelor dinamice de tip general (implementarea schemelor bloc dezvoltate cu limbajul de programare Simulink, din 1990 [16, 17, 23]) și a sistemelor fizice particulare (implementarea schemelor funcționale dezvoltate cu limbajul de programare Simscape, din 2008 [15, 22]).

Principalele momente ale evoluției limbajului de programare MATLAB sunt, [29]:

- Prima versiune comercială a limbajului de programare MATLAB (MATLAB 1.0) a apărut în anul 1984, imediat după înființarea companiei MathWorks.
- În anul 1990 a fost lansată versiunea MATLAB 3.5 pentru sistemul de operare MS-DOS care conținea și prima versiune a limbajului de programare Simulink.
- În anul 1994 a fost lansată versiunea MATLAB 4.2c (R7) pentru sistemul de operare Microsoft Windows.
- În anul 1998, odată cu versiunea MATLAB 5.2 (R10) a fost lansat și limbajul de programare Simulink destinat modelării și simulării sistemelor dinamice.
- În anul 2000, odată cu lansarea versiunii MATLAB 6.0 (R12) a fost încorporat și codul executabil JVM (Java Virtual Machine, [25, 28]) de la Sun Microsystems, [26].
- În anul 2008, a fost lansată versiunea MATLAB 7.6 (R2008a) care conținea sistemul de calcul simbolic MuPAD (SciFace Software, [27]), precum și mediul de simulare a sistemelor fizice Simscape.

- În anul 2009 a fost lansată versiunea MATLAB 7.8 (R2009a) pentru sistemul de operare Microsoft Windows 7 pe 32 și 64 de biți.
- În anul 2012 au fost lansate versiunile MATLAB 7.14 (R2012a) care include versiunea 7.9 a limbajului Simulink și MATLAB 8 (R2012b) care include versiunea 8 a limbajului Simulink. Începând cu versiunea 8, interfața programului este complet modificată.
- În luna martie a anului 2013 a fost lansată versiunea MATLAB 8.1 (R2013a) care include versiunea 8.1 a limbajului Simulink.
- În luna septembrie a anului 2013 a fost lansată versiunea MATLAB 8.2 (R2013b) care include versiunea 8.2 a limbajului Simulink.
- În luna martie a anului 2014 a fost lansată versiunea MATLAB 8.3 (R2014a) care include versiunea 8.3 a limbajului Simulink.

Limbajul de programare MATLAB este format dintr-un nucleu de bază conținând elementele fundamentale ale programului la care se adaugă un mare număr de toolbox-uri, furnizând astfel noi facilități computaționale de nivel general sau proceduri de calcul specifice anumitor domenii ale științelor ingineresti.

Principalele caracteristici computaționale ale limbajului de programare MATLAB sunt, [24]:

- Elemente fundamentale de programare, funcții matematice uzuale, funcții pentru reprezentarea grafică 2D și 3D, proceduri pentru generarea funcțiilor definite de utilizator, definirea și manipularea polinoamelor, funcții pentru controlul datelor și fișierelor, funcții pentru integrarea și derivarea numerică, funcții pentru rezolvarea numerică a ecuațiilor și sistemelor de ecuații algebrice și transcendente, funcții pentru rezolvarea numerică a ecuațiilor și sistemelor de ecuații diferențiale ordinare, [1, 5, 7, 12].
- Efectuarea de calcule simbolice (Symbolic Math Toolbox, [6]): generarea și manipularea variabilelor, expresiilor și funcțiilor simbolice, reprezentarea grafică a funcțiilor simbolice, derivarea și integrarea simbolică, calculul simbolic al limitelor, rezolvarea simbolică a ecuațiilor. Interfață grafică specializată pentru efectuarea de calcule simbolice (MuPAD, [8]).
- Rezolvarea ecuațiilor diferențiale cu derivate parțiale în domeniul bidimensional (Partial Differential Equation Toolbox, [11]). Interfață grafică specializată pentru rezolvarea ecuațiilor diferențiale cu derivate parțiale. Module de calcul specializate pentru: mecanică, electrostatică, magnetostatică, transfer de căldură, difuzie.
- Analiza statistică (Statistics Toolbox) și numerică (Curve Fitting Toolbox) a datelor experimentale. Metode de aproximare a funcțiilor prin interpolare și regresie folosind polinoame, funcții putere,

exponențiale, trigonometrice, Gauss, Weibull, Fourier). Funcții specifice pentru analiza parametrilor de calitate ai aproximărilor: nivelul de încredere, intervalele de eroare, parametri numerici de calitate, reprezentarea grafică a reziduurilor. Interfețe grafice specializate pentru analiza numerică a datelor experimentale.

- Analiza, proiectarea și reglarea sistemelor dinamice folosind metoda procedurilor numerice specializate (Control Systems Toolbox, [2]). Definirea sistemelor dinamice prin metoda funcției de transfer, metoda intrare-stare-ieșire, metoda poli-zerouri. Analiza performanțelor dinamice și de stabilitate ale sistemelor dinamice în domeniul timp (răspuns la semnal treaptă) și în domeniul frecvențial (diagrama Bode).
- Modelarea și simularea sistemelor dinamice folosind metoda schemelor bloc (Simulink, [16, 17]) cu biblioteci de blocuri pentru: simularea sistemelor dinamice continue; simularea sistemelor dinamice discrete; simularea discontinuităților; operații matematice de bază; operații logice; operații la nivel de bit; aproximarea funcțiilor 1D, 2D și nD; simularea semnalelor de intrare; simularea elementelor pentru vizualizarea grafică și numerică a semnalelor de ieșire; manipularea semnalelor; modificarea parametrilor semnalelor; verificarea domeniului de variație a valorilor numerice ale semnalelor; crearea și manipularea subsistemelor; crearea și manipularea blocurilor definite de utilizator.
- Modelarea și simularea sistemelor fizice folosind metoda schemelor funcționale (Simscape) cu biblioteci de elemente funcționale de bază pentru sisteme mecanice, hidraulice, electrice, termice, pneumatice.
- Modelarea și simularea sistemelor hidraulice folosind metoda schemelor funcționale cu biblioteci de elemente funcționale specializate (SimHydraulics, [14]) pentru: pompe centrifuge, pompe volumice, distribuitoare, robinete, orificii, conducte, motoare hidraulice liniare și rotative, acumuloare, rezervoare, actuatori, supape de sens, supape de presiune.
- Modelarea și simularea sistemelor mecanice 3D folosind metoda schemelor funcționale cu biblioteci de elemente funcționale specializate (SimMechanics) pentru: corpuri rigide, stabilirea poziției relative dintre mai multe corpuri rigide, conexiunea de tip oscilator amortizat, cuple, aplicarea sarcinilor, măsurarea parametrilor mecanici.
- Modelarea, simularea și analiza sistemelor biologice prin algoritmi specifici (Bioinformatics Toolbox) și prin metoda schemelor funcționale cu biblioteci de elemente funcționale specializate (SimBiology).

- Modelarea și simularea sistemelor mecanice 1D folosind metoda schemelor funcționale cu biblioteci de elemente funcționale specializate (SimDriveline) pentru: generatoare și tractoare pentru mișcarea de rotație, angrenaje, limitatoare de cursă, amortizoare și elemente elastice pentru mișcarea de rotație.
- Modelarea și simularea sistemelor electronice folosind metoda schemelor funcționale cu biblioteci de elemente funcționale specializate (SimElectronics) pentru: surse de curent, surse de tensiune, rezistențe, rezistențe fuzibile, bobine, condensatoare, diode, circuite integrate, termocuple, termorezistențe, senzori de proximitate, mărci tensometrice, fotodiode, motoare electrice.
- Modelarea și simularea sistemelor electrice de putere folosind metoda schemelor funcționale cu biblioteci de elemente funcționale specializate (SimPowerSystems) pentru: mașini electrice, tractoare pentru mărimile electrice, turbine eoliene.
- Obținerea modelelor matematice ale sistemelor dinamice (identificarea sistemelor dinamice) pe baza măsurării mărimilor de intrare și de ieșire ale sistemelor (System Identification Toolbox).
- Rezolvarea problemelor de optimizare liniară, pătratică, neliniară, multiobiectiv (Optimization Toolbox, [9]).
- Implementarea unor algoritmi specifici pentru generarea, manipularea, vizualizarea și analiza semnalelor digitale și analogice. Interfețe grafice specializate. Analiza spectrală a semnalelor de diferite tipuri: semnale audio, seismice, financiare, biomedicale, radar. (Signal Processing Toolbox, [13]).
- Implementarea unor algoritmi specifici de procesare, analiză și vizualizare a imaginilor (Image Processing Toolbox). Algoritmi și funcții specifice pentru proiectarea și managementul sistemelor de supraveghere video (Computer Vision System Toolbox). Realizarea interfațării cu camere video (Image Acquisition Toolbox).
- Realizarea conectării cu sisteme de achiziție de date (DAQ-Data Acquisition) construite pe diferite platforme: USB, PCI, PCI Express, PXI, PXI Express și dezvoltate de diferiți producători: National Instruments, Measurement Computing, Advantech, Data Translation (Data Acquisition Toolbox).
- Modelarea matematică și analiza statistică a datelor financiare (Financial Toolbox, Econometrics Toolbox, Datafeed Toolbox, Financial Instruments Toolbox, Trading Toolbox).
- Funcții pentru generarea interfețelor grafice de tip GUI (Graphic User Interface) cu ajutorul interfeței GUIDE (Graphic User Interface Development Environment), [3].



- Realizarea documentațiilor tehnice aferente diferitelor aplicații și algoritmi numerici și simbolici de calcul (MATLAB Report Generator). Realizarea documentațiilor tehnice aferente diferitelor scheme bloc și scheme funcționale de simulare a sistemelor dinamice (Simulink Report Generator).
- Transferul bidirecțional al datelor între MATLAB și Microsoft Excel (Spreadsheet Link EX, [18]). Transferul bidirecțional al datelor între MATLAB și baze de date relaționale: Oracle, MySQL, Sybase, Microsoft SQL Server, Informix (Database Toolbox).
- Crearea aplicațiilor executabile prin utilizarea unor compilatoare (Application Deployment). Realizarea partajării aplicațiilor create de utilizator prin definirea fișierelor executabile sau a bibliotecilor partajabile (fișiere de tip .exe sau .dll) (Matlab Compiler), a claselor Java (Matlab Builder JA), a macrourilor de tip VBA-Visual Basic for Applications (MATLAB Builder EX), a obiectelor de tip .COM și NET (MATLAB Builder NE).
- Rezolvarea problemelor numerice complexe utilizând proceduri specifice de paralelizare a calculelor și metode de programare de tip CUDA pe sisteme de calcul multicore (Parallel Computing Toolbox, [10]) și distribuirea calculelor pe sisteme de calcul de tip cluster, cloud și grid (MATLAB Distributed Computing Server).

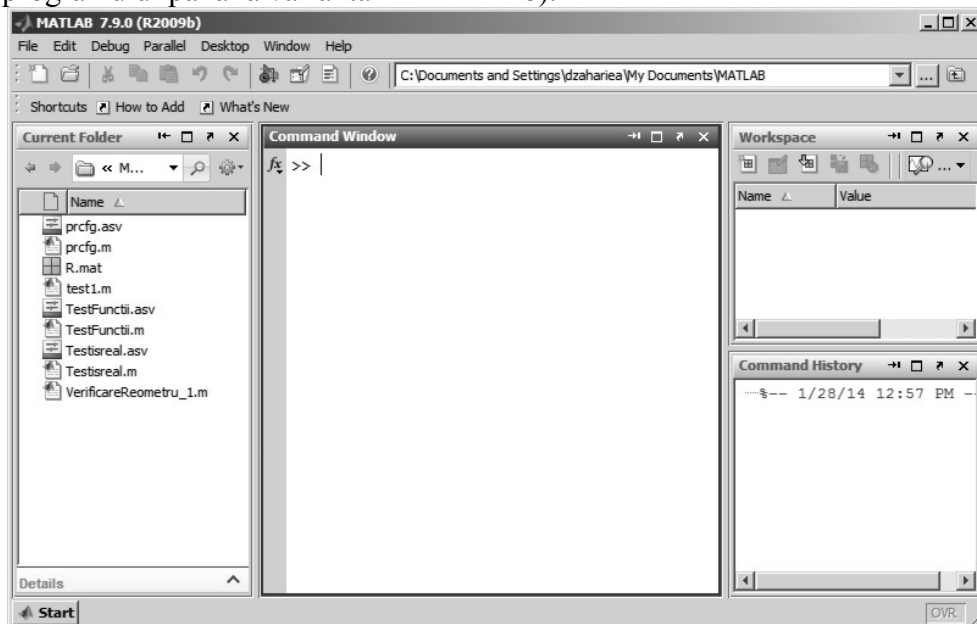
Principalele tipuri de fișiere specifice pe care le manipulează în mod nativ limbajul de programare MATLAB sunt:

- Fișiere cu extensia .m (fișiere de tip `script` sau de tip `function` care conțin instrucțiuni și text explicativ).
- Fișiere cu extensia .fig (fișiere de tip `figure` care conțin informații de tip grafic).
- Fișiere cu extensia .mdl și .slx (fișiere de tip model care conțin scheme bloc de tip Simulink și scheme funcționale de tip Simscape).
- Fișiere cu extensia .mn (fișiere care conțin comenzi de calcul simbolic MuPAD).
- Fișiere cu extensia .mat (fișiere care conțin date formate și care pot fi utilizate pentru salvarea datelor din spațiul de lucru MATLAB, folosind comanda `save`, respectiv pentru încărcarea datelor în spațiul de lucru MATLAB, folosind comanda `load`).

Limbajul de programare MATLAB are o structură deschisă, fiind posibilă dezvoltarea unor biblioteci de programe specifice unui anumit domeniu de activitate (toolbox-uri definite de utilizator). Prin caracteristicile computaționale, prin numărul mare de toolbox-uri specifice, prin caracterul său deschis, MATLAB este unul din cele mai răspândite limbaje de programare academice în domeniul științelor ingineresti.

## 2.2. INTERFAȚA PROGRAMULUI

Lansarea în execuție a limbajului de programare MATLAB conduce, în mod implicit, la interfața prezentată în figura 2.1, [4] (pentru versiuni ale programului până la varianta MATLAB 8).



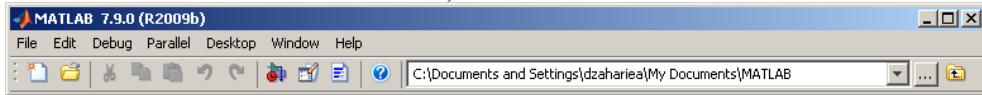
**Figura 2.1.** Ecranul implicit al limbajului de programare MATLAB.

Principalele elemente ale interfeței MATLAB sunt:

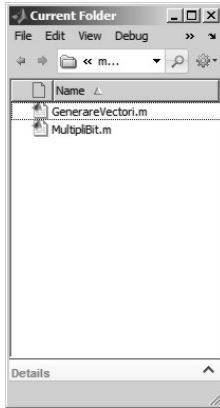
- Bara cu meniurile: File, Edit, Debug, Parallel, Desktop, Window și Help, figura 2.2, a).
- Toolbar-ul cu butoane corespunzătoare comenzilor generale: New, Open, Cut, Copy, Paste, Undo, Redo, Simulink, GUIDE, Profiler, Help, Current Folder, Browse for folder; Go up one level, figura 2.2, b).
- Fereastra Current Folder (directorul curent) reprezintă conținutul directorului în care programul salvează automat fișierele create de utilizator, figura 2.2, c). Directorul Current Folder poate fi oricând configurat de utilizator dar la un moment dat nu poate fi declarat decât un singur director de acest tip. Selectarea directorului dorit pentru a reprezenta Current Folder se poate realiza folosind butoanele Browse for folder și Go up one level din toolbar-ul programului.
- Fereastra Command Window (fereastra de comenzi) reprezintă zona în care utilizatorul introduce comenzi, respectiv în care programul poate furniza rezultate, figura 2.2, d).



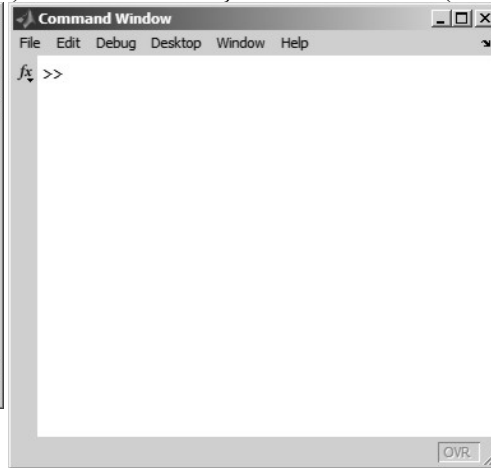
a) bara cu meniuri



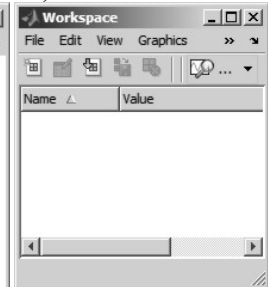
b) bara cu meniuri și bara cu butoane (toolbar)



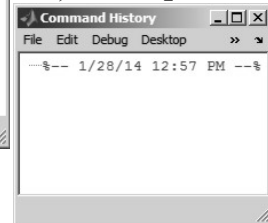
c) Current Folder



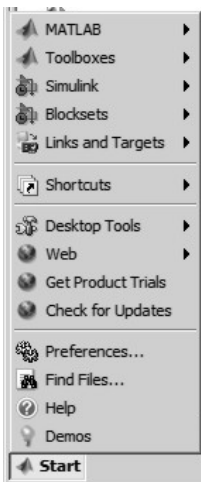
d) fereastra Command Window



e) Workspace



f) Command History



g) butonul Start

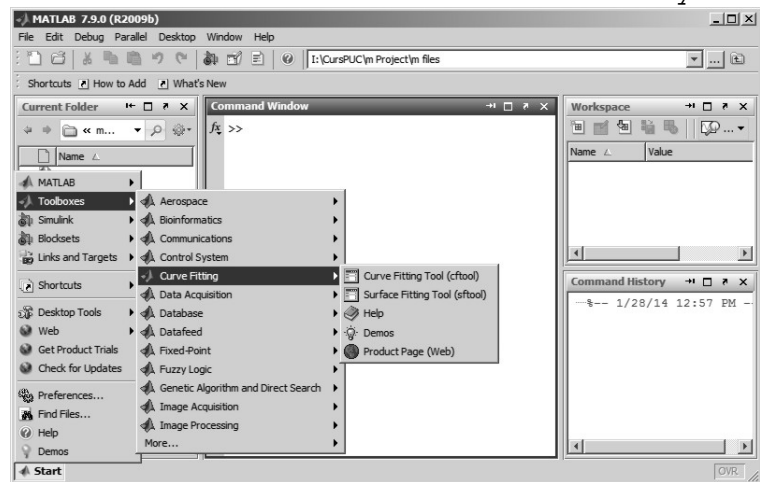


Figura 2.2. Principalele elemente ale interfeței MATLAB.

- Fereastra Workspace (spațiul de lucru) reprezintă fereastra în care se afișează caracteristicile principale ale variabilelor create în program: valoarea în cazul variabilelor scalare, dimensiunea în cazul

variabilelor vectoriale și matriceale, figura 2.2, e). Efectuând dublu click pe numele oricărei variabile existente în Workspace se deschide fereastra Variable Editor, în care se poate edita variabila respectivă.

- Fereastra Command History (istoria comenzilor) reprezintă o statistică a comenzilor introduse în fereastra de comenzi a programului, în diferite sesiuni de lucru anterioare, figura 2.2, f). Orice comanda existentă în fereastra Command History poate fi copiată în spațiul de lucru al programului și executată din nou.
- Butonul Start, aflat în partea stânga-jos a interfeței permite accesul rapid la: fișiere demonstrative (Demos) și de informații (Help); interfețe utilizator specifice unor toolbox-uri instalate în program; link-uri ale unor site-uri Web specifice diferitelor elemente ale limbajului de programare; comanda de configurare a mediului de programare, Preferences; etc., figura 2.2, g).

Structura implicită de ferestre a interfeței programului MATLAB poate fi configurată folosind comanda Desktop/Desktop Layout/Default.

Ferestrele interfeței implicite pot fi manipulate independent, astfel încât se poate lucra cu o configurație diferită a ferestrelor. Principalele operațiuni care se pot executa asupra ferestrelor sunt:

- Minimizare. Modificarea dimensiunii unei ferestre astfel încât aceasta se va reduce la un simplu buton, în timp ce celelalte ferestre se vor mări automat, ocupând întregul spațiu disponibil al ferestrei generale MATLAB. Revenirea la dimensiunile inițiale se realizează prin comanda Restore.
- Maximizare. Modificarea dimensiunii unei ferestre astfel încât aceasta va ocupa întreg spațiul disponibil al ferestrei generale MATLAB. Revenirea la dimensiunile inițiale se realizează prin comanda Restore.
- Incluziune/excluziune în raport cu fereastra generală MATLAB (dock/undock). O fereastră exclusă se poate manipula apoi separat față de fereastra generală.
- Închidere.

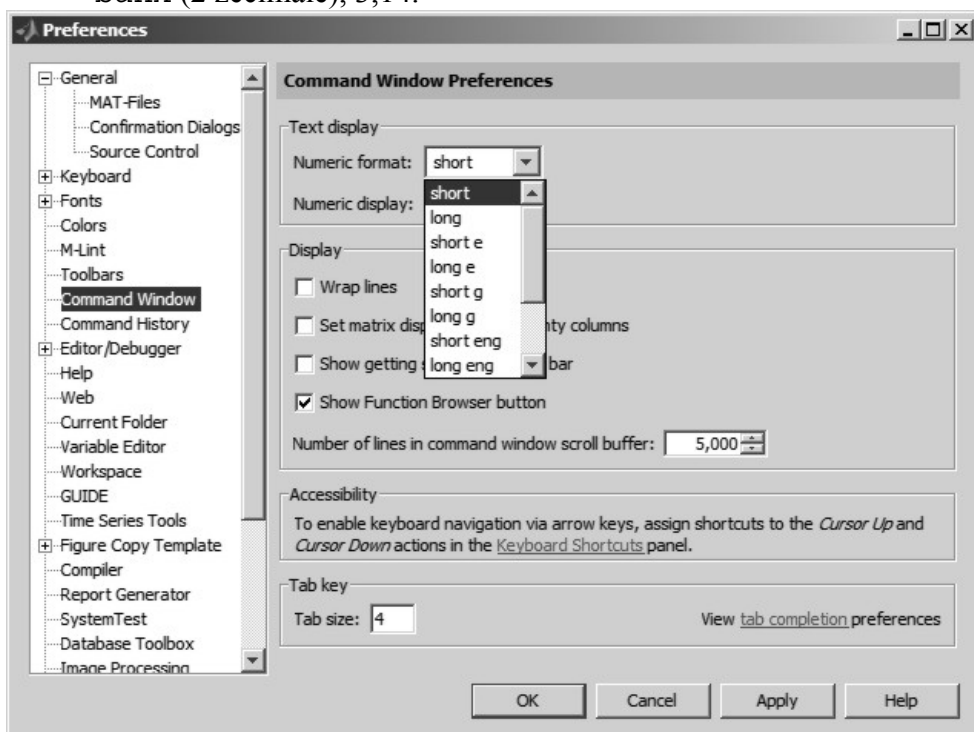
Configurarea mediului de lucru se realizează folosind comanda Preferences din meniul File sau din butonul Start. Fereastra comenzii Preferences asigură accesul utilizatorului la variabilele de configurare a mediului de lucru. Aceste variabile sunt grupate pe diferite categorii: variabile generale; variabile de configurare a tastelor și shortcut-urilor; variabile de configurare a tipului de caractere utilizate; variabile de configurare a culorilor, variabile de configurare a toolbar-ului; variabile de

configurare a ferestrei Command History; variabile de configurare a ferestrei de comenzi; etc.

Între comenzile de configurare existente în categoria Command Window se află și comanda Numeric Format (figura 2.3) care permite specificarea modului de afișare a rezultatelor numerice.

De exemplu numărul  $\pi$  poate fi exprimat prin mai multe metode:

- short (virgulă fixă cu 4 zecimale), 3,1416.
- long (virgulă fixă cu 15 zecimale), 3,141592653589793.
- short e (virgulă mobilă cu 4 zecimale), 3,1416e+000=3,1416 $\cdot 10^0$ .
- long e (virgulă mobilă cu 15 zecimale), 3,141592653589793 e+000.
- hex (sistem de numerație hexazecimal), 400921fb54442d18.
- rat (format fracționar), 355/113.
- bank (2 zecimale), 3,14.



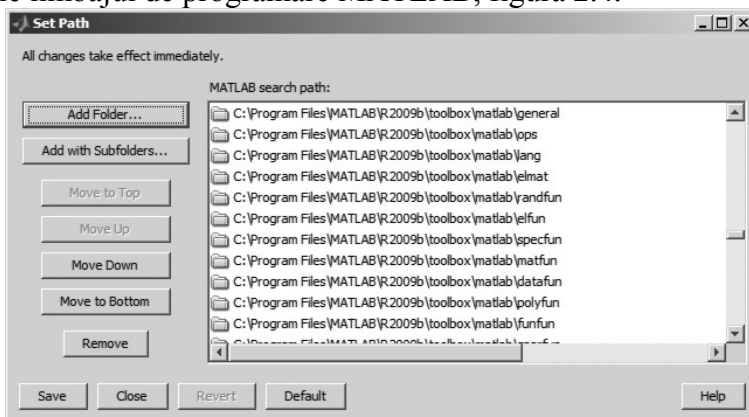
**Figura 2.3.** Comanda de configurare Preferences.

Principalele funcții de control general ale limbajului de programare MATLAB sunt:

- `clc` Șterge conținutul ferestrei de comenzi
- `clear` Șterge variabile din spațiul de lucru.
- `close` Închide fereastra grafică curentă.

Folosirea instrucțiunilor `clc`, `clear all` și `close all` se recomandă înainte de începerea fiecărui nou exemplu de calcul, ca și la începutul fiecărui program (fișier de tip `script`).

- `clear x`                      Șterge doar variabila `x`
- `clear a b`                    Șterge variabilele `a` și `b`
- `clear all`                    Șterge toate variabilele din spațiul de lucru
- `who`                            Prezintă lista variabilelor din spațiul de lucru
- `who`  
Your variables are:  
`x y`
- `whos`                         Prezintă lista variabilelor, inclusiv informații suplimentare despre dimensiunea și tipul acestora.
- `whos`  
Name      Size                      Bytes    Class      Attributes  
`x`        `1x1`                                8    double  
`y`        `1x1000`                            8000   double
- `exit, quit`                    Termină sesiunea de lucru curentă.
- Preferences                  Lansează interfața de configurare a mediului de lucru.
- Pathtool                        Lansează interfața specializată `Set Path` care se utilizează pentru definirea structurii de directoare relevante ale limbajul de programare MATLAB, figura 2.4.

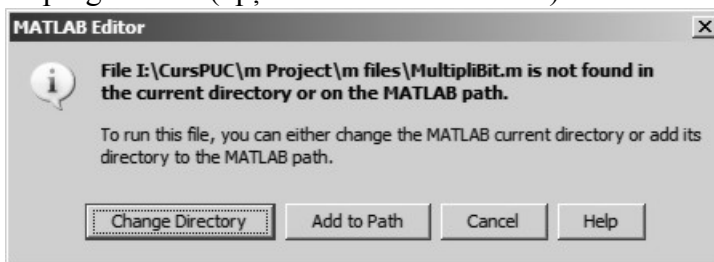


**Figura 2.4.** Interfața `Set Path` pentru configurarea structurii de directoare relevante.

La instalarea programului MATLAB, se generează automat o structură de directoare conținând toate directoarele și subdirectoarele relevante ale programului. Această structură inițială poate fi apoi dezvoltată de utilizator prin adăugarea unor noi directoare (`Add Folder`), inclusiv cu subdirectoarele componente (`Add with Subfolders`).

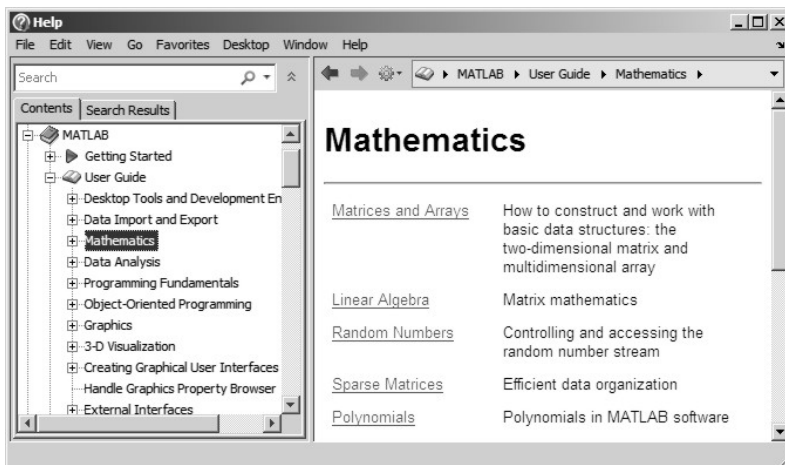
De asemenea, interfața `Set Path` permite stabilirea unei anumite priorități a procesului de căutare în structura de directoare relevante pentru cazul în care același fișier este salvat în două directoare diferite (`Move to Top`, `Move Up`, `Move Down`, `Move to Bottom`), dar și ștergerea unui anumit director din structura de directoare relevante (`Remove`).

Orice program poate fi executat doar dacă este salvat anterior într-un director care face parte din această structura de directoare. În cazul în care se încearcă lansarea în execuție a unui fișier `script` care nu se găsește într-unul din directoarele relevante ale programului programul solicită utilizatorului, printr-o fereastră de dialog (figura 2.5), fie schimbarea directorului curent (opțiunea `Change Directory`), fie adăugarea directorului în care se găsește fișierul de executat la structura de directoare relevante ale programului (opțiunea `Add to Path`).



**Figura 2.5.** Lansarea în execuție a unui fișier `script` care nu se găsește într-unul din directoarele relevante ale programului.

Obținerea de informații despre elementele limbajului de programare MATLAB se poate face folosind baza de date structurată `Help`, figura 2.6. Aceasta poate fi lansată în execuție fie din meniul `Help`, fie prin apăsarea tastei `F1`, fie prin lansarea comenzii `Help` din toolbarul programului (figura 2.2, b).



**Figura 2.6.** Fereastra `Help`.

## 2.3. MODUL DE LUCRU

Principalele moduri de lucru în limbajul de programare MATLAB sunt: modul de lucru în linie de comandă, modul de lucru bazat pe fișiere de tip `script` și modul de lucru bazat pe definirea funcțiilor (prin metoda fișierelor de tip `function`, prin metoda funcțiilor de tip `inline` și prin metoda funcțiilor de tip `anonymous`), [12].

### 2.3.1. Modul de lucru în linie de comandă

Constă în scrierea instrucțiunilor direct la prompterul MATLAB (`>>`), în fereastra de comenzi. După scrierea unei instrucțiuni se apasă tasta ENTER, lansând astfel în execuție instrucțiunea respectivă și trecând în mod automat la linia următoare, unde urmează scrierea următoarei instrucțiuni.

Observații generale pentru scrierea instrucțiunilor:

- Limbajul de programare MATLAB este sensibil la majuscule, altfel spus variabila `x` este diferită de variabila `X`. De exemplu, instrucțiunile:

```
x=1; X=4; y=x+X
```

furnizează rezultatul:

```
y=5
```

- Apăsarea tastei ENTER la sfârșitul unei instrucțiuni are ca efect executarea instrucțiunii respective și în plus afișarea rezultatului obținut. Dacă la sfârșitul unei instrucțiuni se plasează, suplimentar, caracterul `;` atunci nu se mai afișează pe ecran rezultatul obținut în urma executării instrucțiunii respective. În ambele situații însă, după apăsarea tastei ENTER, variabila respectivă se va regăsi în spațiul de lucru al programului. Folosirea caracterului `;` se recomandă la introducerea datelor inițiale ale unui program, ca și după instrucțiunile care conduc la rezultate intermediare.

```
>> x=1 >> x=1;
```

```
x = >>
```

```
1
```

```
>>
```

- Dacă în cursul procesului de execuție a unei anumite instrucțiuni se întâlnesc erori, execuția instrucțiunii se oprește și se afișează un mesaj de eroare. Mesajul de eroare obținut furnizează indicații despre tipul și locația erorii detectate de program.

```
x=1;y=2;
```

```
z=(x+1)/(y+3
```

```
??? z=(x+1)/(y+3
```

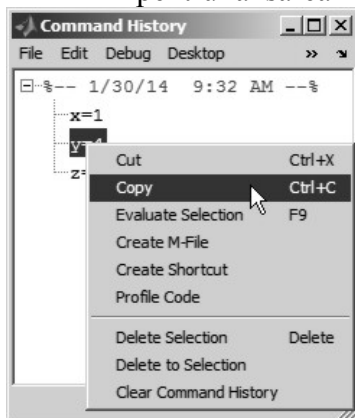
```
|  
Error: Expression or statement is incorrect-  
possibly unbalanced (, {, or [.
```



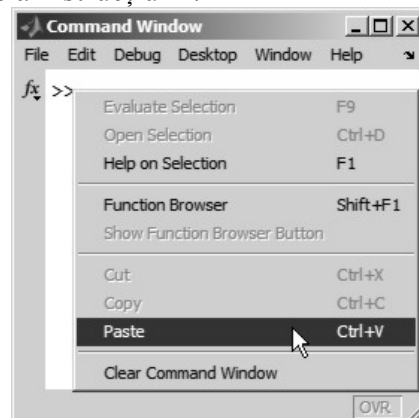
- În cazul instrucțiunilor lungi, pentru scrierea cărora nu este suficient un singur rând, la sfârșitul liniei curente se introduc trei puncte ( . . . ), după care se apăsă tasta ENTER. Se trece astfel la rândul următor pe care se continuă scrierea instrucțiunii.

$$x = (u + 2 * \log(t) / \sqrt{y}) / (\sin(e - \pi) \dots + \cos(e + \pi)) + \exp(2 - q) * (\arccos(t/k) + 1/u^k);$$

- Fiecare instrucțiune care se execută în fereastra Command Window este salvată în mod automat în fișierul history.m sub forma unei baze de date structurată pe zile și ore. Înregistrările din această bază de date se păstrează până la atingerea unei valori maxime a cantității de informație, sau până ce utilizatorul decide să șteargă aceste înregistrări (folosind comanda Clear Command History din meniul Edit). Accesul la această bază de date se realizează cu ajutorul ferestrei Command History. Repetarea unei instrucțiuni anterioare (pentru corectarea unor erori, pentru modificarea unor parametri, sau pur și simplu pentru repetarea unor calcule) se poate realiza prin mai multe metode:
  - Se efectuează un click cu butonul drept al mouse-ului (RMB) pe instrucțiunea dorită din fereastra Command History și apoi se selectează comanda Copy, realizându-se astfel copierea instrucțiunii în Clipboard, figura 2.7, a). Apoi se efectuează un click RMB în fereastra de comenzi Command Window și apoi se selectează comanda Paste din mediul contextual corespunzător, realizându-se astfel copierea instrucțiunii, figura 2.7, b). Apoi se apasă tasta ENTER pentru lansarea în execuție a instrucțiunii.



a) copierea din fereastra Command History în Clipboard



b) copierea din Clipboard în fereastra Command Window

**Figura 2.7.** Lansarea în execuție a unei instrucțiuni anterioare.

- Se deplasează pe verticală cursorul din dreapta ferestrei `Command History` până se găsește instrucțiunea dorită, se selectează instrucțiunea prin efectuarea unui click cu butonul stâng al mouse-ului (LMB), apoi se execută operațiunea `drag&drop` spre fereastra de comenzi `Command Window`, unde se eliberează butonul stâng al mouse-ului. Prin această operațiune se realizează copierea instrucțiunii din fereastra `Command History` în fereastra `Command Window`, unde se poate lansa în execuție prin apăsarea tastei `ENTER`.
- Se efectuează click cu butonul drept al mouse-ului (RMB) pe instrucțiunea dorită din fereastra `Command History` și apoi se selectează comanda `Evaluate Selection` din meniul contextual corespunzător, realizându-se astfel lansarea în execuție a instrucțiunii respective.
- Se selectează fereastra de comenzi `Command Window` și se apasă tastele `↑` (`Up Arrow`) și `↓` (`Down Arrow`) realizându-se astfel parcurgerea, una după alta, a instrucțiunilor din `Command History` și în același timp, transferarea automată și vizualizarea acestora în fereastra de comenzi `Command Window` până se identifică instrucțiunea dorită, moment în care instrucțiunea se va lansa în execuție prin apăsarea tastei `ENTER`.

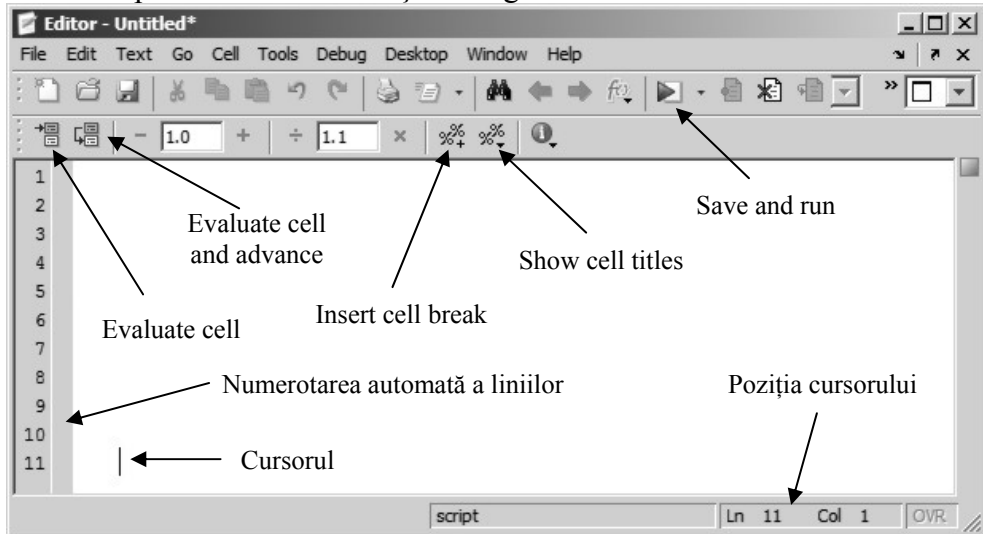
### 2.3.2. Modul de lucru pe bază de fișiere `script`

Un fișier `script` conține o succesiune de instrucțiuni care se vor procesa automat la lansarea fișierului în execuție ca și cum ar fi introduse independent în fereastra de comenzi a programului. Fișierele `script` pot utiliza datele create în structura fișierului, sau datele existente în spațiul de lucru al programului. Fișierele `script` nu au definite variabile de ieșire (și nici variabile de intrare), însă toate variabilele definite sau calculate în structura fișierului, rămân după executarea fișierului `script`, în spațiul de lucru al programului. Folosirea fișierelor `script` este cu atât mai indicată cu cât crește complexitatea și numărul instrucțiunilor de executat, cu atât mai mult cu cât se urmărește utilizarea repetată a instrucțiunilor respective.

La utilizarea fișierelor `script` trebuie să se considere următoarele reguli și observații generale:

- Deschiderea unui nou fișier `script` se poate face prin selectarea comenzii `New/Script` (sau `New/Blank M-File`) din meniul `File`, prin selectarea comenzii `New Script` (sau `New M-File`) din bara cu butoane a programului, sau prin apăsarea tastelor `CTRL+N`. Se deschide astfel fereastra de editare a fișierelor

script, figura 2.8 în care se vor introduce, la poziția curentă a cursorului, linie după linie, toate informațiile specifice fișierului respectiv. Editorul de fișiere asigură numerotarea automată a liniilor.



**Figura 2.8.** Editorul de fișiere script.

- La salvarea unui fișier script trebuie avut în vedere faptul că extensia fișierului este în mod implicit .m și nu trebuie modificată.
- Numele fișierelor script trebuie să înceapă cu o literă, poate să conțină caracterul „\_”, dar nu trebuie să conțină spații libere.
- Directorul în care se salvează fișierele script trebuie adăugat în structura relevanta de directoare a programului.
- În structura fișierelor script pot fi introduse atât instrucțiuni, grupuri de instrucțiuni reunite în celule de calcul, grupuri de instrucțiuni definite ca funcții, precum și linii de text explicativ. O celulă de program poate conține la rândul său, atât instrucțiuni, text explicativ, cât și funcții. De asemenea, funcțiile pot conține instrucțiuni, text explicativ, celule de calcul, precum și alte funcții. Un text care începe cu % este considerat a fi text explicativ, imediat după caracterul % și până la sfârșitul liniei curente. O zonă de program care începe cu %% este considerată o celulă. O celulă începe deci cu %% și se termină fie la întâlnirea, din nou, a caracterelor %% (moment în care începe o nouă celulă), fie la sfârșitul fișierului script respectiv. Se recomandă ca fiecare din grupurile principale de instrucțiuni ale unui fișier script să fie definite ca și celule (de exemplu, celula cu titlul programului, datele de identificare ale autorului, grupul de instrucțiuni `clc`, `clear all`, `close all`; celula cu datele inițiale ale programului; celula

cu calculele; celula cu reprezentările grafice; etc.). Inserarea unei celule la poziția curentă a cursorului dintr-un fișier se poate face și prin comanda `Insert cell break` din bara de butoane a editorului de fișiere, figura 2.8. Comanda `Show cell titles` din bara de butoane a editorului de fișiere prezintă lista tuturor celulelor definite în fișierul curent și, prin selectarea unui titlu oarecare, permite saltul în fișier, direct la celula selectată, figura 2.8.

- Identificarea eventualelor erori logice sau de sintaxă nu poate fi realizată decât odată cu lansarea în execuție a fișierului `script` (prin apăsarea tastei F5, prin selectarea butonului `Save and Run` din bara cu butoane a editorului de fișiere, sau prin lansarea comenzii `Save File and Run` din meniul `Debug`). Rezultă deci, necesitatea lansării repetate în execuția a fișierului `script`, pe întreg parcursul dezvoltării acestuia, pentru a identifica din timp eventualele erori. Scrierea tuturor instrucțiunilor și apoi lansarea în execuție este o metodă de lucru contraproductivă.
- Lansarea în execuție doar a instrucțiunilor din structura unei anumite celule a fișierului se poate realiza prin comanda `Evaluate Cell` (CTRL+ENTER) din bara cu butoane a editorului de fișiere. Dacă se dorește în plus și avansarea la celula următoare trebuie selectată comanda `Evaluate cell and advance` (CTRL+SHIFT+ENTER) din bara de butoane a editorului de fișiere, figura 2.8. În acest mod se pot lansa în execuție doar anumite instrucțiuni ale fișierului `script`.
- După editarea, executarea și verificarea unui fișier `script`, acesta poate fi ulterior îmbunătățit și eventual distribuit altor utilizatori.

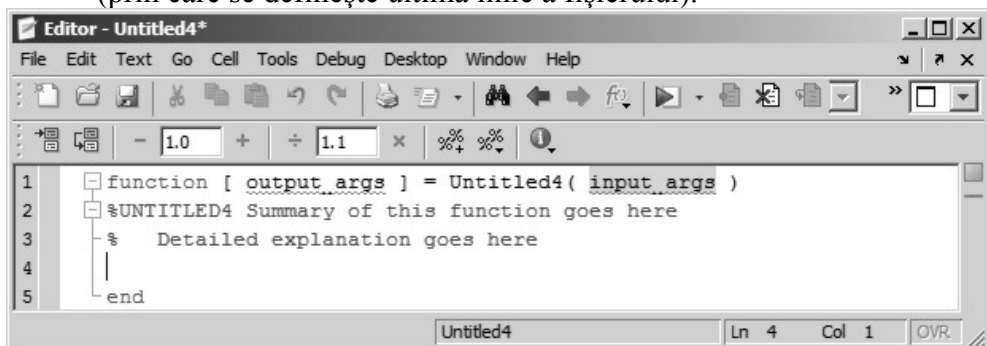
Principalele avantaje ale metodei fișierelor de tip `script` sunt: posibilitatea de a scrie coduri sursă complexe cu foarte multe linii; posibilitatea de utilizare ulterioară, repetată a instrucțiunilor din structura fișierului; posibilitatea introducerii de text explicativ pentru adnotarea instrucțiunilor din structura fișierului; posibilitatea apelării numelui fișierului astfel creat în structura altor fișiere; posibilitatea transferului rapid către alți utilizatori.

### 2.3.3. Modul de lucru bazat pe utilizarea funcțiilor

În cursul rezolvării multor probleme apare necesitatea creării funcțiilor definite de utilizator. Aceste funcții pot fi definite prin mai multe metode: metoda fișierelor de tip `function`, metoda funcțiilor `inline`, metoda funcțiilor `anonymous`. Alegerea metodei pentru definirea unei funcții depinde de mai mulți factori printre care cei mai importanți se referă la tipul funcției și la modul de utilizare a funcției respective.

Principalele caracteristici ale fișierelor de tip `function` sunt:

- Fișierele de tip `function` beneficiază de avantajele fișierelor de tip `script`, dar, spre deosebire de acestea, pot avea variabile de intrare și pot furniza variabile de ieșire.
- Fișierul de tip `function` este un fișier `script` particular care are proprietatea că își construiește un spațiu de lucru independent, cu variabile independente de spațiul de lucru al programului prin intermediul unor variabile de tip `local`. Comunicarea dintre spațiul de lucru al programului și cel al fișierului `function` se realizează prin intermediul variabilelor de intrare ale funcției care sunt transferate din spațiul de lucru al programului în spațiul de lucru al fișierului `function`, precum și prin intermediul variabilelor de ieșire ale funcției care sunt transferate din spațiul de lucru al fișierului `function` în spațiul de lucru al programului. Transferul de date dintre spațiul de lucru al programului și cel al fișierului se poate realiza și prin intermediul unor variabile particulare, definite în structura fișierului `function` ca fiind variabile de tip `global`.
- Fișierul de tip `function` poate fi apelat în cadrul unui al fișier de tip `function`, în cadrul unui fișier de tip `script`, sau direct în fereastra de comenzi. La apelarea unui fișier de tip `function` nu are importanță numele variabilelor care sunt transferate din spațiul de lucru al programului către spațiul de lucru al funcției, ci doar poziția acestora în instrucțiunea de apelare.
- Deschiderea unui nou fișier `function` se poate face prin selectarea comenzii `New/Function` (sau `New/Function M-File`) din meniul `File`. Se deschide astfel fereastra de editare a fișierelor `function` (figura 2.9), care este de fapt identică cu fereastra de editare a fișierelor `script`, cu excepția cuvintelor rezervate `function` (prin care se definește prima linie a fișierului) și `end` (prin care se definește ultima linie a fișierului).



**Figura 2.9.** Editorul de fișiere `function`.

- Structura generală a unui fișier de tip `function` este următoarea:

```
function [Ieșire]=NumeFuncție (Intrare)
%Text explicativ
Instrucțiuni
end
```

în care: `Ieșire` reprezintă lista variabilelor de ieșire ale funcției, `Intrare` reprezintă lista variabilelor de intrare ale funcției, iar `NumeFuncție` este numele funcției și în mod obligatoriu și numele fișierului în care se va salva funcția (`NumeFuncție.m`). Există funcții care nu au variabile de intrare, sau care nu au variabile de ieșire, sau care nu au nici variabile de intrare, nici de ieșire.

- Cuvântul `function` este un cuvânt rezervat care nu se poate utiliza decât în procesul definirii fișierelor de tip `function`. Cuvântul `end` este de asemenea un cuvânt rezervat, care poate însă să fie utilizat și în alte situații.
- În structura unui fișier `function` se pot introduce instrucțiuni, grupuri de instrucțiuni reunite în celule de calcul, grupuri de instrucțiuni definite ca funcții, precum și text explicativ, ca și în cazul fișierelor de tip `script`.
- Dacă în structura unui fișier `function`, cuvântul rezervat `function` apare de mai multe ori, exceptând prima apariție (care corespunde funcției principale), toate celelalte apariții definesc sub-funcții (funcții locale). Sub-funcțiile sunt vizibile doar pentru funcția principală și pentru celelalte funcții locale definite în corpul funcției principale respective.
- Funcțiile pot fi definite atât prin metoda fișierelor de tip `function` (funcții cu mai multe variabile de intrare, mai multe variabile de ieșire și mai mult de o singură instrucțiune), cât și prin metoda funcțiilor `inline` (funcții cu mai multe variabile de intrare, o singură variabilă de ieșire și doar o singură instrucțiune). Funcțiile de tip `inline` sunt funcții locale care pot fi definite în fereastra de comenzi la prompterul MATLAB, în structura unui fișier de tip `script`, sau într-un fișier de tip `function`. Structura generală a unei funcții de tip `inline` este următoarea:

```
NumeFuncție=inline ('Expresie' , 'vi1' , 'vi2' , ...)
```

în care `vi1`, `vi2`, ... sunt toate variabilele de intrare care se află în relația matematică `Expresie` prin care se definește funcția cu numele `NumeFuncție`.

- Tot pentru reprezentarea funcțiilor definite printr-o singură expresie matematică se pot folosi funcțiile de tip `anonymous`. Structura generală a unei funcții de tip `anonymous` este următoarea:

`NumeFuncție=@(vi1, vi2, ...)Expresie`

în care `vi1, vi2, ...` sunt toate variabilele de intrare care se află în relația matematică `Expresie` prin care se definește funcția cu numele `NumeFuncție`.

- Principala deosebire dintre exprimarea funcțiilor prin metoda `inline` și metoda `anonymous` constă în modul de interacțiune între spațiul de lucru al programului și spațiul de lucru al funcției. În cazul funcțiilor definite prin metoda `inline`, toate variabilele din expresia matematică a funcției trebuie definite ca variabile de intrare care vor fi transferate funcției la apelarea acesteia, neexistând altă posibilitate de interacțiune cu spațiul de lucru al programului. Funcțiile de tip `anonymous` permit interacțiunea dintre spațiul de lucru al programului și spațiul de lucru al funcției atât prin transferul valorilor numerice ale variabilelor de intrare la apelarea funcției, cât și prin transferul valorilor numerice ale unor parametri care nu au fost definiți ca variabile de intrare dar care apar totuși în exprimarea matematică a funcției. În acest caz, parametrii respectivi trebuie definiți înainte de definirea funcției `anonymous`, transferul valorilor parametrilor respectivi realizându-se în mod permanent, chiar la momentul definirii funcției.
- Exprimarea funcțiilor prin metoda `anonymous` a fost introdusă odată cu versiunea MATLAB 7 pentru a înlocui metoda `inline`. Chiar dacă ambele metode sunt încă operaționale, se recomandă folosirea metodei `anonymous`.

### Problema 2.1

Pentru exemplificarea modurilor principale de lucru ale limbajului de programare MATLAB (lucrul în linia de comandă, metoda fișierelor `script`, metoda funcțiilor definite în fișiere de tip `function`, a funcțiilor `inline` și a funcțiilor `anonymous`) se consideră problema determinării mediei aritmetice  $\bar{x}$  a două variabile scalare  $x_1$  și  $x_2$ , având valorile  $x_1=18$  și  $x_2=32$ .

Relația de calcul a mediei aritmetice este:

$$\bar{x} = (x_1 + x_2)/2$$

Efectuând calculele se obține valoarea:

$$\bar{x} = (18 + 32)/2 = 25$$

## Rezolvare

Rezolvarea acestei probleme simple, folosind cele cinci metode de lucru, presupune parcurgerea următoarelor etape:

- **Modul de lucru în linie de comandă.** Se lucrează doar în fereastra de comenzi unde se scriu instrucțiunile și unde se obțin și rezultatele calculelor. În figura 2.10 se prezintă trei variante de rezolvare a problemei utilizând modul de lucru în linie de comandă. În varianta din figura 2.10, a) se observă modul de scriere a fiecărei instrucțiuni și apăsarea imediată a tastei `Enter` fără utilizarea la sfârșitul instrucțiunilor a caracterului `;`, rezultând astfel executarea instrucțiunilor și afișarea imediată a rezultatelor obținute atât după introducerea datelor inițiale, cât și după instrucțiunea de calcul a mediei aritmetice. În varianta din figura 2.10, b), s-a utilizat caracterul `;` după fiecare instrucțiune, mai puțin după instrucțiunea de calcul a mediei aritmetice, care furnizează de altfel și rezultatul final al calculelor. În varianta din figura 2.10, c) se observă modul de scriere a mai multor instrucțiuni pe o singură linie, folosind între instrucțiuni separatorul `;`. După instrucțiunea din linia a doua s-a tastat direct `Enter`, fără introducerea caracterului `;`, având în vedere faptul că se dorește vizualizarea rezultatului acestei instrucțiuni. Un dezavantaj al acestui mod de lucru este faptul că dacă se dorește calcularea mediei aritmetice pentru alte două valori numerice ale variabilelor  $x_1$  și  $x_2$ , trebuie rescrise în fereastra de comenzi toate instrucțiunile.

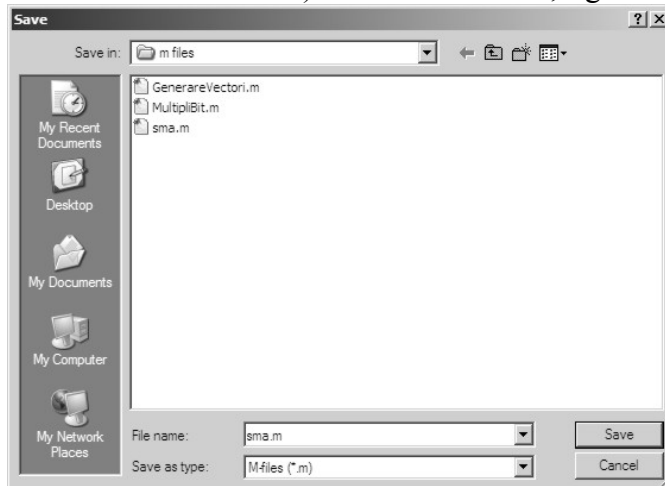
<code>&gt;&gt;x1=18</code>	<code>&gt;&gt;x1=18;</code>
<code>x1=</code>	<code>&gt;&gt;x2=32;</code>
<code>18</code>	<code>&gt;&gt;xm=(x1+x2)/2</code>
<code>&gt;&gt;x2=32</code>	<code>xm=</code>
<code>x2=</code>	<code>25</code>
<code>32</code>	<code>b)</code>
<code>&gt;&gt;xm=(x1+x2)/2</code>	<code>&gt;&gt;x1=18;x2=32;</code>
<code>xm=</code>	<code>&gt;&gt;xm=(x1+x2)/2</code>
<code>25</code>	<code>xm=</code>
	<code>25</code>
a)	c)

**Figura 2.10.** Calculul mediei aritmetice a două variabile scalare: modul de lucru în linie de comandă.

- **Modul de lucru bazat pe crearea unui fișier de tip script.** Din meniul `File` al programului se selectează comanda `New/Script` (sau `New/Blank M-File`). În fereastra de editare a fișierului `script` se scriu instrucțiunile necesare rezolvării problemei,



inclusiv valorile numerice ale variabilelor, se salvează fișierul (de exemplu cu numele sma.m), după care se lansează în execuție. În mod implicit, fișierul nou creat va fi salvat în directorul curent al limbajului de programare; se poate însă indica un alt director specificând calea de căutare către directorul dorit (de exemplu directorul cu numele m files) în fereastra Save, figura 2.11.



**Figura 2.11.** Specificarea directorului în care se va salva fișierul script.

În figura 2.12, a) se prezintă conținutul fișierul script pentru calculul mediei aritmetice. Se observă definirea a trei celule de calcul (%%): celula de titlu, celula datelor inițiale și celula blocului de calcul, precum și a două linii de text explicativ (%): pentru prima, respectiv a doua variabilă. După fiecare instrucțiune s-a introdus caracterul ;, mai puțin după instrucțiunea de calcul a mediei aritmetice, având în vedere faptul că se dorește vizualizarea în fereastra de comenzi doar a rezultatului executării acestei instrucțiuni, figura 2.12, b).

%% Calculul mediei aritmetice	xm=
clc;clear all; close all;	25
%% Date initiale	
%Prima variabila	
x1=18;	
%A doua variabila	
x2=32;	
%% Bloc de calcul	
xm=(x1+x2)/2	
a) fișierul script	b) rezultatul obținut

**Figura 2.12.** Calculul mediei aritmetice a două variabile scalare: modul de lucru bazat pe un fișier de tip script.

Dacă se dorește calcularea mediei aritmetice pentru alte două valori numerice ale variabilelor  $x_1$  și  $x_2$ , trebuie modificate, în structura fișierului `script`, doar cele două instrucțiuni de atribuire prin care se definesc cele două variabile, după care se lansează în execuție fișierul, obținându-se în fereastra de comenzi noul rezultat.

- **Modul de lucru bazat pe crearea unui fișier de tip `function`.** Din meniul `File` al programului se selectează comanda `New/Function` (sau `New/Function M-File`). În fereastra de editare a fișierului `function` se vor scrie instrucțiunile necesare rezolvării problemei, fără însă a specifica valorile numerice ale variabilelor, ci doar numele acestora, după care se salvează fișierul (de exemplu cu numele `fma.m`). În fereastra de comenzi (sau într-un fișier de tip `script` sau `function`) se specifică valorile numerice ale celor două variabile, după care se apelează numele fișierului `function` `fma`, transferând astfel celor două variabile ale funcției valorile numerice definite în fereastra de comenzi. Rezultatul executării fișierului `function` se obține tot în fereastra de comenzi. În figura 2.13 se prezintă atât fișierul de tip `function`, cât și fereastra de comenzi cu definirea valorilor numerice ale variabilelor, apelarea funcției și rezultatul obținut în urma executării funcției `fma`. Instrucțiunea de apelare a funcției realizează deschiderea fișierului de tip `function`, transferul către acest fișier a variabilelor numerice  $x_1=18$  și  $x_2=32$ , execuția instrucțiunii din interiorul funcției și în sfârșit, transferul rezultatului obținut către variabile `xm`. La apelarea funcției importantă este respectarea numărului și ordinii variabilelor de intrare ale funcției și nu numele acestora (`a` și `b` în interiorul funcției; `x1` și `x2` în exteriorul funcției).

```

1 function ma=fma(a,b)
2     ma=(a+b)/2;
3 end

```

a) definirea funcției

```

>> x1=18;x2=32;
>> xm=fma(x1,x2)
xm =
    25
fx >>

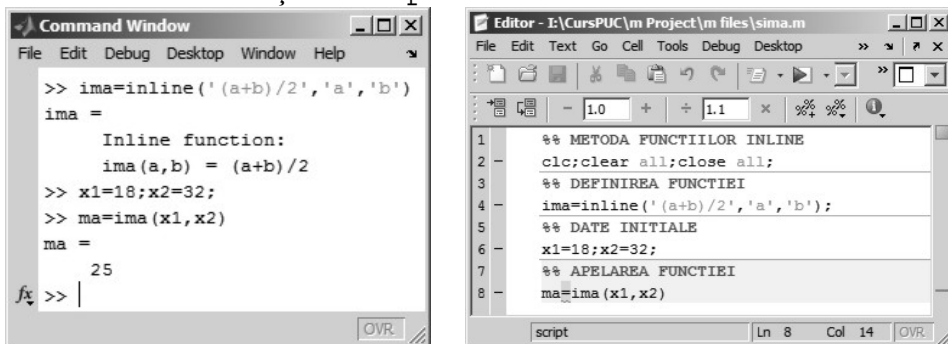
```

b) apelarea funcției

**Figura 2.13.** Calculul mediei aritmetice a două variabile scalare: modul de lucru bazat pe un fișier de tip `function`.

- **Modul de lucru bazat pe utilizarea funcțiilor de tip inline**

În figura 2.14, a) se prezintă rezolvarea problemei determinării mediei aritmetice a celor două numere folosind o funcție de tip inline definită direct în fereastra de comenzi. După definirea funcției se specifică valorile numerice ale celor două variabile, apoi se apelează funcția, transferând astfel celor două variabile locale ale funcției a și b, valorile numerice ale variabilelor x1 și x2 din spațiul de lucru al programului. În figura 2.14, b) se prezintă aceeași problemă rezolvată prin definirea funcției de tip inline în structura unui fișier script.



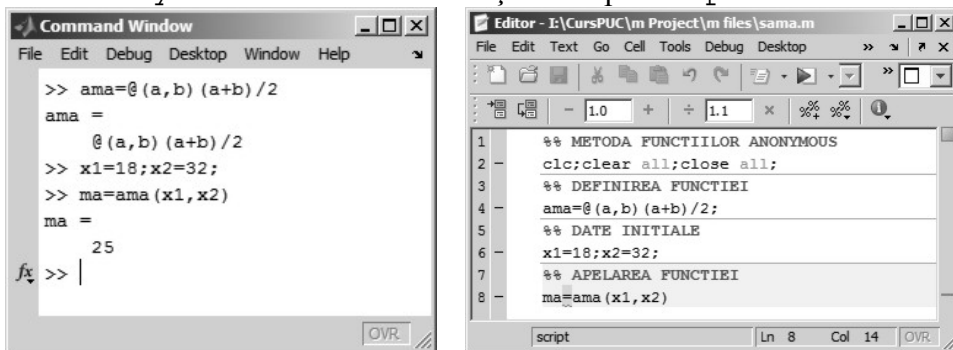
a) fereastra de comenzi

b) fișier de tip script

**Figura 2.14.** Calculul mediei aritmetice a două variabile scalare: modul de lucru bazat pe o funcție de tip inline.

- **Modul de lucru bazat pe utilizarea funcțiilor de tip anonymous**

Rezolvarea problemei determinării mediei aritmetice a celor două numere folosind o funcție de tip anonymous este prezentată în figura 2.15, a) pentru cazul lucrului direct în fereastra de comenzi și, respectiv în figura 2.15, b), pentru cazul definirii funcției anonymous în structura unui fișier de tip script.



a) fereastra de comenzi

b) fișier de tip script

**Figura 2.15.** Calculul mediei aritmetice a două variabile scalare: modul de lucru bazat pe o funcție de tip anonymous.

## 2.4. VARIABLE SCALARE. NUMERE. OPERATORI. ORDINEA OPERAȚIILOR

### 2.4.1. Declararea variabilelor

Declararea variabilelor se realizează prin operația de atribuire (=), prin care unei variabile oarecare, programul îi va atribui o anumită expresie:

```
NumeVariabilă=expresie
```

În mod implicit, programul operează doar cu variabile de tip matrice, astfel că la declararea unei variabile scalare, programul construiește o matrice având dimensiunea 1x1, cu o linie și o coloană.

Pentru a obține valoarea conținută în orice variabilă, la promptul programului se va tasta numele variabilei, urmat de tasta Enter:

```
x=exp(3.34);
```

```
x
```

```
x=
```

```
28.2191
```

Numele variabilei trebuie să înceapă cu o literă, putând apoi conține orice combinație de litere, numere și, eventual caracterul `_`.

```
VitezaMedie=3.5
```

```
alpha_0=pi/3
```

```
W_inf=15.81
```

```
ti=0
```

```
t_final=60
```

Declarația următoare va conduce la un mesaj de eroare datorită cifrei 3 care apare pe prima poziție în numele variabilei:

```
3_sigma=2.725
```

```
Error: The input character is not valid in  
MATLAB statements or expressions.
```

Pentru cazul variabilelor având nume formate din mai multe caractere, trebuie considerat faptul că doar primele `First_N` caractere sunt semnificative. Valoarea parametrului `First_N` se obține cu instrucțiunea:

```
First_N=namelengthmax
```

```
First_N=
```

```
63
```

În cazul în care, numele variabilei și caracterul = sunt omise, atunci programul va crea automat o variabilă generică numită `ans` (answer) care va conține rezultatul executării instrucțiunii respective.

```
4.32*log(3.2)/(6.54-345)*exp(3)
```

```
ans=
```

```
-0.2982
```

Programul nu solicită declararea inițială a tipului ori a dimensiunii unei variabile. La declararea unei variabile, programul creează automat variabila respectivă și alocă spațiul necesar de stocare. Dacă variabila există deja în spațiul de lucru al programului (de exemplu, în urma unei declarații anterioare), atunci programul modifică conținutul variabilei și, după caz realocă spațiul de stocare.

```
x=2.5
x=
    2.5000
y=2*x
y=
    5
x=3
x=
    3
z=2*x
z=
    6
```

Chiar dacă cele două variabile  $y$  și  $z$  se calculează cu aceeași relație ( $y = 2 \cdot x$  și  $z = 2 \cdot x$ ), se observă că  $y \neq z$  datorită faptului că pentru calculul variabilei  $y$ ,  $x = 2,5$ , în timp ce pentru calculul variabilei  $z$ ,  $x = 3$ .

### 2.4.2. Numere

Pentru reprezentarea numerelor reale, programul utilizează notația zecimală convențională cu punct zecimal, în virgulă fixă sau mobilă în funcție de parametrii de formatare stabiliți.

Domeniul de definiție al numerelor este finit, fiind cuprins între limitele definite prin variabilele `realmin` și `realmax`.

```
Limita_inf=realmin
Limita_inf=
    2.2251e-308
Limita_sup=realmax
Limita_sup=
    1.7977e+308
```

Formatul de afișare al numerelor pe ecran poate fi ales de utilizator modificând parametrul `Numeric format` (`File/Preferences/Command Window/Text display`). Principalele tipuri sunt: `short` (virgulă fixă cu 4 zecimale) și `long` (virgulă fixă cu 15 zecimale reprezentative). Formatele de reprezentare de tip `short e` și `long e` (virgulă mobilă cu 4 și respectiv cu 15 zecimale reprezentative) permit afișarea valorilor numerice conform formatului  $b \cdot 10^e$ , în care  $b$  este baza, iar  $e$  este exponentului:

$$3.2514e+004 \quad \Leftrightarrow \quad 3.2514 \cdot 10^{+4}$$

$$1.1028e-010 \quad \Leftrightarrow \quad 1.1028 \cdot 10^{-10}$$

Indiferent de formatul de afișare ales, toate numerele sunt stocate și manipulate la nivelul operațional al programului în formatul de reprezentare care asigură cele mai mici erori de rotunjire.

În cazul unor expresii matematice care generează valori numerice mai mari decât limita maximă `realmax`, sau în cazul unor operații aritmetice de împărțire a unei valori diferite de zero la zero, programul generează o variabilă specială denumită `Inf` (Infinity).

```
x=1.25*realmax
x=
    Inf
x=27.51/0
x=
    Inf
```

În cazul unor expresii matematice care conduc la operații de tipul `0/0`, `Inf-Inf` sau `Inf/Inf`, programul va genera o variabilă specială denumită `NaN` (Not-a-Number).

```
0/0
ans=
    NaN
Inf-Inf
ans=
    NaN
Inf/Inf
ans=
    NaN
```

Pentru reprezentarea numerelor complexe, se utilizează unitatea imaginară ( $\sqrt{-1}$ ) notată fie cu `i` fie cu `j`. Indiferent însă de unitatea imaginară utilizată, rezultatele returnate de program vor conține doar unitatea imaginară notată cu `i`.

```
i
ans=
    0 + 1.0000i
j
ans=
    0 + 1.0000i
```

Principalele metode pentru definirea numerelor complexe sunt:

- Metoda directă presupune scrierea directă a părții reale și a părții imaginare a numărului complex, care astfel se obține prin scrierea unei singure instrucțiuni de atribuire:

```
x=2.51+3.815*i
x=
    2.5100+3.8150i
```

- Metoda indirectă presupune definirea independentă a părții reale și a părții imaginare a numărului complex și utilizarea apoi a instrucțiunii `complex` pentru generarea numărului complex:

```
x_real=2.51;
x_imag=3.815;
x=complex(x_real,x_imag)
x=
    2.5100+3.8150i
```

### 2.4.3. Operatori

Limbajul de programare MATLAB lucrează cu trei tipuri de operatori: operatori aritmetici, operatori relaționali și operatori logici:

- Principalii operatori aritmetici utilizați la definirea expresiilor aritmetice sunt:

Operator	Descriere
+	Adunare, $x+y$
-	Scădere, $x-y$
*	Înmulțire, $x*y$
/	Împărțire, $x/y=x:y$
\	împărțire la stânga, $x\y=y:x$
^	ridicare la putere, $x^y$
'	Transpunere, $x'$

- Principalii operatori relaționali utilizați la compararea variabilelor sunt:

Operator	Descriere
<	Mai mic
>	Mai mare
<=	Mai mic sau egal
>=	Mai mare sau egal
==	Egal
~=	Diferit

- Principalii operatori logici utilizați pentru obținerea expresiilor logice sunt:

Operator	Descriere
&	AND
	OR
~	NOT

Ordinea de precedență a operatorilor stabilește ordinea în care se vor executa operațiile aritmetice, relaționale și logice din cadrul unei expresii. În

cadrul aceluiași nivel de precedență, operatorii având același nivel de precedență se vor evalua de la stânga la dreapta, mai puțin parantezele care se vor evalua de la interior la exterior.

Principalele reguli de precedență sunt:

Nivel de precedență	Operație
1	Paranteze, ()
2	Transpunere (') și ridicare la putere (^)
3	NOT (~)
4	Înmulțire (*) și împărțire (/, \)
5	Adunare (+) și scădere (-)
6	Operatorul :
7	Operatorii relaționali, <, <=, >, >=, ==, ~=
8	AND
9	OR

## 2.5. FUNCȚII MATEMATICE ELEMENTARE

Funcțiile matematice elementare definite în limbajul de programare MATLAB se grupează, în principal, în următoarele categorii: funcții exponențiale, logaritmice și putere; funcții trigonometrice; funcții hiperbolice; funcții pentru manipularea numerelor complexe; funcții pentru aproximarea numerelor.

### 2.5.1. Funcții exponențiale, logaritmice și putere

Funcția	Descriere
exp	Funcția exponențială, $\exp(x) = e^x$
expm1	Calculează $\exp(x) - 1$
log	Logaritm natural, $\log(x) = \ln(x)$
log10	Logaritm în baza 10, $\log_{10}(x) = \log_{10}(x)$
log2	Logaritm în baza 2, $\log_2(x) = \log_2(x)$
sqrt	Rădăcina pătrată, $\text{sqrt}(x) = \sqrt{x}$
pow2(e)	Calculează numărul $\text{pow2}(e) = 2^e$
pow2(m, e)	Calculează numărul real în virgulă mobilă $\text{pow2}(m, e) = m \cdot 2^e$

#### Observație

Pentru calculul logaritmului  $\log_a x$ , în care baza logaritmului  $a \notin \{2; e; 10\}$  nu există o instrucțiune predefinită pentru rezolvare. În acest caz se utilizează relațiile pentru transformarea logaritmului din baza  $a$  într-una din bazele pentru care există instrucțiuni de rezolvare  $b \in \{2; e; 10\}$  folosind relația generală:

$$\log_a x = \frac{\log_b x}{\log_b a}$$



### 2.5.2. Funcții trigonometrice

Funcția	Descriere
$\sin(x)$ $\cos(x)$ $\tan(x)$ $\cot(x)$ $\sec(x)$ $\csc(x)$	Funcții trigonometrice directe având argumentul $x$ exprimat în radiani.
$\text{sind}(x)$ $\text{cosd}(x)$ $\text{tand}(x)$ $\text{cotd}(x)$ $\text{secd}(x)$ $\text{cscd}(x)$	Funcții trigonometrice directe având argumentul $x$ exprimat în grade.
$\text{asin}(x)$ $\text{acos}(x)$ $\text{atan}(x)$ $\text{acot}(x)$ $\text{asec}(x)$ $\text{acsc}(x)$	Funcții trigonometrice inverse furnizând rezultatul în radiani.
$\text{asind}(x)$ $\text{acosd}(x)$ $\text{atand}(x)$ $\text{acotd}(x)$ $\text{asecd}(x)$ $\text{acscd}(x)$	Funcții trigonometrice inverse furnizând rezultatul în grade.

### 2.5.3. Funcții hiperbolice

Funcția	Descriere
Funcții hiperbolice directe	
$\sinh(x)$ $\cosh(x)$ $\tanh(x)$ $\coth(x)$ $\text{sech}(x)$ $\text{csch}(x)$	$\sinh x = \frac{e^x - e^{-x}}{2}; \cosh x = \frac{e^x + e^{-x}}{2};$ $\tanh x = \frac{\sinh x}{\cosh x}; \coth x = \frac{1}{\tanh x};$ $\text{sech } x = \frac{1}{\cosh x}; \text{csch } x = \frac{1}{\sinh x}$
Funcții hiperbolice inverse	
$\text{asinh}(x)$ $\text{acosh}(x)$ $\text{atanh}(x)$ $\text{acoth}(x)$ $\text{asech}(x)$ $\text{acsch}(x)$	$\sinh^{-1} x = \ln(x + \sqrt{x^2 + 1});$ $\cosh^{-1} x = \ln(x + \sqrt{x^2 - 1});$ $\tanh^{-1} x = \frac{1}{2} \ln\left(\frac{1+x}{1-x}\right); \coth^{-1} x = \tanh^{-1}\left(\frac{1}{x}\right)$ $\text{sech}^{-1} x = \cosh^{-1}(1/x); \text{csch}^{-1} x = \sinh^{-1}(1/x)$

### 2.5.4. Funcții pentru manipularea numerelor complexe

Funcția	Descriere
<code>i</code> , <code>j</code>	Unitatea imaginară implicită.
<code>real(z)</code>	Determină partea reală a numărului complex $z$ .
<code>imag(z)</code>	Determină partea imaginară a numărului complex $z$ .
<code>z=complex(a,b)</code> <code>z=a+bi</code>	Definește numărul complex $z$ având partea reală $a$ și partea imaginară $b$ .
<code>zc=conj(z)</code>	Calculează conjugatul numărului complex $z$ .
<code>r=abs(z)</code>	Calculează modulul $r$ al numărului complex $z = x + i \cdot y$ conform relației: $r = \sqrt{x^2 + y^2}$
<code>phi=angle(z)</code>	Calculează argumentul $\varphi$ al numărului complex $z = x + i \cdot y$ conform relației: $\varphi = \tan^{-1}\left(\frac{y}{x}\right)$
<code>z=r*exp(i*phi)</code> <code>z=r*(cos(phi)+i*sin(phi))</code>	Definește forma polară a numărului complex conform relației: $z = r \cdot e^{i\varphi}$
<code>isreal(z)</code>	Verifică dacă numărul $z$ este real sau nu. Rezultatul funcției este 1 dacă numărul $z$ este real, și 0 dacă numărul $z$ este complex.

### 2.5.5. Funcții pentru aproximarea numerelor

Funcția	Descriere
<code>round(x)</code>	Rotunjește numărul $x$ la cel mai apropiat număr întreg.
<code>floor(x)</code>	Rotunjește numărul $x$ la cel mai apropiat număr întreg spre $-\infty$ .
<code>ceil(x)</code>	Rotunjește numărul $x$ la cel mai apropiat număr întreg spre $+\infty$ .
<code>fix(x)</code>	Rotunjește numărul $x$ la cel mai apropiat număr întreg spre 0.
<code>rats(x)</code>	Aproximarea numărului $x$ folosind exprimarea cu numere raționale.
<code>rat(x)</code>	Aproximarea numărului $x$ folosind exprimarea cu fracții continue.

## Problema 2.2

Se consideră variabilele:

$$a=21; b=3,25; c=10,012$$

Folosind modul de lucru în linie de comandă să se calculeze următoarele expresii:

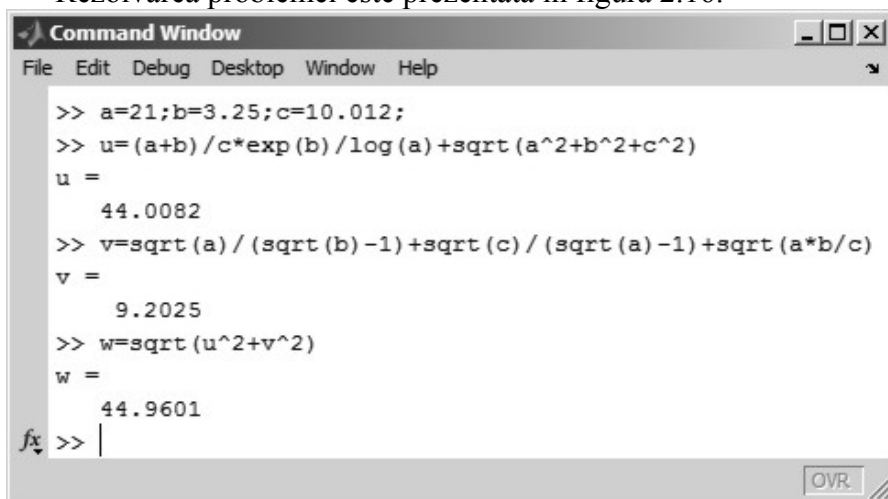
$$u = \frac{a+b}{c} \cdot \frac{e^b}{\ln a} + \sqrt{a^2 + b^2 - c^2}$$

$$v = \frac{\sqrt{a}}{\sqrt{b}-1} + \frac{\sqrt{c}}{\sqrt{a}-1} + \sqrt{\frac{a \cdot b}{c}}$$

$$w = \sqrt{u^2 + v^2}$$

## Rezolvare

Rezolvarea problemei este prezentată în figura 2.16.



```
Command Window
File Edit Debug Desktop Window Help
>> a=21;b=3.25;c=10.012;
>> u=(a+b)/c*exp(b)/log(a)+sqrt(a^2+b^2+c^2)
u =
    44.0082
>> v=sqrt(a)/(sqrt(b)-1)+sqrt(c)/(sqrt(a)-1)+sqrt(a*b/c)
v =
     9.2025
>> w=sqrt(u^2+v^2)
w =
    44.9601
fx >> |
```

Figura 2.16. Rezolvarea problemei 2.2.

## Observații

- Numărătorul primei fracții din expresia variabilei  $u$  trebuie scris între paranteze rotunde pentru că instrucțiunea  $(a+b)/c$  corespunde relației  $\frac{a+b}{c}$ . În cazul în care se omite scrierea parantezelor se obține o alterare a instrucțiunii pentru că instrucțiunea  $a+b/c$  nu corespunde relației  $\frac{a+b}{c}$ , ci relației  $a + \frac{b}{c}$ .
- Numitorii primelor două fracții din expresia variabilei  $v$  trebuie, de asemenea, scriși între paranteze rotunde.
- Numărătorul ultimei fracții din expresia variabilei  $v$  nu trebuie scris între paranteze pentru că atât instrucțiunea  $a*b/c$  cât și instrucțiunea  $(a*b)/c$  corespund aceleiași relații,  $\frac{a \cdot b}{c}$ . Nu trebuie făcut exces de paranteze.

- Argumentele de intrare ale funcțiilor trebuie scrise între paranteze rotunde. În această problemă, este cazul funcției exponențiale  $\exp(b)$ , a funcției logaritm natural  $\log(a)$ , precum și a funcțiilor radical indice 2, de exemplu  $\sqrt{a}$ .
- În cazul produsului a două variabile, nu trebuie să se omită operatorul de înmulțire dintre cele două variabile. De exemplu, pentru a scrie produsul variabilelor  $a$  și  $b$ , instrucțiunea corectă este  $a*b$ , și nu  $ab$ .
- Se recomandă ca la scrierea unei expresii între două paranteze rotunde să se scrie mai întâi ambele paranteze și abia apoi să se scrie expresia. În acest mod, mai ales în cazul expresiilor complexe, se poate ține o evidență mai clară asupra perechilor de paranteze din cadrul unei instrucțiuni.

### Problema 2.3

Se consideră variabilele:

$$\alpha = 30^\circ \text{ și } \beta = 15^\circ$$

Să se calculeze următoarea expresie trigonometrică:

$$u = \frac{\sin \alpha}{\cos \alpha + \cos \beta}$$

### Rezolvare

Rezolvarea problemei este prezentată în figura 2.17, a) pentru cazul funcțiilor trigonometrice care necesită exprimarea unghiurilor în grade ( $\text{sind}$  și  $\text{cosd}$ ) și în figura 2.17, b) pentru cazul funcțiilor trigonometrice care necesită exprimarea unghiurilor în radiani ( $\sin$  și  $\cos$ ).

```

Command Window
File Edit Debug Desktop Window Help
>> alpha=30;beta=15;
>> u=sind(alpha)/(cosd(alpha)+cosd(beta))
u =
0.2729
fx >> |
OVR

```

a) unghiuri exprimate în grade

```

Command Window
File Edit Debug Desktop Window Help
>> alpha=30;beta=15;
>> alphas=alpha*pi/180;
>> betas=beta*pi/180;
>> u=sin(alphas)/(cos(alphas)+cos(betas))
u =
0.2729
fx >> |
OVR

```

a) unghiuri exprimate în radiani

**Figura 2.17.** Rezolvarea problemei 2.3.

### Observații

- În editorul de fișiere nu se pot scrie caractere grecești, deci se impune redenumirea variabilelor, de exemplu:  $\alpha \rightarrow \text{alpha}$  și  $\beta \rightarrow \text{beta}$ .
- La utilizarea funcțiilor trigonometrice trebuie avut în vedere modul de exprimare a variabilelor (grade sau radiani). De exemplu, pentru calculul funcției sinus, se va folosi:

- instrucțiunea  $\sin \alpha$ , pentru unghiul  $\alpha$  exprimat în radiani,
- instrucțiunea  $\text{sind } \alpha$ , pentru unghiul  $\alpha$  exprimat în grade.
- În anumite cazuri este necesară efectuarea transformărilor unităților de măsurare ale unghiurilor:
  - Trecerea de la exprimarea în grade la exprimarea în radiani se face conform relației de transformare:

$$\alpha_r = \alpha_g \cdot \frac{\pi}{180}$$

- Trecerea de la exprimarea în radiani la exprimarea în grade se face conform relației de transformare:

$$\alpha_g = \alpha_r \cdot \frac{180}{\pi}$$

### Problema 2.4

Se consideră variabilele:

$$\alpha = 30^\circ \text{ și } \beta = \pi/2$$

Să se calculeze următoarele expresii trigonometrice:

$$u = \frac{\pi}{3} \sin(\alpha + \beta)$$

a)

$$v = \frac{1 - \sin \alpha}{1 + \cos \beta}$$

b)

### Rezolvare

Rezolvarea problemei este prezentată în figura 2.18.

```

Command Window
File Edit Debug Desktop Window Help
>> a=30;b=pi/2;
>> u=pi/3*sin(a*pi/180+b)
u =
    0.9069
>> v=(1-sind(a))/(1+cos(b))
v =
    0.5000
fx >> |
  
```

Figura 2.18. Rezolvarea problemei 2.4.

### Observații:

- Trebuie redenumite variabilele, de exemplu:  $\alpha \rightarrow a$  și  $\beta \rightarrow b$ .
- În instrucțiunea corespunzătoare variabilei  $u$ , s-a utilizat instrucțiunea  $\text{sin}$  care necesită ca argumentul său să fie exprimat în radiani. Transformarea unghiului  $\alpha$  din grade în radiani a fost făcută direct în instrucțiunea principală, fără definirea unei variabile intermediare.
- În instrucțiunea corespunzătoare variabilei  $v$ , s-au utilizat funcțiile trigonometrice corespunzătoare argumentului, fără definirea unor variabile intermediare.

## Problema 2.5

Se consideră variabilele:

$$a=1,12; b=2,28; c=5,12$$

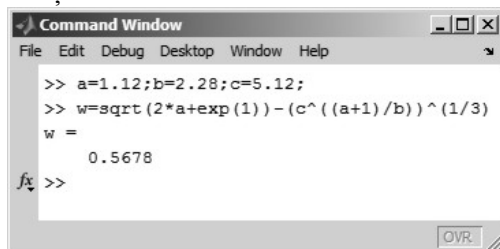
Folosind modul de lucru în linie de comandă să se calculeze:

$$w = \sqrt[2]{2a + e} - \sqrt[3]{c \frac{a+1}{b}}$$

în care  $e$  este baza logaritmului natural.

### Rezolvare

Rezolvarea problemei este prezentată în figura 2.19. Pe prima linie se introduc valorile numerice ale datelor de intrare separate prin caracterul  $;$ , apoi se scrie instrucțiunea care realizează calculul variabilei  $w$ .



```
Command Window
File Edit Debug Desktop Window Help
>> a=1.12;b=2.28;c=5.12;
>> w=sqrt(2*a+exp(1))-(c^((a+1)/b))^(1/3)
w =
    0.5678
fx >>
```

Figura 2.19. Rezolvarea problemei 2.5.

### Observații

- Pentru exprimarea funcției radical indice 2 ( $\sqrt{\quad}$ ) se folosește instrucțiunea `sqrt`. Nu există însă o instrucțiune specifică pentru exprimarea funcției radical indice 3 ( $\sqrt[3]{\quad}$ , sau în caz general, radical indice  $n$ ,  $\sqrt[n]{\quad}$ ). În aceste situații trebuie folosită echivalența algebrică:

$$\sqrt[n]{x} = x^{1/n}$$

care conduce la o instrucțiune de forma  $x^{1/n}$ . Dacă din eroare, se omite scrierea parantezelor, adică  $x^{1/n}$ , se obține o alterare a relației matematice, care în acest caz devine  $\frac{x^1}{n} \neq x^{\frac{1}{n}}$ .

- Considerând faptul că pentru exprimarea funcției exponențiale  $e^x$  trebuie utilizată instrucțiunea `exp(x)`, rezultă că obținerea bazei logaritmului natural  $e$ , se poate face cu instrucțiunea `exp(1)`.
- Puterea variabilei  $c$  este  $\frac{a+1}{b}$ , deci trebuie folosite două paranteze: una pentru definirea numărătorului  $a+1$  și cea de-a doua pentru definirea puterii, rezultând astfel instrucțiunea: `c^((a+1)/b)`. O instrucțiune de forma `c^(a+1)/b`, conduce la un rezultat eronat datorită faptului că este echivalentă cu relația  $\frac{c^{a+1}}{b} \neq c^{\frac{a+1}{b}}$ .
- Pentru exprimarea numerelor zecimale se folosește punctul zecimal.

## Problema 2.6

Se consideră un deversor dreptunghiular având lățimea  $b=10$  m și înălțimea pragului amonte față de baza canalului  $p=2$  m. Deversorul lucrează sub sarcina constantă  $H=0,5$  m.

Debitul deversat  $Q$  [ $\text{m}^3/\text{s}$ ] se calculează cu relația:

$$Q = \frac{2}{3} \mu b \sqrt{2g} \cdot H^{\frac{3}{2}}$$

în care  $g=9,81$   $\text{m/s}^2$  este accelerația gravitațională, iar  $\mu$  este coeficientul de debit care se poate calcula cu relațiile:

- relația Rehbock:

$$\mu_R = \left( 0.6035 + 0.0813 \frac{H}{p} + \frac{0.00009}{p} \right) \cdot \left( 1 + \frac{0.0011}{H} \right)^{\frac{3}{2}}$$

- relația Bazin:

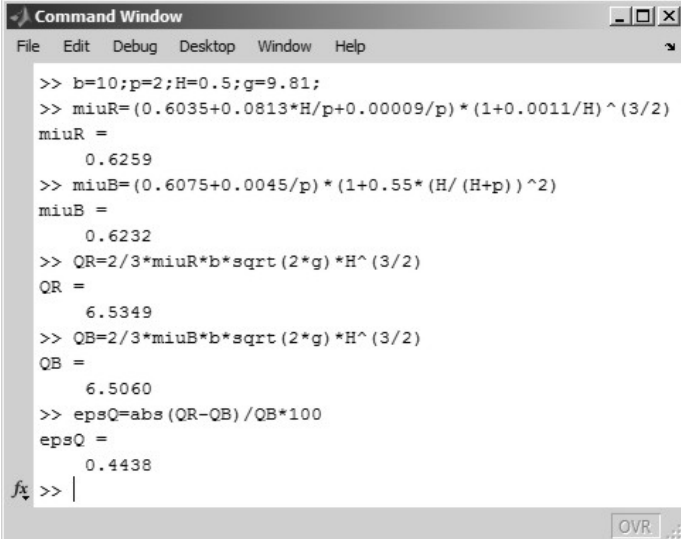
$$\mu_B = \left( 0.6075 + \frac{0.0045}{p} \right) \cdot \left[ 1 + 0.55 \left( \frac{H}{H+p} \right)^2 \right]$$

Considerând că debitul calculat cu relația Bazin ( $Q_B$ ) este valoarea convențional adevărată, să se determine eroarea relativă introdusă în calculul debitului prin exprimarea coeficientului de debit cu relația Rehbock:

$$\varepsilon_Q = \frac{|Q_R - Q_B|}{Q_B} 100 [\%]$$

## Rezolvare

Rezolvarea problemei folosind metoda de lucru în linie de comandă este prezentată în figura 2.20.



```
Command Window
File Edit Debug Desktop Window Help
>> b=10;p=2;H=0.5;g=9.81;
>> miuR=(0.6035+0.0813*H/p+0.00009/p)*(1+0.0011/H)^(3/2)
miuR =
    0.6259
>> miuB=(0.6075+0.0045/p)*(1+0.55*(H/(H+p))^2)
miuB =
    0.6232
>> QR=2/3*miuR*b*sqrt(2*g)*H^(3/2)
QR =
    6.5349
>> QB=2/3*miuB*b*sqrt(2*g)*H^(3/2)
QB =
    6.5060
>> epsQ=abs(QR-QB)/QB*100
epsQ =
    0.4438
fx >> |
```

Figura 2.20. Rezolvarea problemei 2.6 prin metoda de lucru în linie de comandă.

Rezolvarea problemei folosind metoda fișierelor script este prezentată în figura 2.21. În structura fișierului script se observă existența celulelor de calcul, a datelor inițiale și a instrucțiunilor de calcul, fără însă a se utiliza funcții definite de utilizator. Rezultatul obținut în urma lansării în execuție a fișierului script este prezentat în figura 2.22.

```

Editor - G:\CursPUC\m Project\m files\DebitRB_script.m
File Edit Text Go Cell Tools Debug Desktop Window Help
- 1.0 + 1.1 x % %
1 %% PROGRAM pentru calculul debitului unui deversor
2 % Tipul deversorului: dreptunghilar
3 % Metoda fisierelor script
4 - clc;clear all;close all;
5 %% DATE INITIALE
6 % Latimea deversorului [m]
7 - b=10;
8 % Inaltimea pragului amonte [m]
9 - p=2;
10 % Sarcina [m]
11 - H=0.5;
12 % Acceleratia gravitacionala [m/s2]
13 - g=9.81;
14 %% BLOC DE CALCULE
15 % Coeficientul de debit Rehbock
16 - miuR=(0.6035+0.0813*H/p+0.00009/p)*(1+0.0011/H)^(3/2)
17 % Coeficientul de debit Bazin
18 - miuB=(0.6075+0.0045/p)*(1+0.55*(H/(H+p))^2)
19 % Debitul Rehbock QR [m3/s]
20 - QR=2/3*miuR*b*sqrt(2*g)*H^(3/2)
21 % Debitul Bazin QB [m3/s]
22 - QB=2/3*miuB*b*sqrt(2*g)*H^(3/2)
23 % Eroarea relativa [%]
24 - epsQ=abs(QR-QB)/QB*100
Ln 1 Col 1 OVR

```

Figura 2.21. Conținutul fișierului script pentru rezolvarea problemei 2.6.

```

Command Window
File Edit Debug Desktop Window Help
miuR =
    0.6259
miuB =
    0.6232
QR =
    6.5349
QB =
    6.5060
epsQ =
    0.4438
fx >> |
OVR

```

Figura 2.22. Rezultatul obținut prin metoda fișierului script.



Rezolvarea problemei folosind metoda funcțiilor definite în fișiere de tip `function` presupune crearea a două fișiere de tip `function` care permit calcularea coeficientului de debit și a debitului prin cele două metode (Rehbock, figura 2.23 și Bazin, figura 2.24), precum și a unui fișier de tip `script` (figura 2.25) prin care se introduc datele inițiale, se apelează cele două funcții și se calculează eroarea relativă. Se observă modul de definire a unei variabile de tip global în ambele fișiere de tip `function`, precum și în fișierul de tip `script`. Atribuirea unei valori numerice variabilei globale se realizează în structura fișierului `script`, astfel încât după lansarea acestuia în execuție, valoarea numerică a variabilei globale se va găsi în spațiul de lucru al programului de unde poate fi utilizată de cele două fișiere de tip `function`. Rezultatul obținut în urma lansării în execuție a fișierului de tip `script` este prezentat în figura 2.26.

```

1 function [miuR, QR] = fQR(x, y, z)
2 % Funcție pentru calculul coeficientului de debit
3 % și a debitului unui deversor folosind relația Rehbock
4 global g
5 % Coeficientul de debit miuR
6 miuR=(0.6035+0.0813*z/y+0.00009/y)*(1+0.0011/z)^(3/2);
7 % Debitul QR [m3/s]
8 QR=2/3*miuR*x*sqrt(2*g)*z^(3/2);
9 end

```

**Figura 2.23.** Fișier `function` pentru relația Rehbock.

```

1 function [miuB, QB] = fQB(x, y, z)
2 % Funcție pentru calculul coeficientului de debit
3 % și a debitului unui deversor folosind relația Bazin
4 global g
5 % Coeficientul de debit miuB
6 miuB=(0.6075+0.0045/y)*(1+0.55*(z/(z+y))^2);
7 % Debitul QB [m3/s]
8 QB=2/3*miuB*x*sqrt(2*g)*z^(3/2);
9 end

```

**Figura 2.24.** Fișier `function` pentru relația Bazin.

```

1 %% PROGRAM pentru calculul debitului unui deversor
2 % Tipul deversorului: dreptunghilar
3 % Metoda fisierelor function
4 - clc;clear all;close all;
5 %% DATE INITIALE
6 % Latimea deversorului [m]
7 - b=10;
8 % Inaltimea pragului amonte [m]
9 - p=2;
10 % Sarcina [m]
11 - H=0.5;
12 % Acceleratia gravitationala [m/s2]
13 - global g
14 - g=9.81;
15 %% BLOC DE CALCULE
16 % Coeficientul de debit si debitul (Rehbock)
17 - [miuR,QR]=fQR(b,p,H)
18 % Coeficientul de debit si debitul (Bazin)
19 - [miuB,QB]=fQB(b,p,H)
20 % Eroarea relativa [%]
21 - epsQ=abs(QR-QB)/QB*100

```

**Figura 2.25.** Fișierul script pentru metoda funcțiilor definite în fișiere de tip function.

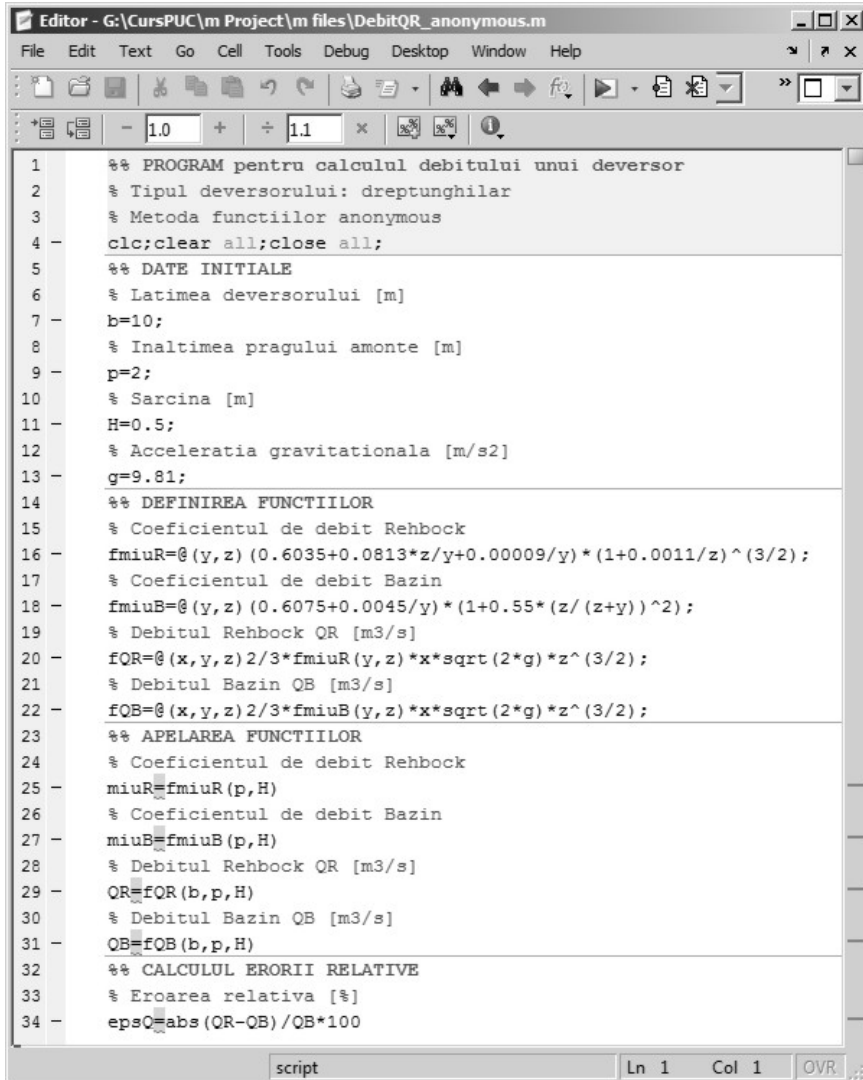
```

miuR =
    0.6259
QR =
    6.5349
miuB =
    0.6232
QB =
    6.5060
epsQ =
    0.4438
fx >> |

```

**Figura 2.26.** Rezultatul obținut prin metoda funcțiilor definite în fișiere de tip function.

Pentru rezolvarea problemei folosind funcții de tip anonymous se prezintă în figura 2.27 structura unui fișier de tip script care conține: definirea datelor inițiale; definirea funcțiilor pentru coeficienții de debit și pentru debite; apelarea funcțiilor prin care se transferă acestora valorile numerice ale variabilelor corespunzătoare și se realizează astfel calculul numeric; calculul erorii relative. Rezultatele obținute în urma rulării acestui fișier script sunt prezentate în figura 2.28.



```
1 %% PROGRAM pentru calculul debitului unui deversor
2 % Tipul deversorului: dreptunghilar
3 % Metoda functiilor anonymous
4 - clc;clear all;close all;
5 %% DATE INITIALE
6 % Latimea deversorului [m]
7 - b=10;
8 % Inaltimea pragului amonte [m]
9 - p=2;
10 % Sarcina [m]
11 - H=0.5;
12 % Acceleratia gravitationala [m/s^2]
13 - g=9.81;
14 %% DEFINIREA FUNCTIILOR
15 % Coeficientul de debit Rehbock
16 - fmiuR=@(y,z)(0.6035+0.0813*z/y+0.00009/y)*(1+0.0011/z)^(3/2);
17 % Coeficientul de debit Bazin
18 - fmiuB=@(y,z)(0.6075+0.0045/y)*(1+0.55*(z/(z+y))^2);
19 % Debitul Rehbock QR [m^3/s]
20 - fQR=@(x,y,z)2/3*fmiuR(y,z)*x*sqrt(2*g)*z^(3/2);
21 % Debitul Bazin QB [m^3/s]
22 - fQB=@(x,y,z)2/3*fmiuB(y,z)*x*sqrt(2*g)*z^(3/2);
23 %% APELAREA FUNCTIILOR
24 % Coeficientul de debit Rehbock
25 - miuR=fmiuR(p,H)
26 % Coeficientul de debit Bazin
27 - miuB=fmiuB(p,H)
28 % Debitul Rehbock QR [m^3/s]
29 - QR=fQR(b,p,H)
30 % Debitul Bazin QB [m^3/s]
31 - QB=fQB(b,p,H)
32 %% CALCULUL ERORII RELATIVE
33 % Eroarea relativa [%]
34 - epsQ=abs(QR-QB)/QB*100
```

Figura 2.27. Fișierul script pentru metoda funcțiilor de tip anonymous.



```
miuR =
    0.6259
miuB =
    0.6232
QR =
    6.5349
QB =
    6.5060
epsQ =
    0.4438
fx >> |
```

Figura 2.28. Rezultatul obținut prin metoda funcțiilor anonymous.

## BIBLIOGRAFIE

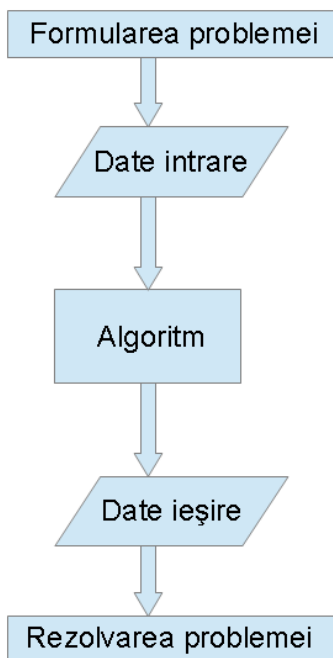
1. MathWorks, MATLAB, 3D Visualization, 2014.
2. MathWorks, MATLAB, Control System Toolbox, Getting Started Guide, 2014.
3. MathWorks, MATLAB, Creating Graphical User Interface, 2014.
4. MathWorks, MATLAB, Desktop Tools and Development Environment, 2014.
5. MathWorks, MATLAB, Graphics, 2014.
6. MathWorks, MATLAB, Symbolic Math Toolbox, User's Guide, 2014.
7. MathWorks, MATLAB, Mathematics, 2014.
8. MathWorks, MATLAB. MuPAD, User's Guide, 2014.
9. MathWorks, MATLAB, Optimization Toolbox, User's Guide, 2014.
10. MathWorks, MATLAB, Parallel Computing Toolbox, User's Guide, 2014.
11. MathWorks, MATLAB, Partial Differential Equation Toolbox, User's Guide, 2014.
12. MathWorks, MATLAB, Primer, 2014.
13. MathWorks, MATLAB, Signal Processing Toolbox, User's Guide, 2014.
14. MathWorks, MATLAB, SimHydraulics, User's Guide, 2014.
15. MathWorks, MATLAB, Simscape, User's Guide, 2014.
16. MathWorks, MATLAB, Simulink, Getting Started Guide, 2014.
17. MathWorks, MATLAB, Simulink, User's Guide, 2014.
18. MathWorks, MATLAB, Spreadsheet Link EX, User's Guide, 2014.
19. MathWorks, Cleve Moler, Chief Mathematician, Chairman, and Cofounder of MathWorks, [http://www.mathworks.com/company/aboutus/founders/cleve\\_moler.html](http://www.mathworks.com/company/aboutus/founders/cleve_moler.html), accesat la data 28.01.2014.
20. MathWorks, Jack Little, President and Cofounder of MathWorks, <http://www.mathworks.com/company/aboutus/founders/jacklittle.html>, accesat la data 28.01.2014.
21. MathWorks, <http://www.mathworks.com/>, accesat la data 28.01.2014.
22. MathWorks, Simscape Toolbox, <http://www.mathworks.com/products/simscape/>, accesat la data 28.01.2014.
23. MathWorks, Simulink Toolbox, <http://www.mathworks.com/products/simulink/>, accesat la data 28.01.2014.
24. MathWorks, MATLAB, <http://www.mathworks.com/>, accesat la data 28.01.2014.
25. Oracle, Create the Future with Java, <https://www.oracle.com/java/index.html>, accesat la 25.08.2014.
26. Oracle, Oracle and Sun Microsystems, <http://www.oracle.com/us/sun/index.html>, accesat la 25.08.2014.
27. SciFace Software, MuPAD, <http://www.mupad.com>, accesat la 17.05.2014.
28. Wikipedia, Java Virtual Machine, [http://en.wikipedia.org/wiki/Java\\_virtual\\_machine](http://en.wikipedia.org/wiki/Java_virtual_machine), accesat la 25.08.2014.
29. Wikipedia, MATLAB, <http://en.wikipedia.org/wiki/MATLAB>, accesat la data 28.01.2014.

## CAPITOLUL 3

### ALGORITMI ȘI SCHEME LOGICE

#### 3.1. DEFINIȚII. PROPRIETĂȚI

**Algoritmul** [1, 2, 3, 4, 5, 6, 15, 17, 21], este o metodă de rezolvarea a unei anumite probleme prin care, pornind de la datele de intrare și parcurgând o anumită succesiune finită de etape, se ajunge în final, la o serie de date de ieșire care reprezintă soluția problemei respective, figura 3.1.



**Figura 3.1.** Algoritmul-calea de rezolvare a unei probleme.

**Datele de intrare** reprezintă toate informațiile necesare pentru rezolvarea problemei respective și rezultă dintr-o bună înțelegere a problemei de rezolvat. Aproximarea și reducerea numărului datelor de intrare pot conduce la reducerea complexității algoritmului, dar și la creșterea gradului de aproximarea a soluției obținute.

**Datele de ieșire** reprezintă datele finale care rezultă în urma aplicării algoritmului de calcul asupra datelor de intrare. Algoritmul de calcul poate să furnizeze și o serie de rezultate intermediare, care au relevanță doar în contextul etapelor intermediare ale algoritmului. Utilizatorul, pe baza problemei de rezolvat, va decide care din rezultatele intermediare reprezintă și datele de ieșire relevante pentru soluția problemei.

Principale proprietăți ale algoritmilor sunt:

- **Determinarea.** Algoritmul trebuie să fie astfel conceput încât toate etapele, ca și ordinea lor de executare, să fie clare, concise și fără ambiguități.
- **Generalitatea.** Algoritmul trebuie să permită rezolvarea unei familii de probleme și nu a unei probleme particulare.
- **Finitudinea.** Algoritmul poate să conțină un număr oricât de mare de etape, totuși, numărul etapelor trebuie să fie finit.

Principalele obiecte care intră în structura algoritmilor sunt:

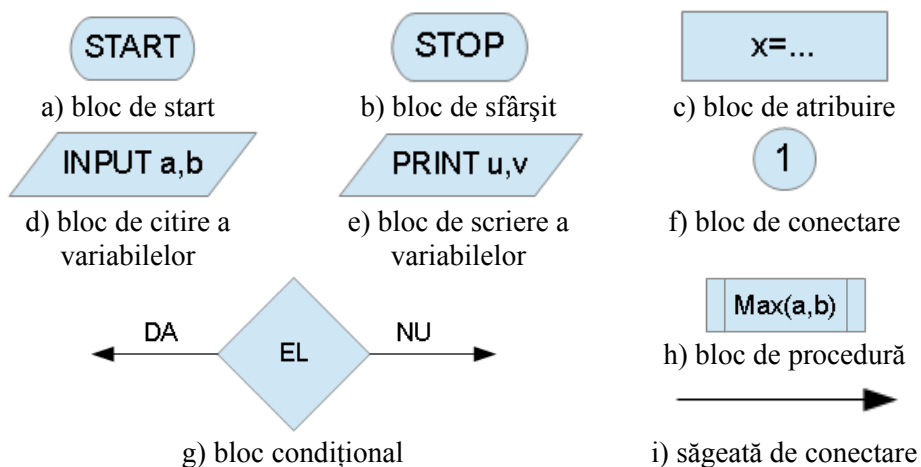
- **Constantele** - reprezintă date (numerice, alfanumerice, logice) care nu se modifică pe parcursul executării unui algoritm.
- **Variabilele** - reprezintă datele care se modifică pe parcursul executării unui algoritm.
- **Operatorii** - reprezintă operațiile care se execută asupra constantelor și variabilelor unui algoritm. Principalele tipuri de operatori sunt:
  - Operatori aritmetici: adunare +, scădere -, înmulțire \*, împărțire /.
  - Operatori relaționali: mai mare >, mai mic <, mai mare sau egal >=, mai mic sau egal <=, egal =, diferit <math>\neq</math>.
  - Operatori logici: conjuncție logică AND, disjuncție logică OR, negație logică NOT.
- **Expresiile** - reprezintă ansamblul constantelor, variabilelor și al operatorilor dintre acestea. Expresiile pot fi aritmetice, relaționale și logice.

### 3.2. METODE DE REPREZENTARE A ALGORITMILOR

Reprezentarea algoritmilor este independentă de un anumit limbaj de programare și poate fi realizată prin diferite metode: metoda schemelor logice, metode de tip pseudocod.

#### 3.2.1. Metoda schemelor logice

Reprezintă un limbaj în care diferitele elemente ale unui algoritm sunt reprezentate cu ajutorul unor simboluri grafice. Principalele simboluri grafice specifice realizării schemelor logice sunt prezentate în figura 3.2, [16, 18, 21].



**Figura 3.2.** Simboluri grafice pentru metoda schemelor logice.

## Observații

- Blocurile de start (START) și de sfârșit (STOP) (figura 3.2, a și figura 3.2, b), denumite și blocuri de tip terminal, au doar rolul de a defini punctele terminale (de început și de sfârșit) ale schemei logice. Orice schemă logică are un bloc de start și un bloc de sfârșit.
- Blocul de atribuire (figura 3.2, c) permite definirea unor variabile prin intermediul expresiilor.
- Blocurile de citire (INPUT) și de scriere (PRINT) (figura 3.2, d și figura 3.2, e) a variabilelor sunt responsabile cu introducerea datelor de intrare (a, b), respectiv cu prezentarea datelor de ieșire (u, v) obținute după executarea schemei logice.
- Blocul de conectare (figura 3.2, f), permite gruparea a două intrări într-o singură ieșire, este identificat printr-un număr și ajută la modularizarea schemei logice.
- Blocul condițional (figura 3.2, g), introduce în schema logică un punct de ramificare pe baza unei expresii logice. În funcție de valoarea de adevăr a acestei expresii logice (adevărat sau fals), blocul condițional realizează o ieșire alternativă, pe una sau cealaltă dintre căile de ieșire ale blocului.
- Blocul de procedură (figura 3.2, h) permite compactarea unei porțiuni a algoritmului într-un singur bloc, asigurând astfel modularizarea schemelor logice. În interiorul blocului se specifică numele simbolic al procedurii, variabilele de intrare și cele de ieșire.
- Săgeata de conectare (figura 3.2, i), asigură conexiunea dintre diferitele blocuri ale unei scheme logice. Săgeata de conectare asigură transferul informației într-un singur sens (specificat prin orientarea săgeții de la capătul liniei).

### 3.2.2. Metode de tip pseudocod

Reprezintă un limbaj în care diferitele elemente ale unui algoritm sunt reprezentate cu ajutorul unor cuvinte cheie. Cuvintele cheie au în corespondență instrucțiuni specifice în diferite limbaje de programare. Principalele cuvinte cheie (în limba română) ale metodei de tip pseudocod sunt prezentate în figura 3.3, [19, 21].

<b>start</b>	<b>citește a,b</b>	<b>x=expresie</b>
<b>stop</b>	<b>scrie u,v</b>	
a) comenzile de start și de sfârșit	b) comenzile de citire și de scriere a variabilelor	c) comanda de atribuire
<b>cât timp</b> expresie logică	<b>execută</b>	<b>repetă</b>
instrucțiune 1		instrucțiune 1
instrucțiune 2		instrucțiune 2
<b>sfârșit cât timp</b>		<b>până când</b> expresie logică
d) structura <b>cât timp</b>		e) structura <b>repetă până când</b>

<b>dacă</b> expresie logică <b>atunci</b> instrucțiune 1	<b>dacă</b> expresie logică <b>atunci</b> instrucțiune 1
<b>altfel</b> instrucțiune 2	instrucțiune 2
<b>sfârșit dacă</b> f) structură condițională 1	<b>sfârșit dacă</b> g) structură condițională 2
<b>pentru</b> variabilă=valoare inițială, valoare finală <b>execută</b> instrucțiune 1	
instrucțiune 2	
<b>sfârșit pentru</b> h) structura <b>pentru</b>	

**Figura 3.3.** Cuvinte cheie pentru metoda pseudocod.

### Observații

- Cuvintele cheie **start** și **stop** corespund începutului și sfârșitului algoritmului. Aceste două cuvinte cheie sunt echivalente blocurilor de start (START) și de sfârșit (STOP), adică unor simboluri grafice specifice reprezentării algoritmilor prin metoda schemelor logice.
- Pentru marea majoritate a simbolurilor grafice specifice metodei de reprezentare a algoritmilor prin metoda schemelor logice există cuvinte cheie echivalente în metoda pseudocod.
- Comanda de atribuire nu conține nici un cuvânt cheie. Această comandă este echivalentă cu blocul de atribuire din metoda schemelor logice.
- Cuvintele cheie **citește** și **scrie** corespund comenzilor de introducere a datelor de intrare (a, b), respectiv de prezentare a datelor de ieșire (u, v) și sunt echivalente blocurilor INPUT, respectiv PRINT din metoda schemelor logice.
- Structura **cât timp** corespunde unui ciclu repetitiv cu test inițial.
- Structura **repetă până când** corespunde unui ciclu repetitiv cu test final.
- Structura **pentru** corespunde unui ciclu repetitiv cu număr cunoscut de pași.
- În cazul structurii condiționale 1, instrucțiunea 1 se execută doar în cazul în care expresia logică este adevărată. În caz contrar, se va executa instrucțiunea 2. În cazul structurii condiționale 2, cele două instrucțiuni 1 și 2 se vor executa doar dacă este îndeplinită expresia logică, în caz contrar, structura **dacă** nu se ia în considerare.
- Structurile condiționale din metoda pseudocod pot fi transpuse în metoda schemelor logice pornind de la unul sau mai multe blocuri condiționale.



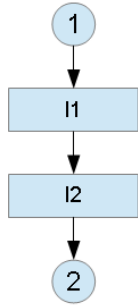
### 3.3. STRUCTURI DE CONTROL ȘI SCHEMELE LOR LOGICE

**Teorema programării structurate (Böhm-Jacopini).** Orice algoritm cu un singur punct de început și un singur punct de sfârșit poate fi descris cu ajutorul doar a trei tipuri de structuri de control, [5, 20]:

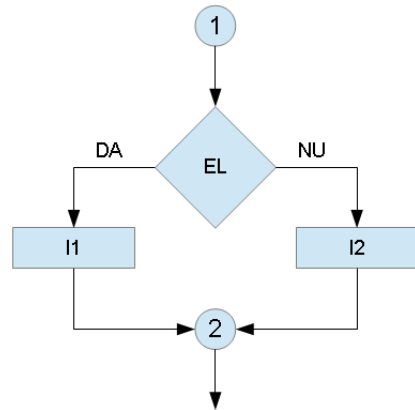
- **structura secvențială.** Instrucțiunile se execută una după alta, exact în ordinea în care apar în program. Schema logică a unei structuri secvențiale, de exemplu, cu doar două instrucțiuni I1 și I2, este prezentată în figura 3.4, a).
- **structura alternativă.** Instrucțiunile se execută în mod alternativ, după cum o anumită expresie logică este adevărată sau falsă. Structurile alternative pot fi: cu două ramuri (figura 3.4, b) sau cu o singură ramură (figura 3.4, c). Valoarea de adevăr a expresiei logice identificată cu numele EL poate fi adevărat sau fals și în funcție de această valoare de adevăr, urmează execuția alternativă a instrucțiunilor.
- **structura repetitivă.** Instrucțiunile se execută în mod repetitiv, de un număr finit de ori. ieșirea din bucla repetitivă se realizează după un număr necunoscut de pași printr-un test inițial (figura 3.4, d) sau printr-un test final (figura 3.4, e), sau după un număr cunoscut de pași, cu ajutorul unui contor (figura 3.4, f).

#### Observații

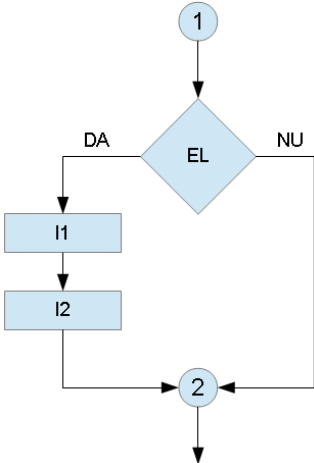
- În cazul structurii secvențiale (figura 3.4, a), cele două instrucțiuni I1 și I2 se execută o singură dată, în ordinea specificată în schema logică: după terminarea instrucțiuni I1 se execută instrucțiunea I2 fără posibilitatea revenirii la instrucțiunea anterioară.
- În cazul structurii alternative cu două ramuri (figura 3.4, b), dacă expresia logică EL este adevărată, se va executa instrucțiunea I1, iar dacă expresia logică EL este falsă se va executa instrucțiunea I2. Cum expresia logică EL nu poate fi decât adevărată sau falsă, rezultă că părăsirea structurii alternative se va face după executarea uneia dintre cele două instrucțiuni, I1 sau I2.
- În cazul structurii alternative cu o singură ramură (figura 3.4, c), dacă expresia logică EL este adevărată, se vor executa în mod secvențial cele două instrucțiuni I1 și I2, în timp ce pentru cazul în care expresia logică EL este falsă, se va părăsi structura alternativă fără a se executa niciuna dintre cele două instrucțiuni.
- În cazul structurii repetitive cu test inițial, dacă expresia logică EL este falsă atunci se părăsește structura repetitivă fără a se executa nici măcar o singură dată instrucțiunea I1. Atât timp, cât expresia logică EL este adevărată, se execută instrucțiunea I1, (figura 3.4, d).



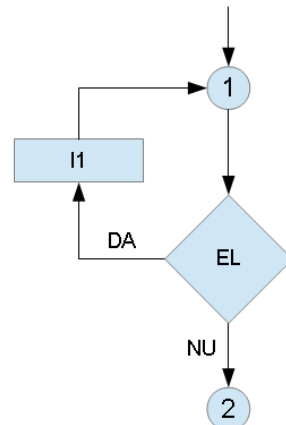
a) structură secvențială



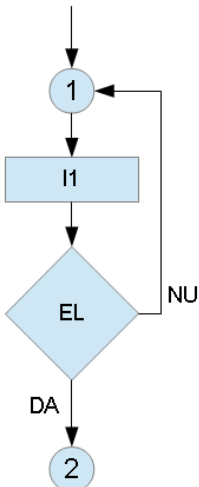
b) structură alternativă cu două ramuri



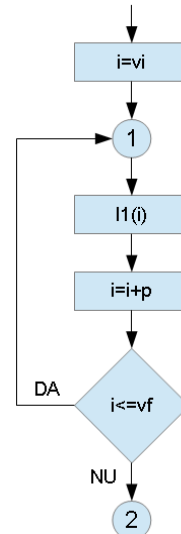
c) structură alternativă cu o ramură



d) structură repetitivă cu test inițial



e) structură repetitivă cu test final



f) structură repetitivă cu contor

**Figura 3.4.** Structuri de control secvențiale, alternative și repetitive.

- În cazul structurii repetitive cu test final (figura 3.4, e), instrucțiunea I1 se va executa cel puțin o singură dată, deoarece abia după executarea instrucțiunii I1 urmează evaluarea expresiei logice EL. Dacă expresia logică EL este adevărată se părăsește structura repetitivă, în caz contrar se va repeta executarea instrucțiunii I1, după care urmează din nou evaluarea expresiei logice EL.
- În cazul structurii repetitive din figura 3.4, f), se definește un contor notat de exemplu cu  $i$ , care pentru început primește valoarea inițială  $i = v_i$  (faza de inițializare a contorului). Apoi se execută instrucțiunea I1 (faza de execuție a corpului structurii repetitive), după care urmează faza de incrementare a valorii contorului cu o valoare caracteristică denumită pas ( $p$ ). În mod implicit pasul incrementării este 1 ( $p=1$ ). Urmează apoi condiția de verificare a valori curente a contorului în funcție de valoarea finală impusă  $v_f$ . În cazul în care valoarea curentă a contorului depășește valoarea finală impusă ( $i > v_f$ ), atunci se părăsește structura repetitivă. În caz contrar se repetă faza de execuție a corpului structurii repetitive, faza de incrementare a contorului și apoi faza de verificare, etc.
- În cazul structurilor repetitive, datorită erorilor logice de stabilire a condițiilor de verificare a valorii de adevăr a expresiilor logice, este posibilă apariția unor bucle infinite. De exemplu, în cazul structurii repetitive cu test inițial, dacă acest test este întotdeauna adevărat, atunci se intră într-o buclă în care instrucțiunile structurii se execută de un număr infinit de ori.

### Problema 3.1

Se consideră mișcarea cu suprafață liberă a apei într-un canal hidraulic cu secțiune transversală dreptunghiulară, având următoarele caracteristici geometrice: lățimea  $b=3$  m și adâncimea normală  $h_0=0,75$  m.

Să se calculeze raza hidraulică a canalului hidraulic cunoscând:

- aria secțiunii vii a apei se calculează cu relația:

$$A = b \cdot h_0$$

- perimetrul udat se calculează cu relația:

$$P = b + 2 \cdot h_0$$

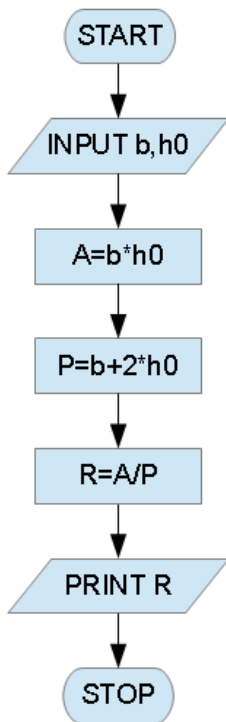
- raza hidraulică se calculează cu relația:

$$R = A/P$$

Să se realizeze schema logică pentru rezolvarea problemei. Să se scrie un fișier de tip `script` pentru implementarea schemei logice.

### Rezolvare

În figura 3.5 se prezintă rezolvarea acestei probleme.



```

1 %% CALCULUL RAZEI HIDRAULICE
2 clc;clear all;close all;
3 %% DATE INITIALE
4 % Latimea canalului [m]
5 b=3;
6 % Adancimea normala [m]
7 h0=0.75;
8 %% BLOC DE CALCULE
9 % Aria sectiunii vii [m2]
10 A=b*h0;
11 % Perimetrul udat [m]
12 P=b+2*h0;
13 % Raza hidraulica [m]
14 R=A/P
  
```

b) fișier de tip script

```

R =
    0.5000
  
```

c) rezultatul executării fișierului de tip script

a) schema logică

**Figura 3.5.** Rezolvarea problemei 3.1.

### Observații

- Pentru a calcula raza hidraulică  $R$  a canalului trebuie mai întâi să se determine aria secțiunii vii  $A$  și perimetrul udat  $P$ . Această problemă conduce la o schemă logică de tip secvențial, figura 3.5, a).
- După blocul care definește punctul de început al schemei logice (START), urmează introducerea datelor de intrare. Datele de intrare ale problemei sunt cele două caracteristici geometrice ale canalului hidraulic: lățimea  $b$  [m] și adâncimea normală  $h_0$  [m]. Introducerea datelor de intrare se realizează cu ajutorul unui bloc pentru citirea variabilelor (INPUT).
- Urmează apoi trei blocuri de atribuire pentru calculul celor trei mărimile  $A$ ,  $P$  și  $R$ .
- Datele obținute în urma executării algoritmului sunt: aria secțiunii vii  $A$  și perimetrul udat  $P$ , care reprezintă rezultatele intermediare și raza hidraulică  $R$ , care reprezintă rezultatul final și care constituie deci, data de ieșire a algoritmului (blocul PRINT).

### 3.4. INSTRUCȚIUNI DE CONTROL MATLAB

Principalele instrucțiuni ale limbajului de programare MATLAB pentru realizarea structurilor de control sunt: instrucțiunea condițională `if`, instrucțiunea repetitivă `for`, instrucțiunea repetitivă `while`, [7]. Cu ajutorul acestor trei instrucțiuni se pot realiza toate structurile de control secvențiale, alternative și repetitive, cu excepția structurii repetitive cu test final.

#### 3.4.1. Instrucțiunea condițională `if`

Instrucțiunea condițională `if` permite realizarea unei structuri alternative pe baza valorii de adevăr a unei expresii logice, [8].

Definirea expresiilor logice se face cu ajutorul operatorilor relaționali și logici specifici, care, în sintaxa limbajului de programare MATLAB, au reprezentările prezentate în tabelul 3.1.

**Tabel 3.1.** Operatori relaționali și logici.

Operatori relaționali	
Semnificația	Simbol MATLAB
Mai mic	<
Mai mare	>
Mai mic sau egal	<=
Mai mare sau egal	>=
Identic	==
Diferit	~=

Operatori logici	
Semnificația	Simbol MATLAB
AND	&
OR	
NOT	~

În limbajul de programare MATLAB, principalele implementări ale instrucțiunii condiționale `if` sunt: `if`; `if-else`; `if-elseif` și `if-elseif-else` și sunt prezentate în figura 3.6.

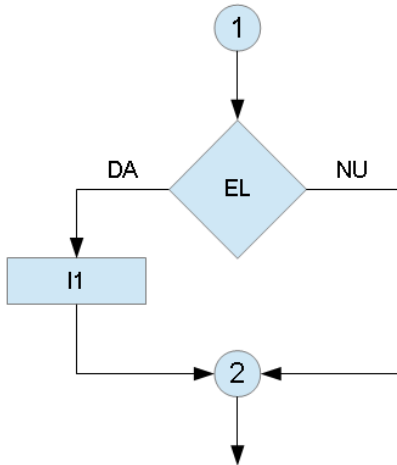
#### Observații

- În cazul variantei `if` (figura 3.6, a), dacă expresia logică `EL` este adevărată, atunci se execută instrucțiunea `I1`, în caz contrar se părăsește structura condițională. Este posibil astfel, ca evaluarea structurii condiționale să nu conducă la executarea nici unei instrucțiuni (cazul expresiei logice false).
- În cazul variantei `if-else` (figura 3.6, b), dacă expresia logică `EL` este adevărată, se execută instrucțiunea `I1`, în caz contrar se execută instrucțiunea `I2`. În acest caz, indiferent de valoarea de adevăr a expresiei logice `EL`, evaluarea structurii condiționale `if-else` conduce la executarea cel puțin a unei instrucțiuni (`I1`, dacă expresia logică `EL` este adevărată, sau `I2`, dacă expresia logică `EL` este falsă).

```

if expresie logică EL
    instrucțiune I1
end

```

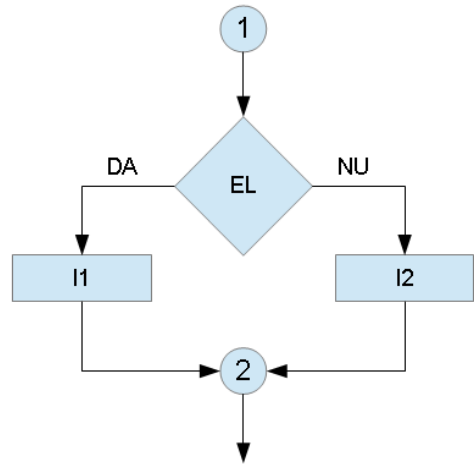


a) varianta if

```

if expresie logică EL
    instrucțiune I1
else
    instrucțiune I2
end

```

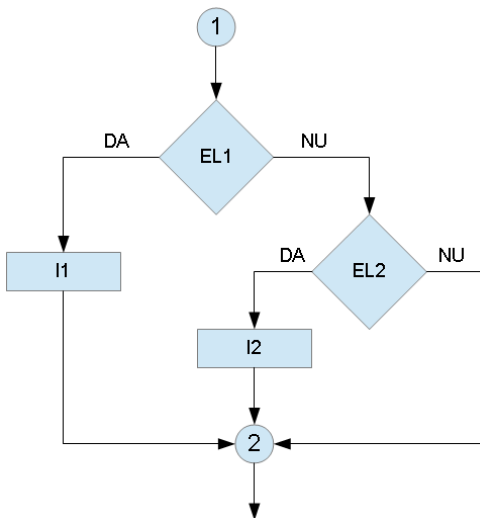


b) varianta if-else

```

if expresie logica EL1
    instrucțiune I1
elseif expresie logică EL2
    instrucțiune I2
end

```

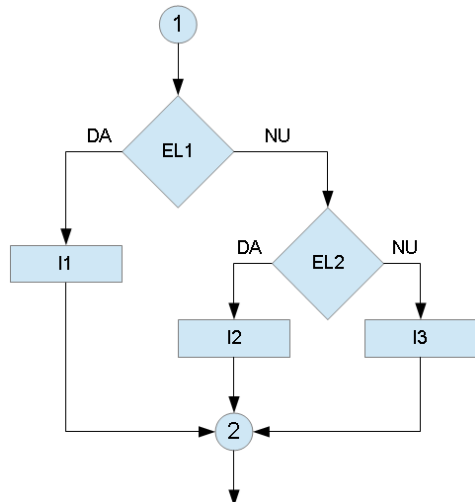


c) varianta if-elseif

```

if expresie logica EL1
    instrucțiune I1
elseif expresie logică EL2
    instrucțiune I2
else
    instrucțiune I3
end

```



d) varianta if-elseif-else

**Figura 3.6.** Instrucțiunea condițională if.

- În cazul variantei `if-elseif` (figura 3.6, c), dacă expresia logică EL1 este adevărată se execută instrucțiunea I1. Dacă expresia logică EL1 este falsă, atunci se testează suplimentar expresia logică EL2: dacă expresia logică EL2 este adevărată se execută instrucțiunea I2, în caz contrar se părăsește structura condițională. Dacă ambele expresii logice EL1 și EL2 sunt false atunci se părăsește structura condițională fără a se executa nici o instrucțiune.
- În cazul variantei `if-elseif-else` (figura 3.6, d), dacă expresia logică EL1 este adevărată se execută instrucțiunea I1. Dacă expresia logică EL1 este falsă, atunci se testează suplimentar expresia logică EL2: dacă expresia logică EL2 este adevărată se execută instrucțiunea I2, în caz contrar se execută instrucțiunea I3. Indiferent de valoarea de adevăr a celor două expresii logice EL1 și EL2, cel puțin o instrucțiune din cadrul structurii condiționale va fi executată.

### Problema 3.2

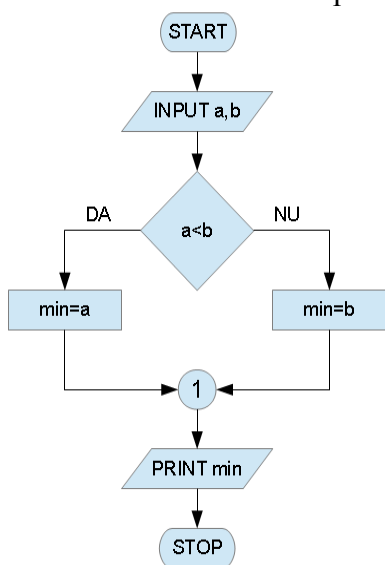
Se consideră două numere reale diferite  $a \in \mathbb{R}$ ,  $b \in \mathbb{R}$ ,  $a \neq b$ . Să se realizeze schema logică pentru descrierea algoritmului de determinare a valorii minime  $\min(a,b)$  a celor două numere.

Să se scrie un fișier de tip `function` pentru implementarea schemei logice.

Să se verifice algoritmul pentru valorile numerice  $a=5$  și  $b=3,24$ .

### Rezolvare

Rezolvarea acestei probleme este prezentată în figura 3.7.



```

function min=fmin2(a,b)
% Minimul a doi scalari
if a<b
    min=a;
else
    min=b;
end
  
```

b) fișier de tip `function`

```

>> a=5;b=3.24;
>> min=fmin2(a,b)
min =
    3.2400
>>
  
```

a) schema logică

c) apelarea funcției `fmin2` cu parametrii doriți

**Figura 3.7.** Rezolvarea problemei 3.2.

## Observații

- După blocul care definește punctul de început al schemei logice (blocul START) urmează introducerea datelor de intrare, figura 3.7, a). Datele de intrare ale problemei sunt cele două variabile reale  $a$  și  $b$ , pentru introducerea cărora se utilizează blocul pentru citirea variabilelor (blocul INPUT).
- Urmează apoi o structură alternativă cu două ramuri. Expresia logică urmărește determinarea valorii de adevăr a inegalității  $a < b$ . Dacă expresia logică este adevărată atunci minimul celor două numere este  $\min(a,b)=a$ , în caz contrar  $\min(a,b)=b$ .
- Pe cele două ramuri ale structurii condiționale se găsește câte o comandă de atribuire prin care se transferă variabilei  $\min$  valoarea numărului  $a$ , sau a numărului  $b$ , după cum expresia logică este sau nu adevărată.
- După părăsirea structurii condiționale se prezintă rezultatul final al algoritmului, respectiv valoarea minimă dintre cele două numere folosind blocul PRINT.
- Structura fișierului de tip `function` este prezentată în figura 3.7, b). Fișierul de tip `function` se salvează în directorul de lucru cu numele `fmin2.m`, exact cu același nume ca și numele funcției.
- Fișierul `function` astfel definit nu se execută, ci se apelează dintr-un alt program, sau direct din fereastra de comenzi. În general, la apelarea unei funcții (indiferent de metoda de definire, fișier de tip `function`, funcție `inline` sau `anonymous`) trebuie să se respecte atât concordanța dintre numărul parametrilor de apelare ai funcției și numărul parametrilor de intrare ai funcției, precum și ordinea parametrilor funcției. În acest caz, funcția are doi parametri de intrare  $a$  și  $b$ . La apelarea funcției, în locul variabilelor generice  $a$  și  $b$ , se transferă în corpul funcției valorile numerice care definesc datele de intrare ale problemei de rezolvat, respectiv 5 și 3,24. Pentru această problemă nu are importanță ordinea în care sunt transferate valorile numerice la apelarea funcției.
- Apelarea fișierului de tip `function` s-a făcut direct din fereastra de comenzi (figura 3.7, c), imediat după introducerea valorilor numerice ale celor două variabile  $a=5$  și  $b=3,24$ .
- Datorită particularităților computaționale ale limbajului de programare MATLAB, fișierul de tip `function` astfel definit poate funcționa și în cazul comparării a două variabile vectoriale sau matriceale, și nu doar pentru variabile de tip scalar. Trebuie asigurată însă condiția egalității dimensionale ale celor două variabile supuse procesului de comparație.



### 3.4.2. Instrucțiunea repetitivă for

Instrucțiunea `for` se utilizează pentru repetarea unor instrucțiuni de un număr  $n_i$  finit, cunoscut de ori, [9]. În acest scop se definește o variabilă specială, denumită contor și notată de exemplu cu litera  $i$ , care va lua anumite valori între valoarea minimă  $i_{min}$  și valoarea maximă  $i_{max}$ . În cazul în care pasul contorului (diferența dintre oricare două valori consecutive) este cunoscut și notat cu  $p_i$ , atunci numărul  $n_i$  de ori pentru care se va realiza repetarea executării instrucțiunilor se poate determina cu relația:

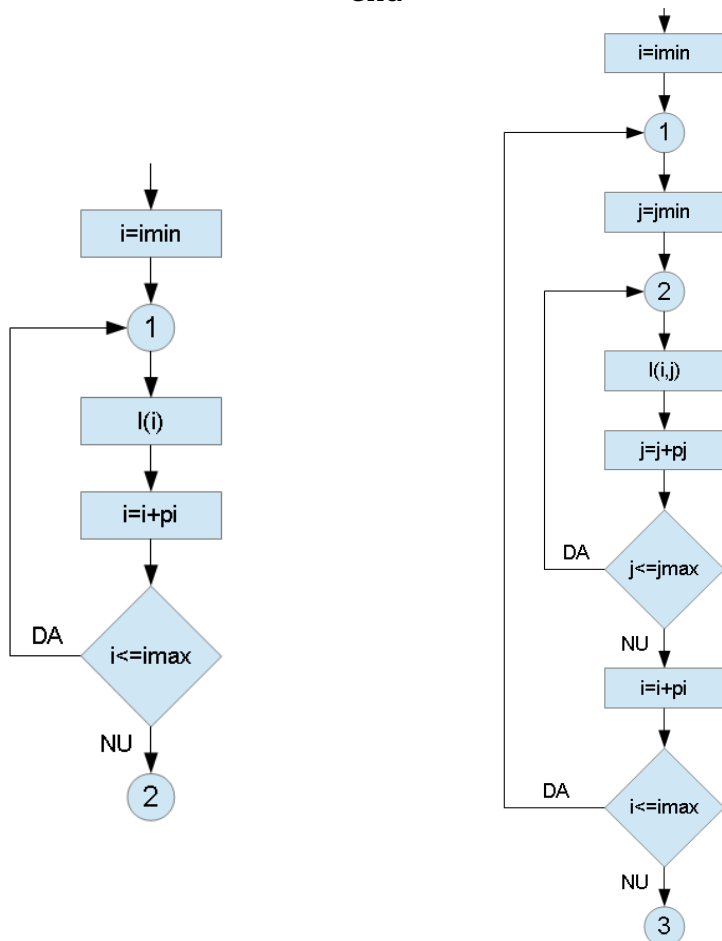
$$n_i = (i_{max} - i_{min})/p_i + 1$$

În figura 3.8 se prezintă două din implementările instrucțiunii `for`.

```

for i=i_min:p_i:i_max      for i=i_min:p_i:i_max
    instrucțiuni(i)      for j=j_min:p_j:j_max
end                          instrucțiuni(i,j)
                             end
                             end
end

```



a) instrucțiune `for` simplă

b) instrucțiune `for` dublă

**Figura 3.8.** Instrucțiunea repetitivă `for`.

## Observații

- În cazul instrucțiunii `for` simple (figura 3.8, a), după execuția structurii repetitive se obține un vector având  $n_i$  elemente. Principalele faze ale algoritmului sunt: faza de inițializare a contorului  $i = i_{min}$ , faza de execuție a instrucțiunilor dependente de valoarea contorului  $i$ , faza de incrementare a contorului  $i$  cu valoarea pasului  $p_i$  și, în sfârșit, faza de testare a valorii curente a contorului. În cazul în care valoarea curentă a contorului este mai mică decât valoarea sa maximă  $i_{max}$  se revine la faza de execuție a instrucțiunilor din corpul structurii, se incrementează din nou contorul și se testează noua valoare a contorului în raport cu valoarea sa maximă. Acest proces repetitiv se desfășoară până când, în urma fazei de testare, se constată că valoarea curentă a contorului este mai mare decât valoarea sa maximă  $i_{max}$ , caz în care se părăsește structura repetitivă.
- În cazul instrucțiunii `for` duble (figura 3.8, b), după execuția structurii repetitive se obține o matrice cu  $n_i$  linii și  $n_j$  coloane. Principalele faze ale algoritmului sunt: inițializarea contorului  $i = i_{min}$ , inițializarea contorului  $j = j_{min}$ , execuția instrucțiunilor dependente de valorile contoarelor  $i$  și  $j$  (este posibil să existe și instrucțiuni dependente doar de contorul  $i$ , sau doar de contorul  $j$ ), incrementarea contorului  $j$  cu valoarea pasului  $p_j$ , testarea valorii curente a contorului  $j$ . Dacă în urma testării contorului  $j$ , se constată că valoarea sa curentă este mai mică decât valoarea sa maximă  $j_{max}$ , se revine în corpul structurii repetitive și se execută din nou toate instrucțiunile, de data aceasta pentru valoarea incrementată a contorului  $j$  (contorul  $i$  nu a fost incrementat încă). După repetarea instrucțiunilor se incrementează din nou contorul  $j$  și se verifică valoarea sa curentă în raport cu valoarea sa maximă. Acest proces repetitiv se desfășoară până când se depășește valoarea maximă a contorului  $j$ , moment în care se părăsește instrucțiunea `for` interioară și se trece, în sfârșit, și la incrementarea contorului  $i$  (cu valoarea pasului  $p_i$ ). Urmează apoi testarea valorii curente a contorului  $i$  față de valoarea sa maximă  $i_{max}$ . Dacă se constată că valoarea curentă a contorului  $i$  este mai mică decât valoarea sa maximă  $i_{max}$ , se revine în corpul structurii la faza de inițializare a contorului  $j$  a cărui valoare va reveni la valoarea sa minimă  $j = j_{min}$ . Urmează apoi repetarea executării instrucțiunilor până când valoarea curentă a contorului  $j$  depășește valoarea sa maximă. Apoi se incrementează din nou contorul  $i$ , ș.a.m.d. În acest mod, instrucțiunile din corpul structurii se execută de  $n_i \times n_j$  ori.

### Problema 3.3

Se consideră doi vectori  $a$  și  $b$  având elementele:

$$a = [a_1 \ a_2 \ \dots \ a_n]$$

$$b = [b_1 \ b_2 \ \dots \ b_n]$$

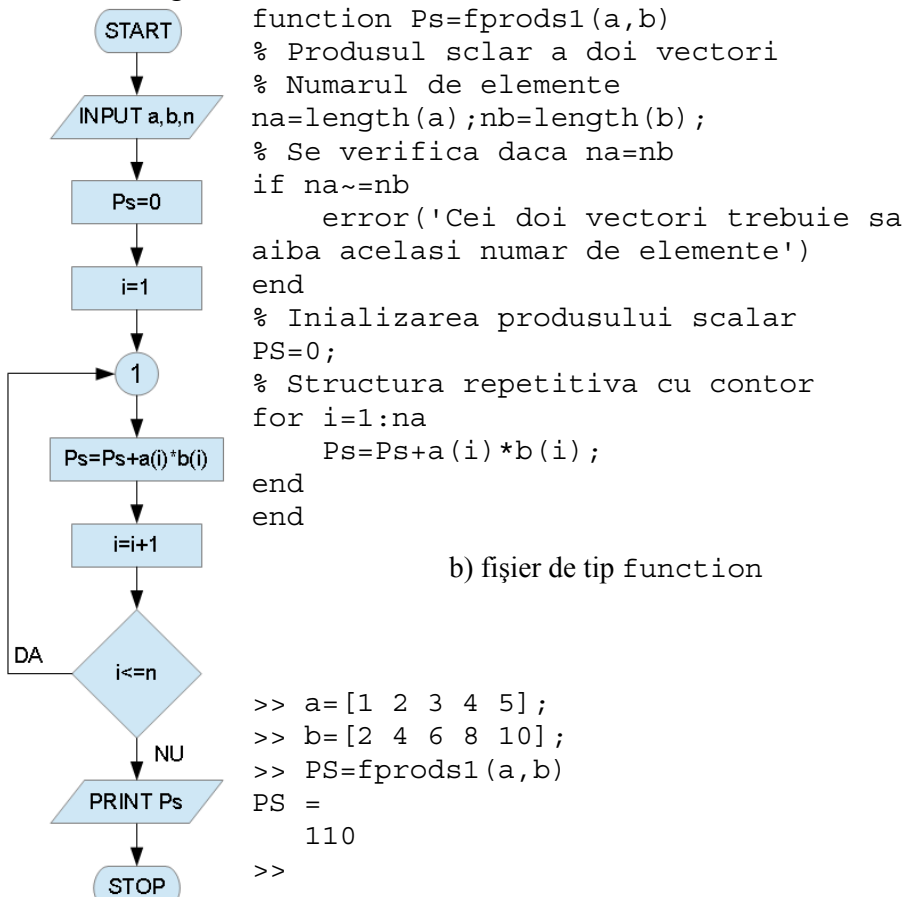
Produsul scalar al celor doi vectori se calculează cu relația:

$$P = \sum_{i=1}^n a_i b_i$$

Să se realizeze o schemă logică pentru descrierea algoritmului de determinare a produsul scalar al celor doi vectori folosind o structură repetitivă cu contor. Să se scrie un fișier de tip function pentru implementarea schemei logice. Să se verifice algoritmul pentru vectorii  $a=[1 \ 2 \ 3 \ 4 \ 5]$  și  $b=[2 \ 4 \ 6 \ 8 \ 10]$ .

### Rezolvare

Rezolvarea problemei folosind o structură repetitivă cu contor este prezentată în figura 3.9.



a) schema logică

c) apelarea funcției fprods1 cu parametrii doriți

**Figura 3.9.** Rezolvarea problemei 3.3, structură repetitivă cu contor.

## Observații

- Schemă logică reprezentată în figura 3.9, a) corespunde algoritmului bazat pe o structură repetitivă cu contor. Schema logică cuprinde:
  - Faza de introducere a datelor de intrare, în care se citesc cei doi vectori  $a$  și  $b$ , precum și dimensiunea acestora  $n$  cu ajutorul unui bloc de tip INPUT.
  - Faza de inițializare, în care se definește valoarea inițială a contorului  $i = 1$ , precum și valoarea inițială a produsului scalar  $Ps = 0$ , cu ajutorul a două blocuri de atribuire.
  - Faza de calcul care constă într-o structură de tip repetitiv cu contor. La fiecare iterație se recalculează produsul scalar adunând la valoarea anterioară un nou termen  $a_i b_i$  după care se incrementează contorul  $i = i + 1$  și apoi se testează dacă valoarea curentă a contorului a depășit sau nu valoarea corespunzătoare numărului de elemente ale celor doi vectori. Calculul repetitiv continuă până la parcurgerea celor  $n$  iterații, după care se părăsește structura repetitivă.
  - Faza de prezentare a datelor de ieșire, care în acest caz sunt reprezentate de valoarea produsului scalar, se realizează cu ajutorul unui bloc de tip PRINT.
- Pentru implementarea schemei logice din figura 3.9, a), a fost creată o funcție prin metoda fișierelor de tip `function`. Structura fișierului de tip `function` este prezentată în figura 3.9, b). Fișierul a fost salvat cu numele `fprods1` în directorul curent.
- Funcția are doi parametri de intrare  $a$  și  $b$ . Dacă  $n_a$  și  $n_b$  reprezintă dimensiunile celor doi vectori, atunci funcția poate furniza un rezultat valabil doar dacă este îndeplinită condiția  $n_a = n_b$ . Dacă de exemplu  $n_a > n_b$ , atunci, la iterația  $n_b + 1$  se constată că elementul  $a_{n_b + 1}$  nu are un element corespondent în cadrul vectorului  $b$ , astfel încât produsul  $a_{n_b + 1} \cdot b_{n_b + 1}$  nu se poate calcula, rezultând blocarea execuției programului și avertizarea utilizatorului în fereastra de comenzi a programului printr-un mesaj de eroare de tipul (pentru cazul  $n_a = 6$  și  $n_b = 5$ ):

```
Attempted to access b(6); index out
of bounds because numel(b)=5.
Error in fprods1 (line 12)
    Ps=Ps+a(i)*b(i);
```

Blocarea execuției programului și avertizarea utilizatorului asupra existenței acestei erori înainte de intrarea în structura repetitivă cu contor se poate face dacă în structura funcției se prevede un bloc specializat în acest scop care să detecteze cazul  $n_a \neq n_b$ :

```

% Se verifica daca na=nb
if na~=nb
    error('Cei doi vectori trebuie sa
aiba acelasi numar de elemente')
end

```

S-a utilizat o structură condițională `if` simplă care verifică valoarea de adevăr a expresiei logice  $n_a \neq n_b$ . Dacă valoarea de adevăr este adevărat, atunci se execută instrucțiunea `error`, [10] din corpul structurii condiționale. Această instrucțiune blochează execuția programului și afișează în fereastra de comenzi Command Window mesajul de eroare introdus ca argument al funcției `error`:

```

>> PS=fprods1(a,b)
Error using fprods1 (line 7)
Cei doi vectori trebuie sa aiba
acelasi numar de elemente
>>

```

Pentru avertizarea utilizatorului asupra apariției unor erori la execuția instrucțiunilor dintr-un program se poate utiliza și instrucțiunea `warning`, [11]:

```

% Se verifica daca na=nb
if na~=nb
    warning('Cei doi vectori trebuie sa
aiba acelasi numar de elemente')
end

```

Deosebirea față de instrucțiunea `error` constă în faptul că la detectarea erorii nu se întrerupe execuția programului, ci doar se afișează mesajul de avertizare. Se intră apoi în structura repetitivă și doar la iterația  $n_b+1$  se ajunge la situația imposibilității efectuării produsului  $a \cdot b$ , moment în care se blochează execuția programului:

```

PS=fprods1(a,b)
Warning: Cei doi vectori trebuie sa
aiba acelasi numar de elemente
> In fprods1 at 7
Attempted to access b(6); index out
of bounds because numel(b)=5.
Error in fprods1 (line 13)
    Ps=Ps+a(i)*b(i);
>>

```

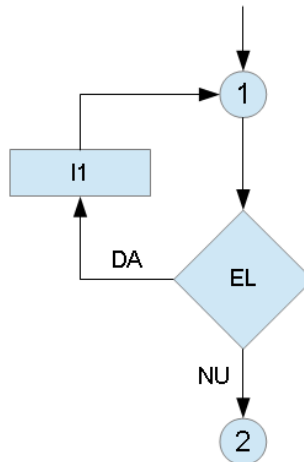
- Apelarea fișierului de tip `function` s-a făcut direct din fereastra de comenzi (figura 3.9, c), imediat după introducerea valorilor numerice ale celor două variabile vectoriale  $a=[1\ 2\ 3\ 4\ 5]$  și  $b=[2\ 4\ 6\ 8\ 10]$ . Rezultatul obținut în urma efectuării calculelor este:  $PS=110$ .

### 3.4.3. Instrucțiunea repetitivă while

Instrucțiunea repetitivă while se utilizează pentru realizarea unei structuri repetitive cu test inițial, [12].

Implementarea instrucțiunii while este prezentată în figura 3.10.

```
while EL
    instrucțiune 1
end
```



**Figura 3.10.** Instrucțiunea repetitivă while.

#### Observații

- Instrucțiunea repetitivă while permite executarea instrucțiunii I1 atât timp cât valoarea de adevăr a expresiei logice EL este adevărat. Numărul de ori de reperare a execuției instrucțiunii din corpul structurii nu este cunoscut la începutul executării structurii repetitive. Atunci când valoarea de adevăr a expresiei logice EL este fals, se părăsește corpul structurii repetitive, moment în care se poate determina de câte ori s-a repetat executarea instrucțiunii I1.
- În cazul în care chiar la intrarea în corpul structurii repetitive, valoarea de adevăr a expresiei logice EL este fals, atunci se părăsește structura repetitivă fără a se executa nici măcar o singură dată instrucțiunea I1.
- În cazul în care, la intrarea în corpul structurii valoarea de adevăr a expresiei logice este adevărat și în urma repetării de oricâte ori a instrucțiunii I1 din corpul structurii, valoarea de adevăr a expresiei logice rămâne neschimbată, se intră într-o buclă infinită care conduce la repetarea la infinit a executării instrucțiunii din corpul structurii repetitive. Ieșirea forțată dintr-o buclă infinită se realizează prin combinația de taste CTRL+C.

### Problema 3.4

Se consideră doi vectori  $a$  și  $b$  având elementele:

$$a = [a_1 \ a_2 \ \dots \ a_n]$$

$$b = [b_1 \ b_2 \ \dots \ b_n]$$

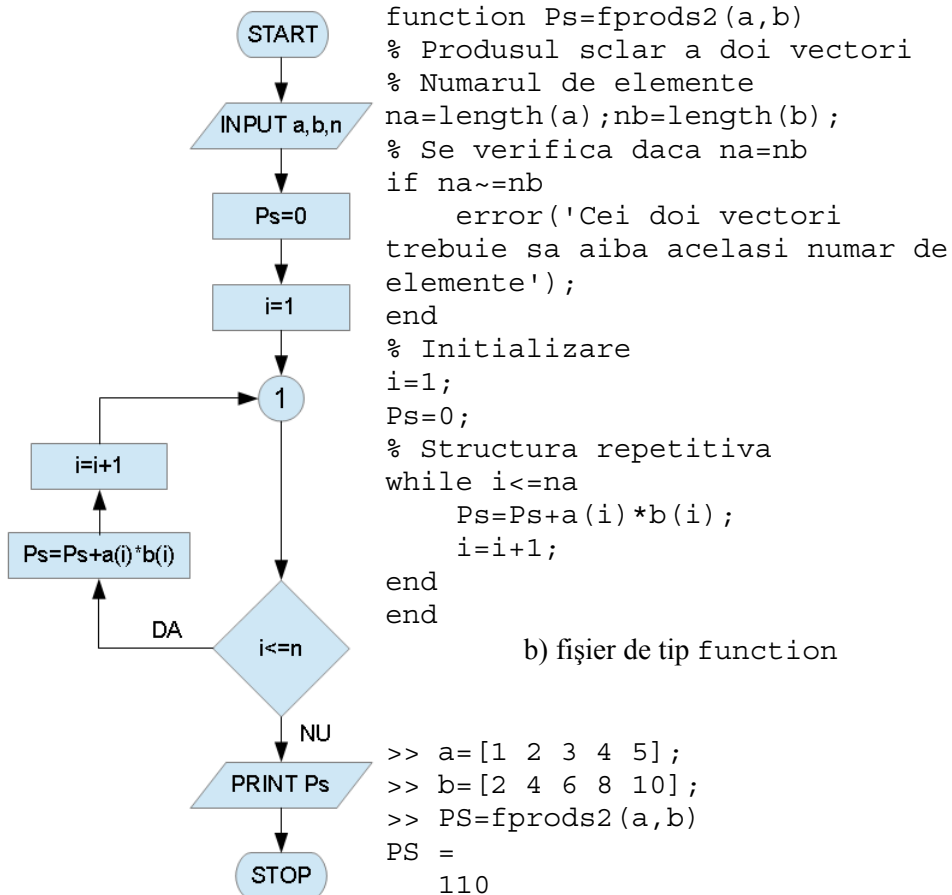
Produsul scalar al celor doi vectori se calculează cu relația:

$$P = \sum_{i=1}^n a_i b_i$$

Să se realizeze o schemă logică pentru descrierea algoritmului de determinare a produsul scalar al celor doi vectori folosind o structură repetitivă cu test inițial. Să se scrie un fișier de tip function pentru implementarea schemei logice. Să se verifice algoritmul pentru vectorii  $a=[1 \ 2 \ 3 \ 4 \ 5]$  și  $b=[2 \ 4 \ 6 \ 8 \ 10]$ .

### Rezolvare

Rezolvarea problemei folosind o structură repetitivă cu test inițial este prezentată în figura 3.11.



a) schema logică

c) apelarea funcției fprods2 cu parametrii doriți

**Figura 3.11.** Rezolvarea problemei 3.4, structură repetitivă cu test inițial.

## Observații

- Schemă logică reprezentată în figura 3.11, a) corespunde algoritmului bazat pe o structură repetitivă cu test inițial și cuprinde:
  - Faza de introducere a datelor de intrare, în care se citesc cei doi vectori  $a$  și  $b$ , precum și dimensiunea acestora  $n$ , cu ajutorul unui bloc de tip INPUT.
  - Faza de inițializare, în care se definește valoarea inițială a contorului  $i = 1$ , precum și valoarea inițială a produsului scalar  $Ps = 0$  cu ajutorul a două blocuri de atribuire.
  - Faza de calcul, care constă într-o structură de tip repetitiv cu test inițial. Se testează valoarea curentă  $i$  a contorului și dacă se constată că  $i \leq n$ , atunci se recalculează produsul scalar adăugând la valoarea anterioară, rezultatul curent al produsului elementelor,  $a_i \cdot b_i$ . Apoi se incrementează contorul  $i = i + 1$  după care se testează din nou valoarea curentă a contorului. Dacă valoarea curentă a contorului a depășit valoarea corespunzătoare numărului de elemente ale celor doi vectori, calculul repetitiv se sfârșește, părăsindu-se structura repetitivă.
  - Faza de prezentare a datelor de ieșire (în acest caz sunt reprezentate de valoarea produsului scalar), care se realizează cu ajutorul unui bloc de tip PRINT.
- Pentru implementarea schemei logice din figura 3.11, a), a fost creată o funcție prin metoda fișierelor de tip `function`. Structura fișierului de tip `function` este prezentată în figura 3.11, b). Fișierul a fost salvat cu numele `fprods2` în directorul curent.
- Datorită unor erori de introducere a datelor sau datorită unor calcule anterioare, este posibil ca dimensiunile celor doi vectori să nu fie identice ( $n_a \neq n_b$ ). Pentru a detecta această situație, a bloca execuția programului înainte de intrarea în structura repetitivă cu test inițial și pentru a avertiza utilizatorul asupra acestei erori, în structura funcției s-a prevăzut un bloc specializat bazat pe o structură condițională:

```
% Se verifica daca na=nb
if na~=nb
    error('Cei doi vectori trebuie sa
    aiba acelasi numar de elemente')
end
```
- Apelarea fișierului de tip `function` s-a făcut direct din fereastra de comenzi (figura 3.11, c), imediat după introducerea valorilor numerice ale celor două variabile vectoriale  $a=[1\ 2\ 3\ 4\ 5]$  și  $b=[2\ 4\ 6\ 8\ 10]$ . Rezultatul obținut în urma efectuării calculelor este:  $PS=110$ .



### Problema 3.5

Se consideră seria numerică  $\sum_{k=1}^{\infty} a_k$  având termenul general:

$$a_k = 1/k^2$$

- Considerând primii  $n=10$  termeni ai seriei:
  - Să se studieze convergența seriei prin analiza șirului sumelor parțiale  $S_1, S_2, \dots, S_n$ :  
 $S_1 = \sum_{k=1}^1 a_k, S_2 = \sum_{k=1}^2 a_k, \dots, S_n = \sum_{k=1}^n a_k$
  - Să se determine viteza de convergență a șirului sumelor parțiale calculând și analizând diferența  $r_k$  dintre fiecare două sume parțiale consecutive:  
 $r_k = S_k - S_{k-1}, \forall k = 2 \div n$
  - Cunoscând valoarea exactă a sumei seriei:  
$$S_{\infty} = \sum_{k=1}^{\infty} a_k = \lim_{n \rightarrow \infty} S_n = \frac{\pi^2}{6}$$
să se calculeze eroarea relativă de aproximare pentru toate valorile șirului sumelor parțiale determinate anterior:  
$$\varepsilon_k = \frac{|S_k - S_{\infty}|}{S_{\infty}} \cdot 100 [\%], \forall k = 1 \div n$$
- Să se determine numărul minim de termeni  $k_{min}$  ai șirului sumelor parțiale astfel încât eroarea relativă dintre valoarea exactă și valoarea aproximativă a sumei seriei să fie mai mică decât  $\varepsilon_{cr}=2\%$ .

### Rezolvare

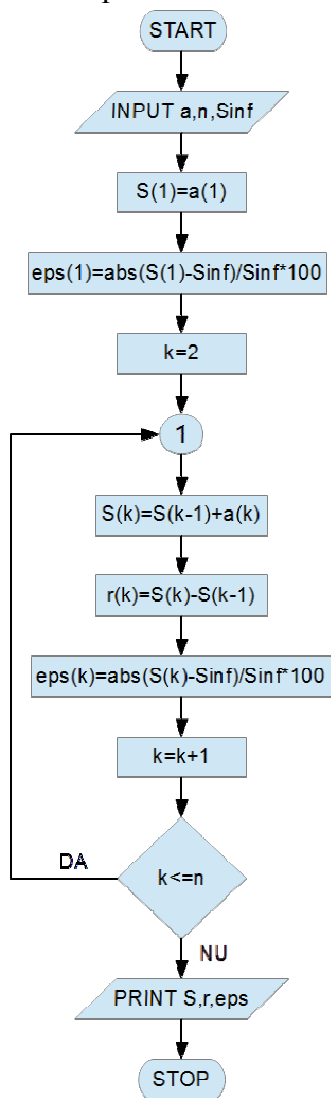
Rezolvarea primului punct al problemei, cuprinzând calculul șirului sumelor parțiale, analiza vitezei de convergență și calculul erorilor relative de aproximare este prezentată în figura 3.12. S-a utilizat o schema logică bazată pe o structură repetitivă cu contor.

### Observații

- După blocul care definește punctul de început al schemei logice (blocul START) urmează introducerea datelor de intrare, figura 3.12, a). Datele de intrare ale problemei sunt: expresia termenului general al seriei  $a_k$ , numărul  $n$  impus de termeni ai șirului sumelor parțiale, precum și valoarea exactă a sumei seriei  $S_{\infty}$ . Pentru introducerea datelor de intrare se utilizează blocul pentru citirea variabilelor (blocul INPUT).
- Faza de inițializare a structurii repetitive cu contor constă în definirea primei valori a șirului sumelor parțiale  $S_1$ , în calculul primei valori a erorii relative  $\varepsilon_1$ , precum și în inițializarea contorului  $k=2$ .
- Urmează apoi structura repetitivă cu contor în care, la fiecare iterație definită prin valoarea curentă  $k$  a contorului, se calculează noua valoare  $S_k$  din șirul sumelor parțiale, diferența față de valoarea

anterioară  $r_k = S_k - S_{k-1}$ , precum și eroarea relativă față de valoarea exactă a sumei seriei  $\varepsilon_k$ .

- Blocul condițional testează valoarea curentă a contorului  $k$  și întrerupe structura repetitivă în cazul depășirii valorii impuse  $n$  a numărului de termeni.
- Faza de prezentare a datelor de ieșire (în acest caz sunt reprezentate de valorile șirului sumelor parțiale  $S_k$ , de valorile diferențelor  $r_k$  și de valorile erorilor relative  $\varepsilon_k$ ) se realizează cu ajutorul unui bloc de tip PRINT



a) schema logică

```

%% ANALIZA SERIILOR NUMERICE (1)
clear all;close all;clc;
%% DATE DE INTRARE
% Numarul de termeni ai sumei
n=10;
% Valoarea exacta a sumei seriei
Sinf=pi^2/6;
% Primul element al sirului
%sumelor partiale
S(1)=1/1^2;
% Prima valoare a erorii relative
e(1)=abs(S(1)-Sinf)/Sinf*100;
%% BLOC DE CALCUL
for k=2:n
    S(k)=S(k-1)+1/k^2;
    r(k)=S(k)-S(k-1);
    e(k)=abs(S(k)-Sinf)/Sinf*100;
end
%% PREZENTAREA REZULTATELOR
disp(' S(k) r(k) e(k) [%] ');
disp([S; r; e]')
  
```

b) fișier de tip script

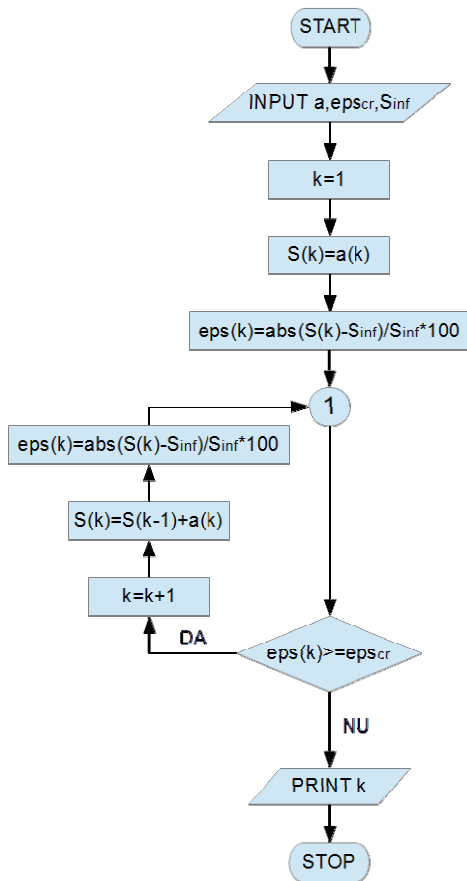
S(k)	r(k)	eps(k) [%]
1.0000	0	39.2073
1.2500	0.2500	24.0091
1.3611	0.1111	17.2544
1.4236	0.0625	13.4548
1.4636	0.0400	11.0231
1.4914	0.0278	9.3344
1.5118	0.0204	8.0938
1.5274	0.0156	7.1439
1.5398	0.0123	6.3933
1.5498	0.0100	5.7854

c) rezultatele executării fișierului script

**Figura 3.12.** Calculul sumelor parțiale, a vitezei de convergență și a erorilor relative de aproximare.

- Pentru prezentarea în fereastra de comenzi Command Window a rezultatelor numerice obținute în urma efectuării calculului s-a utilizat de două ori instrucțiunea `disp`, [13]. Prima instrucțiune `disp` afișează titlurile celor trei coloane cu rezultate numerice. A doua instrucțiune `disp` afișează valorile numerice propriu-zise. Fiecare șir de valori  $S_k$ ,  $r_k$  și  $\varepsilon_k$  reprezintă vectori linie și pentru afișarea pe coloane s-a utilizat operația de transpunere.

Rezolvarea celui de-al doilea punct al problemei, cuprinzând calculul numărului minim de termeni  $k_{min}$  ai șirului sumelor parțiale astfel încât eroarea relativă dintre valoarea exactă și valoarea aproximativă a sumei seriei fie mai mică decât valoarea critică  $\varepsilon_{cr}=2\%$  este prezentată în figura 3.13. S-a utilizat o schema logică bazată pe o structură repetitivă cu test inițial.



```

%% ANALIZA SERIILOR NUMERICE(2)
clear all;close all;clc;
%% DATE DE INTRARE
% Eroarea relativa critica [%]
ecr=2;
% Valoarea exacta a sumei
% seriei:
Sinf=pi^2/6;
%% BLOC DE CALCUL
% Initializarea contorului
k=1;
% Initializarea parametrilor
S(k)=1/k^2;
e(k)=abs(S(k)-Sinf)/Sinf*100;
while e(k) >= ecr
    k=k+1;
    S(k)=S(k-1)+1/k^2;
    e(k)=abs(S(k)-
Sinf)/Sinf*100;
end
%% PREZENTAREA REZULTATELOR
disp(['Eroarea impusa = '
num2str(ecr) '%']);
disp(['Numarul de termeni
necesar = ' num2str(k)]);
  
```

a) schema logică b

c) rezultatele executării fișierului script

**Figura 3.13.** Calculul numărului necesar de termeni astfel încât suma seriei să aproximeze valoarea reală cu o eroare critică impusă.

## Observații

- După blocul care definește punctul de început al schemei logice (blocul START) urmează introducerea datelor de intrare, figura 3.13, a). Datele de intrare ale problemei sunt: expresia termenului general al seriei  $a_k$ , valoarea critică a erorii relative  $\varepsilon_{cr}$  dintre valoarea exactă și valoarea aproximativă a sumei seriei, precum și valoarea exactă a sumei seriei  $S_\infty$ . Pentru introducerea datelor de intrare se utilizează blocul pentru citirea variabilelor (blocul INPUT).
- Faza de inițializare a structurii repetitive cu test inițial constă în inițializarea contorului  $k=1$ , calculul primei valori a șirului sumelor parțiale  $S_k$ , precum și în calculul primei valori a erorii relative  $\varepsilon_k$ .
- Urmează apoi structura repetitivă cu test inițial în care se testează valoarea de adevăr a inegalității  $\varepsilon_k \geq \varepsilon_{cr}$ . Dacă valoarea de adevăr a expresiei logice este adevărat, se efectuează calculele din structura repetitivă: se incrementează contorul  $k = k + 1$ , se calculează noua valoare a sumei seriei prin adăugarea la valoarea anterioară a unui nou termen al seriei  $S_k = S_{k-1} + a_k$ , se determină noua valoare a erorii relative. Apoi se testează din nou valoarea de adevăr a inegalității, însă pentru valoarea curentă a sumei seriei. Calculul iterativ continuă prin contorizarea termenilor seriei, unul după altul până când eroarea relativă calculată scade sub valoarea critică impusă, moment în care se părăsește structura repetitivă.
- Faza de prezentare a datelor de ieșire (în acest caz sunt reprezentate doar de valoarea numărului  $k$  de termeni ai seriei necesari a fi luați în calcul pentru ca eroarea relativă dintre valoarea aproximativă și cea exactă a sumei seriei să fie mai mică decât o valoare critică impusă), se realizează cu ajutorul unui bloc de tip PRINT.
- Pentru prezentarea în fereastra de comenzi Command Window a rezultatelor obținute în urma efectuării calculelor s-a utilizat de două ori instrucțiunea `disp`. Pentru a afișa cu aceeași instrucțiune `disp` atât șiruri de caractere, cât și valori numerice trebuie definit un vector de elemente. Șirurile de caractere trebuie scrise între apostrofuri, în timp ce valorile numerice trebuie transformate în șiruri de caractere echivalente folosind instrucțiunea `num2str`, [14]. De exemplu, instrucțiunea:

```
disp(['Eroarea impusa = ' num2str(egr) '%'])
```

va afișa trei elemente: șirul de caractere Eroarea impusa =, apoi șirul de caractere obținut prin transformarea valori numerice a variabilei `egr` și, în sfârșit, caracterul `%`.

## BIBLIOGRAFIE

1. Andonie R., Algoritmi fundamentali – o perspectivă C++, Ed. Libris, Cluj-Napoca, 1995.
2. DEX Online, (Dicționar explicativ al limbii române online), Algoritm, <http://dexonline.ro/definitie/algoritm>, accesat la data 31.01.2014.
3. Iorga V., Pop F., Metode numerice. Algoritmi și aplicații, Ed. Politehnica Press, București, 2008.
4. Knuth D.E., Arta programării calculatoarelor, vol. 1: Algoritmi fundamentali, Ed. Teora, București, 1999.
5. Lungu V., Petrescu Gh., Structuri de date și algoritmi, Ed. Printech, București, 2007.
6. Maier C., Dima M., Programarea calculatoarelor și limbaje de programare. Teorie și aplicații, Ed. Cartea Universitară, București, 2007.
7. MathWorks, MATLAB, Programming Fundamentals, 2014.
8. MathWorks, Execute Statements if Condition is True, <http://www.mathworks.com/help/matlab/ref/if.html>, accesat la 16.02.2014.
9. MathWorks, Execute Statements Specified Number of Times, <http://www.mathworks.com/help/matlab/ref/for.html>, accesat la 16.02.2014.
10. MathWorks, Display Message and Abort Function, <http://www.mathworks.com/help/matlab/ref/error.html>, accesat la 26.08.2014.
11. MathWorks, Warning Message, <http://www.mathworks.com/help/matlab/ref/warning.html>, accesat la 26.08.2014.
12. MathWorks, Repeatedly Execute Statements while Condition is True, <http://www.mathworks.com/help/matlab/ref/while.html>, accesat la 16.02.2014.
13. MathWorks, Display Text or Array, <http://www.mathworks.com/help/matlab/ref/disp.html>, accesat la 26.08.2014.
14. MathWorks, Convert Number to String, <http://www.mathworks.com/help/matlab/ref/num2str.html>, accesat la 26.08.2014.
15. Pop F., Iorga V., Algoritmi numerici, Ed. BREN, București, 2006.
16. Runceanu A., Programarea și utilizarea calculatoarelor, Editura Academică Brâncuși, Târgu-Jiu, 2003.
17. Wikipedia, Algorithm, <http://en.wikipedia.org/wiki/Algorithm>, accesat la 26.08.2014.
18. Wikipedia, Flowchart, <http://en.wikipedia.org/wiki/Flowchart>, accesat la 26.08.2014.
19. Wikipedia, Pseudocode, <http://en.wikipedia.org/wiki/Pseudocode>, accesat la 26.08.2014.
20. Wikipedia, Structured Program Theorem, [http://en.wikipedia.org/wiki/Structured\\_program\\_theorem](http://en.wikipedia.org/wiki/Structured_program_theorem), accesat la 26.08.2014.
21. Zaharia C., Zaharia M., Să învățăm să programăm, Editura Tehnică, București, 1992.

# CAPITOLUL 4

## VARIABLE VECTORIALE

### 4.1. DEFINIREA VARIABILELOR VECTORIALE

În mod implicit, programul operează cu variabile de tip matrice, astfel că la declararea unei variabile vectoriale  $x$ , de fapt programul construiește o matrice având dimensiunea  $1 \times n_x$ , deci o variabilă vectorială de tip linie, cu o linie și  $n_x$  coloane. Prin variabilă vectorială (vector), în cele ce urmează, se înțelege o variabilă formată din mai multe valori distribuite după o linie sau coloană, fără a considera noțiunile de direcție și sens. Declararea variabilelor vectoriale se realizează prin operația de atribuire = prin care unei variabile vectoriale oarecare identificată prin numele NumeVariabilă, programul îi va atribui o anumită expresie, [2]:

NumeVariabilă=expresie

astfel încât variabila vectorială de tip linie obținută va conține elementele:

$$x = [x_1 \quad x_2 \quad x_3 \quad \dots \quad x_{n_x}]$$

în care indicele primului element este întotdeauna 1.

Dacă variabila vectorială are elementele ordonate crescător, adică:

$$x_1 < x_2 < x_3 < \dots < x_{n_x}$$

atunci:  $x_{\min} = x_1$  este cea mai mică valoare iar  $x_{\max} = x_{n_x}$  este cea mai mare valoare a variabilei vectoriale.

În funcție de valorile numerice efective conținute în variabila vectorială, se pune problema punerii în evidență a unei reguli prin care se face trecerea de la un element al variabilei vectoriale la elementul următor. În cazul în care există o astfel de regulă, atunci se spune că vectorii au **elemente asociate**, fie cu pas constant (liniar sau logaritm), fie cu pas variabil. În cazul în care nu există o astfel de regulă, atunci se spune că vectorii au **elemente neasociate**. Pentru fiecare tip de vector (cu elemente asociate sau cu elemente neasociate) există metode de definire particulare.

#### 4.1.1. Variabile vectoriale cu elemente neasociate

Definirea variabilelor vectoriale cu elemente neasociate se realizează prin metoda “element cu element”. Această metodă constă în scrierea tuturor elementelor variabilei, unul după altul, separate prin spațiu sau virgulă, între paranteze pătrate, caz în care se obține un vector linie.

Dacă se dorește obținerea unui vector coloană atunci fie se transpune vectorul linie, fie se utilizează separatorul ; între elementele variabilei.

### Problema 4.1

Să se definească următorii vectori linie cu elemente neasociate:

$$a) x = [1 \ 56 \ 2 \ 4 \ -34 \ 0]$$

$$b) y = [2 \ 5 \ 1 \ 12 \ 4 \ 15]$$

$$c) u = [\sqrt{7} \ e^2 \ \sqrt{1+\sqrt{2}}]$$

$$d) v = [\ln 7 \ \log_{10}(5/2) \ \sqrt{\sin(1/\pi)}]$$

### Rezolvare

```
>> x=[1 56 2 4 -34 0]
x =
     1     56     2     4    -34     0
>> y=[2 5 1 12 4 15]
y =
     2     5     1    12     4    15
>> u=[sqrt(7) exp(2) sqrt(1+sqrt(2))]
u =
     2.6458     7.3891     1.5538
>> v=[log(7) log10(5/2) sqrt(sin(1/pi))]
v =
     1.9459     0.3979     0.5594
```

#### 4.1.2. Variabile vectoriale cu elemente asociate

Definirea vectorilor cu elemente asociate se face în mod diferit după cum pasul este liniar sau logaritmic. Definirea vectorilor cu elemente asociate cu pas liniar se poate face prin două metode, funcție de parametrii cunoscuți ai vectorului: metoda pasului și metoda numărului de elemente.

**Metoda pasului.** Dacă se cunosc limitele vectorului crescător  $x_{\min}$  și  $x_{\max}$ , precum și pasul  $p_x$  dintre oricare două elemente ale vectorului, atunci pentru definirea vectorilor crescători se va utiliza instrucțiunea, [4]:

```
x=xmin:px:xmax
```

obținându-se următoarele valori:

$$[x_{\min} \ x_{\min} + 1 \cdot p_x \ x_{\min} + 2 \cdot p_x \ \dots \ x_{\max}]$$

Numărul de elemente ale vectorului astfel obținut este:

$$n_x = (x_{\max} - x_{\min})/p_x + 1$$

În cazul pasului unitar ( $p_x=1$ ), se poate utiliza instrucțiunea simplificată:

```
x=xmin:xmax
```

obținându-se următoarele valori:

$$[x_{\min} \quad x_{\min} + 1 \quad x_{\min} + 2 \quad \dots \quad x_{\max}]$$

Generarea vectorilor descrescători cu pas constant liniar se realizează în același mod, printr-o instrucțiune de genul:

$$x=x_{\max}:-p_x:x_{\min}$$

obținându-se următoarele valori:

$$[x_{\max} \quad x_{\max} - 1 \cdot p_x \quad x_{\max} - 2 \cdot p_x \quad \dots \quad x_{\min}]$$

În cazul pasului unitar negativ ( $p_x=-1$ ), nu se poate utiliza instrucțiunea simplificată:  $x=x_{\max}:x_{\min}$ , datorită faptului că întotdeauna operatorul  $:$  incrementează și nu decrementează, așa încât, pentru obținerea vectorului linie:

$$[x_{\max} \quad x_{\max} - 1 \quad x_{\max} - 2 \quad \dots \quad x_{\min}]$$

trebuie să se folosească instrucțiunea:

$$x=x_{\max}:-1:x_{\min}$$

#### Problema 4.2

Să se definească vectorii cu pas constant liniar cunoscând:

- $x_{\min}=0; x_{\max}=10; p_x=2$  (vector linie)
- $y_{\min}=0; y_{\max}=5; p_y=1$  (vector linie)
- $u_{\min}=-10; u_{\max}=0; p_u=-5$  (vector linie)
- $v_{\min}=1; v_{\max}=5; p_v=-1$  (vector linie)
- $w_{\min}=0; w_{\max}=10; p_w=2$  (vector coloană)

#### Rezolvare

```
>> x=0:2:10
x =
     0     2     4     6     8    10
>> y=1:5
y =
     1     2     3     4     5
>> u=10:-5:-10
u =
    10     5     0    -5   -10
>> v=5:1
v =
Empty matrix: 1-by-0
>> v=5:-1:1
v =
     5     4     3     2     1
>> w=[-1:1]'
w =
```



-1  
0  
1

**Metoda numărului de elemente.** Dacă se cunosc limitele vectorului crescător  $x_{\min}$  și  $x_{\max}$ , precum și numărul de elemente  $n_x$  ale vectorului, atunci pentru definirea vectorilor crescători se va utiliza instrucțiunea, [5]:

```
x=linspace(xmin,xmax,nx)
```

obținându-se următoarele valori:

$$[x_{\min} \quad x_{\min} + 1 \cdot p_x \quad x_{\min} + 2 \cdot p_x \quad \dots \quad x_{\max}]$$

în care pasul vectorului are expresia:

$$p_x = (x_{\max} - x_{\min}) / (n_x - 1)$$

În cazul vectorilor cu pas constant liniar având  $n_x=100$  de elemente, se poate utiliza instrucțiunea simplificată:

```
x=linspace(xmin,xmax)
```

Generarea vectorilor descrescători cu pas constant liniar se realizează în același mod, printr-o instrucțiune de genul:

```
x=linspace(xmax,xmin,nx)
```

obținându-se următoarele valori:

$$[x_{\max} \quad x_{\max} - 1 \cdot p_x \quad x_{\max} - 2 \cdot p_x \quad \dots \quad x_{\min}]$$

### Problema 4.3

Să se definească sub forma unui vector linie crescător, a unui vector linie descrescător, precum și a unui vector coloană descrescător, vectorul cu pas constant liniar cunoscând:

$$x_{\min}=0; x_{\max}=100; n_x=5$$

### Rezolvare

```
>> x=linspace(0,100,5)
x =
    0    25    50    75   100
>> x=linspace(100,0,5)
x =
  100    75    50    25     0
>> x=linspace(100,0,5) '
x =
  100
   75
   50
   25
    0
```

**Metoda pasului logaritm.** Pentru definirea vectorilor cu elemente asociate cu pas constant logaritm se utilizează instrucțiunea, [6]:

```
x=logspace (xmin, xmax, nx)
```

care va conduce la un vector având  $n_x$  elemente, distribuite logaritm între limitele  $10^{x_{\min}}$  și  $10^{x_{\max}}$ .

În cazul vectorilor cu pas constant logaritm având  $n_x=50$  de elemente, se poate utiliza instrucțiunea simplificată:

```
x=logspace (xmin, xmax)
```

#### Problema 4.4

Să se definească ca vectori linie și coloană crescători și descrescători, vectori cu pas constant logaritm  $x$  și  $y$ , cunoscând:

a)  $x_{\min}=-2$ ;  $x_{\max}=2$ ;  $n_x=5$

b)  $y_{\min}=-1$ ;  $y_{\max}=1$ ;  $n_y=3$

#### Rezolvare

```
>> x=logspace (-2, 2, 5)
```

```
x =
```

```
    0.0100    0.1000    1.0000    10.0000   100.0000
```

```
>> x=logspace (-2, 2, 5) '
```

```
x =
```

```
    0.0100
```

```
    0.1000
```

```
    1.0000
```

```
   10.0000
```

```
  100.0000
```

```
>> x=logspace (2, -2, 5)
```

```
x =
```

```
  100.0000   10.0000    1.0000    0.1000    0.0100
```

```
>> x=logspace (2, -2, 5) '
```

```
x =
```

```
  100.0000
```

```
   10.0000
```

```
    1.0000
```

```
    0.1000
```

```
    0.0100
```

```
>> y=logspace (-1, 1, 3)
```

```
y =
```

```
    0.1000    1.0000   10.0000
```

```
>> y=logspace (-1, 1, 3) '
```

```
y =
```

```
    0.1000
```

```
    1.0000
```

```

10.0000
>> y=logspace(1,-1,3)
y =
10.0000    1.0000    0.1000
>> y=logspace(1,-1,3) '
y =
10.0000
1.0000
0.1000

```

### Observații

- În cazul generării vectorilor cu elemente asociate pentru care se cunosc parametrii caracteristici  $x_{\min}$ ,  $x_{\max}$ ,  $p_x$  sau  $n_x$ , se pot utiliza și structurile de calcul iterativ prezentate în figura 4.1.

<pre> for i=1:nx     x(i)=xmin+px*(i-1); end </pre> <p>a) structură repetitivă cu contor</p> <pre> i=1; x(i)=xmin; while x(i)&lt;xmax     i=i+1;     x(i)=xmin+px*(i-1); end </pre> <p>c) structură repetitivă cu test inițial</p>	<pre> i=1; x(i)=xmin; while x(i)&lt;xmax     i=i+1;     x(i)=x(i-1)+px; end </pre> <p>b) structură repetitivă cu test inițial</p> <pre> i=1; while i&lt;=nx     x(i)=xmin+(i-1)*px;     i=i+1; end </pre> <p>d) structură repetitivă cu test inițial</p>
--	--

**Figura 4.1.** Structuri de calcul iterativ utilizate pentru generarea vectorilor.

În cazul structurii iterative cu contor de la figura 4.1, a), pentru prima valoare a indicelui,  $i = 1$ , se obține prima valoare a vectorului  $x_1 = x_{\min}$ , iar pentru ultima valoare a indicelui  $i = n_x$  se obține ultima valoare a vectorului  $x_{n_x} = x_{\min} + p_x \cdot (n_x - 1) = x_{\max}$ .

În cazul structurilor repetitive cu test inițial de la figurile 4.1, b) și 4.1, c), expresia logică care se testează pentru a trece la următoarea iterație, constă în verificarea valorii curente a vectorului față de valoarea sa maximă  $x_{\max}$ , motiv pentru care este necesară definirea prealabilă atât a primei valori a indicelui, cât și a primei valori a vectorului.

În cazul structurii iterative cu test inițial de la figura 4.1, d), expresia logică constă în verificarea valorii curente a indicelui față de numărul maxim  $n_x$  de elemente ale vectorului, fiind astfel necesară doar definirea primei valori a indicelui structurii iterative.

- În cazul unui vector cu elemente neasociate, dar care pe porțiuni conține unele elemente asociate, se procedează mai întâi la descompunerea vectorului inițial în subvectori având elemente asociate și apoi, prin metoda concatenării se realizează definirea vectorului.

Spre exemplu, un vector cu 10 elemente de forma:

$$x = [x_1 \ x_2 \ x_3 \ \dots \ x_{10}]$$

care are primele 4 elemente și ultimele 4 elemente independent asociate de forma unor vectori crescători cu pas unitar, poate fi definit, prin metoda concatenării, cu o instrucțiune de tipul:

```
x=[sx1 sx2 sx3]
```

în care `sx1`, `sx2` și `sx3` sunt sub-vectori definiți prin:  
`sx1=x1:x4`; `sx2=[x5 x6]`; și `sx3=x7:x10`.

### Problema 4.5

Să se definească următorii vectori cu elemente asociate pe porțiuni:

a)  $x = [1 \ 2 \ 3 \ 0 \ 7 \ 8 \ 9]$

b)  $y = [-8 \ -6 \ -4 \ -2 \ 0 \ 3 \ 4 \ 5]$

c)  $u = [-15 \ -10 \ -5 \ 0 \ 9 \ 8 \ 7]$

d)  $v = [1 \ 2 \ 3 \ 4 \ 3 \ 2 \ 1]$

e)  $w = [1 \ 2 \ 3 \ 4 \ 1 \ 2 \ 3 \ 4]$

### Rezolvare

```
>> x=[1:3 0 7:9]
```

```
x =
```

```
     1     2     3     0     7     8     9
```

```
>> y=[-8:2:0 3:5]
```

```
y =
```

```
    -8    -6    -4    -2     0     3     4     5
```

```
>> u=[-15:5:0 9:-1:7]
```

```
u =
```

```
   -15   -10    -5     0     9     8     7
```

```
>> >> v=[1:4 3:-1:1]
```

```
v =
```

```
     1     2     3     4     3     2     1
```

```
>> w=[1:4 1:4]
```

```
w =
```

```
     1     2     3     4     1     2     3     4
```

```
>> w1=1:4;w=[w1 w1]
```

```
w =
```

```
     1     2     3     4     1     2     3     4
```

## 4.2. EXTRAGEREA ELEMENTELOR UNUI VECTOR

Se consideră un vector oarecare având  $n_x$  elemente:

$$x = [x_1 \quad x_2 \quad x_3 \quad \dots \quad x_{n_x}]$$

Scopul operațiunii de extragere a elementelor unui vector poate fi efectuarea unor operații la nivelul elementelor sau subvectorilor unui vector sau chiar ștergerea unor elemente sau subvectori dintr-un vector.

Extragerea elementelor unui vector se realizează prin intermediul indicilor, după cum urmează:

- Primul și respectiv, ultimul element al vectorului se obțin cu instrucțiunile:  
 $x(1)$  ;  
 $x(nx)$  ;
- În general, extragerea dintr-un vector a elementului cu indicele  $j$  se obține cu instrucțiunea:  
 $x(j)$  ;
- În cazul extragerii mai multor elemente ale unui vector trebuie specificați între paranteze pătrate indicii tuturor elementelor care urmează a fi extrase. În general, pentru extragerea elementelor cu indicii  $i$ ,  $j$  și  $k$  se utilizează instrucțiunea:  
 $x([i \ j \ k])$  ;
  - Pentru extragerea primilor 5 termeni ai unui vector se utilizează instrucțiunea:  
 $x(1:5)$  ;
  - Pentru extragerea primilor 4 și ultimilor 4 termeni ai unui vector cu 10 elemente se utilizează instrucțiunea:  
 $x([1:4 \ 7:10])$  ;
- Extragerea tuturor elementelor unui vector și afișarea elementelor sub forma unui vector coloană se realizează cu instrucțiunea:  
 $x(:)$  ;
- Ștergerea unui subvector dintr-un vector se realizează cu ajutorul unei instrucțiuni prin care fiecărui element al subvectorului care trebuie să fie șters  $i$  se atribuie elementul nul (matrice goală  $[]$ ) .
  - De exemplu, pentru ștergerea elementului al treilea al vectorului  $x$  se utilizează instrucțiunea:  
 $x(3) = []$   
care conduce la rezultatul:  
$$x = [x_1 \quad x_2 \quad x_4 \quad \dots \quad x_{n_x}]$$
  - Pentru ștergerea primelor trei și a ultimului element al vectorului  $x$  se utilizează instrucțiunea  
 $x([1:3 \ nx]) = []$   
care conduce la rezultatul:  
$$x = [x_4 \quad x_5 \quad \dots \quad x_{n_x-1}]$$

### Problema 4.6

Se consideră vectorul:

$$x = [0 \ 2 \ 4 \ 6 \ 8 \ 10]$$

Să se extragă elementele independente:

$$x_1, x_6, x_7$$

și subvectorii:

$$\begin{bmatrix} x_1 & x_2 & x_3 \\ x_2 & x_3 & x_4 & x_5 \\ x_1 & x_2 & x_5 & x_6 \end{bmatrix}$$

Să se ștergă primele două elemente ale vectorului  $x$ .

Să se ștergă primul și ultimele două elemente ale vectorului  $x$ .

### Rezolvare

```
>> x=0:2:10
x =
     0     2     4     6     8    10
>> x(1)
ans =
     0
>> x(6)
ans =
    10
>> x(7)
Index exceeds matrix dimensions.
>> x(1:3)
ans =
     0     2     4
>> x(2:5)
ans =
     2     4     6     8
>> x([1:2 5:6])
ans =
     0     2     8    10
>> x([1 2])=[]
x =
     4     6     8    10
>> x([1 5 6])=[]
x =
     2     4     6
```

### Observații

Încercarea de a extrage elementul cu indicele 7 ( $x_7$ ) din vectorul  $x$ , conduce la o eroare de tipul `Index exceeds matrix dimensions`, datorită faptului că vectorul  $x$  are doar șase elemente.

### 4.3. OPERAȚII CU ELEMENTELE UNUI VECTOR

Se consideră vectorul oarecare  $x$ , cu o linie și  $n_{Cx}$  coloane:

$$x = [x_1 \quad x_2 \quad x_3 \quad \dots \quad x_{n_{Cx}}]$$

Principalele operații care se pot efectua cu elementele vectorului  $x$  sunt:

- Dimensiunea vectorului,  $n_{Lx} \times n_{Cx}$ , [7]:

$$[n_{Lx} \quad n_{Cx}] = \text{size}(x);$$

în care  $n_{Lx}$  și  $n_{Cx}$  reprezintă numărul de linii, respectiv numărul de coloane ale vectorului.

- Numărul elementelor vectorului,  $n_x$ , [8]:

$$n_x = \max(\text{size}(x));$$

$$n_x = \text{length}(x);$$

- Valoarea maximă a elementelor unui vector,  $x_{max}$ , [9]:

$$[x_{max} \quad ix_{max}] = \max(x);$$

$$x_{max} = \max(x);$$

în care  $ix_{max}$  reprezintă indicele valorii maxime a elementelor unui vector. În cazul în care în același vector se află mai multe valori maxime egale, atunci  $ix_{max}$  reprezintă indicele primei valori maxime.

- Valoarea minimă a elementelor unui vector,  $x_{min}$ , [10]:

$$[x_{min} \quad ix_{min}] = \min(x);$$

$$x_{min} = \min(x);$$

în care  $ix_{min}$  reprezintă indicele valorii minime a elementelor unui vector. În cazul în care în același vector se află mai multe valori minime egale, atunci  $ix_{min}$  reprezintă indicele primei valori minime.

- Suma elementelor unui vector,  $S$ , [11]:

$$S = \sum_{i=1}^n x_i$$

$$S = \text{sum}(x);$$

- Produsul elementelor unui vector,  $P$ , [12]:

$$P = \prod_{i=1}^n x_i$$

$$P = \text{prod}(x);$$

- Media aritmetică a elementelor unui vector,  $\bar{x}$ , [13]:

$$\bar{x} = \frac{1}{n_x} \cdot \sum_{i=1}^{n_x} x_i$$

`ma=mean(x)` ;

- Abaterea medie pătratică a elementelor unui vector,  $s$ , [14]:

$$s = \sqrt{\frac{1}{n_x - 1} \sum_{i=1}^{n_x} (x_i - \bar{x})^2}$$

`s=std(x)` ;

- Transpunerea vectorului este operația prin care un vector cu dimensiunea  $(1, n_x)$  se transformă într-un vector având dimensiunea  $(n_x, 1)$ , adică un vector linie se transformă într-un vector coloană, sau invers, conform relațiilor:

$$[x_1 \quad x_2 \quad \dots \quad x_{n_x}]' = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n_x} \end{bmatrix}$$

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n_x} \end{bmatrix}' = [x_1 \quad x_2 \quad \dots \quad x_{n_x}]$$

Operația de transpunere se realizează cu una din instrucțiunile, [15]:

`xT=x'` ;

`xT=transpose(x)` ;

- Sortarea elementelor unui vector este operația prin care se realizează:

- aranjarea elementelor vectorului în ordine crescătoare, astfel încât vectorul sortat va avea elementele:

$$x = [x_1 \quad x_2 \quad x_3 \quad \dots \quad x_{n_x}]$$

cu proprietatea:

$$x_1 < x_2 < x_3 < \dots < x_{n_x}$$

Sortare în ordine crescătoare a elementelor vectorului  $x$  se realizează cu una din instrucțiunile, [16]:

`xs=sort(x)`

`xs=sort(x, 'ascend')` ;



- o aranjarea elementelor vectorului în ordine descrescătoare, astfel încât vectorul sortat va avea elementele:

$$x = [x_1 \quad x_2 \quad x_3 \quad \dots \quad x_{nx}]$$

cu proprietatea:

$$x_1 > x_2 > x_3 > \dots > x_{nx}$$

Sortare în ordine descrescătoare a elementelor vectorului  $x$  se realizează cu instrucțiunea, [16]:

```
xs=sort(x, 'descend');
```

- Norma-p a vectorului  $x = [x_1 \quad x_2 \quad x_3 \quad \dots \quad x_n]$  se definește prin:

$$|x|_p = \sqrt[p]{\sum_{i=1}^n |x_i|^p}$$

cu trei cazuri particulare remarcabile:

- o norma-1, definită prin  $|x|_1 = \sum_{i=1}^n |x_i|$
- o norma-2 (norma euclidiană), definită prin:  $|x|_2 = \sqrt{\sum_{i=1}^n |x_i|^2}$
- o norma-∞, definită prin:  $|x|_\infty = \max_{i=1,n} |x_i|$

Calculul normei-2 a unui vector se face cu instrucțiunea, [17]:

```
n2x=norm(x)
```

În general, pentru calculul normei-p se utilizează instrucțiunea:

```
npX=norm(x, p)
```

#### Problema 4.7

Se consideră vectorul:

$$x = [1 \quad 3 \quad 7 \quad 2 \quad 9 \quad 8]$$

Să se determine: dimensiunea vectorului; numărul elementelor vectorului; valoarea maximă; valoarea minimă; suma elementelor vectorului; produsul elementelor vectorului; media aritmetică a elementelor vectorului; abaterea media pătratică; sortarea elementelor vectorului în ordine crescătoare și descrescătoare; norma-1, norma-2 și norma-∞.

#### Rezolvare

```
>> x=[1 3 7 2 9 8]
```

```
x =
```

```
    1     3     7     2     9     8
```

```
>> [nLx nCx]=size(x)
```

```
nLx =
```

```
    1
```

```
nCx =
```

```
    6
```

```

>> nx=length(x)
nx =
     6
>> xmax=max(x)
xmax =
     9
>> xmin=min(x)
xmin =
     1
>> S=sum(x)
S =
    30
>> P=prod(x)
P =
    3024
>> ma=mean(x)
ma =
     5
>> s=std(x)
s =
    3.4059
>> xT=x'
xT =
     1
     3
     7
     2
     9
     8
>> xsc=sort(x,'ascend')
xsc =
     1     2     3     7     8     9
>> xsd=sort(x,'descend')
xsd =
     9     8     7     3     2     1
>> n1x=norm(x,1)
n1x =
    30
>> n2x=norm(x,2)
n2x =
    14.4222
ninfx=norm(x,inf)
ninfx =
     9
>>

```

#### 4.4. OPERAȚII ÎNTRE UN SCALAR ȘI UN VECTOR

Se consideră un scalar  $a$  și un vector  $x$ :

$$a \in \mathbb{R}$$

$$x = [x_1 \quad x_2 \quad x_3 \quad \dots \quad x_{nx}]$$

Se pune problema efectuării următoarelor calcule, [1, 2, 3]:

$$x + a = [x_1 + a \quad x_2 + a \quad x_3 + a \quad \dots \quad x_{nx} + a]$$

$$x - a = [x_1 - a \quad x_2 - a \quad x_3 - a \quad \dots \quad x_{nx} - a]$$

$$x \cdot a = [x_1 \cdot a \quad x_2 \cdot a \quad x_3 \cdot a \quad \dots \quad x_{nx} \cdot a]$$

$$x/a = [x_1/a \quad x_2/a \quad x_3/a \quad \dots \quad x_{nx}/a]$$

$$a/x = [a/x_1 \quad a/x_2 \quad a/x_3 \quad \dots \quad a/x_{nx}]$$

$$x^a = [x_1^a \quad x_2^a \quad x_3^a \quad \dots \quad x_{nx}^a]$$

Toate aceste calcule reprezintă operații de tip element-cu-element, pentru care operatorii de adunare, scădere, înmulțire, împărțire și ridicare la putere acționează între variabila scalară  $a$  și fiecare element  $x_1, x_2, \dots, x_{nx}$  din variabila vectorială  $x$ . Operațiile de tip element-cu-element se mai numesc și operații vectorizate, întrucât cel puțin unul din operanți este un vector. Se utilizează operatorii aritmetici obișnuiți, mai puțin în cazul operațiilor  $a/x$  și  $x^a$ , pentru care trebuie să se utilizeze operatorul de „împărțire cu punct” [18], respectiv de „ridicare la putere cu punct” [19], corespunzător operațiilor de tip element-cu-element, conform instrucțiunilor:

```
x+a
x-a
x*a
x/a
a./x
x.^a
```

##### Problema 4.8

Se consideră vectorul:

$$x = [2 \quad 4 \quad 6 \quad 8 \quad 10]$$

și un scalar  $a=2$ .

Să se efectueze următoarele operații:

$$x + a, x - a$$

$$x \cdot a, x/a,$$

$$a/x$$

$$x^a$$

##### Rezolvare

```
>> a=2
```

```
a =
```

```
2
```

```

>> x=2:2:10
x =
     2     4     6     8    10
>> x+a
ans =
     4     6     8    10    12
>> x-a
ans =
     0     2     4     6     8
>> x*a
ans =
     4     8    12    16    20
>> x/a
ans =
     1     2     3     4     5
>> a/x
Error using /
Matrix dimensions must agree.
>> a./x
ans =
  1.0000    0.5000    0.3333    0.2500    0.2000
>> x^a
Error using ^
Inputs must be a scalar and a square matrix.
To compute elementwise POWER, use POWER (.^)
instead.
>> x.^a
ans =
     4    16    36    64   100

```

### Observații

- În cazul operației  $a/x$  vectorul  $x$  se află la numitor, expresia fiind echivalentă cu  $a \cdot x^{-1}$ , prin urmare trebuie utilizat operatorul de „împărțire cu punct” ( $./$ ) corespunzător unei operații element-cu-element. În caz contrar se obține mesajul de eroare: Error using /, Matrix dimensions must agree.
- În cazul operației  $x^a$ , de asemenea trebuie utilizat operatorul cu punct, în caz contrar obținându-se mesajul de eroare: Error using ^. Inputs must be a scalar and a square matrix. To compute elementwise POWER, use POWER (.^) instead. În mod implicit se presupune că  $x$  este o matrice și chiar și în acest caz, operația  $x^a$ , care este echivalentă cu  $x^a = x \cdot x \cdot \dots \cdot x$  de  $a$  ori, ar avea sens doar dacă  $x$  ar fi o matrice pătrată. Cu atât mai mult, în cazul în care  $x$  este un vector trebuie utilizat operatorul cu punct ( $.^$ ) corespunzător unei operații element-cu-element.

#### 4.5. OPERAȚII ÎNTRE DOI VECTORI

Se consideră doi vectori  $x$  și  $y$ :

$$x = [x_1 \quad x_2 \quad x_3 \quad \dots \quad x_{nx}]$$
$$y = [y_1 \quad y_2 \quad y_3 \quad \dots \quad y_{ny}]$$

având același număr de elemente,  $n_x = n_y$ .

Se pune problema efectuării următoarelor calcule, [1, 2, 3]:

$$x + y = [x_1 + y_1 \quad x_2 + y_2 \quad x_3 + y_3 \quad \dots \quad x_{nx} + y_{ny}]$$

$$x - y = [x_1 - y_1 \quad x_2 - y_2 \quad x_3 - y_3 \quad \dots \quad x_{nx} - y_{ny}]$$

$$x \cdot y = [x_1 \cdot y_1 \quad x_2 \cdot y_2 \quad x_3 \cdot y_3 \quad \dots \quad x_{nx} \cdot y_{ny}]$$

$$x/y = [x_1/y_1 \quad x_2/y_2 \quad x_3/y_3 \quad \dots \quad x_{nx}/y_{ny}]$$

$$x^y = [x_1^{y_1} \quad x_2^{y_2} \quad x_3^{y_3} \quad \dots \quad x_{nx}^{y_{ny}}]$$

Toate aceste calcule reprezintă operații de tip element-cu-element, pentru care operatorii de adunare, scădere, înmulțire, împărțire și ridicare la putere acționează între fiecare element al primului vector și elementele corespondente ale celei de-a doua variabile vectoriale.

Se utilizează operatori aritmetici obișnuiți doar în cazul adunării și scăderii, în rest trebuie utilizați operatorii aritmetici cu punct corespunzător operațiilor de tip element-cu-element („înmulțire cu punct” [20], „împărțire cu punct” [18], „ridicare la putere cu punct” [19]), conform instrucțiunilor:

$$x+y$$

$$x-y$$

$$x \cdot y$$

$$x / y$$

$$x.^y$$

În cazul în care se urmărește efectuarea operației aritmetice de înmulțire a celor doi vectori specifică calculului matriceal, atunci trebuie îndeplinită condiția:  $n_{cx} = n_{ly}$ , adică unul din vectori trebuie să fie vector linie, iar celălalt trebuie să fie un vector coloană.

Spre exemplu, dacă se consideră doi vectori  $x$  și  $y$  având dimensiunile  $(1, n_{cx})$  și  $(n_{ly}, 1)$  de forma:

$$x = [x_1 \quad x_2 \quad \dots \quad x_{n_{cx}}]$$

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{n_{ly}} \end{bmatrix}$$

și dacă  $n_{cx} = n_{ly}$ , atunci, conform regulilor calculului matriceal au sens operațiile:

$$x*y;$$

$$y*x;$$

În cazul primei operații, rezultatul este un scalar obținut prin:

$$x \cdot y = x_1 \cdot y_1 + x_2 \cdot y_2 + \dots + x_{n_{Cx}} \cdot y_{n_{Ly}}$$

Acest rezultat are semnificația produsului scalar al celor doi vectori. În general, dat fiind doi vectori  $x$  și  $y$  având aceeași dimensiune  $(1, n_{Cx})$  și  $(1, n_{Cy})$ , cu  $n_{Cx} = n_{Cy} = n$ , atunci produsul scalar al celor doi vectori definit prin:

$$\vec{x} \cdot \vec{y} = \sum_{i=1}^n x_i \cdot y_i$$

se obține cu o instrucțiune de forma:

$$PS = \text{sum}(x \cdot y);$$

În cazul celei de-a doua operații, rezultatul este o matrice având dimensiunea  $(n_{Ly}, n_{Cx})$  obținută prin:

$$y \cdot x = \begin{pmatrix} y_1 \cdot x_1 & y_1 \cdot x_2 & \dots & y_1 \cdot x_{n_{Cx}} \\ y_2 \cdot x_1 & y_2 \cdot x_2 & \dots & y_2 \cdot x_{n_{Cx}} \\ \vdots & \vdots & \ddots & \vdots \\ y_{n_{Ly}} \cdot x_1 & y_{n_{Ly}} \cdot x_2 & \dots & y_{n_{Ly}} \cdot x_{n_{Cx}} \end{pmatrix}$$

#### Problema 4.9

Se consideră vectorul:

$$x = [1 \quad 3 \quad 6 \quad 12]$$

și scalarul  $a = 23.6$

Să se calculeze:

$$a) y = \frac{x+1}{x} + x^2 + 2x(a-1)$$

$$b) z = a \cdot \frac{x + \sqrt{a \cdot y}}{2y+1}$$

$$c) u = \sqrt{axy} + \frac{\sqrt{z}}{x} + \frac{a \cdot y}{z+1}$$

#### Rezolvare

```
>> a=2
```

```
a =
```

```
2
```

```
>> x=[1 3 6 9]
```

```
x =
```

```
1      3      6      9
```

```
>> y=(x+1) ./x+x.^2+2*x*(a-1)
```

```
y =
```

```
5.0000    16.3333    49.1667   100.1111
```

```
>> z=a*(x+sqrt(a*y))./(2*y+1)
```

```
z =
```

```
0.7568    0.5178    0.3205    0.2301
```

```
>> u=sqrt(a*x.*y)+sqrt(z)./x+a*y./(z+1)
u =
    9.7244    31.6624    98.8531   205.2732
>>
```

### Problema 4.10

Se consideră funcția:

$$f: [0; 0,8] \rightarrow \mathbb{R}, f(x) = x^2 + 3x + 1$$

Se cere:

- Să se genereze domeniul de definiție  $x$  al funcției  $f$  cunoscând valoarea pasului  $p_x=0,2$ .
- Să se determine valorile funcției  $f(x)$ .
- Să se calculeze valorile funcțiilor:

$$g(x) = f(x)/2 + 3$$

$$h(x) = [f(x)]^{3/2}$$

$$u(x) = \frac{2g(x) + 1}{3h(x) + 2}$$

$$v(x) = 1 + 2[f(x)]^2 + 3[g(x)]^3$$

$$w(x) = f(x) \cdot g(x) + [3 + u(x)]^{-1}$$

- Să se rezolve problema folosind și structurile de calcul iterativ cu contor (for) și cu test inițial (while).
- Să se rezolve problema folosind funcții de tip anonymous definite în structura unui fișier de tip script.

### Rezolvare

```
>> x=0:0.2:0.8
x =
    0    0.2000    0.4000    0.6000    0.8000
>> f=x.^2+3*x+1
f =
    1.0000    1.6400    2.3600    3.1600    4.0400
>> g=f/2+3
g =
    3.5000    3.8200    4.1800    4.5800    5.0200
>> h=f.^(3/2)
h =
    1.0000    2.1002    3.6255    5.6173    8.1203
>> u=(2*g+1)./(3*h+2)
u =
    1.6000    1.0409    0.7269    0.5389    0.4188
>> v=1+2*f.^2+3*g.^3
v =
   131.6250   173.6081   231.2431   309.1869   413.1612
```

```
>> w=f.*g+(3+u).^(-1)
w =
    3.7174    6.5123   10.1331   14.7554   20.5733
```

Rezolvarea problemei folosind o structură iterativă cu contor este realizată cu ajutorul următorului fișier de tip `script`:

```
%% CALCULE CU VECTORI
clear all;close all;clc;
%% DOMENIUL DE DEFINITIE
xmin=0;xmax=0.8;px=0.2;
x=xmin:px:xmax;
nx=length(x);
%% BLOC DE CALCUL
for i=1:nx
    f(i)=x(i)^2+3*x(i)+1;
    g(i)=f(i)/2+3;
    h(i)=f(i)^(3/2);
    u(i)=(2*g(i)+1)/(3*h(i)+2);
    v(i)=1+2*f(i)^2+3*g(i)^3;
    w(i)=f(i)*g(i)+(3+u(i))^( -1);
end
%% PREZENTAREA REZULTATELOR
disp('x f(x) g(x) h(x) u(x) v(x) w(x)')
disp([x; f; g; h; u; v; w]')
```

Rezolvarea problemei folosind o structură iterativă cu test inițial este realizată cu ajutorul unui fișier de tip `script`, care spre deosebire de fișierul anterior, are un alt bloc de calcul:

```
%% BLOC DE CALCUL
i=1;
while i<=nx
    f(i)=x(i)^2+3*x(i)+1;
    g(i)=f(i)/2+3;
    h(i)=f(i)^(3/2);
    u(i)=(2*g(i)+1)/(3*h(i)+2);
    v(i)=1+2*f(i)^2+3*g(i)^3;
    w(i)=f(i)*g(i)+(3+u(i))^( -1);
    i=i+1;
end
```

Rezolvarea problemei folosind funcții de tip `anonymous` este realizată cu ajutorul unui fișier de tip `script`, care spre deosebire de fișierele anterioare, are un alt bloc de calcul, dar și un alt bloc de prezentare a rezultatelor:



```

%% BLOC DE CALCUL
f=@(x)x.^2+3*x+1;
g=@(x)f(x)/2+3;
h=@(x)f(x).^^(3/2);
u=@(x)(2*g(x)+1)./(3*h(x)+2);
v=@(x)1+2*f(x).^2+3*g(x).^3;
w=@(x)f(x).*g(x)+(3+u(x)).^(-1);
%% PREZENTAREA REZULTATELOR
disp('    x    f(x)    g(x)    h(x)    u(x)    v(x)    w(x)')
disp([x; f(x); g(x); h(x); u(x); v(x); w(x)]')

```

Indiferent însă de metoda utilizată, lansarea în execuție a fișierelor `script` respective, conduce la următorul rezultat:

x	f(x)	g(x)	h(x)	u(x)	v(x)	w(x)
0	1.00	3.50	1.0000	1.6000	131.6250	3.7174
0.2	1.64	3.82	2.1002	1.0409	173.6081	6.5123
0.4	2.36	4.18	3.6255	0.7269	231.2431	10.1331
0.6	3.16	4.58	5.6173	0.5389	309.1869	14.7554
0.8	4.04	5.02	8.1203	0.4188	413.1612	20.5733

### Observații

- Pentru rezolvarea acestor tipuri de probleme se recomandă utilizarea fișierelor de tip `script` și nu lucrul direct în fereastra de comenzi.
- În cazul structurilor iterative (cu contor `for` sau cu test inițial `while`) este necesară determinarea numărului de elemente  $n_x$  ale vectorului  $x$  care reprezintă domeniul de definiție al funcției  $f(x)$ . În plus, în cazul structurii iterative cu test inițial `while` este necesară inițializarea contorului cu instrucțiunea  $i=1$ , dar și incrementarea contorului cu instrucțiunea  $i = i + 1$ , după execuția instrucțiunilor de calcul a funcțiilor și imediat înainte de testarea valorii de adevăr a expresiei logice  $i \leq n_x$ .
- Metoda funcțiilor `anonymous` nu necesită cunoașterea numărului de elemente  $n_x$  ale domeniului de definiție al funcției  $f(x)$ . Spre deosebire de structurile iterative, în cazul funcțiilor de tip `anonymous` trebuie să se folosească operatorii cu punct (`.*`, `./`, `.^`) corespunzători operațiilor element-cu-element. Evaluarea funcțiilor astfel definite și calcularea valorilor acestora în fiecare punct al domeniului de definiție se face doar în momentul apelării acestor funcții, adică în cea de-a doua instrucțiune de prezentare a rezultatelor `disp`. Se recomandă utilizarea metodei funcțiilor de tip `anonymous` pentru că se asigură astfel o mai bună manipulare a funcțiilor, precum și a operației de compunere a funcțiilor.

## BIBLIOGRAFIE

1. Dodun O., Calcul numeric asistat – Teorie și aplicații în MATLAB, Ed. Performantica, Iași, 2013.
2. Palm W.J. III, MATLAB 6 per l'ingegneria e le scienze, McGraw Hill, Milano, 2001.
3. Palm W.J. III, Introduction to MATLAB 7 for Engineers, McGraw Hill, New York, 2005.
4. Mathworks, Create Vectors, Array Subscripting, and for-Loop Iterators, <http://www.mathworks.com/help/matlab/ref/colon.html>, accesat la 14.02.2014.
5. Mathworks, Generate Linearly Spaced Vectors, <http://www.mathworks.com/help/matlab/ref/linspace.html>, accesat la 14.02.2014.
6. Mathworks, Generate Logarithmically Spaced Vectors, <http://www.mathworks.com/help/matlab/ref/logspace.html>, accesat la 14.02.2014.
7. Mathworks, Array Dimensions, <http://www.mathworks.com/help/matlab/ref/size.html>, accesat la 16.02.2014.
8. Mathworks, Length of Vector or Largest Array Dimension, <http://www.mathworks.com/help/matlab/ref/length.html>, accesat la 16.02.2014.
9. Mathworks, Largest Elements in Array, <http://www.mathworks.com/help/matlab/ref/max.html>, accesat la 16.02.2014.
10. Mathworks, Smallest Elements in Array, <http://www.mathworks.com/help/matlab/ref/min.html>, accesat la 16.02.2014.
11. Mathworks, Sum of Array Elements, <http://www.mathworks.com/help/matlab/ref/sum.html>, accesat la 16.02.2014.
12. Mathworks, Product of Array Elements, <http://www.mathworks.com/help/matlab/ref/prod.html>, accesat la 16.02.2014.
13. Mathworks, Average or Mean Value of Array, <http://www.mathworks.com/help/matlab/ref/mean.html>, accesat la 16.02.2014.
14. Mathworks, Standard Deviation, <http://www.mathworks.com/help/matlab/ref/std.html>, accesat la 16.02.2014.
15. Mathworks, Transpose, <http://www.mathworks.com/help/matlab/ref/transpose.html>, accesat la 16.02.2014.
16. Mathworks, Sort Array Elements in Ascending or Descending Order, <http://www.mathworks.com/help/matlab/ref/sort.html>, accesat la 16.02.2014.
17. Mathworks, Vector and Matrix Norms, <http://www.mathworks.com/help/matlab/ref/norm.html>, accesat la 16.02.2014.
18. MathWorks, Right Array Division, <http://www.mathworks.com/help/matlab/ref/rdivide.html>, accesat la 18.09.2014.
19. MathWorks, Element-Wise Power, <http://www.mathworks.com/help/matlab/ref/power.html>, accesat la 18.09.2014.
20. MathWorks, Element-Wise Multiplication, <http://www.mathworks.com/help/matlab/ref/times.html>, accesat la 18.09.2014.

## CAPITOLUL 5

### REPREZENTĂRI GRAFICE 2D

#### 5.1. TIPURI DE OBIECTE GRAFICE

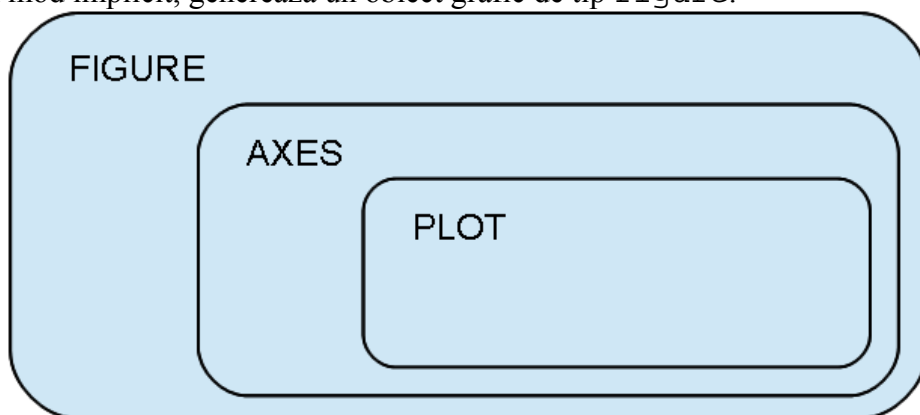
În vederea realizării reprezentării grafice a funcțiilor definite atât la nivel numeric cât și la nivel simbolic, în limbajul de programare MATLAB sunt definite atât structura de elemente caracteristice formată din obiectele grafice specifice, cât și relațiile de dependență dintre acestea. Toate aceste elemente permit realizarea de reprezentări grafice de nivel ridicat care asigură controlul atât al elementelor de reprezentare grafică propriu-zisă, cât și al elementelor de formatare, vizualizare și interogare a reprezentărilor grafice [1, 2, 3, 18].

Principalele obiecte grafice sunt, [4]:

- **obiectul grafic `figure`**. Poate fi creat în mod explicit folosind comanda `figure` și reprezintă o fereastră grafică independentă care conține toate celelalte obiecte grafice. Obiectul grafic de tip `figure` are toate proprietățile obiectelor grafice de tip `window` specifice sistemului de operare: bară de titlu; butoane pentru minimizare, maximizare și închidere; bară cu meniuri; bară cu butoane pentru accesarea rapidă a unor comenzi. De asemenea, obiectul grafic de tip `figure` poate fi redimensionat și deplasat pe display prin metoda `drag&drop`.
- **obiectul grafic `axes`**. Poate fi creat în mod explicit folosind comanda `axes` și reprezintă un obiect grafic inclus în obiectul grafic `figure` care conține diferite obiecte: obiecte de tip date numerice; obiecte geometrice de tip curbă, suprafață sau imagine; obiecte grafice de tip adnotare: titluri, legende, text, bare de culoare. Obiectul grafic `figure` poate conține mai multe obiecte grafice de tip `axes`, crearea acestora făcându-se folosind comanda `subplot`. Obiectele grafice de tip `axes` pot fi definite atât în domeniul 2D cât și în domeniul 3D.
- **obiectul grafic `plot`**. Poate fi creat în mod explicit folosind diferite comenzi specifice (`plot`, `bar`, `plot3`, `contour`, `mesh`, `surf`, etc.) și reprezintă un obiect grafic inclus în obiectul grafic `axes` care conține dependența funcțională propriu-zisă între variabilele

asociate (în cazul reprezentărilor grafice 2D, dependența funcțională  $y = f(x)$  poate fi creată, de exemplu cu instrucțiunea `plot(x, y)`, iar în cazul reprezentărilor grafice 3D, dependența funcțională  $z = f(x, y)$  poate fi creată cu instrucțiunea `surf(X, Y, Z)`, în care  $X, Y$  și  $Z$  sunt matrice asociate vectorilor  $x, y$  și  $z$ ).

Relația de dependență între aceste obiecte grafice este strict determinată, în sensul că, pentru a reprezenta o dependență funcțională, spre exemplu  $y = f(x)$ , este nevoie de utilizarea instrucțiunii `plot(x, y)`, care însă nu poate afișa rezultatul decât în interiorul unui obiect grafic de tip `axes`, care la rândul său, nu poate exista decât în interiorul unui obiect grafic de tip `figure`, figura 5.1. Altfel spus, la utilizarea explicită a instrucțiunilor de reprezentare grafică, de exemplu instrucțiunea `plot`, în mod implicit se creează un obiect grafic de tip `axes`, care la rândul său, tot în mod implicit, generează un obiect grafic de tip `figure`.



**Figura 5.1.** Dependența dintre obiectele grafice `figure`, `axes` și `plot`.

## 5.2. TIPURI DE REPREZENTĂRI GRAFICE 2D

Principalele tipuri de reprezentări grafice 2D pentru care, în limbajul de programare MATLAB, sunt implementate proceduri specifice sunt, [5]

- Grafice de tip `line`:
  - `plot` (grafic 2D de tip linie cu o singura ordonată).
  - `plotyy` (grafic 2D de tip linie cu două ordonate).
  - `loglog` (grafic 2D în coordonate logaritmice).
  - `semilogx` (grafic 2D cu abscisa logaritmică).
  - `semilogy` (grafic 2D cu ordonata logaritmică).
  - `contourslice` (grafic 2D cu linii de contur reprezentate în plane de secțiune ale unui domeniu 3D).
  - `contour` (reprezentare grafică 2D cu linii de contur).
  - `contourf` (reprezentare grafică 2D cu linii de contur și suprafețe colorate).

- `ezcontour` (reprezentare grafică 2D cu linii de contur pentru o funcție simbolică).
  - `ezcontourf` (reprezentare grafică 2D cu linii de contur și suprafețe colorate pentru o funcție simbolică).
  - `ezplot` (grafic 2D de tip linie pentru o funcție simbolică).
- Grafice de tip bar:
  - `bar` (grafic 2D cu bare verticale).
  - `barh` (grafic 2D cu bare orizontale).
  - `hist` (grafic 2D cu bare de tip histogramă).
  - `errorbar` (grafic de tip bare de eroare).
  - `pareto` (grafic de tip Pareto).
- Grafice de tip area:
  - `area` (grafic 2D de tip area).
  - `pie` (grafic 2D de tip pie).
  - `fill` (reprezentare grafică a poligoanelor 2D).
  - `image` (reprezentare grafică a imaginilor 2D).
  - `pcolor` (grafic de tip pseudocolor).
- Grafice de tip direction:
  - `feather` (reprezentare 2D a vectorilor axial echidistanți).
  - `quiver` (reprezentare grafică a câmpurilor vectoriale 2D).
  - `comet` (reprezentare grafică a unei curbe 2D cu animație de tip comet).
  - `streamslice` (reprezentare grafică a liniilor de curent în plane de secțiune ale unui domeniul 3D).
- Grafice de tip radial:
  - `polar` (reprezentare grafică în coordonate polare).
  - `rose` (reprezentare grafică a histogramei în coordonate polare).
  - `compass` (reprezentare grafică a funcțiilor 2D ca vectori orientați pe o rețea de tip circular).
  - `ezpolar` (reprezentare grafică în coordonate polare pentru funcții simbolice).
- Grafice de tip discrete:
  - `stairs` (reprezentare grafică 2D în trepte).
  - `stem` (reprezentare grafică a datelor 2D discrete).
  - `scatter` (reprezentare grafică 2D de tip scatter).
  - `plotmatrix` (reprezentare grafică de tip scatter a datelor numerice de tip matrice).

## 5.3. UTILIZAREA INSTRUCȚIUNII `plot`

### 5.3.1. Reprezentarea unei singure funcții

Se consideră funcția:  $f: [x_{min}, x_{max}] \rightarrow \mathbb{R}$ , definită prin  $y = f(x)$ .

Pentru reprezentarea grafică a dependenței  $y = f(x)$  se utilizează instrucțiunea, [6]:

```
plot(x,y,'pf')
```

în care:  $x$  reprezintă valorile de pe abscisă,  $y$  valorile de pe ordonată, iar  $pf$  sunt parametrii de formatare ai graficului.

Formatarea curbelor se face prin intermediul a trei caracteristici:

- Tip marker (+, o, \*, x, s, d, ^, v, <, >, p, h).
- Tip linie ( - linie continuă, -- linie întreruptă, : linie punctată, - . linie punct).
- Culoare (r - roșu, g - verde, b - albastru, m - mov, y - galben, k - negru, w - alb).

### Problema 5.1

Se consideră funcția:

$$f: [-\pi; \pi] \rightarrow \mathbb{R}, y = \sin x.$$

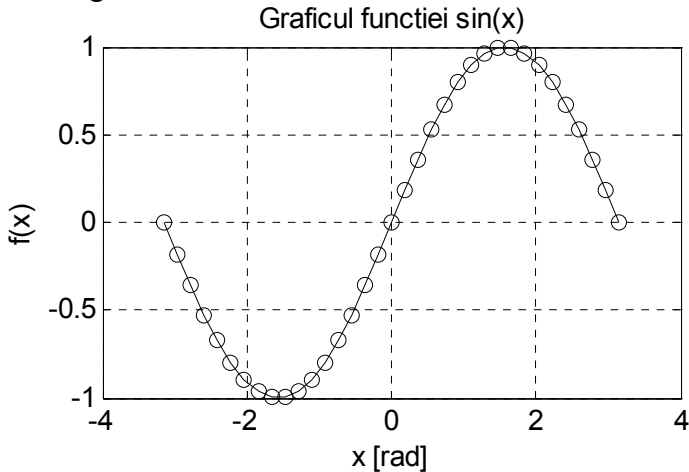
Să se realizeze un fișier de tip `script` prin care să se reprezinte grafic funcția.

### Rezolvare

Rezolvarea problemei este realizată cu următorul fișier `script`:

```
%% GRAFICE 2D (1)
clear all;close all;clc;
%% DATE INITIALE
xmin=-pi;xmax=pi;nx=35;
x=linspace(xmin,xmax,nx);
%% BLOC DE CALCUL
y=sin(x);
%% REPREZENTARE GRAFICA
figure
plot(x,y,'-ok');
grid on;
xlabel('x [rad]');ylabel('f(x)');
title('Graficul functiei sin(x)');
set(gcf,'Position',[100 100 300 200]);
set(gca,'FontSize',11);
set(get(gca,'XLabel'),'FontSize',11);
set(get(gca,'YLabel'),'FontSize',11);
set(get(gca,'Title'),'FontSize',11);
```

Lansarea în execuție a fișierului conduce la reprezentarea grafică prezentată în figura 5.1.



**Figura 5.1.** Graficul funcției sinus.

### Observații

- Crearea unui nou element de tip figură se realizează cu instrucțiunea `figure`.
- Parametrii de formatare ai curbei specificați în interiorul instrucțiunii `plot` sunt `'-ok'` și conduc la obținerea unei curbe cu linie continuă, cu markere de tip `circle`, de culoare neagră.
- Îmbunătățirea calității reprezentărilor grafice se realizează prin aplicarea diferitelor tipuri de adnotări:

- Aplicarea adnotării de tip `grid` se face cu instrucțiunea:

```
grid on
```

- Aplicarea adnotărilor de etichetare a celor două axe se face cu instrucțiunile:

```
xlabel('x [rad]') și ylabel('f(x)')
```

- Aplicarea adnotării de tip titlu se face cu instrucțiunea:

```
title('Graficul funcției sin(x)')
```

Șirurile de caractere care vor constitui etichetele celor două axe și titlul graficului, reprezintă argumentele instrucțiunilor `xlabel`, `ylabel` și `title` și trebuie introduse între apostrofuri și între paranteze rotunde.

- Controlul dimensiunilor ferestrei grafice în care se va reprezenta un anumit obiect grafic se realizează cu instrucțiunea:

```
set(gcf, 'Position', [100 100 400 250]);
```

Această instrucțiune definește (set) proprietatea `Position` a ferestrei grafice curente (`gcf-get current figure`) ca având valorile `[100 100 400 250]`, în care valorile numerice reprezintă coordonatele colțului din stânga-jos (100,100), lungimea (400) și înălțimea (250) ferestrei grafice curente. Măsurarea acestor dimensiuni de face față de colțul stânga-jos al monitorului. Unitatea de măsură implicită este pixel, însă se pot folosi și alte unități de măsură: inch (1 inch=25,4 mm); centimetri; puncte tipografice (1 dot=1/72 inch); caractere; unitate normalizată (coordonatele colțului stânga-jos sunt 0;0 iar ale colțului dreapta-sus sunt 1;1). În timp ce unitățile de măsură inch, centimetri și puncte tipografice sunt absolute, unitățile de măsură caracter și pixel sunt relative la tipul de font utilizat, respectiv la rezoluția monitorului.

- Controlul dimensiunii fontului utilizat pentru scrierea valorilor numerice de pe axele graficului se realizează cu instrucțiunea:

```
set(gca, 'FontSize', 11);
```

Această instrucțiune definește (set) proprietatea `FontSize` a fontului cu care se vor scrie valorile numerice ale axelor curente (`gca-get current axes`) ca având valoarea 11. Unitatea de măsură implicită este punctul tipografic (dot), însă se pot utiliza și alte unități de măsură: pixeli, centimetri, inch, unitate normalizată.

- Controlul dimensiunii fontului utilizat pentru scrierea textelor explicative ale celor două axe și titlului se realizează cu :

```
set(get(gca, 'XLabel'), 'FontSize', 11);
set(get(gca, 'YLabel'), 'FontSize', 11);
set(get(gca, 'Title'), 'FontSize', 11);
```

Aceste instrucțiuni definesc (set) proprietatea `FontSize` a fontului cu care se vor scrie textele explicative ale abscisei (`XLabel`), ordonatei (`YLabel`) și titlului (`Title`) pentru axele curente (`gca`) ca având valorile specificate (în acest caz, 11 dot).

- Există și alte proprietăți ale obiectelor grafice `figure`, `axes` și `line` care intervin în procesul de reprezentare grafică a funcțiilor. Listele tuturor proprietăților obiectelor grafice `figure`, `axes` și `line` pot fi consultate în [7, 8, 9].



### 5.3.2. Reprezentarea a două funcții în ferestre grafice diferite

Se consideră funcțiile:

$$f_1: [x_{min}; x_{max}] \rightarrow \mathbb{R}, y_1 = f_1(x)$$

$$f_2: [x_{min}; x_{max}] \rightarrow \mathbb{R}, y_2 = f_2(x)$$

Pentru reprezentarea grafică a celor două funcții  $y_1 = f_1(x)$  și  $y_2 = f_2(x)$  în două ferestre grafice diferite se utilizează instrucțiunile:

```
figure
plot(x,y1,'-k');grid on;
figure
plot(x,y2,'-k');grid off;
```

În acest caz, instrucțiunea `figure` se va utiliza de două ori, înaintea fiecărei instrucțiuni `plot`, astfel încât să se obțină reprezentarea celor două grafice în ferestre grafice diferite. Dacă se omite scrierea celei de-a doua instrucțiuni `figure`, atunci toate instrucțiunile `plot` vor reprezenta, în mod succesiv, curbele corespunzătoare în același obiect grafic axes din același obiect grafic `figure`, fără păstrarea graficelor anterioare, rezultând în final doar ultimul grafic.

#### Problema 5.2

Se consideră funcțiile:

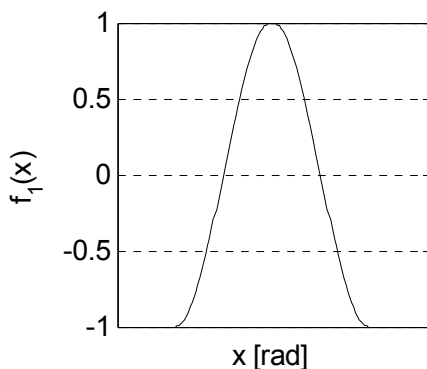
$$f_1: [-\pi; \pi] \rightarrow \mathbb{R}, y_1 = \cos(x)$$

$$f_2: [-\pi; \pi] \rightarrow \mathbb{R}, y_2 = \cos(x^2)$$

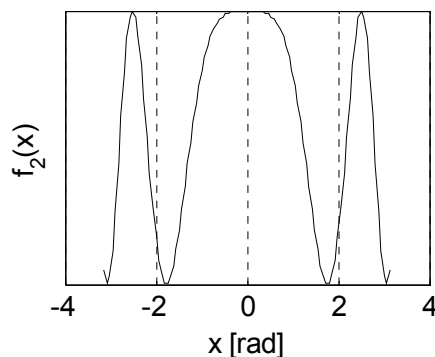
Să se reprezinte grafic cele două funcții în ferestre grafice diferite.

#### Rezolvare

Pe baza fișierului `script` utilizat pentru rezolvarea problemei 5.1, s-a elaborat un alt fișier, care în urma lansării în execuție a condus la reprezentarea graficelor în două ferestre grafice diferite, figura 5.2.



a) funcția  $y_1 = \cos(x)$



b) funcția  $y_2 = \cos(x^2)$

**Figura 5.2.** Grafice reprezentate în ferestre grafice diferite.

## Observații

- Pentru etichetarea celor două ordonate s-au utilizat instrucțiunile:

```
ylabel('f_1(x)');  
ylabel('f_2(x)');
```

Obținerea indicilor inferiori se realizează folosind caracterul `_`, iar a indicilor superiori folosind caracterul `^`. Dacă indicii inferiori, respectiv superiori sunt formați din mai multe caractere, acestea trebuie introduse între acolade, `{ }`.

- Pentru controlul independent al rețelelor de linii ajutătoare (`grid`) se folosesc instrucțiuni de tipul:

```
set(gca, 'XGrid', 'off');  
set(gca, 'YGrid', 'on')
```

Prima instrucțiune împiedică afișarea rețelei de linii ajutătoare ale abscisei (`XGrid`), în timp ce a doua instrucțiune permite afișarea rețelei de linii ajutătoare ale ordonatei (`YGrid`), figura 5.2, a).

- În mod implicit, numărul și localizarea liniilor ajutătoare de tip `grid` se determină după un algoritm implicit, în funcție de valorile de pe axele respective, precum și de dimensiunea fizică a ferestrei grafice. De exemplu, pentru obținerea reprezentării grafice din figura 5.2, b), s-au utilizat instrucțiunile:

```
set(gca, 'XGrid', 'on');  
set(gca, 'YGrid', 'off');  
set(gca, 'XTick', [-4:2:4]);  
set(gca, 'YTick', []);
```

### 5.3.3. Reprezentarea a două funcții în aceeași figură și în aceleași axe

Se consideră funcțiile:

$$f_1: [x_{min}, x_{max}] \rightarrow \mathbb{R}, y_1 = f_1(x)$$

$$f_2: [x_{min}, x_{max}] \rightarrow \mathbb{R}, y_2 = f_2(x)$$

Pentru reprezentarea grafică a celor două funcții  $y_1 = f_1(x)$  și  $y_2 = f_2(x)$  în același obiect grafic de tip `figure` și în același obiect grafic de tip `axes`, se poate utiliza una din variantele:

```
plot(x, y1, '-k', x, y2, '--k');
```

```
plot(x, y1, '-k'); hold on;  
plot(x, y2, '--k'); hold off;
```

Indiferent de numărul funcțiilor de reprezentat se recomandă a doua metodă. Pentru obținerea rezultatului final, instrucțiunea `plot` a fost aplicată, în acest caz, de două ori, odată pentru trasarea graficului funcției  $y_1 = f_1(x)$  și a două oară pentru obținerea graficului funcției  $y_2 = f_2(x)$ . Cele două instrucțiuni `plot` trebuie să afișeze rezultatele în aceeași fereastră grafică și în același obiect grafic de tip `axes`, fiind obligatorie intercalarea între cele două instrucțiuni `plot` a instrucțiunii:

```
hold on
```

După instrucțiunea `hold on`, toate instrucțiunile de reprezentare grafică (până la întâlnirea instrucțiunii `hold off`) vor afișa curbele respective în același obiect grafic de tip `figure` și în același obiect grafic de tip `axes` în care s-a reprezentat și prima curbă.

Formatarea independentă a celor două curbe ajută la identificarea precisă a acestora și este necesară ori de câte ori pe aceeași fereastră grafică se reprezintă mai multe curbe. Astfel, funcția  $y_1 = f_1(x)$  se reprezintă cu linie continuă de culoare neagră ('-k'), iar funcția  $y_2 = f_2(x)$  se reprezintă cu linie întreruptă de culoare neagră ('--k').

Identificarea clară a celor două curbe se realizează prin intermediul obiectului grafic de tip `legend`. Aplicarea adnotării de tip `legend` se face cu instrucțiunea:

```
legend('y_1=f_1(x)', 'y_2=f_2(x)')
```

în care parametrii instrucțiunii `legend` reprezintă specificațiile celor două curbe și trebuie introduși între apostrofuri și separași prin virgulă.

### Problema 5.3

Se consideră funcțiile:

$$f_1: [-\pi; \pi] \rightarrow \mathbb{R}, y_1 = |\sin(x)|$$

$$f_2: [-\pi; \pi] \rightarrow \mathbb{R}, y_2 = \sin(x^2)$$

Să se reprezinte grafic cele două funcții în aceeași fereastră grafică și în același obiect grafic de tip `axes`.

### Rezolvare

Rezolvarea problemei este realizată cu următorul fișier `script`:

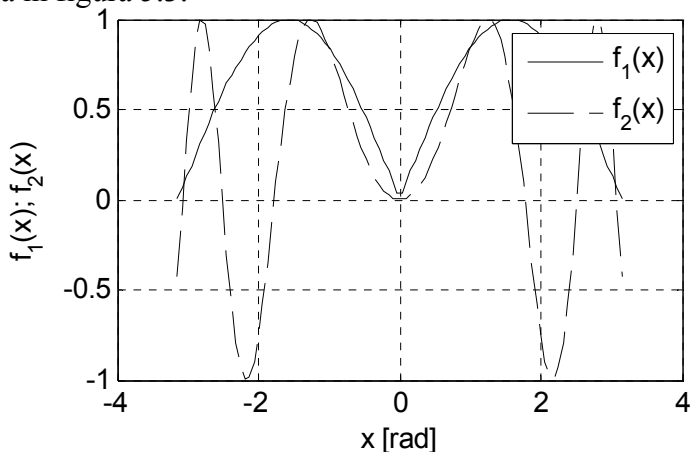
```
%% GRAFICE 2D (3)
clear all;close all;clc;
%% DATE INITIALE
xmin=-pi;xmax=pi;
x=linspace(xmin,xmax);
```

```

%% BLOC DE CALCUL
y1=abs(sin(x));
y2=sin(x.^2);
%% REPREZENTARE GRAFICA
figure
plot(x,y1,'-k');
hold on;
plot(x,y2,'--k');
set(gca,'XGrid','on');set(gca,'YGrid','on')
xlabel('x [rad]');ylabel('f_1(x); f_2(x)');
set(gcf,'Units','pixel');
set(gcf,'Position',[100 100 400 250]);
set(gca,'FontSize',11);
set(get(gca,'XLabel'),'FontSize',11);
set(get(gca,'YLabel'),'FontSize',11);
set(get(gca,'Title'),'FontSize',11);
legend('f_1(x) ','f_2(x) ');
hold off

```

Lansarea în execuție a fișierului conduce la reprezentarea grafică prezentată în figura 5.3.



**Figura 5.3.** Grafice reprezentate în aceeași fereastră grafică și în același obiect grafic de tip axes.

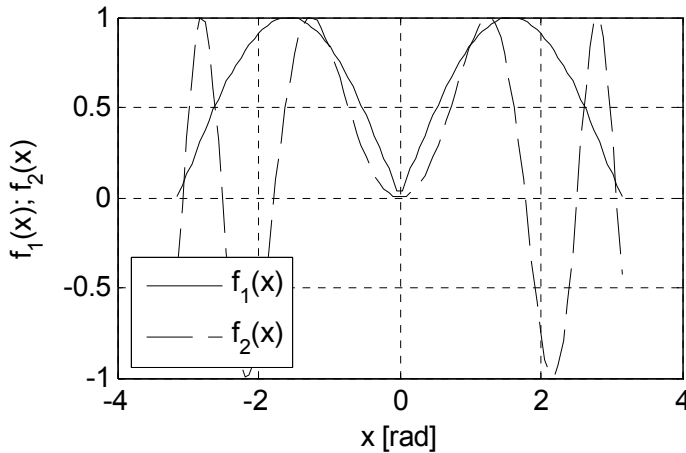
### Observații

- Identificarea clară a celor două curbe se realizează prin intermediul adnotării de tip legendă. Obiectul grafic de tip legend poate fi mutat pe spațiul figurii, folosind tehnica drag&drop, astfel încât, pe cât posibil, să nu acopere porțiuni semnificative ale curbelor.
- Controlul poziției obiectului grafic de tip legend pe spațiul figurii se poate face și direct, prin specificarea valorii parametrului 'Location' din cadrul instrucțiunii legend:

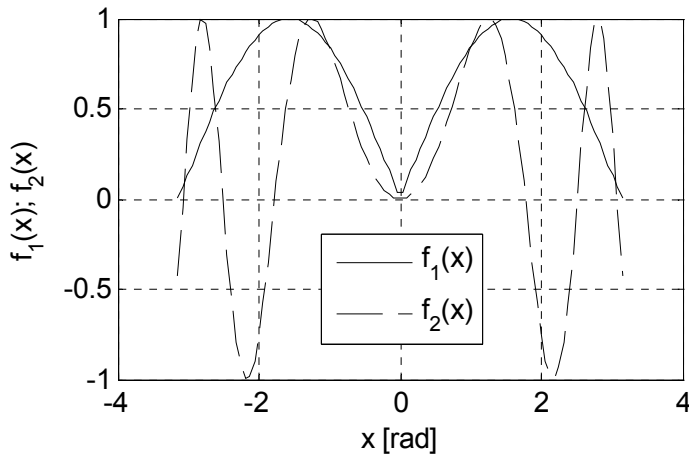
```
legend('f_1(x)', 'f_2(x)', 'Location', pozitie);
```

în care pozitie poate fi: North, South, East, West, NorthEast, NorthWest, SouthEast, SouthWest, NorthOutside, SouthOutside, EastOutside, West Outside, NorthEastOutside, NorthWestOutside, SouthEastOutside, SouthWestOutside, Best, Best Outside. Variantele Best și BestOutside plasează legenda în interiorul axelor, respectiv în exteriorul acestora, în locații optime, astfel încât suprapunerea legendei peste curba reprezentată să fie minimă.

- În figura 5.4, localizarea legendei s-a obținut cu parametrul 'Location' având valoarea SouthWest, iar în figura 5.5, localizarea legendei s-a obținut cu opțiunea Best.



**Figura 5.4.** Localizarea legendei cu opțiunea SouthWest.



**Figura 5.5.** Localizarea legendei cu opțiunea Best.

### 5.3.4. Reprezentarea a două funcții în aceeași figură și în axe diferite

Reprezentarea a două funcții în același obiect grafic `figure` dar în două obiecte grafice de tip `axes` diferite se realizează prin crearea unei structuri de obiecte grafice de tip `axes`, adică a mai multor axe dispuse în interiorul aceluiași obiect grafic de tip `figure`.

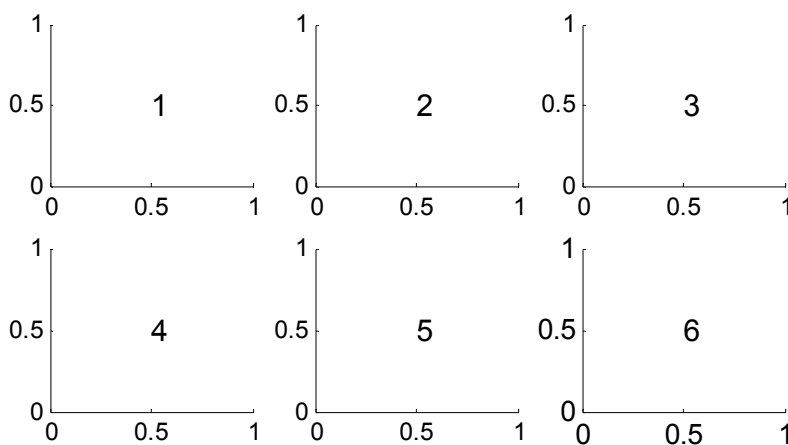
În acest scop se utilizează instrucțiunea:

```
subplot(m,n,p)
```

care realizează atât împărțirea ferestrei grafice curente într-o matrice cu  $m$  rânduri și  $n$  coloane de obiecte grafice de tip `axes`, cât și selectarea obiectului grafic `axes` cu identificatorul numeric  $p$  ca fiind cel curent.

În mod implicit, o structură de obiecte grafice de tip `axes` creată cu această instrucțiune va avea un număr de  $m \times n$  axe, rezultând astfel o structură regulată. Pot fi construite însă și structuri de obiecte grafice de tip `axes` neregulate.

Fiecare obiect grafic de tip `axes` dintr-o astfel de structură primește un identificator numeric unic, notat cu  $p$ , numerotarea făcându-se linie după linie începând cu colțul din stânga sus al structurii. De exemplu, în cazul unei structuri regulate cu dimensiunile  $2 \times 3$ , numerotarea și stabilirea identificatorilor numerici  $p$  se face conform figurii 5.6.



**Figura 5.6.** Structură regulată cu 6 axe pe o matrice de tip  $2 \times 3$ .

Pentru obținerea acestei structuri regulate de axe se utilizează instrucțiunile:

```
subplot(2,3,1);text(0.5,0.5,'1');  
subplot(2,3,2);text(0.5,0.5,'2');  
subplot(2,3,3);text(0.5,0.5,'3');
```

```
subplot(2,3,4);text(0.5,0.5,'4');
subplot(2,3,5);text(0.5,0.5,'5');
subplot(2,3,6);text(0.5,0.5,'6');
```

Scrierea identificatorului numeric  $p$  în interiorul fiecărei axe se realizează cu instrucțiunea `text` al cărei format general este:

```
text(x,y,'sir de caractere')
```

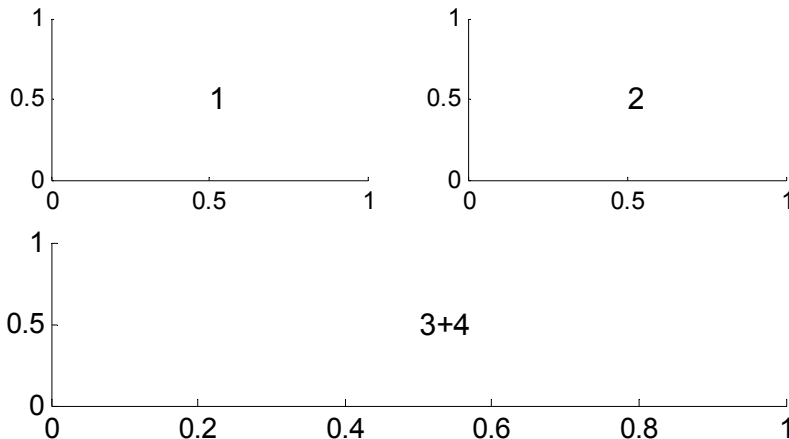
în care  $x$  și  $y$  sunt coordonatele punctului de început al textului, iar 'sir de caractere' este textul propriu-zis.

Ca și în cazul obiectelor grafice de tip `figure`, `axes` și `line` și în cazul obiectului grafic de tip `text` există o serie de parametri care pot fi controlați de către utilizator. Lista tuturor proprietăților obiectului grafic `text` poate fi consultată în [10].

Instrucțiunea `subplot` permite realizarea și a structurilor neregulate de axe. De exemplu, structura neregulată din figura 5.7 este formată din trei obiecte grafice de tip `axes`, din care două sunt plasate pe prima linie, iar cel de-al treilea pe cea de-a doua linie a structurii. Structura de axe are la bază o structură regulată cu dimensiunea  $2 \times 2$  (4 axe elementare), neregularitatea fiind determinată de concatenarea axelor elementare cu identificatorii numerici 3 și 4 și obținerea astfel, pe poziția acestor două axe elementare a unei singure axe.

Pentru obținerea acestei structuri neregulate de axe se utilizează instrucțiunile:

```
subplot(2,2,1); text(0.5,0.5,'1');
subplot(2,2,2); text(0.5,0.5,'2');
subplot(2,2,[3 4]); text(0.5,0.5,'3+4');
```

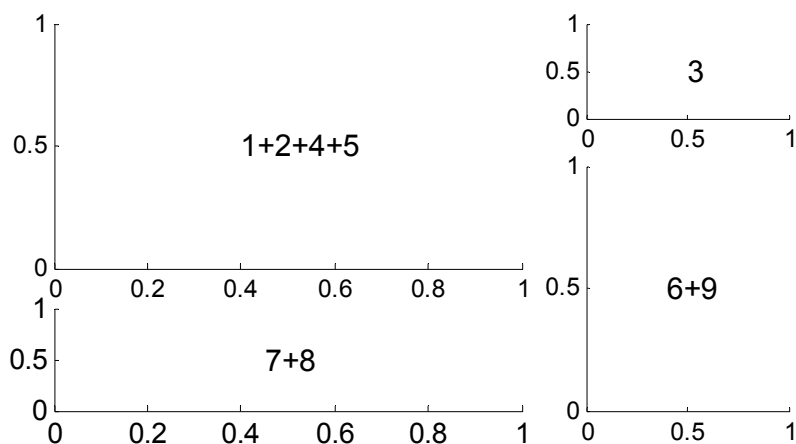


**Figura 5.7.** Structură neregulată cu 3 axe pe o matrice de tip  $2 \times 2$ .

În figura 5.8 este reprezentat un alt exemplu de structură neregulată de obiecte grafice de tip `axes`. În acest caz, structura de axe are la bază o structură regulată cu dimensiunea 3x3 (9 axe elementare), neregularitatea fiind determinată de mai multe procese de combinare a celor 9 axe elementare ale structurii regulate: concatenarea axelor elementare cu identificatorii numerici 1, 2, 4 și 5, concatenarea axelor elementare cu identificatorii 7 și 8, precum și concatenarea axelor elementare cu identificatorii 6 și 9.

Pentru obținerea acestei structuri neregulate de axe se utilizează instrucțiunile:

```
subplot(3,3,[1 2 4 5]);text(0.4,0.5,'1+2+4+5');
subplot(3,3,3);text(0.5,0.5,'3');
subplot(3,3,[6 9]);text(0.4,0.5,'6+9');
subplot(3,3,[7 8]);text(0.45,0.5,'7+8');
```



**Figura 5.8.** Structură neregulată cu 4 axe pe o matrice de tip 3x3.

Una din proprietățile obiectelor grafice de tip `axes` este `Visible` și se referă la capacitatea de a controla afișarea axelor, a etichetelor și a valorilor numerice de pe axe. Valorile acestei proprietăți sunt `on` (în mod implicit) și `off`. Pentru ascunderea obiectului grafic de tip `axes` trebuie modificată valoarea proprietății `Visible` cu ajutorul instrucțiunii:

```
set(gca,'Visible','off');
```

Această proprietate permite ascunderea axelor, însă nu poate controla obiectele grafice reprezentate în interiorul acestor axe. Prin urmare se poate introduce în spațiul unor axe invizibile orice obiect grafic de tip text sau de tip linie.



### 5.3.5. Controlul scalării și modului de vizualizare a axelor

Pentru obiectele grafice de tip `axes` controlul principalelor proprietăți referitoare la scalare și la modul de vizualizare se poate face cu instrucțiunea specifică `axis`. Principalele moduri de configurare pentru aplicarea instrucțiunii `axis` sunt, [11]:

- `axis auto`. Varianta implicită pentru care limitele celor două axe  $x$  și  $y$  se determină în mod automat, pe baza valorilor minime și maxime ale datelor corespunzătoare abscisei  $x$  și ordonatei  $y$ .
- `axis tight`. Limitele celor două axe sunt strict egale cu valorile minime și maxime ale datelor corespunzătoare abscisei și ordonatei.
- `axis off`. Ascunde obiectul grafic `axes`.
- `axis on`. Afișează obiectul grafic `axes`.
- `axis xy`. Originea sistemului de coordonate este amplasată în colțul din stânga-jos al obiectului grafic de tip `axes`, axa abscisei (axa  $x$ ) este orizontală având sensul crescător al valorilor de la stânga la dreapta, iar axa ordonatei (axa  $y$ ) este verticală având sensul crescător al valorilor de jos în sus.
- `axis ij`. Originea sistemului de coordonate este amplasată în colțul din stânga-sus al obiectului grafic de tip `axes`, axa ordonatei  $i$  este verticală având sensul crescător al valorilor de sus în jos, iar axa abscisei  $j$  este orizontală având sensul crescător al valorilor de la stânga la dreapta.
- `axis square`. Redimensionarea lungimii celor două axe  $x$  și  $y$  astfel încât suprafața obiectului grafic `axes` să aibă exact forma unui pătrat.
- `axis equal`. Redimensionarea lungimii axelor astfel încât unitatea de măsură în direcția abscisei trebuie să fie egală cu unitatea de măsură din direcția ordonatei (factorii de scalare ai celor două axe  $x$  și  $y$  trebuie să fie identici).
- `axis image`. Redimensionarea lungimii axelor astfel încât unitatea de măsură în direcția abscisei trebuie să fie egală cu unitatea de măsură din direcția ordonatei și în plus, limitele celor două axe sunt strict egale cu valorile minime și maxime ale datelor corespunzătoare abscisei, respectiv ordonatei.
- `axis([xmin xmax ymin ymax])` controlează limitele domeniului dreptunghiular care va fi afișat în interiorul obiectului grafic de tip `axes`. Indiferent de valorile numerice ale vectorilor corespunzători abscisei și ordonatei, această instrucțiune permite vizualizarea doar a unui subdomeniu dreptunghiular definit de colțul stânga-jos ( $xmin$   $ymin$ ) și colțul dreapta-sus ( $xmax$   $ymax$ ).

### Problema 5.4

Se consideră profilul aerodinamic de tip Gö364 definit prin coordonatele intradosului  $y^-(x)$  și extradosului  $y^+(x)$  conform tabelului de valori, [20]:

$x$	0,0	1,25	2,5	5,0	7,5	10	15	20	30	40	50	60	70	80	90	95	100
$y^-$	0,85	0,0	0,05	0,35	0,55	0,65	1,05	1,3	1,7	1,85	1,8	1,55	1,25	0,9	0,45	0,2	0,1
$y^+$	0,85	4,05	5,45	7,03	8,6	9,65	11	11,85	12,5	12,1	11,1	9,5	7,55	5,35	2,9	1,55	0,1

Să se reprezinte grafic profilul Gö364 utilizând pentru intrados  $y^-(x)$  linia întreruptă de culoare neagră și pentru extrados  $y^+(x)$  linia continuă de culoare neagră.

Să se verifice rezultatul aplicării principalilor parametri de configurare ai instrucțiunii `axis`.

### Rezolvare

Reprezentarea grafică a profilului Gö364 folosind opțiunea `axis auto` este prezentată în figura 5.9. Datorită existenței unor factori de scalare diferiți pentru cele două axe, reprezentarea grafică a profilului este foarte departe de realitate.

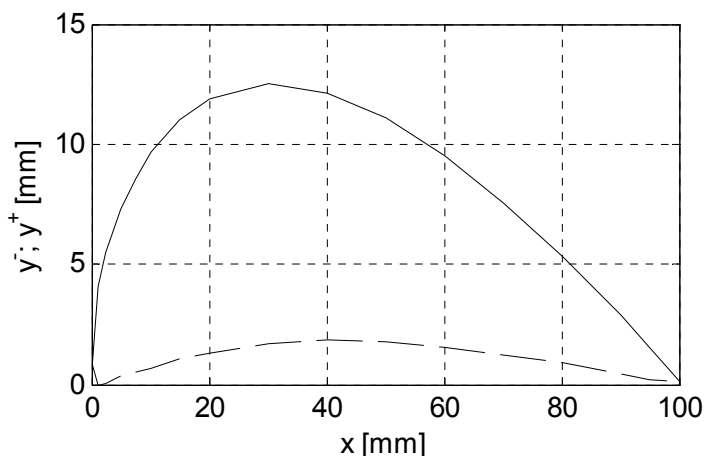
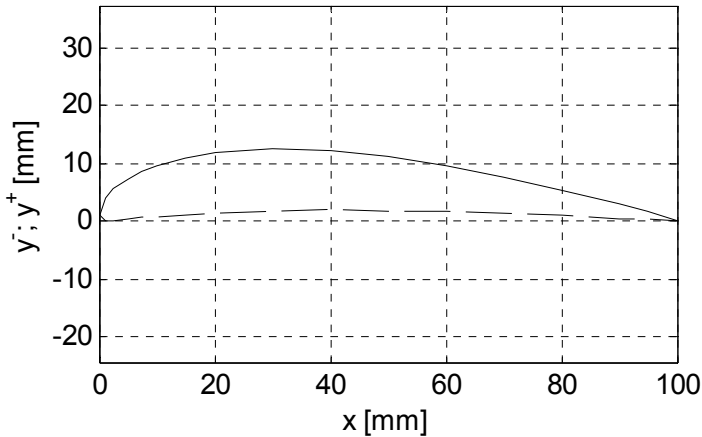


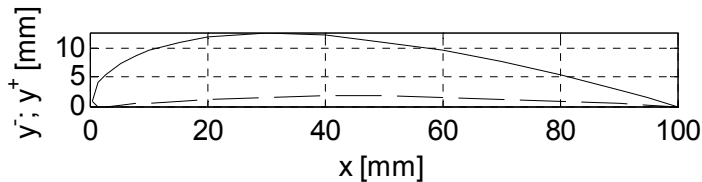
Figura 5.9. Profilul Gö364, varianta `axis auto`.

În cazul în care, la reprezentarea grafică a unei funcții oarecare  $f: [x_{min}, x_{max}] \rightarrow \mathbb{R}$ , definită prin  $y = f(x)$ , atât valorile  $x$ , cât și valorile  $y$  corespunzătoare, au aceeași unitate de măsură (de exemplu la reprezentarea grafică a unui profil aerodinamic, atât abscisa cât și ordonata se exprimă în milimetri), pentru realizarea unei reprezentări grafice corecte, trebuie utilizată instrucțiunea `axis equal`, figura 5.10.



**Figura 5.10.** Profilul Gö364, varianta `axis equal`.

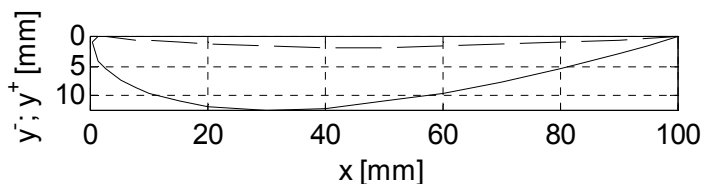
Pentru restrângerea domeniului celor două axe astfel încât limitele acestuia să fie strict egale cu valorile minime și maxime ale datelor corespunzătoare abscisei și ordonatei se utilizează instrucțiunea `axis tight`. Instrucțiunile `axis equal` și `axis tight` sunt echivalente cu instrucțiunea `axis image`, figura 5.11.



**Figura 5.11.** Profilul Gö364, varianta `axis image`.

Modificarea orientării ordonatei astfel încât sensul crescător al valorilor să fie de sus în jos se realizează cu ajutorul opțiunii `axis ij`, figura 5.12. Obținerea acestei reprezentări grafice s-a realizat datorită acțiunii combinate a două opțiuni ale instrucțiunii `axis`: `image` și `ij`. În astfel de cazuri se poate utiliza o singură instrucțiune de forma:

`axis image ij`



**Figura 5.12.** Profilul Gö364, varianta `axis image, axis ij`.

## 5.4. UTILIZAREA INSTRUCȚIUNII `plotyy`

Pentru realizarea obiectelor grafice de tip `axes` cu două ordonate, una plasată la stânga și cealaltă plasată la dreapta obiectului grafic `axes` se utilizează instrucțiunea `plotyy`. Această instrucțiune este deosebit de utilă în cazul în care se urmărește reprezentarea grafică în același obiect grafic de tip `axes` a două funcții  $f_1$  și  $f_2$  care au același domeniu de definiție:

$$f_1: [x_{min}; x_{max}] \rightarrow \mathbb{R}$$

$$f_2: [x_{min}; x_{max}] \rightarrow \mathbb{R}$$

dar care au valori  $y_1 = f_1(x)$  și  $y_2 = f_2(x)$  mult diferite între ele.

Astfel, pentru reprezentarea grafică a celor două funcții  $y_1 = f_1(x)$  și  $y_2 = f_2(x)$  în aceeași fereastră grafică și pe aceleași axe se poate utiliza una din următoarele două variante:

```
plotyy(x, y1, x, y2); grid;  
plotyy(x, y1, '-b', x, y2, '--r'); grid;
```

Se observă în cel de-al doilea caz, introducerea și a parametrilor de formatare pentru cele două curbe: prima curbă va fi reprezentată cu linie continuă de culoare albastră, în timp ce a doua curbă va fi reprezentată cu linie întreruptă de culoare roșie.

În cazul reprezentării grafice cu ajutorul instrucțiunii `plotyy`, datorită scalării independente a celor două ordonate (ordonata din stânga este pusă în corespondență cu domeniul de valori  $y_1 = f_1(x)$  în timp ce ordonata din dreapta este pusă în corespondență cu domeniul de valori  $y_2 = f_2(x)$ ), cele două curbe vor fi reprezentate grafic într-o manieră care va permite observarea clară a evoluției funcțiilor pe domeniul lor de definiție.

### Problema 5.5

Se consideră funcțiile:

$$f_1: [-3\sigma; 3\sigma] \rightarrow \mathbb{R}, f_1(x) = x^{0.4} \sin 10x$$

$$f_2: [-3\sigma; 3\sigma] \rightarrow \mathbb{R}, f_2(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

în care  $\sigma=2,5$  și  $\mu=0,5$ .

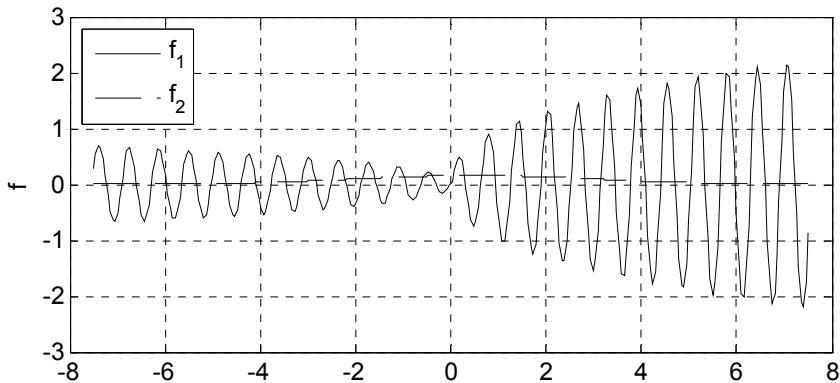
Să se reprezinte grafic cele două funcții în aceeași figură și în aceleași axe folosind, pe rând, instrucțiunile `plot` și `plotyy`.

### Rezolvare

Reprezentarea grafică a celor două funcții folosind instrucțiunea `plot` este prezentată în figura 5.13. Se observă că domeniul de variație al valorilor funcției  $f_1$  este  $\cong[-2; 2]$ , în timp ce valorile funcției  $f_2$  aparțin domeniului  $\cong[0; 0,2]$ . Graficul funcției  $f_1$  poate fi corect vizualizat, analizat

și interpretat, dar în cazul funcției  $f_2$  nu este posibilă vizualizarea corectă a graficului datorită diferenței semnificative între cele două domenii de variație ale valorilor celor două funcții. Principalele instrucțiuni necesare pentru obținerea reprezentării grafice din figura 5.13 sunt:

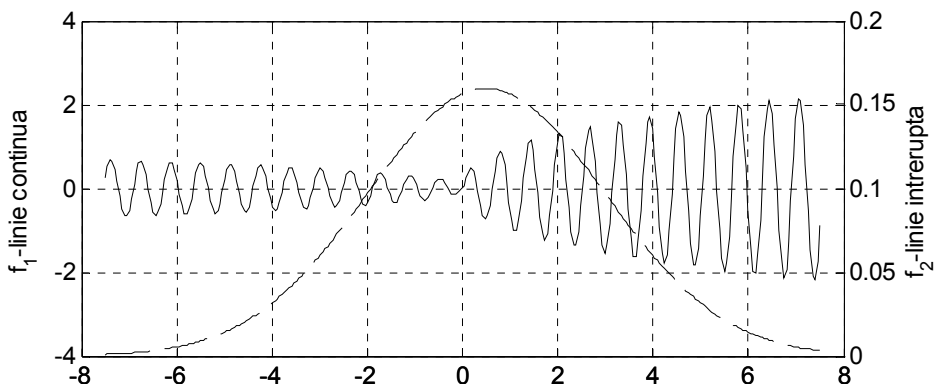
```
plot(x, f1, '-k', x, f2, '--k');
grid on; ylabel('f');
legend('f_1', 'f_2', 'Location', 'NorthWest');
```



**Figura 5.13.** Reprezentarea funcțiilor folosind instrucțiunea `plot`.

Datorită valorilor diferite ale celor două funcții  $f_1$  și  $f_2$ , doar utilizarea instrucțiunii `plotyy` permite reprezentarea corectă a celor două curbe în interiorul aceluiași obiect grafic de tip `axes` datorită scalării independente a celor două ordonate care se vor asocia în mod independent cu cele două funcții de reprezentat, figura 5.14.

Se observă că și graficul funcției  $f_2$  poate fi acum corect vizualizat datorită faptului că cele două funcții nu se mai raportează la o singură ordonată (ca în cazul instrucțiunii `plot`).



**Figura 5.14.** Reprezentarea funcțiilor folosind instrucțiunea `plotyy`.

În cazul aplicării instrucțiunii `plotyy`, formatarea independentă a celor două curbe, precum și etichetarea și formatarea independentă a celor două ordonate se realizează cu următoarele instrucțiuni:

```
%% GRAFICE 2D (5)
clear all;close all;clc;
%% DATE INITIALE
s=2.5;m=0.5;nx=300;
x=linspace(-3*s,3*s,nx);
f1=x.^0.4.*sin(10*x);
f2=1/(s*sqrt(2*pi))*exp(-1/2*((x-m)/s).^2);
%% GRAFIC PLOTYY
[AX H1 H2] = plotyy(x,f1,x,f2,'plot');
%formatarea independenta a celor doua curbe
set(H1,'LineStyle','-','Color','k');
set(H2,'LineStyle','--','Color','k');
%etichetarea independenta a celor doua axe y
set(get(AX(1),'Ylabel'),'String','f_1-linie
continua');
set(get(AX(2),'Ylabel'),'String','f_2-linie
intrerupta');
%formatarea independenta a celor doua axe y
set(AX(1),'YColor','k')
set(AX(2),'YColor','k')
%formatari comune
grid on;
set(gcf,'Position',[100 100 600 250]);
```

### Observații

- Domeniul de definiție al celor două funcții este un vector având un număr  $n_x=300$  de puncte de discretizare.
- Datorită faptului că variabila  $x$  este un vector, în relațiile de definiție ale celor două funcții intervin operatori cu punct ( $\cdot$  și  $\wedge$ ).
- AX reprezintă manipulatorul axelor curente, prima componentă AX(1) pentru ordonata din stânga, iar cea de-a doua componentă AX(2) pentru ordonata din dreapta. În acest mod este posibilă etichetarea și formatarea independentă a celor două ordonate.
- H1 și H2 reprezintă manipulatorii pentru cele două curbe asociate celor două funcții de reprezentat  $f_1$  și  $f_2$ . În acest mod este posibilă formatarea independentă a celor două curbe.
- Există și o serie de formatari comune, care în acest caz se referă la rețeaua de linii ajutătoare și la poziția și dimensiunea ferestrei grafice în care se reprezintă cele două funcții.

## 5.5. GRAFICE ÎN COORDONATE LOGARITMICE

Pentru reprezentarea grafică a funcțiilor utilizând coordonate logaritmice (logaritm în baza 10) se pot utiliza următoarele instrucțiuni:

- `loglog`. Permite realizarea graficelor 2D cu ambele axe trasate în coordonate logaritmice.
- `semilogx`. Permite realizarea graficelor 2D doar cu abscisa trasată în coordonate logaritmice.
- `semilogy`. Permite realizarea graficelor 2D doar cu ordonata trasată în coordonate logaritmice.

### Problema 5.6

Se consideră funcția:

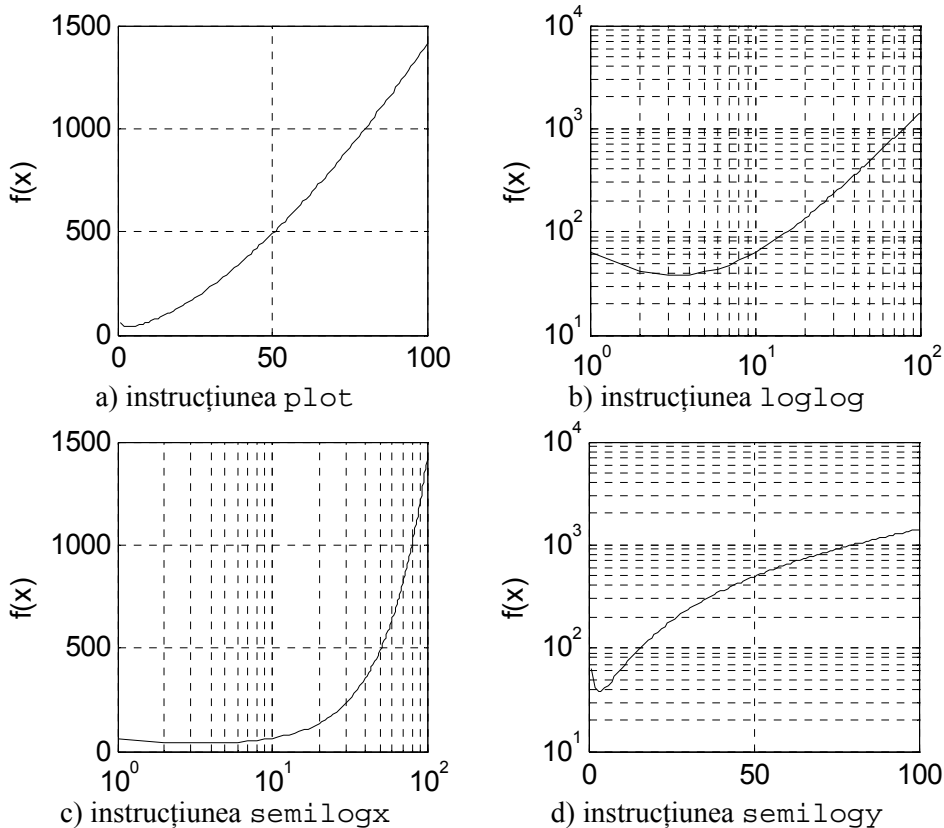
$$f: [1; 100] \rightarrow \mathbb{R}, f(x) = e^{\frac{\pi \cdot x + 1}{x}} + x^{\frac{\pi}{2}}$$

Să se reprezinte grafic funcția  $f(x)$  folosind, pe rând, instrucțiunile `plot`, `loglog`, `semilogx` și `semilogy`.

### Rezolvare

Reprezentarea grafică comparativă a funcției  $f(x)$  folosind toate cele patru instrucțiuni grafice (`plot`, `loglog`, `semilogx` și `semilogy`) este prezentată în figura 5.15. Principalele instrucțiuni necesare pentru obținerea reprezentărilor grafice sunt:

```
%% GRAFICE 2D (6)
clear all;close all;clc;
%% DATE INITIALE
x=linspace(1,100);
f=exp((pi*x+1)./x)+x.^(pi/2);
%% GRAFIC plot
figure
plot(x,f,'-k');
grid on;ylabel('f(x)');
%% GRAFIC loglog
figure
loglog(x,f,'-k');
grid on;ylabel('f(x)');
%% GRAFIC semilogx
figure
semilogx(x,f,'-k');
grid on;ylabel('f(x)');
%% GRAFIC semilogy
figure
semilogy(x,f,'-k');
grid on;ylabel('f(x)');
```



**Figura 5.15.** Reprezentarea grafică comparativă

### Observații

- Instrucțiunile sunt grupate în mai multe celule de calcul: celula de titlu, celula datelor inițiale, precum și câte o celulă pentru fiecare tip de reprezentare grafică. Celula de titlu conține și instrucțiunile `clear all`, `close all` și `clc`. Celula datelor inițiale conține generarea vectorului  $x$  al domeniului de definiție, precum și calculul valorilor funcției  $f(x)$ .
- Indiferent de instrucțiunea grafică utilizată, toate curbele obținute au aceiași parametri de formatare: linie continuă de culoare neagră.
- Se observă scalarea corespunzătoare a abscisei și ordonatei definite în coordonate logaritmice, precum și grid-ul specific logaritmice.
- Chiar dacă în toate cele patru cazuri s-a reprezentat aceeași funcție, datorită scalării diferite a celor două axe (scalare liniară, scalare logaritmice), aspectul reprezentărilor grafice obținute este diferit.
- Există situații în care graficul în coordonate logaritmice permite observarea mai clară a variației, pe anumite domenii, a unei funcții.



## 5.6. GRAFICE ÎN TREPTE

Se consideră funcția:  $f: [x_{min}, x_{max}] \rightarrow \mathbb{R}$ , definită prin  $y = f(x)$ .

Pentru reprezentarea grafică a funcției  $y = f(x)$  folosind metoda graficului în trepte se utilizează instrucțiunea:

```
stairs(x,y)
```

Modul de utilizare al acestei instrucțiuni, prezentarea variantelor de utilizare și a diferiților parametri de configurare, precum și diferite exemple se găsesc în [12].

### Problema 5.7

Se consideră funcția:

$$f: [0; \pi] \rightarrow \mathbb{R}, f(x) = e^x \cdot \sin x$$

Să se reprezinte grafic funcția  $f(x)$ , comparativ, folosind instrucțiunile `plot` și `stairs`.

### Rezolvare

Reprezentarea grafică comparativă a funcției  $f(x)$  folosind cele două instrucțiuni grafice `plot` și `stairs` este prezentată în figura 5.16. Principalele instrucțiuni necesare pentru obținerea reprezentărilor grafice sunt:

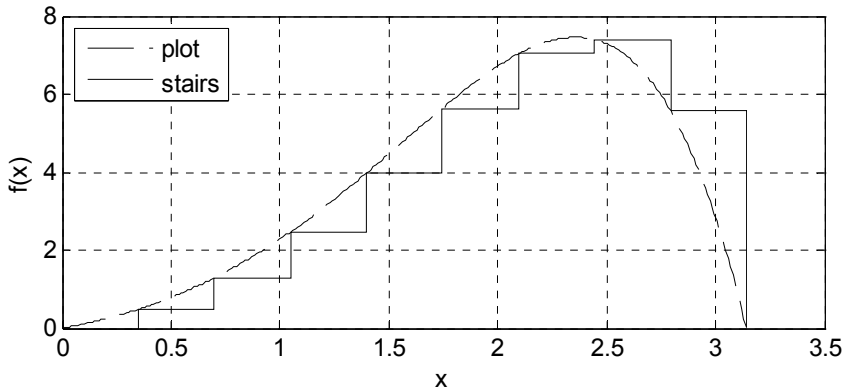
```
%% GRAFICE 2D (7)
clear all;close all;clc;
%% DATE INITIALE
xmin=0;xmax=pi;np=300;ns=10;
xp=linspace(xmin,xmax,np);
xs=linspace(xmin,xmax,ns);
f=@(x)exp(x).*sin(x);
%% GRAFIC plot, stairs
figure
plot(xp,f(xp),'k');hold on;
stairs(xs,f(xs),'-k');
grid on;xlabel('x');ylabel('f(x)');
legend('plot','stairs','Location','NorthWest');
hold off;
```

### Observații

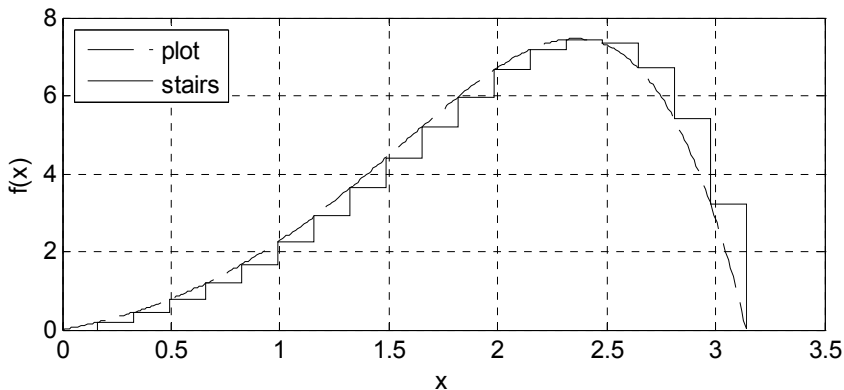
- Numărul treptelor este determinat de numărul punctelor de discretizare ale domeniului de definiție al funcției. Se recomandă generarea unei discretizări fine a domeniului de definiție al funcției pentru utilizarea instrucțiunii `plot` și a unei discretizări cu un număr mai redus de puncte pentru cazul instrucțiunii `stairs`.

Pentru reprezentarea cu instrucțiunea `plot` domeniul de definiție al funcției a fost împărțit în  $n_p=300$  de puncte, în timp ce pentru instrucțiunea `stairs` s-au utilizat  $n_s=10$  puncte pentru figura 5.16, a) și  $n_s=20$  de puncte pentru reprezentarea din figura 5.16, b).

- Expresia matematică  $f(x)$  a fost definită o singură dată sub forma unei funcții de tip anonymous. Apelarea funcției și calcularea valorilor sale pentru cele două discretizări  $x_p$  și  $x_s$  ale domeniului de definiție s-a realizat chiar în corpul instrucțiunilor de reprezentare grafică `plot` și `stairs`.
- Legenda are rolul de a permite identificarea clară a celor două curbe, în conformitate cu parametrii de formatare utilizați pentru cele două instrucțiuni de reprezentare grafică: ambele curbe au culoarea neagră, doar că s-au utilizat două tipuri diferite de linie: linie continuă pentru instrucțiunea `stairs` și linie întreruptă pentru instrucțiunea `plot`.



a) 10 trepte



b) 20 trepte

**Figura 5.16.** Reprezentarea grafică comparativă folosind instrucțiunile `plot` și `stairs`.

## 5.7. GRAFICE CU BARE

Se consideră funcția:  $f: [x_{min}, x_{max}] \rightarrow \mathbb{R}$ , definită prin  $y = f(x)$ .

Pentru reprezentarea grafică a funcției  $y = f(x)$  folosind metoda graficelor cu bare 2D sau 3D se utilizează instrucțiunile:

`bar(x, y)`, pentru bare verticale 2D;

`barh(x, y)`, pentru bare horizontale 2D;

`bar3(x, y)`, pentru bare verticale 3D;

`bar3h(x, y)`, pentru bare horizontale 3D;

În cazul reprezentării grafice cu bare a două sau mai multe funcții care au același domeniu de definiție, atunci trebuie definită o matrice având pe fiecare coloană valorile funcțiilor respective. În acest caz, opțiunile de reprezentare ale comenzilor de reprezentare a graficelor cu bare sunt `group` și `stack`.

### Problema 5.8

Se consideră funcțiile:

$$f: [0; 1] \rightarrow \mathbb{R}, f(x) = e^x$$

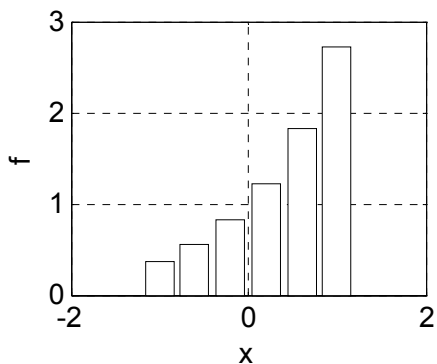
$$g: [0; 1] \rightarrow \mathbb{R}, g(x) = \frac{2}{e^{x+1}}.$$

Să se reprezinte cele două funcții folosind diferite variante ale instrucțiunilor de reprezentare grafică cu bare pentru o discretizare cu 6 puncte a domeniului de definiție al funcțiilor.

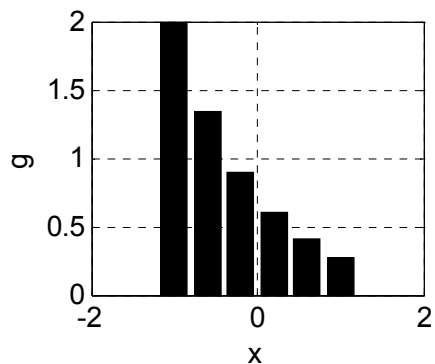
### Rezolvare

În figura 5.17 se prezintă graficele cu bare ale celor două funcții reprezentate în axe diferite: bare verticale 2D pentru cazurile din figurile 5.17, a) și b); bare horizontale 2D pentru cazurile din figurile 5.17, c) și d); bare verticale 3D pentru cazurile din figurile 5.15, e) și f). În toate figurile, graficul funcției  $f(x)$  are culoarea albă, iar graficul funcției  $g(x)$  are culoarea neagră. Instrucțiunile care au generat reprezentările grafice din figura 5.17, a) și b) sunt:

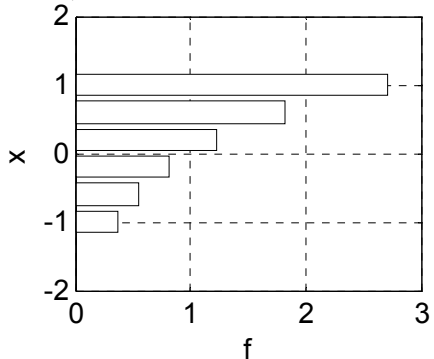
```
%% GRAFICE 2D (8)
clear all;close all;clc;
%% DATE INITIALE
xmin=-1;xmax=1;nx=6;x=linspace(xmin,xmax,nx);
f=@(x) exp(x);g=@(x) 2./exp(x+1);Y=[f(x);g(x)'];
%% GRAFIC 1
figure
bar(x,f(x),'w');xlabel('x');ylabel('f');grid on;
figure
bar(x,g(x),'k');xlabel('x');ylabel('g');grid on;
```



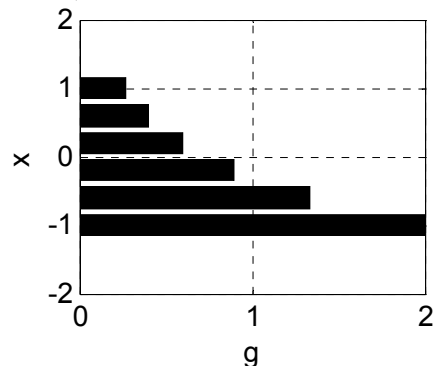
a) `bar(x, f(x), 'w');`



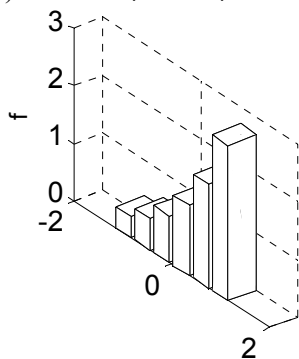
b) `bar(x, g(x), 'k');`



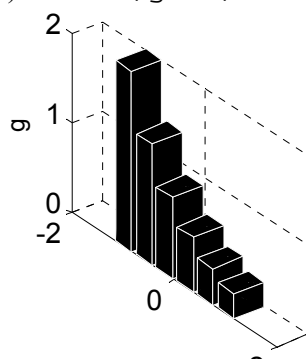
c) `barh(x, f(x), 'w');`



d) `barh(x, g(x), 'k');`



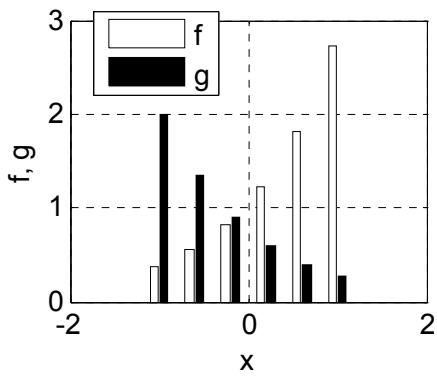
e) `bar3(x, f(x), 'w');`



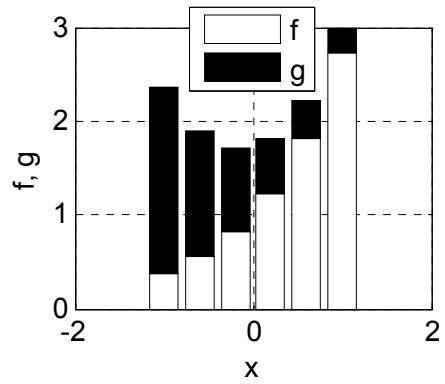
f) `bar3(x, g(x), 'k');`

**Figura 5.17.** Reprezentarea grafică cu bare în axe diferite.

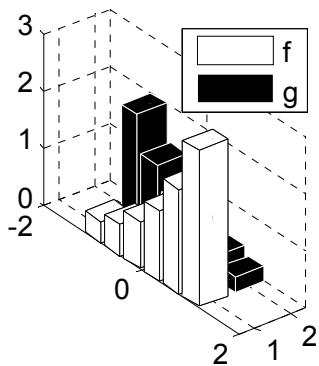
În figura 5.18 se prezintă graficele cu bare ale celor două funcții reprezentate în aceleași axe: bare verticale 2D cu opțiunea `group` în figura 5.18, a); bare verticale 2D cu opțiunea `stack` în figura 5.18, b); bare verticale 3D cu opțiunea implicită în figura 5.18, c); bare verticale orizontale 3D cu opțiunea `stack` în figura 5.18, d); bare orizontale 3D cu opțiunea implicită în figura 5.18, e) și bare orizontale 3D cu opțiunea `stack` în figura 5.18, f).



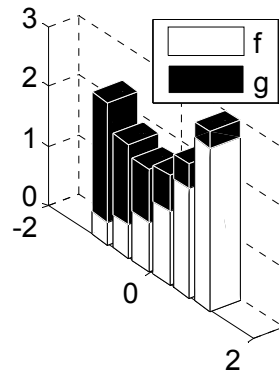
a) `bar(x,Y, 'group')` ;



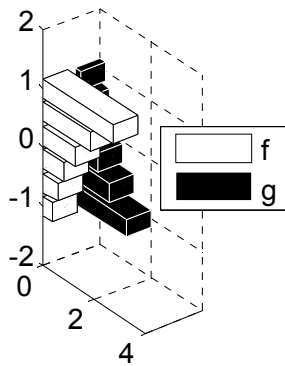
b) `bar(x,Y, 'stack')` ;



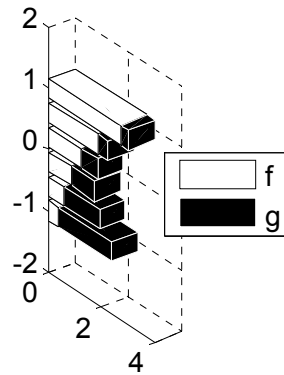
c) `bar3(x,Y)` ;



d) `bar3(x,Y, 'stack')` ;



e) `bar3h(x,Y)` ;



f) `bar3h(x,Y, 'stack')` ;

**Figura 5.18.** Reprezentarea grafică cu bare în aceleași axe.

## 5.8. REPREZENTAREA GRAFICĂ A HISTOGRAMEI

Se consideră un eșantion de volum  $n$  format din următoarele valori ordonate crescător, [21]:

$$x_1, x_2, \dots, x_{n-1}, x_n$$
$$x_{min} = x_1 \text{ și } x_{max} = x_n$$

Amplitudinea sondajului, definită prin  $A = x_{max} - x_{min}$ , se împarte într-un număr  $k$  de intervale egale, numite clase. Amplitudinea unei clase se calculează cu relația  $a = A/n$ .

Frecvența absolută a claselor  $i = 1 \dots k$  se notează cu  $n_i$ ,  $i = 1 \dots k$  și reprezintă numărul de rezultate ale eșantionului care se găsesc în fiecare interval  $i = 1 \dots k$ .

Reprezentarea grafică a frecvențelor absolute ( $n_i$ ) în funcție de intervalele de valori corespunzătoare ( $i = 1 \dots k$ ) poartă numele de histograma repartiției. Histograma repartiției se mai poate exprima și în funcție de frecvențele relative definite prin relația:  $f_i = n_i/n$ . Obținerea histogramei frecvențelor absolute se face prin utilizarea instrucțiunilor:

- Pentru coordonate carteziene:

`hist(x, k)`

în care  $x$  reprezintă vectorul valorilor eșantionului de analizat, iar  $k$  reprezintă numărul de clase în care a fost împărțită amplitudinea sondajului. Dacă se omite scrierea parametrul  $k$ , atunci se consideră în mod implicit că numărul de clase este  $k=10$ .

- Pentru coordonate polare:

`rose(t, k)`

în care  $t$  [rad] reprezintă vectorul valorilor unghiulare ale eșantionului de analizat, iar  $k$  reprezintă numărul de clase în care a fost împărțită amplitudinea sondajului. Dacă se omite scrierea parametrului  $k$ , atunci se consideră în mod implicit că numărul de clase este  $k=20$ .

### Problema 5.9

Se consideră un eșantion  $x_1, x_2, \dots, x_{n-1}, x_n$ , de volum  $n$ , format din valori aleatoare normal distribuite, dintr-o populație având ca parametri media aritmetică  $\bar{x}$  și abaterea medie pătratică  $s$ .

Se cere:

- Să se reprezinte grafic histograma în coordonate carteziene pentru  $n=100$ ,  $\bar{x}=12,65$ ,  $s=0,75$ ,  $k=\{10; 5\}$ .
- Să se reprezinte grafic histograma în coordonate polare pentru  $n=50$ ,  $\bar{t}=\pi$ ,  $s=\pi$ ,  $k=\{20; 10\}$ .

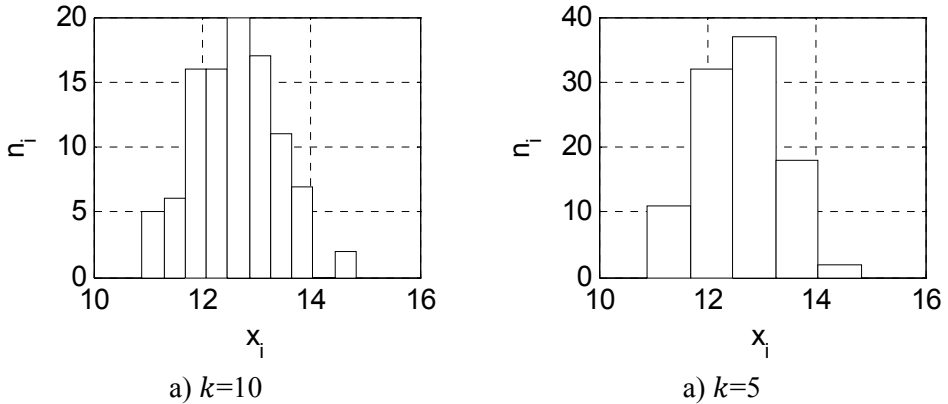
## Rezolvare

Generarea unui sondaj aleator uniform distribuit având media aritmetică  $x_m$  și abaterea medie pătratică  $s$  se face cu instrucțiunea:

$$x = x_m + s * \text{randn}(n, 1)$$

în care parametrii instrucțiunii `randn` determină dimensiunea  $n \times 1$  a structurii sondajului aleator.

Reprezentarea grafică a histogrammei în coordonate carteziene pentru cele două cazuri ( $k=10$  și  $k=5$ ) este prezentată în figura 5.19.



**Figura 5.19.** Histograma în coordonate carteziene.

Principalele instrucțiuni necesare pentru obținerea histogramelor în coordonate carteziene sunt:

```
%% GRAFICE 2D (9)
clear all;close all;clc;
%% DATE INITIALE
% Volumul sondajului
n=100;
% Numarul de clase
k=[10 5];
% Media aritmetica
xm=12.65;
% Abaterea medie patratica
s=0.75;
%% GENERAREA SONDAJULUI ALEATOR
x=xm+s*randn(n,1);
%% REPREZENTAREA HISTOGRAMEI CARTEZIENE
% k=10
figure
hist(x,k(1));grid;xlabel('x_i');ylabel('n_i');
h=findobj(gca,'Type','patch');
```

```

set(h, 'FaceColor', 'w', 'EdgeColor', 'k')
% k=5
figure
hist(x,k(2));grid;
xlabel('x_i');ylabel('n_i');
h=findobj(gca, 'Type', 'patch');
set(h, 'FaceColor', 'w', 'EdgeColor', 'k')

```

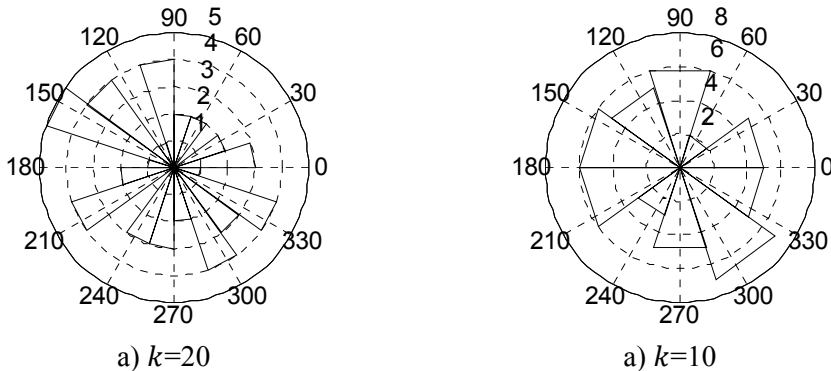
Pentru realizarea histogramelor în coordonate carteziene se utilizează elemente grafice de tip `patch`, [13]. Modificarea culorii histogramelor în coordonate carteziene se face prin identificarea acestor elemente grafice folosind instrucțiunea:

```
h=findobj(gca, 'Type', 'patch');
```

și apoi prin modificarea culorii acestor elemente `FaceColor`, respectiv a culorii muchiilor `EdgeColor`, cu ajutorul instrucțiunii:

```
set(h, 'FaceColor', 'w', 'EdgeColor', 'k')
```

Reprezentarea grafică a histogramei în coordonate polare pentru cele două cazuri ( $k=20$  și  $k=10$ ) este prezentată în figura 5.20.



**Figura 5.20.** Histograma în coordonate polare.

Pentru realizarea histogramelor în coordonate polare se utilizează elemente grafice de tip `line`, [9]. Modificarea culorii histogramelor în coordonate polare se face prin identificarea acestor elemente grafice folosind instrucțiunea:

```
h=findobj(gca, 'Type', 'line');
```

și apoi prin modificarea culorii acestor elemente `Color`, cu ajutorul instrucțiunii:

```
set(h, 'Color', 'k')
```



## 5.9. REPREZENTAREA GRAFICĂ A ERORILOR

Se consideră că în urma efectuării unui proces de măsurare, rezultatele obținute se pot scrie sub forma, [21]:

$$y = \bar{x} \pm e$$

în care  $\bar{x}$  reprezintă valorile medii ale rezultatelor experimentale obținute în fiecare punct de măsură  $x$ , iar  $e$  reprezintă intervalul de eroare corespunzător fiecărui punct de măsurare. Valorile  $\bar{x}$  și  $e$  rezultă în urma repetării procesului de măsurare în condiții practic identice de un anumit număr de ori pentru fiecare punct de măsură  $x$  și prin efectuare apoi a analizei statistice a datelor experimentale. Rezultatele obținute în urma analizei statistice a datelor experimentale sunt valabile în condițiile impunerii unei anumite valori a nivelului de încredere  $P$  [%] (în general 95% pentru măsurări obișnuite și 99% sau mai mult pentru măsurări de precizie).

Reprezentarea grafică a intervalelor de eroare se face în mod diferit după cum intervalele de eroare sunt sau nu simetrice, [14]:

- În cazul intervalelor de eroare simetrice  $e$  asociate setului de date  $\{x, \bar{x}\}$ , se utilizează instrucțiunea:

```
errorbar (x, xm, e)
```

- În cazul intervalelor de eroare asimetrice  $e_I$  (eroarea inferioară) și  $e_S$  (eroarea superioară) asociate setului de date  $\{x, \bar{x}\}$ , se utilizează instrucțiunea:

```
errorbar (x, xm, eI, eS)
```

Instrucțiunea `errorbar` realizează pentru fiecare punct al setului de date  $\{x, \bar{x}\}$ , reprezentarea grafică a unor segmente verticale proporționale cu intervalele de eroare asociate, având lungimea  $e + e$  în cazul erorilor simetrice și  $e_I + e_S$  în cazul erorilor asimetrice.

### Problema 5.10

Se consideră că în urma efectuării unui proces de măsurare oarecare, rezultatele experimentale au fost prelucrate din punct de vedere statistic, rezultând următorul tabel de valori finale:

$x$	3	6	9	12	15	18	21
$\bar{x}$	3,25	7,16	10,95	14,1	16,2	17,05	16,2
$e$	2,5	2,2	1,5	1,0	1,8	1,75	2,75

Să se reprezinte intervalele de eroare simetrice  $e$  asociate setului de date  $\{x, \bar{x}\}$ .

## Rezolvare

În figura 5.21 se prezintă graficul erorilor corespunzător setului de date experimentale.

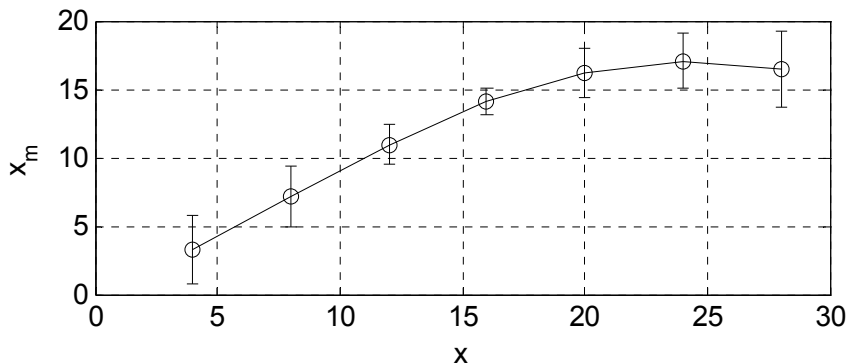


Figura 5.21. Reprezentarea grafică a erorilor.

Principalele instrucțiuni utilizate pentru obținerea reprezentării grafice a erorilor sunt:

```
%% GRAFICE 2D (10)
clear all;close all;clc;
%% DATE INITIALE
% Punctele de masurare
x=[4 8 12 16 20 24 28];
% Valorile medii
xm=[3.25 7.16 10.95 14.1 16.2 17.05 16.5];
% Intervalele de eroare
e=[2.5 2.2 1.5 1 1.8 2 2.75];
%% GRAFICUL ERORILOR
figure
errorbar(x,xm,e,'-ok');grid;
xlabel('x');ylabel('x_m');
```

## Observații

- Fișierul de tip script conține trei celule: celula cu instrucțiunile `clear all`, `close all` și `clc`, celula datelor inițiale în care s-au introdus valorile numerice ale celor trei variabile vectoriale (punctele de măsurare  $x$ , valorile medii  $\bar{x}$ , intervalele de eroare simetrice  $e$ ) și celula pentru reprezentarea grafică.
- Formatarea graficului se realizează direct în structura instrucțiunii `errorbar` prin folosirea parametrilor de formatare `'-ok'` care vor determina obținerea unei linii continue de culoare neagră. Markerele utilizate în acest caz pentru reprezentarea punctelor setului de date  $\{x, \bar{x}\}$  sunt de tip `o`.

## 5.10. GRAFICE DE TIP area

Pentru reprezentarea grafică de tip area se utilizează instrucțiunea:

```
area (y)
```

în care  $y$  reprezintă un vector de date. Suprafața dintre curba obținută și axa absciselor va fi colorată. În cazul în care  $y$  este o matrice se reprezintă fiecare coloană a matricei sub forma unor curbe elementare suprapuse, colorându-se în mod diferit suprafețele dintre două curbe elementare.

În cazul instrucțiunii:

```
area (x, y)
```

rezultatul este identic cu cel al utilizării instrucțiunii `plot`, cu deosebirea că se colorează suprafața de sub curbă.

### Problema 5.11

Să se reprezinte sub forma unui grafic de tip area seturile de date  $y_1$  și  $y_2$  în funcție de valorile  $x$  specificate în tabelul:

$x$	1	2	3	4	5
$y_1$	3	6	5	3	2
$y_2$	1	3	2,5	2	1

### Rezolvare

Reprezentarea grafică de tip area este prezentată în figura 5.22.

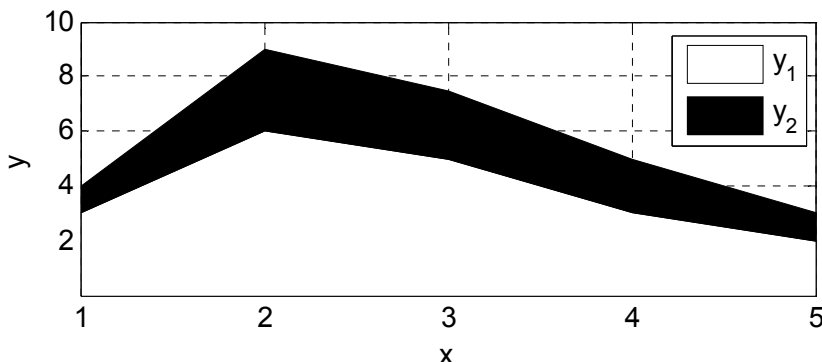


Figura 5.22. Reprezentarea grafică de tip area.

Principalele instrucțiuni utilizate pentru obținerea reprezentării grafice de tip area sunt:

```
%% GRAFICE 2D (11)  
clear all;close all;clc;  
%% DATE INITIALE
```

```

x=1:5;
Y=[3 6 5 3 2;1 3 2.5 2 1]';
%% REPREZENTARE GRAFICA
figure
h=area(x,Y);
grid on;
xlabel('x');ylabel('y');
legend('y_1','y_2');
set(h(1),'FaceColor','w');
set(h(2),'FaceColor','k');
set(gca,'XTick',x);
set(gca,'YTick',2:2:10);

```

### Observații

- Fișierul de tip script conține trei celule: celula cu instrucțiunile `clear all`, `close all` și `clc`, celula datelor inițiale și celula pentru reprezentarea grafică.
- Datele inițiale sunt reprezentate de valorile numerice ale variabilei vectoriale  $x$  și ale variabilei matriceale  $Y$ . Matricea  $Y$  are două coloane, dintre care prima coloană conține setul de date  $y_1$ , iar cea de-a doua coloană conține setul de date  $y_2$ .
- Instrucțiunea `h=area(x,Y)` realizează mai întâi reprezentarea grafică a primului set de date  $y_1$  în funcție de  $x$ . Reprezentarea celui de-al doilea set de date se realizează prin adăugarea la ordonatele  $y_1$  a valorile  $y_2$ , astfel încât ordonatele celei de-a doua curbe au valorile  $y_1+y_2$ . Primul set de date consideră ca element de referință axa absciselor, în timp ce fiecare nou set de date consideră ca element de referință curba anterioară.
- Adnotarea de tip legendă `legend('y_1','y_2')`, ajută la asocierea corectă a celor două curbe cu cele două seturi de date.
- Culorile suprafețelor dintre curbe se stabilesc automat la executarea instrucțiunii folosindu-se culorile implicite. Modificarea acestor culori se poate realiza, în mod separat, pentru fiecare set de date. Instrucțiunea `h=area(x,Y)` atribuie toate proprietățile reprezentării grafice de tip `area` unui identificator notat cu  $h$ , [15]. Numărul de elemente ale identificatorului  $h$  este egal cu numărul coloanelor matricei  $Y$ , adică cu numărul seturilor de date de reprezentat. În acest caz, matricea  $Y$  având două coloane, identificatorul  $h$  va avea două elemente  $h(1)$  pentru prima curbă și  $h(2)$  pentru cea de-a doua curbă. Se atribuie proprietății `'FaceColor'` culoarea `'w'` pentru prima curbă  $h(1)$  și culoarea `'k'` pentru cea de-a doua curbă  $h(2)$ .

### 5.11. GRAFICE DE TIP `pie`

Pentru reprezentarea grafică a unui set de date în care fiecare element reprezintă un procent dintr-un întreg, se utilizează instrucțiunea:

```
pie(y)
```

în care  $y$  reprezintă un vector având  $n$  date care are proprietatea că suma elementelor reprezintă un întreg (de exemplu, 100%), astfel încât fiecare element al vectorului de date  $y$  este pus în corespondență cu valoarea sa procentuală.

În cazul instrucțiunii:

```
pie(y,explode)
```

în care `explode` este un vector având aceeași dimensiune ca vectorul de date  $y$ , dar având doar elemente de 0 și 1, rezultatul este identic cu cel al utilizării instrucțiunii `pie(y)`, cu deosebirea că toate sectoarele cărora le corespund valorile 1 vor fi separate din suprafața circulară a figurii (sectoare explodate).

În cazul instrucțiunilor:

```
pie3(y)
```

```
pie3(y,explode)
```

rezultatul este identic cu cel al utilizării instrucțiunilor `pie(y)` și `pie(y,explode)` cu deosebirea că toate sectoarele vor fi reprezentate în vedere tridimensională.

#### Problema 5.12

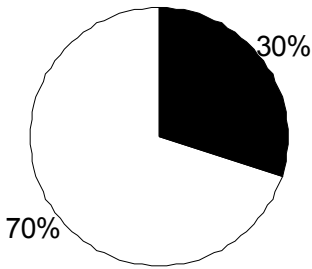
Se consideră următorul set de valori:

	Sectoare		
	1	2	$\sum y.$
$y$	7	3	10
$y[\%]$	70%	30%	100%

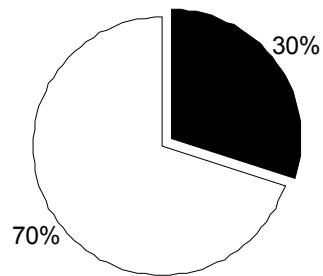
Să se reprezinte graficul de tip `pie` cu sectoare neexplodate. Să se reprezinte apoi sectorul 2 explodat. Să se reprezinte sectoarele și în vedere tridimensională.

#### Rezolvare

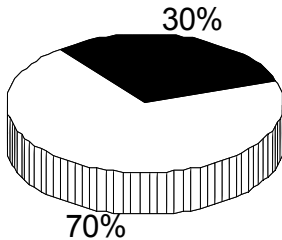
Reprezentarea graficelor de tip `pie` este prezentată în figura 5.23 pentru sectoare 2D și 3D, cu și fără sectoare explodate. Disponerea sectoarelor se face în sens trigonometric.



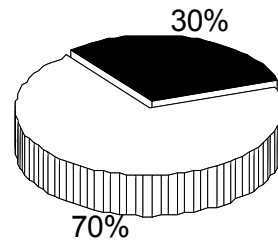
a) vedere 2D fără sectoare explodate



b) vedere 2D cu un sector explodat



c) vedere 3D fără sectoare explodate



d) vedere 3D cu un sector explodat

**Figura 5.23.** Reprezentarea grafică de tip pie.

Principalele instrucțiuni utilizate pentru controlul proprietăților sectoarelor 2D ale graficelor de tip pie sunt:

```
h=pie(y);
set(h(1), 'FaceColor', 'w');
set(h(2), 'FontSize', 11);
set(h(3), 'FaceColor', 'k');
set(h(4), 'FontSize', 11);
```

### Observații

- Proprietățile implicite ale sectoarelor se stabilesc automat la executarea instrucțiunii. Modificarea acestor proprietăți se poate realiza, în mod separat, pentru fiecare sector. Instrucțiunea `h=pie(y)` atribuie toate proprietățile reprezentării grafice unui identificator notat cu `h`. Pentru fiecare sector pot fi controlate proprietățile specifice ale elementelor grafice de tip `patch` și `text` cu ajutorul cărora se construiește graficul de tip pie. În cazul sectoarelor 2D, prima componentă a identificatorului, `h(1)`, controlează suprafața, iar a doua componentă, `h(2)`, controlează textul asociat fiecărui sector. În cazul sectoarelor 3D, primele trei componente ale identificatorului, `h(1)`, `h(2)` și `h(3)`, controlează fețele inferioară, laterală și superioară ale sectorului, iar componentă `h(4)` controlează textul asociat sectorului respectiv.

## 5.12. GRAFICE DE TIP `fill`

Se consideră un număr de  $n$  puncte  $P_i$  având coordonatele  $x_i$  și  $y_i$ ,  $i = 1 \div n$ . Pentru reprezentarea grafică a suprafeței interioare conturului poligonal definit de punctele considerate  $P_i(x_i, y_i)$ ,  $i = 1 \div n$  și colorarea suprafeței respective folosind culoarea  $c$ , se utilizează instrucțiunea:

```
fill(x,y,'c')
```

### Problema 5.13

Se consideră un număr de  $n=5$  puncte având coordonatele:

	1	2	3	4	5
$x$ [m]	1	2	4	7	3
$y$ [m]	5	3	2,5	6	7,5

Să se reprezinte grafic poligonul definit de cele cinci puncte  $P_i(x_i, y_i)$ ,  $i = 1 \dots n$  și să se calculeze aria acestui poligon.

### Rezolvare

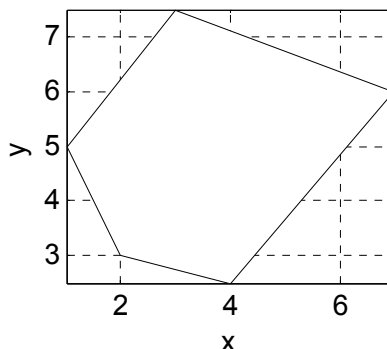
Fișierul `script` este prezentat în figura 5.24, a). Poligonul obținut este reprezentat în figura 5.24, b). Pentru colorarea suprafeței s-a utilizat opțiunea `'w'`. Datorită faptului că mărimile reprezentate pe cele două axe au aceeași unitate de măsură se impune scalarea egală a celor două axe folosind instrucțiunea `axis image`.

Pentru calculul suprafeței poligonului s-a utilizat instrucțiunea:

```
A=polyarea(x,y)
```

Rezultatul obținut este  $A=17,25 \text{ m}^2$ .

```
%% GRAFICE 2D (13)
clear all;close all;clc;
%% DATE INITIALE
%Definire punctelor
x=[1 2 4 7 3];
y=[5 3 2.5 6 7.5];
%% REPREZENTARE GRAFICA
figure
fill(x,y,'w');grid on;
xlabel('x');ylabel('y');
axis image;
%Calculul ariei poligonului
A=polyarea(x,y)
```



a) fișierul `script`

b) reprezentarea grafică

**Figura 5.24.** Reprezentarea grafică de tip `fill`.

### 5.13. GRAFICE ÎN COORDONATE POLARE

Pentru reprezentarea grafică a funcțiilor în coordonate polare  $(r, \theta)$  se utilizează instrucțiunea `polar`. Forma generală de implementare a instrucțiunii este:

```
polar(theta, r)
```

în care `theta` reprezintă unghiul în radiani dintre raza vectorială și axa  $Ox$ , iar `r` reprezintă lungimea razei vectoriale.

Pentru graficele în coordonate polare instrucțiunile de adnotare ale axelor sau ale rețelei grid nu au relevanță.

#### Problema 5.14

Se consideră următoarea funcție definită în coordonate polare  $(r, \theta)$  (spirală lui Arhimede) prin expresia:

$$r = a\theta, \quad a = 1$$

Să se reprezinte în coordonate polare și carteziene spirală lui Arhimede pentru domeniile de definiție  $\theta=[0; 2\pi]$  și  $\theta=[0; 4\pi]$ .

#### Rezolvare

Reprezentarea grafică a funcției este prezentată în figura 5.25 pentru coordonate polare și în figura 5.26 pentru coordonate carteziene. Pentru obținerea acestor reprezentări grafice au fost folosite instrucțiunile:

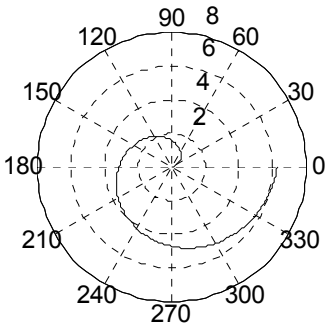
```
%% GRAFICE 2D (14)
clear all;close all;clc;
%% DATE INITIALE
a=1;
tmin=0;tmax1=2*pi;tmax2=4*pi;nt=300;
t1=linspace(tmin,tmax1,nt);
t2=linspace(tmin,tmax2,nt);
r=@(t) a*t;
%% GRAFICE POLARE
figure
polar(t1,r(t1),'k');
figure
polar(t2,r(t2),'k');
%% GRAFICE CARTEZIENE
x1=r(t1).*cos(t1);
y1=r(t1).*sin(t1);
figure
plot(x1,y1,'k');grid on;
xlabel('x');ylabel('y');
axis image;
x2=r(t2).*cos(t2);
```



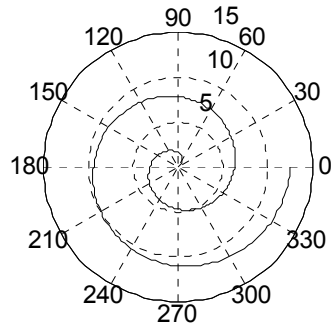
```

y2=r(t2).*sin(t2);
figure
plot(x2,y2,'k');
grid on;
xlabel('x');ylabel('y');
axis image;

```

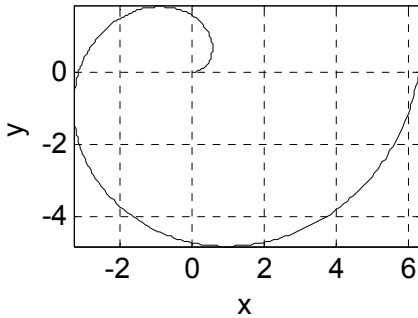


a)  $\theta=[0; 2\pi]$

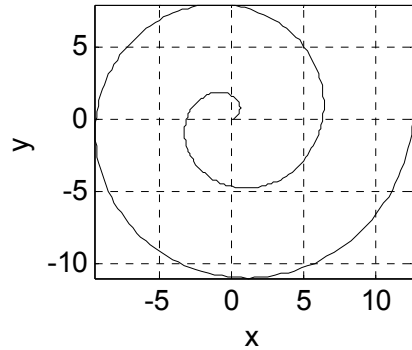


b)  $\theta=[0; 4\pi]$

**Figura 5.25.** Spirala lui Arhimede în coordonate polare.



a)  $\theta=[0; 2\pi]$



b)  $\theta=[0; 4\pi]$

**Figura 5.26.** Spirala lui Arhimede în coordonate carteziene.

### Observații

- S-a utilizat o funcție de tip anonymous pentru specificarea expresiei matematice a funcției de analizat,  $r=@(t) a*t$ . Parametrul acestei funcții este domeniul de definiție ( $t$ ). Apelarea funcției s-a făcut de două ori, pentru  $\theta=[0; 2\pi]$  și  $\theta=[0; 4\pi]$
- Pentru transformarea coordonatelor polare în coordonate carteziene au fost utilizate relațiile:

$$\begin{cases} x = r \cdot \cos \theta \\ y = r \cdot \sin \theta \end{cases}$$

## 5.14. UTILIZAREA CARACTERELOR SPECIALE

Limbajul de programare MATLAB permite aplicarea în cadrul obiectelor grafice de tip `axes` a mai multor tipuri de adnotări, [16]. Adnotările care au ca obiect manipularea șirurilor de caractere sunt:

1. Adnotarea de etichetare a axelor:

```
xlabel('valoare proprietate')
ylabel('valoare proprietate')
```

2. Adnotării de tip titlu:

```
title('valoare proprietate')
```

3. Adnotarea de tip legendă:

```
legend('pic1', 'pic2', ...)
```

în care `pic1` și `pic2` reprezintă parametrii de identificare ai curbelor 1 și 2.

4. Adnotarea de tip text:

```
text(xt, yt, 'text')
```

în care `xt` și `yt` sunt coordonatele de amplasare în fereastra grafică a textului, iar `'text'` reprezintă textul propriu-zis.

Pentru toate aceste tipuri de adnotări pe lângă caracterele normale, se pot utiliza și caractere speciale. Principalele tipuri de caractere speciale sunt:

- Indicii inferiori. Obținerea indicelui inferior simplu se realizează prin utilizarea caracterului special `_` care are rolul de a plasa caracterul imediat următor pe poziția de indice inferior.
- Indicii superiori. În cazul în care se dorește obținerea poziției de indice superior simplu se va folosi caracterul special `^`. În cazul în care pe poziția de indice inferior, respectiv superior se dorește plasarea mai multor caractere (indice inferior și superior de tip multiplu) se impune includerea acestora între acolade, `{}`.
- Simboluri matematice și literele grecești. Introducerea simbolurilor matematice și a caracterelor grecești se face prin utilizarea unui interpretor de caractere speciale. Sunt implementate două seturi de caractere speciale:  $\text{T}_{\text{E}}\text{X}$  și  $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ . Selectarea interpretorului de caractere speciale se face prin modificarea proprietății `Interpreter` la una din valorile particulare `tex`, `latex`, respectiv `none`. Interpretorul implicit este  $\text{T}_{\text{E}}\text{X}$ . Informații suplimentare despre aceste interpretoare se găsesc în [19].

Spre exemplu, introducerea unui text cu folosirea interpretorului de tip  $L_A T_E X$  se face printr-o instrucțiune de forma:

```
text ('Interpreter' , 'latex' , .....
```

În tabelul 5.1 se prezintă modul de obținere a literelor alfabetului grecesc iar în tabelul 5.2 se prezintă modul de obținere a simbolurilor matematice uzuale. Șirul de caractere care generează simbolul matematic sau litera alfabetului grecesc se introduce imediat după caracterul  $\backslash$ .

**Tabel 5.1.** Modul de obținere a literelor alfabetului grecesc.

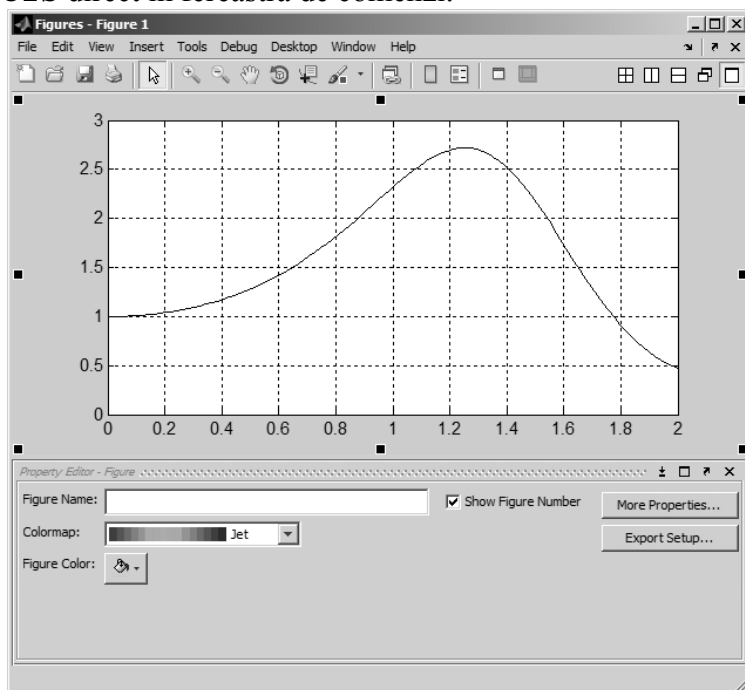
Simbol	Sintaxa	Simbol	Sintaxa	Simbol	Sintaxa
$\alpha$	$\backslash\alpha$	$\nu$	$\backslash\nu$	$\Gamma$	$\backslash\Gamma$
$\beta$	$\backslash\beta$	$\xi$	$\backslash\xi$	$\Delta$	$\backslash\Delta$
$\gamma$	$\backslash\gamma$	$\omicron$	$\omicron$	$\Theta$	$\backslash\Theta$
$\delta$	$\backslash\delta$	$\pi$	$\backslash\pi$	$\Lambda$	$\backslash\Lambda$
$\epsilon$	$\backslash\epsilon$	$\rho$	$\backslash\rho$	$\Xi$	$\backslash\Xi$
$\zeta$	$\backslash\zeta$	$\sigma$	$\backslash\sigma$	$\Pi$	$\backslash\Pi$
$\eta$	$\backslash\eta$	$\tau$	$\backslash\tau$	$\Sigma$	$\backslash\Sigma$
$\theta$	$\backslash\theta$	$\upsilon$	$\backslash\upsilon$	$\Upsilon$	$\backslash\Upsilon$
$\iota$	$\backslash\iota$	$\phi$	$\backslash\phi$	$\Phi$	$\backslash\Phi$
$\kappa$	$\backslash\kappa$	$\chi$	$\backslash\chi$	$\Psi$	$\backslash\Psi$
$\lambda$	$\backslash\lambda$	$\psi$	$\backslash\psi$	$\Omega$	$\backslash\Omega$
$\mu$	$\backslash\mu$	$\omega$	$\backslash\omega$		

**Tabel 5.2.** Modul de obținere a simbolurilor matematice uzuale.

Simbol	Sintaxa	Simbol	Sintaxa	Simbol	Sintaxa
$\equiv$	$\backslash\equiv$	$\cap$	$\backslash\cap$	$\infty$	$\backslash\infty$
$\cong$	$\backslash\cong$	$\cup$	$\backslash\cup$	$\partial$	$\backslash\partial$
$\approx$	$\backslash\approx$	$\supset$	$\backslash\supset$	$\circ$	$\backslash\circ$
$\sim$	$\backslash\sim$	$\supseteq$	$\backslash\supseteq$	$\nabla$	$\backslash\nabla$
$\neq$	$\backslash\neq$	$\subset$	$\backslash\subset$	$\emptyset$	$\backslash\emptyset$
$\leq$	$\backslash\leq$	$\subseteq$	$\backslash\subseteq$	$\int$	$\backslash\int$
$\geq$	$\backslash\geq$	$\in$	$\backslash\in$	$\Re$	$\backslash\Re$
$\pm$	$\backslash\pm$	$\perp$	$\backslash\perp$	$\Im$	$\backslash\Im$
$\div$	$\backslash\div$	$\otimes$	$\backslash\otimes$	$\aleph$	$\backslash\aleph$
$\cdot$	$\backslash\cdot$	$\oplus$	$\backslash\oplus$	$\wp$	$\backslash\wp$

## 5.15. CREAREA ȘI EDITAREA GRAFICELOR UTILIZÂND INTERFAȚA Plot Tools

Crearea și editarea graficelor sunt operațiuni care se pot realiza fie prin folosirea instrucțiunilor specifice, fie cu ajutorul unei interfețe specializate denumită Plot Tools, [17]. În cazul în care se dorește editarea unei reprezentări grafice existente, lansarea interfeței Plot Tools se realizează prin selectarea comenzii Show Plot Tools din toolbar-ul Figure al figurii respective (figura 5.27). În cazul în care se dorește realizarea unei noi reprezentări grafice și ulterior editarea acesteia, lansarea interfeței Plot Tools se realizează prin tastarea instrucțiunii `plottools` direct în fereastra de comenzi.

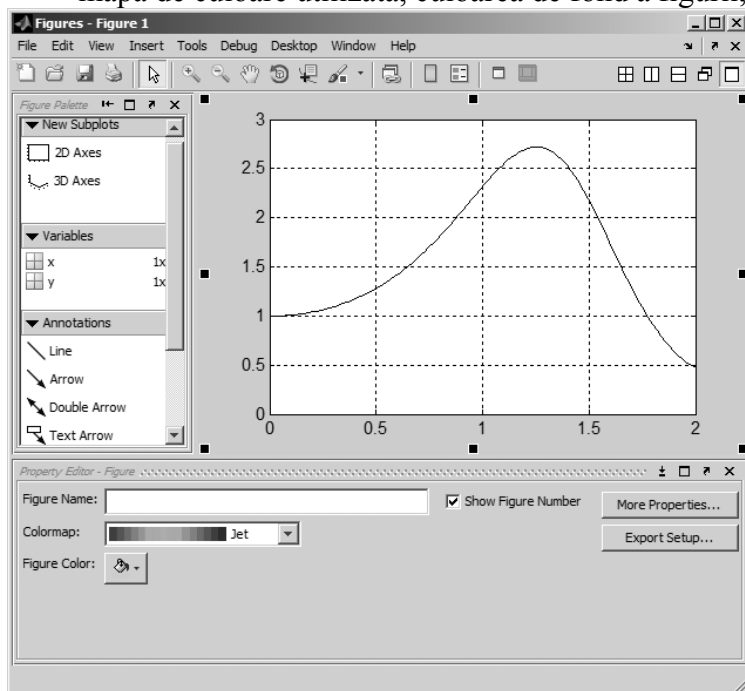


**Figura 5.27.** Lansarea interfeței Plot Tools.

Principalele operațiuni care se pot efectua cu ajutorul interfeței Plot Tools sunt grupate în funcție de obiectul grafic la care se referă:

- Operațiuni specifice toolbar-ului Figure Palette (figura 5.28). Deschiderea acestui toolbar se realizează din meniul View.
  - Adăugarea unor noi axe (de tip 2D sau 3D) și realizarea astfel a unei structuri de mai multe axe pe aceeași fereastră grafică (similar instrucțiunii `subplot`).
  - Adăugarea unor adnotări de tip grafic pe spațiul figurii (linie, săgeată simplă, săgeată dublă, săgeată de poziționare cu text, casetă de text, dreptunghi, elipsă).

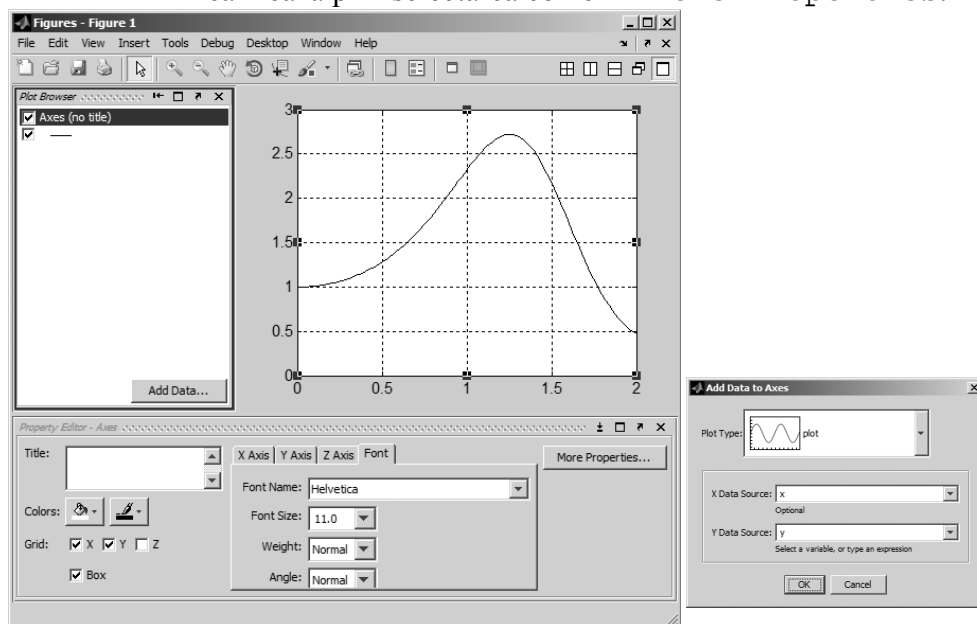
- Modificarea parametrilor specifici ai obiectului grafic figure (Property Editor-Figure): titlul figurii, mapa de culoare utilizată, culoarea de fond a figurii, etc.



**Figura 5.28.** Toolbar-ul Figure Palette.

- Operațiuni specifice toolbar-ului Plot Browser (figura 5.29, a). Deschiderea acestui toolbar se realizează din meniul View.
  - Adăugarea unor noi curbe și realizarea astfel a unei structuri de mai multe curbe pe aceleași axe (similar efectului instrucțiunilor `hold on` și `hold off`). Crearea unor noi seturi de date care se vor reprezenta pe aceleași axe se realizează prin selectarea comenzii `Add Data` (figura 5.29, b) și selectarea apoi a variabilelor corespunzătoare existente deja în spațiul de lucru al programului. Comanda `Add Data` permite și selectarea tipului de grafic care se va realiza.
  - Modificarea parametrilor specifici ai obiectului grafic axes (Property Editor-Axes): titlul graficului; etichetele axelor; rețeaua de tip grid; limitele domeniului vizibil al axelor; tipul de scalare al axelor (liniar sau logaritmic); culoarea de fond a spațiului dintre axe; culoarea axelor, grid-ului și a valorilor numerice de pe axe; poziția și direcția de amplasare a valorilor numerice de pe axe; caracteristicile caracterelor utilizate la scrierea valorilor numerice de pe axe

(tipul de font, dimensiunea font-ului utilizat); caracteristicile caracterelor utilizate la scrierea etichetelor axelor și a titlului (tipul de font, dimensiunea și culoarea font-ului, alinierea etichetei). Obținerea listei tuturor parametrilor specifice se realizează prin selectarea comenzii *More Properties*.



a) modificarea proprietăților axelor

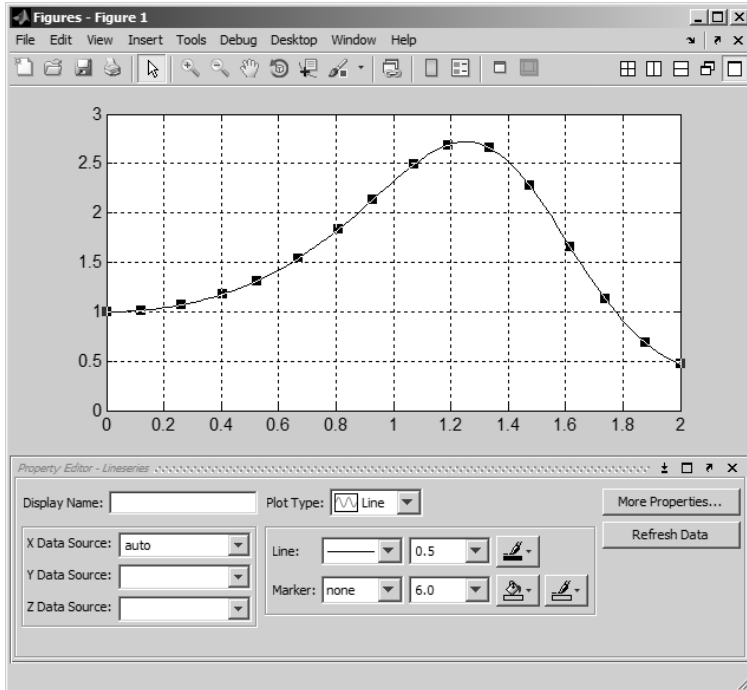
b) adăugarea unui nou set de date

**Figura 5.29.** Toolbar-ul *Plot Browser*.

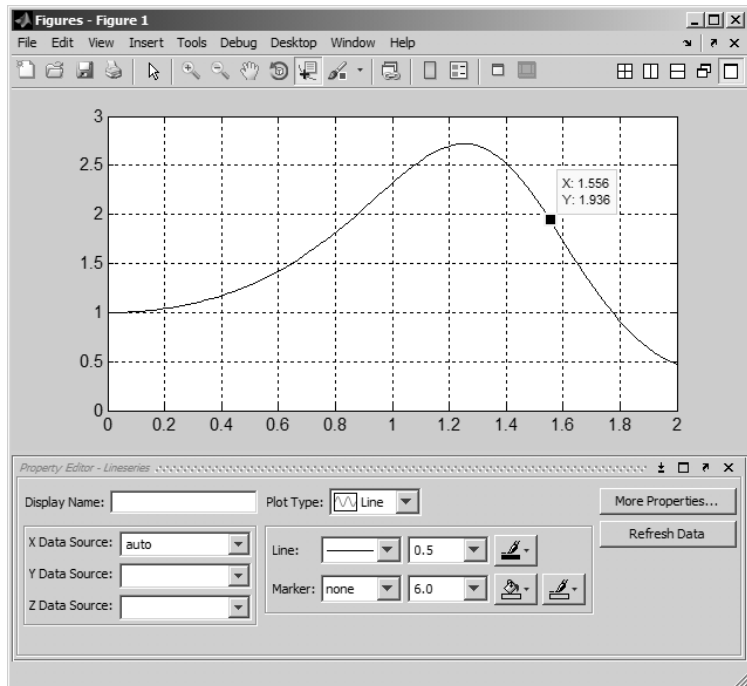
Modificarea diferitelor proprietăți ale elementelor componente ale reprezentării grafice (figură, axe, curbe, etc.) se realizează prin selectarea componenteii respective, după care în fereastra *Property Editor* se vor modifica parametrii doriți. De exemplu, în figura 5.30, după selectarea curbei, se observă în fereastra *Property Editor* parametrii asociați curbei selectate: variabilele asociate setului de date; tipul graficului; tipul, grosimea și culoarea liniei; tipul, dimensiunea și culoarea marker-elor, etc.

Determinarea interactivă a coordonatelor curente ale oricărui punct al unei curbe se poate realiza cu ajutorul funcției de interogare *Data Cursor* care se poate lansa în execuție fie din meniul *Tools* fie direct din bara cu butoane a interfeței *Plot Tools*. În figura 5.31 se observă coordonatele punctului selectat de pe curbă:  $X: 1.556$  și  $Y: 1.936$ .

După finalizarea procesului de formatare a unei reprezentări grafice prin selectarea comenzii *Hide Plot Tools* din bara de butoane se părăsește mediul de editare și se revine la fereastra grafică inițială.



**Figura 5.30.** Modificarea parametrilor curbelor.



**Figura 5.31.** Funcția de interogare Data Cursor.

### Problema 5.15

Pentru construcția analitică a profilului aerodinamic de tip Jukovski se consideră următorii parametri de intrare: raza cercului generator  $r_0=1$ ; raza vectoare a centrului cercului generator  $m=0,125$ ; unghiul dintre coarda profilului și raza vectoare a centrului cercului generator  $\delta=40^\circ$ . Se mai cunosc: unghiul de incidență  $\alpha_\infty=6^\circ$  și viteza incidentă  $V_\infty=20$  m/s.

Să se reprezinte grafic (în coordonate adimensionale) profilul Jukovski. Să se reprezinte distribuția de viteză adimensională, precum și distribuția coeficientului de presiune pe profilul aerodinamic.

### Rezolvare

Pentru determinarea coordonatelor profilului trebuie parcurse următoarele etape, [20]:

- Calculul unghiului dintre coarda profilului și raza vectoare a segmentului de pe semiaxa reală pozitivă interceptat de cercul generator:

$$\tau = \arcsin\left(\frac{m}{r_0} \sin \delta\right)$$

- Calculul lungimii segmentului de pe semiaxa reală pozitivă interceptat de cercul generator:

$$q = r_0 \cos \tau - m \cos \delta$$

- Definirea unghiului de parcurgere al cercului generator:

$$\theta = 0 \dots 2\pi$$

- Calculul coordonatelor profilului Jukovski. Pornind de la transformarea Jukovski, se obțin următoarele relații explicite pentru abscisa și ordonata profilului:

$$x_1 = (-m \cos \delta + r_0 \cos \theta) \left[ 1 + \frac{q^2}{m^2 + r_0^2 - 2mr_0 \cos(\theta + \delta)} \right]$$

$$y_1 = (m \sin \delta + r_0 \sin \theta) \left[ 1 - \frac{q^2}{m^2 + r_0^2 - 2mr_0 \cos(\theta + \delta)} \right]$$

- Reprezentarea grafică a profilului  $\bar{y}_1(\bar{x}_1)$ , în care  $\bar{y}_1 = y_1/L$  și  $\bar{x}_1 = x_1/L$  sunt coordonatele adimensionale.

Pentru determinarea distribuției de viteză și a variației coeficientului de presiune pe profilul aerodinamic de tip Jukovski trebuie parcurse următoarele etape:

- Afixa centrului cercului generator:

$$\zeta_0 = -m \cos \delta + i \cdot m \sin \delta$$

- Afixa punctelor de pe cercul generator:

$$\zeta = \zeta_0 + r_0 \cdot e^{i\theta}$$

- Funcția de transformare Jukovski:



$$z = \zeta + \frac{q^2}{\zeta}$$

- Determinarea coordonatelor profilului se poate face și direct, prin determinarea numerică a părții reale și imaginare a funcției de transformare a lui Jukovski. Astfel, pentru obținerea abscisei profilului se utilizează relația:

$$x_2 = \text{real}(z)$$

ordonata profilului determinându-se cu relația:

$$y_2 = \text{imag}(z)$$

- Circulația vitezei pe profil:

$$\Gamma = 4\pi r_0 V_\infty \sin(\alpha_\infty + \tau)$$

- Distribuția vitezei pe profil:

$$V = \frac{\left| 2V_\infty \sin(\theta - \alpha_\infty) + \frac{\Gamma}{2\pi r_0} \right|}{\left| 1 - \frac{q^2}{\zeta^2} \right|}$$

- Distribuția coeficientului de presiune pe profil:

$$k_p = 1 - \left( \frac{V}{V_\infty} \right)^2$$

- Reprezentarea grafică a profilului  $\bar{y}_2(\bar{x}_2)$ , a distribuției de viteză adimensională  $V/V_0$  și a variației coeficientului de presiune pe profil  $k_p$  ( $\bar{y}_2 = y_2/L$  și  $\bar{x}_2 = x_2/L$  sunt coordonatele adimensionale).

Pentru rezolvarea problemei se definește un fișier de tip script conținând următoarele instrucțiuni principale:

```

%% STUDIUL PROFILULUI JUKOVSKI
close all;clear all;clc;
%% DATE DE INTRARE
% Raza cercului generator
r0=1;
% Raza vectoare a centrului cercului generator
m=0.125;
% Unghiul dintre coarda profilului si raza vectoare:
d=40;
% Unghiul de incidenta
a0=6;
% Viteza incidenta [m/s]
V0=20;
%% CALCULE PRELIMINARE
% Unghiul tau
tau=asin(m/r0*sind(d));
% Lungimea q
q=r0*cos(tau)-m*cosd(d);
% Coarda profilului
L=4*q;

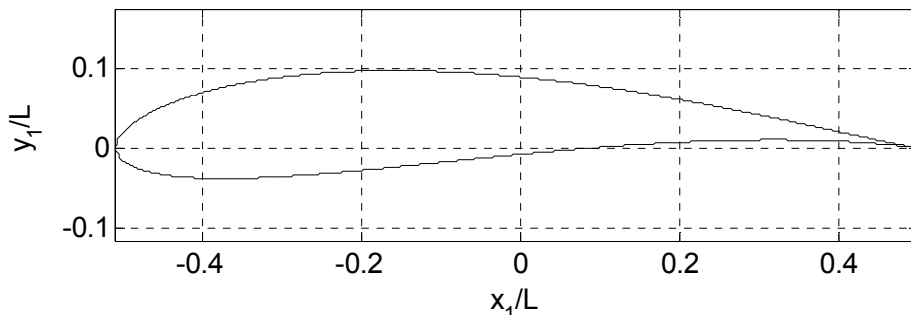
```

```

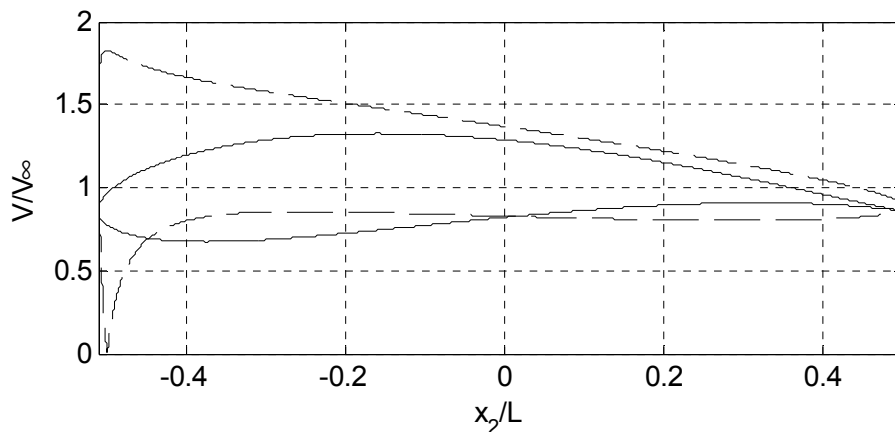
% Unghiul t
t=linspace(0,2*pi,1000);
%% CALCULUL COORDONATELOR
% Abscisa profilului
x1=(-m*cosd(d)+r0*cos(t)).*(1+q^2./(m^2+r0^2-2*m*r0*
cos(t+d*pi/180)));
% Ordonata profilului
y1=(m*sind(d)+r0*sin(t)).*(1-q^2./(m^2+r0^2-2*m*r0*
cos(t+d*pi/180)));
%% CALCULUL VITEZEI SI PRESIUNII PE PROFIL
% Afixa centrului cercului generator
zeta0=-m*cosd(d)+i*m*sind(d);
% Afixa punctelor de pe cercul generator
zeta=@(t) zeta0+r0*exp(i*t);
% Functia de transformare Jukovski
z=@(t) zeta(t)+q^2./zeta(t);
% Abscisa profilului
x2=@(t) real(z(t));
% Ordonata profilului
y2=@(t) imag(z(t));
% Circulatia vitezei
G=4*pi*r0*V0*sin(a0*pi/180+tau);
% Viteza pe profil
V=@(t) abs(2*V0*sin(t-a0*pi/180)+G/(2*pi*r0))./abs(1-q^2./
zeta(t).^2);
% Coeficientul de presiune pe profil
kp=@(t) 1-(V(t)./V0).^2;
%% REPREZENTAREA GRAFICA A PROFILULUI
figure
plot(x1/L,y1/L,'-k');
xlabel('x_1/L');ylabel('y_1/L');grid on;axis equal;
%% REPREZENTAREA GRAFICA A VITEZEI
figure
[AX H1 H2]=plotyy(x2(t)/L,V(t)/V0,x2(t)/L,y2(t)/L,'plot');
% formatarea independenta a celor doua axe y
axes(AX(1));
grid on;box on;xlabel('x_2/L');ylabel('V/V_0');
axes(AX(2));
set(gca,'Visible','off');axis image;box off;
%formatari comune
set(gcf,'Color','w');
%% REPREZENTARE GRAFICA A COEFICIENTULUI DE PRESIUNE
figure
[AX H1 H2]=plotyy(x2(t)/L,kp(t),x2(t)/L,y2(t)/L,'plot');
% formatarea independenta a celor doua axe y
axes(AX(1));
grid on;box on;xlabel('x_2/L');ylabel('k_p');
axes(AX(2));
set(gca,'Visible','off');axis image;box off;
% formatari comune
set(gcf,'Color','w');

```

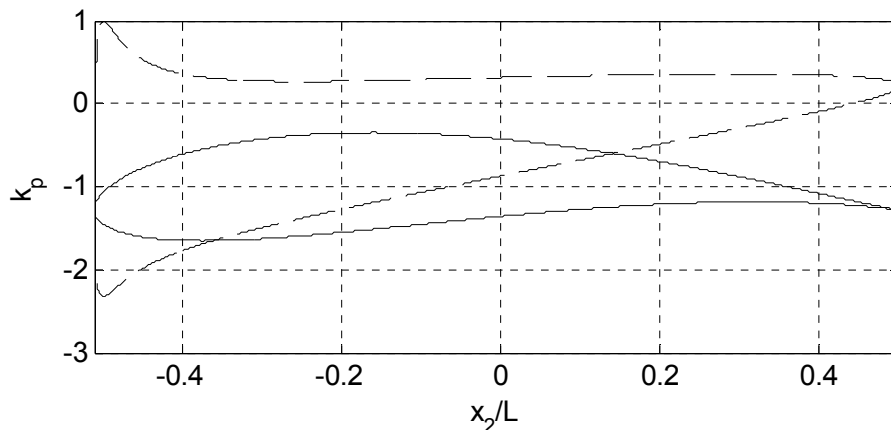
Lansarea în execuție a fișierului `script` conduce la obținerea profilului aerodinamic (figura 5.32), la distribuția de viteză adimensională pe profil (figura 5.33), precum și la distribuția coeficientului de presiune pe profil (figura 5.34).



**Figura 5.32.** Profilul Jukovski,  $\bar{y}_1(\bar{x}_1)$ .



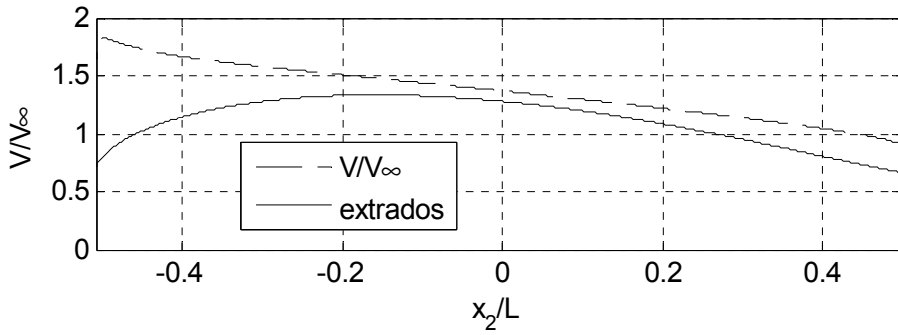
**Figura 5.33.** Profilul  $\bar{y}_2(\bar{x}_2)$  și distribuția de viteză adimensională.



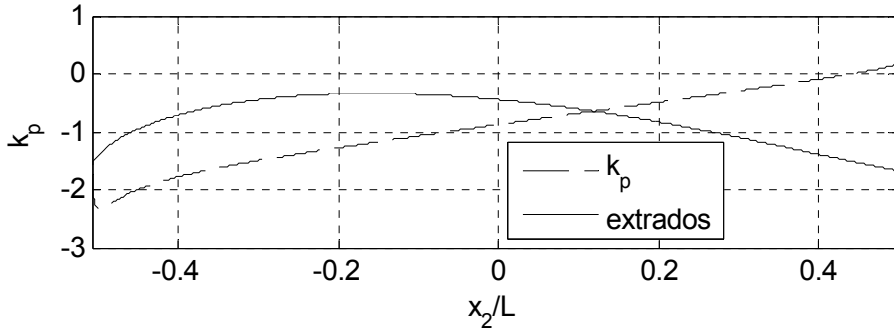
**Figura 5.34.** Profilul  $\bar{y}_2(\bar{x}_2)$  și variația coeficientului de presiune.

## Observații

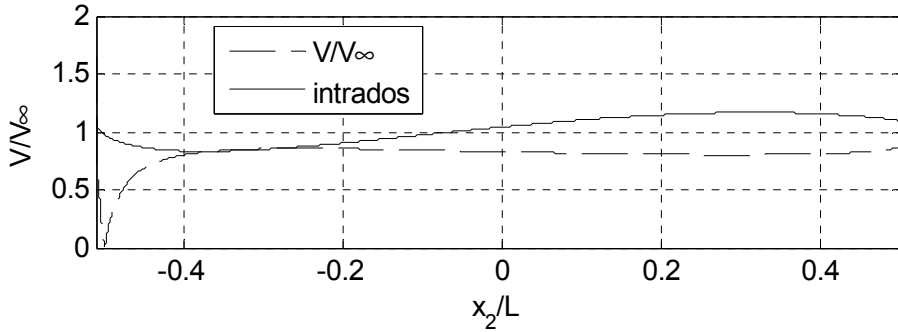
- Datorită faptului că unghiul dintre coarda profilului și raza vectorială a centrului cercului generator este exprimat în grade, se utilizează funcțiile trigonometrice  $\sin(d)$  și  $\cos(d)$ . În relațiile abscisei și ordonatei profilului apare termenul  $\cos(\theta + \delta)$  având ca argument suma dintre un unghi exprimat în radiani și un unghi exprimat în grade. S-a folosit funcția trigonometrică  $\cos(t + d * \pi / 180)$ , în care s-a efectuat transformarea unghiului  $\delta$  în radiani.
- Calitatea reprezentării grafice depinde de numărul de puncte de pe intradosul și extradadosul profilului. Din acest motiv, la discretizarea cercului generator s-a utilizat un număr mare de puncte (1000):  $t = \text{linspace}(0, 2 * \pi, 1000)$ .
- Pentru reprezentarea grafică a profilului  $\bar{y}_1(\bar{x}_1)$  s-a utilizat instrucțiunea  $\text{plot}(x1/L, y1/L, '-k')$ .
- Pentru calculul coordonatelor profilului  $\bar{y}_2(\bar{x}_2)$ , a distribuției de viteză adimensională  $V/V_\infty$  și a coeficientului de presiune  $k_p$ , toate mărimile caracteristice (afixa centrului cercului generator, afixa punctelor de pe cercul generator, funcția de transformare Jukovski, coordonatelor profilului, circulația vitezei pe profil, distribuția vitezei pe profil și distribuția coeficientului de presiune pe profil) au fost definite ca funcții de tip anonymous, având ca parametru unghiul  $\theta = 0 \div 2\pi$  de parcurgere a cercului generator.
- Reprezentarea grafică a distribuției de viteză adimensională  $V/V_\infty$  se suprapune peste profilul aerodinamic  $\bar{y}_2(\bar{x}_2)$ . În acest scop s-a utilizat instrucțiunea  $\text{plotyy}$ , ordonata din partea stângă fiind asociată cu viteza adimensională  $V/V_\infty$ , iar ordonata din partea dreaptă cu ordonata profilului  $\bar{y}_2$ , abscisa  $\bar{x}_2$  fiind comună ambelor reprezentări.
- Reprezentarea grafică a coeficientului adimensional de presiune  $k_p$  se suprapune peste profilul aerodinamic  $\bar{y}_2(\bar{x}_2)$ . În acest scop s-a utilizat instrucțiunea  $\text{plotyy}$ , ordonata din partea stângă fiind asociată cu valorile numerice ale coeficientului de presiune  $k_p$ , iar ordonata din partea dreaptă cu ordonata profilului  $\bar{y}_2$ , abscisa  $\bar{x}_2$  fiind comună ambelor reprezentări.
- Reprezentarea grafică a vitezei adimensionale și a coeficientului de presiune în mod separat, pentru intrados (figura 5.35 și figura 5.36) și pentru extradados (figura 5.37 și figura 5.38) se poate face dacă domeniul  $\theta = 0 \div 2\pi$  de variație al unghiului de parcurgere a cercului generator se împarte în două subdomenii:  $\theta_e = 0 \div \pi$  și  $\theta_i = \pi \div 2\pi$  cu instrucțiunile:  $t1 = \text{linspace}(0, \pi, 1000)$  și  $t2 = \text{linspace}(\pi, 2 * \pi, 1000)$ .



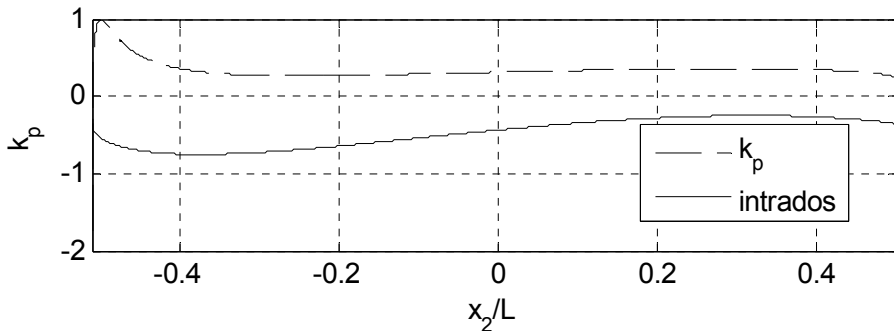
**Figura 5.35.** Variația vitezei adimensionale pe extrados.



**Figura 5.36.** Variația coeficientului de presiune pe extrados.



**Figura 5.37.** Variația vitezei adimensionale pe intrados.



**Figura 5.38.** Variația coeficientului de presiune pe intrados.

## BIBLIOGRAFIE

1. Magrab E.B., Azram S., Balachandran B., Duncan J.H., Herold K.E., Walsh G.C., An Engineer's Guide to MATLAB with Applications from Mechanical, Aerospace, Electrical and Civil Engineering, Second Edition, Pearson Prentice Hall, 2005.
2. Marchand P., Holland O.T., Graphics and GUIs with MATLAB, Chapman&Hall/CRC, 2003.
3. MathWorks, MATLAB, Graphics, 2014.
4. MathWorks, Graphics Objects, <http://www.mathworks.com/help/matlab/graphics-objects.html>, accesat la 2.02.2014.
5. MathWorks, 2D and 3D Plots, <http://www.mathworks.com/help/matlab/2-and-3d-plots.html>, accesat la 2.02.2014.
6. MathWorks, 2D Line Plot, <http://www.mathworks.com/help/matlab/ref/plot.html>, accesat la 26.08.2014.
7. MathWorks, Figure Properties, [http://www.mathworks.com/help/matlab/ref/figure\\_props.html](http://www.mathworks.com/help/matlab/ref/figure_props.html), accesat la 2.02.2014.
8. MathWorks, Axes Properties, [http://www.mathworks.com/help/matlab/ref/axes\\_props.html](http://www.mathworks.com/help/matlab/ref/axes_props.html), accesat la 2.02.2014.
9. MathWorks, Line Properties, [http://www.mathworks.com/help/matlab/ref/line\\_props.html](http://www.mathworks.com/help/matlab/ref/line_props.html), accesat la 2.02.2014.
10. MathWorks, Text Properties, [http://www.mathworks.com/help/matlab/ref/text\\_props.html](http://www.mathworks.com/help/matlab/ref/text_props.html), accesat la 2.02.2014.
11. MathWorks, Axis Scaling and Appearance, <http://www.mathworks.com/help/matlab/ref/axis.html>, accesat 2.02.2014.
12. MathWorks, Stairstep Graph, <http://www.mathworks.com/help/matlab/ref/stairs.html>, accesat la 5.02.2014.
13. MathWorks, Patch Properties, [http://www.mathworks.com/help/matlab/ref/patch\\_props.html](http://www.mathworks.com/help/matlab/ref/patch_props.html), accesat la 5.02.2014.
14. MathWorks, Errorbar, <http://www.mathworks.com/help/matlab/ref/errorbar.html>, accesat la 5.02.2014.
15. MathWorks, Define Areaseries Properties, <http://www.mathworks.com/help/matlab/ref/areaseriesproperties.html>, accesat la 5.02.2014.
16. MathWorks, Legend, Label, Graph Title, <http://www.mathworks.com/help/matlab/titles-and-labels.html>, accesat la 6.02.2014.
17. MathWorks, Customize Graph Using Plot Tools, [http://www.mathworks.com/help/matlab/creating\\_plots/plotting-tools--interactive-plotting.html](http://www.mathworks.com/help/matlab/creating_plots/plotting-tools--interactive-plotting.html), accesat la 26.08.2014.
18. Smith S.T., MATLAB-Advanced GUI Development, Dog Ear Publishing, Indianapolis, 2006.
19. TEX Users Group, (TUG), <http://www.tug.org/>, accesat la 6.02.2014.
20. Zidaru Gh., Mișcări potențiale și hidrodinamica rețelelor de profile, E:D.P., București, 1981.
21. Zahariea D., Măsurări hidraulice, Rotaprint, Universitatea Tehnică „Gheorghe Asachi”, Iași, 1999.

## CAPITOLUL 6

### PROBLEME SPECIFICE DE CALCUL NUMERIC

#### 6.1. CALCULE NUMERICE CU POLINOAME

##### 6.1.1. Definirea polinoamelor

Se consideră un polinom  $P_n(x)$  de grad  $n$ , definit prin:

$$P_n(x) = p_n x^n + p_{n-1} x^{n-1} + \dots + p_1 x^1 + p_0 x^0$$

în care  $p_n, p_{n-1}, \dots, p_1$  și  $p_0$  sunt coeficienții polinomului.

Definirea polinomului  $P_n(x)$  se realizează prin specificarea unui vector linie conținând coeficienții polinomului  $p_n, p_{n-1}, \dots, p_1, p_0$  în ordine descrescătoare a puterilor variabilei  $x$ , folosind instrucțiunea generală, [14]:

$$P_n = [p_n \ p_{n-1} \ \dots \ p_1 \ p_0] ;$$

De exemplu, definirea polinomului  $P_5(x)$  având expresia:

$$P_5(x) = 2x^5 + 4x^4 + 12x^3 - 5x^2 + 7x - 1$$

se realizează cu instrucțiunea:

$$P_5 = [2 \ 4 \ 12 \ -5 \ 7 \ -1] ;$$

Instrucțiunea pentru reprezentarea polinoamelor trebuie să conțină toți coeficienții polinomului, inclusiv coeficienții nuli. De exemplu, definirea polinomului  $P_7(x)$  având expresia:

$$P_7(x) = x^7 + 2x^4 + 8x^3 - 3x + 1$$

se realizează cu instrucțiunea:

$$P_7 = [1 \ 0 \ 0 \ 2 \ 8 \ 0 \ -3 \ 1] ;$$

##### 6.1.2. Evaluarea polinoamelor

Prin evaluarea unui polinom  $P_n(x)$  într-un punct oarecare  $x = a$  se înțelege calculul valorii polinomului în punctul respectiv,  $P_n(a)$ .

Evaluarea polinomului  $P_n(x)$  în punctul  $x = a$  se realizează cu instrucțiunea, [15]:

$$P_a = \text{polyval}(P_n, a) ;$$

în care  $P_n$  reprezintă vectorul coeficienților polinomului. În cazul în care  $a$  este un vector sau o matrice, procesul de evaluare se realizează pentru fiecare valoare a vectorului sau matricei respective.

### 6.1.3. Reprezentarea grafică a polinoamelor

Se consideră polinomul de grad  $n$ ,  $P_n(x): [x_{min}; x_{max}] \rightarrow \mathbb{R}$ :

$$P_n(x) = p_n x^n + p_{n-1} x^{n-1} + \dots + p_1 x^1 + p_0 x^0$$

în care  $p_n, p_{n-1}, \dots, p_1$  și  $p_0$  sunt coeficienții polinomului.

Se pune problema reprezentării grafice a valorilor polinomului  $P_n(x)$  pentru domeniul de definiție  $[x_{min}; x_{max}]$ .

Principalele etape necesare pentru rezolvarea acestei probleme sunt:

- Definirea polinomului prin specificarea unui vector linie conținând coeficienții polinomului  $p_n, p_{n-1}, \dots, p_1, p_0$ , în ordine descrescătoare a puterilor variabilei  $x$ .
- Specificarea vectorului  $x$  al domeniului de definiție  $[x_{min}, x_{max}]$  prin utilizarea metodelor de generare a vectorilor. În general domeniul de definiție este dat prin valorile sale limită, fără referiri asupra numărului de elemente sau pasului, recomandându-se astfel utilizarea instrucțiunii `linspace`.
- Evaluarea valorilor polinomului  $P_n(x)$  în punctele vectorului  $x$  al domeniului de definiție prin utilizarea instrucțiunii `polyval`.
- Reprezentarea grafică a valorilor polinomului  $P_n(x)$  în funcție de variabila  $x \in [x_{min}; x_{max}]$ , prin utilizarea instrucțiunii `plot`.

#### Problema 6.1

Se consideră polinomul definit prin:

$$P_4(x) = 2x^4 + 4x^3 + x^2 - 1$$

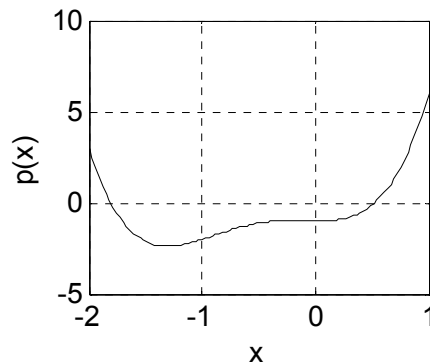
Să se reprezinte grafic polinomul  $P_4(x)$  pe domeniul de definiție  $[x_{min}; x_{max}] = [-2; 1]$ .

#### Rezolvare

Pentru rezolvarea problemei se utilizează fișierul de tip script din figura 6.1, a), graficul obținut fiind prezentat în figura 6.1, b).

```
%% POLINOAME (1)
close all;clear all;clc;
%% DATE DE INTRARE
% Coeficientii polinomului
p=[2 4 1 0 -1];
% Domeniul de definitie
x=linspace(-2,1);
%% EVALUAREA POLINOMULUI
px=polyval(p,x);
%% REPREZENTARE GRAFICA
plot(x,px,'k');grid on;
xlabel('x');ylabel('p(x)');
```

a) fișierul script



b) reprezentarea grafică

**Figura 6.1.** Reprezentarea grafică a polinomului  $P_4(x): [-2; 1] \rightarrow \mathbb{R}$ .



#### 6.1.4. Operații cu polinoame

##### Adunarea și scăderea polinoamelor

Operațiile aritmetice de adunare și scădere a polinoamelor se realizează prin adunarea, respectiv scăderea coeficienților de același ordin ai polinoamelor.

Spre exemplu, se consideră două polinoame  $A(x)$  și  $B(x)$  de grad  $n_A$  și respectiv  $n_B$ , pentru care  $n_A = n_B$ , definite prin:

$$A(x) = 2x^3 + 5x^2 - 3x + 1$$

$$B(x) = 7x^3 + 2x - 8$$

Pentru calculul sumei și diferenței celor două polinoame:

$$S(x) = A(x) + B(x)$$

$$D(x) = A(x) - B(x)$$

se vor utiliza următoarele instrucțiuni:

$$A = [2 \ 5 \ -3 \ 1] ;$$

$$B = [7 \ 0 \ 2 \ -8] ;$$

$$S = A + B ;$$

$$D = A - B ;$$

care vor conduce la următoarele rezultate:

$$S = [9 \ 5 \ -1 \ -7]$$

$$D = [-5 \ 5 \ -5 \ 9]$$

corespunzătoare următoarelor două polinoame:

$$S(x) = 9x^3 + 5x^2 - x - 7$$

$$D(x) = -5x^3 + 5x^2 - 5x + 9$$

O situație deosebită apare atunci când cele două polinoame nu au același grad. Spre exemplu, se consideră două polinoame  $A(x)$  și  $B(x)$  de grad  $n_A$  și respectiv  $n_B$ , pentru care  $n_A \neq n_B$ , definite prin:

$$A(x) = 2x^4 + 3x^3 + 5x^2 - 3x + 1$$

$$B(x) = 3x^2 + 5x - 2$$

pentru care se urmărește calculul sumei și diferenței celor două polinoame:

$$S(x) = A(x) + B(x)$$

$$D(x) = A(x) - B(x)$$

Încercarea de rezolvare a problemei cu ajutorul instrucțiunilor:

$$A = [2 \ 3 \ 5 \ -3 \ 1] ;$$

$$B = [3 \ 5 \ -2] ;$$

$$S = A + B ;$$

$$D = A - B ;$$

conduce la o eroare de tip „Matrix dimension must agree” datorită dimensiunii diferite a celor doi vectori  $A(x)$  și  $B(x)$ .

Rezolvarea corectă constă în redimensionarea polinomului cu grad mai mic, prin adăugarea de termeni nuli până la același grad cu cel al polinomului de grad maxim. În acest caz,  $n_A=4$  și  $n_B=2$ , deci polinomul  $B(x)$  va trebui redimensionat prin adăugarea a  $n_A - n_B=4-2=2$  termeni, astfel încât pentru realizarea operațiilor de adunare și scădere, polinomul  $B(x)$  va avea forma modificată:

$$B(x) = 0x^4 + 0x^3 + 3x^2 + 5x - 2$$

Pentru calculul corect al sumei și diferenței celor două polinoame se vor utiliza instrucțiunile:

$$A = [2 \ 3 \ 5 \ -3 \ 1] ;$$

$$B = [0 \ 0 \ 3 \ 5 \ -2] ;$$

$$S = A + B ;$$

$$D = A - B ;$$

care vor conduce la următoarele rezultate:

$$S = [2 \ 3 \ 8 \ 2 \ -1]$$

$$D = [2 \ 3 \ 2 \ -8 \ 3]$$

corespunzătoare următoarelor două polinoame:

$$S(x) = 2x^4 + 3x^3 + 8x^2 + 2x - 1$$

$$D(x) = 2x^4 + 3x^3 + 2x^2 - 8x + 3$$

### Înmulțirea polinoamelor

Operația aritmetică de înmulțire a două polinoame  $A(x)$  și  $B(x)$ , de grad  $n_A$  și respectiv  $n_B$ , operație definită prin  $P(x) = A(x) \cdot B(x)$ , este o operație de convoluție a celor două polinoame și se realizează cu ajutorul instrucțiunii, [16]:

$$P = \text{conv}(A, B)$$

în care  $A$  și  $B$  sunt vectorii coeficienților celor două polinoame, iar  $P$  este vectorul coeficienților polinomului produs. Gradul polinomului produs  $P(x)$  este  $n_P = n_A + n_B$ .

Spre exemplu, se consideră două polinoame  $A(x)$  și  $B(x)$ , de grad  $n_A=3$  și respectiv  $n_B=2$ , definite prin:

$$A(x) = 6x^3 + 2x^2 + 3x - 2$$

$$B(x) = 3x^2 - 5x + 1$$

Pentru calculul produsului  $P(x)$  al celor două polinoame:

$$P(x) = A(x) \cdot B(x)$$

se vor utiliza următoarele instrucțiuni:

$$A = [6 \ 2 \ 3 \ -2] ;$$

$$B = [3 \ -5 \ 1] ;$$

$$P = \text{conv}(A, B) ;$$

care va conduce la următorul rezultat:

$$P = [18 \quad -24 \quad 5 \quad -19 \quad 13 \quad -2]$$

corespunzător următorului polinom:

$$P(x) = 18x^5 - 24x^4 + 5x^3 - 19x^2 + 13x - 2$$

Se observă gradul polinomului produs  $n_P = n_A + n_B = 3 + 2 = 5$ .

### Împărțirea polinoamelor

Operația aritmetică de împărțire a două polinoame  $A(x)$  și  $B(x)$ , de grad  $n_A$  și respectiv  $n_B$ , operație definită prin  $A(x)/B(x)$ , este o operație de deconvoluție a celor două polinoame și se realizează cu instrucțiunea, [17]:

$$[C, R] = \text{deconv}(A, B)$$

în care  $A$  și  $B$  sunt vectorii coeficienților celor două polinoame, iar  $C$  și  $R$  sunt vectorii coeficienților polinoamelor cât și rest, coresponzător relației de definire a operației de împărțire:

$$A(x) = C(x) \cdot B(x) + R(x)$$

Gradul polinomului produs  $C(x)$  este  $n_C = n_A - n_B$ , iar gradul polinomului rest  $R(x)$  este  $n_R = n_B - 1$ .

Spre exemplu, se consideră două polinoame  $A(x)$  și  $B(x)$ , de grad  $n_A = 3$  și respectiv  $n_B = 2$ , definite prin:

$$A(x) = 6x^3 + 2x^2 + 3x - 2$$

$$B(x) = 3x^2 - 5x + 1$$

Pentru operația de împărțire a celor două polinoame, calculul câtului  $C(x)$  și al restului  $R(x)$  se realizează cu următoarele instrucțiuni:

$$A = [6 \quad 2 \quad 3 \quad -2];$$

$$B = [3 \quad -5 \quad 1];$$

$$[C, R] = \text{deconv}(A, B);$$

care vor conduce la următoarele rezultate:

$$C = [2 \quad 4]$$

$$R = [0 \quad 0 \quad 21 \quad -6]$$

corespunzătoare următoarelor două polinoame:

$$C(x) = 2x + 4$$

$$R(x) = 21x - 6$$

### Derivarea polinoamelor

Operația de derivare a unui polinom  $A(x)$ , de grad  $n_A$ , operație definită prin  $D(x) = \frac{d}{dx}[A(x)] = A'(x)$ , se realizează cu ajutorul unei instrucțiuni de forma, [18]:

$$D = \text{polyder}(A)$$

în care  $A$  este vectorul coeficienților polinomului, iar  $D$  este vectorul coeficienților polinomului derivat. Gradul polinomului  $D(x)$  obținut în urma operației de derivare este  $n_D = n_A - 1$ .

Spre exemplu, se consideră polinomul  $A(x)$ , de grad  $n_A=3$ , definit prin expresia:

$$A(x) = 12x^3 - 3x^2 + 5x + 1$$

Pentru calculul derivatei  $D(x)$ :

$$D(x) = \frac{d}{dx}[A(x)] = A'(x)$$

se vor utiliza următoarele instrucțiuni:

$$A = [12 \ -3 \ 5 \ 1];$$

$$D = \text{polyder}(A);$$

care vor conduce la următorul rezultat:

$$D = [36 \ -6 \ 5]$$

corespunzător următorului polinom:

$$D(x) = 36x^2 - 6x + 5$$

Se observă gradul polinomului derivat  $n_D = n_A - 1 = 3 - 1 = 2$ .

Operația de derivare a produsului dintre două polinoame  $A(x)$  și  $B(x)$ , de grad  $n_A$  și respectiv  $n_B$ , operație definită prin  $D(x) = \frac{d}{dx}[A(x) \cdot B(x)]$ , se realizează cu ajutorul instrucțiunii:

$$D = \text{polyder}(A, B)$$

în care  $A$  și  $B$  sunt vectorii coeficienților celor două polinoame, iar  $D$  este vectorul coeficienților polinomului derivat. Gradul polinomului derivat este  $n_D = n_A + n_B - 1$ .

Spre exemplu, se consideră două polinoame  $A(x)$  și  $B(x)$ , de grad  $n_A=3$  și respectiv  $n_B=1$  definite prin:

$$A(x) = 7x^3 - 5x^2 + x - 2$$

$$B(x) = 3x - 1$$

Pentru calculul derivatei  $D(x)$  a produsului celor două polinoame:

$$D(x) = \frac{d}{dx}[A(x) \cdot B(x)] = A'(x) \cdot B(x) + A(x) \cdot B'(x)$$

se vor utiliza următoarele instrucțiuni:

$$A = [7 \ -5 \ 1 \ -2];$$

$$B = [3 \ -1];$$

$$D = \text{polyder}(A, B);$$

care va conduce la următorul rezultat:

$$D = [84 \quad -66 \quad 16 \quad -10]$$

corespunzător următorului polinom:

$$D(x) = 84x^3 - 66x^2 + 16x - 10$$

Se observă gradul polinomului derivat  $n_p = n_A + n_B - 1 = 3 + 1 - 1 = 3$ .

Operația de derivare a raportului dintre două polinoame  $A(x)$  și  $B(x)$ , de grad  $n_A$  și respectiv  $n_B$ , operație definită prin  $D(x) = d[A(x)/B(x)]/dx$ , se realizează cu ajutorul instrucțiunii:

$$[NUM, DENUM] = \text{polyder}(A, B)$$

în care  $A$  și  $B$  sunt vectorii coeficienților celor două polinoame, iar  $NUM$  și  $DENUM$  sunt vectorii coeficienților numărătorului și numitorului derivatei raportului celor două polinoame, coresponzător relației de definire a operației de derivare a raportului dintre două polinoame:

$$\frac{NUM(x)}{DENUM(x)} = \frac{d}{dx} \left[ \frac{A(x)}{B(x)} \right] = \frac{A'(x) \cdot B(x) - A(x) \cdot B'(x)}{[B(x)]^2}$$

Gradul polinoamelor obținute este  $n_{NUM} = n_A + n_B - 1$  și  $n_{DENUM} = 2n_B$ .

Spre exemplu, se consideră două polinoame  $A(x)$  și  $B(x)$ , de grad  $n_A=4$  și respectiv  $n_B=1$  definite prin:

$$A(x) = 3x^4 + 2x^3 - x^2 + 15x + 12$$

$$B(x) = x + 1$$

Pentru calculul derivatei raportului celor două polinoame se vor utiliza următoarele instrucțiuni:

$$A = [3 \quad 2 \quad -1 \quad 15 \quad 12];$$

$$B = [1 \quad 1];$$

$$[NUM, DENUM] = \text{polyder}(A, B);$$

care vor conduce la următoarele rezultate:

$$NUM = [9 \quad 16 \quad 5 \quad -2 \quad 3]$$

$$DENUM = [1 \quad 2 \quad 1]$$

corespunzătoare următoarelor două polinoame:

$$NUM(x) = 9x^4 + 16x^3 + 5x^2 - 2x + 3$$

$$DENUM(x) = x^2 + 2x + 1$$

Se observă gradul celor două polinoame obținute:

- $n_{NUM} = n_A + n_B - 1 = 4 + 1 - 1 = 4$ ;
- $n_{DENUM} = 2n_B = 2 \cdot 1 = 2$ .

### 6.1.5. Rezolvarea ecuațiilor polinomiale

Se consideră un polinom  $P_n(x)$  de grad  $n$ , definit prin:

$$P_n(x) = p_n x^n + p_{n-1} x^{n-1} + \dots + p_1 x^1 + p_0 x^0$$

în care  $p_n, p_{n-1}, \dots, p_1$  și  $p_0$  sunt coeficienții reali ai polinomului.

Ecuția polinomială de forma:

$$P_n(x) = 0$$

are  $n$  soluții care pot fi reale sau complexe. Soluțiile complexe sunt perechi de numere complex conjugate. Determinarea soluțiilor ecuației polinomiale  $P_n(x)=0$  se realizează cu instrucțiunea, [19]:

$$r = \text{roots}(P)$$

în care  $P$  este vectorul coeficienților polinomului, iar  $r$  este vectorul coloană al soluțiilor ecuației polinomiale.

#### Problema 6.2

Se consideră polinomul  $P(x)$  definit prin expresia:

$$P(x) = 3x^3 + 12x^2 - 6x + 1$$

Să se rezolve ecuația polinomială  $P(x)=0$ . Să se verifice soluțiile obținute prin metoda grafică și metoda analitică.

#### Rezolvare

Pentru calculul soluțiilor ecuației polinomiale  $P(x)=0$  se vor utiliza următoarele instrucțiuni:

$$P = [3 \ 12 \ -6 \ 1];$$

$$r = \text{roots}(P);$$

care vor conduce la următoarele rezultate:

$$r =$$

$$-4.4647$$

$$0.2323 + 0.1438i$$

$$0.2323 - 0.1438i$$

Rezultatele obținute corespund soluției reale  $x_1$  și soluțiilor complex conjugate  $x_2$  și  $x_3$ :

$$x_1 = -4.4647$$

$$x_2 = 0.2323 + 0.1438 \cdot i$$

$$x_3 = 0.2323 - 0.1438 \cdot i$$

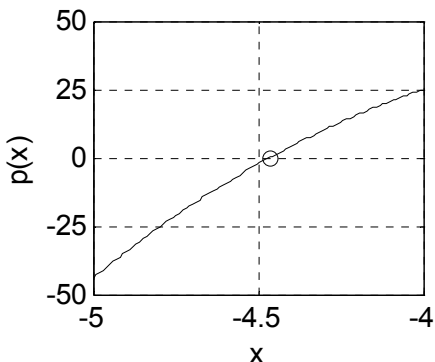
Pentru verificarea soluțiilor se poate utiliza fie metoda grafică (doar pentru soluțiile reale), fie metoda analitică (pentru toate soluțiile).

Metoda grafică constă în reprezentarea grafică a polinomului pe un domeniu de definiție care să includă soluțiile reale obținute. Pentru acest caz, soluția reală fiind  $x_1 = -4,4647$ , se poate considera ca domeniu de definiție intervalul  $x \in [-5, -4]$ .

Pentru reprezentarea grafică a polinomului analizat  $P(x)$ , pe domeniul  $[x_{min}; x_{max}] = [-5, -4]$  și pentru vizualizarea soluției reale a ecuației polinomiale  $P(x)=0$  se utilizează următoarele instrucțiuni:

```
%% POLINOAME (2)
close all;clear all;clc;
%% DATE DE INTRARE
% Coeficientii polinomului
p=[3 12 -6 1];
%% CALCULUL SOLUTIILOR
r=roots(p)
%% VERIFICAREA GRAFICA A SOLUTIILOR
% Domeniul de definitie
x=linspace(-5,-4);
% Evaluarea polinomului
px=polyval(p,x);
% Reprezentarea grafica a polinomului
% si a solutiei reale
plot(x,px,'k');
hold on;
plot(r(1),0,'ok');
grid on;
xlabel('x');
ylabel('p(x)');
axis([-5 -4 -50 50])
set(gca,'YTick',-50:25:50);
hold off;
```

Reprezentarea grafică a soluției reale a polinomului este prezentată în figura 6.2.



**Figura 6.2.** Vizualizarea soluției reale a ecuației polinomiale.

Metoda analitică de verificare a soluțiilor constă în evaluarea polinomului analizat  $P(x)$  în punctele identificate ca soluții  $x_1$ ,  $x_2$  și  $x_3$  și verificarea rezultatelor obținute care ar trebui să fie zero.

Se utilizează instrucțiunea:

```
polyval(p,r)
```

care conduce la valori foarte aproape de zero:

```
ans =
    1.0e-12 *
   -0.1519
    0.0007 + 0.0003i
    0.0007 - 0.0003i
```

### 6.1.6. Determinarea polinomului pentru un set de soluții cunoscute

Reprezintă problema inversă rezolvării ecuațiilor polinomiale și constă în determinarea aceluși polinom care are anumite rădăcini, cunoscute. În acest scop se utilizează instrucțiunea, [20]:

$$P=\text{poly}(r)$$

în care  $r$  este vectorul coloană al soluțiilor ecuației polinomiale, soluții presupuse cunoscute, iar  $P$  este vectorul coeficienților polinomului care are soluțiile  $r$ .

Spre exemplu, se consideră un polinom oarecare  $A(x)$ , care are rădăcinile cunoscute:

$$r = \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix}$$

Datorită faptului că există cinci soluții, gradul polinomului  $A(x)$  va fi cinci, numărul coeficienților acestuia fiind șase. Pentru determinarea celor șase coeficienți  $a_5, a_4, a_3, a_2, a_1$  și  $a_0$  ai polinomului  $A(x)$  definit prin

$$A_5(x) = a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + a_1x^1 + a_0x^0$$

se va utiliza următoarea instrucțiune:

$$P=\text{poly}(r)$$

care va conduce la următorul rezultat:

$$P=1 \quad -5 \quad 85 \quad -225 \quad 274 \quad -120$$

reprezentând coeficienții:

$$\begin{matrix} a_5=1 \\ a_4=-5 \\ a_3=85 \\ a_2=-225 \\ a_1=274 \\ a_0=-120 \end{matrix}$$

corespunzător următorului polinom:

$$A(x) = x^5 - 5x^4 + 85x^3 - 225x^2 + 274x - 120$$



## 6.2. REZOLVAREA ECUAȚIILOR ALGEBRICE ȘI TRANSCENDENTE

Se consideră o funcție oarecare  $f: [a; b] \rightarrow \mathbb{R}$ . Ecuația  $f(x)=0$  este o ecuație algebrică dacă poate fi adusă la o formă polinomială. În caz contrar ecuația  $f(x)=0$  se numește transcendentă.

Pentru determinarea soluțiilor ecuației:

$$f(x)=0$$

trebuie parcurse următoarele etape:

- Definirea funcției  $f(x)$  prin una din metodele recomandate:
  - Metoda fișierelor de tip `function`, cu forma generală:

```
function f=fun(x)
f=expresie
end
```

Acest fișier trebuie salvat cu numele `fun.m`.

- Metoda funcțiilor de tip `anonymous`, cu forma generală:

```
f=@(x) expresie
```

- Scrierea unui fișier de tip `script` care să asigure atât reprezentarea grafică a funcției pe domeniul său de definiție pentru identificarea aproximațiilor inițiale ale soluțiilor, cât și rezolvarea ecuației. Pentru reprezentarea grafică a funcției se utilizează instrucțiunile:

```
x=linspace(a,b);
f=fun(x);
plot(x,f);grid on;
```

De pe graficul astfel obținut se identifică aproximațiile inițiale ale tuturor soluțiilor reale ale ecuației:

$$x_{0a} = \{x_{0a1}, x_{0a2}, \dots, x_{0am}\}$$

- În cazul funcțiilor definite în fișiere de tip `function` determinarea tuturor soluțiilor ecuației se realizează cu instrucțiunea, [21]:

```
[x0, fx0]=fsolve(@fun, x0a)
```

în care: `x0` reprezintă soluțiile ecuației analizate; `fx0` reprezintă valorile funcției corespunzătoare soluțiilor obținute; `fun` este denumirea fișierului de tip `function` care conține definiția funcției de analizat, iar `x0a` reprezintă aproximațiile inițiale ale soluțiilor obținute de pe graficul funcției.

- În cazul utilizării funcțiilor de tip `anonymous`, determinarea tuturor soluțiilor ecuației se realizează cu instrucțiunea:

```
[x0, fx0]=fsolve(f, x0a)
```

- Indiferent de modul de definire al funcției de analizat, determinarea soluțiilor ecuației  $f(x)=0$  se poate face și cu instrucțiunea, [22]:

```
[x0, fx0, exitflag, output] = fzero(f, x0a, options)
```

în care  $x_0$  reprezintă soluția ecuației analizate;  $fx_0$  reprezintă valoarea funcției corespunzătoare soluției obținute;  $f$  este denumirea fișierului de tip `function` care conține definiția funcției de analizat, iar  $x_0a$  reprezintă fie aproximația inițială a unei soluții, fie un interval din jurul unei soluții, la capetele căruiua funcția are valori cu semn schimbat.

Instrucțiunea `fzero` se aplică pe rând, pentru găsirea fiecărei soluții a unei ecuații, spre deosebire de instrucțiunea `fsolve` care permite determinarea tuturor soluțiilor, la o singură apelare a instrucțiunii de rezolvare.

În vederea identificării și a rezultatelor parțiale obținute după fiecare iterație trebuie definit parametrul suplimentar:

```
options=optimset('Display','iter');
```

Dacă nu se dorește vizualizarea acestor rezultate parțiale, atunci parametrului `Display` trebuie să i se atribuie valoarea `'off'`:

```
options=optimset('Display','off');
```

Lista completă a tuturor parametrilor suplimentari, specifici instrucțiunii `fzero`, care pot fi controlați cu instrucțiunea `optimset` poate fi consultată în [23].

Parametrul `exitflag` este un identificator numeric care reprezintă motivul pentru care instrucțiunea `fzero` poate sau nu să furnizeze o soluție a ecuației. Principalele valori ale acestui parametru sunt: 1, algoritmul este convergent spre o soluție; -3, algoritmul a detectat valori NaN sau Inf; -4, algoritmul a detectat valori complexe; -5, algoritmul poate converge spre un punct singular; -6, algoritmul nu a detectat nici o schimbare de semn.

Parametrul `output` furnizează o serie de informații specifice algoritmului de căutare a soluției (numărul de iterații necesare pentru găsirea intervalului la capetele căruiua funcția prezintă o schimbare de semn, numărul de iterații necesare pentru găsirea soluției, numărul de evaluări ale funcției de analizat, algoritmul utilizat în procesul de căutare a soluției). Algoritmul utilizat de instrucțiunea `fzero` pentru găsirea soluțiilor se bazează pe metodele biseției, secantei și interpolării pătratice inverse.

### Problema 6.3

Se consideră funcția:

$$f: [0; \pi] \rightarrow \mathbb{R}$$

definită prin expresia:

$$f(x) = \sin(x^2) + \cos\left(x^2 + \frac{1}{x + \pi}\right)$$

Să se rezolve ecuația  $f(x)=0$  folosind instrucțiunile `fsolve` și `fzero`.

### Rezolvare

Pentru determinarea, pe domeniul  $[0; \pi]$ , a soluțiilor ecuației:

$$f(x) = 0$$

folosind instrucțiunea `fsolve`, trebuie parcurse următoarele etape:

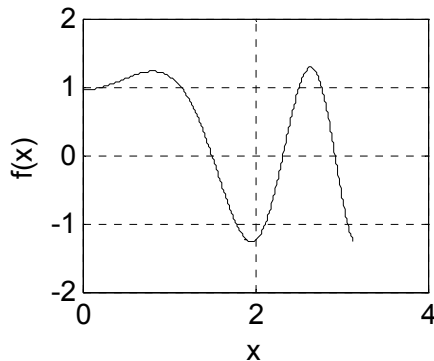
- Scrierea fișierului de tip `function` care să conțină funcția de analizat. Fișierul trebuie salvat cu numele `fun.m`.

```
function f=fun(x)
f=sin(x.^2)+cos(x.^2+1./(x+pi));
end
```

- Scrierea fișierului de tip `script` care să asigure rezolvarea ecuației din fișierul de tip `function`. Fișierul `script` are două module principale: identificarea aproximațiilor inițiale ale soluțiilor prin metoda grafică și rezolvarea și vizualizarea pe grafic a soluțiilor reale ale ecuației.

```
%% ANALIZA FUNCTIILOR TRANSCENDENTE
close all;clear all;clc;
%% APROXIMATIILE INITIALE
% Definirea domeniului
a=0;b=pi;nx=500;
x=linspace(a,b,nx);
y=fun(x);
% Reprezentarea grafica
figure
plot(x,y,'-k');hold on;
grid on;
xlabel('x');ylabel('f(x)');
pause
% Aproximatiile initiale
x0a=[1.5 2.5 3];
%% REZOLVAREA ECUAȚIEI
[x0,fx0]=fsolve(@fun,x0a)
%% REPREZENTAREA GRAFICA A SOLUTIILOR
plot(x0,fx0,'ok');
```

- Reprezentarea grafică a funcției de analizat se realizează cu o instrucțiune de forma `plot(x,y,'-k')`, în care `x` reprezintă domeniul de definiție al funcției, `y` reprezintă valorile corespunzătoare ale funcției pentru domeniul de definiție specificat, iar `'-k'` sunt parametrii de formatare ai curbei (linia continuă de culoare neagră). În acest caz, domeniul de definiție al funcției este un vector cu `nx=500` de elemente obținut cu ajutorul instrucțiunii `linspace`. Instrucțiunea `hold on` este necesară deoarece se urmărește adăugarea ulterioară, pe același grafic, a unor puncte suplimentare, reprezentând soluțiile reale ale ecuației  $f(x)=0$ .
- După parcurgerea primului modul, se introduce instrucțiunea `pause` care are rolul de a bloca temporar execuția celorlalte instrucțiuni. Pe durata suspendării execuției programului, se analizează graficul funcției (figura 6.3) căutând aproximațiile inițiale ale soluțiilor reale ale ecuației de analizat (dacă există). În acest caz, se observă existența a trei soluții reale. Pentru îmbunătățirea preciziei citirii aproximațiilor inițiale ale soluțiilor se va utiliza în mod repetat funcția grafică `zoom` în jurul fiecărei soluții reale. În acest mod, se identifică de pe grafic, următoarele valori ale aproximațiilor inițiale ale soluțiilor:  $x_{0a1}=1,5$ ;  $x_{0a2}=2,5$ ;  $x_{0a3}=2,8$ .



**Figura 6.3.** Rezolvarea ecuațiilor transcendente: identificarea aproximațiilor inițiale ale soluțiilor.

- După identificarea aproximațiilor inițiale ale soluțiilor, se revine în fișierul de tip `script` și se completează valorile numerice ale aproximațiilor inițiale ale celor trei soluții, definind astfel vectorul aproximațiilor inițiale ale soluțiilor `x0a=[1.5 2.5 2.8]`. Apoi se revine în fereastra de comenzi `Command Window` și se tastează `Space`, reluând astfel execuția celorlalte instrucțiuni care se află în cadrul fișierului de tip `script`, după instrucțiunea de suspendare temporară `pause`.

- Pentru rezolvarea și identificarea valorilor exacte ale soluțiilor ecuației transcendente  $f(x)=0$  se utilizează instrucțiunea:

```
[x0, fx0] = fsolve(@fun, x0a)
```

în care parametrii de ieșire  $x0$  și  $fx0$  reprezintă soluțiile exacte ale ecuației și valorile funcției corespunzătoare soluțiilor obținute.

- Se obțin următoarele rezultate:

```
x0 =1.4995      2.3252      2.9252
fx0 =1.0e-010 *
      0.0000    0.0000    -0.2448
```

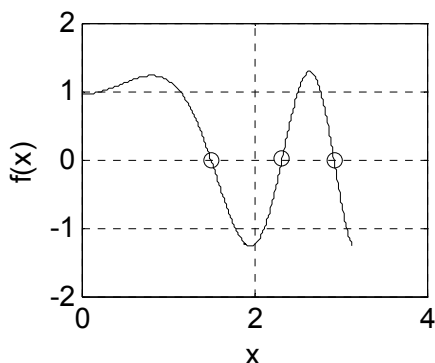
Aceste rezultate corespund soluțiilor:

$$x_{01}=1,4995; x_{02}=2,3252; x_{03}=2,9252$$

Valorile funcției  $f(x)$  corespunzătoare soluțiilor sunt:

$$f(x_{01})=0,0000 \cdot 10^{-10}; f(x_{02})=0,0000 \cdot 10^{-10}; f(x_{03})=-0,2448 \cdot 10^{-10}$$

- Ultima etapă constă în reprezentarea grafică a soluțiilor exacte folosind instrucțiunea `plot(x0, fx0, 'ok')`, figura 6.4.



**Figura 6.3.** Rezolvarea ecuațiilor transcendente: reprezentarea soluțiilor reale.

- Dacă se utilizează ca metodă de definire a funcției, varianta funcțiilor de tip anonymous, atunci în structura fișierului `script` trebuie efectuate următoarele modificări:

```
f=@(x) sin(x.^2)+cos(x.^2+1./(x+pi));
plot(x, f(x), '-k')
[x0, fx0] = fsolve(f, x0a)
```

restul instrucțiunilor rămânând neschimbate.

În cazul utilizării instrucțiunii `fzero` trebuie considerate câteva aspecte particulare:

- Definirea funcției se poate face prin oricare din metodele utilizate la instrucțiunea `fsolve` (fișiere `function` sau funcții anonime).
- Ca și în cazul instrucțiunii `fsolve` trebuie identificate aproximațiile inițiale ale soluțiilor. Pentru cazul instrucțiunii `fzero` se poate utiliza fie o valoare aproximativă a soluției, fie un interval aproximativ la capetele căruia funcția își schimbă semnul.
- Analizând graficul funcției (figura 6.3) se observă existența a trei soluții, prin urmare instrucțiunea `fzero` trebuie aplicată de trei ori.
- Spre exemplu, lansarea în execuție a instrucțiunii `fzero` având ca aproximație inițială valoarea  $x_0 = 1,45$  conduce la:

```
Search for an interval around 1.45 containing a sign change:
Func-count  a      f(a)      b      f(b)      Procedure
   1      1.45      0.180677  1.45      0.180677  initial interval
   3      1.40899  0.322803  1.49101  0.0315214 search
   5      1.392      0.379221  1.508     -0.031862 search
Search for a zero in the interval [1.392, 1.508]:
Func-count  x      f(x)      Procedure
   5          1.508     -0.031862  initial
   6          1.49901    0.00178245  interpolation
   7          1.49949    5.36692e-006  interpolation
   8          1.49949   -3.78901e-011  interpolation
   9          1.49949    2.22045e-016  interpolation
  10         1.49949    2.22045e-016  interpolation
Zero found in the interval [1.392, 1.508]
x0 =
    1.4995
fx0 =
    2.2204e-016
exitflag =
    1
output =
    intervaliterations: 2
    iterations: 5
    funcCount: 10
    algorithm: 'bisection, interpolation'
    message: 'Zero found in the interval [1.392, 1.508]'
```

- Se observă că pentru etapa identificării intervalului din jurul aproximației inițiale la capetele căruia funcția are valori cu semn schimbat au fost necesare două iterații. Pentru identificarea soluției din acest interval au fost necesare 5 iterații. În cursul acestor iterații au fost efectuate 10 evaluări ale funcției în diferite puncte până la identificarea soluției.

### 6.3. REZOLVAREA SISTEMELOR DE ECUAȚII NELINIARE

Se consideră un sistem  $S$  format din două ecuații algebrice neliniare:

$$S: \begin{cases} f(x_1, x_2) = 0 \\ g(x_1, x_2) = 0 \end{cases}$$

cu necunoscutele  $x_1$  și  $x_2$  și aproximațiile inițiale ale soluțiilor:

$$x_{0a} = [x_{0a1} \ x_{0a2}]$$

În categoria ecuațiilor algebrice neliniare intră ecuațiile transcendente și ecuațiile algebrice, cu excepția celor de gradul unu.

Pentru determinarea soluțiilor sistemului de ecuații  $S$ , trebuie parcurse următoarele etape:

- Definirea sistemului de ecuații  $S$ , prin scrierea unui fișier de tip `function` având forma generală:

```
function f=funs(x)
f=[f(x(1),x(2));g(x(1),x(2))]
```

Acest fișier trebuie salvat cu numele `funs.m`.

- Scrierea unui fișier de tip `script` care să asigure rezolvarea numerică a sistemului de ecuații. Determinarea soluțiilor sistemului de ecuații neliniare se realizează prin instrucțiunea de rezolvare:

```
[x0,fx0]=fsolve(@funs,x0a,options)
```

în care:  $x_0 = [x_{10} \ x_{20}]$  reprezintă soluțiile sistemului de ecuații;  $fx_0$  reprezintă valorile funcțiilor  $f(x_{10}, x_{20})$  și  $g(x_{10}, x_{20})$  corespunzătoare soluțiilor obținute; `funs` este denumirea fișierului de tip `function` care conține definiția sistemului de ecuații de analizat;  $x_{0a}$  reprezintă aproximațiile inițiale ale soluțiilor, iar `options` reprezintă eventualii parametri suplimentari utilizați pentru configurarea instrucțiunii `fsolve`.

În vederea identificării rezultatelor parțiale obținute după fiecare iterație parametrul opțional `options` se definește prin:

```
options=optimset('Display','iter');
```

Dacă nu se dorește vizualizarea acestor rezultate parțiale, atunci parametrului opțional `Display` trebuie să i se atribuie valoarea `'off'`:

```
options=optimset('Display','off');
```

### Problema 6.4

Să se rezolve sistemul de ecuații:

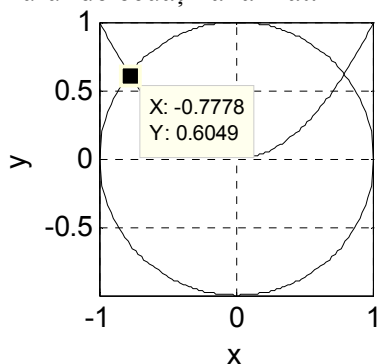
$$S: \begin{cases} x^2 + y^2 = R^2 \\ x^2 = y \end{cases}, R=1$$

### Rezolvare

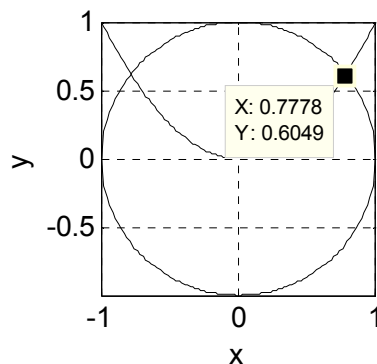
În acest caz, aproximațiile inițiale ale soluțiilor se pot determina prin metoda grafică. Se observă că sistemul de ecuații se poate aduce la forma echivalentă:

$$S: \begin{cases} y = \pm\sqrt{R^2 - x^2} \\ y = x^2 \end{cases}, R=1$$

în care prima expresie reprezintă ecuația unui cerc având centrul în originea sistemului de coordonate și raza  $R=1$ , iar a doua relație este ecuația unei parabole (figura 6.4). Se observă existența a două puncte în care cele două curbe se intersectează. Din analiza graficului se determină, cu aproximație, coordonatele celor două puncte de intersecție:  $S_1(-0,7778;0,6049)$  și  $S_2(0,7778;0,6049)$ , care reprezintă și aproximațiile inițiale ale soluțiilor sistemului de ecuații analizat.



a) primul punct de intersecție



b) al doilea punct de intersecție

**Figura 6.4.** Determinarea grafică a aproximațiilor inițiale ale soluțiilor.

Pentru rezolvarea sistemului de ecuații se vor utiliza două fișiere:

- Un fișier de tip `function` (identificat prin numele `funs.m`, în care au fost definite cele două ecuații ale sistemului analizat:

```
function F=funs(x)
R=1;
F=[x(1)^2+x(2)^2-R^2;x(1)^2-x(2)];
end
```

Ecuațiile se definesc ca linii separate ale unui vector având atâtea elemente câte ecuații există în sistemul de analizat. Se observă că ecuațiile au fost aduse la forma recomandată  $f(x_1, x_2) = 0$  și  $g(x_1, x_2) = 0$ , ceea ce pentru cazul analizat conduce la formele



$x^2 + y^2 - R^2 = 0$  și  $x^2 - y = 0$ . Se observă, de asemenea, definirea unei variabile locale pentru specificarea valorii razei cercului ( $R=1$ ). Salvarea fișierului de tip `function` trebuie să se facă sub același nume (`funs`) care este specificat în corpul funcției, ca denumire a funcției respective:  $F=funs(x)$ .

- Un fișier de tip `script` care conține aproximațiile inițiale ale soluțiilor și instrucțiunile de rezolvare ale sistemului de ecuații. În acest caz, aproximațiile inițiale ale soluțiilor au fost definite în două variabile separate `x01a` și `x02a`. Ca urmare, instrucțiunea de rezolvare se va utiliza în mod repetat pentru identificarea fiecărei soluții exacte `x01` și `x02`. Parametrii de ieșire `fx01` și `fx02` ai instrucțiunilor de rezolvare reprezintă valorile celor două funcții corespunzătoare soluțiilor exacte obținute.

```
%% REZOLVAREA SISTEMELOR DE ECUATII NELINIARE
close all;clear all;clc;
%% APROXIMATIILE INITIALE
x01a=[-0.7778 0.6049];
x02a=[0.7778 0.6049];
%% REZOLVAREA SISTEMULUI
options = optimset('Display','off');
% Prima solutie
[x01,fx01]=fsolve(@funs,x01a,options)
% A doua solutie
[x02,fx02]=fsolve(@funs,x02a,options)
```

- În urma lansării în execuție a fișierului `script` se obțin următoarele rezultate:

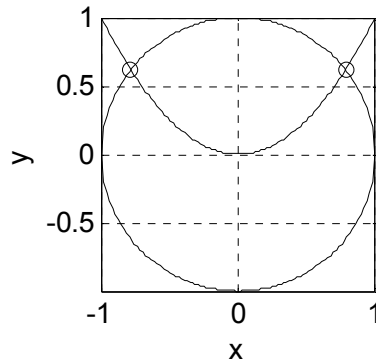
```
x01 =
    -0.7862     0.6180
fx01 =
    1.0e-007 *
     0.1512
     0.0903
x02 =
     0.7862     0.6180
fx02 =
    1.0e-007 *
     0.1512
     0.0903
```

În figura 6.5 se prezintă cele două curbe și punctele de intersecție având coordonatele exacte rezultate din rezolvarea sistemului de ecuații. Pentru realizarea reprezentării grafice s-au utilizat instrucțiunile:

```

%% REPREZENTAREA GRAFICA A SOLUTIILOR
% Determinarea intersectiei dintre un cerc si o
% parabola
clear all;close all;clc;
%% DATE INITIALE
R=1;
x=linspace(-R,R);
y1=sqrt(R^2-x.^2);
y2=-sqrt(R^2-x.^2);
y3=x.^2;
%% REPREZENTARE GRAFICA
figure
plot(x,y1,'-k');hold on;
plot(x,y2,'-k');
plot(x,y3,'k');
x01=[-0.7862 0.618];
x02=[0.7862 0.618];
plot(x01(1),x01(2),'ok');
plot(x02(1),x02(2),'ok');
grid on;
xlabel('x');ylabel('y');
axis image;hold off;

```



**Figura 6.5.** Reprezentarea grafică a soluțiilor.

### Observații

- Domeniul de definiție este un vector cu 100 de elemente, generat cu instrucțiunea `x=linspace(-R,R)`.
- Se reprezintă grafic semicercul superior  $y_1=\sqrt{R^2-x.^2}$ , semicercul inferior  $y_2=-\sqrt{R^2-x.^2}$  și parabola  $y_3=x.^2$ . Se reprezintă grafic apoi și soluțiile date prin coordonate.
- Instrucțiunea `axis image` are rolul de a introduce același factor de scală la reprezentarea ambelor axe ale graficului, astfel încât să nu se obțină deformarea cercului.

## 6.4. CALCULUL MINIMULUI ȘI MAXIMULUI UNEI FUNCȚII

### 6.4.1. Miniumul unei funcții de o variabilă

Se consideră o funcție oarecare  $f: [a; b] \rightarrow \mathbb{R}$ .

Pentru calculul minimumului funcției pe intervalul său de definiție:

$$f_{min} = \min_{a \leq x \leq b} f(x)$$

trebuie parcurse următoarele etape:

- Definirea funcției  $f(x)$  prin una din metodele recomandate:
  - Metoda fișierelor de tip `function`, cu forma generală:  

```
function f=fun(x)  
f=expresie
```

Acest fișier trebuie salvat cu numele `fun.m`.
  - Metoda funcțiilor de tip `anonymous`, cu forma generală:  

```
f=@(x) expresie
```

- Calculul abscisei minimumului funcției cu instrucțiunea, [24]:

```
xmin=fminbnd(@fun, a, b)
```

în care:  $a$  și  $b$  reprezintă limitele domeniului de analiză; `fun` este denumirea fișierului de tip `function` care conține definiția funcției de analizat, iar `xmin` reprezintă valoarea numerică a abscisei pentru care funcția are valoare minimă.

- Calculul valorii minime a funcției se efectuează cu instrucțiunea:

```
fmin=fun(xmin)
```

în care: `fun` este denumirea fișierului de tip `function` care conține definiția funcției de analizat; `xmin` reprezintă valoarea numerică a abscisei pentru care funcția are valoare minimă, iar `fmin` reprezintă valoarea minimă a funcției.

- Reprezentarea grafică a funcției pe domeniul său de definiție cu instrucțiunile:

```
x=linspace(a, b) ;  
f=fun(x) ;  
plot(x, f) ;grid on;hold on;  
plot(xmin, fmin, 'sr') ;hold off;
```

Pe graficul astfel obținut se realizează și interpretarea grafică a minimumului funcției de analizat, respectiv se reprezintă și punctul având coordonatele  $(x_{min}; f_{min})$ .

### 6.4.2. Maximul unei funcții de o variabilă

Pentru calculul maximului unei funcții trebuie făcută observația că abscisa maximului funcției  $f(x)$  coincide cu abscisa minimumului funcției  $-f(x)$ . Prin urmare se negativatează expresia funcției de analizat din fișierul de tip `function` identificat prin numele `fun.m` și se salvează fișierul sub un alt nume, de exemplu `funn.m`, după care se determină abscisa minimumului funcției negativate, care reprezintă de fapt abscisa căutăată, corespunzătoare maximului funcției inițiale.

Pentru calculul maximului unei funcții  $f(x)$  pe intervalul său de definiție  $[a, b]$ :

$$f_{max} = \max_{a \leq x \leq b} f(x)$$

trebuie parcurse următoarele etape:

- Definirea celor două fișiere de tip `function`. Dacă pentru definirea funcției  $f(x)$  se utilizează un fișier `fun.m` de tipul:

```
function f=fun(x)
f=expresie
```

atunci pentru definire funcției negativate se va utiliza un fișier `funn.m` de tipul:

```
function f=funn(x)
f=-fun(x)
```

- Calculul abscisei maximului funcției  $f(x)$  (care corespunde cu abscisa minimumului funcției  $-f(x)$ ) cu instrucțiunea:

```
xmax=fminbnd(@funn, a, b)
```

în care:  $a$  și  $b$  reprezintă limitele domeniului de analiză; `funn` este denumirea fișierului de tip `function` care conține definiția funcției de analizat negativate, iar `xmax` reprezintă valoarea numerică a abscisei pentru care funcția de analizat are valoare maximă.

- Calculul valorii maxime a funcției se efectuează cu instrucțiunea:

```
fmax=fun(xmax)
```

în care: `fun` este denumirea fișierului de tip `function` care conține definiția funcției de analizat; `xmax` reprezintă valoarea numerică a abscisei pentru care funcția are valoare maximă, iar `fmax` reprezintă valoarea maximă a funcției.

- Reprezentarea grafică a funcției pe domeniul său de definiție cu instrucțiunile:

```
x=linspace(a,b);
f=fun(x);
plot(x,f);grid on;hold on;
plot(xmax,fmax,'^k');hold off;
```

Pe graficul astfel obținut se realizează și interpretarea grafică a maximului funcției de analizat, respectiv se reprezintă și punctul având coordonatele  $(x_{max}; f_{max})$ .

- Pentru rezolvarea problemelor care implică determinarea punctelor de extrem ale unei funcții (puncte de minim și puncte de maxim) se recomandă utilizarea funcțiilor de tip anonymous (dacă funcția de analizat se poate defini printr-o singură instrucțiune):

```
f=@(x) expresie;
fn=@(x) -f(x)
xmin=fminbnd(f(x),a,b)
fmin=f(xmin)
xmax=fminbnd(fn(x),a,b)
fmax=f(xmax)
```

### Problema 6.5

Se consideră funcția:

$$f: [-5; 5] \rightarrow \mathbb{R}$$

definită prin expresia:

$$f(x) = \frac{2x^2 + 5x - 1}{5x^2 + 2}$$

Să se determine punctele de extrem ale funcției  $f(x)$  pe intervalul său de definiție  $[-5; 5]$ :

$$f_{min} = \min_{-5 \leq x \leq 5} f(x)$$

$$f_{max} = \max_{-5 \leq x \leq 5} f(x)$$

### Rezolvare

Prima metodă de rezolvare se bazează pe definirea funcției de analizat cu ajutorul fișierelor de tip `function`. Se vor utiliza trei fișiere:

- Un prim fișier de tip `function` care va conține funcția de analizat. Fișierul va fi salvat cu numele `fun.m`.

```
function f=fun(x)
f=(2*x.^2+5*x-1)/(5*x.^2+2);
end
```

- Un al doilea fișier de tip `function` care va conține funcția de analizat negativată. Fișierul va fi salvat cu numele `funn.m`.

```
function f=funn(x)
f=-fun(x);
end
```

- Un fișier de tip script care va asigura determinarea și vizualizarea grafică a extremelor funcției de analizat.

```
%% ANALIZA EXTREMELOR UNEI FUNCTII
% Metoda fisierelor function
close all;clear all;clc;
%% REPREZENTAREA GRAFICA A FUNCTIEI
%Definirea domeniului
a=-5;b=5;nx=500;x=linspace(a,b,nx);
f=fun(x);
% Reprezentarea grafica
plot(x,f,'-k');grid on;hold on;
xlabel('x');ylabel('f(x)');
pause
%% CALCULUL MINIMULUI FUNCTIEI
xmin=fminbnd(@fun,a,b)
fmin=fun(xmin)
plot(xmin,fmin,'sk');
%% CALCULUL MAXIMULUI FUNCTIEI
xmax=fminbnd(@funn,a,b)
fmax=fun(xmax)
plot(xmax,fmax,'^k');
hold off
```

Principalele module ale fișierului script sunt:

- Reprezentarea grafică a funcției pe domeniul său de definiție. Se definesc limitele domeniului de definiție ( $a=-5$ ;  $b=5$ ); se discretizează domeniul de definiție al funcției în  $nx=500$  de puncte ( $x=linspace(a,b,nx)$ ); se determină valorile funcției în cele 500 de puncte ( $y=fun(x)$ ), după care se reprezintă grafic funcția ( $plot(x,y,'-k')$ ), figura 6.6, a).
- După parcurgerea primului modul, se introduce instrucțiunea pause care are rolul de a bloca temporar execuția celorlalte instrucțiuni. Pe durata suspendării execuției programului se analizează graficul funcției (figura 6.6, a) căutând eventualele puncte de extrem. În acest caz, se observă prezența unui punct de minim și a unui punct de maxim.
- Determinarea minimului funcției și vizualizarea pe grafic a punctului de minim (figura 6.6, b). Pentru determinarea abscisei punctului de minim se utilizează instrucțiunea  $xmin=fminbnd(@fun,a,b)$ , iar pentru calculul valorii

minime a funcției se utilizează instrucțiunea `fmin=fun(xmin)`. Se obțin următoarele rezultate:

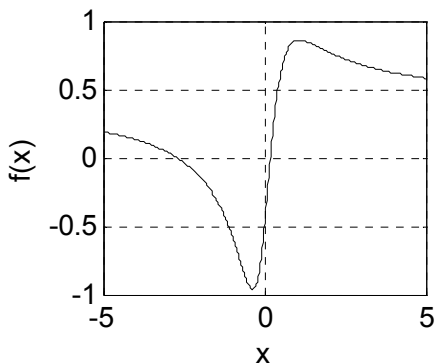
```
xmin=-0.3677  
fmin=-0.9597
```

Pentru reprezentarea pe grafic a punctului de minim se utilizează instrucțiunea `plot(xmin,fmin,'sk')`.

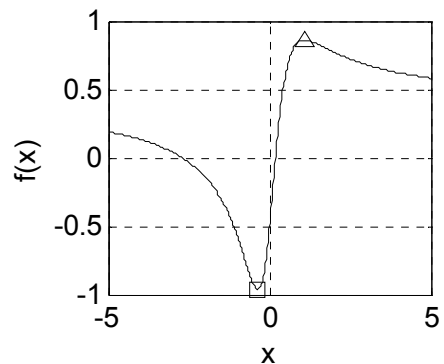
- Determinarea maximului funcției și vizualizarea pe grafic a punctului de maxim (figura 6.6, b). Pentru determinarea abscisei punctului de maxim se utilizează instrucțiunea `xmax=fminbnd(@funn,a,b)`, iar pentru calculul valorii maxime a funcției se utilizează instrucțiunea `fmax=fun(xmax)`. Se obțin următoarele rezultate:

```
xmax=1.0877  
fmax=0.8597
```

Pentru reprezentarea pe grafic a punctului de maxim se utilizează instrucțiunea `plot(xmax,fmax,'^k')`.



a) analiza preliminară a graficului funcției



b) vizualizarea punctelor de extrem

**Figura 6.6.** Analiza extremelor unei funcții.

A doua metodă de rezolvare se bazează pe utilizarea funcțiilor de tip anonymous. Se va utiliza un singur fișier de tip script:

```
%% ANALIZA EXTREMELOR UNEI FUNCTII  
% Metoda functiilor anonymous  
close all;clear all;clc;  
%% REPREZENTAREA GRAFICA A FUNCTIEI  
% Definirea domeniului  
a=-5;b=5;nx=500;x=linspace(a,b,nx);  
f=@(x) (2*x.^2+5*x-1)./(5*x.^2+2);
```

```

fn=@(x) -f(x);
% Reprezentarea grafica
plot(x,f(x),'-k');grid on;hold on;
xlabel('x');ylabel('f(x)');
pause
%% CALCULUL MINIMULUI FUNCTIEI
[xmin fmin]=fminbnd(f,a,b)
plot(xmin,fmin,'sk');
%% CALCULUL MAXIMULUI FUNCTIEI
[xmax fminn]=fminbnd(fn,a,b)
fmax=-fminn
plot(xmax,fmax,'^k');

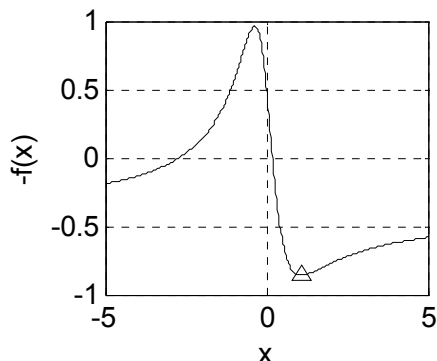
```

### Observații

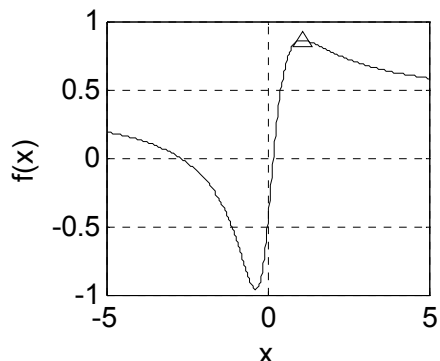
- Funcția de analizat a fost definită ca o funcție de tip anonymous prin instrucțiunea  $f=@(x) (2*x.^2+5*x-1) ./ (5*x.^2+2)$ . Tot sub forma unei funcții de tip anonymous a fost definită și funcția negativată, cu instrucțiunea  $fn=@(x) -f(x)$ .
- Determinarea coordonatelor punctului de minim se poate face cu o singură instrucțiune de forma:

```
[xmin fmin]=fminbnd(f,a,b)
```

- În cazul punctului de maxim se definește o variabilă intermediară,  $fminn$ , care reprezintă valoarea minimă a funcției negativate și care corespunde de fapt valorii maxime a funcției inițiale:  $fmax=-fminn$ .
- În figura 6.7, a) se prezintă graficul funcției negativate. Se observă că punctul de minim al funcției negativate corespunde punctului de maxim al funcției inițiale (figura 6.7, b).



a) minimul funcției negativate



b) maximul funcției inițiale

**Figura 6.7.** Analiza punctului de maxim al funcției.



## 6.5. DERIVAREA NUMERICĂ A FUNCȚIILOR

Se consideră o funcție  $f$  definită pe un interval  $[a; b]$  cu valori în  $\mathbb{R}$ ,  $f: [a; b] \rightarrow \mathbb{R}$  și fie un punct oarecare  $x_0 \in [a; b]$ . Funcția  $f$  este derivabilă în punctul  $x_0$ , dacă raportul  $\frac{f(x)-f(x_0)}{x-x_0}$  are limită finită în punctul  $x_0$ .

Această limită se numește derivata funcției  $f$  în punctul  $x_0$  și se notează  $f'(x_0)$ :

$$f'(x_0) = \lim_{x \rightarrow x_0} \frac{f(x) - f(x_0)}{x - x_0}$$

Prin relațiile:

$$\begin{cases} \Delta x = x - x_0 \\ \Delta f = f(x) - f(x_0) \end{cases}$$

se definește creșterea argumentului  $x$  în punctul  $x_0$ , respectiv creșterea funcției  $f$ , corespunzătoare creșterii argumentului de la  $x_0$  la  $x$ . În aceste condiții derivata funcției se exprimă prin:

$$f'(x_0) = \lim_{\Delta x \rightarrow 0} \frac{\Delta f}{\Delta x}$$

Cu schimbarea de variabilă:

$$x - x_0 = h$$

se obține:

$$f'(x_0) = \lim_{h \rightarrow 0} \frac{f(x_0 + h) - f(x_0)}{h}$$

Dacă derivata  $f'(x_0)$  există și este finită, atunci funcția  $f$  este derivabilă în punctul  $x_0$ .

Aproximarea numerică a derivatei unei funcții utilizând diferențele finite se poate realiza prin trei metode:

- Metoda diferențelor înainte.

$$f'(x_k) \cong \frac{f(x_{k+1}) - f(x_k)}{x_{k+1} - x_k}$$

- Metoda diferențelor înapoi.

$$f'(x_k) \cong \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}$$

- Metoda diferențelor centrale.

$$f'(x_k) \cong \frac{f(x_{k+1}) - f(x_{k-1})}{x_{k+1} - x_{k-1}}$$

Se consideră un vector  $x$  având valorile:

$$x = [x_1 \ x_2 \ x_3 \ \cdots \ x_{n-1} \ x_n]$$

Calculul diferențelor de ordinul 1 dintre fiecare două valori succesive ale vectorului  $x$ :

$$[x_2 - x_1 \ x_3 - x_2 \ \cdots \ x_n - x_{n-1}]$$

se realizează cu instrucțiunea, [25]:

`diff(x)`

Vectorul diferențelor `diff(x)` conține cu un element mai puțin decât vectorul inițial `x`.

Pentru calculul diferențelor de ordin superior (`p`) se utilizează instrucțiunea:

`diff(x,p)`

De exemplu, instrucțiunea `diff(x,2)` este echivalentă cu `diff(diff(x))`.

În cazul în care domeniul de definiție al unei funcții este conținut în vectorul `x`, iar valorile funcției sunt conținute în vectorul `f`, pentru calculul derivatei  $f'(x)$  se utilizează instrucțiunea:

`fp=diff(f)./diff(x)`

Valorile corespunzătoare ale vectorului `x` se determină în mod diferit în funcție de cele trei metode de exprimare a diferențelor:

- Pentru metoda diferențelor înainte:

`xp=x(1:n-1)`

- Pentru metoda diferențelor înapoi:

`xp=x(2:n)`

- Pentru metoda diferențelor centrale:

`xp=x(1:n-1)+diff(x)/2`

### **Problema 6.6**

Se consideră funcția  $f: [1; 2] \rightarrow \mathbb{R}$ , definită prin expresia:

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

Să se calculeze derivata de ordinul 1,  $f'(x)$  și să se reprezinte grafic folosind metodele diferențelor înainte, înapoi și centrale.

### **Rezolvare**

Rezolvarea problemei s-a făcut cu ajutorul unui fișier de tip `script` care conține următoarele instrucțiuni:

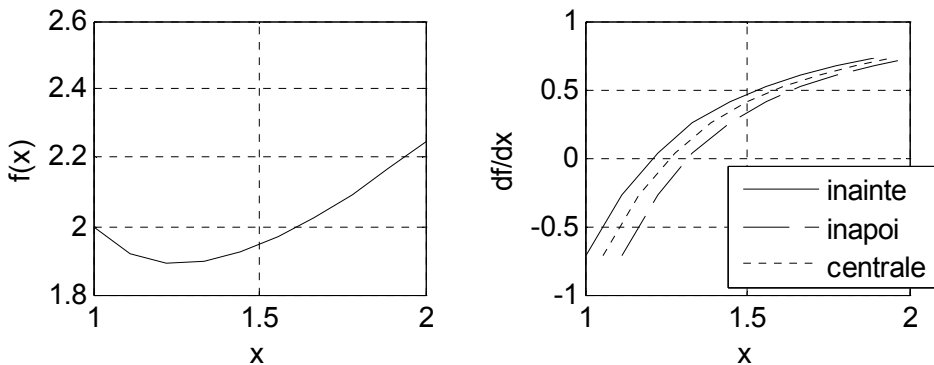
```
%% CALCULUL NUMERIC AL DERIVATEI
close all;clear all;clc;
%% DEFINIREA FUNCTIEI
xmin=1;xmax=2;nx=10;
x=linspace(xmin,xmax,nx);
```

```

f=x+1./x.^2;
%% CALCULUL NUMERIC AL DERIVATEI
df=diff(f);dx=diff(x);fp=df./dx;
% diferente inainte
xp1=x(1:nx-1);
% diferente inapoi
xp2=x(2:nx);
% diferente centrale
xp3=x(1:nx-1)+diff(x)/2;
%% REPREZENTARE GRAFICA
figure
plot(x,f,'-k');grid on;xlabel('x');ylabel('y');
figure
plot(xp1,fp,'-k');hold on;
plot(xp2,fp,'--k');plot(xp3,fp,':k');
grid on;xlabel('x');ylabel('df/dx');
legend('inainte','inapoi','centrale');hold off

```

În figura 6.8, a) se prezintă graficul funcției  $f(x)$ , iar în figura 6.8, b) graficul derivatei  $f'(x)$ , prin cele trei metode de exprimare: diferențe înainte, diferențe înapoi și diferențe centrale.



a) graficul funcției  $f(x)$

b) graficul derivatei  $f'(x)$

**Figura 6.8.** Calculul numeric al derivatei.

### Observații

- Pentru specificarea domeniului de definiție al funcției se utilizează instrucțiunea  $x=\text{linspace}(x_{\min},x_{\max},nx)$ , în care  $x_{\min}$  și  $x_{\max}$  reprezintă limitele domeniului de definiție al funcției, iar  $nx$  este numărul de elemente ale vectorului.
- Diferențele care apar între cele trei curbe, se diminuează odată cu creșterea numărului  $nx$  de puncte în care se efectuează discretizarea domeniului de definiție al funcției. Pentru acest caz s-a utilizat o discretizare a domeniului de definiție al funcției în 10 puncte.

## 6.6. INTEGRAREA NUMERICĂ A FUNCȚIILOR

### 6.6.1. Integrarea numerică a funcțiilor de o variabilă

Se consideră o funcție oarecare  $f: [a; b] \rightarrow \mathbb{R}$ .

Pentru calculul integralei definite:

$$I = \int_a^b f(x) dx$$

trebuie parcurse următoarele etape:

- Definirea funcției  $f(x)$  prin scrierea unui fișier de tip `function` având forma generală:

```
function f=fun(x)
f=expresie
```

Acest fișier trebuie salvat cu numele `fun.m`.

- Calculul integralei definite cu instrucțiunea, [26]:

```
IS=quad(@fun,a,b,tol)
```

în care:  $a$  și  $b$  reprezintă limitele domeniului de integrare; `fun` este denumirea fișierului de tip `function` care conține definiția funcției de analizat; `tol` este eroarea limită (implicit  $10^{-6}$ ), iar `IS` reprezintă valoarea numerică a integralei. Calculul efectiv al integralei se poate realiza utilizând mai multe proceduri:

- Procedura de calcul numeric `quad` reprezintă o implementare a metodei de integrare recursiv-adaptivă Simpson cu o eroare limită având valoarea  $10^{-6}$ .
- Procedura de calcul numeric `quadl` reprezintă o implementare a metodei de integrare recursiv-adaptivă Gauss-Lobatto cu o eroare limită având valoarea  $10^{-6}$ .
- Procedura de calcul numeric `quadgk` reprezintă o implementare a metodei de integrare adaptivă Gauss-Konrod cu o eroare limită de  $10^{-6}$ . Această procedură permite calculul integralelor funcțiilor care prezintă singularități la limitele finite ale domeniului de integrare. Dacă funcția prezintă o singularitate în interiorul domeniului de integrare atunci se recomandă împărțirea domeniului de integrare în două subintervale astfel încât punctul singular să fie una din limitele de integrare. Se aplică apoi procedura `quadgk` pe cele două subintervale, adunându-se valorile obținute.
- Procedura de calcul numeric `quadv` reprezintă o implementare a metodei de integrare recursiv-adaptivă Simpson cu o eroare limită având valoarea  $10^{-6}$  aplicată însă unei structuri complexe de funcții.

- o Procedura de calcul numeric trapz reprezintă o implementare a metodei trapezelor. În acest caz instrucțiunea de rezolvare este de forma, [27]:

```
IT=trapz(x,y)
```

în care:  $x=\text{linspace}(a,b)$  reprezintă discretizarea domeniului de integrare, iar  $y=\text{fun}(x)$  reprezintă valorile funcției de integrat pe domeniul de integrare. Rezultatul procedurii trapz este cu atât mai precis cu cât discretizarea domeniului de integrare este mai fină.

- Reprezentarea grafică a funcției pe domeniul său de definiție cu instrucțiunile:

```
x=linspace(a,b);
y=fun(x);
area(x,y);grid on;
```

Pe graficul astfel obținut se realizează și interpretarea grafică a integralei definite, respectiv se identifică aria mărginită de curba funcției de integrat  $y = f(x)$ , de axa Ox, precum și de cele două verticale definite prin  $x = a$  și  $x = b$ .

### Problema 6.7

Se consideră funcția:

$$f: [-\pi; \pi] \rightarrow \mathbb{R}$$

definită prin expresia:

$$f(x) = \frac{x^3 + 5x + 1}{2x^2 + 1}$$

Să se calculeze valoarea numerică a integralei definite:

$$I = \int_{-\pi}^{\pi} f(x) dx$$

### Rezolvare

Prima metodă de rezolvare se bazează pe definirea funcției de integrat cu ajutorul fișierelor de tip function. Se vor utiliza două fișiere:

- Un prim fișier, de tip function, care va conține funcția de analizat. Fișierul va fi salvat cu numele fun.m.

```
function f=fun(x)
f=(x.^3+5*x+1)/(2*x.^2+1);
end
```

În structura instrucțiunii de definire a funcției de integrat se utilizează operații de tip element-cu-element. De asemenea, se

recomandă plasarea caracterului ; la sfârșitul instrucțiunii, datorită faptului că în cursul procedurii de calcul a integralei, funcția de integrat va fi evaluată de mai multe ori, rezultând transferul unor rezultate parțiale spre fereastra de comenzi Command Window, transfer care poate fi astfel oprit.

- Un fișier de tip script, care va asigura calculul integralei definite pentru funcția din fișierul de tip function. Fișierul de tip script trebuie să permită și reprezentarea grafică a funcției de analizat.

```
%% CALCULUL INTEGRALEI DEFINITE
% Metoda fisierelor function
close all;clear all;clc;
%% DATE DE INTRARE
a=-pi;b=pi;nx=20;x=linspace(a,b,nx);
f=fun(x);tol=1e-6;
%% CALCULUL INTEGRALEI
% Metoda Simpson
IS=quad(@fun,a,b,tol)
% Metoda trapezelor
IT=trapz(x,f)
% Eroarea relative de calcul a integralei
eps=abs(IT-IS)/IS*100
%% REPREZENTARE GRAFICA
h=area(x,f);
hold on;grid on;
set(h,'FaceColor','w');
xlabel('x');ylabel('f(x)');
title(['IS=' num2str(IS) '; IT=' num2str(IT)...
'\epsilon=' num2str(eps) '%']);
set(gca,'XTick',-4:2:4);set(gca,'YTick',-3:1.5:3);
axis tight;
```

- Se observă existența în structura fișierului de tip script a patru celule:
  - Prima celulă a fișierului conține instrucțiunile generale close all, clear all și clc care asigură închiderea tuturor ferestrelor grafice, ștergerea tuturor variabilelor din spațiul de lucru MATLAB, precum și ștergerea liniilor de text de pe ecran.
  - Cea de-a doua celulă definește limitele domeniului de integrare ( $a=-\pi$  și  $b=\pi$ ), discretizarea domeniului de integrare al funcției  $x=\text{linspace}(a,b)$ , precum și calculul valorilor funcției  $f=\text{fun}(x)$ . În acest caz, s-a utilizat o discretizare a domeniului funcției în  $n_x=20$  de puncte.

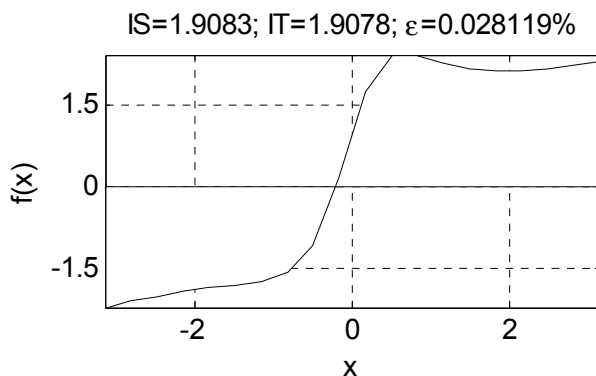
- Calculul integralei se efectuează în celula a treia cu ajutorul instrucțiunii  $I=\text{quad}(@\text{fun}, a, b)$  pentru metoda Simpson și cu instrucțiunea  $It=\text{trapz}(x, f)$  pentru metoda trapezelor. Se calculează apoi eroarea relativă la calculul integralei considerând rezultatul obținut prin metoda Simpson ca valoare convențional adevărată  $\text{eps}=\text{abs}(It - Iq) / It * 100$ . Se obțin următoarele rezultate:

```
IS =
    1.908347592911301
IT =
    1.907811139006418
eps =
    0.028118816056502
```

- Creșterea numărului de puncte de discretizare conduce la îmbunătățirea rezultatului procedurii `trapz`. De exemplu pentru  $n_x=40$  se obțin valorile:

```
IT =
    1.908222808526748
eps =
    0.006539298450658
```

- Scăderea toleranței `tol` conduce la îmbunătățirea rezultatului procedurii `quad`. De exemplu pentru  $\text{tol}=1e-12$  se obține valoarea:  $IS = 1.90834913913505$ .
- În celula a patra se reprezintă grafic suprafața mărginită de curba funcției de integrat, de axa absciselor, precum și de cele două verticale definite prin  $y = a$  și  $y = b$  folosind instrucțiunea `h=area(x, f)`, figura 6.9.



**Figura 6.9.** Calculul numeric al integralei.

A doua metodă de rezolvare se bazează pe utilizarea funcțiilor de tip anonymous. Se va utiliza un singur fișier de tip script:

```
%% CALCULUL INTEGRALEI DEFINITE
% Metoda functiilor anonymous
close all;clear all;clc;
%% DATE DE INTRARE
a=-pi;b=pi;nx=20;
x=linspace(a,b,nx);
f=@(x) (x.^3+5*x+1)./(2*x.^2+1);
tol=1e-6;
%% CALCULUL INTEGRALEI
IS=quad(f,a,b,tol)
IT=trapz(x,f(x))
eps=abs(IT-IS)/IT*100
%% Reprezentarea grafica
h=area(x,f(x));hold on;grid on;
set(h,'FaceColor','w');
xlabel('x');ylabel('f(x)');
title(['IS=' num2str(IS) ' ; IT=' num2str(IT) ...
'\epsilon=' num2str(eps) '%']);
set(gca,'XTick',-4:2:4);
set(gca,'YTick',-3:1.5:3);
axis tight;
```

### Observații

- Funcția de analizat a fost definită ca o funcție de tip anonymous prin instrucțiunea:  $f=@(x) (x.^3+5*x+1)./(2*x.^2+1)$ .
- În comparație cu fișierul script utilizat la metoda funcțiilor de tip function, în acest caz nu există alte deosebiri decât în ceea ce privește modul de apelare a funcției. Astfel, calculul integralei se realizează cu instrucțiunea  $IS=quad(f,a,b,tol)$  pentru metoda Simpson și cu instrucțiunea  $IT=trapz(x,f(x))$  pentru metoda trapezelor. De asemenea și instrucțiunea pentru realizarea reprezentării grafice diferă, în acest caz având forma:  $h=area(x,f(x))$ .
- Ca și în cazul fișierului script utilizat la metoda funcțiilor de tip function, și în acest caz localizarea liniilor ajutătoare de tip grid, precum și a valorilor de pe cele două axe sunt controlate de utilizator prin instrucțiunile  $set(gca,'XTick',-4:2:4)$  pentru abscisă și  $set(gca,'YTick',-3:1.5:3)$  pentru ordonată.
- Rezultatele obținute prin cele două metode sunt identice.



### Problema 6.8

Se consideră funcțiile, [5]:

$$f: [a; b] \rightarrow \mathbb{R} \text{ și } g: [a; b] \rightarrow \mathbb{R}$$

definite prin expresiile:

$$f(x) = 2 - x^2 \text{ și } g(x) = |x^{2/3}|$$

Să se calculeze aria  $A$  mărginită de cele două curbe  $y = f(x)$ ,  $y = g(x)$  și de cele două verticale  $x = a$ ,  $x = b$  pentru  $x \in [a; b]$ ,  $a=-1$  și  $b=1$ .

### Rezolvare

Aria mărginită de două curbe  $y = f(x)$ ,  $y = g(x)$  și de cele două verticale  $x = a$ ,  $x = b$  pentru  $f(x) \geq g(x)$  se calculează cu relația:

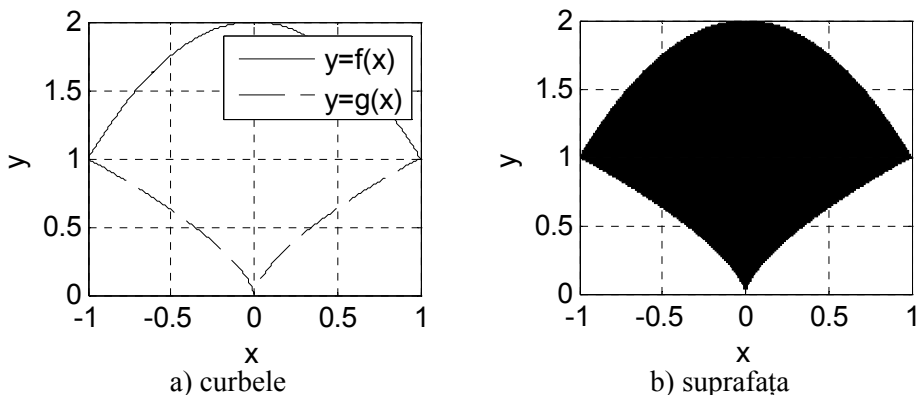
$$A = \int_a^b [f(x) - g(x)] dx$$

Fișierul script pentru rezolvarea problemei conține următoarele instrucțiuni:

```
%% CALCULUL ARIEI UNEI SUPRAFETE
% Metoda functiilor anonymous
close all;clear all;clc;
%% DATE DE INTRARE
a=-1;b=1;nx=1000;x=linspace(a,b,nx);
f=@(x) 2-x.^2;
g=@(x) abs(x.^(2/3));
u=@(x) f(x)-g(x);
%% CALCULUL SUPRAFETEI
A=quad(u,a,b)
%% REPREZENTARE GRAFICA
figure
plot(x,f(x),'-k');
hold on;grid on;
plot(x,g(x),'--k');
legend('y=f(x)','y=g(x)','Location','NorthEast');
hold off;
figure
fill([x x],[f(x) g(x)],'k');
grid on;
```

În urma lansării în execuție a fișierului se obțin reprezentările grafice din figura 6.10 (a-curbele  $y = f(x)$  și  $y = g(x)$ ; b-suprafața delimitată de cele două curbe) și următorul rezultat numeric:

$$A = 2.1333$$



**Figura 6.10.** Calculul ariei dintre două curbe.

**Observații**

- Pentru generarea domeniului de definiție s-a utilizat instrucțiunea `x=linspace(a,b,nx)` cu `nx=1000` puncte de discretizare.
- Funcțiile sunt definite folosind metoda anonymous prin instrucțiunile `f=@(x) 2-x.^2` și `g=@(x) abs(x.^(2/3))`. S-a definit și funcția diferența prin `u=@(x) f(x)-g(x)`.
- Calculul suprafeței dintre cele două curbe se reduce la calculul integralei `A=quad(u,a,b)`.
- Prima reprezentare grafică (figura 6.10, a) conține cele două curbe `plot(x,f(x),'-k')` și `plot(x,g(x),'--k')`.
- A doua reprezentare grafică (figura 6.10, b) conține suprafața dintre cele două curbe `fill([x x],[f(x) g(x)],'k')`.

**Problema 6.9**

Pentru construcția analitică a profilului aerodinamic de tip NACA 8410 se consideră următorii parametri de intrare: curbura adimensională a profilului  $\bar{f}=8\%$ ; abscisa adimensională a săgeții profilului  $\bar{x}_f=0,4$ ; grosimea relativă a profilului  $\bar{d}=0,1$ ; raza adimensională a bordului de fugă  $r_F=0,105 \cdot \bar{d}^2$ . Să se reprezinte grafic profilul NACA 8410 și să se calculeze aria profilului aerodinamic.

**Rezolvare**

Pentru determinarea coordonatelor profilului trebuie parcurse următoarele etape, [43]:

- Se definește coarda adimensională a profilului:  
$$\bar{x}=0\dots 1$$
- Se definește funcția de grosime adimensională a profilului:  
$$\bar{y}_d = \bar{d}[1,4845\sqrt{\bar{x}} - 0,63\bar{x} - 1,758\bar{x}^2 + 1,4215\bar{x}^3 - 0,5075\bar{x}^4]$$
  
Se reprezintă grafic cercurile înscrise în profil.

- Se definește ecuația adimensională a scheletului profilului:

$$\bar{y}_f = \frac{\bar{f}}{\bar{x}_f} (2\bar{x}_f \bar{x} - \bar{x}^2) \quad \text{pentru } 0 \leq \bar{x} \leq \bar{x}_f$$

$$\bar{y}_f = \frac{\bar{f}}{(1 - \bar{x}_f)^2} [1 - 2\bar{x}_f(1 - \bar{x}) - \bar{x}^2] \quad \text{pentru } \bar{x}_f \leq \bar{x} \leq 1$$

- Se calculează derivata ecuației scheletului profilului:

$$D = d\bar{y}_f/d\bar{x}$$

- Se calculează unghiul tangentei la scheletul profilului:

$$\theta = \arctan D$$

- Se calculează coordonatele intradosului profilului:

$$\bar{x}_{int} = \bar{x} + \bar{y}_d \sin \theta$$

$$\bar{y}_{int} = \bar{y}_f - \bar{y}_d \cos \theta$$

- Se calculează coordonatele extradosului profilului:

$$\bar{x}_{ext} = \bar{x} - \bar{y}_d \sin \theta$$

$$\bar{y}_{ext} = \bar{y}_f + \bar{y}_d \cos \theta$$

- Se reprezintă grafic profilul prin curba intradosului  $\bar{y}_{int}[\bar{x}_{int}(\bar{x})]$  și curba extradosului  $\bar{y}_{ext}[\bar{x}_{ext}(\bar{x})]$ . Aceste două curbe reprezintă înfășurătoarele cercurilor definite prin funcția de grosime  $\bar{y}_d(\bar{x})$ .

- Se consideră domeniul plan, interior profilului aerodinamic:

$$D = \{(x, y) \in \mathbb{R}^2: 0 \leq \bar{x} \leq 1, \bar{y}_{int} \leq y \leq \bar{y}_{ext}\}$$

Aria suprafeței mărginită de ecuația extradosului  $\bar{y}_{ext}[\bar{x}_{ext}(\bar{x})]$ , ecuația intradosului  $\bar{y}_{int}[\bar{x}_{int}(\bar{x})]$  și de cele două verticale  $\bar{x} = a$ ,  $\bar{x} = b$ ,  $a=0$ ,  $b=1$  se calculează cu relațiile:

$$A = \int_a^b \{\bar{y}_{ext}[\bar{x}_{ext}(\bar{x})] - \bar{y}_{int}[\bar{x}_{int}(\bar{x})]\} d\bar{x}$$

Pentru rezolvarea problemei se definește un fișier de tip script conținând următoarele instrucțiuni principale:

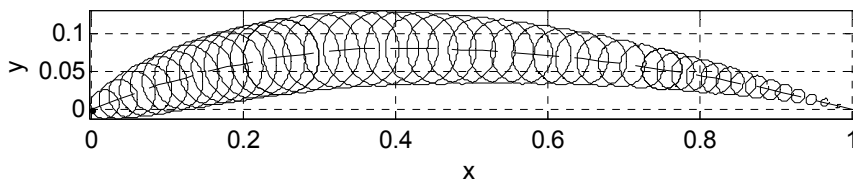
```
%% PROFIL NACA 8410
close all;clear all;clc;
%% DATE DE INTRARE
% Curbura profilului
f=0.08;
% Sageata scheletului
xf=0.4;
% Grosimea relativa a profilului
d=0.1;
% Raza bordului de fuga
rF=0.105*d^2;
%% CALCULE
% Coarda adimensionala a profilului
%x=[linspace(0,0.1,2000) linspace(0.1,0.9,100)
%linspace(0.9,1,2000)];
```

```

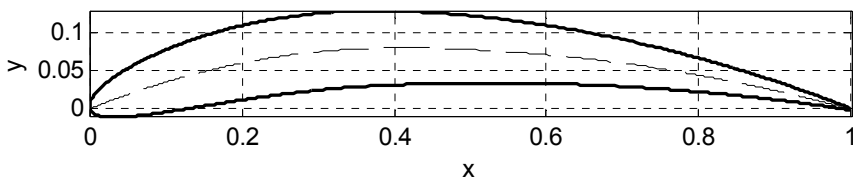
x=[linspace(0,0.25,15) linspace(0.25,0.75,20) ...
linspace(0.75,1,15)];
nx=length(x);
for i=1:nx
    % Functia de grosime a profilului
    yd(i)=d*(1.4845*sqrt(x(i))-0.63*x(i)-...
1.758*x(i).^2+1.4215*x(i).^3-0.5075*x(i).^4);
    % Scheletul si derivata scheletului
    if x(i)<=xf
        yf(i)=f/xf^2*(2*xf*x(i)-x(i).^2);
        D(i)=f/xf^2*2*(xf-x(i));
    else
        yf(i)=f/(1-xf)^2*(1-2*xf*(1-x(i))-x(i).^2);
        D(i)=f/(1-xf)^2*2*(xf-x(i));
    end
    % Unghiul tangentei la scheletul profilului
    t(i)=atan(D(i));
    % Coordonatele intradosului
    xi(i)=x(i)+yd(i).*sin(t(i));
    yi(i)=yf(i)-yd(i).*cos(t(i));
    % Coordonatele extradosului
    xe(i)=x(i)-yd(i).*sin(t(i));
    ye(i)=yf(i)+yd(i).*cos(t(i));
end
%% REPREZENTARE PROFILULUI
figure
% Scheletul
plot(x,yf,'--k');hold on;
% Extradosul
plot(xi,yi,'-k');
% Intradosul
plot(xe,ye,'-k');
% Bordul de fuga
circle(1,0,rf);
% Formatarea graficului
xlabel('x');ylabel('y');grid on;axis image;hold off;
%% REPREZENTAREA CERCURILOR INSCRISE IN PROFIL
figure
% Scheletul
plot(x,yf,'--k');hold on;
% Cercurile inscise in profil
for i=1:nx
    circle(x(i),yf(i),yd(i));
end
% Bordul de fuga
circle(1,0,rf);
% Formatarea graficului
xlabel('x');ylabel('y');grid on;axis image;hold off;
%% CALCULUL ARIEI PROFILULUI
% Metoda integrarii cu procedura trapz
A=trapz(xe,ye)-trapz(xi,yi)

```

Lansarea în execuție a fișierului script conduce la obținerea rezultatului numeric  $A=0.069285315234247$  și a reprezentărilor grafice din figura 6.11 (cercurile înscrise în profilul NACA 8410) și figura 6.12 (intradosul, extradosul și scheletul profilului NACA 8410).



**Figura 6.11.** Cercurile înscrise în profilul NACA 8410.



**Figura 6.12.** Intradosul, extradosul și scheletul profilului NACA 8410.

### Observații

- Calitatea reprezentării grafice a intradosului, extradosului și scheletului profilului depinde de numărul de puncte de discretizare a domeniului  $\bar{x}=0\dots 1$ , în special în zona bordului de atac și a bordului de fugă al profilului. Din acest motiv, coarda adimensională a profilului conține trei subdomenii: zona bordului de atac, cu 2000 de puncte de discretizare, `linspace(0,0.25,2000)`; zona centrală cu un număr mai mic de puncte, `linspace(0.25,0.75,100)` și zona bordului de fugă, tot cu 2000 de puncte de discretizare, `linspace(0.75,1,2000)`, în total 4100 de puncte.
- Pentru reprezentarea grafică a cercurilor înscrise în profil se utilizează o discretizare cu un număr mult mai mic de puncte. În acest scop s-a definit o a doua discretizare cu numai 50 de puncte din care câte 15 puncte în zona bordului de atac și a bordului de fugă și 20 de puncte în zona centrală a profilului. Reprezentarea propriu-zisă a cercurilor s-a făcut cu ajutorul unei funcții având următoarea structură:

```
function circle(x0,y0,R)
theta=linspace(0,2*pi);
x=R.*cos(theta);
y=R.*sin(theta);
plot(x+x0,y+y0,'-k')
end
```

Funcția astfel definită admite ca parametri de intrare coordonatele  $x_0$  și  $y_0$  ale centrului, precum și raza  $R$  a cercului. Funcția se apelează pentru toate punctele de discretizare ale domeniului corzii aerodinamice a profilului cu ajutorul unei instrucțiuni repetitive cu contor de forma:

```
for i=1:nx
    circle(x(i),yf(i),yd(i));
end
```

Pentru fiecare valoare a contorului ( $i=1:nx$ , în care  $nx=length(x)$  reprezintă numărul de elemente ale domeniului corzii aerodinamice a profilului) se apelează funcția `circle` transferând acesteia coordonatele curente de pe schelet  $x(i)$  și  $yf(i)$ , precum și raza curenta  $yd(i)$ .

- Fișierul `script` se execută de două ori, o dată pentru obținerea reprezentării grafice a cercurilor înscrise în profil (în acest caz se utilizează discretizarea cu număr mic de puncte și se obține reprezentarea grafică din figura 6.11) și a doua oară pentru reprezentarea grafică a intradosului, extradadosului și a scheletului profilului (în acest caz se utilizează discretizarea cu număr mare de puncte, obținându-se reprezentarea grafică din figura 6.12).
- Pentru calculul mărimilor caracteristice ale profilului (ecuația adimensională a scheletului, funcția de grosime adimensională, derivata scheletului, unghiul tangentei la scheletul profilului, coordonatele intradosului și extradadosului profilului) s-a utilizat o structură iterativă cu contor definită prin instrucțiunea repetitivă `for`. Pentru fiecare valoare a contorului ( $i=1:nx$ ), se verifică în care subdomeniu ( $0 \leq \bar{x} \leq \bar{x}_f$  sau  $\bar{x}_f \leq \bar{x} \leq 1$ ) se încadrează valoarea curentă a corzii adimensionale a scheletului cu ajutorul unei structuri alternative cu două ramuri definită prin instrucțiunea condițională `if`.
- Pentru calculul ariei profilului aerodinamic, din aria delimitată de extradados și axa abscisei, calculată cu instrucțiunea `trapz(xe,ye)`, se scade aria dintre intrados și axa abscisei, calculată cu instrucțiunea `trapz(xi,yi)`. În cazul unui profil aerodinamic simetric, cele două arii sunt egale, doar că extradadosul, fiind plasat deasupra abscisei, definește o arie pozitivă, în timp ce intradosul, plasat în întregime sub abscisă, definește o arie negativă.

### Problema 6.10

Se consideră cercul cu raza  $R=1$  având centrul plasat în punctul de coordonate  $(x_0, y_0)=(2,2)$ . Să se calculeze volumul corpurilor de revoluție obținute prin rotația cercului de ecuație  $y = f(x)$  în jurul celor două axe de coordonate  $Ox$  și  $Oy$ .

#### Rezolvare

Pentru cazul general al unei curbe  $y = f(x)$ ,  $a \leq x \leq b$ , volumul corpurilor de revoluție obținute prin rotația suprafeței plane mărginite de curba  $y = f(x)$ , axa  $Ox$  și cele două drepte verticale  $x = a$ ,  $x = b$  în jurul celor două axe de coordonate  $Ox$  și  $Oy$  se calculează cu relațiile:

$$V_x = \pi \cdot \int_a^b y^2 dx$$
$$V_y = 2\pi \cdot \int_a^b xy dx$$

În acest caz, ecuația cercului este:

$$(x - x_0)^2 + (y - y_0)^2 = R^2 \Leftrightarrow y = y_0 \pm \sqrt{R^2 - (x - x_0)^2} \Leftrightarrow$$
$$\begin{cases} y_1 = y_0 + \sqrt{R^2 - (x - x_0)^2} \\ y_2 = y_0 - \sqrt{R^2 - (x - x_0)^2} \end{cases}$$

Volumul torului obținut prin rotația curbei în jurul axei  $Ox$  se obține prin calculul integralei:

$$V_x = \pi \cdot \int_{x_0-R}^{x_0+R} [y_1^2(x) - y_2^2(x)] dx$$

Volumul torului obținut prin rotația curbei în jurul axei  $Oy$  se obține prin calculul integralei:

$$V_y = 2\pi \cdot \int_{x_0-R}^{x_0+R} x[y_1(x) - y_2(x)] dx$$

Fișierul script pentru rezolvarea problemei conține următoarele instrucțiuni:

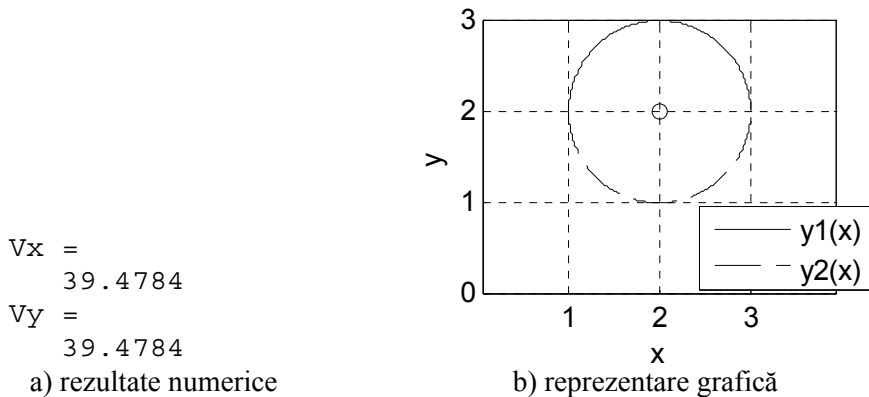
```
%% CALCULUL VOLUMULUI CORPURILOR DE REVOLUTIE
% Metoda functiilor anonymous
close all;clear all;clc;
%% DATE DE INTRARE
x0=2;y0=2;R=1;
a=x0-R;
b=x0+R;
nx=1000;
x=linspace(a,b,nx);
y1=@(x) y0+sqrt(R^2-(x-x0).^2);
y2=@(x) y0-sqrt(R^2-(x-x0).^2);
```

```

y=@(x) y1(x)-y2(x);
u=@(x) y1(x).^2-y2(x).^2;
v=@(x) x.*y(x);
%% CALCULUL VOLUMULUI
Vx=pi*quad(u,x0-R,x0+R)
Vy=2*pi*quad(v,x0-R,x0+R)
%% REPREZENTARE GRAFICA
plot(x,y1(x),'-k');
hold on;
plot(x,y2(x),'--k');
plot(x0,y0,'ok');
grid on;
xlabel('x');ylabel('y');
legend('y1(x)', 'y2(x)', 'Location', 'Best');
axis([0 x0+R 0 y0+R]);
axis equal;hold off;

```

În urma lansării în execuție a fișierului se obțin rezultatele numerice din figura 6.13, a), precum și reprezentarea grafică din figura 6.13, b).



**Figura 6.13.** Cercul generator al celor două toruri.

### Observații

- Pentru generarea domeniului de definiție s-a utilizat instrucțiunea  $x=\text{linspace}(a,b,nx)$ , în care limitele domeniului sunt  $a=x0-R$  și  $b=x0+R$ , iar numărul punctelor de discretizare este  $nx=1000$ .
- Cele două semicercuri sunt definite folosind metoda funcțiilor de tip anonymous prin instrucțiunile  $y1=@(x) y0+\text{sqrt}(R^2-(x-x0).^2)$  și  $y2=@(x) y0-\text{sqrt}(R^2-(x-x0).^2)$ . În plus, s-au definit următoarele funcții: funcția diferență  $y=@(x) y1(x)-y2(x)$ ; funcțiile care vor fi integrate  $u=@(x) y1(x).^2-y2(x).^2$  și  $v=@(x) x.*y(x)$ .



- Calculul volumului torului obținut prin rotația cercului generator în jurul axei  $Ox$  se reduce la calculul integralei  $Vx = \pi \int_{x_0-R}^{x_0+R} u^2 dx$ . S-a obținut valoarea  $Vx = 39.4784$ .
- Calculul volumului torului obținut prin rotația cercului generator în jurul axei  $Ox$  se reduce la calculul integralei  $Vy = 2\pi \int_{x_0-R}^{x_0+R} v^2 dx$ . S-a obținut valoarea  $Vy = 39.4784$ .
- Datorită simetriei centrului cercului generator  $(x_0, y_0) = (2, 2)$  față de cele două axe de coordonate  $Ox$  și  $Oy$ , cele două rezultate numerice obținute sunt identice,  $Vx = Vy$ .
- Reprezentare grafică din figura 6.13 conține cele două semicercuri  $\text{plot}(x, y1(x), 'k')$  și  $\text{plot}(x, y2(x), 'k')$ , precum și centrul cercului generator  $\text{plot}(x0, y0, 'ok')$ .

### Problema 6.11

Se consideră funcția:

$$f: [a; b] \rightarrow \mathbb{R}$$

definită prin expresia:

$$f(x) = \sqrt{1 - x^2}$$

Să se calculeze aria  $A$  mărginită de curba  $y = f(x)$  și de cele două verticale  $x = a$ ,  $x = b$  pentru  $x \in [a; b]$ ,  $a=0$  și  $b=1$ .

Să se calculeze coordonatele  $x_G$  și  $y_G$  ale centrului de greutate pentru suprafața definită de curba  $y = f(x)$  și de cele două verticale  $x = a$ ,  $x = b$ .

### Rezolvare

Aria mărginită de curba  $y = f(x)$  și de cele două verticale  $x = a$  și  $x = b$  se calculează cu relația:

$$A = \int_a^b f(x) dx$$

Coordonatele centrului de greutate pentru suprafața definită de curba  $y = f(x)$  și de cele două verticale  $x = a$  și  $x = b$  se calculează cu relațiile:

$$\begin{cases} x_G = \frac{1}{A} \int_a^b xy dx \\ y_G = \frac{1}{2A} \int_a^b y^2 dx \end{cases}$$

Fișierul script pentru rezolvarea problemei conține următoarele instrucțiuni:

```
%% CALCULUL CENTRULUI DE GREUTATE (1)
% Metoda functiilor anonymous
close all;clear all;clc;
```

```

%% DATE DE INTRARE
a=0;b=1;
f=@(x) sqrt(1-x.^2);
fxg=@(x) x.*f(x);
fyg=@(x) f(x).^2;
%% CALCULUL ARIEI SUPRAFETEI
A=quad(f,a,b)
%% CALCULUL COORDONATELOR CENTRULUI DE GREUTATE
xG=1/A*quad(fxg,a,b)
yG=1/(2*A)*quad(fyg,a,b)
%% REPREZENTARE GRAFICA
figure
nx=1000;x=linspace(a,b,nx);
plot(x,f(x),'-k','LineWidth',2);hold on;
plot(xG,yG,'ok','LineWidth',2);
xlabel('x');ylabel('y');
grid on;axis image;hold off;

```

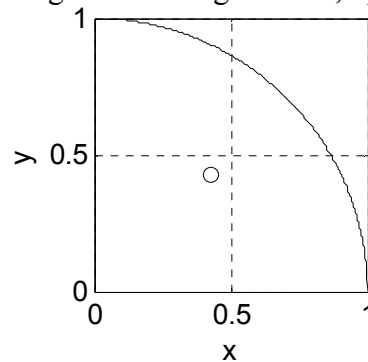
În urma lansării în execuție a fișierului se obțin rezultatele numerice din figura 6.14, a), precum și reprezentarea grafică din figura 6.14, b).

```

A =
    0.7854
xG =
    0.4244
yG =
    0.4244

```

a) rezultate numerice



b) reprezentare grafică

**Figura 6.14.** Centrul de greutate al unui sfert de cerc.

### Observații

- Se observă că suprafața analizată este reprezentată de un sfert de cerc, cu centrul în originea sistemului de coordonate și cu raza  $R=1$ , pentru care se cunosc expresiile ariei  $A = \pi R^2/4$  și coordonatelor centrului de greutate  $x_G = y_G = 4R/(3\pi)$ . Pornind de la aceste expresii și efectuând calculele se obțin valorile exacte  $A=0,7854$  și  $x_G = y_G=0,4244$ , verificându-se astfel valorile numerice aproximative obținute prin calculul integralelor (figura 6.14, a).
- Se reprezintă grafic în aceeași figură și în aceleași axe atât curba `plot(x,f(x),'-k')` cât și centrul de greutate al suprafeței `plot(xG,yG,'ok')`.

### Problema 6.12

Se consideră funcțiile:

$$f: [a; b] \rightarrow \mathbb{R} \text{ și } g: [a; b] \rightarrow \mathbb{R}$$

definite prin expresiile:

$$f(x) = \sqrt{x} \text{ și } g(x) = x^2$$

Să se calculeze aria  $A$  mărginită de curbele  $y = f(x)$ ,  $y = g(x)$  și de cele două verticale  $x = a$ ,  $x = b$  pentru  $x \in [a, b]$ ,  $a=0$  și  $b=1$ .

Să se calculeze coordonatele  $x_G$  și  $y_G$  ale centrului de greutate pentru suprafața definită de curbele  $y = f(x)$ ,  $y = g(x)$  și de cele două verticale  $x = a$ ,  $x = b$ .

### Rezolvare

Aria mărginită de două curbe  $y = f(x)$ ,  $y = g(x)$  și de cele două verticale  $x = a$ ,  $x = b$  pentru  $f(x) \geq g(x)$  se calculează cu relația:

$$A = \int_a^b [f(x) - g(x)] dx$$

Coordonatele centrului de greutate pentru suprafața definită de cele două curbe  $y = f(x)$ ,  $y = g(x)$  și de cele două verticale  $x = a$  și  $x = b$  se calculează cu relațiile:

$$\begin{cases} x_G = \frac{1}{A} \int_a^b x[f(x) - g(x)] dx \\ y_G = \frac{1}{2A} \int_a^b [f^2(x) - g^2(x)] dx \end{cases}$$

Fișierul script pentru rezolvarea problemei conține următoarele instrucțiuni:

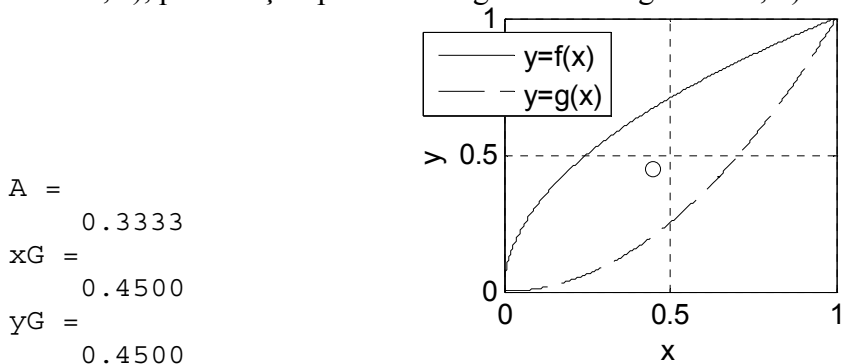
```
%% CALCULUL CENTRULUI DE GREUTATE (2)
% Metoda functiilor anonymous
close all;clear all;clc;
%% DATE DE INTRARE
a=0;b=1;
f=@(x) sqrt(x);g=@(x) x.^2;
fa=@(x) f(x)-g(x);
fxg=@(x) x.*(f(x)-g(x));
fyg=@(x) f(x).^2-g(x).^2;
%% CALCULUL ARIEI SUPRAFETEI
A=quad(fa,a,b)
%% CALCULUL COORDONATELOR CENTRULUI DE GREUTATE
xG=1/A*quad(fxg,a,b);
yG=1/(2*A)*quad(fyg,a,b)
%% REPREZENTARE GRAFICA
figure
nx=1000;x=linspace(a,b,nx);
```

```

plot(x,f(x),'-k','LineWidth',2);hold on;
plot(x,g(x),'--k','LineWidth',2)
plot(xG,yG,'ok','LineWidth',2);
xlabel('x');ylabel('y');grid on;hold off;
legend('y=f(x)','y=g(x)');

```

În urma lansării în execuție a fișierului se obțin rezultatele numerice din figura 6.15, a), precum și reprezentarea grafică din figura 6.15, b).



a) rezultate numerice

b) reprezentare grafică

**Figura 6.15.** Centrul de greutate al unui suprafețe compuse.

### Observații

- Cele două curbe  $y = f(x)$  și  $y = g(x)$  sunt definite folosind metoda funcțiilor de tip anonymous prin instrucțiunile  $f=@(x)$   $\sqrt{x}$  și  $g=@(x)$   $x.^2$ . În plus, s-au definit următoarele funcții: funcția diferență necesară pentru calculul ariei suprafeței  $fa=@(x)$   $f(x)-g(x)$ ; funcțiile necesare pentru calculul coordonatelor centrului de greutate  $fxg=@(x)$   $x.*(f(x)-g(x))$  și  $fyg=@(x)$   $f(x).^2-g(x).^2$ .
- Calculul ariei suprafeței dintre cele două curbe se face cu instrucțiunea  $A=quad(fa,a,b)$ .
- Calculul coordonatelor centrului de greutate se face cu instrucțiunile  $xG=1/A*quad(fxg,a,b)$  și  $yG=1/(2*A)*quad(fyg,a,b)$ .
- Pentru generarea domeniului de definiție s-a utilizat instrucțiunea  $x=linspace(a,b,nx)$  în care limitele domeniului sunt  $a=0$  și  $b=1$ , iar numărul punctelor de discretizare este  $nx=1000$ .
- Se reprezintă grafic în aceeași figură și în aceeași axe atât cele două curbe  $plot(x,f(x),'-k')$  și  $plot(x,g(x),'--k')$ , cât și centrul de greutate al suprafeței  $plot(xG,yG,'ok')$ . În acest scop după prima instrucțiune  $plot$  se introduce instrucțiunea  $hold on$ . Pentru identificarea clară a celor două curbe se utilizează instrucțiunea  $legend('y=f(x)','y=g(x)')$ .

### 6.6.2. Calculul numeric al integralelor multiple

Se consideră o funcție  $f(x_1, x_2, \dots, x_n)$  definită pe un domeniu mărginit  $D \subset \mathbb{R}^n$ . Integrala multiplă pe domeniul  $D$  se exprimă prin:

$$I = \int \dots \int_D f(x_1, x_2, \dots, x_n) dx_1 dx_2 \dots dx_n$$

Calculul integralelor multiple se reduce la calculul unei serii de integrale simple conform relației, [38]:

$$I = \int_{a_n}^{b_n} \dots \left( \int_{a_2}^{b_2} \left( \int_{a_1}^{b_1} f(x_1, x_2, \dots, x_n) dx_1 \right) dx_2 \right) \dots dx_n$$

în care  $a_1 \leq x_1 \leq b_1, a_2 \leq x_2 \leq b_2, \dots, a_n \leq x_n \leq b_n$ .

**Integrale duble.** Pentru  $n=2$  și folosind notațiile  $x_1 = x, x_2 = y$ , se obține cazul particular al integralelor duble pe domenii mărginite  $D \subset \mathbb{R}^2$ :

$$I = \iint_D f(x, y) dx dy$$

Calculul integralelor duble se face în funcție de modul de exprimare al domeniului de integrare [3, 39]:

- Dacă domeniul de integrare  $D$  este definit prin:

$$a \leq x \leq b \text{ și } c(x) \leq y \leq d(x), \forall x \in [a; b]$$

(domeniu simplu în raport cu axa  $Oy$ ) atunci:

$$I = \iint_D f(x, y) dx dy = \int_a^b \left( \int_{c(x)}^{d(x)} f(x, y) dy \right) dx$$

- Dacă domeniul de integrare  $D$  este definit prin:

$$c \leq y \leq d \text{ și } a(y) \leq x \leq b(y), \forall y \in [c; d]$$

(domeniu simplu în raport cu axa  $Ox$ ) atunci:

$$I = \iint_D f(x, y) dx dy = \int_c^d \left( \int_{a(y)}^{b(y)} f(x, y) dx \right) dy$$

- În cazul particular al unui domeniu dreptunghiular:

$$a \leq x \leq b \text{ și } c \leq y \leq d$$

atunci:

$$I = \int_a^b \left( \int_c^d f(x, y) dy \right) dx = \int_c^d \left( \int_a^b f(x, y) dx \right) dy$$

Dacă într-o integrală dublă se face schimbarea de variabile, [8]:

$$\begin{cases} x = \alpha(u, v) \\ y = \beta(u, v) \end{cases}$$

prin care domeniul  $D_{x,y}$  se transformă în domeniul  $D_{u,v}$ , atunci:

$$\iint_{D_{x,y}} f(x, y) dx dy = \iint_{D_{u,v}} f(\alpha(u, v), \beta(u, v)) \cdot |J(u, v)| du dv$$

în care  $J(u, v) \neq 0$  este Jacobianul transformării definit prin:

$$J(u, v) = \begin{vmatrix} \frac{\partial \alpha}{\partial u} & \frac{\partial \alpha}{\partial v} \\ \frac{\partial \beta}{\partial u} & \frac{\partial \beta}{\partial v} \end{vmatrix}$$

Pentru cazul particular al transformării din coordonate carteziene  $(x, y)$  în coordonate polare  $(r, \theta)$ , schimbarea de variabile corespunde relațiilor:

$$\begin{cases} x = r \cos \theta \\ y = r \sin \theta \end{cases}$$

pentru care Jacobianul transformării este:

$$J(r, \theta) = \begin{vmatrix} \cos \theta & -r \sin \theta \\ \sin \theta & r \cos \theta \end{vmatrix} = r$$

În aceste condiții, integrala dublă se calculează cu relația:

$$\iint_{D_{x,y}} f(x, y) dx dy = \iint_{D_{r,\theta}} f(r \cos \theta, r \sin \theta) r dr d\theta$$

Pentru calculul numeric al integralei duble:

$$I = \iint_{D_{x,y}} f(x, y) dx dy$$

pe un domeniu de integrare  $D_{x,y}$  dreptunghiular de forma

$$\begin{cases} a \leq x \leq b \\ c \leq y \leq d \end{cases}$$

în funcție de modul de definire a funcției de integrat se folosește una din instrucțiunile, [28]:

```
I=dblquad(f, a, b, c, d, tol)
I=dblquad(@fun, a, b, c, d, tol)
```

în care:  $a \leq x \leq b$  și  $c \leq y \leq d$  reprezintă limitele domeniului de integrare;  $f$  este denumirea funcției de integrat definită ca funcție de tip anonymous;  $fun$  este denumirea fișierului de tip `function` care conține definiția funcției de integrat;  $tol$  este eroarea limită (implicit  $10^{-6}$ ), iar  $I$  reprezintă valoarea numerică a integralei.

În cazul unui domeniu de integrare  $D_{x,y}$  de forma:

$$\begin{cases} a \leq x \leq b \\ c(x) \leq y \leq d(x) \end{cases}$$

în funcție de modul de definire a funcției de integrat se folosește una din instrucțiunile [29, 30]:

```
I=quad2d(f, a, b, c, d)
I=quad2d(@fun, a, b, c, d)
I=integral2(f, a, b, c, d)
I=integral2(@fun, a, b, c, d)
```

### Problema 6.13

Să se reprezinte grafic domeniul de integrare și să se calculeze următoarea integrală dublă:

$$I = \iint_D f(x, y) dx dy$$
$$D = \{(x, y) \in \mathbb{R}^2: -1 \leq x \leq 1, -1 \leq y \leq 1\}$$
$$f(x, y) = \sqrt{x^2 + y^2}$$

### Rezolvare

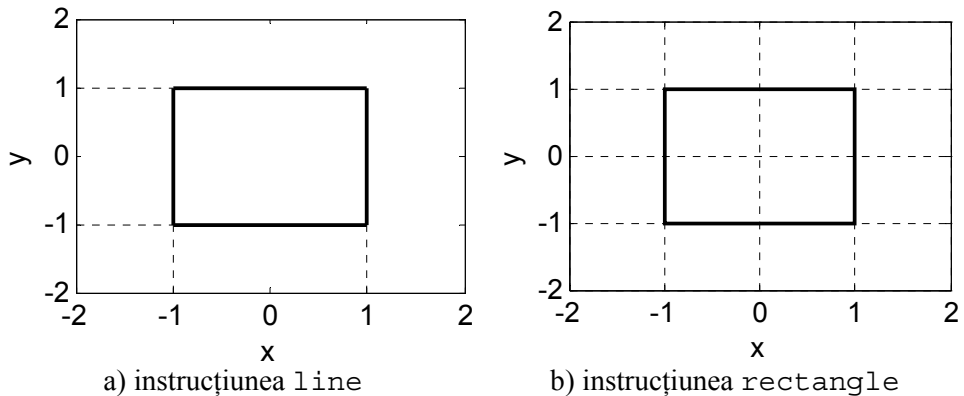
Se utilizează metoda de rezolvare bazată pe utilizarea funcțiilor de tip anonymous. Se va utiliza un singur fișier de tip script:

```
%% CALCULUL INTEGRALEI DUBLE (1)
% Metoda functiilor anonymous
close all;clear all;clc;
%% DATE DE INTRARE
% Limitele variabilei x
a=-1;b=1;
% Limitele variabilei y
c=-1;d=1;
% Functia de integrat
f=@(x,y) sqrt(x.^2+y.^2);
%% CALCULUL INTEGRALEI DUBLE
I=dblquad(f,a,b,c,d)
%% REPREZENTAREA DOMENIULUI
nx=20;x=linspace(a,b,nx);
ny=20;y=linspace(c,d,ny);
figure
xmin=-2;xmax=2;
ymin=-2;ymax=2;
axis([xmin xmax ymin ymax]);box on;
for i=[1 nx]
    line([x(i) x(i)], [y(1) y(ny)], 'Color', 'k');hold on;
    line([x(i) x(i)], [ymin y(1)], 'LineStyle', ':', 'Color', 'k');
end
for j=[1 ny]
    line([x(1) x(nx)], [y(j) y(j)], 'Color', 'k');
    line([xmin x(1)], [y(j) y(j)], 'LineStyle', ':', 'Color', 'k');
end
xlabel('x');ylabel('y');
```

În urma execuției fișierului script se obține valoarea numerică a integralei duble:

$$I = 3.0608$$

și reprezentarea grafică a domeniului de integrare din figura 6.16, a).



a) instrucțiunea line                      b) instrucțiunea rectangle

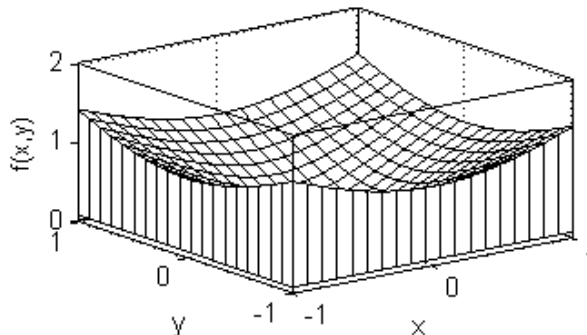
**Figura 6.16.** Reprezentarea domeniului dreptunghiular de integrare.

**Observații**

- Limitele domeniului de variație sunt definite prin instrucțiunile  $a=-1$ ;  $b=1$  pentru variabila  $x$  și prin instrucțiunile  $c=-1$ ;  $d=1$  pentru variabila  $y$ .
- Funcția de integrat este definită ca o funcție de tip anonymous prin instrucțiunea  $f=@(x,y) \text{ sqrt}(x.^2+y.^2)$ .
- Calculul numeric al integralei duble se realizează cu instrucțiunea  $I=\text{dblquad}(f, a, b, c, d)$ .
- Reprezentarea grafică a domeniului de integrare se face prin desenarea liniilor domeniului (comanda `line`, figura 6.16, a).
- Pentru domenii dreptunghiulare reprezentarea grafică se poate obține mai simplu folosind instrucțiunea `rectangle`, figura 6.16, b):

```
rectangle('Position',[a c b-a d-c], 'LineWidth',2);
axis([xmin xmax ymin ymax]);
box on;grid on;
```

- Integrala dublă  $I = \iint_D f(x,y) dx dy$  reprezintă volumul cilindroidului  $C$  (figura 6.17) definit prin mulțimea punctelor din spațiu  $C = \{(x, y, z) \in \mathbb{R}^3: (x, y) \in D, 0 \leq z \leq f(x, y)\}$ .



**Figura 6.17.** Cilindroidul  $C$ .



### Problema 6.14

Să se reprezinte grafic domeniul de integrare și să se calculeze următoarea integrală dublă:

$$I = \iint_D f(x, y) \, dx dy$$
$$D = \left\{ (x, y) \in \mathbb{R}^2: -1 \leq x \leq 1, -\sqrt{1-x^2} \leq y \leq \sqrt{1-x^2} \right\}$$
$$f(x, y) = \sqrt{1-x^2-y^2}$$

### Rezolvare

Se utilizează metoda de rezolvare bazată pe utilizarea funcțiilor de tip anonymous. Se va utiliza un singur fișier de tip script:

```
%% CALCULUL INTEGRALEI DUBLE (2)
% Metoda functiilor anonymous
close all;clear all;clc;
%% DATE DE INTRARE
% Limitele variabilei x
a=-1;b=1;
% Limitele variabilei y
c=@(x) -sqrt(1-x.^2);
d=@(x) sqrt(1-x.^2);
% Functia de integrat
f=@(x,y) sqrt(1-x.^2-y.^2);
%% CALCULUL INTEGRALEI DUBLE
I=quad2d(f,a,b,c,d)
%% REPREZENTAREA DOMENIULUI
figure
nx=30;x=linspace(a,b,nx);
ny=30;y=linspace(min(c(x)),max(d(x)),ny);
plot(x,c(x),'k','LineWidth',2);hold on;
plot(x,d(x),'k','LineWidth',2)
xlabel('x');ylabel('y');grid on;axis equal;
```

În urma execuției fișierului script se obține valoarea numerică a integralei duble:

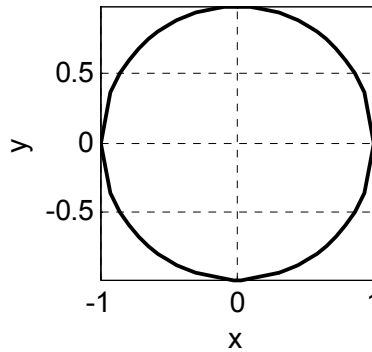
$$I = 2.0944$$

și reprezentarea grafică a domeniului de integrare din figura 6.18.

### Observații

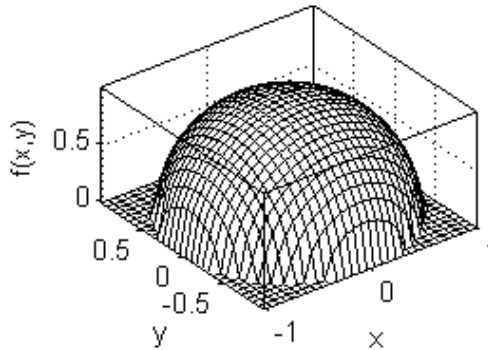
- Limitele de variație ale domeniului pentru variabila  $x$  sunt definite prin instrucțiunile  $a=-1$ ;  $b=1$ .

- Limitele de variație ale domeniului pentru variabila  $y$  sunt funcții care depind de variabila  $x$  și se definesc cu ajutorul a două funcții de tip anonymous prin instrucțiunile  $c=@(x) -\text{sqrt}(1-x.^2)$  și  $d=@(x) \text{sqrt}(1-x.^2)$ .
- Funcția de integrat este definită ca o funcție de tip anonymous prin instrucțiunea  $f=@(x,y) \text{sqrt}(1-x.^2-y.^2)$ .
- Calculul numeric al integralei duble se realizează cu instrucțiunea  $I=\text{quad2d}(f,a,b,c,d)$ .



**Figura 6.18.** Reprezentarea domeniului de integrare.

- Integrala dublă  $I = \iint_D f(x,y) dx dy$  reprezintă volumul cilindroidului  $C$  (figura 6.19) definit prin mulțimea punctelor din spațiu  $C = \{(x,y,z) \in \mathbb{R}^3: (x,y) \in D, 0 \leq z \leq f(x,y)\}$ .



**Figura 6.19.** Cilindroidul  $C$ .

- Se observă că cilindroidul  $C$  este în acest caz semisfera cu centrul în originea sistemului de coordonate, cu raza  $R$  egală cu unitatea și plasată deasupra domeniului  $D$ . Considerând expresia cunoscută a volumului semisferei  $4\pi R^3/6$  și efectuând calculele, se obține valoarea exactă 2,0944, verificându-se astfel valoarea aproximativă obținută prin calculul integralei duble.

### Problema 6.15

Se consideră domeniul plan definit prin:

$$D = \{(x, y) \in \mathbb{R}^2: 0 \leq x \leq \pi, \sin x \leq y \leq \pi \cdot \sin x\}$$

Să se calculeze aria  $A$  și coordonatele  $x_G$  și  $y_G$  ale centrului de greutate pentru domeniul  $D$ .

### Rezolvare

Aria domeniul  $D$  corespunde suprafeței mărginite de curbele  $c(x) = \sin x$  și  $d(x) = \pi \cdot \sin x$  și de cele două verticale  $x = a$  și  $x = b$ ,  $a=0$  și  $b=\pi$ . Folosind integralele duble, această arie se calculează cu relația:

$$A = \iint_D dA = \int_a^b \left( \int_{c(x)}^{d(x)} dy \right) dx$$

Coordonatele  $x_G$  și  $y_G$  ale centrului de greutate pentru domeniul  $D$  se calculează cu relațiile:

$$\begin{cases} x_G = \frac{1}{A} \iint_D x dA \\ y_G = \frac{1}{A} \iint_D y dA \end{cases}$$

Fișierul script pentru rezolvarea problemei conține următoarele instrucțiuni:

```
%% CALCULUL INTEGRALEI DUBLE (3)
% Metoda functiilor anonymous
close all;clear all;clc;
%% DATE DE INTRARE
% Limitele variabilei x
a=0;b=pi;
% Limitele variabilei y
c=@(x) sin(x);
d=@(x) pi*sin(x);
% Functia de integrat
f=@(x,y) (x) ./ (x);
fxg=@(x,y) x;
fyg=@(x,y) y;
%% CALCULUL ARIEI DOMENIULUI
A=quad2d(f,a,b,c,d)
%% CALCULUL COORDONATELOR
xG=1/A*quad2d(fxg,a,b,c,d)
yG=1/A*quad2d(fyg,a,b,c,d)
%% REPREZENTAREA DOMENIULUI
figure
nx=1000;x=linspace(a,b,nx);
plot(x,c(x),'k-');hold on;
```

```

plot(x,d(x),'k--');
plot(xG,yG,'ok');
xlabel('x');ylabel('y');grid on;hold off;
legend('y=c(x)','y=d(x)');

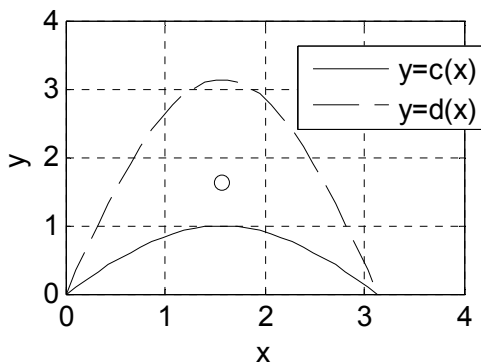
```

În urma lansării în execuție a fișierului se obțin rezultatele numerice din figura 6.20, a), precum și reprezentarea grafică din figura 6.20, b).

```

A =
    4.2832
xG =
    1.5708
yG =
    1.6264

```



a) rezultate numerice

b) reprezentare grafică

**Figura 6.20.** Centrul de greutate al unui suprafețe compuse.

### Observații

- Cele două curbe  $y = c(x)$  și  $y = d(x)$  sunt definite folosind metoda funcțiilor de tip anonymous prin instrucțiunile  $c=@(x) \sin(x)$  și  $d=@(x) \pi*\sin(x)$ . În plus, s-au definit următoarele funcții: funcția identitate necesară pentru calculul ariei  $f=@(x,y) (x) ./ (x)$ ; funcțiile necesare pentru calculul coordonatelor centrului de greutate  $fxg=@(x,y) x$  și  $fyg=@(x,y) y$ .
- Calculul ariei domeniului  $D$  se face cu instrucțiunea  $A=quad2d(f,a,b,c,d)$ .
- Calculul coordonatelor  $x_G$  și  $y_G$  ale centrului de greutate se face cu instrucțiunile  $xG=1/A*quad2d(fxg,a,b,c,d)$  și  $yG=1/A*quad2d(fyg,a,b,c,d)$ .
- Pentru generarea domeniului de definiție s-a utilizat instrucțiunea  $x=linspace(a,b,nx)$  în care limitele domeniului sunt  $a=0$  și  $b=\pi$ , iar numărul punctelor de discretizare este  $nx=1000$ .
- Se reprezintă grafic în aceeași figură și în aceeași axe atât cele două curbe  $\text{plot}(x,c(x),'-k')$  și  $\text{plot}(x,d(x),'--k')$  cât și centrul de greutate al suprafeței  $\text{plot}(xG,yG,'ok')$ . În acest scop după prima instrucțiune  $\text{plot}$  se introduce instrucțiunea  $\text{hold on}$ . Pentru identificarea clară a celor două curbe se utilizează instrucțiunea  $\text{legend}('y=c(x)','y=d(x)')$ .

### Problema 6.16

Pentru construcția analitică a profilului aerodinamic de tip NACA65-0415 se consideră următorii parametri de intrare: coeficientul adimensional de curbură  $c_{p0}=0,4$ ; grosimea relativă a profilului  $\bar{d}=0,15$ .

Să se reprezinte grafic profilul NACA65-0415, să se calculeze aria profilului aerodinamic și coordonatele centrului de greutate folosind comparativ, exprimarea prin intermediul integralelor simple și duble.

### Rezolvare

Pentru determinarea coordonatelor profilului trebuie parcurse următoarele etape, [43]:

- Se definește coarda adimensională a profilului:

$$\bar{x} = 0 \cdots 1$$

- Se definește ecuația adimensională a scheletului profilului:

$$\bar{y}_f = -\frac{c_{p0}}{4\pi} [(1 - \bar{x}) \ln(1 - \bar{x}) + \bar{x} \ln \bar{x}]$$

- Funcția de grosime a profilului reprezintă raza cercurilor înscrise în profil. Se definește funcția de grosime adimensională a profilului:

$$\bar{y}_d = \bar{d} \frac{1 - \bar{x}}{1 - 0,176\bar{x}} [1,0675\sqrt{\bar{x}} - 0,2758\bar{x} + 2,4478\bar{x}^2 - 2,8385\bar{x}^3]$$

Se reprezintă grafic cercurile înscrise în profil.

- Se calculează derivata ecuației scheletului profilului:

$$D = \frac{d\bar{y}_f}{d\bar{x}} = -\frac{c_{p0}}{4\pi} [-\ln(1 - \bar{x}) + \ln \bar{x}]$$

- Se calculează unghiul tangentei la scheletul profilului:

$$\theta = \arctan D$$

- Se calculează coordonatele intradosului profilului:

$$\bar{x}_{int} = \bar{x} + \bar{y}_d \sin \theta$$

$$\bar{y}_{int} = \bar{y}_f - \bar{y}_d \cos \theta$$

- Se calculează coordonatele extradadosului profilului:

$$\bar{x}_{ext} = \bar{x} - \bar{y}_d \sin \theta$$

$$\bar{y}_{ext} = \bar{y}_f + \bar{y}_d \cos \theta$$

- Se reprezintă grafic profilul prin curba intradosului  $\bar{y}_{int}[\bar{x}_{int}(\bar{x})]$  și curba extradadosului  $\bar{y}_{ext}[\bar{x}_{ext}(\bar{x})]$ . Aceste două curbe reprezintă înfășurătoarele cercurilor definite prin funcția de grosime  $\bar{y}_d(\bar{x})$ .

- Se consideră domeniul plan, interior profilului aerodinamic definit prin:

$$D = \{(x, y) \in \mathbb{R}^2: 0 \leq \bar{x} \leq 1, \bar{y}_{int} \leq y \leq \bar{y}_{ext}\}$$

Aria suprafeței mărginită de ecuația extradadosului  $\bar{y}_{ext}[\bar{x}_{ext}(\bar{x})]$ , ecuația intradosului  $\bar{y}_{int}[\bar{x}_{int}(\bar{x})]$  și de cele două verticale  $\bar{x} = a$ ,  $\bar{x} = b$ ,  $a=0$ ,  $b=1$  se calculează cu relațiile:

$$A_1 = \int_a^b \{\bar{y}_{ext}[\bar{x}_{ext}(\bar{x})] - \bar{y}_{int}[\bar{x}_{int}(\bar{x})]\} dx$$

$$A_2 = \iint_D dA = \int_a^b \left( \int_{\bar{y}_{int}(\bar{x}_{int}(x))}^{\bar{y}_{ext}(\bar{x}_{ext}(x))} dy \right) dx$$

- Coordonatele centrului de greutate pentru suprafața mărginită de ecuația extradosului  $\bar{y}_{ext}[\bar{x}_{ext}(\bar{x})]$ , ecuația intradosului  $\bar{y}_{int}[\bar{x}_{int}(\bar{x})]$  și de cele două verticale  $\bar{x} = a$ ,  $\bar{x} = b$ ,  $a=0$ ,  $b=1$  se calculează cu relațiile:

$$\begin{cases} x_{G1} = \frac{1}{A} \int_a^b x \{\bar{y}_{ext}[\bar{x}_{ext}(\bar{x})] - \bar{y}_{int}[\bar{x}_{int}(\bar{x})]\} dx \\ y_{G1} = \frac{1}{2A} \int_a^b \{\bar{y}_{ext}^2(\bar{x}_{ext}(x)) - \bar{y}_{int}^2(\bar{x}_{int}(x))\} dx \end{cases}$$

$$\begin{cases} x_{G2} = \frac{1}{A} \iint_D x dA \\ y_{G2} = \frac{1}{A} \iint_D y dA \end{cases}$$

În relațiile de calcul ale ariei și ale coordonatelor centrului de greutate, indicele inferior 1 se referă la cazul utilizării integralelor simple, iar indicele inferior 2 se referă la cazul utilizării integralelor duble.

Pentru rezolvarea problemei se definește un fișier de tip script conținând următoarele instrucțiuni principale:

```
%% PROFIL NACA 65-0415
close all;clear all;clc;
%% DATE DE INTRARE
% Coeficientul adimensional de curbura
cp0=0.4;
% Grosimea relativa a profilului
d=0.15;
%% CALCULUL COORDONATELOR PROFILULUI
% Coarda adimensională a profilului
x=[linspace(0+eps,0.1,1000) linspace(0.1,0.9,100)...
linspace(0.9,1-eps,1000)];
% Ecuația adimensională a scheletului profilului
yf=@(x) -cp0/(4*pi)*((1-x).*log(1-x)+x.*log(x));
% Funcția de grosime adimensională a profilului
yd=@(x) d*(1-x)./(1-0.176*x).* (1.0675*sqrt(x)...
-0.2758*x+2.4478*x.^2-2.8385*x.^3);
% Derivata scheletului profilului
D=@(x) -cp0/(4*pi)*(-log(1-x)+log(x));
% Unghiul tangentei la scheletul profilului
t=@(x) atan(D(x));
```

```

% Coordonatele intradosului
xi=@(x) x+yd(x).*sin(t(x));
yi=@(x) yf(x)-yd(x).*cos(t(x));
% Coordonatele extradosului
xe=@(x) x-yd(x).*sin(t(x));
ye=@(x) yf(x)+yd(x).*cos(t(x));
%% CALCULUL ARIEI PROFILULUI SI COORODNATELOR
% CENTRULUI DE GREUTATE
% Metoda integralei simple, TRAPZ
Alt=trapz(xe(x),ye(x))-trapz(xi(x),yi(x))
% Metoda integralei simple, QUAD
fa=@(x) ye(x)-yi(x);
fxg=@(x) x.*(ye(x)-yi(x));
fyg=@(x) ye(x).^2-yi(x).^2;
A1q=quad(fa,0,1)
xG1=1/A1q*quad(fxg,0,1)
yG1=1/(2*A1q)*quad(fyg,0,1)
% Metoda integralei duble, QUAD2D
fa2=@(x,y) x./x;
fxg2=@(x,y) x;fyg2=@(x,y) y;
A2=quad2d(fa2,0,1,yi, ye)
xG2=1/A2*quad2d(fxg2,0,1,yi, ye)
yG2=1/A2*quad2d(fyg2,0,1,yi, ye)
%% REPREZENTARE GRAFICA
figure
% Cercurile inscrise in profil
xc=[linspace(0,0.2,10) linspace(0.2,0.8,20)
linspace(0.8,1,10)];nc=length(xc);
for i=1:nc
    circle(xc(i),yf(xc(i)),yd(xc(i)));hold on;
end
% Scheletul
plot(x,yf(x),'--k');xlabel('x');ylabel('y');
grid on;axis image;hold off;
figure
% Extradosul
plot(xi(x),yi(x),'-k','LineWidth',2);hold on;
% Intradosul
plot(xe(x),ye(x),'-k','LineWidth',2);grid on;
% Scheletul
plot(x,yf(x),'--k');
xlabel('x');ylabel('y');axis image;
% Centrul de greutate al profilului
% calculat cu integrala simpla
plot(xG1,yG1,'ok');
% calculat cu integrala dubla
plot(xG2,yG2,'sk');hold off;

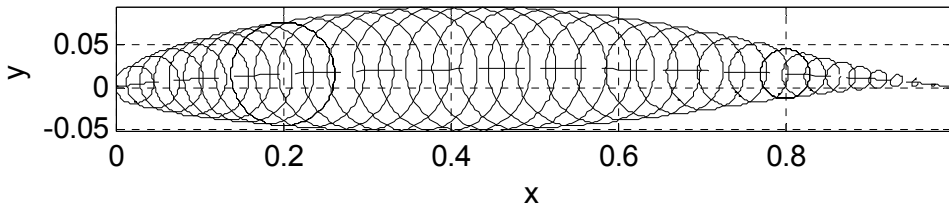
```

Lansarea în execuție a fișierului script conduce la obținerea reprezentărilor grafice din figura 6.21 (cercurile înscrise în profilul NACA65-0415) și figura 6.22 (intradosul, extradadosul, scheletul și centrul de greutate al profilului NACA65-0415), precum și a următoarelor rezultate numerice:

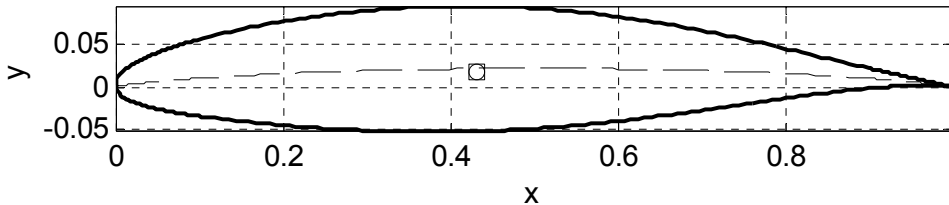
```

A1t =                                xG1 =
    0.097701257016042                0.431033086079837
A1q =                                yG1 =
    0.097550686490347                0.018291613025910
A2 =                                  xG2 =
    0.097556759533807                0.430966104691014
                                       yG2 =
                                       0.018284977837779

```



**Figura 6.21.** Cercurile înscrise în profilul NACA65-0415.



**Figura 6.22.** Intradosul, extradadosul, scheletul și centrul de greutate al profilului NACA65-0415.

### Observații

- Calitatea reprezentării grafice depinde de numărul de puncte de pe intradosul și extradadosul profilului, în special în zona bordului de atac și a bordului de fugă al profilului. Din acest motiv, coarda adimensională a profilului conține trei subdomenii: zona bordului de atac `linspace(0+eps, 0.1, 2000)`, zona centrală `linspace(0.1, 0.9, 100)` și zona bordului de fugă `linspace(0.9, 1-eps, 2000)`; în total 4100 de puncte de discretizare.
- Parametrul predefinit `eps= 2.2204e-016` intervine în punctul de început `0+eps` și în cel de sfârșit `1-eps` pentru a elimina nedeterminările NaN care intervin în bordul de atac și în bordul de fugă datorită formei particulare a ecuației adimensionale a scheletului profilului.



- Toate mărimile caracteristice ale profilului (ecuația adimensională a scheletului, funcția de grosime adimensională, derivata scheletului, unghiul tangentei la scheletul profilului, coordonatele intradosului și extradadosului profilului) se definesc ca funcții de tip anonymous pentru a facilita utilizarea procedurilor de integrare numerică `quad` și `quad2d` (aceste două proceduri de integrare numerică au ca parametru de intrare numele funcției de integrat).
- Pentru calculul ariei profilului aerodinamic s-au utilizat două metode: metoda integralei simple (procedura `trapz` și procedura `quad`) și metoda integralei duble (procedura `quad2d`). În cazul utilizării procedurii `trapz`, pentru calculul ariei profilului  $A1t$ , din aria delimitată de extradados și axa abscisei, calculată cu instrucțiunea `trapz(xe(x), ye(x))`, se scade aria dintre intrados și axa abscisei, calculată cu instrucțiunea `trapz(xi(x), yi(x))`. În cazul unui profil aerodinamic simetric, cele două arii sunt egale, doar că extradadosul, fiind plasat deasupra abscisei, definește o arie pozitivă, în timp ce intradosul, plasat în întregime sub abscisă, definește o arie negativă. În cazul utilizării procedurii `quad`, pentru calculul ariei profilului  $A1q$ , s-a definit funcția de integrat prin instrucțiunea `fa=@(x) ye(x)-yi(x)`, apoi s-a utilizat instrucțiunea de calcul a integralei simple `A1q=quad(fa,0,1)`. În cazul utilizării procedurii `quad2d`, pentru calculul ariei profilului  $A2$ , s-a definit funcția unitară de integrat cu instrucțiunea `fa2=@(x,y) x./x`, apoi s-a utilizat instrucțiunea de calcul a integralei duble `A2=quad2d(fa2,0,1,yi,ye)`.
- Pentru calculul coordonatelor centrului de greutate al profilului aerodinamic s-au utilizat două metode: metoda integralei simple (procedura `quad`) și metoda integralei duble (procedura `quad2d`). În cazul utilizării procedurii `quad`, pentru calculul coordonatelor  $xG1$  și  $yG1$ , s-au definit funcțiile de integrat prin instrucțiunile `fxg=@(x) x.*(ye(x)-yi(x))` și `fyg=@(x) ye(x).^2-yi(x).^2`, apoi s-au utilizat instrucțiunile de calcul a integralelor duble `xG1=1/A1q*quad(fxg,0,1)` și `yG1=1/(2*A1q)*quad(fyg,0,1)`. În cazul utilizării procedurii `quad2d`, pentru calculul coordonatelor  $xG2$  și  $yG2$ , s-au definit funcțiile de integrat prin instrucțiunile `fxg2=@(x,y) x` și `fyg2=@(x,y) y`, apoi s-au utilizat instrucțiunile de calcul a integralelor duble `xG2=1/A2*quad2d(fxg2,0,1,yi,ye)` și `yG2=1/A2*quad2d(fyg2,0,1,yi,ye)`.

**Integrale triple.** Pentru  $n=3$  și folosind notațiile  $x_1 = x$ ,  $x_2 = y$ ,  $x_3 = z$ , se obține cazul integralelor triple pe domenii mărginite  $V \subset \mathbb{R}^3$ :

$$I = \iiint_V f(x, y, z) dx dy dz$$

În cazul în care domeniul  $V \subset \mathbb{R}^3$  este un domeniu simplu în raport cu axa  $Oz$ , adică este definit de o suprafață:

$$e(x, y) \leq z \leq g(x, y)$$

a cărei proiecție în planul  $xOy$  corespunde unui domeniu  $D \subset \mathbb{R}^2$  definit de:

$$c(x) \leq y \leq d(x) \text{ cu } a \leq x \leq b$$

atunci integrala triplă se calculează cu relația [3, 39]:

$$\begin{aligned} I &= \iiint_V f(x, y, z) dx dy dz = \iint_D \left( \int_{e(x,y)}^{g(x,y)} f(x, y, z) dz \right) dx dy \\ &= \int_a^b \left[ \int_{c(x)}^{d(x)} \left( \int_{e(x,y)}^{g(x,y)} f(x, y, z) dz \right) dy \right] dx \end{aligned}$$

Dacă într-o integrală triplă se face schimbarea de variabile, [8]:

$$\begin{cases} x = \alpha(u, v, w) \\ y = \beta(u, v, w) \\ z = \gamma(u, v, w) \end{cases}$$

ceea ce este echivalent cu transformarea domeniului  $V_{x,y,z}$  în domeniul  $V_{u,v,w}$ , atunci se poate scrie următoarea egalitate:

$$\begin{aligned} &\iiint_{V_{x,y,z}} f(x, y, z) dx dy dz = \\ &= \iiint_{V_{u,v,w}} f(\alpha(u, v, w), \beta(u, v, w), \gamma(u, v, w)) \cdot |J(u, v, w)| du dv dw \end{aligned}$$

în care  $J(u, v, w) \neq 0$  este Jacobianul transformării definit prin:

$$J(u, v, w) = \begin{vmatrix} \frac{\partial \alpha}{\partial u} & \frac{\partial \alpha}{\partial v} & \frac{\partial \alpha}{\partial w} \\ \frac{\partial \beta}{\partial u} & \frac{\partial \beta}{\partial v} & \frac{\partial \beta}{\partial w} \\ \frac{\partial \gamma}{\partial u} & \frac{\partial \gamma}{\partial v} & \frac{\partial \gamma}{\partial w} \end{vmatrix}$$

Pentru cazul particular al transformării din coordonate carteziene  $(x, y, z)$  în coordonate cilindrice  $(r, \theta, z)$ , schimbarea de variabile corespunde relațiilor:

$$\begin{cases} x = r \cos \theta \\ y = r \sin \theta \\ z = z \end{cases}$$

pentru care Jacobianul transformării este:

$$J(r, \theta, z) = r$$

În aceste condiții, integrala triplă se calculează cu relația:

$$\iiint_{V_{x,y,z}} f(x, y, z) dx dy dz = \iiint_{V_{r,\theta,z}} f(r \cos \theta, r \sin \theta, z) r dr d\theta dz$$

Pentru cazul particular al transformării din coordonate carteziene  $(x, y, z)$  în coordonate sferice  $(r, \theta, \varphi)$ , schimbarea de variabile corespunde relațiilor:

$$\begin{cases} x = r \cos \theta \sin \varphi \\ y = r \sin \theta \sin \varphi \\ z = r \cos \varphi \end{cases}$$

pentru care Jacobianul transformării este:

$$J(r, \theta, z) = -r^2 \sin \varphi$$

În aceste condiții, integrala triplă se calculează cu relația:

$$\begin{aligned} & \iiint_{V_{x,y,z}} f(x, y, z) dx dy dz = \\ & = \iiint_{V_{r,\theta,z}} f(r \cos \theta \sin \varphi, r \sin \theta \sin \varphi, r \cos \varphi) r^2 \sin \varphi dr d\theta d\varphi \end{aligned}$$

Pentru calculul numeric al integralei triple:

$$I = \iiint_{V_{x,y,z}} f(x, y, z) dx dy dz$$

pe un domeniu paralelipedic de integrare  $V_{x,y,z}$  de forma

$$a \leq x \leq b, c \leq y \leq d, e \leq z \leq g$$

în funcție de modul de definire a funcției de integrat se folosește una din instrucțiunile, [31]:

```
I=triplequad(f, a, b, c, d, e, g, tol)
I=triplequad(@fun, a, b, c, d, e, g, tol)
```

în care:  $a \leq x \leq b$ ,  $c \leq y \leq d$  și  $e \leq z \leq g$  reprezintă limitele domeniului de integrare;  $f$  este denumirea funcției de integrat definită ca funcție de tip anonymous;  $fun$  este denumirea fișierului de tip `function` care conține definiția funcției de integrat; `tol` este eroarea limită (implicit  $10^{-6}$ ), iar  $I$  reprezintă valoarea numerică a integralei triple.

În cazul unui domeniu de integrare simplu în raport cu axa  $Oz$ ,  $V_{x,y,z}$  de forma:

$$a \leq x \leq b, c(x) \leq y \leq d(x), e(x, y) \leq z \leq g(x, y)$$

în funcție de modul de definire a funcției de integrat se folosește una din instrucțiunile, [32]:

```
I=integral3(f, a, b, c, d, e, g)
I=integral3(@fun, a, b, c, d, e, g)
```

### Problema 6.17

Să se calculeze integrala triplă:

$$I = \iiint_{V_{x,y,z}} f(x,y,z) dx dy dz$$

cunoscând:

- Volumul paralelipipedic de integrare este definit prin:  
 $V_{x,y,z} = \{(x,y,z) \in \mathbb{R}^3: a \leq x \leq b, c \leq y \leq d, e \leq z \leq g\}$   
în care  $a=0, b=1, c=0, d=1, e=0$  și  $g=1$ .
- Funcția de integrat are expresia:

$$f(x,y,z) = \frac{1}{\sqrt{x+y+z+1}}$$

### Rezolvare

Fișierul script pentru rezolvarea problemei conține următoarele instrucțiuni:

```
%% CALCULUL INTEGRALEI TRIPLE (1)
% Metoda functiilor anonymous
close all;clear all;clc;
%% DATE DE INTRARE
% Limitele variabilei x
a=0;b=1;
% Limitele variabilei y
c=0;d=1;
% Limitele variabilei z
e=0;g=1;
% Functia de integrat
f=@(x,y,z) 1./sqrt(x+y+z+1);
%% CALCULUL INTEGRALEI TRIPLE
I=triplequad(f,a,b,c,d,e,g)
```

În urma lansării în execuție a fișierului se obține următorul rezultat numeric:

```
I =
    0.6428
```

### Observatii

- Introducerea limitelor domeniilor de definiție pentru cele trei variabile se face cu instrucțiunile  $a=0; b=1; c=0; d=1; e=0; g=1$ .
- Funcția de integrat este definită ca funcție de tip anonymous prin instrucțiunea  $f=@(x,y,z) 1./sqrt(x+y+z+1)$ .
- Calculul integralei triple se face cu ajutorul instrucțiunii  $I=triplequad(f,a,b,c,d,e,g)$ .

### Problema 6.18

Se considera domeniul paralelipipedic definit prin:

$$V_{x,y,z} = \{(x, y, z) \in \mathbb{R}^3: a \leq x \leq b, c \leq y \leq d, e \leq z \leq g\}$$

în care  $a=0, b=2, c=0, d=1, e=0$  și  $g=1$ .

În ipoteza unui corp neomogen din punct de vedere al distribuției masei pentru care densitatea corpului este definită prin relația:

$$\rho(x, y, z) = x + y + z$$

să se calculeze următoarele mărimi, [4]:

- Volumul  $V$  și masa  $M$ .
- Momentele statice în raport cu cele trei plane de coordonate  $M_{xy}$ ,  $M_{yz}$  și  $M_{zx}$ .
- Coordonatele centrului de greutate  $x_G, y_G$  și  $z_G$ .
- Momentele de inerție axiale în raport cu cele trei axe de coordonate  $I_x, I_y$  și  $I_z$ .
- Momentele de inerție planare în raport cu cele trei plane de coordonate  $I_{Oxy}, I_{Oyz}$  și  $I_{Ozx}$ .
- Momentele de inerție centrifugale în raport cu axele de coordonate  $I_{xy}, I_{yz}$  și  $I_{zx}$ .
- Momentul de inerție polar în raport cu originea sistemului de coordonate  $I_p$ .

### Rezolvare

Volumul  $V$  al domeniului  $V_{x,y,z}$  se calculează cu relația:

$$V = \iiint_{V_{x,y,z}} dx dy dz$$

Masa  $M$  a domeniului  $V_{x,y,z}$  se calculează cu relația:

$$M = \iiint_{V_{x,y,z}} \rho(x, y, z) dx dy dz$$

Momentele statice  $M_{xy}, M_{yz}$  și  $M_{zx}$  ale domeniului  $V_{x,y,z}$  în raport cu planele de coordonate  $Oxy, Oyz$  și  $Ozx$  se calculează cu relațiile:

$$M_{xy} = \iiint_{V_{x,y,z}} z \rho(x, y, z) dx dy dz$$

$$M_{yz} = \iiint_{V_{x,y,z}} x \rho(x, y, z) dx dy dz$$

$$M_{zx} = \iiint_{V_{x,y,z}} y \rho(x, y, z) dx dy dz$$

Coordonatele centrului de greutate  $x_G, y_G$  și  $z_G$  ale domeniului  $V_{x,y,z}$  se calculează cu relațiile:

$$x_G = \frac{M_{yz}}{M}$$

$$y_G = \frac{M_{zx}}{M}$$

$$z_G = \frac{M_{xy}}{M}$$

Momentele de inerție axiale  $I_x$ ,  $I_y$  și  $I_z$  ale domeniului  $V_{x,y,z}$  în raport cu axele de coordonate  $Ox$ ,  $Oy$  și  $Oz$  se calculează cu relațiile:

$$I_x = \iiint_{V_{x,y,z}} (y^2 + z^2) \rho(x, y, z) dx dy dz$$

$$I_y = \iiint_{V_{x,y,z}} (z^2 + x^2) \rho(x, y, z) dx dy dz$$

$$I_z = \iiint_{V_{x,y,z}} (x^2 + y^2) \rho(x, y, z) dx dy dz$$

Momentele de inerție planare  $I_{Oxy}$ ,  $I_{Oyz}$  și  $I_{Ozx}$  ale domeniului  $V_{x,y,z}$  în raport cu planele de coordonate  $Oxy$ ,  $Oyz$  și  $Ozx$  se calculează cu relațiile:

$$I_{Oxy} = \iiint_{V_{x,y,z}} z^2 \rho(x, y, z) dx dy dz$$

$$I_{Oyz} = \iiint_{V_{x,y,z}} x^2 \rho(x, y, z) dx dy dz$$

$$I_{Ozx} = \iiint_{V_{x,y,z}} y^2 \rho(x, y, z) dx dy dz$$

Momentele de inerție centrifugale  $I_{xy}$ ,  $I_{yz}$  și  $I_{zx}$  ale domeniului  $V_{x,y,z}$  în raport cu axele de coordonate  $xy$ ,  $yz$  și  $zx$  se calculează cu relațiile:

$$I_{xy} = \iiint_{V_{x,y,z}} xy \rho(x, y, z) dx dy dz$$

$$I_{yz} = \iiint_{V_{x,y,z}} yz \rho(x, y, z) dx dy dz$$

$$I_{zx} = \iiint_{V_{x,y,z}} zx \rho(x, y, z) dx dy dz$$

Momentul de inerție polar  $I_p$  al domeniului  $V_{x,y,z}$  în raport cu originea sistemului de coordonate se calculează cu relația:

$$I_p = \iiint_{V_{x,y,z}} (x^2 + y^2 + z^2) \rho(x, y, z) dx dy dz$$

Fișierul script pentru rezolvarea problemei conține instrucțiunile:

```
%% CALCULUL INTEGRALEI TRIPLE (2)
% Metoda functiilor anonymous
close all;clear all;clc;
%% DATE DE INTRARE
% Limitele variabilei x
a=0;b=2;
% Limitele variabilei y
c=0;d=1;
% Limitele variabilei z
e=0;g=1;
% Functiile de integrat
fv=@(x,y,z) x./x;
fr=@(x,y,z) x+y+z;
fmxy=@(x,y,z) z.*fr(x,y,z);
fmyz=@(x,y,z) x.*fr(x,y,z);
fmzx=@(x,y,z) y.*fr(x,y,z);
fix=@(x,y,z) (y.^2+z.^2).*fr(x,y,z);
fiy=@(x,y,z) (z.^2+x.^2).*fr(x,y,z);
fiz=@(x,y,z) (x.^2+y.^2).*fr(x,y,z);
fiOxy=@(x,y,z) z.^2.*fr(x,y,z);
fiOyz=@(x,y,z) x.^2.*fr(x,y,z);
fiOzx=@(x,y,z) y.^2.*fr(x,y,z);
fixy=@(x,y,z) x.*y.*fr(x,y,z);
fiyz=@(x,y,z) y.*z.*fr(x,y,z);
fizx=@(x,y,z) z.*x.*fr(x,y,z);
fip=@(x,y,z) (x.^2+y.^2+z.^2).*fr(x,y,z);
%% CALCULUL INTEGRALELOR TRIPLE
% Volumul
V=triplequad(fv,a,b,c,d,e,g)
% Masa
M=triplequad(fr,a,b,c,d,e,g)
% Momentele statice
Mxy=triplequad(fmxy,a,b,c,d,e,g)
Myz=triplequad(fmyz,a,b,c,d,e,g)
Mzx=triplequad(fmzx,a,b,c,d,e,g)
% Coordonatele centrului de greutate
xG=Myz/M;yG=Mzx/M;zG=Mxy/M;
% Momentele de inertie axiale
Ix=triplequad(fix,a,b,c,d,e,g)
Iy=triplequad(fiy,a,b,c,d,e,g)
Iz=triplequad(fiz,a,b,c,d,e,g)
% Momentele de inertie planare
IOxy=triplequad(fiOxy,a,b,c,d,e,g)
IOyz=triplequad(fiOyz,a,b,c,d,e,g)
IOzx=triplequad(fiOzx,a,b,c,d,e,g)
```

```

% Momentele de inertie centrifugale
Ixy=triplequad(fixy,a,b,c,d,e,g)
Iyz=triplequad(fiyz,a,b,c,d,e,g)
Izx=triplequad(fizx,a,b,c,d,e,g)
% Momentul de inertie polar
Ip=triplequad(fip,a,b,c,d,e,g)

```

În urma lansării în execuție a fișierului se obțin următoarele rezultate numerice:

V =	Ix =	Ixy =
2	3	2.5000
M =	Iy =	Iyz =
4	8.1667	1.1667
Mxy =	Iz =	Izx =
2.1667	8.1667	2.5000
Myz =	IOxy =	Ip =
4.6667	1.5000	9.6667
Mzx =	IOyz =	
2.1667	6.6667	
xG =	IOzx =	
1.1667	1.5000	
yG =		
0.5417		
zG =		
0.5417		

### Observații

- Toate funcțiile de integrat sunt definite prin metoda funcțiilor anonymous.
- Toate mărimile calculate se reduc la integrale triple în funcție de elementul de volum  $dV = dx dy dz$  sau elementul de masă  $dm = \rho(x, y, z) dx dy dz = \rho(x, y, z) dV$ .
- Corpurile tridimensionale pentru care densitatea  $\rho(x, y, z)$  este constantă se numesc corpuri omogene din punct de vedere al distribuției maselor. În acest caz, densitatea fiind constantă  $\rho(x, y, z) = \rho_0$ , relațiile de calcul se simplifică și, de exemplu, pentru calculul coordonatelor centrului de greutate capătă formele:

$$x_G = \frac{1}{V} \iiint_{V_{x,y,z}} x dV$$

$$y_G = \frac{1}{V} \iiint_{V_{x,y,z}} y dV$$

$$z_G = \frac{1}{V} \iiint_{V_{x,y,z}} z dV$$



## 6.7. REZOLVAREA ECUAȚIILOR DIFERENȚIALE ORDINARE

### 6.7.1. Generalități

Se consideră o funcție reală  $F: [x_{min}; x_{max}] \times Y$ ,  $Y \subset \mathbb{R}^{n+1}$  dependentă de variabila reală  $x \in [x_{min}; x_{max}]$ , de funcția reală  $y$ , precum și de derivatele sale  $y', y'', \dots, y^{(n)}$  definită prin expresia:

$$F(x, y, y', y'', \dots, y^{(n)})$$

Relația:

$$F(x, y, y', y'', \dots, y^{(n)}) = 0$$

se numește ecuație diferențială de ordinul  $n$ , [39].

În cazul în care funcția necunoscută  $y = f(x)$  depinde doar de o singură variabilă, ecuația este o ecuație diferențială ordinară (ODE). În cazul funcțiilor care depind de mai mulți parametri, ecuația este o ecuație diferențială cu derivate parțiale (PDE).

Funcția  $y = \varphi(x)$ , derivabilă de  $n$  ori pe intervalul  $[x_{min}; x_{max}]$ , pentru care este verificată relația:

$$F(x, \varphi(x), \varphi'(x), \varphi''(x), \dots, \varphi^{(n)}(x)) = 0$$

se numește soluție particulară a ecuației diferențiale. Graficul funcției  $y = \varphi(x)$  se numește curbă integrală.

Funcția  $\varphi(x, c_1, c_2, \dots, c_n)$  este soluția generală a ecuației diferențiale dacă  $\varphi$  este soluție a ecuației și dacă prin alegerea convenabilă a constantelor  $c_1, c_2, \dots, c_n$ , funcția  $\varphi(x, c_1, c_2, \dots, c_n)$  conduce la toate soluțiile ecuației diferențiale.

Determinarea constantelor  $c_1, c_2, \dots, c_n$  și obținerea astfel a unei soluții particulare a ecuației diferențiale se realizează considerând un set suplimentar de condiții pe care trebuie să le îndeplinească soluția ecuației. În funcție de tipul condițiilor suplimentare, ecuațiile diferențiale se clasifică în următoarele două categorii:

- Probleme cu condiții inițiale (initial conditions) pentru care se cunosc valorile funcției  $y$  și derivatelor sale la începutul domeniului de definiție ( $x = x_0$ ):

$$y(x_0) = y_0, y'(x_0) = y'_0, y''(x_0) = y''_0, \dots, y^{(n)}(x_0) = y_0^{(n)}$$

Acest tip de problemă se mai numește problema lui Cauchy:

- Problema lui Cauchy pentru ecuația diferențială de ordinul 1 de forma generală  $y' = f(x, y)$ , constă în determinarea soluției  $y(x)$  cunoscând condiția inițială  $y(x_0) = y_0$ .
- Problema lui Cauchy pentru ecuația diferențială de ordinul 2 constă în determinarea soluției  $y(x)$  pentru ecuația diferențială generală  $y'' = f(x, y, y')$  cunoscând condițiile inițiale  $y(x_0) = y_0$  și  $y'(x_0) = y'_0$ .

- Probleme cu condiții pe frontieră (boundary conditions) pentru care se cunosc valorile funcției  $y$ , la începutul și la sfârșitul domeniului de definiție:

$$\begin{aligned}x = x_{min} &\Rightarrow y(x_{min}) = y_0^{min} \\x = x_{max} &\Rightarrow y(x_{max}) = y_0^{max}\end{aligned}$$

### Observații

- Orice ecuație diferențială de ordin superior se poate transforma într-un sistem de ecuații diferențiale de ordinul 1 cu ajutorul unei schimbări convenabile de variabile. Spre exemplu, ecuația diferențială de ordinul 2:

$$y'' = f(x, y, y')$$

se poate transforma folosind schimbarea de variabile:

$$\begin{cases}y_1 = y \\y_2 = y'\end{cases}$$

în următorul sistem de două ecuații diferențiale de ordinul 1:

$$\begin{cases}y'_1 = y_2 \\y'_2 = f(x, y_1, y_2)\end{cases}$$

- Se consideră ecuația diferențială ordinară de ordinul 1:

$$y' = f(t, u(t), y)$$

și condiția inițială:

$$t = 0 \Rightarrow y(0) = y_0$$

În cazul în care variabila independentă este timpul, atunci pentru operatorul de derivare se poate utiliza notația cu punct:

$$\frac{dy}{dt} \Leftrightarrow \dot{y}$$

astfel încât ecuația diferențială se poate rescrie sub forma:

$$\dot{y} = f(t, u(t), y)$$

Soluția ecuației diferențiale se poate obține evaluând integrala:

$$y(t) = y_0 + \int_{t_0}^t f(t, u(t), y) dt$$

- Se consideră ecuația diferențială ordinară de ordinul 2:

$$y'' = f(t, u(t), y, y')$$

și condițiile inițiale:

$$t = 0 \Rightarrow \begin{cases}y(0) = y_0 \\y'(0) = y'_0\end{cases}$$

În cazul în care variabila independentă este timpul, atunci pentru operatorii de derivare se poate utiliza notația cu punct:

$$\frac{dy}{dt} \Leftrightarrow \dot{y}, \frac{d^2y}{dt^2} \Leftrightarrow \ddot{y}$$

astfel încât ecuația diferențială se poate rescrie sub forma:

$$\ddot{y} = f(t, u(t), y, \dot{y})$$

Soluția ecuației diferențiale se poate obține evaluând integralele:

$$\dot{y}(t) = y'_0 + \int_{t_0}^t f(t, u(t), y, \dot{y}) dt$$

$$y(t) = y_0 + \int_{t_0}^t \dot{y}(t) dt$$

### Problema 6.19

Se consideră ecuația diferențială ordinară de ordinul 1:

$$\begin{cases} \dot{y}(t) = \frac{1}{t+1} \cdot \sin(t+1)^2 \\ y(t_0) = 0 \end{cases}$$

și intervalul de integrare  $t=0 \div 4$ .

Să se determine soluția ecuației diferențiale folosind relația:

$$y(t) = y_0 + \int_{t_0}^t f(t) dt$$

în care  $f(t) = \frac{1}{t+1} \cdot \sin(t+1)^2$ .

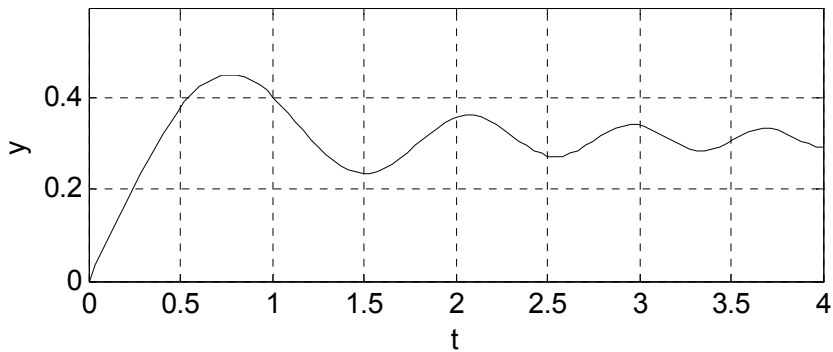
Să se reprezinte grafic soluția obținută.

### Rezolvare

Pentru rezolvarea problemei se definește un fișier de tip script conținând următoarele instrucțiuni principale:

```
%% REZOLVAREA ECUATIILOR DIFERENTIALE ORDINARE (1)
clear all;close all;clc;
%% DATE DE INTRARE
% Functia de integrat
f=@(t) 1./(t+1).*sin((t+1).^2);
% Domeniul de integrare
t0=0;tmax=4;nt=100;t=linspace(t0,tmax,nt);
% Conditia initiala
y0=0;
%% SOLUTIA NUMERICA
% Determinarea solutiei
for i=1:nt
    y(i)=y0+quad(f,t0,t(i));
end
% Reprezentarea grafica a solutiei
figure
plot(t,y,'-k');grid on;xlabel('t');ylabel('y');
```

În urma lansării în execuție a fișierului script se obține reprezentarea grafică din figura 6.23.



**Figura 6.23.** Soluția ecuației diferențiale de la problema 6.19.

### Observații

- Expresia de integrat se definește printr-o funcție de tip anonymous de forma  $f=@(t) 1./(t+1) .* \sin ((t+1).^2)$ .
- Calculul integralei  $\int_{t_0}^t f(t) dt$  se face folosind instrucțiunea `quad` într-o structură repetitivă cu `contor` de forma:
 

```
for i=1:nt
    y(i)=y0+quad(f,t0,t(i));
end
```
- Contorul `i` parcurge toate cele `nt` valori ale domeniului de integrare pentru definirea căruia s-au folosit instrucțiunile:
 

```
t0=0;tmax=4;nt=100;t=linspace(t0,tmax,nt);
```
- Pentru reprezentarea grafică a soluției ecuației diferențiale s-a folosit instrucțiunea `plot(t,y,'-k')`. Parametrii de formatare ai curbei `'-k'` conduc la obținerea unei linii continue de culoare neagră.
- Formatarea graficului constă în activarea rețelei `grid` (`grid on`) și în etichetarea celor două axe (`xlabel('t');`;`ylabel('y')`).

### 6.7.2. Rezolvarea numerică a ecuațiilor diferențiale ordinare de ordinul 1

Se consideră ecuația diferențială ordinară de ordinul 1 exprimată în formă explicită prin:

$$\begin{cases} y'(t) = f(t, y) \\ y(t_0) = y_0 \end{cases}$$

În funcție de metoda de rezolvare utilizată, ecuația diferențială poate conduce la mai multe categorii de soluții, [13]: soluții închise obținute prin utilizarea metodei analitice de rezolvare; soluții pseudo-închise obținute prin transformarea ecuației diferențiale într-o ecuație cu diferențe finite de ordinul 1 și rezolvarea acestora prin metoda analitică; soluții deschise obținute prin utilizarea metodelor numerice de integrare a ecuației diferențiale.

Aplicarea metodei analitice de rezolvare și obținerea soluțiilor închise ale ecuațiilor diferențiale ordinare se poate realiza în unele cazuri particulare dintre care cele mai importante sunt: ecuații diferențiale cu variabile separabile, ecuații diferențiale omogene, ecuații reductibile la ecuații omogene, ecuații cu diferențiale totale, ecuații diferențiale care admit factor integrant, ecuații diferențiale liniare de ordinul întâi, ecuații diferențiale de tip Bernoulli, ecuații diferențiale de tip Riccati, ecuații diferențiale de tip Lagrange, ecuații diferențiale de tip Clairaut.

Metodele numerice de rezolvare a ecuațiilor diferențiale se bazează pe determinarea valorilor aproximative ale soluției  $y_i = y(t_i)$ ,  $\forall i = 1 \div n$  în punctele  $t_i$  de discretizare ale domeniului de integrare, [12, 13, 37, 40].

În funcție de cantitatea de informație necesară pentru determinarea soluției, metodele numerice de rezolvare se clasifică în două categorii:

- Metode numerice directe (uni-pas) pentru care valoarea curentă a soluției  $y_{i+1}$  se determină doar pe baza valorii anterioare  $y_i$ . Principalele metode numerice uni-pas sunt:
  - Metodele numerice de tip Taylor (metoda Euler, metoda Euler îmbunătățită).
  - Metodele numerice de tip Runge-Kutta.
- Metode numerice indirecte (multi-pas) pentru care valoarea curentă a soluției  $y_{i+1}$  se determină pe baza mai multor valori anterioare. Principalele metode numerice multi-pas sunt:
  - Metode numerice de tip Adams-Bashforth.
  - Metode numerice de tip Adams-Moulton.
  - Metode numerice de tip predictor-corector (metoda Adams-Bashforth-Moulton, metoda Milne-Hamming).

Rezolvarea în MATLAB a ecuațiilor diferențiale ordinare de ordinul 1 se face utilizând instrucțiunea generală, [33]:

```
[t, y] = solver(odefun, [t0 tmax], y0, options)
```

în care:

- `solver` reprezintă una din metodele numerice de rezolvare:
  - `ode45`, metodă uni-pas de tip Runge-Kutta de ordinul 4-5 în varianta Dormand-Prince
  - `ode23`, metodă uni-pas de tip Runge-Kutta de ordinul 2-3, varianta Bogacki-Shampine.
  - `ode113`, metodă multi-pas de tip predictor-corector Adams-Bashforth-Moulton.
  - `ode15s`, metodă multi-pas de tip diferențe numerice (NDFs-Numerical Differentiation Formulas).
  - `ode23s`, metodă uni-pas de tip Rosenbrock modificată de ordinul 2.

- ode23t, metoda trapezului cu interpolant liber.
- ode23tb, metoda trapezului combinată cu metoda diferențelor înapoi de ordinul 2 (BDF-Backward Differentiation Formula).
- odefun este numele funcției  $f(t, y)$  din membrul drept al ecuației diferențiale (definită ca funcție de tip anonymous, sau ca fișier de tip function).
- $[t_0 \ t_{max}]$  este intervalul de integrare pe care se determină soluția ecuației diferențiale.
- $y_0$  reprezintă condițiile inițiale ale ecuației diferențiale.
- $t$  reprezintă vectorul punctelor de discretizare a intervalului de integrare  $[t_0 \ t_{max}]$ .
- $y$  reprezintă vectorul soluție al ecuației diferențiale în fiecare punct  $t$  din intervalului de integrare  $[t_0 \ t_{max}]$ .
- options reprezintă parametrii opționali de configurare a procedurii numerice de rezolvare a ecuației diferențiale. Obținerea valorilor implicite ale tuturor parametrilor opționali de configurare se realizează cu instrucțiunea:

odeset

```

    AbsTol:[positive scalar or vector{1e-6}]
    RelTol:[positive scalar {1e-3}]
    NormControl:[on | {off}]
    NonNegative:[vector of integers]
    OutputFcn:[function_handle]
    OutputSel:[vector of integers]
    Refine:[positive integer]
    Stats:[on | {off}]
    InitialStep:[positive scalar]
    MaxStep:[positive scalar]
    BDF:[on | {off}]
    MaxOrder:[1 | 2 | 3 | 4 | {5}]
    Jacobian:[matrix | function_handle]
    JPattern:[sparse matrix]
    Vectorized:[on | {off}]
    Mass:[matrix | function_handle]
MStateDependence:[none | {weak} | strong]
    MvPattern:[sparse matrix]
    MassSingular:[yes | no | {maybe}]
    InitialSlope:[vector]
    Events:[function_handle]

```

Modificarea parametrilor opționali de configurare se realizează cu instrucțiunea:

```
options=odeset('p1',vp1,'p2',vp2,...)
```

în care 'p1' și 'p2' sunt parametrii opționali, iar vp1 și vp2 sunt noile valori ale acestor parametri.

Parametrii AbsTol și RelTol definesc eroarea  $e_i$  a fiecărei componente  $y_i$  a soluției, conform relației:

$$|e_i| \leq \max(\text{RelTol} \cdot |y_i|, \text{AbsTol}_i)$$

### Problema 6.20

Se consideră ecuația diferențială ordinară de ordinul 1:

$$\begin{cases} \dot{y}(t) = t^3 - 2ty \\ y(t_0) = 1 \end{cases}$$

și intervalul de integrare  $t=1 \div 5$ .

Să se determine soluția numerică  $y_1(t)$  a ecuației diferențiale folosind procedura ode23 pentru valoarea implicită a erorii relative RelTol=1e-3. Să se modifice eroarea relativă la valoarea RelTol=1e-4 și să se determine apoi noua soluție numerică  $y_2(t)$ .

Cunoscând că soluția exactă determinată prin metoda analitică se poate exprima prin relația:

$$y_e = \frac{1}{2}(t^2 - 1) + e^{1-t^2}$$

să se calculeze erorile absolute dintre cele două soluții numerice  $y_1(t)$ , respectiv  $y_2(t)$  și soluția exactă a ecuației diferențiale:

$$e_1 = |y_e - y_1| \text{ și } e_2 = |y_e - y_2|$$

Să se reprezinte grafic soluțiile și erorile absolute ale acestora.

### Rezolvare

Pentru rezolvarea problemei se definește un fișier de tip script conținând următoarele instrucțiuni principale:

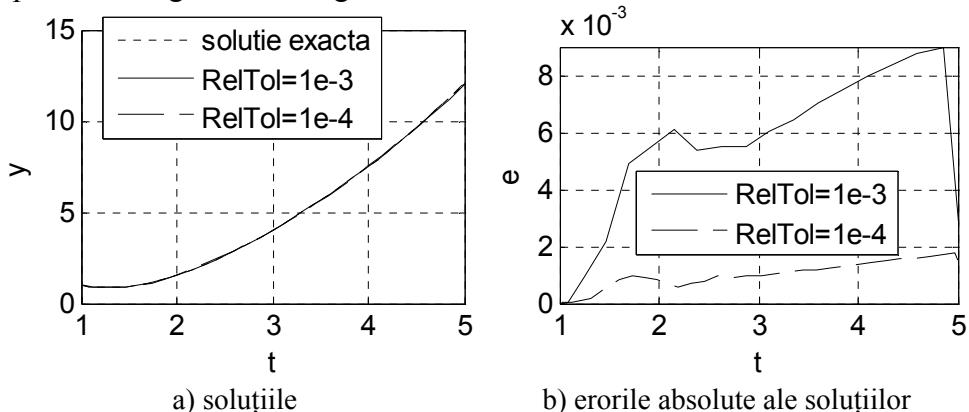
```
%% REZOLVAREA ECUATIILOR DIFERENTIALE ORDINARE (2)
clear all;close all;clc;
%% DATE DE INTRARE
% Functia de integrat
f=@(t,y) t.^3-2*t.*y;
% Domeniul de integrare
tmin=1;tmax=5;
% Conditia initiala
y0=1;
% Solutia analitica exacta
ye=@(t) 1/2*(t.^2-1)+exp(1-t.^2);
%% SOLUTIA NUMERICA APROXIMATIVA
% Pentru eroarea relativa implicita 1e-3
[t1,y1]=ode23(f,[tmin tmax],y0);
```

```

% Pentru eroarea relativa implicita 1e-4
options=odeset('RelTol',1e-4);
[t2,y2]=ode23(f,[tmin tmax],y0,options);
% Eroarea absoluta dintre cele doua solutii
e1=abs(ye(t1)-y1);e2=abs(ye(t2)-y2);
%% REPREZENTARI GRAFICE
% Reprezentarea grafica a solutiilor
figure
plot(t1,ye(t1),'k');hold on;
plot(t1,y1,'-k');plot(t2,y2,'--k');hold off;
xlabel('t');ylabel('y');grid on;
legend('solutie exacta','RelTol=1e-3','RelTol=1e-4');
% Reprezentarea grafica a erorilor
figure;
plot(t1,e1,'-k');hold on;
plot(t2,e2,'--k');hold off;
xlabel('t');ylabel('e');grid on;
legend('RelTol=1e-3','RelTol=1e-4');

```

În urma lansării în execuție a fișierului script se obține reprezentarea grafică din figura 6.24.



**Figura 6.24.** Soluția ecuației diferențiale de la problema 6.20.

### Observații

- Din punct de vedere al reprezentării grafice a soluțiilor (figura 6.24, a), nu se constată diferențe semnificative între soluția exactă și cele două soluții numerice.
- Din punct de vedere al erorilor absolute (figura 6.23, b), scăderea valorii erorii relative la  $10^{-4}$  conduce la scăderea semnificativă a valorii erorii absolute dintre soluția exactă și soluția numerică.
- Discretizarea domeniului de integrare s-a efectuat în mod automat rezultând 19 puncte pentru  $y_1$  și 33 de puncte soluția  $y_2$ .



### Problema 6.21

Se consideră ecuația diferențială ordinară de ordinul 1:

$$\begin{cases} \dot{y}(t) = t^2 \cos t + \frac{2y}{t} \\ y(t_0) = 0 \end{cases}$$

și intervalul de integrare  $t=2\pi \div 4\pi$ .

Să se determine soluția numerică  $y_1(t)$  a ecuației diferențiale folosind procedura ode23 pentru o discretizare a domeniului de integrare cu  $n_{t1}=20$  de puncte. Să se modifice numărul de puncte de discretizare la valoarea  $n_{t2}=200$  și să se determine apoi noua soluția numerică  $y_2(t)$ .

Cunoscând că soluția exactă determinată prin metoda analitică se poate exprima prin relația:

$$y_e = t^2 \cdot \sin t$$

să se calculeze erorile absolute dintre cele două soluții numerice  $y_1(t)$ , respectiv  $y_2(t)$  și soluția exactă a ecuației diferențiale:

$$e_1 = |y_e - y_1| \text{ și } e_2 = |y_e - y_2|$$

Să se reprezinte grafic soluțiile și erorile absolute ale acestora.

### Rezolvare

Pentru rezolvarea problemei se definește un fișier de tip script conținând următoarele instrucțiuni principale:

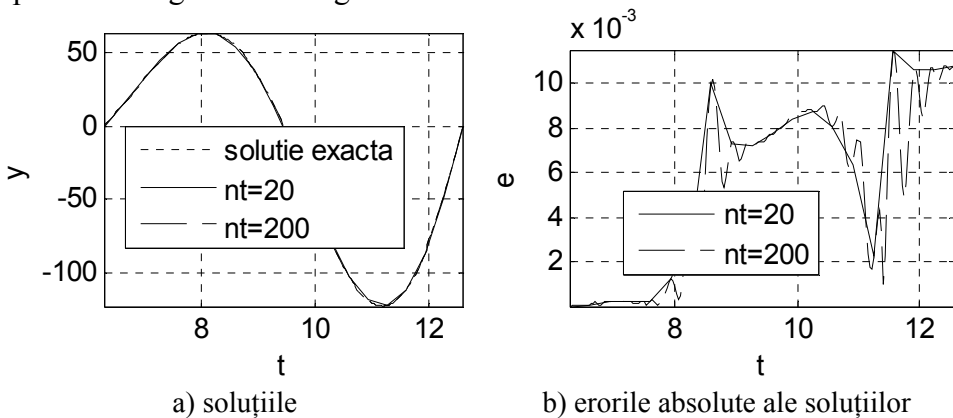
```
%% REZOLVAREA ECUATIILOR DIFERENTIALE ORDINARE (3)
clear all;close all;clc;
%% DATE DE INTRARE
% Functia de integrat
f=@(t,y) t.^2.*cos(t)+2*y./t;
% Domeniul de integrare
tmin=2*pi;tmax=4*pi;
nt1=20;t1=linspace(tmin,tmax,nt1);
nt2=200;t2=linspace(tmin,tmax,nt2);
% Conditia initiala
y0=0;
% Solutia analitica exacta
ye=@(t) t.^2.*sin(t);
%% SOLUTIA NUMERICA APROXIMATIVA
% Pentru o discretizare cu nt=20 puncte
[t1,y1]=ode23(f,t1,y0);
% Pentru o discretizare cu nt=200 puncte
[t2,y2]=ode23(f,t2,y0);
% Eroarea absoluta dintre cele doua solutii
e1=abs(ye(t1)-y1);e2=abs(ye(t2)-y2);
%% REPREZENTARI GRAFICE
% Reprezentarea grafica a solutiilor
```

```

figure
plot(t2,ye(t2),'k');hold on;
plot(t1,y1,'-k');plot(t2,y2,'--k');hold off;
xlabel('t');ylabel('y');grid on;
legend('solutie exacta','nt=20','nt=200');
% Reprezentarea grafica a erorilor
figure
plot(t1,e1,'-k');hold on;
plot(t2,e2,'--k');hold off;
xlabel('t');ylabel('e');grid on;
legend('nt=20','nt=200');

```

În urma lansării în execuție a fișierului script se obține reprezentarea grafică din figura 6.25.



**Figura 6.25.** Soluția ecuației diferențiale de la problema 6.21.

### Observații

- Atât funcția de integrat, cât și soluția exactă a ecuației diferențiale sunt definite sub forma unor funcții de tip anonymous prin instrucțiunile  $f=@(t,y) t.^2.*\cos(t)+2*y./t$  și  $ye=@(t) t.^2.*\sin(t)$ .
- Domeniul de integrare este discretizat în  $nt_1=20$  puncte ( $t_1$ ) și respectiv în  $nt_2=200$  puncte ( $t_2$ ).
- Cele două soluții numerice se obțin prin instrucțiunile  $[t_1,y_1]=ode23(f,t_1,y_0)$  și  $[t_2,y_2]=ode23(f,t_2,y_0)$ .
- Din punct de vedere al reprezentării grafice a soluțiilor (figura 6.24, a), se constată foarte mici diferențe între soluția numerică obținută pentru  $n_{t_1}=20$  puncte și celelalte două soluții.
- Din punct de vedere al erorilor absolute (figura 6.24, b), creșterea numărului de puncte de discretizare a intervalului de integrare la valoarea  $n_{t_2}=200$  are ca urmare obținerea unei evoluții mult mai exacte față de cazul  $n_{t_1}=20$ .

### Problema 6.22

Se consideră ecuația diferențială ordinară de ordinul 1:

$$\begin{cases} \dot{y}(t) = \frac{4t - ty^2}{y} \\ y(t_0) = 3 \end{cases}$$

și intervalul de integrare  $t=0 \div 4$ .

Să se determine soluția numerică  $y_1(t)$  a ecuației diferențiale folosind procedura `ode23` pentru o discretizare a domeniului de integrare cu  $n_t=1000$  de puncte. Pentru aceeași discretizare a domeniului de integrare, să se determine soluția  $y_2(t)$  corespunzătoare utilizării procedurii `ode45`.

Cunoscând că soluția exactă determinată prin metoda analitică se poate exprima prin relația:

$$y_e = \sqrt{4 + 5e^{-x^2}}$$

să se calculeze erorile absolute dintre cele două soluții numerice  $y_1(t)$ , respectiv  $y_2(t)$  și soluția exactă a ecuației diferențiale:

$$e_1 = |y_e - y_1| \text{ și } e_2 = |y_e - y_2|$$

Să se reprezinte grafic soluțiile și erorile absolute ale acestora.

### Rezolvare

Pentru rezolvarea problemei se definește un fișier de tip `script` conținând următoarele instrucțiuni principale:

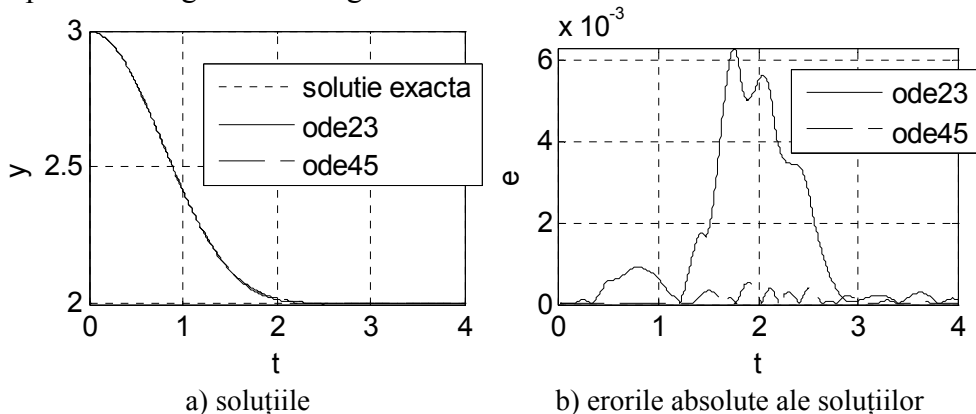
```
%% REZOLVAREA ECUATIILOR DIFERENTIALE ORDINARE (4)
clear all;close all;clc;
%% DATE DE INTRARE
% Functia de integrat
f=@(t,y) (4*t-t.*y.^2)./y;
% Domeniul de integrare
tmin=0;tmax=4;nt=1000;
t=linspace(tmin,tmax,nt);
% Conditia initiala
y0=3;
% Solutia exacta
ye=@(t) sqrt(4+5*exp(-t.^2));
%% SOLUTIA NUMERICA APROXIMATIVA
% Pentru procedura ode23
[t1,y1]=ode23(f,t,y0);
% Pentru procedura ode45
[t2,y2]=ode45(f,t,y0);
% Eroarea absoluta dintre cele doua solutii
e1=abs(ye(t1)-y1);e2=abs(ye(t2)-y2);
%% REPREZENTARI GRAFICE
% Reprezentarea grafica a solutiilor
figure
```

```

plot(t, ye(t), ':k'); hold on;
plot(t1, y1, '-k');
plot(t2, y2, '--k'); hold off;
xlabel('t'); ylabel('y'); grid on;
legend('solutie exacta', 'ode23', 'ode45');
% Reprezentarea grafica a erorilor
figure
plot(t1, e1, '-k'); hold on;
plot(t2, e2, '--k'); hold off;
xlabel('t'); ylabel('e'); grid on;
legend('ode23', 'ode45');

```

În urma lansării în execuție a fișierului script se obține reprezentarea grafică din figura 6.26.



**Figura 6.26.** Soluția ecuației diferențiale de la problema 6.22.

### Observații

- Domeniul de integrare este discretizat în  $nt=1000$  puncte folosind instrucțiunea  $t=linspace(tmin, tmax, nt)$ .
- Cele două soluții numerice  $y_1(t)$  și  $y_2(t)$  se calculează pentru aceeași discretizare a domeniului de integrare, folosind instrucțiunile  $[t1, y1]=ode23(f, t, y0)$  și  $[t2, y2]=ode4(f, t, y0)$ .
- Calculul erorilor absolute dintre cele două soluții numerice  $y_1(t)$ , respectiv  $y_2(t)$  și soluția analitică  $y_e(t)$  se realizează cu instrucțiunile  $e1=abs(ye(t1)-y1)$  și  $e2=abs(ye(t2)-y2)$ .
- Din punct de vedere al reprezentării grafice a soluțiilor (figura 6.25, a), nu se constată diferențe semnificative între soluția analitică  $y_e(t)$  și cele două soluții numerice  $y_1(t)$  și  $y_2(t)$ .
- Din punct de vedere al erorilor absolute (figura 6.25, b), folosirea procedurii ode45 conduce la erori  $e_2$  mult mai mici față de erorile  $e_1$  specifice utilizării procedurii ode23.

### Problema 6.23

Se consideră ecuația diferențială ordinară de ordinul 1:

$$\dot{y}(t) = y \cdot (2 - t)$$

și intervalul de integrare  $t=1 \div 4$ .

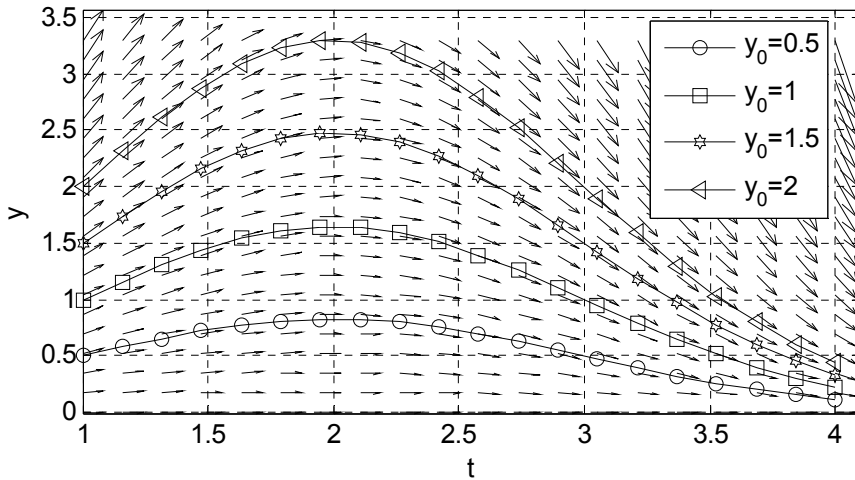
Să se determine soluțiile ecuației diferențiale folosind procedura ode45, și să se reprezinte grafic curbele integrale corespunzătoare următoarelor condiții inițiale  $y_0=\{0,5; 1,0; 1,5; 2,0\}$ . Să se reprezinte de asemenea, câmpul de vectori tangenți la curbele integrale ale ecuației diferențiale. Pentru fiecare punct  $(t, y)$  al reprezentării grafice, vectorul tangent curbei integrale este centrat în punctul respectiv și are panta egală cu  $\dot{y}(t) = dy/dt = f(t, y)$ , în care funcția pantă este  $f(t, y) = y \cdot (2 - t)$ .

### Rezolvare

Pentru rezolvarea problemei se definește un fișier de tip script conținând următoarele instrucțiuni principale:

```
%% REZOLVAREA ECUATIILOR DIFERENTIALE ORDINARE (5)
clear all;close all;clc;
%% DATE DE INTRARE
% Functia de integrat
f=@(t,y) y.*(2-t);
% Domeniul de integrare
tmin=1;tmax=4;nt=20;
t=linspace(tmin,tmax,nt);
% Condițiile initiale
y0=[0.5 1 1.5 2];ny0=length(y0);
%% SOLUTIILE ECUATIEI DIFERENTIALE
% Definirea parametrilor de formatare
s={'-ko','-ks','-kh','-k<'};
% Reprezentarea grafica a curbelor integrale
figure
hold on;
for i=1:ny0
    [t,y]=ode45(f,t,y0(i));
    plot(t,y,s{i});
end
% Definirea domeniului matriceal
[T,Y]=meshgrid(t,linspace(0,max(y),nt));
% Definirea componentelor vectorilor
DY=f(T,Y);DT=ones(size(DY));
% Reprezentarea grafica a vectorilor
quiver(T,Y,DT,DY,'k','AutoScaleFactor',2);
% Formatarea graficului
xlabel('t');ylabel('y');
grid on;box on;axis tight;hold off;
```

În urma lansării în execuție a fișierului script se obține reprezentarea grafică din figura 6.27.



**Figura 6.27.** Câmpul vectorilor tangenți curbelor integrale și curbele integrale corespunzătoare condițiilor inițiale  $y_0 = \{0,5; 1,0; 1,5; 2,0\}$ .

### Observații

- Condițiile inițiale sunt definite sub forma unui vector cu următoarele componente  $y_0 = [0.5 \ 1 \ 1.5 \ 2]$ . Numărul de condiții inițiale se obține prin instrucțiunea  $ny_0 = \text{length}(y_0)$ .
- Rezolvarea ecuației diferențiale și obținerea curbelor integrale pentru cele  $ny_0$  condiții inițiale considerate se realizează cu următoarea structură iterativă cu contor:

```
for i=1:ny0
    [t,y]=ode45(f,t,y0(i));
    plot(t,y,s{i});
end
```

Această structură iterativă realizează și reprezentarea grafică a tuturor celor  $ny_0$  curbe integrale. Atribuirea unor parametri de formatare diferiți pentru fiecare curbă integrală se realizează prin parametrul  $s\{i\}$ . Astfel, curba integrală cu indicele  $i$ , va avea parametrii de formatare de pe poziția corespunzătoare din vectorul definit anterior  $s = \{ '-ko', '-ks', '-kh', '-k<' \}$ .

- Reprezentarea grafică a câmpului de vectori tangenți curbelor integrale ale ecuației diferențiale se realizează cu instrucțiunea: `quiver(T,Y,DT,DY,'k','AutoScaleFactor',2)`. Cele două componente ale vectorilor se determină conform relațiilor:

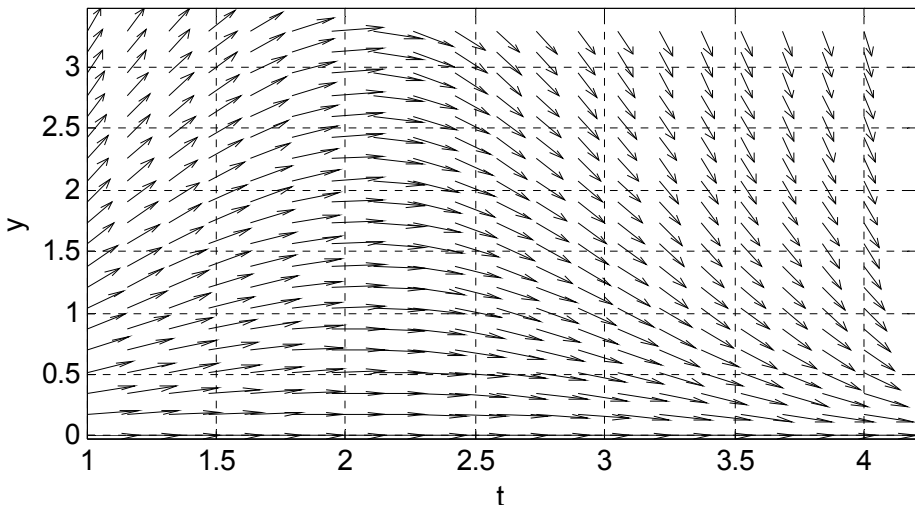
$$\dot{y}(t) = \frac{dy}{dt} = \frac{f(t,y)}{1} \Rightarrow \begin{cases} dy = f(t,y) \\ dt = 1 \end{cases}$$

cu instrucțiunile  $DY=f(T, Y)$  și  $DT=ones(size(DY))$ .

- Culoarea vectorilor este specificată prin parametrul 'k', iar lungimea vectorilor este determinată prin parametrul de scalare 'AutoScaleFactor'. Indiferent de valoarea factorului de scalare, dimensiunea fiecărui vector este proporțională cu valoarea pantei în punctul considerat, motiv pentru care lungimea vectorilor este diferită în puncte în care panta este diferită.
- În cazul în care la reprezentarea câmpului vectorial interesează doar direcția nu și valoarea numerică a pantei în punctul considerat, printr-un proces de normalizare a componentelor vectorilor se pot utiliza vectori având aceeași lungime. În acest scop, fișierul script se completează cu următoarele instrucțiuni:

```
%% CAMP DE VECTORI UNIFORMI
% Scalarea uniforma a vectorilor
DYn=DY./sqrt(DY.^2+DT.^2);
DTn=DT./sqrt(DY.^2+DT.^2);
% Reprezentarea grafica a campului de vectori
figure
quiver(T,Y,DTn,DYn,'k');
% Formatarea graficului
xlabel('t');ylabel('y');
grid on;box on;
axis tight;
```

În urma lansării în execuție a fișierului script completat se obține și reprezentarea grafică din figura 6.28.



**Figura 6.28.** Câmpul vectorilor normalizați tangenți curbelor integrale.

### 6.7.3. Rezolvarea numerică a ecuațiilor diferențiale ordinare de ordinul 2

Se consideră ecuația diferențială ordinară de ordinul 2:

$$\begin{cases} y''(t) = f(t, y, y') \\ y(t_0) = y_0 \\ y'(t_0) = y'_0 \end{cases}$$

Folosind schimbarea de variabile:

$$\begin{cases} y_1 = y \\ y_2 = y' \end{cases}$$

ecuația diferențială de ordinul 2 se transformă în următorul sistem de două ecuații diferențiale de ordinul 1:

$$\begin{cases} y'_1 = y_2 \\ y'_2 = f(t, y_1, y_2) \\ y_1(t_0) = y_0 \\ y_2(t_0) = y'_0 \end{cases}$$

Forma generală a sistemului de ecuații diferențiale este:

$$\begin{cases} y'_1 = f_1(t, y_1, y_2) \\ y'_2 = f_2(t, y_1, y_2) \\ y_1(t_0) = y_{10} \\ y_2(t_0) = y_{20} \end{cases}$$

#### Observații

- În cazul în care funcțiile  $f_1$  și  $f_2$  nu depind de variabila  $t$ , sistemul se numește autonom. În caz contrar, sistemul se numește neautonom.
- În cazul în care funcțiile  $f_1$  și  $f_2$  se pot exprima sub forma:

$$\begin{cases} f_1(y_1, y_2) = a_{11}y_1 + a_{12}y_2 \\ f_2(y_1, y_2) = a_{21}y_1 + a_{22}y_2 \end{cases}$$

atunci definesc un sistem de ecuații diferențiale liniare cu coeficienți constanți. În acest caz, sistemul de ecuații diferențiale se poate exprima prin relația:

$$Y' = A \cdot Y$$

în care  $Y'$  este vectorul derivatelor necunoscute,  $A$  este matricea coeficienților, iar  $Y$  este vectorul funcțiilor necunoscute:

$$\begin{aligned} Y' &= \begin{pmatrix} y'_1 \\ y'_2 \end{pmatrix} \\ A &= \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \\ Y &= \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \end{aligned}$$

- Toate condițiile teoremei de existență și unicitate a soluțiilor ecuațiilor diferențiale se presupune a fi satisfăcute, [41].



### Problema 6.24

Mișcarea oscilatorie a unui lichid într-un tub în formă de U cu ramuri egale, pentru ipoteza regimului de curgere laminar, este modelată de ecuația diferențială de ordinul 2, [10]:

$$\ddot{y} + 2\delta\dot{y} + \omega_0^2 y = 0$$

în care:  $\omega_0 = \sqrt{2g/l}$  este pulsația naturală (proprie) a sistemului;  $l$  este lungimea coloanei de lichid;  $g$  este accelerația gravitațională;  $\delta = 16\nu/D^2$  este coeficientul de amortizare;  $\nu$  este coeficientul de vâscozitate cinematică;  $D$  este diametrul celor două ramuri ale tubului U.

Să se rezolve ecuația diferențială și să se studieze influența diametrului asupra comportării sistemului. Se cunosc următoarele valori numerice:  $l=2,3$  m;  $g=9.81$  m/s<sup>2</sup>; lichidul de lucru este MIL-F 83282 cu  $\nu=35,9115 \cdot 10^{-6}$  m<sup>2</sup>/s (la 20 °C). Condițiile inițiale sunt: denivelarea inițială  $y_0=1$  m și viteza inițială  $\dot{y}_0=0$  m/s.

### Rezolvare

Folosind schimbarea de variabile:

$$\begin{cases} y_1 = y \\ y_2 = \dot{y} \end{cases}$$

ecuația diferențială de ordinul 2 se transformă în următorul sistem de două ecuații diferențiale de ordinul 1:

$$\begin{cases} \dot{y}_1 = y_2 \\ \dot{y}_2 = -2\delta y_2 - \omega_0^2 y_1 \\ y_1(t_0) = y_0 \\ y_2(t_0) = \dot{y}_0 \end{cases}$$

Pulsația naturală a sistemului are valoarea  $\omega_0 = \sqrt{2g/l}=2,9207$  rad/s. Caracterul răspunsului sistemului dinamic depinde de semnul expresiei  $\xi^2 - 1$ , în care  $\xi = \delta/\delta_{cr}$  este raportul de amortizare, iar  $\delta_{cr} = \omega_0$  este coeficientul critic de amortizare. Din relația  $\delta_{cr} = \omega_0$  rezultă valoarea critică a diametrului tubului în formă de U:  $D_{cr}=14$  mm.

Astfel:

- Pentru  $\xi > 1 \Rightarrow D > D_{cr}$  (de exemplu 18 mm), regim cu amortizare supracritică caracterizat printr-un răspuns aperiodic amortizat.
- Pentru  $\xi = 1 \Rightarrow D = D_{cr}$  (14 mm), regim cu amortizare critică caracterizat printr-un răspuns aperiodic amortizat.
- Pentru  $\xi < 1 \Rightarrow D < D_{cr}$  (de exemplu 10 mm), regim cu amortizare subcritică caracterizat printr-un răspuns oscilatoriu amortizat.

Pentru rezolvarea problemei se definește un fișier de tip script conținând următoarele instrucțiuni principale:

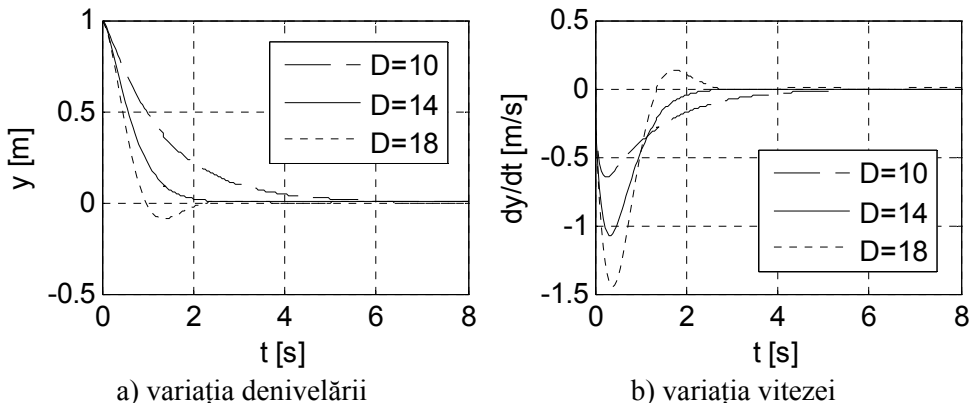
```
%% OSCILATII IN TUB U
clear all;close all;clc;
```

```

%% DATE DE INTRARE
% Acceleratia gravitacionala
g=9.81;
% Lungimea coloanei de lichid
l=2.3;
% Vascozitatea cinematica a lichidului
niu=35.9115*10^(-6);
% Pulsatia naturala a sistemului
omega0=sqrt(2*g/l);
% Diametrul critic al tubului
Dcr=floor(sqrt(16*niu/omega0)*1000)/1000;
% Diametrul tubului U
D=[Dcr-0.004 Dcr Dcr+0.004];nD=length(D);
% Coeficientul de amortizare
delta=@(D) 16*niu./D.^2;
% Domeniul de integrare
tmin=0;tmax=8;nt=200;t=linspace(tmin,tmax,nt);
% Condițiile initiale
y0=[1;0];
% Definirea figurii 1, y(t)
fig1=figure(1);hold all;axe1=gca;
% Definirea figurii 2, dy/dt(t)
fig2=figure(2);hold all;axe2=gca;
% Definirea parametrilor de formatare ai curbelor
s={'--k','-k',':k'};
% Calculul si reprezentarea grafica a solutiilor
for i=1:nD
    f=@(t,y) [y(2); -2*delta(D(i))*y(2)-omega0^2*y(1)];
    [t,y]=ode45(f,t,y0);
    plot(axe1,t,y(:,1),s{i});hold all;
    plot(axe2,t,y(:,2),s{i});hold all;
end

```

În urma lansării în execuție a fișierului script se obține reprezentarea grafică din figura 6.29.



**Figura 6.29.** Soluția problemei mișcării oscilatorii a unui lichid într-un tub în formă de U.

## Observații

- Valorile caracteristice ale diametrului tubului U sunt definite prin instrucțiunea  $D = [D_{cr} - 0.004 \quad D_{cr} \quad D_{cr} + 0.004]$ . Numărul de valori se obține prin instrucțiunea  $nD = \text{length}(D)$ .
- Coeficientul de amortizare fiind dependent de diametrul tubului, se definește ca o funcție de tip anonymous prin instrucțiunea  $\text{delta} = @(D) \ 16 * \text{niu} ./ D.^2$ .
- Rezolvarea ecuației diferențiale și obținerea curbelor integrale pentru valorile diferite ale diametrului tubului U se realizează cu următoarea structură iterativă cu contor:

```
for i=1:nD
    f=@(t,y) [y(2); -2*delta(D(i))*y(2)-omega0^2*y(1)];
    [t,y]=ode45(f,t,y0);
    plot(axe1,t,y(:,1),s{i});hold all;
    plot(axe2,t,y(:,2),s{i});hold all;
end
```

Această structură iterativă realizează definirea sistemului de ecuații diferențiale, rezolvarea sistemului folosind procedura ode45 și reprezentarea grafică a tuturor celor nD curbe integrale atât pentru denivelarea  $y(t)$ , cât și pentru viteza  $\dot{y}(t) = dy/dt$ . Atribuirea unor parametri de formatare diferiți pentru fiecare curbă integrală se realizează prin parametrul  $s\{i\}$ . Astfel, curba integrală cu indicele i, va avea parametrii de formatare de pe poziția corespunzătoare din vectorul definit anterior  $s = \{ '-k', '-k', ':k' \}$ .

- În urma executării structurii iterative vor rezulta două figuri, una pentru reprezentarea funcției  $y(t)$  și cealaltă pentru reprezentarea funcției  $\dot{y}(t)$ . În acest scop în structura instrucțiunilor plot corespunzătoare se specifică și axele în care se reprezintă curbele denivelării (axe1), respectiv ale vitezei (axe2). Cele două axe, împreună cu figurile corespunzătoare sunt definite anterior prin instrucțiunile:  $\text{fig1} = \text{figure}(1); \text{hold all}; \text{axe1} = \text{gca}$  și  $\text{fig2} = \text{figure}(2); \text{hold all}; \text{axe2} = \text{gca}$ . Prin această metodă se realizează accesul independent la fiecare figură atât în faza de reprezentare grafică (în interiorul structurii iterative), cât și în faza de formatare a graficelor (în exteriorul structurii iterative).
- De exemplu, pentru formatarea primei figuri, reprezentând variația denivelării  $y(t)$ , se utilizează instrucțiunile:

```
% Formatarea figurii 1
set(fig1,'Position',[100 100 275 200]);
xlabel(axe1,'t [s]');ylabel(axe1,'y [m]');
set(axe1,'Xgrid','on','Ygrid','on','Box','on');
legend(axe1,'D=10','D=14','D=20');
```

### Problema 6.25

Se consideră modelul clasic Lotka-Volterra pentru descrierea dinamicii a două populații  $y_1(t)$  și  $y_2(t)$ , [2, 7, 9]:

$$\begin{cases} \dot{y}_1 = a_{11}y_1 - a_{12}y_2 \\ \dot{y}_2 = a_{21}y_1 - a_{22}y_2 \end{cases}$$

în care:  $y_1(t)$  reprezintă populația pradă;  $y_2(t)$  reprezintă populația prădătoare;  $a_{11}>0$  reprezintă rata de creștere a populației pradă în absența prădătorilor;  $a_{12}>0$  reprezintă rata de scădere a populației pradă ca urmare a consumării acesteia de către prădători (agresivitatea prădătorului asupra prăzii);  $a_{21}>0$  reprezintă rata de creștere a populației de prădători ca urmare a consumării populației pradă;  $a_{22}>0$  reprezintă rata de scădere a populației de prădători ca urmare a absenței prăzii (mortalitatea prădătorilor în absența populației pradă).

Sistemul de ecuații diferențiale se rescrie sub forma:

$$\begin{cases} \dot{y}_1 = a_{11}y_1 - a_{12}y_1y_2 = f_1(y_1, y_2) \\ \dot{y}_2 = a_{21}y_1y_2 - a_{22}y_2 = f_2(y_1, y_2) \end{cases}$$

Condițiile inițiale sunt:

$$\begin{cases} y_1(t_0) = 2 \\ y_2(t_0) = 1 \end{cases}$$

Să se rezolve sistemul de ecuații diferențiale și să se reprezinte grafic soluțiile  $y_1(t)$  și  $y_2(t)$  pentru următoarele seturi de valori ale coeficienților:

$$S_1: a_{11}=1,0; a_{12}=1,0; a_{21}=1,0; a_{22}=1,0$$

$$S_2: a_{11}=1,0; a_{12}=0,5; a_{21}=1,0; a_{22}=1,0$$

$$S_3: a_{11}=1,0; a_{12}=2,0; a_{21}=1,0; a_{22}=1,0$$

Să se reprezinte grafic orbitele corespunzătoare din spațiul fazelor.

Să se reprezinte grafic, de asemenea, câmpul de vectori asociați spațiului fazelor, precum și punctul de echilibru nenul pentru fiecare set de valori. Să se determine natura punctului de echilibru.

### Rezolvare

Punctele de echilibru se determină rezolvând sistemul de ecuații:

$$\begin{cases} \dot{y}_1 = 0 \\ \dot{y}_2 = 0 \end{cases} \Rightarrow \begin{cases} 0 = (a_{11} - a_{12}y_2)y_1 \\ 0 = (a_{21}y_1 - a_{22})y_2 \end{cases}$$

Se obțin două puncte de echilibru având coordonatele:

$$\begin{aligned} &(0; 0) \\ &\left(\frac{a_{22}}{a_{21}}; \frac{a_{11}}{a_{12}}\right) \end{aligned}$$

Natura punctelor de echilibru se obține analizând valorile proprii ale Jacobianului sistemului de ecuații aplicat în cele două puncte de echilibru.

Jacobianul sistemului de ecuații se obține prin relațiile:

$$J(y_1, y_2) = \begin{pmatrix} \frac{\partial f_1}{\partial y_1} & \frac{\partial f_1}{\partial y_2} \\ \frac{\partial f_2}{\partial y_1} & \frac{\partial f_2}{\partial y_2} \end{pmatrix} = \begin{pmatrix} a_{11} - a_{12}y_2 & -a_{12}y_1 \\ a_{21}y_2 & a_{21}y_1 - a_{22} \end{pmatrix}$$

Calculul Jacobianului în cele două puncte de echilibru conduce la:

$$J(0,0) = \begin{pmatrix} a_{11} & 0 \\ 0 & -a_{22} \end{pmatrix}$$

$$J\left(\frac{a_{22}}{a_{21}}, \frac{a_{11}}{a_{12}}\right) = \begin{pmatrix} 0 & -\frac{a_{12}a_{22}}{a_{21}} \\ \frac{a_{11}a_{21}}{a_{12}} & 0 \end{pmatrix}$$

Valorile proprii  $\lambda$  se determină rezolvând ecuația caracteristică:

$$\det(J - \lambda I) = 0$$

în care  $I$  este matricea identitate.

Pentru punctul de echilibru  $(0; 0)$ , se obțin valorile proprii:

$$\begin{cases} \lambda_1 = a_{11} \\ \lambda_2 = -a_{22} \end{cases}$$

Pentru punctul de echilibru  $\left(\frac{a_{22}}{a_{21}}; \frac{a_{11}}{a_{12}}\right)$ , se obțin valorile proprii:

$$\begin{cases} \lambda_1 = i\sqrt{a_{11}a_{22}} \\ \lambda_2 = -i\sqrt{a_{11}a_{22}} \end{cases}$$

Analiza valorilor proprii  $\lambda_1$  și  $\lambda_2$ , conduce la următoarele cazuri particulare:

- $\lambda_1 \in \mathbb{R}, \lambda_2 \in \mathbb{R}, \lambda_1 \neq \lambda_2, \lambda_1 > 0, \lambda_2 < 0$ : șa instabilă.
- $\lambda_1 \in \mathbb{R}, \lambda_2 \in \mathbb{R}, \lambda_1 \neq \lambda_2, \lambda_1 < 0, \lambda_2 < 0$ : nod asimptotic stabil (puț, nod atractiv).
- $\lambda_1 \in \mathbb{R}, \lambda_2 \in \mathbb{R}, \lambda_1 = \lambda_2 < 0$ : nod asimptotic stabil (puț, nod atractiv).
- $\lambda_1 \in \mathbb{R}, \lambda_2 \in \mathbb{R}, \lambda_1 = 0, \lambda_2 < 0$ : dreaptă infinită de puncte de echilibru atractive.
- $\lambda_1 \in \mathbb{R}, \lambda_2 \in \mathbb{R}, \lambda_1 \neq \lambda_2, \lambda_1 > 0, \lambda_2 > 0$ : nod instabil (sursă, nod repulsiv).
- $\lambda_1 \in \mathbb{R}, \lambda_2 \in \mathbb{R}, \lambda_1 = \lambda_2 > 0$ : nod instabil (sursă, nod repulsiv).
- $\lambda_1 \in \mathbb{R}, \lambda_2 \in \mathbb{R}, \lambda_1 = 0, \lambda_2 > 0$ : dreaptă infinită de puncte de echilibru repulsive.
- $\lambda_1 \in \mathbb{C}, \lambda_2 \in \mathbb{C}, \lambda_{1,2} = \pm bi, b > 0$ : centru stabil.
- $\lambda_1 \in \mathbb{C}, \lambda_2 \in \mathbb{C}, \lambda_{1,2} = a \pm bi, a < 0, b > 0$ : spirală asimptotic stabilă (puț, focar atractiv).
- $\lambda_1 \in \mathbb{C}, \lambda_2 \in \mathbb{C}, \lambda_{1,2} = a \pm bi, a > 0, b > 0$ : spirală instabilă (sursă, focar repulsiv).

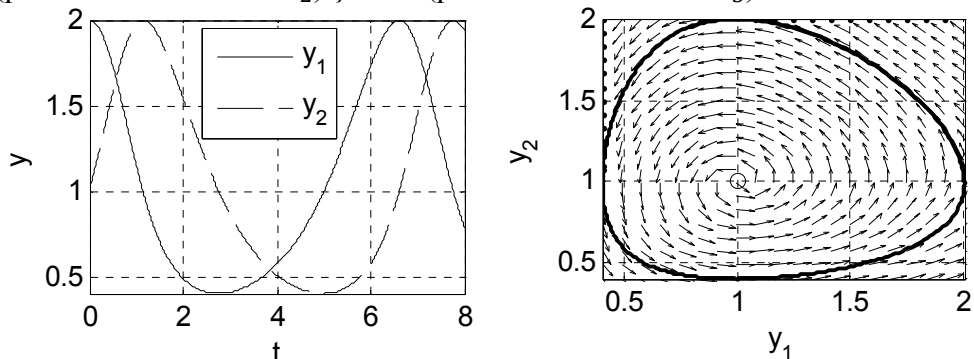
Pentru rezolvarea problemei se definește un fișier de tip script conținând următoarele instrucțiuni principale:

```

%% SISTEMUL LOTKA-VOLTERRA
clear all;close all;clc;
%% DATE DE INTRARE
% Coeficientii sistemului
a11=1;a12=1;a21=1;a22=1;
% Functiile de integrat
f1=@(t,y) a11*y(1)-a12*y(1).*y(2);
f2=@(t,y) a21*y(1).*y(2)-a22*y(2);
f=@(t,y) [f1(t,y);f2(t,y)];
% Domeniul de integrare
tmin=0;tmax=8;nt=400;t=linspace(tmin,tmax,nt);
% Condițiile initiale
y0=[2;1];
%% SOLUTIA NUMERICA
[t,y]=ode45(f,t,y0);
%% REPREZENTARI GRAFICE
% Reprezentarea grafica a curbelor integrale
figure
plot(t,y(:,1),'-k');hold on;legend('y_1','y_2');
plot(t,y(:,2),'--k');hold off;
% Reprezentarea grafica a orbitei
figure
plot(y(:,1),y(:,2),'-k','LineWidth',2);hold on;
% Definirea domeniului vectorial
y1=linspace(min(y(:,1)),max(y(:,1)),nt/20);
y2=linspace(min(y(:,2)),max(y(:,2)),nt/20);
ny1=length(y1);ny2=length(y2);
% Definirea domeniului matriceal
[Y1,Y2]=meshgrid(y1,y2);
% Calculul celor doua componente ale vectorilor
for i=1:ny1
    for j=1:ny2
        DY1(i,j)=f1(t,[Y1(i,j) Y2(i,j)]);
        DY2(i,j)=f2(t,[Y1(i,j) Y2(i,j)]);
    end
end
% Graficul campului vectorial normalizati
Yn=sqrt(DY1.^2+DY2.^2);DY1n=DY1./Yn;DY2n=DY2./Yn;
quiver(Y1,Y2,DY1n,DY2n,'k');
% Reprezentarea grafica a punctului de echilibru
plot(a22/a21,a11/a12,'ok');
% Jacobianul punctului de echilibru
J=[0 -a12*a22/a21;a11*a21/a12 0];
% Valorile proprii
lambda=eig(J)

```

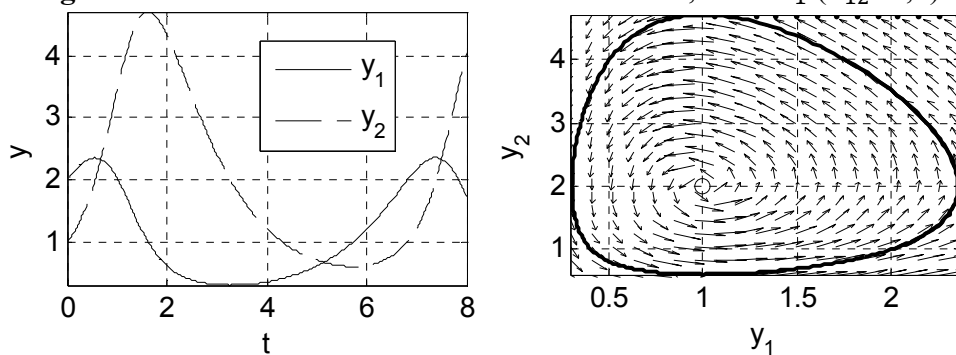
În urma lansării repetate în execuție a fișierului script (pentru fiecare set de valori ale coeficienților sistemului de ecuații diferențiale) se obțin reprezentările grafice din figurile 6.30 (pentru setul de valori  $S_1$ ), 6.31 (pentru setul de valori  $S_2$ ) și 6.32 (pentru setul de valori  $S_3$ ).



a) soluțiile  $y_1(t)$  - pradă și  $y_2(t)$  - prădător

b) orbita, câmpul de vectori normalizați și punctul fix din spațiul fazelor

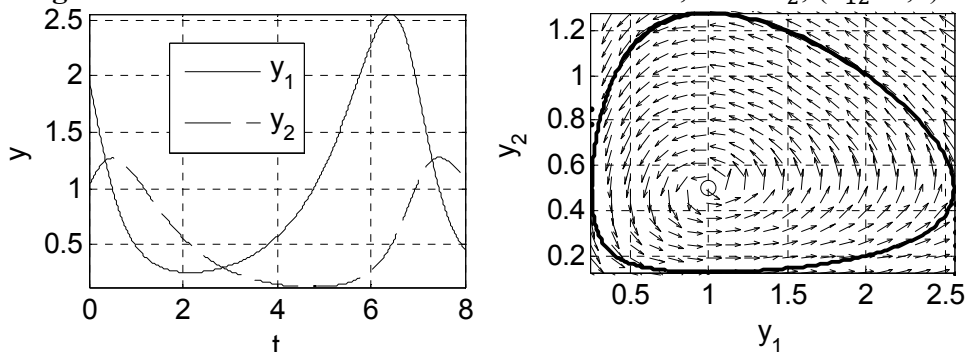
**Figura 6.30.** Rezolvarea sistemului Lotka-Volterra, cazul  $S_1$  ( $a_{12}=1,0$ )



a) soluțiile  $y_1(t)$  - pradă și  $y_2(t)$  - prădător

b) orbita, câmpul de vectori normalizați și punctul fix din spațiul fazelor

**Figura 6.31.** Rezolvarea sistemului Lotka-Volterra, cazul  $S_2$ , ( $a_{12}=0,5$ ).



a) soluțiile  $y_1(t)$  - pradă și  $y_2(t)$  - prădător

b) orbita, câmpul de vectori normalizați și punctul fix din spațiul fazelor

**Figura 6.32.** Rezolvarea sistemului Lotka-Volterra, cazul  $S_3$ , ( $a_{12}=2,0$ ).

## Observații

- Sistemul de ecuații diferențiale este definit sub forma unei funcții de tip anonymous prin instrucțiunea  $f=@(t,y) [f1(t,y); f2(t,y)]$ , în care funcțiile  $f1(t,y)$  și  $f2(t,y)$  reprezintă expresiile  $a_{11}y_1 - a_{12}y_1y_2$  și  $a_{21}y_1y_2 - a_{22}y_2$ , definite de asemenea sub forma unor funcții de tip anonymous.
- Rezolvarea sistemului de ecuații se realizează cu ajutorul procedurii ode45 prin instrucțiunea  $[t,y]=ode45(f,t,y0)$ . Cele două componente ale variabilei  $y$  reprezintă soluțiile sistemului de ecuații.
- Pentru reprezentarea grafică a celor două curbe integrale  $y_1(t)$  - prima componentă a soluției și  $y_2(t)$  - a doua componentă a soluției, se utilizează instrucțiunile:

```
plot(t,y(:,1),'-k');hold on;  
plot(t,y(:,2),'--k');hold off;
```

Identificarea clară a celor două curbe se realizează cu ajutorul unei legende pentru definirea căreia se utilizează instrucțiunea  $legend('y_1','y_2')$ .

- Pentru reprezentarea grafică a orbitei în spațiul fazelor se utilizează instrucțiunea  $plot(y(:,1),y(:,2),'-k','LineWidth',2)$ .
- Pentru reprezentarea grafică a vectorilor tangenți orbitelor din spațiul fazelor se definește domeniul matriceal al reprezentării cu ajutorul instrucțiunii  $[Y1,Y2]=meshgrid(y1,y2)$ , în care  $y1$  și  $y2$  reprezintă discretizări ale soluțiilor  $y_1(t)$  și  $y_2(t)$ . Calculul celor două componente ale vectorilor se realizează cu structura iterativă cu două contoare  $i=1:ny1$  și  $j=1:ny2$ :

```
for i=1:ny1  
    for j=1:ny2  
        DY1(i,j)=f1(t,[Y1(i,j) Y2(i,j)]);  
        DY2(i,j)=f2(t,[Y1(i,j) Y2(i,j)]);  
    end  
end
```

Reprezentarea grafică a câmpului de vectori normalizați se realizează cu instrucțiunea  $quiver(Y1,Y2,DY1n,DY2n)$ .

- Reprezentarea grafică a punctului de echilibru se realizează cu instrucțiunea  $plot(a22/a21,a11/a12,'ok')$ .
- Pentru determinarea naturii punctului de echilibru  $\left(\frac{a_{22}}{a_{21}}; \frac{a_{11}}{a_{12}}\right)$  se determina Jacobianul sistemului de ecuații diferențiale pentru punctul de echilibru respectiv cu instrucțiunea  $J=[0 \quad -a_{12} \cdot a_{22}/a_{21}; a_{11} \cdot a_{21}/a_{12} \quad 0]$ .



- Calculul valorilor proprii ale ecuației caracteristice de  $\det(J - \lambda I) = 0$  se realizează cu instrucțiunea, [34]:  $\lambda = \text{eig}(J)$ . S-au obținut următoarele valori:  $\lambda_{1,2} = \pm i$ . Rezultă deci că punctul de echilibru  $\left(\frac{a_{22}}{a_{21}}; \frac{a_{11}}{a_{12}}\right)$  este un centru stabil.
- Pentru punctul de echilibru  $(0; 0)$ , valorile proprii sunt:  $\lambda_1 = a_{11}$  și  $\lambda_2 = -a_{22}$ . Rezultă că punctul de echilibru  $(0; 0)$  este o șarpe instabilă.
- În cazul  $S_2$ , populația pradă scade mai greu ca urmare a influenței prădătorilor, ca urmare, fiind mai multă pradă și numărul prădătorilor va crește mai mult. Numărul mare de prădători are ca efect dispariția aproape completă a prăzii.
- În cazul  $S_3$ , ferocitatea prădătorilor este foarte mare, prada este puțină și, în consecință, numărul prădătorilor va avea o creștere mult mai redusă față de cazul  $S_2$ . Mai mult, prada redusă are ca efect dispariția aproape completă a prădătorilor.
- Pe baza modelului clasic Lotka-Volterra au fost definite modele evaluate pentru simularea dinamicii populațiilor care consideră diferite aspecte: relații de competiție sau de cooperare în cadrul aceleiași populații; influența resurselor asupra creșterii populației pradă; efectul controlului chimic sau biologic asupra celor două populații; reacția prădătorului la schimbarea cantității de pradă; efectul saturației prădătorilor și populației pradă, [7].

### Problema 6.26

Simularea mișcării straturilor de aer pe verticală datorită încălzirii neuniforme a atmosferei se poate realiza cu sistemul de ecuații diferențiale Lorenz, [6, 11, 42]:

$$\begin{cases} \dot{y}_1 = \delta(y_2 - y_1) = f_1(y_1, y_2, y_3) \\ \dot{y}_2 = r y_1 - y_2 - y_1 y_3 = f_2(y_1, y_2, y_3) \\ \dot{y}_3 = y_1 y_2 - b y_3 = f_3(y_1, y_2, y_3) \end{cases}$$

în care:  $y_1$  exprimă intensitatea fenomenelor atmosferice convective;  $y_2$  caracterizează diferența de temperatură dintre straturile de aer care coboară și straturile de aer care urcă;  $y_3$  exprimă abaterea distribuției de temperatură pe verticală față de variația liniară;  $\delta=10$  reprezintă numărul adimensional Prandtl;  $r=28$  este diferența de temperatură dintre stratul inferior și stratul superior;  $b=8/3$  este factorul geometric proporțional cu raportul dintre lățimea și înălțimea domeniului considerat.

Să se rezolve sistemul de ecuații diferențiale și să se reprezinte variația în timp a tuturor celor trei soluții  $y_1(t)$ ,  $y_2(t)$  și  $y_3(t)$ . Să se reprezinte în spațiul fazelor orbita corespunzătoare condițiilor inițiale  $(-8; 8; 27)$ . Să se reprezinte, de asemenea, proiecția acestei orbite în planele  $(y_1, y_2)$ ,  $(y_2, y_3)$  și  $(y_1, y_3)$ .

## Rezolvare

Pentru rezolvarea problemei se definește un fișier de tip script conținând următoarele instrucțiuni principale:

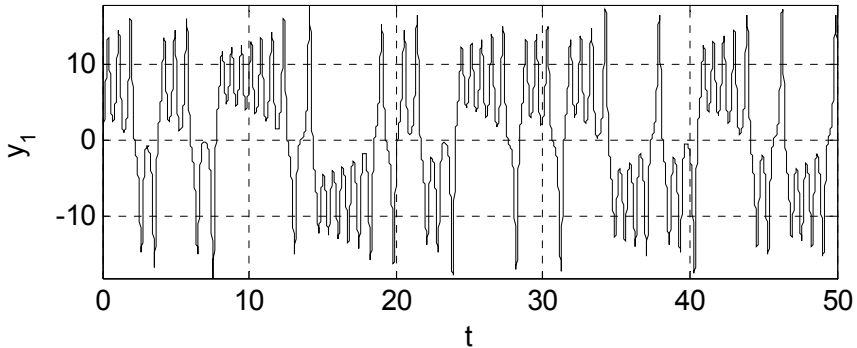
```
%% ATRACTORUL LORENZ
clear all;close all;clc;
%% DATE DE INTRARE
% Coeficientii sistemului
delta=10;r=28;b=8/3;
% Functiile de integrat
f1=@(t,y) delta*(y(2)-y(1));
f2=@(t,y) r*y(1)-y(2)-y(1).*y(3);
f3=@(t,y) y(1).*y(2)-b*y(3);
f=@(t,y) [f1(t,y);f2(t,y);f3(t,y)];
% Domeniul de integrare
tmin=0;tmax=50;nt=5000;
t=linspace(tmin,tmax,nt);
% Condițiile initiale
y0=[-8;8;27];
%% SOLUTIA NUMERICA
options=odeset('RelTol',1e-9);
[t,y]=ode45(f,t,y0,options);
%% REPREZENTARI GRAFICE
% Reprezentarea grafica a solutiilor
figure
plot(t,y(:,1),'-k');
xlabel('t');ylabel('y_1');grid on;
axis tight;
figure
plot(t,y(:,2),'-k');
xlabel('t');ylabel('y_2');grid on;
axis tight;
figure
plot(t,y(:,3),'-k');
xlabel('t');ylabel('y_3');grid on;
axis tight;
% Atractorul straniu in spatiul fazelor 3D
figure
plot3(y(:,1),y(:,2),y(:,3),'-k');
xlabel('y_1');ylabel('y_2');zlabel('y_3');grid on;
axis tight;box on;
view(-45,60);
% Proiectia atractorului straniu in planul (y1,y2)
figure
plot(y(:,1),y(:,2),'-k');
xlabel('y_1');ylabel('y_2');grid on;axis tight;
% Proiectia atractorului straniu in planul (y2,y3)
```

```

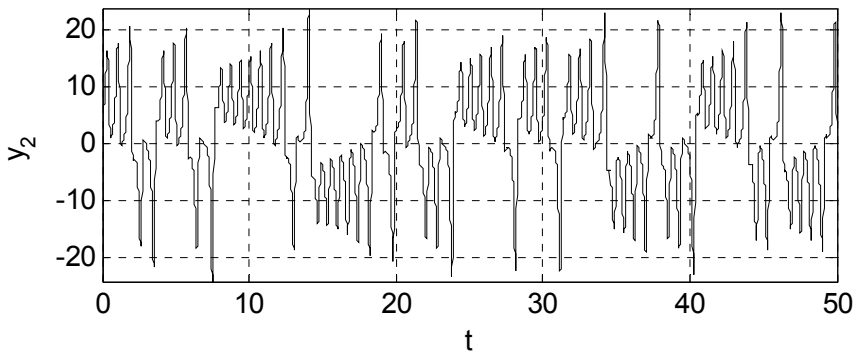
figure
plot(y(:,2),y(:,3),'-k');
xlabel('y_2');ylabel('y_3');grid on;axis tight;
% Proiectia atracteurului straniu in planul (y1,y3)
figure
plot(y(:,1),y(:,3),'-k');
xlabel('y_1');ylabel('y_3');grid on;axis tight;

```

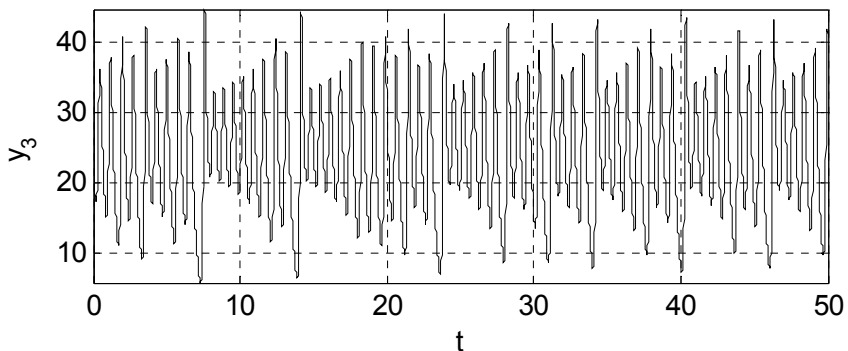
În urma lansării în execuție a fișierului script se obțin următoarele reprezentări grafice:



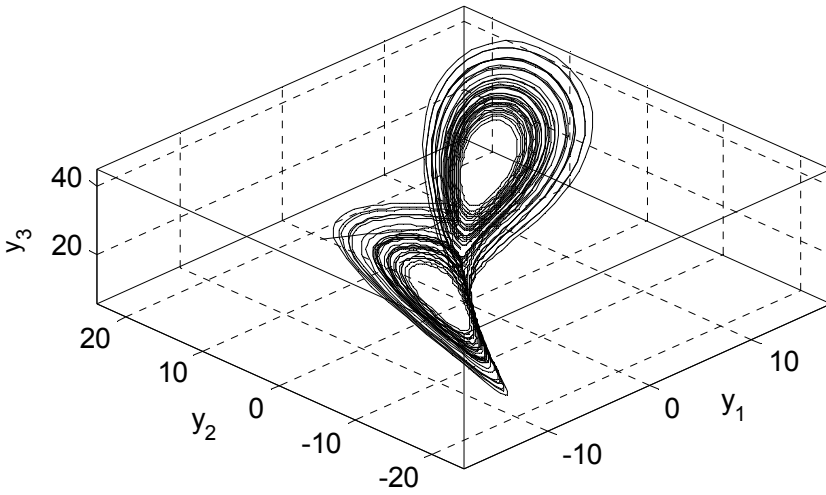
**Figura 6.33.** Componenta  $y_1(t)$ .



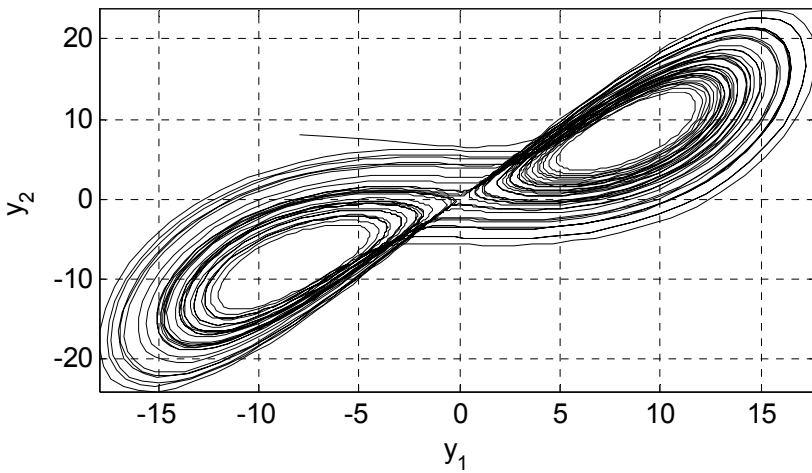
**Figura 6.34.** Componenta  $y_2(t)$ .



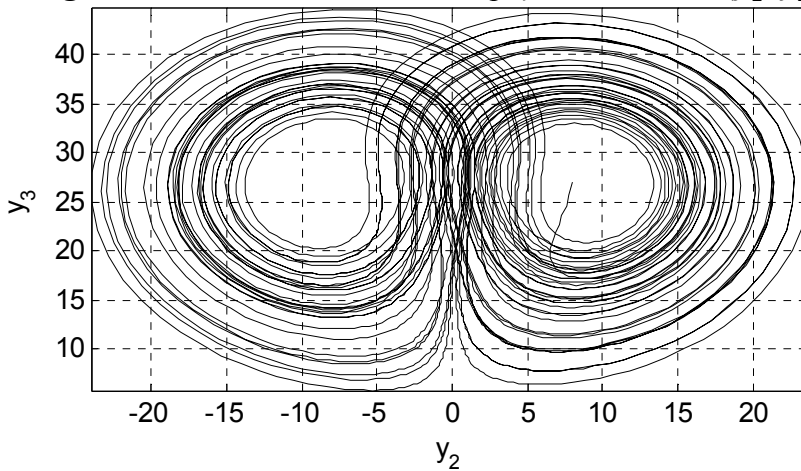
**Figura 6.35.** Componenta  $y_3(t)$ .



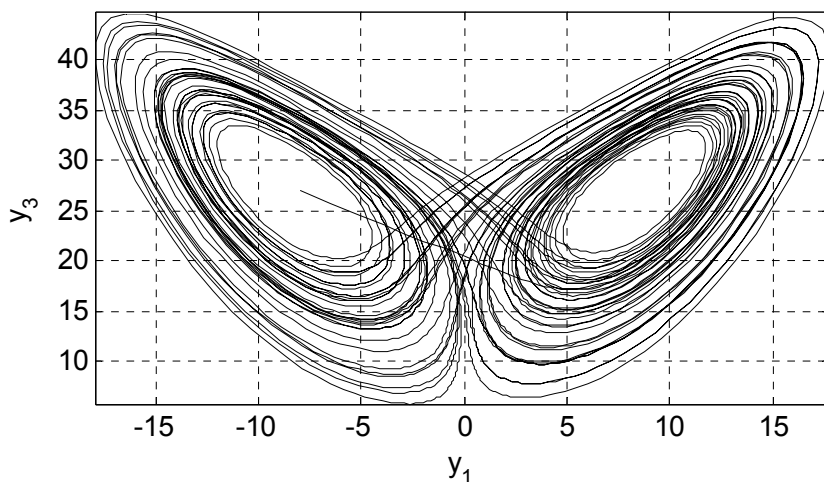
**Figura 6.36.** Atractorul Lorenz în spațiul fazelor 3D ( $y_1, y_2, y_3$ ).



**Figura 6.37.** Atractorul Lorenz în spațiul fazelor 2D, ( $y_1, y_2$ ).



**Figura 6.38.** Atractorul Lorenz în spațiul fazelor 2D, ( $y_2, y_3$ ).



**Figura 6.39.** Atractorul Lorenz în spațiul fazelor 2D,  $(y_1, y_3)$ .

### Observații

- Sistemul de ecuații diferențiale este definit sub forma unei funcții de tip anonymous prin instrucțiunea  $f=@(t,y) [f1(t,y); f2(t,y); f3(t,y)]$ , în care funcțiile  $f1(t,y)$ ,  $f2(t,y)$  și  $f3(t,y)$  reprezintă expresiile  $\delta(y_2 - y_1)$ ,  $ry_1 - y_2 - y_1y_3$  și  $y_1y_2 - by_3$ , definite de asemenea sub forma unor funcții de tip anonymous.
- Rezolvarea sistemului de ecuații se realizează cu ajutorul procedurii `ode45` prin instrucțiunea `[t,y]=ode45(f,t,y0,options)`. Cele trei componente ale variabilei  $y$  reprezintă soluțiile  $y_1(t)$ ,  $y_2(t)$  și  $y_3(t)$  ale sistemului de ecuații diferențiale.
- Reprezentarea grafică a atractorului straniu al lui Lorenz în spațiul fazelor 3D se face cu instrucțiunea `plot3(y(:,1),y(:,2),y(:,3),'-k')`. Vizualizarea atractorului în proiecție pe planele  $(y_1, y_2)$ ,  $(y_2, y_3)$  și  $(y_1, y_3)$  se realizează cu instrucțiunile `plot(y(:,1),y(:,2),'-k')`, `plot(y(:,2),y(:,3),'-k')` și `plot(y(:,1),y(:,3),'-k')`.
- Vizualizarea dinamică a modului de parcurgere a orbitei atractorului Lorenz odată cu variația timpului se poate realiza în spațiul fazelor 3D, respectiv în proiecția pe unul din planele de coordonate cu instrucțiunile, [35, 36]:

```
figure
comet3(y(:,1),y(:,2),y(:,3));
figure
comet(y(:,1),y(:,2));
```

## BIBLIOGRAFIE

1. Abbot I.H., Doenhoff A.E., Theory of Wing Sections: Including a Summary of Airfoil Data. Dover Publications, New York, 1959.
2. Baigent S., Lotka-Volterra Dynamics - An introduction, [http://www.ltcc.ac.uk/courses/BioMathematics/LTCC\\_LV2010.pdf](http://www.ltcc.ac.uk/courses/BioMathematics/LTCC_LV2010.pdf), accesat la 30.03.2014.
3. Chiriță S., Probleme de matematici superioare, EDP, București, 1989.
4. Crăciun I., Analiză matematică. Calcul integral., Editura PIM, Iași, 2007.
5. Demidovitch B., Recueil d'exercices et de problèmes d'analyse mathématique, MIR, Moscou, 1974.
6. Elert G., Strange and Complex, The Chaos Hypertextbook, <http://hypertextbook.com/chaos/>, accesat la 30.03.2014.
7. Georgescu R.M., Bifurcație în dinamica biologică cu metode de teoria grupurilor, Ed. Universității din Pitești, 2009.
8. Harman Th.L., Dabney J., Richert N., Advanced Engineering Mathematics with MATLAB, Brooks Cole, 2000.
9. Hyde D., Predator-Prey Modeling with the Lotka-Volterra Equations, <http://onlinemathcircle.com/wp-content/uploads/2012/03/Lotka-Volterra-Equations.pdf>, accesat la 30.03.2014.
10. Isbășoiu E.C.Gh., Bucur D.M., Tratat de mecanica fluidelor, Ed. AGIR, București, 2011.
11. Johns Hopkins University, Chaos and Fractals, <http://www.stsci.edu/~lbradley/seminar/index.html>, accesat la 30.03.2014.
12. Kovacs A., ș.a., Modern Numerical Methods in Engineering, Ed. Politehnica, Timișoara, 2012.
13. Lăzăroiu Gh., Sisteme de programare pentru modelare și simulare, Ed. Politehnica Press, București, 2005.
14. MathWorks, Representing Polynomials, <http://www.mathworks.com/help/matlab/math/representing-polynomials.html>, accesat la 9.02.2014.
15. MathWorks, Evaluating Polynomials, <http://www.mathworks.com/help/matlab/math/evaluating-polynomials.html>, accesat la 9.02.2014.
16. MathWorks, Convolution and Polynomial Multiplication, <http://www.mathworks.com/help/matlab/ref/conv.html>, accesat la 9.02.2014.
17. MathWorks, Deconvolution and Polynomial Division, <http://www.mathworks.com/help/matlab/ref/deconv.html>, accesat la 9.02.2014.
18. MathWorks, Polynomial Derivative, <http://www.mathworks.com/help/matlab/ref/polyder.html>, accesat la 9.02.2014.
19. MathWorks, Polynomial Roots, <http://www.mathworks.com/help/matlab/ref/roots.html>, accesat la 9.02.2014.
20. MathWorks, Polynomial with Specified Roots, <http://www.mathworks.com/help/matlab/ref/poly.html>, accesat la 9.02.2014.
21. MathWorks, Solve System of Nonlinear Equations, <http://www.mathworks.com/help/optim/ug/fsolve.html>, accesat la 9.02.2014.
22. MathWorks, Root of Nonlinear Function, <http://www.mathworks.com/help/matlab/ref/fzero.html>, accesat la 9.02.2014.

23. MathWorks, Create or Edit Optimization Options Structure, <http://www.mathworks.com/help/matlab/ref/optimset.html>, accesat la 9.02.2014.
24. MathWorks, Find Minimum of Single-Variable Function on Fixed Interval, <http://www.mathworks.com/help/matlab/ref/fminbnd.html>, accesat la 9.02.2014.
25. MathWorks, Differences and Approximate Derivative, <http://www.mathworks.com/help/matlab/ref/diff.html>, accesat la 9.02.2014.
26. MathWorks, Numerically Evaluate Integral, Adaptive Simpson Quadrature, <http://www.mathworks.com/help/matlab/ref/quad.html>, accesat la 9.02.2104.
27. MathWorks, Trapezoidal Numerical Integration, <http://www.mathworks.com/help/matlab/ref/trapz.html>, accesat la 9.02.2014.
28. MathWorks, Numerically Evaluate Double Integral Over Rectangle, <http://www.mathworks.com/help/matlab/ref/dblquad.html>, accesat la 9.02.2014.
29. MathWorks, Numerically Evaluate Double Integral, Tiled Method, <http://www.mathworks.com/help/matlab/ref/quad2d.html>, accesat la 9.02.2014.
30. MathWorks, Numerically Evaluate Double Integral, <http://www.mathworks.com/help/matlab/ref/integral2.html>, accesat la 10.02.2014.
31. MathWorks, Numerically Evaluate Triple Integral, <http://www.mathworks.com/help/matlab/ref/triplequad.html>, accesat la 9.02.2014.
32. MathWorks, Numerically Evaluate Triple Integral, <http://www.mathworks.com/help/matlab/ref/integral3.html>, accesat la 10.02.2014.
33. MathWorks, Ordinary Differential Equations, <http://www.mathworks.com/help/matlab/math/ordinary-differential-equations.html>, accesat la 22.03.2014.
34. MathWorks, Eigenvalues and Eigenvectors, <http://www.mathworks.com/help/matlab/ref/eig.html>, accesat la 30.03.2014.
35. MathWorks, 3D Comet Plot, <http://www.mathworks.com/help/matlab/ref/comet3.html>, accesat la 30.03.2014.
36. MathWorks, 2D Comet Plot, <http://www.mathworks.com/help/matlab/ref/comet.html>, accesat la 30.03.2014.
37. Năslău P., ș.a., *Matematici asistate de calculator*, Ed. Politehnica, Timișoara, 2007.
38. Olariu V., *Analiză matematică*, EDP, București, 1981.
39. Roșculeț M., *Analiză matematică*, EDP, București, 1984.
40. Stanoyevitch A., *Introduction to Numerical Ordinary and Partial Differential Equations Using MATLAB*, Wiley-Interscience, New Jersey, 2005.
41. Vrabie I., *Ecuatii diferențiale*, Ed. Universității „Al. I. Cuza”, Iași, 2012.
42. Wolfram MathWorld, Lorenz Attractor, <http://mathworld.wolfram.com/LorenzAttractor.html>, accesat la 30.04.2014.
43. Zidaru Gh., *Mișcări potențiale și hidrodinamica rețelelor de profile*, E.D.P., București, 1981.

## CAPITOLUL 7

### ANALIZA DATELOR EXPERIMENTALE

#### 7.1. ANALIZA STATISTICĂ A DATELOR EXPERIMENTALE

Se consideră un sondaj de volum  $n$  obținut prin repetarea unei măsurări în condiții practic identice. Analiza statistică a datelor experimentale presupune, în principal, parcurgerea etapelor, [17, 18]:

- **Definirea vectorului  $x$**  conținând valorile măsurate în ordinea rezultată în urma efectuării procesului de măsurare. Se utilizează metoda de definire a vectorilor prin specificarea element-cu-element a componentelor acestuia, între paranteze pătrate, separate prin spațiu sau virgulă.

- **Determinarea numărului de elemente** ale vectorului  $x$ :

$n = \text{length}(x)$

- **Ordonarea în sens crescător** a elementelor vectorului  $x$ :

$x = \text{sort}(x)$

Se obține astfel vectorul ordonat crescător al datelor experimentale:

$$x = x_1, x_2, \dots, x_{n-1}, x_n$$

astfel încât:  $x_1 = \min_{i=1 \div n} x_i$  și  $x_n = \max_{i=1 \div n} x_i$

- **Calculul parametrilor statistici principali:**

- Media aritmetică de sondaj:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$$

Se utilizează instrucțiunea:

$xm = \text{mean}(x)$

- Abateră medie pătratică de sondaj:

$$s = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2}$$

Se utilizează instrucțiunea:

$s = \text{std}(x)$



- **Stabilirea nivelului de încredere:**

$$P_1=95\%, P_2=98\% \text{ sau } P_3=99\%$$

- **Determinarea parametrului distribuției Student  $t(n-1,P)$ , în funcție de numărul de rezultate  $(n - 1)$  și de nivelul de încredere  $P$  [%], conform tabelului 7.1**

**Tabel 7.1.** Parametrul distribuției Student.

$n - 1 \backslash P$	95%	98%	99%
10	2,228	2,764	3,169
15	2,131	2,602	2,947
20	2,086	2,528	2,845
25	2,060	2,485	2,787
30	2,042	2,457	2,750
35	2,030	2,437	2,724
40	2,021	2,423	2,704
45	2,014	2,412	2,689
50	2,008	2,403	2,677
60	2,000	2,390	2,660
100	1,984	2,364	2,626

- **Determinarea intervalelor de eroare:**

$$e = t \cdot s / \sqrt{n}$$

- **Prezentarea rezultatului procesului de măsurare:**

$$X = \bar{x} \pm e$$

- **Reprezentarea grafică a histogramei repartiției cu ajutorul instrucțiunii, [1, 16]:**

$$\text{hist}(x, k)$$

în care  $x$  reprezintă vectorul datelor experimentale, iar  $k$  reprezintă numărul de clase (în mod implicit  $k=10$ ).

- **Reprezentarea grafică a erorilor.** Etapele specificate se parcurg pentru toate cele  $m$  valori de referință  $x_r$  pentru care au fost efectuate măsurători. Rezultatele finale obținute pentru o anumită valoare a nivelului de încredere, se centralizează în tabelul:

$x_r$	$x_{r1}$	$x_{r2}$	...	$x_{rm}$
$\bar{x}$	$\bar{x}_1$	$\bar{x}_2$	...	$\bar{x}_m$
$e$	$e_1$	$e_2$	...	$e_m$

Reprezentarea grafică a erorilor se realizează cu instrucțiunea:

$$\text{errorbar}(x_r, x_m, e)$$

### Problema 7.1

Se consideră un proces de măsurare care constă în efectuarea a 5 serii independente de măsurări asupra unei mărimi fizice oarecare, pentru valorile de referință:  $x_r = \{75; 81; 95; 113; 130\}$ . Datele experimentale obținute în urma măsurărilor sunt prezentate în tabelul 7.2.

**Tabel 7.2.** Datele experimentale.

Nr.	$x_r$				
	75	81	95	113	130
1	65,00	81,2500	97,500	113,7500	121,8750
2	66,50	83,1250	99,750	116,3750	124,6875
3	67,00	83,7500	101,250	117,2500	125,6250
4	67,50	83,9375	102,000	114,1250	125,9063
5	68,00	84,0625	102,750	114,5625	126,0938
6	68,15	84,3750	103,125	118,0000	126,5625
7	68,76	84,6875	108,250	125,2625	127,0313
8	68,75	84,8125	105,000	120,3300	127,2188
9	68,85	85,0000	105,375	120,3125	127,5000
10	69,25	84,1875	105,750	118,4875	122,7813
11	69,50	85,6250	103,875	116,1875	128,4375
12	69,75	85,8125	106,725	116,6250	132,7188
13	71,50	85,9375	107,250	114,0625	128,9063
14	73,85	85,0625	107,625	114,1500	122,8438
15	70,00	85,5625	107,850	115,2375	130,3125
16	70,15	86,8750	108,225	125,5000	130,7813
17	70,25	86,1875	108,375	125,7625	131,2500
18	70,50	87,5000	108,750	125,9375	131,7188
19	70,75	87,8125	109,125	123,3750	132,1875
20	71,00	88,1250	109,425	123,8125	133,1250
21	71,15	88,7500	109,725	124,2500	134,0625
22	71,50	88,9375	109,875	124,5125	134,5313
23	71,75	83,3750	110,250	125,1250	135,9375
24	72,50	89,6875	110,625	125,5625	136,4063
25	72,75	90,6250	111,225	126,8750	137,3438
26	69,25	86,9375	112,500	126,3125	136,6250
27	75,00	83,0625	109,275	126,1875	133,4063
28	67,75	82,7500	103,875	126,2500	129,0938
29	69,85	83,5000	106,000	122,2375	124,1250
30	69,90	83,8000	106,500	122,3250	122,2000
31	70,00	83,2560	106,750	115,3597	131,0126

Să se calculeze următorii parametri statistici: valorile extreme  $x_{min}$  și  $x_{max}$ ; valoarea medie  $\bar{x}$ ; abaterea medie pătratică de sondaj  $s$ ; parametrul distribuției Student  $t$  și intervalele de eroare  $e$  pentru fiecare valoare a nivelului de încredere  $P=95\%$ ,  $98\%$  și  $99\%$ .

Să se reprezinte grafic histogramele repartițiilor corespunzătoare celor cinci valori de referință. Să se reprezinte distribuția grafică a erorilor pentru cele trei valori ale nivelului de încredere.

## Rezolvare

În cazul prelucrării unor variabile conținând un mare număr de date, în particular date obținute din măsurări, se recomandă utilizarea a două fișiere: un fișier de date (existent pe o unitatea de stocare accesibilă programului) care conține valorile numerice ale variabilelor și un fișier `script` care conține instrucțiunile pentru efectuarea calculelor (în particular, calculele specifice prelucrării statistice a datelor numerice).

Generarea fișierului de date se realizează cu instrucțiunea, [5]:

```
save('NumeFisier','Variabila','Format')
```

în care `NumeFisier` este numele fișierului de date în care se vor salva, în formatul dorit `Format`, valorile numerice ale variabilei identificate prin numele `Variabila`.

Instrucțiunea de salvare a datelor poate fi scrisă și sub forma echivalentă:

```
save NumeFisier Variabila Format
```

În mod implicit, comanda `save` salvează datele în formatul binar `MAT (-mat)`, însă este posibilă utilizarea și a formatului `ASCII` cu 8 cifre semnificative (`-ascii`) sau cu 16 cifre semnificative (`-ascii -double`).

Citirea datelor dintr-un fișier de date și încărcarea acestora în spațiul de lucru al programului se realizează cu una din instrucțiunile, [6]:

```
load('NumeFisier','Format','Variabila')  
load NumeFisier Format Variabila
```

Pentru cazul analizat, salvarea datelor se realizează cu următoarele instrucțiuni:

```
x(:,1)=[65.00 66.50 ... 69.90 70.00];  
x(:,2)=[81.25 83.125 ... 83.8 83.256];  
x(:,3)=[97.5 99.75 ...106.5 106.75];  
x(:,4)=[113.75 116.375 ...122.325 115.3597];  
x(:,5)=[121.875 124.6875 ...122.2 131.0126];  
save('date_exp.txt','x','-ascii')
```

Datorită numărului mare de date experimentale se preferă introducerea separată a datelor, pe coloane (`x(:,1)`, `x(:,2)`, `x(:,3)`, `x(:,4)` și `x(:,5)`), pentru fiecare din cele 5 serii de măsurări. Se obține deci, în directorul curent, un fișier de date de tip `ASCII` cu numele `date_exp.txt`, care va conține toate cele cinci serii de rezultate experimentale sub forma unei matrice având dimensiunea  $31 \times 5$ . Fișierul de date astfel creat poate fi analizat, arhivat sau distribuit altor utilizatori.

Fișierul de tip script pentru analiza statistică a datelor experimentale conținute în fișierul de date `date_exp.txt` conține următoarele instrucțiuni:

```

%% ANALIZA STATISTICA A DATELOR EXPERIMENTALE
% Prelucrarea datelor
close all;clear all;clc;
%% DATE INITIALE
% Definirea valorilor de referinta
xr=[75 81 95 113 130];
% Citirea datelor din fisierul de date
x=load('date_exp.txt','-ascii');
%% ANALIZA STATISTICA A DATELOR
% Definirea nivelului de incredere
P=[95 98 99];
% Calculul numarului de elemente
[n,m]=size(x)
% Ordonarea valorilor vectorului
x=sort(x)
% Valoarea minima
xmin=min(x)
% Valoarea maxima
xmax=max(x)
% Media aritmetica
xm=mean(x)
% Abaterea media patratica
s=std(x)
% Parametrul t(n-1,P)
t=[2.042 2.457 2.75];
% Calculul erorii
for k=1:3
    for j=1:m
        e(k,j)=t(k)*s(j)/sqrt(n);
    end
end
%% REPREZENTAREA HISTOGRAMELOR
for j=1:m
    figure
    hist(x(:,j),10);
    xlabel('x_i');ylabel('n_i');
end
%% REPREZENTAREA ERORILOR
for k=1:3
    figure
    errorbar(xr,xm,e(k,:), '-k');
    grid on;    xlabel('x');ylabel('x_m');
end

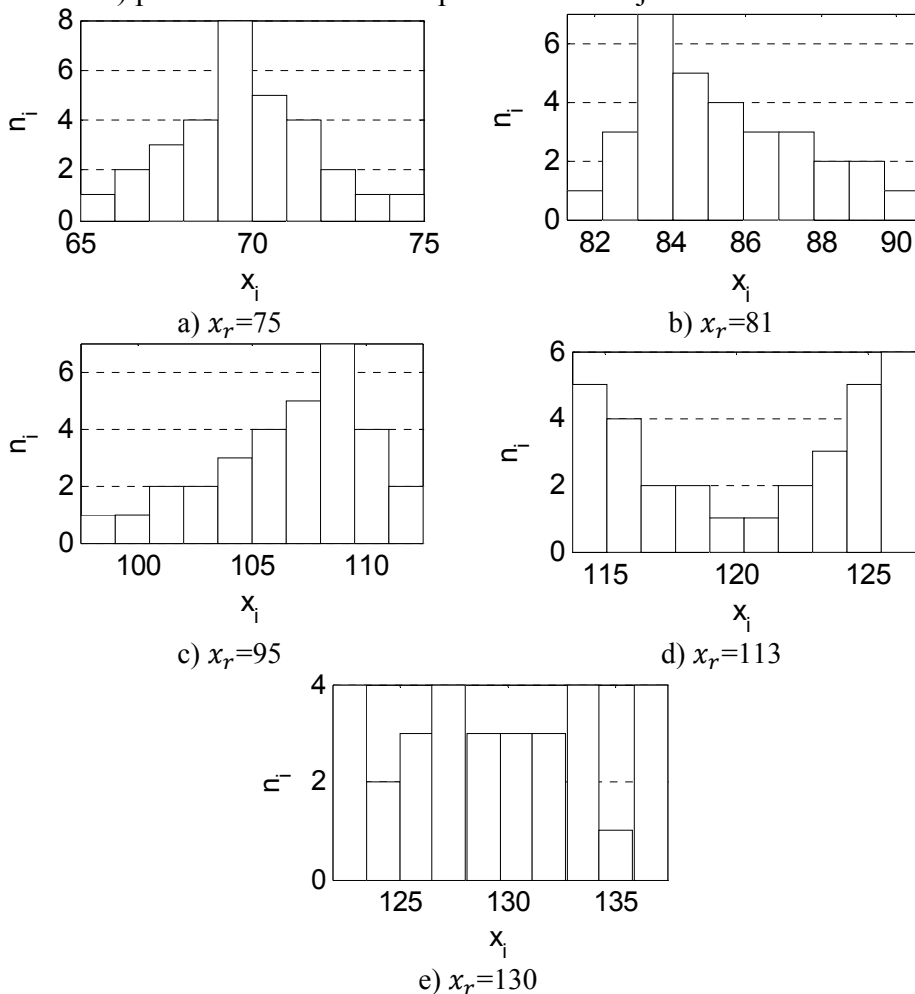
```

Rezultatele obținute sunt prezentate în tabelul 7.3.

**Tabel 7.3.** Rezultate obținute.

		$x_r$				
		75,00	81,00	95,00	113,00	130,00
$x_{min}$		65,00	81,25	97,5	113,75	121,875
$x_{max}$		75,00	90,625	112,5	126,875	137,3438
$\bar{x}$		69,8842	85,4635	106,5992	120,7775	129,4293
$s$		2,1204	2,2585	3,5136	4,7401	4,5607
$e$	95%	0,7777	0,8283	1,2886	1,7384	1,6727
	98%	0,9357	0,9966	1,5505	2,0918	2,0126
	99%	1,0473	1,1155	1,7354	2,3412	2,2526

În figura 7.1 se prezintă histogramele celor cinci serii de date experimentale (frecvențele absolute  $n_i$  în funcție de intervalele de valori  $i = 1 \dots k$ ) pentru o diviziune a amplitudinii sondajului în  $k=10$  clase.

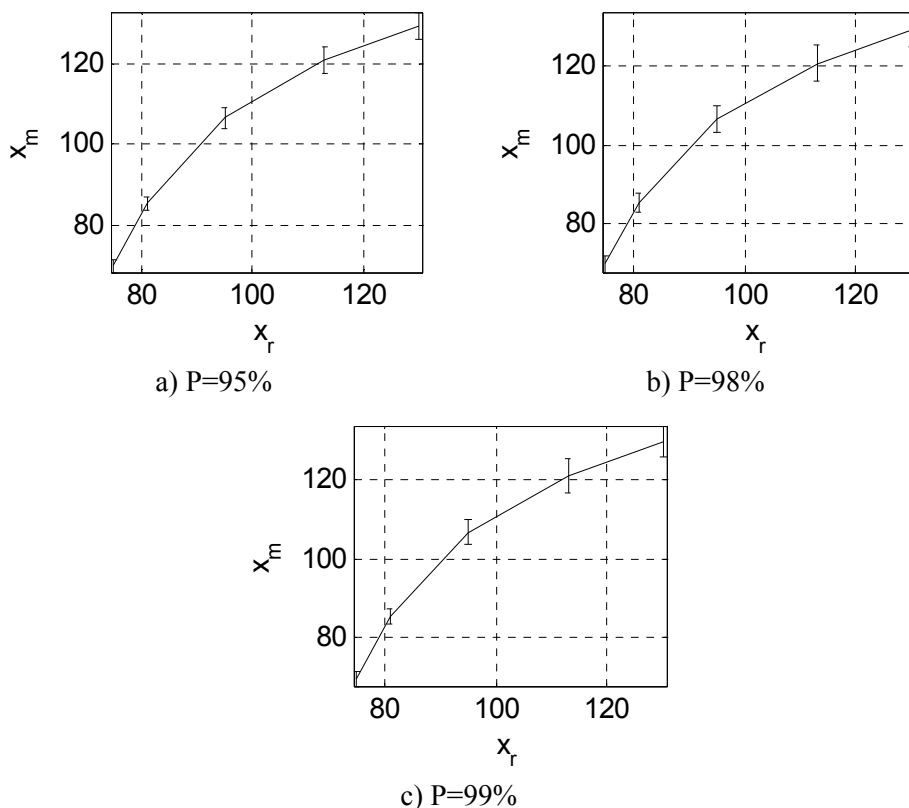


**Figura 7.1.** Histogramele repartițiilor.

### Observații

- Histograma obținută pentru  $x_r=75$  se apropie cel mai mult de repartiția normală teoretică, în timp ce histograma pentru  $x_r=130$  definește o repartiție foarte departe de repartiția normală teoretică.
- Histograma corespunzătoare seriei de date experimentale  $x_r=81$  definește o repartiție asimetrică la stânga, în timp ce histograma obținută pentru  $x_r=95$  definește o repartiție asimetrică la dreapta.
- Histograma corespunzătoare seriei de date experimentale  $x_r=113$  definește o repartiție antimodală.

În figura 7.2 se prezintă distribuția grafică a erorilor pentru setul de date experimentale, în funcție de nivelul de încredere ales.



**Figura 7.2.** Reprezentarea grafică a erorilor.

### Observații

- Din figura 7.2 și din tabelul 7.3 se observă că pentru valori ridicate ale nivelului de încredere și intervalele de eroare cresc.
- Indiferent însă de nivelul de încredere, se observă că precizia efectuării încercărilor experimentale este mai mare pentru primele valori de referință.

## 7.2. ANALIZA NUMERICĂ A DATELOR EXPERIMENTALE

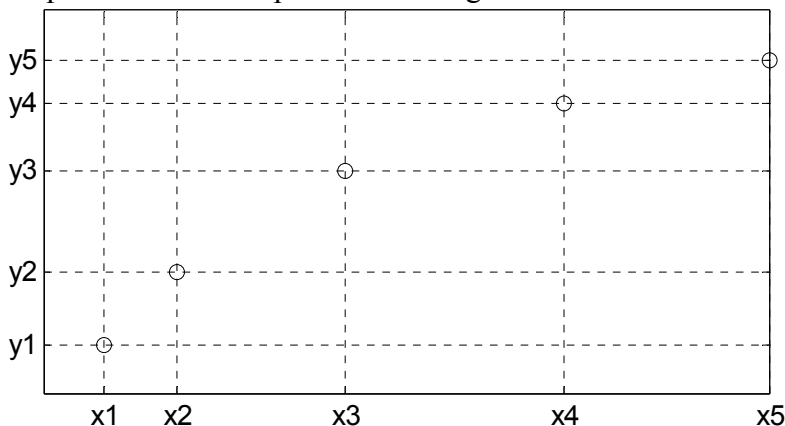
### 7.2.1. Metode de aproximare a funcțiilor

Se consideră o funcție  $f: [x_1, x_n] \rightarrow \mathbb{R}$  definită sub forma tabelară generală, [16]:

$$f(x_i) = y_i, i = 1 \dots n$$

$x$	$x_1$	$x_2$	$x_3$	$\dots$	$x_{n-1}$	$x_n$
$y$	$y_1$	$y_2$	$y_3$	$\dots$	$y_{n-1}$	$y_n$

Reprezentarea grafică a funcției constă în reprezentarea punctelor  $P_i(x_i, y_i)$ ,  $i = 1 \dots n$ . Pentru un caz particular în care funcția este cunoscută, de exemplu în  $n=5$  puncte de rețea, reprezentarea grafică a unei funcții oarecare ar putea fi de forma prezentată în figura 7.3.



**Figura 7.3.** Reprezentarea grafică a funcției tabelare,

Funcția  $f(x)$  este cunoscută doar în punctele de rețea  $x_1, x_2, \dots, x_n$ . Acesta este și motivul pentru care în reprezentarea grafică din figura 7.3 sunt vizualizate doar punctele  $P_i(x_i, y_i)$ ,  $i = 1 \dots n$ .

Problema principală, din perspectiva metodelor de aproximare a funcțiilor, este a determina ce se întâmplă între punctele  $P_i(x_i, y_i)$ , altfel spus care sunt valorile funcției  $f$  pentru argumente  $x_a$  diferite de punctele de rețea  $x_a \neq x_i$ ,  $i = 1 \dots n$ .

Rezolvarea acestei probleme constă în determinarea unei expresii analitice  $F: [x_1, x_n] \rightarrow \mathbb{R}$  care să aproximeze funcția  $f$  dată sub formă tabelară. Funcția de aproximare conține un număr  $m$  de parametri, astfel încât forma generală a funcției de aproximare este:

$$F(x_i, \alpha_j), i = 1 \dots n, j = 1 \dots m$$

Forma funcției de aproximare și valorile parametrilor acesteia se determină din condiția ca funcția  $F$  să aproximeze cât mai bine funcția  $f$ . În acest sens, se introduce noțiunea de distanță dintre cele două funcții, notată cu  $d(f, F)$  și definită prin relația:

$$d(f, F) = \sqrt{\sum_{i=1}^n [f(x_i) - F(x_i, \alpha_j)]^2}$$

Pornind de la noțiunea de distanță dintre cele două funcții, aproximarea funcțiilor se poate realiza prin două metode:

- **metoda de aproximare prin interpolare**, pentru care se impune condiția de anulare a distanței dintre cele două funcții:

$$d(f, F) = 0$$

- **metoda de aproximare prin regresie** (aproximare în medie prin metoda celor mai mici pătrate), pentru care se impune condiția de minimizare a distanței dintre cele două funcții:

$$d(f, F) = \min$$

### 7.2.2. Aproximarea prin interpolare

Aproximarea funcțiilor prin interpolare se bazează pe condiția de anulare a distanței dintre funcția tabelară  $f$  și funcția de aproximare  $F$ , adică funcția de aproximare va avea aceleași valori ca și funcția tabelară în toate punctele de rețea:

$$F(x_i, \alpha_j) = f(x_i), i = 1 \dots n,$$

Din punct de vedere grafic, curba funcției de aproximare prin interpolare trebuie în mod necesar să treacă prin toate punctele  $P_i(x_i, y_i)$ ,  $i = 1 \dots n$ , fără însă a exista nici un fel de restricție asupra modului de variație al funcției de aproximare între punctele de rețea.

Realizarea aproximării prin interpolare se realizează în mod diferit după cum punctele de aproximare  $x_a$  aparțin sau nu domeniului de definiție al funcției tabelare, [2, 7]:

- Determinarea valorilor funcției de aproximare  $F(x_a)$  pentru un set de puncte  $x_a \in [x_1, x_n]$  se realizează cu instrucțiunea:

$$y_i = \text{interp1}(x, y, x_a, 'metoda');$$

- Determinarea valorilor funcției de aproximare  $F(x_a)$  pentru un set de puncte  $x_a \notin [x_1, x_n]$  se realizează cu instrucțiunea:

$$y_e = \text{interp1}(x, y, x_a, 'metoda', 'extrap');$$

în care  $y$  reprezintă valorile funcției  $f$  în puncte de rețea  $x$ ;  $x_a$  reprezintă setul de puncte în care se dorește aproximarea funcției, iar  $metoda$  reprezintă metoda de interpolare utilizată.

Principalele metode de interpolare sunt, [3, 4]:

- Interpolarea liniară, pentru care  $metoda = 'linear'$ .
- Interpolarea cubică, pentru care  $metoda = 'cubic'$ .
- Interpolarea spline, pentru care  $metoda = 'spline'$ .



## Problema 7.2

În urma efectuării analizei statistice a datelor experimentale (problema 7.1), pentru fiecare valoare de referință  $x_r = \{75; 81; 95; 113; 130\}$  se obțin valorile medii  $\bar{x}$  corespunzătoare. Rezultatele finale ale analizei statistice reprezintă datele de intrare ale analizei numerice a datelor experimentale. Se obține deci funcția  $f: [75; 130] \rightarrow \mathbb{R}$  definită sub forma tabelară prin valorile:

$x$	75	81	95	113	130
$y = f(x)$	69,8842	85,4635	106,5992	120,7775	129,4293

Să se determine valoarea funcției în punctele  $x_{a1}=90$  și  $x_{a2}=135$  prin cele trei metode de interpolare: liniară, cubică și spline. Să se reprezinte grafic funcția tabelară și funcțiile de interpolare prin metoda liniară și spline.

## Rezolvare

Se observă că în primul caz  $x_{a1} \in [75; 130]$ , ceea ce corespunde unei aproximări prin interpolare, în timp ce în cel de-al doilea caz  $x_{a2} \notin [75; 130]$ , aproximarea realizându-se prin extrapolare, caz în care în structura instrucțiunii `interp1` trebuie să se menționeze și parametrul 'extrap'.

Pentru rezolvarea problemei se va scrie un fișier de tip `script` care va conține următoarele instrucțiuni principale:

```
x=[75 81 95 113 130];
y=[69.8842 85.4635 106.5992 120.7775 129.4293];
xa1=90;
yil=interp1(x,y,xa1,'linear')
yic=interp1(x,y,xa1,'cubic')
yis=interp1(x,y,xa1,'spline')
xa2=130;
yel=interp1(x,y,xa2,'linear','extrap')
yec=interp1(x,y,xa2,'cubic','extrap')
yes=interp1(x,y,xa2,'spline','extrap')
```

obținându-se următoarele rezultate pentru interpolare:

```
yil=99,0507
yic=100,5416
yis=100,7725
```

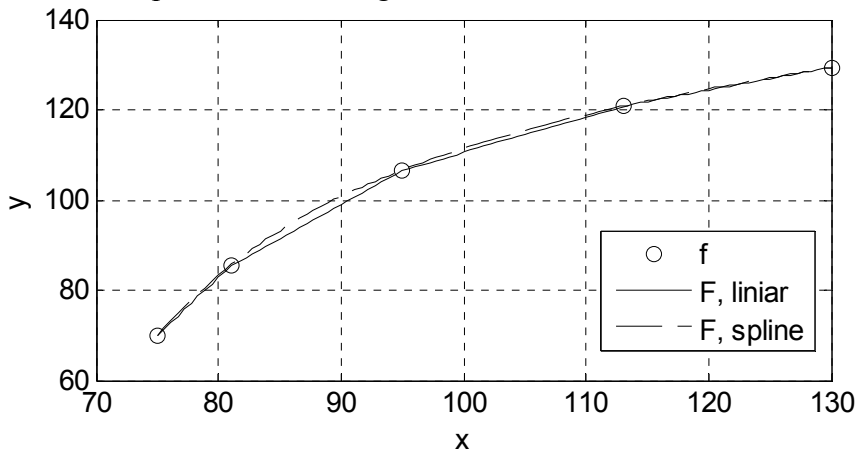
și următoarele rezultate pentru extrapolare:

```
yel=131,9739
yec=100,5416
yes=131,9222
```

Pentru reprezentarea grafică a celor două funcții de aproximare prin interpolare se va realiza o discretizare mai fină a domeniului de definiție al funcției tabelare (100 de puncte), pentru care se vor calcula valorile obținute prin interpolare, valori care ulterior vor fi reprezentate grafic. Pentru aceasta în structura aceluiași fișier de tip `script` se introduc instrucțiunile:

```
xi=linspace(75,130);
yil=interp1(x,y,xi,'linear');
yis=interp1(x,y,xi,'spline');
figure
plot(x,y,'ok');hold on;
plot(xi,yil,'-b');
plot(xi,yis,':r');
legend('f','F-liniar','F-spline');
grid on;xlabel('x');ylabel('y');hold off;
```

obținându-se reprezentarea din figura 7.4.



**Figura 7.4.** Aproximarea prin interpolare.

### Observații

- Se observă existența a trei instrucțiuni de tip `plot`, prima instrucțiune realizează reprezentarea grafică a punctelor din definiția tabelară a funcției de analizat, în timp ce celelalte două instrucțiuni de tip `plot` realizează reprezentarea grafică a funcției de interpolare de tip liniar, respectiv de tip spline.
- Pentru ca reprezentările grafice determinate de cele trei instrucțiuni de tip `plot` să fie afișate în aceeași fereastră grafică, după prima instrucțiune `plot` trebuie specificată instrucțiunea `hold on`.
- Se observă: etichetarea celor două axe ale graficului; prezența instrucțiunii `grid on`; definirea unei legende pentru identificarea clară a celor trei curbe; introducerea instrucțiunii `hold off`.

### 7.2.3. Aproximarea prin regresie

Aproximarea funcțiilor prin regresie se bazează pe condiția de minimizare a distanței dintre funcția tabelară  $f$  și funcția de aproximare  $F$ :

$$d(f, F) = \min$$

Din punct de vedere grafic, curba funcției de aproximare prin regresie nu trebuie în mod necesar să treacă prin nici unul din punctele  $P_i(x_i, y_i)$ ,  $i = 1 \dots n$ , existând însă restricția ca distanța dintre cele două curbe, pe tot domeniul de variație al celor două funcții, să fie minimă.

Pentru cazul particular al regresiei polinomiale, determinarea coeficienților polinomului de aproximare se realizează cu instrucțiunea, [8]:

```
p=polyfit(x,y,nr);
```

în care  $y$  reprezintă valorile funcției  $f$  în puncte de rețea  $x$ , iar  $nr$  reprezintă gradul polinomului de regresie utilizat pentru aproximare,

După determinarea coeficienților polinomului de regresie se va evalua acest polinom în punctul dorit.

#### Problema 7.3

Se consideră funcția tabelară definită la problema 7.2. Să se determine valoarea funcției în punctele  $x_{a1}=90$  și  $x_{a2}=135$  prin regresie polinomială de grad  $n_r=1$  și  $n_r=3$ . Să se reprezinte grafic funcția tabelară și polinoamele de regresie de grad  $n_r=1$  și  $n_r=3$ .

#### Rezolvare

Pentru rezolvarea problemei se va scrie un fișier de tip script care va conține instrucțiunile:

```
x=[75 81 95 113 130];  
y=[69.8842 85.4635 106.5992 120.7775 129.4293];  
xa=[90 135];  
p1=polyfit(x,y,1);p3=polyfit(x,y,3);  
yr1=polyval(p1,xa);yr3=polyval(p3,xa);
```

obținându-se următoarele rezultate:

```
p1=      1.0438      -0.7003  
p3=      0.0004      -0.1445      17.5123      -600.6174  
yr1=     93.2450     140.2176  
yr3=    100.7414     133.2141
```

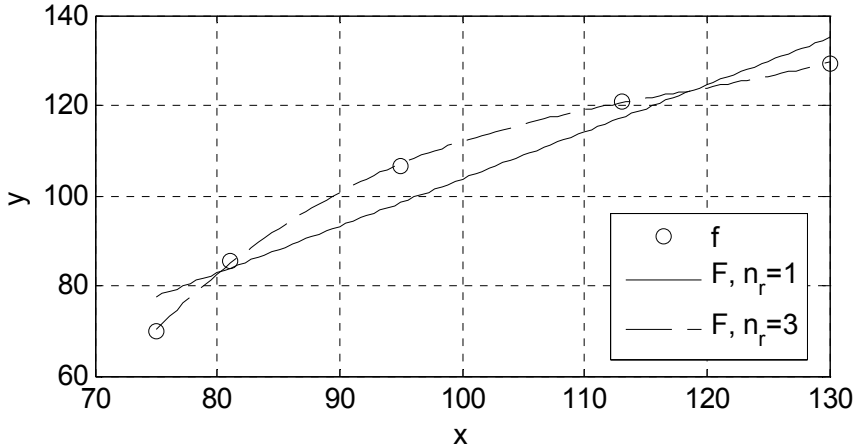
Pentru reprezentarea grafică a celor două funcții de aproximare prin regresie se va realiza o discretizare mai fină a domeniului de definiție al funcției tabelare. Pentru fiecare punct de discretizare se calculează valorile funcției prin interpolare, valori care ulterior vor fi reprezentate grafic:

```

xr=linspace(min(x),max(x));
yr1=polyval(p1,xr);
yr3=polyval(p3,xr);
figure
plot(x,y,'ok');hold on;
plot(xr,yr1,'-k');
plot(xr,yr3,'--k'); hold off;
grid on;xlabel('x');ylabel('y');
legend('f','F, nr=1','F, nr=3');

```

obținându-se reprezentarea din figura 7.5.



**Figura 7.5.** Aproximarea prin regresie.

#### 7.2.4. Aprecierea calității aproximării prin regresie

Pentru aprecierea calității aproximării prin regresie se utilizează mai multe metode: metoda analizei reziduurilor, metoda parametrilor numerici sintetici, metoda intervalelor de eroare.

Reziduurile aproximării prin regresie reprezintă diferența dintre valorile funcției tabelare și valorile polinomului de regresie în fiecare punct de rețea, [9, 10]:

$$r_i = y_i - F_i, i = 1 \dots n$$

Pentru calculul și reprezentarea grafică a reziduurilor obținute la aproximarea cu polinoame de regresie de grad  $n_r=1$  și  $n_r=3$  fișierul script anterior se completează cu instrucțiunile:

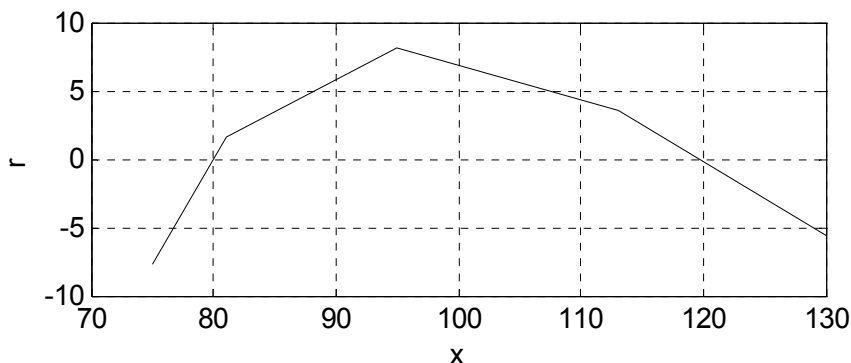
```

r1=y-yr1;
r3=y-yr3;
figure
plot(x,r1,'-k');
grid;xlabel('x');ylabel('r');
figure
plot(x,r3,'--k');
grid;xlabel('x');ylabel('r');

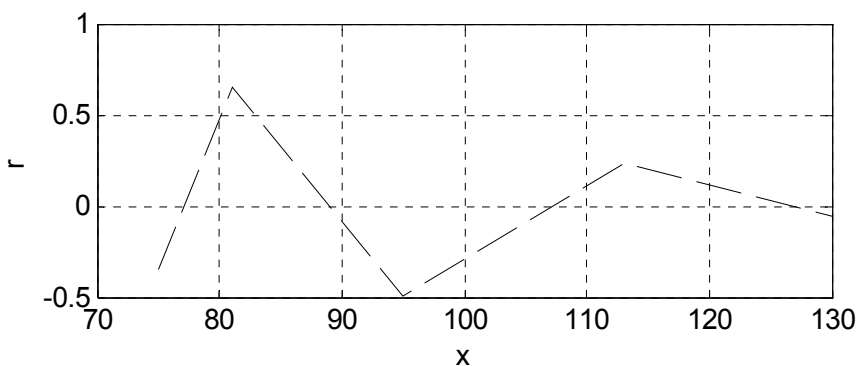
```

obținându-se reprezentarea din figura 7.6 dacă se utilizează o reprezentare grafică de tip linie, sau reprezentarea din figura 7.7 dacă se utilizează o reprezentare grafică cu bare, caz în care instrucțiunile de reprezentare grafică au forma:

```
figure
bar(x,r1,'w','LineStyle','-');grid on;
xlabel('x');ylabel('r');
figure
bar(x,r3,'w','LineStyle','--');
grid on;
xlabel('x');ylabel('r');
```



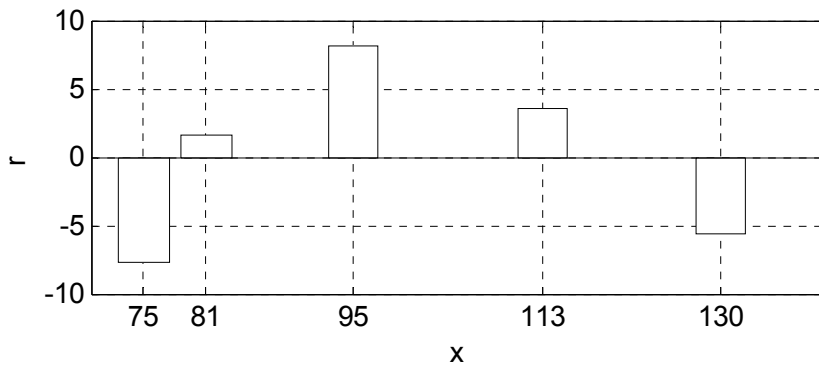
a)  $n_r=1$



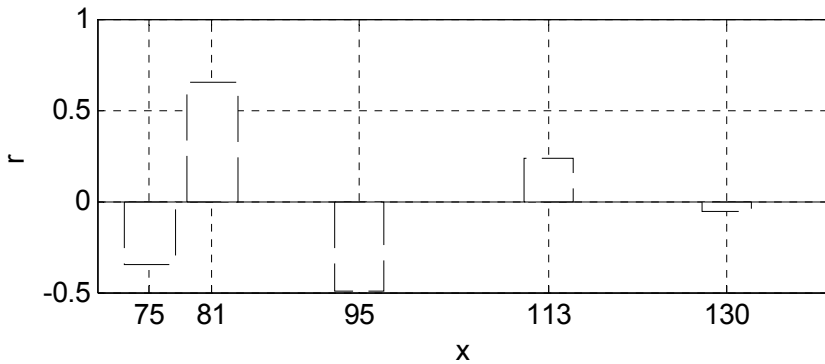
b)  $n_r=3$

**Figura 7.6.** Reprezentarea grafică de tip line a reziduurilor.

Reziduurile reprezintă o măsură de apreciere a erorilor aleatoare specifice metodei de aproximare prin regresie. Prin urmare, dacă reziduurile au o distribuție aleatoare, atunci funcția de regresie respectivă aproximează bine funcția tabelară. În cazul în care reziduurile par să aibă o structură definită în mod sistematic, atunci funcția de regresie nu aproximează corect funcția tabelară.



a)  $n_r=1$



b)  $n_r=3$

**Figura 7.7.** Reprezentarea grafică de tip bar a reziduurilor.

Pentru aprecierea sintetică a calității unei aproximări prin regresie se introduc următorii parametri numerici, [10]:

- norma reziduurilor (trebuie să aibă o valoare cât mai mică):

$$|r| = \sqrt{\sum_{i=1}^n (y_i - F_i)^2}$$

- suma pătratelor reziduurilor (trebuie să aibă o valoare cât mai mică):

$$SSE = \sum_{i=1}^n (y_i - F_i)^2$$

- suma pătratelor diferențelor funcției  $F$  față de media aritmetică:

$$SSR = \sum_{i=1}^n (F_i - \bar{y})^2$$

- suma pătratelor diferențelor funcției  $f$  față de media aritmetică:

$$SST = \sum_{i=1}^n (y_i - \bar{y})^2$$

- raportul dintre suma pătratelor diferențelor funcției  $F$  față de media aritmetică și suma pătratelor diferențelor funcției  $f$  față de media aritmetică (trebuie să aibă o valoare cât mai aproape de 1):

$$R^2 = \frac{SSR}{SST} = 1 - \frac{SSE}{SST}$$

în care  $\bar{y}$  reprezintă media aritmetică a valorilor funcției tabelare:

$$\bar{y} = \frac{1}{n} \sum_{i=1}^n y_i$$

#### Problema 7.4

Să se calculeze valorile parametrilor sintetici de apreciere a calității aproximării prin regresie pentru cazul celor două polinoame de regresie, de grad  $n_r=1$  și  $n_r=3$ , obținute la problema 7.3.

#### Rezolvare

Pentru rezolvarea problemei, fișierul `script` anterior se completează cu instrucțiunile:

```
ymed=mean(y)
normr1=norm(r1)
SSE1=sum((y-yr1),^2)
SSR1=sum((yr1-ymed),^2)
SST1=sum((y-ymed),^2)
RR1=SSR1/SST1
normr3=norm(r3)
SSE3=sum((y-yr3),^2)
SSR3=sum((yr3-ymed),^2)
SST3=sum((y-ymed),^2)
RR3=SSR3/SST3
```

obținându-se rezultatele:

```
ymed=102.4307
norm1=13.098
SSE1=171.5564
SSR1=2258.5
SST1=2430.1
RR1=0.9294
norm3=0.9197
SSE3=0.8458
SSR3=2429.2
SST3=2430.1
RR3=0.9997
```

Dacă se consideră, de exemplu, parametrul  $R^2$ , acesta are un domeniu de variație între 0 și 1, aproximația fiind cu atât mai bună cu cât valoarea obținută este mai aproape de limita superioară.

O altă modalitate de apreciere a calității aproximării prin regresie este analiza intervalelor de eroare definite prin, [11, 12]:

$$\delta = p \pm t\sqrt{S}$$

în care  $p$  reprezintă coeficienții obținuți prin metoda de regresie utilizată;  $t$  reprezintă inversa funcției  $T$  a lui Student, iar  $S$  reprezintă vectorul elementelor diagonale ale matricii covarianței coeficienților funcției de aproximare.

Dacă se consideră un nivel de încredere  $P=95\%$  (deci un risc al deciziei  $\alpha = 1 - P/100=0,05$ ), pentru reprezentarea grafică a intervalelor de eroare se utilizează instrucțiunile:

- Cazul polinomului de regresie de grad  $n_r=1$ , figura 7.8, a):

```
P=95;alpha=1-P/100;
[p1,S1]=polyfit(x,y,1);
[yr1,delta1]=polyconf(p1,x,S1,alpha);
plot(x,y,'ok');hold on;
plot(x,yr1,'-k');
plot(x,yr1+delta1,':k');
plot(x,yr1-delta1,':k');
grid;xlabel('x');ylabel('y');hold off
```

- Cazul polinomului de regresie de grad  $n_r=3$ , figura 7.8, b):

```
P=95;alpha=1-P/100;
[p3,S3]=polyfit(x,y,3);
[yr3,delta3]=polyconf(p3,x,S3,alpha);
plot(x,y,'ok');hold on;
plot(x,yr3,'--k');
plot(x,yr3+delta3,':k');
plot(x,yr3-delta3,':k');
grid;xlabel('x');ylabel('y');hold off
```

Dacă se consideră un nivel de încredere  $P=99\%$  (deci un risc al deciziei  $\alpha = 1 - P/100=0,01$ ), pentru reprezentarea grafică a intervalelor de eroare se utilizează instrucțiunile:

- Cazul polinomului de regresie de grad  $n_r=1$ , figura 7.8, c):

```
P=99;alpha=1-P/100;
[p1,S1]=polyfit(x,y,1);
[yr1,delta1]=polyconf(p1,x,S1,alpha);
plot(x,y,'ok');hold on;plot(x,yr1,'-k');
plot(x,yr1+delta1,':k');plot(x,yr1-delta1,':k');
grid;xlabel('x');ylabel('y');hold off
```

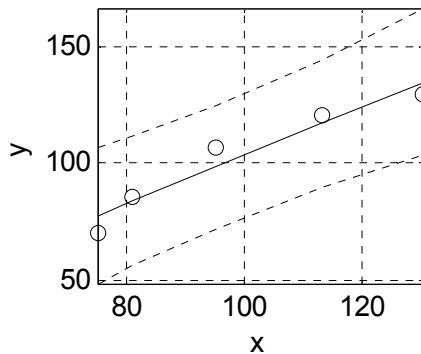


- Cazul polinomului de regresie de grad  $n_r=3$ , figura 7.8, d):

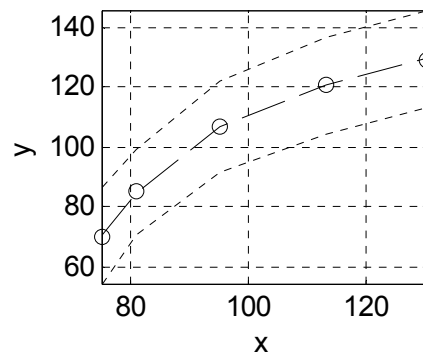
```
P=99;alpha=1-P/100;
[p3,S3]=polyfit(x,y,3);
[yr3,delta3]=polyconf(p3,x,S3,alpha);
plot(x,y,'ok');hold on;
plot(x,yr3,'--k');
plot(x,yr3+delta3,':k');
plot(x,yr3-delta3,':k');
grid;xlabel('x');ylabel('y');hold off
```

Din analiza reprezentării grafice a curbelor de eroare se poate interpreta calitatea procesului de aproximare. Astfel, dacă pe tot domeniul de variație al funcției se constată că variația intervalului dintre cele două curbe de eroare este nesemnificativă, atunci calitatea aproximării este bună.

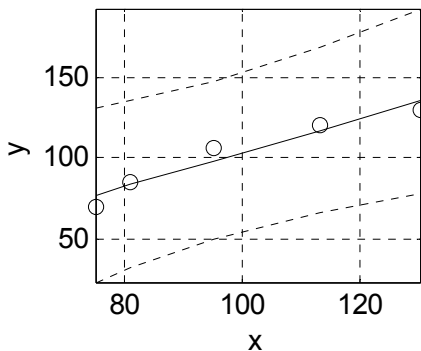
Dacă dimpotrivă, se constată variații importante ale intervalului dintre cele două curbe limită atunci calitatea aproximării este slabă. Variații importante ale intervalului de eroare apar, în general, în vecinătatea limitelor domeniului de definiție al funcției tabelare.



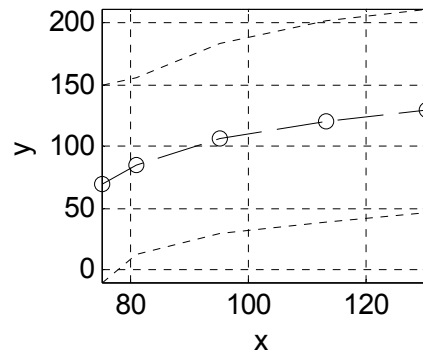
a)  $n_r=1$  și  $P=95\%$



b)  $n_r=3$  și  $P=95\%$



c)  $n_r=1$  și  $P=99\%$



d)  $n_r=3$  și  $P=99\%$

**Figura 7.8.** Reprezentarea intervalului de eroare.

### 7.3. INTERFEȚE SPECIALIZATE PENTRU ANALIZA NUMERICĂ A DATELOR EXPERIMENTALE

Pentru analiza numerică a datelor experimentale se pot utiliza instrucțiuni specifice, ca în exemplele anterioare, sau interfețele specializate de analiză existente în structura mediului de programare: interfața Basic Fitting [13], interfața polytool, [14] și interfața cftool, [15].

#### 7.3.1. Interfața Basic Fitting

Pentru deschiderea interfeței de analiză Basic Fitting trebuie parcurse următoarele etape:

- Definirea celor doi vectori  $x$  și  $y$  (reprezentarea tabelară a funcției de analizat).
- Reprezentarea grafică a funcției tabelare cu instrucțiunea `plot(x,y,'o')`.
- Din meniul Tools al figurii se selectează opțiunea Basic Fitting, rezultând deschiderea interfeței de analiză, figura 7.9.

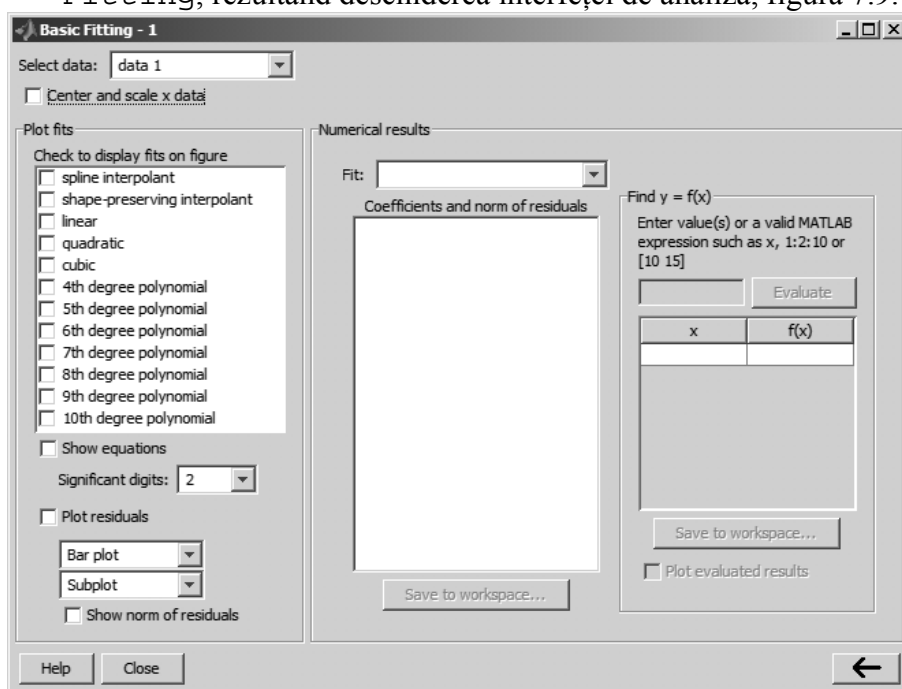


Figura 7.9. Interfața Basic Fitting.

Principalele operațiuni specifice care se pot efectua cu ajutorul interfeței Basic Fitting sunt, figura 7.9:

- Alegerea unor polinoame de regresie de diferite grade prin selectarea polinomului dorit din zona Plot fits.

- Afișarea ecuației polinomului de regresie prin selectarea opțiunii `Show equations` și specificarea numărului de zecimale ale coeficienților polinomului de regresie prin selectarea opțiunii dorite din `Significant digits`.
- Reprezentarea grafică a reziduurilor prin selectarea opțiunii `Plot residuals`, utilizând grafice de tip `line` sau de tip `bar`, în aceeași fereastră grafică cu graficul polinomului de aproximare, sau într-o fereastră grafică separată.
- Calcularea și afișarea valorii normei reziduurilor prin selectarea opțiunii `Show norm of residuals`.
- Calcularea și afișarea valorilor coeficienților polinomului de regresie în zona `Numerical results`.
- Evaluarea polinomului de regresie în diferite puncte din interiorul (interpolare) sau exteriorul (extrapolare) domeniului de definiție al funcției tabelare și reprezentarea grafică a valorilor astfel obținute pe graficul polinomului de aproximare (doar în cazul analizei efectuate în interiorul domeniului de definiție al funcției tabelare) în zona de evaluare `Find y=f(x)`.
- Salvarea principalilor parametri ai aproximării în mediul de lucru al programului pentru efectuarea unor analize ulterioare, prin selectarea opțiunii `Save to workspace`.

### Problema 7.5

Se consideră funcția tabelară definită în problema 7.2 prin valorile:

$x$	75	81	95	113	130
$y = f(x)$	69,8842	85,4635	106,5992	120,7775	129,4293

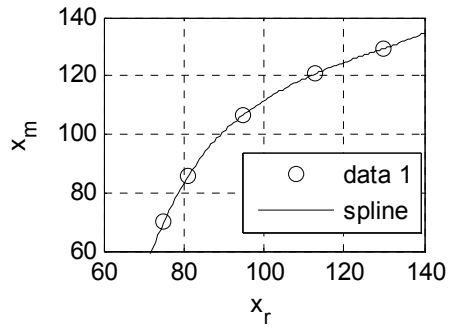
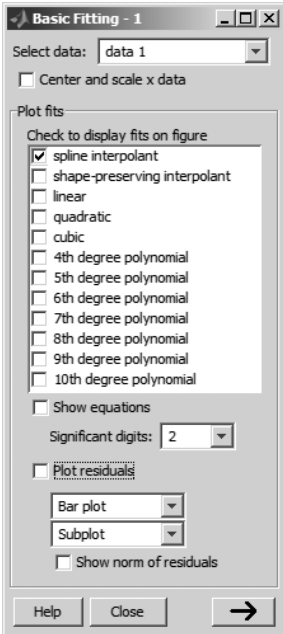
Utilizând interfața `Basic Fitting` să se analizeze următoarele tipuri de aproximări:

- Aproximare prin interpolare cu polinoame de tip spline.
- Aproximare prin regresie polinomială de grad  $n_r=1$ .
- Aproximare prin regresie polinomială de grad  $n_r=2$ .
- Aproximare prin regresie polinomială de grad  $n_r=3$ .

Să se stabilească care polinom de regresie conduce la cea mai bună aproximare pe baza valorilor numerice și a distribuției reziduurilor fiecărei aproximări.

### Rezolvare

În figura 7.10, a) se prezintă setările efectuate în interfața `Basic Fitting` pentru realizarea aproximării prin interpolare cu polinoame de tip spline, iar în figura 7.10, b) se observă reprezentarea grafică obținută.

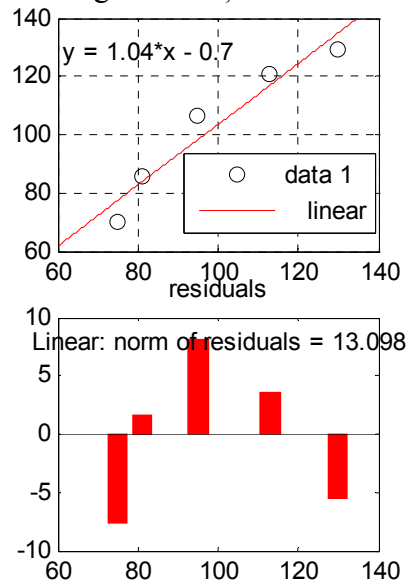
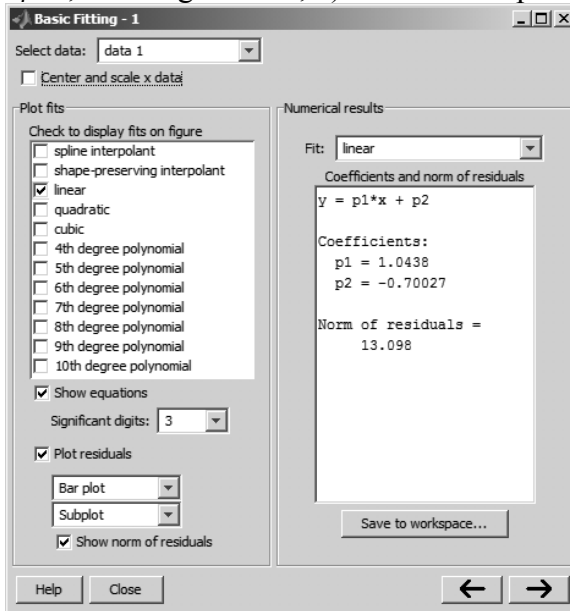


a) interfața Basic Fitting

b) interpolare cu polinoame spline

**Figura 7.10.** Interfața Basic Fitting: interpolare spline.

În figura 7.11, a) se prezintă setările efectuate în interfața Basic Fitting pentru realizarea aproximării prin regresie cu polinoame de grad  $n_r=1$ , iar în figura 7.11, b) se observă reprezentarea grafică obținută.

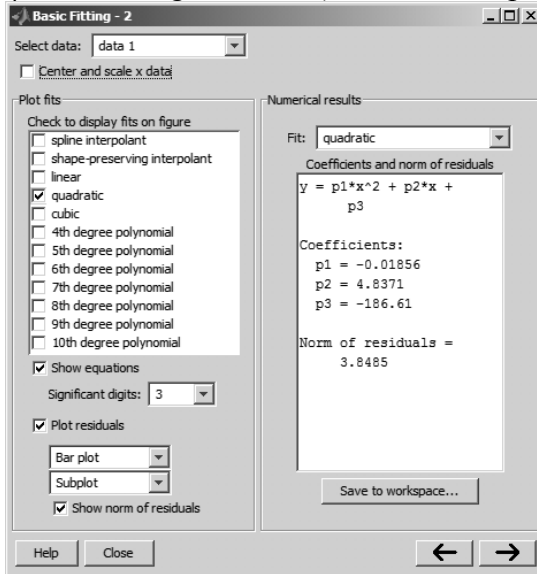


a) interfața Basic Fitting

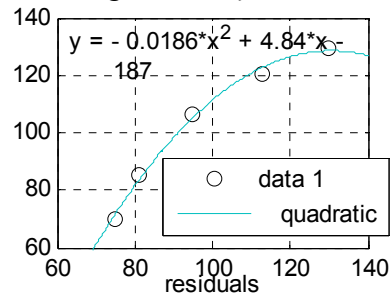
b) polinom de regresie de grad 1

**Figura 7.11.** Interfața Basic Fitting: regresie liniară.

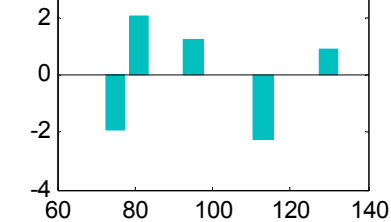
În figura 7.12, a) se prezintă setările efectuate în interfața Basic Fitting pentru realizarea aproximării prin regresie cu polinoame de grad  $n_p=2$ , iar în figura 7.12, b) se observă reprezentarea grafică obținută.



a) interfața Basic Fitting



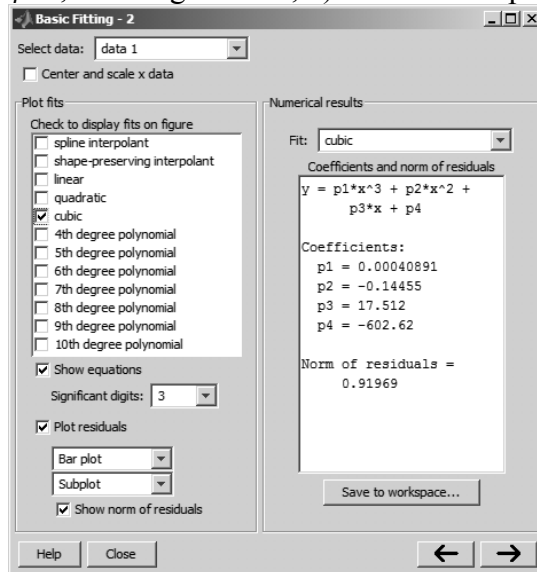
Quadratic: norm of residuals = 3.8485



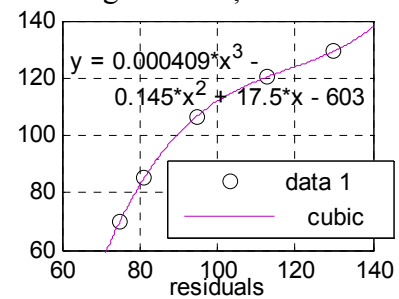
b) polinom de regresie de grad 2

**Figura 7.12.** Interfața Basic Fitting: regresie polinomială de grad 2,

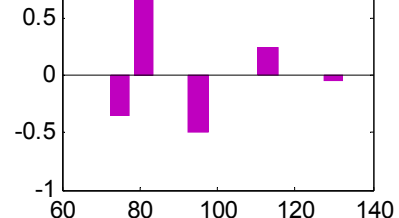
În figura 7.13, a) se prezintă setările efectuate în interfața Basic Fitting pentru realizarea aproximării prin regresie cu polinoame de grad  $n_p=3$ , iar în figura 7.13, b) se observă reprezentarea grafică obținută.



a) interfața Basic Fitting



Cubic: norm of residuals = 0.91969



b) polinom de regresie de grad 3

**Figura 7.13.** Interfața Basic Fitting: regresie polinomială de grad 3.

Alegerea celei mai bune aproximări se realizează în funcție de valoarea și distribuția reziduurilor, Astfel, o aproximare este cu atât mai bună cu cât distribuția reziduurilor are un caracter mai puternic aleatoriu (cât mai multe schimbări de semn) și cu cât reziduurile au valori mai mici (norma reziduurilor are o valoare mai mică). În acest caz, atât din punct de vedere al distribuției reziduurilor, cât și al valorii normei reziduurilor (13,098 pentru  $n_r=1$ ; 3,8485 pentru  $n_r=2$ ; 0,91969 pentru  $n_r=3$ ), cea mai bună aproximare este cea cu polinoame de grad 3.

### 7.3.2. Interfața polytool

Pentru deschiderea interfeței de analiză `polytool` trebuie parcurse următoarele etape:

- Definirea celor doi vectori  $x$  și  $y$ .
- Lansarea în execuție a interfeței folosind instrucțiunea:

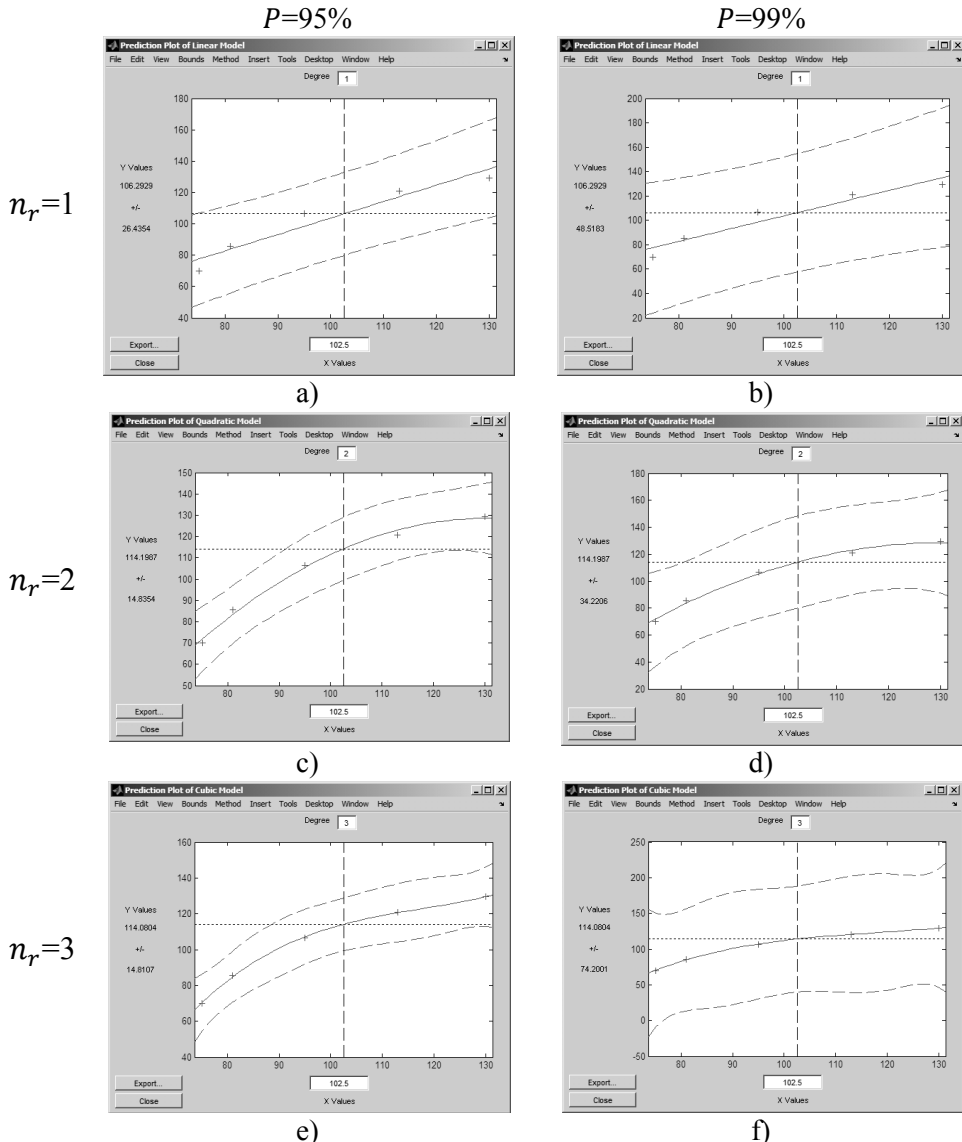
```
polytool(x,y,nr,alpha)
```

în care  $y$  reprezintă valorile funcției  $f$  în puncte de rețea  $x$ ;  $nr$  reprezintă gradul polinomului de regresie utilizat pentru aproximare, iar  $alpha$  reprezintă riscul decizie ( $\alpha = 1 - P/100$ , în care  $P$  este nivelul de încredere ales).

Principalele operațiuni specifice care se pot efectua cu ajutorul interfeței `polytool` sunt:

- Alegerea unor polinoame de regresie de diferite grade, prin specificarea gradului polinomului într-o casetă de control.
- Vizualizarea grafică a intervalelor de eroare (distanța dintre cele două curbe limită) pentru  $P = (1 - \alpha)100$  %. Nivelul de încredere  $P$  se specifică odată cu scrierea instrucțiunii `polytool`.
- Evaluarea polinomului de regresie în câte un punct de interes din interiorul domeniului de definiție al funcției tabelare și reprezentarea grafică a valorii astfel obținute pe graficul polinomului de aproximare sub forma unui cursor format din intersecția liniei verticale a abscisei punctului de interes cu orizontala ordonatei corespunzătoare. Acest cursor poate fi deplasat cu mouse-ul pe tot domeniul de variație al funcției tabelare, permițând astfel parcurgerea interactivă a valorilor polinomului de regresie. Interogarea valorilor curbei de regresie se poate face doar în interiorul domeniului de definiție al funcției tabelare.
- Valoarea polinomului de regresie obținută pentru orice punct de interes aflat în interiorul domeniului de variație al funcției tabelare este însoțită și de limitele intervalului de eroare corespunzătoare punctului respectiv.

În figura 7.14 se prezintă rezultatele obținute prin utilizarea interfeței polytool pentru regresia polinomială de grad 1, 2 și 3. Rezultatele sunt prezentate comparativ, pentru două valori ale nivelului de încredere:  $P=95\%$  și  $P=99\%$ . Se observă creșterea intervalelor de eroare odată cu creșterea nivelului de încredere. Alegerea celei mai bune aproximări se realizează pe baza analizei formei intervalelor de eroare. Se urmărește ca intervalele de eroare să aibă o variație cu cât mai puține discontinuități (mărima intervalelor de eroare să fie aproximativ egală pe tot domeniul de variație al funcției). De asemenea, o aproximare este cu atât mai bună cu cât la capetele intervalului, tendința de divergență a curbelor limită este mai mică.



**Figura 7.14.** Interfața polytool: regresie polinomială de grad 1, 2 și 3.

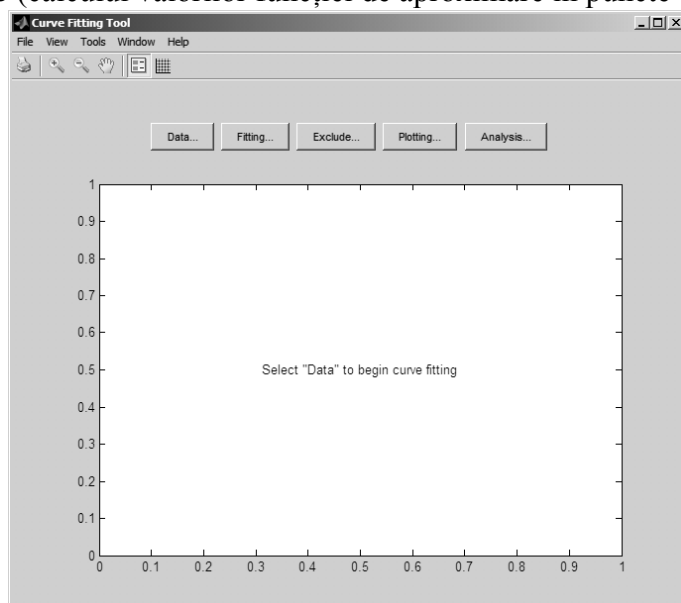
### 7.3.3. Interfața `cftool`

Pentru deschiderea interfeței de analiză `cftool` trebuie parcurse următoarele etape:

- Definirea celor doi vectori  $x$  și  $y$ .
- Lansarea în execuție a interfeței cu instrucțiunea:

```
cftool
```

În figura 7.15 se prezintă interfața `cftool` înainte de introducerea datelor experimentale de analizat. Se observă prezența a cinci butoane de comenzi: `Data` (introducerea datelor de analizat), `Fitting` (efectuarea aproximării), `Exclude` (definirea unor reguli pentru excluderea anumitor subdomenii ale variabilelor  $x$  și  $y$ ), `Plotting` (reprezentarea grafică) și `Analysis` (calculul valorilor funcției de aproximare în puncte de interes).

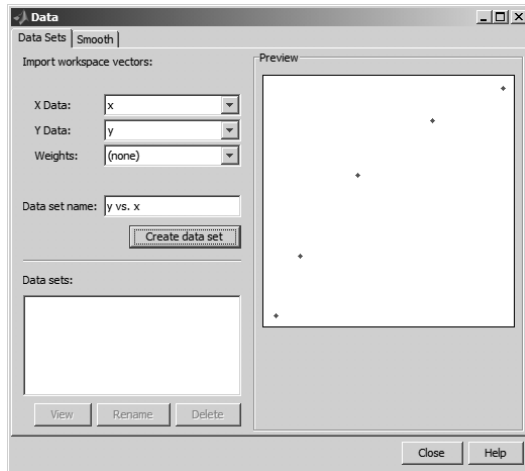


**Figura 7.15.** Interfața `cftool` înainte de introducerea datelor,

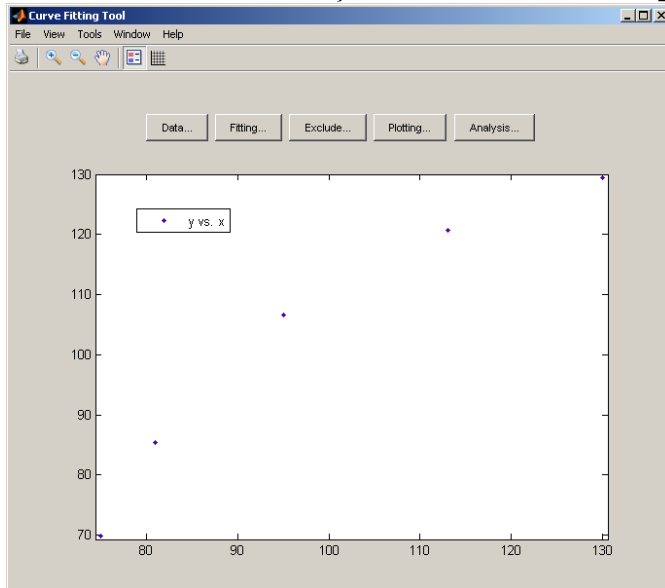
Principalele operațiuni specifice care se pot efectua cu ajutorul interfeței `cftool` sunt:

- Încărcarea în interfața de analiză a vectorilor  $x$  și  $y$  și definirea astfel a unui set de date prin selectarea comenzii `Data`, figura 7.16. Cei doi vectori  $x$  și  $y$  trebuie definiți în prealabil în spațiul de lucru al programului. După selectarea celor doi vectori de date  $x$  și  $y$ , se selectează comanda `Create data set`. După definirea setului de date, în fereastra grafică a interfeței se realizează, în mod automat, graficul funcției tabelare. În figura 7.17 se prezintă interfața `cftool` după introducerea datelor experimentale de analizat.





**Figura 7.16.** Introducerea datelor și crearea setului de date  $y$  vs  $x$ .

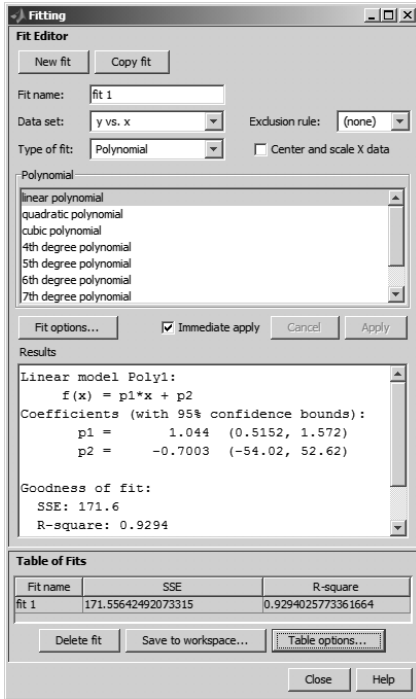


**Figura 7.17.** Interfața cftool după introducerea datelor.

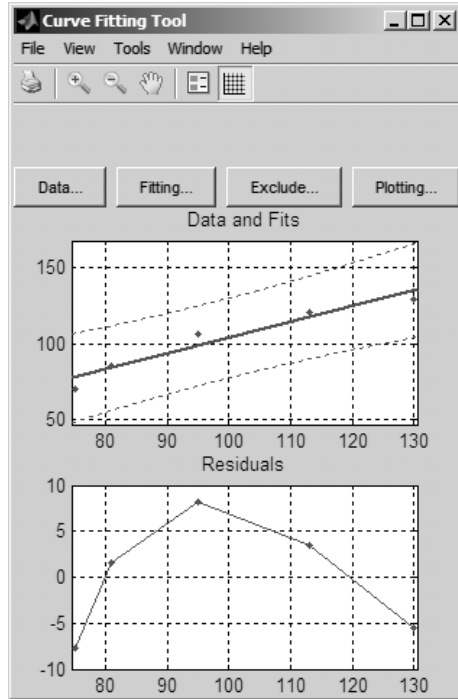
- După definirea setului de date de analizat se selectează comanda **Fitting** care asigură accesul la principalii parametri de configurare ai aproximării: definirea unei noi aproximări (**New fit**); specificarea numelui aproximării (**Fit name**); selectarea setului de date de aproximat (**Data set**); selectarea tipului aproximării (**Type of fit**); specificarea parametrilor sintetici de apreciere a calității aproximării (**Table options**); salvarea caracteristicilor aproximării în spațiul de lucru al programului pentru prelucrări ulterioare (**Save to workspace**); ștergerea unei aproximări (**Delete fit**).

- Spre deosebire de celelalte interfețe, pentru care este posibilă doar aproximarea prin interpolare polinomială, în cazul interfeței `cftool` sunt implementate și alte tipuri de aproximări: de tip Gauss, de tip funcție putere, de tip funcție exponențială, de tip Fourier, de tip sumă de funcții sinus, de tip Weibull, de tip funcție rațională și evident de tip polinomial.
- Pentru cazul funcției de regresie de tip polinomial se poate alege gradul polinomului de regresie. Se afișează ecuația polinomului de regresie, valorile coeficienților polinomului, precum și parametrii numerici sintetici ai calității aproximării.
- Reprezentarea grafică a reziduurilor, utilizând grafice de tip `line` sau de tip `scatter plot`, în aceeași fereastră grafică cu graficul polinomului de aproximare. Pentru aceasta se selectează comanda `Residuals` din meniul `View`.
- Alegerea nivelului de încredere dorit din meniul `View`, comanda `Confidence level`.
- Reprezentarea grafică a intervalelor de eroare conform nivelului de încredere ales prin selectarea opțiunii `Prediction bounds` din meniul `View`.
- După selectarea diferitelor opțiuni, pentru efectuarea calculelor se selectează comanda `Apply`. Efectuarea automată a calculelor se face prin selectarea opțiunii `Immediate apply`.
- Evaluarea polinomului de regresie în diferite puncte din interiorul sau exteriorul domeniului de definiție al funcției tabelare și reprezentarea grafică a valorilor astfel obținute pe graficul polinomului de aproximare. Pentru aceasta se selectează comanda `Analysis`. În fereastra de configurare se introduc punctele în care se dorește evaluarea funcției de aproximare, se selectează opțiunile pentru reprezentarea grafică a rezultatelor, se observă valorile numerice obținute.
- Modificarea limitelor valorilor de pe abscisă și de pe ordonată prin selectarea comenzii `Axis limit control` din meniul `Tools`. Creșterea limitelor celor două axe permite o mai bună analiză a modului de variație a intervalelor de eroare spre capetele domeniului, precum și analiza cazurilor de evaluare a diferitelor puncte din afara domeniului de definiție al funcției.

În figurile 7.18, a), 7.19, a) și 7.20, a) se prezintă setările efectuate în interfața `cftool:Fitting` pentru realizarea aproximării prin regresie cu polinoame de grad  $n_r=1, 2$  și  $3$ . În figurile 7.18, b), 7.19, b) și 7.20, b) se observă reprezentările grafice corespunzătoare celor trei tipuri de aproximare prin regresie: cu polinoame de grad  $n_r=1, 2$  și  $3$ .

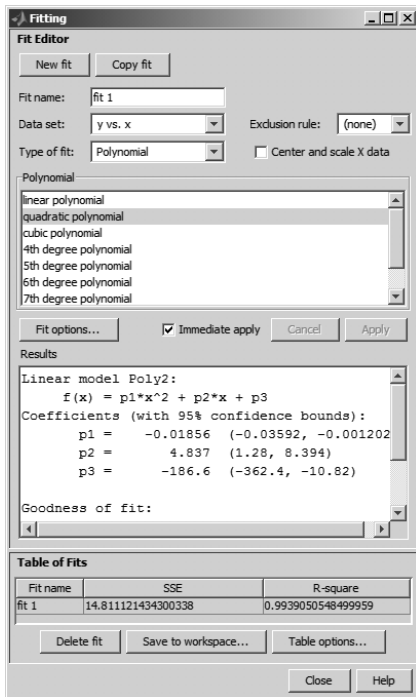


a) interfața cftool:Fitting

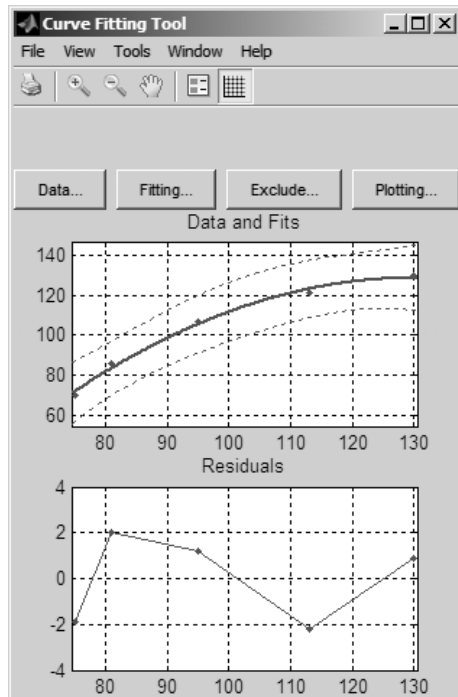


b) polinom de regresie de grad 1

**Figura 7.18.** Interfața cftool: regresie liniară.

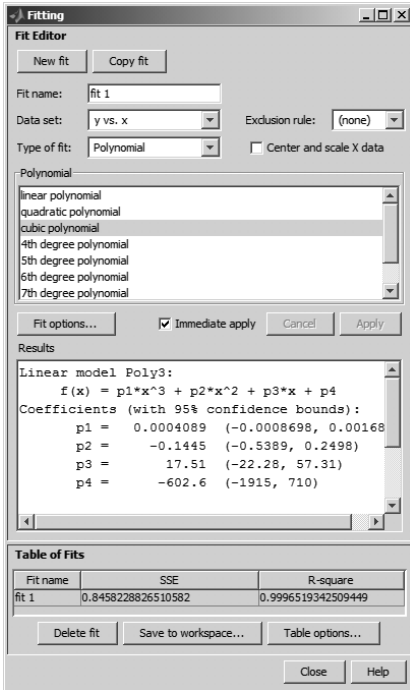


a) interfața cftool:Fitting

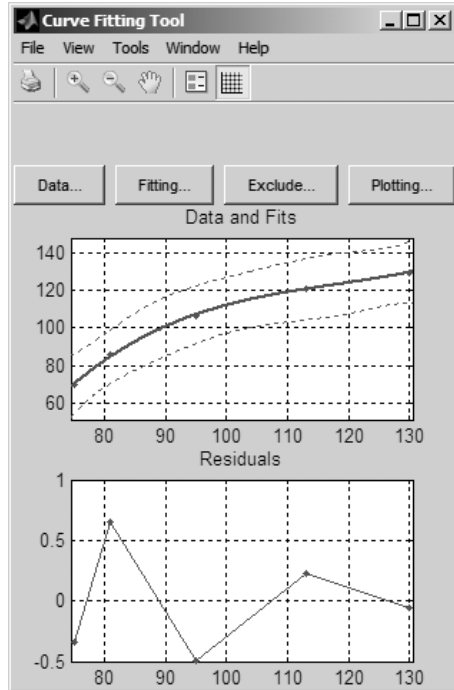


b) polinom de regresie de grad 2

**Figura 7.19.** Interfața cftool: regresie polinomială de grad 2.



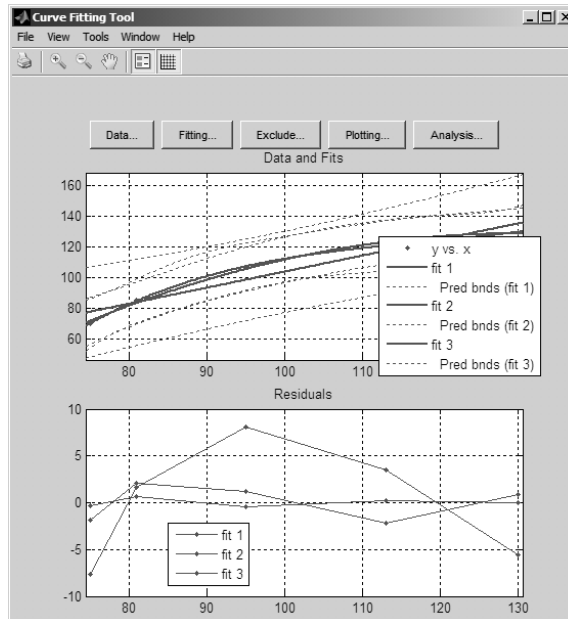
a) interfața cftool :Fitting



b) polinom de regresie de grad 3

**Figura 7.20.** Interfața cftool: regresie polinomială de grad 3.

În figura 7.21 se prezintă analiza grafică comparativă a celor trei polinoame de aproximare: curbele de aproximare, intervalele de eroare și variația reziduurilor.



**Figura 7.21.** Analiza grafică comparativă a polinoamelor de aproximare.

Se analizează atât forma curbelor de aproximare, a curbelor limită cât și distribuția și valorile reziduurilor, Din analiza grafică a celor trei aproximări, rezultă că cea mai bună aproximare este aproximarea fit 3 (polinom de regrese de grad 3).

Alegerea celei mai bune aproximări se realizează și pe baza unor parametri numerici specifici: norma reziduurilor ( $|r|$ , trebuie să aibă o valoare cât mai mică), suma pătratelor reziduurilor (SSE, trebuie să aibă o valoare cât mai mică), raportul dintre suma pătratelor funcției de aproximare  $F$  față de media aritmetică și suma pătratelor funcției tabelare  $f$  față de media aritmetică ( $R^2$ , trebuie să aibă o valoare cât mai apropiată de 1), etc.

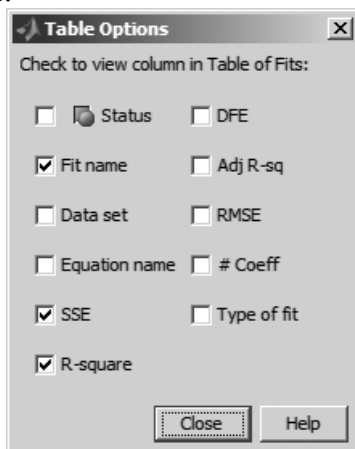
Pentru cazul analizat, valorile parametrilor sintetici de apreciere a calității aproximării sunt prezentați în tabelul 7.4.

**Tabel 7.4.** Parametrii sintetici de apreciere a calității aproximării,

Aproximare polinomială		SSE	$R^2$
fit 1	Grad 1	171,556	0,929402
fit 2	Grad 2	14,811	0,993905
fit 3	Grad 3	0,8458	0,999651

Și din analiza parametrilor numerici rezultă că cea mai bună aproximare este tot aproximarea fit 3 (polinom de regrese de grad 3) pentru că are atât valoarea SSE cea mai mică (0,8458), cât și valoarea  $R^2$  cea mai apropiată de 1 (0,999651).

Parametrii sintetici de apreciere a calității aproximării sunt calculați automat și afișați în zona `Table of fits` (pentru acest caz doar SSE și  $R^2$ ). Adăugarea și a altor informații în această zonă de prezentare a rezultatelor numerice se realizează prin selectarea comenzii `Table options`. Se selectează apoi parametrii doriți din fereastra de configurare, prezentată în figura 7.22.



**Figura 7.22.** Parametrii sintetici de apreciere a calității aproximării.

## BIBLIOGRAFIE

1. Curteanu S., Inițiere în MATLAB, Ed. Polirom, Iași, 2008.
2. Davidescu A., Analiza și procesarea datelor în MATLAB, Ed. Politehnica, Timișoara, 2003.
3. Ghinea M., Firețeanu V., MATLAB. Calcul numeric. Grafică. Aplicații, Ed. Teora, București, 2003.
4. Lăzăroi Gh., Sisteme de programare pentru modelare și simulare, Ed. Politehnica Press, București, 2005.
5. MathWorks, Save Workspace Variables to File, <http://www.mathworks.com/help/matlab/ref/save.html>, accesat la 9.02.2014.
6. MathWorks, Load Variables From File Into Workspace, <http://www.mathworks.com/help/matlab/ref/load.html>, accesat la 9.02.2014.
7. MathWorks, 1-D Data Interpolation (table lookup), <http://www.mathworks.com/help/matlab/ref/interp1.html>, accesat la 9.02.2014.
8. MathWorks, Polynomial Curve Fitting, <http://www.mathworks.com/help/matlab/ref/polyfit.html>, accesat la 9.02.2014.
9. MathWorks, Residuals, <http://www.mathworks.com/help/stats/residuals.html>, accesat la 7.09.2014.
10. MathWorks, Residual Analysis, <http://www.mathworks.com/help/curvefit/residual-analysis.html>, accesat la 7.09.2014.
11. MathWorks, Confidence and Prediction Bounds, <http://www.mathworks.com/help/curvefit/confidence-and-prediction-bounds.html>, accesat la 7.09.2014.
12. MathWorks, Polynomial Confidence Intervals, <http://www.mathworks.com/help/stats/polyconf.html>, accesat la 9.02.2014.
13. MathWorks, Interactive Fitting, [http://www.mathworks.com/help/matlab/data\\_analysis/interactive-fitting.html](http://www.mathworks.com/help/matlab/data_analysis/interactive-fitting.html), accesat la 9.02.2014.
14. MathWorks, Interactive Polynomial Fitting, <http://www.mathworks.com/help/stats/polytool.html>, accesat la 9.02.2014.
15. MathWorks, Open Curve Fitting app, <http://www.mathworks.com/help/curvefit/cftool.html>, accesat la 9.02.2014.
16. Palm W.J., Introduction to MATLAB 7 for Engineers, McGraw Hill, New York, 2005.
17. Zahariea D., Măsurări hidraulice, Rotaprint, Universitatea Tehnică „Gheorghe Asachi”, Iași, 1999.
18. Zahariea D., Măsurări hidraulice. Îndrumar de laborator, Rotaprint, Universitatea Tehnică „Gheorghe Asachi”, Iași, 2003.

## CAPITOLUL 8

### VARIABLE MATRICEALE

#### 8.1. DEFINIRE ȘI OPERAȚII SPECIFICE

##### 8.1.1. Definirea variabilelor matriceale oarecare

În mod implicit, limbajul de programare MATLAB operează doar cu variabile de tip matrice, astfel că la declararea unei variabile  $X$ , programul construiește o matrice având dimensiunea  $n_L \times n_C$ , cu  $n_L$  linii și  $n_C$  coloane.

Definirea variabilelor matriceale se realizează prin operația de atribuire (folosind operatorul de atribuire, =), prin care unei variabile matriceale oarecare, programul îi va atribui o anumită expresie:

NumeVariabilă=expresie

astfel încât matricea obținută va conține elementele:

$$X = \begin{pmatrix} x_{11} & \cdots & x_{1n_C} \\ \vdots & \ddots & \vdots \\ x_{n_L1} & \cdots & x_{n_Ln_C} \end{pmatrix}$$

în care indicii primului element sunt (1,1).

Definirea matricelor oarecare se face cu ajutorul operatorului de construcție a matricelor [ ], prin specificarea elementelor, linie după linie. La sfârșitul fiecărei linii, se inserează ca separator de linie caracterul ;. În cazul în care elementele unei linii reprezintă un vector cu elemente neasociate atunci trebuie introduse toate elementele, unul după altul.

Spre exemplu, matricea:

$$A = \begin{pmatrix} 2 & -3.4 & 11 \\ 0,5 & 3/4 & 7 \\ 12 & 4,02 & 1 \end{pmatrix}$$

are toate liniile formate din elemente neasociate, așa încât pentru definirea matricei se va utiliza metoda introducerii element-cu-element, conform instrucțiunii:

A=[2 -3.4 11;0.5 -3/4 7;12 4.02 1]

Dacă însă elementele unei linii reprezintă un vector cu elemente asociate, total sau pe porțiuni, atunci se vor utiliza regulile pentru definirea vectorilor cu elemente asociate (xmin:pas:xmax, sau linspace (xmin, xmax, nx)). Spre exemplu, dacă se consideră matricea:

$$B = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 8 & 7 & 6 & 5 & 4 & 3 & 2 \\ 2 & 4 & 6 & 8 & 1 & 2 & 3 \end{pmatrix}$$

se observă că fiecare linie este un vector cu elemente asociate, astfel încât pentru definirea matricei se poate utiliza instrucțiunea simplificată:

$$A = [1:7; 8:-1:2; 2:2:8 \quad 1:3]$$

### 8.1.2. Definirea variabilelor matriceale particulare

Există o serie de matrice particulare pentru definirea cărora se utilizează următoarele instrucțiuni specifice:

- **Matricea goală**

Definirea unei matrice goale  $A$ , are sens doar la începutul unui program în vederea alocării unui spațiu de memorie pentru variabila matriceală respectivă. Astfel, instrucțiunea:

$$A = []$$

atribuie variabilei  $A$  o matrice având dimensiunea  $0 \times 0$ .

- **Matricea zero**

Matricea zero este matricea având toate elementele egale cu zero. Pentru definirea matricei zero se utilizează instrucțiunile, [2]:

$$\begin{aligned} O &= \text{zeros}(n) \\ O &= \text{zeros}(n_L, n_C) \\ O &= \text{zeros}(\text{size}(A)) \end{aligned}$$

În cazul în care instrucțiunea `zeros` se apelează cu un singur argument scalar ( $n$ ), atunci rezultatul executării instrucțiunii este o matrice pătrată având toate elementele zero și dimensiunea  $n \times n$ . Spre exemplu, instrucțiunea:

$$O = \text{zeros}(3)$$

va genera următorul rezultat:

$$O = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

În cazul în care instrucțiunea `zeros` se apelează cu două argumente scalare ( $n_L$  și  $n_C$ ), atunci rezultatul executării instrucțiunii este o matrice având toate elementele zero și dimensiunea  $n_L \times n_C$ .

Spre exemplu, instrucțiunea:

$$O = \text{zeros}(3, 4)$$

va genera următorul rezultat:



$$O = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Pentru cel de-al treilea caz de apelare a comenzii, rezultatul executării instrucțiunii va fi o matrice având toate elementele zero și dimensiunea identică cu cea a matricei  $A$ . Spre exemplu, se consideră o matrice  $A$  având dimensiunea  $3 \times 2$ , de forma:

$$A = \begin{pmatrix} 1 & 2 \\ 12 & -5 \\ 2 & 33 \end{pmatrix}$$

Instrucțiunea:

```
O=zeros(size(A))
```

va genera matricea cu toate elementele egale cu zero, având aceeași dimensiune cu cea a matricei  $A$ , adică  $(3 \times 2)$ :

$$O = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}$$

- **Matricea unitate**

Matricea unitate este matricea având toate elementele egale cu 1. Pentru definirea matricei unitate se utilizează instrucțiunile, [3]:

```
U=ones(n)
```

```
U=ones(nL, nC)
```

```
U=ones(size(A))
```

În cazul în care instrucțiunea `ones` se apelează cu un singur argument scalar ( $n$ ), atunci rezultatul executării instrucțiunii este o matrice pătrată având toate elementele egale cu unu și dimensiunea  $n \times n$ . Spre exemplu, instrucțiunea:

```
U=ones(3)
```

va genera următorul rezultat:

$$U = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

În cazul în care instrucțiunea `ones` se apelează cu două argumente scalare ( $n_L$  și  $n_C$ ), atunci rezultatul executării instrucțiunii este o matrice având toate elementele egale cu unu și dimensiunea  $n_L \times n_C$ . Spre exemplu, instrucțiunea:

```
U=ones(3,4)
```

va genera următorul rezultat:

$$U = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

Pentru cel de-al treilea caz de apelare a comenzii, rezultatul executării instrucțiunii va fi o matrice având toate elementele egale cu unu și dimensiunea identică cu cea a matricei  $A$ . Spre exemplu, se consideră o matrice  $A$  având dimensiunea  $4 \times 3$ , de forma:

$$A = \begin{pmatrix} 1 & 21 & 2 \\ 12 & 4 & 3 \\ -1 & 0 & 6 \\ 3 & 7 & 11 \end{pmatrix}$$

Instrucțiunea:

```
U=ones(size(A))
```

va genera matricea cu toate elementele egale cu unu, având aceeași dimensiune cu cea a matricei  $A$ , adică  $(4 \times 3)$ :

$$U = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

- **Matricea identitate**

Matricea identitate este matricea având toate elementele de pe diagonala principală egale cu unu, restul elementelor fiind egale cu zero. Pentru definirea matricei identitate se utilizează una din instrucțiunile, [4]:

```
I=eye(n)
I=eye(nL, nC)
I=eye(size(A))
```

În cazul în care instrucțiunea `eye` se apelează cu un singur argument scalar ( $n$ ), atunci rezultatul executării instrucțiunii este o matrice pătrată având toate elementele de pe diagonala principală egale cu unu, restul elementelor zero și dimensiunea  $n \times n$ . Spre exemplu, instrucțiunea:

```
I=eye(3)
```

va genera următorul rezultat:

$$I = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

În cazul în care instrucțiunea `ones` se apelează cu două argumente scalare ( $n_L$  și  $n_C$ ), atunci rezultatul executării instrucțiunii este o matrice având toate elementele de pe diagonala principală egale cu

unu, restul elementelor zero și dimensiunea  $n_L \times n_C$ . Spre exemplu, instrucțiunea:

`I=eye(3,4)`

va genera următorul rezultat:

$$I = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Pentru cel de-al treilea caz de apelare a comenzii, rezultatul instrucțiunii va fi o matrice identitate și dimensiunea identică cu cea a matricei  $A$ .

Spre exemplu, se consideră o matrice  $A$  având dimensiunea  $4 \times 3$ , de forma:

$$A = \begin{pmatrix} 1 & 21 & 2 \\ 12 & 4 & 3 \\ -1 & 0 & 6 \\ 3 & 7 & 11 \end{pmatrix}$$

Instrucțiunea:

`I=eye(size(A))`

va genera matricea identitate, având aceeași dimensiune cu cea a matricei  $A$ , adică  $(4 \times 3)$ :

$$I = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$$

- **Matricea diagonală**

Matricea diagonală este matricea pătrată având specificate elementele unei anumite diagonale, restul elementelor fiind egale cu zero. Identificarea exactă a diagonalei cu elemente nenule se realizează prin intermediul parametrului  $k$ . Astfel, pentru  $k = 0$ , se obține diagonala principală, pentru  $k > 0$  se obține diagonala  $k$  deasupra diagonalei principale, iar pentru  $k < 0$  se obține diagonala  $k$  sub diagonala principală.

Pentru definirea matricei diagonale pentru care elementele de pe diagonala  $k$  trebuie să coincidă cu elementele unui vector oarecare  $v$ , se utilizează una din instrucțiunile, [5]:

`D=diag(v,k)`

`D=diag(v)`

Dimensiunea matricei diagonale obținută va fi  $n_v + |k|$ , în care  $n_v$  reprezintă numărul elementelor vectorului  $v$ .

Spre exemplu, se consideră un vector oarecare de forma:

$$v = \{v_1; v_2; v_3; v_4\}$$

Instrucțiunea:

$$D = \text{diag}(v, 2)$$

va genera matricea diagonală, având elementele vectorului  $v$  distribuite pe diagonala a doua ( $k = 2$ ), deasupra celei principale:

$$D = \begin{pmatrix} 0 & 0 & v_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & v_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & v_3 & 0 \\ 0 & 0 & 0 & 0 & 0 & v_4 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Instrucțiunea:

$$D = \text{diag}(v, -3)$$

va genera matricea diagonală, având elementele vectorului  $v$  distribuite pe diagonala a treia ( $k = -3$ ), sub diagonala principală:

$$D = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ v_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & v_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & v_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & v_4 & 0 & 0 & 0 \end{pmatrix}$$

În cazul în care instrucțiunea `diag` se apelează cu un singur parametru, atunci elementele vectorului vor fi distribuite în mod automat pe diagonala principală a matricei, presupunându-se că valoarea parametrului  $k$  este zero.

Instrucțiunea:

$$D = \text{diag}(v)$$

va genera matricea diagonală, având elementele vectorului  $v$  distribuite pe diagonala principală ( $k = 0$ ):

$$D = \begin{pmatrix} v_1 & 0 & 0 & 0 \\ 0 & v_2 & 0 & 0 \\ 0 & 0 & v_3 & 0 \\ 0 & 0 & 0 & v_4 \end{pmatrix}$$

Pentru definirea matricei diagonale pentru care elementele de pe diagonala  $k_d$  coincid cu elementele aflate pe diagonala  $k_v$  a unei matrice oarecare  $V$ , se utilizează instrucțiunea:

$$D = \text{diag}(\text{diag}(V, k_v), k_d)$$

Spre exemplu, se consideră o matrice oarecare de forma:

$$V = \begin{pmatrix} v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \\ v_{31} & v_{32} & v_{33} \end{pmatrix}$$

Instrucțiunea:

$$D = \text{diag}(\text{diag}(V, 1), -2)$$

va genera matricea diagonală, având elementele de pe prima diagonală deasupra celei principale ( $k_v = 1$ ) a matricei  $V$  distribuite pe diagonală a doua ( $k_d = -2$ ) sub cea principală a noii matrice  $D$ :

$$D = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ v_{12} & 0 & 0 & 0 \\ 0 & v_{23} & 0 & 0 \end{pmatrix}$$

Instrucțiunea:

$$D = \text{diag}(\text{diag}(V))$$

va genera matricea diagonală, având elementele de pe diagonală principală ( $k_v = 0$ ) a matricei  $V$  distribuite pe diagonală principală ( $k_d = 0$ ) a noii matrice  $D$ :

$$D = \begin{pmatrix} v_{11} & 0 & 0 \\ 0 & v_{22} & 0 \\ 0 & 0 & v_{33} \end{pmatrix}$$

- **Matricea aleatoare**

Distribuția elementelor aleatoare se poate realiza fie după repartiția standard uniformă, fie după repartiția standard normală.

Pentru cazul distribuției uniforme a numerelor aleatoare în intervalul standard (0; 1) se utilizează una din instrucțiunile, [6]:

```
Ru=rand(n)
Ru=rand(nL, nC)
Ru=rand(size(A))
```

În cazul unui interval oarecare ( $r_{min}; r_{max}$ )  $\neq(0; 1)$  se vor utiliza instrucțiunile:

```
Ru=rmin+(rmax-rmin)*rand(n)
Ru=rmin+(rmax-rmin)*rand(nL, nC)
Ru=rmin+(rmax-rmin)*rand(size(A))
```

Pentru cazul distribuției standard normale a numerelor corespunzătoare mediei aritmetice zero și abaterii medii pătratice unitare se utilizează una din instrucțiunile, [7]:

```
Rn=randn (n)
Rn=randn (nL, nC)
Rn=randn (size (A) )
```

În cazul unei distribuții normale oarecare caracterizată prin media aritmetică  $r_m$  și abaterea medie pătratică  $s$  se vor utiliza instrucțiunile:

```
Rn=rm+s*rand (n)
Rn=rm+s*rand (nL, nC)
Rn=rm+s*rand (size (A) )
```

În cazul în care instrucțiunile `rand` și `randn` se apelează cu un singur argument scalar ( $n$ ), atunci rezultatul executării instrucțiunilor respective este o matrice pătrată având elemente aleatoare, uniform sau normal distribuite și dimensiunea  $n \times n$ . În cazul în care instrucțiunile `rand` și `randn` se apelează cu două argumente scalare ( $n_L$  și  $n_C$ ), atunci rezultatul executării instrucțiunilor respective este o matrice având elemente aleatoare, uniform sau normal distribuite și dimensiunea  $n_L \times n_C$ . Pentru cel de-al treilea caz de apelare a comenzilor `rand` și `randn`, rezultatul executării instrucțiunilor va fi o matrice având elemente aleatoare, uniform sau normal distribuite și dimensiunea identică cu cea a matricei  $A$ .

- **Matricea pătrată „pătratul magic”**

Matricea denumită „pătratul magic” este o matrice pătrată particulară care se bucură de proprietatea că are aceeași sumă a elementelor pe fiecare linie, pe fiecare coloană, pe diagonala principală și pe anti-diagonala principală. De exemplu, pentru generarea „pătratului magic” cu dimensiunea  $n=3$  se utilizează instrucțiunea:

```
M=magic (3)
```

obținându-se rezultatul:

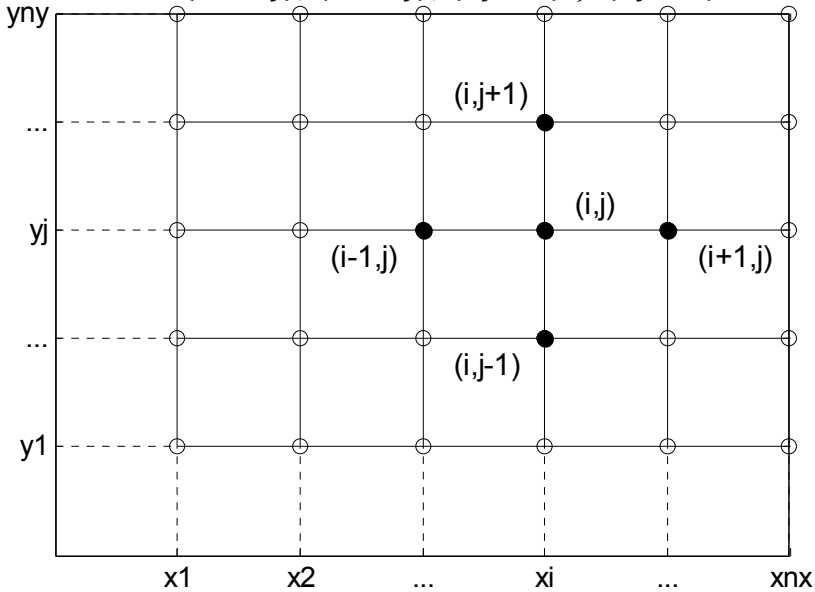
```
M =
     8     1     6
     3     5     7
     4     9     2
```

- **Matricele coordonatelor unui set de puncte din plan**

Se consideră vectorii  $x = \{x_1, x_2, \dots, x_{n_x}\}$  și  $y = \{y_1, y_2, \dots, y_{n_y}\}$ .

Dacă valorile celor doi vectori reprezintă puncte de discretizare ale celor două axe  $Ox$  și  $Oy$  ale planului cartezian  $xOy$ , atunci perechile  $(x_i, y_j)$ , cu  $i = 1 \div n_x$  și  $j = 1 \div n_y$  reprezintă coordonatele celor  $n_x \times n_y$

puncte  $P_{i,j}$  în care s-a efectuat discretizarea domeniului plan  $\{(x, y), x_1 \leq x \leq x_{n_x}, y_1 \leq y \leq y_{n_y}\}$ , figura 8.1. Considerând punctul  $P_{i,j}$  aflat la intersecția abscisei  $x_i$  cu ordonata  $y_j$  ca element de referință, punctele învecinate au indicii  $(i - 1, j)$ ,  $(i + 1, j)$ ,  $(i, j - 1)$  și  $(i, j + 1)$ .



**Figura 8.1.** Discretizarea domeniului plan.

Pentru obținerea matricelor  $X$  și  $Y$  ale absciselor și ordonatelor punctelor  $P_{i,j}$  (cele două seturi de coordonate) de discretizare ale domeniului plan se utilizează instrucțiunea, [8]:

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

Proprietățile celor două matrice  $X$  și  $Y$  sunt:

- Ambele matrice au aceleași dimensiuni,  $n_y$  rânduri și  $n_x$  coloane.
- Numărul de rânduri ale matricei  $X$  este egal cu numărul  $n_y$  de elemente ale vectorului  $y$ .
- Rândurile matricei  $X$  sunt copii ale vectorului  $x$ .
- Numărul de rânduri ale matricei  $Y$  este egal cu numărul  $n_x$  de elemente ale vectorului  $x$ .
- Coloanele matricei  $Y$  sunt copii ale vectorului  $y$ .

Dacă pentru oricare punct  $P_{i,j}$ , valoarea abscisei este egală cu valoarea ordonatei ( $x_i = y_j$ ), atunci pentru generarea celor două matrice de coordonate  $X$  și  $Y$  se utilizează instrucțiunea:

```
[X, Y]=meshgrid(x)
```

### 8.1.3. Extragerea elementelor unei matrice

Se consideră matricea oarecare:

$$X = \begin{pmatrix} x_{11} & \cdots & x_{1n_C} \\ \vdots & \ddots & \vdots \\ x_{n_L1} & \cdots & x_{n_Ln_C} \end{pmatrix}$$

Extragerea elementelor unei matrice se realizează prin intermediul indicilor. Scopul acestei operațiuni poate fi efectuarea unor operațiuni la nivelul submatricelor, subvectorilor sau a unor elemente individuale ale unei matrice, sau chiar ștergerea unor elemente individuale, submatrice sau subvectori dintr-o matrice.

Primul și respectiv, ultimul element al matricei se obțin cu instrucțiunile:

`X(1,1)`

`X(nL,nC)`

În general, extragerea dintr-o matrice a elementului cu indicii  $(i, j)$  se obține cu instrucțiunea:

`X(i,j)`

În cazul extragerii mai multor elemente ale unei matrice trebuie să se specifice indicii tuturor elementelor care urmează a fi extrase:

- Pentru extragerea liniei  $i$  se utilizează instrucțiunea:

`X(i, :)`

- Pentru extragerea coloanei  $j$  se utilizează instrucțiunea:

`X(:, j)`

- Pentru extragerea doar a primelor 5 elemente ale liniei  $i$  se utilizează instrucțiunea:

`X(i, [1:5])`

- Pentru extragerea doar a primelor 3 elemente ale coloanei  $j$  se utilizează instrucțiunea:

`X([1:3], j)`

- Pentru extragerea submatricei formată din primele trei linii și primele 2 coloane se utilizează instrucțiunea:

`X([1:3], [1 2])`



- Pentru extragerea submatricei formată din liniile 3, 4 și 7 și coloanele 2, 3 și 5 se utilizează instrucțiunea:

$$X([3 \ 4 \ 7], [2 \ 3 \ 5])$$

- Pentru extragerea tuturor elementelor matricei se utilizează instrucțiunea:

$$X(:, :)$$

Ștergerea unei linii sau coloane dintr-o matrice se realizează cu ajutorul unei instrucțiuni prin care fiecărui element al liniei sau coloanei care trebuie să fie șterse i se atribuie elementul nul. De exemplu, pentru ștergerea liniei a treia a matricei  $X$  se utilizează instrucțiunea:

$$X(3, :) = []$$

Pentru ștergerea primelor două coloane ale matricei  $X$  se utilizează instrucțiunea:

$$X(:, [1 \ 2]) = []$$

#### 8.1.4. Operații cu elementele unei matrice

Se consideră matricea oarecare:

$$X = \begin{pmatrix} x_{11} & \cdots & x_{1n_C} \\ \vdots & \ddots & \vdots \\ x_{n_L1} & \cdots & x_{n_Ln_C} \end{pmatrix}$$

Asupra elementelor matricei  $X$  se pot efectua o serie de operații specifice, însă indiferent de operația executată, aceasta se referă doar la coloanele matricei.

Principalele operații care se pot efectua cu elementele matricei sunt:

- Dimensiunea matricei:

$$[n_L \ n_C] = \text{size}(X)$$

în care  $n_L$  și  $n_C$  reprezintă numărul de linii, respectiv numărul de coloane ale matricei.

- Valoarea maximă a elementelor din fiecare coloană a unei matrice:

$$[x_{\max} \ ix_{\max}] = \text{max}(X)$$

$$x_{\max} = \text{max}(X)$$

în care  $x_{\max}$  este valoarea maximă iar  $ix_{\max}$  reprezintă indicele valorii maxime. Pentru matricea  $X$ , identificarea valorii maxime se

face pentru fiecare coloană în parte. Dacă în aceeași coloană se află mai multe valori maxime egale, atunci  $ix_{max}$  reprezintă indicele primei valori maxime din coloana respectivă.

- Valoarea minimă a elementelor din fiecare coloană a unei matrice:

$$[x_{min} \quad ix_{min}] = \min(X)$$

$$x_{min} = \min(X)$$

în care  $x_{min}$  este valoarea minimă iar  $ix_{min}$  reprezintă indicele valorii minime. Pentru matricea  $X$ , identificarea valorii minime se face pentru fiecare coloană în parte. Dacă în aceeași coloană se află mai multe valori minime egale, atunci  $ix_{min}$  reprezintă indicele primei valori minime din coloana respectivă

- Suma elementelor de pe fiecare coloană a unei matrice:

$$S = \text{sum}(X)$$

permite evaluarea relațiilor:

$$S_j = \sum_{i=1}^{n_L} x_{i,j}, \forall j = 1 \div n_C$$

- Produsul elementelor de pe fiecare coloană a unei matrice:

$$P = \text{prod}(X)$$

permite evaluarea relații:

$$P_j = \prod_{i=1}^{n_L} x_{i,j}, \forall j = 1 \div n_C$$

- Media aritmetică a elementelor de pe fiecare coloană a unei matrice:

$$ma = \text{mean}(X)$$

permite evaluarea relațiilor:

$$ma_j = \frac{1}{n_L} \sum_{i=1}^{n_L} x_{i,j}, \forall j = 1 \div n_C$$

- Abaterea medie pătratică a elementelor de pe fiecare coloană a unei matrice:

$$s = \text{std}(X)$$

permite evaluarea relațiilor:

$$s_j = \sqrt{\frac{1}{n_L - 1} \sum_{i=1}^{n_L} (x_{i,j} - ma_j)^2}, \forall j = 1 \div n_C$$

### 8.1.5. Operații între un scalar și o matrice

Se consideră un scalar  $a$  și o matrice  $X$  cu dimensiunea  $(n_{Lx}, n_{Cx})$ :

$$a \in \mathbb{R}$$
$$X = \begin{pmatrix} x_{11} & \cdots & x_{1n_{Cx}} \\ \vdots & \ddots & \vdots \\ x_{n_{Lx}1} & \cdots & x_{n_{Lx}n_{Cx}} \end{pmatrix}$$

Se pune problema efectuării următoarelor calcule:

$$X + a = \begin{pmatrix} x_{11} + a & \cdots & x_{1n_{Cx}} + a \\ \vdots & \ddots & \vdots \\ x_{n_{Lx}1} + a & \cdots & x_{n_{Lx}n_{Cx}} + a \end{pmatrix}$$

$$X - a = \begin{pmatrix} x_{11} - a & \cdots & x_{1n_{Cx}} - a \\ \vdots & \ddots & \vdots \\ x_{n_{Lx}1} - a & \cdots & x_{n_{Lx}n_{Cx}} - a \end{pmatrix}$$

$$X \cdot a = \begin{pmatrix} x_{11} \cdot a & \cdots & x_{1n_{Cx}} \cdot a \\ \vdots & \ddots & \vdots \\ x_{n_{Lx}1} \cdot a & \cdots & x_{n_{Lx}n_{Cx}} \cdot a \end{pmatrix}$$

$$X/a = \begin{pmatrix} x_{11}/a & \cdots & x_{1n_{Cx}}/a \\ \vdots & \ddots & \vdots \\ x_{n_{Lx}1}/a & \cdots & x_{n_{Lx}n_{Cx}}/a \end{pmatrix}$$

$$X^a = \begin{pmatrix} x_{11}^a & \cdots & x_{1n_{Cx}}^a \\ \vdots & \ddots & \vdots \\ x_{n_{Lx}1}^a & \cdots & x_{n_{Lx}n_{Cx}}^a \end{pmatrix}$$

Toate aceste calcule reprezintă operații de tip element-cu-element, pentru care operatorii de adunare, scădere, înmulțire, împărțire și ridicare la putere acționează între variabila scalară  $a$  și fiecare element al matricei  $X$ .

Se utilizează operatorii aritmetici obișnuiți, mai puțin în cazul operației  $X^a$  pentru care trebuie să se utilizeze operatorul de ridicare la putere de tip element-cu-element conform instrucțiunilor:

$X+a$   
 $X-a$   
 $X \cdot a$   
 $X/a$   
 $X.^a$

În cazul în care se urmărește efectuarea operației de ridicare la putere specifică calcului matriceal, atunci trebuie îndeplinită condiția:  $n_{Lx} = n_{Cx}$  (matrice pătrată). Pentru acest caz, realizarea calcului:

$$X^a = X \cdot X \cdot \dots \cdot X \text{ de } a \text{ ori.}$$

se obține cu ajutorul instrucțiunii:

$X^{\wedge}a$

### 8.1.6. Operații între două matrice

Se consideră două matrice  $X$  și  $Y$  cu dimensiunile  $(n_{Lx}, n_{Cx})$  și  $(n_{Ly}, n_{Cy})$ :

$$X = \begin{pmatrix} x_{11} & \cdots & x_{1n_{Cx}} \\ \vdots & \ddots & \vdots \\ x_{n_{Lx}1} & \cdots & x_{n_{Lx}n_{Cx}} \end{pmatrix}, Y = \begin{pmatrix} y_{11} & \cdots & y_{1n_{Cy}} \\ \vdots & \ddots & \vdots \\ y_{n_{Ly}1} & \cdots & y_{n_{Ly}n_{Cy}} \end{pmatrix}$$

Se pune problema efectuării următoarelor calcule:

$$X + Y = \begin{pmatrix} x_{11} + y_{11} & \cdots & x_{1n_{Cx}} + y_{1n_{Cy}} \\ \vdots & \ddots & \vdots \\ x_{n_{Lx}1} + y_{n_{Ly}1} & \cdots & x_{n_{Lx}n_{Cx}} + y_{n_{Ly}n_{Cy}} \end{pmatrix}$$

$$X - Y = \begin{pmatrix} x_{11} - y_{11} & \cdots & x_{1n_{Cx}} - y_{1n_{Cy}} \\ \vdots & \ddots & \vdots \\ x_{n_{Lx}1} - y_{n_{Ly}1} & \cdots & x_{n_{Lx}n_{Cx}} - y_{n_{Ly}n_{Cy}} \end{pmatrix}$$

$$X \cdot Y = \begin{pmatrix} x_{11} \cdot y_{11} & \cdots & x_{1n_{Cx}} \cdot y_{1n_{Cy}} \\ \vdots & \ddots & \vdots \\ x_{n_{Lx}1} \cdot y_{n_{Ly}1} & \cdots & x_{n_{Lx}n_{Cx}} \cdot y_{n_{Ly}n_{Cy}} \end{pmatrix}$$

$$X/Y = \begin{pmatrix} x_{11}/y_{11} & \cdots & x_{1n_{Cx}}/y_{1n_{Cy}} \\ \vdots & \ddots & \vdots \\ x_{n_{Lx}1}/y_{n_{Ly}1} & \cdots & x_{n_{Lx}n_{Cx}}/y_{n_{Ly}n_{Cy}} \end{pmatrix}$$

$$X^Y = \begin{pmatrix} x_{11}^{y_{11}} & \cdots & x_{1n_{Cx}}^{y_{1n_{Cy}}} \\ \vdots & \ddots & \vdots \\ x_{n_{Lx}1}^{y_{n_{Ly}1}} & \cdots & x_{n_{Lx}n_{Cx}}^{y_{n_{Ly}n_{Cy}}} \end{pmatrix}$$

Toate aceste calcule reprezintă operații de tip element-cu-element, pentru care operatorii de adunare, scădere, înmulțire, împărțire și ridicare la putere acționează între fiecare element al primei matrice și elementele corespondente ale celei de-a doua variabile matriceale. Pentru efectuarea tuturor acestor calcule cele două matrice trebuie să aibă aceeași dimensiune, adică trebuie îndeplinite condițiile:  $n_{Lx} = n_{Ly}$  și  $n_{Cx} = n_{Cy}$ .

Se utilizează operatorii aritmetici obișnuiți doar în cazul adunării și scăderii, în rest trebuie utilizați operatorii aritmetici de tip element-cu-element, conform instrucțiunilor:

X+Y  
X-Y  
X.\*Y  
X./Y  
X.^Y

Operația aritmetică de tip element cu element  $X + Y$  (la fiecare element al matricei  $X$  se adună elementul corespondent al matricei  $Y$ ) se poate realiza și cu instrucțiunea:

plus (X, Y)

Operația aritmetică de tip element cu element  $X - Y$  (din fiecare element al matricei  $X$  se scade elementul corespondent al matricei  $Y$ ) se poate realiza și cu instrucțiunea:

minus (X, Y)

Operația aritmetică de tip element cu element  $X \cdot Y$  (fiecare element al matricei  $X$  se înmulțește cu elementul corespondent al matricei  $Y$ ) se poate realiza și cu instrucțiunea:

times (X, Y)

Operația aritmetică de tip element-cu-element  $X/Y$  (fiecare element al matricei  $X$  se împarte la elementul corespondent al matricei  $Y$ ) se realizează cu instrucțiunea  $X ./ Y$  și corespunde împărțirii la dreapta.

Această operație aritmetică se poate efectua și cu instrucțiunea:

rdivide (X, Y)

Pentru cele două matrice  $X$  și  $Y$  se poate pune problema efectuării operației aritmetice de împărțire la stânga  $Y \setminus X$  (fiecare element al matricei  $X$  se împarte la elementul corespondent al matricei  $Y$ ), caz în care se pot utiliza instrucțiunile:

$Y \setminus X$   
ldivide (Y, X)

În cazul efectuării operației aritmetice de înmulțire, specifice calculului matriceal, trebuie îndeplinită condiția  $n_{Cx} = n_{Ly}$ . Pentru acest caz, efectuarea calculelor se realizează cu ajutorul instrucțiunilor:

$X * Y$   
mtimes (X, Y)

care conduc la obținerea matricei:

$$C = \begin{pmatrix} c_{11} & \cdots & c_{1n_{Cy}} \\ \vdots & \ddots & \vdots \\ c_{n_{Lx}1} & \cdots & c_{n_{Lx}n_{Cy}} \end{pmatrix}$$

având elementele calculate conform relației:

$$c_{ij} = \sum_{k=1}^{n_{Cx}} x_{i,k} \cdot y_{k,j}$$
$$\forall i = 1 \div n_{Lx}, \forall j = 1 \div n_{Cy}$$

### 8.1.7. Operații specifice analizei matriceale

Se consideră o matrice pătrată având dimensiunea  $n \times n$ :

$$X = \begin{pmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{pmatrix}$$

Pentru calculul determinantului se utilizează instrucțiunea, [9]:

$$d = \det(X)$$

În cazul în care determinantul matricei este diferit de zero atunci matricea este nesingulară. Pentru acest caz, are sens determinarea matricei inverse  $X^{-1}$ , care prin definiție, este matricea ce satisface egalitățile:

$$X \cdot X^{-1} = X^{-1} \cdot X = I$$

în care  $I$  este matricea identitate, având aceeași dimensiune cu cea a matricei  $X$  ( $n \times n$ ).

Pentru calculul matricei inverse se utilizează instrucțiunea, [10]:

$$iX = \text{inv}(X)$$

Suma elementelor de pe diagonala principală (urma matricei) se calculează cu instrucțiunea, [11]:

$$t = \text{trace}(X)$$

Se consideră o matrice oarecare având dimensiunea  $n_L \times n_C$ :

$$X = \begin{pmatrix} x_{11} & \cdots & x_{1n_C} \\ \vdots & \ddots & \vdots \\ x_{n_L 1} & \cdots & x_{n_L n_C} \end{pmatrix}$$

Rangul matricei  $X$  este ordinul  $k$  al minorului nenul (dacă există), toți minorii de ordin mai mare decât  $k$  (dacă există) fiind nuli. Rangul matricei  $X$  reprezintă deci numărul de linii sau coloane liniar independente, fiind un scalar care verifică inegalitățile:

$$1 \leq k \leq \min(n_L, n_C)$$

Pentru calculul rangului matricei se utilizează instrucțiunea, [12]:

$$rx = \text{rank}(X)$$

În cazul matricelor pătrate singulare, sau al matricelor nepătrate calculul matricei inverse nu are sens. Pentru aceste cazuri, se poate utiliza însă o matrice particulară, numită matricea pseudo-inversă care are doar unele din proprietățile matricei inverse.

Pentru calculul matricei pseudo-inverse (Moore-Penrose) se utilizează instrucțiunea, [13]:

$$pX = \text{pinv}(X)$$

### Problema 8.1

Distribuția de viteză a vântului în stratul limită atmosferic este aproximată prin legea logaritmică:

$$V_z = V_r \cdot \frac{\ln \frac{z}{z_0}}{\ln \frac{z_r}{z_0}}$$

în care:  $V_z$  [m/s] este viteza medie a vântului la înălțimea  $z$ ,  $V_r=7$  m/s este viteza medie a vântului măsurată la înălțimea de referință  $z_r=10$  m, iar  $z_0$  [m] este parametrul de rugozitate al terenului.

Cunoscând înălțimea de amplasare a rotorului turbinei eoliene  $\delta=80$  m, să se calculeze și să se reprezinte grafic profilul vitezei vântului pentru  $1 \leq z \leq \delta$  și trei locații diferite având caracteristicile, [21]:

- Suprafața apei, pentru care  $z_0=0,0002$  m.
- Zonă agricolă deschisă cu suprafață netedă sau ușor ondulată, pentru care  $z_0=0,03$  m.
- Sate, orașe mici, zone agricole cu multe elemente de vegetație, păduri și terenuri cu rugozitate foarte ridicată, pentru care parametrul de rugozitate al terenului este  $z_0=0,4$  m.

### Rezolvare

Conform datelor problemei, în relația de calcul a variației vitezei vântului în stratul limită atmosferic există două variabile vectoriale: parametrul de rugozitate al terenului  $z_0=\{0,0002; 0,03; 0,4\}$  m și înălțimea  $z=1 \div \delta$  m. Efectuarea calculelor presupune că pentru prima valoare a parametrului de rugozitate  $z_0=0,0002$  m, se calculează viteza vântului  $V_z$  [m/s] la fiecare înălțime  $z=1 \div \delta$  m. Apoi se repeta calculele pentru a doua valoare  $z_0=0,03$  m, respectiv pentru cea de-a treia valoare a parametrului de rugozitate  $z_0=0,4$  m.

Pentru rezolvarea acestei probleme se pot utiliza mai multe metode: metoda structurilor iterative (de exemplu cu contor); metoda discretizării meshgrid; metoda funcțiilor de tip anonymous.

**Metoda structurii iterative cu contor.** Rezolvarea problemei presupune scrierea unui fișier de tip script de forma:

```
%% DISTRIBUTIA DE VITEZA IN STRATUL LIMITA
% ATMOSFERIC
% Legea logaritmica
close all;clear all;clc;
%% DATE DE INTRARE
% Inaltimea de referinta [m]:
zr=10;
% Viteza de referinta masurata [m/s]:
Vr=7;
```

```

% Parametrul de rugozitate al terenului [m]:
z0=[0.0002 0.03 0.4];nz0=length(z0);
% Grosimea stratului limita atmosferic [m]:
delta=80;
% Discretizarea stratului limita atmosferic:
z=linspace(1,delta,100);nz=length(z);
%% 1.METODA STRUCTURII ITERATIVE CU CONTOR
for i=1:nz0
    for j=1:nz
        V1(i,j)=Vr*log(z(j)/z0(i))/log(zr/z0(i));
    end
end
figure
plot(V1(1,:),z,'-k');hold on;
plot(V1(2,:),z,'--k');
plot(V1(3,:),z,':k');hold off;grid on;

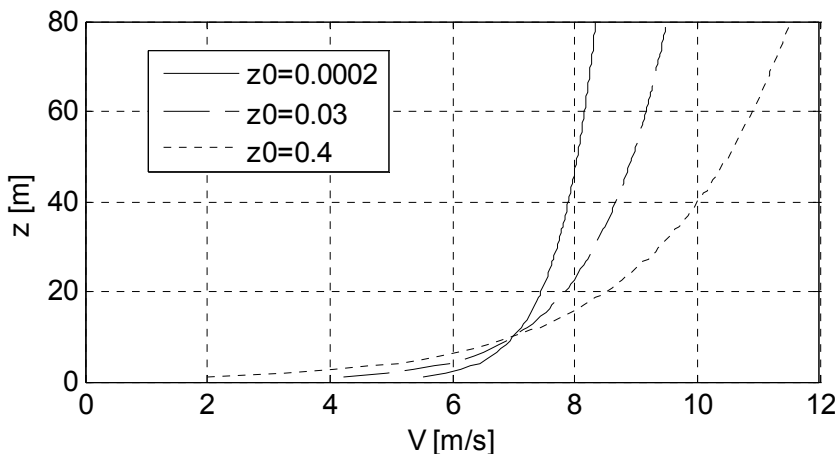
```

În urma lansării în execuție a fișierului se obține reprezentare grafică comparativă din figura 8.2, precum și următoarele rezultate numerice:

```

V1'
ans =
    5.5103    4.2254    1.9926
    6.9855    6.9729    6.9511
    7.3999    7.7449    8.3443
    7.6505    8.2117    9.1867
    7.8307    8.5472    9.7923
    7.9715    8.8094   10.2655
    8.0870    9.0246   10.6539
    8.1850    9.2072   10.9833
    8.2701    9.3657   11.2694
    8.3453    9.5057   11.5221

```



**Figura 8.2.** Viteza vântului în stratul limită atmosferic.



## Observații

- Instrucțiunea repetitivă este o instrucțiune `for` dublă în care primul contor, `for i=1:nz0` parcurge toate valorile parametrului de rugozitate  $z_0$ , iar cel de-al doilea contor, `for j=1:nz` parcurge toate valorile  $z$  ale discretizării grosimii stratului limită atmosferic.
- Viteza vântului se calculează cu instrucțiunea  $V1(i,j) = V_r * \log(z(j)/z_0(i)) / \log(z_r/z_0(i))$ . Rezultatul obținut,  $V1$ , este o matrice cu dimensiunea  $nz0 \times nz$ . În cadrul instrucțiunii de calcul a vitezei vântului nu este necesară folosirea operațiilor cu punct (`.` sau `./`) datorită faptului că la fiecare iterație ( $i, j$ ) se operează doar cu variabile scalare.
- Reprezentarea grafică se obține cu trei instrucțiuni de tip `plot`, fiecare realizând trasarea curbei de variație a vitezei vântului pentru câte o valoare a parametrului de rugozitate a terenului. Se utilizează parametri de formatare diferiți pentru cele trei curbe: `'-k'`, `'--k'` și `':k'`. De asemenea, s-au utilizat instrucțiunile `hold on` și `hold off` pentru ca cele trei curbe să se reprezinte în aceleași axe. Îmbunătățirea calității reprezentării grafice implică creșterea numărului de puncte de discretizate pentru domeniul  $z=1 \div \delta$  m. Astfel, pentru reprezentările grafice s-au utilizat 100 puncte, în timp ce rezultatele numerice corespund unei discretizări cu 10 puncte.

**Metoda discretizării `meshgrid`.** Rezolvarea problemei presupune completarea fișierului `script` anterior cu următoarele instrucțiuni:

```
[Z,Z0]=meshgrid(z,z0);  
V2=Vr*log(Z./Z0)./log(zr./Z0);  
figure  
plot(V2(1,:),z,'-k');hold on;grid on;  
plot(V2(2,:),z,'--k');  
plot(V2(3,:),z,':k');hold off;
```

## Observații

- Domeniile vectoriale  $z$  și  $z_0$ , având dimensiunile  $1 \times nz$  și  $1 \times nz_0$ , se transformă în două domenii matriceale  $Z$  și  $Z_0$ , având dimensiunea  $nz \times nz_0$ , cu ajutorul instrucțiunii `[Z,Z0]=meshgrid(z,z0)`.
- Instrucțiunea prin care se determină valorile numerice ale vitezei vântului,  $V2=V_r * \log(Z./Z_0) ./ \log(z_r./Z_0)$  realizează calcule de tip element-cu-element între fiecare valoare a matricei  $Z$  și valoarea corespondentă a matricei  $Z_0$ . Prin urmare este necesară folosirea operațiilor cu punct (`./`).

**Metoda funcțiilor de tip anonymous.** Rezolvarea problemei presupune completarea fișierului script anterior cu următoarele instrucțiuni:

```
V3=@(z0,z) Vr*log(z./z0)./log(zr./z0);
figure
plot(V3(z0(1),z),z,'-k');hold on;
plot(V3(z0(2),z),z,'--k');
plot(V3(z0(3),z),z,':k');hold off;
```

### Observații

- A fost definită funcția V3, cu doi parametri z0 și z, prin instrucțiunea V3=@(z0,z) Vr\*log(z/z0)/log(zr/z0).
- Calculul efectiv al valorilor numerice ale vitezei vântului se face la apelarea funcției, în structura instrucțiunilor de reprezentare grafică. La apelarea funcției, se transferă variabila vectorială z și doar câte una din cele trei valori ale parametrului de rugozitate z0(1), z0(2) și z0(3), astfel încât calculele se reduc la operații între un scalar și un vector.

### Problema 8.2

Se consideră un ventilator centrifugal proiectat prin metoda valorii constante a unghiului paletei ( $\beta = \text{ct.}$ ) pentru care se cunosc următorii parametri: numărul de palete  $n_z = 12$ ; raza paletei la intrare  $r_1 = 0,25$  m; raza paletei la ieșire  $r_2 = 0,5$  m și unghiul paletei  $\beta = 30^\circ$ , [1].

Să se reprezinte grafic paletele, precum și cercurile de intrare și de ieșire ale rotorului ventilatorului centrifugal.

### Rezolvare

Ecuatiile cercurilor de intrare și de ieșire sunt:

$$\begin{cases} x_{c1}(k) = r_1 \cdot \cos[\theta(k)] \\ y_{c1}(k) = r_1 \cdot \sin[\theta(k)] \end{cases} \text{ și } \begin{cases} x_{c2}(k) = r_2 \cdot \cos[\theta(k)] \\ y_{c2}(k) = r_2 \cdot \sin[\theta(k)] \end{cases}$$

în care  $k=1:n_\theta$  reprezintă indicele celor  $n_\theta$  valori în care se discretizează domeniul  $\theta = 0 \cdots 2\pi$ .

Calculul coordonatelor unei palete a ventilatorului centrifugal presupune parcurgerea următoarelor etape:

- Se definesc razele de calcul în intervalul  $r = r_1 \cdots r_2$ , folosind  $n_r = 100$  de puncte.
- Se calculează unghiul caracteristic  $\varphi$  pentru trasarea prin puncte a unei paletei:

$$\varphi(i) = \frac{\ln[r(i)/r_1]}{\tan(\beta)}$$

în care  $i=1:n_r$  este indicele razelor de calcul.

- Se calculează coordonatele unei paleți  $x_p$  și  $y_p$ :

$$\begin{cases} x_p(i) = r(i) \cdot \cos[\varphi(i)] \\ y_p(i) = r(i) \cdot \sin[\varphi(i)] \end{cases}$$

Calculul coordonatelor tuturor celor  $n_z$  palete conduce la modificarea relațiilor de calcul după cum urmează:

$$\begin{cases} x_p(i,j) = r(i) \cdot \cos\left[\varphi + (j-1) \cdot \frac{2\pi}{n_z}\right] \\ y_p(i,j) = r(i) \cdot \sin\left[\varphi + (j-1) \cdot \frac{2\pi}{n_z}\right] \end{cases}$$

în care  $j=1:n_z$  este indicele fiecărei palete.

Pentru rezolvare, se utilizează metoda structurii iterative cu contor:

```
%% CALCULUL COORDONATELOR PALETELOR UNUI
% VENTILATOR CENTRIFUGAL
close all;clear all;clc;
%% DATE DE INTRARE
% Numarul de palete
nz=12;
% Unghiul constant al paletei
beta=30;
% Raza paletei la intrare si la iesire [m]
r1=0.25;
r2=0.5;
% Razele de calcul
nr=100;r=linspace(r1,r2,nr);
% Unghiurile de calcul ale cercurilor
ntheta=50;theta=linspace(0,2*pi,ntheta);
%% CALCULUL SI REPREZENTAREA GRAFICA A PALETELOR
figure
for i=1:nz
    for j=1:nr
        phi(j)=1/(tan(beta*pi/180))*log(r(j)/r1);
        xp(i,j)=r(j).*cos(phi(j)+(i-1)*2*pi/nz);
        yp(i,j)=r(j).*sin(phi(j)+(i-1)*2*pi/nz);
    end
    plot(xp(i,:),yp(i,:),'-k');hold on;grid on;
end
%% CALCULUL SI REPREZENTAREA GRAFICA A CERCURILOR
xc1=r1*cos(theta);yc1=r1*sin(theta);
xc2=r2*cos(theta);yc2=r2*sin(theta);
plot(xc1,yc1,'k');plot(xc2,yc2,'k');
%% FORMATAREA GRAFICULUI
text(-0.1,0.005,['n_z=' num2str(nz) ' palete';
'\beta=' num2str(beta) '\circ'],'FontSize',12);
xlabel('x[m]');ylabel('y[m]');axis image;hold off;
```

Lansarea în execuție a fișierului `script` conduce la reprezentarea grafică din figura 8.3.

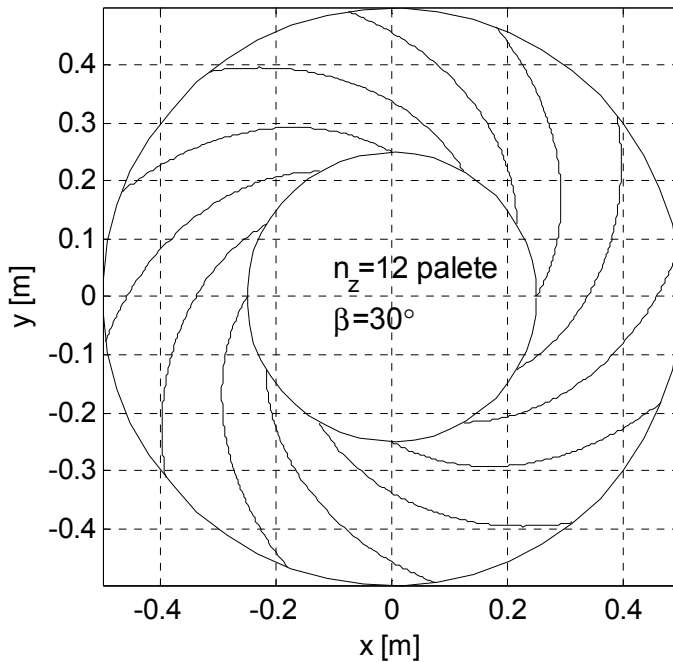


Figura 8.3. Rotorul ventilatorului centrifugal.

### Observații

- Instrucțiunea repetitivă este o instrucțiune `for` dublă în care primul contor `for i=1:nz` parcurge toate cele  $n_z=12$  palete ale ventilatorului, iar cel de-al doilea contor `for j=1:nr` parcurge toate cele  $n_r=100$  raze de calcul.
- Pentru prima paletă  $i=1$ , se parcurge structura iterativă interioară prin care se calculează valorile numerice ale unghiului caracteristic și coordonatele paletei la fiecare rază de calcul. După ieșirea din structura iterativă interioară, se reprezintă grafic prima paletă. Instrucțiunea `hold on` determină ca reprezentarea celorlalte palete, ca și a cercurilor de intrare și de ieșire să se realizeze în aceeași fereastră grafică în care s-a reprezentat prima paletă.
- Pentru celelalte palete  $i=2:nz$ , în relațiile de calcul ale coordonatelor se adaugă un termen  $(i-1) * 2 * \pi / nz$  proporțional cu unghiul dintre prima paletă și paleta curentă. Pentru prima paletă,  $i=1$ , deci acest termen se anulează.
- Pentru amplasarea textului explicativ în interiorul reprezentării grafice s-a utilizat instrucțiunea `text`.

## 8.2. REZOLVAREA SISTEMELOR DE ECUAȚII ALGEBRICE LINIARE

### 8.2.1. Generalități

Se consideră un sistem general de  $m$  ecuații algebrice liniare cu  $n$  necunoscute  $x_1, x_2, \dots, x_n$ , de forma:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2 \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n = b_m \end{cases}$$

Sistemul de ecuații poate fi scris sub formă matriceală:

$$A \cdot X = B$$

în care:

- $A$  este matricea coeficienților:

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$$

- $X$  este vectorul coloană al necunoscutelor:

$$X = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

- $B$  este vectorul coloană al termenilor liberi:

$$B = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}$$

Rezolvarea sistemului de ecuații algebrice liniare presupune, în principal, parcurgerea următoarelor etape:

- Analiza existenței și unicității soluției sistemului.
- Rezolvarea propriu-zisă și găsirea soluțiilor.
- Verificarea soluțiilor obținute.

### 8.2.2. Analiza existenței și unicității soluției sistemului

Presupune parcurgerea mai multor etape:

- Se definesc datele de intrare: matricea coeficienților  $A$  și vectorul coloană al termenilor liberi  $B$ .
- Se determină matricea extinsă prin concatenarea matricei  $A$  cu vectorul coloană  $B$  folosind instrucțiunea:

$$C = [A \ B]$$

- Se calculează rangul  $r_A$  al matricei coeficienților  $A$  cu instrucțiunea:  
 $rA = \text{rank}(A)$
- Se calculează rangul  $r_C$  al matricei extinse  $C$  cu instrucțiunea:  
 $rC = \text{rank}(C)$
- Dacă  $r_A = r_C$ , sistemul de ecuații admite soluții exacte.
  - Dacă în plus  $r_A = r_C = n$ , atunci sistemul de ecuații admite soluție exactă unică.
  - Dacă însă  $r_A = r_C < n$ , atunci sistemul de ecuații admite o infinitate de soluții exacte și  $r_A$  necunoscute (necunoscute principale) se pot exprima în funcție de celelalte  $n - r_A$  necunoscute rămase (necunoscute secundare).
- Dacă  $r_A \neq r_C$  sistemul de ecuații nu admite soluții exacte. În acest caz se poate determina totuși, o soluție aproximativă a sistemului de ecuații (în sensul celor mai mici pătrate).

### 8.2.3. Rezolvarea propriu-zisă și găsirea soluțiilor

Rezolvarea sistemului de ecuații algebrice liniare se face în mod diferit în funcție de rangul matricei coeficienților și cel al matricei extinse, de tipul sistemului de ecuații (determinat, supradeterminat, subdeterminat), de valoarea determinantului sistemului:

- Dacă  $r_A = r_C = n$ 
  - Dacă  $m = n$ , se calculează determinantul sistemului cu instrucțiunea:  
 $D = \det(A)$ 
    - Dacă determinantul sistemului este diferit de zero ( $D \neq 0$ ), atunci rezolvarea sistemului se efectuează prin metoda matricei inverse, obținându-se soluția exactă unică a sistemului de ecuații:  
 $X = \text{inv}(A) * B$
    - Dacă determinantul sistemului este egal cu zero ( $D = 0$ ), atunci rezolvarea sistemului se efectuează prin metoda matricei pseudo-inverse, obținându-se soluția exactă unică a sistemului de ecuații:  
 $X = \text{pinv}(A) * B$
  - Dacă  $m \neq n$ , rezolvarea sistemului se efectuează prin metoda matricei pseudo-inverse, obținându-se soluția exactă unică a sistemului de ecuații:

$$X = \text{pinv}(A) * B$$

- Dacă  $r_A = r_C < n$ 
  - Dacă  $m = n$ , se calculează determinantul sistemului cu instrucțiunea:

$$D = \det(A)$$

- Dacă determinantul sistemului este diferit de zero ( $D \neq 0$ ), atunci se utilizează metoda matricei inverse, obținându-se o soluție exactă particulară:

$$X = \text{inv}(A) * B$$

Pentru a determina forma generală a soluțiilor sistemului se va determina matricea redusă cu instrucțiunea, [14]:

$$R = \text{rref}(C)$$

Se verifică ultima linie a matricei obținute care trebuie să conțină doare elemente zero. Celelalte linii ale matricei reduse permit determinarea formai generale a soluției sistemului.

- Dacă determinantul sistemului este egal cu zero ( $D = 0$ ), atunci rezolvarea sistemului se efectuează prin metoda matricei pseudo-inverse, obținându-se o soluția particulară (exactă sau aproximativă) a sistemului de ecuații:

$$X = \text{pinv}(A) * B$$

Pentru a verifica dacă soluția obținută este exactă sau aproximativă și pentru a determina forma generală a soluțiilor sistemului se va determina matricea redusă cu instrucțiunea:

$$R = \text{rref}(C)$$

Dacă matricea astfel obținută conține pe ultima linie doar elemente zero, atunci soluția sistemului este o soluție exactă.

Dacă în schimb, pe ultima linie există cel puțin o valoare diferită de zero, atunci soluția va fi aproximativă.

Celelalte linii ale matricei reduse permit determinarea formai generale a soluției sistemului.

- Dacă  $m \neq n$ , rezolvarea sistemului se efectuează prin metoda matricei pseudo-inverse, obținându-se o soluție particulară (exactă sau aproximativă) a sistemului de ecuații:

$$X = \text{pinv}(A) * B$$

Pentru a verifica dacă soluția obținută este exactă sau aproximativă și pentru a determina forma generală a soluțiilor sistemului se va determina matricea redusă cu instrucțiunea:

$$R = \text{rref}(C)$$

Dacă matricea astfel obținută conține pe ultima linie doar elemente zero, atunci soluția sistemului este o soluție exactă. Dacă în schimb, pe ultima linie există cel puțin o valoare diferită de zero, atunci soluția va fi aproximativă. Celelalte linii ale matricei reduse permit determinarea formei generale a soluției sistemului.

- Dacă  $r_A \neq r_C$ , rezolvarea sistemului se efectuează prin metoda matricei pseudo-inverse, obținându-se o soluție aproximativă a sistemului de ecuații:

$$X = \text{pinv}(A) * B$$

Pentru a verifica faptul că soluția obținută este aproximativă se va determina matricea redusă cu instrucțiunea:

$$R = \text{rref}(C)$$

Se verifică ultima linie a matricei obținute care trebuie să conțină cel puțin un element diferit de zero.

Rezolvarea sistemului de ecuații  $A \cdot X = B$  se poate face și direct, cu instrucțiunea, [15]:

$$X = \text{linsolve}(A, B)$$

Pentru a verifica sensibilitatea soluțiilor obținute față de erorile coeficienților sistemului, se calculează numărul de condiționare al matricei coeficienților cu instrucțiunea, [16]:

$$c = \text{cond}(A)$$

Dacă matricea  $A$  este bine condiționată, atunci  $c \cong 1$ .



### Problema 8.3

Se consideră sistemul de ecuații algebrice liniare:

$$\begin{cases} 2x_1 + x_2 + 3x_3 = 2 \\ 3x_1 + 3x_2 + 2x_3 = 11 \\ x_1 + 2x_2 + x_3 = 5 \end{cases}$$

Se cere:

- Să se studieze existența și unicitatea soluției.
- Să se rezolve.
- Să se verifice soluția obținută.

### Rezolvare

- Se definește matricea coeficienților:

```
>> A=[2 1 3;3 3 2;1 2 1]
```

```
A =
```

```
     2     1     3
     3     3     2
     1     2     1
```

- Se definește vectorul termenilor liberi:

```
>> B=[2 11 5]'
```

```
B =
```

```
     2
    11
     5
```

- Se definește matricea extinsă:

```
>> C=[A B]
```

```
C =
```

```
     2     1     3     2
     3     3     2    11
     1     2     1     5
```

- Se calculează rangul matricei A:

```
>> rA=rank(A)
```

```
rA =
```

```
     3
```

- Se calculează rangul matricei C:

```
>> rC=rank(C)
```

```
rC =
```

```
     3
```

- Rangurile sunt egale între ele și în plus, egale cu numărul de necunoscute, deci sistemul de ecuații admite soluție exactă unică.

- Se calculează determinantul matricei A:

```
>> D=det(A)
```

```
D =
```

6

- Se determină soluția unică a sistemului de ecuații:

```
>> X=inv(A)*B
```

```
X =
```

```
3
```

```
2
```

```
-2
```

- Se verifică soluția astfel obținută:

```
>> A*X
```

```
ans =
```

```
2
```

```
11
```

```
5
```

- Se analizează matricea redusă:

```
>> rref(C)
```

```
ans =
```

```
1 0 0 3
```

```
0 1 0 2
```

```
0 0 1 -2
```

- În acest caz, matricea redusă furnizează soluția exactă unică a sistemului de ecuații:

$$\begin{cases} 1 \cdot x_1 + 0 \cdot x_2 + 0 \cdot x_3 = 3 \\ 0 \cdot x_1 + 1 \cdot x_2 + 0 \cdot x_3 = 2 \\ 0 \cdot x_1 + 0 \cdot x_2 + 1 \cdot x_3 = -2 \end{cases} \Rightarrow \begin{cases} x_1 = 3 \\ x_2 = 2 \\ x_3 = -2 \end{cases}$$

#### Problema 8.4

Se consideră sistemul de ecuații algebrice liniare:

$$\begin{cases} x_1 + 3x_2 + 7x_3 = 5 \\ -x_1 + 4x_2 + 4x_3 = 2 \\ x_1 + 10x_2 + 18x_3 = 12 \end{cases}$$

Se cere:

- Să se studieze existența și unicitatea soluției.
- Să se rezolve.
- Să se verifice soluția obținută.

#### Rezolvare

- Se definește matricea coeficienților:

```
>> A=[1 3 7;-1 4 4;1 10 18]
```

```
A =
```

```
1 3 7
```

```
-1 4 4
```

```
1 10 18
```

- Se definește vectorul termenilor liberi:  

```
>> B=[5 2 12]'
```

```
B =
```

```
    5
```

```
    2
```

```
   12
```
- Se definește matricea extinsă:  

```
>> C=[A B]
```

```
C =
```

```
    1    3    7    5
```

```
   -1    4    4    2
```

```
    1   10   18   12
```
- Se calculează rangul matricei A:  

```
>> rA=rank(A)
```

```
rA =
```

```
    2
```
- Se calculează rangul matricei C:  

```
>> rC=rank(C)
```

```
rC =
```

```
    2
```
- Rangurile sunt egale între ele dar nu sunt egale și cu numărul de necunoscute, prin urmare sistemul de ecuații admite o infinitate de soluții exacte.
- Se calculează determinantul matricei A:  

```
>> D=det(A)
```

```
D =
```

```
    0
```
- Se determină o soluție particulară a sistemului de ecuații:  

```
>> X=pinv(A)*B
```

```
X =
```

```
    0.3850
```

```
   -0.1103
```

```
    0.7066
```
- Se verifică soluția astfel obținută:  

```
>> A*X
```

```
ans =
```

```
    5.0000
```

```
    2.0000
```

```
   12.0000
```
- Se analizează matricea redusă:  

```
>> rref(C)
```

```
ans =
```

$$\begin{array}{cccc} 1.0000 & 0 & 2.2857 & 2.0000 \\ 0 & 1.0000 & 1.5714 & 1.0000 \\ 0 & 0 & 0 & 0 \end{array}$$

- În acest caz, matricea redusă furnizează soluția generală a sistemului de ecuații:

$$\begin{cases} 1 \cdot x_1 + 0 \cdot x_2 + 2.2857 \cdot x_3 = 2 \\ 0 \cdot x_1 + 1 \cdot x_2 + 1.5714 \cdot x_3 = 1 \end{cases} \Rightarrow \begin{cases} x_1 = 2 - 2.2857 \cdot x_3 \\ x_2 = 1 - 1.5714 \cdot x_3 \end{cases}$$

Se observă că două din soluțiile sistemului ( $x_1$  și  $x_2$ ) se exprimă în funcție de cea de-a treia soluție ( $x_3$ ). Astfel, dând valori lui  $x_3$  (în domeniul  $-\infty, +\infty$ ), se obțin celelalte două soluții,  $x_1$  și  $x_2$ .

### Problema 8.5

Se consideră sistemul de ecuații algebrice liniare:

$$\begin{cases} x_1 + 3x_2 + 7x_3 = 3 \\ -x_1 + 4x_2 + 4x_3 = 6 \\ x_1 + 10x_2 + 18x_3 = 0 \end{cases}$$

Se cere:

- Să se studieze existența și unicitatea soluției.
- Să se rezolve.
- Să se verifice soluția obținută.

### Rezolvare

- Se definește matricea coeficienților:

```
>> A = [1 3 7; -1 4 4; 1 10 18]
```

A =

$$\begin{array}{ccc} 1 & 3 & 7 \\ -1 & 4 & 4 \\ 1 & 10 & 18 \end{array}$$

- Se definește vectorul termenilor liberi:

```
>> B = [3 6 0]'
```

B =

$$\begin{array}{c} 3 \\ 6 \\ 0 \end{array}$$

- Se definește matricea extinsă:

```
>> C = [A B]
```

C =

$$\begin{array}{cccc} 1 & 3 & 7 & 3 \\ -1 & 4 & 4 & 6 \\ 1 & 10 & 18 & 0 \end{array}$$

- Se calculează rangul matricei A:

```
>> rA = rank(A)
```

```
rA =  
    2
```

- Se calculează rangul matricei C:

```
>> rC=rank(C)  
rC =  
    3
```

- Rangurile sunt diferite, prin urmare sistemul de ecuații nu are soluții exacte.

- Se calculează determinantul matricei A:

```
>> D=det(A)  
D =  
    0
```

- Se determină o soluție aproximativă a sistemului de ecuații:

```
>> X=pinv(A)*B  
X =  
   -1.0892  
    1.2512  
   -0.5235
```

- Se verifică soluția astfel obținută:

```
>> A*X  
ans =  
   -1.0000  
    4.0000  
    2.0000
```

Pentru cazul unei soluții exacte, etapa de verificare trebuie să conducă la obținerea vectorului termenilor liberi. Cum în acest caz, rezultatul etapei de verificare (-1; 4; 2) diferă de valorile vectorului termenilor liberi (3; 6; 0), se poate stabili ca și concluzie, faptul că soluția obținută nu este exactă.

- Se analizează matricea redusă:

```
>> rref(C)  
ans =  
    1.0000         0    2.2857         0  
         0    1.0000    1.5714         0  
         0         0         0    1.0000
```

În acest caz, matricea redusă are pe ultima linie o valoare diferită de zero, prin urmare soluția obținută este o soluție aproximativă (în sensul celor mai mici pătrate).

### 8.3. TRANSFORMĂRI GEOMETRICE AFINE

Se consideră un sistem de axe cartezian  $Oxyz$  și un punct  $P_1$  oarecare, având coordonatele  $(x_1, y_1, z_1)$ , figura 8.4. Transformarea geometrică  $T(x, y, z)$ , asociază punctului  $P_1(x_1, y_1, z_1)$  un nou punct  $P_2$ , ale cărui coordonate  $(x_2, y_2, z_2)$  depind de coordonatele punctului  $P_1$  și de forma particulară a transformării geometrice  $T$ , [17].

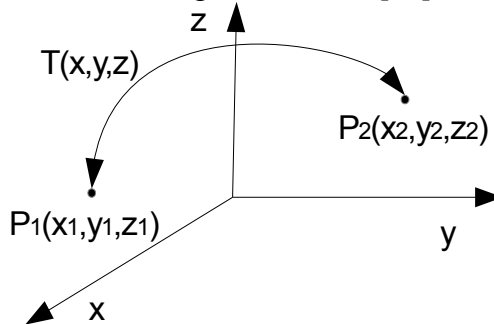


Figura 8.4. Transformarea geometrică generală.

Principalele tipuri de transformări geometrice sunt:

- **Translația.** Este transformarea geometrică prin care punctul  $P_1(x_1, y_1, z_1)$  se translatează în punctul  $P_2(x_2, y_2, z_2)$ , printr-o transformare de forma:

$$\begin{cases} x_2 = x_1 + t_x \\ y_2 = y_1 + t_y \\ z_2 = z_1 + t_z \end{cases}$$

în care  $t_x$ ,  $t_y$  și  $t_z$  sunt deplasările pe cele trei axe de coordonate.

- **Scalarea.** Este transformarea geometrică prin care punctul  $P_1(x_1, y_1, z_1)$  se scalează față de origine în punctul  $P_2(x_2, y_2, z_2)$ , printr-o transformare de forma:

$$\begin{cases} x_2 = s_x \cdot x_1 \\ y_2 = s_y \cdot y_1 \\ z_2 = s_z \cdot z_1 \end{cases}$$

în care  $s_x$ ,  $s_y$  și  $s_z$  sunt factorii de scalare față de cele trei axe de coordonate. Factori de scalare supraunitari determină mărire, iar cei subunitari determină micșorare. Pentru cazul  $s_x=s_y=s_z$  se obține scalarea omogenă.

- **Rotația.** Este transformarea geometrică prin care punctul  $P_1(x_1, y_1, z_1)$  se rotește în punctul  $P_2(x_2, y_2, z_2)$ , printr-o transformare de forma:

$$\begin{cases} x_2 = x_1 \cos \theta_z - y_1 \sin \theta_z \\ y_2 = x_1 \sin \theta_z + y_1 \cos \theta_z \\ z_2 = z_1 \end{cases}$$

în care  $\theta_z$  [rad] este unghiul de rotație în jurul axei  $Oz$ .

Pentru reprezentarea matematică unitară a tuturor tipurilor de transformări geometrice prin intermediul operației de înmulțire a matricelor, se definesc coordonatele geometrice omogene prin adăugarea unei dimensiuni suplimentare astfel încât să se realizeze echivalența:

$$(x, y, z) \leftrightarrow (x_w, y_w, z_w, w)$$

Legătura dintre sistemul de coordonate carteziene și sistemul de coordonate omogene se face prin intermediul relațiilor de legătură:

$$x = x_w/w, y = y_w/w, z = z_w/w$$

în care  $w \neq 0$  reprezintă transformarea de scalare globală. Punctele pentru care  $w=0$  definesc mulțimea punctelor de la infinit (planul de la infinit).

Folosind sistemul de coordonate omogene, transformările geometrice se definesc printr-o relație generală de forma:

$$X_2 = X_1 \cdot M$$

în care  $X_1 = (x_1, y_1, z_1, 1)$  este vectorul linie al coordonatelor omogene ale punctului inițial;  $X_2 = (x_2, y_2, z_2, 1)$  este vectorul linie al coordonatelor omogene ale punctului transformat, iar  $M$  este matricea de transformare.

Forma generală a matricei de transformare este:

$$M = \begin{pmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & 1 \end{pmatrix}$$

Submatricea:

$$\begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{pmatrix}$$

definește transformările geometrice de rotație, scalare și simetrie.

Vectorul linie:

$$m_{41} \quad m_{42} \quad m_{43}$$

definește transformarea geometrică de translație.

Vectorul coloană:

$$\begin{pmatrix} m_{14} \\ m_{24} \\ m_{34} \end{pmatrix}$$

definește transformarea geometrică de proiecție în perspectivă.

Transformările geometrice pentru care matricea de transformare este de forma  $M$  și prin care se conservă paralelismul liniilor și planelor, se numesc transformări geometrice afine.

Matricele de transformare ale principalelor transformări afine sunt:

- Translația:

$$T(t_x, t_y, t_z) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ t_x & t_y & t_z & 1 \end{pmatrix}$$

- Scalarea și simetria:

$$S(s_x, s_y, s_z) = \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Cazuri particulare:

- Scalare uniformă,  $s_x=s_y=s_z$ .
  - Simetrie față de planul  $Oyz$ ,  $s_y=s_z=1$ ,  $s_x=-1$ .
  - Simetrie față de planul  $Ozx$ ,  $s_z=s_x=1$ ,  $s_y=-1$ .
  - Simetrie față de planul  $Oxy$ ,  $s_x=s_y=1$ ,  $s_z=-1$ .
  - Simetrie față de axa  $Oz$ ,  $s_z=1$ ,  $s_x=s_y=-1$ .
  - Simetrie față de axa  $Oy$ ,  $s_y=1$ ,  $s_x=s_z=-1$ .
  - Simetrie e față de axa  $Ox$ ,  $s_x=1$ ,  $s_z=s_y=-1$ .
- Rotația în jurul axei  $Ox$ ,  $Oy$  și  $Oz$ :

$$R_x(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta & \sin \theta & 0 \\ 0 & -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$R_y(\theta) = \begin{pmatrix} \cos \theta & 0 & -\sin \theta & 0 \\ 0 & 1 & 0 & 0 \\ \sin \theta & 0 & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$R_z(\theta) = \begin{pmatrix} \cos \theta & \sin \theta & 0 & 0 \\ -\sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Forfecarea în planul  $Oxy$ ,  $Oyz$  și  $Ozx$ :

$$SH_{xy}(sh_x, sh_y) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ sh_x & sh_y & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$SH_{yz}(sh_y, sh_z) = \begin{pmatrix} 1 & sh_y & sh_z & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$SH_{zx}(sh_z, sh_x) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ sh_x & 1 & sh_z & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Matricea unei transformări geometrice afine complexe formată din  $n$  transformări elementare efectuate în ordinea  $M_1, M_2, \dots, M_n$  se obține prin:

$$M = M_n \cdot \dots \cdot M_2 \cdot M_1$$



Cunoscând coordonatele punctului inițial  $P_1(x_1, y_1, z_1)$  și formele particulare ale matricelor de transformare  $M_1, M_2, \dots, M_n$ , coordonatele punctului final  $P_2(x_2, y_2, z_2)$  se obțin prin parcurgerea următoarelor etape:

- Definirea transformărilor afine elementare  $T_1, T_2, \dots, T_n$  pentru fiecare matrice de transformare, se face cu instrucțiunile, [18]:

```
T1=maketform('affine',M1);
T2=maketform('affine',M2);
...
Tn=maketform('affine',Mn);
```

- Definirea transformării geometrice afine compuse, formată din cele  $n$  transformări elementare aplicate în ordinea dorită  $T_1, T_2, \dots, T_n$ , se face cu instrucțiunea:

```
T=maketform('composite',[Tn...T2 T1]);
```

- Determinarea coordonatelor punctului final obținut în urma aplicării celor  $n$  transformări elementare se face cu instrucțiunea, [19]:

```
[x2 y2 z2]=tformfwd([x1 y1 z1],T);
```

### Problema 8.6

Se consideră profilul aerodinamic de tip Gö 624 definit prin coordonatele intradosului  $y^-(x)$  și extradodusului  $y^+(x)$ , conform tabelului de valori, [20]:

$x$	0,0	1,25	2,5	5,0	7,5	10	15	20	30	40	50	60	70	80	90	95	100
$y^-$	4,0	2,25	1,65	0,95	0,6	0,4	0,15	0,05	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
$y^+$	4,0	7,15	8,5	10,4	11,75	12,85	14,35	15,3	16,0	15,4	14,05	12,0	9,5	6,6	3,55	2,0	0,5

Folosind transformările geometrice afine față de centrul de greutate al profilului, să se reprezinte grafic profilul aerodinamic scalat omogen cu factorul de scalare  $s_x=s_y=0,75$  și rotit cu unghiul  $\theta=45^\circ$  în sens trigonometric, cunoscând coordonatele centrului de greutate al profilului  $x_g=41,7679$  mm și  $y_g=6,51102$  mm.

### Rezolvare

Obținerea profilului final se realizează prin aplicarea următoarelor transformări geometrice afine:

- Transformarea  $T_1$ , translația profilului astfel încât centrul său de greutate să coincidă cu originea sistemului de coordonate. Matricea transformării geometrice este:

$$M_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -x_g & -y_g & 1 \end{pmatrix}$$

- Transformarea  $T_2$ , scalarea profilului față de originea sistemului de coordonate. Matricea transformării geometrice este:

$$M_2 = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

- Transformarea  $T_3$ , rotația profilului față de originea sistemului de coordonate. Matricea transformării geometrice este:

$$M_3 = \begin{pmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

- Transformarea  $T_4$ , translația profilului în poziția inițială. Matricea transformării geometrice este:

$$M_4 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ x_g & y_g & 1 \end{pmatrix}$$

Pentru rezolvarea problemei se definește un fișier de tip script conținând următoarele instrucțiuni principale:

```

%% TRANSFORMARI AFINE ALE PROFILULUI GOTTINGEN 624
clear all;close all;clc;
%% DATE DE INTRARE
% Abscisa profilului, x[mm]
x=[0 1.25 2.5 5 7.5 10 15 20:10:90 95 100];
% Extradosul profilului, ye [mm]
ye=[4 7.15 8.5 10.4 11.75 12.85 14.35 15.3 16 15.4 14.05 12
9.5 6.6 3.55 2 0.5];
% Intradosul profilului, yi [mm]
yi=[4 2.25 1.65 0.95 0.6 0.4 0.15 0.05 0 0 0 0 0 0 0 0];
% Coordonatele centrului de greutate xg si yg [mm]
xg=41.7679;yg=6.51102;
% Factorul de scalare omogena
s=0.75;
% Unghiul de rotatie al profilului
t=pi/4;
%% CALCULE PRELIMINARE
% Definirea absciselor si ordonatelor profilului
x1=[x fliplr(x)]';y1=[yi fliplr(ye)]';
%% REPREZENTAREA GRAFICA A PROFILULUI INITIAL
% Reprezentarea profilului
figure
plot(x1,y1,'--k');hold on;
%% DEFINIREA MATRICELOR DE TRANSFORMARE
% Matricea de translatie in originea sistemului de coordonate
M1=[1 0 0;0 1 0;-xg -yg 1];
% Matricea de scalare a profilului
M2=[s 0 0;0 s 0;0 0 1];
% Matricea de rotatie
M3=[cos(t) sin(t) 0;-sin(t) cos(t) 0;0 0 1];

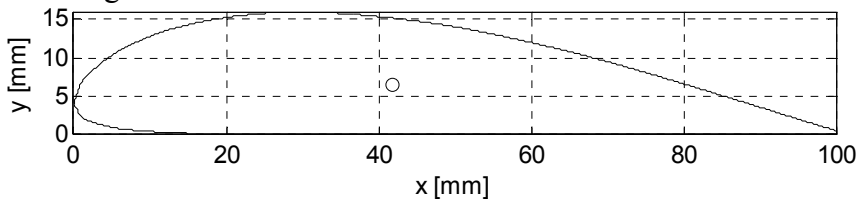
```

```

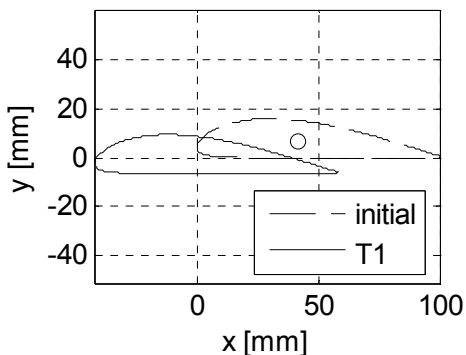
% Matricea de translatie in centrul de greutate
M4=[1 0 0;0 1 0;0 0 1];
%% DEFINIREA TRANSFORMARII GEOMETRICE AFINE COMPUSE
% Translatia profilului in origine
T1=maketform('affine',M1);
% Scalarea profilului
T2=maketform('affine',M2);
% Rotirea profilului
T3=maketform('affine',M3);
% Translatia profilului in pozitia initiala
T4=maketform('affine',M4);
% Matricea de transformare geometrica afina compusa
T=maketform('composite',[T4 T3 T2 T1]);
%% CALCULUL COORDONATELOR PROFILULUI DUPA TRANSFORMARE
[x2 y2]=tformfwd([x1 y1],T);
%% REPREZENTAREA GRAFICA A PROFILULUI TRANSFORMAT
plot(x2,y2,'-k');
% Reprezentarea centrului de greutate al profilului
plot(xg,yg,'ok');
% Definirea legendei
legend('initial','T1 T2 T3 T4');
% Etichetarea axelor
xlabel('x [mm]');ylabel('y [mm]');
% Formatarea axelor
grid on;box on;axis equal;
% Definirea dimensiunii ferestrei grafice
set(gcf,'Position',[25 25 300 200]);

```

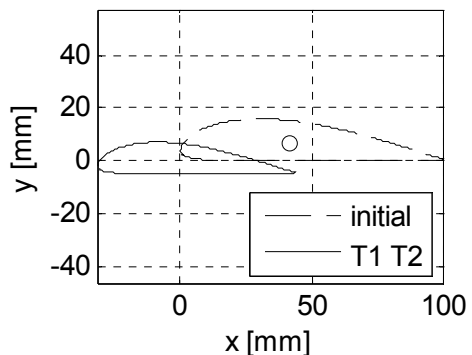
Lansarea în execuție a fișierului script conduce la reprezentările grafice din figura 8.4.



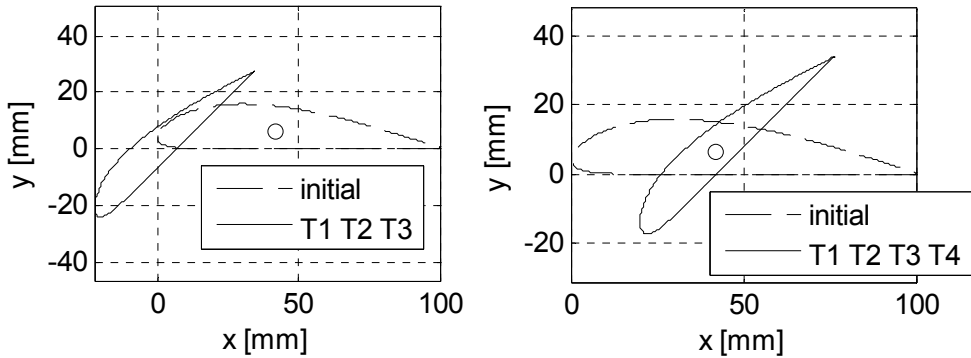
a) profil inițial



b) translație în origine



c) scalare omogenă față de origine



d) rotație față de axa  $Oz$

e) translație în poziția inițială

**Figura 8.4.** Transformările geometrice afine ale profilului aerodinamic.

### Observații

- În figura 8.4, a) se prezintă profilul aerodinamic Gö 624 inițial, înainte de aplicarea transformărilor geometrice.
- În celelalte figuri se prezintă fiecare etapă a transformării profilului: translația în origine a profilului inițial (figura 8.4, b); scalarea omogenă a profilului translat față de originea sistemului de coordonate (figura 8.4, c); rotația profilului scalat față de axa  $Oz$  (figura 8.4, d); translația profilului scalar și rotit, înapoi în coordonatele centrului său de greutate (figura 8.4, e).
- Definirea vectorilor imaginii inițiale, conținând valorile abscisei și ordonatei profilului, se face cu instrucțiunile:  $x1=[x \text{ flipplr}(x)]'$  și  $y1=[y_i \text{ flipplr}(y_e)]'$ . Folosirea instrucțiunilor  $\text{flipplr}(x)$  și  $\text{flipplr}(y_e)$  este necesară pentru a asigura reprezentarea grafică corectă a profilului aerodinamic, în următoarea ordine: mai întâi se reprezintă intradosul pornind de la bordul de atac până la bordul de fugă, după care urmează extradosul pornind de la bordul de fugă până la bordul de atac. În caz contrar, pe lângă curbele intradosului și extradosului s-ar obține și o linie nedorită, din bordul de atac în bordul de fugă al profilului aerodinamic.
- Pentru reprezentarea grafică comparativă a profilului original, a imaginii sale transformate, precum și a centrului de greutate al profilului se folosește o structură de forma:

```
figure
plot(x1,y1,'--k');hold on;
plot(x2,y2,'-k');plot(xg,yg,'ok');hold off;
```

- Pentru identificarea corectă a celor două profile se definește o legendă, prin utilizarea instrucțiunii `legend('initial','T1 T2 T3 T4')` (acest caz corespunde figurii 8.4, e).

## BIBLIOGRAFIE

1. Andrei Șt., Proiectarea unui ventilator centrifugal, Proiect de Diplomă, coordonator științific: Zahariea D., Universitatea Tehnică „Gh. Asachi”, Iași, 2005.
2. MathWorks, Create Array of All Zeros, <http://www.mathworks.com/help/matlab/ref/zeros.html>, accesat la 12.02.2014.
3. MathWorks, Create Array of All Ones, <http://www.mathworks.com/help/matlab/ref/ones.html>, accesat la 12.02.2014.
4. MathWorks, Identity Matrix, <http://www.mathworks.com/help/matlab/ref/eye.html>, accesat la 12.02.2014.
5. MathWorks, Diagonal Matrices and Diagonals of Matrix, <http://www.mathworks.com/help/matlab/ref/diag.html>, accesat la 12.02.2014.
6. MathWorks, Uniformly Distributed Pseudorandom Numbers, <http://www.mathworks.com/help/matlab/ref/rand.html>, accesat la 12.02.2014.
7. MathWorks, Normally Distributed Pseudorandom Numbers, <http://www.mathworks.com/help/matlab/ref/randn.html>, accesat la 12.02.2014.
8. MathWorks, Rectangular Grid in 2-D and 3-D Space, <http://www.mathworks.com/help/matlab/ref/meshgrid.html>, accesat la 14.02.2014.
9. MathWorks, Matrix Determinant, <http://www.mathworks.com/help/matlab/ref/det.html>, accesat la 12.02.2014.
10. MathWorks, Matrix Inverse, <http://www.mathworks.com/help/matlab/ref/inv.html>, accesat la 12.02.2014.
11. MathWorks, Sum of Diagonal Elements, <http://www.mathworks.com/help/matlab/ref/trace.html>, accesat la 12.02.2014.
12. MathWorks, Rank of Matrix, <http://www.mathworks.com/help/matlab/ref/rank.html>, accesat la 12.02.2014.
13. MathWorks, Moore-Penrose Pseudoinverse of Matrix, <http://www.mathworks.com/help/matlab/ref/pinv.html>, accesat la 12.02.2014.
14. MathWorks, Reduced Row Echelon Form, <http://www.mathworks.com/help/matlab/ref/rref.html>, accesat la 12.02.2104.
15. MathWorks, Solve Linear System of Equations, <http://www.mathworks.com/help/matlab/ref/linsolve.html>, accesat la 12.02.2014.
16. MathWorks, Condition Number With Respect to Inversion, <http://www.mathworks.com/help/matlab/ref/cond.html>, accesat la 12.02.2014.
17. MathWorks, Image Processing Toolbox, Users's Guide, 2014.
18. MathWorks, Create Spatial Transformation Structure, <http://www.mathworks.com/help/images/ref/maketform.html>, accesat la 17.03.2014.
19. MathWorks, Apply Forward Spatial Transformation, <http://www.mathworks.com/help/images/ref/tformfwd.html>, accesat la 17.03.2014.
20. Zidaru Gh., Mișcări potențiale și hidrodinamica rețelelor de profile, E.D.P., București, 1981.
21. Zaharia C., Proiectarea sistemului de monitorizare a parametrilor atmosferici pentru instalații de utilizare a energiei eoliene, Proiect de Diplomă, coordonator: Zahariea D., Universitatea Tehnică „Gh. Asachi”, Iași, 2011.

## CAPITOLUL 9

### REPREZENTĂRI GRAFICE 3D

#### 9.1. TIPURI DE REPREZENTĂRI GRAFICE 3D

În limbajul de programare MATLAB sunt implementate proceduri specifice pentru realizarea reprezentărilor grafice 3D pentru următoarele tipuri principale de grafice, [1, 2]:

- Grafice de tip line:
  - `plot3` (reprezentare grafică 3D de tip linie).
  - `contour3` (reprezentare grafică 3D cu linii de contur).
  - `ezplot3` (reprezentare grafică 3D de tip linie pentru o funcție simbolică).
  - `waterfall` (reprezentare grafică 3D de tip waterfall).
  
- Grafice de tip mesh:
  - `mesh` (reprezentare grafică 3D cu suprafețe parametrice de tip wireframe).
  - `meshc` (reprezentare grafică 3D cu suprafețe parametrice de tip wireframe și linii de contur).
  - `meshz` (reprezentare grafică 3D cu suprafețe parametrice de tip wireframe și suport).
  - `ezmesh` (reprezentare grafică 3D cu suprafețe parametrice de tip wireframe pentru o funcție simbolică).
  - `ezmeshc` (reprezentare grafică 3D cu suprafețe parametrice de tip wireframe și linii de contur pentru o funcție simbolică).
  
- Grafice de tip bar:
  - `bar3` (reprezentare grafică 3D cu bare verticale).
  - `bar3h` (reprezentare grafică 3D cu bare orizontale).
  
- Grafice de tip area:
  - `pie3` (reprezentare grafică 3D de tip pie).
  - `fill3` (reprezentare grafică a poligoanelor 3D).

- `patch` (reprezentare grafică a poligoanelor 2D și 3D).
  - `cylinder` (reprezentare grafică a cilindrului).
  - `ellipsoid` (reprezentare grafică a elipsoidului).
  - `sphere` (reprezentare grafică a sferei).
- Grafice de tip `surface`:
    - `surf` (reprezentare grafică 3D cu suprafețe parametrice colorate).
    - `surface` (reprezentare grafică 3D cu suprafețe parametrice colorate).
    - `surf1` (reprezentare grafică 3D cu suprafețe parametrice colorate și iluminare).
    - `surfz` (reprezentare grafică 3D cu suprafețe parametrice colorate și linii de contur).
    - `ribbon` (reprezentare grafică 3D cu suprafețe parametrice colorate de tip `ribbon`).
    - `ezsurf` (reprezentare grafică 3D cu suprafețe parametrice colorate pentru o funcție simbolică).
    - `ezsurfz` (reprezentare grafică 3D cu suprafețe parametrice colorate și linii de contur pentru o funcție simbolică).
- Grafice de tip `direction`:
    - `quiver3` (reprezentare grafică a câmpurilor vectoriale 3D).
    - `comet3` (reprezentare grafică a unei curbe 3D cu animație de tip comet).
- Grafice de tip volumetric:
    - `coneplot` (reprezentare grafică a vectorilor viteză ai unui câmp vectorial 3D sub forma conurilor orientate).
    - `streamline` (reprezentare grafică a liniilor de curent ale câmpurilor vectoriale 2D și 3D).
    - `streamtube` (reprezentare grafică 3D a tuburilor de curent pentru un câmp vectorial 3D).
- Grafice de tip discrete:
    - `stem3` (reprezentare grafică 3D a datelor discrete),
    - `scatter3` (reprezentare grafică 3D de tip `scatter`).

## 9.2. UTILIZAREA INSTRUCȚIUNII `plot3`

Realizarea reprezentărilor grafice ale curbelor 3D prin utilizarea obiectelor grafice de tip linie, pentru funcții matematice exprimate sub formă parametrică pe domenii de definiție oarecare, se obține cu instrucțiunea `plot3`, [3].

Astfel, instrucțiunea:

```
plot3 (fx, fy, fz)
```

realizează graficul 3D al funcției parametrice  $f_x(t)$ ,  $f_y(t)$  și  $f_z(t)$ , pe domeniul de variație al variabilei  $t = [t_{min}, t_{max}]$ . Atunci când variabila independentă parcurge domeniul de variație impus, valorile  $f_x(t)$ ,  $f_y(t)$  și  $f_z(t)$  reprezintă coordonatele spațiale ale punctelor graficului.

### Problema 9.1

Se consideră spirala sferică definită parametric prin relațiile:

$$\begin{cases} x = \frac{\cos t}{\sqrt{1 + a^2 t^2}} \\ y = \frac{\sin t}{\sqrt{1 + a^2 t^2}} \\ z = \frac{-at}{\sqrt{1 + a^2 t^2}} \end{cases}$$

$$a=0,05, t=[-50\pi; 50\pi]$$

Să se realizeze un fișier de tip `script` prin care să se reprezinte grafic spirala sferică utilizând instrucțiunea `plot3`.

### Rezolvare

Rezolvarea problemei este realizată cu următorul fișier `script`:

```
%% GRAFICE 3D (1)
% Instrucțiunea plot3
clear all;close all;clc;
%% DATE DE INTRARE
% Domeniul de definiție
tmin=-50*pi;tmax=50*pi;nt=5000;
t=linspace(tmin,tmax,nt);
% Parametrul caracteristic
a=0.05;
% Definirea funcției
x=cos(t)./sqrt(1+a^2*t.^2);
y=sin(t)./sqrt(1+a^2*t.^2);
z=-a*t./sqrt(1+a^2*t.^2);
%% REPREZENTARE GRAFICA
```

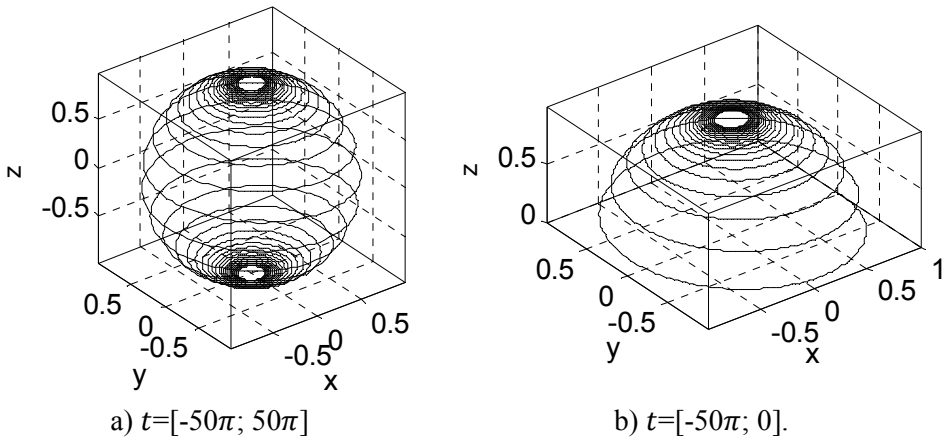


```

figure
plot3(x,y,z,'-k');
% Formatarea graficului
xlabel('x');ylabel('y');zlabel('z');
axis equal;
grid on;
set(gca,'Box','on');

```

Lansarea în execuție a fișierului conduce la reprezentările grafice prezentate în figura 9.1, a) pentru domeniul  $t=[-50\pi; 50\pi]$  și în figura 9.1, b) pentru domeniul  $t=[-50\pi; 0]$ .



**Figura 9.1.** Spirala sferică.

### Observații

- Domeniul de definiție al funcției este specificat prin instrucțiunea  $t=\text{linspace}(t_{\min},t_{\max},n_t)$ , în care valoarea inițială este  $t_{\min}=-50*\pi$ , valoarea finală este  $t_{\max}=50*\pi$ , iar numărul punctelor de calcul este  $n_t=5000$ .
- Calculul coordonatelor spațiale ale spiralei sferice se face cu instrucțiunile  $x=\cos(t)/\sqrt{1+a^2*t.^2}$ ,  $y=\sin(t)/\sqrt{1+a^2*t.^2}$  și  $z=-a*t./\sqrt{1+a^2*t.^2}$ .
- Reprezentarea grafică a curbei spațiale se face cu instrucțiunea  $\text{plot3}(x,y,z,'-k')$ . Parametrii de formatare ai curbei specificați în interiorul instrucțiunii  $\text{plot3d}$  sunt  $'-k'$  și conduc la obținerea unei curbe cu linie continuă de culoare neagră.
- Aplicarea adnotărilor de etichetare a celor trei axe se face cu instrucțiunile  $\text{xlabel}('x')$ ,  $\text{ylabel}('y')$  și  $\text{zlabel}('z')$ . Șirurile de caractere care vor constitui etichetele celor trei axe și care reprezintă argumentele instrucțiunilor  $\text{xlabel}$ ,  $\text{ylabel}$  și  $\text{zlabel}$  trebuie introduse între apostrofuri.

### 9.3. UTILIZAREA INSTRUCȚIUNII `mesh`

Realizarea reprezentărilor grafice 3D prin utilizarea obiectelor grafice de tip `wireframe`, pentru funcții matematice exprimate sub formă explicită sau parametrică pe domenii de definiție oarecare, se obține cu instrucțiunea `mesh`, [4].

Astfel, pentru reprezentarea grafică a funcției exprimate în mod explicit prin relația:

$$z = f(x, y)$$

pe domeniul oarecare:

$$x \times y = [x_{min}, x_{max}] \times [y_{min}, y_{max}]$$

se utilizează instrucțiunea:

```
mesh (X, Y, Z)
```

în care `X` și `Y` reprezintă matricele absciselor și ordonatelor punctelor de discretizare ale domeniului plan de definiție, determinate cu instrucțiunea:

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

iar `Z` reprezintă matricea valorilor funcției determinată cu instrucțiunea:

```
Z = f(X, Y)
```

Pentru realizarea reprezentării grafice în cazul funcțiilor exprimate prin ecuațiile parametrice:

$$f_x(u, v), f_y(u, v) \text{ și } f_z(u, v)$$

definite pe un domeniu oarecare:

$$u \times v = [u_{min}, u_{max}] \times [v_{min}, v_{max}]$$

se utilizează instrucțiunea:

```
mesh (X, Y, Z)
```

în care matricele coordonatelor `X`, `Y` și `Z` se calculează conform instrucțiunilor:

```
X = fx(U, V)
```

```
Y = fy(U, V)
```

```
Z = fz(U, V)
```

iar matricele parametrilor `U` și `V` se determină cu instrucțiunea:

```
[U, V] = meshgrid(u, v)
```

Realizarea reprezentărilor grafice 3D prin utilizarea combinată a obiectelor grafice de tip `wireframe` și a liniilor de contur, pentru funcții matematice exprimate sub formă explicită sau parametrică pe domenii de definiție oarecare, se obține cu instrucțiunea `meshc` [5], conform sintaxei:

```
meshc (X, Y, Z)
```

Realizarea reprezentărilor grafice 3D prin utilizarea combinată a obiectelor grafice de tip `wireframe` și a unui suport plasat la partea inferioară a suprafeței, pentru funcții matematice exprimate sub formă explicită sau parametrică pe domenii de definiție oarecare, se obține cu instrucțiunea `meshz` [6], conform sintaxei:

```
meshz (X, Y, Z)
```

### Problema 9.2

Se consideră paraboloidul hiperbolic definit prin relațiile:

$$z(x, y) = \frac{y^2}{b^2} - \frac{x^2}{a^2}$$
$$a = 1, b = 1,$$
$$x \times y = [-\pi; \pi] \times [-\pi; \pi].$$

Să se realizeze un fișier de tip `script` prin care să se reprezinte grafic paraboloidul hiperbolic utilizând instrucțiunea `mesh`.

### Rezolvare

Rezolvarea problemei este realizată cu următorul fișier `script`:

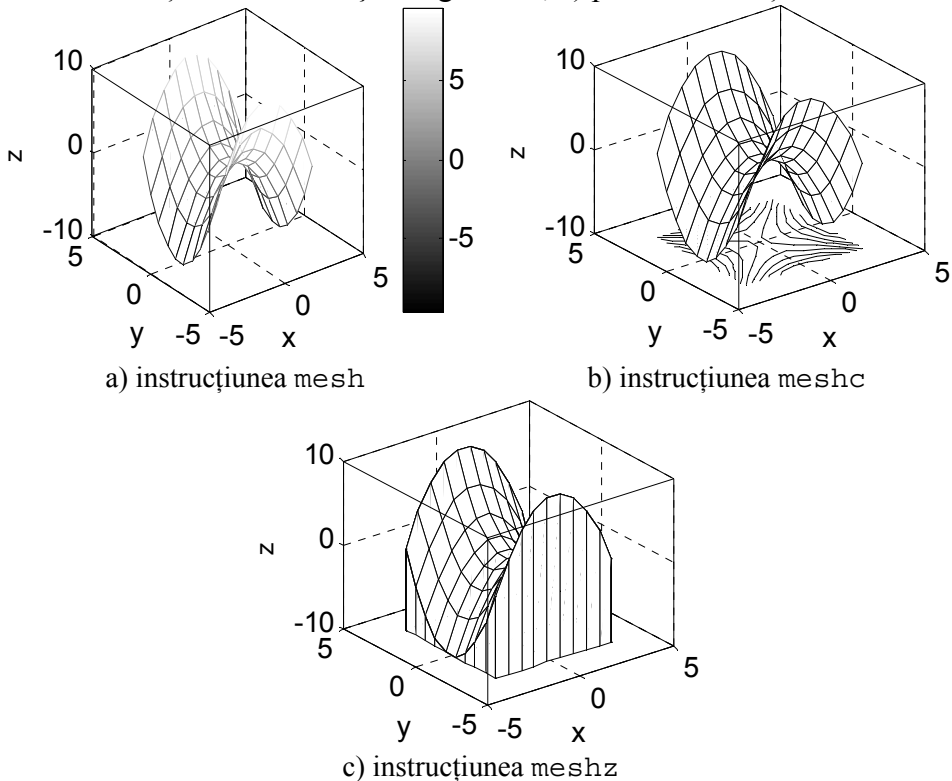
```
%% GRAFICE 3D (2)
% Instrucțiunea mesh
clear all;close all;clc;
%% DATE DE INTRARE
% Domeniul vectorial
xmin=-pi;xmax=pi;nx=25;x=linspace(xmin,xmax,nx);
ymin=-pi;ymax=pi;ny=25;y=linspace(ymin,ymax,ny);
% Domeniul matriceal
[X,Y]=meshgrid(x,y);
% Parametrii caracteristici
a=1;b=1;
% Definirea funcției
Z=Y.^2/b^2-X.^2/a^2;
%% REPREZENTARE GRAFICA
% Instrucțiunea mesh
figure
mesh(X,Y,Z);box on;colormap gray;colorbar;
xlabel('x');ylabel('y');zlabel('z');
% Instrucțiunea meshc
figure
meshc(X,Y,Z);box on;colormap([0 0 0]);
xlabel('x');ylabel('y');zlabel('z');
% Instrucțiunea meshz
```

```

figure
meshz(X,Y,Z);xlabel('x');ylabel('y');zlabel('z');
box on;colormap([0 0 0]);

```

Lansarea în execuție a fișierului conduce la reprezentările grafice prezentate în figura 9.2, a) pentru instrucțiunea mesh, în figura 9.2, b) pentru instrucțiunea meshc și în figura 9.2, c) pentru instrucțiunea meshz.



**Figura 9.2.** Paraboloidul hiperbolic.

### Observații

- Atribuirea unei anumite mape de culoare reprezentării grafice 3D se face cu instrucțiunea `colormap(map)` [7], în care `map` reprezintă una din mapele de culoare predefinite: `jet`, `hsv`, `hot`, `cool`, `spring`, `summer`, `autumn`, `winter`, `gray`, `bone`, `cooper`, `pink`, `lines`. Se pot utiliza și mape de culoare definite de utilizator prin orice combinație de forma `[r g b]`, în care `r`, `g` și `b` reprezintă nuanțe ale culorilor roșu, verde și albastru. Mapa de culoare `colormap([0 0 0])` corespunde culorii negru.
- Afișarea scalei culorilor pentru a permite identificarea corespondenței dintre valorile numerice și culorile asociate se face cu instrucțiunea `colorbar` [8], figura 9.2, a).

#### 9.4. UTILIZAREA INSTRUCȚIUNII `surf`

Realizarea reprezentărilor grafice 3D prin utilizarea obiectelor grafice de tip `surface`, pentru funcții matematice exprimate sub formă explicită sau parametrică pe domenii de definiție oarecare, se obține cu instrucțiunea `surf`, [9].

Astfel, pentru reprezentarea grafică a funcției exprimate în mod explicit prin relația:

$$z = f(x, y)$$

pe domeniul oarecare:

$$x \times y = [x_{min}, x_{max}] \times [y_{min}, y_{max}]$$

se utilizează instrucțiunea:

```
surf (X, Y, Z)
```

în care `X` și `Y` reprezintă matricele absciselor și ordonatelor punctelor de discretizare ale domeniului plan de definiție, determinate cu instrucțiunea:

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

iar `Z` reprezintă matricea valorilor funcției determinată cu instrucțiunea:

```
Z = f (X, Y)
```

Pentru realizarea reprezentării grafice în cazul funcțiilor exprimate prin ecuațiile parametrice:

$$f_x(u, v), f_y(u, v) \text{ și } f_z(u, v)$$

definite pe un domeniu oarecare:

$$u \times v = [u_{min}, u_{max}] \times [v_{min}, v_{max}]$$

se utilizează instrucțiunea:

```
surf (X, Y, Z)
```

în care matricele coordonatelor `X`, `Y` și `Z` se calculează conform instrucțiunilor:

$$X = f_x (U, V) ; Y = f_y (U, V) ; Z = f_z (U, V)$$

iar matricele parametrilor `U` și `V` se determină cu instrucțiunea:

```
[U, V] = meshgrid(u, v)
```

Realizarea reprezentărilor grafice 3D prin utilizarea combinată a obiectelor grafice de tip `surface` și a liniilor de contur, pentru funcții matematice exprimate sub formă explicită sau parametrică pe domenii de definiție oarecare, se obține cu instrucțiunea `surf c` [10], conform sintaxei:

```
surf c (X, Y, Z)
```

### Problema 9.3

Se consideră torul de tip inel circular definit prin relațiile:

$$\begin{cases} x = (R_1 + R_2 \cos v) \cos u \\ y = (R_1 + R_2 \cos v) \sin u \\ z = R_2 \sin v \end{cases}$$
$$u \times v = [0; 2\pi] \times [0; 2\pi]$$
$$R_1 = 2; R_2 = 0,75$$

Să se realizeze un fișier de tip script prin care să se reprezinte grafic torul utilizând instrucțiunea surf.

### Rezolvare

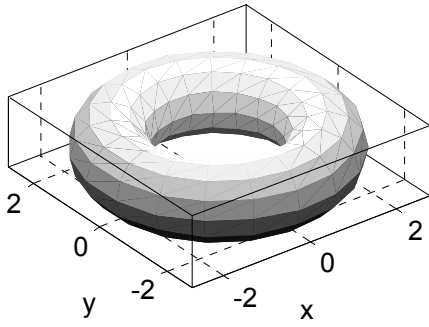
Rezolvarea problemei este realizată cu următorul fișier script:

```
%% GRAFICE 3D (2)
% Instrucțiunea surf
clear all;close all;clc;
%% DATE DE INTRARE
% Domeniul vectorial
umin=0;umax=2*pi;nu=25;
u=linspace(umin,umax,nu);
vmin=0;vmax=2*pi;nv=25;
v=linspace(vmin,vmax,nv);
% Domeniul matriceal
[U,V]=meshgrid(u,v);
% Parametrii caracteristici
R1=2;R2=0.75;
% Definirea funcției
X=(R1+R2*cos(V)).*cos(U);
Y=(R1+R2*cos(V)).*sin(U);
Z=R2*sin(V);
%% REPREZENTARE GRAFICA
surf(X,Y,Z);
xlabel('x');ylabel('y');zlabel('z');
box on;axis equal;
colormap(gray);shading interp;
```

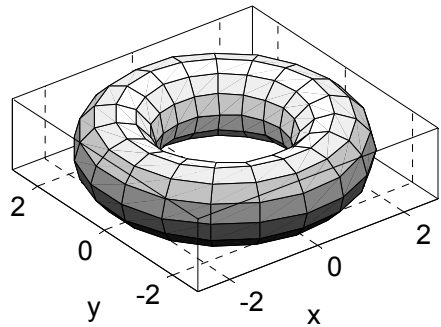
Lansarea în execuție a fișierului conduce la reprezentările grafice prezentate în figura 9.3.

### Observații

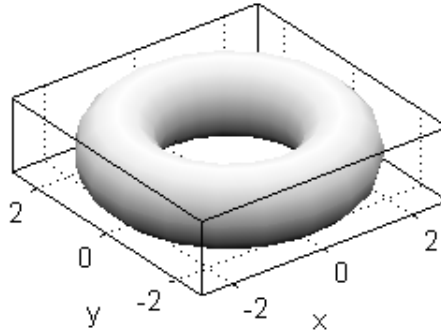
- Modificarea proprietății de umbrire a suprafețelor se realizează cu ajutorul instrucțiunii shading, [11]. Parametrii instrucțiunii shading pot fi: flat (figura 9.3, a), faceted (figura 9.3, b) sau interp (figura 9.3, c).



a) parametrul shading flat



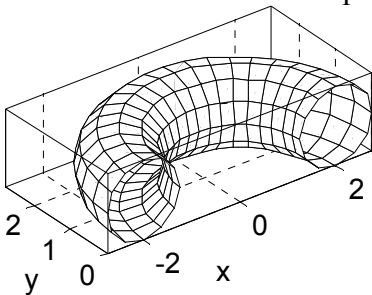
b) parametrul shading faceted



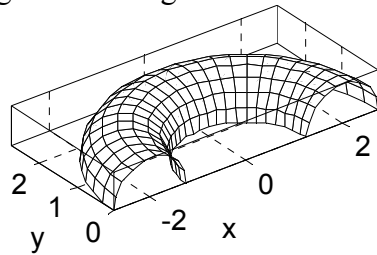
c) parametrul shading interp

**Figura 9.3.** Reprezentarea grafică a torului.

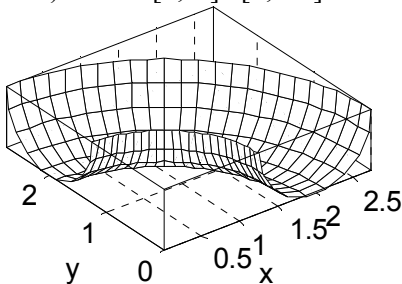
- Modificarea domeniului vectorial de definiere a ecuațiilor parametrice ale torului conduce la reprezentările grafice din figura 9.4.



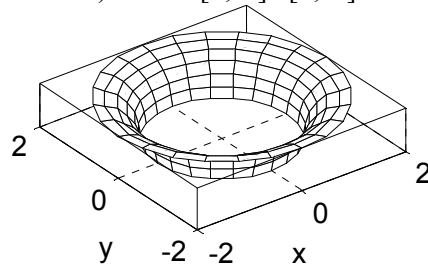
a)  $u \times v = [0; \pi] \times [0; 2\pi]$



b)  $u \times v = [0; \pi] \times [0; \pi]$



c)  $u \times v = [0; \pi/2] \times [0; 2\pi]$



d)  $u \times v = [0; 2\pi] \times [\pi/2; \pi]$

**Figura 9.4.** Reprezentarea grafică a unor porțiuni ale torului.

## 9.5. UTILIZAREA INSTRUCȚIUNII `contour`

Realizarea reprezentărilor grafice 2D prin utilizarea liniilor de contur, pentru funcții matematice exprimate sub formă explicită sau parametrică pe domenii de definiție oarecare, se obține cu instrucțiunea `contour`, [12].

Astfel, pentru reprezentarea grafică a funcției exprimate în mod explicit prin relația:

$$z = f(x, y)$$

pe domeniul oarecare:

$$x \times y = [x_{min}, x_{max}] \times [y_{min}, y_{max}]$$

se utilizează instrucțiunea:

$$C = \text{contour}(X, Y, Z)$$

în care  $X$  și  $Y$  reprezintă matricele absciselor și ordonatelor punctelor de discretizare ale domeniului plan de definiție, determinate cu instrucțiunea:

$$[X, Y] = \text{meshgrid}(x, y)$$

iar  $Z$  reprezintă matricea valorilor funcției determinată cu instrucțiunea:

$$Z = f(X, Y)$$

Pentru realizarea reprezentării grafice în cazul funcțiilor exprimate prin ecuațiile parametrice:

$$f_x(u, v), f_y(u, v) \text{ și } f_z(u, v)$$

definite pe un domeniu oarecare:

$$u \times v = [u_{min}, u_{max}] \times [v_{min}, v_{max}]$$

se utilizează instrucțiunea:

$$C = \text{contour}(X, Y, Z)$$

în care matricele coordonatelor  $X$ ,  $Y$  și  $Z$  se calculează conform instrucțiunilor:

$$X = f_x(U, V) ; Y = f_y(U, V) ; Z = f_z(U, V) ;$$

iar matricele parametrilor  $U$  și  $V$  se determină cu instrucțiunea:

$$[U, V] = \text{meshgrid}(u, v)$$

Realizarea reprezentărilor grafice 2D prin utilizarea combinată a liniilor de contur și a colorării spațiului dintre liniile de contur (folosind culorile specificate în mapa de culoare definită prin instrucțiunea `colormap`), se obține cu instrucțiunea `contourf`, conform sintaxei:



```
contourf (X, Y, Z)
```

Realizarea reprezentărilor grafice 3D prin utilizarea liniilor de contur, pentru funcții matematice exprimate sub formă explicită sau parametrică pe domenii de definiție oarecare, se obține cu instrucțiunea `contour3`, conform sintaxei:

```
C=contour3 (X, Y, Z)
```

Numărul liniilor de contur este identificat automat în funcție de domeniul de valori ale variabilei Z. Controlul numărului liniilor de contur poate fi realizat cu ajutorul instrucțiunilor:

```
C=contour (X, Y, Z, n)
```

```
C=contour (X, Y, Z, v)
```

în care  $n$  reprezintă numărul liniilor de contur, iar  $v = [v_1 \ v_2 \dots v_m]$  este un vector cu valorile variabilei Z prin care se vor trasa linii de contur.

Etichetarea liniilor de contur se poate face prin următoarele două metode:

```
C=contour (X, Y, Z) ;
```

```
clabel (C) ;
```

sau:

```
contour (X, Y, Z, 'ShowText', 'on') ;
```

#### Problema 9.4

Se consideră două variabile cantitative continue normal distribuite  $x$  și  $y$  având mediile aritmetice  $\mu_x$  și  $\mu_y$  și abaterile medii pătratice  $\sigma_x$  și  $\sigma_y$ .

În ipoteza că cele două variabile  $x$  și  $y$  sunt necorelate, distribuția normală bivariată a celor două variabile se exprimă prin relația:

$$P(x, y) = \frac{1}{2\pi\sigma_x\sigma_y} \cdot e^{-\frac{(x-\mu_x)^2}{2\sigma_x^2} - \frac{(y-\mu_y)^2}{2\sigma_y^2}}$$

Să se reprezinte grafic funcția  $P(x, y)$  folosind metoda liniilor de contur pentru cazul:  $\mu_x=1$ ;  $\mu_y=2$ ;  $\sigma_x=4$ ;  $\sigma_y=1$ ;  $u \times v = [-11; 13] \times [-1; 5]$ .

#### Rezolvare

Rezolvarea problemei este realizată cu următorul fișier script:

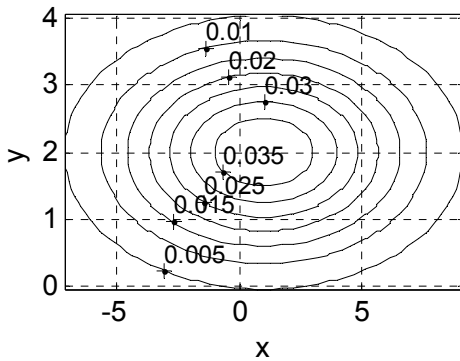
```
% GRAFICE 3D (4)
% Instrucțiunea contour
clear all;close all;clc;
```

```

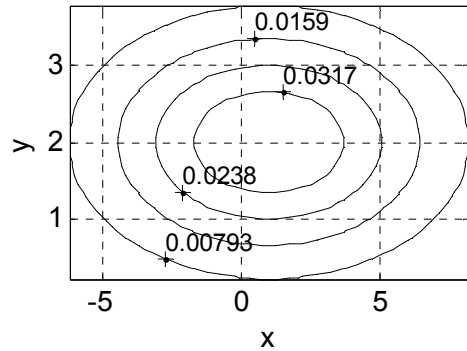
%% DATE DE INTRARE
% Domeniul vectorial
xmin=-11;xmax=13;nx=50;
x=linspace(xmin,xmax,nx);
ymin=-1;ymax=5;ny=50;
y=linspace(ymin,ymax,ny);
% Domeniul matriceal
[X,Y]=meshgrid(x,y);
% Definirea functiei
mx=1;my=2;sx=4;sy=1;
Z=1/(2*pi*sx*sy)*exp(-(X-mx).^2/(2*sx^2)-(Y-my).^2/(2*sy^2));
%% REPREZENTARE GRAFICA
figure
C=contour(X,Y,Z);grid on;box on;colormap([0 0 0]);
xlabel('x');ylabel('y');clabel(C);axis tight;

```

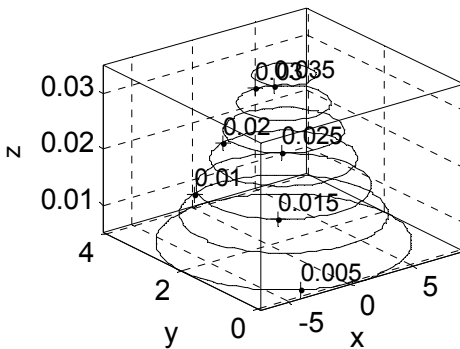
Lansarea în execuție a fișierului conduce la reprezentările grafice prezentate în figura 9.5.



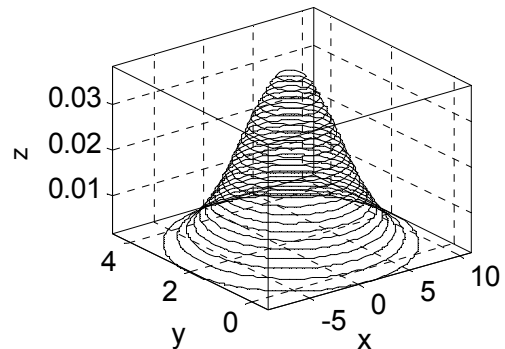
a) contour cu număr implicit de linii de contur



b) contour cu 4 linii de contur



c) contour3 cu număr implicit de linii de contur



d) contour3 cu 25 linii de contur fără etichete

**Figura 9.5.** Distribuția normală bivariată.

## 9.6. SPECTRUL HIDRODINAMIC AL MIȘCĂRILOR POTENȚIALE

Se consideră potențialul complex definit prin:

$$f(x, y) = \varphi(x, y) + i \cdot \psi(x, y)$$

pe un domeniu  $x \times y = [x_{min}, x_{max}] \times [y_{min}, y_{max}]$ .

Pentru reprezentarea grafică a spectrului hidrodinamic al mișcării potențiale trebuie parcurse următoarele etape:

- Cunoscând limitele domeniului de mișcare se determină vectorii domeniului vectorial al potențialului complex:

```
x=linspace(xmin, xmax, nx) ;
```

```
y=linspace(ymin, ymax, ny) ;
```

- Se determină domeniul matriceal de definiție al potențialului complex:

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

- Se definește variabila complexă:

```
Z=X+i*Y ;
```

- Se definește potențialul complex funcție de problema analizată:

```
F=...
```

- Se separă partea reală și partea imaginară a potențialului complex:

```
PHI=real(F) ;
```

```
PSI=imag(F) ;
```

- Se calculează componentele vitezei  $\vec{V} = u(x, y) \cdot \vec{i} + v(x, y) \cdot \vec{j}$  folosind relația, [13]:

```
[U, V]=gradient(PHI) ;
```

- Se reprezintă grafic spectrul hidrodinamic format din  $n_\varphi$  linii echipotențiale și  $n_\psi$  linii de curent:

```
Cphi=contour(X, Y, PHI, nphi) ;hold on ;grid on ;
```

```
Cpsi=contour(X, Y, PSI, npsi) ;hold off ;
```

- Se reprezintă grafic câmpul vectorial al vitezelor și liniile de curent cunoscând densitatea liniilor de curent  $d_\psi$ :

```
streamslice(X, Y, U, V, dpsi) ;
```

### Problema 9.5

Să se reprezinte grafic spectrul hidrodinamic al mișcării potențiale plane pentru care se cunosc următoarele caracteristici:

- Mișcarea potențială plană este definită prin suprapunerea unui vârtej punctiform plasat în punctul  $z_{\Gamma_1} = x_{\Gamma_1} + i \cdot y_{\Gamma_1}$  având intensitatea  $\Gamma_1$  cu un al doilea vârtej punctiform plasat în punctul  $z_{\Gamma_2} = x_{\Gamma_2} + i \cdot y_{\Gamma_2}$  având intensitatea  $\Gamma_2$ . Potențialul complex al mișcării este, [21]:

$$f(z) = -i \cdot \frac{\Gamma_1}{2\pi} \ln(z - z_{\Gamma_1}) - i \cdot \frac{\Gamma_2}{2\pi} \ln(z - z_{\Gamma_2})$$

$$\Gamma_1 = -1; \Gamma_2 = 1$$

$$z_{\Gamma_1} = 0,0 + i \cdot 0,5; z_{\Gamma_2} = 0,0 - i \cdot 0,5$$

- Domeniul de mișcare  $x \times y = [x_{min}, x_{max}] \times [y_{min}, y_{max}]$  este definit prin:

$$x \times y = [-1; 1] \times [-1; 1]$$

- Spectrul hidrodinamic este caracterizat prin  $n_\varphi$  linii echipotențiale și  $n_\psi$  linii de curent.

$$n_\varphi = 50; n_\psi = 50$$

- Parametrul de densitate al liniilor de curent pentru reprezentarea câmpului vectorial al vitezelor este  $d_\psi$ .

$$d_\psi = 1$$

### Rezolvare

Rezolvarea problemei este realizată cu următorul fișier script:

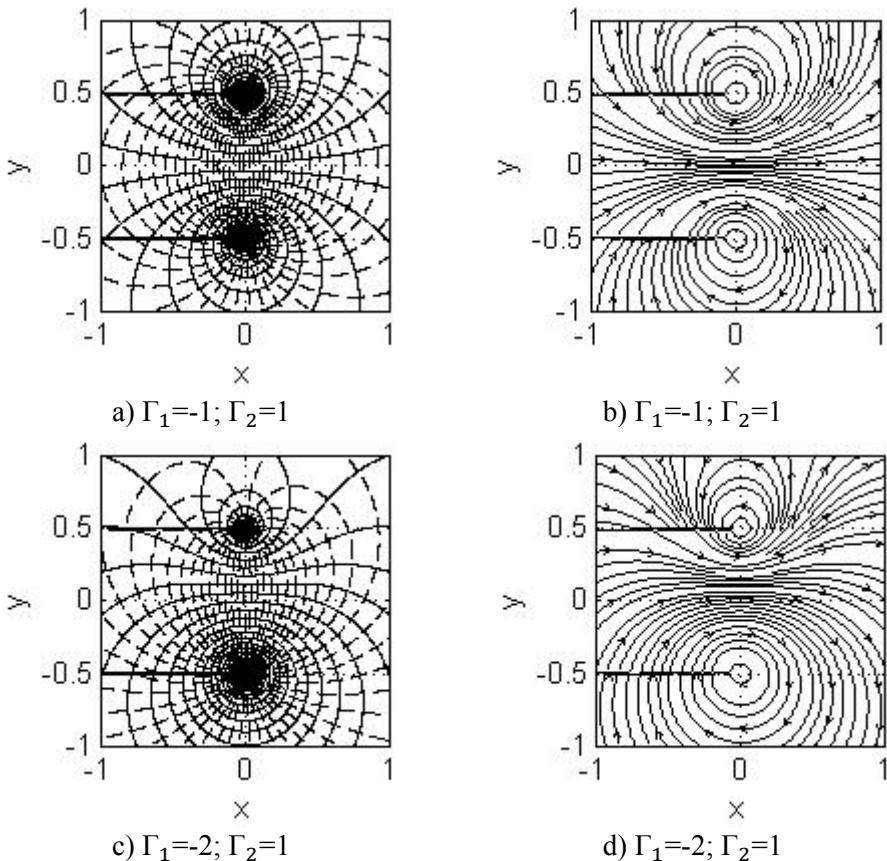
```
%% REPREZENTAREA SPECTRULUI HIDRODINAMIC
% Sistem de 2 vartejuri
clear all;close all;clc;
%% DATE DE INTRARE
% Domeniul vectorial
xmin=-1;xmax=1;nx=150;x=linspace(xmin,xmax,nx);
ymin=-1;ymax=1;ny=150;y=linspace(ymin,ymax,ny);
% Domeniul matriceal
[X,Y]=meshgrid(x,y);
% Definirea variabilei complexe
Z=X+i*Y;
% Definirea potentialului complex
G1=-1;G2=2;
Zg1=-0.0-0.5*i;Zg2=0.0+0.5*i;
F=-i*G1/(2*pi)*log(Z-Zg1)-i*G2/(2*pi)*log(Z-Zg2);
% Extragerea partii reale si a partii imaginare
PHI=real(F);PSI=imag(F);
% Calculul vitezelor
[U,V]=gradient(PHI);
```

```

% SPECTRUL HIDRODINAMIC
% Numarul liniilor echipotentiale
nphi=50;
% Numarul liniilor de curent
npsi=50;
figure
Cphi=contour(X,Y,PHI,nphi,'--k'); hold on;
Cpsi=contour(X,Y,PSI,npsi,'-k');hold off;
grid on;box on;axis image;xlabel('x');ylabel('y');
% LINIILE DE CURENT SI VITEZELE ABSOLUTE
% parametrul de densitate al liniilor de curent
d=2;
figure
[v av]=streamslice(X,Y,U,V,d);
hlines=streamline([v av]);
set(hlines,'Color','k');
grid on;box on;axis image;xlabel('x');ylabel('y');

```

Lansarea în execuție a fișierului conduce la reprezentările grafice prezentate în figura 9.6.



**Figura 9.6.** Spectrul hidrodinamic al unui sistem format din două surse.

## 9.7. UTILIZAREA INSTRUCȚIUNII `quiver`

Se consideră câmpul vectorial de viteze:

$$\vec{V} = u(x, y) \cdot \vec{i} + v(x, y) \cdot \vec{j}$$

definit pe domeniul plan:

$$x \times y = [x_{min}, x_{max}] \times [y_{min}, y_{max}]$$

Instrucțiunea `quiver` [14], realizează reprezentarea grafică, pentru un domeniu de mișcare bidimensional, a vitezelor absolute:

$$V(x, y) = \sqrt{u^2(x, y) + v^2(x, y)}$$

Pentru reprezentarea grafică a vitezelor absolute trebuie parcurse următoarele etape:

- Cunoscând limitele domeniului de mișcare se determină vectorii:

```
x=linspace(xmin, xmax, nx) ;
```

```
y=linspace(ymin, ymax, ny) ;
```

- Se determină domeniul matriceal de discretizare al domeniului vectorial de mișcare:

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

- Se definesc vitezele  $U(X, Y)$  și  $V(X, Y)$  în funcție de problema analizată pornind de la relațiile componentelor vitezei  $u(x, y)$  și  $v(x, y)$ , calculate însă pe domeniul matriceal de discretizare  $[X, Y]$ .

- Se reprezintă grafic vitezele absolute cunoscând factorul de scalare al vectorilor  $f_s$  cu instrucțiunea:

```
quiver(X, Y, U, V, fs)
```

### Problema 9.6

Să se reprezinte grafic vitezele absolute ale mișcării plane definite prin potențialul complex pentru care se cunosc următoarele caracteristici:

- Mișcarea potențială plană este definită prin suprapunerea unei surse pozitive, având debitul  $Q$ , plasată în origine și a unei translații uniforme având viteza  $V_\infty$  orientată în direcția pozitivă a axei  $Ox$ . Potențialul complex al acestei mișcări este definit prin relația, [21]:

$$f(z) = V_\infty z \cdot e^{-i\alpha_\infty} + \frac{Q}{2\pi} \cdot \ln z$$

$$Q=1; V_\infty=1; \alpha_\infty=0$$

- Domeniul de mișcare  $x \times y = [x_{min}, x_{max}] \times [y_{min}, y_{max}]$  este definit prin:

$$x \times y = [-1; 1] \times [-1; 1]$$

- Vectorii viteză sunt caracterizați prin factorul de scalare  $f_s$  având valorile:

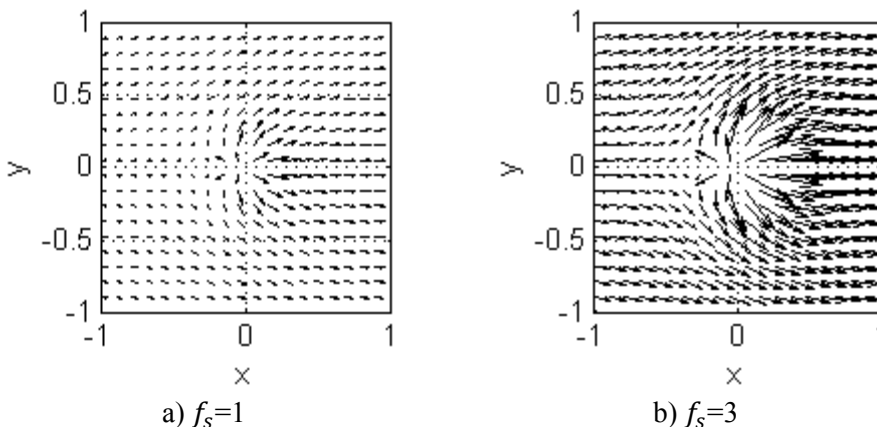
$$f_s=1 \text{ și } f_s=3$$

## Rezolvare

Rezolvarea problemei este realizată cu următorul fișier script:

```
%% DATE DE INTRARE
% Domeniul vectorial
xmin=-1;xmax=1;nx=20;x=linspace(xmin,xmax,nx);
ymin=-1;ymax=1;ny=20;y=linspace(ymin,ymax,ny);
% Domeniul matriceal
[Xq,Yq]=meshgrid(xq,yq);
%% CALCULUL VITEZELOR U si V
% Definirea variabilei complexe
Z=@(X,Y) X+i*Y;
% Definirea potentialului complex
Q=2;Vinf=1;ainf=0;
F=@(X,Y) Vinf*Z(X,Y)*exp(-i*ainf)+Q/(2*pi)*
log(Z(X,Y));
% Extragerea partii reale si a partii imaginare
PHI=@(X,Y) real(F(X,Y));
PSI=@(X,Y) imag(F(X,Y));
% Calculul vitezelor
[U,V]=gradient(PHI(X,Y));
% REPREZENTAREA GRAFICA A VITEZELOR
% Factorul de scalare
fs=1;%pentru cazul 2, fs=3
figure
qlines=quiver(X,Y,U,V,fs);
set(qlines,'Color','k');
grid on;box on;xlabel('x');ylabel('y');
axis equal;axis([xmin xmax ymin ymax]);
```

Lansarea în execuție a fișierului conduce la reprezentările grafice prezentate în figura 9.7.



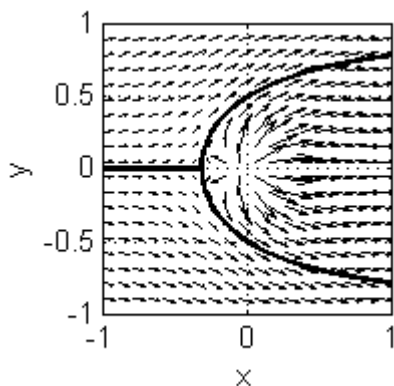
**Figura 9.7.** Sursă în prezența unui curent uniform, vectorii viteze.

În figura 9.8, a) se prezintă suprapunerea peste reprezentarea grafică a vectorilor viteză (instrucțiunea `quiver`), a curbei reprezentând linia de curent care materializează suprafața unei carene, obținută cu ajutorul instrucțiunii `contour`.

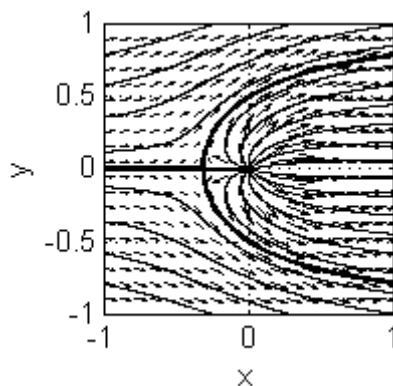
În figura 9.8, b) se reprezintă, în plus, și un număr de  $n_\psi=20$  linii de curent folosind instrucțiunea `contour`. Pentru reprezentările grafice din figura 9.8, a) și figura 9.8, b) s-a utilizat un factor de scalare al vectorilor viteză având valoarea  $f_s=2$ .

În figura 9.8, c) se prezintă spectrul hidrodinamic al mișcării pentru  $n_\phi=30$  linii echipotențiale și  $n_\psi=30$  linii de curent, obținut cu ajutorul instrucțiunii `contour`.

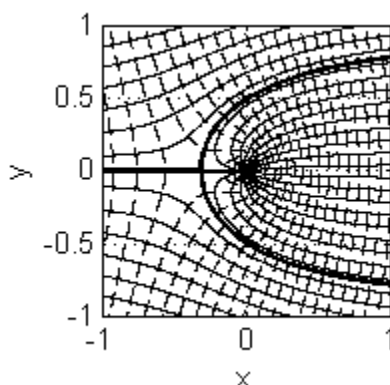
În figura 9.8, d) se prezintă, pentru comparație, liniile de curent și vitezele absolute obținute folosind instrucțiunile `streamslice` și `streamline`. Parametrul de densitate al liniilor de curent pentru reprezentarea câmpului vectorial al vitezelor are valoarea  $d_\psi=1$ .



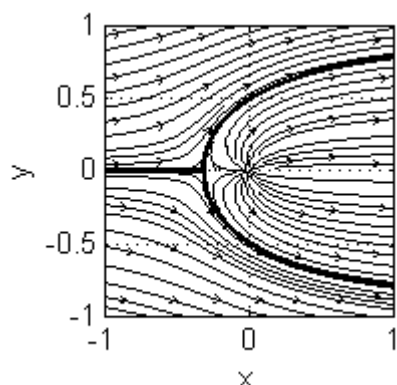
a) carena (`contour`) și vectorii viteză (`quiver`)



b) carena (`contour`), vectorii viteză (`quiver`) și liniile de curent (`contour`)



c) carena (`contour`) și spectrul hidrodinamic (`contour`)



d) carena (`contour`), vectorii viteză și liniile de curent (`streamslice`)

**Figura 9.8.** Sursă în prezența unui curent uniform.



## 9.8. UTILIZAREA INSTRUCȚIUNII `quiver3`

Se consideră câmpul vectorial de viteze:

$$\vec{V} = u(x, y, z) \cdot \vec{i} + v(x, y, z) \cdot \vec{j} + w(x, y, z) \cdot \vec{k}$$

definit pe domeniul tridimensional:

$$x \times y \times z = [x_{min}, x_{max}] \times [y_{min}, y_{max}] \times [z_{min}, z_{max}]$$

Instrucțiunea `quiver3` [15], realizează reprezentarea grafică, pentru un domeniu de mișcare tridimensional, a vitezelor absolute:

$$V(x, y, z) = \sqrt{u^2(x, y, z) + v^2(x, y, z) + w^2(x, y, z)}$$

Pentru reprezentarea grafică a vitezelor absolute trebuie parcurse următoarele etape:

- Cunoscând limitele domeniului de mișcare se determină vectorii:

```
x=linspace(xmin, xmax, nx) ;
```

```
y=linspace(ymin, ymax, ny) ;
```

```
z=linspace(zmin, zmax, nz) ;
```

- Se determină domeniul matriceal de discretizare al domeniului vectorial de mișcare:

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

- Se definesc vitezele  $U(X, Y, Z)$ ,  $V(X, Y, Z)$  și  $W(X, Y, Z)$  în funcție de problema analizată pornind de la relațiile componentelor vitezei  $u(x, y, z)$ ,  $v(x, y, z)$  și  $w(x, y, z)$ , calculate însă pe domeniul matriceal de discretizare  $[X, Y, Z]$ .

- Se reprezintă grafic vitezele absolute cunoscând factorul de scalare al vectorilor  $f_s$  cu instrucțiunea:

```
quiver3(X, Y, Z, U, V, W, fs)
```

### Problema 9.7

Să se reprezinte grafic vitezele absolute și liniile de curent ale mișcării tridimensionale pentru care se cunosc următoarele caracteristici:

- Componentele vitezei:

$$\begin{cases} u(x, y, z) = x^2 y \\ v(x, y, z) = y^2 z \\ w(x, y, z) = -2xyz - z^2 x \end{cases}$$

- Domeniul de mișcare  $x \times y \times z = [x_{min}, x_{max}] \times [y_{min}, y_{max}] \times [z_{min}, z_{max}]$  este definit prin:

$$x \times y \times z = [-1; 1] \times [0; 1] \times [-1; 1]$$

- Vectorii viteze sunt caracterizați prin factorul de scalare  $f_s$  având valoarea:

$$f_s = 5$$

## Rezolvare

Rezolvarea problemei este realizată cu următorul fișier script:

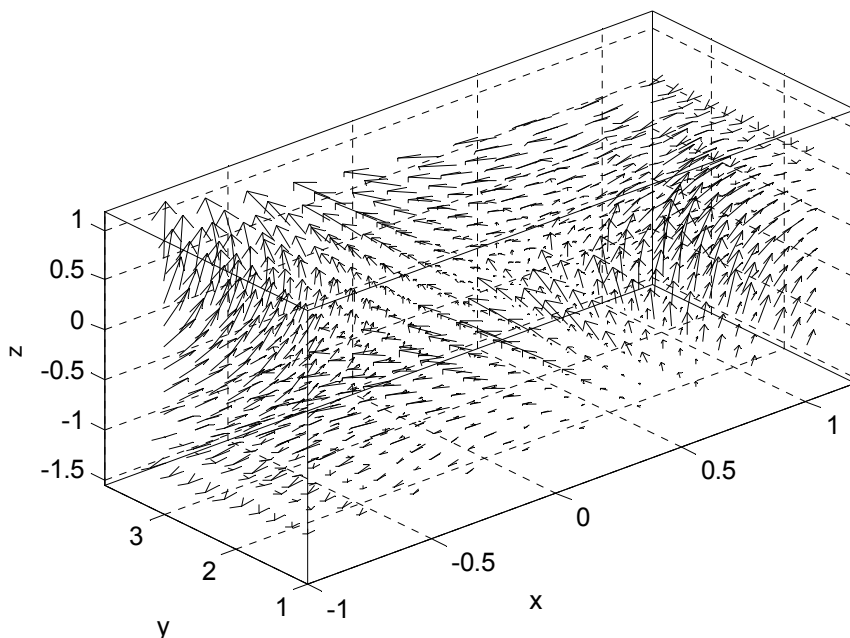
```
%% DATE DE INTRARE
% Domeniul vectorial de miscare
xmin=-1;xmax=1;nx=10;x=linspace(xmin,xmax,nx);
ymin=1;ymax=3;ny=10;y=linspace(ymin,ymax,ny);
zmin=-1;zmax=1;nz=10;z=linspace(zmin,zmax,nz);
% Domeniul matriceal de miscare
[X,Y,Z]=meshgrid(x,y,z);
% Componentele vitezei
U=@(X,Y,Z) X.^2.*Y;V=@(X,Y,Z) Y.^2.*Z;
W=@(X,Y,Z) -Y.*Z.^2-2*X.*Y.*Z;
% Viteza absoluta
V=@(X,Y,Z) sqrt(U(X,Y,Z).^2+V(X,Y,Z).^2+W(X,Y,Z).^2);
%% REPREZENTAREA GRAFICA A VITEZELOR ABSOLUTE
figure
hq=quiver3(X,Y,Z,U(X,Y,Z),V(X,Y,Z),W(X,Y,Z),5);
% Parametrii de formatare
set(hq,'LineWidth',1,'Color','k')
xlabel('x');ylabel('y');zlabel('z');
daspect([1,3,3]);axis tight;box on;grid on;view(-40,25);
%% REPREZENTAREA GRAFICA A LINIILOR DE CURENT
figure
[sx,sy,sz]=meshgrid(xmin:0.1:xmax,1,zmin:0.2:zmax);
hlines=streamline(X,Y,Z,U(X,Y,Z),V(X,Y,Z),W(X,Y,Z),sx,sy,sz);
set(hlines,'LineWidth',1,'Color','k');
% Parametrii de formatare
xlabel('x');ylabel('y');zlabel('z');
daspect([1,3,3]);axis tight;box on;grid on;view(-40,25);
```

Lansarea în execuție a fișierului conduce la reprezentările grafice prezentate în figura 9.9 (pentru cazul vitezelor absolute) și figura 9.10 (pentru cazul liniilor de curent).

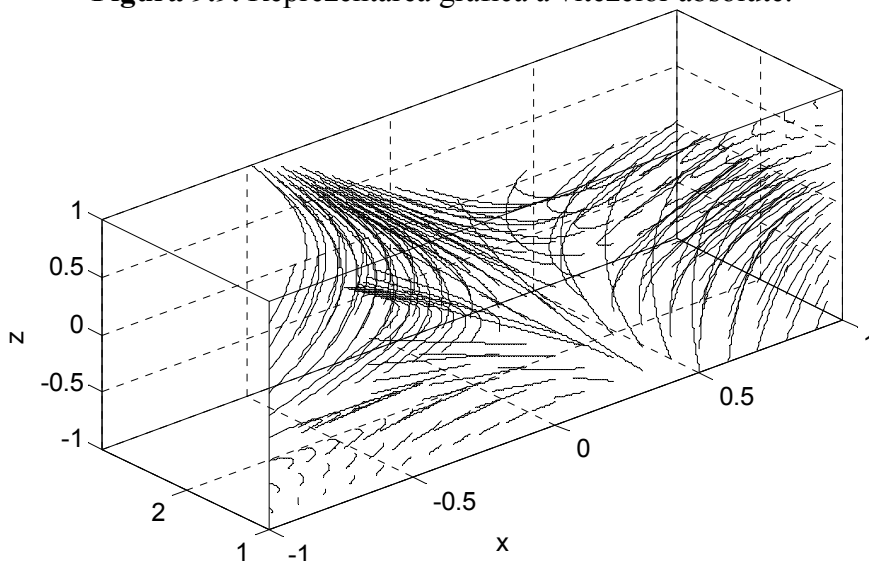
### Observații

- Domeniul matriceal de mișcare este definit prin instrucțiunea:  $[X, Y, Z]=\text{meshgrid}(x, y, z)$ .
- Componentele vitezei  $u(x, y, z)$ ,  $v(x, y, z)$  și  $w(x, y, z)$ , precum și viteza absolută  $V(x, y, z)$  sunt definite ca funcții anonymous.
- Reprezentarea grafică a vectorilor vitezelor absolute se obține cu instrucțiunea `quiver3`.
- Modificarea factorilor de scalare relativă pentru cele trei axe de coordonate se realizează cu instrucțiunea `daspect([kx ky kz])`, [16]. Coeficienții de scalare  $k_x$ ,  $k_y$  și  $k_z$  sunt corelați cu unitățile de măsură ale celor trei axe  $[u_x]$ ,  $[u_y]$  și  $[u_z]$  prin relația:

$$[u_x] \cdot k_x = [u_y] \cdot k_y = [u_z] \cdot k_z$$



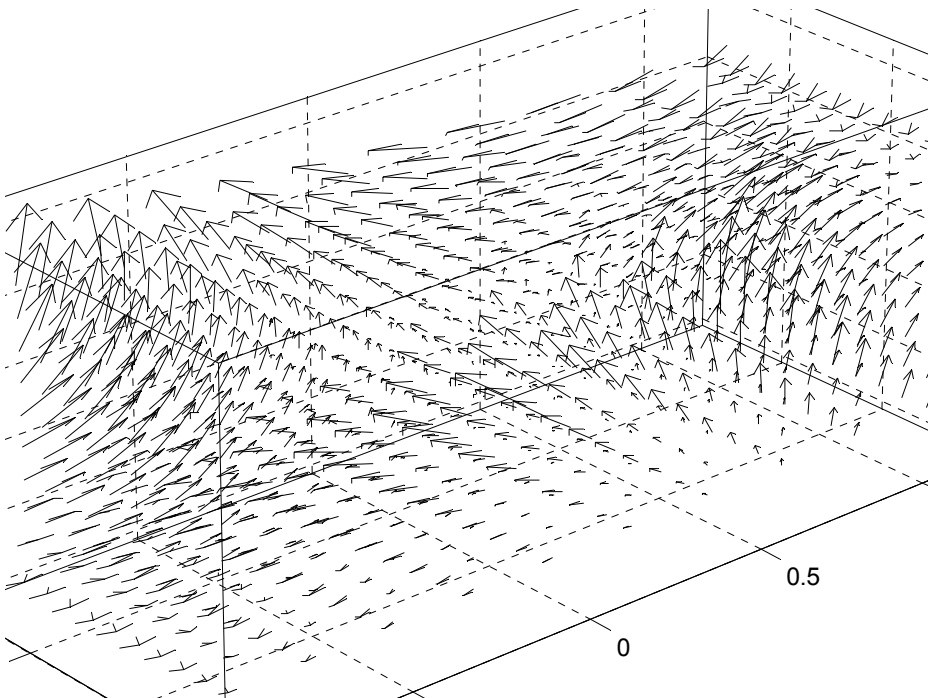
**Figura 9.9.** Reprezentarea grafică a vitezelor absolute.



**Figura 9.10.** Reprezentarea grafică a liniilor de curent.

- Modificarea direcției de vizualizare a reprezentării grafice se obține cu instrucțiunea `view([az e1])`, [17]. Parametrul  $az$  [°] (azimut) exprimă rotația în jurul axei  $Oz$  măsurată față de direcția negativă a axei  $Oy$ . Pentru  $az > 0$  se obține o rotație în sens trigonometric a direcției de vizualizare. Parametrul  $e1$  [°] (elevație) exprimă rotația față de planul  $Oxy$ . Pentru  $az > 0$  se obține vizualizarea de deasupra obiectului.

- Pentru reprezentarea grafică a liniilor de curent este necesară definirea punctelor de început ale acestora  $s_x$ ,  $s_y$  și  $s_z$ . Pentru cazul analizat, s-a utilizat o structură de puncte plasată într-un plan paralel cu planul  $Ozx$ , la distanța 1 față de acesta, structură definită printr-o discretizare cu pasul 0,1 pe axa  $Ox$ , respectiv cu pasul 0,2 pe axa  $Oz$ . Stabilirea grosimii și culorii liniilor de curent se realizează cu instrucțiunea de formatare `set(hlines,'LineWidth',1,'Color','k')`. Sensul de parcurgere al liniilor de curent se determină prin corelare cu reprezentarea grafică a vitezelor absolute din figura 9.9.
- Vizualizarea graficelor tridimensionale se poate face în proiecție ortografică (paralelă) sau în perspectivă. Controlul tipului de proiecție se realizează cu instrucțiunea `camproj`, [18]. În figura 9.10 se prezintă o vedere în perspectivă pentru realizarea căreia s-a utilizat instrucțiunea `camproj('perspective')`.
- Modificarea factorului de scalare generală a reprezentării grafice se realizează cu instrucțiunea `camzoom(fz)`, [19], în care valori supraunitare ale factorului de scalare  $f_z$  (factor de zoom) determină mărirea obiectului vizualizat. În figura 9.11 se prezintă o imagine mărită a vitezelor absolute pentru obținerea căreia s-a utilizat un factor de zoom având valoarea  $f_z = 1,5$ .



**Figura 9.11.** Proiecție în perspectivă și zoom 1,5.

### Problema 9.8

Calcululele aerodinamice ale unui ventilator axial permit obținerea parametrilor geometrici necesari pentru realizarea paletei și a rețelei de palete a ventilatorului:

- Raza la butuc este  $r_b=150$  mm; raza la vârf este  $r_v=300$  mm.
- Razele de calcul pe anvergura paletei sunt:  $r=\{0; 25; 50; 75; 100; 125; 150\}$  mm.
- Pentru construcția paletei, la butuc se folosește un profil aerodinamic de tip Gö 744 definit prin coordonatele intradosului  $y^-(x)$  și extradadosului  $y^+(x)$  conform tabelului de valori, [20]:

$x$	0,0	1,25	2,5	5,0	7,5	10	15	20	30
$y^-$	0,0	-1,886	-2,172	-2,145	-2,068	-1,89	-1,455	-0,88	-0,15
$y^+$	0,0	3,464	5,077	7,355	9,132	10,53	12,495	13,6	14,18
$x$	40	50	60	70	80	90	95	100	
$y^-$	0,11	-0,18	-0,74	-1,18	-1,32	-1,06	-0,655	0,0	
$y^+$	13,13	10,84	8,09	5,2	2,63	0,77	0,225	0,0	

- Coarda aerodinamică a profilului la butuc este  $L_b=100$  mm. Coarda aerodinamică a profilului la vârf este  $L_v=70$  mm. Valorile corzii aerodinamice la razele de calcul ale paletei sunt:  $L_r=\{100; 95; 90; 85; 80; 75; 70\}$  mm.
- La toate razele de calcul pe anvergura paletei se folosește același profil aerodinamic (Gö 744) dar scalat omogen cu factorii de scalare calculați cu relația  $f = L_r/L_b$ .
- Unghiurile de așezare ale profilelor aerodinamice la toate razele de calcul rezultate din calcululele aerodinamice au valorile:  $\beta_0=\{45; 39; 34; 30; 27; 25; 23\}$  °, [22].
- Numărul de palete ale ventilatorului axial este  $n_p=12$ .

Se cere să se determine geometria paletei prin reprezentarea grafică a profilului de la butuc, a profilelor aerodinamice din secțiunile de calcul, precum și prin reprezentarea suprafeței paletei. Se cere, de asemenea, să se reprezinte grafic rețeaua de palete rotorice a ventilatorului axial.

### Rezolvare

Pentru reprezentarea grafică a profilului aerodinamic din secțiunea butucului ventilatorului axial se utilizează un fișier de tip script conținând următoarele instrucțiuni principale:

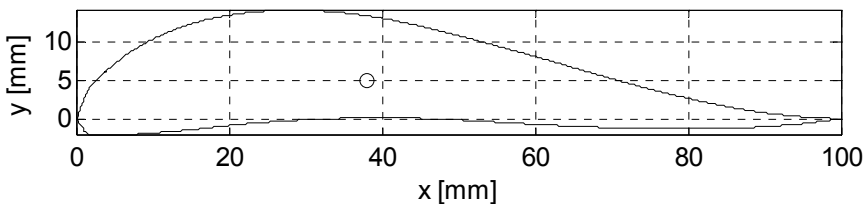
```
%% GEOMETRIA REȚELEI DE PALETE A UNUI VENTILATOR AXIAL
clear all;close all;clc;
%% DATE DE INTRARE
% Abscisa, x[mm]
x=[0 1.25 2.5 5 7.5 10 15 20:10:90 95 100];
% Coarda aerodinamica a profilului la butuc
```

```

L=max(x);
% Coordonatele extradadosului, ye [mm]
ye=[0.85 4.05 5.45 7.3 8.6 9.65 11 11.85 12.5 12.1 11.1 9.5
7.55 5.35 2.9 1.55 0.1];
% Coordonatele intradosului, yi [mm]
yi=[0.85 0 0.05 0.35 0.55 0.65 1.05 1.3 1.7 1.85 1.8 1.55
1.25 0.9 0.45 0.2 0.1];
%% CALCULE PRELIMINARE
% Definirea vectorului absciselor profilului
x1=[x fliplr(x)]';
% Calculul numarului de elemente ale vectorului x1
nx=length(x1);
% Definirea vectorului ordonatelor profilului
y1=[yi fliplr(ye)]';
% Definirea vectorului z pentru prima sectiune de calcul
z1=zeros(size(x1));
% Discretizarea fina a abscisei si ordonatei
xs=linspace(0,L,1000);
yes=interp1(x,ye,xs,'spline');yis=interp1(x,yi,xs,'spline');
% Calculul ariei profilului
AA=trapez(xs,yes-yis)
% Calculul coordonatelor centrului de greutate al profilului
xg=1/AA*trapez(xs,xs.*(yes-yis));
yg=1/(2*AA)*trapez(xs,yes.^2-yis.^2);zg=0;
%% REPREZENTARI GRAFICE
% Reprezentarea grafica a profilului aerodinamic
figure
plot(x1,y1,'-k');hold on;
grid on;axis image;xlabel('x [mm]');ylabel('y [mm]');
% Reprezentarea grafica a centrului de greutate al profilului
plot(xg,yg,'ok');

```

În urma lansării în execuție a acestui fișier de tip script se obține reprezentarea grafică din figura 9.12.



**Figura 9.12.** Profilul aerodinamic Gö 744.

### Observații

- Pentru obținerea unor valori numerice cât mai precise ale coordonatelor  $x_g$  și  $y_g$  ale centrului de greutate se recalculează coordonatele intradosului și extradadosului prin interpolare de tip spline folosind o discretizare mult mai fină (1000 de puncte) față de cea existentă în catalogul de profile (17 puncte).

Determinarea profilelor aerodinamice din fiecare secțiune de calcul se realizează prin aplicarea următoarelor transformări afine:

- Transformarea  $T_1$ , translația profilului astfel încât centrul său de greutate să coincidă cu originea sistemului de coordonate. Matricea transformării afine este:

$$M_1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -x_g & -y_g & -z_g & 1 \end{pmatrix}$$

- Transformarea  $T_2$ , scalarea profilului față de originea sistemului de coordonate. Matricea transformării este:

$$M_2 = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Transformarea  $T_3$ , rotația profilului față de originea sistemului de coordonate. Matricea transformării este:

$$M_3 = \begin{pmatrix} \cos \beta_0 & \sin \beta_0 & 0 & 0 \\ -\sin \beta_0 & \cos \beta_0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Transformarea  $T_4$ , translația profilului scalat și rotit, înapoi în poziția inițială. Matricea transformării este:

$$M_4 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ x_g & y_g & z_g & 1 \end{pmatrix}$$

- Transformarea  $T_5$ , translația profilului scalat și rotit din poziția inițială, pe anvergura paletii, la următoarea rază de calcul. Matricea transformării este:

$$M_5 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & r & 1 \end{pmatrix}$$

Pentru determinarea profilelor aerodinamice din secțiunile de calcul, precum și pentru reprezentarea grafică a acestora se completează fișierul de tip script cu următoarele instrucțiuni:

```
%% DATE DE INTRARE
% Raza sectiunilor de calcul
r=[0 25 50 75 100 125 150];nr=length(r);
% Coarda aerodinamica a profilelor in sectiunile de calcul
Lr=[100 95 90 85 80 75 70];
% Factorul de scalare
```

```

f=Lr/L;
% Unghiul de asezare al profilelor in sectiunile de calcul
t=[45 39 34 30 27 25 23]*pi/180;
%% MATRICELE TRANSFORMARILOR AFINE
% Matricea de translatie in originea sistemului de coordonate
M1=[1 0 0 0;0 1 0 0;0 0 1 0;-xg -yg -zg 1];
% Matricea de scalare a profilului
M2=@(f) [f 0 0 0;0 f 0 0;0 0 1 0;0 0 0 1];
% Matricea de rotatie
M3=@(t) [cos(t) -sin(t) 0 0;sin(t) cos(t) 0 0;0 0 1 0;0 0 0 1]
% Matricea de translatie in centrul de greutate
M4=[1 0 0 0;0 1 0 0;0 0 1 0;xg yg zg 1];
% Matricea de translatie radiala
M5=@(r) [1 0 0 0;0 1 0 0;0 0 1 0;0 0 r 1];
%% CALCULUL COORDONATELOR PROFILULUI; REPREZENTAREA GRAFICA
figure
for i=1:nr
    for j=1:nx
        % Translatia profilului in origine
        T1=maketform('affine',M1);
        % Scalarea profilului
        T2=maketform('affine',M2(f(i)));
        % Rotirea profilului
        T3=maketform('affine',M3(t(i)));
        % Translatia profilului in pozitia initiala
        T4=maketform('affine',M4);
        % Translatia profilului pe anvergura paletei
        T5=maketform('affine',M5(r(i)));
        % Matricea de transformare afina
        Ta=maketform('composite',[T5 T4 T3 T2 T1]);
        % Coordonatele profilului
        [x2(i,j) y2(i,j) z2(i,j)]=tformfwd([x1(j) y1(j) z1(j)],Ta)
    end
    % Reprezentarea grafica a profilului
    plot3(x2(i,:),y2(i,:),z2(i:,:),'k');hold on;
end
% Reprezentarea grafica a centrului de greutate
plot3(xg,yg,zg,'ok');grid on;axis image;box on;

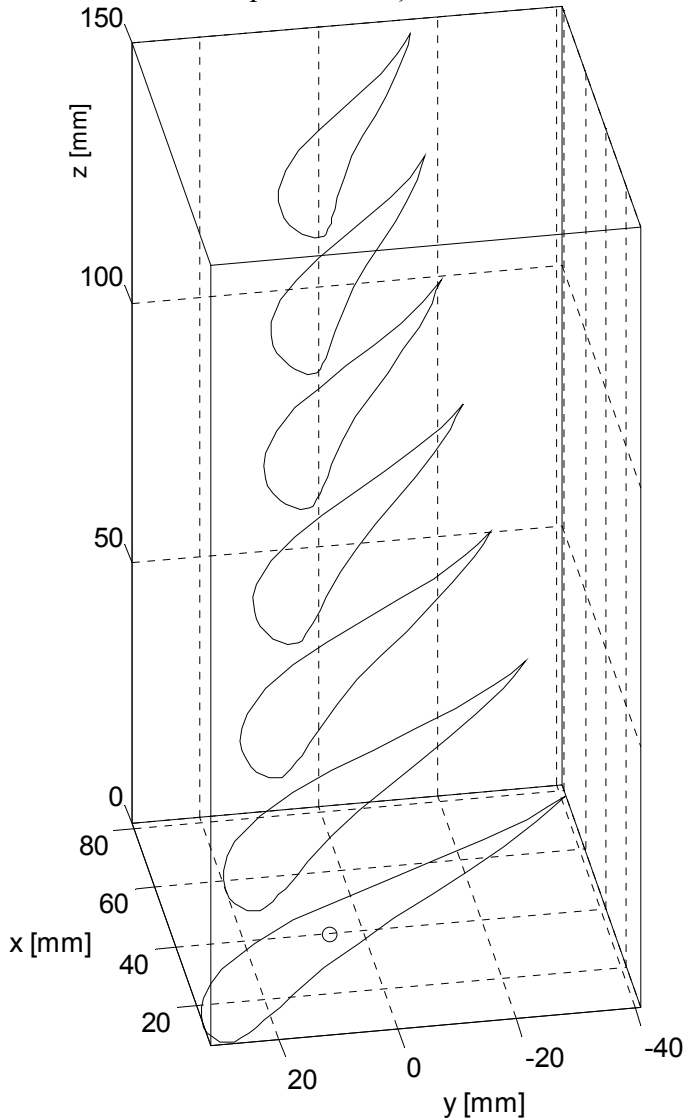
```

### Observații

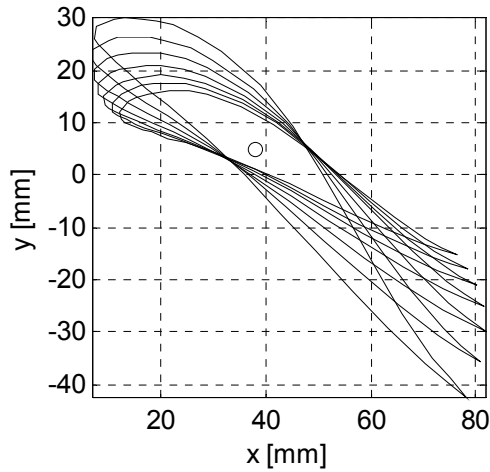
- Se observă existența a trei matrice de transformare care depind de câte un parametru ( $M_2$ ,  $M_3$  și  $M_5$ ) pentru definirea cărora s-a preferat utilizarea funcțiilor de tip anonymous.
- După definirea transformărilor afine elementare  $T_1$ ,  $T_2$ ,  $T_3$ ,  $T_4$  și  $T_5$  se determină transformarea compusă cu instrucțiunea  $Ta=maketform('composite',[T5 T4 T3 T2 T1])$ .
- Calculul coordonatelor după aplicarea transformării compuse se realizează cu instrucțiunea:  $[x2(i,j) y2(i,j) z2(i,j)]=tformfwd([x1(j) y1(j) z1(j)],Ta)$ .



- Toate aceste calcule se efectuează în cadrul unei structuri iterative duble, cu două contoare:  $i=1:nr$  pentru numărul secțiunilor de calcul și  $j=1:nx$  pentru numărul punctelor de pe profil. La ieșirea din structura iterativă interioară se reprezintă profilul aerodinamic din secțiunea respectivă cu instrucțiunea: `plot3(x2(i,:), y2(i,:), z2(i,:), 'k')`.
- În urma lansării în execuție a noului fișier de tip `script` se obține reprezentarea grafică din figura 9.13, a), pentru vederea definită prin instrucțiunea `view(-100,30)`, respectiv din figura 9.13, b), pentru vederea definită prin instrucțiunea `view(0,90)`.



a) vedere 3D, `view(-100,30)`



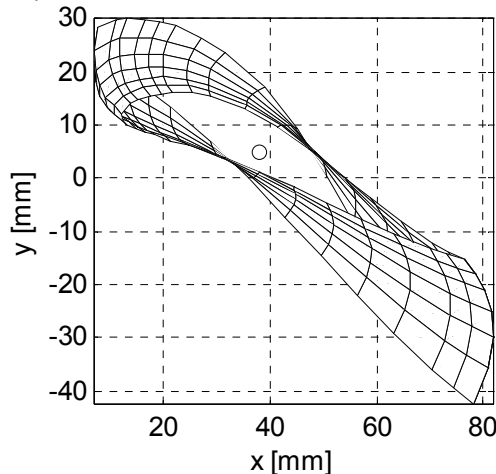
b) vedere 2D, view(0,90)

**Figura 9.13.** Profilele aerodinamice din secțiunile de calcul.

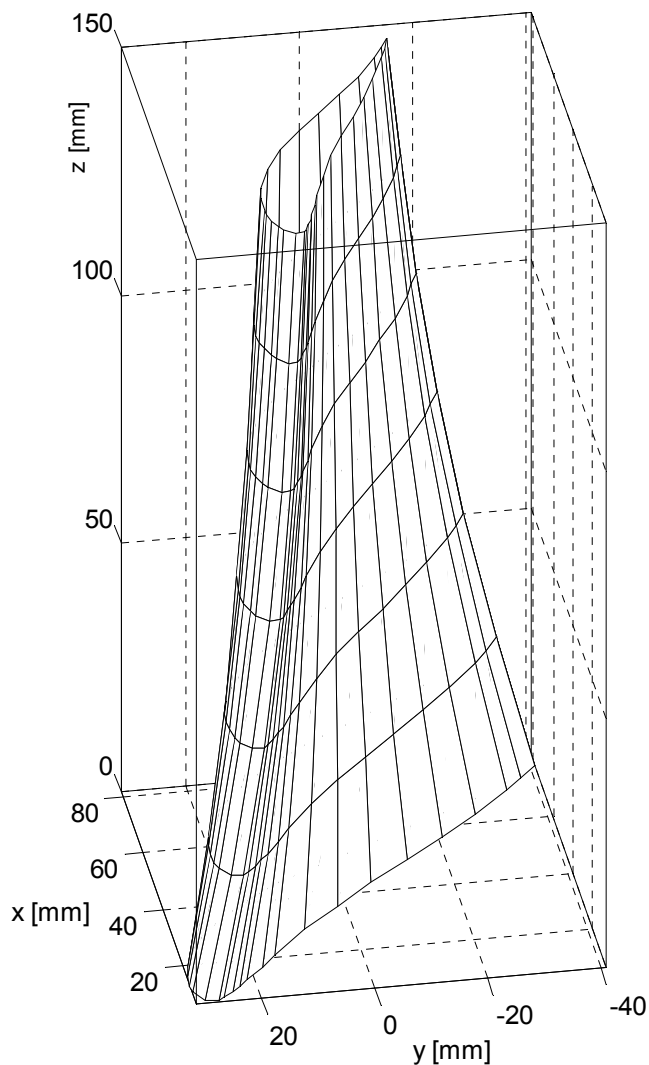
Pentru reprezentarea grafică a suprafeței paletelor se completează fișierul de tip script cu următoarele instrucțiuni:

```
%% REPREZENTAREA GRAFICA A PALETEI, mesh
figure
mesh(x2,y2,z2);
% Formatarea figurii
colormap([0 0 0]);grid on;axis image;box on;
xlabel('x [mm]');ylabel('y [mm]');zlabel('z [mm]');
```

În urma lansării în execuție a noului fișier de tip script se obține reprezentarea grafică din figura 9.14, a), pentru vederea definită prin instrucțiunea view(0,90), respectiv din figura 9.14, b) pentru vederea definită prin instrucțiunea view(-100,30).



a) vedere 2D, view(0,90)



b) vedere 3D, view(-100, 30)

**Figura 9.14.** Paleta ventilatorului axial.

Obținerea tuturor paletelor și poziționarea acestora la raza butucului se realizează prin aplicarea următoarelor transformări afine:

- Transformarea  $T_6$ , translația paletelor în direcția pozitivă a axei  $Oz$ , astfel încât profilul de la baza paletelor să fie tangent la butuc. Matricea transformării afine este:

$$M_6 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & rb & 1 \end{pmatrix}$$

- Transformarea  $T_7$ , rotația paletelor în jurul axei  $Oy$ . Matricea transformării este:

$$M_7 = \begin{pmatrix} \cos \theta_p & 0 & \sin \theta_p & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \theta_p & 0 & \cos \theta_p & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

în care  $\theta_p$  este vectorul unghiurilor de poziționare ale celor  $n_p=12$  palete ale ventilatorului având valorile  $k \cdot 2\pi/n_p$ ,  $k=0 \div n_p-1$ .

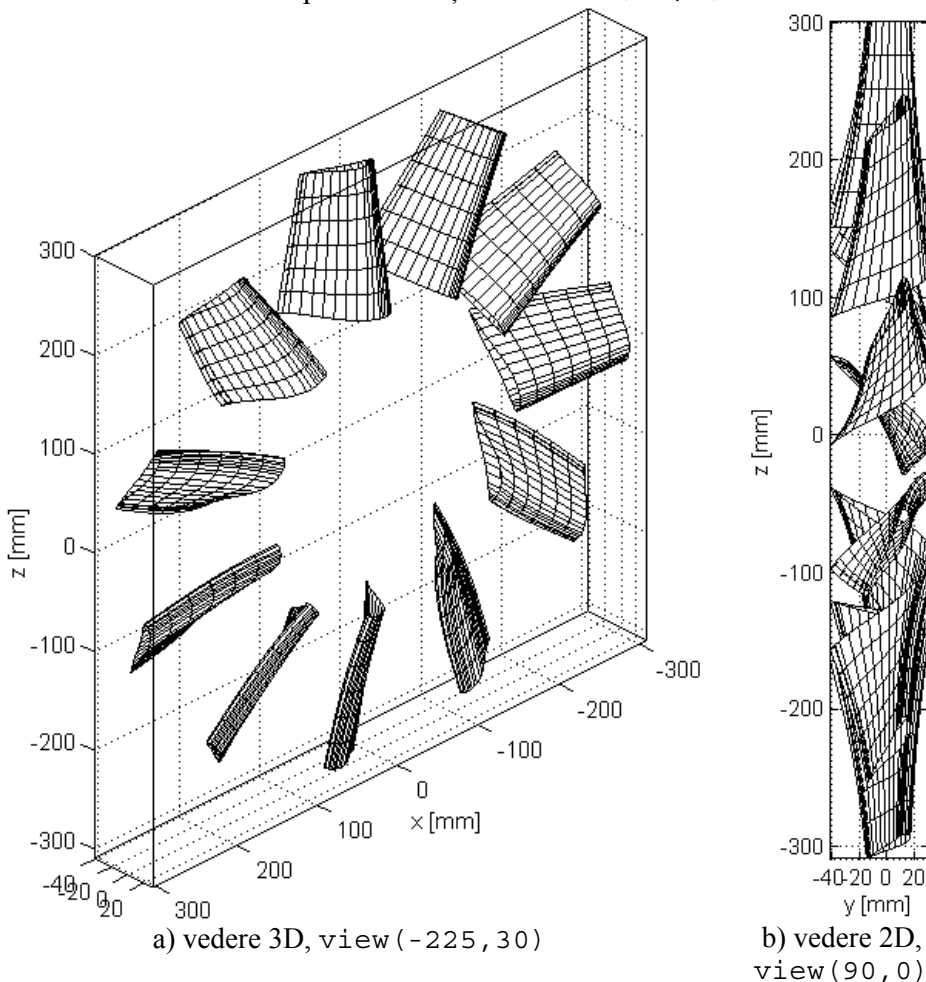
Pentru reprezentarea grafică a rețelei de palete rotorice se completează fișierul de tip script cu următoarele instrucțiuni:

```
%% DATE DE INTRARE
% Raza la butuc, [mm]
rb=150;
% Numarul palete
np=12;
% Unghiurile de pozitionare ale paletelor pe butuc
tp=linspace(0,2*pi,np);
%% MATRICELE TRANSFORMARILOR AFINE
% Matricea de pozitionare a paletii pe butuc
M6=[1 0 0 0;0 1 0 0;0 0 1 0;0 0 rb 1];
%Matricea de rotatie a paletii:
M7=@(tp) [cos(tp) 0 sin(tp) 0;0 1 0 0;-sin(tp) 0 cos(tp) 0;0
0 0 1];
% Translatia de pozitionare a paletii pe butuc
T6=maketform('affine',M6);
%% REPREZENTAREA GRAFICA A RETELEI DE PALETE ROTORICE
figure
for k=1:np
    % Rotatia paletii
    T7=maketform('affine',M7(tp(k)));
    % Matricea transformarii compuse
    Tb=maketform('composite',[T7 T6]);
    for i=1:nr
        for j=1:nx
            [x3(i,j) y3(i,j) z3(i,j)]=tformfwd([x2(i,j) y2(i,j)
z2(i,j)],Tb);
        end
    end
    mesh(x3,y3,z3);hold on;end
```

### Observații

- Se observă că matricea de transformare  $M_7$  depinde de parametrul  $\theta_p$ . Prin urmare, pentru definirea matricei de transformare  $M_7$  s-a preferat utilizarea unei funcții de tip anonymous.
- După definirea transformărilor afine elementare  $T_6$  și  $T_7$  se determină transformarea compusă cu instrucțiunea  $T_b = \text{maketform}('composite', [T_7 T_6])$ .

- Calculul coordonatelor obiectului transformat se face prin aplicarea asupra imaginii inițiale a transformării compuse  $T_b$ , folosind instrucțiunea  $[x3(i,j) \ y3(i,j) \ z3(i,j)] = \text{tformfwd}([x2(j) \ y2(j) \ z2(j)], T_b)$ .
- Toate aceste calcule se efectuează în cadrul unei structuri iterative triple, cu trei contoare:  $k=1:n_p$  pentru numărul paletelor,  $i=1:n_r$  pentru numărul secțiunilor de calcul și  $j=1:n_x$  pentru numărul punctelor de pe profil. La ieșirea din cele două structuri iterative interioare se reprezintă paleta cu indicele curent  $k$  din cele  $n_p$  palete, folosind instrucțiunea  $\text{mesh}(x3, y3, z3)$ .
- În urma lansării în execuție a noului fișier de tip `script` se obține reprezentarea grafică din figura 9.15, a), pentru vederea definită prin instrucțiunea `view(-225, 30)`, respectiv din figura 9.15, b) pentru vederea definită prin instrucțiunea `view(90, 0)`.



**Figura 9.15.** Rețeaua de palete a ventilatorului axial.

## BIBLIOGRAFIE

1. MathWorks, MATLAB, Graphics, 2014.
2. MathWorks, 2D and 3D Plots, <http://www.mathworks.com/help/matlab/2-and-3d-plots.html>, accesat la 2.02.2014.
3. MathWorks, 3D Line Plot, <http://www.mathworks.com/help/matlab/ref/plot3.html>, accesat la 6.03.2014.
4. MathWorks, Mesh Plot, <http://www.mathworks.com/help/matlab/ref/mesh.html>, accesat la 6.03.2014.
5. MathWorks, Plot a Contour Graph Under Mesh Graph, <http://www.mathworks.com/help/matlab/ref/meshc.html>, accesat la 6.03.2014.
6. MathWorks, Plot a Curtain Around Mesh Plot, <http://www.mathworks.com/help/matlab/ref/meshz.html>, accesat la 6.03.2014.
7. MathWorks, Set and Get Current Colormap, <http://www.mathworks.com/help/matlab/ref/colormap.html>, accesat la 6/03.2014.
8. MathWorks, Colorbar Showing Color Scale, <http://www.mathworks.com/help/matlab/ref/colorbar.html>, accesat la 6.03.2014.
9. MathWorks, 3-D Shaded Surface Plot, <http://www.mathworks.com/help/matlab/ref/surf.html>, accesat la 6.03.2014.
10. MathWorks, Contour Plot Under a 3-D Shaded Surface Plot, <http://www.mathworks.com/help/matlab/ref/surfc.html>, accesat la 6.03.2014.
11. MathWorks, Set Color Shading Properties, <http://www.mathworks.com/help/matlab/ref/shading.html>, accesat la 6.03.2014.
12. MathWorks, Contour Plot of Matrix, <http://www.mathworks.com/help/matlab/ref/contour.html>, accesat la 8.03.2014.
13. MathWorks, Numerical Gradient, <http://www.mathworks.com/help/matlab/ref/gradient.html>, accesat la 8.03.2014.
14. MathWorks, Quiver or Velocity Plot, <http://www.mathworks.com/help/matlab/ref/quiver.html>, accesat la 8.03.2014.
15. MathWorks, 3D Quiver or Velocity Plot, <http://www.mathworks.com/help/matlab/ref/quiver3.html>, accesat la 8.03.2014.
16. MathWorks, Set or Query Axes Data Aspect Ratio, <http://www.mathworks.com/help/matlab/ref/daspect.html>, accesat la 8.03.2014.
17. MathWorks, Viewpoint Specification, <http://www.mathworks.com/help/matlab/ref/view.html>, accesat la 8.03.2014.
18. MathWorks, Set or Query Projection Type, <http://www.mathworks.com/help/matlab/ref/camproj.html>, accesat la 8.03.2014.
19. MathWorks, Zoom In and Out on Scene, <http://www.mathworks.com/help/matlab/ref/camzoom.html>, accesat la 8.03.2014.
20. UIUC Applied Aerodynamics Group, <http://aerospace.illinois.edu/m-selig/>, accesat la 19.03.2014.
21. Zahariea D., Mișcări potențiale plane. Simulări numerice., Ed. CERMI, Iași, 2005.
22. Zahariea D., Proiectarea unui ventilator axial, Proiect de Diplomă, coordonator: Rusu I. Ilie., Universitatea Tehnică „Gh.Asachi”, Iași, 1989.

## CAPITOLUL 10

### MODELAREA ȘI SIMULAREA SISTEMELOR DINAMICE ÎN SIMULINK

#### 10.1. GENERALITĂȚI

Simulink este un mediu de programare special dezvoltat pentru modelarea, simularea și analiza sistemelor dinamice. Simulink este integrat în limbajul de programare MATLAB și folosește capacitățile computaționale ale acestuia. Prima versiune a mediului de programare Simulink (1.0) a fost lansată în anul 1990, odată cu versiunea MATLAB 3.5. Cea mai recentă versiune a mediului de programare Simulink (8.3) a fost lansată în anul 2014, odată cu versiunea MATLAB 8.3 (R2014a).

Principalele etape ale procesului de simulare a sistemelor dinamice utilizând mediul de programare Simulink sunt, [1, 2, 3, 4]:

- Modelarea sistemului fizic supus analizei prin crearea unui model grafic cu ajutorul editorului de modele Simulink. Realizarea modelului grafic trebuie să se facă în concordanță cu relațiile matematice care descriu comportarea sistemului fizic analizat. Prin urmare, modelarea sistemului fizic se desfășoară în două etape:
  - Crearea modelului analitic al sistemului fizic reprezentat de un set de ecuații care descriu comportarea dinamică a sistemului fizic.
  - Crearea modelului numeric de simulare reprezentat de modelul grafic obținut cu ajutorul editorului de modele Simulink.
- Simularea comportării dinamice a sistemului fizic analizat prin rezolvarea numerică a sistemului de ecuații pe baza cărora a fost realizat modelul grafic al sistemului fizic.

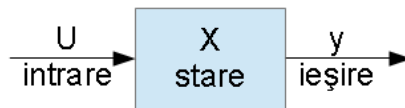
#### 10.1.1. Modelarea sistemelor dinamice

Crearea modelului analitic al sistemului depinde de natura și particularitățile sistemului fizic analizat și reprezintă faza principală a procesului de modelare. Erorile în evaluarea și interpretarea diferitelor aspecte ale sistemului fizic analizat, precum și unele ipoteze simplificatoare neadecvate pot conduce la un model analitic care să corespundă doar parțial cu sistemul fizic real analizat.

Crearea modelului numeric de simulare pe baza modelului analitic reprezintă cea de-a doua fază a procesului de modelare a sistemelor dinamice. Buna cunoaștere a facilităților computaționale ale mediului de programare Simulink asigură obținerea unor modele numerice aflate în bună concordanță cu modelul analitic și în ultimă instanță cu sistemul fizic analizat.

Principalele elemente specifice fazei de modelare a sistemelor dinamice sunt, [3, 4]:

- **Schema bloc (diagrama de blocuri).** Modelul grafic al sistemului dinamic analizat se numește schemă bloc sau diagramă de blocuri. Schema bloc constă dintr-un set de simboluri, numite blocuri, interconectate prin linii de conexiune prin care circulă semnale numerice. Fiecare bloc reprezintă un sistem dinamic elementar care, în general, primește un semnal de intrare și generează un semnal de ieșire care depinde după o lege oarecare (starea sistemului) de semnalul de intrare, figura 10.1. Tipul blocului determină relația de corespondență dintre ieșire, intrare, stare și timpul de simulare.



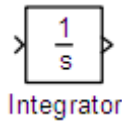
**Figura 10.1.** Intrarea, starea și ieșirea blocurilor Simulink.

Liniile de conexiune reprezintă modalitatea de interconectare a ieșirilor unor blocuri cu intrările altor blocuri. O diagramă de blocuri poate conține un număr oricât de mare de blocuri, de același fel sau diferite.

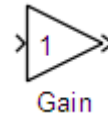
- **Starea unui sistem** este o variabilă care determină ieșirea unui bloc și a cărei valoare curentă este o funcție de valorile anterioare ale stărilor blocului respectiv. Prin urmare, pentru a putea determina starea curentă, blocul respectiv trebuie să conserve valorile anterioare ale stării sistemului. Pentru aceasta blocul trebuie să posede o zonă de memorie internă în care vor fi conservate atât valorile stărilor anterioare, cât și valorile semnalelor de intrare pe baza cărora se va putea determina, în orice moment, starea curentă. Din acest punct de vedere blocurile se clasifică în două categorii:
  - Blocuri cu stări, figura 10.2, a). De exemplu, blocul de integrare (Integrator) furnizează la ieșire valoarea numerică a integralei semnalului primit pe portul de intrare începând cu momentul de start până în momentul de sfârșit al simulării. Valoarea integralei la un moment dat depinde de valorile anterioare calculate și conservate în memoria internă a blocului.



- Blocuri fără stări, figura 10.2, b). De exemplu, blocul multiplicator (Gain) furnizează pe portul de ieșire valoarea primită pe portul de intrare multiplicată cu o valoare constantă, denumită factor de multiplicare (gain). Ieșirea blocului este în acest caz determinată doar de valoarea curentă de pe portul de intrare și de factorul de multiplicare, care sunt mărimi constante.



a) bloc cu stări



b) bloc fără stări

**Figura 10.2.** Tipuri de blocuri Simulink.

- **Funcțiile unui sistem.** Fiecare bloc Simulink este asociat cu un set de funcții prin care se specifică relația de dependență dintre semnalele de intrare, stările sistemului și semnalele de ieșire. Funcțiile asociate blocurilor Simulink utilizate pentru simularea sistemelor continue sunt:
  - Funcția de ieșire  $f_0$  care specifică relația de dependență dintre semnalul de ieșire  $y$  și semnalul de intrare  $u$ , stările  $x$  și timpul de simulare  $t$ :
 
$$y = f_0(u, x, t)$$
  - Funcția derivată  $f_d$  care reprezintă derivata  $x'$  a stării sistemului continuu în timp în funcție de valorile curente ale semnalul de intrare  $u$ , de stările  $x$  și de timpul de simulare  $t$ :
 
$$x' = f_d(u, x, t)$$
- **Parametrii caracteristici.** Reprezintă proprietățile fundamentale ale fiecărui tip de bloc. Fiecare bloc cu parametri dispune de o fereastră de dialog care permite modificarea valorilor parametrilor caracteristici atât în faza de editare, cât și în faza de simulare a modelului. Pentru introducerea parametrilor se pot utiliza atât valori numerice, expresii MATLAB, cât și numele unor variabile existente în spațiul de lucru al programului. Mediul de programare Simulink evaluează expresiile și citește variabilele din spațiul de lucru înainte de începerea simulării astfel încât, în cursul procesului de simulare, toți parametrii sunt cunoscuți, prin valorile lor numerice. Parametrii blocurilor nu pot fi modificați în timpul procesului de simulare, astfel încât după pornirea simulării, fereastra de dialog care permitea modificarea parametrilor nu mai este disponibilă. Se asigură astfel nu numai interpretarea corectă a relației de dependență dintre intrări și ieșiri, dar și o viteză ridicată de execuție a procesului de simulare.

- **Blocuri de tip continuu sau discret.** În mediul de programare Simulink sunt definite două tipuri de blocuri:
  - Blocuri de tip continuu, care răspund în mod continuu la variația continuă a semnalului de intrare.
  - Blocuri de tip discret, care răspund la variația intrării doar la valori discrete de timp, valori determinate în funcție de perioadă de eșantionare (*sample time*). În acest mod, un bloc discret păstrează constantă valoarea de ieșire între două momente succesive de eșantionare. Blocurile de tip discret conțin un parametru caracteristic prin intermediul căruia se poate specifica valoarea perioadei de eșantionare.

În funcție de tipul semnalului de pe portul de intrare (semnal continuu sau semnal discret) unele blocuri pot fi de tip continuu sau de tip discret. Aceste blocuri au o perioadă de eșantionare implicită care poate fi:

  - Continuă, dacă există cel puțin un semnal continuu pe porturile de intrare.
  - Cea mai mică perioadă de eșantionare, dacă toate semnalele de intrare au perioade de eșantionare multipli întregi ai celei mai mici perioade de eșantionare.
  - Perioada de eșantionare fundamentală, egală cu cel mai mare divizor comun al perioadelor de eșantionare ale semnalelor de intrare.
- **Subsisteme.** Simulink permite modelarea sistemelor complexe folosind principiul modularizării prin crearea subsistemelor (fiecare subsistem conține câte o schemă bloc cu semnale de intrare și de ieșire proprii), precum și principiul ierarhizării prin interconectarea în cadrul unui sistem a mai multor subsisteme.
- **Blocuri definite de utilizator.** Creșterea performanțelor computaționale ale mediului de programare Simulink se poate face și prin crearea bibliotecilor de blocuri definite de utilizator destinate rezolvării unor anumite tipuri de probleme. Crearea blocurilor definite de utilizator se poate face prin două metode: metoda grafică, pornind de la o schemă bloc anterioară și metoda prin programare, pornind de la un fișier de tip `script`.
- **Semnale.** Semnalele Simulink reprezintă succesiuni de valori numerice adimensionale care circulă pe liniile de comunicație dintre porturile de ieșire ale unor blocuri și porturile de intrare ale altor blocuri. Principalele caracteristici ale semnalelor Simulink sunt: numele semnalului; tipul datelor numerice asociate semnalului (8 biți, 16 biți, 32 biți); semnal de tip real sau complex; dimensiunea semnalului (semnale unidimensionale sau bidimensionale).

- **Solverul numeric.** Modelele Simulink determină derivatele în funcție de timp ale stărilor continue dar nu și valorile propriu-zise ale stărilor. Astfel, în cursul procesului de simulare, valorile stărilor continue se obțin prin integrarea numerică a derivatelor stărilor.

Din multitudinea de tehnici de integrare numerică specifice diferitelor domenii, mediul de programare Simulink folosește cele mai stabile, eficiente și precise metode de integrare cu pas constant sau cu pas variabil. Alegerea tipului de solver numeric se poate face în oricare din fazele de modelare sau de simulare ale unui sistem dinamic prin selectarea din meniul *Simulation* a opțiunii *Configuration Parameters*. Se deschide astfel o fereastră de configurare, care în partea stângă, cuprinde o structură arborescentă cu toate elementele configurabile. După selectarea elementului dorit, în partea dreaptă a ferestrei apar parametrii de configurare particulari, cu valorile numerice implicite. Pentru cazul solverului numeric, utilizatorul poate configura diferiți parametri, printre care: tipul solverului, valoarea inițială, minimă și maximă a pasului de integrare, metoda utilizată pentru detectarea discontinuităților, eroarea relativă limită și eroarea absolută limită care intervin în cursul procesului automat de adaptare a pasului de integrare, intervalul de timp al simulării.

În cazul tehnicilor de integrare cu pas constant rezolvarea modelului se face la momente constante de timp pe toată durata timpului de simulare. Mărimea intervalului de timp al simulării (pasul simulării) poate fi specificat de utilizator sau poate fi determinat automat de către solver. În general, micșorarea pasului simulării conduce la îmbunătățirea preciziei de calcul dar și la creșterea timpului necesar pentru finalizarea simulării sistemului.

Principalele tehnici de integrare cu pas constant sunt: ode 1 (algoritm Euler); ode 2 (algoritm Heun); ode 3 (algoritm Bogacki-Shampine); ode 4 (algoritm Runge-Kutta, RK4); ode 5 (algoritm Dormand-Prince, RK5); ode8 (algoritm Dormand-Prince, RK8).

În cazul metodelor de integrare cu pas variabil pasul variază în timpul simulării în funcție de dinamica stărilor sistemului: pasul se micșorează în cazul în care stările se modifică rapid, sau se mărește pentru cazul unei variații lente a stărilor. Principalele tehnici de integrare cu pas variabil sunt: ode45 (algoritm Dormand-Prince, RK 4-5); ode23 (algoritm Bogacki-Shampine, RK 2-3); ode113 (algoritm Adams-Bashforth-Moulton); ode15s (algoritm NDF); ode23s (algoritm Rosenbrock); ode23t (algoritm conform metodei trapezului cu interpolant liber); ode23tb (algoritm combinat metoda trapezului TR-metoda diferențelor înapoi de ordinul doi BDF2).

### 10.1.2. Simularea sistemelor dinamice

După ce în faza de modelare, a fost creată schema bloc pe baza modelului analitic al sistemului fizic de analizat, urmează lansarea în execuție a procesului de simulare. Principalele elemente specifice fazei de simulare a comportării dinamice a sistemelor fizice sunt:

- **Inițializarea modelului.** Presupune realizarea mai multor operațiuni: evaluarea expresiilor care definesc valorile parametrilor tuturor blocurilor din structura schemei bloc; determinarea caracteristicilor semnalelor și verificarea existenței unor eventuale incompatibilități între semnale și tipurile blocurilor din structura schemei bloc; evaluarea necesarului de memorie pentru executarea procesului de simulare; evaluarea gradului de ierarhizare al schemei bloc, substituind eventualele subsisteme existente în structura schemei bloc principale cu schemele bloc corespunzătoare; sortarea blocurilor în ordinea în care acestea se vor executa în cursul procesului de simulare; determinarea perioadei de eșantionare pentru toate blocurile din structura schemei bloc care nu au definită în mod explicit o perioadă de eșantionare; alocarea și inițializarea spațiului de memorie pentru conservarea valorilor curente ale stărilor și ieșirilor tuturor blocurilor din structura schemei bloc principale, inclusiv a eventualelor subsisteme.
- **Execuția modelului.** În această fază, pe baza informațiilor obținute în faza de inițializare, se calculează succesiv, stările și ieșirile sistemului la intervale de timp cuprinse între momentul de start și momentul de sfârșit al simulării. Momentele succesive de timp la care se vor calcula stările și ieșirile sistemului se numesc pași de timp ai simulării (*time steps*). Intervalul de timp cuprins între doi pași de timp succesivi se numește interval elementar de calcul. Valoarea efectivă a intervalului elementar de calcul depinde de tipul solverului numeric utilizat pentru a calcula stările unui sistem continuu, de perioada fundamentală de eșantionare pentru sisteme discrete, precum și de existența în cadrul sistemelor continue a unor eventuale discontinuități.

La începutul simulării, pe baza modelului, se stabilesc stările inițiale și ieșirile sistemului care va fi simulat. La fiecare interval elementar de calcul se calculează noile valori ale intrărilor, stărilor și ieșirilor realizându-se astfel actualizarea modelului pentru a reflecta noile valori calculate, astfel încât la sfârșitul simulării, modelul va reflecta valorile finale ale intrărilor, stărilor și ieșirilor.

Actualizarea modelului la fiecare interval elementar de calcul este un proces complex care presupune: calculul valorii de ieșire pe baza intrărilor, a stării și a timpului; actualizarea stărilor blocurilor în

ordinea prestabilită în cursul fazei de inițializare a modelului; verificarea discontinuităților, precum și calculul valorii următoare a timpului de calcul pe baza valorii curente a timpului și a intervalului elementar de calcul.

Ordinea prestabilită de actualizare a blocurilor se realizează pe baza tipului fiecărui bloc, mai exact pe baza relației de dependență dintre ieșirile și intrările unui bloc. Din acest punct de vedere, blocurile Simulink pot fi:

- Cu transfer direct, pentru care portul de intrare determină în mod direct valoarea curentă de pe portul de ieșire (de exemplu blocurile Gain, Product, Sum).
- Fără transfer direct, pentru care există alte tipuri de dependențe. De exemplu, există blocuri la care valoarea de ieșire depinde doar de starea sistemului (Integrator), sau blocuri fără semnal de intrare (Constant).

Indiferent însă de tipul blocurilor, la stabilirea ordinii de actualizare a blocurilor sunt considerate și următoarele reguli de ordonare: fiecare bloc se va actualiza înainte de toate blocurile cu care este conectat prin porturi cu transfer direct; blocurile fără porturi cu transfer direct pot fi actualizate în orice ordine, atât timp cât nu intervin în blocuri cu transfer direct.

În cursul procesului de simulare, la fiecare interval elementar de calcul, obținerea valorilor curente ale stării și ieșirii sistemului se realizează printr-un calcul iterativ. Numărul necesar de iterații depinde de valoarea erorii locale care se calculează pentru fiecare interval elementar și care nu trebuie să depășească eroarea limită. În cazul în care eroarea locală calculată este mai mare decât valoarea limită admisă, solverul micșorează automat pasul de integrare și repetă calculul.

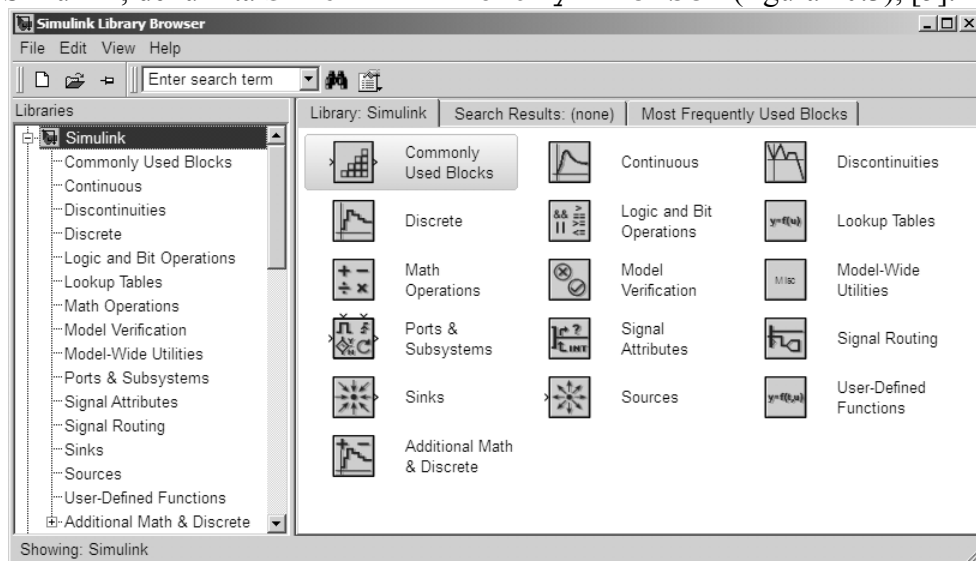
- **Eliminarea discontinuităților.** Una din etapele procesului de actualizare a modelului este detectarea și eliminarea discontinuităților. Pentru aceasta se determină cu exactitate momentul de timp la care apare o discontinuitate. Eliminarea discontinuității se realizează prin introducerea unor intervale de timp înainte și după momentul apariției discontinuității astfel încât discontinuitatea să coincidă cu domeniul interior al unui interval elementar de calcul. Pentru aceasta se impune utilizarea solverelor cu pas variabil care realizează automat modificarea pasului de discretizare în jurul discontinuității astfel încât momentul discontinuității să nu coincidă cu nici unul din momentele simulării. Ajustarea pasului de integrare se realizează în mod automat în funcție de valorile erorilor limită relative și absolute.

## 10.2. MODUL DE LUCRU ÎN SIMULINK

Modelarea și simularea sistemelor dinamice în mediul de programare Simulink presupune parcurgerea mai multor etape, atât în faza de modelare, cât și în faza de simulare: lansarea în execuție a programului Simulink; identificarea bibliotecilor de blocuri Simulink; crearea, salvarea, deschiderea și printarea modelelor Simulink; introducerea comenzilor Simulink; identificarea ferestrelor Simulink; introducerea, modificarea parametrilor, formatarea și conectarea blocurilor Simulink; selectarea obiectelor într-o schemă bloc Simulink; introducerea textelor explicative într-o schemă bloc Simulink; crearea subsistemelor Simulink.

### 10.2.1. Lansarea în execuție a programului Simulink

După inițierea unei sesiuni de lucru în limbajul de programare MATLAB, lansarea în execuție a mediului de programare Simulink se poate face prin două metode: se selectează pictograma programului Simulink de pe bara cu butoane sau se introduce comanda `simulink` direct în fereastra de comenzi. În acest mod se deschide fereastra principală a programului Simulink, denumită Simulink Library Browser (figura 10.3), [5].



**Figura 10.3.** Fereastra Simulink Library Browser.

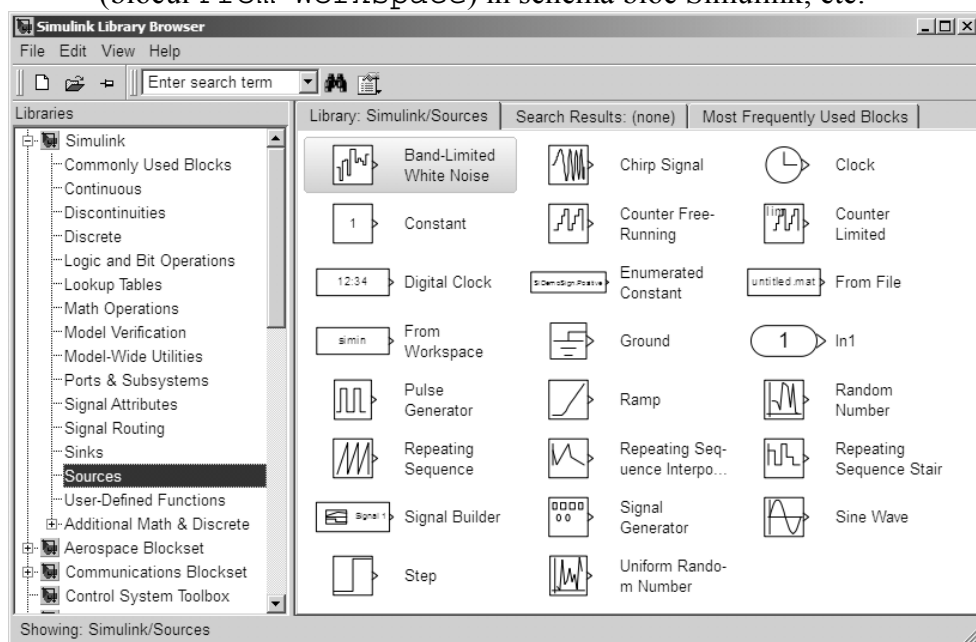
Această fereastră conține toate blocurile Simulink grupate în biblioteci de blocuri. Accesul la un anumit bloc se face prin identificarea bibliotecii corespunzătoare și prin selectarea apoi a blocului dorit. Cunoașterea locației fiecărui bloc este o sarcină dificilă și nu neapărat necesară dacă se consideră următoarele aspecte: există o bibliotecă denumită `Commonly Used Blocks` care conține cele mai des utilizate blocuri, există o funcție de căutare a blocurilor, `Enter search term`.

## 10.2.2. Biblioteci de blocuri Simulink

În partea stângă a ferestrei Simulink Library Browser se afișează bibliotecile Simulink, iar în partea dreaptă se prezintă blocurile individuale din biblioteca selectată.

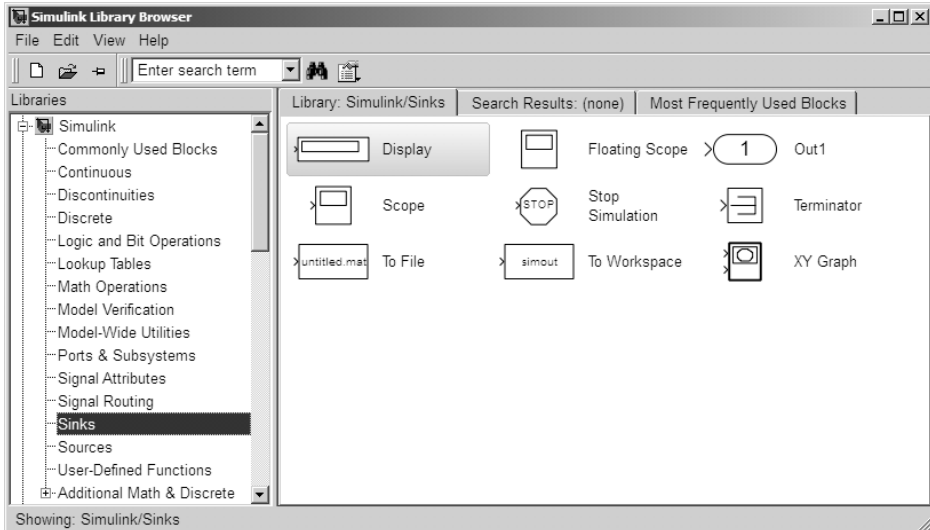
Principalele biblioteci de blocuri utilizate la modelarea și simularea sistemelor dinamice continue sunt:

- Biblioteca Sources, figura 10.4, [6]. Conține blocuri pentru specificarea mărimilor de intrare: timpul de simulare (blocul Clock); semnal constant (blocul Constant); semnal de tip sinusoidal (blocul Sine Wave); semnal de tip treaptă (blocul Step); semnal de tip rampă (blocul Ramp); semnal de tip impuls (blocul Pulse Generator); semnal de tip secvență repetitivă (blocul Repeating Sequence); bloc generator de semnal (blocul Signal Generator); transferul unor parametri din fișiere (blocul From File) sau din spațiul de lucru MATLAB (blocul From Workspace) în schema bloc Simulink; etc.



**Figura 10.4.** Blocurile specifice bibliotecii Sources.

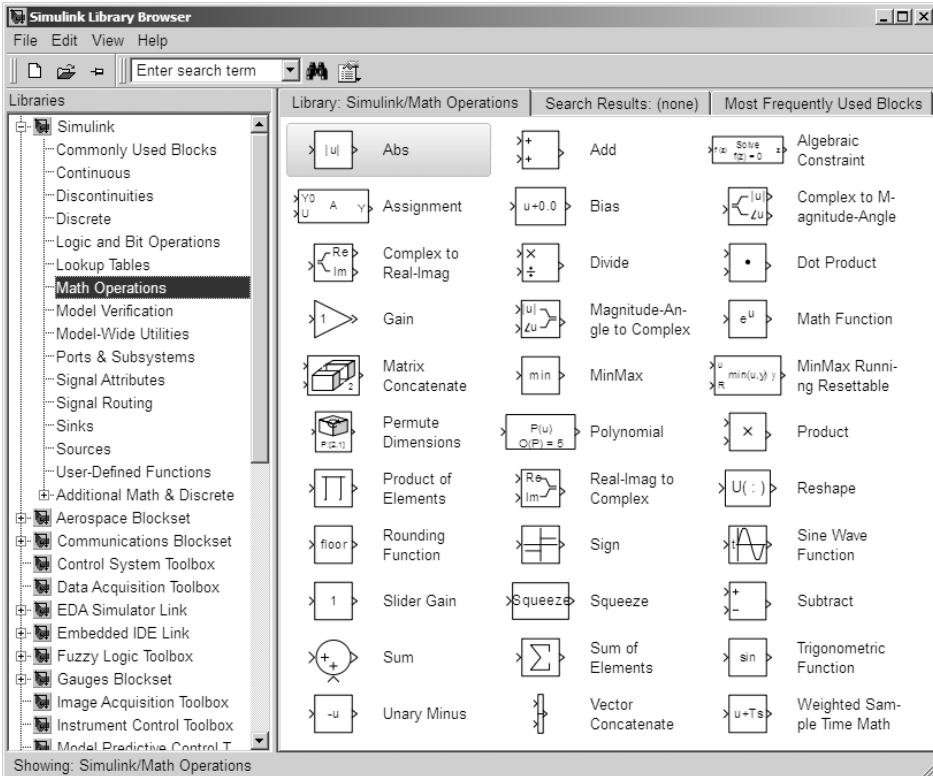
- Biblioteca Sinks, figura 10.5, [7]. Conține blocuri pentru specificarea mărimilor de ieșire: vizualizarea valorilor numerice (blocul Display); vizualizarea variației în timp a unor semnale (blocul Scope); vizualizarea dependenței dintre două semnale (blocul XZ Graph); transferul unor parametri din schema bloc Simulink în spațiul de lucru MATLAB (blocul To Workspace).



**Figura 10.5.** Blocurile specifice bibliotecii Sinks.

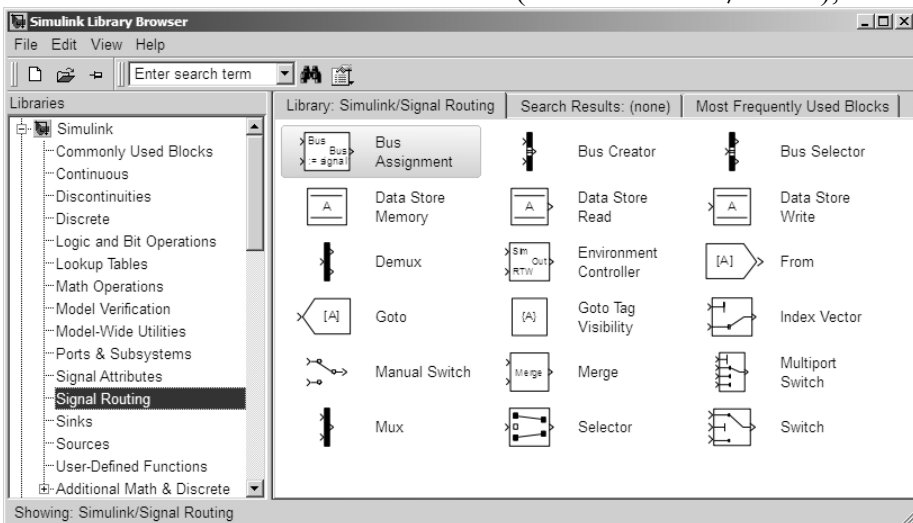
- Biblioteca Math Operations, figura 10.6, [8]. Conține blocuri pentru:
  - Efectuarea operațiilor matematice de bază: adunare și scădere (blocurile Add, Subtract, Sum, Sum of elements); înmulțire și împărțire (blocurile Product, Divide, Product of Elements); factor de multiplicare (blocul Gain); produs scalar (blocul Dot Product).
  - Funcții matematice elementare: valoare absolută (blocul Abs); funcția exponențială, logaritm natural, logaritm în baza 10, puterea lui 10, putere, rădăcină pătrată, reciproca rădăcinii pătrate, ridicare la pătrat, conjugare complexă, etc. (blocul Math function); funcția semn (blocul Sign); funcția de negativare (blocul Unary minus).
  - Funcții trigonometrice directe și inverse; funcții hiperbolice directe și inverse (blocul Trigonometric function).
  - Funcția sinus, (blocul Sine wave function).
  - Determinarea părții reale și imaginare a numerelor complexe (blocul Complex to real-imag).
  - Determinarea modulului și argumentului numerelor complexe (blocul Complex to magnitude-angle).
  - Determinarea punctelor de extrem ale unui semnal - minim și maxim (blocul MinMax).
  - Aproximarea numerelor prin metodele floor, ceil, round și fix (blocul Rounding function);
  - Evaluarea polinoamelor (blocul Polynomial).





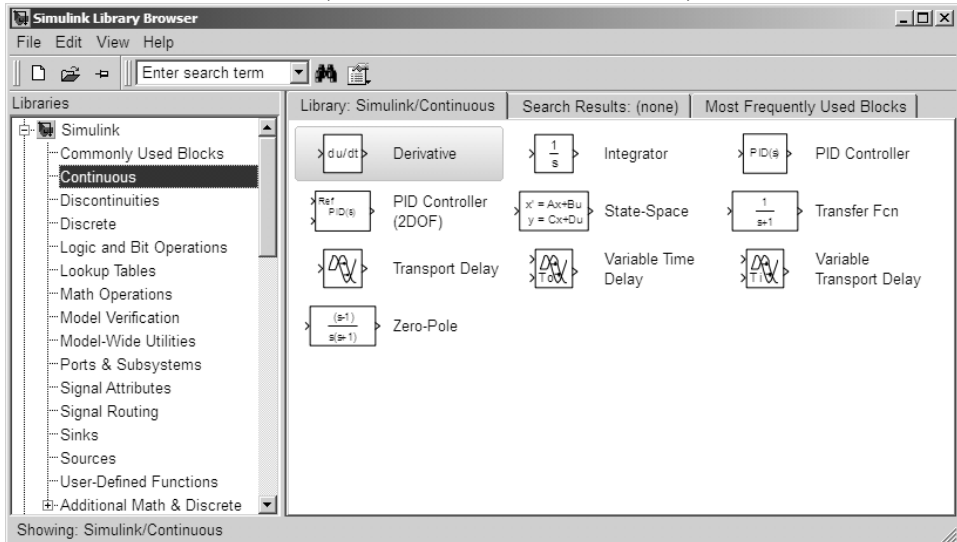
**Figura 10.6.** Blocurile specifice bibliotecii Math Operations.

- Biblioteca Signal Routing, figura 10.7, [9]. Conține blocuri utilizate pentru manipularea semnalelor: multiplexarea/demultiplexarea semnalelor (blocurile Mux/Demux); scrierea/citirea valorilor unor semnale în variabile (blocurile Goto/From), etc.



**Figura 10.7.** Blocurile specifice bibliotecii Signal Routing.

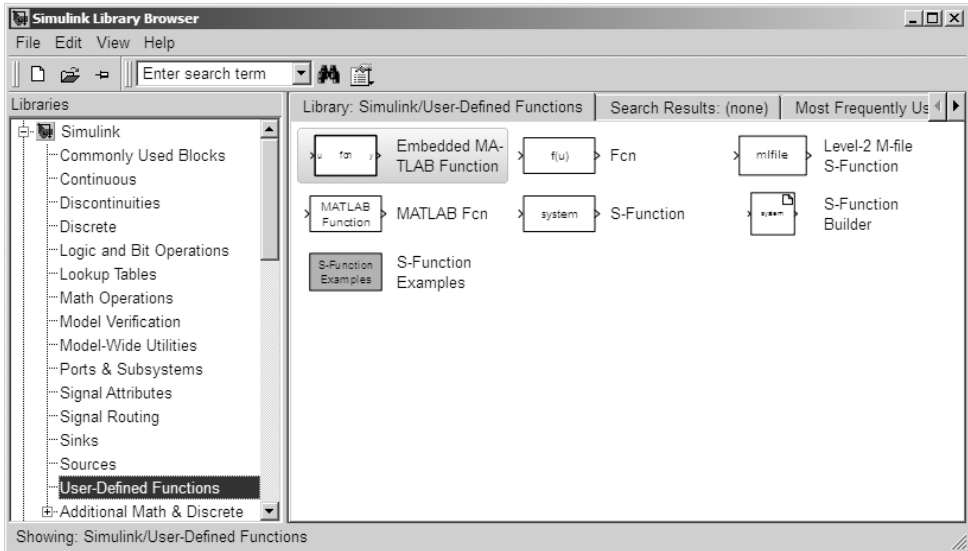
- Biblioteca Continuous, figura 10.8, [10]. Conține blocuri pentru modelarea și simularea sistemelor dinamice continue: integrarea/derivarea numerică (blocul Integrator/Derivative); modelarea sistemelor dinamice liniare prin funcții de transfer (blocul Transfer Fcn), prin metoda intrare-stare-ieșire (blocul State-Space), prin metoda zerouri-poli (blocul Zero-Pole); modelarea controlerelor PID (blocul PID Controller), etc.



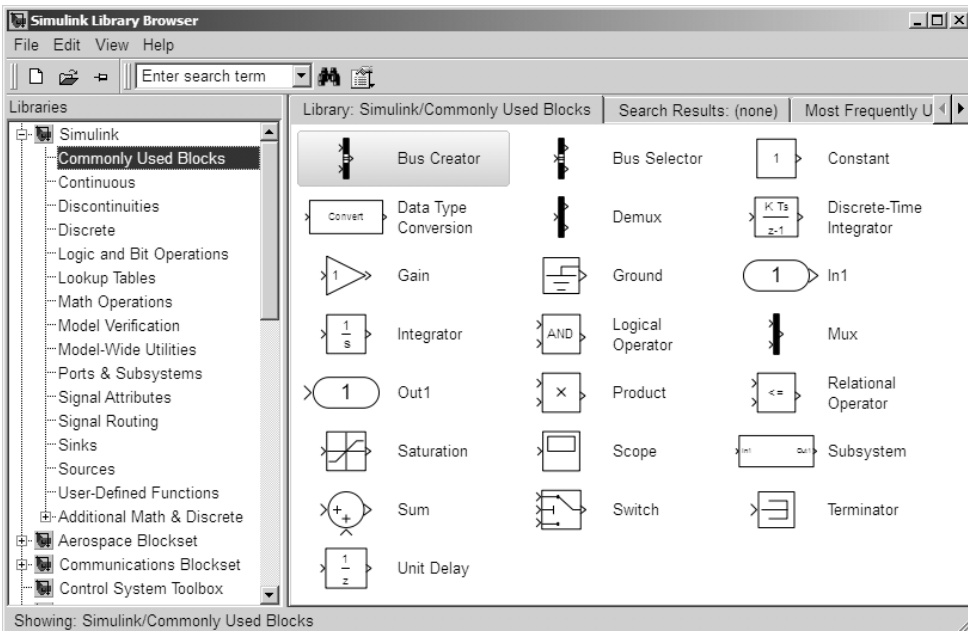
**Figura 10.8.** Blocurile specifice bibliotecii Continuous.

- Biblioteca User-Defined Functions, figura 10.9, [11]. Conține blocuri pentru definirea unor funcții speciale definite de utilizator: aplicarea funcțiilor MATLAB unor semnale de intrare (blocul MATLAB Fcn); aplicarea expresiilor MATLAB unor semnale de intrare (blocul Fcn); aplicarea funcțiilor definite de utilizator unor semnale de intrare (blocul Embedded MATLAB Function); generarea blocurilor Simulink definite de utilizator (blocurile S-Function și S-Function Builder); etc.
- Biblioteca Commonly Used Blocks, figura 10.10. Conține cele mai des utilizate blocuri: blocul Constant pentru specificarea valorilor de intrare constante; blocurile Sum, Product și Gain pentru realizarea operațiilor aritmetice simple (adunare, scădere, înmulțire, împărțire); blocul Scope pentru vizualizarea variației în timp a valorilor numerice ale semnalelor; blocul Integrator pentru integrarea semnalelor variabile în timp; blocurile In1, Out1 și Subsystem pentru crearea subsistemelor și manipularea semnalelor de intrare și de ieșire specifice subsistemelor; blocurile

Mux și Demux pentru multiplexarea și demultiplexarea semnalelor; blocul Switch pentru selectarea semnalelor; blocul Relational Operator pentru implementarea operatorilor relaționali între două semnale (identic, diferit, mai mic, mai mic sau egal, mai mare, mai mare sau egal); blocul Saturation pentru limitarea inferioară și superioară a unui semnal; etc.



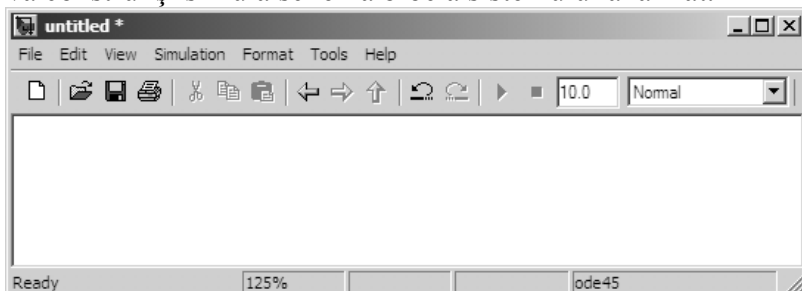
**Figura 10.9.** Blocurile bibliotecii User-Defined Functions.



**Figura 10.10.** Blocurile specifice bibliotecii Commonly Used Blocks.

### 10.2.3. Crearea, salvarea, deschiderea și printarea modelelor Simulink

Crearea unui nou model Simulink se realizează prin selectarea comenzii *File/New/Model* sau prin selectarea butonului *New model* din bara de butoane a programului. Se obține fereastra din figura 10.11 în care se va construi și simula schema bloc a sistemului analizat.



**Figura 10.11.** Fereastra de lucru a unui nou model.

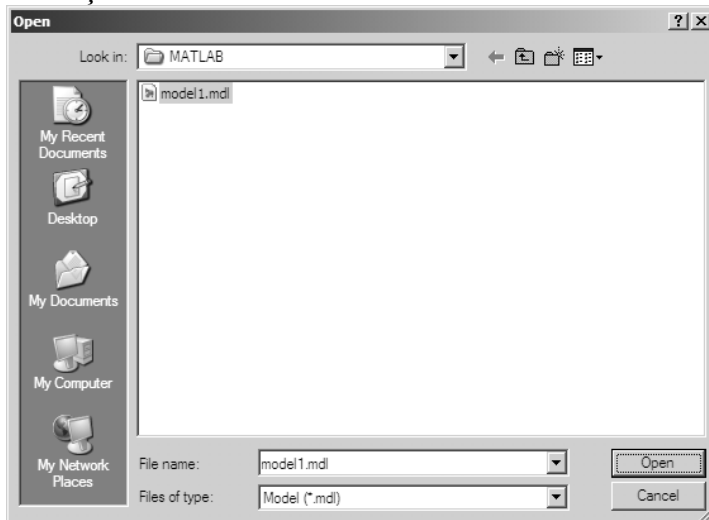
Salvarea unei scheme bloc se realizează prin selectarea comenzii *File/Save* sau prin selectarea butonului *Save* din bara de butoane a programului. Se obține fereastra din figura 10.12 în care se vor menționa atât numele modelului (*model1.mdl*), cât și directorul în care se va salva fișierul respectiv (în acest caz directorul implicit MATLAB). La prima salvare a unui model se va utiliza comanda *Save*.

La salvările ulterioare (pentru un model deja existent) se va utiliza fie comanda *Save* (care are ca efect înlocuirea conținutului fișierului cu ultima variantă modificată a schemei bloc), fie comanda *Save As* (care are ca efect salvarea modelului sub un alt nume sau într-un alt director).



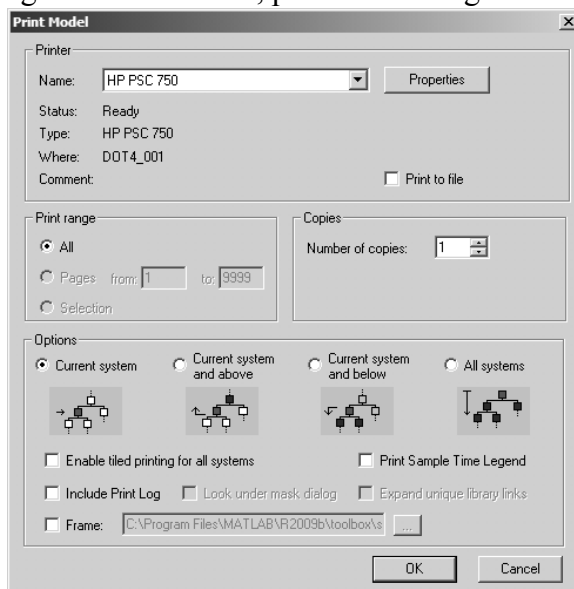
**Figura 10.12.** Fereastra pentru salvarea unui model.

Deschiderea unui model Simulink creat și salvat anterior se realizează prin selectarea comenzii File/Open sau prin selectarea butonului Open din bara de butoane a programului. Se obține fereastra din figura 10.13 în care se va deschide directorul corespunzător din care se va selecta numele fișierului dorit.



**Figura 10.13.** Fereastra pentru deschiderea unui model.

Printarea schemei bloc a unui model Simulink se realizează prin selectarea comenzii Print din meniul File sau prin selectarea comenzii Print din bara cu butoane. După lansarea comenzii Print se deschide fereastra de dialog Print Model, prezentată în figura 10.14.

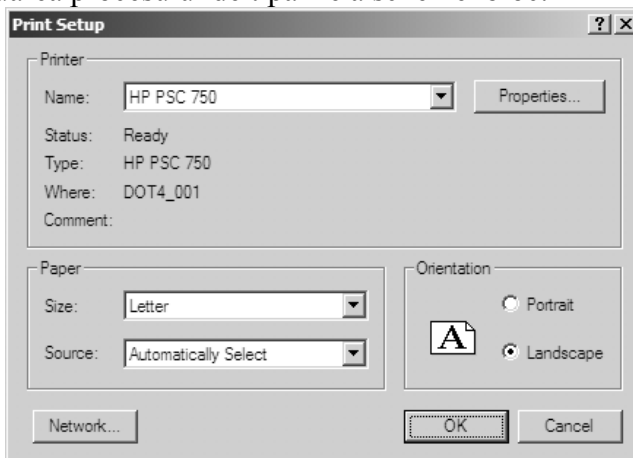


**Figura 10.14.** Fereastra de dialog Print Model.

Fereastra de dialog **Print Model** permite efectuarea unor setări specifice procesului de tipărire dintre care cele mai importante sunt:

- Alegerea și configurarea imprimantei (tipul imprimantei, dimensiunea și orientarea paginii, calitatea procesului de tipărire).
- Selectarea paginilor care vor fi tipărite și a numărului de copii.
- Alegerea sistemelor care vor fi tipărite:
  - Schema bloc curentă.
  - Schema bloc curentă și toate subsistemele plasate în ierarhia modelului deasupra schemei bloc curente.
  - Schema bloc curentă și toate subsistemele plasate în ierarhia modelului sub schema bloc curentă.
  - Toate sistemele modelului.

Configurarea imprimantei se poate realiza anterior lansării în execuție a comenzii **Print**, prin folosirea comenzii **Print Setup** din meniul **File**. Fereastra de dialog **Print Setup** este prezentată în figura 10.15. Comanda **Network** permite selectarea unei imprimante oarecare din rețeaua în care se găsește conectat sistemul de calcul de pe care se dorește efectuarea procesului de tipărire a schemei bloc.



**Figura 10.15.** Fereastra de dialog **Print Setup**.

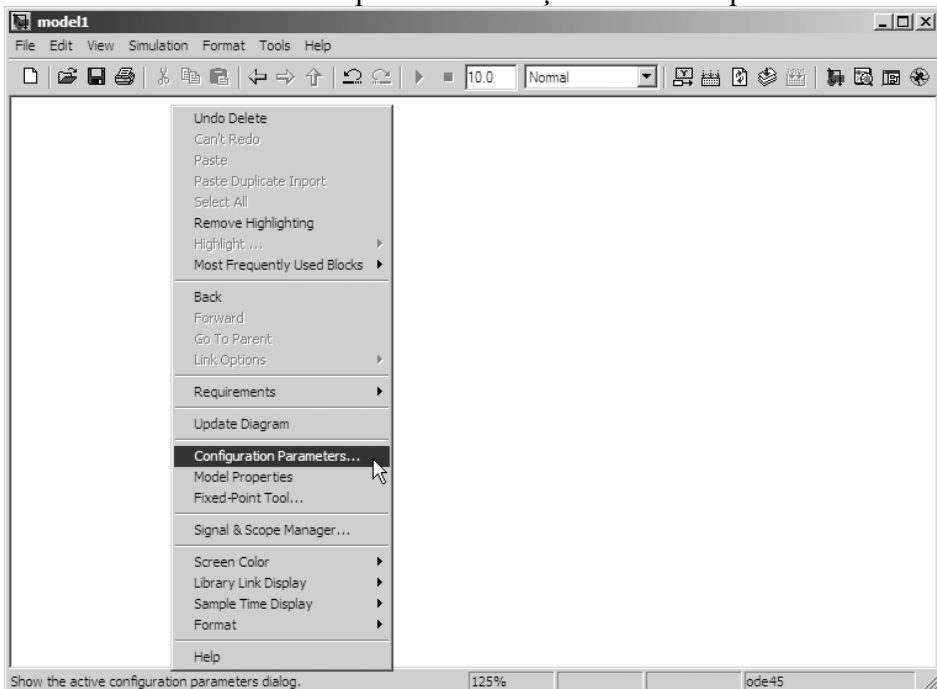
#### **10.2.4. Introducerea comenzilor Simulink**

Introducerea comenzilor într-un model Simulink se poate face prin mai multe metode:

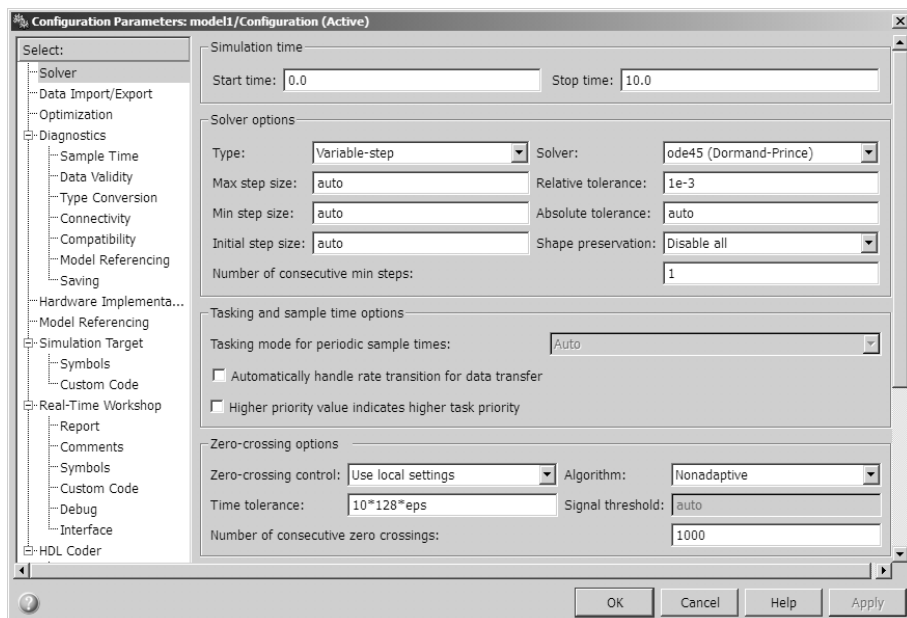
- Selectarea comenzii din bara de meniuri. Meniurile specifice ferestrei de editare și simulare a modelelor Simulink sunt: **File**, **Edit**, **View**, **Simulation**, **Format**, **Tools** și **Help**.
- Selectarea comenzii din bara cu butoane (toolbar). Toolbar-ul specific ferestrei de editare și simulare a modelelor Simulink include cele mai des utilizate comenzi, dintre se menționează: **New model**,

Open model, Save, Print, Cut, Copy, Paste, Navigate back, Navigate forward, Go to parent system, Undo, Redo, Start simulation, Stop simulation, Simulation stop time, Simulation mode, etc.

- Lansarea rapidă a comenzii cu ajutorul unor combinații de taste: Ctrl+O (Open model); Ctrl+S (Save model); Ctrl+P (Print); Ctrl+C (Copy); Ctrl+X (Cut); Ctrl+V (Paste); Ctrl+D (Update diagram); etc.
- Selectarea comenzii din meniul contextual care se deschide la efectuarea unui click cu butonul drept al mouse-ului (RMB). Comenzile din acest meniu contextual depind de obiectul selectat pe care s-a efectuat RMB. De exemplu, dacă se efectuează un RMB pe fundalul unei scheme bloc (zona liberă sau background) se deschide meniul contextual prezentat în figura 10.16. Din acest meniu contextual, de exemplu, se poate selecta apoi comanda Configuration Parameters prin care se pot modifica parametrii de configurare ai mediului de programare, figura 10.17, [12]. În momentul creării, fiecărui nou model îi sunt asociate în mod automat valorile implicite ale parametrilor de configurare. Modificările efectuate asupra diferiților parametri de configurare vor fi asociate doar schemelor bloc din structura modelului curent și trebuie făcute doar după buna cunoaștere a acestor parametri.



**Figura 10.16.** Meniul contextual asociat unei scheme bloc.



**Figura 10.17.** Fereastra de configurare a parametrilor modelului.

În cursul procesului de realizare a schemelor bloc, deseori apare necesitatea revenirii la o comandă anterioară. Parcurgerea în sens invers a listei comenzilor executate se realizează folosind comanda Undo. Această comandă permite revenirea la până la 100 de comenzi anterioare. Comenzile care pot fi anulate folosind comanda Undo fac parte din următoarele categorii de comenzi: adăugarea, ștergerea sau mutarea unui bloc, linie sau text; editarea numelui unui bloc; crearea unui subsistem.

### 10.2.5. Ferestrele Simulink

Principalele tipuri de ferestre specifice programului Simulink sunt:

- Fereastra principală, Simulink Library Browser (figura 10.3) care conține blocurile elementare Simulink grupate pe biblioteci.
- Fereastra de lucru, în care se creează schema bloc (figura 10.11). La partea inferioară a ferestrei de lucru se găsește bara de stare (Status Bar) care prezintă diferite informații referitoare la procesul de simulare: momentul de timp al simulării, solverul utilizat, etc. Activarea sau dezactivarea barei de stare se poate face din meniul View/Status Bar. Pentru vizualizare în întregime a unei scheme bloc complexe sau pentru vizualizarea unor detalii ale unei scheme bloc se utilizează diferite opțiuni ale meniului View: Zoom In, Zoom Out, Fit System to View, Normal. Varianta implicită de vizualizare este Fit System to View.



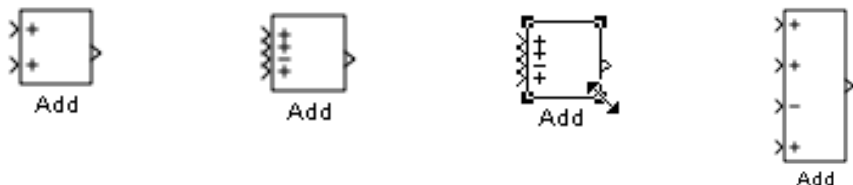
- Ferestrele de dialog, care permit modificarea parametrilor caracteristici ai unor comenzi sau blocuri Simulink (de exemplu fereastra Print Model, figura 10.14).
- Ferestrele pentru vizualizarea grafică a rezultatelor simulării generate de blocurile specifice Scope sau XY Graph.

### 10.2.6. Operații cu blocuri Simulink

Pe baza modelului analitic al sistemului fizic de analizat se construiește schema bloc cu ajutorul editorului de modele Simulink. Principalele operațiuni specifice fazei de construcție a schemelor bloc sunt:

- **Identificarea blocurilor** necesare pentru a realiza modelarea numerică a sistemului fizic prin analiza tipurilor de blocuri din fiecare bibliotecă de blocuri Simulink. În general, cele mai des utilizate blocuri se găsesc în bibliotecile: Sources (figura 10.4), Sinks (figura 10.5), Math Operations (figura 10.6), Signal Routing (figura 10.7), Continuous (figura 10.8), User-Defined Functions (figura 10.9), Ports & Subsystems. De altfel, Simulink grupează unele dintre cele mai des utilizate blocuri într-o bibliotecă specială denumită Commonly Used Blocks, (figura 10.10).
- **Introducerea blocurilor** respective în fereastra de lucru a modelului se realizează prin selectarea blocului cu click LMB pe suprafața blocului, menținerea apăsată a butonului mouse-ului și tragerea blocului deasupra spațiului de lucru al modelului, unde se eliberează butonul mouse-ului (metoda drag&drop cu butonul stâng al mouse-ului). În cazul în care într-o schemă bloc este nevoie de mai multe ori de același bloc, blocul respectiv se poate introduce de fiecare dată folosind tehnica drag&drop cu butonul stâng al mouse-ului din biblioteca Simulink în spațiul de lucru al modelului sau, imediat după prima introducere a blocului se poate utiliza tehnica Duplicate. Pentru acesta se efectuează drag&drop asupra blocului existent, dar cu butonul drept al mouse-ului. O altă metodă pentru multiplicarea unui bloc deja introdus în spațiul de lucru al modelului este metoda Copy-Paste.
- **Selectarea blocurilor** și poziționarea corespunzătoare a acestora. La selectarea diferitelor obiecte ale unei scheme bloc Simulink trebuie să se țină seama de următoarele observații:
  - Pentru a selecta un singur obiect, existent în fereastra de lucru a modelului, se efectuează LMB pe obiectul respectiv. Obiectul odată selectat, va fi încadrat automat într-un chenar având în cele patru colțuri câte un mic dreptunghi negru.

- Pentru a selecta mai multe obiecte se efectuează Shift+LMB pe obiectele respective sau se realizează cu ajutorul mouse-ului un dreptunghi de selecție în jurul obiectelor respective. În cazul dreptunghiului de selecție trebuie precizat faptul că din mulțimea elementelor selectate vor face parte atât obiectele conținute în totalitate în interiorul conturului de selecție, cât și obiectele doar parțial incluse în interiorul conturului dreptunghiular de selecție.
- Pentru a selecta toate obiectele existente în fereastra de lucru a modelului se utilizează comanda Edit/Select All.
- **Redimensionarea blocurilor** este operațiunea de mărire/micșorare a dimensiunii unui bloc având în vedere două aspecte: accesul ușor la porturile de intrare/ieșire și obținerea unor linii de conexiune cât mai puțin segmentate. În figura 10.18, a) se prezintă blocul destinat realizării operației matematice de adunare (Add, biblioteca Math Operations). În mod implicit, acest bloc are două porturi de intrare pentru introducerea celor două semnale care se vor aduna și un port de ieșire pentru transmiterea rezultatului obținut. În unele cazuri, există mai multe semnale de intrare, de exemplu 4 semnale (figura 10.18, b). Modificarea numărului de semnale se realizează prin efectuarea unui dublu click LMB pe blocul Add și deschiderea astfel a ferestrei de configurare a parametrilor caracteristici ai blocului, după care se modifică lista semnelor în caseta de control List of Signs (în acest caz lista semnelor este ++-+). Numărul semnalelor de intrare poate fi deci crescut, însă odată cu aceasta nu se modifică în mod corespunzător și dimensiunea blocului așa încât accesul spre porturile de intrare ale blocului devine dificil și imprecis. Pentru eliminarea acestui dezavantaj se impune mărirea dimensiunii blocului. Se selectează blocul, se poziționează cursorul mouse-ului pe unul de punctele negre existente în cele patru colțuri ale blocului astfel încât cursorul să ia forma unei săgeți duble (figura 10.18, c) și se aplică tehnica drag&drop cu butonul stâng al mouse-ului până se obține dimensiunea dorită (figura 10.18, d)



a) blocul inițial

b) modificarea numărului de porturi de intrare

c) modificarea dimensiunii blocului

d) blocul modificat

**Figura 10.18.** Redimensionarea blocurilor.

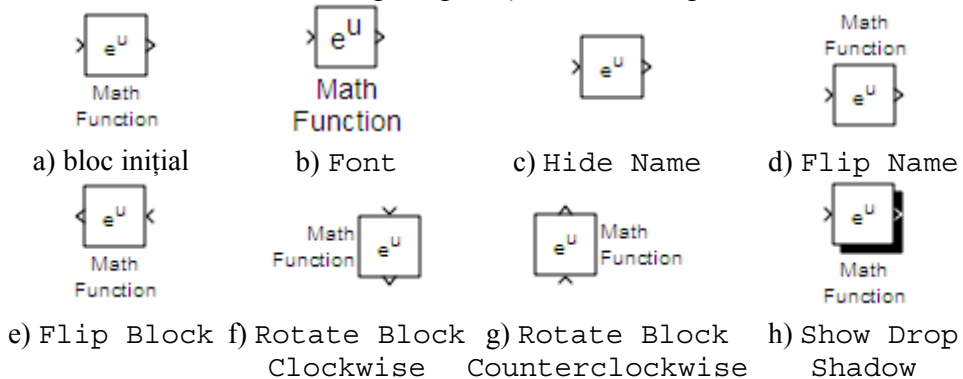
- **Conectarea blocurilor** cu linii de comunicație urmărind atât respectarea condiției funcționale, cât și a condiției ca liniile de conexiune, pe cât posibil, să nu se intersecteze accidental. Pentru respectarea celei de-a doua condiții, blocurile se pot re poziționa în mod convenabil prin deplasare și rotire. O altă modalitate prin care se poate evita apariția intersecțiilor accidentale ale liniilor de conexiune este utilizarea tehnicii `GoTo-From` pentru mărimile de intrare și eventual, pentru mărimile intermediare ale modelului analitic al sistemului fizic de analizat.

Conectarea blocurilor se poate face în mod automat sau manual. Conectarea automată se realizează prin selectarea blocului sursă și apoi prin apăsarea tastei `Ctrl` și selectarea blocului destinație. Conectarea manuală a blocurilor presupune parcurgerea mai multor etape: se poziționează cursorul pe portul de ieșire al blocului sursă astfel încât cursorul de tip săgeată să ia forma unei cruci; se apasă și se menține apăsat `LMB`; se deplasează cursorul peste portul de intrare al blocului destinație astfel încât cursorul să ia forma unei cruci duble; se eliberează butonul mouse-ului.

Realizarea liniilor de conexiune dintre o linie de conexiune existentă și portul de intrare al unui bloc (realizarea unei ramificații) presupune parcurgerea următoarelor etape: se poziționează cursorul mouse-ului pe linia de pe care va porni ramificația; se apasă tasta `Ctrl` și simultan `LMB`; se menține apăsat butonul mouse-ului timp în care se mută cursorul spre portul de intrare al blocului destinație până când cursorul cruce se transformă într-un cursor de tip cruce dublă, moment în care se eliberează butonul mouse-ului. Operațiunea se poate realiza și fără apăsarea tastei `Ctrl`, caz în care în loc de apăsarea `LMB` se va apăsa `RMB`. O altă modalitate de realizare a liniilor de ramificație este conectarea inversă, pornind de la portul de intrare al blocului destinație către linia de comunicație pe care se va conecta linia de ramificație. Indiferent însă de metoda folosită, intersecția dintre linia de ramificație și linia de conexiune existentă este simbolizată cu un punct.

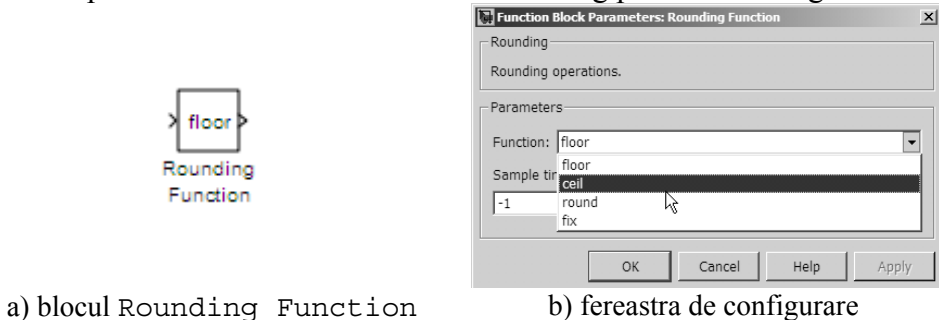
- **Formatarea blocurilor** constă în efectuarea unor operațiuni care au ca scop, în principal, îmbunătățirea aspectului grafic al informațiilor asociate blocurilor Simulink. Pentru aceasta se efectuează un `RMB` pe suprafața blocului respectiv și din meniul contextual se selectează comanda `Format`. Principalele opțiuni ale comenzii `Format` asociate blocurilor Simulink sunt: modificarea fontului (`Font`), figura 10.19, b); ascunderea numelui blocului (`Hide Name`), figura 10.19, c); re poziționarea numelui de partea cealaltă a blocului (`Flip`

Name), figura 10.19, d); rotirea blocului cu 180° (Flip Block sau Ctrl+I), figura 10.19, e); rotirea blocului cu 90° în sensul acelor de ceasornic (Rotate Block Clockwise sau Ctrl+R), figura 10.19, f), sau în sens trigonometric (Rotate Block Counterclockwise), figura 10.19, g); umbrirea blocului (Show Drop Shadow), figura 10.19, h). Tot din categoria comenzilor de formatare a blocurilor fac parte și comenzile pentru specificarea culorilor pentru contur și text (Foreground Color), respectiv pentru fondul blocurilor (Background Color). Aceste ultime două comenzi se găsesc tot în meniul contextual care apare la efectuarea unui RMB pe suprafața blocului respectiv.



**Figura 10.19.** Formatarea blocurilor.

- **Modificarea parametrilor caracteristici ai blocurilor** în conformitate cu specificațiile numerice ale ecuațiilor care guvernează comportamentul sistemului fizic analizat. Se efectuează dublu click LMB pe suprafața blocului respectiv, deschizându-se astfel fereastra de dialog care permite modificarea parametrilor caracteristici ai blocului. Numărul și tipul parametrilor caracteristici depind de tipul blocului. Pentru blocul de aproximare a numerelor (blocul Rounding Function, [13]), alegerea funcției de aproximare se face din fereastra de dialog prezentată în figura 10.20.



**Figura 10.20.** Parametrii caracteristici ai blocurilor.

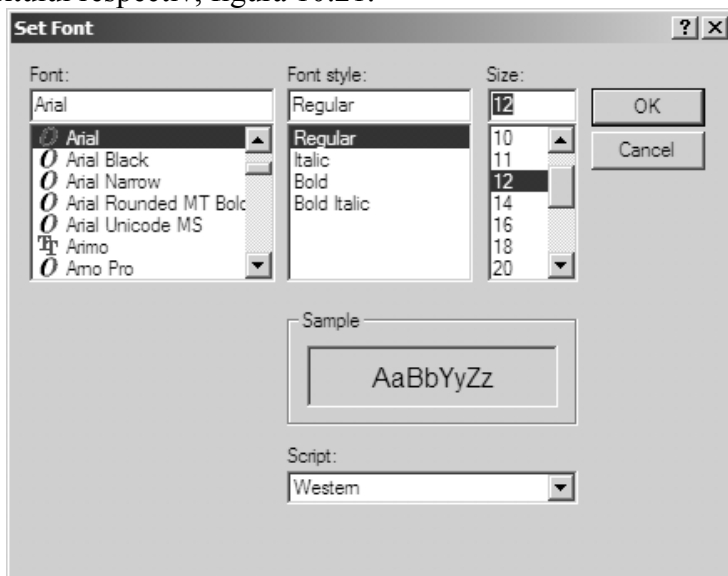
### 10.2.7. Introducerea textelor explicative într-o schemă bloc

#### Simulink

În vederea documentării unei scheme bloc se poate utiliza metoda textului explicativ introdus direct în spațiul de lucru al modelului.

Principalele elemente specifice procesului de introducere a textelor explicative în cadrul schemelor bloc Simulink sunt:

- Se efectuează dublu click cu LMB pe o zonă liberă a schemei bloc deschizându-se astfel o zonă specifică introducerii textelor explicative. Se observă și forma specifică a cursorului, caracteristică introducerii textului.
- Se introduce textul explicativ la cursorul din interiorul zonei de text, linie după linie, folosind tasta `Enter` ca separator de linii.
- Textul explicativ se poate muta în altă poziție prin tehnica `drag&drop` cu butonul stâng al mouse-ului aplicată frame-ului textului respectiv.
- Textul explicativ se poate modifica ulterior prin efectuarea unui click cu LMB în zona de text și apoi prin ștergerea sau adăugarea de text.
- Textul explicativ poate fi formatat prin lansarea comenzii `Font` din meniul `Format` sau din meniul contextual care se deschide la efectuarea unui click RMB pe suprafața textului respectiv. Se deschide fereastra `Set Font` care permite: modificarea fontului, a stilului de scriere și a dimensiunii caracterelor utilizate la scrierea textului respectiv, figura 10.21.



**Figura 10.21.** Modificarea parametrilor fontului utilizat la scrierea textelor explicative.

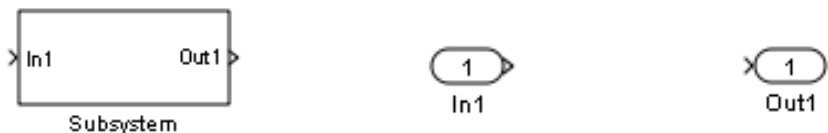
Tot din categoria comenzilor de formatare a textului explicativ fac parte și comenzile pentru specificarea culorilor pentru contur și text (Foreground Color), respectiv pentru fondul textului (Background Color). Aceste ultime două comenzi se găsesc tot în meniul contextual generat de efectuarea unui RMB pe suprafața textului respectiv.

### 10.2.8. Crearea subsistemelor Simulink

Crearea subsistemelor în cadrul unei scheme bloc Simulink este o metodă utilizată în general pentru simplificarea schemei bloc, [14]. Principalele avantaje ale utilizării subsistemelor sunt: reducerea numărului de blocuri din schema bloc principală; gruparea din punct de vedere funcțional a mai multor blocuri în subsisteme; protejarea conținutului unor subsisteme față de eventuale modificări neautorizate prin acordarea dreptului de citire/scriere (ReadWrite, ReadOnly, NoReadOrWrite); realizarea schemelor bloc cu structură ierarhică.

Crearea subsistemelor se poate face prin două metode:

- Introducerea blocului Subsystem din biblioteca Port & Subsystems și apoi crearea subsistemului respectiv. În figura 10.22, a) se prezintă blocul Subsystem, iar în figurile 10.22, b) și 10.22, c) se prezintă blocurile destinate specificării semnalelor de intrare (blocul In) și de ieșire (blocul Out) ale subsistemului. Modificarea numărului de semnale de intrare și de ieșire se realizează prin multiplicarea corespunzătoare a blocurilor In și Out din structura subsistemului. Denumirea semnalelor de intrare și de ieșire ale unui subsistem se poate modifica prin modificarea denumirii fiecărui bloc de tip In și Out din structura subsistemului.



a) bloc subsistem

b) bloc pentru semnale de intrare

c) bloc pentru semnale de ieșire

**Figura 10.22.** Blocuri specifice subsistemelor.

- Crearea întregii scheme bloc și apoi transformarea diferitelor regiuni ale schemei în subsisteme. În acest scop se selectează blocurile dorite, se efectuează un click RMB pe zona selectată și apoi din meniul contextual se selectează comanda Create Subsystem. Selectarea comenzii Create Subsystem se poate face și din meniul Edit.

### Problema 10.1

Se consideră mișcarea cu suprafață liberă a apei într-un canal cu secțiune transversală dreptunghiulară având lățimea  $b=3$  m și adâncimea normală  $h_0=0,75$  m.

Să se realizeze o schemă bloc Simulink pentru calculul razei hidraulice a canalului cunoscând relațiile de calcul pentru:

- aria secțiunii vii a apei:

$$A = b \cdot h_0$$

- perimetrul udat:

$$P = b + 2 \cdot h_0$$

- raza hidraulică:

$$R = A/P$$

### Rezolvare

Din analiza modelului analitic al sistemului fizic analizat rezultă următoarele observații:

- Există două mărimi de intrare ( $b=3$  și  $h_0=0,75$ ) pentru introducerea cărora se vor folosi două blocuri de tip Constant [15], (biblioteca Sources).
- Există două relații algebrice pentru calculul unor mărimi intermediare ( $A = b \cdot h_0$  și  $P = b + 2 \cdot h_0$ ) pentru modelarea cărora se vor folosi două blocuri Product [16], și un bloc Add [17], (biblioteca Math Operations).
- Există o relație algebrică pentru calculul mărimii de ieșire ( $R = A/P$ ) pentru modelarea căreia se va folosi blocul Divide [18], (biblioteca Math Operations).
- Modelul analitic analizat nu conține mărimi variabile în timp ci doar mărimi constante, prin urmare pentru vizualizarea rezultatelor numerice ale calculelor se vor folosi doar blocuri de tip Display [19], (biblioteca Sinks).

Principalele etape ale realizării schemei bloc sunt:

- Se deschide un nou model Simulink fie prin selectarea comenzii New/Model din meniul File, fie prin selectarea comenzii New model din bara de butoane a programului, figura 10.23, a). Dimensiunea ferestrei în care se va construi schema bloc se va modifica în cursul procesului de realizare a modelului în funcție de complexitatea schemei.
- Se introduc două blocuri de tip Constant din biblioteca Sources, figura 10.23, b). Se poziționează cele două blocuri Constant prin selectarea și mutarea acestora în poziții

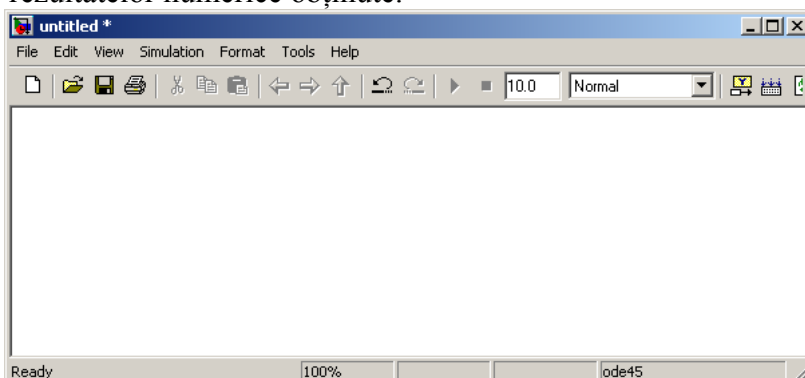
convenabile. La poziționarea blocurilor trebuie să se respecte mai multe condiții: funcționalitatea schemei, simetria blocurilor față de operatori; realizarea unei scheme bloc cât mai compacte.

- Se modifică denumirile celor două blocuri de tip Constant. Denumirile implicite ale celor două blocuri sunt Constant și Constant1. Se efectuează click cu LMB în zona textului explicativ al fiecărui bloc și se modifică denumirea în conformitate cu semnificația mărimilor de intrare ( $b$  și  $h_0$ ), figura 10.23, c).
- Se modifică parametrii caracteristici ai celor două blocuri Constant. Pentru aceasta se efectuează câte un dublu click cu LMB pe fiecare din cele două blocuri Constant, modificându-se valoarea numerică a fiecărui bloc în ferestre de dialog corespunzătoare, în conformitate cu valorile numerice  $b=3$  și  $h_0=0,75$ , figura 10.23, d).
- Se introduce blocul Product din biblioteca Math Operations, pentru a realiza operația de înmulțire, figura 10.23, e).
- Se conectează ieșirile celor două blocuri Constant cu cele două intrări ale blocului Product, figura 10.23, f).
- Se introduce blocul Display din biblioteca Sinks. Se modifică denumirea blocului Display în conformitate cu semnificația fizică a mărimii care rezultă prin înmulțirea lățimii  $b$  cu adâncimea normală  $h_0$  (aria secțiunii vii a apei A). Se conectează ieșirea blocului Product cu intrarea blocului Display. Se lansează în execuție procesul de simulare al modelului prin selectarea comenzii Start Simulation din bara cu butoane. Se observă valoarea numerică obținută în blocul Display, figura 10.23, g).
- Pentru calculul perimetrului udat, conform relației  $P = b + 2 \cdot h_0$ , se vor introduce trei noi blocuri: un bloc Gain pentru realizarea operației  $2 \cdot h_0$ , un bloc Add pentru realizarea operației de adunare și un bloc Display pentru vizualizarea rezultatului obținut. Parametrul caracteristic al blocului Gain trebuie modificat în conformitate cu factorul de amplificare având în acest caz valoarea 2. Denumirea noului bloc Display trebuie modificată în P. Se realizează conectarea corectă a acestor blocuri rezultând schema bloc din figura 10.23, h).
- Pentru calculul razei hidraulice conform relației  $R = A/P$  se vor introduce două noi blocuri: un bloc Divide pentru realizarea operației de împărțire și un bloc Display pentru vizualizarea rezultatului obținut. Denumirea noului bloc Display trebuie modificată în R. Se realizează conectarea corectă a acestor blocuri

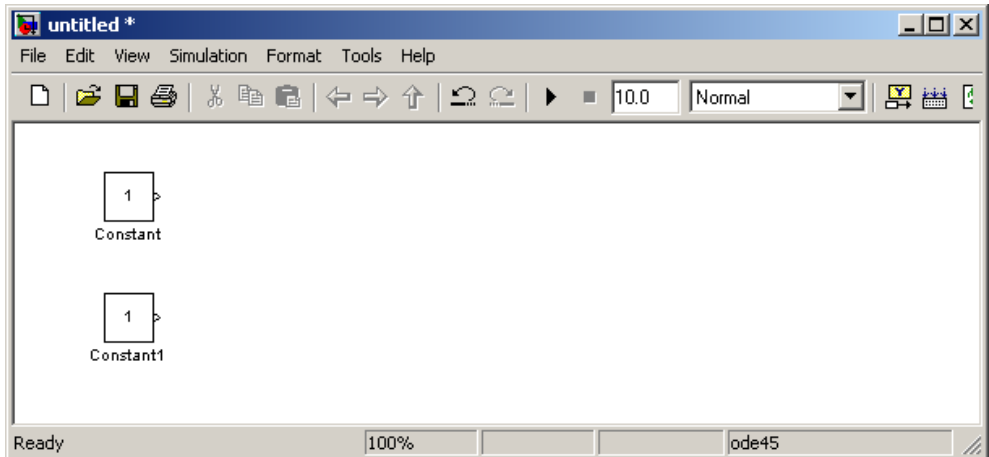


rezultând schema bloc din figura 10.23, i). Se observă intersectarea accidentală a două linii de conexiune, situație care ar trebui evitată. Din păcate, în cazul acestei scheme bloc, așa cum a fost concepută, nu se poate realiza operație de calcul a razei hidraulice fără apariția unei intersecții accidentale a liniilor de conexiune.

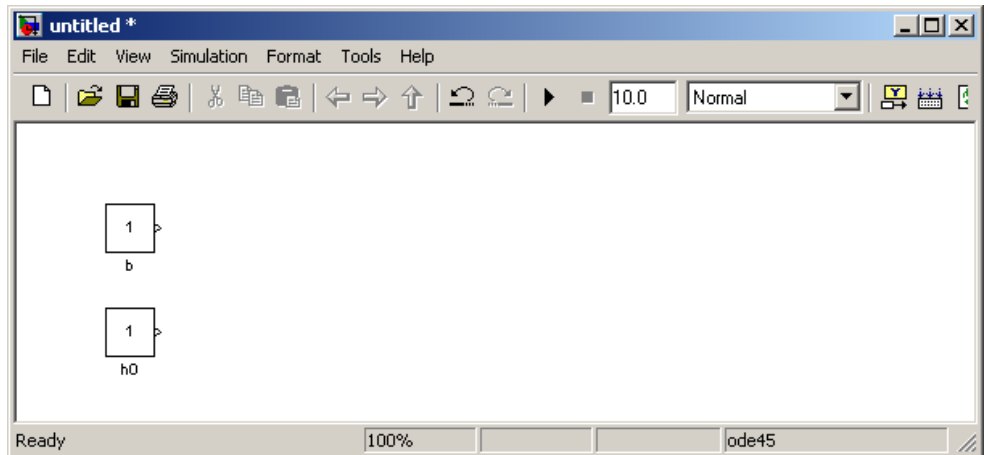
- Evitarea intersecției accidentale a liniilor de conexiune se poate realiza totuși și în acest caz prin utilizarea tehnicii Goto-From, figura 10.23, j). Pentru aceasta atât valorile numerice ale datelor de intrare ( $b$  și  $h_0$ ), cât și ale mărimilor intermediare ( $A$  și  $P$ ) și pentru acest caz, chiar și a mărimii de ieșire ( $R$ ), se vor memora în variabile de tip global asociate schemei bloc folosind blocul Goto [20], (biblioteca Signal Routing). După aceasta, ori de câte ori este nevoie de aceste mărimi, se vor utiliza blocurile corespunzătoare From [21], (biblioteca Signal Routing). La configurarea blocurilor Goto se efectuează două operațiuni: definirea numelui variabilei în zona Goto tag și specificarea domeniului de vizibilitate al variabilei (global - vizibilitate în toate subsistemele schemei bloc indiferent de poziția acestora în structura ierarhică a schemei bloc, local - vizibilitate doar în subsistemul curent; scoped - vizibilitate în subsistemul curent și în toate subsistemele conținute în subsistemul curent). La configurarea blocurilor From se efectuează două operațiuni: actualizarea variabilelor definite în schema bloc în concordanță cu domeniul lor de vizibilitate (Update Tags) și selectarea numelui variabilei dorite din lista variabilelor vizibile în subsistemul curent (Goto Tag).
- În figura 10.23, k) se prezintă modul de utilizare a subsistemelor. Au fost create trei subsisteme: subsistemul Date inițiale pentru introducerea datelor inițiale; subsistemul Calcule pentru efectuarea calculelor și subsistemul Rezultate pentru prezentarea rezultatelor numerice obținute.



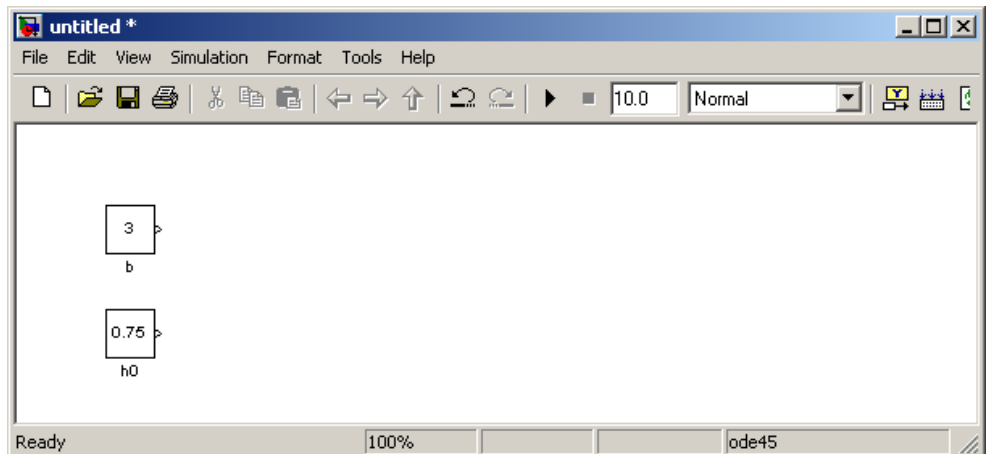
a) deschiderea unui nou model Simulink



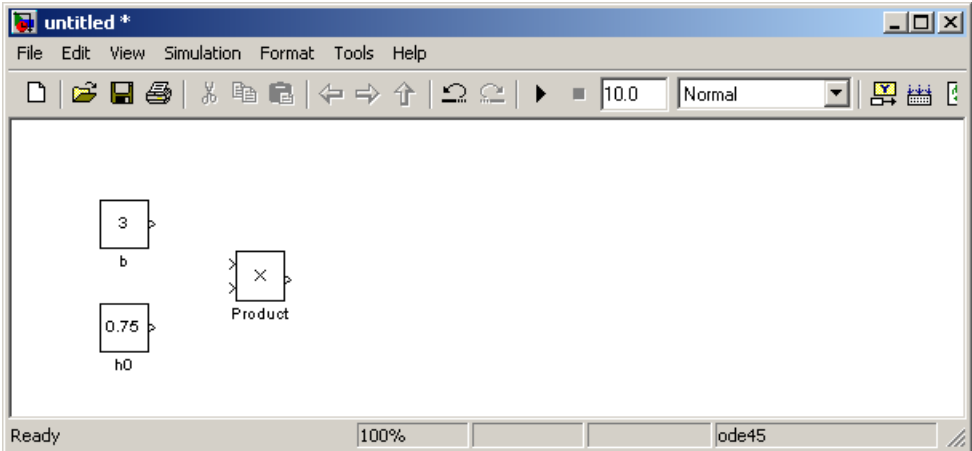
b) introducerea celor două blocuri de tip Constant



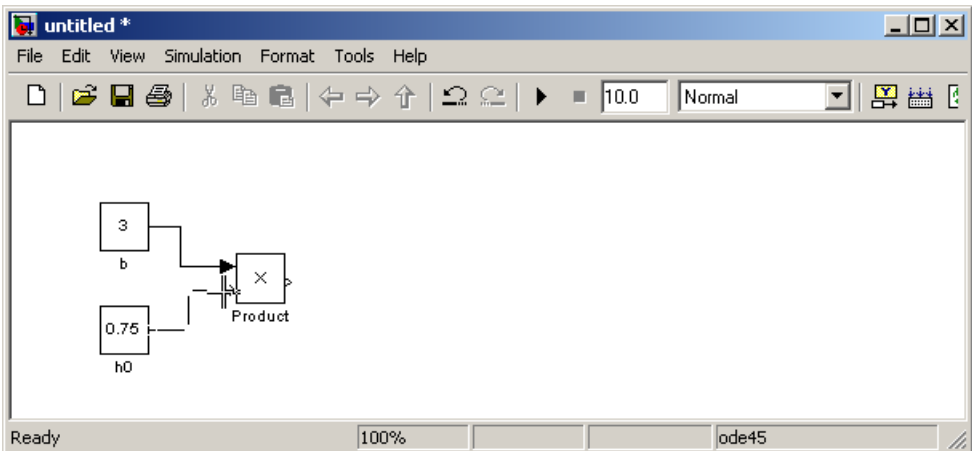
c) modificarea denumirii celor două blocuri de tip Constant



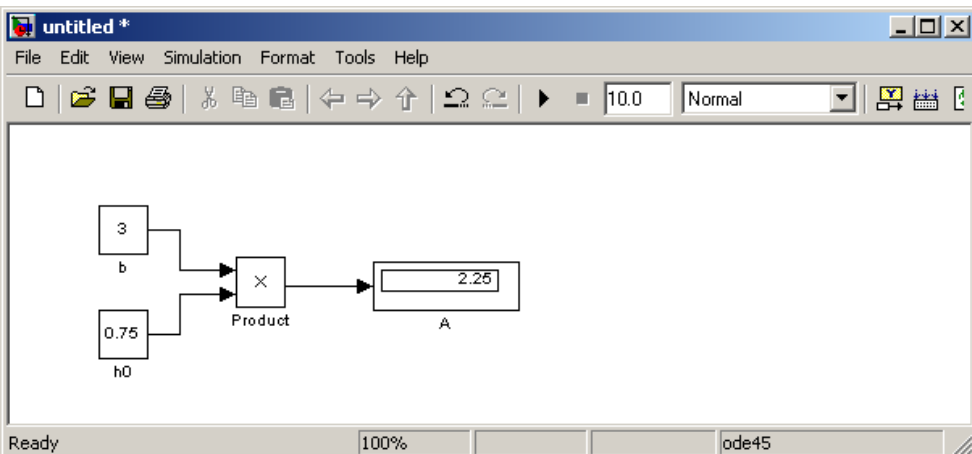
d) modificarea valorilor numerice ale celor două blocuri de tip Constant



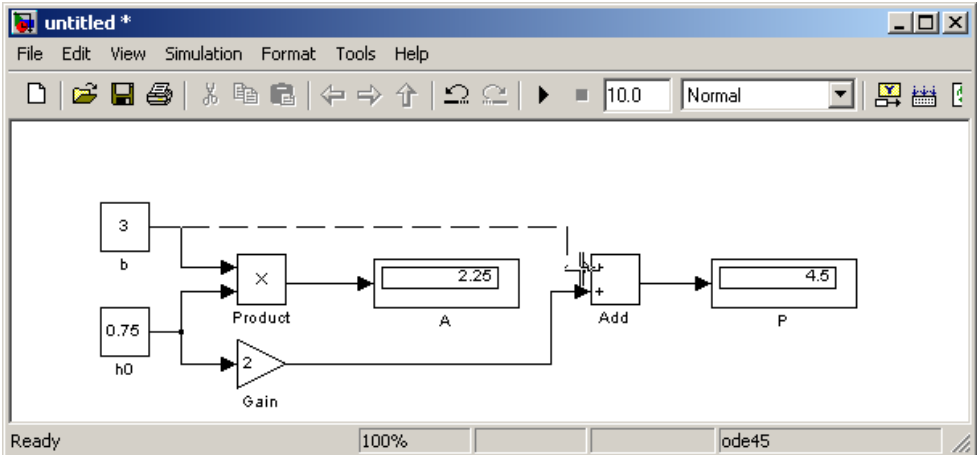
e) introducerea blocului Product



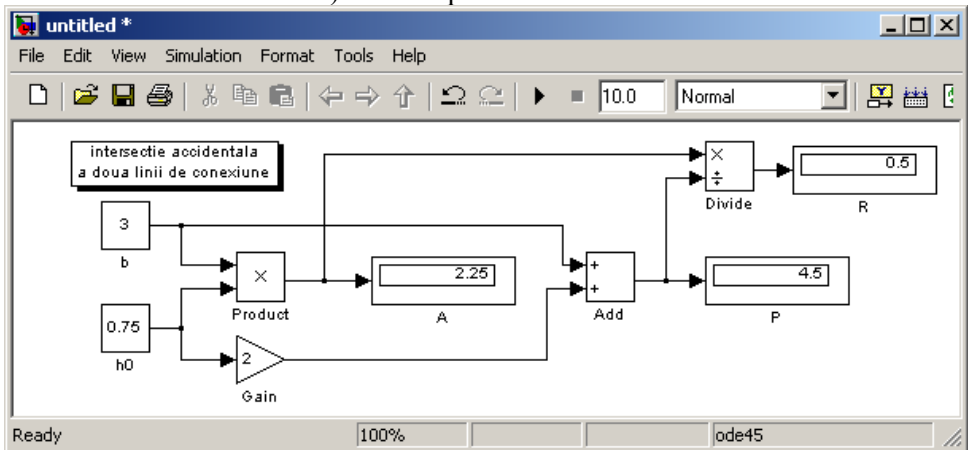
f) conectarea celor două blocuri Constant cu blocul Product



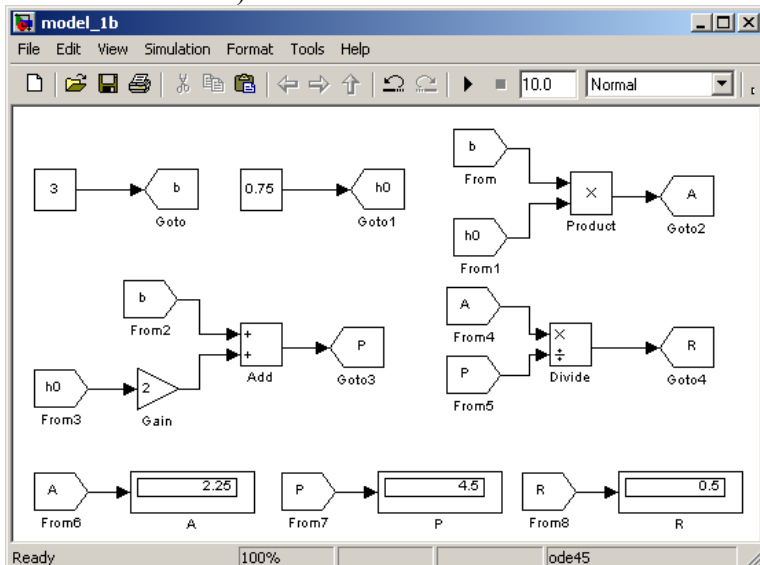
g) conectarea blocului Display și calculul ariei secțiunii vii a apei



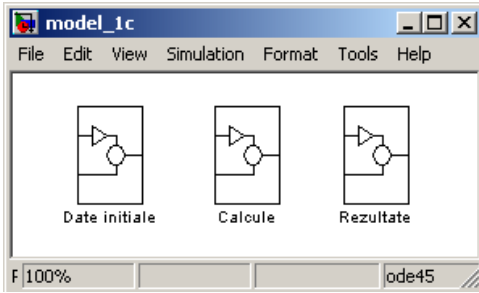
h) calculul perimetrului udat



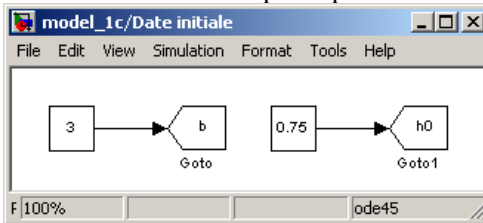
i) calculul razei hidraulice



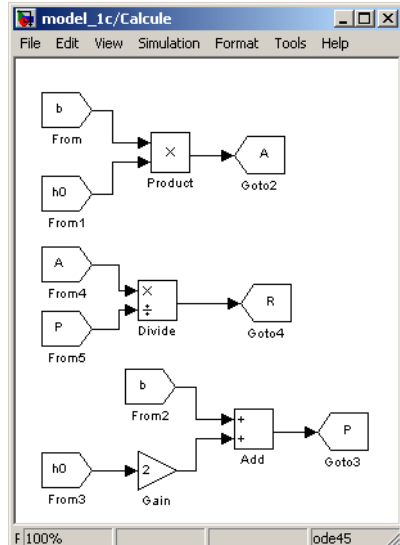
j) utilizarea tehnicii de lucru Goto-From



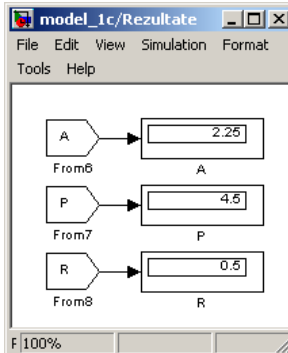
schema bloc principală



subsistemul Date inițiale



subsistemul Calcule



subsistemul Rezultate

k) utilizarea tehnicii de lucru bazată pe subsisteme

**Figura 10.23.** Realizarea schemei bloc pentru rezolvarea problemei 10.1.

### Observații

- În cazul schemelor bloc cu un mare număr de date inițiale, cu multe rezultate numerice intermediare se recomandă utilizarea tehnicii de lucru bazată de blocurile Goto-From. Pentru scheme cu o complexitate ridicată, din considerente de simplificare a schemei bloc principale și pentru a grupa diferitele etape ale algoritmului de rezolvare în blocuri unitare de calcule se recomandă în plus, utilizarea și a subsistemelor. Indiferent însă de complexitatea schemei bloc principale, dacă prin nici o altă metodă (amplasarea blocurilor în altă ordine, modificarea ordinii operațiilor aritmetice, rotirea blocurilor) nu se poate evita intersecția accidentală a liniilor de conexiune, atunci se impune utilizarea tehnicii de lucru bazată pe blocurile Goto-From și eventual, introducerea subsistemelor.

## Problema 10.2

Se consideră funcția sinusoidală definită prin:

$$y = y_0 + A \cdot \sin(\omega t + \varphi)$$

în care:  $y_0$  este ordonata în jurul căreia evoluează funcția sinus (bias);  $A$  este amplitudinea;  $\omega$  [rad/s] este pulsația sau frecvența unghiulară;  $\varphi$  [rad] este defazajul sau faza inițială la momentul  $t=0$  s;  $t$  [s] este timpul simulării.

Argumentul funcției sinusoidale (unghiul de fază), se exprimă prin:

$$\theta = \omega t + \varphi$$

Pulsația (frecvența unghiulară) se poate exprima și prin relația:  $\omega = 2\pi \cdot f$ , în care:  $f$  este frecvența exprimată în [1/s] sau [Hz].

Se cere:

a) Să se realizeze o schemă bloc pentru vizualizarea variației funcției sinusoidale pentru următorii parametri:  $y_0=0$ ;  $A=1$ ;  $\omega=1$ ;  $\varphi=0$ ;  $t=0 \div 2\pi$  s.

b) Să se modifice schema bloc inițială astfel încât să permită efectuarea următoarelor analize: influența bias-ului:  $y_0=\{0; 1\}$ ; influența amplitudinii:  $A=\{1; 2\}$ ; influența pulsației:  $\omega=\{1; 2\}$ .

## Rezolvare

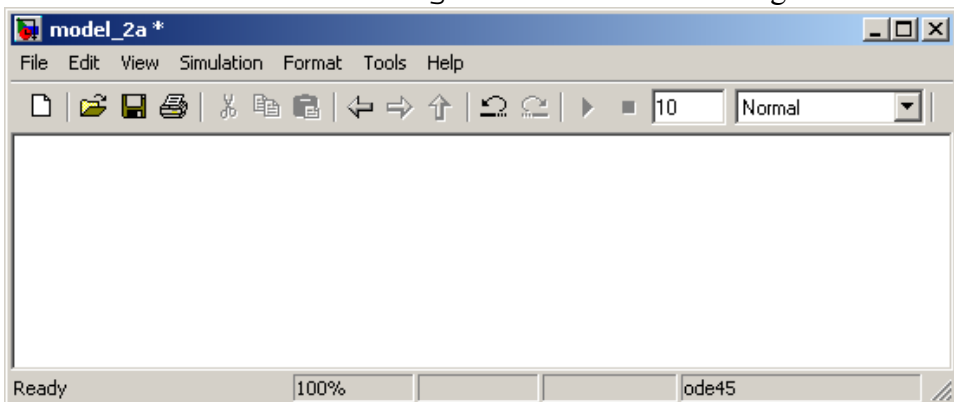
Din analiza modelului analitic al sistemului fizic analizat rezultă următoarele observații: există o singură mărime de intrare, care de altfel este și mărimea de ieșire a modelului-funcția sinusoidală pentru modelarea căreia se folosește blocul Sine Wave, (biblioteca Sources); modelul analitic analizat conține o mărime variabilă în timp pentru vizualizarea căreia se va folosi un bloc de tip Scope, (biblioteca Sinks).

Rezolvarea punctului a) presupune parcurgerea următoarelor etape:

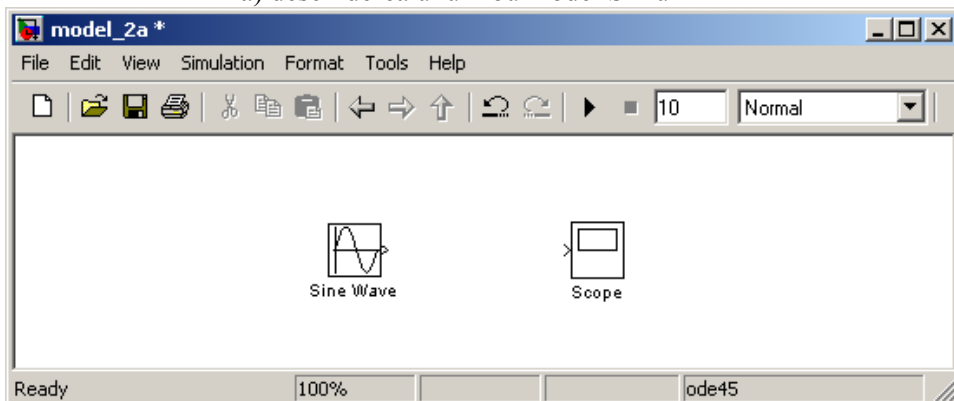
- Se deschide un nou model Simulink fie prin selectarea comenzii New/Model din meniul File, fie prin selectarea comenzii New model din bara de butoane a programului, figura 10.24, a). Dimensiunea ferestrei în care se va construi schema bloc se va modifica în cursul procesului de realizare a modelului în funcție de complexitatea schemei.
- Se introduc blocurile Sine Wave și Scope din bibliotecile Sources, respectiv Sinks, figura 10.24, b). Se poziționează cele două blocuri prin selectarea și mutarea acestora în poziții convenabile. La poziționarea blocurilor trebuie să se respecte mai multe condiții: funcționalitatea schemei, simetria blocurilor față de operatori; realizarea unei scheme bloc cât mai compacte.
- Se conectează cele două blocuri, figura 10.25, c). Conectarea manuală a blocurilor presupune parcurgerea mai multor etape: se poziționează cursorul pe portul de ieșire al blocului sursă (Sine

Wave) astfel încât cursorul de tip săgeată să ia forma unei cruci; se apasă și se menține apăsat LMB; se deplasează cursorul peste portul de intrare al blocului destinație (Scope) astfel încât cursorul să ia forma unei cruci duble; se eliberează butonul mouse-ului.

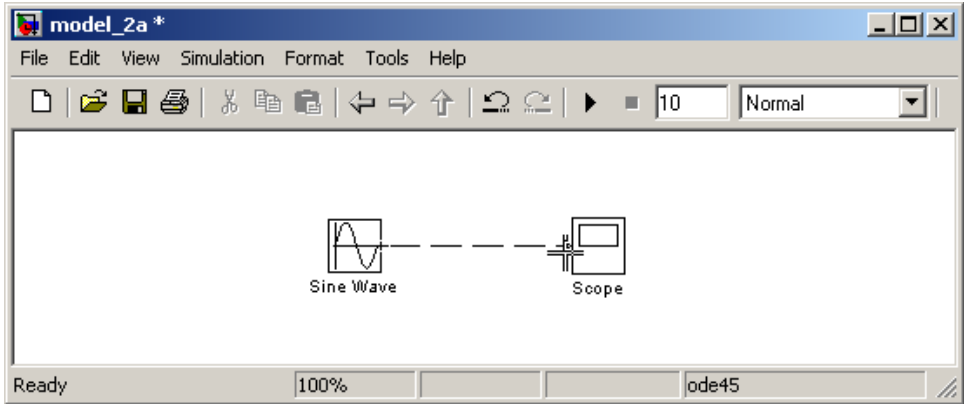
- Se modifică parametrii caracteristici ai blocului Sine Wave. Pentru aceasta se efectuează un dublu click cu LMB pe suprafața blocului, modificându-se valorile numerice ale parametrilor caracteristici în fereastra de dialog corespunzătoare, figura 10.24, d). Se modifică valoarea finală a timpului de simulare ( $t_f = 2\pi$  s) în caseta de control Simulation Stop Time din toolbar-ul modelului, figura 10.24, e).
- Se lansează în execuție simularea și se vizualizează în blocul Scope variația în timp a funcției sinusoidale analizate, figura 10.24, f). Scalarea corespunzătoare a ordonatei se realizează prin selectarea comenzii Autoscale din bara de butoane a ferestrei grafice. În cazul în care scalarea curentă a axelor corespunde și altor simulări se poate menține actuala scalare prin selectarea comenzii Save current axis settings din toolbar-ul ferestrei grafice.



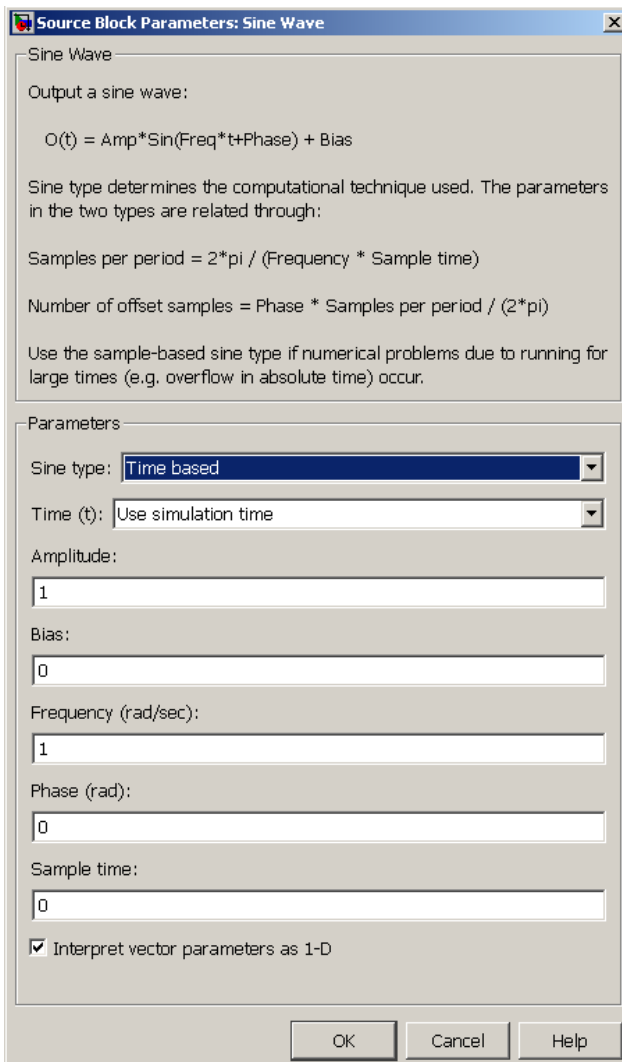
a) deschiderea unui nou model Simulink



b) introducerea celor două blocuri

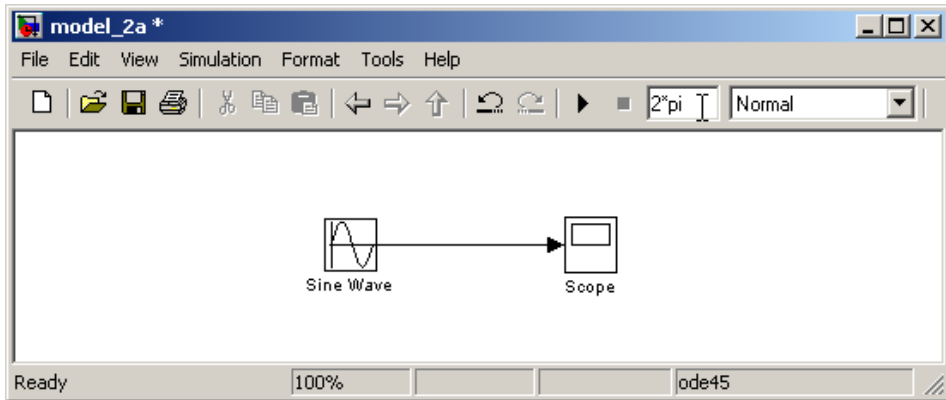


c) conectarea celor două blocuri

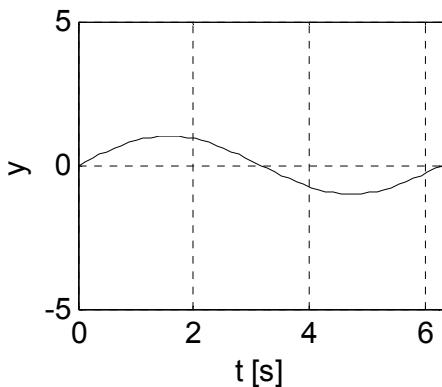


d) modificarea parametrilor caracteristici ai blocului Sine Wave

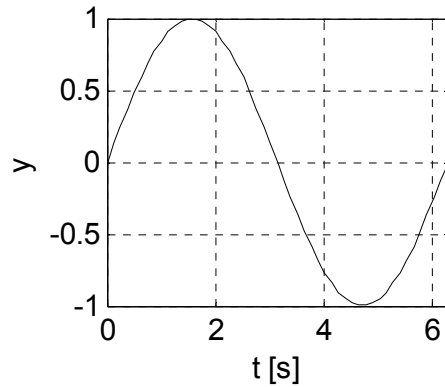




e) modificarea valorii finale a timpului de simulare



graficul inițial



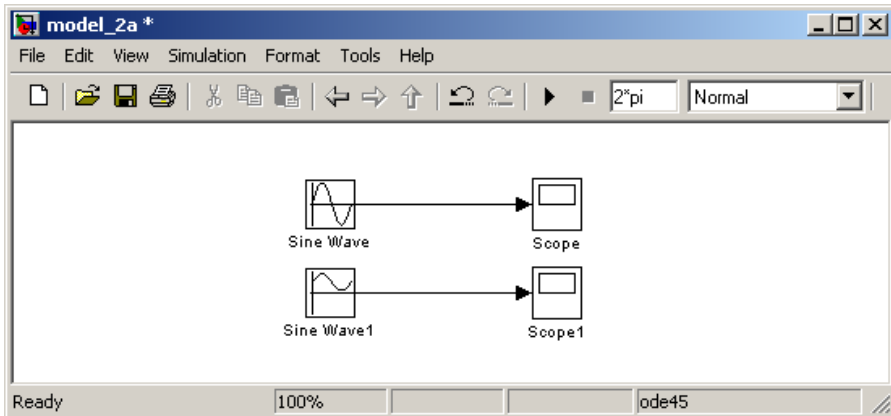
graficul scalat

f) vizualizarea variației în timp a funcției

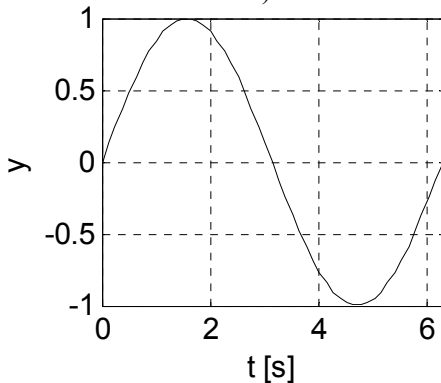
**Figura 10.24.** Realizarea schemei bloc pentru rezolvarea problemei 10.2, a).

Pentru rezolvarea punctului b), se prezintă trei variante:

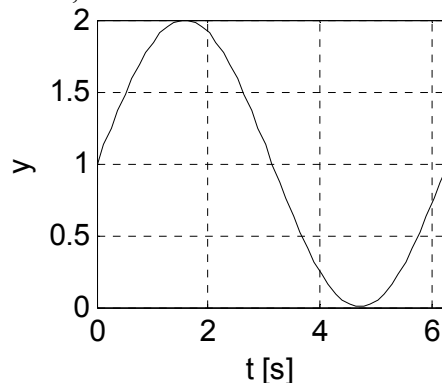
- **Varianta 1.** Se introduce în schema bloc un alt bloc Sine Wave și un alt bloc Scope care se conectează apoi între ele, figura 10.25, a). Aceste două noi blocuri vor avea în mod implicit alte nume: Sine Wave 1 și Scope 1. Parametrii blocului Sine Wave vor rămâne nemodificați, fiind considerați de referință, în timp ce parametrii blocului Sine Wave 1 se vor modifica în concordanță cu analiza influenței fiecărui parametru caracteristic. De exemplu, pentru acest caz se va considera influența bias-ului pentru modificarea căruia se deschide fereastra de configurare a blocului Sine Wave 1 și se introduce noua valoare a parametrului bias. După lansarea în execuție se obțin reprezentările grafice din figura 10.25, b), pentru cazul semnalului de referință (bias  $y_0=0$ ) și, respectiv din figura 10.25, c), pentru cazul semnalului modificat, (bias  $y_0=1$ ). Scalarea graficelor se realizează cu ajutorul comenzii Autoscale.



a) schema bloc modificată, varianta 1



b)  $y_0=0$



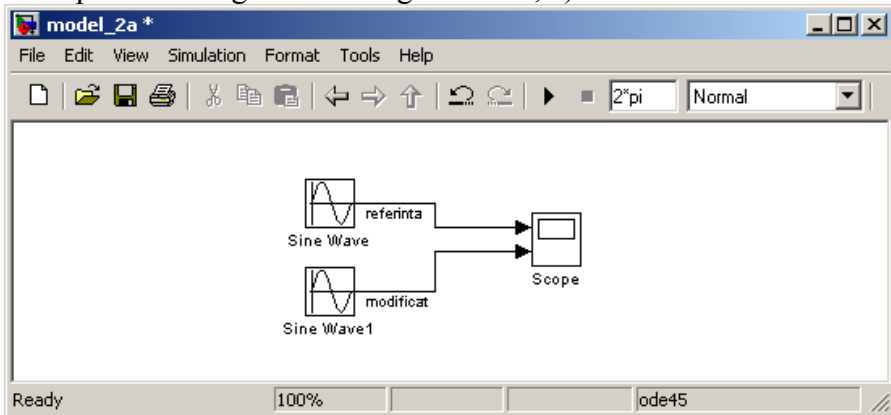
c)  $y_0=1$

**Figura 10.25.** Influența bias-ului.

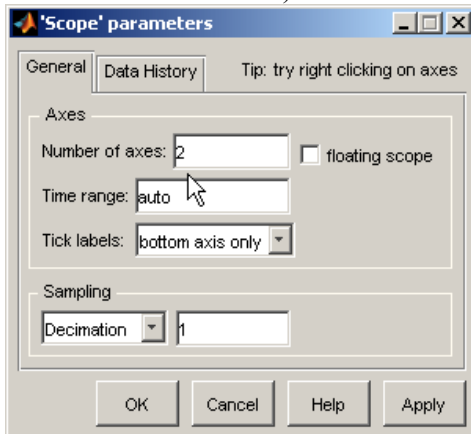
- **Varianta 2.** Se introduce în schema bloc doar un alt bloc Sine Wave care va primi în mod implicit numele Sine Wave 1, figura 10.26, a). Pentru acest caz se consideră influența amplitudinii. Pentru modificarea amplitudinii se deschide fereastra de configurare a blocului Sine Wave 1 și se introduce noua valoare a parametrului Amplitude ( $A=1$ ). Blocul Scope existent are în mod implicit o singură pereche de axe.

Pentru acest caz este nevoie însă de reprezentarea a două funcții. Una din metodele de rezolvare este configurarea blocului Scope existent pentru două perechi de axe. Pentru aceasta se efectuează dublu click LMB pe suprafața blocului Scope și în fereastra de configurare, figura 10.26, b), se modifică parametrul din caseta de dialog Number of Axes (noua valoare este 2). Se observă apariția a încă unui port de intrare pentru blocul Scope. Se conectează ieșirile celor două blocuri Sine Wave cu cele două intrări ale blocului Scope. Pentru identificarea clară a

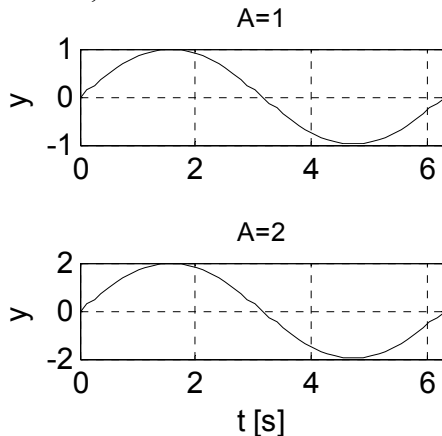
reprezentărilor grafice se specifică numele semnalelor de intrare ale blocului Scope. Pentru aceasta se efectuează pe rând, dublu click cu LMB pe liniile de comunicație dintre blocurile Sine Wave și blocul Scope și se introduce denumirea semnalelor care circulă pe liniile de conexiune respective (semnalul de referință și, respectiv semnalul modificat). Se lansează în execuție simularea după care se efectuează dublu click LMB pe blocul Scope obținându-se reprezentarea grafică din figura 10.26, c).



a) schema bloc modificată, varianta 2



b) modificarea numărului de axe

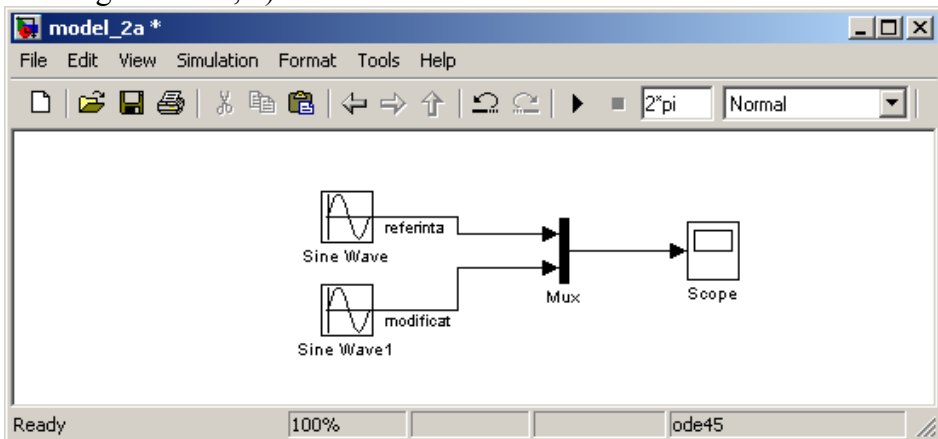


c) reprezentările grafice obținute

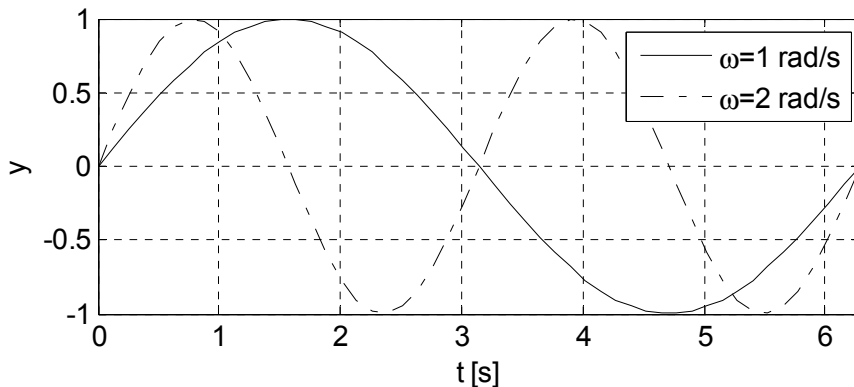
**Figura 10.26.** Influența amplitudinii.

- **Varianta 3.** Se introduce în schema bloc doar un alt bloc Sine Wave care va primi în mod implicit numele Sine Wave 1, figura 10.27, a). Pentru acest caz se consideră influența pulsației. Pentru modificarea pulsației se deschide fereastra de configurare a blocului Sine Wave 1 și se introduce noua valoare a parametrului Frequency ( $\omega=2$ ). Blocul Scope existent are în mod implicit o singură pereche de axe.

Pentru acest caz este nevoie însă de reprezentarea a două funcții. A doua metodă de rezolvare este configurarea blocului Scope pentru o singură pereche de axe, pe portul său de intrare intrând însă un semnal multiplexat, pentru acest caz din două semnale (semnalul de referință și semnalul modificat). Pentru aceasta se introduce în schema bloc un alt element, blocul Mux, din biblioteca Signal Routing. În mod implicit, blocul Mux are două intrări însă acest parametru se poate modifica din fereastra sa de dialog (parametrul Number of Inputs). Se conectează ieșirile celor două blocuri de intrare Sine Wave cu cele două intrări ale blocului de multiplexare. Semnalul multiplexat obținut la ieșirea blocului Mux se conectează cu intrarea blocului Scope. Se lansează în execuție simularea după care se efectuează dublu click LMB pe blocul Scope obținându-se astfel reprezentarea grafică din figura 10.27, b).



a) schema bloc modificată, varianta 3



b) reprezentările grafice obținute  
**Figura 10.27.** Influența pulsației.

### Problema 10.3

Se consideră funcția sinusoidală definită prin:

$$y = y_0 + A \cdot \sin(\omega t + \varphi)$$

în care:  $y_0$  este ordonata în jurul căreia evoluează funcția sinus (bias);  $A$  este amplitudinea;  $\omega$  [rad/s] este pulsația sau frecvența unghiulară;  $\varphi$  [rad] este defazajul sau faza inițială la momentul  $t_0=0$  s;  $t$  [s] este timpul simulării.

Se cere să se realizeze o schemă bloc Simulink pentru vizualizarea grafică a variației funcției:

$$u(t) = \frac{\sqrt{|y|} + \cos t}{\pi^2}$$

pentru următorii parametri:  $y_0=0$ ;  $A=1$ ;  $\omega=1$ ;  $\varphi=0$ ;  $t=0 \div 2\pi$  s.

### Rezolvare

Din analiza modelului analitic al sistemului fizic analizat rezultă următoarele observații:

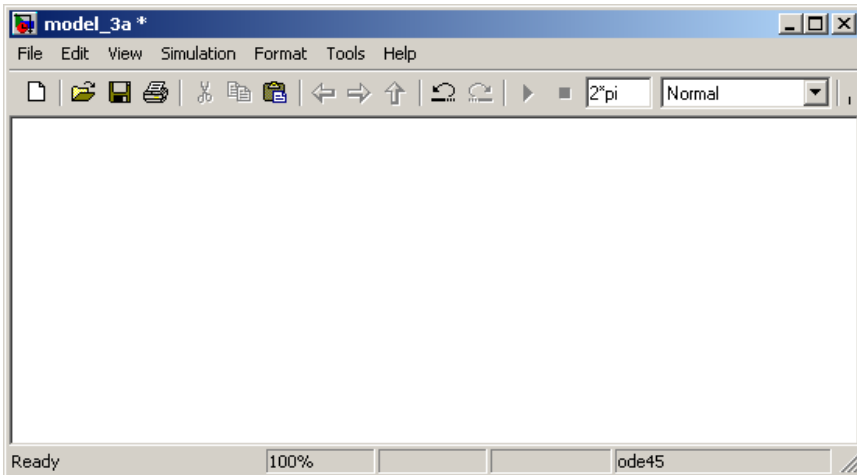
- Există trei mărimi de intrare: funcția sinusoidală pentru modelarea căreia se folosește blocul Sine Wave, timpul de simulare pentru modelarea căreia se folosește blocul Clock și constanta  $\pi$  pentru modelarea căreia se utilizează blocul Constant. Toate blocurile pentru modelarea mărimilor de intrare se găsesc în biblioteca Sources.
- Mărimea de ieșire a modelului analitic analizat  $u(t)$  este o mărime variabilă în timp pentru vizualizarea căreia se va folosi un bloc de tip Scope (biblioteca Sinks).
- Pentru modelarea numărătorului expresiei  $u(t)$  se utilizează blocul Abs (semnalul de ieșire este egal cu valoarea absolută a semnalului de intrare), blocul Math Function (configurat pentru obținerea funcției rădăcină pătrată), blocul Trigonometric Function (configurat pentru obținerea funcției cosinus) și blocul Add (configurat pentru adunarea a două semnale). Toate aceste blocuri se obțin din biblioteca Math Operations.
- Pentru modelarea numitorului expresiei  $u(t)$  se utilizează blocul Math Function (biblioteca Math Operations).
- Realizarea operației de împărțire dintre numărătorul și numitorul expresiei  $u(t)$  se realizează cu ajutorul blocului Divide (biblioteca Math Operations).

Principalele etape ale realizării schemei bloc sunt:

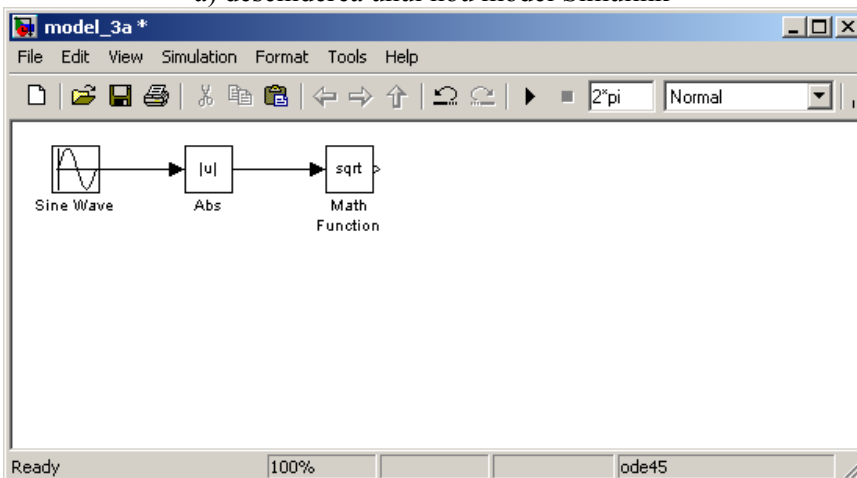
- Se deschide un nou model Simulink fie prin selectarea comenzii New/Model din meniul File, fie prin selectarea comenzii New model din bara de butoane a programului, figura 10.28, a).

Dimensiunea ferestrei în care se va construi schema bloc se va modifica în cursul procesului de realizare a modelului în funcție de complexitatea schemei.

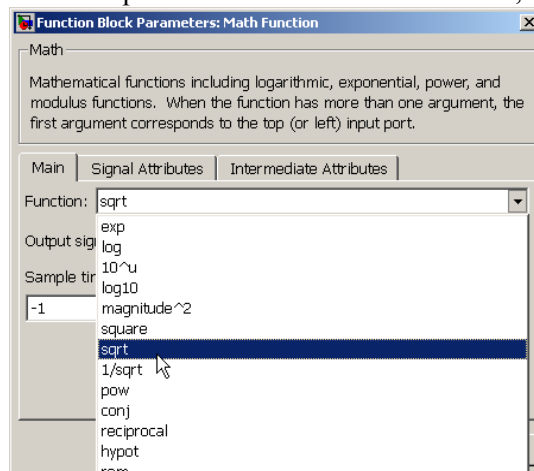
- Pentru modelarea primului termen al numărătorului ( $\sqrt{|y|}$ ) se introduc trei blocuri: `Sine Wave` din biblioteca `Sources`, `Abs` (obținerea valorii absolute a unui semnal) și `Math Functions` din biblioteca `Math Operations`, figura 10.28, b). Se modifică parametrii de configurare ai blocului `Sine Wave` în conformitate cu datele problemei ( $y_0=0$ ;  $A=1$ ;  $\omega=1$ ;  $\varphi=0$ ). Blocul `Math Functions` permite aplicarea unei funcții MATLAB asupra semnalului de intrare. Din lista de funcții MATLAB disponibile, în acest caz, pentru specificarea funcției rădăcină pătrată se alege funcția `sqrt`, figura 10.28, c).
- Pentru modelarea celui de-al doilea termen al numărătorului ( $\cos t$ ) se introduc două blocuri: `Clock` (obținerea timpului de simulare  $t$ ) din biblioteca `Sources` și `Trigonometric Functions` din biblioteca `Math Operations`, figura 10.28, d). Blocul `Trigonometric Functions` permite aplicarea unei funcții trigonometrice directe sau inverse asupra semnalului de intrare. Din lista de funcții disponibile, în acest caz se alege funcția `cos`, figura 10.28, e).
- Pentru modelarea numărătorului ( $\sqrt{|y|} + \cos t$ ) se introduce blocul `Add` (biblioteca `Math Operations`) care realizează adunarea dintre cei doi termeni ai numărătorului, figura 10.28, f).
- Pentru modelarea numitorului ( $\pi^2$ ) se introduc două blocuri: blocul `Constant` din biblioteca `Sources` și un alt bloc `Math Functions` (biblioteca `Math Operations`), figura 10.28, g). Se configurează blocul `Constant` astfel încât valoarea numerică a acestuia să fie egală cu  $\pi$ , figura 10.28, h). Din lista de funcții MATLAB disponibile pentru blocul `Math Functions 1`, în acest caz se alege funcția `square`, figura 10.28, i).
- Pentru realizarea operației de împărțire dintre numărător și numitor și modelarea astfel a expresiei  $u$ , se utilizează blocul `Divide` (biblioteca `Math Operations`, figura 10.28, j). Pentru vizualizarea variației în timp a funcției  $u$  se folosește blocul `Scope` (biblioteca `Sinks`). Se lansează în execuție simularea după care se efectuează dublu click LMB pe blocul `Scope` obținându-se reprezentarea din figura 10.28, k) pentru graficul inițial nescalat și reprezentarea grafică din figura 10.28, l) pentru graficul scalat cu ajutorul comenzii `Autoscale`.



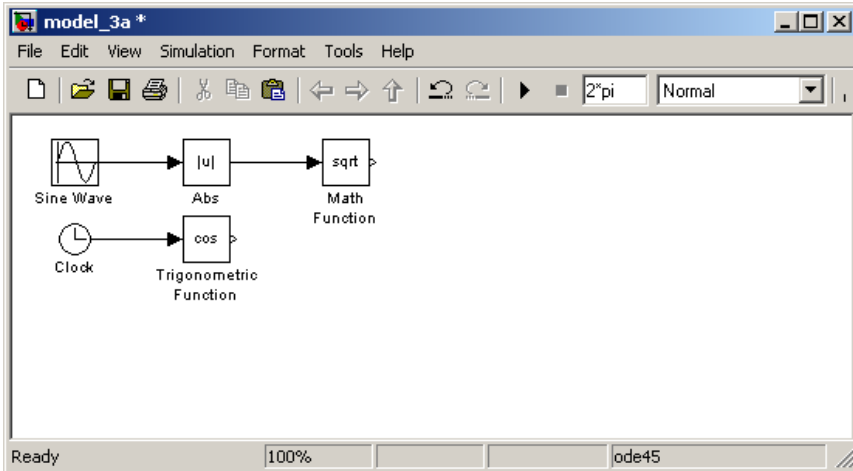
a) deschiderea unui nou model Simulink



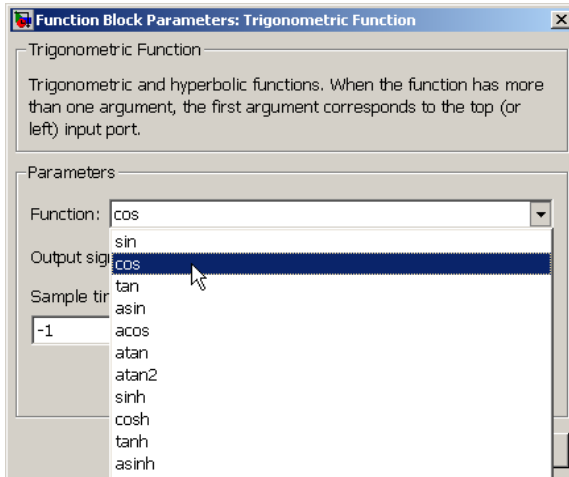
b) modelarea primului termen al numărătorului,  $\sqrt{|y|}$



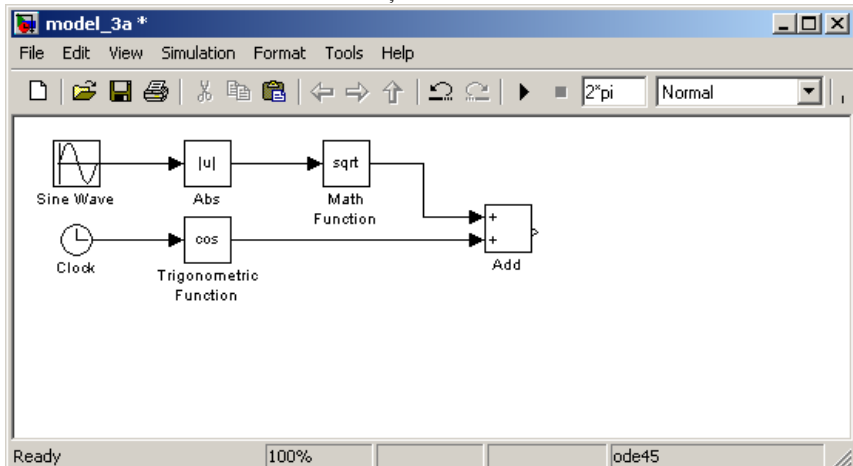
c) configurarea blocului Math Functions pentru obținerea funcției rădăcină pătrată



d) modelarea celui de-al doilea termen al numărătorului,  $\cos t$

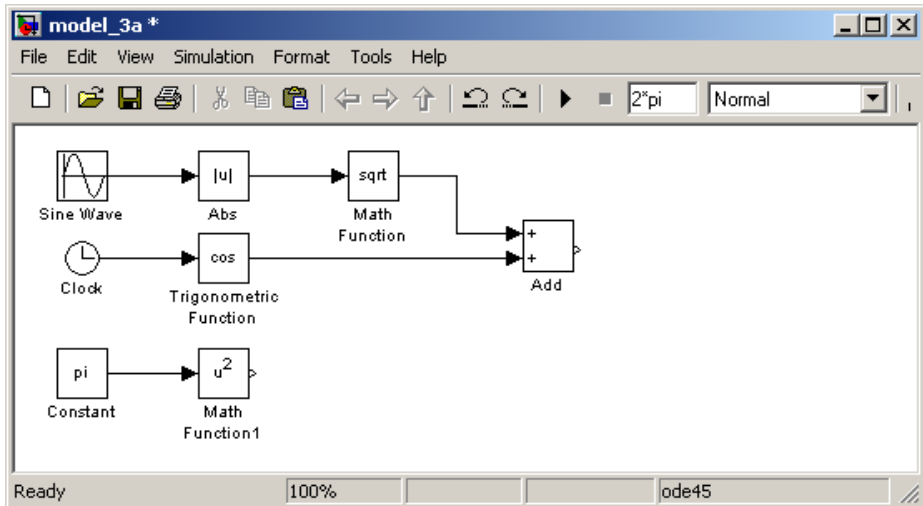


e) configurarea blocului Trigonometric Functions pentru obținerea funcției cosinus

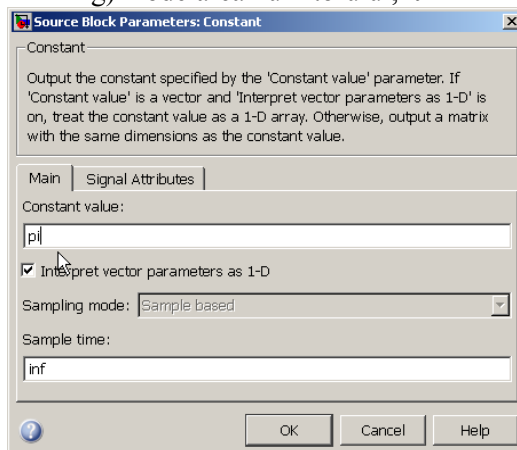


f) modelarea numărătorului,  $\sqrt{|y|} + \cos t$

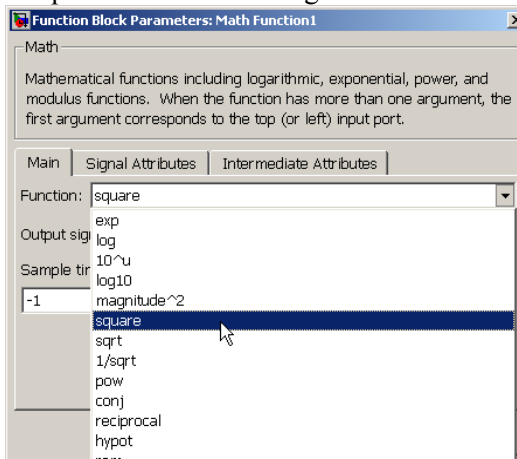




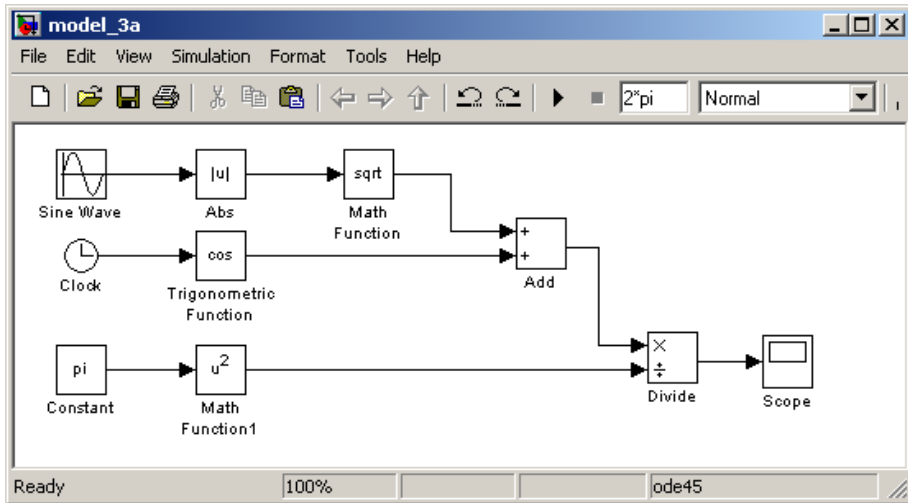
g) modelarea numitorului,  $\pi^2$



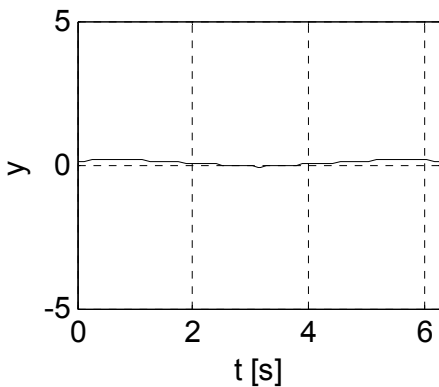
h) modificarea parametrului de configurare al blocului Constant



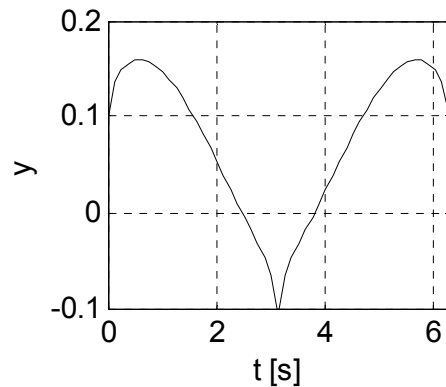
i) configurarea blocului Math Functions 1 pentru obținerea funcției de ridicare la puterea a doua



j) schema bloc finală



k) graficul inițial, nescalat



l) graficul scalat

**Figura 10.28.** Etapele realizării schemei bloc.

### Observații

- Principala cerință a schemei bloc trebuie să fie, evident, cea de reprezentare fidelă a funcției  $u(t)$ . Dezvoltarea schemei bloc se realizează în ordinea logică a operațiilor aritmetice din structura funcției de analizat.
- Pe lângă aspectul funcțional, schema bloc trebuie să respecte și alte cerințe: trebuie să fie cât mai simplă, compactă și fără intersecții accidentale ale liniilor de conexiune. Obținerea unei scheme bloc cu o structură cât mai ordonată se realizează prin alinierea pe orizontală și pe verticală a blocurilor. Prin modificarea culorii blocurilor se pot pune în evidență anumite grupuri de blocuri. De asemenea, se recomandă ca toate blocurile din structura schemei bloc să fie redenumite, în concordanță cu semnificația fizică a variabilelor respective.

## 10.3. SCHEME BLOC SIMULINK PENTRU REZOLVAREA ECUAȚIILOR DIFERENȚIALE ORDINARE

### 10.3.1. Scheme bloc pentru ecuații diferențiale de ordinul 1

Se consideră ecuația diferențială ordinară de ordinul 1:

$$y' = f(t, u(t), y)$$

și condiția inițială:

$$t = 0 \Rightarrow y(0) = y_0$$

În cazul în care variabila independentă este timpul, atunci pentru operatorul de derivare se poate utiliza notația cu punct:

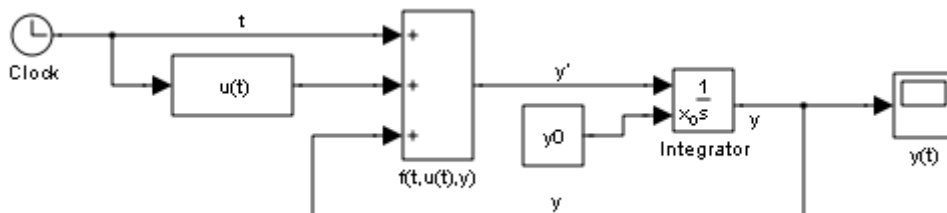
$$\frac{dy}{dt} \Leftrightarrow \dot{y}$$

astfel încât ecuația diferențială se poate rescrie sub forma:

$$\dot{y} = f(t, u(t), y)$$

(formă recomandată pentru realizarea schemei bloc).

Schema bloc Simulink generală pentru rezolvarea ecuațiilor diferențiale ordinare de ordinul 1 este prezentată în figura 10.29.



**Figura 10.29.** Schemă bloc Simulink generală pentru rezolvarea ecuațiilor diferențiale ordinare de ordinul 1.

Principalele elemente ale schemei bloc sunt:

- Construcția funcției de integrat  $f(t, u(t), y)$ . Timpul  $t$  se obține cu ajutorul blocului Clock (biblioteca Sources). Funcția  $u(t)$  se construiește pornind de la variabila timp și folosind diferite funcții algebrice (blocul Math Functions), funcții trigonometrice (blocul Trigonometric Functions), etc., în concordanță cu ecuația diferențială de rezolvat. Funcția  $y$  se obține cu ajutorul unei linii de conexiune inversă pornind de la semnalul obținut după blocul de integrare.

Obținerea formei finale a funcției de integrat  $f(t, u(t), y)$  se realizează cu ajutorul unui bloc de tip Add având semnele configurate în mod corespunzător în funcție de semnul fiecărei componente a funcției de integrat. Pentru cazul prezentat în figura 10.29, blocul Add are semnele (+ + +), configurate conform următoarei relații de definiție a funcției de integrat:

$$f(t, u(t), y) = t + u(t) + y$$

- Integrarea funcției de integrat  $f(t, u(t), y)$  conform relației:

$$y(t) = y_0 + \int_{t_0}^t f(t, u(t), y) dt$$

se realizează cu ajutorul blocului Integrator (biblioteca Continuous), figura 10.30. Condiția inițială se poate introduce intern, direct în fereastra de configurare a blocului Integrator (figura 10.30, a) sau extern, cu un bloc de tip Constant (figura 10.30, b). Ieșirea blocului Integrator reprezintă soluția  $y(t)$ .

- Reprezentarea grafică a soluției  $y(t)$  se obține cu ajutorul unui bloc de tip Scope (biblioteca Sinks).



a) condiție inițială de tip intern

b) condiție inițială de tip extern

**Figura 10.30.** Blocul Integrator.

#### Problema 10.4

Se consideră ecuația diferențială de ordinul 1:

$$\tau \dot{y} + y = a + b \cdot u(t)$$

pentru care se cunosc următoarele:

- condiția inițială:

$$y(0) = 0$$

- valorile parametrilor numerici:

$$a=0,25; b=1$$

$$\tau=\{0,01; 0,1; 1; 10\}$$

- funcția  $u(t)$  este definită prin expresia:

$$u(t) = t + \sin(\omega_f t), \omega_f=\{1; 10\}$$

- intervalul de timp de simulare:

$$t=[t_i; t_f]$$

$$\text{în care } t_i=0 \text{ s și } t_f=50 \text{ s}$$

Se cere să se realizeze schema bloc Simulink pentru rezolvarea ecuației diferențiale și să se analizeze influența modificării valorilor parametrilor  $\tau$  și  $\omega_f$  asupra soluției ecuației diferențiale.

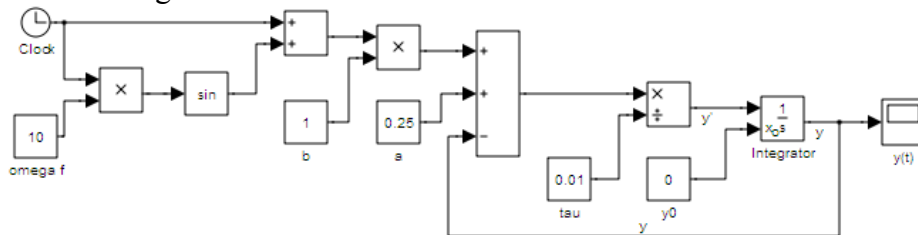
#### Rezolvare

Ecuația diferențială se aduce la forma recomandată:

$$\dot{y} = \frac{1}{\tau} [a + b \cdot u(t) - y]$$

identificându-se astfel în membrul drept al relației funcția de integrat.

Schema bloc Simulink pentru rezolvarea ecuației diferențiale este prezentată în figura 10.31.



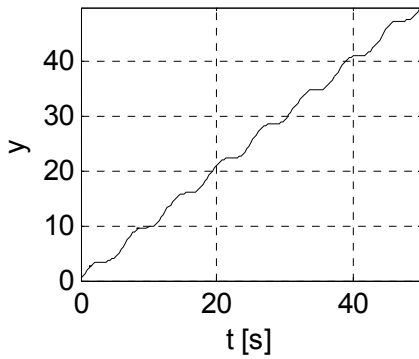
**Figura 10.31.** Schema bloc Simulink pentru rezolvarea ecuației diferențiale de ordinul 1, problema 10.4.

### Observații

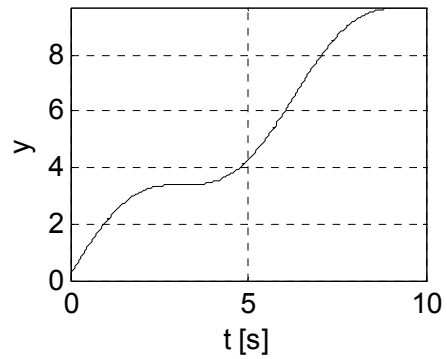
- Funcția de integrat se obține cu ajutorul unui bloc de tip Add configurat cu următoarele semne: + + - și cu un bloc de tip Divide pentru a realiza împărțirea cu  $\tau$  (biblioteca Math Operations).
- Introducerea parametrilor  $a$ ,  $b$ ,  $\tau$  și  $\omega_f$  se realizează cu ajutorul unor blocuri de tip Constant (biblioteca Sources).
- Funcția sinus se realizează cu ajutorul blocului Trigonometric Functions (biblioteca Math Operations).
- Timpul de simulare se obține cu ajutorul blocului Clock (biblioteca Sources).
- Realizarea operației de integrare se realizează cu ajutorul unui bloc de tip Integrator (biblioteca Continuous), având condiția inițială de tip extern, introdusă cu ajutorul unui bloc de tip Constant, identificat prin numele  $y_0$ .
- Pe portul - al blocului Add de construcție a funcției de integrat se introduce semnalul obținut după blocul integrator printr-o linie de conexiune inversă.
- Vizualizarea variației în timp a soluției  $y(t)$  a ecuației diferențiale se realizează cu ajutorul unui bloc de tip Scope (biblioteca Sinks).
- Valoarea finală a timpului de simulare  $t_f$ , se introduce în caseta de control Simulation stop time.

În urma simulării schemei bloc se obțin reprezentările grafice din figura 10.32 pentru diferite combinații ale valorilor parametrilor de intrare:

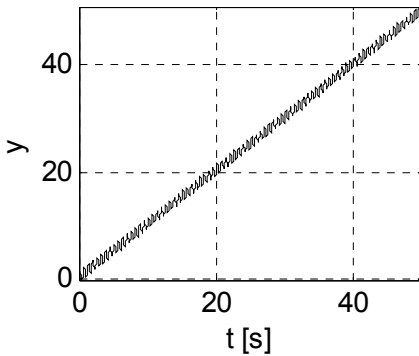
- Pentru cazul  $\omega_f=1$ ,  $\tau=0,01$ ;  $t=50$  s.
- Pentru cazul  $\omega_f=1$ ,  $\tau=0,01$ ;  $t=10$  s.
- Pentru cazul  $\omega_f=10$ ,  $\tau=0,01$ ;  $t=50$  s.
- Pentru cazul  $\omega_f=10$ ,  $\tau=0,01$ ;  $t=10$  s.
- Pentru cazul  $\omega_f=1$ ,  $\tau=10$ ;  $t=50$  s.
- Pentru cazul  $\omega_f=10$ ,  $\tau=10$ ;  $t=50$  s.



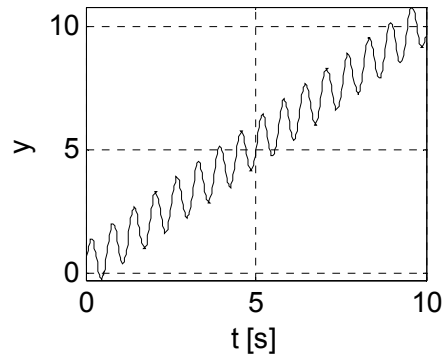
a)  $\omega_f=1; \tau=0,01; t=50$  s



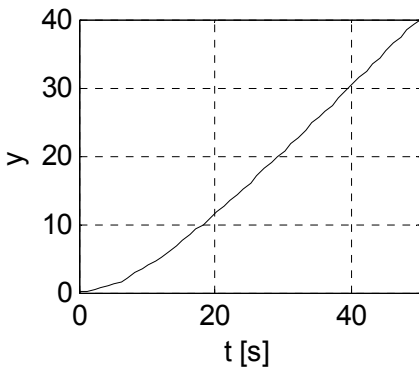
b)  $\omega_f=1; \tau=0,01; t=10$  s



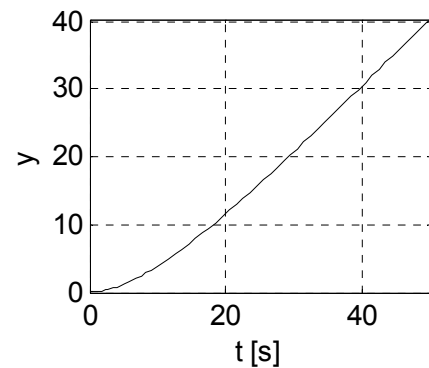
c)  $\omega_f=10; \tau=0,01; t=50$  s



d)  $\omega_f=10; \tau=0,01; t=10$  s



e)  $\omega_f=1; \tau=10; t=50$  s



f)  $\omega_f=10; \tau=10; t=50$  s

**Figura 10.32.** Soluția ecuației diferențiale de ordinul 1, problema 10.4.

În cazul unor variații mai intense a soluției ecuației diferențiale, pentru efectuarea unor analize și interpretări corecte, se recomandă vizualizarea grafică a soluției și pentru diferite subintervale caracteristice din domeniul total al timpului de rezolvare. De exemplu, în figurile 10.32, b) și 10.32, d) se reprezintă soluția ecuației diferențiale pentru intervalul  $[0; 10]$  s, un subinterval al domeniului de integrare  $[0; 50]$  s.

### 10.3.2. Scheme bloc pentru ecuații diferențiale de ordinul 2

Se consideră ecuația diferențială ordinară de ordinul 2:

$$y'' = f(t, u(t), y, y')$$

și condițiile inițiale:

$$t = 0 \Rightarrow \begin{cases} y(0) = y_0 \\ y'(0) = y'_0 \end{cases}$$

În cazul în care variabila independentă este timpul, atunci pentru operatorii de derivare se poate utiliza notația cu punct:

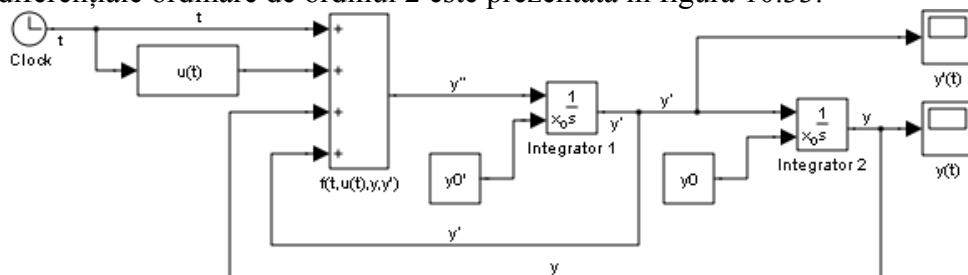
$$\frac{dy}{dt} \Leftrightarrow \dot{y}, \frac{d^2y}{dt^2} \Leftrightarrow \ddot{y}$$

astfel încât ecuația diferențială se poate rescrie sub forma:

$$\ddot{y} = f(t, u(t), y, \dot{y})$$

(formă recomandată pentru realizarea schemei bloc).

Schema bloc Simulink generală pentru rezolvarea ecuațiilor diferențiale ordinare de ordinul 2 este prezentată în figura 10.33.



**Figura 10.33.** Schemă bloc Simulink generală pentru rezolvarea ecuațiilor diferențiale ordinare de ordinul 2.

Principalele elemente ale schemei bloc sunt:

- Construcția funcției de integrat  $f(t, u(t), y, y')$ . Timpul  $t$  se obține cu ajutorul blocului Clock (biblioteca Sources). Funcția  $u(t)$  se construiește pornind de la variabila timp și folosind diferite funcții algebrice (blocul Math Functions), funcții trigonometrice (blocul Trigonometric Functions), etc. Funcția  $y$  se obține cu ajutorul unei linii de conexiune inversă pornind de la semnalul obținut după cel de-al doilea bloc de integrare. Funcția  $y'$  se obține cu ajutorul unei linii de conexiune inversă pornind de la semnalul obținut după primul bloc de integrare. Obținerea formei finale a funcției de integrat  $f(t, u(t), y, y')$  se realizează cu ajutorul unui bloc de tip Add având semnele configurate în mod corespunzător în funcție de semnul fiecărei componente a funcției de integrat. Pentru acest caz blocul Add are semnele (+ + + +), configurate conform următoarei relații de definiție a funcției de integrat:

$$f(t, u(t), y) = t + u(t) + y + y'$$

- Integrarea funcției  $f(t, u(t), y, y')$  se realizează de două ori, conform relațiilor:

$$y'(t) = y'_0 + \int_{t_0}^t f(t, u(t), y, y') dt$$

$$y(t) = y_0 + \int_{t_0}^t y'(t) dt$$

În acest scop se utilizează două blocuri de tip Integrator (biblioteca Continuous) conectate în serie, identificate prin numele Integrator 1 și Integrator 2. Condiția inițială se poate introduce intern, direct în fereastra de configurare a blocurilor Integrator, sau extern, cu ajutorul unor blocuri de tip Constant. Ieșirea blocului Integrator 1 reprezintă derivata de ordinul 1,  $y'(t)$ , care se transmite ulterior la intrarea celui de-al doilea bloc de integrare, Integrator 2 și care după integrare furnizează pe portul său de ieșire un semnal reprezentând soluția  $y(t)$  a ecuației diferențiale.

- Reprezentarea grafică a soluției  $y(t)$  ca și a derivatei de ordinul 1  $y'(t)$ , se realizează cu ajutorul a două blocuri de tip Scope (biblioteca Sinks), identificate prin numele  $y(t)$  și  $y'(t)$ .

### Problema 10.5

Se consideră ecuația diferențială de ordinul 2:

$$\ddot{y} + a\dot{y} + by = c \cdot u(t)$$

pentru care se cunosc următoarele:

- Condițiile inițiale:

$$\begin{cases} y(0) = 0 \\ y'(0) = 1 \end{cases}$$

- Valorile parametrilor:

$$a=0,1; b=1,2; c=5$$

- Funcția  $u(t)$  este definită prin expresia:

$$u(t) = \sin(\omega_f t), \omega_f = \{0,1; 10\}$$

- Intervalul de timp de simulare:

$$t=[t_i; t_f], \text{ în care } t_i=0 \text{ s și } t_f=60 \text{ s}$$

Se cere să se realizeze schema bloc Simulink pentru rezolvarea ecuației diferențiale.

### Rezolvare

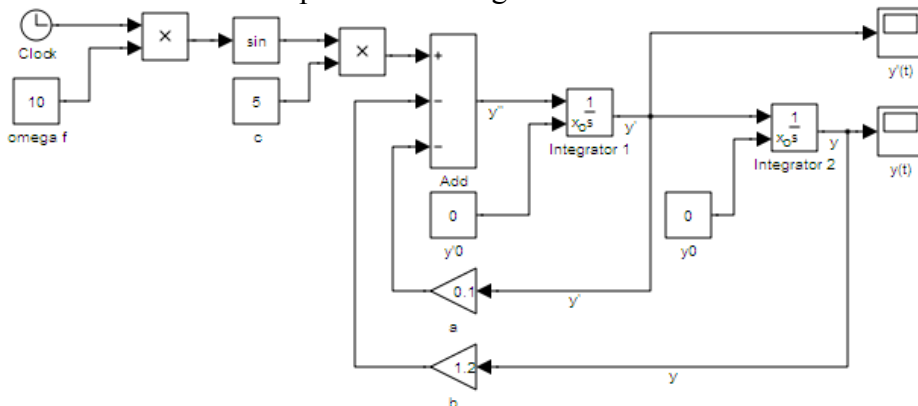
Ecuația diferențială se aduce la forma recomandată:

$$\ddot{y} = c \cdot u(t) - a\dot{y} - by$$



identificându-se astfel în membrul drept al relației funcția de integrat formată din trei termeni având semnele + - -.

Schema bloc este prezentată în figura 10.34.



**Figura 10.34.** Schema bloc Simulink pentru rezolvarea ecuației diferențiale de ordinul 2, problema 10.5.

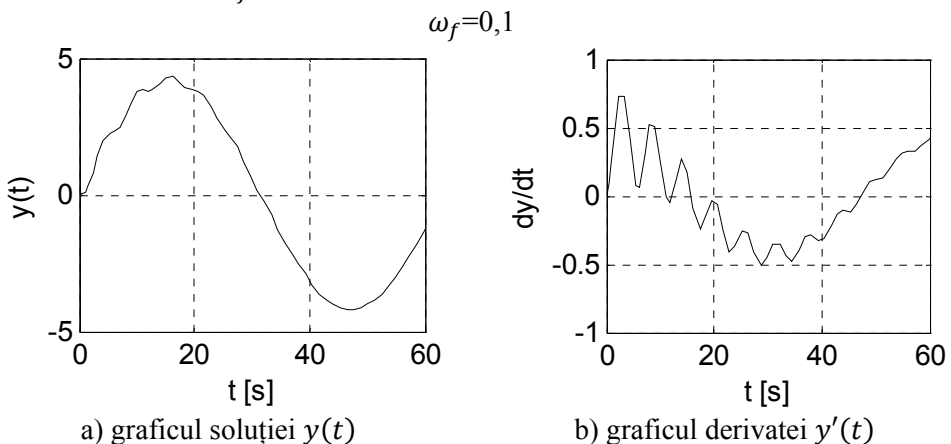
### Observații

- Funcția de integrat se obține cu ajutorul unui bloc de tip Add (biblioteca Math Operations) configurat cu trei intrări și având următoarele semne: + - -.
- Introducerea parametrilor  $c$  și  $\omega_f$  se realizează cu ajutorul unor blocuri de tip Constant (biblioteca Sources) identificate prin numele  $c$  și  $\omega_f$ .
- Introducerea funcției sinus se realizează cu ajutorul blocului Trigonometric Functions (biblioteca Math Operations).
- Timpul de simulare se obține cu ajutorul blocului Clock (biblioteca Sources).
- Realizarea operațiilor de integrare se realizează cu ajutorul a două blocuri de tip Integrator (biblioteca Continuous) identificate prin numele Integrator 1 și Integrator 2, având condițiile inițiale de tip extern introduse cu ajutorul a două blocuri de tip Constant identificate prin numele  $y_0$  și  $y'_0$ . Semnalul de intrare al primului bloc de integrare reprezintă chiar funcția de integrat  $f(t, u(t), y, \dot{y})$ , care în acest caz are expresia  $c \cdot u(t) - a\dot{y} - by$ . Semnalul de ieșire din primul bloc de integrare reprezintă derivata  $\dot{y}(t)$  a soluției ecuației diferențiale. Acest semnal se va integra a doua oară în cel de-al doilea bloc de integrare. Semnalul de ieșire din cel de-al doilea bloc de integrare reprezintă soluția  $y(t)$  a ecuației diferențiale.

- Pe primul port – al blocului Add de construcție a funcției de integrat se introduce semnalul obținut după blocul Integrator 2 (acest semnal reprezintă soluția  $y(t)$  a ecuației diferențiale) printr-o linie de conexiune inversă, semnal multiplicat în prealabil cu ajutorul unui bloc Gain (biblioteca Math Operations) identificat prin numele b având factorul de multiplicare 1,2.
- Pe cel de-al doilea port – al blocului Add de construcție a funcției de integrat se introduce semnalul obținut după blocul Integrator 1 (acest semnal reprezintă derivata  $\dot{y}(t)$  a soluției ecuației diferențiale) printr-o linie de conexiune inversă, semnal multiplicat în prealabil cu ajutorul unui bloc Gain (biblioteca Math Operations) identificat prin numele a având factorul de multiplicare 0,1.
- Vizualizarea variației în timp a soluției  $y(t)$  se realizează cu ajutorul unui bloc de tip Scope (biblioteca Sinks) identificat prin numele  $y(\tau)$ . Acest bloc primește pe portul său de intrare semnalul de ieșire din cel de-al doilea bloc de integrare, Integrator 2.
- Vizualizarea variației în timp a derivatei soluției  $y'(t)$  se realizează cu ajutorul unui bloc de tip Scope (biblioteca Sinks) identificat prin numele  $y'(\tau)$ . Acest bloc primește pe portul său de intrare semnalul de ieșire din primul bloc de integrare, Integrator 1.
- Valoarea finală a timpului de simulare  $t_f=60$  s, se introduce în caseta Simulation stop time.

În urma simulării schemei bloc se obțin reprezentările grafice din figura 10.35 pentru cele două valori ale parametrului  $\omega_f$ :

- Pentru  $\omega_f=0,1$ , graficul soluției  $y(t)$ .
- Pentru  $\omega_f=0,1$ , graficul derivatei soluției  $y'(t)$ .
- Pentru  $\omega_f=10$ , graficul soluției  $y(t)$ .
- Pentru  $\omega_f=10$ , graficul derivatei soluției  $y'(t)$ .

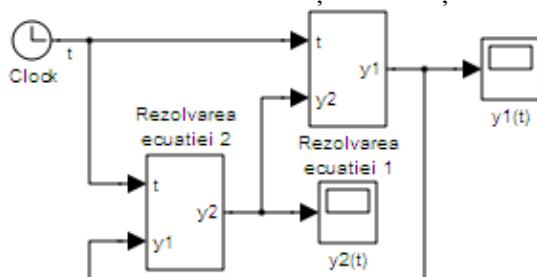




se aplică și la intrarea funcției de integrat din scheme elementară 2, iar soluția  $y_2(t)$  obținută la ieșirea schemei elementare 2 se aplică și la intrarea funcției de integrat din scheme elementară 1.

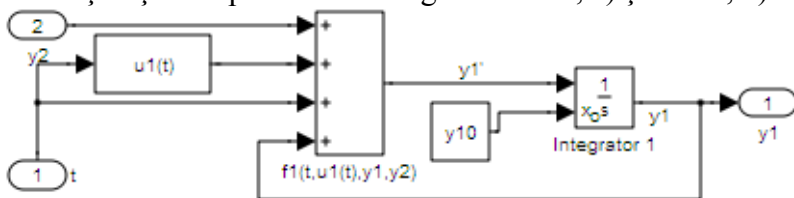
- Principalele blocuri ale schemei elementare 1 sunt: blocul de construcție a funcției de integrat  $f_1(t, u_1(t), y_1, y_2)$ , blocul de integrare identificat prin numele Integrator 1 și blocul Scope pentru vizualizarea soluției  $y_1(t)$ .
- Principalele blocuri ale schemei elementare 2 sunt: blocul de construcție a funcției de integrat  $f_2(t, u_2(t), y_1, y_2)$ , blocul de integrare identificat prin numele Integrator 2 și blocul Scope pentru vizualizarea soluției  $y_2(t)$ .

Utilizând metoda de lucru bazată pe crearea subsistemelor, schema bloc pentru rezolvarea sistemului de ecuații diferențiale de ordinul 1 devine:

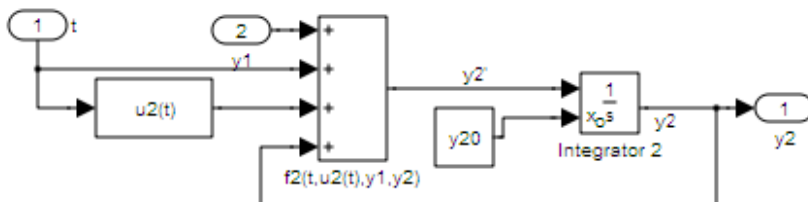


**Figura 10.37.** Schemă bloc Simulink generală pentru rezolvarea sistemelor de ecuații diferențiale de ordinul 1, metoda subsistemelor.

Cele două subsisteme identificate prin numele Rezolvarea ecuației 1 și Rezolvarea ecuației 2 conțin schemele elementare 1 și 2 și sunt prezentate în figurile 10.38, a) și 10.38, b).



a) Subsistemul Rezolvarea ecuației 1



b) Subsistemul Rezolvarea ecuației 2

**Figura 10.38.** Structura celor două subsisteme ale schemei bloc pentru rezolvarea sistemelor de ecuații diferențiale de ordinul 1.

### Problema 10.6

Se consideră sistemul de ecuații diferențiale ordinare:

$$\begin{cases} y_1' = -y_1 - y_2^2 + u_1(t) \\ y_2' = -y_2 + e^{y_1} + u_2(t) \end{cases}$$

pentru care se cunosc următoarele:

- Condițiile inițiale:

$$\begin{cases} y_1(0) = 0 \\ y_2(0) = 0 \end{cases}$$

- Funcția  $u_1(t)$  este definită prin expresia:

$$u_1(t) = \sin(\omega_{f1}t), \omega_{f1}=0,1$$

- Funcția  $u_2(t)$  este definită prin expresia:

$$u_2(t) = \sin(\omega_{f2}t), \omega_{f2}=0,25$$

- Intervalul de timp de simulare:

$$t=[t_i; t_f], \text{ în care } t_i=0 \text{ s și } t_f=150 \text{ s}$$

Se cere să se realizeze schema bloc Simulink pentru rezolvarea ecuației diferențiale.

### Rezolvare

Schema bloc este prezentată în figura 10.39.

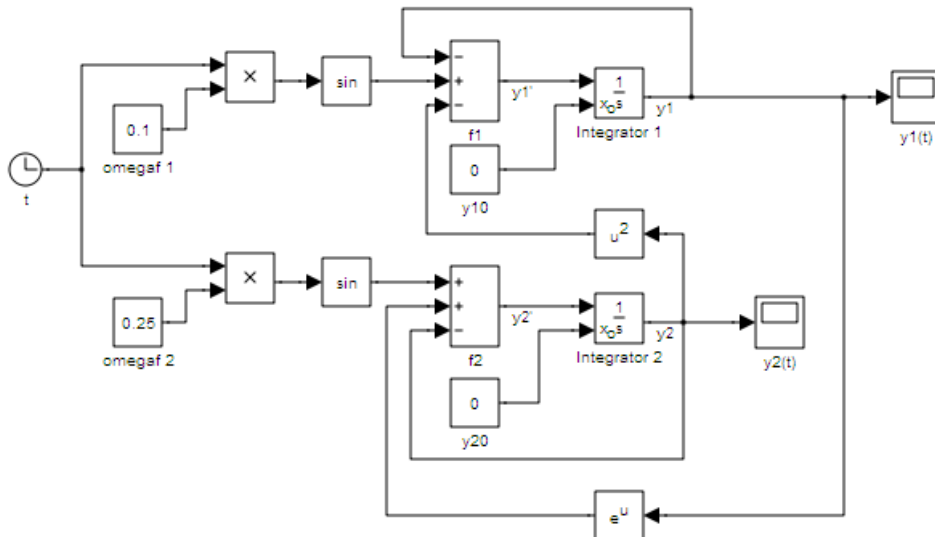


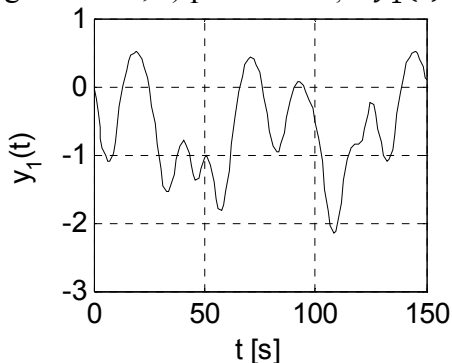
Figura 10.39. Schemă bloc Simulink pentru rezolvarea sistemului de ecuații diferențiale de ordinul 1, problema 10.6.

### Observații

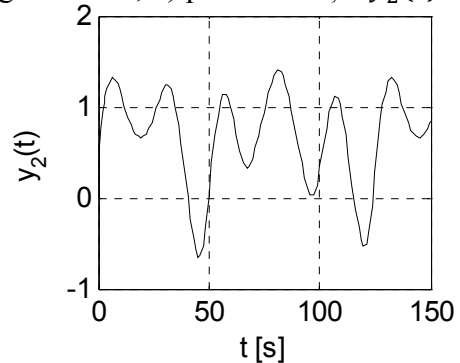
- Introducerea funcției trigonometrice sinus se realizează cu ajutorul blocului Trigonometric Functions (biblioteca Math Operations).

- Timpul de simulare se obține cu ajutorul blocului Clock (biblioteca Sources).
- Funcțiile de integrat se obțin cu ajutorul a două blocuri de tip Add (biblioteca Math Operations) configurate cu următoarele semne: +- pentru funcția de integrat  $f_1 = -y_1 - y_2^2 + u_1(t)$  și ++ pentru funcția de integrat  $f_2 = -y_2 + e^{y_1} + u_2(t)$ .
- Pentru exprimarea termenului  $y_2^2$  se utilizează blocul Math Function (biblioteca Math Operations) configurat pentru obținerea funcției de ridicare la puterea a doua (funcția square).
- Pentru exprimarea termenului  $e^{y_1}$  se utilizează blocul Math Function (biblioteca Math Operations) configurat pentru obținerea funcției exponențiale (funcția exp).
- Realizarea operațiilor de integrare se realizează cu ajutorul a două blocuri de tip Integrator (biblioteca Continuous) identificate prin numele Integrator 1 și Integrator 2, având condițiile inițiale de tip extern introduse cu ajutorul a două blocuri de tip Constant identificate prin numele y10 și y20. Semnalul de intrare al primului bloc de integrare reprezintă funcția de integrat  $f_1$ . Semnalul de ieșire din primul bloc de integrare reprezintă soluția  $y_1(t)$ . Semnalul de intrare al celui de-al doilea bloc de integrare reprezintă funcția de integrat  $f_2$ . Semnalul de ieșire din cel de-al doilea bloc de integrare reprezintă soluția  $y_2(t)$ .
- Vizualizarea variației în timp a soluțiilor  $y_1(t)$  și  $y_2(t)$  a sistemului de ecuații diferențiale se realizează cu ajutorul a două blocuri de tip Scope (biblioteca Sinks) identificate prin numele y1(t) și y2(t).

În urma simulării schemei bloc se obțin reprezentările grafice din figura 10.40, a) pentru soluția  $y_1(t)$  și figura 10.40, b) pentru soluția  $y_2(t)$ .



a) graficul soluției  $y_1(t)$



b) graficul derivatei  $y_2(t)$

**Figura 10.40.** Soluțiile sistemului de ecuații diferențiale de ordinul 1, problema 10.6.

## BIBLIOGRAFIE

1. Cavallo A., Setola R., Vasca F., La nuova guida a MATLAB, Simulink e Control Toolbox, Liguori Editore, Napoli, 2002.
2. Dabney J.B., Harman T.L., Mastering Simulink, Pearson, Prentice Hall, New Jersey, 2004.
3. MathWorks, Simulink. Users's Guide; Getting Started Guide; Graphical User Interface, 2014.
4. MathWorks, Simulation and Model-Based Design, <http://www.mathworks.com/help/simulink/index.html>, accesat la 1.04.2014.
5. MathWorks, About the Library Browser, <http://www.mathworks.com/help/simulink/gui/about-the-library-browser.html>, accesat la 1.04.2014.
6. MathWorks, Sources, <http://www.mathworks.com/help/simulink/sources.html>, accesat la 1.04.2014.
7. MathWorks, Sinks, <http://www.mathworks.com/help/simulink/sinks.html>, accesat la 1.04.2014.
8. MathWorks, Math Operations, <http://www.mathworks.com/help/simulink/math-operations.html>, accesat la 1.04.2014.
9. MathWorks, Signal Routing, <http://www.mathworks.com/help/simulink/signal-routing.html>, accesat la 1.04.2014.
10. MathWorks, Continuous, <http://www.mathworks.com/help/simulink/continuous.html>, accesat la 1.04.2014.
11. MathWorks, User-Defined Functions, <http://www.mathworks.com/help/simulink/user-defined-functions.html>, accesat la 1.04.2014.
12. MathWorks, Configuration Parameters Dialog Box Overview, <http://www.mathworks.com/help/simulink/gui/configuration-parameters-dialog-box-overview.html>, accesat la 17.05.2014.
13. MathWorks, Rounding Function, <http://www.mathworks.com/help/simulink/slref/roundingfunction.html>, accesat la 17.05.2014.
14. MathWorks, Subsystem, Atomic Subsystem, Nonvirtual Subsystem, CodeReuse Subsystem, <http://www.mathworks.com/help/simulink/slref/subsystem.html>, accesat la 17.05.2014.
15. MathWorks, Constant, <http://www.mathworks.com/help/simulink/slref/constant.html>, accesat la 17.05.2014.
16. MathWorks, Product, <http://www.mathworks.com/help/simulink/slref/product.html>, accesat la 17.05.2014.
17. MathWorks, Sum, Add, Subtract, Sum of Elements, <http://www.mathworks.com/help/simulink/slref/add.html>, accesat la 17.05.2014.
18. MathWorks, Divide, <http://www.mathworks.com/help/simulink/slref/divide.html>, accesat la 17.05.2014.
19. MathWorks, Display, <http://www.mathworks.com/help/simulink/slref/display.html>, accesat la 17.05.2014.
20. MathWorks, Goto, <http://www.mathworks.com/help/simulink/slref/goto.html>, accesat la 17.05.2014.
21. MathWorks, From, <http://www.mathworks.com/help/simulink/slref/from.html>, accesat la 17.05.2014.

## CAPITOLUL 11

### ELEMENTE DE CALCUL SIMBOLIC

#### 11.1. GENERALITĂȚI

Rezolvarea problemelor matematice ca și a aplicațiilor specifice domeniului științelor ingineresti cu ajutorul calculatorului se poate realiza prin două metode:

- **Metode de calcul numeric.** Soluția problemelor se determină cu ajutorul unor algoritmi care operează asupra reprezentărilor numerice ale obiectelor matematice. Valorile numerice sunt stocate în calculator în format zecimal cu virgulă mobilă și oricât de performant ar fi sistemul de reprezentare al numerelor (32, 64, 128 biți), deseori apar erori de aproximare datorită faptului că numărul de zecimale ale numărului poate fi mai mare decât numărul disponibil de biți (eliminarea cifrelor zecimale în exces). Erorile de aproximare ale numerelor se pot amplifica în cursul diferitelor etape de calcul specifice algoritmului de rezolvare a problemei respective. Rezultă că prin metodele de calcul numeric se poate obține doar o soluție aproximativă a problemei de rezolvat.
- **Metode de calcul simbolic.** Soluția problemelor se determină cu ajutorul unor algoritmi care operează asupra reprezentărilor simbolice ale obiectelor matematice (funcții, expresii, ecuații, chiar numere). Dacă există, soluțiile unei probleme determinate prin metoda de calcul simbolic sunt întotdeauna exacte, deoarece operează cu reprezentările exacte ale obiectelor matematice (chiar și numere zecimale).

Pentru exemplificarea celor două metode de analiza se consideră problema calculului integralei definite:

$$f = \int_0^1 F(x) dx$$

în care funcția de integrat este definită prin:

$$F(x) = \frac{e^x}{e^{2x} + 1}$$

Rezolvarea problemei prin cele două metode se prezintă în figura 11.1, (a-metoda de calcul numeric; b-metoda de calcul simbolic).

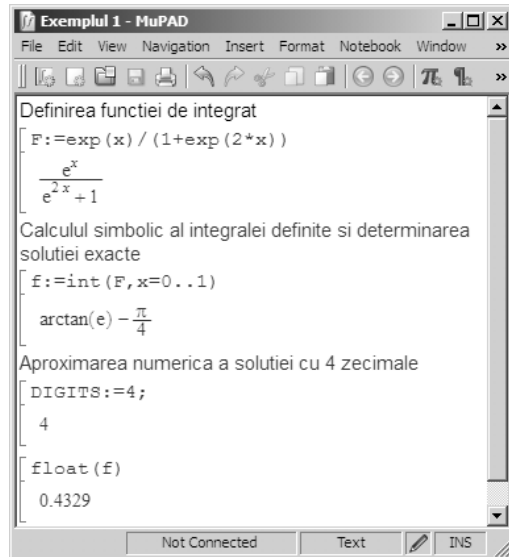


```

% Definirea domeniului
x=linspace(0,1,1000);
% Definirea functiei
F=exp(x)./(1+exp(2*x));
% Calculul numeric al
integralei
F=trapz(x,F)
F=
    0.4329

```

a) calcul numeric



b) calcul simbolic

**Figura 11.1.** Exemplificarea comparativă a metodelor de calcul numeric și simbolic.

## 11.2. MuPAD-PREZENTARE GENERALĂ

Rezolvarea problemelor de calcul simbolic în limbajul de programare MATLAB se poate realiza prin două metode:

- Metoda directă, în fereastra de comenzi Command Window, începând cu declararea variabilelor de tip simbolic printr-o instrucțiune de forma `syms a b`, în care `a` și `b` sunt două variabile de tip simbolic, după care urmează scrierea instrucțiunilor de calcul simbolic.
- Metoda interfeței specializate pentru calcule simbolice, MuPAD.

Limbajul de programare MuPAD (Multi Processing Algebra Data tool), a fost lansat în anul 1990 de către un grup de cercetători de la Universitatea din Paderborn, Germania. Din anul 1997, la dezvoltarea și promovarea limbajului de programare MuPAD s-a adăugat și compania germană SciFace Software [58]. În anul 2008, compania MathWorks a cumpărat compania SciFace Software și astfel limbajul de programare MuPAD a fost adăugat în structura limbajului de programare MATLAB sub forma unui toolbox (Symbolic Math Toolbox [1], [2], [3]).

Principalele facilități computaționale ale limbajului MuPAD sunt:

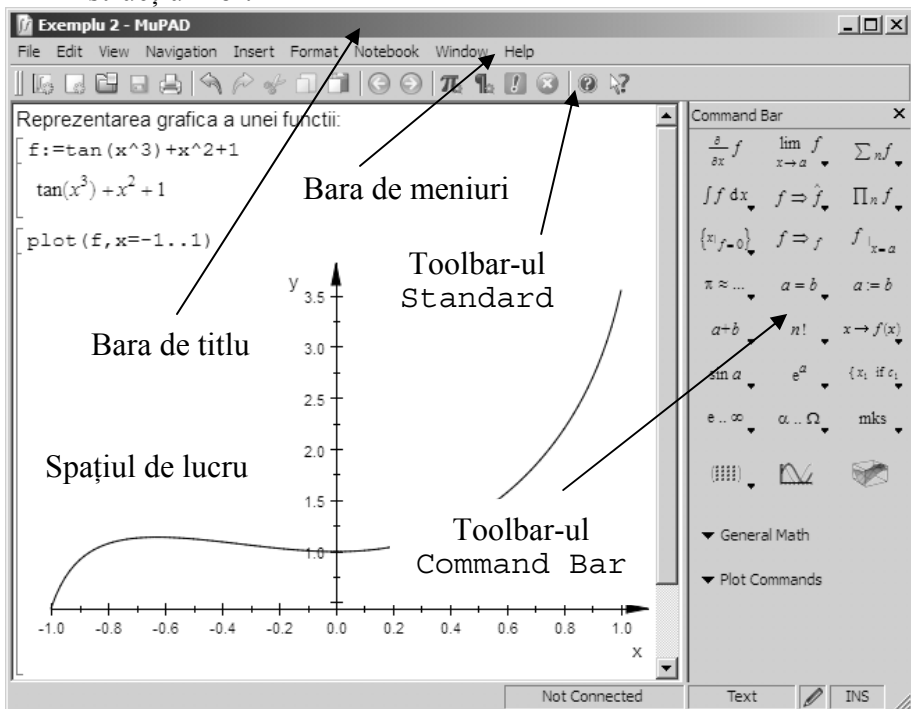
- Definirea și manipularea expresiilor simbolice.
- Calculul simbolic al integralelor, derivatelor, sumelor și limitelor.
- Rezolvarea simbolică a ecuațiilor și sistemelor de ecuații algebrice.
- Rezolvarea simbolică a ecuațiilor și sistemelor de ecuații diferențiale.

- Elemente de calcul statistic.
- Elemente de algebră liniară.
- Reprezentarea grafică a funcțiilor și realizarea animațiilor.

Lansarea în execuție a interfeței grafice MuPAD se realizează fie tastând mupad în fereastra de comenzi MATLAB, fie prin butonul Start/Toolboxes/Symbolic Math/MuPAD.

Principalele elemente ale interfeței MuPAD (figura 11.2) sunt:

- Bara de titlu a programului cu butoanele clasice de minimizare, maximizare și închidere.
- Bara de meniuri conținând comenzi specifice grupate în următoarele meniuri: File, Edit, View, Navigation, Insert, Format, Notebook, Window și Help.
- Butoane corespunzătoare principalelor comenzi generale grupate în toolbar-urile Standard și Format.
- Butoane corespunzătoare principalelor comenzi specifice de calcul simbolic grupate în toolbar-ul Command Bar. Bara de comenzi Command Bar poate fi activată/dezactivată selectând comanda Command Bar din meniul View.
- Spațiul de lucru (fereastra de comenzi) în care se scriu comentariile și instrucțiunile și în care se vizualizează rezultatele executării instrucțiunilor.

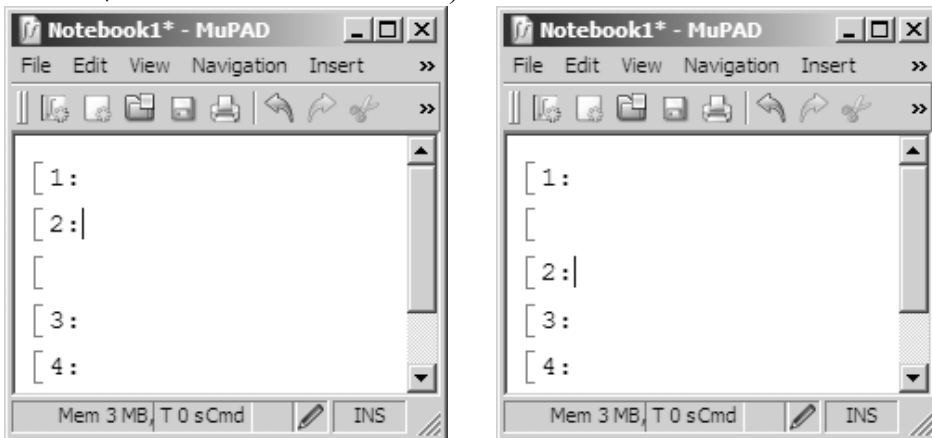


**Figura 11.2.** Interfața limbajului de programare MuPAD.

Diferitele elemente care rezultă în cursul procesului interactiv de operare cu interfața programului MuPAD au culori diferite. Astfel, în mod implicit: comenzile și instrucțiunile scrise de utilizator au culoarea roșie; rezultatul obținut în urma evaluării unei comenzi are culoarea albastră; comentariile introduse de utilizator au culoarea neagră. Modificarea acestor culori se poate face prin comanda de configurare `Configure MuPAD`, [4] din meniul `View`.

În mod implicit, la deschiderea unui nou fișier MuPAD apare prompterul specific introducerii comenzilor, `[`. Comenzile se scriu la prompterul MuPAD în ceea ce se numește linie de calcul (`Calculation`). Introducerea unei noi linii de calcul se realizează automat la apăsarea tastei `Enter` după linia de comenzi anterioară. Dacă ulterior scrierii unei succesiuni de comenzi, se dorește intercalarea între două comenzi existente a unei noi linii de comenzi se utilizează una din comenzile `Insert/Calculation` sau `Insert/Calculation Above` după cum noua linie de comenzi va fi introdusă după sau înainte de linia curentă.

În exemplul prezentat în figura 11.3 se consideră linia 2 ca fiind linia curentă și se observă efectul utilizării celor două comenzi de introducere a unei noi linii de comenzi: figura 11.3, a) - introducerea unei noi linii de comenzi după linia curentă (`Insert/Calculation`); figura 11.3, b) - introducerea unei noi linii de comenzi înainte de linia curentă (`Insert/Calculation Above`).



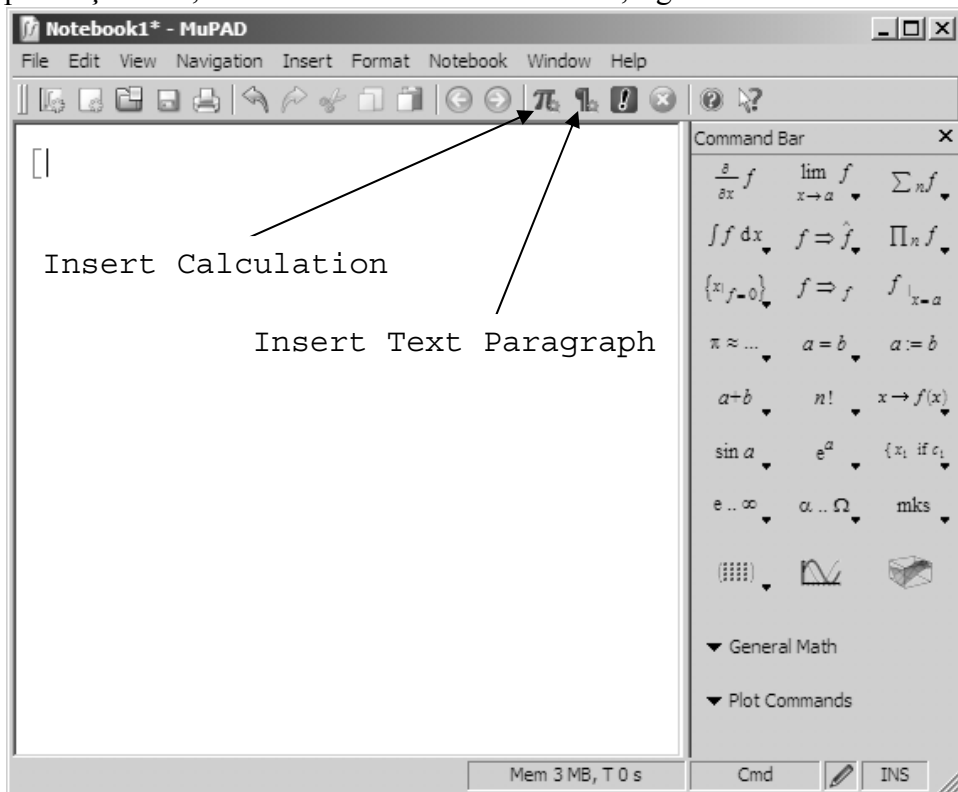
a) `Insert/Calculation`

b) `Insert/Calculation Above`

**Figura 11.3.** Introducerea unei noi linii de comenzi.

Introducerea textului explicativ (care nu este asociat instrucțiunilor și comenzilor MuPAD) se realizează cu ajutorul comenzilor `Insert/Text Paragraph` sau `Insert/Text Paragraph Above`, după cum se dorește ca noua linie de text explicativ să fie plasată după, sau înainte de linia curentă.

Comenzile Insert/Calculation (introducerea unei comenzi) și Insert/Text Paragraph (introducerea unui text explicativ) pot fi apelate și direct, din bara de butoane Standard, figura 11.4.

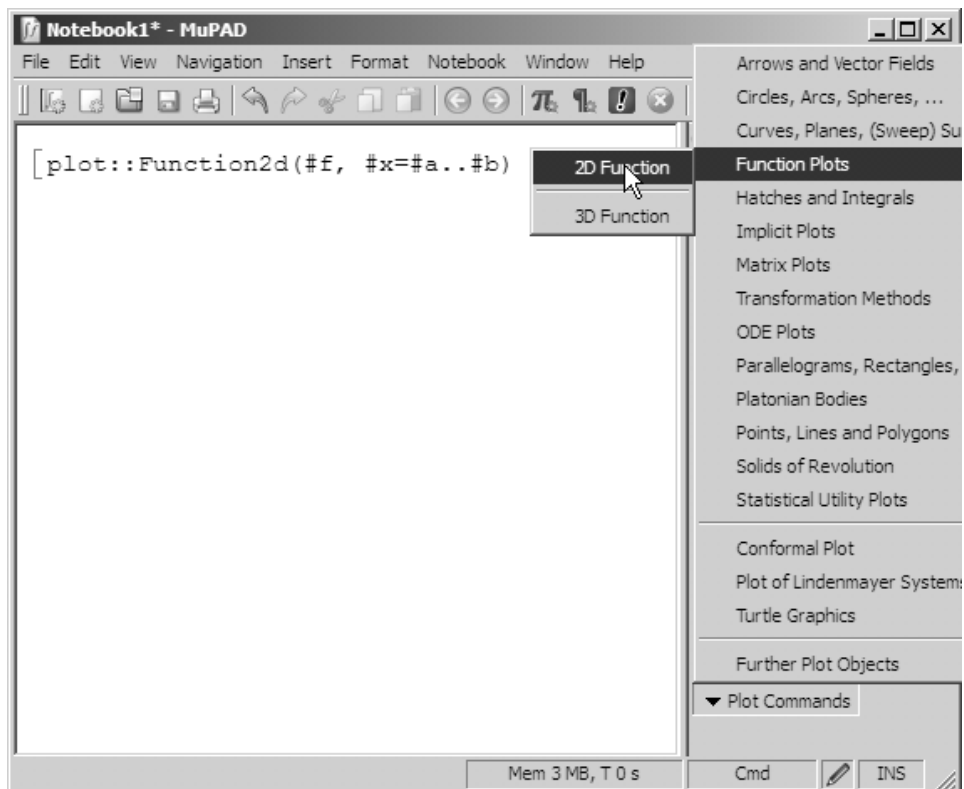


**Figura 11.4.** Comenzile Insert Calculation și Insert Text Paragraph.

Introducerea comenzilor în cadrul fișierelor MuPAD se poate face prin două metode:

- Direct, prin scrierea acestora în integralitate în fereastra de comenzi.
- Indirect, prin introducerea formatului general al comenzii din bara de butoane Command Bar, după care se procedează la modificarea corespunzătoare a parametrilor comenzii respective prin editare directă.

De exemplu, în figura 11.5 se prezintă metoda indirectă de introducere a comenzii de reprezentare a unui grafic 2D. Selectarea comenzii 2D Function din bara de comenzi Command Bar are ca efect introducerea în fereastra de comenzi a formatului general al instrucțiunii pentru reprezentare grafică, după care urmează modificare parametrilor care sunt precedați de caracterul # în concordanță cu funcția care va fi reprezentată grafic, precum și cu domeniul de definiție al acesteia.



**Figura 11.5.** Introducerea comenzilor din toolbar-ul Command Bar.

După introducerea tuturor comenzilor necesare pentru rezolvarea unei probleme, fișierul se salvează în directorul curent, în mod implicit cu extensia `.mn`.

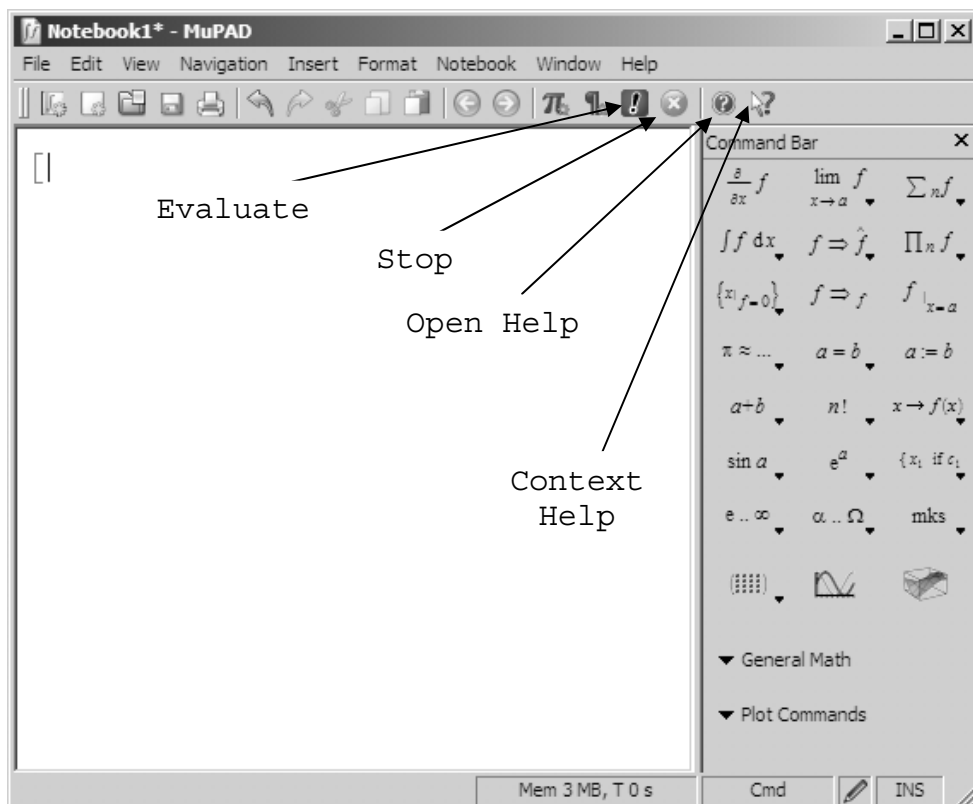
Executarea instrucțiunilor existente într-un fișier de tip MuPAD se poate face prin mai multe metode:

- Execuția instrucțiune cu instrucțiune - prin apăsarea tastei `Enter` pe fiecare instrucțiune care se dorește a fi executată. Același efect se poate obține cu ajutorul comenzii `Notebook/Evaluate` sau prin selectarea comenzii `Evaluate` din bara de butoane `Standard`, figura 11.6.
- Execuția tuturor instrucțiunilor pornind de la prima instrucțiune până la poziția curentă a cursorului: `Notebook/Evaluate From Beginning`.
- Execuția tuturor instrucțiunilor pornind de la poziția curentă a cursorului până la ultima instrucțiune din fișier: `Notebook/Evaluate To End`.
- Execuția tuturor instrucțiunilor din fișier, indiferent de poziția curentă a cursorului: `Notebook/Evaluate All`.

Înteruperea executării unei comenzi se poate realiza prin intermediul comenzii Stop din meniul Notebook sau din toolbar-ul Standard, figura 11.6.

Obținerea unor informații suplimentare despre comenzile MuPAD se poate face prin intermediul comenzii Open Help, din meniul Help sau din toolbar-ul Standard, figura 11.6. Baza de date cu informații despre modul de lucru și despre comenzile și parametrii de configurare specifici limbajului MuPAD este organizată sub forma unei structuri de fișiere de tip html. Căutarea unei anumite informații se poate face prin mai multe metode: pornind de la un cuprins general (Content), pornind de la un cuprins specific pentru funcțiile MuPAD (Topic), pornind de la o procedură de interogare a bazei de date pentru căutarea comenzilor (Commands) sau pornind de la o procedură de interogare a bazei de date pentru căutarea informațiilor generale (Search).

Obținerea de informații despre un anumit obiect selectabil în interfața de lucru MuPAD se poate face folosind comanda Context Help, din meniul Help sau din toolbar-ul Standard, figura 11.6.



**Figura 11.6.** Comenzile Evaluate, Stop, Open Help, Context Help.

## 11.3. DEFINIREA ȘI MANIPULAREA EXPRESIILOR

### 11.3.1. Definirea expresiilor simbolice

Definirea expresiilor simbolice se realizează cu instrucțiunea:

nume expresie simbolică:=expresie simbolică

în care: nume expresie simbolică este numele variabilei sau expresiei simbolice, expresie simbolică este expresia simbolică propriu-zisă, iar := este operatorul de atribuire, [5].

La sfârșitul instrucțiunii se poate tasta direct Enter, caz în care instrucțiunea se execută iar rezultatul se afișează pe ecran, sau se poate introduce caracterul :, după care se tastează Enter, caz în care instrucțiunea se execută, dar rezultatul nu se mai afișează pe ecran.

În figura 11.7 se prezintă câteva exemple de expresii simbolice.

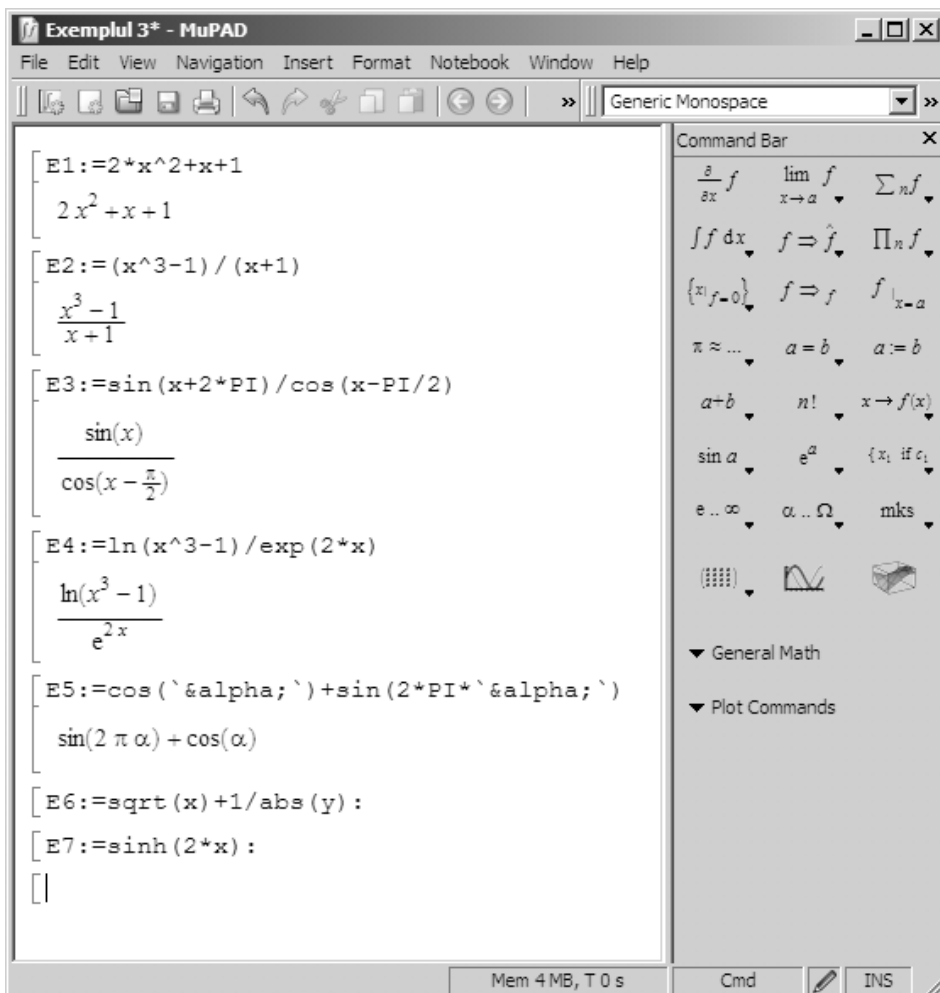


Figura 11.7. Definirea expresiilor simbolice.

## Observații

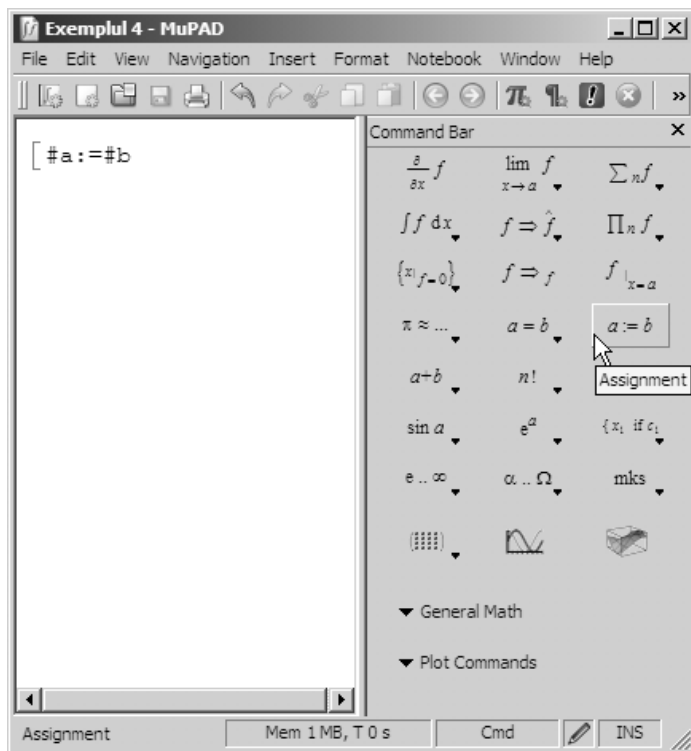
- Rezultatele obținute în urma executării fiecărei instrucțiuni de atribuire sunt scrise în mod implicit folosind culoarea albastră și respectă notația matematică uzuală, spre deosebire de instrucțiunile propriu-zise care sunt scrise în mod implicit folosind culoarea roșie și respectă sintaxa instrucțiunilor MuPAD. Modificarea culorii de afișare a comenzilor (input) și a rezultatelor executării acestora (output) se poate face folosind comanda `Color` din toolbarul `Format`.
- În cazul expresiei E3 se observă folosirea simbolului special (PI) corespunzător numărului  $\pi$ , precum și folosirea funcțiilor trigonometrice (`sin`, `cos`).
- În cazul expresiei E4 se observă folosirea funcției logaritm natural (`ln`) și a funcției exponențiale (`exp`).
- În cazul expresiei E5 se observă folosirea funcțiilor trigonometrice (`sin`, `cos`), dar și a caracterelor speciale corespunzătoare alfabetului grecesc. Obținerea caracterului  $\alpha$  se realizează prin selectarea comenzii corespunzătoare `&alpha;` din toolbar-ul `Command Bar`.
- În cazul instrucțiunilor E1, E2, E3, E4 și E5 s-a tastat direct `Enter` la sfârșitul expresiilor respective realizându-se atât execuția instrucțiunilor, cât și afișarea rezultatului obținut. Pentru expresiile E6 și E7 s-a introdus la sfârșitul expresiilor caracterul `:` după care s-a apăsat tasta `Enter`. În acest mod se realizează doar execuția expresiilor respective, rezultatele obținute nemaifiind afișate pe ecran.
- La completarea unei expresii, trecerea de la un parametru la altul se face prin apăsarea tastei `Tab`.

### 11.3.2. Comenzile toolbar-ului `Command Bar`

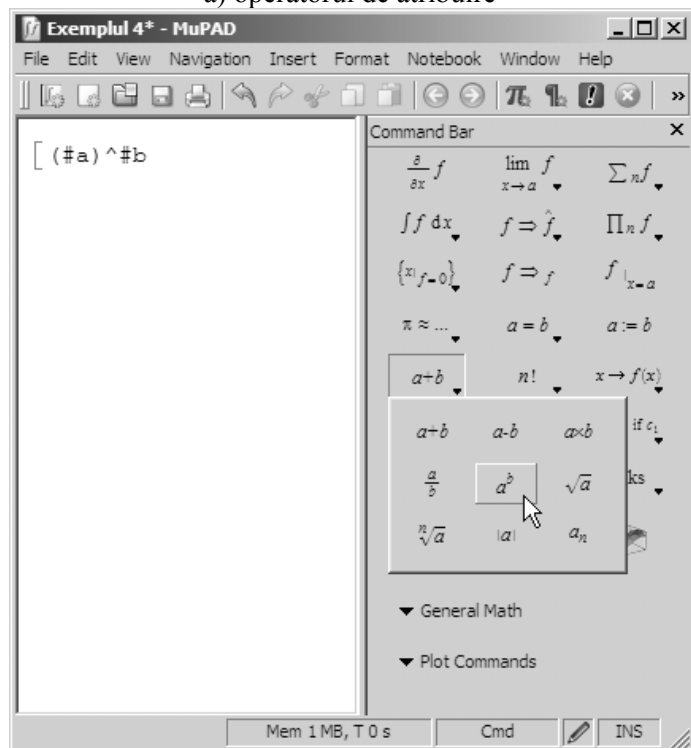
La definirea expresiilor simbolice se pot utiliza butoanele corespunzătoare unor comenzi specifice existente în toolbar-ul `Command Bar`. De exemplu, în figura 11.8 se prezintă modul de utilizare a principalelor comenzi existente în toolbar-ul `Command Bar`:

- Operatorul de atribuire, figura 11.8, a).
- Operații aritmetice, figura 11.8, b).
- Funcții trigonometrice, figura 11.8, c).
- Funcții exponențiale și logaritmice, figura 11.8, d).
- Simboluri speciale, figura 11.8, e).
- Caractere grecești, figura 11.8, f).

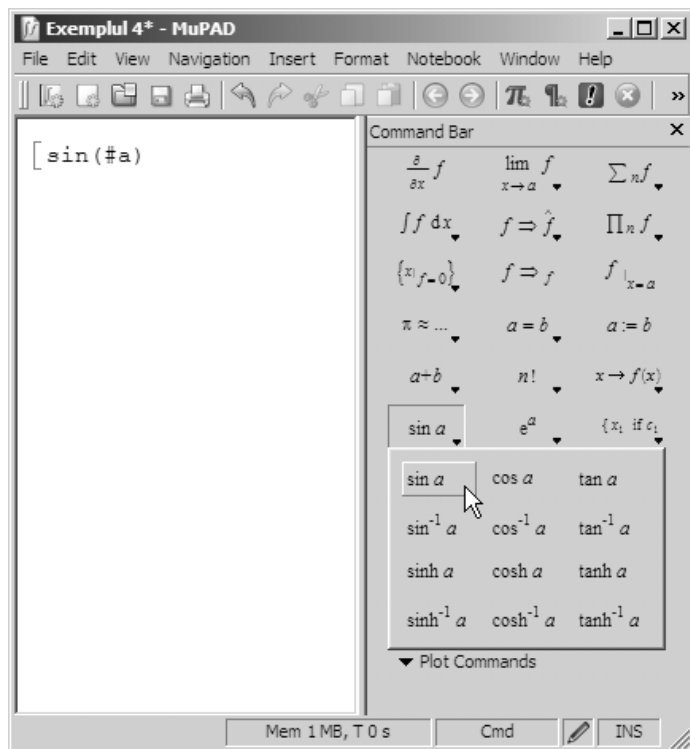




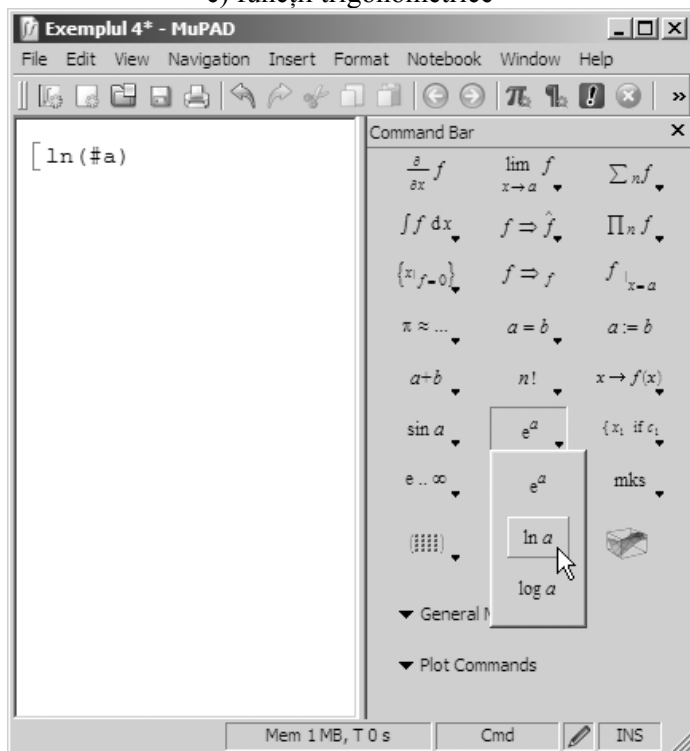
a) operatorul de atribuire



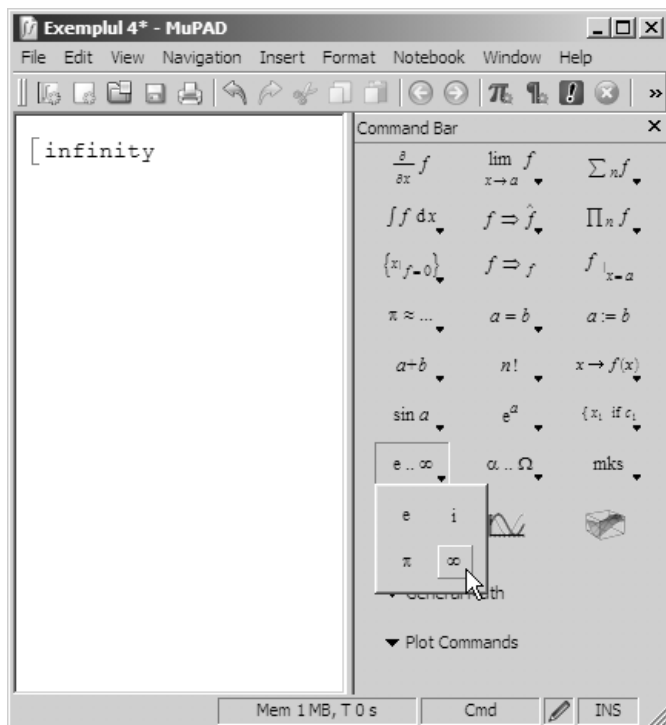
b) operații aritmetice



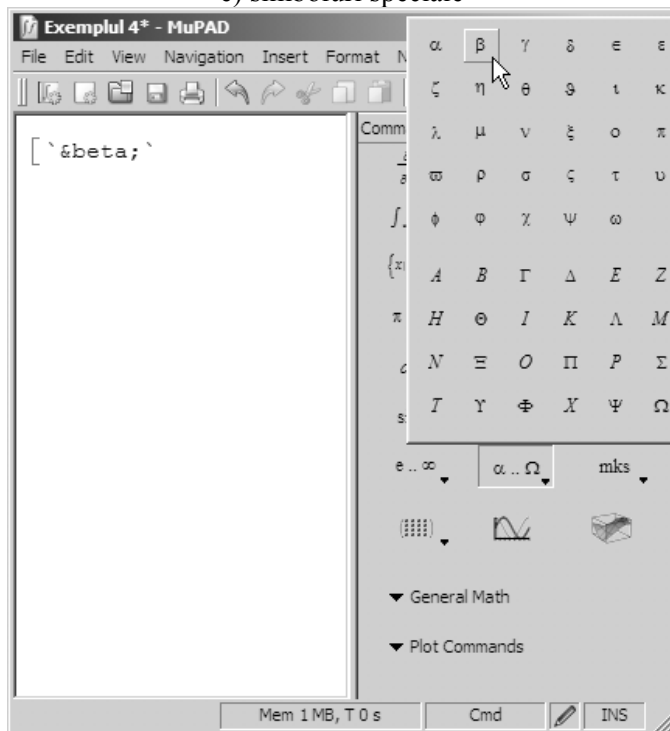
c) funcții trigonometrice



d) funcții exponențiale și logaritmice



e) simboluri speciale



f) caractere grecești

Figura 11.8. Principalele comenzi ale toolbar-ului Command Bar.

### 11.3.3. Definirea expresiilor complexe

Pentru definirea expresiilor folosind numere complexe, se consideră următoarele observații:

- La definirea numerelor complexe ( $z = x + i \cdot y$ ) se utilizează unitatea imaginară notată cu  $i$ .
- Partea reală a unui număr complex se determină cu instrucțiunea:

$$x := \text{Re}(z)$$

- Partea imaginară a unui număr complex se determină cu instrucțiunea:

$$y := \text{Im}(z)$$

- Valoarea absolută a unui număr complex ( $r = \sqrt{x^2 + y^2}$ ) se determină cu instrucțiunea:

$$r := \text{abs}(z)$$

- Argumentul unui număr complex ( $f = \text{arctg}(y/x)$ ) se determină cu instrucțiunea:

$$f := \text{arg}(z)$$

- Cunoscând modulul  $r$  și argumentul  $f$ , numărul complex  $z$  corespunzător se poate defini folosind relațiile:

$$z = r \cdot e^{if}$$
$$z = r \cdot (\cos f + i \cdot \sin f)$$

### 11.3.4. Definirea ipotezelor

În mod implicit, la definirea variabilelor și expresiilor simbolice se presupune că acestea aparțin mulțimii numerelor complexe  $\mathbb{C}$ . În anumite cazuri se impune însă efectuarea unor precizări suplimentare asupra tipului variabilelor simbolice, prin intermediul aplicării unor ipoteze specifice.

Definirea ipotezelor se realizează prin instrucțiunea [6]:

```
assume(nume expresie simbolică, proprietate)
```

în care: `nume expresie simbolică` este numele variabilei sau expresiei simbolice, iar `proprietate` este condiția care se va aplica variabilei simbolice respective.

Principalele proprietăți care se referă la condițiile matematice (de tip domeniu sau de tip interval) care se pot aplica diferitelor variabile și expresii simbolice sunt:

- `Type := Complex`, mulțimea numerelor complexe,  $\mathbb{C}$ .

- `Type::Even`, mulțimea numerelor întregi pare,  $2\mathbb{Z}$ .
- `Type::Imaginary`, mulțimea numerelor imaginare,  $\mathbb{R} \cdot i$ .
- `Type::Integer`, mulțimea numerelor întregi,  $\mathbb{Z}$ .
- `Type::Negative`, mulțimea numerelor reale negative,  $\mathbb{R}_{<0}$ .
- `Type::NegInt`, mulțimea numerelor întregi negative,  $\mathbb{Z}_{<0}$ .
- `Type::NegRat`, mulțimea numerelor raționale negative,  $\mathbb{Q}_{<0}$ .
- `Type::NonNegative`, mulțimea numerelor reale mai mari sau egale cu zero,  $\mathbb{R}_{\geq 0}$ .
- `Type::NonNegInt`, mulțimea numerelor întregi mai mari sau egale cu zero,  $\mathbb{Z}_{\geq 0}$ .
- `Type::NonNegRat`, mulțimea numerelor raționale mai mari sau egale cu zero,  $\mathbb{Q}_{\geq 0}$ .
- `Type::NonZero`, mulțimea numerelor diferite de zero,  $\mathbb{C} - \{0\}$ .
- `Type::Odd`, mulțimea numerelor întregi impare,  $2\mathbb{Z} + 1$ .
- `Type::PosInt`, mulțimea numerelor întregi strict pozitive,  $\mathbb{Z}_{>0}$ .
- `Type::Positive`, mulțimea numerelor reale strict pozitive,  $\mathbb{R}_{>0}$ .
- `Type::PosRat`, mulțimea numerelor raționale strict pozitive,  $\mathbb{Q}_{>0}$ .
- `Type::Rational`, mulțimea numerelor raționale,  $\mathbb{Q}$ .
- `Type::Real`, mulțimea numerelor reale,  $\mathbb{R}$ .
- `Type::Zero`, numărul zero,  $\{0\}$ .
- `Type::Interval(a, b, M)`, definește valorile corespunzătoare intervalului  $\{x \in M, a < x < b\}$ .
- `Type::Interval([a], b, M)`, definește valorile corespunzătoare intervalului  $\{x \in M, a \leq x < b\}$ .
- `Type::Interval(a, [b], M)`, definește valorile corespunzătoare intervalului  $\{x \in M, a < x \leq b\}$ .
- `Type::Interval([a], [b], M)`, definește valorile corespunzătoare intervalului  $\{x \in M, a \leq x \leq b\}$ .

Fiind dată o expresie simbolică oarecare, obținerea tipului expresiei respective se poate face cu ajutorul instrucțiunii:

```
getprop(expresie simbolică)
```

Aplicarea a două ipoteze diferite asupra aceleiași expresii simbolice se face în două etape: `assume`, pentru prima proprietate și apoi `assumeAlso` pentru cea de-a doua proprietate (ambele proprietăți activează simultan). În figura 11.9 se prezintă câteva exemple reprezentative de aplicare a unor ipotezelor (de tip domeniu și de tip interval) asupra variabilelor simbolice.

```

[ assume(x, Type::Positive)
[ getprop(x)
(0, ∞)
[ assume(x, Type::PosInt)
[ getprop(x)
 $\mathbb{Z} \cap [1, \infty)$ 
[ assume(x, Type::Real) : assume(y, Type::Real)
z:=x+I*y
x+yi
[ getprop(z)
 $\mathbb{C}$ 
[ assume(x, Type::Real) : assume(y, Type::Imaginary)
z:=x+I*y
x+yi
[ getprop(z)
 $\mathbb{R}$ 
[ assume(x>=0) : assumeAlso(x<=10)
[ getprop(x)
[0, 10]

```

Figura 11.9. Aplicarea ipotezelor de tip domeniu și de tip interval.

### 11.3.5. Operații de manipulare a expresiilor simbolice

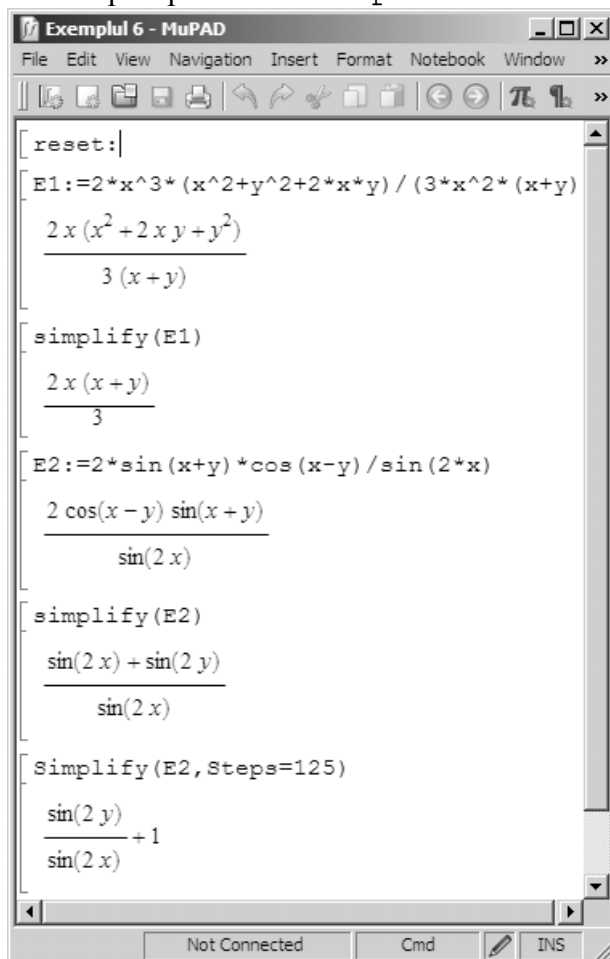
Atât asupra expresiilor simbolice definite de utilizator, cât și a celor rezultate din calcule simbolice anterioare se pot efectua o serie de operații simbolice grupate sub denumirea de operații de manipulare a expresiilor simbolice.

Principalele operații de manipulare a expresiilor simbolice sunt:

- **Aducerea la o formă mai simplă a expresiilor simbolice.** Reprezintă operațiile care au ca scop aducerea unei expresii simbolice oarecare la o formă echivalentă, mai simplă. Instrucțiunile care permit aducerea la o formă mai simplă a expresiilor simbolice sunt:
  - `simplify(expresie simbolică)`, [7]. Este metoda prin care se caută o formă mai simplă a unei expresii

simbolice prin rescrierea termenilor expresiei inițiale (figura 11.10). Procedura de rescriere se aplică, automat, de mai multe ori, la fiecare nouă aplicare căutându-se o formă mai simplă față de forma anterioară.

- `Simplify`(*expresie simbolică*), [8]. Este o metodă mai avansată de căutare a unei forme mai simple a unei expresii simbolice (figura 11.10). Și în acest caz procedura de simplificare se aplică în mod automat, de mai multe ori, doar că la fiecare nouă aplicare se caută o formă mai simplă față de toate formele anterioare, obținându-se în final cea mai bună formă simplificată. În mod implicit, instrucțiunea de aducere la o formă mai simplă `Simplify` lucrează cu maxim 100 de etape. Numărul maxim de etape se poate modifica prin parametrul `Steps`.



**Figura 11.10.** Aplicarea instrucțiunilor de aducere la o formă mai simplă `simplify` și `Simplify`.

- Radsimp, [9]. Este metoda de aducere la o formă mai simplă a expresiilor simbolice cu radicali (figura 11.11). Prin această metodă se încearcă să se determine cea mai simplă formă posibilă a unei expresii prin simplificarea radicalilor.

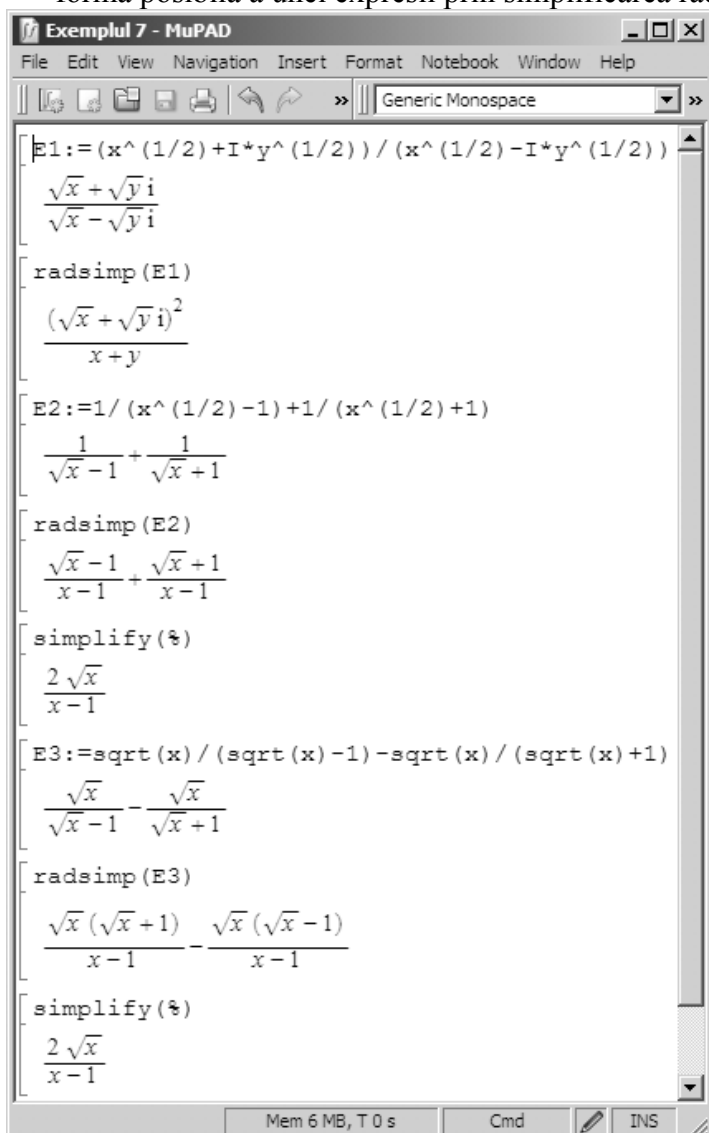


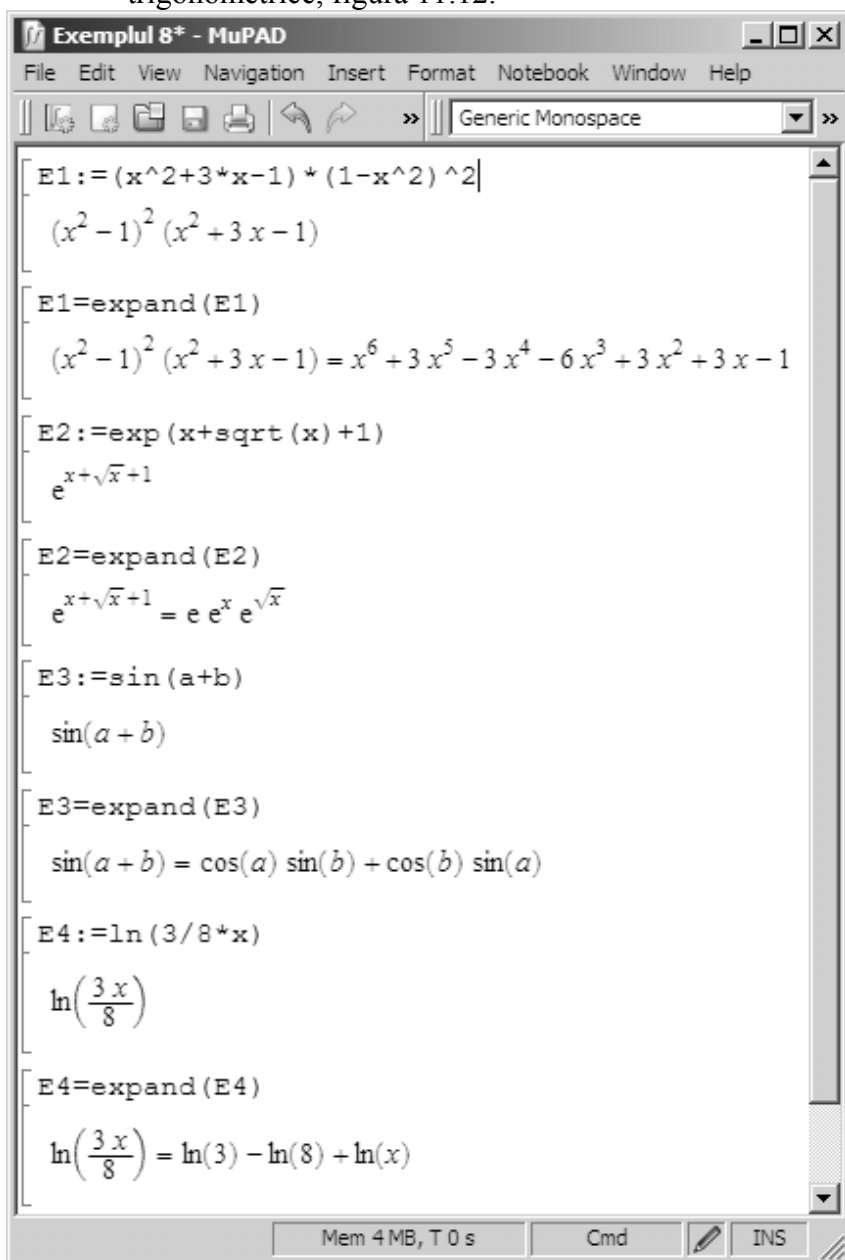
Figura 11.11. Aplicarea instrucțiunii radsimp.

- **Transformarea expresiilor simbolice.** Reprezintă operațiile care au ca scop aducerea unei expresii simbolice la o formă echivalentă, mai simplă, dar având un anumit format impus (produs de termeni, sumă de termeni, gruparea după puterile unei variabile), sau identificarea anumitor funcții din structura unei expresii simbolice și înlocuirea lor completă cu alte funcții.



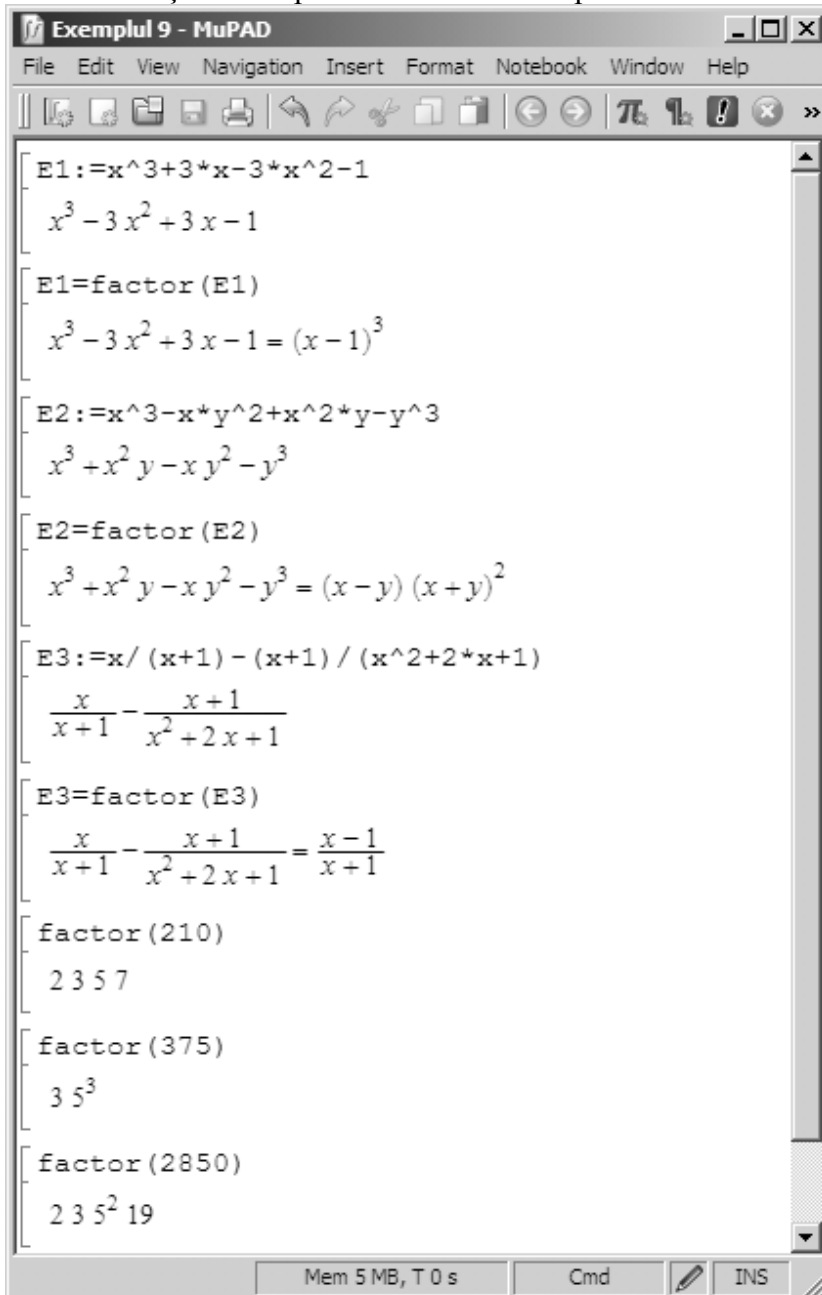
Instrucțiunile care permit transformarea expresiilor simbolice sunt:

- o `expand(expresie simbolică)`, [10]. Este metoda prin care se caută o formă echivalentă a unei expresii simbolice prin transformarea, în general, a unui produs de sume într-o sumă de produse. Instrucțiunea are ca efect și transformarea unor expresii logaritmice, exponențiale și trigonometrice, figura 11.12.



**Figura 11.12.** Aplicarea instrucțiunii de transformare `expand`.

- o `factor`(expresie simbolică), [11]. Este metoda prin care se caută o formă echivalentă a unei expresii simbolice prin transformarea, în general, a unei sume de produse într-un produs de sume, figura 11.13. Instrucțiunea are ca efect aducerea la același numitor a mai multor fracții, dar și descompunerea în factori simpli a numerelor.



**Figura 11.13.** Aplicarea instrucțiunii de transformare `factor`.

- o `collect`(expresie simbolică, variabila simbolică), [12]. Este metoda prin care se caută o formă echivalentă a unei expresii simbolice prin gruparea termenilor din expresia simbolică (expresie simbolică) după puterile variabilei simbolice specificate (variabila simbolică), figura 11.14. În cazul fracțiilor simbolice, instrucțiunea `collect` se aplică în mod independent pentru numitorul și numărătorul fracției.

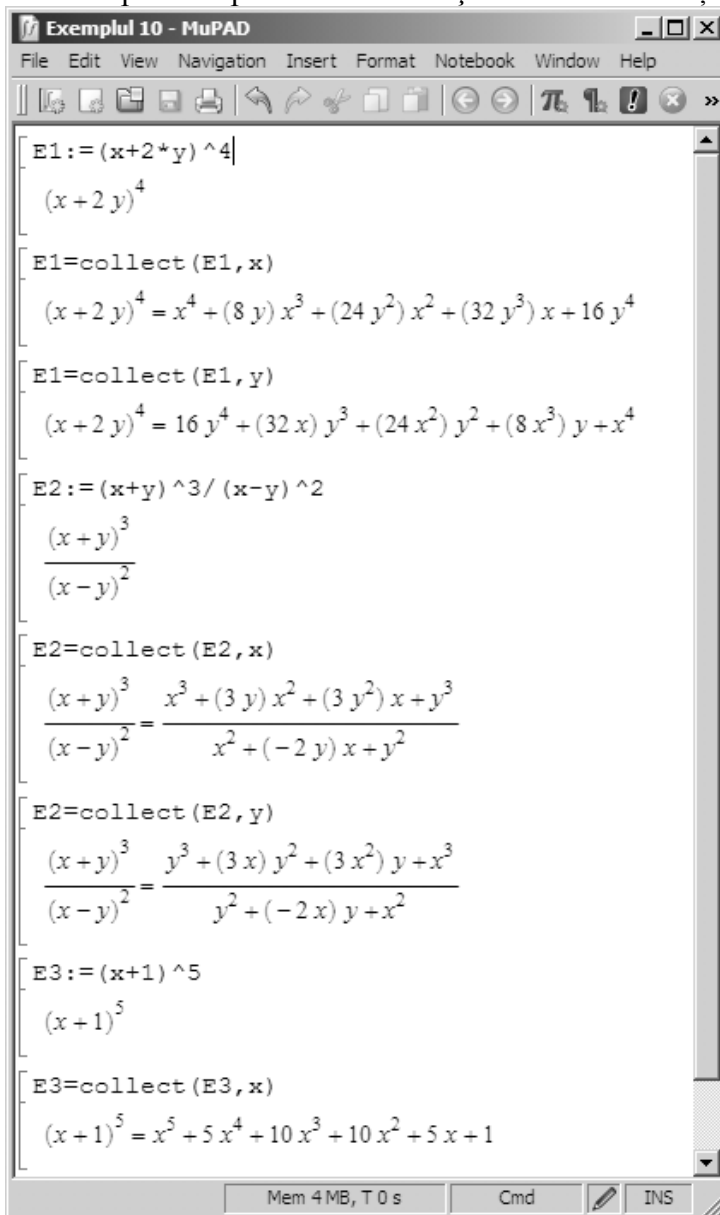
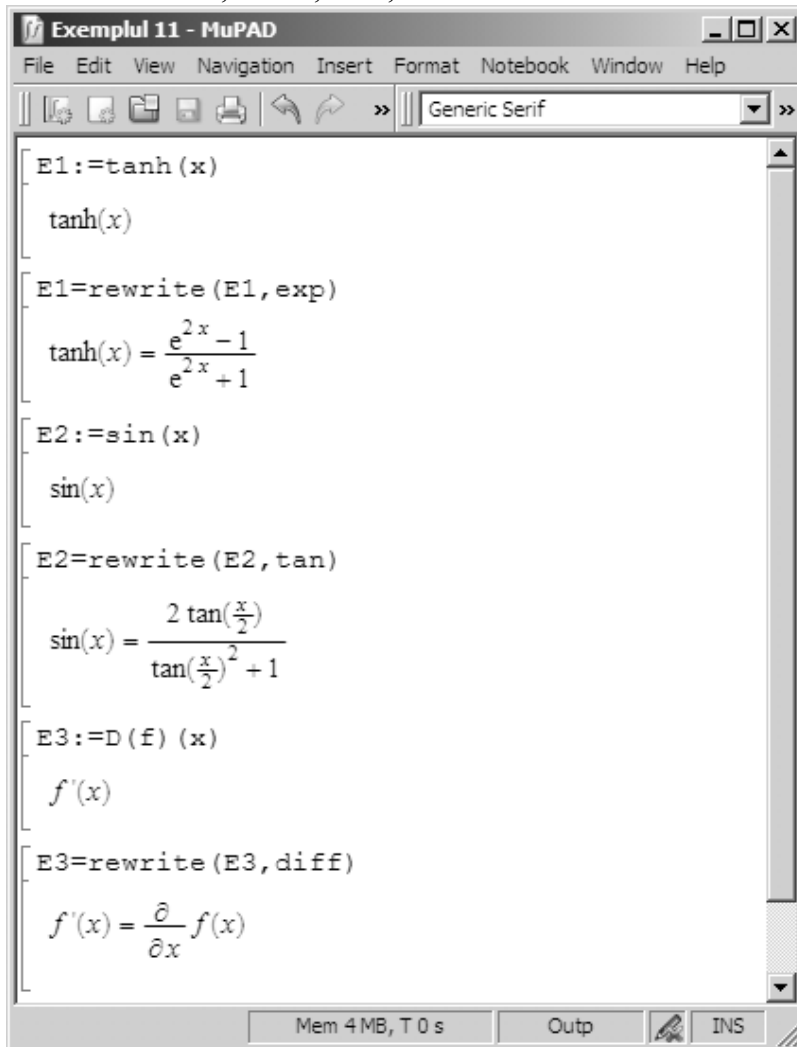


Figura 11.14. Aplicarea instrucțiunii de transformare `collect`.

- o `rewrite`(expresie simbolică, funcție simbolică), [13]. Este metoda prin care se caută o formă echivalentă a unei expresii simbolice prin rescrierea expresiei simbolice (expresie simbolică) în raport cu funcția simbolică specificată (funcție simbolică), figura 11.15. Principalele variabile simbolice utilizate la transformarea expresiilor cu instrucțiunea `rewrite` sunt: `arccos`, `arccosh`, `arccot`, `arccoth`, `arcsin`, `arcsinh`, `arctan`, `arctanh`, `cos`, `cosh`, `cot`, `coth`, `diff`, `D`, `erf`, `erfc`, `erfi`, `exp`, `ln`, `max`, `min`, `sin`, `sincos`, `sinh`, `tan`, `tanh`.



**Figura 11.15.** Aplicarea instrucțiunii de transformare `rewrite`.

- o `rectform`(expresie simbolică), [14]. Este metoda prin care se caută o formă echivalentă a unei expresii simbolice complexe  $E(z)$ ,  $z = x + i \cdot y$ , prin explicitarea părții reale și a părții imaginare, în concordanță cu forma carteziană a numerelor complexe  $E(z) = \Re(E) + i \cdot \Im(E)$ , figura 11.16 (în care  $\Re(E)$  este partea reală a numărului complex iar  $\Im(E)$  este partea imaginară a numărului complex). Instrucțiunea `rectform(E)` este mai puternică decât instrucțiunile simple utilizate pentru determinarea părții reale  $\text{Re}(E)$  și a părții imaginare  $\text{Im}(E)$  a unei expresii simbolice. Pentru identificarea corectă a părții reale, respectiv a părții imaginare a unei expresii simbolice  $E(z)$  este necesară efectuarea unor ipoteze suplimentare asupra variabilelor  $x$  și  $y$ , de tipul `assume(x, Type::Real)` și `assume(y, Type::Real)`.

```

[ reset:
[ assume(x, Type::Real) : assume(y, Type::Real) :
[ E1:=sin(x+I*y) :
[ E1r:=rectform(E1)
  cosh(y) sin(x) + (cos(x) sinh(y)) i
[ E2:=sin(z) :
[ E2=rectform(E2)
  sin(z) = cosh(Im(z)) sin(Re(z)) + (cos(Re(z)) sinh(Im(z))) i
[ Re(E2) ; Im(E2) ;
  Re(sin(z))
  Im(sin(z))

```

Figura 11.16. Aplicarea instrucțiunii de transformare `rectform`.

- o `denom`(expresie simbolică), [15] și `numer`(expresie simbolică), [16]. Sunt metodele prin care se extrag numitorul și numărătorul unei expresii simbolice, figura 11.17.

```

Exemplul 13 - MuPAD
File Edit View Navigation Insert Format Notebook Window Help
[Icons]
E1:=(x^3+2*x-1)/(x^2-1)

$$\frac{x^3+2x-1}{x^2-1}$$

NumitorE1:=denom(E1)

$$x^2-1$$

NumaratorE1:=numer(E1)

$$x^3+2x-1$$

E2:=exp(x)/(ln(x)+1)+1/x

$$\frac{1}{x} + \frac{e^x}{\ln(x)+1}$$

NumitorE2:=denom(E2)

$$x+x \ln(x)$$

NumaratorE2:=numer(E2)

$$\ln(x)+x e^x+1$$

E3:=x^2+(x-1)/(x+1)

$$\frac{x-1}{x+1}+x^2$$

NumitorE3:=denom(E3)

$$x+1$$

NumaratorE3:=numer(E3)

$$x^3+x^2+x-1$$

Mem 4 MB, T 0 s Cmd INS

```

**Figura 11.17.** Aplicarea instrucțiunilor `denom` și `numer`.

## 11.4. CALCULUL SIMBOLIC AL LIMITELOR, DERIVATELOR ȘI INTEGRALELOR

### 11.4.1. Calculul simbolic al limitelor

Pentru calculul limitelor expresiilor simbolice se utilizează instrucțiunea `limit`, [17]. Principalele implementări ale instrucțiunii `limit` permit efectuarea următoarelor evaluări simbolice:

- În general, pentru calculul limitei unei expresii simbolice  $E(x)$  cu  $x \rightarrow x_0$  de forma  $\lim_{x \rightarrow x_0} E(x)$  se utilizează instrucțiunea:

```
limit (E, x=x0)
```

- Pentru calculul limitei unei expresii simbolice  $E(x)$  cu  $x \rightarrow \infty$  de forma  $\lim_{x \rightarrow \infty} E(x)$  se utilizează instrucțiunea:

```
limit (E, x=infinity)
```

- Pentru calculul limitei unei expresii simbolice  $E(x)$  cu  $x \rightarrow -\infty$  de forma  $\lim_{x \rightarrow -\infty} E(x)$  se utilizează instrucțiunea:

```
limit (E, x=-infinity)
```

- Pentru calculul limitei unei expresii simbolice  $E(x, y)$  cu  $x \rightarrow x_0$  de forma  $\lim_{x \rightarrow x_0} E(x, y)$  se utilizează instrucțiunea:

```
limit (E, x=x0)
```

- Pentru calculul limitei unei expresii simbolice  $E(x, y)$  cu  $y \rightarrow y_0$  de forma  $\lim_{y \rightarrow y_0} E(x, y)$  se utilizează instrucțiunea:

```
limit (E, y=y0)
```

- Pentru calculul limitei la stânga a unei expresii simbolice  $E(x)$  cu  $x \rightarrow x_0^-$  de forma  $\lim_{x \rightarrow x_0^-} E(x)$  se utilizează instrucțiunea:

```
limit (E, x=x0, Left)
```

- Pentru calculul limitei la dreapta a unei expresii simbolice  $E(x)$  cu  $x \rightarrow x_0^+$  de forma  $\lim_{x \rightarrow x_0^+} E(x)$  se utilizează instrucțiunea:

```
limit (E, x=x0, Right)
```

Modul de utilizare al instrucțiunii pentru calculul simbolic al limitelor (`limit`) în diferite cazuri concrete este prezentat în figura 11.18. Pentru fiecare caz în parte, se definește mai întâi expresia simbolică folosind operatorul de atribuire, după care se aplică instrucțiunea `limit`.

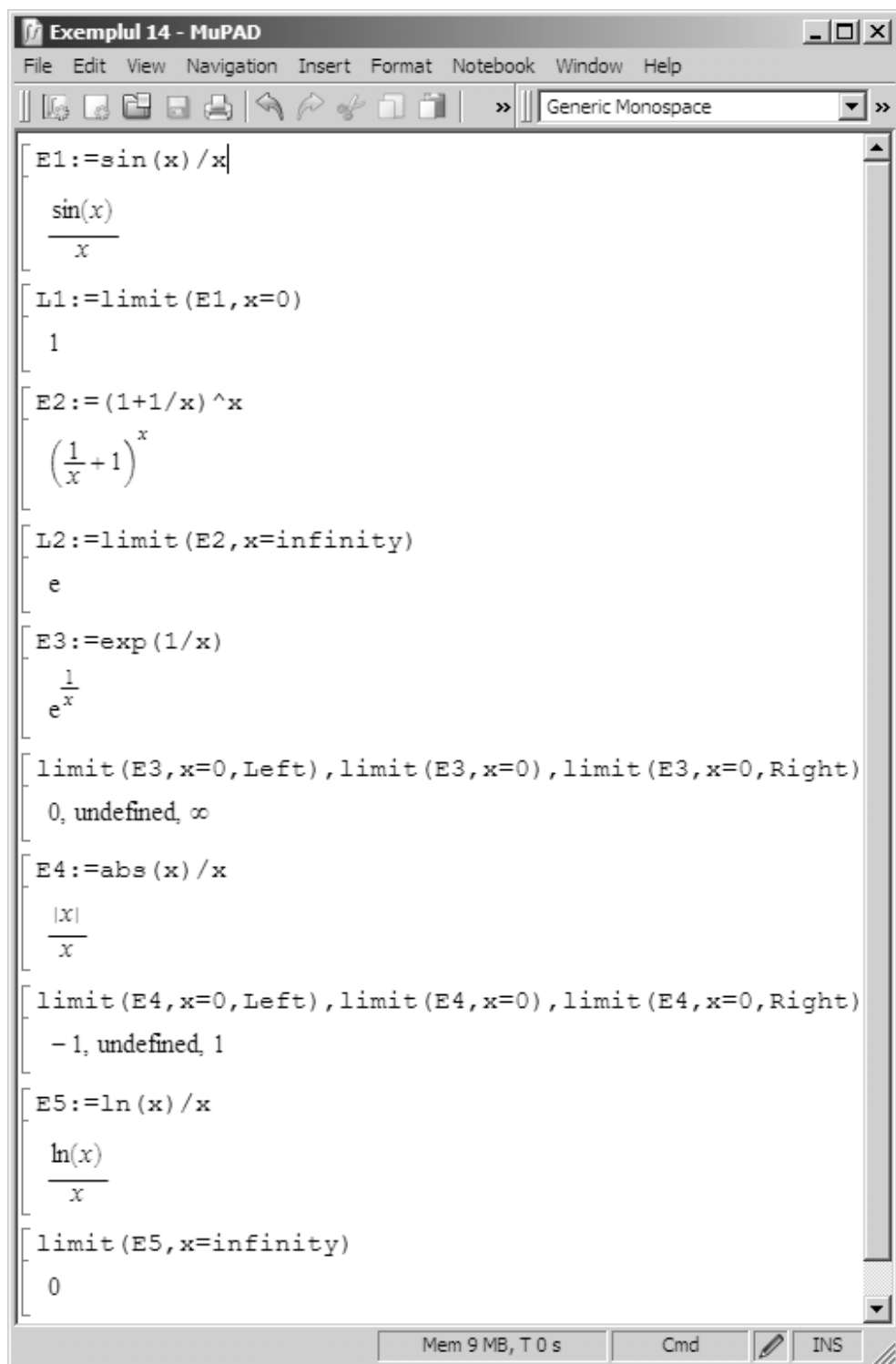


Figura 11.18. Utilizarea instrucțiunii limit.



### 11.4.2. Calculul simbolic al derivatelor

Pentru calculul derivatelor expresiilor simbolice se utilizează instrucțiunea `diff`, [18]. Principalele implementări ale instrucțiunii `diff` permit efectuarea următoarelor evaluări simbolice:

- Pentru calculul derivatei unei expresii simbolice  $E(x)$  în raport cu variabila  $x$ ,

$$\frac{d}{dx} E(x)$$

se utilizează instrucțiunea:

```
diff (E, x)
```

- Pentru calculul derivatei a doua a unei expresii simbolice  $E(x)$  în raport cu variabila  $x$ ,

$$\frac{d^2}{dx^2} E(x)$$

se pot utiliza instrucțiunile:

```
diff (diff (E, x) , x)
```

```
diff (E, x, 2)
```

- În general, pentru calculul derivatei de ordinul  $n$  pentru expresia simbolică  $E(x)$  în raport cu variabila  $x$

$$\frac{d^n}{dx^n} E(x)$$

se utilizează instrucțiunea:

```
diff (E, x, n)
```

- Pentru calculul derivatei parțiale a unei expresii simbolice  $E(x, y)$  în raport cu variabila  $x$

$$\frac{\partial}{\partial x} E(x, y)$$

se utilizează instrucțiunea:

```
diff (E, x)
```

- Pentru calculul derivatei parțiale a unei expresii simbolice  $E(x, y)$  în raport cu variabila  $y$

$$\frac{\partial}{\partial y} E(x, y)$$

se utilizează instrucțiunea:

```
diff (E, y)
```

Modul de utilizare al instrucțiunii pentru calculul simbolic al derivatelor (`diff`) în diferite cazuri concrete este prezentat în figura 11.19.

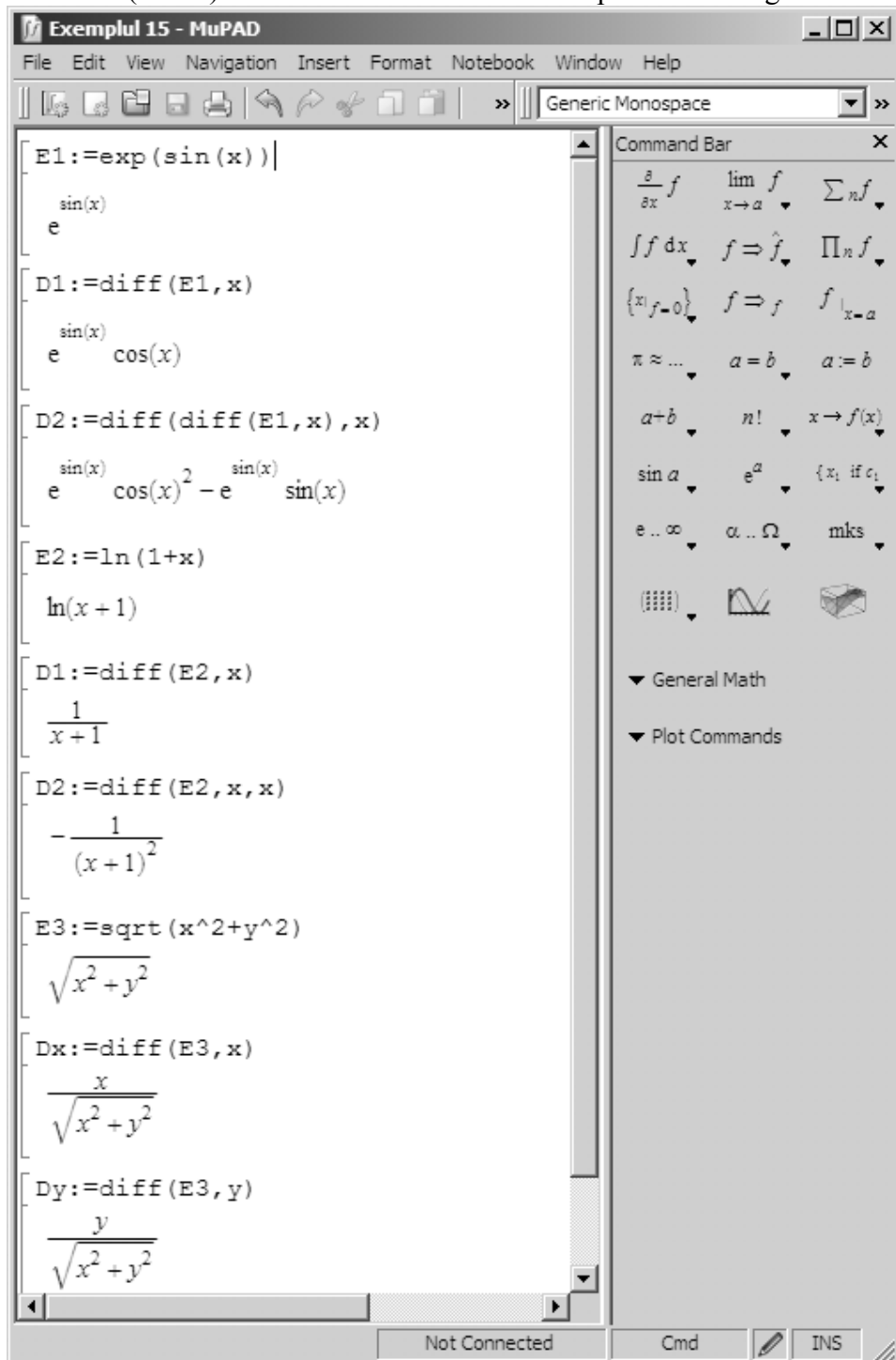


Figura 11.19. Utilizarea instrucțiunii `diff`.

### 11.4.3. Calculul simbolic al integralelor

Pentru calculul integralelor expresiilor simbolice se utilizează instrucțiunea `int`, [19]. Principalele implementări ale instrucțiunii `int` permit efectuarea următoarelor evaluări simbolice:

- Calculul integralei nedefinite  $\int f(x)dx$  a unei funcții simbolice  $f(x)$  în raport cu variabila  $x$  se face cu instrucțiunea:

`int (f, x)`

- Calculul integralei definite  $\int_a^b f(x)dx$  a unei funcții simbolice  $f(x)$  în raport cu variabila  $x$ , între limitele de integrare  $x = a$  și  $x = b$  se face cu instrucțiunea:

`int (f, x=a..b)`

- Calculul integralei duble definite  $\int_{x_1}^{x_2} dx \int_{y_1}^{y_2} f(x, y) dy$  a unei funcții simbolice  $f(x, y)$  în raport cu variabilele  $x = x_1 \dots x_2$  și  $y = y_1 \dots y_2$  se face cu instrucțiunea:

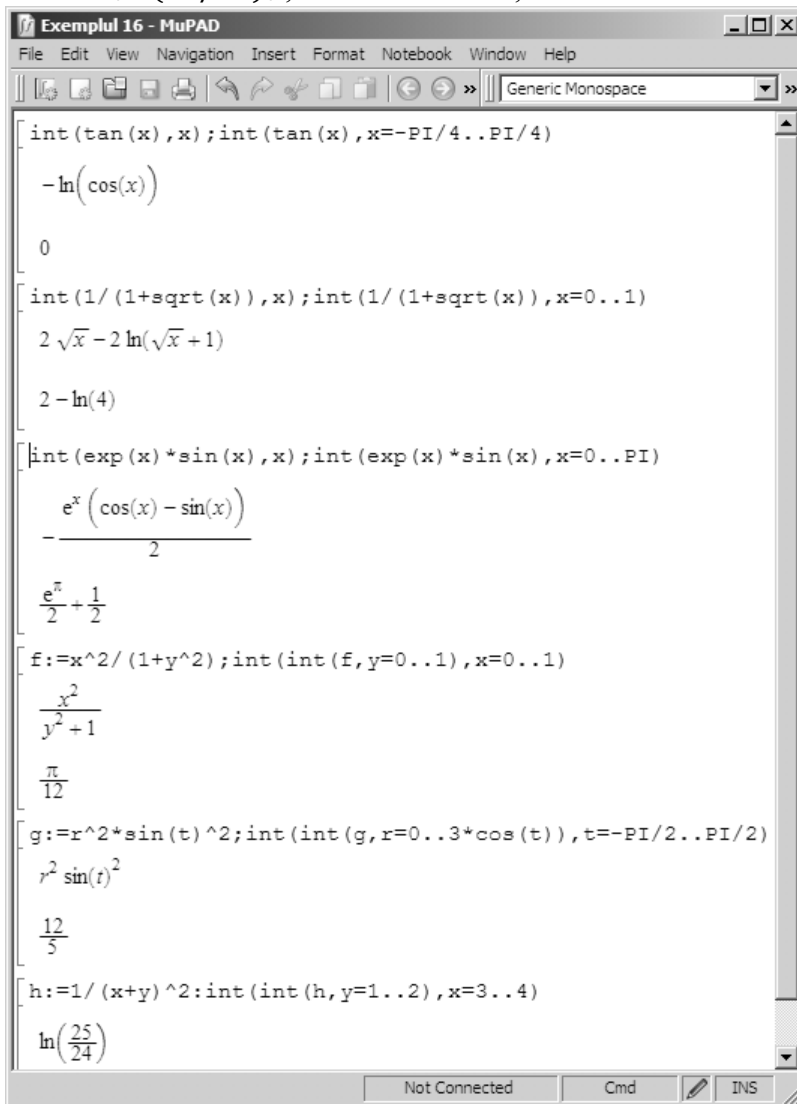
`int (int (f, y=y1..y2), x=x1..x2)`

Modul de utilizare al instrucțiunii `int` pentru calculul simbolic al integralelor în diferite cazuri concrete este prezentat în figura 11.20.

#### Observații

- În primul caz se calculează integrala nedefinită  $\int \operatorname{tg} x dx$  obținându-se rezultatul  $-\ln(\cos x)$ , precum și integrala definită  $\int_{-\pi/2}^{+\pi/2} \operatorname{tg} x dx$  obținându-se rezultatul 0.
- În cel de-al doilea caz se calculează integrala nedefinită  $\int \frac{1}{1+\sqrt{x}} dx$ , precum și integrala definită  $\int_0^1 \frac{1}{1+\sqrt{x}} dx$  pentru care se obține rezultatul simbolic  $2 - \ln 4$ . Valoarea numerică aproximativă se determină cu `float (2 - ln 4)`, rezultatul fiind 0,6137056389.
- În cel de-al treilea caz se calculează integrala nedefinită  $\int e^x \sin x dx$ , precum și integrala definită  $\int_0^\pi e^x \sin x dx$  pentru care se obține rezultatul simbolic  $(e^\pi + 1)/2$ . Valoarea numerică aproximativă a rezultatului simbolic se obține cu instrucțiunea `float ((e^\pi + 1)/2)`, rezultatul fiind 12,07034632.
- În cel de-al patrulea caz se definește expresia simbolică  $f(x, y) = x^2/(1 + y^2)$  și se calculează integrala dublă  $\int_0^1 dx \int_0^1 f(x, y) dy$  obținându-se rezultatul simbolic  $\pi/12$ . Valoarea numerică aproximativă a rezultatului simbolic se obține cu instrucțiunea `float (\pi/12)`, rezultatul fiind 0,2617993878.

- În cel de-al cincilea caz se definește expresia simbolică  $g(r, t) = r^2 \sin t^2$  și se calculează integrala dublă  $\int_{-\pi/2}^{+\pi/2} dt \int_0^{3 \cos t} g(r, t) dr$  obținându-se rezultatul simbolic  $12/5$ . Valoarea numerică aproximativă a rezultatului simbolic se obține cu instrucțiunea `float(12/5)`, rezultatul fiind 2,4.
- În cel de-al șaselea caz se definește expresia simbolică  $h(x, y) = \frac{1}{(x+y)^2}$  și se calculează integrala dublă  $\int_3^4 dx \int_1^2 h(x, y) dy$  obținându-se rezultatul simbolic  $\ln(25/24)$ . Valoarea numerică aproximativă a rezultatului simbolic se obține cu instrucțiunea `float(ln(25/24))`, rezultatul fiind 0,04082199452.



**Figura 11.20.** Utilizarea instrucțiunii `int`.

## 11.5. REPREZENTAREA GRAFICĂ A EXPRESIILOR SIMBOLICE

Reprezentarea grafică a expresiilor simbolice presupune parcurgerea mai multor etape:

- Definirea expresiei simbolice. Pentru expresii simbolice simple se folosește operatorul de atribuire  $:=$ . De exemplu, pentru definirea expresiei simbolice  $E_1 = x^2 + 1$  se folosește instrucțiunea:

```
E1 := x^2 + 1
```

Pentru funcții se folosește operatorul de atribuire  $:=$  combinat cu operatorul  $->$ . Înaintea operatorului  $->$  se introduc variabilele funcției, iar după operatorul  $->$  se introduce expresia funcției. De exemplu, pentru definirea funcției  $E_2(x, y) = x^2 + y - 2x$  se utilizează următoarea instrucțiune:

```
E2 := (x, y) -> x^2 + y - 2*x
```

- Definirea obiectului grafic și configurarea parametrilor caracteristici ai acestuia. Obiectele grafice disponibile se găsesc în comanda Plot Commands din toolbar-ului Command Bar.

Principalele obiecte grafice sunt:

- Functions Plot pentru reprezentarea grafică a funcțiilor 2D, respectiv a funcțiilor 3D.
- Points, Lines and Polygons pentru reprezentarea grafică a punctelor, liniilor și poligoanelor 2D și 3D.
- Curves, Planes, (Sweep) Surfaces pentru reprezentarea grafică a curbilor 2D și 3D, a planelor și suprafețelor în coordonate carteziane, coordonate cilindrice și coordonate sferice.
- Circles, Arcs, Spheres pentru reprezentarea grafică a arcelor de cerc, a cercurilor, a elipselor, a sferelor și a elipsoizilor.
- Implicit Plots pentru reprezentarea grafică a funcțiilor 2D și 3D exprimate sub formă implicită.
- Solids of Revolution pentru reprezentarea grafică a suprafețelor de revoluție obținute prin rotația unei curbe plane în jurul unei axe.
- Platonian Bodies pentru reprezentarea grafică a poliedrelor regulate: tetraedrul, hexaedrul, octaedrul, dodecaedrul, icosaedrul (corpurile Platonice).
- Parallelograms, Rectangles, Boxes pentru reprezentarea grafică a paralelogramului 2D și 3D, a dreptunghiului, a conului și a cilindrului.

- Hatch and Integrals pentru reprezentarea grafică a hașurării unei suprafețe precum și pentru analiza interpretării grafice a procedurilor numerice de integrare a funcțiilor (metoda dreptunghiului sau a sumelor Riemann, metoda trapezului, metoda lui Simpson).
- Arrows and Vector Fields pentru reprezentarea grafică a vectorilor și a câmpurilor de vectori.
- ODE Plots pentru reprezentarea grafică 2D și 3D a soluțiilor ecuațiilor diferențiale.
- Statistical Utility Plots pentru reprezentarea grafică a graficelor cu bare, a histogramelor, a graficelor cu sectoare circulare, a seturilor de puncte 2D (inclusiv a dreptei de regresie) și 3D.
- Conformal Plot pentru reprezentarea grafică a transformărilor conforme.

Parametrii caracteristici depind de tipul obiectului grafic. De exemplu, pentru obiectul grafic de tip `plot::Function2d`, principalii parametrii caracteristici sunt [20]:

- Numărul punctelor de discretizare a domeniului de definiție al funcției, `Mesh` (valoarea implicită 121).
- Tipul liniei, `LineStyle` (`solid`, `dashed`, `dotted`).
- Culoarea liniei, `LineColor` (implicit `RGB::Blue`). Culoarea liniei pentru a doua curbă, `LineColor2`.
- Grosimea liniei, `LineWidth` (valoarea implicită 0,35).
- Dimensiunea punctelor utilizate pentru reprezentarea grafică, `PointSize` (valoarea implicită 1,5).
- Titlul, `Title` și afișarea legendei, `LegendVisible`.
- Domeniul de valori ale funcției, `ViewingBoxYRange`.
- Identificarea punctelor singulare prin asimptote verticale, `VerticalAsymptotesVisible` (`TRUE` sau `FALSE`).

Accesul interactiv la totalitatea parametrilor caracteristici ai unui obiect grafic se realizează prin selectarea obiectului grafic respectiv. În acest mod, se deschide fereastra de configurare `Object Browser` în care se găsesc toți parametrii de configurare ai obiectului grafic grupați pe trei nivele subordonate ierarhic: grup de figuri, figură, axe și curbe.

- Reprezentarea grafică propriu-zisă a obiectului grafic. Pentru reprezentarea grafică a unui obiect grafic se folosesc instrucțiunile `2D graphics and animation`, respectiv `3D graphics and animation` după cum obiectul grafic de reprezentat este de tip 2D sau 3D.

### Problema 11.1

Se consideră funcția:

$$f: [-\pi; \pi] \rightarrow \mathbb{R}, f(x) = (x + 1) \cdot \sin x$$

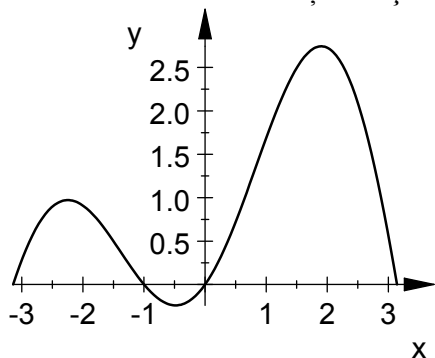
Să se reprezinte grafic funcția  $f(x)$  folosind procedurile de reprezentare grafică la nivel simbolic. Să se studieze influența următorilor parametri caracteristici asupra reprezentării grafice: culoarea liniei, tipul de linie, scalarea axelor, rețeaua de linii ajutătoare `grid`.

### Rezolvare

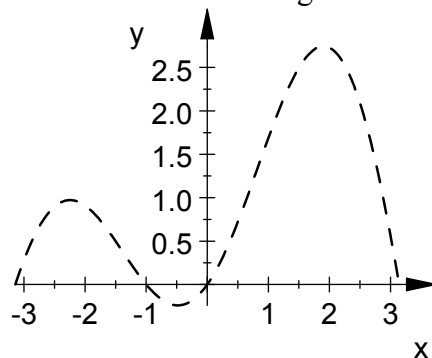
Pentru rezolvarea problemei, într-un fișier de tip MuPAD se scriu următoarele instrucțiuni:

```
f := (x) -> (x+1) * sin(x) :  
G1 := plot :: Function2d(f, x = -PI .. PI, Color = RGB :: Black,  
LineStyle = Solid) :  
G2 := plot :: Function2d(f, x = -PI .. PI, Color = RGB :: Black,  
LineStyle = Dashed) :  
G3 := plot :: Function2d(f, x = -PI .. PI, Color = RGB :: Black,  
Scaling = Constrained) :  
G4 := plot :: Function2d(f, x = -PI .. PI, Color = RGB :: Black,  
Scaling = Constrained, XGridVisible, YGridVisible) :  
plot(G1) ; plot(G2) ; plot(G3) ; plot(G4) ;
```

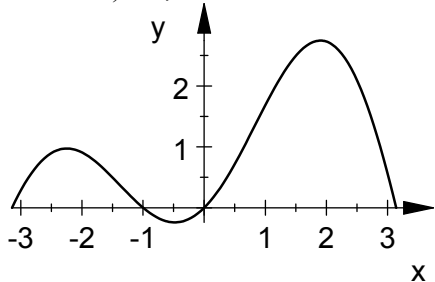
Lansarea în execuție a fișierului conduce la următoarele grafice:



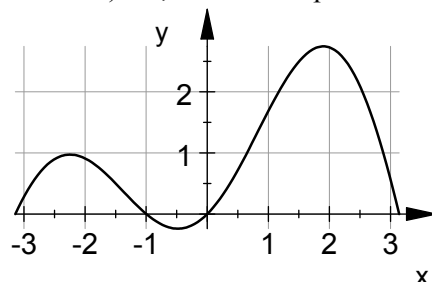
a) G1, linie continuă



b) G2, linie întreruptă



c) G3, scalare identică a axelor



d) G4, afișarea rețelei de linii `grid`

**Figura 11.21.** Reprezentări grafice, problema 11.1.

## Problema 11.2

Se consideră funcția dată prin legea:

$$f(x) = \frac{x^3}{x^2 - 1}$$

Se cere:

- Să se determine domeniul de definiție al funcției.
- Să se studieze asimptotele verticale ale funcției.
- Să se studieze asimptotele orizontale ale funcției.
- Să se studieze asimptotele oblice ale funcției.
- Să se studieze derivata întâi.
- Să se studieze derivata a doua.
- Să se completeze tabloul de variație al funcției.
- Să se reprezinte grafic funcția.

## Rezolvare

Pentru rezolvarea problemei, într-un fișier de tip MuPAD se scriu următoarele instrucțiuni:

```
f := (x) -> x^3 / (x^2 - 1) :
NUM := (x) -> denom(f(x)) :
s := solve(NUM, x) :
si := discont(f(x), x) :
limit(f(x), x=s[1], Left), limit(f(x), x=s[1], Right) :
limit(f(x), x=s[2], Left), limit(f(x), x=s[2], Right) :
limit(f(x), x=-infinity), limit(f(x), x=infinity) :
m := limit(f(x)/x, x=-infinity) :
m := limit(f(x)/x, x=infinity) :
n := limit(f(x)-m*x, x=-infinity) :
n := limit(f(x)-m*x, x=infinity) :
ya := (x) -> x :
D1 := diff(f(x), x) : D1 := simplify(D1) : xe := solve(D1, x) :
ye[1] := f(xe[1]) : ye[2] := f(xe[2]) : ye[3] := f(xe[3]) :
D2 := diff(f(x), x, x) : D2 := simplify(D2) : xi := solve(D2, x) :
yi[1] := f(xi[1]) : yi[2] := f(xi[2]) : yi[3] := f(xi[3]) :
G1 := plot::Function2d(f, x=-4..4, Color=RGB::Black,
LineStyle=Solid) :
G2 := plot::Point2d(xi[1], yi[1], PointSize = 2*unit::mm,
Color = RGB::Black, PointStyle = Squares) :
G3 := plot::Point2d(xe[2], ye[2], PointSize = 2*unit::mm,
Color = RGB::Black, PointStyle = Circles) :
G4 := plot::Point2d(xe[3], ye[3], PointSize = 2*unit::mm,
Color = RGB::Black, PointStyle = Circles) :
G5 := plot::Function2d(ya, x=-4..4, Color=RGB::Black,
LineStyle=Dashed) :
plot(G1, G2, G3, G4, G5) :
```



## Observații

Rezolvarea problemei presupune parcurgerea mai multor etape:

- Definirea funcției se face folosind operatorul de atribuire := combinat cu operatorul pentru definirea funcțiilor ->.
- Stabilirea domeniului de definiție al funcției. Funcția  $f(x)$  are sens dacă numitorul funcției este diferit de zero. Se extrage numitorul funcției folosind instrucțiunea `denom`, apoi se caută punctele în care numitorul este diferit de zero prin rezolvarea ecuației `denom(f(x))=0`. Soluțiile ecuației sunt  $\{-1; 1\}$ , prin urmare domeniul de definiție al funcției  $f(x)$  este  $\mathbb{R}-\{-1; 1\}$ . Determinarea directă a punctelor de discontinuitate, în care funcția nu are sens, se poate face și folosind instrucțiunea `discont`, [21].
- Asimptotele verticale se caută în punctele în care funcția  $f(x)$  nu este definită dar care sunt puncte de acumulare pentru domeniul de definiție al funcției, respectiv punctele  $\{-1; 1\}$ . Pentru cele două puncte se calculează limitele laterale folosind instrucțiunea `limit`. Se obțin următoarele rezultate:

$$\lim_{\substack{x \rightarrow -1 \\ x < -1}} f(x) = -\infty$$

$$\lim_{\substack{x \rightarrow -1 \\ x > -1}} f(x) = +\infty$$

$$\lim_{\substack{x \rightarrow +1 \\ x < +1}} f(x) = -\infty$$

$$\lim_{\substack{x \rightarrow +1 \\ x > +1}} f(x) = +\infty$$

Prin urmare funcția  $f(x)$  are două asimptote verticale în punctele:

$$x = -1$$

$$x = +1$$

- Asimptotele orizontale se determină calculând limitele funcției  $f(x)$  pentru  $x \rightarrow \pm\infty$ , dacă punctele  $\pm\infty$  aparțin domeniului de definiție al funcției. Se obțin următoarele rezultate:

$$\lim_{x \rightarrow -\infty} f(x) = -\infty$$

$$\lim_{x \rightarrow +\infty} f(x) = +\infty$$

Cum valorile obținute nu sunt diferite de  $\pm\infty$ , rezultă că funcția  $f(x)$  nu are asimptotă orizontală. Prin urmare, are sens căutarea asimptotelor oblice.

- Asimptotele oblice se caută sub forma ecuației  $y = mx + n$ , în care:

$$m = \lim_{x \rightarrow \pm\infty} \frac{f(x)}{x}$$
$$n = \lim_{x \rightarrow \pm\infty} [f(x) - mx]$$

Se obțin rezultatele  $m = 1$  și  $n = 0$ , deci funcția  $f(x)$  are asimptota oblică definită prin  $y = x$  spre  $\pm\infty$ .

- Se calculează derivata întâi a funcției folosind instrucțiunea `diff(f(x), x)`, obținându-se rezultatul:

$$\frac{d}{dx}f(x) = \frac{x^2(x^2 - 3)}{(x^2 - 1)^2}$$

Se calculează punctele critice ale derivatei întâi prin rezolvarea ecuației:

$$\frac{d}{dx}f(x) = 0$$

Se obțin valorile:  $\{0; -\sqrt{3}; +\sqrt{3}\}$ .

Se calculează valorile funcției în aceste puncte și se obțin rezultatele:

$$\left\{0; -\frac{3\sqrt{3}}{2}; +\frac{3\sqrt{3}}{2}\right\}.$$

- Se calculează derivata a doua a funcției folosind instrucțiunea `diff(f(x), x, x)`, obținându-se rezultatul:

$$\frac{d^2}{dx^2}f(x) = \frac{2x(x^2 + 3)}{(x^2 - 1)^3}$$

Se calculează punctele critice ale derivatei a doua prin rezolvarea ecuației:

$$\frac{d^2}{dx^2}f(x) = 0$$

Se obțin valorile:  $\{0; -i\sqrt{3}; +i\sqrt{3}\}$ .

Se calculează valorile funcției în aceste puncte și se obțin rezultatele:

$$\left\{0; -i\frac{3\sqrt{3}}{4}; +i\frac{3\sqrt{3}}{4}\right\}.$$

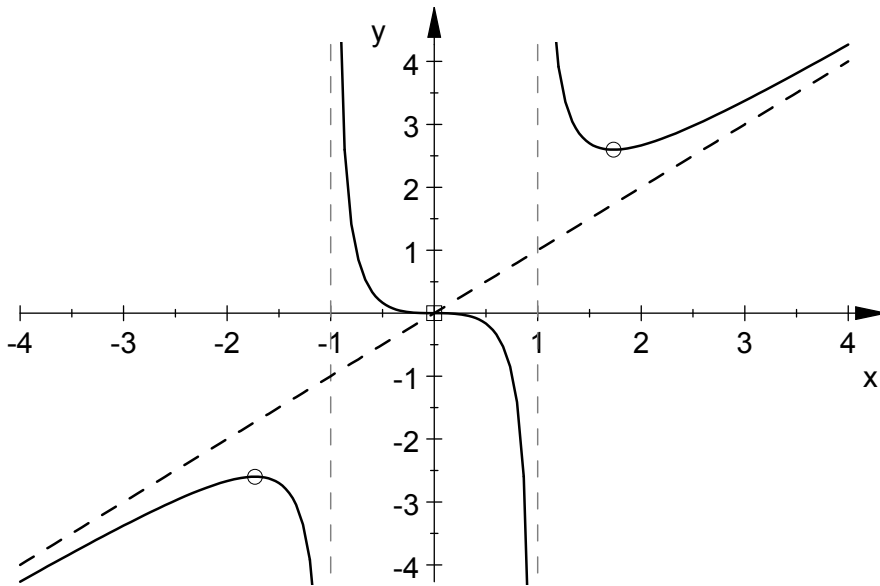
Prin urmare punctul de coordonate  $\{0; 0\}$  este un punct de inflexiune al funcției  $f(x)$ .

- Se completează tabloul de variație al funcției:

$x$	$-\infty$	$-\sqrt{3}$	$-1$	$0$	$+1$	$\sqrt{3}$	$+\infty$								
$f'(x)$	$+$	$0$	$-$	$-$	$0$	$-$	$0$	$+$							
$f''(x)$	$-$	$-$	$-$	$+$	$0$	$-$	$+$	$+$							
$f(x)$	$-\infty$	$\nearrow$	$-\frac{3\sqrt{3}}{2}$	$\searrow$	$-\infty$	$+\infty$	$\searrow$	$0$	$\searrow$	$-\infty$	$+\infty$	$\searrow$	$+\frac{3\sqrt{3}}{2}$	$\nearrow$	$+\infty$

Din analiza tabloului de variație al funcției se constată că funcția  $f(x)$  are două puncte de extrem: un punct de minim local având coordonatele  $\left\{\sqrt{3}; \frac{3\sqrt{3}}{2}\right\}$  și un punct de maxim local având coordonatele  $\left\{-\sqrt{3}; -\frac{3\sqrt{3}}{2}\right\}$ . De asemenea, funcția  $f(x)$  are un punct de inflexiune având coordonatele  $\{0; 0\}$ .

- Se reprezintă grafic funcția, figura 11.22.



**Figura 11.22.** Graficul funcției  $f(x)$ , problema 11.2.

### Problema 11.3

Se consideră funcția  $f: \mathbb{R} \rightarrow \mathbb{R}$  dată prin legea:

$$f(x) = \begin{cases} e^x, & x < -2 \\ \sin x, & x \in [-2; 2] \\ x^2 - 2x, & x > 2 \end{cases}$$

Să se reprezinte grafic funcția  $f(x)$  folosind diferite proceduri de reprezentare grafică la nivel simbolic.

### Rezolvare

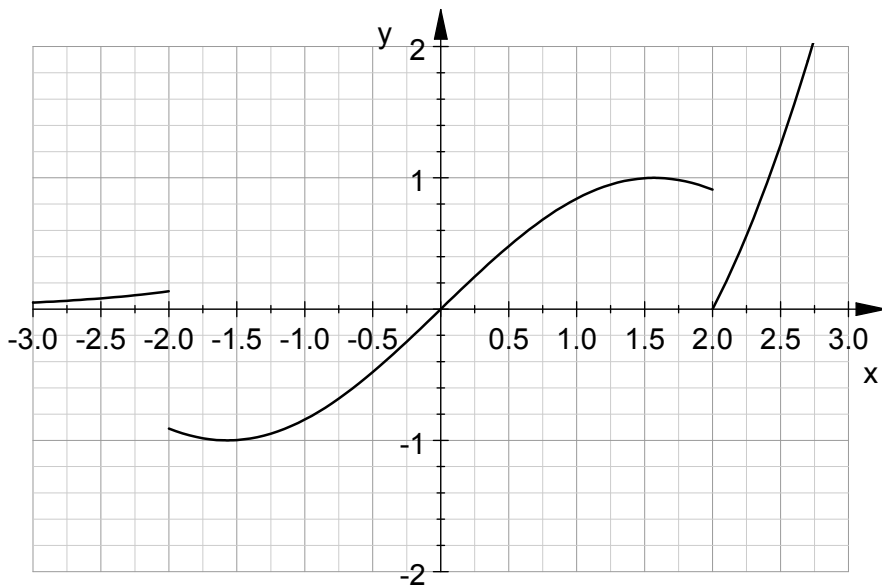
Pentru rezolvarea problemei, într-un fișier de tip MuPAD se scriu următoarele instrucțiuni:

```
f:=piecewise([x<-2,exp(x)],[x>=-2 and x<=2,sin(x)],[x>2,x^2-2*x]):
G1:=plot::Function2d(f(x),x=-3..3,Color=RGB::Black,
LineStyle=Solid,GridVisible=TRUE,SubgridVisible=TRUE,
XTicksDistance=0.5,XTicksBetween=1,YTicksDistance=1,
YTicksBetween=4,ViewingBoxYRange=-2..2):
plot(G1)
```

### Observații

- Funcția analizată este definită prin trei forme  $e^x$ ,  $\sin x$  și  $x^2 - 2x$ , pe trei intervale ale domeniului său de definiție  $x < -2$ ,  $x \in [-2; 2]$  și respectiv  $x > 2$ . Definirea acestor tipuri de funcții se realizează folosind instrucțiunea `piecewise`, [22].

- Instrucțiunea `plot::Function2d` definește o structură grafică, identificată prin numele `G`, care poate fi vizualizată ulterior folosind instrucțiunea `plot(G)`. Avantajul acestei abordări este faptul că dacă de exemplu se dorește reprezentarea graficelor a două funcții  $f_1(x)$  și  $f_2(x)$ , atunci se poate realiza configurarea independentă a parametrilor celor două curbe prin construirea a două obiecte grafice distincte `G1` și `G2` și reprezentarea grafică ulterioară cu o instrucțiune de forma `plot(G1, G2)`.
- În urma executării instrucțiunilor se obține reprezentarea grafică din figura 11.23.



**Figura 11.23.** Graficul funcției, problema 11.3.

- Vizualizarea liniilor principale ale rețelei de linii ajutătoare `grid` se face cu parametrul `GridVisible=TRUE`. Vizualizarea liniilor secundare ale rețelei de linii ajutătoare se face cu parametrul `SubgridVisible=TRUE`.
- Controlul distanței dintre liniile principale și a numărului de linii secundare se realizează cu parametrii `XTicksDistance=0.5`, `XTicksBetween=1` pentru axa `Ox` și respectiv `YTicksDistance=1`, `YTicksBetween=4` pentru axa `Oy`.
- Controlul domeniului de vizualizare pentru axa `Oy` se realizează cu ajutorul parametrului `ViewingBoxYRange=-2..2`.
- Reprezentarea grafică directă a funcției se poate face și folosind instrucțiunile simple `plotfunc2d`, [23] și `plot`, [24].

### Problema 11.4

Se consideră funcția  $f: [0; 3] \rightarrow \mathbb{R}$  dată prin legea:

$$f(x) = \frac{6x}{x^6 + x + 1}$$

Să se calculeze valoare numerică a integralei:

$$I_n = \int_0^3 f(x) dx$$

Să se studieze, folosind metoda grafică, modul de calcul al integralei folosind următoarele metode numerice: metoda sumelor Riemann (cu eșantionare inferioară, superioară, mediană), metoda trapezului, metoda Simpson.

### Rezolvare

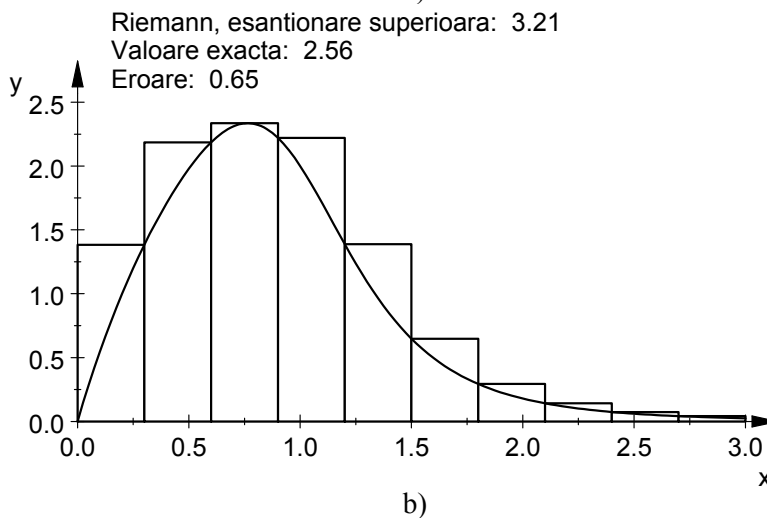
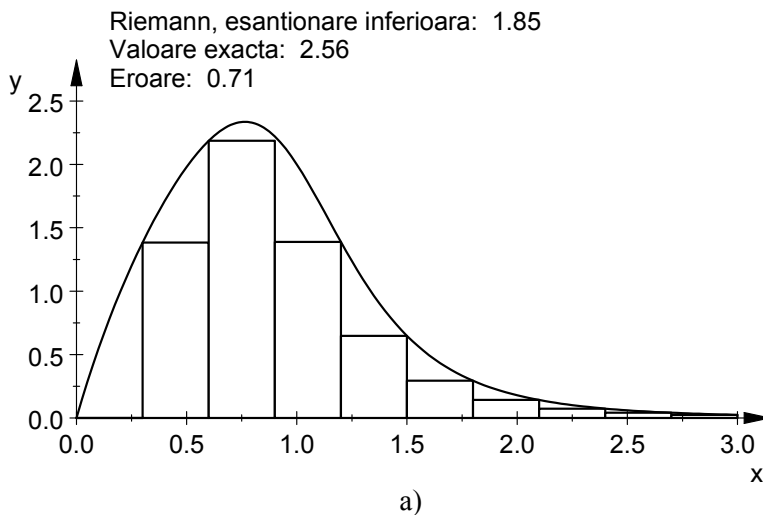
Pentru rezolvarea problemei, într-un fișier de tip MuPAD se scriu următoarele instrucțiuni:

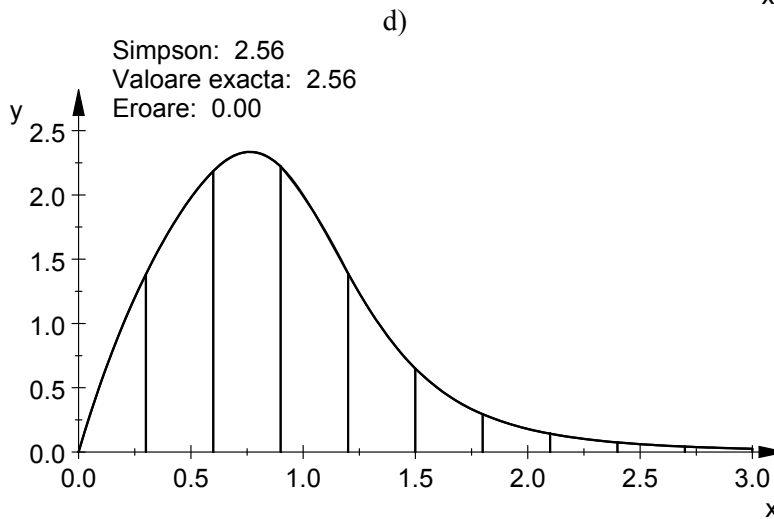
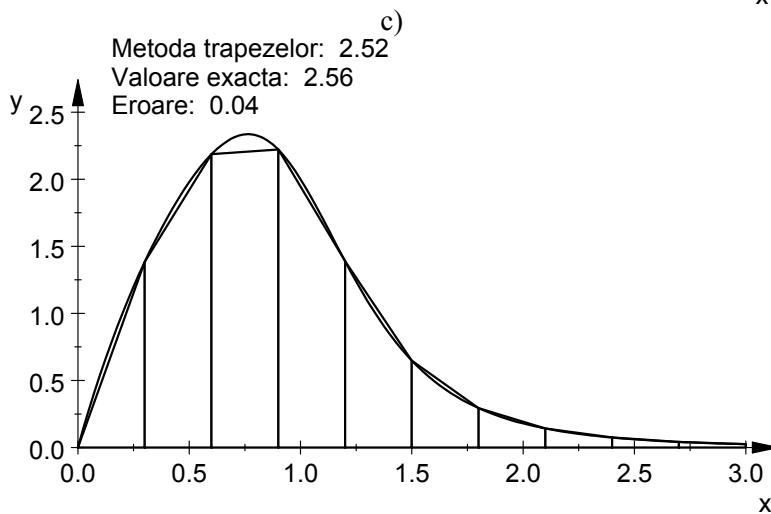
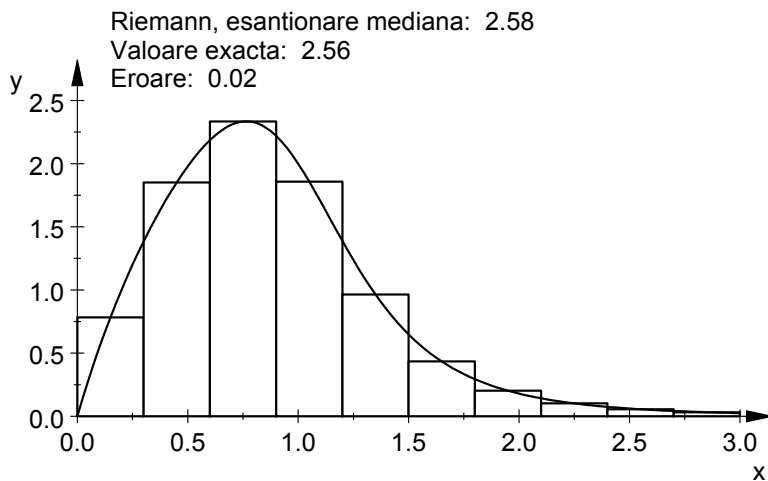
```
f:=x->6*x/(x^6+x+1):
In:=float(int(f(x),x=0..3)):
G1:=plot::Function2d(f,x=0..3,Color=RGB::Black):
G2:=plot::Integral(G1,10,IntMethod=RiemannLower,Color=
RGB::White,ShowInfo=[IntMethod,Integral="Valoare
exacta",Error="Eroare"]):
G3:=plot::Integral(G1,10,IntMethod=RiemannUpper,Color=
RGB::White,ShowInfo=[IntMethod,Integral="Valoare
exacta",Error="Eroare"]):
G4:=plot::Integral(G1,10,IntMethod=RiemannMiddle,Color=
RGB::White,ShowInfo=[IntMethod,Integral="Valoare
exacta",Error="Eroare"]):
G5:=plot::Integral(G1,10,IntMethod=Trapezoid,Color=RGB:
:White,ShowInfo=[IntMethod,Integral="Valoare
exacta",Error="Eroare"]):
G6:=plot::Integral(G1,10,IntMethod=Simpson,Color=RGB::
White,ShowInfo=[IntMethod,Integral="Valoare
exacta",Error="Eroare"]):
plot(G2,G1):
plot(G3,G1):
plot(G4,G1):
plot(G5,G1):plot(G6,G1):
```

### Observații

- După definirea funcției de analizat se calculează valoarea numerică a integralei, obținându-se rezultatul:  $I_n=2,56194351$ .
- Se definește obiectul grafic G1 conținând reprezentarea grafică a funcției de analizat.

- Se definesc apoi obiectele grafice G2, G3, G4, G5 și G6 conținând vizualizările grafice ale modului de calcul al integralei definite prin metoda sumelor Riemann cu eșantionare inferioară (G2), superioară (G3), mediană (G4), metoda trapezului (G5), respectiv prin metoda Simpson (G6) folosind instrucțiunea `plot :: Integral, [25]`.
- Se reprezintă, combinat, obiectele grafice (G2, G1) în figura 11.24, a); (G3, G1) în figura 11.24, b); (G4, G1) în figura 11.24, c); (G5, G1) în figura 11.24, d), (G2, G1) în figura 11.24, e), folosind în mod repetat instrucțiunea `plot`.
- Metoda cea mai precisă este metoda Simpson (eroare 0,0), urmată de metoda sumelor Riemann cu eșantionare mediană (eroare 0,02), de metoda trapezului (eroare 0,04), metoda sumelor Riemann cu eșantionare superioară (eroare 0,65) și inferioară (eroare 0,71).





e)

**Figura 11.24.** Interpretarea grafică a calcului integralei definite.

### Problema 11.5

Să se reprezinte grafic următoarele funcții 2D parametrice:

- a) Epicicloida, definită prin:

$$\begin{cases} x = (a + b) \cos t - b \cos \left[ \left( \frac{a}{b} + 1 \right) t \right] \\ y = (a + b) \sin t - b \sin \left[ \left( \frac{a}{b} + 1 \right) t \right] \end{cases}$$
$$a=8; b=5; t=[0; 10\pi]$$

- b) Epitrohoida, definită prin:

$$\begin{cases} x = (a + b) \cos t - c \cos \left[ \left( \frac{a}{b} + 1 \right) t \right] \\ y = (a + b) \sin t - c \sin \left[ \left( \frac{a}{b} + 1 \right) t \right] \end{cases}$$
$$a=5; b=3; c=5; t=[0; 6\pi]$$

- c) Hipotrohoida, definită prin:

$$\begin{cases} x = (a - b) \cos t + c \cos \left[ \left( \frac{a}{b} - 1 \right) t \right] \\ y = (a - b) \sin t - c \sin \left[ \left( \frac{a}{b} - 1 \right) t \right] \end{cases}$$
$$a=5; b=7; c=2,2; t=[-8\pi; 8\pi]$$

- d) Astroida, definită prin:

$$\begin{cases} x = a(\cos t)^3 \\ y = a(\sin t)^3 \end{cases}$$
$$a=1; t=[-\pi; \pi]$$

- e) Hipocicloida, definită prin:

$$\begin{cases} x = (a - b) \cos t + b \cos \left[ \left( \frac{a}{b} - 1 \right) t \right] \\ y = (a - b) \sin t - b \sin \left[ \left( \frac{a}{b} - 1 \right) t \right] \end{cases}$$
$$a=5; b=3; t=[-3\pi; 3\pi]$$

- f) Tricuspoida, definită prin:

$$\begin{cases} x = a(2 \cos t + \cos 2t) \\ y = a(2 \sin t - \sin 2t) \end{cases}$$
$$a=0,1; t=[-\pi; \pi]$$

- g) Curba Lissajous, definită prin:

$$\begin{cases} x = a \sin(nt + c) \\ y = b \sin t \end{cases}$$

$$a=1, b=1, c=1, n=3, t=[-\pi; \pi]$$

- h) Foliumul lui Descartes, definit prin:

$$\begin{cases} x = 3at/(1 + t^3) \\ y = 3at^2/(1 + t^3) \end{cases}$$
$$a=1; t=[-20\pi; 20\pi]$$



i) Curba Tallbot, definită prin:

$$\begin{cases} x = [a^2 + f^2(\sin t)^2] \frac{\cos t}{a} \\ y = [a^2 - 2f^2 + f^2(\sin t)^2] \frac{\sin t}{b} \end{cases}$$

$a=1,1; b=1; f=1; t=[-\pi; \pi]$

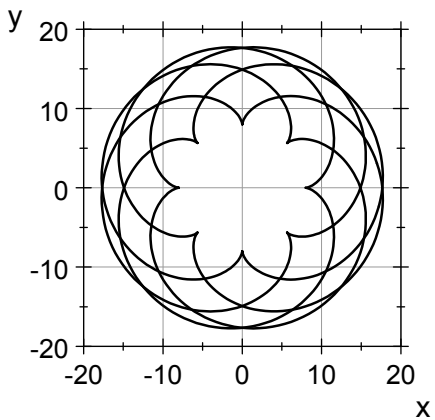
### Rezolvare

Pentru reprezentarea grafică a epicloidei, într-un fișier de tip MuPAD se scriu următoarele instrucțiuni:

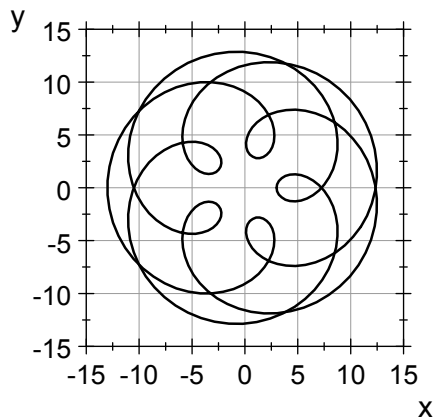
```
reset():
x:=(a+b)*cos(t)-b*cos((a/b+1)*t):
y:=(a+b)*sin(t)-b*sin((a/b+1)*t):
a:=8:b:=5:
G:=plot::Curve2d([x,y],t=0..10*PI,Color=RGB::Black,
Scaling=Constrained,Mesh=1000,GridVisible=TRUE,
XTicksDistance=10,XTicksBetween=1,YTicksDistance=10,
YTicksBetween=1,Axes=Boxed,ViewingBoxYRange=-20..20,
ViewingBoxXRange=-20..20):
plot(G)
```

### Observații

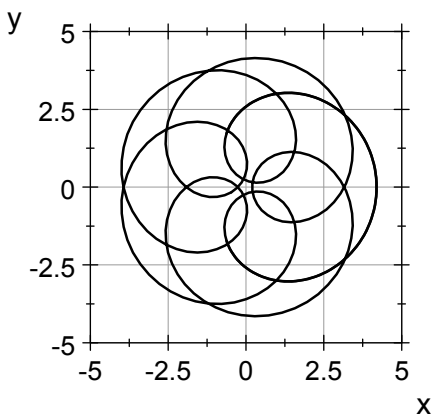
- Pentru reprezentarea grafică a funcțiilor definite sub formă parametrică se utilizează instrucțiunea `plot::Curve2d`, [26].
- Instrucțiunea pentru realizarea obiectului grafic `G` permite modificarea parametrilor specifici care controlează: culoarea curbei (`Color=RGB::Black`), scalarea celor două axe (`Scaling=Constrained`), numărul de puncte de discretizare (`Mesh=1000`), afișarea rețelei de linii ajutătoare (`GridVisible=TRUE`), distanța dintre liniile principale (`XTicksDistance=10`, `YticksDistance=10`) și a numărului de linii secundare (`XticksBetween=1`, `YTicksBetween=1`) pentru cele două axe  $Ox$  și  $Oy$ , tipul axelor de coordonate (`Axes=Boxed`), domeniul de vizualizare pentru cele două axe (`ViewingBoxYRange=-20..20`, `ViewingBox XRange=-20..20`).
- Instrucțiunile pentru reprezentarea grafică a epicloidei se adaptează și pentru celelalte funcții, obținându-se următoarele grafice: epicloida în figura 11.25, a); epitrohoida în figura 11.25, b); hipotrohoida în figura 11.25, c); astroida în figura 11.25, d); hipocicloida în figura 11.25, e); tricuspoida în figura 11.25, f); curba Lissajous în figura 11.25, g); foliumul lui Descartes în figura 11.25, h); curba Tallbot în figura 11.25, i).



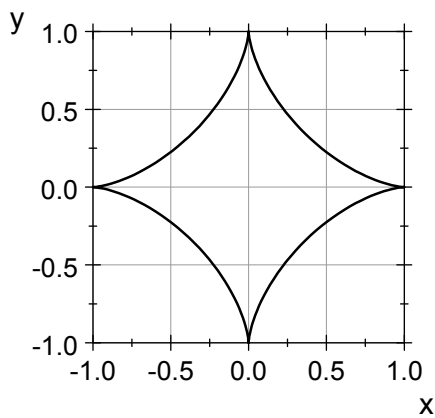
a) epicicloida



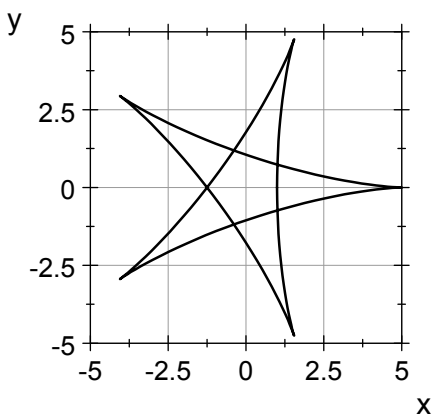
b) epitrochoida



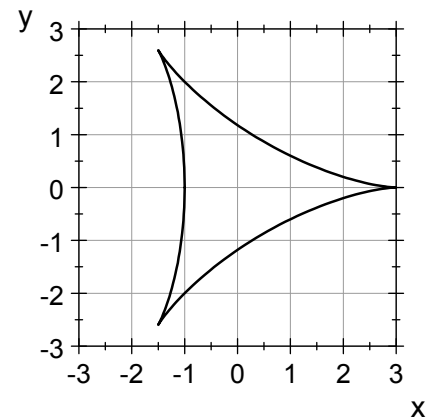
c) hipotrochoida



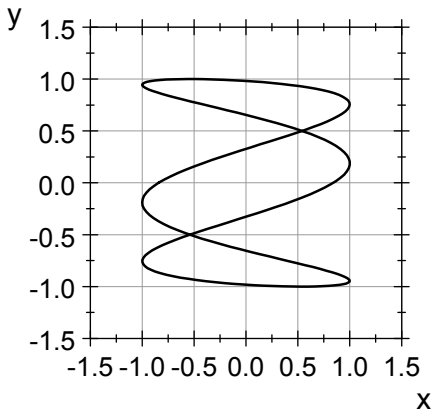
d) astroida



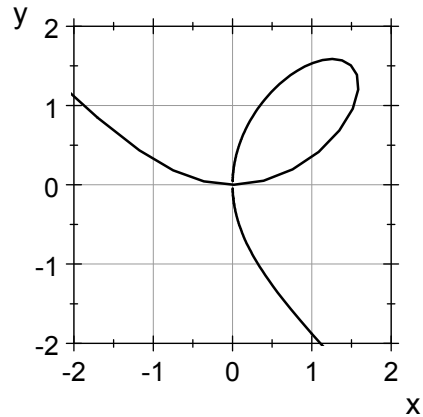
e) hipocicloida



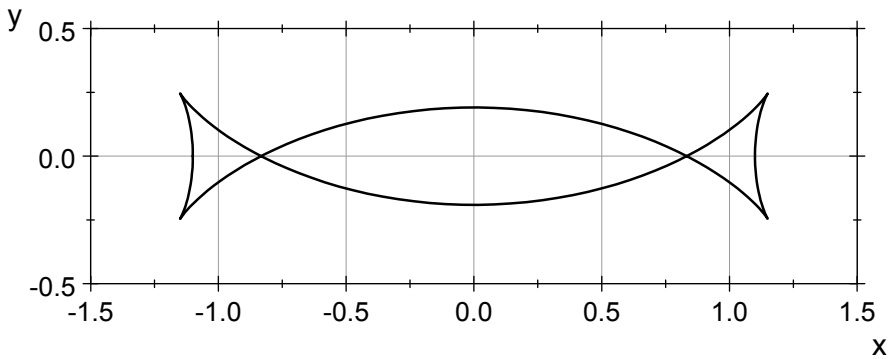
f) tricuspoida



g) curba Lissajous



h) foliumul lui Descartes



i) curba Tallbot

**Figura 11.25.** Reprezentarea grafică a funcțiilor 2D parametrice.

### Problema 11.6

Se consideră funcția, denumită cardioidă, definită sub formă implicită prin legea:

$$(x^2 + y^2 - 2ax)^2 = 4a^2(x^2 + y^2), \quad a = 1$$

Să se reprezinte grafic funcția pe domeniul de definiție specificat prin  $x \times y = [-2\pi; 2\pi] \times [-\pi; \pi]$ .

### Rezolvare

Pentru reprezentarea grafică a cardioidei, într-un fișier de tip MuPAD se scriu următoarele instrucțiuni:

```
reset():
eq:=(x^2+y^2-2*a*x)^2-4*a^2*(x^2+y^2):
a:=1:
G:=plot::Implicit2d(eq,x=-2*PI..2*PI,y=-PI..PI,
Color=RGB::Black,Scaling=Constrained,Mesh=[20,20],
```

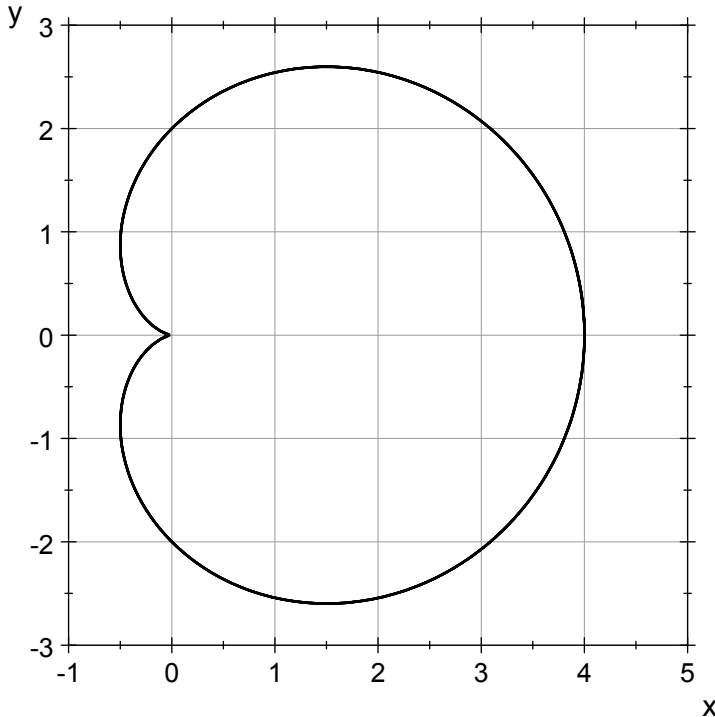
```

GridVisible=TRUE,XTicksDistance=1,XTicksBetween=1,
YTicksDistance=1,YTicksBetween=1,Axes=Boxed,
ViewingBoxYRange=-3..3,ViewingBoxXRange=-1..5):
plot(G)

```

### Observații

- Pentru reprezentarea grafică a funcțiilor definite sub formă implicită se utilizează instrucțiunea `plot::Implicit2d`, [27].
- Primul parametru al instrucțiunii pentru crearea obiectului grafic este chiar ecuația implicită a curbei de analizat, definită în prealabil prin  $eq := (x^2 + y^2 - 2ax)^2 - 4a^2(x^2 + y^2)$ .
- Specificarea domeniului de definiție pentru cele două axe de coordonate  $Ox$  și  $Oy$  se realizează cu instrucțiunile  $x = -2\pi..2\pi$  și  $y = -\pi..\pi$ .
- Odată cu crearea obiectului grafic se modifică și parametrii caracteristici ai acestuia. Se remarcă parametrul `Mesh=[20,20]` care definește o discretizare identică a celor două axe de coordonate  $Ox$  și  $Oy$ , fiecare cu câte 20 de puncte.
- După executarea instrucțiunilor se obține reprezentarea grafică din figura 11.26.



**Figura 11.26.** Reprezentarea grafică a cardioidei.

### Problema 11.7

Să se reprezinte grafic următoarele funcții în coordonate polare:

- a) Spirala echiunghiulară, definită prin:

$$r = ae^{\theta \cot b}$$

$$a=1; b=1,5; \theta=[0; 2\pi]$$

- b) Spirala lui Fermat, definită prin:

$$r^2 = a^2 \theta$$

$$a=1; \theta=[0; 2\pi]$$

- c) Spirala sinusoidală, definită prin:

$$r = a(\cos p\theta)^{1/p}$$

$$a=1; p=1/20, \theta=[0; 10\pi]$$

- d) Curba Kappa, definită prin:

$$r = a \operatorname{ctg} \theta$$

$$a=1; \theta=[0; 2\pi]$$

- e) Foliumul dublu, definit prin:

$$r = 4a \cos \theta (\sin \theta)^2$$

$$a=1; \theta=[0; \pi]$$

- f) Foliumul triplu, definit prin:

$$r = a \cos \theta [4(\sin \theta)^2 - 1]$$

$$a=1; \theta=[0; \pi]$$

- g) Lemniscata lui Bernoulli, definită prin:

$$r^2 = a^2 \cos 2\theta$$

$$a=1, \theta=[-\pi/4; \pi/4]$$

### Rezolvare

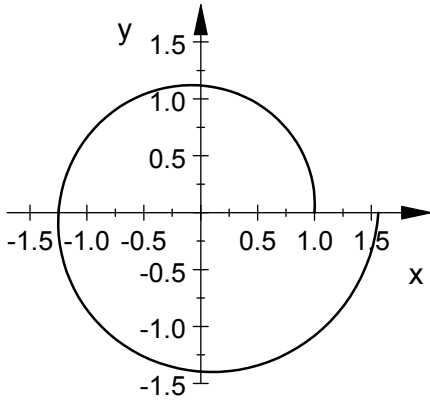
Pentru reprezentarea grafică a spiralei echiunghiulare, într-un fișier de tip MuPAD se scriu următoarele instrucțiuni:

```
reset() :  
r1:=a*exp(t *cot(b)) :a:=1:b:=1.5 :  
G1:=plot::Polar([r1,t],t=0..2*PI,Color=RGB::Black,  
Scaling=Constrained,Mesh=100,  
XTicksDistance=0.5,XTicksBetween=1,  
YTicksDistance=0.5,YTicksBetween=1,  
ViewingBoxYRange=-1.5..1.5,ViewingBoxXRange=-1.7..1.7) :  
plot(G1)
```

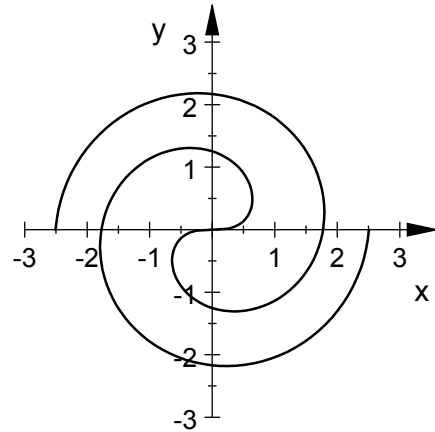
### Observații

- Pentru reprezentarea grafică a funcțiilor definite în coordonate polare se utilizează instrucțiunea `plot::Polar`, [28].
- Instrucțiunile pentru reprezentarea grafică a spiralei echiunghiulare se adaptează și pentru celelalte funcții, obținându-se următoarele grafice: spirala echiunghiulară în figura 11.27, a); spirala lui Fermat

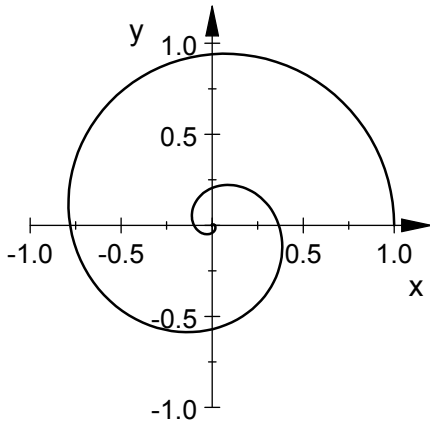
în figura 11.27, b); spirala sinusoidală în figura 11.27, c); curba Kappa în figura 11.27,d); foliumul dublu în figura 11.27,e); foliumul triplu în figura 11.27,f); lemniscata lui Bernoulli în figura 11.27,g).



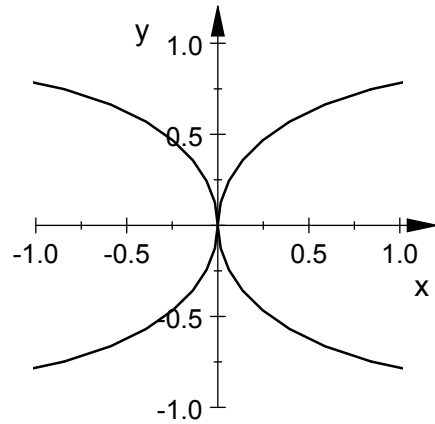
a) spirală echiunghiulară



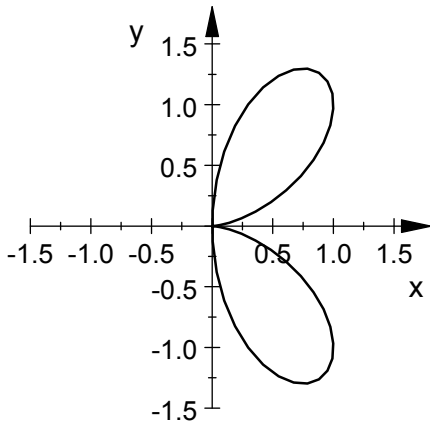
b) spirală lui Fermat



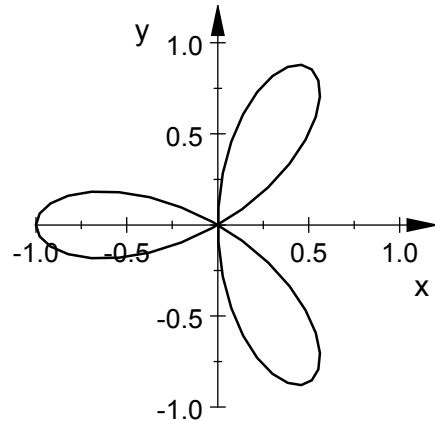
c) spirală sinusoidală



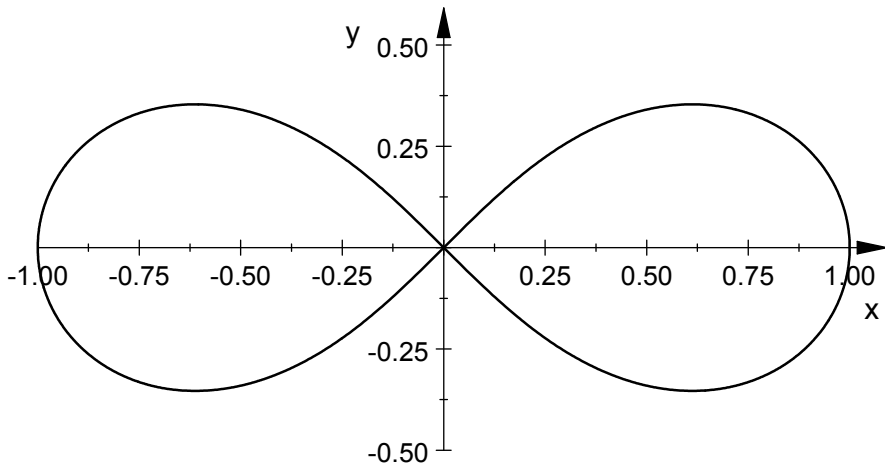
d) curba Kappa



e) foliumul dublu



f) foliumul triplu



g) lemniscata lui Bernoulli

**Figura 11.27.** Reprezentarea grafică a funcțiilor în coordonate polare.

### Problema 11.8

Să se reprezinte grafic următoarele funcții 3D parametrice, [57]:

a) Banda lui Moebius, definită prin:

$$\begin{cases} x = [R + u \cos(v/2)] \cos v \\ y = [R + u \cos(v/2)] \sin v \\ z = u \sin(v/2) \end{cases}$$

$$u \times v = [-w; w] \times [0; 2\pi]; R=1; w=0,5$$

b) Suprafața de tip Sine, definită prin:

$$\begin{cases} x = a \sin u \\ y = a \sin v \\ z = a \sin(u + v) \end{cases}$$

$$u \times v = [-2\pi; 2\pi] \times [-2\pi; 2\pi]; a=1$$

c) Suprafața de tip octaedru hiperbolic, definită prin:

$$\begin{cases} x = (\cos u \cos v)^3 \\ y = (\sin u \cos v)^3 \\ z = (\sin v)^3 \end{cases}$$

$$u \times v = [-\pi/2; \pi/2] \times [-\pi; \pi]$$

d) Suprafața de tip pseudosferă, definită prin:

$$\begin{cases} x = \cos u \sin v \\ y = \sin u \sin v \\ z = \cos v + \ln[\tan(v/2)] \end{cases}$$

$$u \times v = [0; 2\pi] \times [0; \pi]$$

e) Suprafața de tip Corkscrew, definită prin:

$$\begin{cases} x = a \cos u \cos v \\ y = a \sin u \cos v \\ z = a \sin v + bu \end{cases}$$

$$u \times v = [0; \pi] \times [0; 2\pi]; a=1; b=1$$

f) Suprafața de tip Cross Cap, definită prin:

$$\begin{cases} x = 1/2 \cos u \sin 2v \\ y = 1/2 \sin u \sin 2v \\ z = 1/2 [(\cos v)^2 - (\cos u)^2 (\sin v)^2] \end{cases}$$

$$u \times v = [0; 2\pi] \times [0; \pi/2]$$

g) Suprafața de tip Seashell, definită prin:

$$\begin{cases} x = 2[1 - e^{u/6\pi}] \cos u [\cos(v/2)]^2 \\ y = 2[-1 + e^{u/6\pi}] \sin u [\cos(v/2)]^2 \\ z = 1 - e^{u/3\pi} - \sin v + e^{u/6\pi} \sin v \end{cases}$$

$$u \times v = [0; 5\pi] \times [0; 2\pi]$$

### Rezolvare

Pentru reprezentarea grafică a suprafeței de tip octaedru hiperbolic, într-un fișier de tip MuPAD se scriu următoarele instrucțiuni:

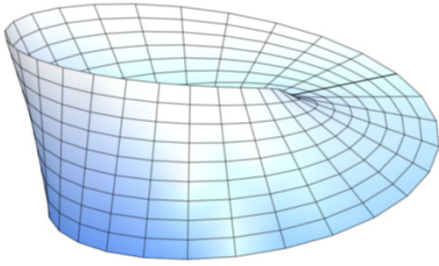
```
reset():
x:=(cos(u)*cos(v))^3:
y:=(sin(u)*cos(v))^3:
z:=(sin(v))^3:
G:=plot::Surface([x,y,z],u=-PI/2..PI/2,v=-PI..PI,
Color RGB::White,Scaling=Constrained,UMesh=40,VMesh=40,
ULinesVisible=TRUE,VLinesVisible=TRUE):
plot(G3)
```

### Observații

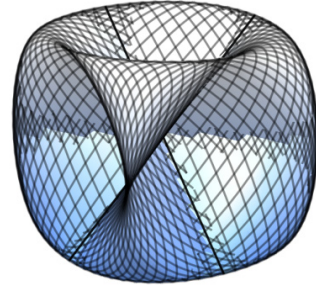
- Pentru reprezentarea grafică a funcțiilor 3D parametrice se utilizează instrucțiunea `plot::Surface`, [29].
- Se definesc expresiile  $x(u, v)$ ,  $y(u, v)$  și  $z(u, v)$  prin care este specificată suprafața de analizat.
- Specificarea domeniului de definiție pentru cei doi parametri  $u$  și  $v$  se face în corpul instrucțiunii `plot::Surface` prin instrucțiunile  $u=-PI/2..PI/2$  și  $v=-PI..PI$ .
- Odată cu crearea obiectului grafic  $G$  se modifică și parametrii caracteristici ai acestuia. Se remarcă parametrii  $UMesh=40$  și  $VMesh=40$  care definesc numărul punctelor de discretizare pentru domeniile de variație ale celor doi parametri  $u$  și  $v$ . De asemenea, parametrii  $ULinesVisible=TRUE$  și  $VLinesVisible=TRUE$  controlează afișarea sau nu a liniilor rețelei de tip `wireframe` prin care se construiește suprafața de analizat.



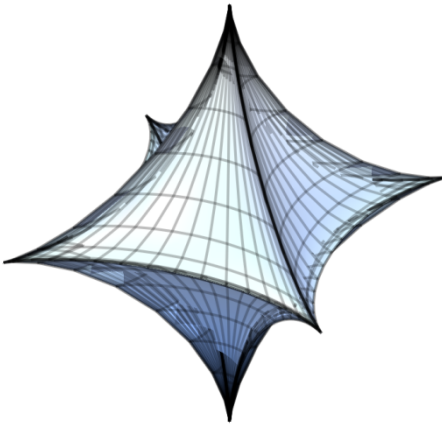
- Instrucțiunile pentru reprezentarea grafică a suprafeții de tip octaedru hiperbolic se adaptează și pentru celelalte funcții, obținându-se următoarele grafice: banda lui Moebius în figura 11.28,a); suprafața de tip Sine în figura 11.28, b); suprafața de tip octaedru hiperbolic în figura 11.28, c); suprafața de tip pseudosferă în figura 11.28, d); suprafața de tip Corkscrew în figura 11.28, e); suprafața de tip Cross Cap în figura 11.28, f); suprafața de tip Seashell în figura 11.28, g).



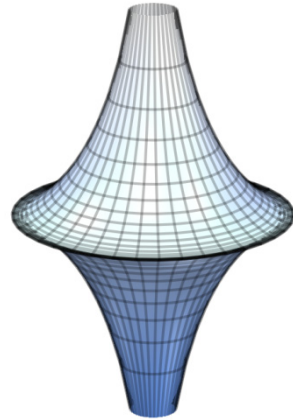
a) banda lui Moebius



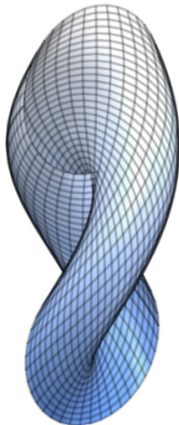
b) suprafață de tip Sine



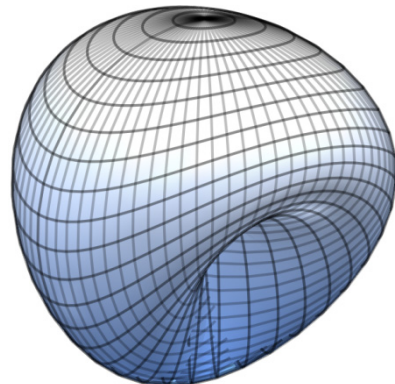
c) octaedru hiperbolic



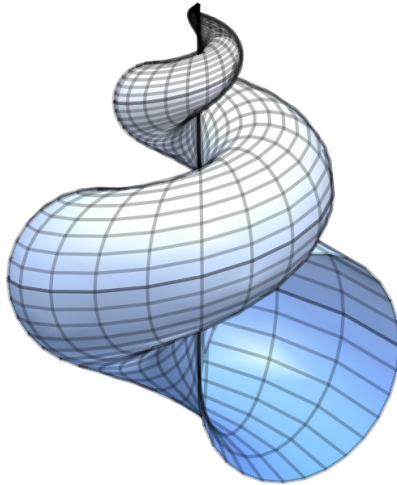
d) pseudosferă



e) suprafață Corkscrew



f) suprafață Cross Cap



g) suprafața de tip Seashell

**Figura 11.28.** Reprezentarea grafică a funcțiilor 3D parametrice.

### Problema 11.9

Să se reprezinte grafic următoarele primitive grafice 3D:

- Hexaedrul, tetraedrul, octaedrul, icosaedrul, dodecaedrul având centrul în originea sistemului de coordonate și raza sferei circumscrise egală cu 1 (pentru hexaedru, raza sferei înscrise=1).
- Prisma regulată, piramida și trunchiul de piramidă.
- Conul, cilindrul, sfera, elipsoidul.

### Rezolvare

Pentru reprezentarea grafică a hexaedrului, într-un fișier de tip MuPAD se scriu următoarele instrucțiuni:

```
G1:=plot::Hexahedron(Center=[0,0,0],Radius=1,
Color=RGB::White,XAxisVisible=FALSE,YAxisVisible=FALSE,
ZAxisVisible=FALSE,Scaling=Constrained):
plot(G1)
```

Instrucțiunile pentru controlul parametrilor caracteristici (culoarea, vizibilitatea axelor de coordonate, scalarea) se vor utiliza și pentru reprezentarea celorlalte primitive grafice 3D.

Pentru reprezentarea grafică a prisme regulate (cu 8 fețe laterale), a piramidei și a trunchiului de piramidă, într-un fișier de tip MuPAD se scriu următoarele instrucțiuni:

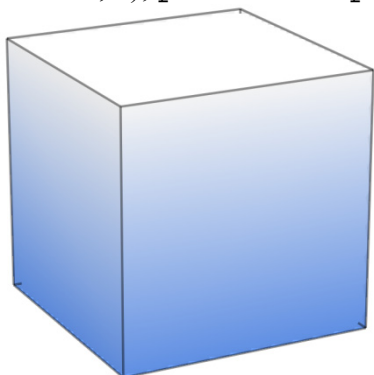
```
G2:=plot::Prism(1,[0,0,0],[0,0,1],Edges=8):
G3:=plot::Pyramid(1,[0,0,0],0,[0,0,1]):
G4:=plot::Pyramid(1,[0,0,0],.5,[0,0,.5]):
plot(G2);plot(G3);plot(G4);
```

Pentru reprezentarea grafică a conului, cilindrului, sferei și elipsoidului, într-un fișier de tip MuPAD se scriu următoarele instrucțiuni:

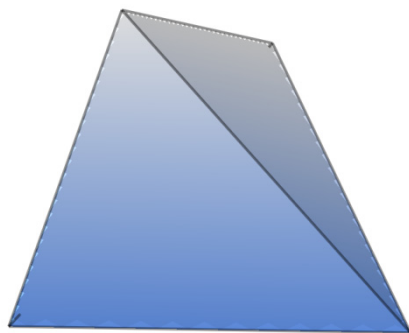
```
G5:=plot::Cone(1,[0,0,0],[0,0,1]):  
G6:=plot::Cylinder(1,[0,0,0],[0,0,1]):  
G7:=plot::Sphere(1,[0,0,0]):  
G8:=plot::Ellipsoid(1,.5,.5,[0,0,0]):  
plot(G5);plot(G6);plot(G7);plot(G8);
```

### Observații

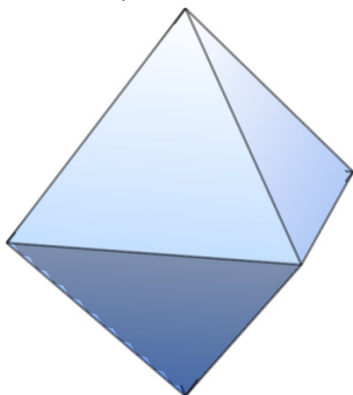
- Pentru reprezentarea grafică se utilizează instrucțiunile `plot::Hexahedron`, [30], figura 11.29, a); `plot::Tetrahedron`, [31], figura 11.29, b); `plot::Octahedron`, [32], figura 11.29, c) `plot::Icosahedron`, [33], figura 11.29, d); `plot::Dodecahedron`, [34], figura 11.29, e); `plot::Prism`, [35], figura 11.29, f), `plot::Pyramid`, [36], figura 11.29, g) și h); `plot::Cone`, [37], figura 11.29, i); `plot::Cylinder`, [38], figura 11.29, j); `plot::Sphere`, [39], figura 11.29, k); `plot::Ellipsoid`, [40], figura 11.29, l).



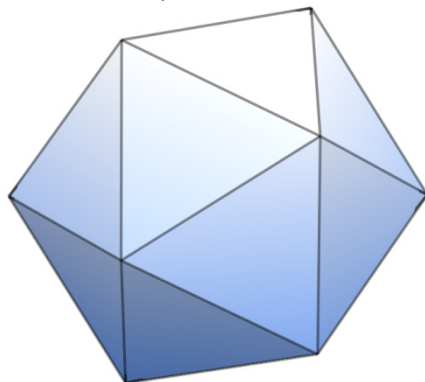
a) hexaedrul



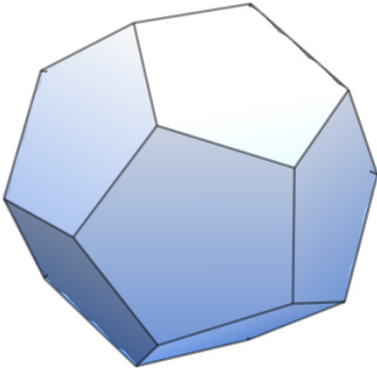
b) tetraedrul



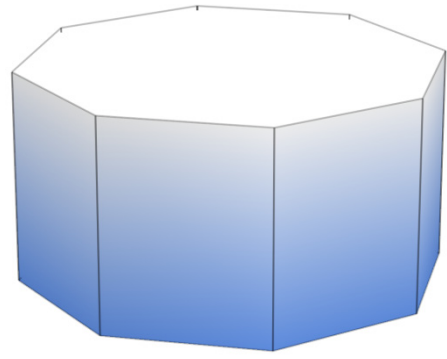
c) octaedrul



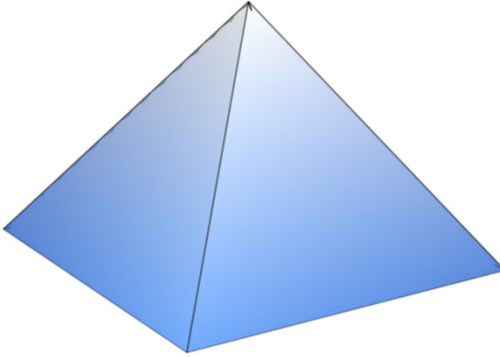
d) icosaedrul



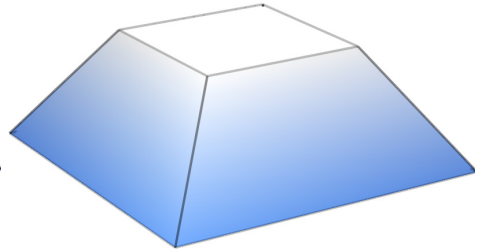
e) dodecaedrul



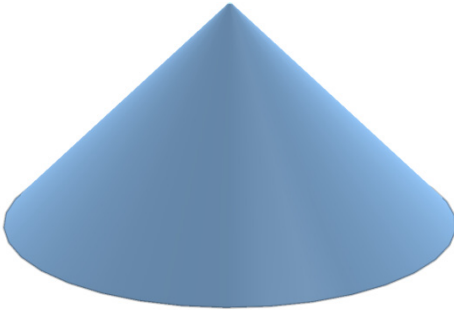
f) prisma regulată



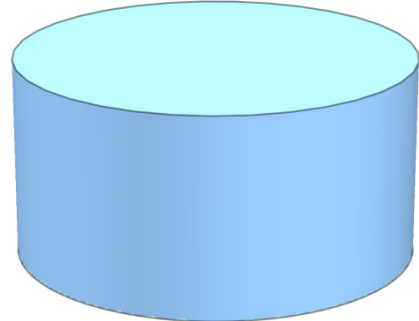
g) piramida



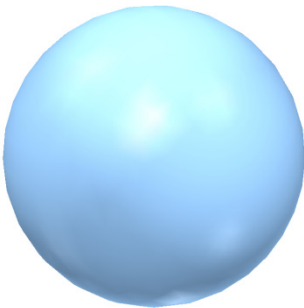
h) trunchiul de piramidă



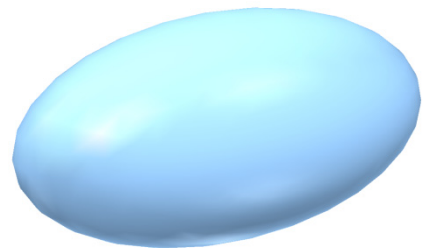
i) conul



j) cilindrul



k) sfera



l) elipsoidul

**Figura 11.29.** Reprezentarea grafică a unor primitive grafice 3D.

## 11.6. REZOLVAREA ECUAȚILOR ȘI SISTEMELOR DE ECUAȚII ALGEBRICE ȘI TRANSCENDENTE

Principalele etape necesare pentru rezolvarea ecuațiilor și sistemelor de ecuații algebrice sunt:

- Definirea ecuației sau sistemului de ecuații de rezolvat. În funcție de tipul problemei de rezolvat se utilizează diferite metode: metoda expresiilor matematice sau metoda funcțiilor pentru definirea ecuațiilor, respectiv metoda mulțimii de expresii sau metoda matriceală pentru definirea sistemelor de ecuații.
- Rezolvarea propriu-zisă. Se realizează folosind metode simbolice (instrucțiunea generală `solve`, [41]) sau numerice (instrucțiunea generală `numeric::solve`, [42]). Principalele formate de utilizare ale celor două instrucțiuni de rezolvare sunt:

```
solve (eq, x, opt)
solve (eq, x=a..b, opt)
numeric::solve (eq, x, opt)
numeric::solve (eq, x=a..b, opt)
```

în care `eq` reprezintă ecuația sau sistemul de ecuații de rezolvat; `x` reprezintă variabila sau variabilele în raport cu care se dorește rezolvarea; `x=a..b` reprezintă intervalul închis al variabilei în care se dorește căutarea soluțiilor; `opt` reprezintă o listă opțională de parametri specifici procesului de căutare a soluțiilor.

Instrucțiunile generale `solve` și `numeric::solve` se utilizează pentru orice tip de problemă de rezolvat. În cazul anumitor probleme particulare, de exemplu pentru rezolvarea sistemelor de ecuații algebrice liniare, instrucțiunea generală `solve` va apela, intern, instrucțiunea specifică `linsolve` pentru metoda simbolică de rezolvare, respectiv instrucțiunea `numeric::linsolve` pentru metoda numerică de rezolvare. Obținerea unui rezultat numeric se poate face prin două metode: rezolvarea problemei folosind metoda simbolică conduce la obținerea soluțiilor simbolice, după care se procedează la aproximarea numerică a soluțiilor simbolice (comanda `float`); rezolvarea problemei folosind metoda numerică care conduce direct la obținerea soluției numerice. Dezavantajul utilizării metodei numerice este că în cazul ecuațiilor diferite de cele polinomiale se obține doar prima soluție a ecuației. Obținerea și a celorlalte soluții reale (dacă există) se face prin utilizarea parametrului opțional `AllRealRoots`. Verificarea existenței mai multor soluții reale se poate face și prin metoda reprezentării grafice.

### Problema 11.10

Se consideră funcția:

$$f: \mathbb{R} \rightarrow \mathbb{R}$$

definită prin legea:

$$f(x) = x^3 - 2x - 1$$

Se cere:

- Să se rezolve ecuația  $f(x)=0$  folosind instrucțiunile `solve` și `numeric::solve`. Să se utilizeze pentru definirea ecuației de rezolvat atât metoda expresiilor matematice, cât și metoda funcțiilor.
- Să se reprezinte grafic funcția  $f(x)$  și soluțiile ecuației  $f(x)=0$ .

### Rezolvare

Rezolvarea problemei prin metoda simbolică și definirea ecuației printr-o expresie este prezentată în figura 11.30, a).

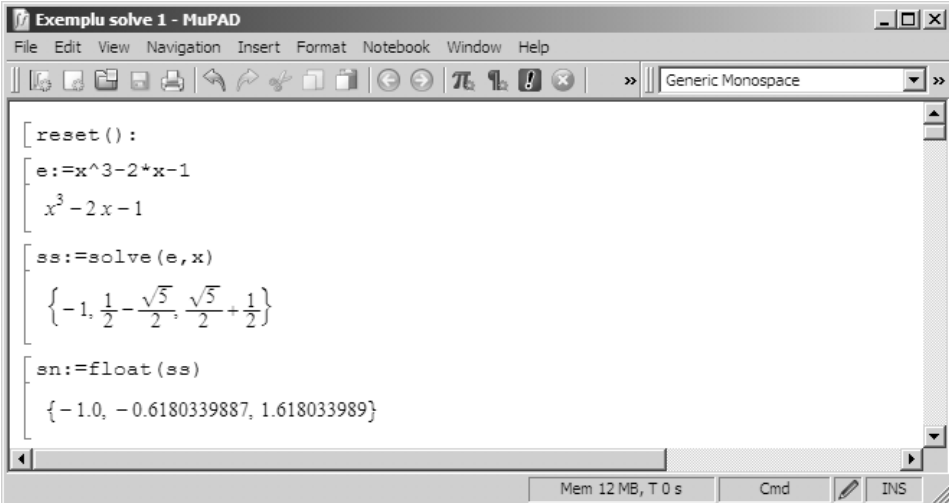
Rezolvarea problemei prin metoda numerică și definirea ecuației printr-o funcție este prezentată în figura 11.30, b).

Reprezentarea grafică a funcției  $f(x)$ , precum și a celor trei soluții se realizează cu ajutorul instrucțiunilor prezentate în figura 11.30, c).

Graficul obținut este prezentat în figura 11.30, d).

### Observații

- Definirea ecuației de analizat ca expresie matematică se realizează prin instrucțiunea `e:=x^3-2*x-1`.
- Rezolvarea simbolică a ecuației se realizează prin instrucțiunea `ss:=solve(e,x)`.
- Obținerea aproximărilor numerice ale soluțiilor simbolice se realizează prin instrucțiunea `sn:=float(ss)`.



```
Exemplu solve 1 - MuPAD
File Edit View Navigation Insert Format Notebook Window Help
Generic Monospace
[ reset():
[ e:=x^3-2*x-1
  x^3-2x-1
[ ss:=solve(e,x)
  { -1, 1/2 - sqrt(5)/2, sqrt(5)/2 + 1/2 }
[ sn:=float(ss)
  { -1.0, -0.6180339887, 1.618033989 }
```

a) definirea ecuației ca expresie și metoda simbolică de rezolvare

```

[reset():]
[f:=x->x^3-2*x-1
 x -> x^3 - 2x - 1
 sn:=numeric::solve(f(x),x)
 {-1.0, -0.6180339887, 1.618033989}
 x0:=op(sn)
 1.618033989, -0.6180339887, -1.0
 f(x0[1]);f(x0[2]);f(x0[3])
 0.0
 0.0
 0.0

```

b) definirea ecuației ca funcție și metoda numerică de rezolvare

- Definirea ecuației de analizat ca funcție matematică se realizează prin instrucțiunea  $f := (x) \rightarrow x^3 - 2x - 1$ .
- Rezolvarea numerică a ecuației se realizează prin instrucțiunea  $sn := numeric::solve(f(x), x)$ . Se obține mulțimea soluțiilor numerice ale ecuației formată din trei elemente.
- Specificarea independentă a fiecărei soluții din mulțimea soluțiilor ecuației se realizează prin instrucțiunea  $x0 := op(sn)$ , [43].
- Verificarea fiecărei soluții a ecuației  $f(x) = 0$  se realizează cu instrucțiunile  $f(x0[1])$ ,  $f(x0[2])$  și  $f(x0[3])$ , obținându-se rezultatul 0. Verificarea soluțiilor se poate face și cu instrucțiunea  $teste(f(x) | x = sn[1])$ , obținându-se rezultatul TRUE.
- Pentru realizarea reprezentării grafice se construiesc patru obiecte grafice: G pentru funcția de analizat  $f(x)$  și G1, G2, G3 pentru cele trei soluții ale ecuației  $f(x) = 0$ .
- Pentru reprezentarea curbei se folosește instrucțiunea  $plot::Function2d$ . Curba se trasează cu linie continuă ( $LineStyle=Solid$ ), culoarea neagră ( $Color=RGB::Black$ ).
- Pentru reprezentarea punctelor se utilizează instrucțiunea  $plot::Point2d$ . Punctele se reprezintă folosind simbolul cerc ( $PointStyle=Circles$ ) cu dimensiunea  $PointSize=2 * unit::mm$  și culoarea neagră ( $Color=RGB::Black$ ).

- Pentru modificarea tuturor parametrilor caracteristici ai reprezentării grafice se deschide interfața Object Browser, [44], prin selectarea domeniului reprezentării grafice și lansarea comenzii Object Browser din toolbar-ul Standard. Această interfață permite accesul la toți parametrii caracteristici ai obiectelor grafice: curbe (Definition, Animation, Annotation, Calculation, Style), puncte (Definition, Animation, Annotation, Style), sistemul de coordonate (Definition, Axes, Tick Marks, Grid Lines), etc. În fereastra superioară Object Browser se selectează obiectul grafic dorit, iar în fereastra inferioară Properties se selectează proprietatea și se modifică valoarea proprietății respective.

```

G:=plot::Function2d(f(x),x=-1.5..2,Color=RGB::Black,LineStyle=Solid,
XTicksDistance=0.5,XTicksBetween=1,YTicksDistance=1,YTicksBetween=1,
GridVisible=TRUE,SubgridVisible=TRUE,ViewingBoxYRange=-3..3):

[G1:=plot::Point2d(x0[1],f(x0[1]),PointSize=2*unit::mm,
Color = RGB::Black,PointStyle=Circles):

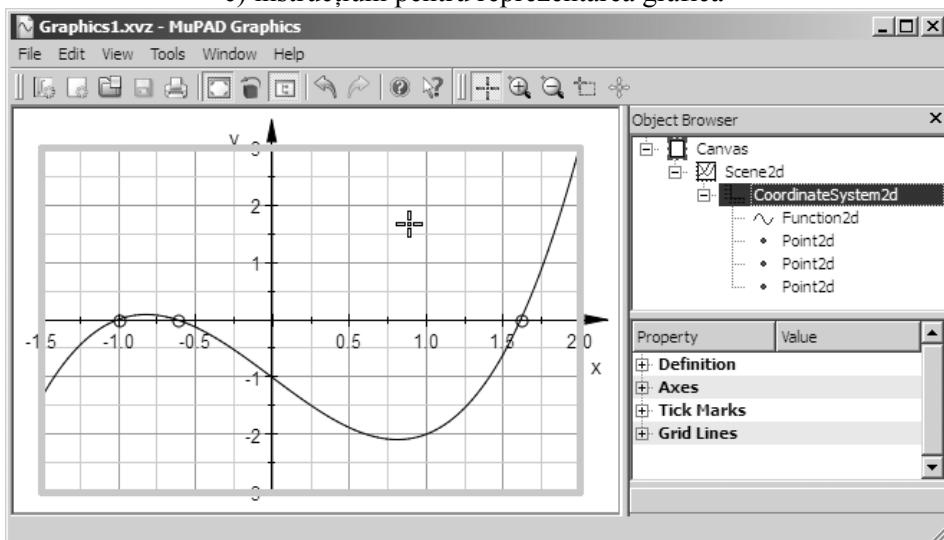
[G2:=plot::Point2d(x0[2],f(x0[2]),PointSize =2*unit::mm,
Color = RGB::Black,PointStyle=Circles):

[G3:=plot::Point2d(x0[3],f(x0[3]),PointSize =2*unit::mm,
Color = RGB::Black,PointStyle=Circles):

plot(G,G1,G2,G3)

```

c) instrucțiuni pentru reprezentarea grafică



d) graficul funcției și soluțiile ecuației

**Figura 11.30.** Rezolvarea problemei 11.9.



### Problema 11.11

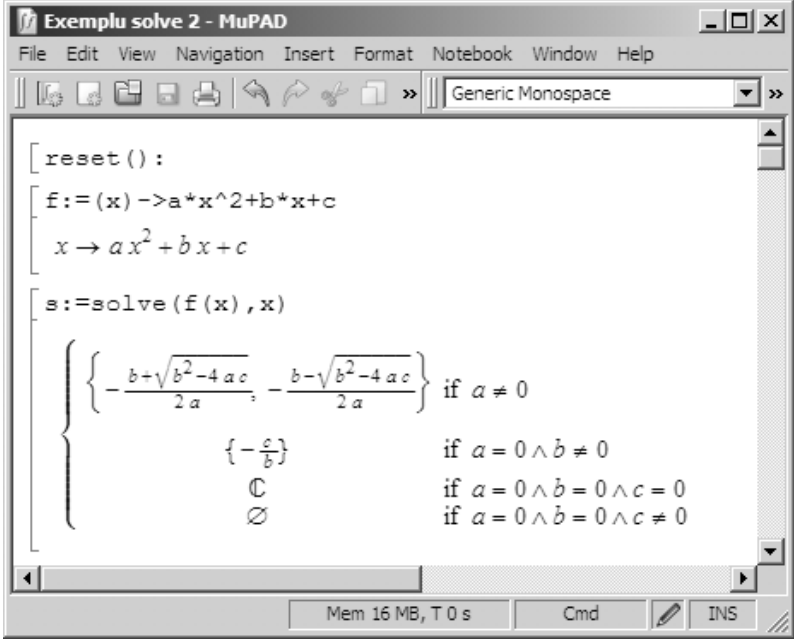
Se consideră ecuația algebrică de gradul al doilea:

$$ax^2 + bx + c = 0$$

pentru care nu se face nici o precizare suplimentară în legătură cu coeficienții  $a$ ,  $b$  și  $c$ . Să se rezolve ecuația (cazul general). Să se găsească rădăcinile ecuației pentru cazul  $a \neq 0$ .

### Rezolvare

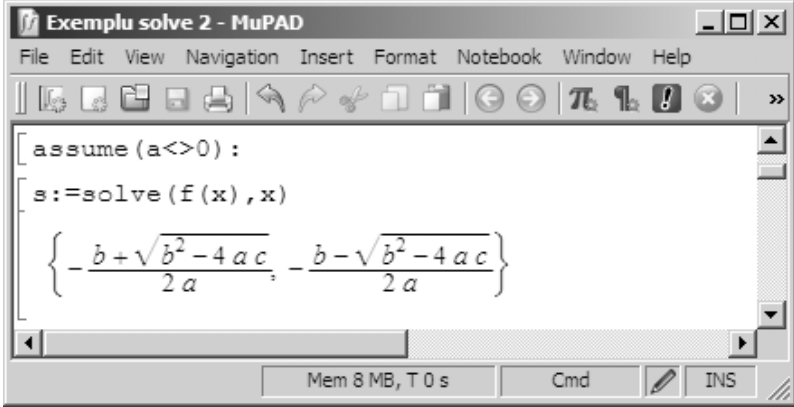
Rezolvarea problemei este prezentată în figura 11.31,a). Introducerea condiției  $a \neq 0$  se face prin intermediul ipotezelor `assume(a<>0)` și `assumeAlso(a>0)`, sau direct `assume(a<>0)`, figura 11.31, b).



```
Exemplu solve 2 - MuPAD
File Edit View Navigation Insert Format Notebook Window Help
Generic Monospace

[ reset() :
[ f:=(x)->a*x^2+b*x+c
[ x -> a x^2 + b x + c
[ s:=solve(f(x),x)
{
  { - (b + sqrt(b^2 - 4 a c)) / (2 a), - (b - sqrt(b^2 - 4 a c)) / (2 a) } if a != 0
  { - c / b } if a = 0 ^ b != 0
  C if a = 0 ^ b = 0 ^ c = 0
  O if a = 0 ^ b = 0 ^ c != 0
}
Mem 16 MB, T 0 s Cmd INS
```

a) cazul general



```
Exemplu solve 2 - MuPAD
File Edit View Navigation Insert Format Notebook Window Help
Mem 8 MB, T 0 s Cmd INS

[ assume(a<>0) :
[ s:=solve(f(x),x)
{
  - (b + sqrt(b^2 - 4 a c)) / (2 a), - (b - sqrt(b^2 - 4 a c)) / (2 a)
}
Mem 8 MB, T 0 s Cmd INS
```

b) cazul  $a \neq 0$

Figura 11.31. Rezolvarea ecuației de gradul doi.

### Problema 11.12

Se consideră funcția definită prin legea:

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

Se cere:

- Să se determine domeniul de definiție al funcției.
- Să se determine soluțiile ecuației  $f(x) = 0$ .
- Să se determine soluțiile inecuațiilor  $f(x) > 0$ , respectiv  $f(x) < 0$ .
- Să se reprezinte grafic funcția.

### Rezolvare

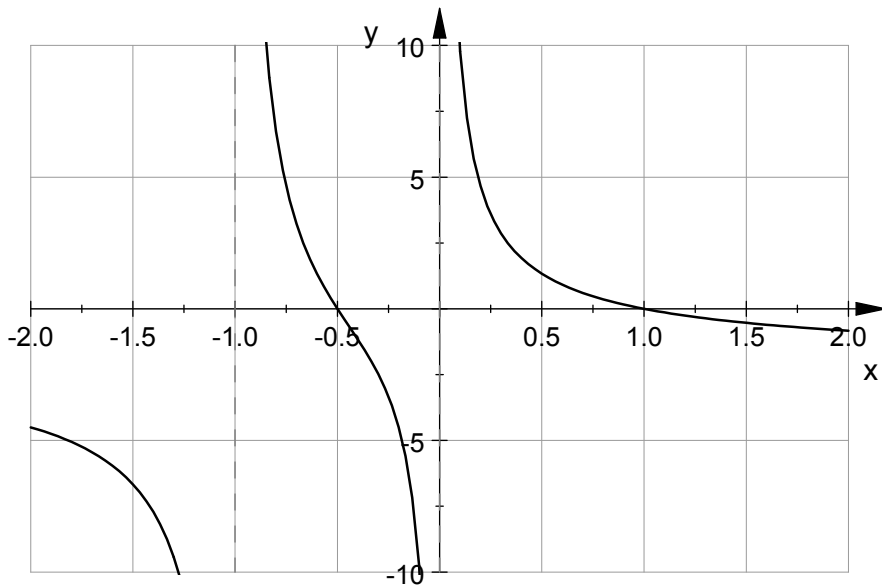
Rezolvarea problemei este prezentată în figura 11.32, a). Graficul obținut este prezentat în figura 11.32, b).

### Observații

- Determinarea punctelor în care funcția nu are sens se face cu instrucțiunea `discont(f(x), x)`. Prin urmare, domeniul de definiție al funcției este:  $\mathbb{R} - \{-1; 0\}$ .
- Rezolvarea ecuației  $f(x) = 0$  se face cu instrucțiunea `s:=solve(f(x)=0, x)`. Se obțin soluțiile  $-1/2$  și  $1$ .

```
reset():
f:=(x)->(x+1)/x-2*x/(x+1)-1
x -> (x+1)/x - 2*x/(x+1) - 1
discont(f(x), x)
{-1, 0}
s:=solve(f(x)=0, x)
{-1/2, 1}
Ip:=solve(f(x)>0, x)
(0, 1) union (-1, -1/2)
In:=solve(f(x)<0, x)
(-1/2, 0) union (1, infinity) union (-infinity, -1)
G1:=plot::Function2d(f(x), x=-2..2, Color=RGB::Black,
LineStyle=Solid, GridVisible=TRUE):
plot(G1)
```

a) instrucțiuni



b) reprezentarea grafică a funcției  
**Figura 11.32.** Rezolvarea problemei 11.11.

### Problema 11.13

Se consideră funcțiile  $f_1: \mathbb{R} \rightarrow \mathbb{R}$  și  $f_2: \mathbb{R} \rightarrow \mathbb{R}$  definite prin legile:

$$\begin{cases} f_1(x, y) = y^2 + x^2 - 1 \\ f_2(x, y) = \pi y - x \end{cases}$$

Să se rezolve și să se reprezinte grafic soluțiile reale ale sistemului de ecuații:

$$S_1: \begin{cases} f_1(x, y) = 0 \\ f_2(x, y) = 0 \end{cases}$$

Să se rezolve și să se reprezinte grafic soluțiile reale ale următoarelor două sisteme de inecuații:

$$S_2: \begin{cases} f_1(x, y) \leq 0 \\ f_2(x, y) \leq 0 \end{cases} \text{ și } S_3: \begin{cases} f_1(x, y) \leq 0 \\ f_2(x, y) \geq 0 \end{cases}$$

### Rezolvare

Pentru definirea celor două funcții  $f_1(x, y)$  și  $f_2(x, y)$ , pentru rezolvarea sistemului de ecuații  $S_1$ , precum și a celor două sisteme de inecuații  $S_2$  și  $S_3$ , într-un fișier MuPAD se scriu instrucțiunile prezentate în figura 11.33, a). Pentru reprezentarea grafică a celor două funcții (G1 și G2), a celor două soluții ale sistemului de ecuații (G3 și G4), precum și a domeniilor reale rezultate din rezolvarea celor două sisteme de inecuații (G5 și G6) se folosesc instrucțiunile prezentate în figura 11.33, b). Graficele rezultate sunt prezentate în figura 11.33, c) (G1, G2, G3, G4), figura 11.33, d) (G5) și figura 11.33, e) (G6).

```

Exemplu solve 4 - MuPAD
File Edit View Navigation Insert Format Notebook Window Help
[
reset():
f1:=(x,y)->x^2+y^2-1
(x,y) -> x^2 + y^2 - 1
f2:=(x,y)->PI*y-x
(x,y) -> pi y - x
s1:=solve([f1(x,y)=0,f2(x,y)=0],[x,y],Real)
{ [x = pi/sqrt(pi^2+1), y = 1/sqrt(pi^2+1)], [x = -pi/sqrt(pi^2+1), y = -1/sqrt(pi^2+1)] }
s2:=solve([f1(x,y)<=0,f2(x,y)<=0],[x,y],Real)
(x,y) in R^2 cap ( union_{v in [-1,1]} { (u/v) | u in [pi*v, sqrt(1-v^2)] cap [-sqrt(1-v^2), sqrt(1-v^2)] } )
s3:=solve([f1(x,y)<=0,f2(x,y)>=0],[x,y],Real)
(x,y) in R^2 cap ( union_{v in [-1,1]} { (u/v) | u in [-sqrt(1-v^2), sqrt(1-v^2)] cap [-sqrt(1-v^2), pi*v] } )
]
Mem 17 MB, T 0 s Cmd INS

```

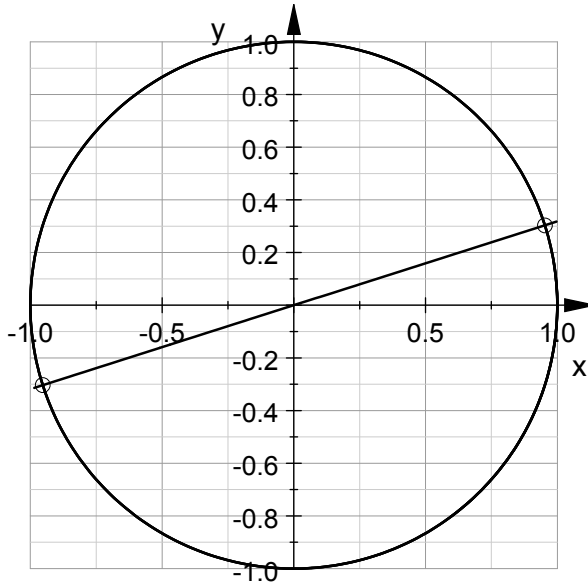
a) instrucțiuni pentru rezolvarea sistemelor de ecuații și inecuații

```

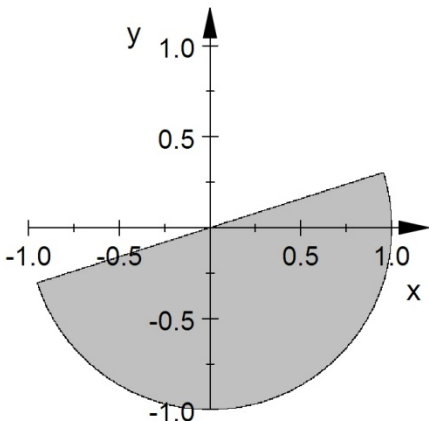
Exemplu solve 4 - MuPAD
File Edit View Navigation Insert Format Notebook Window Help
[G1:=plot::Implicit2d(f1(x,y),x=-1..1,y=-1..1,
Color=RGB::Black,LineStyle=Solid,GridVisible=TRUE,
SubgridVisible=TRUE,Scaling=Constrained):
G2:=plot::Implicit2d(f2(x,y),x=-1..1,y=-1..1,
Color=RGB::Black,LineStyle=Solid,GridVisible=TRUE,
SubgridVisible=TRUE,Scaling=Constrained):
G3:=plot::Point2d(s1[1],Color=RGB::Black,PointStyle=Circles,
PointSize=2*unit:mm):
G4:=plot::Point2d(s1[2],Color=RGB::Black,PointStyle=Circles,
PointSize=2*unit:mm):
G5:=plot::Inequality([f1(x,y)<=0,f2(x,y)<=0],x=-1..1,y=-1..1,
Mesh=[400,400],FillColorFalse=RGB::White,FillColorTrue= RGB::Gray,
Scaling=Constrained):
G6:=plot::Inequality([f1(x,y)<=0,f2(x,y)>=0],x=-1..1,y=-1..1,
Mesh=[400,400],FillColorFalse=RGB::White,FillColorTrue= RGB::Gray,
Scaling=Constrained):
plot(G1,G2,G3,G4);plot(G5);plot(G6)
]
Mem 17 MB, T 14 s Cmd INS

```

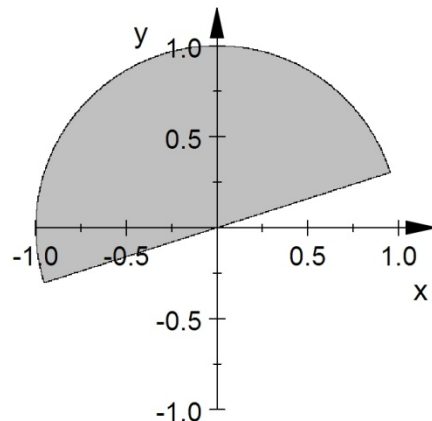
b) instrucțiuni pentru reprezentarea grafică a soluțiilor



c) soluțiile sistemului de ecuații  $S_1$



d) soluțiile sistemului de inecuații  $S_2$



e) soluțiile sistemului de inecuații  $S_3$

**Figura 11.33.** Rezolvarea problemei 11.12.

### Observații

- Pentru identificarea doar a soluțiilor reale, în corpul instrucțiunii `solve` se folosește parametrul opțional `Real`.
- Pentru reprezentarea grafică a domeniilor reale care definesc soluțiile celor două sisteme de inecuații se folosește instrucțiunea `plot::Inequality`, [45]. Domeniul care satisface sistemul de inecuații are culoarea definită prin parametrul opțional `FillColorTrue=RGB::Gray`. Domeniul care nu satisface sistemul de inecuații are culoarea definită prin parametrul opțional `FillColorFalse=RGB::White`.

### Problema 11.14

Se consideră sistemul de ecuații algebrice liniare:

$$S: \begin{cases} 2x_1 + x_2 + 3x_3 = 2 \\ 3x_1 + 3x_2 + 2x_3 = 11 \\ x_1 + 2x_2 + x_3 = 5 \end{cases}$$

Să se rezolve sistemul de ecuații.

### Rezolvare

În cazul în care sistemul de ecuații este definit prin setul de ecuații componente, rezolvarea problemei este prezentată în figura 11.34, a) pentru cazul instrucțiunii `solve` și în figura 11.34, b) pentru cazul instrucțiunii `linsolve`.

Metoda transformării setului de ecuații în formă matriceală este prezentată în figura 11.34, c).

În cazul în care sistemul de ecuații este definit direct sub formă matriceală, rezolvarea problemei este prezentată în figura 11.34, d).

### Observații

- Sistemul de ecuații este definit prin setul celor trei ecuații componente cu ajutorul instrucțiunii:  $S := \{x+2*y+z=1, 2*x-y+3*z=2, x+y-z=-1\}$ .
- Folosind instrucțiunea `solve`, soluția sistemului de ecuații se obține cu instrucțiunea `s1:=solve(S, [x,y,z])`, figura 11.34, a). În acest caz soluțiile sistemului se obțin sub forma unei mulțimi de valori. Valorile obținute sunt  $\{[x=-3/11, y=2/11, z=10/11]\}$ . Verificarea soluției obținute se face prin asignarea valorilor numerice  $-3/11$ ,  $2/11$  și  $10/11$  celor trei variabile  $x, y$  și  $z$  și evaluarea apoi a ecuațiilor componente ale sistemului folosind instrucțiunile `assign(op(s1)):S`.
- Folosind instrucțiunea `linsolve`, soluția sistemului de ecuații se obține, în mod direct, cu instrucțiunea `s2:=linsolve(S, [x,y,z])`, figura 11.34, b). Verificarea soluției obținute se face prin înlocuirea directă a valorilor obținute  $[x=-3/11, y=2/11, z=10/11]$  în sistemul de ecuații  $S$  folosind instrucțiunea `S|s2`.
- Sistemul de ecuații  $S$ , definit în prealabil prin setul celor trei ecuații componente se poate transforma în formă matriceală folosind instrucțiunea `C:=linalg::expr2Matrix(S, [x,y,z])`, [46], figura 11.34, c). Matricea obținută  $C$ , reprezintă matricea extinsă definită prin concatenarea matricei coeficienților și vectorului termenilor liberi. Pornind de la matricea extinsă  $C$ , rezolvarea

sistemului de ecuații se realizează cu ajutorul instrucțiunii  $s3:=linalg::matlinsolve(C)$ , [47].

- Metoda matriceală de rezolvare constă în definirea matricei coeficienților și a vectorului termenilor liberi cu ajutorul instrucțiunilor  $A:=matrix([ [1,2,1], [2,-1,3], [1,1,-1] ])$  și  $b:=matrix([1,2,-1])$ , [48] și prin aplicarea apoi a instrucțiunii de rezolvare  $s4:=linalg::matlinsolve(A,b)$ .

```

[ reset() :
[ S:={x+2*y+z=1, 2*x-y+3*z=2, x+y-z=-1}
  {2x-y+3z=2, x+y-z=-1, x+2y+z=1}
[ s1:=solve(S, [x, y, z])
  { [x = -3/11, y = 2/11, z = 10/11] }
[ assign(op(s1)) : S
  {-1 = -1, 1 = 1, 2 = 2}

```

a) metoda setului de ecuații, instrucțiunea solve

```

[ reset() :
[ S:={x+2*y+z=1, 2*x-y+3*z=2, x+y-z=-1}
  {2x-y+3z=2, x+y-z=-1, x+2y+z=1}
[ s2:=linsolve(S, [x, y, z])
  [x = -3/11, y = 2/11, z = 10/11]
[ S|s2
  {-1 = -1, 1 = 1, 2 = 2}

```

b) metoda setului de ecuații, instrucțiunea linsolve

```

[ reset() :
[ S:={x+2*y+z=1, 2*x-y+3*z=2, x+y-z=-1}
  {2x-y+3z=2, x+y-z=-1, x+2y+z=1}
[ C:=linalg::expr2Matrix(S, [x, y, z])
  ( 1 2 1 1 )
  ( 2 -1 3 2 )
  ( 1 1 -1 -1 )
[ s3:=linalg::matlinsolve(C)
  ( -3/11 )
  ( 2/11 )
  ( 10/11 )

```

c) transformarea setului de ecuații în formă matriceală

```

[ reset() :
[ A:=matrix([[1, 2, 1], [2, -1, 3], [1, 1, -1]])
  ( 1 2 1 )
  ( 2 -1 3 )
  ( 1 1 -1 )
[ b:=matrix([1, 2, -1])
  ( 1 )
  ( 2 )
  ( -1 )
[ s4:=linalg::matlinsolve(A, b)
  ( -3/11 )
  ( 2/11 )
  ( 10/11 )
[ A*s4
  ( 1 )
  ( 2 )
  ( -1 )

```

d) metoda matriceală

**Figura 11.34.** Rezolvarea sistemelor de ecuații algebrice liniare.



## 11.7. REZOLVAREA ECUAȚILOR ȘI SISTEMELOR DE ECUAȚII DIFERENȚIALE ORDINARE

Principalele etape necesare pentru rezolvarea ecuațiilor și sistemelor de ecuații diferențiale ordinare sunt:

- Definirea ecuației diferențiale de rezolvat. Se realizează folosind instrucțiunea `ode`, [49]. În procesul de definire a ecuației diferențiale se consideră următoarele exprimări ale operatorilor de derivare de ordinul 1, respectiv de ordinul 2:

$$\frac{dy}{dt} \Leftrightarrow \text{diff}(y(t), t) \text{ sau } y'(t)$$
$$\frac{d^2y}{dt^2} \Leftrightarrow \text{diff}(y(t), t, t) \text{ sau } y''(t)$$

- Rezolvarea ecuației diferențiale poate avea ca scop:
  - Determinarea soluției generale. Se realizează cu ajutorul instrucțiunii `solve`.
  - Determinarea unor soluții particulare, considerând o serie de condiții suplimentare pe care aceste soluții trebuie să le respecte. Condițiile suplimentare, fie că sunt condiții inițiale, fie condiții pe frontieră, se introduc în procesul de definire a ecuației diferențiale.
  - Reprezentarea grafică a curbelor integrale. Se realizează cu ajutorul instrucțiunii `plot::Function2d`. Reprezentarea grafică a curbelor integrale se poate realiza și fără obținerea soluțiilor propriu-zise ale ecuației diferențiale, folosindu-se instrucțiunile `plot::Ode2d`, [50], respectiv `plot::Ode3d`, [51].
  - Reprezentarea grafică a câmpului de vectori atașat ecuației diferențiale de rezolvat. Se realizează cu ajutorul instrucțiunile `plot::VectorField2d`, [52], respectiv `plot::VectorField3d`, [53].
- Instrucțiunea generală `solve` se utilizează pentru orice tip de ecuație diferențială de rezolvat. În cazul unor ecuații diferențiale particulare (de exemplu pentru rezolvarea ecuațiilor diferențiale de tip Abel, Bernoulli, Chini, Clairaut, Lagrange, Riccati, ecuații omogene, ecuații cu variabile separabile) se poate introduce în corpul instrucțiunii `solve` un parametru opțional care să specifice tipul ecuației de rezolvat, de exemplu `Type=Lagrange`.
- Definirea sistemelor de ecuații diferențiale ordinare se poate face fie sub forma unei mulțimi de ecuații diferențiale, fie sub formă matriceală. Pentru rezolvarea numerică a ecuațiilor și sistemelor de ecuații diferențiale ordinare se utilizează instrucțiunea `numeric::odesolve`, [54].

### Problema 11.15

Se consideră ecuația diferențială ordinară de ordinul 1:

$$y'(t) = \sin t - \pi \cdot y(t)$$

Se cere:

- Să se găsească soluția generală a ecuației diferențiale.
- Să se găsească soluția particulară corespunzătoare condiției inițiale  $y(0) = 0$ . Să se reprezinte curba integrală obținută.
- Să se reprezintă curba integrală corespunzătoare condiției inițiale  $y(0) = 0$ , precum și câmpul de vectori tangenți la curbele integrale ale ecuației diferențiale.

### Rezolvare

Pentru determinarea soluției generale (sg) a ecuației diferențiale se folosesc instrucțiunile prezentate în figura 11.35, a). Definirea ecuației diferențiale se face cu instrucțiunea  $eq:=diff(y(t),t)=f(t,y)$ . Același rezultat se obține prin utilizarea instrucțiunii  $eq:=y'(t)=f(t,y)$ . Instrucțiunile necesare pentru determinarea soluției particulare (sp) corespunzătoare condiției inițiale  $y(0) = 0$  sunt prezentate în figura 11.35, b). Constanta de integrare C4 din expresia soluției generale sn se determină pornind de la condiția inițială  $y(0) = 0$ , rezultând astfel soluția particulară sp. Pentru reprezentarea grafică a curbei integrale, precum și a câmpului de vectori tangenți curbelor integrale ale ecuației diferențiale se utilizează instrucțiunile prezentate în figura 11.35, c). Graficul obținut este prezentat în figura 11.35, d).

```
[ reset() :  
[ f:=(t,y)->sin(t)-PI*y(t)  
(t,y) -> sin(t) - pi y(t)  
[ eq:=diff(y(t),t)=f(t,y)  
 $\frac{\partial}{\partial t} y(t) = \sin(t) - \pi y(t)$   
[ s:=ode({eq},y(t)):  
[ sg:=solve(s)  
 $\left\{ \frac{C4}{e^{\pi t}} - \frac{\cos(t) - \pi \sin(t)}{\pi^2 + 1} \right\}$ 
```

a) soluția generală sg

```

Exemplu odesolve 1 - MuPAD
File Edit View Navigation Insert Format Notebook Window Help
[
y0:=y(0)=0
y(0)=0
S:=ode({eq,y0},y(t)):
sp:=solve(S)
{
1/(e^pi*t*(pi^2+1)) - (cos(t)-pi*sin(t))/(pi^2+1)
}
Mem 21 MB, T 0 s Cmd INS

```

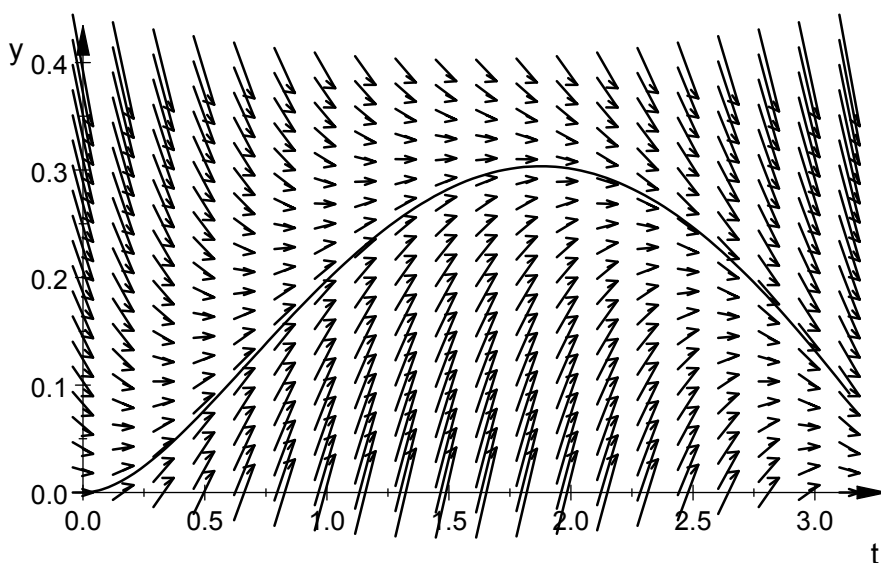
b) soluția particulară  $sp$

```

Exemplu odesolve 1 - MuPAD
File Edit View Navigation Insert Format Notebook Window Help
[
G1:=plot::Function2d(sp[1],t=0..PI,Color=RGB::Black):
G2:=plot::VectorField2d([1,f(t,y)],t=0..PI,y=0..PI/8,
Mesh=[20,20],Color=RGB::Black):
plot(G1,G2)
Mem 21 MB, T 0 s Cmd INS

```

c) instrucțiuni pentru realizarea reprezentării grafice



d) curba integrală și câmpul de vectori asociați ecuației diferențiale  
**Figura 11.35.** Rezolvarea problemei 11.14.

### Problema 11.16

Se consideră ecuația diferențială ordinară de ordinul 2:

$$\ddot{y} + 2\delta\dot{y} + \omega^2 y = 0$$

Se cere:

- Să se găsească soluția generală a ecuației diferențiale.
- Să se găsească soluția particulară corespunzătoare condițiilor inițiale  $y(0) = 1$  și  $y'(0) = 0$ .
- Să se determine soluția particulară pentru următoarele cazuri:  $\omega=3, \delta=1$ ;  $\omega=3, \delta=3$  și  $\omega=3, \delta=6$ . Să se reprezinte grafic cele trei soluții particulare.

### Rezolvare

Pentru determinarea soluției generale (sg) a ecuației diferențiale se folosesc instrucțiunile prezentate în figura 11.36, a). Definirea operatorilor de derivare de ordinul 1, respectiv de ordinul 2 se face folosind exprimările  $y'(t)$ , respectiv  $y''(t)$ . Se observă modul de introducere a caracterelor grecești  $\delta$  și  $\omega$ . În expresia analitică a soluției generale astfel obținute se observă prezența a două constante de integrare identificate prin C4 și C13.

Instrucțiunile necesare pentru determinarea soluției particulare (sp) corespunzătoare condițiilor inițiale  $y(0) = 0$  și  $y'(0) = 0$  sunt prezentate în figura 11.36, b). Pentru determinarea soluțiilor particulare sp1 ( $\omega=3, \delta=1$ ), sp2 ( $\omega=3, \delta=3$ ) și sp3 ( $\omega=3, \delta=6$ ) se folosesc instrucțiunile prezentate în figura 11.36, c). Se observă modul de utilizare a instrucțiunii subs, [55] pentru realizarea substituțiilor simbolice între variabilele  $\omega$  și  $\delta$  și valorile lor numerice.

```
reset();
eq:=y''(t)+2*delta*y'(t)+omega*y(t);
y'(t)+0*y(t)+2*delta*y'(t);
s:=ode({eq},y(t));
sg:=solve(s)
{
  C13 / e^{t*(delta+sqrt(delta^2-omega))} +
  C4 / e^{t*(delta-sqrt(delta^2-omega))}
}
```

a) soluția generală sg

Exemplu odesolve 2 - MuPAD

```

File Edit View Navigation Insert Format Notebook Window Help
[
y0:=y(0)=1, y'(0)=0
y(0)=1, y'(0)=0
S:=ode({eq, y0}, y(t)):
sp:=solve(S)

```

$$\left\{ \begin{array}{ll} \left\{ \frac{\delta + \sigma_1}{2e^{t(\delta - \sigma_1)}} - \frac{\delta - \sigma_1}{2e^{t(\delta + \sigma_1)}} \right\} & \text{if } \omega \neq \delta^2 \\ \{1\} & \text{if } \delta = 0 \wedge \omega = 0 \\ \emptyset & \text{if } \omega = \delta^2 \wedge \delta \neq 0 \end{array} \right.$$

where

$$\sigma_1 = \sqrt{\delta^2 - \omega}$$

Mem 12 MB, T 0 s    Cmd    INS

b) soluția particulară sp

Exemplu odesolve 2 - MuPAD

```

File Edit View Navigation Insert Format Notebook Window Help
[
sp1:=subs(sp, `&omega;`=3, `&delta;`=1)

```

$$\left\{ -\frac{\sqrt{-2} e^{t(-1+\sqrt{2}i)} (1+\sqrt{2}i)}{4} - \frac{\sqrt{-2} (-1+\sqrt{2}i)}{4 e^{t(1+\sqrt{2}i)}} \right\}$$

```

[
sp2:=subs(sp, `&omega;`=3, `&delta;`=3)

```

$$\left\{ \frac{\sqrt{6} e^{t(\sqrt{6}-3)} (\sqrt{6}+3)}{12} + \frac{\sqrt{6} (\sqrt{6}-3)}{12 e^{t(\sqrt{6}+3)}} \right\}$$

```

[
sp3:=subs(sp, `&omega;`=3, `&delta;`=6)

```

$$\left\{ \frac{\sqrt{33} e^{t(\sqrt{33}-6)} (\sqrt{33}+6)}{66} + \frac{\sqrt{33} (\sqrt{33}-6)}{66 e^{t(\sqrt{33}+6)}} \right\}$$

Mem 12 MB, T 0 s    Cmd    INS

c) soluțiile particulare sp1, sp2 și sp3

Pentru reprezentarea grafică a celor trei curbe integrale se utilizează instrucțiunile prezentate în figura 11.36, d). Se definesc trei obiecte grafice G1, G2 și G3 folosind instrucțiunea `plot::Function2d` pentru fiecare din cele trei soluții particulare `sp1`, `sp2` și `sp3`. Intervalul de timp utilizat pentru cele trei reprezentări grafice este  $t=0 \dots 10$ . Toate cele trei curbe au culoarea neagră (`Color=RGB::Black`), însă pentru identificarea clară a acestora se utilizează linii de diferite tipuri: linia continuă pentru soluția `sp1`, linia întreruptă pentru soluția `sp2` și linie punctată pentru soluția `sp3`. Se modifică de asemenea și grosimea liniei, în mod unitar, pentru toate cele trei curbe (`LineWidth=.4*unit::mm`). Afișarea rețelei de linii ajutătoare de tip grid se face cu parametrul `GridVisible=TRUE`.

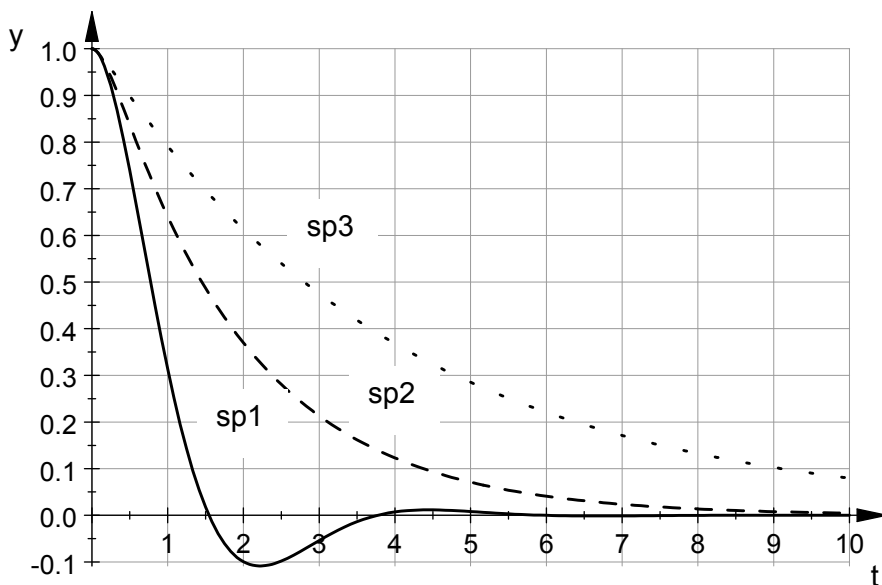
Graficul obținut este prezentat în figura 11.36, e).

```

Exemplu odesolve 2 - MuPAD
File Edit View Navigation Insert Format Notebook Window Help
[G1:=plot::Function2d(sp1[1],t=0..10,LineStyle=Solid):
[G2:=plot::Function2d(sp2[1],t=0..10,LineStyle=Dashed):
[G3:=plot::Function2d(sp3[1],t=0..10,LineStyle=Dotted):
plot(G1,G2,G3,GridVisible=TRUE,Color=RGB::Black,
LineWidth=.4*unit::mm)
Mem 12 MB, T 0 s Cmd INS

```

d) instrucțiuni pentru realizarea reprezentării grafice



e) curbele integrale ale soluțiilor particulare

**Figura 11.36.** Rezolvarea problemei 11.15.

### Problema 11.17

Se consideră sistemul de ecuații diferențiale:

$$S: \begin{cases} y'(t) = ay(t) + bz(t) \\ z'(t) = cy(t) + dz(t) \end{cases}$$

Se cere:

- Să se găsească soluția generală a sistemului de ecuații diferențiale.
- Să se găsească soluția particulară sp corespunzătoare condițiilor inițiale  $y(0) = 1, z(0) = 0$ .
- Să se determine soluțiile particulare sp1 și sp2 pentru următoarele cazuri:  $a=1, b=2, c=-2, d=1$  și  $a=0, b=1, c=-1, d=-1$ . Să se reprezinte grafic soluțiile  $y(t), z(t)$ , precum și orbitele corespunzătoare din spațiul fazelor  $(z(t), y(t))$ .

### Rezolvare

Pentru determinarea soluției generale (sg) a sistemului de ecuații diferențiale se folosesc instrucțiunile prezentate în figura 11.37, a). Se observă modul de definire a celor două ecuații diferențiale, precum și a sistemului  $S$  (instrucțiunea `ode`). Rezolvarea sistemului de ecuații diferențiale se face cu instrucțiunea `solve`. În soluția generală astfel obținută se observă prezența a două constante de integrare identificate prin  $C1$  și  $C4$ , dar și a trei parametri numerici  $\sigma_1, \sigma_2$  și  $\sigma_3$ .

```
Exemplu odesolve 3 - MuPAD
File Edit View Navigation Insert Format Notebook Window Help
[reset():
[eq:={y'(t)-a*y(t)-b*z(t)=0, z'(t)-c*y(t)-d*z(t)=0}
{y'(t)-a*y(t)-b*z(t)=0, z'(t)-c*y(t)-d*z(t)=0}
[S:=ode(eq, {y(t), z(t)}):
[sg:=solve(S)
{[z(t) = -C1*a*sigma_1+C4*a*sigma_2-C1*d*sigma_1-C4*d*sigma_2+C1*sigma_1*sigma_3-C4*sigma_2*sigma_3, y(t) = C1*sigma_1+C4*sigma_2]}
where
sigma_1 = e^(a*t/2 + d*t/2 - t*sigma_3/2)
sigma_2 = e^(a*t/2 - d*t/2 + t*sigma_3/2)
sigma_3 = sqrt(a^2 - 2*a*d + d^2 + 4*b*c)
Mem 16 MB, T 0 s Cmd INS
```

a) soluția generală sg

În figura 11.37, b) sunt prezentate instrucțiunile necesare pentru determinarea soluției particulare  $sp$  corespunzătoare condițiilor inițiale  $y(0) = 1$  și  $z(0) = 0$ , precum și pentru determinarea soluțiilor particulare  $sp1$  și  $sp2$  corespunzătoare valorilor particulare ale coeficienților  $a=1, b=2, c=-2, d=1$ , respectiv  $a=0, b=1, c=-1, d=-1$ . Se observă expresiile analitice ale soluțiilor sistemului de ecuații diferențiale în care intervin o serie de parametri numerici:  $\sigma_1$  și  $\sigma_2$  pentru soluția particulară  $sp1$ , respectiv  $\sigma_1, \sigma_2, \sigma_3$  și  $\sigma_4$  pentru soluția particulară  $sp2$ . Se observă de asemenea, modul de utilizare a instrucțiunii `subs` pentru realizarea substituțiilor simbolice între variabilele  $a, b, c$  și  $d$  și valorile lor numerice pentru ambele soluții particulare  $sp1$ , respectiv  $sp2$ .

```

Exemplu odesolve 3 - MuPAD
File Edit View Navigation Insert Format Notebook Window Help
[ y0:=y(0)=1, z(0)=0
  y(0) = 1, z(0) = 0
[ S:=ode({op(eq), y0}, {y(t), z(t)}):
[ sp:=solve(S):
[ sp1:=subs(sp, a=1, b=2, c=-2, d=1)
  { [ z(t) = -\frac{\sqrt{16}\sigma_1 i}{8} + \frac{\sqrt{16}\sigma_2 i}{8}, y(t) = -\frac{\sqrt{-16}\sqrt{16}\sigma_1 i}{32} - \frac{\sqrt{-16}\sqrt{16}\sigma_2 i}{32} ] }
  where
    \sigma_1 = e^{\frac{t - \sqrt{16} t i}{2}}
    \sigma_2 = e^{\frac{t + \sqrt{16} t i}{2}}
[ sp2:=subs(sp, a=0, b=1, c=-1, d=-1)
  { [ z(t) = -\frac{\sigma_3}{4\sigma_1} + \frac{\sigma_4}{4\sigma_2} + \frac{\sqrt{-3}\sigma_3}{12\sigma_1} + \frac{\sqrt{-3}\sigma_4}{12\sigma_2}, y(t) = -\frac{\sqrt{-3}\sigma_3}{6\sigma_1} - \frac{\sqrt{-3}\sigma_4}{6\sigma_2} ] }
  where
    \sigma_1 = e^{\frac{t}{2} + \frac{\sqrt{3} t i}{2}}
    \sigma_2 = e^{\frac{t}{2} - \frac{\sqrt{3} t i}{2}}
    \sigma_3 = -1 + \sqrt{3} i
    \sigma_4 = 1 + \sqrt{3} i
  
```

b) soluțiile particulare  $sp$ ,  $sp1$  și  $sp2$



Instrucțiunile pentru extragerea soluțiilor individuale  $y_1(t)$ ,  $z_1(t)$ , respectiv  $y_2(t)$ ,  $z_2(t)$  din mulțimea soluțiilor  $sp_1$ , respectiv  $sp_2$  sunt prezentate în figura 11.37, c) : Simplify, op și rhs, [56].

```

Exemplu odesolve 3 - MuPAD
File Edit View Navigation Insert Format Notebook Window Help
[ y1:=Simplify(rhs(op(sp1[1])[2]))
  cos(2 t) e^t
[ z1:=Simplify(rhs(op(sp1[1])[1]));
  -sin(2 t) e^t
[ y2:=Simplify(rhs(op(sp2[1])[2]))
  3 cos(\frac{\sqrt{3} t}{2}) + \sqrt{3} sin(\frac{\sqrt{3} t}{2})
  -----
  3 e^{\frac{t}{2}}
[ z2:=Simplify(rhs(op(sp2[1])[1]));
  2 \sqrt{3} sin(\frac{\sqrt{3} t}{2})
  -----
  3 e^{\frac{t}{2}}
  
```

c) soluțiile individuale  $y_1(t)$ ,  $z_1(t)$ , respectiv  $y_2(t)$ ,  $z_2(t)$

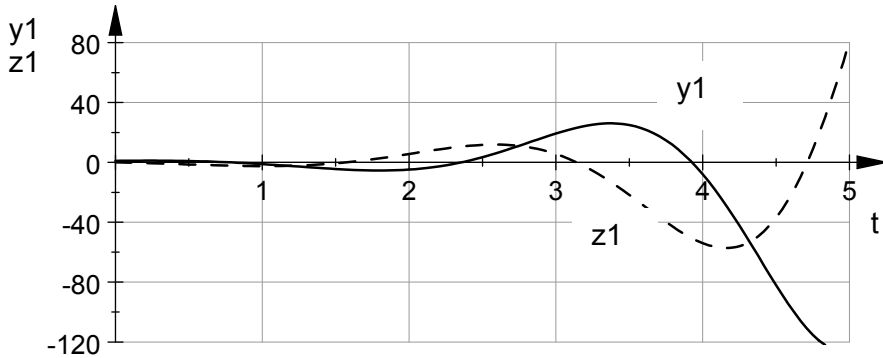
Pentru reprezentarea grafică a soluțiilor precum și a orbitelor corespunzătoare din spațiul fazelor se utilizează instrucțiunile prezentate în figura 11.37, d).

```

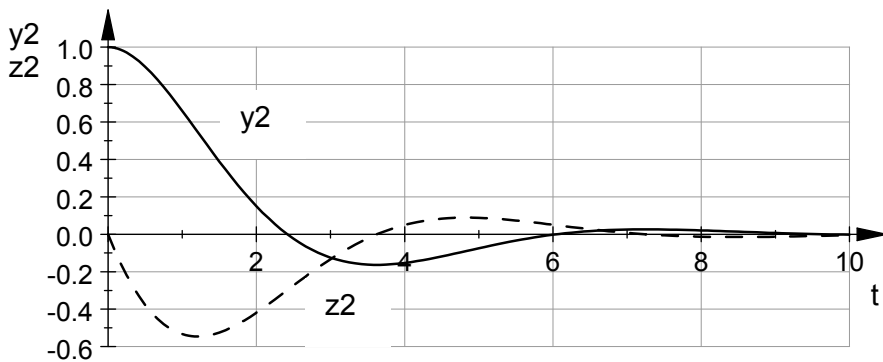
Exemplu odesolve 3 - MuPAD
File Edit View Navigation Insert Format Notebook Window Help
[ G11:=plot::Function2d(y1,t=0..5,LineStyle=Solid):
[ G12:=plot::Function2d(z1,t=0..5,LineStyle=Dashed):
[ G13:=plot::Curve2d([y1,z1],t=0..5):
[ G21:=plot::Function2d(y2,t=0..10,LineStyle=Solid):
[ G22:=plot::Function2d(z2,t=0..10,LineStyle=Dashed):
[ G23:=plot::Curve2d([y2,z2],t=0..10):
plot(G11,G12,GridVisible=TRUE,Color=RGB::Black,
  LineWidth=.4*unit:mm):
plot(G13,GridVisible=TRUE,Color=RGB::Black,
  Scaling=Constrained)
plot(G21,G22,GridVisible=TRUE,Color=RGB::Black,
  LineWidth=.4*unit:mm)
plot(G23,GridVisible=TRUE,Color=RGB::Black,
  Scaling=Constrained)
  
```

d) instrucțiuni pentru realizarea reprezentărilor grafice

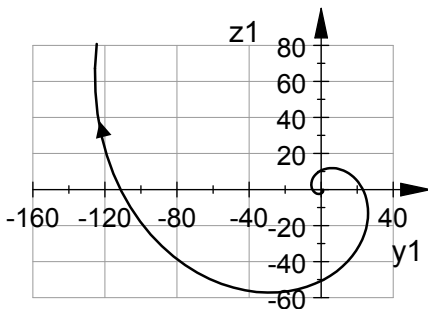
Graficele obținute sunt prezentate în figura 11.37, e) pentru curbele integrale ale soluțiilor particulare  $y_1(t)$ ,  $z_1(t)$ ; în figura 11.37, f) pentru curbele integrale ale soluțiilor particulare  $y_2(t)$ ,  $z_2(t)$ ; în figura 11.37, g) pentru orbita  $(z_1, y_1)$  din spațiul fazelor (spirală instabilă); în figura 11.37, h) pentru orbita  $(z_2, y_2)$  din spațiul fazelor (spirală asimptotic stabilă).



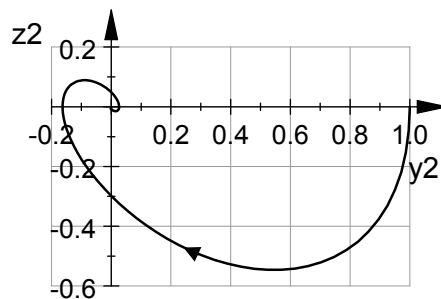
e) curbele integrale ale soluțiilor particulare  $y_1(t)$ ,  $z_1(t)$



f) curbele integrale ale soluțiilor particulare  $y_2(t)$ ,  $z_2(t)$



g) orbita  $(z_1, y_1)$



h) orbita  $(z_2, y_2)$

**Figura 11.37.** Rezolvarea problemei 11.16.

## BIBLIOGRAFIE

1. Gehrs K., Postel F., MuPAD, A Practical Guide, Mathematics Made Anew: Tools and Texts for Computer Aided Learning, Vol. 1, SciFace Software, Paderborn, 2003.
2. MathWorks, MuPAD-User's Guide, 2014.
3. MathWorks, MuPAD, <http://www.mathworks.com/help/symbolic/mupad.html>, accesat la 17.05.2014.
4. MathWorks, Set Preferences for Notebooks, <http://www.mathworks.com/help/symbolic/mupad Ug/set-preferences-for-notebooks.html>, accesat la 14.08.2014.
5. MathWorks, Assign Variables, [http://www.mathworks.com/help/symbolic/mupad\\_ref/\\_assign.html](http://www.mathworks.com/help/symbolic/mupad_ref/_assign.html), accesat la 14.08.2014.
6. MathWorks, Set Assumption on Symbolic Object, <http://www.mathworks.com/help/symbolic/assume.html>, accesat la 14.08.2014.
7. MathWorks, Simplify Representation of Uncertain Object, <http://www.mathworks.com/help/robust/ref/simplify.html>, accesat la 14.08.2014.
8. MathWorks, Simplify an Expression, [http://www.mathworks.com/help/symbolic/mupad\\_ref/simplify1.html](http://www.mathworks.com/help/symbolic/mupad_ref/simplify1.html), accesat la 14.08.2014.
9. MathWorks, Simplify Radicals in Arithmetical Expressions, [http://www.mathworks.com/help/symbolic/mupad\\_ref/radsimp.html](http://www.mathworks.com/help/symbolic/mupad_ref/radsimp.html), accesat la 14.08.2014.
10. MathWorks, Expand an Expression, [http://www.mathworks.com/help/symbolic/mupad\\_ref/expand.html](http://www.mathworks.com/help/symbolic/mupad_ref/expand.html), accesat la 14.08.2014.
11. MathWorks, Factor a Polynomial Into Irreducible Polynomials, [http://www.mathworks.com/help/symbolic/mupad\\_ref/factor.html](http://www.mathworks.com/help/symbolic/mupad_ref/factor.html), accesat la 14.08.2014.
12. MathWorks, Collect Terms With the Same Powers, [http://www.mathworks.com/help/symbolic/mupad\\_ref/collect.html](http://www.mathworks.com/help/symbolic/mupad_ref/collect.html), accesat la 14.08.2014.
13. MathWorks, Rewrite an Expression, [http://www.mathworks.com/help/symbolic/mupad\\_ref/rewrite.html](http://www.mathworks.com/help/symbolic/mupad_ref/rewrite.html), accesat la 14.08.2014.
14. MathWorks, Rectangular Form of a Complex Expression, [http://www.mathworks.com/help/symbolic/mupad\\_ref/rectform.html](http://www.mathworks.com/help/symbolic/mupad_ref/rectform.html), accesat la 14.08.2014.
15. MathWorks, Denominator of a Rational Expression, [http://www.mathworks.com/help/symbolic/mupad\\_ref/denom.html](http://www.mathworks.com/help/symbolic/mupad_ref/denom.html), accesat la 14.08.2014.
16. MathWorks, Numerator of a Rational Expression, [http://www.mathworks.com/help/symbolic/mupad\\_ref/numer.html](http://www.mathworks.com/help/symbolic/mupad_ref/numer.html), accesat la 14.08.2014.
17. MathWorks, Compute Limit of Symbolic Expression, <http://www.mathworks.com/help/symbolic/limit.html>, accesat la 14.08.2014.
18. MathWorks, Differentiate Symbolic Expression or Function, <http://www.mathworks.com/help/symbolic/diff.html>, accesat la 14.08.2014.

19. MathWorks, Definite and Indefinite Integrals, <http://www.mathworks.com/help/symbolic/int.html>, accesat la 14.08.2014.
20. MathWorks, 2D Function Graphs, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-function2d.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-function2d.html), accesat la 14.08.2014.
21. MathWorks, Discontinuities of a Function, [http://www.mathworks.com/help/symbolic/mupad\\_ref/discont.html](http://www.mathworks.com/help/symbolic/mupad_ref/discont.html), accesat la 14.08.2014.
22. MathWorks, Domain of Conditionally Defined Objects, [http://www.mathworks.com/help/symbolic/mupad\\_ref/piecewise.html](http://www.mathworks.com/help/symbolic/mupad_ref/piecewise.html), accesat la 14.08.2014.
23. MathWorks, Function Plots in 2D, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plotfunc2d.html](http://www.mathworks.com/help/symbolic/mupad_ref/plotfunc2d.html), accesat la 14.08.2014.
24. MathWorks, Display Graphical Objects on the Screen, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot.html), accesat la 14.08.2014.
25. MathWorks, Numerical Approximation of an Integral, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-integral.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-integral.html), accesat la 14.08.2014.
26. MathWorks, Parametrized 2D Curves, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-curve2d.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-curve2d.html), accesat la 15.08.2014.
27. MathWorks, Contour Lines of a Function from  $\mathbb{R}^2$  to  $\mathbb{R}$ , [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-implicit2d.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-implicit2d.html), accesat la 15.08.2014.
28. MathWorks, Curves in 2D Parametrized in Polar Coordinates, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-polar.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-polar.html), accesat la 15.08.2014.
29. MathWorks, Parametrized Surfaces in 3D, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-surface.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-surface.html), accesat la 15.08.2014.
30. MathWorks, Regular Hexaedra, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-hexahedron.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-hexahedron.html), accesat la 16.08.2014.
31. MathWorks, Regular Tetrahedra, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-tetrahedron.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-tetrahedron.html), accesat la 16.08.2014.
32. MathWorks, Regular Octahedra, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-octahedron.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-octahedron.html), accesat la 16.08.2014.
33. MathWorks, Regular Icosahedra, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-icosahedron.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-icosahedron.html), accesat la 16.08.2014.
34. MathWorks, Regular Dodecahedra, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-dodecahedron.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-dodecahedron.html), accesat la 16.08.2014.
35. MathWorks, Prisms, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-prism.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-prism.html), accesat la 16.08.2014.
36. MathWorks, Pyramids and Frustums of Pyramids, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-pyramid.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-pyramid.html), accesat la 16.08.2014.
37. MathWorks, Cones and Frustums, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-cone.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-cone.html), accesat la 18.08.2014.
38. MathWorks, Cylinders, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-cylinder.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-cylinder.html), accesat la 18.08.2014.

39. MathWorks, Graphical Primitive for Spheres, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-sphere.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-sphere.html), accesat la 18.08.2014.
40. MathWorks, Graphical Primitive for Ellipsoids, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-ellipsoid.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-ellipsoid.html), accesat la 18.08.2014.
41. MathWorks, Solve Equations and Inequalities, [http://www.mathworks.com/help/symbolic/mupad\\_ref/solve.html](http://www.mathworks.com/help/symbolic/mupad_ref/solve.html), accesat la 18.08.2014.
42. MathWorks, Numerical Solution of Equations (the float attribute of solve), [http://www.mathworks.com/help/symbolic/mupad\\_ref/numeric-solve.html](http://www.mathworks.com/help/symbolic/mupad_ref/numeric-solve.html), accesat la 18.08.2014.
43. MathWorks, Operands of an Object, [http://www.mathworks.com/help/symbolic/mupad\\_ref/op.html](http://www.mathworks.com/help/symbolic/mupad_ref/op.html), accesat la 18.08.2014.
44. MathWorks, Viewer, Browser and Inspector: Interactive Manipulation, [http://www.mathworks.com/help/symbolic/mupad\\_ug/viewer-browser-and-inspector-interactive-manipulation.html](http://www.mathworks.com/help/symbolic/mupad_ug/viewer-browser-and-inspector-interactive-manipulation.html), accesat la 18.08.2014.
45. MathWorks, Display Areas where Inequalities are Fulfilled, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-inequality.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-inequality.html), accesat la 19.08.2014.
46. MathWorks, Construct a Matrix from Equations, [http://www.mathworks.com/help/symbolic/mupad\\_ref/linalg-expr2matrix.html](http://www.mathworks.com/help/symbolic/mupad_ref/linalg-expr2matrix.html), accesat la 19.08.2014.
47. MathWorks, Solving Systems of Linear Equations, [http://www.mathworks.com/help/symbolic/mupad\\_ref/linalg-matlinsolve.html](http://www.mathworks.com/help/symbolic/mupad_ref/linalg-matlinsolve.html), accesat la 19.08.2014.
48. MathWorks, Create a Matrix or a Vector, [http://www.mathworks.com/help/symbolic/mupad\\_ref/matrix.html](http://www.mathworks.com/help/symbolic/mupad_ref/matrix.html), accesat la 19.08.2014.
49. MathWorks, Domain of Ordinary Differential Equations, [http://www.mathworks.com/help/symbolic/mupad\\_ref/ode.html](http://www.mathworks.com/help/symbolic/mupad_ref/ode.html), accesat la 19.08.2014.
50. MathWorks, 2D Plots of ODE Solutions, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-ode2d.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-ode2d.html), accesat la 19.08.2014.
51. MathWorks, 3D Plots of ODE Solutions, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-ode3d.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-ode3d.html), accesat la 19.08.2014.
52. MathWorks, 2D Vector Field, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-vectorfield2d.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-vectorfield2d.html), accesat la 19.08.2014.
53. MathWorks, 3D Vector Field, [http://www.mathworks.com/help/symbolic/mupad\\_ref/plot-vectorfield3d.html](http://www.mathworks.com/help/symbolic/mupad_ref/plot-vectorfield3d.html), accesat la 19.08.2014.
54. MathWorks, Numerical Solution of an Ordinary Differential Equation, [http://www.mathworks.com/help/symbolic/mupad\\_ref/numeric-odesolve.html](http://www.mathworks.com/help/symbolic/mupad_ref/numeric-odesolve.html), accesat la 19.08.2014.
55. MathWorks, Symbolic Substitution, <http://www.mathworks.com/help/symbolic/ubs.html>, accesat la 20.08.2014.
56. MathWorks, Right Hand Side of Equations, Inequalities, Relations, Intervals, Ranges and Tables, [http://www.mathworks.com/help/symbolic/mupad\\_ref/rhs.html](http://www.mathworks.com/help/symbolic/mupad_ref/rhs.html), accesat la 20.08.2014.
57. MathWorld, A Wolfram Web Resource, Surfaces, <http://mathworld.wolfram.com/topics/Surfaces.html>, accesat la 16.08.2014.
58. SciFace Software, MuPAD, <http://www.mupad.com/>, accesat 17.05.2014.

## CUPRINS

<b>1. INTRODUCERE</b>	<b>7</b>
1.1. TEHNOLOGIA INFORMAȚIEI	7
1.2. ARHITECTURA CALCULATOARELOR PERSONALE	11
1.3. STRUCTURA CALCULATOARELOR PERSONALE	16
1.3.1. Elemente de tip hardware	16
1.3.2. Elemente de tip software	20
1.4. ELEMENTE DE ALGEBRĂ BOOLEANĂ	22
1.4.1. Operații binare și unare	22
1.4.2. Proprietățile algebrei boolene	23
1.5. SISTEME DE NUMERAȚIE	26
1.5.1. Clasificarea sistemelor de numerație	26
1.5.2. Exprimarea numerelor în diferite sisteme de numerație poziționale	27
Problema 1.1	28
1.5.3. Conversia unui număr dintr-un sistem de numerație oarecare în sistemul de numerație zecimal	28
Problema 1.2	28
1.5.4. Conversia unui număr din sistemul de numerație zecimal într-un sistem de numerație oarecare	29
Problema 1.3	30
1.5.5. Operații aritmetice în sistem binar	30
1.6. REPREZENTAREA NUMERELOR ÎN CALCULATOR	31
1.6.1. Reprezentarea numerelor naturale	31
1.6.2. Reprezentarea numerelor întregi cu semn	32
1.6.3. Reprezentarea numerelor zecimale în virgulă fixă	34
1.6.4. Reprezentarea numerelor reale în virgulă mobilă	37
Problema 1.4	37
Problema 1.5	39
Problema 1.6	39
1.7. ERORI ALE CALCULELOR NUMERICE	40
1.7.1. Clasificarea erorilor	40
1.7.2. Propagarea erorilor	42
BIBLIOGRAFIE	44
<b>2. MATLAB-PREZENTARE GENERALĂ</b>	<b>45</b>
2.1. ISTORIC. CARACTERISTICI GENERALE	45
2.2. INTERFAȚA PROGRAMULUI	50
2.3. MODUL DE LUCRU	56
2.3.1. Modul de lucru în linie de comandă	56
2.3.2. Modul de lucru pe bază de fișiere <code>script</code>	58
2.3.3. Modul de lucru bazat pe utilizarea funcțiilor	60
Problema 2.1	63
2.4. VARIABILE SCALARE. NUMERE. OPERATORI. ORDINEA OPERAȚIILOR	68
2.4.1. Declararea variabilelor	68

2.4.2.	Numere	69
2.4.3.	Operatori	71
2.5.	<b>FUNCȚII MATEMATICE ELEMENTARE</b>	72
2.5.1.	Funcții exponențiale, logaritmice și putere	72
2.5.2.	Funcții trigonometrice	73
2.5.3.	Funcții hiperbolice	73
2.5.4.	Funcții pentru manipularea numerelor complexe	74
2.5.5.	Funcții pentru aproximarea numerelor	74
	Problema 2.2	75
	Problema 2.3	76
	Problema 2.4	77
	Problema 2.5	78
	Problema 2.6	79
	<b>BIBLIOGRAFIE</b>	84
<b>3.</b>	<b>ALGORITMI ȘI SCHEME LOGICE</b>	<b>85</b>
3.1.	DEFINIȚII. PROPRIETĂȚI	85
3.2.	METODE DE REPREZENTARE A ALGORITMILOR	86
3.2.1.	Metoda schemelor logice	86
3.2.2.	Metode de tip pseudocod	87
3.3.	STRUCTURI DE CONTROL ȘI SCHEMELE LOR LOGICE	89
	Problema 3.1	91
3.4.	INSTRUCȚIUNI DE CONTROL MATLAB	93
3.4.1.	Instrucțiunea condițională <code>if</code>	93
	Problema 3.2	95
3.4.2.	Instrucțiunea repetitivă <code>for</code>	97
	Problema 3.3	99
3.4.3.	Instrucțiunea repetitivă <code>while</code>	102
	Problema 3.4	103
	Problema 3.5	105
	<b>BIBLIOGRAFIE</b>	109
<b>4.</b>	<b>VARIABLE VECTORIALE</b>	<b>110</b>
4.1.	DEFINIREA VARIABILELOR VECTORIALE	110
4.1.1.	Variabile vectoriale cu elemente neasociate	110
	Problema 4.1	111
4.1.2.	Variabile vectoriale cu elemente asociate	111
	Problema 4.2	112
	Problema 4.3	113
	Problema 4.4	114
	Problema 4.5	116
4.2.	EXTRAGEREA ELEMENTELOR UNUI VECTOR	117
	Problema 4.6	118
4.3.	OPERAȚII CU ELEMENTELE UNUI VECTOR	119
	Problema 4.7	121
4.4.	OPERAȚII ÎNTRE UN SCALAR ȘI UN VECTOR	123

Problema 4.8	123
4.5. OPERAȚII ÎNTRE DOI VECTORI	125
Problema 4.9	126
Problema 4.10	127
BIBLIOGRAFIE	130
<b>5. REPREZENTĂRI GRAFICE 2D</b>	<b>131</b>
5.1. TIPURI DE OBIECTE GRAFICE	131
5.2. TIPURI DE REPREZENTĂRI GRAFICE 2D	132
5.3. UTILIZAREA INSTRUCȚIUNII <code>plot</code>	134
5.3.1. Reprezentarea unei singure funcții	134
Problema 5.1	134
5.3.2. Reprezentarea a două funcții în ferestre grafice diferite	137
Problema 5.2	137
5.3.3. Reprezentarea a două funcții în aceeași figură și în aceeași axe	138
Problema 5.3	139
5.3.4. Reprezentarea a două funcții în aceeași figură și în axe diferite	142
5.3.5. Controlul scalării și modului de vizualizare a al axelor	145
Problema 5.4	146
5.4. UTILIZAREA INSTRUCȚIUNII <code>plotyy</code>	148
Problema 5.5	148
5.5. GRAFICE ÎN COORDONATE LOGARITMICE	151
Problema 5.6	151
5.6. GRAFICE ÎN TREPTE	153
Problema 5.7	153
5.7. GRAFICE CU BARE	155
Problema 5.8	155
5.8. REPREZENTAREA GRAFICĂ A HISTOGRAMEI	158
Problema 5.9	158
5.9. REPREZENTAREA GRAFICĂ A ERORILOR	161
Problema 5.10	161
5.10. GRAFICE DE TIP <code>area</code>	163
Problema 5.11	163
5.11. GRAFICE DE TIP <code>pie</code>	165
Problema 5.12	165
5.12. GRAFICE DE TIP <code>fill</code>	167
Problema 5.13	167
5.13. GRAFICE ÎN COORDONATE POLARE	168
Problema 5.14	168
5.14. UTILIZAREA CARACTERELOR SPECIALE	170
5.15. CREAREA ȘI EDITAREA GRAFICELOR UTILIZÂND INTERFAȚA <code>Plot Tools</code>	172
Problema 5.15	176
BIBLIOGRAFIE	182



<b>6. PROBLEME SPECIFICE DE CALCUL NUMERIC</b>	<b>183</b>
6.1. CALCULE NUMERICE CU POLINOAME	183
6.1.1. Definirea polinoamelor	183
6.1.2. Evaluarea polinoamelor	183
6.1.3. Reprezentarea grafică a polinoamelor	184
Problema 6.1	184
6.1.4. Operații cu polinoame	185
6.1.5. Rezolvarea ecuațiilor polinomiale	190
Problema 6.2	190
6.1.6. Determinarea polinomului pentru un set de soluții cunoscute	192
6.2. REZOLVAREA ECUAȚIILOR ALGEBRICE ȘI TRANSCENDENTE	193
Problema 6.3	195
6.3. REZOLVAREA SISTEMELOR DE ECUAȚII NELINIARE	199
Problema 6.4	200
6.4. CALCULUL MINIMULUI ȘI MAXIMULUI UNEI FUNCȚII	203
6.4.1. Miniumul unei funcții de o variabilă	203
6.4.2. Maximul unei funcții de o variabilă	204
Problema 6.5	205
6.5. DERIVAREA NUMERICĂ A FUNCȚIILOR	209
Problema 6.6	210
6.6. INTEGRAREA NUMERICĂ A FUNCȚIILOR	212
6.6.1. Integrarea numerică a funcțiilor de o variabilă	212
Problema 6.7	213
Problema 6.8	217
Problema 6.9	218
Problema 6.10	223
Problema 6.11	225
Problema 6.12	227
6.6.2. Calculul numeric al integralelor multiple	229
Problema 6.13	231
Problema 6.14	233
Problema 6.15	235
Problema 6.16	237
Problema 6.17	244
Problema 6.18	245
6.7. REZOLVAREA ECUAȚIILOR DIFERENȚIALE ORDINARE	249
6.7.1. Generalități	249
Problema 6.19	251
6.7.2. Rezolvarea numerică a ecuațiilor diferențiale ordinare de ordinul 1	252
Problema 6.20	255
Problema 6.21	257
Problema 6.22	259

	Problema 6.23	261
6.7.3.	Rezolvarea numerică a ecuațiilor diferențiale ordinare de ordinul 2	264
	Problema 6.24	265
	Problema 6.25	268
	Problema 6.26	273
	<b>BIBLIOGRAFIE</b>	<b>278</b>
<b>7.</b>	<b>ANALIZA DATELOR EXPERIMENTALE</b>	<b>280</b>
7.1.	ANALIZA STATISTICĂ A DATELOR EXPERIMENTALE	280
	Problema 7.1	282
7.2.	ANALIZA NUMERICĂ A DATELOR EXPERIMENTALE	287
7.2.1.	Metode de aproximare a funcțiilor	287
7.2.2.	Aproximarea prin interpolare	288
	Problema 7.2	289
7.2.3.	Aproximarea prin regresie	291
	Problema 7.3	291
7.2.4.	Aprecierea calității aproximării prin regresie	292
	Problema 7.4	295
7.3.	INTERFEȚE SPECIALIZATE PENTRU ANALIZA NUMERICĂ A DATELOR EXPERIMENTALE	298
7.3.1.	Interfața Basic Fitting	298
	Problema 7.5	299
7.3.2.	Interfața polytool	302
7.3.3.	Interfața cftool	304
	<b>BIBLIOGRAFIE</b>	<b>310</b>
<b>8.</b>	<b>VARIABLE MATRICEALE</b>	<b>311</b>
8.1.	DEFINIRE ȘI OPERAȚII SPECIFICE	311
8.1.1.	Definirea variabilelor matriceale oarecare	311
8.1.2.	Definirea variabilelor matriceale particulare	312
8.1.3.	Extragerea elementelor unei matrice	320
8.1.4.	Operații cu elementele unei matrice	321
8.1.5.	Operații între un scalar și o matrice	323
8.1.6.	Operații între două matrice	324
8.1.7.	Operații specifice analizei matriceale	325
	Problema 8.1	327
	Problema 8.2	330
8.2.	REZOLVAREA SISTEMELOR DE ECUAȚII ALGEBRICE LINIARE	333
8.2.1.	Generalități	333
8.2.2.	Analiza existenței și unicității soluției sistemului	333
8.2.3.	Rezolvarea propriu-zisă și găsirea soluțiilor	334
	Problema 8.3	337
	Problema 8.4	338
	Problema 8.5	340

8.3.	TRANSFORMĂRI GEOMETRICE AFINE	342
	Problema 8.6	345
	BIBLIOGRAFIE	349
<b>9.</b>	<b>REPREZENTĂRI GRAFICE 3D</b>	<b>350</b>
9.1.	TIPURI DE REPREZENTĂRI GRAFICE 3D	350
9.2.	UTILIZAREA INSTRUCȚIUNII <code>plot3</code>	352
	Problema 9.1	352
9.3.	UTILIZAREA INSTRUCȚIUNII <code>mesh</code>	354
	Problema 9.2	355
9.4.	UTILIZAREA INSTRUCȚIUNII <code>surf</code>	357
	Problema 9.3	358
9.5.	UTILIZAREA INSTRUCȚIUNII <code>contour</code>	360
	Problema 9.4	361
9.6.	SPECTRUL HIDRODINAMIC AL MIȘCĂRILOR POTENȚIALE	363
	Problema 9.5	364
9.7.	UTILIZAREA INSTRUCȚIUNII <code>quiver</code>	366
	Problema 9.6	366
9.8.	UTILIZAREA INSTRUCȚIUNII <code>quiver3</code>	369
	Problema 9.7	369
	Problema 9.8	373
	BIBLIOGRAFIE	382
<b>10.</b>	<b>MODELAREA ȘI SIMULAREA SISTEMELOR DINAMICE ÎN SIMULINK</b>	<b>383</b>
10.1.	GENERALITĂȚI	383
	10.1.1. Modelarea sistemelor dinamice	383
	10.1.2. Simularea sistemelor dinamice	388
10.2.	MODUL DE LUCRU ÎN SIMULINK	390
	10.2.1. Lansarea în execuție a programului Simulink	390
	10.2.2. Biblioteci de blocuri Simulink	391
	10.2.3. Crearea, salvarea, deschiderea și printarea modelelor Simulink	396
	10.2.4. Introducerea comenzilor Simulink	398
	10.2.5. Ferestrele Simulink	400
	10.2.6. Operații cu blocuri Simulink	401
	10.2.7. Introducerea textelor explicative într-o schemă bloc Simulink	405
	10.2.8. Crearea subsistemelor Simulink	406
	Problema 10.1	407
	Problema 10.2	414
	Problema 10.3	421
10.3.	SCHEME BLOC SIMULINK PENTRU REZOLVAREA ECUAȚIILOR DIFERENȚIALE ORDINARE	427
	10.3.1. Scheme bloc pentru ecuații diferențiale de ordinul 1	427

Problema 10.4	428
10.3.2. Scheme bloc pentru ecuații diferențiale de ordinul 2	431
Problema 10.5	432
10.3.3. Scheme bloc pentru sisteme de ecuații diferențiale	435
Problema 10.6	437
BIBLIOGRAFIE	439
<b>11. ELEMENTE DE CALCUL SIMBOLIC</b>	<b>440</b>
11.1. GENERALITĂȚI	440
11.2. MuPAD-PREZENTARE GENERALĂ	441
11.3. DEFINIREA ȘI MANIPULAREA EXPRESIILOR	446
11.3.1. Definierea expresiilor simbolice	446
11.3.2. Comenzile toolbar-ului Command Bar	448
11.3.3. Definierea expresiilor complexe	452
11.3.4. Definierea ipotezelor	452
11.3.5. Operații de manipulare a expresiilor simbolice	454
11.4. CALCULUL SIMBOLIC AL LIMITELOR, DERIVATELOR ȘI INTEGRALELOR	463
11.4.1. Calculul simbolic al limitelor	463
11.4.2. Calculul simbolic al derivatelor	465
11.4.3. Calculul simbolic al integralelor	467
11.5. REPREZENTAREA GRAFICĂ A EXPRESIILOR SIMBOLICE	469
Problema 11.1	471
Problema 11.2	472
Problema 11.3	475
Problema 11.4	477
Problema 11.5	480
Problema 11.6	483
Problema 11.7	485
Problema 11.8	487
Problema 11.9	490
11.6. REZOLVAREA ECUAȚIILOR ȘI SISTEMELOR DE ECUAȚII ALGEBRICE ȘI TRANSCENDENTE	493
Problema 11.10	494
Problema 11.11	497
Problema 11.12	498
Problema 11.13	499
Problema 11.14	502
11.7. REZOLVAREA ECUAȚIILOR ȘI SISTEMELOR DE ECUAȚII DIFERENȚIALE ORDINARE	505
Problema 11.15	506
Problema 11.16	508
Problema 11.17	511
BIBLIOGRAFIE	515
<b>CUPRINS</b>	<b>518</b>