Lucrarea de laborator nr. 10

I. Scopul lucrării

Aproximarea în medie prin metoda celor mai mici pătrate

II. Conținutul lucrării

1. Metoda celor mai mici pătrate
2. Proceduri MAPLE şi exemple

III. Prezentarea lucrării

III.1. Metoda celor mai mici pătrate

Fie f : [a, b] $\rightarrow$ **R** o funcție. Fie $x_0, x_1, \ldots, x_n$ n+1 puncte distincte din intervalul [a, b] pentru care se cunosc valorile funcției $y_i = f(x_i)$ pentru orice i = 0,1,…n. Aproximarea funcției f printr-un polinom de interpolare nu este indicată în următoarele situații:

- când n este un număr foarte mare, ceea ce determină un volum mare de calcul pentru determinarea coeficienților de interpolare
- când valorile $y_i = f(x_i)$ nu sunt exacte.

În aceste situații se poate folosi aproximarea funcției prin metoda celor mai mici pătrate.

Fie H un spațiu Hilbert şi H_0 un subspațiu Hilbert al său. Fie u un element al lui H. Aproximarea lui u pe H_0 presupune determinarea unui element $u_0 \in H_0$ cu proprietatea că

$$\|u - u_0\| = \inf_{v \in H_0} \|u - v\|$$

u_0 se numeşte ***element de cea mai bună aproximare*** pentru u. Elementul de cea mai bună aproximare este unic şi are proprietatea că $u - u_0$ este ortogonal tuturor elementelor din H_0:

$$< u - u_0, v> = 0 \text{ pentru orice } v \in H_0.$$

Considerăm mulțimea funcțiilor definite pe intervalul [a, b], şi cele n+1 puncte distincte din intervalul [a, b] $x_0, x_1, \ldots, x_n$. Spunem că funcțiile f şi g din această mulțime sunt egale aproape peste tot (și vor fi identificate) dacă $f(x_i) = g(x_i)$ pentru orice i = 0,1,…, n.

Introducem următorul produs scalar

$$<f, g> = \sum_{i=0}^{n} p(x_i) f(x_i) g(x_i)$$

unde p(x) este o ***funcție pondere*** introdusă în ipoteza că aproximările $f(x_i)$ sunt diferite ca ordin de mărime. Funcția p are următoarele proprietăți:

- $p(x_i) > 0$
- $\sum_{i=0}^{n} p(x_i) = 1.$

Relativ la produsul scalar introdus norma lui f este definită prin

$$\|f\|^2 = \sum_{i=0}^{n} p(x_i) f^2(x_i).$$

Fie $\varphi_1, \varphi_2, \ldots \varphi_m$ un sistem de m funcții liniar independente definite pe [a, b], cu $m \le n$. Convenim să numit spațiul generat de ele spațiul polinoamelor generalizate şi să-l notăm H_m. Deci un ***polinom generalizat*** $F \in H_m$ este de forma

$$F(x) = \sum_{i=0}^{m} c_i \varphi_i(x).$$

În cazul metodei celor mai mici pătrate, elementul F_0 care dă cea mai bună aproximare a funcției f pe H_m trebuie să satisfacă condiția

$$\sum_{i=0}^{n} p(x_i)(f(x_i) - F_0(x_i))^2 = \inf_{F \in H_m} \sum_{i=0}^{n} p(x_i)(f(x_i) - F(x_i))^2$$

Determinarea coeficienților c_j ai polinomului generalizat F_0 cu ajutorul acestei relații este dificilă. Se folosește proprietatea

$$<f - F_0, \varphi_j> = 0 \text{ pentru orice } j = 0, 1, 2 \ldots m.$$

ceea ce revine la

$$<f, \varphi_j> = c_0<\varphi_0, \varphi_j> + c_1<\varphi_1, \varphi_j> + \ldots + c_m<\varphi_m, \varphi_j>, \; j = 0, 1, \ldots, m.$$

Notăm

$$a_{ij} = <\varphi_i, \varphi_j> = \sum_{k=0}^{n} p(x_k) \varphi_i(x_k) \varphi_j(x_k)$$

$$b_j = <f, \varphi_j> = \sum_{k=0}^{n} p(x_k) f(x_k) \varphi_j(x_k)$$

Pentru determinarea coeficienților c_j ai polinomului generalizat F_0 se rezolvă sistemul

$$\begin{cases} c_0a_{00} + c_1a_{10} + \ldots + c_ma_{m0} = b_0 \\ c_0a_{01} + c_1a_{11} + \ldots + c_ma_{m1} = b_1 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ c_0a_{0m} + c_1a_{1m} + \ldots + c_ma_{mm} = b_m \end{cases}$$

Determinantul acestui sistem fiind un determinant Gramm (elementele sale sunt produse scalare) este diferit de zero, deoarece sistemul de funcții φ_1, φ_2, $\ldots\varphi_m$ este un sistem liniar independent.

Dacă $\varphi_j(x) = x^j$, j =0, 1,…, m, m ≤ n, şi $p \equiv \frac{1}{n+1}$, atunci

$$F_0(x) = \sum_{i=0}^{m} c_i x^i \ .$$

iar sistemul anterior devine

$$\begin{cases} c_0(n+1) + c_1 \sum_{i=0}^{m} x_i + \ldots + c_m \sum_{i=0}^{m} x_i^m = \sum_{i=0}^{m} f(x_i) \\ c_0 \sum_{i=0}^{m} x_i + c_1 \sum_{i=0}^{m} x_i^2 + \ldots + c_m \sum_{i=0}^{m} x_i^{m+1} = \sum_{i=0}^{m} x_i f(x_i) \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ c_0 \sum_{i=0}^{m} x_i^m + c_1 \sum_{i=0}^{m} x_i^{m+1} + \ldots + c_m \sum_{i=0}^{m} x_i^{2m} = \sum_{i=0}^{m} x_i^m f(x_i) \end{cases}$$

Acest sistem este numit ***sistemul normal al lui Gauss***.

III.2. Proceduri MAPLE şi exemple

Parametrii procedurii `patrateMici` de mai jos sunt:

n - în intervalul [a, b] se consideră n +1 puncte

x - tablou ce conține ce n +1 puncte: $x_0, x_1, \ldots, x_n$

y - tablou ce conține valorile funcției f în ce n +1 puncte: $x_0, x_1, \ldots, x_n$($y_0 = f(x_0)$, $y_1 = f(x_1)$, ….,$y_n = f(x_n)$)

m –numărul funcțiilor liniar independente $\varphi_1, \varphi_2, \ldots\varphi_m$

g – tablou ce conține expresiile funcțiilor $\varphi_1, \varphi_2, \ldots \varphi_m$; convenim să notăm variabila independentă a acestor funcții cu t.

Procedura întoarce un vector ce conține coeficienții polinomului generalizat ce aproximează f.

Se consideră că ponderea $p \equiv \frac{1}{n+1}$.

```
>patrateMici := proc(n, y, x, m, g)
local i, j, k, a, b, c;
    a := matrix(m + 1, m + 1);
    b := vector(m + 1);
    c := vector(m + 1);
    for i from 0 to m do for j from 0 to m do
            a[i + 1, j + 1] := 0;
            for k from 0 to n do
              a[i + 1, j + 1] :=a[i + 1, j + 1]
                 + evalf(subs(t = x[k], g[i]))*
                     evalf(subs(t = x[k], g[j]))
            od
        od
    od;
    for i from 0 to m do
        b[i + 1] := 0;
        for k from 0 to n do
           b[i + 1] :=b[i + 1]
            + evalf(subs(t = x[k], g[i]))*y[k]
        od
    od;
    if det(a) = 0 then
        print(`Sistem de functii liniar dependent`);
        RETURN(NULL)
    fi;
    c := linsolve(a, b);
    RETURN(evalm(c))
end;
```

În exemplele ce urmează vom folosi şi procedurile `poligeneral` şi `desen`. Procedura `poligeneral` primeşte drept parametri m, tabloul ce conține expresiile funcțiilor din sistemul $\varphi_1, \varphi_2, \ldots \varphi_m$, vectorul coeficienților polinomului generalizat ce aproximează f şi un punct a. Procedura întoarce

valoarea polinomului generalizat în a. Procedura `desen` reprezintă grafic în același sistem de axe de coordonate polinomul generalizat ce aproximează funcția şi cele n+1 puncte date inițial. Punctele sunt reprezentate prin elipse. Parametrii procedurii sunt funcția determinată de polinomul generalizat, numărul n, vectorul x ce conține punctele x_0, x_1, …, x_n, şi vectorul y ce conține valorile funcției. Înainte de a folosi aceste proceduri trebuie încărcate pachetele linalg, plots şi plottools.

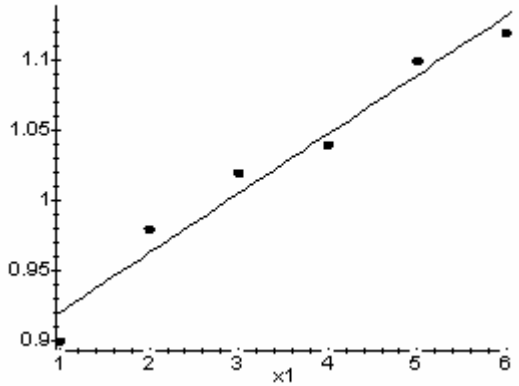
```
>poligeneral := proc(m, g, c, a)
local i, fx, f;
    fx := 0;
    for i from 0 to m do
        fx := fx + evalf(subs(t = a, g[i]))*c[i + 1]
    od;
    RETURN(evalf(fx))
end;
```

```
>desen := proc(f, n, x, y)
local i, ra, rb, d1, d2, d3, x1, x2, y1, y2;
    x1 := min(seq(x[i], i = 0 .. n));
    x2 := max(seq(x[i], i = 0 .. n));
    y1 := min(seq(y[i], i = 0 .. n));
    y2 := max(seq(y[i], i = 0 .. n));
    ra := 1/100*x2 - 1/100*x1;
    rb := 1/100*y2 - 1/100*y1;
    d1 := seq(
        ellipse([x[i], y[i]], ra, rb, filled = true,
        color = black), i = 0 .. n);
    d2 := plot(f(x),
        x = 101/100*x1 - 1/100*x2 .. 101/100*x2 -
1/100*x1);
    d3 := d1, d2;
    display(d3)
end;
```

Exemple

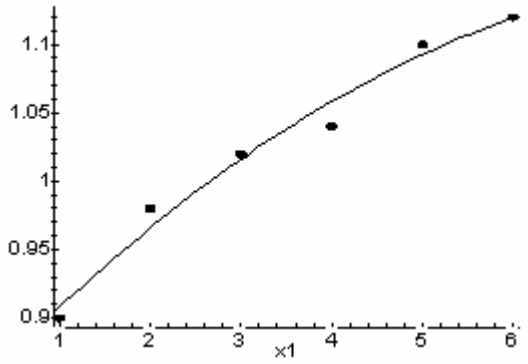
```
> with(plots);
> with(plottools);
> with(linalg);
> x1:=array(0..5,[1,2,3,4,5,6]);
```

```
x1 := array(0 .. 5, [
     (0) = 1
     (1) = 2
     (2) = 3
     (3) = 4
     (4) = 5
     (5) = 6
     ])
> y1:=array(0..5,[0.9,0.98,1.02,1.04,1.10,1.12]);
y1 := array(0 .. 5, [
     (0) = .9
     (1) = .98
     (2) = 1.02
     (3) = 1.04
     (4) = 1.10
     (5) = 1.12
     ])
> g1:=array(0..1,[1,t]);
g1 := array(0 .. 1, [
     (0) = 1
     (1) = t
     ])
> c1:=patrateMici(5,y1,x1,1,g1);
          c1 := [.8786666666, .0422857143]
> f1:=(x->poligeneral(1,g1,c1,x));
        f1 := x -> poligeneral(1, g1, c1, x)
> desen(f1,5,x1,y1);
```

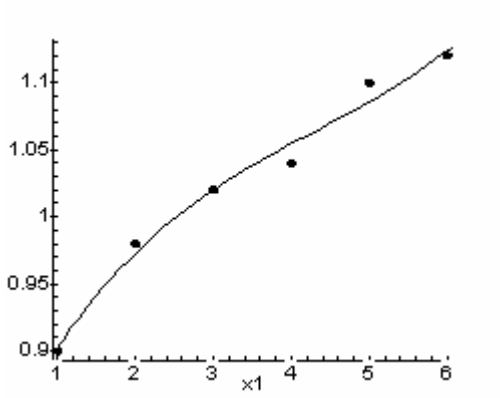



```
> g2:=array(0..2,[1,t,t^2]);
g2 := array(0 .. 2, [
```

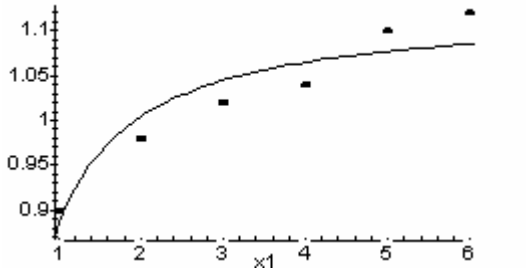
```
    (0) = 1
    (1) = t
    (2) = t^2
    ])
> c2:=patrateMici(5,y1,x1,2,g2);
   c2 := [.8420000017, .0697857129, -.00392857123]
> f2:=(x->poligeneral(2,g2,c2,x));
        f2 := x -> poligeneral(2, g2, c2, x)
> desen(f2,5,x1,y1);
```




```
> g3:=array(0..3,[1,t,t^2,t^3]);
g3 := array(0 .. 3, [
    (0) = 1
    (1) = t
    (2) = t^2
    (3) = t^3
])
> c3:=patrateMici(5,y1,x1,3,g3);
   c3 := [.7999998922, .1226191802, -.0214286144,
                    .001666670689]
> f3:=(x->poligeneral(3,g3,c3,x));
        f3 := x -> poligeneral(3, g3, c3, x)
> desen(f3,5,x1,y1);
```



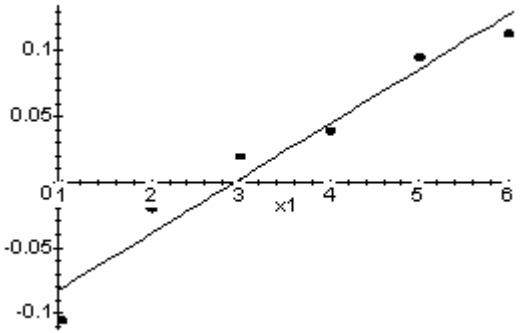

```
> g4:=array(0..1,[1,1/t]);
g4 := array(0 .. 1, [
    (0) = 1
    (1) = 1/t
    ])
> c4:=patrateMici(5,y1,x1,1,g4);
          c4 := [1.125359736, -.2416973117]
> f4:=(x->poligeneral(1,g4,c4,x));
        f4 := x -> poligeneral(1, g4, c4, x)
> desen(f4,5,x1,y1);
```



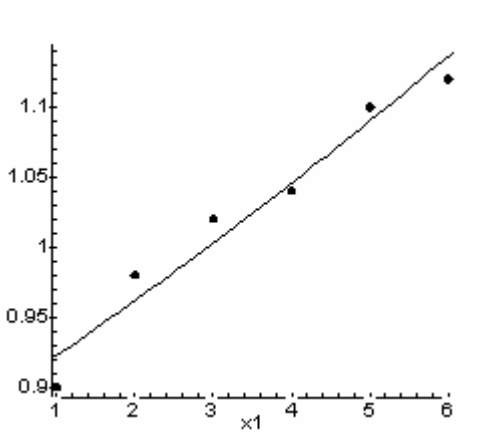
Procedura `compune` de mai jos întoarce tabloul obținut prin aplicarea funcției indicată ca prim parametru tabloului de valori y. Acest nou tablou poate fi folosit mai departe pentru determinarea unui polinom generalizat care să aproximeze funcția din care provine. Aproximația funcției inițiale (din care provine tabloul de valori y) se obține compunând la stânga polinomul generalizat cu inversa funcției din procedura `compune`.

```
>compune := proc(f, y, n)
local i, y1;
```

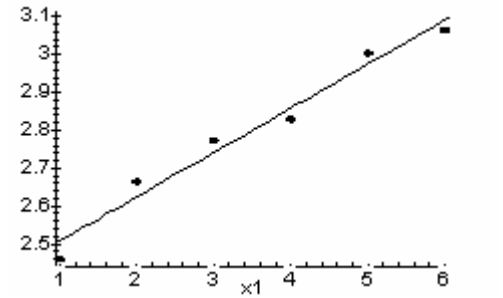
```
    y1 := array(0 .. n);
    for i from 0 to n do y1[i] := f(y[i]) od;
    RETURN(evalm(y1))
end;


> h1:=(x->ln(x));
                          h1 := ln
> y2:=compune(h1,y1,5);
                          y2 := y1
> c5:=patrateMici(5,y2,x1,1,g1);
         c5 := [-.1222571114, .04169722148]
> f5:=(x->poligeneral(1,g1,c5,x));
        f5 := x -> poligeneral(1, g1, c5, x)
> desen(f5,5,x1,y2);
```



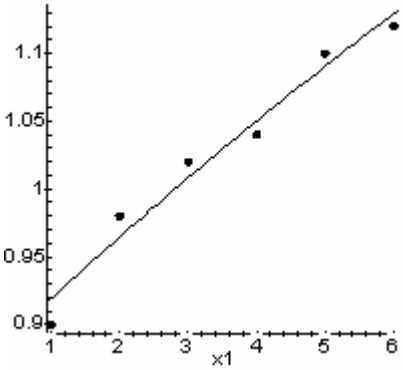

```
> f6:=(x->exp(poligeneral(1,g1,c5,x)));
        f6 := x -> exp(functie(1, g1, c5, x))
> desen(f6,5,x1,y1);
```




```
> h2:=(x->exp(x));
                          h2 := exp
> y3:=compune(h2,y1,5);
                           y3 := y1
> c7:=patrateMici(5,y3,x1,1,g1);
          c7 := [2.389107848, .1171830602]
> f7:=(x->poligeneral(1,g1,c7,x));
        f7 := x -> poligeneral(1, g1, c7, x)
> desen(f7,5,x1,y3);
```




```
> f8:=(x->ln(poligeneral(1,g1,c7,x)));
      f8 := x -> ln(poligeneral(1, g1, c7, x))
> desen(f8,5,x1,y1);
```



Probleme propuse

Se consideră funcția f dată prin tabelul

x	f(x)
2.8	6.7
2.9	6.9
3.0	7.2
3.1	7.3
3.4	8.4
3.9	8.8
4.0	9.1
4.8	9.8
4.9	10.6
5.2	10.7
5.4	11.1
5.5	11.8
6.2	12.1
7.0	12.4

Să se determine elementele care dau cea mai bună aproximare a lui f, asociate unor sisteme diverse de funcții liniar independente φ_1, φ_2, …φ_{14}. Utilizați procedurile MAPLE definite mai sus.