Calcul Numeric

Prof. Bogdan Gavrea

ISA 2018-2019

Șiruri de operatori de aproximare

Operatori liniari și pozitivi

Definiție (Operatori liniari și pozitivi)

Un operator $L : C[a,b] \to C[a,b]$ *se numește operator liniar și pozitiv dacă pentru orice* $f, g \in C[a,b]$ *și* $\alpha, \beta \in \mathbb{R}$ *avem*

- **liniaritate:** $L(\alpha f + \beta g) = \alpha L(f) + \beta L(g)$,
- **pozitivitate:** $L(f) \geq 0$, $\forall f \in C[a,b]$, $f \geq 0$.

Exemple.

a) Fie $\Delta : a = x_0 < x_1 < \ldots < x_n = b$ o diviziune a intervalului $[a,b]$. Atunci operatorul: $s_\Delta : C[a,b] \to C[a,b]$ este un operator liniar și pozitiv. Aici $s_\Delta(f)$ reprezintă spline-ul liniar de interpolare al funcției f relativ la diviziunea Δ.

b) Operatorul $L : C[a,b] \to C[a,b]$, dat prin

$$L(f)(x) = \frac{b-x}{b-a} f(a) + \frac{x-a}{b-a} f(b)$$

este un operator liniar și pozitiv. Se observă că $L(f) = L_1(f; a, b)$ (polinomul de interpolare Lagrange de gradul 1 pentru nodurile $x_0 = a$, $x_1 = b$).

Teorema Popoviciu-Bohmann-Korovkin.

Teoremă (Popoviciu-Bohmann-Korovkin)

Fie $(L_n)_{n\in\mathbb{N}}$, $L_n : C[a,b] \to C[a,b]$ un șir de operatori liniari șI pozitivi. Condiția necesară și suficientă pentru ca

$$\lim_{n\to\infty} ||f - L_n(f)||_\infty = 0, \ \ \forall f \in C[a,b]$$

este ca

$$\lim_{n\to\infty} ||e_k - L_n(e_k)||_\infty = 0, \ \ k = 0,1,2,$$

unde $e_k(x) = x^k$, $k \in \mathbb{N}$.

Definiție (Operatorul Bernstein)

Fie $f \in C[0,1]$ și $n \in \mathbb{N}$. Se numește polinom Bernstein de grad n, polinomul

$$B_n(f)(x) = \sum_{k=0}^{n} p_{n,k}(x) f\left(\frac{k}{n}\right),$$

unde polinoamele de grad (exact) n, $p_{n,k}(x) = \binom{n}{k} x^k (1-x)^{n-k}$, $k = \overline{0,n}$ se numesc polinoamele fundamentale Bernstein de grad n.

Proprietăți ale operatorilor Bernstein

Teoremă (Convergența operatorilor Bernstein)

Șirul de operatori Bernstein $(B_n)_{n\in\mathbb{N}}$ *este un sir de operatori de aproximare, i.e.,*

$$\lim_{n\to\infty} ||B_n(f) - f||_\infty = 0, \forall f \in C[0,1].$$

Observație. Avem:

$$B_n(e_0) = e_0, \quad B_n(e_1) = e_1, \quad B_n(e_2)(x) = x^2 + \frac{x(1-x)}{n}.$$

Teoremă (Pastrarea monotoniei si convexității)

Fie $f \in C[0,1]$.

a) *Dacă* f *este o funcție monotonă pe* $[0,1]$ *atunci* $B_n(f)$ *este o funcție monotonă și de aceeași monotonie cu* f **(păstrarea monotoniei)**;

b) *Dacă* f *este o funcție convexă (concavă) pe* $[0,1]$ *atunci* $B_n(f)$ *este o funcție convexă (concavă) pe* $[0,1]$ **(păstrarea convexității)**.

Algoritmul Popoviciu-Casteljau

Din identitatea:

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1},$$

obținem identitatea:

$$p_{n,k}(x) = (1-x)p_{n-1,k}(x) + xp_{n-1,k-1}(x).$$

Algoritmul Popoviciu-Casteljau

Din identitatea:

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1},$$

obținem identitatea:

$$p_{n,k}(x) = (1-x)p_{n-1,k}(x) + xp_{n-1,k-1}(x).$$

Mulțimea $\mathcal{B} = \{p_{n,0}(x), ..., p_{n,n}(x)\}$ reprezintă o bază în Π_n. Fie $P \in \Pi_n$ cu reprezentarea

$$P(x) = \sum_{k=0}^{n} a_k p_{n,k}(x).$$

Următorul algoritm calculează iterativ valoarea lui $P(x)$.

Algoritmul Popoviciu-Casteljau

Din identitatea:

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1},$$

obținem identitatea:

$$p_{n,k}(x) = (1-x)p_{n-1,k}(x) + xp_{n-1,k-1}(x).$$

Mulțimea $\mathcal{B} = \{p_{n,0}(x), ..., p_{n,n}(x)\}$ reprezintă o bază în Π_n. Fie $P \in \Pi_n$ cu reprezentarea

$$P(x) = \sum_{k=0}^{n} a_k p_{n,k}(x).$$

Următorul algoritm calculează iterativ valoarea lui $P(x)$.

Algoritmul Popoviciu-Casteljau. Fie $p_i^0 := a_i$, $i = \overline{0, n}$.

for $r = 1 : n$
  for $i = 0 : n - r$
    $p_i^r := (1-x)p_i^{r-1} + xp_{i+1}^{r-1};$
  end;
end;
Out: $P(x) := p_0^n$.

Curbe Bezier. Algoritmul Popoviciu-Casteljau

Fie $A_k \in \mathbb{R}^2$, $k = \overline{0,n}$ puncte distincte în plan. Curba Bezier determinată de punctele A_k, $k = \overline{0,n}$ este

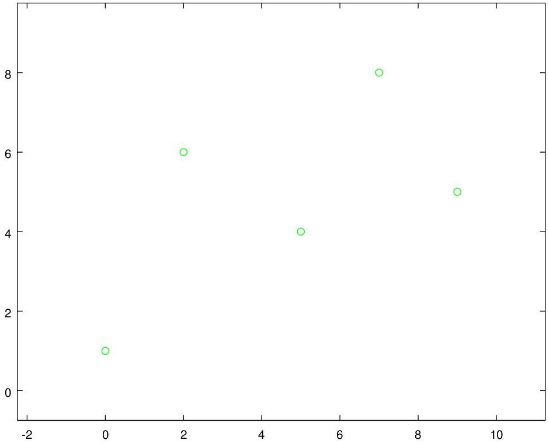
$$P(t) = \sum_{k=0}^{n} p_{n,k}(t) A_k, \, t \in [0,1],$$

unde $P(t)$ este punctul curent pe curba Bezier, $P(t) = (x(t), y(t))^T$ și $A_k = (x_k, y_k)^T$, $k = \overline{0,n}$. Punctul curent $P(t)$ se poate obține folosind algortimul Popoviciu-Casteljau.

Algoritmul Popoviciu-Casteljau (Curbe Bezier). Fie $p_i^0 := A_i = (x_i, y_i)^T$, $i = \overline{0,n}$.

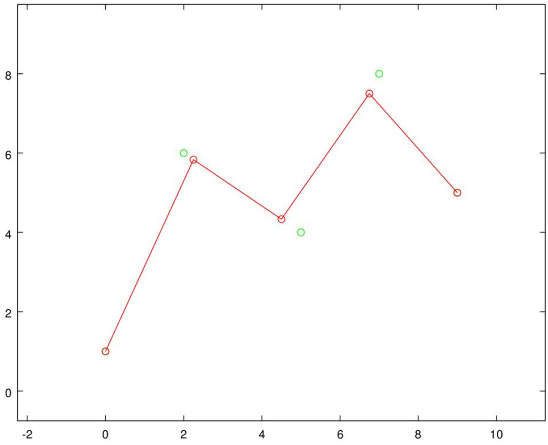
for $r = 1 : n$
 for $i = 0 : n - r$
  $p_i^r := (1-t)p_i^{r-1} + tp_{i+1}^{r-1}$;
 end;
end;
Out: $P(t) = (x(t), y(t))^T := p_0^n$.

Aplicații. Spline-uri Bezier.



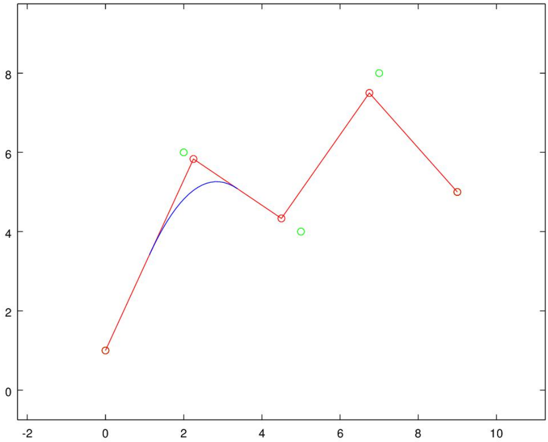
Punctele verzi $(x^{(data)}, y^{(data)})$ obținute din măsurători. Diviziunea pe axa Ox, $\Delta_x^{(data)} : x_0^{(data)} < ... < x_N^{(data)}$, nu este în general echidistantă.

Aplicații. Spline-uri Bezier.



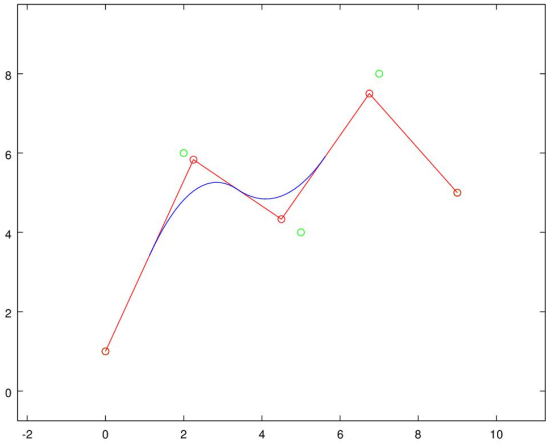
Punctele roșii $(x^{(equi)}, y^{(equi)})$ determină o diviziunea echidistantă pe Ox. Numărul lor este egal cu cel al punctelor verzi, iar $y^{(equi)}$ se obține din spline-ul liniar atasat punctelor verzi. Spline-ul liniar atasat punctelor roșii devine spline-ul de referință.

Aplicații. Spline-uri Bezier.



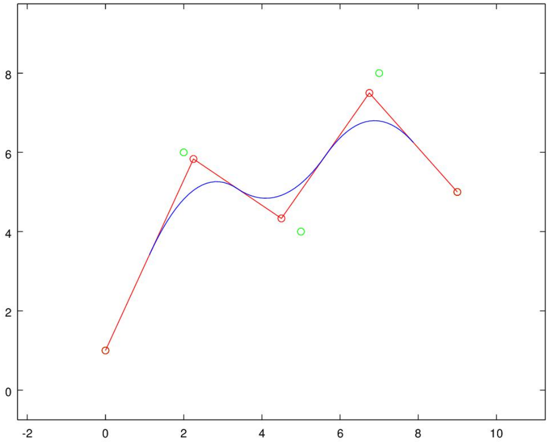
Fiecare subinterval al diviziunii $\Delta_x^{(equi)}$ se înjumătățește $\Rightarrow$ punctele $(x^{(fine)}, y^{(fine)})$ și diviziunea $\Delta_x^{(fine)}$. Pe primul subinterval $[x_0^{(fine)}, x_1^{(fine)}]$ rămâne spline-ul liniar, iar pe intervalul $[x_1^{(fine)}, x_3^{(fine)}]$ folosim curba Bezier de ordinul 2...

Aplicații. Spline-uri Bezier.



Pe subintervalul $[x_3^{(fine)}, x_5^{(fine)}]$ folosim din nou curba Bezier de ordinul 2...

Aplicații. Spline-uri Bezier.



Pe subintervalul $[x_5^{(fine)}, x_7^{(fine)}]$ folosim curba Bezier de ordinul 2, iar pe intervalul $[x_7^{(fine)}, x_8^{(fine)}]$ folosim spline-ul liniar inițial $\Rightarrow$ traiectorie de clasă C^1.