

## CALCULE NUMERICE CU POLINOAME

### 9.1 Definirea polinoamelor

Forma generală a polinoamelor, în scrierea algebrică:  $f(x) = a_nx^n + a_{n-1}x^{n-1} + \dots + a_2x^2 + a_1x + a_0$ , iar reprezentarea acestuia, în MATLAB, este dată prin vectorul linie  $f=[a_n, a_{n-1}, \dots, a_2, a_1, a_0]$ , în care elementele sunt coeficienții polinomului de grad  $n$ . De exemplu, polinomul  $p(x)=x^4+7x^3+x-8$  se definește astfel:



```
>> p=[1, 7, 0, 1, -8]
```

Fie polinoamele:  $f(x) = x^3+2x^2+0,5x-6,5$ ;  $h(x)=x^5+2x^2-6\sqrt[3]{5}$ . Aceste polinoame se scriu, în MATLAB:



```
>> f = [1, 2, 0.5, -6.5];  
>> h=[1, 0, 0, 2, 0, -6*(5^(1/3))];
```



Obs.: În polinomul  $h(x)$  coeficienții lui  $x^4$ ,  $x^3$  respectiv  $x$ , care lipsesc, sunt înlocuiți cu valoarea 0 (zero). Astfel, MATLAB va defini un polinom de grad inferior celui real (de exemplu, dacă  $h(x)$  ar fi introdus de la tastatură numai cu coeficienții  $h=[1,2,-6*(5^(1/3))]$  acesta reprezintă polinomul  $h(x)=x^2+2x-6\sqrt[3]{5}$  )

### 9.2 Adunarea și scăderea polinoamelor

Acste operații se pot executa numai dacă polinoamele au același grad. De aceea, polinoamele trebuie transformate, prin extindere spre stânga, astfel încât acestea să fie de același grad. După uniformizarea gradului, adunarea algebrică a polinoamelor se realizează, în MATLAB, ca o însumare a doi vectori de același dimensiune.

Să se calculeze suma și diferența polinoamelor:  $g(x) = x^4-5x^2+6x-1$  și  $h(x) = x^3+3x-2$ .

Întrucât în MATLAB, polinomul  $h(x)$  are gradul III, acesta va fi „adus” la gradul IV corespunzător polinomului  $g(x)$  (gradul 4), prin impunerea unui coeficient „fals” (zero):  
Astfel:



```
>> g = [1 0 -5 6 -1]; h = [0 1 0 3 -2];  
>> suma = g+h  
suma =  
1 1 -5 9 -3
```

```

» diferența = g-h
diferența =
      1          -1          -5           3          1

```

### 9.3 Produsul polinoamelor

Produsul a două polinoame se realizează, în MATLAB, folosind funcția **conv(a, b)**, unde **a**, **b** sunt vectorii coeficienților polinoamelor care se înmulțesc. De exemplu, produsul polinoamelor  $(x^2+x-2)*(x+1)$  se determină astfel, în MATLAB,:



```

>> g1 = [1 1 -2]; g2 = [1 1];
>> g = conv(g1, g2)
g = [1 2 -1 -2]

```

care, în formă algebrică ușuală, se scrie :  $g(x) = x^3 + 2x^2 - x - 2$ .



*Notă: definirea simbolică a unui polinom, se realizează cu funcția MATLAB **poly2sym()**. Astfel, forma simbolică a polinomului definit prin vectorul **g** = [1 2 -1 -2] se determină cu succesiunea de instrucțiuni:*



```

>> g = [1 2 -1 -2];
>> g_x=poly2sym(g)
g_x =
x^3+2*x^2-x-2

```

### 9.4 Câtul polinoamelor

Împărțirea a două polinoame se realizează cu funcția MATLAB **[c, r] = deconv(a, b)**, în care : **c** - vectorul coeficienților polinomului cât; **r** - vectorul coeficienților polinomului rest. Polinoamele **c(x)** și **r(x)** respectă teorema împărțirii cu rest  $p(x) = d(x)*c(x) + r(x)$ . De exemplu, câtul și restul împărțirii polinoamelor  $h_1(x) = x^3 + 2x^2 - 2$  și  $h_2(x) = x^2 + x$ , este:



```

>> h1=[1 2 0 -2]; h2=[1 1 0];
>> [c, r] = deconv(h1, h2)
c =
      1          1
r =
      0          0         -1         -2

```



```
>> c_x=poly2sym(c)
c_x =
x+1
>> r_x=poly2sym(r)
r_x =
-x-2
```

## 9.5 Determinarea rădăcinilor unui polinom

Determinarea rădăcinilor unui polinom, se realizează folosind funcția MATLAB **r = roots(f)**, în care, **f** - vector linie al coeficienților polinomului

Fie polinomul  $f(x) = x^3 - 2x^2 - 3x + 10$ . Să se determine rădăcinile ecuației  $f(x)=0$ , folosind funcția MATLAB specifică **roots()**:



```
>>f = [1 -2 -3 10]
>>r=roots(f)
r =
2+i
2-i
-2
```

## 9.6 Definirea polinomului, dacă se cunosc rădăcinile

Pentru determinarea unui polinom, dacă se cunosc rădăcinile, se utilizează funcția MATLAB **poly(z)** în care **z** este **vectorul- coloană** al rădăcinilor.

Fie rădăcinile unui polinom :  $x_1=2$ ;  $x_2=3$ ;  $x_3=4$ . Să se determine polinomul cu rădăcinile  $x_1$ ;  $x_2$ ;  $x_3$ .



```
>> b=[2;3;4];
>> p=poly(b)
p =
1    -9    26   -24
>> p_x=poly2sym(p)
p_x =
x^3-9*x^2+26*x-24
```

## 9.7 Calculul valorii numerice a unui polinom

-în MATLAB, se declară, mai întâi, valoarea variabilei, apoi se definește polinomul de evaluat, folosind regulile standard:



```
>> x = 2;
>> f = 3*(x^4)-5*(x^3)+3*x-1
f =
    13
```

-dacă variabila este un vector, calculul valorii polinomului, în acest caz, se poate realiza numai dacă expresia polinomului este introdusă de la tastatură utilizând regulile pentru calculul cu tablouri de valori (variabila este considerată de MATLAB un tablou de valori, deci se va folosi caracterul „.” –**punct**, plasat “în față” operatorului de ridicare la putere, pentru operațiile în care este implicată variabila vector sau matrice) altfel, MATLAB afișează un mesaj de eroare:



```
>> x= [1 2 0];
>> f = 3*x^4-5*x^3+3*x-1
??? Error using ==> ^
Matrix must be square.
```

??? Eroare la utilizarea ==> ^
Matricile trebuie sa fie pătratice



```
>> x= [1 2 0];
>> f = 3*x.^4-5*x.^3+3*x-1
f =
    0      13      -1
```

-dacă variabila este o matrice și se cere valoarea polinomului  $f(x) = 3 \cdot x^4 - 5 \cdot x^3 + 3 \cdot x - 1$  când  $x = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$ , se obține:



```
>> x=[1,2;3,4];
>> f = 3*x^4-5*x^3+3*x-1
f =
    414      605
    908     1323
```

## 9.8 Evaluarea polinoamelor cu funcția MATLAB polyval(p, q)

Evaluarea expresiilor polinomiale, în MATLAB, se poate realiza folosind funcția **polyval(p,q)**, în care: **p** - vectorul linie al coeficienților polinomului; **q** – vectorul , respectiv, matricea în care se evaluatează polinomul **p**. De exemplu, dacă se cere să se determine valoarea polinomului  $p(x)=3x^4+x^3+2x^2-1$  în punctele vectorului  $q=[1,2,3,4,0]$ , respectiv ale

matricelor  $Q = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$  și  $R = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$ , folosind funcția specifică MATLAB

**polyval()**:



```
>> p = [3 1 2 0 -1];
>> q = [1 2 3 4 0]; Q = [1 2 ; 3 4]; R = [1 2 3 ; 3 4 5];
>> f1=polyval(p,q)
f1=
      5   63   287   863   -1
```



```
>> f2=polyval(p,Q)
f2 =
      5   63
     287   863
```



```
>> f3=polyval(p,R)
f3 =
      5       63       287
     287       863      2049
```

## 9.9 Evaluarea pe intervale

Dacă se cere evaluarea unui polinom pe un interval în **n** puncte:

- se împarte intervalul în **n** puncte, după o regulă prestabilită;

- se evaluatează polinomul folosind funcția MATLAB **polyval()**.

De exemplu, dacă se cere să se evaluateze polinomul  $p(x)=3x^4+x^3+2x^2-1$ , pentru valorile  $x \in [0,5]$ , în punctele fixe distanțate la **0,2** unități, în MATLAB se obțin următoarele rezultate:



```
>> x = 0:0.2:5; p = [3 1 2 0 -1];
>> f = polyval(p, x)
```

```

f =
1.0e+003 *
Columns 1 through 6
-0.0010 -0.0009 -0.0005 0.0003 0.0020 0.0050
Columns 7 through 12
0.0098 0.0172 0.0279 0.0428 0.0630 0.0896
Columns 13 through 18
0.1239 0.1672 0.2210 0.2870 0.3668 0.4623
Columns 19 through 24
0.5755 0.7083 0.8630 1.0419 1.2473 1.4819
Columns 25 through 26
1.7482 2.0490

```

## 9.10 Calculul derivatei unui polinom

Derivata polinomului se calculează folosind funcția MATLAB **polyder()**. De exemplu dacă se cere să se calculeze derivata polinomului  $A(x) = x^3 + 2x^2 - x + 2$ , în MATLAB se obțin următoarele rezultate:



```

» A = [1 2 -1 2];
» D = polyder(A)
D =
3 4 -1
D_x=poly2sym(D)
D_x =
3*x^2+4*x-1

```

## 9.11 Calculul derivatei produsului, a două polinoame

Prin definiție, derivata produsului a două polinoame este tot un polinom. Așadar, calculul acestei derivate implică folosirea aceleiași funcții MATLAB **polyder()**, în care, argumentul de intrare (înscris, întotdeauna, între paranteze rotunde), este format din vectorii linie ai coeficienților celor două polinoame: **D = polyder(A, B)**. De exemplu, dacă se cere să se determine derivata produsului  $(x^3 + 2x^2 - x + 2)(x - 1)$ , se obțin următoarele rezultate:



```

» A = [1 2 -1 2]; B = [1 -1];
» D = polyder(A, B)
D =
4 3 -6 3
» D_x=poly2sym(D)
D_x =
4*x^3+3*x^2-6*x+3

```

## 9.12 Calculul derivatei câtului, a două polinoame

Pentru calculul derivatei câtului a două polinoame se folosește funcția MATLAB  $[M, N] = \text{polyder}(A, B)$ , în care,  $-M$  și  $N$  sunt polinoamele de la numărătorul respectiv numitorul expresiei derivate;  $-A$  și  $B$  sunt polinoamele de la numărătorul respectiv numitorul expresiei ce urmează a fi derivată.

De exemplu, dacă se cere să se determine derivata expresiei rationale:  $\frac{x^3 + 2x^2 - x + 2}{x - 1}$ , atunci în MATLAB se obțin următoarele rezultate:



```
>> A = [1 2 -1 2]; B = [1 -1];
>> [M, N] = polyder(A, B)
M =
    2           -1           -4           -1
N =
    1           -2            1
```



```
>> M_x=poly2sym(M)
M_x =
    2*x^3-x^2-4*x-1
>> N_x=poly2sym(N)
N_x =
    x^2-2*x+1
```

ceea ce, în format algebric uzual, se scrie:  $\frac{2x^3 - x^2 - 4x - 1}{(x - 1)^2}$ .

## 9.13 Descompunerea în fracții simple

Întrucât descompunerea în fracții simple are ca rezultat trei vectori, funcția MATLAB **residue()**, se definește cu parametri de ieșire. Funcția MATLAB pentru descompunerea în fracții simple este:

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

în care:

- r - vectorul coloană al reziduurilor;
- p - vectorul coloană al polilor;
- k - vectorul linie al polinomului cât ( $A/B$ )

De exemplu, următoarea expresie:  $\frac{A(x)}{B(x)} = \frac{x^3 - 6x^2 + 11x - 6}{x^2 - 9x + 20}$ , descompusă în fracții simple, în MATLAB, devine:



```

» A = [1 -6 11 -6];B = [1 -9 20];
» [reziduuri_partiale, polii_fractiei, coeficientii_catului] = residue(A,B)

reziduuri_partiale =
              24
              -6

polii_fractiei =
              5
              4

coeficientii_catului =
              1      3

```

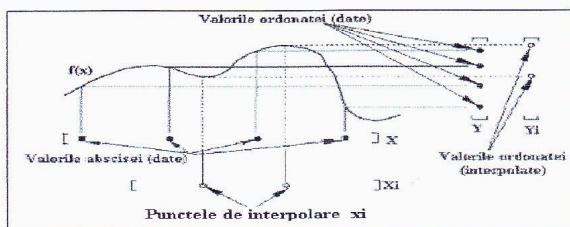
Acest rezultat poate fi transcris, în format algebric canonic, astfel:

$$\frac{x^3 - 6x^2 + 11x - 6}{x^2 - 9x + 20} = \frac{24}{x - 5} + \frac{(-6)}{x - 4} + (x + 3)$$

## 10.1 Interpolarea liniară a datelor

Funcția MATLAB **interp1()** interpolează între punctele date, găsind valorile unei funcții uni-dimensionale  $f(x)$  pe baza acestor puncte. În MATLAB sunt disponibile mai multe variante ale funcției **interp1()**:

**yi = interp1(x,Y,xi)**  
**yi = interp1(Y,xi)**  
**yi = interp1(x,Y,xi,metoda)**



Varianta **yi = interp1(x,Y,xi,metoda)**, permite realizarea interpolării folosind metode alternative:

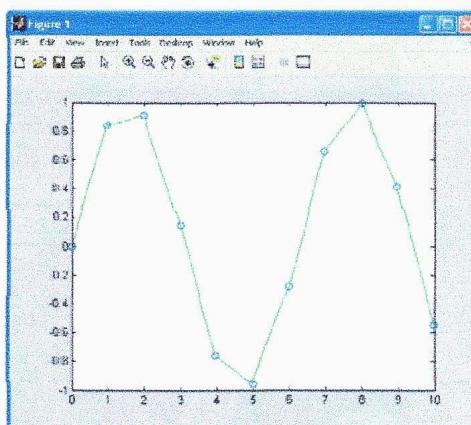
- |           |  |
|-----------|--|
| 'nearest' | -interpolarea prin puncte apropiate vecine |
| 'linear'  | -interpolare liniară (implicită)           |
| 'spline'  | -interpolare Spline cubică                 |
| 'pchip'   | -interpolare cubică Hermite                |
| 'cubic'   | -Similară interpolării 'pchip'             |
| 'vScubic' | -interpolare cubică, utilizată în MATLAB 5 |

De exemplu, dacă se cere să se genereze o curbă sinusoidală „primară”, când vectorul variabilei, x, ia valorile în intervalul **[0,10]**, cu « pasul » egal cu unitatea, apoi să se interpolateze peste un număr mai mare de abscise, vectorul variabilei, x, ia valorile în intervalul **[0,10]**, cu « pasul » **0.1**, se folosește următoarea succesiune de instrucțiuni MATLAB:

### Varianta A

- vectorul variabilei, x, ia valorile în intervalul **[0,10]**, cu « pasul » unitar :

```
>>x=0:10;
>>y=sin(x);
>>xi=0:0.25:10;
>>yi=interp1(x,y,xi);
>>plot(x,y,'o',xi,yi)
```



### Varianta B

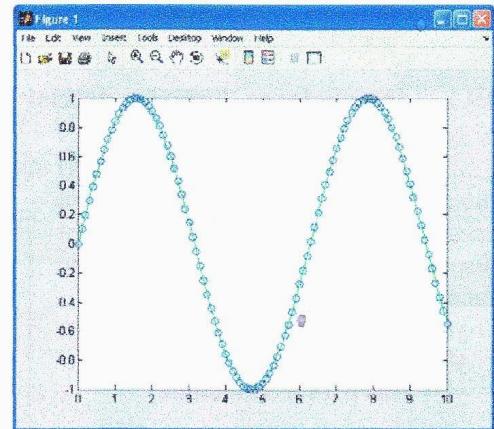
- vectorul variabilei,  $x$ , ia valorile în intervalul [0,10], cu « pasul » 0.1 :



```

>>x=0:0.1 :10;
>>y=sin(x);
>>xi= 0:0.25:10;
>>yi=interp1(x,y,xi);
>>plot(x,y,'o',xi,yi)

```



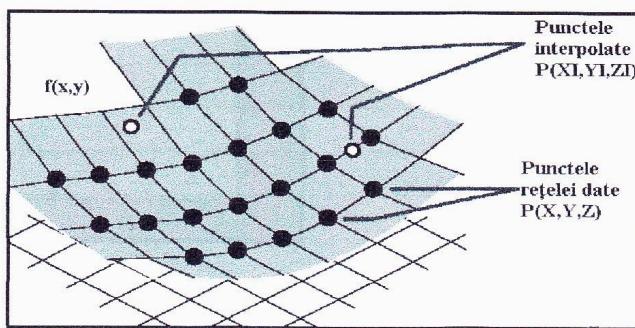
## 10.2 Interpolarea bidimensională a datelor

Interpolarea bi-dimensională se realizează prin funcția MATLAB **interp2()** care se poate apela în următoarele variante :

**ZI = interp2(X,Y,Z,XI,YI)**  
**ZI = interp2(Z,XI,YI)**  
**ZI = interp2(Z,de\_n\_ori)**  
**ZI = interp2(X,Y,Z,XI,YI,metoda)**

- ❖ Varianta **ZI = interp2(Z,de\_n\_ori)** specifică repetarea interpolării bidimensionale, **de n ori**.
- ❖ Varianta **ZI = interp2(X,Y,Z,XI,YI,metoda)** specifică **metoda** alternativă de interpolare:
  - **'linear'** - pentru interpolare bi-lineară
  - **'nearest'** - pentru interpolarea pe vecinătăți
  - **'spline'** - pentru interpolarea spline cubică
  - **'cubic'** - pentru interpolarea bi-cubică

**Concluzie :** Oricare dintre aceste variante interpolează între punctele unei rețele bidimensionale, fiind găsite valorile unei funcții  $f(x,y)$  care face conexiunea între punctele intermediare.



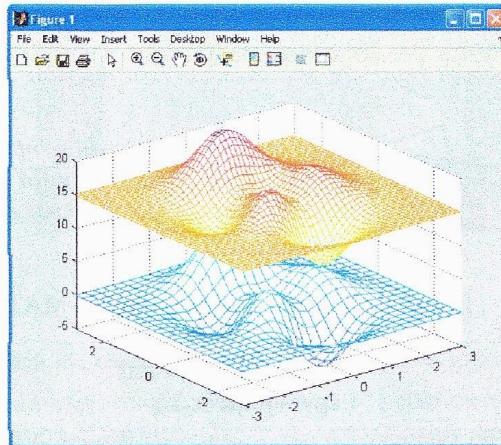
*Interpolarea bi-dimensională*

De exemplu, dacă se cere interpolarea, pe o retea  $[X, Y]$  (definită în MATLAB, prin funcția specifică **meshgrid()** ), a funcției MATLAB **peaks()** (această funcție, de două variabile, este obținută prin translatarea și scalarea distribuțiilor Gauss), definită pe intervalul  $[-3, 3]$  cu « pasul » **0.25**, pe o rețea  $[XI, YI]$  mai fină decât  $[X, Y]$  , pe același interval, dar cu « pasul » **0.125**.

Soluția este :



```
>> [X, Y] = meshgrid(-3:0.25:3);
>> Z = peaks(X, Y);
>> [XI, YI] = meshgrid(-3:0.125:3);
>> ZI = interp2(X, Y, Z, XI, YI);
>> mesh(X, Y, Z), hold,
>> mesh(XI, YI, ZI+15)
>> hold off
>> axis([-3 3 -3 3 -5 20])
```



### 10.3 Interpolarea tridimensională a datelor

Interpolarea tridimensională, în MATLAB, se realizează cu funcția **interp3()** care se poate apela în următoarele variante :

```
>> VI = interp3(X, Y, Z, V, XI, YI, ZI)
>> VI = interp3(V, XI, YI, ZI)
>> VI = interp3(V, de_n_ori)
>> VI = interp3(..., metoda)
```

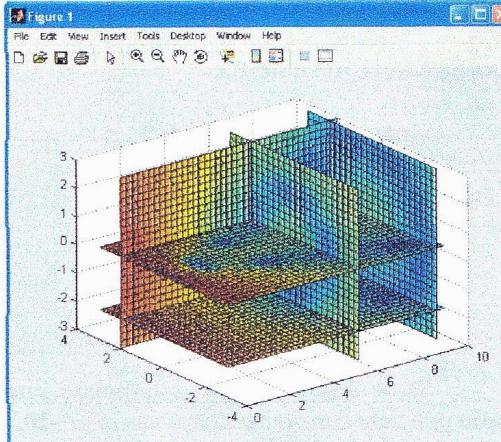
Oricare dintre aceste variante realizează interpolarea prin căutarea valorilor matricei **vi**, reprezentând valorile intermediare ale funcției tridimensionale **v** în punctele rețelei date de matricile **xi**, **yi** și **zi**, ordonate monoton.

De exemplu, folosind și funcția MATLAB **slice()**, se interpolează datele rețelei tridimensionale  $[x, y, z, v] = \text{flow}(10)$ , pentru a găsi valorile corespunzătoare punctelor  $[xi, yi, zi]$  :



```
>> [x, y, z, v] = flow(10);
>> [xi, yi, zi] = meshgrid(0.1:0.25:
10, -3:0.25:3, ...
-3:0.25:3);

>> vi = interp3(x, y, z, v, xi, yi, zi);
>> slice(xi, yi, zi, vi,
[6 9.5], 2, [-2 .2])
```



## 10.4 Interpolarea spline- cubică a datelor

Curba **spline cubică** este o curbă netedă, definită de un set de polinoame de gradul trei. Curba, « de conectare lină », dintre fiecare set de puncte este definită printr-un polinom de gradul trei, determinat astfel încât să conducă la tranziții netede de la un polinom de gradul III la altul. Spre exemplu, trei puncte sunt conectate cu două curbe de gradul trei, diferite, ceea ce constituie o funcție netedă între toate cele trei puncte.



**Notă:** De reținut: diferența dintre metoda de interpolare **lineară** și interpolarea **spline**, constă în aceea că metoda **spline** permite « extrapolarea », adică se pot estima valori care nu se găsesc în setul ordonat.

Interpolarea spline cubică se face în MATLAB cu funcția:

```
xy = spline(x,Y,xx)
sp = spline(x,Y)
```

unde: **x** este obligatoriu un vector ; **Y** poate fi scalar, vector sau rețea structurală; **xx** poate fi scalar, vector sau rețea structurală. Parametrii **x**, **Y** și **xx** trebuie să respecte condițiile dimensionale impuse de funcția de interpolare **spline()**.

De exemplu, dacă se cere să se determine valoarea **y1** de interpolare spline, pentru **x1=2.7**, fiind date: **x=[0,1,2,3,4,5]** ; **y=[0,20,60,68,77,100]** , se folosește următoarea succesiune de instrucțiuni MATLAB :



```
>>x=[0,1,2,3,4,5] ; y=[0,20,60,68,77,100] ;
>>y1=spline(x,y,2.7)
y1=67.4340
```

## 10.5 Interpolarea multiplă a datelor

Funcția MATLAB utilizată:

```
>>interp1(x,y,xi,'metoda')
```

în care, ‘**metoda**’, este cuvântul cheie:

‘**linear**’ – interpolare lineară;  
‘**spline**’ – interpolare spline;  
‘**cubic**’ – interpolare cubică.

Să se determine prin interpolare lineară, spline și cubică, și apoi să se reprezinte grafic punctele corespunzătoare valorilor **-10, -9.75, -4.25, 4.15, 0.1, 6.35** dacă **x=-10:0.5:10**, iar **y=(log(abs(x+12))+2)./(exp(x)+12)**.



```
>>x=-10:0.5:10;
>>y=(log(abs(x+12))+2)./(exp(x)+12);xi=[-10,-9.75,-4.25,4.15,0.1,6.35];
>>yi1=interp1(x,y,xi,'linear');
>>yi2=interp1(x,y,xi,'spline');
>>yi3=interp1(x,y,xi,'cubic');
>>plot(x,y,'r-',xi,yi1,'m:o',
xi,yi2,'k--s',xi,yi3,'b-.+')
```

