Calcul Numeric – Laboratorul#5
Calculatoare şi Tehnologia Informaţiei, Anul I

Algorithm 1: Interpolare Lagrange (metoda directă)

Input: $\mathbf{X} \in \mathbb{R}^{n+1}$, $\mathbf{Y} \in \mathbb{R}^{n+1}$, $\mathbf{z} \in \mathbb{R}$
Result: $\mathbf{t} \in \mathbb{R}$

Pasul 1: Determină matricea $\mathbf{A} \in \mathbb{R}^{(n+1)\times(n+1)}$, unde:

$$\mathbf{A} = \begin{bmatrix} 1 & x_1 & x_1^2 & \dots & x_1^n \\ 1 & x_2 & x_2^2 & \dots & x_2^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n+1} & x_{n+1}^2 & \dots & x_{n+1}^n \end{bmatrix}.$$

Pasul 2: Determină $\underline{\boldsymbol{a}} \in \mathbb{R}^{n+1}$ (coeficienţii polinomului Lagrange), rezolvând sistemul:

$\mathbf{A} \cdot \underline{\boldsymbol{a}} = \mathbf{Y}$.

Pasul 3: (Determină aproximarea în punctul $\mathbf{z}$)

$$\mathbf{t} \longleftarrow \sum_{i=1}^{n+1} \boldsymbol{a_i} \cdot \mathbf{z}^{i-1}.$$

Pasul 4: OUTPUT($\mathbf{t}$)
STOP.

Algorithm 2: Interpolare Lagrange (metoda Lagrange)

Input: $\mathbf{X} \in \mathbb{R}^{n+1}$, $\mathbf{Y} \in \mathbb{R}^{n+1}$, $\mathbf{z} \in \mathbb{R}$
Result: $\mathbf{t} \in \mathbb{R}$

Pasul 1: (Determină funcţiile de bază $L_{n,k}(\mathbf{z})$)

for $k \leftarrow 1$ **to** $n+1$ **do**

$$L_{n,k} \longleftarrow \prod_{\substack{j=1 \\ j\neq k}}^{n+1} \frac{\mathbf{z} - x_j}{x_k - x_j}$$

end

Pasul 2: (Determină aproximarea în punctul $\mathbf{z}$)

$$\mathbf{t} \longleftarrow \sum_{k=1}^{n+1} \mathbf{L}_{n,k} \cdot y_k.$$

Pasul 3: OUTPUT($\mathbf{t}$)
STOP.

Algorithm 3: Interpolare Lagrange (metoda Newton)

Input: $\mathbf{X} \in \mathbb{R}^{n+1}$, $\mathbf{Y} \in \mathbb{R}^{n+1}$, $\mathbf{z} \in \mathbb{R}$
Result: $\mathbf{t} \in \mathbb{R}$

Pasul 1: Determină matricea $\mathbf{A} \in \mathbb{R}^{(n+1)\times(n+1)}$, unde:

$$a_{i,1} \longleftarrow 1, \quad i = \overline{1, n+1};$$

$$a_{i,j} \longleftarrow \prod_{k=1}^{j-1}(x_i - x_k), \quad i = \overline{2, n+1}, \quad j = \overline{2, i}.$$

Pasul 2: (Determină coeficienţii $\underline{\boldsymbol{c}} \in \mathbb{R}^{n+1}$ (coeficienţii polinomului Lagrange), rezolvând sistemul:

$$\mathbf{A} \cdot \underline{\boldsymbol{c}} = \mathbf{Y}.$$

Pasul 3: Determină aproximarea în punctul $\mathbf{z}$

$$\mathbf{t} \longleftarrow c_1 + \sum_{i=2}^{n+1} c_i \prod_{j=1}^{i-1}(\mathbf{z} - x_j).$$

Pasul 4: OUTPUT($\mathbf{t}$)
STOP.

Algorithm 4: Interpolare Lagrange (metoda Newton cu Diferenţe Divizate)

Input: $\mathbf{X} \in \mathbb{R}^{n+1}$, $\mathbf{Y} \in \mathbb{R}^{n+1}$, $\mathbf{z} \in \mathbb{R}$
Result: $\mathbf{t} \in \mathbb{R}$

Pasul 1: Determină matricea $\mathbf{Q} \in \mathbb{R}^{(n+1)\times(n+1)}$, unde:

$$q_{i,1} \longleftarrow y_i, \quad i = \overline{1, n+1};$$

$$q_{i,j} \longleftarrow \frac{q_{i,j-1} - q_{i-1,j-1}}{x_i - x_{i-j+1}}, \quad i = \overline{2, n+1}, \quad j = \overline{2, i}.$$

Pasul 2: Determină aproximarea în punctul $\mathbf{z}$

$$\mathbf{t} \longleftarrow q_{1,1} + \sum_{k=2}^{n+1} q_{k,k} \prod_{j=1}^{k-1}(\mathbf{z} - x_j).$$

Pasul 3: OUTPUT($\mathbf{t}$)
STOP.

– Exerciţii –

Ex. 1
Implementează în **python** *metoda directă de interpolare Lagrange* cu numele **interp_direct**. Pentru implementare, urmăreşte algoritmul de mai sus.

Pentru verificare, rezolvă problema:

(a) Presupunem că avem datele cunoscute $\mathbf{X}$ (date de client) în punctele obţinute din discretizarea intervalului $[-\pi, \pi]$ în 20 de puncte echidistante. Valorile corespunzătoare punctelor rezultate $\mathbf{Y}$ sunt obţinute prin evaluarea funcţiei $f(x) = \sin(2x) - 2\cos(3x)$ în acele puncte. Într-o figură, afişează datele clientului;

(b) Clientul doreşte aproximarea valorilor funcţiei în toate punctele din discretizarea cu 100 de puncte echidistante a domeniului. Pentru aproximarea valorilor lipsă, foloseşte *metoda directă de interpolare Lagrange*;

(c) Pentru verificare (comparaţie pe grafic), generează graficul funcţiei considerate (date exacte), pe domeniul $[-\pi, \pi]$ folosind o discretizare a domeniului cu 100 de noduri echidistante. În aceeaşi figură, generează şi graficul aproximării obţinute la punctul (b);

(d) Într-o figură nouă, generează graficul erorii de interpolare $e_t = |P_n(x) - f(x)|$.

Ex. 2
Implementează în **python** *metoda Lagrange de interpolare Lagrange* cu numele **interp_lagrange**. Pentru implementare, urmăreşte algoritmul de mai sus.

Pentru verificare, rezolvă problema de la exerciţiul 1.

Ex. 3
Implementează în **python** *metoda Newton de interpolare Lagrange* cu numele **interp_newton**. Pentru implementare, urmăreşte algoritmul de mai sus.

Pentru verificare, rezolvă problema de la exerciţiul 1.

Ex. 4
Implementează în **python** *metoda Newton cu Diferenţe Divizate de interpolare Lagrange* cu numele **interp_newton_dd**. Pentru implementare, urmăreşte algoritmul de mai sus.

Pentru verificare, rezolvă problema de la exerciţiul 1.