

**Calcul Numeric – Proba practică
Informatică, Anul III**

INSTRUCȚIUNI:

1. Comentați și explicați toate rezolvările trimise. Codurile necomentate/neexplicate nu se punctează.
2. Codurile vor fi salvate cu următoarea denumire [Nume_Prenume_Grupa.py](#) și vor fi trimise titularului de laborator până în data de **29 ianuarie 2021, ora 14:30**.

Factorizarea QR :

Fie $A = (a_{ij})_{i,j=\overline{1,n}} \in \mathcal{M}_n(\mathbb{R})$. Numim descompunere QR a matricei A , descompunerea de forma $A = QR$ unde $Q = (q_{ij})_{i,j=\overline{1,n}} \in \mathcal{M}_n(\mathbb{R})$ este o matrice ortogonală, i.e. $Q^T Q = Q Q^T = I$, iar $R = (r_{ij})_{i,j=\overline{1,n}} \in \mathcal{M}_n(\mathbb{R})$ este o matrice superior triunghiulară.

Dacă $A = (a_{ij})_{i,j=\overline{1,n}} \in \mathcal{M}_n(\mathbb{R})$ este o matrice inversabilă, atunci există și este unică descompunerea QR a matricei A cu $Q \in \mathcal{M}_n(\mathbb{R})$ o matrice ortogonală și $R \in \mathcal{M}_n(\mathbb{R})$ o matrice superior triunghiulară având componentele pe diagonala principală $r_{kk} > 0, k = \overline{1, n}$.

Sistemul $Ax = b$ devine $QRx = b$. Cum Q este ortogonală ($Q^T Q = I$), înmulțind relația $QRx = b$ cu Q^T obținem $Rx = Q^T b$. Cum R este superior triunghiulară sistemul $Rx = Q^T b$ se rezolvă conform metodei substituției descendente.

ALGORITM (Metoda de descompunere QR)

Date de intrare: $A = (a_{ij})_{i,j=\overline{1,n}} \in \mathcal{M}_n(\mathbb{R})$, $b = (b_i)_{i=\overline{1,n}} \in \mathcal{M}_{n,1}(\mathbb{R})$;

Date de ieșire: $Q = (q_{ij})_{i,j=\overline{1,n}} \in \mathcal{M}_n(\mathbb{R})$, $R = (r_{ij})_{i,j=\overline{1,n}} \in \mathcal{M}_n(\mathbb{R})$, $x = (x_i)_{i=\overline{1,n}} \in \mathcal{M}_{n,1}(\mathbb{R})$;

PASUL 1: Determină prima coloană a matricei Q și prima linie a matricei R :

$$\begin{aligned} & \bullet r_{11} \leftarrow \sqrt{\sum_{i=1}^n a_{i1}^2} \\ & \bullet q_{i1} \leftarrow \frac{a_{i1}}{r_{11}}, \quad \forall i = \overline{1, n}; \\ & \bullet r_{1j} \leftarrow \sum_{s=1}^n q_{s1} a_{sj}, \quad \forall j = \overline{2, n}; \end{aligned}$$

PASUL 2: Pentru $k = \overline{2, n}$ completează coloana k a matricei Q și linia k a matricei R :

$$\begin{aligned} & \bullet r_{kk} \leftarrow \sqrt{\sum_{i=1}^n a_{ik}^2 - \sum_{s=1}^{k-1} r_{sk}^2}; \\ & \bullet q_{ik} \leftarrow \frac{1}{r_{kk}} \left(a_{ik} - \sum_{s=1}^{k-1} q_{is} r_{sk} \right), \quad \forall i = \overline{1, n}; \\ & \bullet r_{kj} \leftarrow \sum_{s=1}^n q_{sk} a_{sj}, \quad \forall j = \overline{k+1, n}; \end{aligned}$$

PASUL 3: Determină soluția x a sistemului $Rx = Q^T b$ conform metodei substituției descendente

Ex. 1 Fie matricea $A = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}$ și vectorul $b = (1, 2, 5)^T$.

- Să se implementeze în Python procedura **DescQR**(A) care returnează matricele Q, R și soluția x a sistemului $Ax = b$;
- Să se rezolve numeric sistemul $Ax = b$ apelând procedura **DescQR**;
- Să se verifice soluția obținută.

Ex. 2

- (a) Creați funcția **newton_raphson** care determină numeric soluția ecuației:

$$f(x) = x^3 + x^2 - 25x - 25 = 0, \quad (1)$$

prin metoda Newton-Raphson și are ca **date de intrare**:

- funcția care determină ecuația (1), f ;
- derivata funcției care determină ecuația (1), df ;
- punctul de start al metodei Newton-Raphson, x_0 ;
- toleranța erorii specifice metodei Newton-Raphson, **eps**;

iar ca **date de ieșire**:

- soluția numerică obținută, x_{approx} ;
- numărul de iterații necesare, N;

- (b) Alegeți subintervalele și punctele de start ale metodei respectând ipotezele teoremei de convergență ale metodei Newton-Raphson, astfel încât șirurile aproximărilor să rămână în subintervalele selectate și să converge la soluții. Justificați atât alegerea subintervalelor, cât și a valorilor inițiale.

Aflați toate soluțiile ecuației (1) apelând funcția **newton_raphson** cu eroarea de aproximare **eps** = 10^{-3} și construiți punctele obținute pe graficul funcției.