```
> ec2:=proc(a,b,c)
    local delta,x1,x2;
    description ‘Rezolvarea ecuatiei de gradul 2‘;
    delta:=b^2-4*a*c;
    if delta>0 then
      x1:=(-b+sqrt(delta))/(2*a);
      x2:=(-b-sqrt(delta))/(2*a);
      RETURN(x1,x2);
    elif delta=0 then RETURN(-b/(2*a));
    else
      RETURN(‘ecuatia nu ere solutii reale‘);
    fi;
  end:
```

care produce următoarele rezultate:

```
> ec2(1,6,9); # ecuatia x^2+6*x+9=0
```

-3

```
> ec2(1,2,9); # ecuatia x^2+2*x+9=0
```

ecuatia nu are solutii reale

```
> ec2(1,2,-3); # ecuatia x^2+2*x-3=0
```

$1, -3$

Pentru a defini tipul unui argument, se foloseşte sintaxa `argument::tip`. De exemplu, să luăm următoarea procedură şi situaţiile care pot apărea:

```
> # procedura care returneaza determinantul unei matrice
> determinant:=proc(A) RETURN(det(A)) end:
> determinant(2);
Error, (in linalg:-det) expecting a matrix
```

Procedura `determinant` se poate ”imbunătăţi” astfel:

```
> determinant1:=proc(A)
    if not type(A, matrix)
      then ERROR(‘argumentul trebuie sa fie matrice!!!‘)
    fi;
    RETURN(det(A))
    end:
```

care produce următorul rezultat:

```
> determinant1(2);
Error, (in determinant1) argumentul trebuie sa fie matrice!!!
```

Se mai poate defini argumentul `A` ca fiind de tipul matrice:

```
> determinant3:=proc(A::matrix) RETURN(det(A)) end:
```

şi se obţine următorul rezultat:

```
> determinant3(2);
Error, invalid input: determinant3 expects its 1st argument, A,
 to be of type matrix, but received 2
```

Mai multe detalii despre tipurile existente se pot găsi accesând pagina de help (cuvântul cheie este `type`).

Un alt exemplu este procedura `rdc`, procedură pentru calculul lui $\frac{1}{\sqrt{x}}$:

```
> rdc:=proc(x)
    if x<0 then ERROR(‘numar negativ!‘)
    elif x=0 then RETURN(infinity)
    else simplify(x^(-1/2));
    fi;
  end;
```

$$rdc := \textbf{proc}(x)\ \textbf{if}\ x < 0\ \textbf{then}\ \text{ERROR}(\text{‘}numar\ negativ!\text{‘})$$
$$\textbf{elif}\ x = 0\ \textbf{then}\ \text{RETURN}(\infty)$$
$$\textbf{else}\ \text{simplify}(1/(\text{x}\hat{}\ (1/2)))\ \textbf{end if end proc}$$

```
> rdc(-1);
Error, (in rdc) numar negativ!
> rdc(0);
```
$$\infty$$
```
> rdc(4);
```
$$\frac{1}{2}$$

Pentru a putea urmări execuţia unei proceduri, se foloseşte `debug`, iar pentru a stopa urmărirea, se foloseşte `undebug`. De exemplu, putem avea:

```
> f:=proc(a,b)
      local y,z;
      y:=a+b/2;
      z:=1/y;
      RETURN(y+z)
    end;
```

$$f := \textbf{proc}(a, b)$$
$$\textbf{local}\ y, z;$$
$$y := a + 1/2 * b;\ z := 1/y$$
$$\text{RETURN}(y + z)\ \textbf{end proc}$$

```
> debug(f);
```
$$f$$
```
> f(2,4);
```

```
{--> enter f, args = 2, 4
```
$$y := 4$$
$$z := \frac{1}{4}$$
```
<-- exit f (now at top level) = 17/4}
```
$$\frac{17}{4}$$
```
> f(0,1);
{--> enter f, args = 0, 1
```
$$y := \frac{1}{2}$$
$$z := 2$$
```
<-- exit f (now at top level) = 5/2}
```
$$\frac{5}{2}$$
```
undebug(f)
```
$$f$$
```
> f(10,20);
```
$$\frac{401}{20}$$

Alte detalii despre funcţii şi proceduri, precum şi despre opţiunile `debug` şi `undebug`, puteţi găsi pe paginile de help referitoare la acestea.

Capitolul 2

Rezolvarea sistemelor liniare

În acest capitol vom prezenta metode de rezolvare a sistemelor liniare de tip Cramer (numărul de ecuaţii este egal cu numărul de necunoscute, şi determinantul matricei sistemului este nenul):

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2 \\ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n = b_n \end{cases} \tag{2.1}$$

în care a_{ij} şi b_i sunt numere reale date, $i = 1 \ldots n, j = 1 \ldots n$, iar $x_1, x_2, \ldots, x_n$ sunt numere reale necunoscute.

Sistemul (2.1) se poate scrie matriceal sub forma:

$$Ax = b$$

unde: $A = (a_{ij})_{i,j=\overline{1,n}}$, $b = (b_1, b_2, \ldots, b_n)^T$, $x = (x_1, x_2, \ldots, x_n)^T$.

Dacă matricea A este nesingulară, sistemul $Ax = b$ are soluţie unică:

$$x = A^{-1}b.$$

Deoarece în cele mai multe cazuri matricea A are număr mare de linii şi coloane, iar calculul matricei A^{-1} este dificil şi acumulează erori, se impun metode directe şi metode iterative pentru rezolvarea acestor sisteme.

2.1 Metoda lui Gauss

2.1.1 Breviar teoretic

Metoda lui Gauss presupune transformarea sistemului $Ax = b$ într-un sistem superior triunghiular, şi apoi rezolvarea acestuia prin substituţie inversă.

Construcţia sistemului superior triunghiular se face astfel: la pasul k se elimină x_k din ecuaţiile $k + 1$, ..., n, prin înmulţirea ecuaţiei k cu

$$m_{ik} = -\frac{a_{ik}}{a_{kk}}$$

(elementul a_{kk} se numeşte pivot) şi adunarea acestora la ecuaţia i $(i > k)$.

În funcţie de alegerea pivotului, există următoarele **variante ale metodei lui Gauss**:

1. **metoda lui Gauss clasică** - în care la fiecare pas, pivotul este elementul a_{kk}, $k = \overline{1,n}$;

2. **metoda lui Gauss cu semipivot** - în care la fiecare pas, se alege ca pivot elementul a_{ik} maxim în valoare absolută pe coloană, pentru $i > k$, permutându-se linia k cu linia i;

3. **metoda lui Gauss cu pivot total** - în care la fiecare pas, se alege ca pivot elementul maxim atât pe linie, cât şi pe coloană, pentru $i > k$, $j > k$, permutându-se linia k cu linia i şi coloana k cu coloana j;

În acest fel, sistemul (2.1) se reduce la forma superior triunghiulară

$$\begin{pmatrix} \tilde{a}_{11} & \tilde{a}_{12} & ... & \tilde{a}_{1,n-2} & \tilde{a}_{1,n-1} & \tilde{a}_{1,n} \\ 0 & \tilde{a}_{22} & ... & \tilde{a}_{2,n-2} & \tilde{a}_{2,n-1} & \tilde{a}_{2,n} \\ ... & ... & ... & ... & ... & ... \\ 0 & 0 & ... & 0 & \tilde{a}_{n-1,n-1} & \tilde{a}_{n-1,n} \\ 0 & 0 & ... & 0 & 0 & \tilde{a}_{nn} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ ... \\ x_{n-1} \\ x_n \end{pmatrix} = \begin{pmatrix} \tilde{b}_1 \\ \tilde{b}_2 \\ ... \\ \tilde{b}_{n-1} \\ \tilde{b}_n \end{pmatrix} \tag{2.2}$$

iar rezolvarea sistemului (2.2) se face prin substituţie inversă:

$$\begin{aligned} x_n &= \frac{\tilde{b}_n}{\tilde{a}_{nn}} \\ x_k &= \left(\tilde{b}_k - \sum_{j=k+1}^{n} \tilde{a}_{kj} \cdot x_j \right) \cdot \frac{1}{\tilde{a}_{kk}},\ k = n-1, n-2, \ldots, 1 \end{aligned} \tag{2.3}$$

Observaţia 2.1.1. Cu ajutorul eliminării gaussiene se poate determina şi inversa unei matrice. Redăm în continuare algoritmul de aflare a inversei unei matrice A.

1. generarea matricei B prin concatenarea matricelor A (de dimensiune n) cu matricea I_n

2. pentru $i = \overline{1,n}$

   $m = B_{ii}$
   pentru $j = \overline{1,2n}$
      $B_{ij} = \dfrac{B_{ij}}{m}$
   pentru $j = \overline{1,n}$, $j \neq i$
      $m_1 = B_{ji}$
      pentru $k = \overline{1,2n}$
         $B_{jk} = B_{jk} - m_1 \, B_{ik}$

3. prin ştergerea primelor n coloane ale matricei B astfel transformate, se obţine inversa matricei A

2.1.2 Probleme rezolvate

Exerciţiul 2.1.1. Să se rezolve următorul sistem folosind cele trei variante ale eliminării Gauss:

$$\begin{cases} x+y+z=6 \\ 2x-y+3z=9 \\ x+4y+z=12. \end{cases}$$

Matricea sistemului este

$$A=\begin{pmatrix} 1 & 1 & 1 \\ 2 & -1 & 3 \\ 1 & 4 & 1 \end{pmatrix},$$

iar $\bar{A}$ este matricea sa extinsă:

$$\bar{A}=(A,b)=\begin{pmatrix} 1 & 1 & 1 & 6 \\ 2 & -1 & 3 & 9 \\ 1 & 4 & 1 & 12 \end{pmatrix}.$$

Deoarece numărul ecuaţiilor este egal cu cel al necunoscutelor şi

$$\det A=-3\neq 0,$$

sistemul este compatibil determinat (de tip Cramer), şi deci metoda eliminării a lui Gauss este aplicabilă.

În continuare, pentru a efectua operaţiile asupra matricei extinse a sistemului vom nota linia i cu L_i, iar coloana j cu C_j.

Rezolvare utilizând metoda lui Gauss clasică

A. Construcţia sistemului superior triunghiular

Pasul 1

- pivot: $a_{11}=1$
- $m_{21}=-\dfrac{2}{1}=-2$
- $m_{31}=-\dfrac{1}{1}=-1$

$$\begin{pmatrix} 1 & 1 & 1 & 6 \\ 2 & -1 & 3 & 9 \\ 1 & 4 & 1 & 12 \end{pmatrix} \xrightarrow[L_3\to L_3+m_{31}L_1]{L_2\to L_2+m_{21}L_1} \begin{pmatrix} 1 & 1 & 1 & 6 \\ 0 & -3 & 1 & -3 \\ 0 & 3 & 0 & 6 \end{pmatrix}$$

Pasul 2

- pivot: $a_{22}=-3$
- $m_{32}=-\dfrac{3}{-3}=1$

$$\begin{pmatrix} 1 & 1 & 1 & 6 \\ 0 & -3 & 1 & -3 \\ 0 & 3 & 0 & 6 \end{pmatrix} \xrightarrow{L_3\to L_3+m_{32}L_2} \begin{pmatrix} 1 & 1 & 1 & 6 \\ 0 & -3 & 1 & -3 \\ 0 & 0 & 1 & 3 \end{pmatrix}$$

În acest moment am ajuns la un sistem de forma $\tilde{A}\mathrm{x} = \tilde{b}$, echivalent cu sistemul iniţial, în care matricea $\tilde{A}$ este superior triunghiulară, unde:

$$\tilde{A} = \begin{pmatrix} 1 & 1 & 1 \\ 0 & -3 & 1 \\ 0 & 0 & 1 \end{pmatrix}, \qquad \mathrm{x} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}, \qquad \tilde{b} = \begin{pmatrix} 6 \\ -3 \\ 3 \end{pmatrix}.$$

B. Rezolvarea sistemului superior triunghiular
Prin metoda substituţiei inverse, avem:

$$\begin{aligned} z &= \frac{3}{1} \\ y &= \frac{1}{-3}(-3 - 1 \cdot z) \\ x &= \frac{1}{1}(6 - 1 \cdot y - 1 \cdot z), \end{aligned}$$

de unde obţinem soluţia sistemului: $x = 1$, $y = 2$, $z = 3$.

Rezolvare cu metoda lui Gauss cu semipivot

A. Construcţia sistemului superior triunghiular

Pasul 1

• Ca pivot se ia elementul a_{i1} de modul maxim de pe coloana 1. În cazul nostru, pivotul este a_{12}, deci se permută linia 1 cu linia 2, şi se fac zerouri pe coloana 1 pentru $i > 1$:

$$\begin{pmatrix} 1 & 1 & 1 & 6 \\ 2 & -1 & 3 & 9 \\ 1 & 4 & 1 & 12 \end{pmatrix} \xrightarrow{L_2 \leftrightarrow L_1} \begin{pmatrix} 2 & -1 & 3 & 9 \\ 1 & 1 & 1 & 6 \\ 1 & 4 & 1 & 12 \end{pmatrix}$$

$$\begin{pmatrix} 2 & -1 & 3 & 9 \\ 1 & 1 & 1 & 6 \\ 1 & 4 & 1 & 12 \end{pmatrix} \xrightarrow[L_3 \to L_3 - \frac{1}{2}L_1]{L_2 \to L_2 - \frac{1}{2}L_1} \begin{pmatrix} 2 & -1 & 3 & 9 \\ 0 & \frac{3}{2} & -\frac{1}{2} & \frac{3}{2} \\ 0 & \frac{9}{2} & -\frac{1}{2} & \frac{15}{2} \end{pmatrix}$$

Pasul 2

• Ca pivot se ia elementul a_{i2} de modul maxim de pe coloana 2, pentru $i \geq 2$. În cazul nostru, pivotul este a_{32}, deci se permută linia 2 cu linia 3 şi se fac zerouri pe coloana 2, pentru $i > 2$:

$$\begin{pmatrix} 2 & -1 & 3 & 9 \\ 0 & \frac{3}{2} & -\frac{1}{2} & \frac{3}{2} \\ 0 & \frac{9}{2} & -\frac{1}{2} & \frac{15}{2} \end{pmatrix} \xrightarrow{L_3 \leftrightarrow L_2} \begin{pmatrix} 2 & -1 & 3 & 9 \\ 0 & \frac{9}{2} & -\frac{1}{2} & \frac{15}{2} \\ 0 & \frac{3}{2} & -\frac{1}{2} & \frac{3}{2} \end{pmatrix}$$

$$\begin{pmatrix} 2 & -1 & 3 & 9 \\ 0 & \frac{9}{2} & -\frac{1}{2} & \frac{15}{2} \\ 0 & \frac{3}{2} & -\frac{1}{2} & \frac{3}{2} \end{pmatrix} \xrightarrow{L_3 \to L_3 - \frac{1}{3}L_2} \begin{pmatrix} 2 & -1 & 3 & 9 \\ 0 & \frac{9}{2} & -\frac{1}{2} & \frac{15}{2} \\ 0 & 0 & -\frac{1}{3} & -1 \end{pmatrix}$$

În acest moment am ajuns la un sistem de forma $\tilde{A}\mathrm{x} = \tilde{b}$, echivalent cu sistemul iniţial, unde matricea $\tilde{A}$ este superior triunghiulară, iar:

$$\tilde{A} = \begin{pmatrix} 2 & -1 & 3 \\ 0 & \frac{9}{2} & -\frac{1}{2} \\ 0 & 0 & -\frac{1}{3} \end{pmatrix}, \qquad \mathrm{x} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}, \qquad \tilde{b} = \begin{pmatrix} 9 \\ \frac{15}{2} \\ -1 \end{pmatrix}.$$

B. Rezolvarea sistemului superior triunghiular se face ca şi în cazul metodei lui Gauss clasice, şi conduce la soluţia $x = 1$, $y = 2$, $z = 3$.

Rezolvare cu metoda lui Gauss cu pivot total

A. Construcţia sistemului superior triunghiular

Pasul 1

• ca pivot se alege elementul a_{ij} de modul maxim pentru $i, j \geq 1$. În cazul nostru pivotul este a_{32}, deci se permută linia 3 cu linia 1, şi coloana 2 cu coloana 1:

$$\begin{pmatrix} 1 & 1 & 1 & 6 \\ 2 & -1 & 3 & 9 \\ 1 & 4 & 1 & 12 \end{pmatrix} \xrightarrow{L_3 \leftrightarrow L_1} \begin{pmatrix} 1 & 4 & 1 & 12 \\ 2 & -1 & 3 & 9 \\ 1 & 1 & 1 & 6 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 4 & 1 & 12 \\ 2 & -1 & 3 & 9 \\ 1 & 1 & 1 & 6 \end{pmatrix} \xrightarrow{C_2 \leftrightarrow C_1} \begin{pmatrix} 4 & 1 & 1 & 12 \\ -1 & 2 & 3 & 9 \\ 1 & 1 & 1 & 6 \end{pmatrix}$$

• pentru corectitudinea rezultatului final este necesar ca, ori de câte ori se permută coloanele matricei extinse, să se permute şi elementele corespunzătoare ale vectorului x. Astfel, avem:

$$\mathrm{x} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \xrightarrow{\mathrm{x}_2 \leftrightarrow \mathrm{x}_1} \begin{pmatrix} y \\ x \\ z \end{pmatrix}$$

• în final, obţinem:

$$\begin{pmatrix} 4 & 1 & 1 & 12 \\ -1 & 2 & 3 & 9 \\ 1 & 1 & 1 & 6 \end{pmatrix} \xrightarrow[L_3 \to L_3 - \frac{1}{4}L_1]{L_2 \to L_2 + \frac{1}{4}L_1} \begin{pmatrix} 4 & 1 & 1 & 12 \\ 0 & \frac{9}{4} & \frac{13}{4} & 12 \\ 0 & \frac{3}{4} & \frac{3}{4} & 3 \end{pmatrix}$$

Pasul 2

• ca pivot se alege elementul a_{ij} de modul maxim pentru $i, j \geq 2$. Deoarece pivotul este a_{23}, se permută coloana 3 cu coloana 2:

$$\begin{pmatrix} 4 & 1 & 1 & 12 \\ 0 & \frac{9}{4} & \frac{13}{4} & 12 \\ 0 & \frac{3}{4} & \frac{3}{4} & 3 \end{pmatrix} \xrightarrow{C_3 \leftrightarrow C_2} \begin{pmatrix} 4 & 1 & 1 & 12 \\ 0 & \frac{13}{4} & \frac{9}{4} & 12 \\ 0 & \frac{3}{4} & \frac{3}{4} & 3 \end{pmatrix}$$

$$l := [x, y, z]$$

$$n := 3$$

$$A := \begin{bmatrix} 1 & 1 & 1 & 6 \\ 2 & -1 & 3 & 9 \\ 1 & 4 & 1 & 12 \end{bmatrix}$$

```
{--> enter cgauss, args = A
```

$$n := 3$$

$$A1 := A$$

$$A2 := \begin{bmatrix} 1 & 1 & 1 \\ 2 & -1 & 3 \\ 1 & 4 & 1 \end{bmatrix}$$

$$m := 2$$

$$A1_{2,1} := 0$$

$$A1_{2,2} := -3$$

$$A1_{2,3} := 1$$

$$A1_{2,4} := -3$$

$$m := 1$$

$$A1_{3,1} := 0$$

$$A1_{3,2} := 3$$

$$A1_{3,3} := 0$$

$$A1_{3,4} := 6$$

$$m := -1$$

$$A1_{3,2} := 0$$

$$A1_{3,3} := 1$$

$$A1_{3,4} := 3$$

```
<-- exit cgauss (now in gauss) = array(1 .. 3, 1 .. 4,[(3, 3)=1,(2,
1)=0,(3, 2)=0,(1, 3)=1,(3, 1)=0,(1, 4)=6,(2, 2)=-3,(2, 3)=1,(3,
4)=3,(1, 2)=1,(1, 1)=1,(2, 4)=-3])}
```

$$A1 := \begin{bmatrix} 1 & 1 & 1 & 6 \\ 0 & -3 & 1 & -3 \\ 0 & 0 & 1 & 3 \end{bmatrix}$$

$$x_3 := 3$$

$$s := 0$$

$$s := 3$$

$$x_2 := 2$$

$$s := 0$$

$$s := 2$$

$$s := 5$$

$$x_1 := 1$$

```
<-- exit gauss (now at top level) = x = 1, y = 2, z = 3}
```

$$x = 1, y = 2, z = 3$$

```
>  gauss({x+y+z=6,2*x-y+3*z=9,x+4*y+z=12},semipivot);
```

```
{--> enter gauss, args = {x+y+z = 6, 2*x-y+3*z = 9, x+4*y+z = 12},
semipivot
```

$$l := [x,\ y,\ z]$$

$$n := 3$$

$$A := \begin{bmatrix} 1 & 1 & 1 & 6 \\ 2 & -1 & 3 & 9 \\ 1 & 4 & 1 & 12 \end{bmatrix}$$

```
{--> enter spgauss, args = A
```

$$n := 3$$

$$A1 := A$$

$$A2 := \begin{bmatrix} 1 & 1 & 1 \\ 2 & -1 & 3 \\ 1 & 4 & 1 \end{bmatrix}$$

$$mx := 1$$

$$mx := 2$$

$$A1 := \begin{bmatrix} 2 & -1 & 3 & 9 \\ 1 & 1 & 1 & 6 \\ 1 & 4 & 1 & 12 \end{bmatrix}$$

$$m := \frac{1}{2}$$

$$A1_{2,\,1} := 0$$

$$A1_{2,\,2} := \frac{3}{2}$$

$$A1_{2,\,3} := \frac{-1}{2}$$

$$A1_{2,\,4} := \frac{3}{2}$$

$$m := \frac{1}{2}$$

$$A1_{3,\,1} := 0$$

$$A1_{3,\,2} := \frac{9}{2}$$

$$A1_{3,\,3} := \frac{-1}{2}$$

$$A1_{3,\,4} := \frac{15}{2}$$

$$mx := 2$$

$$mx := 3$$

$$A1 := \begin{bmatrix} 2 & -1 & 3 & 9 \\ 0 & \frac{9}{2} & \frac{-1}{2} & \frac{15}{2} \\ 0 & \frac{3}{2} & \frac{-1}{2} & \frac{3}{2} \end{bmatrix}$$

$$m := \frac{1}{3}$$

$$A1_{3,\,2} := 0$$

$$A1_{3,3} := \frac{-1}{3}$$
$$A1_{3,4} := -1$$

```
<-- exit spgauss (now in gauss) = array(1 .. 3, 1 .. 4,[(3,
4)=-1,(2, 3)=-1/2,(1, 4)=9,(1, 1)=2,(3, 1)=0,(2, 1)=0,(1, 3)=3,(2,
4)=15/2,(3, 2)=0,(1, 2)=-1,(3, 3)=-1/3,(2, 2)=9/2])}
```

$$A1 := \begin{bmatrix} 2 & -1 & 3 & 9 \\ 0 & \frac{9}{2} & \frac{-1}{2} & \frac{15}{2} \\ 0 & 0 & \frac{-1}{3} & -1 \end{bmatrix}$$
$$x_3 := 3$$
$$s := 0$$
$$s := \frac{-3}{2}$$
$$x_2 := 2$$
$$s := 0$$
$$s := -2$$
$$s := 7$$
$$x_1 := 1$$

```
<-- exit gauss (now at top level) = x = 1, y = 2, z = 3}
```

$$x = 1,\ y = 2,\ z = 3$$

Observaţia 2.1.4. Pachetul `linalg` furnizează procedurile `gausselim` şi `backsub`. Astfel, procedura `gausselim` efectuează eliminarea gaussiană cu pivot parţial asupra unei matrice $n \times m$. Procedura `backsub` ia ca argument rezultatul procedurii `gausselim` si furnizează soluţia sistemului. Astfel, pentru matricea din exemplul precedent, avem:

```
> A := matrix([[1, 1, 1, 6], [2, -1, 3, 9], [1, 4, 1, 12]]);
```

$$A := \begin{bmatrix} 1 & 1 & 1 & 6 \\ 2 & -1 & 3 & 9 \\ 1 & 4 & 1 & 12 \end{bmatrix}$$

```
> gausselim(A);
```

$$\begin{bmatrix} 1 & 1 & 1 & 6 \\ 0 & -3 & 1 & -3 \\ 0 & 0 & 1 & 3 \end{bmatrix}$$

```
> backsub(%);
```

$$[1,\ 2,\ 3]$$

2.2 Factorizarea LU

2.2.1 Breviar teoretic

Fie sistemul compatibil determinat

$$Ax = b. \tag{2.4}$$

Factorizarea LU presupune descompunerea matricei A într-un produs de matrice $L \cdot U$, unde

$$L = \begin{pmatrix} \lambda_{11} & 0 & \dots & 0 \\ \lambda_{21} & \lambda_{22} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ \lambda_{n1} & \lambda_{n2} & \dots & \lambda_{nn} \end{pmatrix} \qquad U = \begin{pmatrix} \mu_{11} & \mu_{12} & \dots & \mu_{1n} \\ 0 & \mu_{22} & \dots & \mu_{2n} \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \mu_{nn} \end{pmatrix}. \tag{2.5}$$

Această descompunere este posibilă dacă toţi determinanţii de colţ ai matricei A sunt nenuli.

Pentru a asigura unicitatea descompunerii, trebuie precizate n elemente ale matricei L sau U. În mod tradiţional, se specifică λ_{ii} sau μ_{ii}; dacă $\lambda_{ii} = 1$ atunci factorizarea LU se numeşte **factorizare Doolittle**, iar dacă $\mu_{ii} = 1$ se numeşte **factorizare Crout**.

Astfel, rezolvarea sistemului (2.4) se reduce la rezolvarea sistemelor triunghiulare

$$Ly = b \tag{2.6}$$

cu soluţia

$$\begin{cases} y_1 = \dfrac{b_1}{\lambda_{11}} \\ y_i = \left(b_i - \displaystyle\sum_{j=1}^{i-1} \lambda_{ij} y_j \right) \cdot \dfrac{1}{\lambda_{ii}}, \ i = 2, 3, \dots, n \end{cases} \tag{2.7}$$

şi

$$Ux = y \tag{2.8}$$

cu soluţia

$$\begin{cases} x_n = \dfrac{y_n}{\mu_{nn}} \\ x_i = \left(y_i - \displaystyle\sum_{j=i+1}^{n} \mu_{ij} x_j \right) \cdot \dfrac{1}{\mu_{ii}}, \ i = 2, 3, \dots, n. \end{cases} \tag{2.9}$$

2.2.2 Problemă rezolvată

Exerciţiul 2.2.1. Să se determine soluţia sistemului următor, folosind factorizarea LU:

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

Sistemul se scrie în forma matriceală:

$$A\mathrm{x} = b,$$

unde

$$A = \begin{pmatrix} 1 & 1 & -1 \\ 2 & -1 & 1 \\ 1 & 3 & -2 \end{pmatrix}, \quad \mathrm{x} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}, \quad b = \begin{pmatrix} 2 \\ 1 \\ 5 \end{pmatrix}.$$

Deoarece

$$1 \neq 0\,, \ \begin{vmatrix} 1 & 1 \\ 2 & -1 \end{vmatrix} = -3 \neq 0\,, \ \begin{vmatrix} 1 & 1 & -1 \\ 2 & -1 & 1 \\ 1 & 3 & -2 \end{vmatrix} = -3 \neq 0\,,$$

```
n:=nops(l);
for i from 1 to n do
ops[i]:=[seq(op(op(l[i])[1])[j] /
         coeff(op(l[i])[1],op(indets(op(op(l[i] )[1])[j]))),
         j=1..nops(op(l[i])[1]))];
od;
opst:=ops[1];
for i from 1 to n do
  for j from 1 to nops(ops[i]) do
    if not(ops[i][j] in opst) then
      opst:=[op(opst),ops[i][j]]
    fi;
  od;
od;
RETURN(opst);
end:

tridiagonalsist:=proc(l::set(equation))
local eqm, opst, A, b, n, lu, L, U, i, s, j, aux, rez;
n:=nops(l);
opst:=nedeterminate(l);
eqm:=genmatrix(l, opst, flag);
A:=delcols(eqm,n+1..n+1);
b:=col(eqm,n+1);
lu:=tridiagonal(A);
L:=lu[1];
U:=lu[2];
aux[1]:=b[1]/L[1,1];
for i from 2 to n do
  aux[i]:=1/L[i,i]*(b[i]-L[i,i-1]*aux[i-1])
od;
rez[n]:=aux[n];
for i from n-1 by -1 to 1 do
  rez[i]:=aux[i]-U[i,i+1]*rez[i+1];
od;
RETURN(seq(opst[i]=rez[i], i=1..n));
end:

debug(tridiagonalsist):

tridiagonalsist({x+2*y=3,2*x-y+z=2, 3*y+2*z-t=4, -2*z+t=-1});
```

```
{--> enter tridiagonalsist, args = {x+2*y = 3, 2*x-y+z = 2, 3*y+2*z-t
= 4, -2*z+t = -1}
```

$$n := 4$$

$$opst := [x,\ y,\ z,\ t]$$

$$eqm := \begin{bmatrix} 1 & 2 & 0 & 0 & 3 \\ 2 & -1 & 1 & 0 & 2 \\ 0 & 3 & 2 & -1 & 4 \\ 0 & 0 & -2 & 1 & -1 \end{bmatrix}$$

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

$$b := [3,\ 2,\ 4,\ -1]$$

$$lu := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 2 & -5 & 0 & 0 \\ 0 & 3 & \frac{13}{5} & 0 \\ 0 & 0 & -2 & \frac{3}{13} \end{bmatrix}, \begin{bmatrix} 1 & 2 & 0 & 0 \\ 0 & 1 & \frac{-1}{5} & 0 \\ 0 & 0 & 1 & \frac{-5}{13} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$L := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 2 & -5 & 0 & 0 \\ 0 & 3 & \frac{13}{5} & 0 \\ 0 & 0 & -2 & \frac{3}{13} \end{bmatrix}$$

$$U := \begin{bmatrix} 1 & 2 & 0 & 0 \\ 0 & 1 & \frac{-1}{5} & 0 \\ 0 & 0 & 1 & \frac{-5}{13} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$aux_1 := 3$$

$$aux_2 := \frac{4}{5}$$

$$aux_3 := \frac{8}{13}$$

$$aux_4 := 1$$

$$rez_4 := 1$$

$$rez_3 := 1$$

$$rez_2 := 1$$

$$rez_1 := 1$$

```
<-- exit tridiagonalsist (now at top level) = x = 1, y = 1, z = 1, t =
1}
```

$$x = 1,\ y = 1,\ z = 1,\ t = 1$$

2.4 Factorizarea Cholesky

2.4.1 Breviar teoretic

Un caz particular de sisteme liniare este acela în care matricea A a sistemului este simetrică şi pozitiv definită (adică toţi determinanţii de colţ sunt strict pozitivi). Pentru astfel de sisteme putem folosi un caz particular al factorizării LU: descompunem matricea A a sistemului într-un produs $L \cdot L^T$, unde

$$L = \begin{pmatrix} \lambda_{11} & 0 & \dots & 0 \\ \lambda_{21} & \lambda_{22} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ \lambda_{n1} & \lambda_{n2} & \dots & \lambda_{nn} \end{pmatrix}. \tag{2.14}$$

Coeficienţii λ_{ij} se obţin din definiţia produsului a două matrice.

2.4.2 Problemă rezolvată

Exerciţiul 2.4.1. Să se rezolve sistemul:

$$\begin{cases} x + 2y + z = 5 \\ 2x + 5y + 2z = 11 \\ x + 2y + 3z = 7. \end{cases}$$

Rezolvare

A. Factorizarea Cholesky

Matricea sistemului este

$$A = \begin{pmatrix} 1 & 2 & 1 \\ 2 & 5 & 2 \\ 1 & 2 & 3 \end{pmatrix}$$

Se observă că $a_{ij} = a_{ji}$, adică matricea A este simetrică. Deoarece

$$1 > 0\,, \quad \begin{vmatrix} 1 & 2 \\ 2 & 5 \end{vmatrix} = 1 > 0\,, \quad \begin{vmatrix} 1 & 2 & 1 \\ 2 & 5 & 2 \\ 1 & 2 & 3 \end{vmatrix} = 2 > 0\,,$$

matricea A este pozitiv definită. Aplicând factorizarea Cholesky avem:

$$A = \begin{pmatrix} 1 & 2 & 1 \\ 2 & 5 & 2 \\ 1 & 2 & 3 \end{pmatrix} = \begin{pmatrix} \lambda_{11} & 0 & 0 \\ \lambda_{21} & \lambda_{22} & 0 \\ \lambda_{31} & \lambda_{32} & \lambda_{33} \end{pmatrix} \cdot \begin{pmatrix} \lambda_{11} & \lambda_{21} & \lambda_{31} \\ 0 & \lambda_{22} & \lambda_{32} \\ 0 & 0 & \lambda_{33} \end{pmatrix}.$$

Folosind definiţia produsului a două matrice, obţinem:

$$\begin{aligned} a_{11} &= \lambda_{11}^2 && \Rightarrow \lambda_{11} = 1 \\ a_{12} &= \lambda_{11} \cdot \lambda_{21} && \Rightarrow \lambda_{21} = 2 \\ a_{13} &= \lambda_{11} \cdot \lambda_{31} && \Rightarrow \lambda_{31} = 1 \\ a_{22} &= \lambda_{21}^2 + \lambda_{22}^2 && \Rightarrow \lambda_{22} = 1 \\ a_{23} &= \lambda_{21} \cdot \lambda_{31} + \lambda_{22} \cdot \lambda_{32} && \Rightarrow \lambda_{32} = 0 \\ a_{33} &= \lambda_{31}^2 + \lambda_{32}^2 + \lambda_{33}^2 && \Rightarrow \lambda_{33} = \sqrt{2}. \end{aligned}$$

Se observă că pentru găsirea elementelor λ_{ij}, $i = \overline{1,n}$, $j = \overline{i,n}$, este suficient să calculăm dezvoltările corespunzătoare elementelor a_{ij}, $i = \overline{1,n}$, $j = \overline{i,n}$.

B. Rezolvarea sistemelor triunghiulare

Pentru a determina soluţia sistemului iniţial, avem de rezolvat două sisteme triunghiulare:

$$\begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 1 & 0 & \sqrt{2} \end{pmatrix} \cdot \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 5 \\ 11 \\ 7 \end{pmatrix},$$

cu soluţia

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 5 \\ 1 \\ \sqrt{2} \end{pmatrix},$$

şi

$$\begin{pmatrix} 1 & 2 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & \sqrt{2} \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 5 \\ 1 \\ \sqrt{2} \end{pmatrix},$$

cu soluţia

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \\ 1 \end{pmatrix}.$$

2.4.3 Probleme propuse

Exerciţiul 2.4.2. Să se găsească soluţia sistemului următor, folosind factorizarea Cholesky:

$$\begin{cases} 3x + y + 3z = 11 \\ x + y + 2z = 6 \\ 3x + y + 4z = 12. \end{cases}$$

2.4.4 Implementare

A. Algoritm

Date de intrare: un sistem de ecuaţii

Date de ieşire: soluţia sistemului

Algoritm

1. generarea matricei A a sistemului (simetrică şi pozitiv definită) şi a vectorului b
   - n = numărul de ecuaţii ale sistemului (numărul de linii ale matricei A)
2. factorizarea Cholesky

   $\lambda_{11} = a_{11}$

   pentru $i = \overline{2,n}$

   pentru $j = \overline{1,i-1}$

   $$\lambda_{ij} = \frac{1}{\lambda_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} \lambda_{ik}\lambda_{jk} \right)$$

$$\lambda_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} \lambda_{ik}^2}$$

3. rezolvarea sistemelor triunghiulare

$y_1 = \dfrac{b_1}{\lambda_{11}}$

pentru $i = \overline{2, n}$

$$y_i = \left(b_i - \sum_{k=1}^{i-1} \lambda_{ik} \cdot y_k \right) \cdot \frac{1}{\lambda_{ii}}$$

$x_n = \dfrac{y_n}{\lambda_{nn}}$

pentru $i = \overline{n-1, 1}$

$$x_i = \left(y_i - \sum_{k=i+1}^{n} \lambda_{ki} \cdot x_k \right) \cdot \frac{1}{\lambda_{ii}}$$

B. Programe MAPLE şi rezultate

2.5 Factorizarea Householder

2.5.1 Breviar teoretic

Factorizarea Householder este o metodă de rezolvare numerică a sistemelor de tip Cramer simetrice, şi constă în determinarea unei matrice simetrice nesingulare U, astfel încât $UAU = T$ să fie o matrice tridiagonală. Atunci soluţia sistemului $Ax = b$ este dată de

$$x = Uy, \tag{2.15}$$

unde y este soluţia sistemului

$$Ty = Ub. \tag{2.16}$$

Factorizarea Householder se bazează pe următoarele rezultate.

Propoziţia 2.5.1. *Oricare ar fi A o matrice pătratică de ordinul n şi simetrică, există un vector $v = (v_1, v_2, \ldots, v_n)^T$ astfel încât vectorul coloană $a^1 = Ae_1, e_1 = (1, 0, \ldots, 0)^T$ (a^1 este prima coloană a matricei A) are proprietatea*

$$a^1 - \frac{2v \cdot < v, a^1 >}{\|v\|^2} = \lambda \cdot e_1. \tag{2.17}$$

Pentru evitarea ambiguităţilor vom considera vectorul v dat de:

$$v = a^1 + sign(a_{11}) \cdot \|a^1\| \cdot e_1. \tag{2.18}$$

Propoziţia 2.5.2. *Oricare ar fi matricea simetrică A, matricea P definită prin:*

$$P = I - \frac{2 \cdot v \cdot v^T}{\|v\|^2} \tag{2.19}$$

este simetrică şi are proprietatea că elementele $2, 3, \ldots, n$ de pe prima coloană a matricei PA sunt nule, unde vectorul v este dat de relaţia (2.18).

Definiţia 2.5.1. *Se numeşte* ***matrice Householder*** *de ordin $n-1$ asociată matricei A şi se notează cu P_{n-1} o matrice de ordin $n-1$ de forma:*

$$P_{n-1} = I_{n-1} - \frac{2 \cdot v \cdot v^T}{\|v\|^2} \tag{2.20}$$

unde: $v = a^1_{n-1} + sign(a_{21}) \cdot \|a^1_{n-1}\| \cdot e_1$ este vectorul format cu componentele vectorului a^1 care este prima coloană a matricei A, $e_1 = (\underbrace{1, 0, \ldots, 0}_{n-1})^T$ şi I_{n-1} este matricea unitate de ordin $n-1$.

Propoziţia 2.5.3. *Matricea Hauseholder P_{n-1} asociată unei matrice simetrice A este simetrică şi are proprietatea că matricea U_1 definită prin:*

$$U_1 = \begin{pmatrix} 1 & 0 & \ldots & 0 \\ 0 & & & \\ & & P_{n-1} & \\ 0 & & & \end{pmatrix} \tag{2.21}$$

este simetrică şi verifică relaţia:

$$A^{(1)} = U_1 \, A \, U_1 = \begin{pmatrix} a_{11} & \alpha_1 & 0 & \ldots & 0 \\ \alpha^1 & a^{(1)}_{22} & a^{(1)}_{23} & \ldots & a^{(1)}_{2n} \\ 0 & a^{(1)}_{32} & a^{(1)}_{33} & \ldots & a^{(1)}_{3n} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & a^{(1)}_{n2} & a^{(1)}_{n3} & \ldots & a^{(1)}_{nn} \end{pmatrix} \tag{2.22}$$

Se consideră vectorul coloană a^2_{n-2} cu ultimele $n-2$ elemente ale coloanei matrice $A^{(1)}$. Cu acest vector se construieşte o matrice Householder de ordinul $n-2$, P_{n-2}. Matricea U_2 definită prin:

$$U_2 = \begin{pmatrix} 1 & 0 & 0 & \ldots & 0 \\ 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & & & \\ \vdots & \vdots & & P_{n-2} & \\ 0 & 0 & & & \end{pmatrix} \tag{2.23}$$

are proprietatea:

$$A^{(2)} = U_2 \, A^{(1)} \, U_2 = \begin{pmatrix} a_{11} & \alpha_1 & 0 & 0 & \ldots & 0 \\ \alpha_1 & a^{(1)}_{22} & \alpha_2 & 0 & \ldots & 0 \\ 0 & \alpha_2 & a^{(2)}_{33} & a^{(2)}_{34} & \ldots & a^{(2)}_{3n} \\ \ldots & \ldots & \ldots & \ldots & \ldots & \\ 0 & 0 & a^{(2)}_{n3} & a^{(2)}_{n4} & \ldots & a^{(2)}_{nn} \end{pmatrix} \tag{2.24}$$

Soluţia sistemului tridiagonal

$$Ty = Ub$$

este

$$y = \begin{pmatrix} 1 \\ \frac{20-10\sqrt{5}}{21} \\ \frac{5-6\sqrt{5}}{15} \end{pmatrix},$$

iar soluţia sistemului iniţial este

$$x = Uy = \begin{pmatrix} \frac{2+\sqrt{5}}{3} \\ \frac{2-\sqrt{5}}{3} \\ -\frac{2}{3} \end{pmatrix},$$

adică $x = \frac{2+\sqrt{5}}{3}$, $y = \frac{2-\sqrt{5}}{3}$ $z = -\frac{2}{3}$.

2.5.3 Probleme propuse

Exerciţiul 2.5.2. Să se găsească soluţiile următoarelor sisteme folosind factorizarea Householder:

a) $\begin{cases} x + 2y + z = 5 \\ 2x - y + 3z = 6 \\ x + 3y - 2z = 3 \end{cases}$

b) $\begin{cases} x + y - z + t = 3 \\ x + 2y + 3z + t = 9 \\ -x + 3y + 2t = 7 \\ x + y + 2z = 5. \end{cases}$

2.5.4 Implementare

A. Algoritm

Date de intrare: un sistem de ecuaţii

Date de ieşire: soluţia sistemului, obţinută folosind factorizarea Householder

Algoritmul constă din următoarele etape:

1. generarea matricei tridiagonale pentru $i = 1 \ldots n-2$

pentru $l = 1 \ldots n$

pentru $m = 1 \ldots n$

dacă $m = l$ atunci $u_{ml} = 1$

dacă $m \neq l$ atunci $u_{ml} = 0$

//Generăm vectorul v

$$\text{norm a} = \sqrt{\sum_{j=i+1}^{n} a_{ij}^2}$$

$e_{i+1} = 1$

pentru $j = i+2 \ldots n$

$\quad e_j = 0$

pentru $j = i+1 \ldots n$

$\quad v_j = a_{ij} + sign(a_{i,i+1}) \cdot \text{norm a} \cdot e_j$

$$\text{norm v} = \sum_{j=i+1}^{n} v_j^2$$

//Generăm matricea U

$j = i+1 \ldots n$

$\quad k = i+1 \ldots n$

$\quad\quad u_{jk} = u_{jk} - 2 \cdot v_j \cdot v_k / \text{ norm v}$

// D=AU

$m = 1 \ldots n$

$\quad l = 1 \ldots n$

$$\quad\quad d_{ml} = \sum_{k=1}^{n} a_{mk} \cdot u_{kl}$$

//A=UD=UAU

$m = 1 \ldots n$

$\quad l = 1 \ldots n$

$$\quad\quad a_{ml} = \sum_{k=1}^{n} u_{mk} \cdot d_{kl}$$

2. rezolvarea sistemului tridiagonal (vezi paragraful 2.3.4):

$$Ty = Ub$$

3. găsirea soluţiei sistemului iniţial:

$$x = Uy$$

B. Programe MAPLE şi rezultate

Observaţia 2.5.1. Deoarece o condiţie neceesară pentru aplicarea factorizării Householder este ca sistemul să fie simetric, folosim procedura `nedeterminate` (prezentată în paragraful 2.4.4). De asemenea, pentru descompunerea matricei tridiagonale, am folosit procedura `tridiagonal` prezentată în paragraful 2.3.4.

$$xo_2 := 0$$
$$test := 1$$
$$x_1 := 1.666666667$$
$$x_2 := 2.500000000$$
$$test := 3.004626063$$
$$xo_1 := 1.666666667$$
$$xo_2 := 2.500000000$$
$$x_1 := 0.8333333333$$
$$x_2 := 1.666666666$$
$$test := 1.178511303$$
$$xo_1 := 0.8333333333$$
$$xo_2 := 1.666666666$$
$$x_1 := 1.111111111$$
$$x_2 := 2.083333334$$
$$test := 0.5007710115$$
$$xo_1 := 1.111111111$$
$$xo_2 := 2.083333334$$
$$x_1 := 0.9722222220$$
$$x_2 := 1.944444444$$
$$test := 0.1964185512$$
$$xo_1 := 0.9722222220$$
$$xo_2 := 1.944444444$$
$$x_1 := 1.018518519$$
$$x_2 := 2.013888889$$
$$test := 0.08346183593$$
$$xo_1 := 1.018518519$$
$$xo_2 := 2.013888889$$
$$x_1 := 0.9953703703$$
$$x_2 := 1.990740740$$
$$test := 0.03273642604$$
$$xo_1 := 0.9953703703$$
$$xo_2 := 1.990740740$$
$$x_1 := 1.003086420$$
$$x_2 := 2.002314815$$
$$test := 0.01391030679$$
$$xo_1 := 1.003086420$$
$$xo_2 := 2.002314815$$

```
<-- exit jacobi (now at top level) = x = 1.003086420, y = 2.002314815}
```

$$x = 1.003086420,\ y = 2.002314815$$

Se pot compara rezultatele obţinute dacă eroarea ε se modifică:

```
> jacobi({3*x+y=5, x+2*y=5}, vector(2,[0,0]),0.01);
```

$$x = 1.003086420,\ y = 2.002314815$$

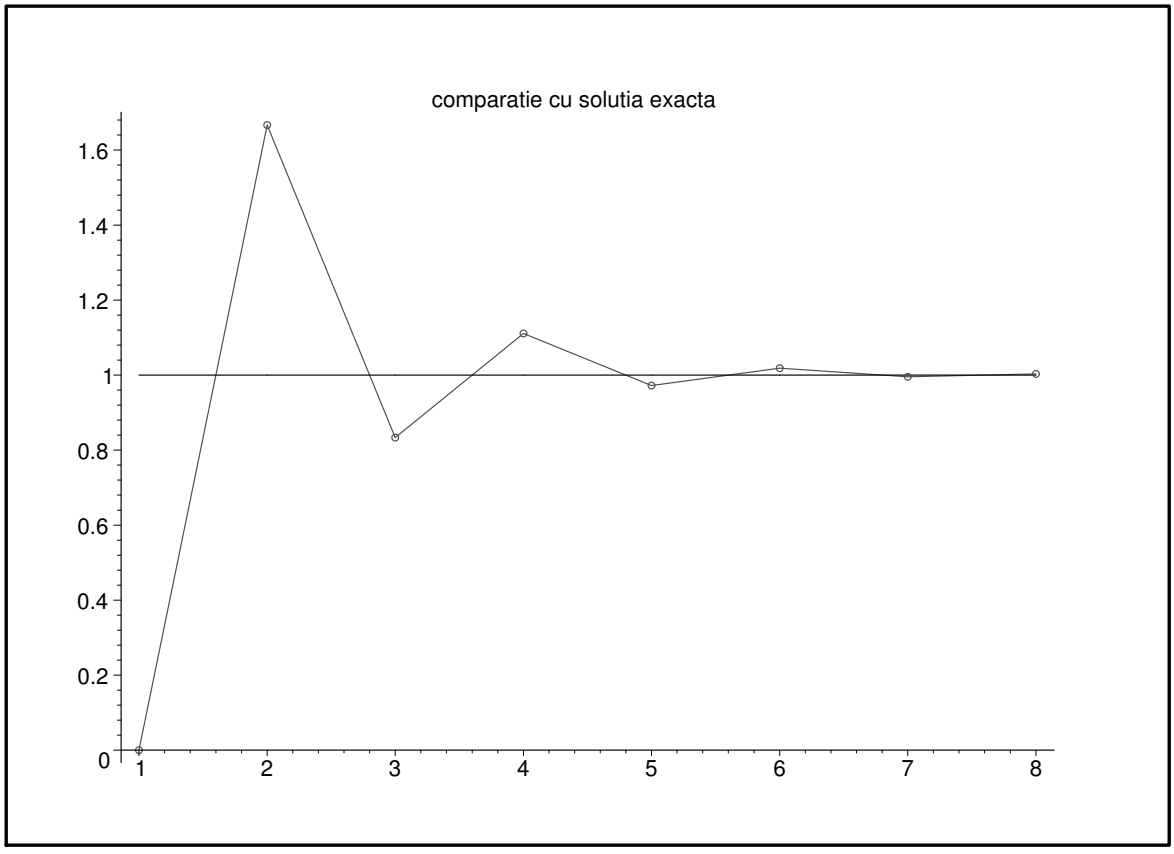
```
> jacobi({3*x+y=5, x+2*y=5}, vector(2,[0,0]),0.001);
```

$$x = 0.9998713993,\ y = 1.999742798$$

```
> jacobi({3*x+y=5, x+2*y=5}, vector(2,[0,0]),0.00001);
```

$$x = 1.000002381,\ y = 2.000001786$$

De asemenea, se poate compara şirul soluţiilor parţiale cu soluţia exactă obţinută rezolvând sistemul $Ax = b$ cu ajutorul procedurii `linsolve`. Pentru sistemul considerat mai sus, a cărui soluţie exactă este $x = 1$, $y = 2$, obţinem următoarele grafice:



traiectoria Gauss-Seidel a vectorului $x^{(0)}$, sunt date de relaţiile:

$$x_1^{(k+1)} = \left(b_1 - \sum_{j=2}^{n} a_{1j} \cdot x_j^{(k)} \right) \cdot \frac{1}{a_{11}} \tag{2.36}$$

$$x_i^{(k+1)} = \left(b_i - \sum_{j=1}^{i-1} a_{ij} \cdot x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} \cdot x_j^{(k)} \right) \cdot \frac{1}{a_{ii}},\ i = 2, \ldots, n. \tag{2.37}$$

2.7.2 Problemă rezolvată

Exerciţiul 2.7.1. Să se determine primele 3 puncte de pe traiectoria Gauss-Seidel a vectorului $(0, 0)^T$ pentru sistemul următor:

$$\begin{cases} 4x + y = -1 \\ 4x + 3y = -2. \end{cases}$$

Rezolvare

Sistemul se poate scrie sub forma $A\mathrm{x} = b$, unde

$$A = \begin{pmatrix} 4 & 1 \\ 4 & 3 \end{pmatrix},\ b = \begin{pmatrix} -1 \\ -2 \end{pmatrix}$$

iar matricea A se descompune în suma $L + D + U$, după cum urmează:

$$L = \begin{pmatrix} 0 & 0 \\ 4 & 0 \end{pmatrix},\ D = \begin{pmatrix} 4 & 0 \\ 0 & 3 \end{pmatrix},\ U = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}.$$

Algoritmul converge dacă raza spectrală a matricei

$$M = -(L + D)^{-1}U = \begin{pmatrix} 0 & -\frac{1}{4} \\ -\frac{4}{3} & 0 \end{pmatrix}$$

este strict subunitară. Efectuând calculele, obţinem

$$\rho(M) = \frac{\sqrt{3}}{3} < 1$$

şi deci algoritmul este convergent.
În continuare, aplicăm formulele (2.36)-(2.37) şi, plecând de la punctele $x^{(0)} = 0$, $y^{(0)} = 0$, obţinem:

$$x^{(1)} = -0.2500000000$$
$$y^{(1)} = -0.3333333333$$
$$x^{(2)} = -0.1666666667$$
$$y^{(2)} = -0.4444444443$$
$$x^{(3)} = -0.1388888889$$
$$y^{(3)} = -0.4814814813$$

Pentru comparaţie, determinăm soluţia exactă a sistemului considerat:

$$x = -\frac{1}{8},\ y = -\frac{1}{2}$$

adică $x = -0.125$, $y = -0.5$.

2.7.3 Probleme propuse

Exerciţiul 2.7.2. Să se verifice dacă se poate aplica metoda iterativă a lui Gauss-Seidel, şi în caz afirmativ să se găsească primele 3 elemente ale şirului de soluţii parţiale. Comparaţi cu soluţia exactă şi cu şirul de soluţii parţiale obţinut prin metoda lui Jacobi:

a) $\begin{cases} x + 3y = -2 \\ 2x + y = 6 \end{cases}$

b) $\begin{cases} x + 2y + z = 1 \\ 3x - y + 5z = 14 \\ x + y - z = -2 \end{cases}$

2.7.4 Implementare

A. Algoritm

Observaţia 2.7.1. Deoarece metoda Gauss-Seidel este o metodă iterativă, trebuie specificată o condiţie de oprire a algoritmului. În continuare vom folosi aceeaşi condiţie de oprire a algoritmului ca şi cea prezentată în paragraful 2.6.4:

$$\sqrt{\sum_{i=1}^{n}(x_i^{(k+1)} - x_i^{(k)})} < \varepsilon\sqrt{n}. \tag{2.38}$$

Date de intrare: un sistem de ecuaţii (o mulţime de ecuaţii), un punct iniţial, $x^{(0)}$, o eroare ε.

Date de ieşire: soluţia aproximativă a sistemului, obţinută în urma aplicării traiectoriei Gauss-Seidel vectorului $x^{(0)}$ până când este îndeplinită condiţia (2.38).

Algoritmul constă în următoarele etape:

1. generarea matricei A a sistemului şi a vectorului b
   - n - numărul de necunoscute (numărul de linii ale matricei A)
2. generarea matricelor L, D, U şi verificarea convergenţei metodei:
   $$\rho(-(L + D)^{-1} U) < 1$$
3. construcţia traiectoriei Gauss-Seidel

   repetă

$$x_1^{(k+1)} = \left(b_1 - \sum_{j=2}^{n} a_{1j} \cdot x_j^{(k)}\right) \cdot \frac{1}{a_{11}}$$
$$x_i^{(k+1)} = \left(b_i - \sum_{j=1}^{i-1} a_{ij} \cdot x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} \cdot x_j^{(k)}\right) \cdot \frac{1}{a_{ii}},\ i = \overline{2, n}.$$

   până când $\sqrt{\sum_{i=1}^{n}(x_i^{(k+1)} - x_i^{(k)})} < \varepsilon\sqrt{n}$

2.8.4 Implementare

A. Algoritm

Observaţia 2.8.2. Deoarece metoda relaxării succesive este o metodă iterativă, trebuie specificată o condiţie de oprire a algoritmului. Această condiţie este similară cu cea folosită în paragrafele anterioare, şi anume:

$$\sqrt{\sum_{i=1}^{n}(x_i^{(k+1)} - x_i^{(k)})} < \varepsilon\sqrt{n}. \tag{2.44}$$

Date de intrare: un sistem de ecuaţii (o mulţime de ecuaţii), un punct iniţial, $x^{(0)}$, o relaxare, ω, o eroare ε.

Date de ieşire: soluţia aproximativă a sistemului, obţinută în urma aplicării traiectoriei vectorului $x^{(0)}$ obţinută prin relaxări succesive până când este îndeplinită condiţia (2.44).

Algoritmul constă în următoarele etape:

1. generarea matricei A a sistemului şi a vectorului b
   - n - numărul de necunoscute (numărul de linii ale matricei A)
2. generarea matricelor L, D, U şi verificarea convergenţei metodei:

$$\rho\left(-\left(L+\frac{1}{\omega}D\right)^{-1}\left[\left(1-\frac{1}{\omega}\right)D+U\right]\right)<1$$

   şi

$$0<\omega<2$$

3. construcţia traiectoriei relaxărilor succesive

   repetă

$$x_1^{(k+1)} = (1-\omega)\cdot x_1^{(k)} + \frac{\omega}{a_{11}}\left[b_1 - \sum_{j=1}^{n} a_{1j}\cdot x_j^{(k)}\right] \tag{2.45}$$

$$x_i^{(k+1)} = (1-\omega)\cdot x_i^{(k)} + \frac{\omega}{a_{ii}}\left[b_i - \sum_{j=1}^{i-1} a_{ij}\cdot x_j^{(k+1)} - \sum_{j=i}^{n} a_{ij}\cdot x_j^{(k)}\right],\ i=\overline{2,n} \tag{2.46}$$

   până când $\sqrt{\sum_{i=1}^{n}(x_i^{(k+1)} - x_i^{(k)})} < \varepsilon\sqrt{n}$

B. Programe MAPLE şi rezultate

Prezentăm comparativ rezultatele aplicării metodei relaxării succesive pentru diferite valori ale lui ω:

```
> relsucc({3*x+y=5, x+2*y=5}, vector(2,[0,0]), 0.99, 0.0001);
```

$x = 0.9919288541,\ y = 2.024277742$

```
> relsucc({3*x+y=5, x+2*y=5}, vector(2,[0,0]), 0.9, 0.0001);
```

$x = 0.9091141587,\ y = 2.272710330$

```
> relsucc({3*x+y=5, x+2*y=5}, vector(2,[0,0]), 1.01, 0.0001);
```

$x = 1.007912178,\ y = 1.976281288$

```
> relsucc({3*x+y=5, x+2*y=5}, vector(2,[0,0]), 1.1, 0.0001);
```

$x = 1.071425885,\ y = 1.785716899$

```
> relsucc({3*x+y=5, x+2*y=5}, vector(2,[0,0]), 0.7, 0.0001);
```

$x = 0.6250722364,\ y = 3.124921273$

```
> relsucc({3*x+y=5, x+2*y=5}, vector(2,[0,0]), 1.3, 0.0001);
```

$x = 1.176464330,\ y = 1.470600344$

```
> relsucc({3*x+y=5, x+2*y=5}, vector(2,[0,0]), 1, 0.0001); #
> gauss-seidel
```

$x = 1.000014289,\ y = 1.999992856$

Capitolul 3

Rezolvarea ecuaţiilor şi a sistemelor de ecuaţii neliniare

Fie sistemul neliniar

$$F(x) = 0 \tag{3.1}$$

unde $F(x)$ este vectorul $(f_1(x), ..., f_n(x))^T$, funcţiile $f_1, ..., f_n : D \in \mathbb{R}^n \to \mathbb{R}^1$ sunt considerate cunoscute, iar vectorul $x = (x_1, ..., x_n)^T$ este necunoscut.

3.1 Metoda punctului fix

3.1.1 Breviar teoretic

Soluţiile sistemului neliniar (3.1) se caută printre punctele fixe $x^{(*)}$, adică printre soluţiile sistemului

$$G(x) = x \tag{3.2}$$

obţinut din sistemul iniţial prin alegerea

$$G(x) = x - [F'(x)]^{-1} \cdot F(x) \tag{3.3}$$

unde $F'(x)$ este matricea Jacobi asociată vectorului F, matrice despre care s-a presupus că este continuă şi inversabilă.

Presupunem că operatorul G are un punct fix $x^{(*)}$.

Definiţia 3.1.1. *Vom spune că $x^{(*)}$ este un punct de atracţie dacă există o sferă deschisă $S(x^{(*)}, r) = \{x \in \mathbb{R}^n \mid \|x - x^{(*)}\| < r\}$ cu următoarele proprietăţi:*

1. *$S(x^{(*)}, r) \subset D$ şi $x^{(k)}$ generat de*

$$x^{(k+1)} = G(x^{(k)}) \tag{3.4}$$

   este un şir bine definit pentru $\forall\, x^{(0)} \in S(x^{(*)}, r)$;

2. *$\forall\, x^{(0)} \in S(x^{(*)}, r)$ şirul $x^{(k)}$ definit de (3.4) aparţine lui D şi $x^{(k)} \xrightarrow[k\to\infty]{} x^{(*)}$.*

```
fixedpoint:=proc(eqn::set(equation), x0::vector, eps::float)
local x, n, g, i, j, jac, yo, test, k, y, jac1, eig;
if nops(eqn) <> vectdim(x0) then
  ERROR(‘problema nu este bine pusa!‘)
fi;
n:=nops(eqn);
x:=[op(indets(eqn, name))];
if nops(x) <> n then
  ERROR(‘numarul de ecuatii nu coincide cu numarul de
    necunoscute!‘)
fi;
g:=vector(n,0);
for i from 1 to n do g[i]:=x[i]+lhs(eqn[i])-rhs(eqn[i]); od;
evalm(g);
jac:=Matrix(n,n,0);
for i from 1 to n do for j from 1 to n do
jac[i,j]:=jacobian(g,x)[i,j]; od;od;
jac1:=[];
for i from 1 to n do
  for j from 1 to n do
    jac1:=[op(jac1),jacobian(g,x)[i,j]];
  od;
od;
eig:=max(op(jac1));
WARNING(‘punctul de plecare trebuie ales astfel incat expresia
  \%1 sa fie strict subunitara‘,eig);
yo:=x0;
test:=1; k:=1;
while test>=eps and k<50 do
  for i from 1 to n do
    test:=0;
    y[i]:=evalf(subs(seq(x[k]=yo[k],k=1..n),g[i]));
    if abs(y[i]-yo[i])>test then test:=abs(y[i]-yo[i]); fi;
    yo[i]:=y[i];
    if test>10^10 then ERROR(‘Algoritmul nu converge!‘);fi;
  od;
  k:=k+1;
od;
if k>=50 then
  ERROR(‘sunt necesare mai mult de 50 de iteratii pentru gasirea
    solutiei‘)
fi;
RETURN(seq(x[k]=y[k],k=1..n));
end:

debug(fixedpoint):
```

```
fixedpoint({(x^2+y)/6-x=0, (x+y^2)/8-y=0},vector(2,[0.5,0.5]),
  0.0001);
```

```
{--> enter fixedpoint, args = {1/6*x^2+1/6*y-x = 0, 1/8*x+1/8*y^2-y =
0}, array(1 .. 2,[(1)=.5,(2)=.5]), .1e-3
```

$$n := 2$$

$$x := [x,\ y]$$

$$g := [0,\ 0]$$

$$g_1 := \frac{x^2}{6} + \frac{y}{6}$$

$$g_2 := \frac{x}{8} + \frac{y^2}{8}$$

$$\left[\frac{x^2}{6} + \frac{y}{6},\ \frac{x}{8} + \frac{y^2}{8}\right]$$

$$jac := \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$jac_{1,\,1} := \frac{x}{3}$$

$$jac_{1,\,2} := \frac{1}{6}$$

$$jac_{2,\,1} := \frac{1}{8}$$

$$jac_{2,\,2} := \frac{y}{4}$$

$$jac1 := []$$

$$jac1 := [\frac{x}{3}]$$

$$jac1 := [\frac{x}{3},\ \frac{1}{6}]$$

$$jac1 := [\frac{x}{3},\ \frac{1}{6},\ \frac{1}{8}]$$

$$jac1 := [\frac{x}{3},\ \frac{1}{6},\ \frac{1}{8},\ \frac{y}{4}]$$

$$eig := \max(\frac{1}{6},\ \frac{x}{3},\ \frac{y}{4})$$

```
Warning, punctul de plecare trebuie ales astfel incat expresia
max(1/6,1/3*x,1/4*y) sa fie strict subunitara
```

$$yo := [0.5,\ 0.5]$$

$$test := 1$$

$$k := 1$$

$$test := 0$$

$$y_1 := 0.1250000000$$

$$test := 0.3750000000$$

$$yo_1 := 0.1250000000$$

$$test := 0$$

$$y_2 := 0.04687500000$$
$$test := 0.4531250000$$
$$yo_2 := 0.04687500000$$
$$k := 2$$
$$test := 0$$
$$y_1 := 0.01041666667$$
$$test := 0.1145833333$$
$$yo_1 := 0.01041666667$$
$$test := 0$$
$$y_2 := 0.001576741537$$
$$test := 0.04529825846$$
$$yo_2 := 0.001576741537$$
$$k := 3$$
$$test := 0$$
$$y_1 := 0.0002808747470$$
$$test := 0.01013579192$$
$$yo_1 := 0.0002808747470$$
$$test := 0$$
$$y_2 := 0.00003542010761$$
$$test := 0.001541321429$$
$$yo_2 := 0.00003542010761$$
$$k := 4$$
$$test := 0$$
$$y_1 := 0.5916499705\,10^{-5}$$
$$test := 0.0002749582473$$
$$yo_1 := 0.5916499705\,10^{-5}$$
$$test := 0$$
$$y_2 := 0.7397192861\,10^{-6}$$
$$test := 0.00003468038832$$
$$yo_2 := 0.7397192861\,10^{-6}$$
$$k := 5$$

```
<-- exit fixedpoint (now at top level) = x = .5916499705e-5, y =
.7397192861e-6}
```

$$x = 0.5916499705\,10^{-5},\ y = 0.7397192861\,10^{-6}$$

3.2 Metoda lui Newton

3.2.1 Breviar teoretic

Soluţia $x^{(*)}$ a sistemului neliniar (3.1) este dată de limita şirului $x^{(k)}$, unde:

$$x^{(k+1)} = x^{(k)} - [F'(x^{(k)})]^{-1} \cdot F(x^{(k)}), \qquad k = 0, 1, 2, \ldots \tag{3.7}$$

în cazul şirului de iteraţii succesive clasic al lui Newton, respectiv

$$x^{(k+1)} = x^{(k)} - [F'(x^{(0)})]^{-1} \cdot F(x^{(k)}), \qquad k = 0, 1, 2, \ldots \tag{3.8}$$

în cazul şirului de iteraţii succesive simplificat al lui Newton.

Teorema 3.2.1. *Fie $F : D \subset \mathbb{R}^n \to \mathbb{R}^n$ şi ecuaţia $F(x) = 0$, despre care presupunem că are o soluţie $x^{(*)} \in D$. Dacă există o sferă deschisă $S(x^{(*)}, r) = \{x \in \mathbb{R}^n \mid \|x - x^{(*)}\| < r\}$ pe care F este de clasă $\mathcal{C}^1$ şi $F'(x^{(*)})$ este nesingulară atunci, în cazul metodei lui Newton clasice, $x^{(*)}$ este un punct de atracţie.*

Teorema 3.2.2. *În condiţiile teoremei precedente, dacă raza spectrală ρ a matricei*

$$I - [F'(x^{(0)})]^{-1} \cdot F'(x^{(*)})$$

este strict subunitară atunci, în cazul metodei lui Newton simplificată, $x^{(*)}$ este un punct fix atractiv.

Un caz particular al metodei lui Newton este acela pentru care $n = 1$. În acest caz ecuaţia (3.1) devine

$$f(x) = 0, \tag{3.9}$$

şirul de iteraţii succesive clasic al lui Newton se scrie

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}, \qquad k = 0, 1, 2, \ldots \tag{3.10}$$

iar şirul de iteraţii succesive simplificat al lui Newton se scrie

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(0)})}, \qquad k = 0, 1, 2, \ldots \tag{3.11}$$

3.2.2 Probleme rezolvate

Exerciţiul 3.2.1. Să se găsească primii 3 termeni ai şirului de iteraţii succesive (clasic şi simplificat) al lui Newton, cu $x^{(0)} = (0.7, 0.4)^T$ pentru sistemul:

$$\begin{cases} x^2 + y^2 - 1 = 0 \\ x^3 - y = 0. \end{cases}$$

Rezolvare

Sistemul se mai scrie $F(\mathrm{x}) = 0$, unde

$$F(\mathrm{x}) = (x^2 + y^2 - 1,\, x^3 - y)^T \,,\ \mathrm{x} = (x, y)^T.$$

Jacobianul operatorului F este:

$$F'(\mathrm{x}) = \begin{pmatrix} 2x & 2y \\ 3x^2 & -1 \end{pmatrix}$$

A. Rezolvare folosind metoda lui Newton clasică

Determinăm produsul:

$$[F'(\mathrm{x})]^{-1} F(\mathrm{x}) = \left(\frac{(x^2 + y^2 - 1)}{2x(1 + 3yx)} + \frac{y(x^3 - y)}{x(1 + 3yx)}, \frac{3x(x^2 + y^2 - 1)}{2 + 6yx} - \frac{(x^3 - y)}{1 + 3yx} \right)^T$$

```
while test>eps do
  if c>1000 then ERROR(‘Algoritmul nu converge!‘) fi;
  y0:=y;
  y:=evalf(y-subs(x=y,f)/subs(x=y,fp));
  test:=abs(y-y0);
  c:=c+1;
od;
RETURN(x=y);
end:

debug(cnewton1d):

cnewton1d(sin(xx)-xx=0, 0.2,0.001);
```

```
> cnewton1d(sin(xx)-xx=0, 0.2,0.001);
```

```
{--> enter cnewton1d, args = sin(xx)-xx = 0, .2, .1e-2
```

$$x := xx$$
$$f := \sin(xx) - xx$$
$$fp := \cos(xx) - 1$$
$$y := 0.2$$
$$c := 1$$
$$test := 1$$
$$y0 := 0.2$$
$$y := 0.1332443177$$
$$test := 0.0667556823$$
$$c := 2$$
$$y0 := 0.1332443177$$
$$y := 0.08880323922$$
$$test := 0.04444107848$$
$$c := 3$$
$$y0 := 0.08880323922$$
$$y := 0.05919437624$$
$$test := 0.02960886298$$
$$c := 4$$
$$y0 := 0.05919437624$$
$$y := 0.03946061575$$
$$test := 0.01973376049$$
$$c := 5$$
$$y0 := 0.03946061575$$
$$y := 0.02630640064$$
$$test := 0.01315421511$$
$$c := 6$$
$$y0 := 0.02630640064$$
$$y := 0.01753738944$$
$$test := 0.00876901120$$

$$c := 7$$
$$y0 := 0.01753738944$$
$$y := 0.01169155254$$
$$test := 0.00584583690$$
$$c := 8$$
$$y0 := 0.01169155254$$
$$y := 0.007794289520$$
$$test := 0.003897263020$$
$$c := 9$$
$$y0 := 0.007794289520$$
$$y := 0.005196191723$$
$$test := 0.002598097797$$
$$c := 10$$
$$y0 := 0.005196191723$$
$$y := 0.003464143309$$
$$test := 0.001732048414$$
$$c := 11$$
$$y0 := 0.003464143309$$
$$y := 0.002309495886$$
$$test := 0.001154647423$$
$$c := 12$$
$$y0 := 0.002309495886$$
$$y := 0.001539688244$$
$$test := 0.000769807642$$
$$c := 13$$

```
<-- exit cnewton1d (now at top level) = xx = .1539688244e-2}
```

$$xx = 0.001539688244$$

Capitolul 4

Interpolare polinomială. Funcţii spline

În practică este des întâlnită situaţia în care se cunoaşte valoarea unei funcţii f în diferite puncte x_i şi se cere valoarea sa într-un punct intermediar. De exemplu, se poate cere valoarea temperaturii aerului la ora 14.30, cunoscându-se temperaturile aerului luate din oră în oră.

Astfel, se pune problema ca, plecând de la punctele date (x_i, y_i) să se determine o funcţie de interpolare al cărei grafic să treacă prin toate punctele date. Interpolarea se numeşte polinomială atunci când se caută funcţii polinomiale având proprietăţile menţionate.

4.1 Polinomul lui Newton cu diferenţe divizate

4.1.1 Breviar teoretic

Fie funcţia $f : X \to \mathbb{R}^1$ dată prin: $y_i = f(x_i), i = 0, 1, \ldots, m$.

Diferenţa divizată de ordinul întâi a lui f relativ la punctul x_r este numărul definit de fracţia:

$$(\mathcal{D}^1 f)(x_r) = \frac{f(x_{r+1}) - f(x_r)}{x_{r+1} - x_r}. \tag{4.1}$$

Diferenţa divizată de ordinul al doilea a funcţiei f relativ la punctul x_r este prin definiţie numărul:

$$(\mathcal{D}^2 f)(x_r) = \frac{(\mathcal{D}^1 f)(x_{r+1}) - (\mathcal{D}^1 f)(x_r)}{x_{r+2} - x_r} = \frac{[x_{r+1}, x_{r+2}, f] - [x_r, x_{r+1}, f]}{x_{r+2} - x_r} \tag{4.2}$$

Prin calcul se găseşte că diferenţa divizată de ordin k a lui f în x_r este:

$$(\mathcal{D}^k f)(x_r) = \sum_{i=0}^{k} \frac{f(x_{r+i})}{\prod\limits_{\substack{m=0 \\ m \neq i}}^{k} (x_{r+i} - x_{r+m})} \tag{4.3}$$

Polinomul lui Newton cu diferenţe divizate se defineşte ca fiind polinomul:

$$\begin{aligned} P_m(x) =& f(x_0) + (\mathcal{D}^1 f)(x_0)(x - x_0) + (\mathcal{D}^2 f)(x_0)(x - x_0)(x - x_1) + \dots \\ &+ (\mathcal{D}^m f)(x_0)(x - x_0)(x - x_1)\dots(x - x_{m-1}) \end{aligned} \tag{4.4}$$

Astfel, funcţia f se poate aproxima după cum urmează:

$$f(x) = P_m(x) + R_m(x), \tag{4.5}$$

unde

$$R_m(x) = \frac{f^{(m+1)}(\xi)}{(m+1)!}(x - x_0)(x - x_1)\dots(x - x_{m-1})(x - x_m) \tag{4.6}$$

este restul sau eroarea de aproximare la interpolarea polinomială.

În cazul în care $x_{i+1} - x_i = h = const$ (i.e. nodurile x_i sunt echidistante), se pot introduce diferenţele finite. Astfel,

$$\triangle f(x_k) = f(x_{k+1}) - f(x_k) \tag{4.7}$$

se numeşte diferenţă finită la dreapta, iar

$$\triangledown f(x_k) = f(x_k) - f(x_{k-1}) \tag{4.8}$$

se numeşte diferenţă finită la stânga.

Diferenţele finite de ordin superior se definesc recursiv, după cum urmează:

$$\triangle^n f(x_k) = \triangle^{n-1} f(x_{k+1}) - \triangle^{n-1} f(x_k) \tag{4.9}$$

$$\triangledown^n f(x_k) = \triangledown^{n-1} f(x_k) - \triangledown^{n-1} f(x_{k-1}) \tag{4.10}$$

Legătura dintre diferenţele divizate şi diferenţele finite este următoarea:

$$(\mathcal{D}^m f)(x_0) = \frac{\triangle^m f(x_0)}{m! h^m}, \qquad (\mathcal{D}^m f)(x_m) = \frac{\triangledown^m f(x_m)}{m! h^m}. \tag{4.11}$$

În acest fel, obţinem polinomul lui Newton cu diferenţe finite la dreapta:

$$\begin{aligned} p_m(x) =& f(x_0) + \frac{\triangle f(x_0)}{h}(x - x_0) + \frac{\triangle^2 f(x_0)}{2! h^2}(x - x_0)(x - x_1) + \dots + \\ &+ \frac{\triangle^m f(x_0)}{m! h^m}(x - x_0)\dots(x - x_{m-1}), \end{aligned} \tag{4.12}$$

respectiv polinomul lui Newton cu diferenţe finite la stânga:

$$\begin{aligned} p_m(x) =& f(x_m) + \frac{\triangledown f(x_m)}{h}(x - x_m) + \frac{\triangledown^2 f(x_m)}{2! h^2}(x - x_m)(x - x_{m-1}) + \dots \\ &+ \frac{\triangledown^m f(x_m)}{m! h^m}(x - x_m)\dots(x - x_1). \end{aligned} \tag{4.13}$$

4.1.2 Probleme rezolvate

Exerciţiul 4.1.1. Să se găsească polinomul de interpolare pentru următorul set de date:

x	1	2	3	4	5
$f(x)$	2	5	10	17	26

folosind

a) diferenţe divizate

b) diferenţe finite la dreapta

c) diferenţe finite la stânga.

Rezolvare

A. Polinomul lui Newton cu diferenţe divizate

1. Obţinerea diferenţelor divizate:

x	$f(x)$	$\mathcal{D}^1 f(x)$	$\mathcal{D}^2 f(x)$	$\mathcal{D}^3 f(x)$	$\mathcal{D}^4 f(x)$
1	2				
		$\frac{5-2}{2-1}=3$			
2	5		$\frac{5-3}{3-1}=1$		
		$\frac{10-5}{3-2}=5$		$\frac{1-1}{4-1}=0$	
3	10		$\frac{7-5}{4-2}=1$		$\frac{0-0}{5-1}=0$
		$\frac{17-10}{4-3}=7$		$\frac{1-1}{5-2}=0$	
4	17		$\frac{9-7}{5-3}=1$		
		$\frac{26-17}{5-4}=9$			
5	26				

Şirul diferenţelor divizate este şirul primelor valori de pe fiecare coloană, începând cu valorile funcţiei, adică:

$$[2, 3, 1, 0, 0].$$

2. Obţinerea polinomului de interpolare

$$\begin{aligned} P(x) =& 2 + 3 \cdot (x-1) + 1 \cdot (x-1)(x-2) + 0 \cdot (x-1)(x-2)(x-3) + \\ & + 0 \cdot (x-1)(x-2)(x-3)(x-4) = \\ =& 2 + 3x - 3 + x^2 - 3x + 2 = \\ =& x^2 + 1. \end{aligned}$$

B. Polinomul lui Newton cu diferenţe finite la dreapta

Această metodă este aplicabilă, deoarece $x_{i+1} - x_i = 1 \overset{not}{=} h$, $i = \overline{1,4}$.

1. Obţinerea diferenţelor finite la dreapta

```
ddinterp(x,fx,10);

Warning, Polinomul de interpolare da rezultate corecte doar
pentru x in intervalul [1,5]
```


sau

```
ddinterp(x,fx,z);
```

$$z^2 + 1$$

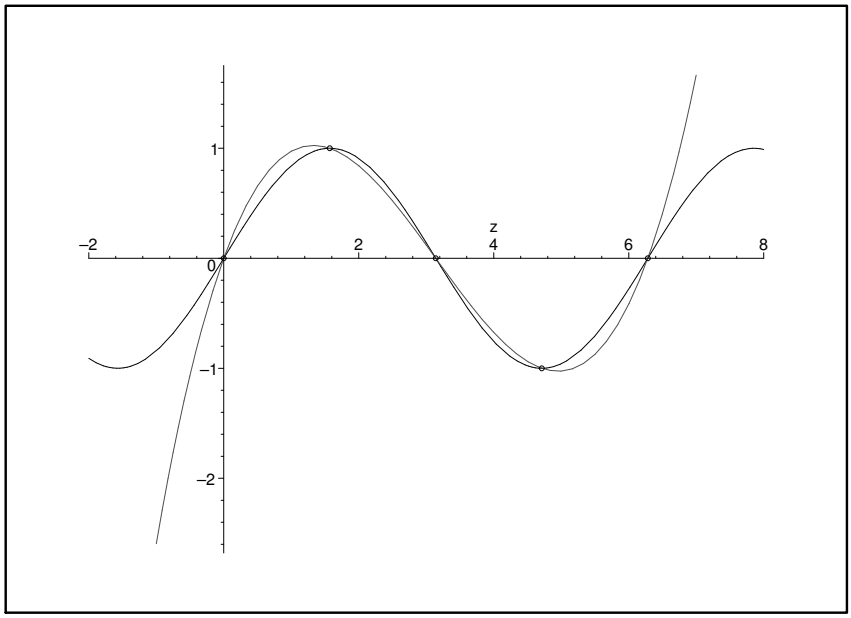
Observaţia 4.1.1. Polinomul de interpolare dă rezultate exacte pentru funcţii polinomiale de grad maxim n, unde n este numărul de puncte în care se cunoaşte valoarea funcţiei căutate.

Observaţia 4.1.2. Fie o funcţie tabelată, dată prin lista ordonată a variabilelor $[x_1 = min, x_2, ..., x_n = max]$, şi lista valorilor sale $[f_1, f_2, ..., f_n]$. Polinomul de interpolare dă rezultate apropiate de valoarea funcţiei doar pentru x în intervalul $[x_1, x_n]$. Pentru a ilustra acest fapt, considerăm funcţia $\sin x$ pe intervalul $[0, 2\pi]$, şi construim polinomul de interpolare cu diferenţe divizate. Pentru o vizualizare mai bună, am reprezentat grafic punctele de interpolare, funcţia $\sin x$ şi polinomul de interpolare.

Avem astfel:

```
x:=[0,Pi/2,Pi,3*Pi/2,2*Pi]: fx:=[0,1,0,-1,0]:
f:=ddinterp(x,fx,z);
p1:=plot(f, z=-2..8, color=red):
p2:=pointplot([seq([x[i],fx[i]],i=1..nops(x))], symbol=circle):
p3:=plot(sin(t), t=-2..8, color=green):
display(p1,p2,p3);
```

$$f := \frac{8(z^2 - 3z\pi + 2\pi^2)z}{3\pi^3}$$



Pentru polinomul lui Newton cu diferenţe finite la dreapta, respectiv la stânga, am folosit proceduri similare celor folosite pentru polinomul lui Newton cu diferenţe divizate.

Observaţia 4.1.3. Fie o funcţie tabelată, dată prin lista ordonată a variabilelor, $[x_1 = min, x_2, ..., x_n = max]$, şi lista valorilor sale, $[f_1, f_2, ..., f_n]$. Polinomul de interpolare cu diferenţe finite nu aproximează bine funcţia, pentru valori ale lui x în afara intervalului $[x_1, x_n]$. Un exemplu intuitiv în acest sens îl constituie următoarea secvenţă de program:

```
x:=[0,Pi/2,Pi,3*Pi/2,2*Pi]: fx:=[1,0,-1,0,1]:
f:=dfdinterp(x,fx,z);
g:=dfsinterp(x,fx,z);
p1d:=plot(f, z=-1.5..7.5, color=red):
p1s:=plot(f, z=-1.5..7.5, color=red):
p2:=pointplot([seq([x[i],fx[i]],i=1..nops(x))], symbol=circle):
p3:=plot(cos(t), t=-2..8, color=black, thickness=2):
display(p1d,p2,p3);
display(p1s,p2,p3);
```

al cărei rezultat este:

$$f := -\frac{8\,z^4 - 32\,z^3\,\pi + 34\,z^2\,\pi^2 - 4\,z\,\pi^3 - 3\,\pi^4}{3\,\pi^4}$$

$$g := -\frac{8\,z^4 - 16\,z^3\,\pi + 10\,z^2\,\pi^2 - 8\,z\,\pi^3 - 3\,\pi^4}{3\,\pi^4}$$