Lucrarea de laborator nr. 9

I. Scopul lucrării

Aproximarea funcțiilor. Polinoame de interpolare.

II. Conținutul lucrării

1. Polinom de interpolare. Definiție. Eroarea de interpolare.
2. Polinomul Lagrange de interpolare.
3. Polinomul Newton de interpolare de speța I (ascendent).
4. Polinomul Newton de interpolare de speța a II-a (descendent).
5. Polinomul Newton cu diferențe divizate.

III. Prezentarea lucrării

III.1. Polinom de interpolare. Definiție. Eroarea de interpolare.

Fie f : [a, b] $\rightarrow$ **R** o funcție. Se pune problema determinării unei funcții F care să aproximeze funcția f. O asfel de aproximație este necesară în următoarele situații:

1) Când nu se cunoaşte expresia analitică a lui f, ci doar valorile sale într-un număr finit de puncte x_0, x_1, …, x_n din intervalul [a, b].
2) Când expresia analitică a lui f este prea complicată şi calculele efectuate cu ajutorul acesteia ar fi prea dificile.

F trebuie să fie o funcție simplă, uşor de evaluat, de diferențiat şi de integrat. În cele ce urmează F va fi un polinom.

Fie x_0, x_1, …, x_n n+1 puncte distincte din intervalul [a, b], şi $y_i = f(x_i)$ pentru orice i = 0,1,…n. Pentru ca polinomul de grad mai mic sau egal cu n, P_n, să fie un polinom de interpolare pentru f trebuie satisfăcute relațiile:

$$P_n(x_i) = y_i,\ i = 0, 1, \ldots, n \quad (1)$$

Polinomul determinat de condițiile anterioare este unic. Într-adevăr fie

$$P_n(x) = a_nX^n + a_{n-1}X^{n-1} + \ldots + a_1X + a_0$$

Relațiile anterioare conduc la sistemul:

$$(2)\quad \begin{cases} a_0 + a_1x_0 + a_2 x_0^2 + \ldots + a_n x_0^n = y_0 \\ a_0 + a_1x_1 + a_2 x_1^2 + \ldots + a_n x_1^n = y_1 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_0 + a_1x_n + a_2 x_n^2 + \ldots + a_n x_n^n = y_n \end{cases}$$

Determinatul acestui sistem este

$$\Delta = \begin{vmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^n \\ 1 & x_1 & x_1^2 & \ldots & x_1^n \\ & & \ldots\ldots\ldots & & \\ 1 & x_n & x_n^2 & \ldots & x_n^n \end{vmatrix} = \prod_{0\le j<i\le n} (x_i - x_j)$$

(fiind un determinant Vandermonde). Deci $\Delta \neq 0$, şi în consecință sistemul (2) este compatibil determinat, adică polinomul de interpolare P_n este unic determinat. Coeficienții polinomului de interpolare pot fi determinați rezolvând sistemul (2), dar dacă n este mare, atunci volumul de calcul este mare. De aceea se utilizează diferite forme comode ale polinoamelor care conduc la polinoame de interpolare Lagrange, Newton, etc.

Teorema următoare stabileşte eroarea maximă cu care polinomul P_n aproximează funcția f:

Teoremă. *Fie f : [a, b]* $\to$ ***R*** *o funcție de clasă* C^{n+1}. *Fie* $x_0, x_1, \ldots, x_n$ *n+1 puncte distincte din intervalul [a, b], şi* $y_i = f(x_i)$ *pentru orice i = 0,1,…n. Fie* P_n *polinomul de interpolare, adică polinomul de gradul n ce satisface relațiile* $P_n(x_i) = y_i$ *pentru orice i = 0, 1, ..., n . Atunci oricare ar fi x* $\in$ *[a, b], avem:*

$$|f(x) - P_n(x)| \le \frac{\sup_{t\in[a,b]} \left|f^{(n+1)}(t)\right|}{(n+1)!} |(x - x_0)(x - x_1)\ldots(x - x_n)|.$$

III.2. Polinomul Lagrange de interpolare.

Fie f : [a, b] $\to$ **R** o funcție, fie $x_0, x_1, \ldots, x_n$ n+1 puncte distincte din intervalul [a, b], şi $y_i = f(x_i)$ pentru orice i = 0,1,…n. Forma polinomului de interpolare Lagrange este:

$$L_n(x) = y_0 b_0(x) + y_1 b_1(x) + \ldots + y_n b_n(x)$$

unde b_i sunt polinoame de grad n. Punând condițiile $L_n(x_i) = y_i$ pentru orice i = 0, 1, …, n , se obține

$$b_i(x) = \frac{(x-x_0)(x-x_1)\ldots(x-x_{i-1})(x-x_{i+1})\ldots(x-x_n)}{(x_i-x_0)(x_i-x_1)\ldots(x_i-x_{i-1})(x_i-x_{i+1})\ldots(x_i-x_n)}$$

şi deci

$$L_n(x) = \sum_{i=0}^{n} y_i \frac{(x-x_0)(x-x_1)\ldots(x-x_{i-1})(x-x_{i+1})\ldots(x-x_n)}{(x_i-x_0)(x_i-x_1)\ldots(x_i-x_{i-1})(x_i-x_{i+1})\ldots(x_i-x_n)}$$

Algoritm pentru determinarea polinomului Lagrange

Date de intrare:

n – numărul de puncte distincte din [a, b] este n +1

xy – tablou ce conține pe prima linie cele n+1 puncte distincte din intervalul [a, b], iar pe a doua linie valorile corespunzătoare ale funcției.

x_0	x_1	…	x_n
y_0	y_1	…	y_n

x – punctul în care se evaluează polinomul.

Date de ieşire:

lx - valoarea polonimului în x

```
lx : =0;
pentru i = 0,n,1 executa
        t : =yi;
        pentru j = 0,i - 1,1 executa
            t: = t * (x – xj)/ (xi – xj)

        pentru j = i + 1,n,1 executa
            t: = t * (x – xj)/ (xi – xj)


        lx : = lx + t;
```

Procedură MAPLE pentru calculul valorii polinomului Lagrange

```
>Lagrange := proc(n, xy, x)
local lx, i, j, t;
    lx := 0;
```

```
    for i from 0 to n do
        t := xy[2, i];
        for j from 0 to i - 1 do
         t := t*(x - xy[1, j])/(xy[1, i] - xy[1, j])
        od;
        for j from i + 1 to n do
         t := t*(x - xy[1, j])/(xy[1, i] - xy[1, j])
        od;
        lx := lx  + t
    od;
    RETURN(lx)
end;
```

Procedura de mai jos are drept parametri o funcție f un număr n şi un tablou x ce conține n+1 puncte distincte din domeniul de definiție al lui f . Procedura întoarce un tablou xy ce conține pe prima linie cele n+1 puncte distincte, iar pe a doua linie valorile corespunzătoare ale funcției.

```
>tabvalori := proc(f, n, x)
local xy, i;
    xy := array(1 .. 2, 0 .. n);
    for i from 0 to n do
        xy[1, i] := evalf(x[i]);
        xy[2, i] := evalf(f(x[i]))
    od;
    RETURN(xy)
end;
```

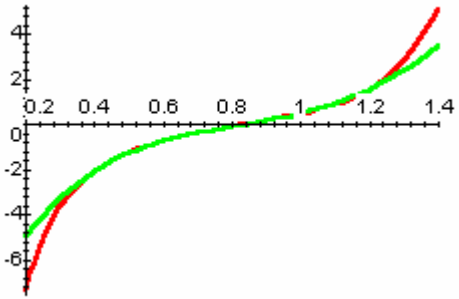
Exemple

```
> f: =(x->ln(x)*sin(x^5+2)/x+x^7/(1+x^(1/2)));
```

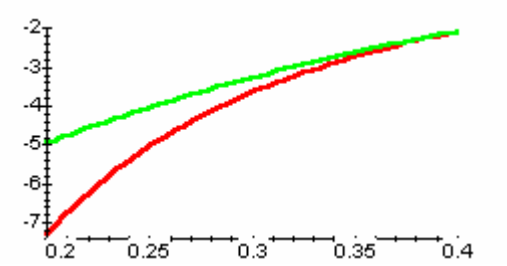
$$x \mapsto \frac{\ln(x)\sin(x^5+2)}{x}+\frac{x^7}{1+\sqrt{x}}$$

```
 > x1:=array(0..4,[0.4,0.6,0.8,1,1.2]);

    x1 := array(0 .. 4, [
      0) = .4
      1) = .6
      (2) = .8
      (3) = 1
    (4) = 1.2
```

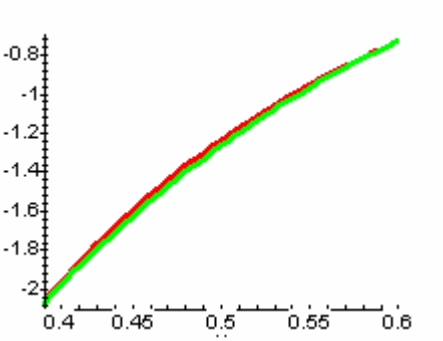
```
    ])
> xy1:=tabvalori(f,4,x1);
                          xy1 := xy
> Lagrange(4,xy1,0.8);

                                -.09207484030

> evalf(f(0.8));

                                 -.0920748403

> Lagrange(4,xy1,0.9);

                                 .1738917140

> evalf(f(0.9));

                                 .1841466329

>with(plots);
> plot([f(x),Lagrange(4,xy1,x)],x=0.2..1.4);
```



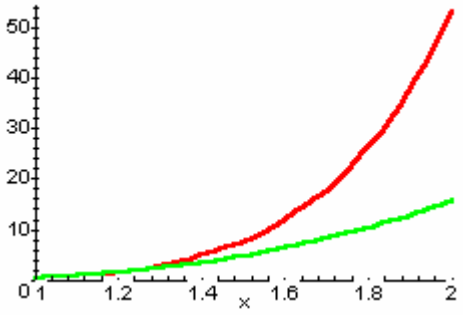

```
> plot([f(x),Lagrange(4,xy1,x)],x=0.2..0.4);
```




```
>plot([f(x),Lagrange(4,xy1,x)],x=0.4..0.6);
```




```
> Lagrange(4,xy1,1.9);
                                  12.91425926
> evalf(f(1.9));
                                  37.92008534
> plot([f(x),Lagrange(4,xy1,x)],x=1..2);
```



```
> x2:=array(0..4,[1,2,3,4,5]);
x2 := array(0 .. 4, [
    (0) = 1
    (1) = 2
    (2) = 3
    (3) = 4
    (4) = 5
    ])
> xy2:=tabvalori(f,4,x2);
                          xy2 := xy
> Lagrange(4,xy2,2);
                        53.20270210
> evalf(f(2));
                        53.20270208
> Lagrange(4,xy2,1.9);
                         10.1492169
> evalf(f(1.9));
                        37.92008534
> Lagrange(4,xy2,19);
                      .2380404278 10^8
> evalf(f(19));
                      .1668013797 10^9
```

III.3. Polinomul Newton de interpolare de speța I (ascendent)

Fie f : [a, b] $\rightarrow$ **R** o funcție, fie $x_0, x_1, \ldots, x_n$ n+1 puncte distincte din intervalul [a, b], şi $y_i = f(x_i)$ pentru orice i = 0,1,…n. Punctele $x_0, x_1, \ldots, x_n$ se presupun echidistante, adică

$x_i = x_0 + ih$, i = 0,1, …, n

h fiind pasul rețelei. Pentru determinarea polinomului de grad mai mic sau egal cu n ce satisface relațiile:

$$P_n(x_i) = y_i,\ i = 0, 1, \ldots, n\ .$$

se pleacă de la un polinom de forma:

$$P_n(x) = C_0 + C_1(x - x_0) + C_2(x - x_0)(x - x_1) + \ldots +$$
$$+ C_n(x - x_0)(x - x_1)\ldots(x - x_{n-1})$$

Coeficienții $C_0, C_1, \ldots, C_n$ se determină luând în considerare condițiile $P_n(x_i) = y_i$, i = 0, 1, …, n . Pentru exprimarea acestor coeficienți este necesară cunoaşterea noțiunilor de putere generalizată şi diferență finită.

Fie f : [a, b] → **R** o funcție, și fie rețeaua cu nodurile echidistante x_0, x_1, …, x_n, având pasul h > 0. Produsul

$$x^{[n]} = x(x-h)\ldots(x-(n-1)h)$$

se numește putere generalizată a lui x. Dacă h ar fi 0, atunci ***puterea generalizată*** ar coincide cu puterea obișnuită. Expresia

$$\Delta f(x) = f(x+h) - f(x)$$

se numește ***diferență finită (la dreapta) de ordinul întâi***. ***Diferențele finite de ordin superior*** se definesc cu ajutorul relației:

$$\Delta^n f(x) = \Delta(\Delta^{n-1} f(x))$$

Se verifică cu ușurință relațiile

$$\Delta(f+g) = \Delta(f) + \Delta(g)$$
$$\Delta(cf) = c\Delta(f)$$
$$\Delta^{(m+n)}(f) = \Delta^m(\Delta^n f).$$

Pentru aplicații, calculul diferențelor finite este comod dacă se construiește tabelul diagonal următor:

$$\begin{array}{ccccccccccc}
f(x_0) & f(x_1) & f(x_2) & f(x_3) & \cdots & f(x_{n-3}) & f(x_{n-2}) & f(x_{n-1}) & f(x_n) \\
\Delta f(x_0) & \Delta f(x_1) & \Delta f(x_2) & \cdots & \Delta f(x_{n-3}) & \Delta f(x_{n-2}) & \Delta f(x_{n-1}) \\
\Delta^2 f(x_0) & \Delta^2 f(x_1) & \cdots & \Delta^2 f(x_{n-3}) & \Delta^2 f(x_{n-2}) \\
\Delta^3 f(x_0) & \cdots & \Delta^3 f(x_{n-3}) \\
& \searrow & \swarrow \\
& & \Delta^n f(x_0)
\end{array}$$

Astfel

$$\Delta f(x_0) = f(x_1) - f(x_0)$$

$$\begin{aligned}
\Delta^2 f(x_0) &= \Delta f(x_1) - \Delta f(x_0) \\
&= f(x_2) - f(x_1) - (f(x_1) - f(x_0)) \\
&= f(x_2) - 2f(x_1) + f(x_0)
\end{aligned}$$

$$\begin{aligned}
\Delta^3 f(x_0) &= \Delta^2 f(x_1) - \Delta^2 f(x_0) \\
&= f(x_3) - 3f(x_2) + 3f(x_1) + f(x_0)
\end{aligned}$$

Se poate demonstra prin inducție că

$$\Delta^n f(x) = \sum_{i=0}^{n} (-1)^i C_n^i f(x + (n-i)h)$$

dar în aplicații se va folosi tabelul diagonal de mai sus.

Pentru determinarea coeficientului C_i al polinomului de interpolare P_n se utilizează diferența finită de ordinul i a lui P_n în x_0. Se obține:

$$P_n(x) = f(x_0) + \frac{\Delta f(x_0)}{1!h}(x-x_0)^{[1]} + \frac{\Delta^2 f(x_0)}{2!h^2}(x-x_0)^{[2]} + \frac{\Delta^3 f(x_0)}{3!h^3}(x-x_0)^{[3]} + \\ + ... + \frac{\Delta^n f(x_0)}{n!h^n}(x-x_0)^{[n]}$$

Expresia polinomului de interpolare de mai sus poartă denumirea de ***polinom Newton de speța I*** (sau ***ascendent***). Expresia polinomului Newton se poate pune sub o formă mai comodă pentru aplicații, dacă se face schimbarea de variabilă

$$t = \frac{x - x_0}{h}$$

şi se ține seama că

$$\frac{(x-x_0)^{[i]}}{h^i} = \frac{(x-x_0)}{h}\frac{(x-x_0-h)}{h}\frac{(x-x_0-2h)}{h}....\frac{(x-x_0-(i-1)h)}{h} \\ = t(t-1)(t-2)...(t-i+1)$$

pentru orice i = 1,2,…,n.

Se obține

$$P_n(x) = P_n(x_0 + t*h) \\ = f(x_0) + \frac{\Delta f(x_0)}{1!}t + \frac{\Delta^2 f(x_0)}{2!}*t(t-1) + \frac{\Delta^3 f(x_0)}{3!}*t(t-1)(t-2) + \\ + ... + \frac{\Delta^n f(x_0)}{n!}*t(t-1)...(t-n+1)$$

Aproximarea funcției f prin polinomul Newton ascendent este avantajosă pentru valorile x din vecinătatea lui x_0, eroarea fiind mică pentru t în modul mic.

Algoritm pentru determinarea polinomului Newton ascendent

Date de intrare:

n – numărul de puncte echidistante din [a, b] este n +1

x0- primul punct din rețea

h – pasul rețelei

y – tablou ce conține valorile funcției în punctele rețelei.

y_0	y_1	…	y_n

$y_i = f(x_i) = f(x0 + ih)$, i = 0,1,…, n

x – punctul în care se evaluează polinomul.

Date de ieşire:

px - valoarea polinomului în x

px : =y_0; t : = (x-x0)/h; u: = 1;
pentru i = 0,n-1,1 executa
    pentru j =n-1,i,0,-1 executa
        y_{j+1} := y_{j+1} –y_i;
pentru i = 1,n,1 executa
    u :=u*(t-i+1)/i;
    px:=px+u*y_i ;

Procedură MAPLE pentru calculul valorii polinomului Newton de speța I (ascendent)

```
>pnewton1 := proc(n, x0, h, y, x)
local t, u, yl, i, j, px;
    yl := array(0 .. n);
    for i from 0 to n do yl[i] := y[i] od;
    px := yl[0];
    t := (x - x0)/h;
    u := 1;
    for i from 0 to n - 1 do
      for j from n - 1 by -1 to i do
            yl[j + 1] := yl[j + 1] - yl[j]
        od
    od;
    for i to n do
      u := u*(t - i + 1)/i;
      px := px + u*yl[i] od;
    RETURN(evalf(px))
end;
```

Procedura tabval de mai jos are drept parametri o funcție f, primul punct din rețeaua de n+1 puncte, x0, şi pasul rețelei, h. Procedura întoarce un tablou unidimensional ce conține valorile funcției în punctele rețelei.

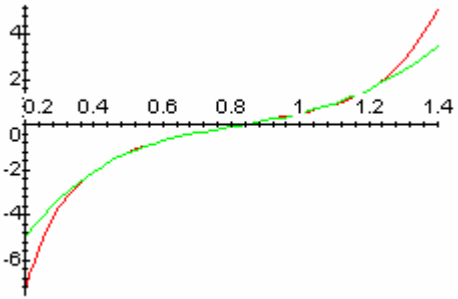
```
>tabval := proc(f, n, x0, h)
local y, i;
    y := array(0 .. n);
    for i from 0 to n do
      y[i] := f(x0 + i*h) od;
    RETURN(evalm(y))
end;
```

Exemple

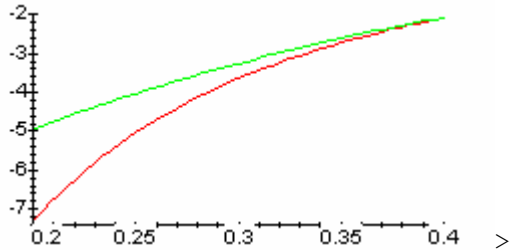
```
> f: =(x->ln(x)*sin(x^5+2)/x+x^7/(1+x^(1/2)));
```

$$x \mapsto \frac{\ln(x)\sin(x^5+2)}{x}+\frac{x^7}{1+\sqrt{x}}$$

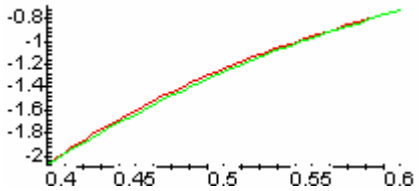
```
> y1:=tabval(f,4,0.4,0.2);
                              y1 := y
> pnewton1(4,0.4,0.2,y1,0.8);
                        -.0920748400
> evalf(f(0.8));
                        -.0920748403
> pnewton1(4,0.4,0.2,y1,0.9);
                         .1738917144
> evalf(f(0.9));
                         .1841466329
> with(plots);
> plot([f(x),pnewton1(4,0.4,0.2,y1,x)],x=0.2..1.4);
```



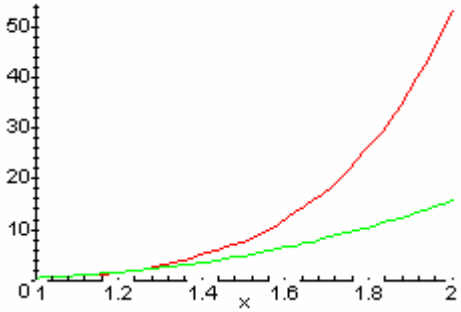

```
>plot([f(x),pnewton1(4,0.4,0.2,y1,x)],x=0.2..0.4);
```




```
>plot([f(x),pnewton1(4,0.4,0.2,y1,x)],x=0.4..0.6);
```




```
> pnewton1(4,0.4,0.2,y1,1.9);
                                  12.91425931
> evalf(f(1.9));
                        37.92008534
> plot([f(x),pnewton1(4,0.4,0.2,y1,x)],x=1..2);
```




```
> y2:=tabval(f,4,1,1);
                          y2 := y
> pnewton1(4,1,1,y2,2);
                        53.20270208
> evalf(f(2));
                        53.20270208
> pnewton1(4,1,1,y2,1.9);
                         10.1492169
> evalf(f(1.9));
                        37.92008534
```

```
> pnewton1(4,1,1,y2,19);
                        .2380404277 10^8
> evalf(f(19));
                        .1668013797 10^9
```

III.3. Polinomul Newton de interpolare de speța II (descendent)

Fie f : [a, b] $\rightarrow$ **R** o funcție, fie $x_0, x_1, \ldots, x_n$ n+1 puncte distincte din intervalul [a, b], şi $y_i = f(x_i)$ pentru orice i = 0,1,…n. Punctele $x_0, x_1, \ldots, x_n$ se presupun echidistante, adică

$x_i = x_0 + ih$, i = 0,1, …, n

h fiind pasul rețelei. Pentru determinarea polinomului de grad mai mic sau egal cu n ce satisface relațiile:

$$P_n(x_i) = y_i,\ i = 0, 1, \ldots, n .$$

se pleacă de la un polinom de forma:

$$P_n(x) = C_0 + C_1(x - x_n) + C_2(x - x_n)(x - x_{n-1}) + \ldots +$$
$$+ C_n(x - x_n)(x - x_{n-1})\ldots(x - x_1)$$

Pentru exprimarea coeficienților $C_0, C_1, \ldots, C_n$ se utilizează diferențele finite la stânga lui y_n.

Expresia

$$\nabla f(x) = f(x) - f(x-h)$$

se numeşte ***diferență finită (la stânga) de ordinul întâi***. ***Diferențele finite la stânga de ordin superior*** se definesc cu ajutorul relației:

$$\nabla^n f(x) = \nabla(\nabla^{n-1} f(x))$$

Se observă că

$$\nabla f(x) = \Delta f(x-h).$$

În aplicații, pentru calculul diferențelor finite la stânga se utilizează tabelul diagonal următor:

$f(x_0)$ $f(x_1)$ $f(x_2)$ $f(x_3)$ ..... $f(x_{n-3})$ $f(x_{n-2})$ $f(x_{n-1})$ $f(x_n)$

$\nabla f(x_1)$ $\nabla f(x_2)$ $\nabla f(x_3)$ ..... $\nabla f(x_{n-2})$ $\nabla f(x_{n-1})$ $\nabla f(x_n)$

$\nabla^2 f(x_2)$ $\nabla^2 f(x_3)$ ..... $\nabla^2 f(x_{n-2})$ $\nabla^2 f(x_n)$

$\nabla^3 f(x_3)$ ..... $\nabla^3 f(x_n)$

$\nabla^n f(x_n)$

Pentru determinarea coeficientului C_i al polinomului de interpolare P_n se utilizează diferența finită de ordinul i a lui P_n în x_0. Se obține:

$$P_n(x) = f(x_0) + \frac{\nabla f(x_0)}{1!h}(x - x_n)^{[1]} + \frac{\nabla^2 f(x_0)}{2!h^2}(x - x_{n-1})^{[2]} + \frac{\nabla^3 f(x_0)}{3!h^3}(x - x_{n-2})^{[3]} +$$
$$+ .... + \frac{\nabla^n f(x_0)}{n!h^n}(x - x_1)^{[n]}$$

Expresia polinomului de interpolare de mai sus poartă denumirea de ***polinom Newton de spețа II*** (sau ***descendent***). Expresia polinomului Newton descendent se poate pune sub o formă mai comodă pentru aplicații, dacă se face schimbarea de variabilă

$$t = \frac{x - x_n}{h}$$

Se obține

$$P_n(x) = P_n(x_n + t*h)$$
$$= f(x_n) + \frac{\nabla f(x_n)}{1!}t + \frac{\nabla^2 f(x_n)}{2!} * t(t+1) + \frac{\nabla^3 f(x_n)}{3!} * t(t+1)(t+2) +$$
$$+ .... + \frac{\nabla^n f(x_n)}{n!} * t(t+1)...(t+n-1)$$

Aproximarea funcției f prin polinomul Newton descendent este avantajosă pentru valorile x din vecinătatea lui x_n, eroarea fiind mică pentru t în modul mic.

Algoritm pentru determinarea polinomului Newton descendent

Date de intrare:

n – numărul de puncte echidistante din [a, b] este n +1
x0- primul punct din rețea
h – pasul rețelei
y – tablou ce conține valorile funcției în punctele rețelei.

y_0	y_1	…	y_n

$y_i = f(x_i) = f(x0 + ih)$, i = 0,1,…, n

x – punctul în care se evaluează polinomul.

Date de ieşire:

px - valoarea polinomului în x

px : $= y_n$; t : = (x-x0-nh)/h; u: = 1;

```
pentru i = 1,n,1 executa
        pentru j =0,n-i executa
                y_j := y_{j+1} –y_j;
pentru i = 1,n,1 executa
        u :=u*(t+i-1)/i;
        px:=px+u*y_{n-i} ;
```

Procedură MAPLE pentru calculul valorii polinomului Newton de speța II (descendent)

```
>pnewton2 := proc(n, x0, h, y, x)
local t, u, yl, i, j, px;
    yl := array(0 .. n);
    for i from 0 to n do yl[i] := y[i] od;
    px := yl[n];
    t := (x - x0 - n*h)/h;
    u := 1;
    for i to n do
        for j from 0 to n - i do
            yl[j] := yl[j + 1] - yl[j]
      od
    od;
    for i to n do
      u := u*(t + i - 1)/i;
      px := px + u*yl[n - i]
    od;
    RETURN(evalf(px))
end;
```

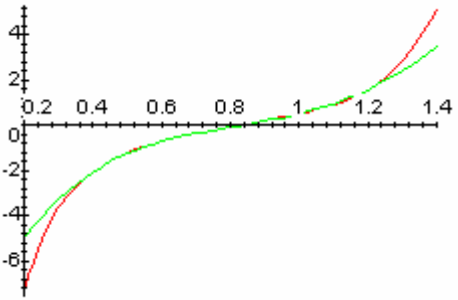
Exemple. În exemplele următoarea procedura tabval este aceeaşi din secțiunea precedentă.

```
> f: =(x->ln(x)*sin(x^5+2)/x+x^7/(1+x^(1/2)));
```

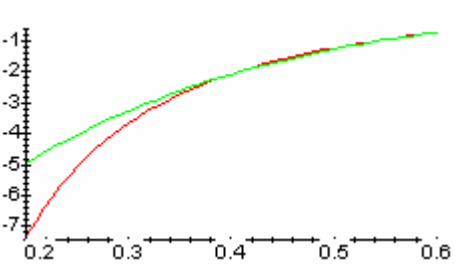
$$x \mapsto \frac{\ln(x)\sin(x^5+2)}{x}+\frac{x^7}{1+\sqrt{x}}$$

```
> y1:=tabval(f,4,0.4,0.2);
```

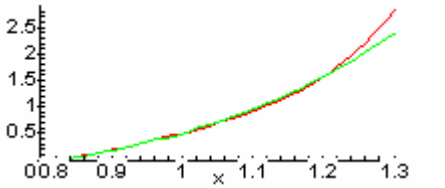
```
                                  y1 := y
> pnewton2(4,0.4,0.2,y1,0.8);
                          -.0920748403
> evalf(f(0.8));
                          -.0920748403
> pnewton2(4,0.4,0.2,y1,0.9);
                           .1738917139
> evalf(f(0.9));
                           .1841466329
> with(plots);
> plot([f(x),pnewton2(4,0.4,0.2,y1,x)],x=0.2..1.4);
```




```
>plot([f(x),pnewton2(4,0.4,0.2,y1,x)],x=0.2..0.6);
```

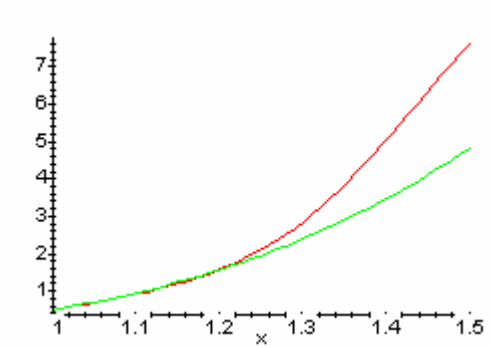



```
> plot([f(x),pnewton2(4,0.4,0.2,y1,x)],x=0.8..1.3);
```




```
> pnewton2(4,0.4,0.2,y1,1.3);
```

```
                          2.388973575
> evalf(f(1.3));
                          2.822982152
> plot([f(x),pnewton2(4,0.4,0.2,y1,x)],x=1..1.5);
```




```
> y2:=tabval(f,4,1,1);

                                    y2 := y

> pnewton2(4,1,1,y2,2);
                          53.20270208
> evalf(f(2));
                          53.20270208
> pnewton2(4,1,1,y2,5.5);
                          43779.27126
> evalf(f(5.5));
                          45511.21075
> pnewton1(4,1,1,y2,55);
                        .2258417985 10^10
> evalf(f(55));
                        .1808934564 10^12
```

III.3. Polinomul Newton cu diferențe divizate

Fie f : [a, b] $\rightarrow$ **R** o funcție, fie x_0, x_1, …, x_n n+1 puncte distincte din intervalul [a, b], şi $y_i = f(x_i)$ pentru orice i = 0,1,…n. Pentru determinarea polinomului de grad mai mic sau egal cu n ce satisface relațiile:

$$P_n(x_i) = y_i,\ i = 0, 1, \ldots, n\ .$$

se pleacă de la un polinom de forma:

$$P_n(x) = C_0 + C_1(x - x_0) + C_2(x - x_0)(x - x_1) + ... +$$
$$+ C_n(x - x_0)(x - x_1)...(x - x_{n-1})$$

Pentru exprimarea coeficienţilor C_0, C_1, …, C_n se utilizează diferenţele divizate ale lui.

Expresia

$$f(x_i, x_j) = \frac{f(x_j) - f(x_i)}{x_j - x_i}, \quad i \neq j$$

se numeşte ***diferenţă divizată de ordinul întâi***. ***Diferenţele divizate de ordin 2*** se definesc cu ajutorul diferenţelor divizate de ordinul întâi:

$$f(x_i, x_j, x_k) = \frac{f(x_j, x_k) - f(x_i, x_j)}{x_k - x_i}$$

Cunoscând diferenţele divizate de ordinul m, diferenţele divizate de ordinul m+1 se definesc prin:

$$f(x_{i-1}, x_i, x_{i+1}, ..., x_{i+m}) = \frac{f(x_i, x_{i+1}, ..., x_{i+m}) - f(x_{i-1}, x_i, ..., x_{i+m-1})}{x_{i+m} - x_{i-1}}$$

Polinomul P_n are forma

$$P_n = f(x_0) + f(x_0, x_1)(x - x_0) + f(x_0, x_2)(x - x_0)(x - x_1)$$
$$+ .... + f(x_0, x_1, ..., x_n)(x - x_0)(x - x_1)...(x - x_{n-1})$$

Algoritm pentru determinarea polinomului Newton descendent

Date de intrare:

n – numărul de puncte din [a, b] este n +1

xy – tablou ce conţine pe prima linie cele n+1 puncte distincte din intervalul [a, b], iar pe a doua linie valorile corespunzătoare ale funcţiei.

x_0	x_1	…	x_n
y_0	y_1	…	y_n

$y_i = f(x_i)$, i = 0,1,…, n

x – punctul în care se evaluează polinomul.

Date de ieşire:

px - valoarea polinomului în x

```
px : =y0; u: = 1;
pentru i = 0,n-1,1 executa
        pentru j =n-1,i,0,-1 executa
```

$y_{j+1} := (y_{j+1} - y_j)/(x_{j+1} - x_{j-i})$;

pentru i = 1,n,1 executa
    u :=u*(x-x_{i-1});
    px:=px+u*y_i ;

Procedură MAPLE pentru calculul valorii polinomului Newton cu diferențe divizate

```
>pnewtond := proc(n, xy, x)
local u, y, i, j, px;
    y := array(0 .. n);
    for i from 0 to n do
      y[i] := xy[2, i]
    od;
    px := y[0];
    u := 1;
    for i from 0 to n - 1 do
      for j from n - 1 by -1 to i do
            y[j + 1] := evalf(
               (y[j+1]- y[j]) /
                  (xy[1, j + 1] - xy[1, j - i]))
      od
    od;
    for i to n do
      u := u*(x - xy[1, i - 1]);
      px := px + u*y[i]
    od;
    RETURN(evalf(px))
end;
```

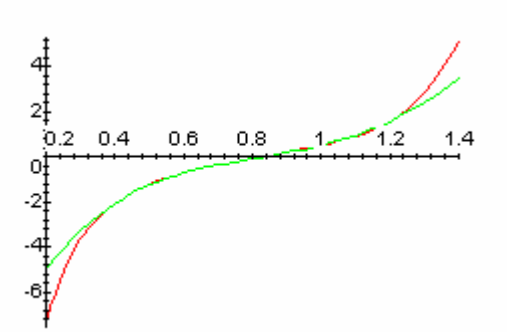
Exemple. Procedura tabvalori din exemplele următoare a fost definită anterior în secțiunea referitoare la polinomul Lagrange.

```
> f: =(x->ln(x)*sin(x^5+2)/x+x^7/(1+x^(1/2)));
```

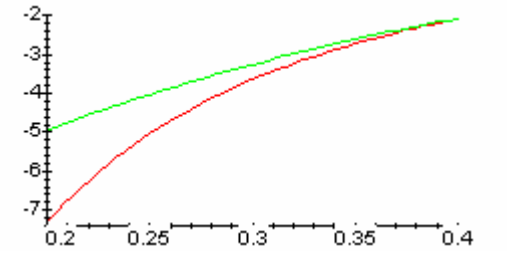
$$x \mapsto \frac{\ln(x)\sin(x^5+2)}{x} + \frac{x^7}{1+\sqrt{x}}$$

```
> x1:=array(0..4,[0.4,0.6,0.8,1,1.2]);
```

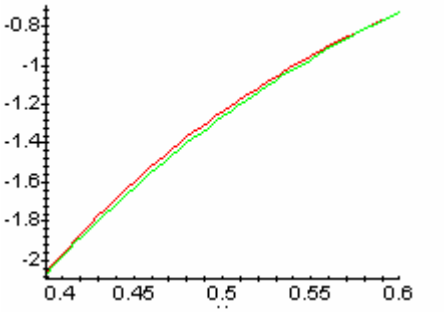
```
x1 := array(0 .. 4, [
     (0) = .4
     (1) = .6
     (2) = .8
     (3) = 1
     (4) = 1.2
     ])

> xy1:=tabvalori(f,4,x1);
                          xy1 := xy
> pnewtond(4,xy1,0.8);
                        -.0920748400
> evalf(f(0.8));
                        -.0920748403
> pnewtond(4,xy1,0.9);
                         .1738917145
> evalf(f(0.9));
                         .1841466329
> plot([f(x),pnewtond(4,xy1,x)],x=0.2..1.4);
```



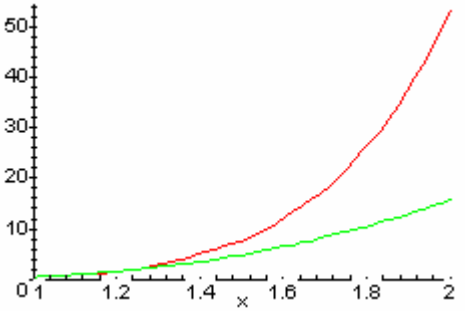

```
> plot([f(x),pnewtond(4,xy1,x)],x=0.2..0.4);
```



```
>plot([f(x),pnewtond(4,xy1,x)],x=0.4..0.6);
```




```
> pnewtond(4,xy1,1.9);
                        12.91425930
> evalf(f(1.9));
                        37.92008534
> plot([f(x),pnewtond(4,xy1,x)],x=1..2);
```




```
> x2:=array(0..4,[1,2,3,4,5]);
x2 := array(0 .. 4, [
     (0) = 1
     (1) = 2
     (2) = 3
     (3) = 4
     (4) = 5
    ])
> xy2:=tabvalori(f,4,x2);
                         xy2 := xy
> pnewtond(4,xy2,2);
                        53.20270208
```

```
> evalf(f(2));
                      53.20270208
> pnewtond(4,xy2,1.9);
                      10.14921692
> evalf(f(1.9));
                      37.92008534
> pnewtond(4,xy2,19);
                    .2380404277 10^8
> evalf(f(19));
                    .1668013797 10^9
```

Probleme propuse

Fie funcția f: $(0,\infty) \to R$, definită prin f(x) = lg(x). Se presupune că se cunosc valorile funcției în punctele echidistante $x_0 = 1000$, $x_1 = 1010,\ldots, x_5 = 1050$. Să se găsească, folosind un polinom Newton de gradul trei, valoarea funcției în x=1044 şi în x = 1006. Să se compare rezultatele obținute folosind diversele polinoame Newton.

Se dă tabelul de valori

x	f(x)
1000	3.0000000
1010	3.0043214
1020	3.0086002
1030	3.0128372
1040	3.0170333
1050	3.30211893