C2

• **SST** $\{Ux = b,\ U \in \mathbb{R}^{n\times n},\ u_{ij} = 0,\ i > j,\ u_{ii} \neq 0,\ b \in \mathbb{R}^n,\ x = ?\}$

$$\begin{pmatrix} u_{11} & u_{12} & \dots & u_{1n} \\ 0 & u_{22} & \dots & u_{2n} \\ 0 & 0 & u_{32} \dots & u_{3n} \\ & \vdots & & \\ 0 & 0 & 0 \dots & u_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

$$u_{nn} x_n = b_n \Rightarrow x_n = \frac{b_n}{u_{nn}}$$

$$u_{n-1\,n-1} x_{n-1} + u_{n-1\,n} x_n = b_{n-1} \Rightarrow$$

$$x_{n-1} = \frac{b_{n-1} - u_{n-1\,n} x_n}{u_{n-1\,n-1}}$$

$$\vdots$$

$$x_1 = \frac{b_1 - u_{1n} x_n - u_{1\,n-1} x_{n-1} - \dots u_{12} x_2}{u_{11}} \Rightarrow$$

$$x_i = \frac{b_i - \sum_{k=i+1}^{n} u_{ik} x_k}{u_{ii}} = \frac{b_i - \sum_{j=i+1}^{n} u_{ij} x_j}{u_{ii}}$$

• **SiT** $(Lx = b,\ L \in \mathbb{R}^{n\times n},\ l_{ij} = 0,\ i < j,\ l_{ii} \neq 0,\ b \in \mathbb{R}^n,\ x = ?)$

$$\begin{pmatrix} l_{11} & 0 & 0 & \dots & 0 \\ l_{12} & l_{22} & 0 & \dots & 0 \\ l_{13} & l_{23} & l_{33} & 0 \dots & 0 \\ \dots & & & & \\ l_{n1} & l_{n2} & l_{n3} & \dots & l_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

$$l_{11} x_1 = b_1 \Rightarrow x_1 = \frac{b_1}{l_{11}}$$

$$l_{12} x_1 + l_{22} x_2 = b_2 \Rightarrow x_2 = \frac{b_2 - l_{12} x_1}{l_{22}}$$

$$\dots$$

$$l_{n1} x_1 + l_{n2} x_2 + \dots + l_{nn} x_n = b_n \Rightarrow$$

Scanat cu CamScanner

$$\Rightarrow x_n = \frac{b_n - l_{n1}x_1 - l_{n2}x_2 - \ldots - l_{n\,n-1}x_{n-1}}{l_{nn}} \Rightarrow$$

$$x_i = \frac{b_i - \sum_{j=1}^{i-1} l_{ij} \cdot x_j}{l_{ii}}$$

• <u>Factorizarea Crout</u> $(u_{ii} = 1)$ $\Rightarrow$ Se face identificarea formulelor folosind o matrice $A \in \mathbb{R}^{3\times3} \Rightarrow$

$$\begin{pmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{pmatrix} \cdot \begin{pmatrix} 1 & u_{12} & u_{13} \\ 0 & 1 & u_{23} \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

$$\Leftrightarrow \begin{cases} l_{11} = a_{11} \\ l_{21} = a_{21} \\ l_{31} = a_{31} \end{cases} \quad \begin{cases} l_{11}u_{12} = a_{12} \Rightarrow u_{12} = \dfrac{a_{12}}{l_{11}} \\ l_{11}u_{13} = a_{13} \Rightarrow u_{13} = \dfrac{a_{13}}{l_{11}} \end{cases}$$

$$\begin{cases} l_{21}u_{12} + l_{22} = a_{22} \Rightarrow l_{22} = a_{22} - l_{21}u_{12} \\ l_{31}u_{12} + l_{32} = a_{32} \Rightarrow l_{32} = a_{32} - l_{31}u_{12} \end{cases}$$

$$\{ l_{21}u_{13} + l_{22}u_{23} = a_{23} \Rightarrow u_{23} = \frac{a_{23} - l_{21}u_{13}}{l_{22}}$$

$$l_{31}u_{13} + l_{32}u_{23} + l_{33} = a_{33} \Rightarrow l_{33} = a_{33} - l_{31}u_{13} - l_{32}u_{23}$$

$\Rightarrow$ La pasul p $(p = 1:n) \Rightarrow$

$$l_{ip} = a_{ip} - \sum_{k=1}^{p-1} l_{ik}u_{kp} \;; \; i = p:n$$

$$u_{pj} = \frac{a_{pj} - \sum_{k=1}^{p-1} l_{pk}u_{kj}}{l_{pp}} \;, \; j = p+1:n$$

Scanat cu CamScanner

• Factorizarea Doolittle ($l_{ii}=1$) $\Rightarrow$ Se face identificarea formulelor folosind o matrice $A\in\mathbb{R}^{3\times 3}$ $\Rightarrow$

$$\begin{pmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{pmatrix}\cdot\begin{pmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \Rightarrow$$

$$\begin{cases} u_{11}=a_{11} \\ u_{12}=a_{12} \\ u_{13}=a_{13} \end{cases} \quad \begin{cases} l_{21}u_{11}=a_{21} \Rightarrow l_{21}=\dfrac{a_{21}}{u_{11}} \\ l_{31}u_{11}=a_{31} \Rightarrow l_{31}=\dfrac{a_{31}}{u_{11}} \end{cases}$$

$$l_{21}u_{12}+u_{22}=a_{22} \Rightarrow \begin{cases} u_{22}=a_{22}-l_{21}u_{12} \\ u_{23}=a_{23}-l_{21}u_{13} \end{cases}$$
$$l_{21}u_{13}+u_{23}=a_{23} \Rightarrow$$

$$l_{31}u_{12}+l_{32}u_{22}=a_{32} \Rightarrow \{ l_{32}=\frac{a_{32}-l_{31}u_{12}}{u_{22}}$$

$$\{ l_{31}u_{13}+l_{32}u_{23}+u_{33}=a_{33} \Rightarrow u_{33}=a_{33}-l_{31}u_{13}-l_{32}u_{23}$$

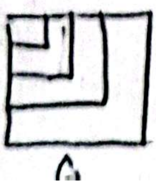
La pasul p ($p=1:n$) $\Rightarrow$

$$u_{pj}=a_{pj}-\sum_{k=1}^{p-1} l_{pk}u_{kj} \ ; \ j=p:n$$

$$l_{ip}=\frac{a_{ip}-\sum_{k=1}^{p-1} l_{ik}u_{kp}}{u_{pp}} \ , \ i=p+1:n$$

• Rezolvare SEL: Fie $A\in\mathbb{R}^{n\times n}$. Dacă submatricile $A^{[p]}=(a_{ij})_{\substack{1\le i\le p \\ 1\le j\le p}}\in\mathbb{R}^{p\times p}$, $p=1:n$ sunt nesingulare, at

$\exists\ L\in\mathbb{R}^{n\times n}$ și $U\in\mathbb{R}^{n\times n}$ a.î. $A=L\cdot U$



A

Scanat cu CamScanner

$$\left.\begin{array}{l} Ax = b \Leftrightarrow L\cdot\underbrace{(Ux)}_{y} = b \Leftrightarrow y = SIT(L, b) \\ Ux = y \Rightarrow x = SST(U, y) \end{array}\right\} \Rightarrow$$

$x = SST(U, SIT(L, b))$, unde $A = L\cdot U$ - matrici obținute prin aplicarea unei factorizări

• __Metoda partiționării pt calcularea inversei unei matrice__

Fie $A \in \mathbb{R}^{m\times m} \Rightarrow A = \begin{pmatrix} A_1 & A_3 \\ A_2 & A_4 \end{pmatrix}$, A_1, A_4 - matrici pătratice

$$A^{-1} = X = \begin{pmatrix} X_1 & X_3 \\ X_2 & X_4 \end{pmatrix} \Rightarrow$$

$$AA^{-1} = AX = I_m \Leftrightarrow \begin{pmatrix} A_1 & A_3 \\ A_2 & A_4 \end{pmatrix}\cdot\begin{pmatrix} X_1 & X_3 \\ X_2 & X_4 \end{pmatrix} = \begin{pmatrix} I & 0 \\ 0 & I \end{pmatrix} \Leftrightarrow$$

$$\left.\begin{array}{l} A_1X_1 + A_3X_2 = I \\ A_1X_3 + A_3X_4 = 0 \Rightarrow X_3 = -A_1^{-1}A_3X_4 \\ A_2X_1 + A_4X_2 = 0 \Rightarrow X_2 = -A_4^{-1}A_2X_1 \end{array}\right\} \Rightarrow$$

$$A_2X_3 + A_4X_4 = I$$

$$A_1X_1 - A_3A_4^{-1}A_2X_1 = I \Leftrightarrow X_1(A_1 - A_3A_4^{-1}A_2) = I \Rightarrow$$

$$\left.\begin{cases} X_1 = (A_1 - A_3A_4^{-1}A_2)^{-1} \\ X_2 = -A_4^{-1}A_2X_1 \\ X_3 = -A_1^{-1}A_3X_4 \\ X_4 = (A_4 - A_2A_1^{-1}A_3)^{-1} \end{cases}\right\} \text{Se obțin în mod analog prin înlocuiri}$$

• __Crout3diag__

$A = diag(d, -1) + diag(a,) + diag(-c, 1)$

Scanat cu CamScanner

$$\underbrace{\begin{pmatrix} a_1 & c_1 & 0 & 0 \\ d_1 & a_2 & c_2 & 0 \\ 0 & d_2 & a_3 & c_3 \\ 0 & 0 & d_3 & a_4 \end{pmatrix}}_{A} = \underbrace{\begin{array}{cc} & ld \\ l1 & \begin{pmatrix} l_{11} & 0 & 0 & 0 \\ l_{21} & l_{22} & 0 & 0 \\ 0 & l_{32} & l_{33} & 0 \\ 0 & 0 & l_{43} & l_{44} \end{pmatrix} \end{array}}_{L} \underbrace{\begin{array}{c} ud \quad u1 \\ \begin{pmatrix} 1 & u_{12} & 0 & 0 \\ 0 & 1 & u_{23} & 0 \\ 0 & 0 & 1 & u_{34} \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array}}_{U}$$

1) $\begin{cases} l_{11} = a_1 \\ l_{21} = d_1 \end{cases}$ $\quad l_{11} u_{12} = c_1 \Rightarrow \{ u_{12} = \dfrac{c_1}{l_{11}}$

2) $l_{21} u_{12} + l_{22} = a_2 \Rightarrow \begin{cases} l_{22} = a_2 - l_{21} u_{12} \\ l_{32} = d_2 \end{cases}$

$l_{32} = d_2 \quad \Rightarrow$

$l_{22} u_{23} = c_2 \Rightarrow \{ u_{23} = \dfrac{c_2}{l_{22}}$

3) $l_{32} u_{23} + l_{33} = a_3 \Rightarrow \begin{cases} l_{33} = a_3 - l_{32} u_{23} \\ l_{43} = d_3 \end{cases}$

$l_{43} = d_3 \quad \Rightarrow$

$l_{33} u_{34} = c_3 \Rightarrow \{ u_{34} = \dfrac{c_3}{l_{33}}$

4) $l_{43} u_{34} + l_{44} = a_4 \Rightarrow \{ l_{44} = a_4 - l_{43} u_{34}$

La pasul p $(p = 1:m) \Rightarrow$

$ld(p) = a_p - l1(p-1) u1(p-1)$; $l1(0) = 0$

$l1(p) = d_p$

$u1(p) = \dfrac{c_p}{ld(p)}$, $l1(m), u1(m)$ nu se calculează

$l1, u1 \in \mathbb{R}^{m-1}$

Scanat cu CamScanner

• <u>Cholesky</u> (A - pozitiv definită)

$L = ?$ și L - inferior triunghiulară a.î. $A = L \cdot L^T$

$$\begin{pmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{pmatrix} \cdot \begin{pmatrix} l_{11} & l_{21} & l_{31} \\ 0 & l_{22} & l_{32} \\ 0 & 0 & l_{33} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

$$1)\begin{cases} l_{11}^2 = a_{11} \Rightarrow l_{11} = \sqrt{a_{11}} \\ l_{11} l_{21} = a_{12} \Rightarrow l_{21} = \dfrac{a_{12}}{l_{11}} \\ l_{11} l_{31} = a_{13} \Rightarrow l_{31} = \dfrac{a_{13}}{l_{11}} \end{cases}$$

$$2)\begin{cases} l_{21}^2 + l_{22}^2 = a_{22} \Rightarrow l_{22} = \sqrt{a_{22} - l_{21}^2} \\ l_{21} l_{31} + l_{22} l_{32} = a_{23} \Rightarrow l_{32} = \dfrac{a_{23} - l_{21} l_{31}}{l_{22}} \end{cases}$$

$$3)\{ l_{31}^2 + l_{32}^2 + l_{33}^2 = a_{33} \Rightarrow l_{33} = \sqrt{a_{33} - l_{31}^2 - l_{32}^2}$$

pt pasul p ($p = 1:n$)

$$l_{pp} = \sqrt{a_{pp} - \sum_{k=1}^{p-1} l_{pk}^2}$$

$$l_{ip} = \frac{a_{ip} - \sum_{k=1}^{p-1} l_{pk} l_{ik}}{l_{pp}}, \quad i = p+1:n$$

Scanat cu CamScanner

C3

· **Gram Schmidt**

Ne dorim noi determinăm factorizarea QR a unei matrice A cu coloane liniar independente folosind procedeul Gram-Schmidt.

Vom deduce algoritmul Gram-Schmidt modificat folosind drept exemplu o matrice $A \in \mathbb{R}^{3\times 3}$

$$A = \begin{array}{c} \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \\ \begin{array}{ccc} \uparrow & \uparrow & \uparrow \\ a_1 & a_2 & a_3 \end{array} \end{array} = Q \cdot R;$$

$Q = [q_1\ q_2\ q_3]$ - matrice ortogonală

$$R = \begin{pmatrix} \|u_1\| & \langle q_1, a_2\rangle & \langle q_1, a_3\rangle \\ 0 & \|u_2\| & \langle q_2, a_3\rangle \\ 0 & 0 & \|u_3\| \end{pmatrix}$$

Procedeul decurge în felul următor:

$u_1 = a_1 \Rightarrow$

$$q_1 = \frac{u_1}{\|u_1\|}$$

$u_2 = a_2 - \langle q_1, a_2\rangle \cdot q_1$ (proiecția lui a_2 pe q_1)

$q_2 = \dfrac{u_2}{\|u_2\|}$ (se normează vectorul)

$$u_3 = a_3 - \langle q_1, a_3\rangle q_1 - \langle q_2, a_3\rangle q_2$$

$$q_3 = \frac{u_3}{\|u_3\|}$$

Algoritmul Gram-Schmidt modificat construiește în același timp matricele Q și R, modificând matricea A în timpul construcției fiecărei linii 1)

Scanat cu CamScanner

a matricii R.

Algoritmul are forma:

for $i = 1 : n$

$r_{ii} = \|a_i\|$ - se populează diagonala principală a matricii R

$q_i = \frac{a_i}{r_{ii}}$ - nu va fi egal cu a_i, deoarece in al doilea for se modifică matricea A

for $j = i+1 : n$ - se parcurge linia curentă pt a

$r_{ij} = \langle q_i, a_j \rangle$ se populează matricea R și

$a_j = a_j - r_{ij} \cdot q_i$ cu aceasta o coarne se modifică matricea A

a_i la pasul (i), nu să fie egal cu a_i (se calculează din timp proiecția vectorului inițial a_i pe subspațiul determinat de $q_1 \dots q_{i-1}$

• Householder

$A \in \mathbb{R}^{m \times n}$ și ne dorim să obț Q și R ai $A = Q \cdot R$, unde $Q \in \mathbb{R}^{m \times m}$ matrice ortogonală și $R \in \mathbb{R}^{m \times n}$ matrice superior triunghiulară.

Algoritm:

1) se calculează vectorul Householder v_p

$$v_p = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ v_{pp} \\ \vdots \\ v_{mp} \end{pmatrix} \leftarrow \text{al } p\text{-lea vector}$$

$v_{ip} = a_{ip}$, $\forall i > p$ (toate elementele de sub v_{pp} sunt cele din matricea A)

$$\sigma_p = \operatorname{sgn}(a_{pp}) \cdot \sqrt{\sum_{i=p}^{m} a_{ip}^2} \Rightarrow v_{pp} = a_{pp} + \sigma_p$$

Scanat cu CamScanner

2) se calculează reflectorul Householder

$$H_p = I_m - 2\frac{v_p \cdot v_p^T}{v_p^T v_p}$$

3) se calculează H și A:

$A = H_p A$

$H = H_p \cdot H$

4) se repetă până când se termină toate coloanele / liniile ($\min(m-1, n)$ pași)

5) $R = A$, $Q = H^T$

• Proprietăți matrice H_p:

1) H_p - simetrică $\Leftrightarrow H_p = H_p^T$

2) H_p - ortogonală $\Leftrightarrow H_p^T \cdot H_p = I_m$

Fie $H = H_{\min(m-1,n)} \cdot \ldots \cdot H_p \cdot \ldots \cdot H_1$ - ortogonală, $H \in \mathbb{R}^{m \times m}$

Algoritmul Householder construiește matricea R prin crearea de zerouri sub elementele (i,i) ale matricei A

Modul prin care acest algoritm funcționează se face evidentă această notație

• Givens

$A \in \mathbb{R}^{m \times n}$, $Q, R = ?$ a.î. $A = QR$, unde $Q \in \mathbb{R}^{m \times m}$ - matrice ortogonală, $R \in \mathbb{R}^{m \times n}$ - matrice superior triunghiulară

Algoritmul de factorizare QR Givens se folosește în detrimentul celui Householder atunci când sub diagonala principală a matricei A există deja un număr mare de zerouri.

Dacă această condiție este îndeplinită, factorizarea Givens este mai eficientă decât factorizarea Householder

Algoritm:

• Se iterează pe fiecare coloană toate elementele de sub diagonala principală până când se ajunge la $\min(m-1, n)$ iterații (până când s-a trecut prin toate coloanele principalei diagonale)

Pt G_{kl} (k - coloana curentă, l - linia curentă):

1) se calculează $\rho = \sqrt{A(k,k)^2 + A(l,k)^2}$

2) se calculează c și s

$$c = \frac{A(k,k)}{\rho}, \quad s = -\frac{A(l,k)}{\rho}$$

Obs: pt ρ și pt s se folosește elementul $A(l,k)$, indicii sunt fie inverși față de indicii matricei G_{kl}

3) se construiește matricea G_{kl} care este matricea unitate I care are 4 elemente modificate, pe liniile și coloanele k și l; aceste elemente sunt, în ordine: $\begin{pmatrix} c & -s \\ s & c \end{pmatrix}$

4) se calculează noul $A = G_{kl} A$ și $G = G_{kl} G$

• Se repetă acest procedeu până când s-au realizat $\min(m-1, n)$ iterații (s-au vizitat $\min(m-1, n)$ coloane)

• În final: $R = A$, $Q = G^T$; $G = G_{\min(m-1,n)} \cdot \ldots \cdot G_{13} G_{12}$

Algoritmul Givens produce zero-uri în locul elementelor (l, k), iar mecanismul din spatele acestuia depășește scopul 4)

• Eliminarea Gaussiană

Acest algoritm este identic cu cel învățat în cadrul cursului de ALGAED și presupunea eliminarea numerelor de sub diagonala principală a unei matrice pătratice $A \in \mathbb{R}^{n \times n}$ sau eliminarea numerelor de sub pivoții unei matrice ce nu este pătratică.

Algoritmul poate fi folosit pt a rezolva un SEL dacă vectorul termenilor liberi este transformat odată cu matricea A

Dacă matricea A se aduce la forma scară redusă (după eliminare, matricea A devine matricea unitate) atunci algoritmul poate fi folosit pentru a calcula inversa unei matrice

Eliminarea Gaussiană are însă și un defect, deoarece linia curentă folosită pentru eliminări poate conține un zero pe poziția de pivot și astfel algoritmul ar putea să împartă la zero (nu este recomandat!)

Pentru a corecta acest defect se propun următorii algoritmi:

• GPP (Gauss cu pivotare parțială)

Pentru coloana curentă se caută elementul maxim în modul de sub diagonala principală și de pe diagonala principală inclusiv, ce se află pe coloana curentă

Linia elementului maxim se interschimbă cu linia curentă și se aplică apoi eliminarea gaussiană obișnuită pentru a crea zerouri sub diagonala principală.

Astfel, se evită împărțirea la zero.

Scanat cu CamScanner

• **GPPS (Gauss cu pivotare și pivot scalat)**

Pentru coloana curentă se alege maximul scalat de pe și de sub diagonala principală și se interschimbă linia acestui maxim cu linia curentă.

Maximul scalat = valoarea maximă a valorilor scalate; o valoare scalată este obținută prin împărțirea elementului la valoarea maximă de pe linia sa.

• **GPT (Gauss cu pivotare totală)**

Pentru coloana curentă se caută în submatricea ce începe de la elementul (i,i) și este bordată de elementul (n,n) valoarea absolută maximă. După determinarea acestei valori coloana curentă se interschimbă cu coloana maximului și mai apoi linia curentă se interschimbă cu linia maximului.

Obs: Dacă se dorește determinarea soluției unui sistem folosind această metodă ordinea elementelor vectorului x al soluției nu va mai fi cea corectă.

• **Algoritmul lui Thomas**

Acest algoritm reprezintă algoritmul de eliminare gaussiană pt o matrice A tridiagonală.

$$\begin{pmatrix} b_1 & c_1 & \cdots & \cdots & 0 \\ a_2 & b_2 & c_2 & & \vdots \\ \vdots & a_3 & b_3 & c_3 & \vdots \\ \vdots & & \ddots & \ddots & c_{n-1} \\ 0 & \cdots & \cdots & a_n & b_n \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ \vdots \\ d_n \end{pmatrix}$$

Scanat cu CamScanner

$$\Leftrightarrow \left(\begin{array}{cccccc|c} 1 & -c_1/b_1 & \dots & \dots & & 0 & d_1/b_1 \\ a_2 & b_2 & -c_2 & & & \vdots & d_2 \\ \vdots & a_3 & b_3 & -c_3 & & \vdots & d_3 \\ \vdots & & \ddots & \ddots & \ddots & -c_{n-1} & \vdots \\ 0 & \dots & \dots & & a_n & b_n & d_n \end{array}\right) \rightarrow \left(\begin{array}{cccccc|c} 1 & -c_1/b_1 & \dots & \dots & & 0 & d_1/b_1 \\ 0 & b_2 - \frac{a_2 c_1}{b_1} & -c_2 & & & & d_2 - \frac{a_2 d_1}{b_1} \\ \vdots & a_3 & b_3 & c_3 & & & d_3 \\ \vdots & & \ddots & \ddots & \ddots & -c_{n-1} & \vdots \\ 0 & \dots & \dots & & a_n & b_n & d_n \end{array}\right)$$

Ne dorim să obținem 1 pe diagonala principală și observăm că vectorul a devine nul, vectorul b va avea numai valorea 1, iar vectorii $-c$ și d se vor supune următoarelor relații:

$$-c'_i = \begin{cases} \dfrac{-c_1}{b_1}, & i = 1 \\ \dfrac{-c_i}{b_i - a_i c'_{i-1}}, & i = 2:n-1 \end{cases} \qquad a_1 = 0,\ -c_n = 0$$

$$d'_i = \begin{cases} \dfrac{d_1}{b_1}, & i = 1 \\ \dfrac{d_i - a_i d'_{i-1}}{b_i - a_i c'_{i-1}}, & i = 2:n \end{cases}$$

$$x_i = \begin{cases} x_n = d'_n, & i = n \\ x_i = d'_i - c'_i x_{i+1}, & i = n-1:-1:1 \end{cases}$$

Scanat cu CamScanner

C5

• **Metode iterative**

Ne dorim să obț. soluția ec $Ax = b$ și pentru a determina această soluție vom porni cu o estimare inițială a acesteia pe care o vom îmbunătăți cu fiecare pas.

Matricea A se va scrie sub forma $A = N - P \Rightarrow$

$Ax = b \Leftrightarrow (N-P)x = b \Leftrightarrow Nx = Px + b \,/\, N^{-1} \Leftrightarrow$

$x = N^{-1}Px + N^{-1}b$

La pasul $x^{(k)}$ se presupune că aceo aproximație este suficient de bună și converge spre soluția reală atunci:

$x^{(k)} \simeq N^{-1}Px^{(k)} + N^{-1}b$, deci următorul x va fi membrul drept $\Rightarrow x^{(k+1)} = N^{-1}Px^{(k)} + N^{-1}b$

Într-un fel, soluția se îmbunătățește pe sine cu fiecare iterație, afirmație a cărei demonstrații nu face obiectul acestor notițe.

• **Metoda Jacobi**

$A = N - P$

$P = L + U$ (se partiționează matricea în două matrici, una inferior și cealaltă superior triunghiulare)

$N = D$ (matrice diagonală)

$Ax = b \Leftrightarrow (N - (L+U))x = b \Leftrightarrow Nx = (L+U)x + b \Leftrightarrow$

$x = N^{-1}(L+U)x + N^{-1}b$

Scanat cu CamScanner

$N x^{(n+1)} = (L+U)x^{(n)} + b \iff$

$$\begin{pmatrix} a_{11} & \cdots & & 0 \\ 0 & a_{22} & & \vdots \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & & a_{mm} \end{pmatrix} \begin{pmatrix} x_1^{(n+1)} \\ x_2^{(n+1)} \\ \vdots \\ x_m^{(n+1)} \end{pmatrix} = -\begin{pmatrix} 0 & a_{12} & a_{13} \cdots & a_{1m} \\ a_{21} & 0 & & \vdots \\ \vdots & & 0 \ \ddots & \vdots \\ a_{2m} & \cdots & & 0 \end{pmatrix} \begin{pmatrix} x_1^{(n)} \\ x_2^{(n)} \\ \vdots \\ x_m^{(n)} \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}$$

$a_{11} x_1^{(n+1)} = -(a_{12} x_2^{(n)} + a_{13} x_3^{(n)} + \ldots + a_{1m} x_m^{(n)}) + b_1 \Rightarrow$

$a_{ii} x_i^{(n+1)} = -(a_{i1} x_1^{(n)} + \ldots a_{ii-1} x_{i-1}^{(n)} + a_{ii+1} x_{i+1}^{(n)} +$

$+ a_{im} x_m^{(n)}) + b_i \Rightarrow$

$$x_i^{(n+1)} = \frac{-\sum\limits_{j=1, j\neq i}^{m} a_{ij} x_j^{(n)} + b_i}{a_{ii}}$$

• <u>Metoda Gauss-Seidel</u>

$N = D - L$ (diferența de matrice diagonală și matrice inferior triunghiulară)

$P = U$ (matrice superior triunghiulară)

$A = N - P \Rightarrow Ax = b \iff (N-P)x = b \iff$

$Nx = Px + b \iff (D-L)x = Ux + b \iff$

$$\begin{pmatrix} a_{11} & \cdots & & 0 \\ a_{21} & a_{22} & & \vdots \\ \vdots & & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mm} \end{pmatrix} \begin{pmatrix} x_1^{(n+1)} \\ x_2^{(n+1)} \\ \vdots \\ x_m^{(n+1)} \end{pmatrix} = -\begin{pmatrix} 0 & a_{12} & \cdots & a_{1m} \\ \vdots & 0 & \cdots & a_{2m} \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & & 0 \end{pmatrix} \begin{pmatrix} x_1^{(n)} \\ x_2^{(n)} \\ \vdots \\ x_n^{(n)} \end{pmatrix}$$

$$+ \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b \end{pmatrix}$$

Scanat cu CamScanner

$\Leftrightarrow a_{11}x_1^{(k+1)} = a_{12}x_2^{(k)} + \dots + a_{1n}x_n^{(k)} + b_1$

$\vdots$

$a_{i1}x_1^{(k+1)} + a_{i2}x_2^{(k+1)} + \dots + a_{ii}x_i^{(k+1)} =$

$= -(a_{i\,i+1}x_{i+1}^{(k)} + \dots + a_{in}x_n^{(k)}) + b_i$

$$x_i^{(k+1)} = \frac{b_i - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)} - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)}}{a_{ii}}$$

- Matricele L și U din ambele metode au pe diagonală zero (strict triunghiulare)

$G = N^{-1}P$ – matricea de iterație

$c = N^{-1}b$ – vector de iterație

- <u>Convergența metodelor iterative de rezolvare a SEL</u>

$\rho(G) \overset{def}{=} \max\{|\lambda_i|\}$, $i = 1:n$, λ_i – valorile proprii ale matricei G

$\rho(G)$ – raza spectrală a matricei G

Orice metodă iterativă de rezolvare a SEL converge dacă $\rho(G) < 1$. Cu cât $\rho(G)$ se apropie de zero, metoda iterativă converge mai repede.

Dacă A este matrice diagonal dominantă atunci metoda converge în mod sigur.

- <u>Metoda suprarelaxării (SOR)</u>:

SOR – Successive Overrelaxation

Scanat cu CamScanner

Se alege un $w \in (0,2)$ care se numește factor de relaxare

Pt $w = 1 \Rightarrow$ Gauss - Siedel

Metoda se aplică astfel:
- în algoritmul Gauss - Siedel obișnuit, după calcularea lui $x(i)$, soluția se relaxează:

$$x(i) = w\, x(i) + (1-w)\, x_0(i)$$

Mecanismul din spatele acestor metode nu face parte din sfera noastră actuală de interese.

Scanat cu CamScanner

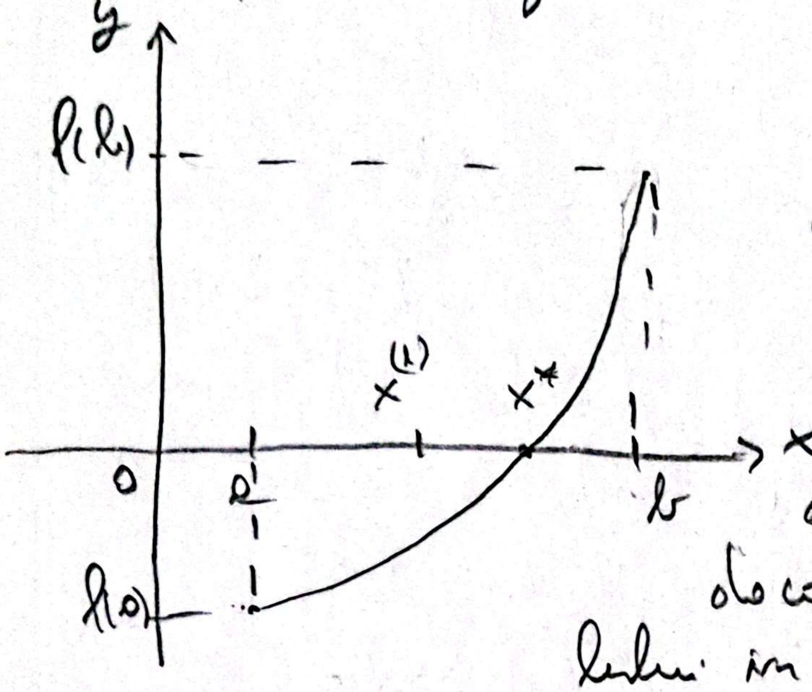
L6

• **Metoda bisecției**

Fie ecuația neliniară $f(x)=0$

$f:[a,b]\to\mathbb{R}$, f cont pe $[a,b]$, $f(a)\cdot f(b)<0$

$[a,b]$ conține o singură rădăcină reală a ec



Ne dorim să obț o soluție aproximativă a ec $f(x)=0$, așa că vom încerca să determinăm succesiv doar mijlocul intervalului în care conține resp o aproximație bună a soluției

$$x^{(1)} = a^{(1)} + \frac{b^{(1)} - a^{(1)}}{2}$$

Dacă $|f(x^{(1)})| < \varepsilon$ sau $|b^{(1)} - a^{(1)}| < \varepsilon$ atunci $x^{(1)}$ resp o aproximație suficient de bună, în caz contrar alegem noul interval a.î.

$f(a^{(2)})\cdot f(b^{(2)}) < 0$ ($a^{(2)} = x^{(1)}$ sau $b^{(2)} = x^{(1)}$ alese a.î. să fie satisfăcută condiție)

La interația (n), aproximația devine:

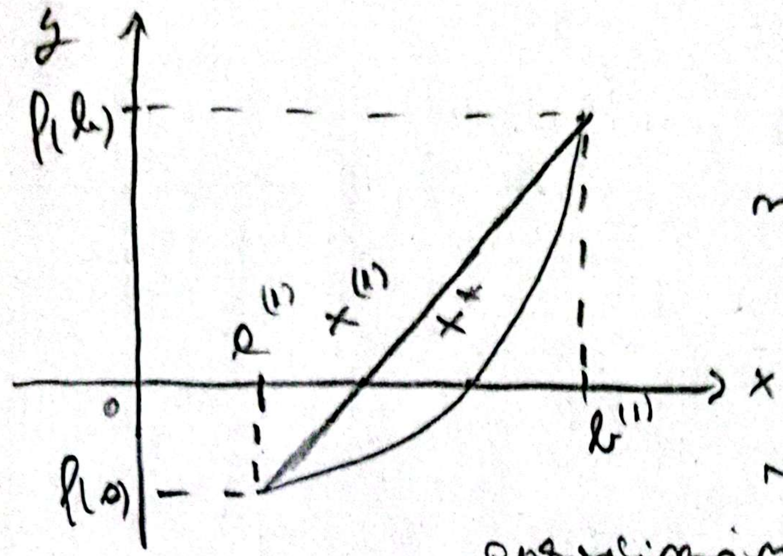
$$x^{(n)} = a^{(n)} + \frac{b^{(n)} - a^{(n)}}{2} \quad (1)$$

Se iterează până când aproximația devine suficient de bună.

1)

Scanat cu CamScanner

• Metoda secantei



Ne dorim să determinăm o aproximație cât mai bună pt ec $f(x)=0$ și în acest sens vom începe să aproximăm soluția cu punctul de intersecție al dreptei ce unește punctele $(a, f(a))$, $(b, f(b))$, unde a și b sunt capetele intervalului în care căutăm

$$m = \frac{f(b)-f(a)}{b-a} \Rightarrow d: y = mx+n \Leftrightarrow$$

$$f(a) = ma+n \Rightarrow n = f(a) - \frac{a f(b) - a f(a)}{b-a} =$$

$$= \frac{b f(a) - \cancel{a f(a)} - a f(b) + \cancel{a f(a)}}{b-a} = \frac{b f(a) - a f(b)}{b-a} \Rightarrow$$

$$\text{Pt } y=0 \Leftrightarrow \frac{f(b)-f(a)}{\cancel{b-a}} x = \frac{a f(b) - b f(a)}{\cancel{b-a}} \Rightarrow$$

$$x = \frac{a f(b) - b f(a)}{f(b)-f(a)} = x^{(1)}$$

Aproximația după prima iterație va fi deci $x^{(1)}$, iar noul interval se alege a.î. $f(a^{(2)}) f(b^{(2)}) < 0$ ($a^{(2)} = x^{(1)}$ sau $b^{(2)} = x^{(1)}$ a.î. condiția să fie îndeplinită). Când $|f(x^{(n)})| < \varepsilon$ sau $|(b^n - a^n)| < \varepsilon$, aproximația a devenit suficient de bună și algoritmul se oprește

Scanat cu CamScanner

• Metoda aproximațiilor succesive

Ne dorim să obț. sol ec $f(x)=0$

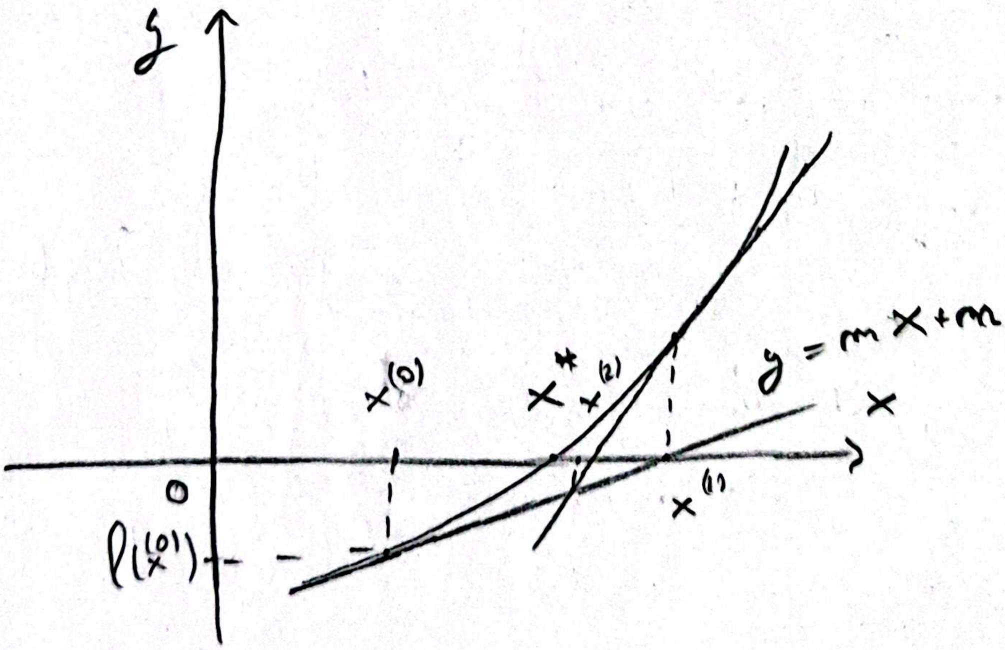
Notăm : $g(x) = f(x) + x$ și pt fiecare nouă iterație

$x^{(k+1)} = g(x^{(k)}) = f(x^{(k)}) + x^{(k)}$ în speranța că $f(x^{(k+1)})$ se va apropia cât de mult de zero

Obs: Metoda poate fi divergentă pt anumite ecuații

• Metoda tangentei :

Ne dorim să determinăm soluția aproximativă a ecuației $f(x)=0$ și în acest sens vom încerca să determinăm această soluție prin folosirea punctului de intersecție al dreptei tangente la grafic într-un punct inițial arbitrar ales. Când soluția aproximată se apropie de cea reală intersecția dreptei tangente la grafic cu axa Ox devine foarte apropiată de vechea aproximație de la pasul anterior, iar când diferența $|x^{(k+1)} - x^{(k)}| < \varepsilon$, s-a găsit o aproximație suficient de bună.



Scanat cu CamScanner

$d: y = mx + n$
$m = f'(x^{(0)})$
$f(x^{(0)}) = m x^{(0)} + n \Rightarrow n = f(x^{(0)}) - x^{(0)} \cdot f'(x^{(0)})$ $\Big\} \Rightarrow$

$$y = f'(x^{(0)})x + f(x^{(0)}) - x^{(0)} f'(x^{(0)})$$

$y = 0 \Leftrightarrow \quad x^{(1)} f'(x^{(0)}) = x^{(0)} f'(x^{(0)}) - f(x^{(0)}) \Leftrightarrow$

$$x^{(1)} = x^{(0)} - \frac{f(x^{(0)})}{f'(x^{(0)})}$$ (aproximația la pasul următor)

La iterația (n), aproximația devine:

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

Căutarea se oprește când $|x^{(n)} - x^{(n-1)}| < \varepsilon$ sau $|f(x^{(n)})| < \varepsilon$, pt un prag ε ales.

• Dacă ne dorim să determinăm soluția unei ec. neliniare $f(x) = 0$ putem folosi dezvoltarea în serie Taylor a acesteia

$$f(x) = f(a) + \frac{f'(a)}{1!}(x - a) + \dots$$, unde $a \in \mathbb{R}$ este punctul în jurul căruia se dezvoltă fc.

Dacă $|x^* - a| < \varepsilon$, unde ε reprezintă toleranța aproximației, atunci dezvoltarea va aproxima foarte bine funcția chiar și cu un nr. mic de termeni. Dacă soluția este însă depărtată de a, atunci aproximația nu va fi așa de bună, însă procedând se poate reține numai când aproximația va 4) devenii suficient de buna =>

$$f(x)=0 \Rightarrow \left.f(x^{(1)}) \cong f(x^{(0)}) + f'(x^{(0)})(x^{(1)}-x^{(0)})\right\} \Rightarrow$$

$x^{(1)}$ - aproximare a soluției => $f(x^{(1)}) \cong 0$

$$f(x^{(0)}) + f'(x^{(0)})(x^{(1)}-x^{(0)}) \cong 0 \Rightarrow$$

$$f(x^{(0)}) + f'(x^{(0)})x^{(1)} - f'(x^{(0)})x^{(0)} \cong 0 \Leftrightarrow$$

$$x^{(1)} = \frac{x^{(0)} f'(x^{(0)}) - f(x^{(0)})}{f'(x^{(0)})} = x^{(0)} - \frac{f(x^{(0)})}{f'(x^{(0)})}$$

$x^{(1)}$ - s-a apropiat de x^* => Procedeul se repetă până când $|f(x^{(n)})| < \varepsilon$

- <u>Sisteme neliniare</u>

$$\begin{cases} f_1(x_1, x_2, \dots, x_n) = 0 \\ \vdots \\ f_n(x_1, x_2, \dots, x_n) = 0 \end{cases}$$

Notații: $x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$, $F(x) = \begin{pmatrix} f_1(x) \\ f_2(x) \\ \vdots \\ f_n(x) \end{pmatrix}$

$F(x) = 0$

- <u>Metoda tangentei (metoda Newton-Raphson):</u>

Există o formulă asemănătoare formulei Taylor pentru un sistem de funcții =>

$$F(x) = F(x^{(n)}) + J(x^{(n)})(x - x^{(n)}) + \dots$$

unde $J(x) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \dots & \frac{\partial f_1}{\partial x_n} \\ \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \dots & \frac{\partial f_n}{\partial x_n} \end{pmatrix}$ - Jacobianul fc $F(x)$

P.p că $x^{(n+1)}$ - soluția sistemului =>

$$F(x^{(n+1)}) = 0 = F(x^{(n)}) + J(x^{(n)})(x^{(n+1)} - x^{(n)}) \Leftrightarrow$$

$$J(x^{(n)})x^{n+1} = J(x^{n})x^{(n)} - F(x^{(n)})$$

Scanat cu CamScanner

Se rezolvă SEL și apoi apoi se testează dacă soluția x^{n+1} este satisfăcătoare sau nu. În caz contrar procedeul se repetă.

Drept date se introduce o rază de toleranță soluției și un $x^{(0)}$ - o aproximație inițială a soluției.

Ex: $\begin{cases} x_1^2 + x_2 + 1 = 0 \\ 2x_1^2 + 3 = 0 \end{cases}$ $\quad x^{(0)} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \; x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$

$$F(x) = \begin{pmatrix} f_1(x_1, x_2) \\ f_2(x_1, x_2) \end{pmatrix} = \begin{pmatrix} x_1^2 + x_2 + 1 \\ 2x_1^2 + 3 \end{pmatrix} \Rightarrow$$

$$J(x) = \begin{pmatrix} 2x_1 & 1 \\ 4x_1 & 0 \end{pmatrix} \Rightarrow$$

$$J(x^{(0)}) \cdot x^{(1)} = J(x^{(0)})\, x^{(0)} - F(x^{(0)}) \Leftrightarrow$$

$$\begin{pmatrix} 2 & 1 \\ 4 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 2 & 1 \\ 4 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 2 \end{pmatrix} - \begin{pmatrix} 4 \\ 5 \end{pmatrix} \Leftrightarrow$$

$$\begin{pmatrix} 2 & 1 \\ 4 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 4 \\ 4 \end{pmatrix} - \begin{pmatrix} 4 \\ 5 \end{pmatrix} = \begin{pmatrix} 0 \\ -1 \end{pmatrix} \Leftrightarrow$$

$$\left(\begin{array}{cc|c} 2 & 1 & 0 \\ 4 & 0 & -1 \end{array}\right) \to \left(\begin{array}{cc|c} 2 & 1 & 0 \\ 0 & -2 & -1 \end{array}\right) \to \left(\begin{array}{cc|c} 1 & \frac{1}{2} & 0 \\ 0 & 1 & \frac{1}{2} \end{array}\right) \to \left(\begin{array}{cc|c} 1 & 0 & -\frac{1}{4} \\ 0 & 1 & \frac{1}{2} \end{array}\right)$$

$$\Rightarrow x^{(1)} = \begin{pmatrix} -0,25 \\ 0,5 \end{pmatrix}$$

$$\text{cond } A = \|A\| \, \|A^{-1}\|$$

$$\varepsilon_x = \frac{\|\tilde{x} - x\|}{\|x\|} \; ; \; \varepsilon_x \leq \text{cond}(A)\, \varepsilon_b$$

Scanat cu CamScanner

- **Vectori și valori proprii**

$A \in \mathbb{R}^{n\times n}$, $Ax = \lambda x$ - ecuație caracteristică, $x \neq 0$

λ - val proprie, x - vector propriu asociat

$Ax = \lambda x \Leftrightarrow Ax - \lambda x = 0 \Leftrightarrow x(A - \lambda I) = 0 \Rightarrow$

$p_A(\lambda) = |A - \lambda I|$ - polinom caracteristic

$\lambda(A) = \{\lambda_1, \lambda_2, \dots \lambda_n\}$ - spectrul matricei A

$\rho(A) = \max\limits_{i=1:n} \{|\lambda_i|\}$ raza spectrală

- **Proprietăți**

1) $\lambda(A - \mu I) = \{\lambda_1 - \mu, \lambda_2 - \mu, \dots \lambda_n - \mu\}$, $\mu \in \mathbb{C}$

λ - val proprie a matricei $A \Rightarrow$

$Av = \lambda v \;|-\mu v \Leftrightarrow Av - \mu v = \lambda v - \mu v \Leftrightarrow$

$v(A - \mu I) = v(\lambda - \mu) \Leftrightarrow (A - \mu I)v = (\lambda - \mu)v \Rightarrow$

$(\lambda - \mu)$ - val proprie a matricei $(A - \mu I)$

2) $\lambda(A^{-1}) = \left\{\frac{1}{\lambda_1}, \frac{1}{\lambda_2}, \dots, \frac{1}{\lambda_n}\right\}$

λ - val proprie a matricei $A \Rightarrow$

$Av = \lambda v \;| \cdot A^{-1} \Leftrightarrow A^{-1}Av = A^{-1}\lambda v \Leftrightarrow v = \lambda A^{-1} v \Leftrightarrow$

$A^{-1}v = \frac{1}{\lambda} v \Rightarrow \frac{1}{\lambda}$ - val proprie a matricei A^{-1}

3) $\lambda(A^k) = \{\lambda_1^k, \lambda_2^k, \dots \lambda_n^k\}$

λ - val proprie a matricei $A \Rightarrow$

Scanat cu CamScanner

$A v = \lambda v \mid \cdot A \Leftrightarrow A^2 v = \lambda (A v) = \lambda \cdot \lambda v = \lambda^2 v \Rightarrow$

$A^2 v = \lambda^2 v \xRightarrow{\text{inducție}} A^k v = \lambda^k v \Rightarrow$

λ^k - val proprie a matricei A^k

4) $\lambda((A-\mu I)^{-1}) = \left\{ \frac{1}{\lambda_1 - \mu}, \frac{1}{\lambda_2 - \mu}, \dots, \frac{1}{\lambda_n - \mu} \right\}$

$\mu \in \mathbb{C} \setminus \lambda(A)$

λ - val proprie a matricei $A \Rightarrow$

$A v = \lambda v \mid -\mu v \Leftrightarrow (A - \mu I) v = (\lambda - \mu) v \mid (A-\mu I)^{-1}$

$\Leftrightarrow v = (\lambda - \mu)(A - \mu I)^{-1} v \Rightarrow (A - \mu I)^{-1} v = \left(\frac{1}{\lambda - \mu}\right) v \Rightarrow$

$\frac{1}{\lambda - \mu}$ - valoare proprie a matricei $(A - \mu I)^{-1}$,
$\mu \in \mathbb{C} \setminus \lambda(A)$

• Metoda puterii directe (MPD):

Presupunem $|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \dots \geq |\lambda_n|$

MPD calculează (λ_1, x_1)

Se știe că șirul $y^{(0)} \neq 0$; $y^{(1)} = A y^{(0)}$; $y^{(2)} = A y^{(1)} = A^2 y^{(0)} \dots$

$\boxed{y^{(k)} = A y^{(k-1)} = A^{(k)} y^{(0)}}$ - converge către x_1

Demonstrația convergenței acestui șir și găsirea valorii spre care converge nu face obiectul acestor notițe.

Fie $\lambda^{(k)}$ - o aproximație a valorii proprii $\lambda_1 \Rightarrow$

$A y^{(k)} \cong \lambda^{(k)} y^{(k)} \mid (y^{(k)})' \Leftrightarrow (y^{(k)})' A y^{(k)} \cong \lambda^{(k)} (y^{(k)})' y^{(k)} \Leftrightarrow$ 2)

Scanat cu CamScanner

$$\boxed{\lambda^{(k)} = \frac{(y^{(k)})' A y^{(k)}}{(y^{(k)})' y^{(k)}}}$$ - câtul Rayleigh

Se va calcula astfel la pasul k aproximarea $y^{(k)}$ a vectorului propriu x_1 și apoi folosind această valoare se va calcula o aproximare a valorii proprii λ_1 calculând $\lambda^{(k)}$.

Algoritmul este iterativ și folosește o aproximare inițială $y^{(0)}$ a vectorului propriu x_1.

Algoritm MPD

D.I: $A, y^{(0)}$, tol, maxiter

D.O: λ_1, x_1

for $k = 1$: maxiter

$\quad z = A y^{(k-1)}$

$\quad y^{(k)} = z / norm(z)$

$\quad \lambda^{(k)} = (y^{(k)})' A y^{(k)}$ - deoarece s-a normat $y^{(k)}$

$\quad$ if $\| A y^{(k)} - \lambda^{(k)} y^{(k)} \| < tol$

$\quad\quad$ return $(\lambda^{(k)}, y^{(k)})$

• MPD pentru determinarea celei mai mici valori proprii

$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \dots \geq |\lambda_{m-1}| > |\lambda_m|$

O proprietate a vectorilor și valorilor proprii ale unei matrici spune că $\lambda(A^{-1}) = \left\{\frac{1}{\lambda_1}, \frac{1}{\lambda_2}, \dots, \frac{1}{\lambda_m}\right\} \Rightarrow$

$\left|\frac{1}{\lambda_m}\right| > \left|\frac{1}{\lambda_{m-1}}\right| \geq \dots \geq \left|\frac{1}{\lambda_2}\right| > \left|\frac{1}{\lambda_1}\right| \Rightarrow$

Vom aplica MPD asupra matricei A^{-1} și astfel vom deter-

Scanat cu CamScanner

$\min\left(\frac{1}{\lambda_n}\right)$ - cea mai mare valoare proprie a matricei A^{-1}

Algoritm

D.i: $A, y^{(0)}$, maxiter, tol

D.O: $(\lambda n, x n)$

for $K = 1$: maxiter

$z = A^{-1} y^{(K-1)}$ - se poate rezolva SEL $Az = y^{(K-1)}$

$y^{(K)} = z / \text{norm}(z)$

$\lambda^{(K)} = (y^{(K)})' A^{-1} y^{(K)}$

if $\| A^{-1} y^{(K)} - \lambda^{(K)} y^{(K)} \| < tol$

break

return $(1/\lambda^{(K)}, y^{(K)})$

• Metoda puterii inverse (MPI)

MPI calculează cea mai mică valoare proprie a unei matrici A folosind matricea $B = (A - \mu I)^{-1}$, unde μ reprezintă o aproximație a unei valori proprii a matricei A

Algoritm (se aplică un MPD modificat pe matricea $B = (A - \mu I)^{-1}$

D.i: $A, y^{(0)}$, maxiter, tol, $\boxed{\mu}$

D.O: (λ, x)

Se folosește matricea B pt a face această metodă iterativă să convergă către o soluție suficient de exactă mai rapid decât MPD. Valoarea de deplasare μ este modificată de la o iterație la alta utilizând aproximația curentă $\mu = \lambda^{(K)}$ pentru a crește viteza de

Scanat cu CamScanner

convergență a algoritmului.

for $K = 1 :$ maxiter

$z = (A - \mu I)^{-1} \cdot y^{(K-1)}$ - se poate rezolva SEL $(A - \mu I) z = y^{(K-1)}$

$y^{(K)} = z / norm(z)$

$\lambda^{(K)} = (y^{(K)})' A y^{(K)}$

if $\| A y^{(K)} - \lambda^{(K)} y^{(K)} \| < tol$

return $(\lambda^{(K)}, y^{(K)})$

$\mu = \lambda^{(K)}$

• <u>Metoda deflației</u>

Metoda deflației se folosește doar în cazul matricelor simetrice, cu elementele reale. Folosind metoda puterii directe se determină valoarea proprie dominantă și vectorul propriu corespunzător. Pentru a afla celelalte perechi proprii se procedează astfel:

• inițial avem o matrice A cu vectorii proprii $x_1, x_2, \dots, x_n$ și valorile proprii $\lambda_1, \lambda_2, \dots, \lambda_n$. Pe λ_1 și x_1 îi calculăm folosind metoda puterii directe.

• Construim matricea $B = (I_n - x_1 y^T) A$, cu y ales a.î. $x_1^T y = 1$. Această matrice va avea valorile proprii $\{0, \lambda_2, \lambda_3, \dots, \lambda_n\}$ și vectorii proprii $\{x_1, z_2, z_3, \dots, z_n\}$, unde $z_i = x_i - y^T x_i x_1$, $\forall$ $i = 2:n$

• Dacă din matricea B eliminăm prima linie și prima coloană, matricea obținută de rang $n-1$, își păstrează perechile proprii mai puțin prima pereche proprie $(0, x_1)$. Pe matricea redusă, putem aplica același procedeu, obținându-se valoarea proprie dominantă λ_2. Procesul continuă până se obțin toate valorile proprii ale matricii inițiale.

Scanat cu CamScanner

- Cercurile Gershgorin

Localizarea valorilor proprii

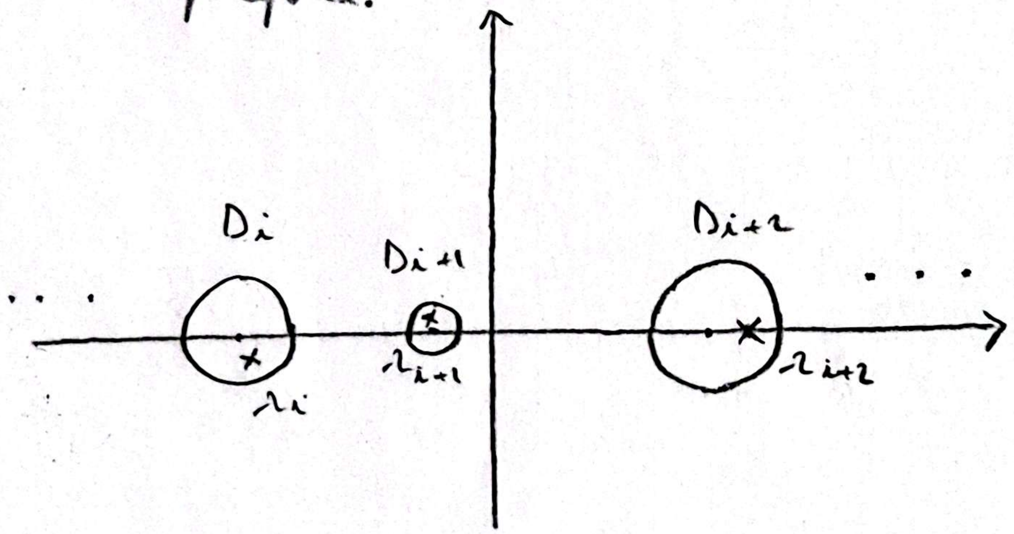
Fie $A \in \mathbb{R}^{n \times n}$ și cercurile Gershgorin definite astfel:

$$D_i = \underbrace{\left\{ z \in \mathbb{C} \,/\, |z - a_{ii}| \le \sum_{\substack{j=1 \\ j \ne i}}^{n} |a_{ij}| \right\}}_{\text{disc (dar aparent i se spune cerc în acest material)}} \quad i = 1:n$$

D_i are centrul $(a_{ii}, 0)$ și raza $\sum_{\substack{j=1 \\ j \ne i}}^{n} |a_{ij}|$

Atunci, $\lambda(A) \subset \bigcup_{i=1}^{n} D_i$

Orice cerc Gershgorin disjunct conține o singură valoare proprie.



• Matrice asemenea

Două matrice $A, B \in \mathbb{R}^{m \times m}$ sunt asemenea dacă există matricea nesingulară $T \in \mathbb{R}^{m \times m}$ a.î.

$$B = T^{-1} A T$$

• Proprietăți matrice asemenea

1) $\lambda(A) = \lambda(B)$, unde A, B - asemenea $B = T^{-1} A T$

$\lambda(A)$ - mulțimea valorilor proprii a matricei A - mulțimea soluțiilor ecuației $|A - \lambda I_m| = 0$

$|A - \lambda I_m| = \det(A - \lambda I_m) = \det(A - \lambda I_m) \cdot \det(I_m) =$

$= \det(A - \lambda I_m) \cdot \det(T \cdot T^{-1}) = \det(T) \cdot \det(A - \lambda I_m) \cdot \det(T^{-1})$

$= \det(TAT^{-1} - \lambda T I_m T^{-1}) = \det(B - \lambda I_m) = |B - \lambda I_m|$

Matricele A și B au același polinom caracteristic $\Rightarrow$

$\lambda(A) = \lambda(B)$

2) Fie (λ, x) - pereche proprie pentru matricea A

(λ, y) - pereche proprie pentru matricea B

$$By = \lambda y \Leftrightarrow T^{-1} A T y = \lambda y \Leftrightarrow T^{-1} \cancel{\lambda} x T y = \cancel{\lambda} y \Leftrightarrow$$

$x T y = T y$

$B(T^{-1} x) = T^{-1} A \cancel{T}\cancel{T^{-1}} x = T^{-1} A x = T^{-1} \lambda x \Rightarrow$

$B(T^{-1} x) = \lambda (T^{-1} x) \Rightarrow y = T^{-1} x \Rightarrow \boxed{x = Ty}$

• Forma superior Hessenberg

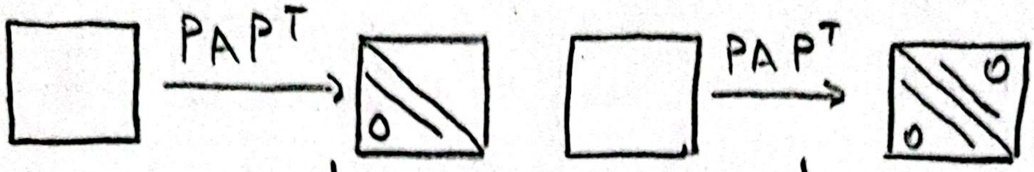
Aducerea unei matrice la forma superior Hessenberg

O matrice $A \in \mathbb{R}^{m \times m}$ poate fi adusă la forma superior Hessenberg H_u folosind transformarea de asemănare ortogonală

Scanat cu CamScanner

$H_n = PAP^T$

P- matrice ortogonală



A - nesimetrică

A simetrică

tridiagonală simetrică (superior Hessenberg + simetrică)

Fie $\boxed{P = P_{n-2}\cdot \ldots \cdot P_2 \cdot P_1}$, unde $P_1, P_2, \ldots, P_{n-2}$ sunt matrice elementare Householder

$$PAP^T = P_{n-2}\cdot\ldots\cdot \underbrace{P_2 \underbrace{P_1 A P_1^T} P_2^T} \cdot \ldots \cdot P_{n-2}^T$$

Fie $A_0 = A$

$A_1 = P_1 A_0 P_1^T$

$A_2 = P_2 A_1 P_2^T$

$\vdots$

$H_n = P_{n-2} A_{n-3} P_{n-2}^T$

<u>Pasul k:</u>

Fie $P_k = \begin{pmatrix} I_k & 0 \\ 0 & \hat{P}_k \end{pmatrix}$

$\hat{P}_k = ?$ a.î. $\hat{P}_k \begin{pmatrix} a_{k+1\,k} \\ a_{k+2\,k} \\ \vdots \\ a_{nk} \end{pmatrix} = \begin{pmatrix} x \\ 0 \\ \vdots \\ 0 \end{pmatrix}$

$\hat{P}_k$ - matricea elementară Householder de dimensiune $(n-k)\times(n-k)$

$v_k = \begin{pmatrix} v_{k+1\,k+1} \\ \vdots \\ v_{nk} \end{pmatrix} \in \mathbb{R}^{n-k}$

$$\hat{P}_k = I_{n-k} - 2\frac{v_k v_k^T}{v_k^T v_k}$$

$v_{k+1\,k+1} = a_{k+1\,k} + \sigma_k$

$\sigma_k = \text{sgn}(a_{k+1\,k}) \sqrt{\sum_{i=k+1}^{n} a_{ik}^2}$; $v_{ik} = a_{ik}$, $i = k+2:n$

Scanat cu CamScanner

• **Algoritmul QR**

Forma Schur reală a unei matrice

Există matricea ortogonală $Q \in \mathbb{R}^{n \times n}$ a.î. $S = Q^T A Q$, unde S se numește forma Schur reală a matricei A. (este matrice cvasi superior triunghiulară)

$$S = \begin{pmatrix} s_{11} & s_{12} & \dots & s_{1n} \\ & s_{22} & \dots & s_{2n} \\ 0 & & \ddots & \vdots \\ & & & s_{nn} \end{pmatrix}$$

• **Observații și precizări:**

Se cunoaște faptul că două matrici asemenea A și B ($B = T^{-1} A T$, T-nesingulară) au aceleași valori proprii ($\lambda(A) = \lambda(B)$)

Pentru a aduce o matrice $A \in \mathbb{R}^{n \times n}$ la forma superior Hessenberg ne folosim de matrici elementare Householder. Cum o matrice elementară Householder este ortogonală (P_K^T - inversa matricei P_K) $\Rightarrow$

O transformare de tipul $A_0 \to A_1$, $A_1 = P_1 A_0 P_1^T$ nu modifică valorile proprii ale matricei rezultate $\lambda(A_0) = \lambda(A_1)$, deoarece cele două matrici sunt asemenea. $\Rightarrow$

H_u - asemenea cu A $\Rightarrow \lambda(H_u) = \lambda(A)$

Metodele ce vor fi prezentate în continuare se vor folosi de transformări Householder și Givens adiționale aducerii matricei A la forma superior Hessenberg pentru a determina valorile proprii ale matricei A ce se vor afla pe diagonala principală a transformării.

Scanat cu CamScanner

După un număr de iterații matricea transformată va conține elemente apropiate de valorile reale proprii pe diagonala principală și valori aproape nule în rest. După cum se știe, o matrice diagonală își are valorile proprii pe diagonală.

• <u>Algoritmul QR fără deplasare pentru matrice nesimetrică A</u>

$A_0 = H_n$

$A_0 = Q_0 R_0$

$A_1 = R_0 Q_0 = Q_1 R_1$

$A_2 = R_1 Q_1 = Q_2 R_2$

...

$A_K = R_{K-1} Q_{K-1}$

A_1 este ortogonal asemenea cu A_0

$$\left.\begin{array}{l} A_1 = R_0 Q_0 \\ A_0 = Q_0 R_0 \Rightarrow R_0 = Q_0^T A_0 \end{array}\right\} \Rightarrow \boxed{A_1 = Q_0^T A_0 Q_0}$$

A_2 este ortogonal asemenea cu A_0

$$\left.\begin{array}{l} A_2 = R_1 Q_1 \\ A_1 = Q_1 R_1 \Rightarrow R_1 = Q_1^T A_1 \end{array}\right\} \Rightarrow A_2 = Q_1^T A_1 Q_1 =$$

$$= Q_1^T Q_0^T A_0 Q_0 Q_1 = (Q_0 Q_1)^T A_0 (Q_0 Q_1)$$

...

<u>Algoritm</u>

calculăm H_n

$A_0 = H_n$

for $K = 1 : maxiter$

$[Q_{K-1}\ R_{K-1}] = householder(A_{K-1})$

$A_K = R_{K-1} Q_{K-1}$

→ test convergență

Scanat cu CamScanner

• **Algoritm QR cu deplasare simplă pentru matrice nesimetrică A**

Algoritm

calculăm H_n

$A_0 = H_n$

for $K = 0$: maxiter

$\mu_K = A_K(m, m)$

$[Q_K\ R_K] = \text{householder}(A_K - \mu_K I)$

$A_{K+1} = R_K Q_K + \mu_K \cdot I$

→ test convergență

• **Algoritmul QR fără deplasare pentru matrice simetrică A**

calculare H_n

$A_0 = H_n$

for $K = 0$: maxiter

$[Q_K\ R_K] = \text{householder}(A_K)$

$A_{K+1} = R_K \cdot Q_K$

→ test convergență

• **Algoritmul QR cu deplasare simplă pentru matrice simetrică A**

calculare H_n

$A_0 = H_n$

for $K = 0$: maxiter

calculare μ_K

$[Q_K\ R_K] = \text{householder}(A_K - \mu_K I_n)$

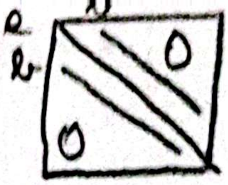
$A_{K+1} = R_K Q_K + \mu_K I_n$

→ test convergență

Calculare μ_K

Scanat cu CamScanner

Fie $E^{(k)} = \begin{pmatrix} a_{m-1}^{(k)} & b_{m-1}^{(k)} \\ b_{m-1}^{(k)} & a_m^{(k)} \end{pmatrix}$



AK

$$|E^{(k)} - \lambda I_2| = \dots = \lambda^2 - \lambda\left(a_{m-1}^{(k)} + a_m^{(k)}\right) + a_{m-1}^{(k)} a_m^{(k)} - \left(b_{m-1}^{(k)}\right)^2$$

roots $\Rightarrow \lambda_1, \lambda_2$

μ_k va fi egal cu valoarea proprie (λ_1 sau λ_2) cea mai apropiată de elementul $a_m^{(k)}$

• SVD

• Descompunerea valorilor singulare

$A = U \Sigma V^T$, $A \in \mathbb{R}^{m \times n}$

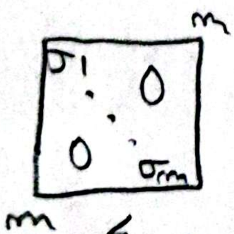
$U = [u_1\ u_2\ \dots\ u_m] \in \mathbb{R}^{m \times m}$ - matrice ortogonală

$u_1, u_2, \dots, u_m$ se numesc vectori singulari stânga ai matricei A

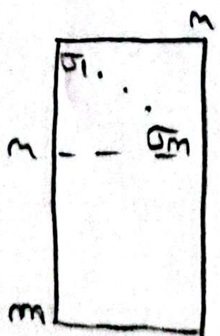
$V^T = \begin{pmatrix} v_1^T \\ v_2^T \\ \dots \\ v_n^T \end{pmatrix} \in \mathbb{R}^{n \times n}$ - matrice ortogonală

$v_1, v_2, \dots, v_n$ se numesc vectori singulari dreapta ai matricei A

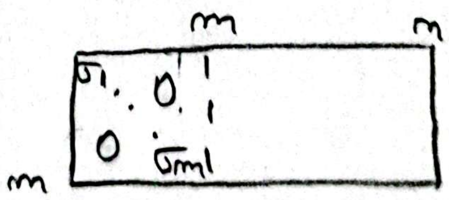
$\Sigma = \operatorname{diag}(\sigma_1, \sigma_2, \dots, \sigma_p) \in \mathbb{R}^{m \times n}$, $p = \min(m, n)$



Σ, $m = n$



Σ, $m > n$



Σ, $m < n$

Scanat cu CamScanner

$\sigma_1, \sigma_2, \dots, \sigma_p$ se numesc valorile singulare ale matricei A

$\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_r > 0,\ \sigma_{r+1} = \sigma_{r+2} = \dots = \sigma_p = 0$

<u>Calculare Σ</u>

$\boxed{\sigma_i = \sqrt{\lambda_i},\ \text{unde } \lambda_i \in \lambda(A^TA)}$

<u>Dem:</u>

$A = U\Sigma V^T \,|\cdot V \iff AV = U\Sigma \iff$

$A[v_1\ v_2 \dots v_m] = [u_1\ u_2 \dots u_m]\,\Sigma \iff$

$[Av_1\ Av_2 \dots Av_m] = [\sigma_1 u_1\ \sigma_2 u_2 \dots \sigma_m u_m] \iff$

$Av_i = \sigma_i u_i,\ i = \overline{1:m}$ (1)

$A = U\Sigma V^T \,|^T \iff A^T = (U\Sigma V^T)^T = V(U\Sigma)^T =$

$= V\Sigma U^T \,|\, U \iff A^TU = V\Sigma \iff A^Tu_i = \sigma_i v_i, i = \overline{1:m}$ (2)

Din (1) $\Rightarrow Av_i = \sigma_i u_i \,|\cdot A^T \iff (A^TA)v_i = \sigma_i(A^Tu_i) \overset{(2)}{\Rightarrow}$

$(A^TA)v_i = \sigma_i^2 v_i \Rightarrow \sigma_i^2 = \lambda_i \Rightarrow \sigma_i = \sqrt{\lambda_i}$, unde

λ_i - valoare proprie a matricei (A^TA)

<u>Calculare V</u>

$(A^TA)v_i = \lambda_i v_i,\ i = \overline{1:m} \Rightarrow v_i$ - sunt vectorii proprii

ai matricei A^TA, $V = [v_1\ v_2 \dots v_m]$

<u>Calculare U</u>

Din (1) $\Rightarrow Av_i = \sigma_i u_i \Rightarrow \boxed{u_i = \frac{1}{\sigma_i} Av_i,\ i = \overline{1:r}}$

Scanat cu CamScanner

$U = [u_1\ u_2 \dots u_r\ u_{r+1} \dots u_m]$

$u_{r+1}, \dots, u_m$ se calculează folosind algoritmul Gram-Schmidt

<u>Proprietăți SVD, $A \in \mathbb{R}^{m \times n}$</u>

1) $\|A\|_2 = \sigma_1$

$\|A\|_2 = \|U \Sigma V^T\|_2 = \|\Sigma V^T\|_2 = \|\Sigma\|_2 = \sqrt{\rho(\Sigma^T \Sigma)} =$

$= \sqrt{\max\{\sigma_1^2, \sigma_2^2, \dots, \sigma_n^2\}} = \sigma_1$

2) $\|A\|_F = \sqrt{\sigma_1^2 + \sigma_2^2 + \dots + \sigma_n^2}$

$\|A\|_F = \|U \Sigma V^T\|_F = \|\Sigma\|_F = \sqrt{\sum_i \sigma_i^2}$

3) $\|A^{-1}\|_2 = \frac{1}{\sigma_n}$

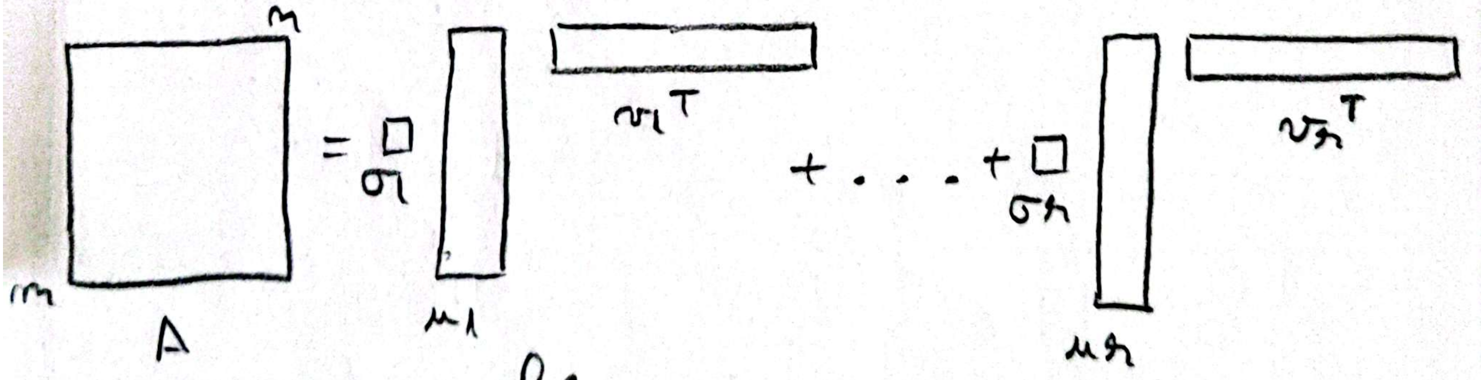
4) $\text{cond}(A) = \|A\|_2 \cdot \|A^{-1}\|_2 = \frac{\sigma_1}{\sigma_n}$

5) $\text{rang}(A) =$ nr. de valori singulare > 0

- <u>Teoremă</u>

Fie $A \in \mathbb{R}^{m \times n}$

$A = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \dots + \sigma_r u_r v_r^T$



<u>Compresia imaginilor</u>

Fie $A_k = \sigma_1 u_1 v_1^T + \dots + \sigma_k u_k v_k^T$, $k < r$

$A_k = U_k \Sigma_k V_k^T$, $U_k \in \mathbb{R}^{m \times k}$, $\Sigma_k \in \mathbb{R}^{k \times k}$, $V_k^T \in \mathbb{R}^{k \times n}$, $A^k \in \mathbb{R}^{m \times n}$

Scanat cu CamScanner

Matricea A_k va aproxima matricea A. Pentru a compune matricea A, vom salva $m \times n$ elemente. Pentru a compune matricele U_k, Σ_k, V_k^T, vom salva $mk + k + nk = (m+1+n)k$ elemente

Factorul de compresie: $\frac{(m+1+n)k}{mn}$

Scanat cu CamScanner

• Interpolare introducere

Fie funcția $f:[x_0, x_m] \to \mathbb{R}$ cunoscută doar în $x_0, x_1, \ldots, x_m$ prin valorile $f(x_0), f(x_1), \ldots, f(x_m)$.

$$x_0 < x_1 < \ldots < x_m$$

$\{x_0, x_1, \ldots, x_m\}$ suportul interpolării

Fie polinomul de interpolare de gradul m

$$P_m(x) = a_0 u_0(x) + a_1 u_1(x) + \ldots + a_m u_m(x), \text{ unde}$$

$\{u_0(x), u_1(x), \ldots, u_m(x)\}$ baza de interpolare

Se pune problema aproximării funcției $f(x)$ pe mulțimea $[x_0, x_m] \setminus \{x_0, x_1, \ldots, x_m\}$

Ne dorim ca $f(x) \simeq P_m(x)$ și să putem determina o valoare aproximativă a funcției f în punctul $x \in [x_0, x_m]$

Pentru a calcula coeficienții $a_0, a_1, \ldots, a_m$, se folosesc condițiile de interpolare:

$$f(x_i) = P_m(x_i), \; i = \overline{0:m}$$

• interpolare polinomială

Se folosește baza de interpolare $\{1, x, \ldots, x^m\}$

Pt a calcula coeficienții $a_0, a_1, \ldots a_m$ se folosesc condițiile de interpolare $f(x_i) = P_m(x_i), \; i = \overline{0:m}$

$$P_m(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_m x^m$$

Scanat cu CamScanner

$$\left.\begin{cases} a_0 + a_1 x_0 + a_2 x_0^2 + \dots + a_m x_0^m = f(x_0) \\ a_0 + a_1 x_1 + a_2 x_1^2 + \dots + a_m x_1^m = f(x_1) \\ \dots \\ a_0 + a_1 x_m + a_2 x_m^2 + \dots + a_m x_m^m = f(x_m) \end{cases}\right\} \Rightarrow$$

Se obține un sistem de $m+1$ ecuații liniare cu $m+1$ necunoscute $(a_0, a_1, \dots, a_m)$

$$\underbrace{\begin{pmatrix} 1 & x_0 & x_0^2 & \dots & x_0^m \\ 1 & x_1 & x_1^2 & \dots & x_1^m \\ \dots \\ 1 & x_m & x_m^2 & \dots & x_m^m \end{pmatrix}}\begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_m \end{pmatrix} = \begin{pmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_m) \end{pmatrix}$$

matrice Vandermonde (matrice rău condiționată)

Această metodă încearcă să aproximeze orice funcție f folosind un polinom de grad m unde m reprezintă numărul condițiilor de interpolare -1 (numărul de valori cunoscute ale funcției $f - 1$)

- <u>interpolare Lagrange</u>

Se folosește baza de interpolare $\{L_0(x), L_1(x), \dots, L_m(x)\}$

$$L_0(x) = \frac{(x-x_1)(x-x_2)\cdot\ldots\cdot(x-x_m)}{(x_0-x_1)(x_0-x_2)\cdot\ldots\cdot(x_0-x_m)}$$

...

$$L_m(x) = \frac{(x-x_0)(x-x_1)\cdot\ldots\cdot(x-x_{m-1})}{(x_m-x_0)(x_m-x_1)\cdot\ldots\cdot(x_m-x_{m-1})}$$

$\boxed{L_K(x) = \prod_{\substack{i=0 \\ i\neq K}}^{m} \frac{x-x_i}{x_K-x_i}}$ - multiplicator Lagrange (polinom de gradul m)

Scanat cu CamScanner

$P_m(x) = \sum_{k=0}^{m} a_k u_k(x)$ polinom de interpolare Lagrange (polinom de gradul m), unde

$u_k(x) = L_k(x),\ k = \overline{0:m}$

$a_k = f(x_k)$

$$P_m(x_k) = f(x_k) = a_k \cdot \prod_{\substack{i=0 \\ i \neq k}}^{m} \frac{\cancel{x_k - x_i}}{\cancel{x_k - x_i}} = a_k \Rightarrow a_k = f(x_k)$$

$$\boxed{P_m(x) = \sum_{k=0}^{m} f(x_k) \prod_{\substack{i=0 \\ i \neq k}}^{m} \frac{x - x_i}{x_k - x_i}}$$

• Interpolare Newton

Pentru a putea deduce formula de interpolare Newton vom defini mai întâi diferențele divizate

Diferențe divizate

F_0 (dif. div. de ord 0) $\quad$ F_1 (dif. div. de ord 1) $\quad$ F_2 (dif. div. de ord 2)

$$F_0[x_0] = f(x_0) \rightarrow F_1[x_0, x_1] = \frac{F_0[x_0] - F_0[x_1]}{x_0 - x_1} \rightarrow F_2[x_0, x_1, x_2] = \frac{F_1[x_0, x_1] - F_1[x_1, x_2]}{x_0 - x_2} \ \ldots$$

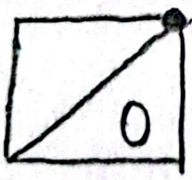
$$F_0[x_1] = f(x_1)$$

$$F_0[x_2] = f(x_2) \rightarrow F_1[x_1, x_2] = \frac{F_0[x_1] - F_0[x_2]}{x_1 - x_2}$$

...

$$F_0[x_m] = f(x_m)$$

Diferențele divizate pot fi organizate sub forma unei matrici.



$$\rightarrow F_m[x_0, x_1, \ldots, x_m] = \frac{F_{m-1}[x_0, x_1, \ldots, x_{m-1}] - F_{m-1}[x_1, x_2, \ldots, x_m]}{x_0 - x_m}$$

Pentru a determina formula polinomului de interpolare vom proceda astfel:

Scanat cu CamScanner

$$F_1[x, x_0] = \frac{F_0[x] - F_0[x_0]}{x - x_0} = \frac{f(x) - f(x_0)}{x - x_0} \Rightarrow$$

$$(x - x_0)\, \cancel{F_1[x, x_0]} = f(x) - f(x_0)$$

$$(x - x_1)\, \cancel{F_2[x, x_0, x_1]} = \cancel{F_1[x, x_0]} - F_1[x_0, x_1] \quad |(x - x_0)$$

$$(x - x_2)\, \cancel{F_3[x, x_0, x_1, x_2]} = \cancel{F_2[x, x_0, x_1]} - F_2[x_0, x_1, x_2] \quad |(x - x_0)(x - x_1)$$

$$\dots$$

$$(x - x_m)\, F_{m+1}[x, x_0, \dots, x_m] = F_m[\cancel{x, x_0, \dots, x_{m-1}}] - F_m[x_0, \dots, x_m]$$

$$|(x - x_0) \cdot \dots \cdot (x - x_{m-1})$$

După însumarea tuturor egalităților se va obține:

$$f(x) = \underline{f(x_0) + F_1[x_0, x_1](x - x_0) + F_2[x_0, x_1, x_2](x - x_0)(x - x_1) +}$$

$$\underline{\dots + F_m[x_0, \dots, x_m](x - x_0)(x - x_1) \cdot \dots \cdot (x - x_{m-1})} +$$

$$+ \underline{\underline{F_{m+1}[x, x_0, \dots, x_m](x - x_0) \cdot \dots \cdot (x - x_m)}}$$

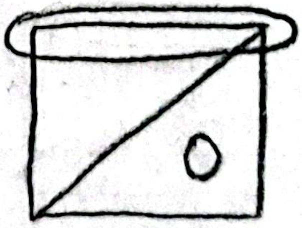
$f(x) = \underline{P_m(x)} + \underline{\underline{E(x)}}$, unde $P_m(x)$ este polinomul de interpolare, iar $E(x)$ este eroarea interpolării

Baza de interpolare este $\{u_0(x), u_1(x), \dots, u_m(x)\} =$

$$= \{1, (x - x_0), (x - x_0)(x - x_1), \dots, (x - x_0)(x - x_1) \cdot \dots \cdot (x - x_{m-1})\}$$

$$\left.\begin{aligned} a_0 &= f(x_0) = F_0[x_0] \\ a_1 &= F_1[x_0, x_1] \\ a_2 &= F_2[x_0, x_1, x_2] \\ &\vdots \\ a_m &= F_m[x_0, x_1, \dots, x_m] \end{aligned}\right\}$$

Prima linie din coloana matricei de diferențe divizate



Scanat cu CamScanner

• Polinoame Bernstein de gradul n

$$\boxed{B_{i,n}(t) = C_n^i (1-t)^{n-i} t^i, \ i = 0:n, \ t \in [0,1]}$$

$n = 1$ (Polinoame Bernstein de gradul 1)

$B_{0,1}(t) = C_1^0 (1-t)^1 t^0 = 1-t$

$B_{1,1}(t) = C_1^1 (1-t)^0 t^1 = t$

$n = 2$ (Polinoame Bernstein de gradul 2)

$B_{0,2}(t) = C_2^0 (1-t)^2 t^0 = (1-t)^2$

$B_{1,2}(t) = C_2^1 (1-t)^1 t^1 = 2(1-t)\cdot t$

$B_{2,2}(t) = C_2^2 (1-t)^0 t^2 = t^2$

...

• interpolare liniară

Un spline $S: [x_0, x_n] \to \mathbb{R}$ este o funcție

$$S(x) = \begin{cases} S_0(x), & x \in [x_0, x_1] \\ S_1(x), & x \in [x_1, x_2] \\ \dots \\ S_{n-1}(x), & x \in [x_{n-1}, x_n] \end{cases}, \text{ unde } S_i : [x_i, x_{i+1}] \to \mathbb{R}$$

Dacă $S_i(x) = a_i x + b_i$, $i = 0:n-1 \Rightarrow$ S - spline liniar

Calcularea coeficienților a_i, b_i, $i = 0:n-1$ ($2n$ coeficienți)

Metoda 1

Condiții de interpolare: $S_i(x_i) = f(x_i), i = \overline{0:n-1}$

$S_{n-1}(x_n) = f(x_n)$

condiții de racordare (polinoamele create creează o curbă continuă pe intervalul $[x_0, x_n]$):

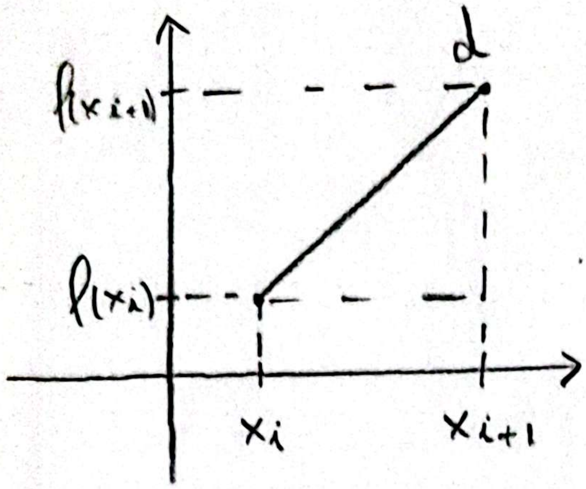
$S_i(x_{i+1}) = S_{i+1}(x_{i+1})$, $i = \overline{0:n-2}$

$\Rightarrow$ Pentru determinarea coeficienților se va rezolva un

Scanat cu CamScanner

Metoda 2

Vom determina fiecare dreaptă ce trece prin două puncte consecutive de forma $(x_i, f(x_i)), (x_{i+1}, f(x_{i+1}))$.



$d: y = a_i x + b_i$

$$\begin{array}{l} f(x_{i+1}) = a_i x_{i+1} + b_i \\ f(x_i) = a_i x_i + b_i \\ \hline \end{array}$$

$$(-) \quad a_i = \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i}$$

$$\begin{array}{l} x_i f(x_{i+1}) = a_i x_i x_{i+1} + b_i x_i \\ x_{i+1} f(x_i) = a_i x_i x_{i+1} + b_i x_{i+1} \\ \hline \end{array}$$

$$(-) \quad b_i = \frac{x_i f(x_{i+1}) - x_{i+1} f(x_i)}{x_i - x_{i+1}} \Rightarrow$$

$$b_i = \frac{x_{i+1} f(x_i) - x_i f(x_{i+1})}{x_{i+1} - x_i} \Rightarrow$$

$$\boxed{a_i = \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i}; \; b_i = \frac{x_{i+1} f(x_i) - x_i f(x_{i+1})}{x_{i+1} - x_i}, \; i = \overline{0 : m-1}}$$

Spline-ul creat va fi folosit pentru a aproxima valoarea funcției în $x \in [x_0, x_n]$ și pentru a construi graficul acesteia.

• <u>Interpolare cubică de clasă C_1</u>

Fie spline-ul cubic $S : [x_0, x_m] \to \mathbb{R}$

$$S(x) = \begin{cases} S_0(x), & x \in [x_0, x_1] \\ S_1(x), & x \in [x_1, x_2] \\ \dots \\ S_{m-1}(x), & x \in [x_{m-1}, x_m] \end{cases}$$

unde $S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3, i = \overline{0:m-1}$

$S_i : [x_i, x_{i+1}] \to \mathbb{R}$

Se dorim să determinăm a_i, b_i, c_i, d_i, $i = \overline{0:m-1}$ ($4m$ necunoscute) astfel încât spline-ul S să aproximeze cât mai bine funcția f.

<u>Condiții de interpolare de tip Hermite:</u>

$$\begin{cases} S_i(x_i) = f(x_i), & i = \overline{0, m-1} \\ S_{m-1}(x_m) = f(x_m) \end{cases} \quad \begin{cases} S_i'(x_i) = f'(x_i), & i = \overline{0:m-1} \\ S_{m-1}'(x_m) = f'(x_m) \end{cases}$$

<u>Condiții de racordare:</u>

$$\begin{cases} S_i(x_{i+1}) = S_{i+1}(x_{i+1}), & i = \overline{0:m-2} \\ S_i'(x_{i+1}) = S_{i+1}'(x_{i+1}), & i = \overline{0:m-2} \end{cases}$$

<u>Spline-ul cubic de clasă C_1 în forma parametrică</u>

Fie $\boxed{h_i = x_{i+1} - x_i \,;\; t = \frac{x - x_i}{h_i}}$ h_i - lungimea intervalului $[x_i, x_{i+1}] \Rightarrow$

$x - x_i = t \cdot h_i \Rightarrow$ Polinomul $S_i(x)$ devine

$$\boxed{S_i(t) = a_i + b_i t h_i + c_i t^2 h_i^2 + d_i t^3 h_i^3, \; i = \overline{0:m-1}}$$

Scanat cu CamScanner

Spline-ul cubic de clasă C_1 în forma parametrică folosind polinoamele Bernstein de gradul 3:

Această scriere ne permite să determinăm direct coeficienții polinomului $S_i(x)$ astfel:

$$S_i(t) = a_i'(1-t)^3 + b_i 3(1-t)^2 t + c_i 3(1-t)t^2 + d_i t^3$$

$$i = \overline{0:n-1} \Rightarrow$$

$$\boxed{\begin{aligned} h_i &= x_{i+1} - x_i \\ a_i' &= f(x_i) \\ b_i' &= f(x_i) + \frac{h_i}{3} f'(x_i) \\ c_i' &= f(x_{i+1}) - \frac{h_i}{3} f'(x_{i+1}) \\ d_i' &= f(x_{i+1}) \end{aligned}}$$

După ce s-a determinat $S_i(t)$ se face trecerea $t \to x \left(t = \frac{x - x_i}{h_i} \right)$

- Interpolare cubică de clasă C_2

Fie spline-ul cubic $S : [x_0, x_n] \to \mathbb{R}$

$$S(x) = \begin{cases} S_0(x), & x \in [x_0, x_1] \\ S_1(x), & x \in [x_1, x_2] \\ \dots \\ S_{n-1}(x), & x \in [x_{n-1}, x_n] \end{cases}$$

unde $S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$, $i = \overline{0:n-1}$

Ne dorim să determinăm a_i, b_i, c_i, d_i, $i = \overline{0:n-1}$

Condiții de interpolare

$$S_i(x_i) = f(x_i), \quad i = \overline{0:n-1}$$

$$S_{n-1}(x_n) = f(x_n)$$

Scanat cu CamScanner

Condiții de racordare

$$S_i(x_{i+1}) = S_{i+1}(x_{i+1}),\ i=\overline{0:m-2}$$

$$S_i'(x_{i+1}) = S_{i+1}'(x_{i+1}),\ i=\overline{0:m-2}$$

$$S_i''(x_{i+1}) = S_{i+1}''(x_{i+1}),\ i=\overline{0:m-2}$$

Spline cubic natural de clasă C2

$$S_0''(x_0) = S_{m-1}''(x_m) = 0$$

Spline cubic tensionat de clasă C2:

$$S_0'(x_0) = f'(x_0)$$

$$S_{m-1}'(x_m) = f'(x_m)$$

Cele 4m necunoscute pot fi determinate prin rezolvarea sistemului de ecuații liniare (SEL) (se construiește o matrice cu coeficienții ecuațiilor A și se rezolvă sistemul $A \cdot coef = b$, unde b este vectorul termenilor liberi)

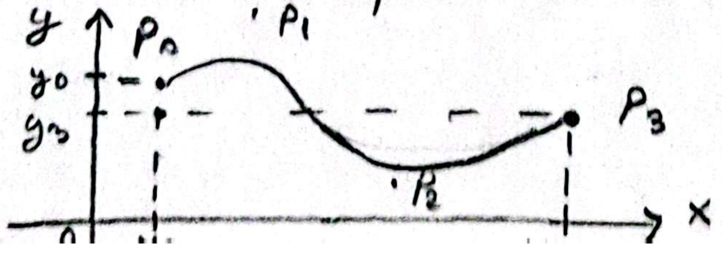
$$coef = \begin{pmatrix} a_0 \\ b_0 \\ c_0 \\ d_0 \\ \vdots \\ a_{m-1} \\ b_{m-1} \\ c_{m-1} \\ d_{m-1} \end{pmatrix}$$; A - matrice cu multe zerouri

• Curbe Bézier

Fie punctele $P_0(x_0,y_0), P_1(x_1,y_1), \dots, P_m(x_m,y_m)$

P_0, P_m - puncte de interpolare

$P_i,\ i=\overline{1:m-1}$ - puncte de control



Scanat cu CamScanner

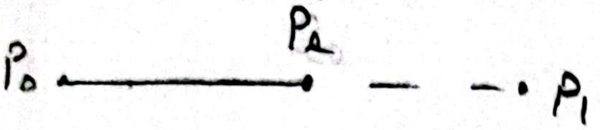
$$\boxed{B_m(t) = \sum_{i=0}^{m} P_i \cdot B_{i,m}(t),\ t \in [0,1]}$$ - curba Bézier de ordinul m

$B_{i,m}(t)$ - polinomul Bernstein de grad m
$i = \overline{0:m}$

Curbele Bézier sunt curbe formate prin interpolarea punctelor obținute prin interpolarea altor puncte.

<u>Curba Bézier liniară</u> $m = 1\ (P_0, P_1)$

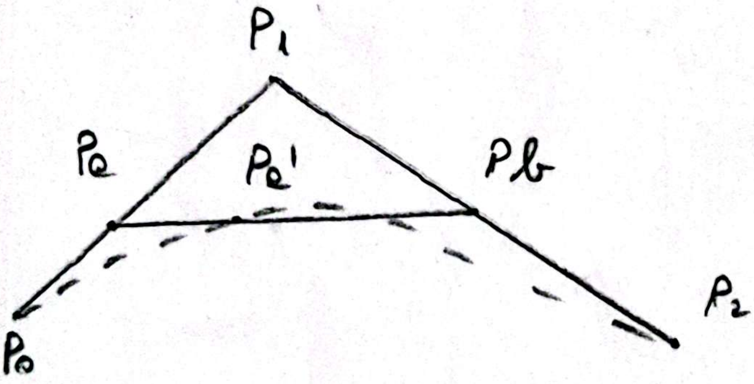
$$B_1(t) = P_0 B_{0,1}(t) + P_1 B_{1,1}(t) = (1-t)P_0 + tP_1,\ t \in [0,1]$$



Variația lui t produce desenarea curbei determinate de $B_1(t)$, P_a fiind punctul $B_1(t')$ pt un $t \in [0,1]$

<u>Curba Bézier pătratică</u> $m = 2\ (P_0, P_1, P_2)$

$$B_2(t) = P_0 B_{0,2}(t) + P_1 B_{1,2}(t) + P_2 B_{2,2}(t)$$
$$B_2(t) = (1-t)^2 P_0 + 2t(1-t)P_1 + t^2 P_2,\ t \in [0,1]$$



P_a' - punctul ce desenează curba formată prin interpolarea punctelor P_a și P_b ce sunt la rândul lor interpolări liniare

$P_0 \longrightarrow P_a = (1-t)P_0 + tP_1 \longrightarrow P_a' = B_2(t) = (1-t)P_a + tP_b$
$P_1 \nearrow$
$P_1 \longrightarrow P_b = (1-t)P_1 + tP_2 \nearrow$
$P_2 \nearrow$

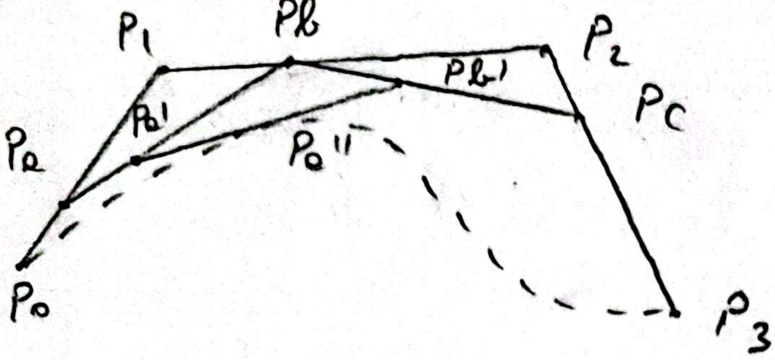
Această curbă este formată prin interpolarea unor puncte care se mișcă când t variază.

Scanat cu CamScanner

Curba Bézier cubică $n=3$ (P_0, P_1, P_2, P_3)

$$B_3(t) = P_0 B_{0,3}(t) + P_1 B_{1,3}(t) + P_2 B_{2,3}(t) + P_3 B_{3,3}(t)$$

$$B_3(t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t) P_2 + t^3 P_3$$



$$\begin{array}{l} P_0 \to P_a = (1-t)P_0 + tP_1 \to P_a' = (1-t)P_a + tP_b \to P_a'' = (1-t)P_a' + tP_b' \\ P_1 \to P_b = (1-t)P_1 + tP_2 \to P_b' = (1-t)P_b + tP_c \qquad = B_3(t) \\ P_2 \to P_c = (1-t)P_2 + tP_3 \\ P_3 \to \\ \ldots \end{array}$$

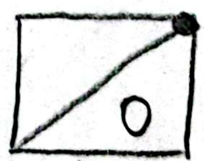
Într-o abordare directă, calcularea unui punct aflat pe o curbă Bézier se obține folosind ecuația parametrică $B(t) = \sum_{i=0}^{n} P_i B_{i,n}(t)$, $t \in [0,1]$. Această metodă este ineficientă deoarece numerele mici ridicate la puteri mari generează erori mari.

Vom folosi algoritmul De Casteljau pentru a rezolva această problemă eficient.

Algoritmul De Casteljau

D.i: $P_0(x_0, y_0), \ldots P_n(x_n, y_n)$ $t \in [0,1]$

```
for j = 2 : n
    for i = 1 : n - j + 1
        Pi^(j) = (1 - t) Pi^(j-1) + t Pi+1^(j-1)
return P1^(n)
```



matricea punctelor $P_i^{(j)}$ formate prin interpolare

Algoritmul aplică procedeul de determinare a curbei $B_n(t) = P^{(n)}$ ... [illegible]

• **Metoda Neville**

Se consideră o funcție $f:[a,b]\to\mathbb{R}$ cunoscută într-o mulțime finită de puncte $x_0, x_1, \dots, x_n$ (suportul interpolării) prin valorile:

$f(x_0), f(x_1), \dots f(x_n)$

Metoda Neville este o metodă de interpolare care aproximează comportamentul funcției f în afara acestor $n+1$ puncte. În cele ce urmează, folosim notația $P_{ij}(x)$ pentru a reprezenta polinomul de interpolare de grad $j-i$ care trece prin punctele $(x_l, f(x_l))$, $l=i, i+1, \dots, j$.

Polinomul de interpolare P_{ij} este dat de relația de recurență:

$$P_{ij}(x) = \frac{x-x_j}{x_i-x_j} P_{i,j-1}(x) + \frac{x_i-x}{x_i-x_j} P_{i+1,j}(x),$$

$$0 \le i < j \le n, \text{ unde } P_{ii}(x) = f(x_i),\ i=\overline{0:n}$$

Recurența continuă până când se obține polinomul de interpolare $P_{0,n}$ care trece prin toate cele $n+1$ puncte $(x_l, f(x_l))$, $l=0,\dots,n$.

Exemplu:

$$\begin{tikzcd}
P_{00}(x)=f(x_0) \arrow[dr] & & & \\
 & P_{01}(x) \arrow[dr] & & \\
P_{11}(x)=f(x_1) \arrow[ur] \arrow[dr] & & P_{02}(x) \arrow[dr] & \\
 & P_{12}(x) \arrow[ur] \arrow[dr] & & P_{03}(x) \\
P_{22}(x)=f(x_2) \arrow[ur] \arrow[dr] & & P_{13}(x) \arrow[ur] & \\
 & P_{23}(x) \arrow[ur] & & \\
P_{33}(x)=f(x_3) \arrow[ur] & & &
\end{tikzcd}$$

Scanat cu CamScanner

C10

• Derivare numerică.

Cunoaștem expresia unei funcții $f(x)$ și ne dorim să determinăm valoarea derivatei acesteia într-un punct x_0 ce aparține domeniului de definiție.

Vom folosi polinomul de interpolare Lagrange cu diferite suporturi de interpolare pentru a aproxima funcția $f(x)$ pe un interval. Procedăm astfel deoarece derivata unui polinom poate fi determinată cu ușurință, iar polinomul de interpolare Lagrange poate oferi o aproximație bună a funcției.

$$f(x) \simeq P_m(x) = \sum_{k=0}^{m} f(x_k) L_k(x) = \sum_{k=0}^{m} f(x_k) \prod_{\substack{i=0 \\ i \neq k}}^{m} \frac{x - x_i}{x_k - x_i}$$

$m = 1 \ (x_0, x_1)$

$$L_0(x) = \frac{x - x_1}{x_0 - x_1} \ ; \ L_1(x) = \frac{x - x_0}{x_1 - x_0} \Rightarrow$$

$$P_1(x) = f(x_0) \frac{x - x_1}{x_0 - x_1} + f(x_1) \frac{x - x_0}{x_1 - x_0}$$

$$\left.\begin{array}{l} f(x) \simeq P_1(x) \Leftrightarrow f'(x_0) \simeq P_1'(x_0) \\ P_1'(x) = \dfrac{f(x_0)}{x_0 - x_1} + \dfrac{f(x_1)}{x_1 - x_0} \end{array}\right\} \Rightarrow$$

$$\left.\begin{array}{l} f'(x_0) \simeq \dfrac{f(x_0)}{x_0 - x_1} + \dfrac{f(x_1)}{x_1 - x_0} \\ \text{Fie } x_1 = x_0 + h \end{array}\right\} \Rightarrow f'(x_0) \simeq \frac{f(x_0)}{-h} + \frac{f(x_0 + h)}{h} = \frac{f(x_0 + h) - f(x_0)}{h} \Rightarrow$$

$$\boxed{f'(x_0) \simeq \frac{f(x_0 + h) - f(x_0)}{h}}$$

– s-a obținut aproximarea cu diferențe finite a derivatei de ordinul I

$\underline{n = 2 \, (x_0, x_1, x_2)}$

$$L_0(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} \; ; \; L_1(x) = \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} ,$$

$$L_2(x) = \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} \Rightarrow$$

$$f(x) \simeq P_2(x) = f(x_0) L_0(x) + f(x_1) L_1(x) + f(x_2) L_2(x) \Rightarrow$$

Pentru $x = x_0, \; x_1 = x_0 + h \;\; x_2 = x_0 + 2h \Rightarrow$

$$f'(x_0) \simeq P_2'(x_0)$$

$$P_2'(x) = f(x_0) \frac{2x - (x_1 + x_2)}{(x_0-x_1)(x_0-x_2)} + f(x_1) \frac{2x - (x_0+x_2)}{(x_1-x_0)(x_1-x_2)} +$$

$$+ f(x_2) \frac{2x - (x_0 + x_1)}{(x_2-x_0)(x_2-x_1)} \Rightarrow$$

$$f'(x_0) \simeq P_2'(x_0) = f(x_0) \frac{\cancel{2x_0} - \cancel{2x_0} - 3h}{(+\cancel{h}) \cdot (+2h)} +$$

$$+ f(x_0+h) \frac{+2h}{\cancel{h} \cdot (+h)} + f(x_0+2h) \cdot \frac{-h}{2h \cdot \cancel{h}} \Rightarrow$$

$$\boxed{f'(x_0) = \frac{1}{h}\left(-\frac{3}{2} f(x_0) + 2 f(x_0+h) - \frac{1}{2} f(x_0+2h)\right)}$$

Pentru $x = x_1, \; x_0 = x_1 - h, \; x_2 = x_1 + h$

$$f'(x_1) \simeq P_2'(x_1) = f(x_1 - h) \frac{-\cancel{h}}{(+\cancel{h})(+2h)} + f(x_1) \cdot 0 +$$

$$+ f(x_1+h) \frac{\cancel{h}}{2h \cdot \cancel{h}} \Rightarrow$$

$$f'(x_1) \simeq \frac{f(x_1+h) - f(x_1-h)}{2h} \; ; \text{ Notăm } x_1 = x_0 \Rightarrow$$

$$\boxed{f'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h}}$$

2)

Scanat cu CamScanner

• **Derivata de ordinul II**

Vom determina o formulă aproximativă pentru derivata de ordinul II a unei funcții $f(x)$ folosind dezvoltarea în serie Taylor a funcției.

$$f(x) \simeq f(x_0) + \frac{f'(x_0)}{1!}(x-x_0) + \frac{f''(x_0)}{2!}(x-x_0)^2 + \dots$$

$x = x_0 + h \Rightarrow$

$$f(x_0+h) = f(x_0) + \frac{f'(x_0)}{1!}\cdot h + \frac{f''(x_0)}{2!}h^2 \quad (1)$$

$x = x_0 - h \Rightarrow$

$$f(x_0-h) \simeq f(x_0) + \frac{f'(x_0)}{1!}(-h) + \frac{f''(x_0)}{2!}h^2 \quad (2)$$

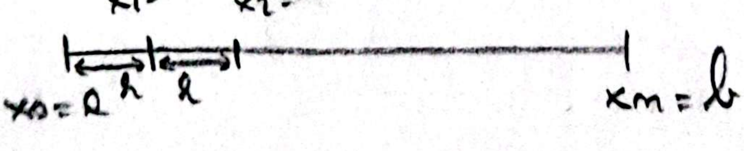
Din (1) și (2) $\Rightarrow f(x_0+h) + f(x_0-h) \simeq 2f(x_0) + f''(x_0)h^2 \Rightarrow$

$$\boxed{f''(x_0) = \frac{f(x_0+h) + f(x_0-h) - 2f(x_0)}{h^2}}$$

• **Integrare numerică - Metode Newton-Cotes**

Fie funcția $f:[a,b] \to \mathbb{R}$ continuă. Ne dorim să determinăm $\int_a^b f(x)\,dx$

Fie $\boxed{x_k = a + kh,\ k = 0:n}$, $\boxed{h = \frac{b-a}{n}}$ - o diviziune a intervalului $[a,b]$ în n subintervale de lungimi egale



Ne dorim să aproximăm integrala definită folosind următoarea sumă:

Scanat cu CamScanner

$$\boxed{\int_a^b f(x)\,dx \simeq \sum_{k=0}^{m} a_k f(x_k)}$$

Problema acestei formule stă în găsirea coeficienților a_k.

Vom folosi polinomul de interpolare Lagrange pentru a aproxima funcția $f(x)$

$$f(x) \simeq P_m(x) = \sum_{k=0}^{m} f(x_k) L_k(x) = \sum_{k=0}^{m} f(x_k) \prod_{\substack{i=0 \\ i \neq k}}^{m} \frac{x - x_i}{x_k - x_i}$$

$$\Rightarrow \int_a^b f(x)\,dx \simeq \int_a^b P_m(x)\,dx = \int_a^b \left(\sum_{k=0}^{m} f(x_k) L_k(x)\right) dx =$$

$$= \sum_{k=0}^{m} f(x_k) \cdot \int_a^b L_k(x)\,dx = \sum_{k=0}^{m} f(x_k)\, a_k \Rightarrow$$

$$\boxed{a_k = \int_a^b L_k(x)\,dx}$$

- <u>Metoda simplă a trapezelor</u>

Dacă $m = 1$ (numarul de diviziuni al intervalului)

$x_0 = a,\ x_1 = b,\ h = b - a \Rightarrow$

$P_1(x) = f(x_0) L_0(x) + f(x_1) L_1(x) \Rightarrow$

$$P_1(x) = f(x_0)\,\frac{x - x_1}{x_0 - x_1} + f(x_1)\,\frac{x - x_0}{x_1 - x_0}$$

Cum $f(x) \simeq P_1(x) \Rightarrow$

$$\int_a^b f(x)\,dx = \int_a^b P_1(x)\,dx = \left(f(x_0)\frac{(x - x_1)^2}{2(x_0 - x_1)} + f(x_1)\frac{(x - x_0)^2}{2(x_1 - x_0)}\right)\Bigg|_a^b =$$

$$+ f(a)\frac{(b-a)^{\cancel{2}}}{2\cancel{(b-a)}} + f(b)\frac{(b-a)^{\cancel{2}}}{2\cancel{(b-a)}} = \frac{(b-a)(f(b) + f(a))}{2} =$$

$$= \frac{h(f(x_0 + h) + f(x_0))}{2},\ h = b - a$$

Scanat cu CamScanner

$$\Rightarrow \boxed{\int_a^b f(x)\,dx \simeq \frac{h}{2}\left(f(x_0) + f(x_0+h)\right) = \frac{b-a}{2}\left(f(a) + f(b)\right)}$$

• <u>Metoda simplă Simpson</u>

Dacă $n = 2$ (două diviziuni ale intervalului)

Se va folosi dezvoltarea în serie Taylor a funcției $f(x)$ pentru a o aproxima

$$x_0 = a,\ x_1 = a + h = \frac{a+b}{2},\ x_2 = b,\ h = \frac{b-a}{2}$$

$$f(x) = f(x_1) + \frac{f'(x_1)}{1!}(x - x_1) + \frac{f''(x_1)}{2!}(x - x_1)^2 + \dots \Rightarrow$$

$$\int_a^b f(x)\,dx = \int_{x_0}^{x_2} f(x)\,dx \simeq \int_{x_0}^{x_2} f(x_1) + \frac{f'(x_1)}{1!}(x - x_1) +$$

$$+ \frac{f''(x_1)}{2!}(x - x_1)^2\,dx =$$

$$= x f(x_1) + f'(x_1)\frac{(x - x_1)^2}{2} + f''(x_1)\frac{(x - x_1)^3}{6}\Big|_{x_0}^{x_2} =$$

$$= (x_1 + h) f(x_1) - (x_1 - h) f(x_1) + \cancel{f'(x_1)\frac{h^2}{2}} - \cancel{f'(x_1)\frac{h^2}{2}} +$$

$$+ f''(x_1)\frac{h^3}{6} + f''(x_1)\frac{h^3}{6} = 2h f(x_1) + f''(x_1)\frac{h^3}{3} =$$

$$= 2h f(x_1) + \frac{h^{\cancel{2}}}{3}\left(\frac{f(x_1 + h) + f(x_1 - h) - 2f(x_1)}{\cancel{h^2}}\right) =$$

$$= \frac{h}{3}\left(f(x_1 + h) + 4f(x_1) + f(x_1 - h)\right) \Rightarrow$$

$$\boxed{\int_a^b f(x)\,dx \simeq \frac{h}{3}\left(f(b) + 4f\left(\frac{b+a}{2}\right) + f(a)\right),\ h = \frac{b-a}{2}}$$

• <u>Formula compusă a trapezelor</u>

$$x_i = a + ih,\ i = 0 : n,\ h = \frac{b-a}{n}$$

$$x_0 = a,\ x_n = b$$

Scanat cu CamScanner

Intervalul de integrare a fost împărțit în n diviziuni de lungime egală => Vom aplica metoda simplă a trapezului pe fiecare subinterval și apoi vom aduna toate integralele calculate.

$$\int_a^b f(x)\,dx = \int_{x_0}^{x_m} f(x)\,dx = \int_{x_0}^{x_1} f(x)\,dx + \int_{x_1}^{x_2} f(x)\,dx + \dots + \int_{x_{m-1}}^{x_m} f(x)\,dx \simeq$$

$$\simeq \frac{h}{2}(f(x_0)+f(x_1)) + \frac{h}{2}(f(x_1)+f(x_2)) + \dots + \frac{h}{2}(f(x_{m-1})+f(x_m)) =$$

$$= \frac{h}{2}\left(f(x_0) + 2\sum_{i=1}^{m-1} f(x_i) + f(x_m)\right) \Rightarrow$$

$$\boxed{\int_a^b f(x)\,dx = \frac{h}{2}\left(f(x_0) + 2\sum_{i=1}^{m-1} f(x_i) + f(x_m)\right),\ h = \frac{b-a}{m}}$$

- Formula compusă Simpson

$x_i = a + ih,\ i = 0:m,\ h = \frac{b-a}{m}$

$x_0 = a,\ x_m = b$

Analog formulei compuse a trapezelor integrala va fi egală cu

$$\int_a^b f(x)\,dx = \int_{x_0}^{x_m} f(x)\,dx = \int_{x_0}^{x_2} f(x)\,dx + \int_{x_2}^{x_4} f(x)\,dx + \dots + \int_{x_{m-2}}^{x_m} f(x)\,dx \simeq$$

$$= \frac{h}{3}(f(x_2) + 4f(x_1) + f(x_0)) + \frac{h}{3}(f(x_4) + 4f(x_3) + f(x_2)) + \dots +$$

$$+ \frac{h}{3}(f(x_m) + 4f(x_{m-1}) + f(x_{m-2})) \Rightarrow$$

$$\boxed{\int_a^b f(x)\,dx = \frac{h}{3}\left(f(x_0) + 4\sum_{i=1}^{\frac{m}{2}} f(x_{2i-1}) + 2\sum_{i=1}^{\frac{m}{2}-1} f(x_{2i}) + f(x_m)\right),\quad h = \frac{b-a}{m}}$$

Scanat cu CamScanner

- <u>Integrale duble</u>

Fie funcția $f:[a,b]\times[c,d]\to\mathbb{R}$. Ne dorim să determinăm $\iint_D f(x,y)\,dA$ și vom folosi în acest demers formula simplă a trapezului:

$$\int_a^b\int_c^d f(x,y)\,dy\,dx = \int_a^b \left(f(x,d)+f(x,c)\right)\frac{d-c}{2}\,dx =$$

$$= \frac{d-c}{2}\,\frac{b-a}{2}\left(f(b,d)+f(b,c)+f(a,d)+f(a,c)\right) \Rightarrow$$

$$\boxed{\int_a^b\int_c^d f(x,y)\,dy\,dx = \frac{(b-a)(d-c)}{4}\left(f(b,d)+f(b,c)+f(a,d)+f(a,c)\right)}$$

- <u>Metoda Romberg</u>

$$\int_a^b f(x)\,dx = ?$$

<u>Matricea Romberg</u> arată astfel:

$$\begin{pmatrix} R_{11} & 0 & 0 & \dots & 0 \\ R_{21} & R_{22} & 0 & & \vdots \\ R_{31} & R_{32} & R_{33} & & \vdots \\ \vdots & & & \ddots & \\ R_{m1} & R_{m2} & \dots & & R_{mm} \end{pmatrix}$$

Prima coloană este formată din aproximări ale integralei R_{K1} $(K=1:m)$ ce se calculează folosind metoda compusă a trapezelor pt 2^{K-1} subintervale.

$$R_{11} = \frac{b-a}{2}\left(f(b)+f(a)\right)$$

$$R_{21} = \frac{b-a}{4}\left(f(b)+2f\left(\frac{b+a}{2}\right)+f(a)\right)$$

$$\dots$$

$$R_{K1} = \frac{b-a}{2^K}\left(f(a)+2\sum_{i=1}^{2^{K-1}-1} f(x_i)+f(b)\right)$$

Scanat cu CamScanner

Celelalte coloane sunt calculate în mod recursiv folosind coloana anterioară, aproximațiile îmbunătățindu-se pe măsură ce crește indicele coloanei curent

În final $R_{mm} \simeq \int_a^b f(x)\,dx$; R_{m} - cea mai bună aproximare

$$R_{k,j} = R_{k,j-1} + \frac{R_{k,j-1} - R_{k-1,j-1}}{4^{k-1}}$$

• <u>integrala numerica - Cuadraturi Gaussiene</u>

Ne dorim să determinăm valoarea unei integrale definite de forma $\int_a^b f(x)w(x)\,dx$.

Vom aproxima integrala astfel:

$$\boxed{\int_a^b f(x)\,w(x)\,dx \simeq \sum_{k=0}^{m} a_k f(x_k)}$$

, unde x_k sunt rădăcinile unui polinom ortogonal, a_k se calculează din condiția ca formula de integrare să fie exactă dacă $f(x) = 1, x, \dots, x^m$

<u>Produsul scalar a două polinoame</u>

$$\boxed{\langle p_i, p_j \rangle = \int_a^b p_i(x)\,p_j(x)\,w(x)\,dx}$$

Două polinoame sunt ortogonale dacă au produsul scalar zero.

<u>Polinoame ortogonale</u>

Nume	a	b	w(x)	Relații de recurență
Cebîșev de ord I	-1	1	$\frac{1}{\sqrt{1-x^2}}$	$T_{m+1} - 2xT_m + T_{m-1} = 0$, $T_0 = 1$, $T_1 = x$
Cebîșev de ord II	-1	1	$\sqrt{1-x^2}$	$T_{m+1} - 2xT_m + T_{m-1} = 0$, $T_0 = 1$, $T_1 = 2x$
Legendre	-1	1	1	$(m+1)L_{m+1} - (2m+1)xL_m + mL_{m-1} = 0$, $L_0 = 0$, $L_1 = x$
Laguerre	0	∞	e^{-x}	$G_{m-1} - (2m+1-x)G_m + m^2 G_{m-1} = 0$, $G_0 = 1$, $G_1 = 1-x$
Hermite	$-\infty$	∞	e^{-x^2}	$H_{m+1} - 2xH_m + 2mH_{m-1} = 0$, $H_0 = 1$, $H_2 = 2x$

Scanat cu CamScanner

Metode numerice cu pași legați

Fie intervalul $I=[a,b]$ și funcțiile $y: I \to \mathbb{R}$

$f: I \times \mathbb{R} \to \mathbb{R}$

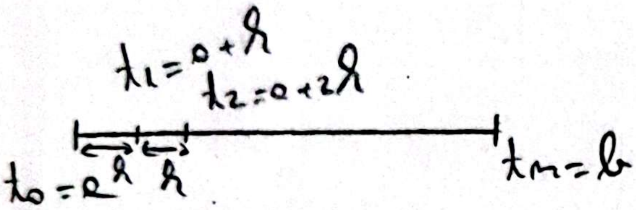
Problema diferențială de ordinul I

$$y'(t)=f(t,y(t)),\quad y(t)=?$$

Problema Cauchy de ordinul I

$$y'(t)=f(t,y(t))\quad y(t_0)=y_0,\ t_0=a,\ y(t)=?$$

Fie $t_i=a+ih$, $i=\overline{0:m}$, $h=\dfrac{b-a}{m}$ - partiționarea intervalului $[a\ b]$



$$y(t_i)=?\qquad i=\overline{0:m}$$

Ne dorim să determinăm soluția ecuației diferențiale în punctele t_i, $i=\overline{0:m}$.

- **Metoda Euler (metoda Runge-Kutta de ordinul I)**

$$y(t)\simeq y(t_i)+\frac{y'(t_i)}{1!}(t-t_i)+\frac{y''(t_i)}{2!}(t-t_i)^2+\dots$$

Fie $y(t_{i+1})\simeq y(t_i)+y'(t_i)(t_{i+1}-t_i)$ - aproximație folosind doi termeni ai polinomului Taylor

$$y'(t_i)=f(t_i,y(t_i)) \Rightarrow$$

$$\left.\begin{array}{l} y(t_{i+1})\simeq y(t_i)+(t_{i+1}-t_i)f(t_i,y(t_i)) \\ t_{i+1}-t_i=h \end{array}\right\} \Rightarrow$$

Scanat cu CamScanner

Notăm $K = h \cdot f(t_i, y(t_i)) \Rightarrow$

$$\boxed{\begin{array}{l} y(t_{i+1}) \approx y(t_i) + K, \quad i = \overline{0:n}, \quad y(t_0) = \eta_0 \\ K = h\, f(t_i, y(t_i)), \quad h = \dfrac{b-a}{n} \end{array}}$$

• <u>Metoda Runge-Kutta de ordinul II</u>

$$\boxed{\begin{array}{l} K_1 = h\, f(t_i, y_i) \\ K_2 = h\, f(t_i + h, y_i + K_1) \\ y_{i+1} = y_i + \dfrac{K_1 + K_2}{2}, \quad i = \overline{0:n}, \quad h = \dfrac{b-a}{n}, \quad y_0 = \eta_0 \end{array}}$$

S-a notat $y(t_i) = y_i \Rightarrow y(t_0) = y_0 = \eta_0$

• <u>Metoda Runge-Kutta de ordinul IV</u>

$$\boxed{\begin{array}{l} K_1 = h\, f(t_i, y_i) \\ K_2 = h\, f\left(t_i + \dfrac{h}{2}, y_i + \dfrac{K_1}{2}\right) \\ K_3 = h\, f\left(t_i + \dfrac{h}{2}, y_i + \dfrac{K_2}{2}\right) \\ K_4 = h\, f(t_i + h, y_i + K_3) \\ y_{i+1} = y_i + \dfrac{K_1 + 2K_2 + 2K_3 + K_4}{6}, \quad i = \overline{0:n}, \quad h = \dfrac{b-a}{n}, \quad y_0 = \eta_0 \end{array}}$$

• <u>Sisteme de n ecuații diferențiale de ordinul I</u>

$$\begin{cases} y_1' = f_1(t, y_1, y_2, \dots, y_n), \quad y_1(t_0) = \eta_{10} \\ y_2' = f_2(t, y_1, y_2, \dots, y_n), \quad y_2(t_0) = \eta_{20} \\ \dots \\ y_n' = f_n(t, y_1, y_2, \dots, y_n), \quad y_n(t_0) = \eta_{n0} \end{cases}$$

Scanat cu CamScanner

$y_1(t), y_2(t), \dots, y_m(t)$ necunoscute

$y_1(t_i) = ?,\ i = \overline{0:m}$

$\dots$

$y_m(t_i) = ?,\ i = \overline{0:m}$

<u>Dacă $m = 2$:</u>

$$\begin{cases} y_1' = f_1(t, y_1, y_2),\ y_1(t_0) = \eta_{10} \\ y_2' = f_2(t, y_1, y_2),\ y_2(t_0) = \eta_{20} \end{cases}$$

$y_1(t_i) = ?,\ y_2(t_i) = ?\quad i = 0:m$

$$K_{11} = h f_1(t_i, y_1^{(i)}, y_2^{(i)})$$

$$K_{21} = h f_1\left(t_i + \frac{h}{2}, y_1^{(i)} + \frac{K_{11}}{2}, y_2^{(i)} + \frac{K_{12}}{2}\right)$$

$$K_{31} = h f_1\left(t_i + \frac{h}{2}, y_1^{(i)} + \frac{K_{21}}{2}, y_2^{(i)} + \frac{K_{22}}{2}\right)$$

$$K_{41} = h f_1(t_i + h, y_1^{(i)} + K_{31}, y_2^{(i)} + K_{32})$$

$$K_{12} = h f_2(t_i, y_1^{(i)}, y_2^{(i)})$$

$$K_{22} = h f_2\left(t_i + \frac{h}{2}, y_1^{(i)} + \frac{K_{11}}{2}, y_2^{(i)} + \frac{K_{12}}{2}\right)$$

$$K_{32} = h f_2\left(t_i + \frac{h}{2}, y_1^{(i)} + \frac{K_{21}}{2}, y_2^{(i)} + \frac{K_{22}}{2}\right)$$

$$K_{42} = h f_2(t_i + h, y_1^{(i)} + K_{31}, y_2^{(i)} + K_{32})$$

$$y_1^{(i+1)} = y_1^{(i)} + \frac{K_{11} + 2K_{21} + 2K_{31} + K_{41}}{6},\quad y_2^{(i+1)} = y_2^{(i)} + \frac{K_{12} + 2K_{22} + 2K_{32} + K_{42}}{6}$$

$i = \overline{0:m},\ h = \frac{b-a}{m},\ y_1^{(0)} = \eta_{10},\ y_2^{(0)} = \eta_{20}$

• <u>Rezolvarea de ecuații diferențiale de ordinul m</u>

Scanat cu CamScanner

$y^{(m)}(t) = f(t, y(t), y'(t), \dots y^{(m-1)}(t))$

$y(t_0) = \eta_{10}, y'(t_0) = \eta_{20}, \dots, y^{(m-1)}(t_0) = \eta_{m0}$

$y(t)$ - necunoscuta

$y(t_i) = ?$, $i = \overline{0:n}$

Fie $u_1(t) = y(t)$, $u_2(t) = y'(t)$, $u_3(t) = y''(t)$, ..., $u_m(t) = y^{(m-1)}(t)$ $\Rightarrow$

$$\begin{cases} u_1'(t) = u_2(t), u_1(t_0) = y(t_0) = \eta_{10} \\ u_2'(t) = u_3(t), u_2(t_0) = y'(t_0) = \eta_{20} \\ \dots \\ u_m'(t) = y^{(m)}(t) = f(t, u_1(t), u_2(t), \dots, u_m(t)) \\ \quad u_m(t_0) = y^{(m-1)}(t_0) = \eta_{m0} \end{cases}$$

Sistem de m ec. diferențiale de ordinul I, cu necunoscutele $u_1(t), u_2(t), \dots u_m(t)$

$y(t_i) = u_1(t_i)$, $i = \overline{0:n}$

- **Metode numerice cu pași legați**

$y'(t) = f(t, y(t))$

$y(t_0) = \eta_0$

$t_i = a + ih$, $i = \overline{0:n}$, $h = \frac{b-a}{n}$, $y(t)$ – necunoscută

$y(t_i) = ?$, $i = \overline{0:n}$

Se va nota: $y_i = y(t_i)$

$f_i = f(t_i, y_i) = f(t_i, y(t_i))$

- **Metoda Adams-Bashforth**

$$\boxed{y_{i+1} = y_i + h \sum_{j=0}^{k-1} \beta_j f_{i-j}, \quad i = \overline{0:n} \quad (1)}$$

k - ordinul metodei

Calculare β_j, $j=0:r-1$:

- punctele $t_{i-r+1}, \dots, t_{i+1}$ se consideră echidistante cu $h=1$ și $t_{i-r+1}=0$
- se pune condiția ca formula (1) să fie exactă dacă $y(t)=1, t, \dots, t^r$

• Metoda Adams-Moulton

$$\boxed{y_{i+1} = y_i + h\sum_{j=-1}^{r-1} \beta_j f_{i-j}, \quad y_0 = x_0 \quad (2)}$$

r - ordinul metodei

Calculare β_j, $j=-1:r-1$:

- alegem punctele $t_{i-r+1}, \dots, t_{i+1}$ echidistante cu pasul $h=1$ și $t_{i-r+1}=0$
- se pune condiția ca formula (2) să fie exactă dacă $y(t)=1, t, \dots, t^{r+1}$

• Metode predictor - corector

Fie

$y_{i+1}^{pred} = y_i + \frac{h}{12}(23 f_i - 16 f_{i-1} + 5 f_{i-2})$, $i=\overline{2:m}$ - soluția prezisă folosind metoda Adams-Bashforth

$y_{i+1}^{cor} = y_i + \frac{h}{24}[9 f(t_{i+1}, y_{i+1}^{pred}) + 19 f_i - 5 f_{i-1} + f_{i-2}]$, $i=\overline{2:m}$

- soluția corectată folosind soluția prezisă drept input și soluția ecuației dedusă prin metoda Adams - Moulton

Scanat cu CamScanner

$y_{i+1} = y_{i+1}^{cor}, \quad i = \overline{2:n}$

$y = RK4(f, x, a, b, n)$

$y = [y(x_0) \; y(x_1) \; y(x_2) \; \dots \; y(x_n)]$

$y = [\underbrace{y_0}_{x} \; \underbrace{y_1 \; y_2}_{RK4} \; \underbrace{y_3 \dots y_n}_{\text{se calculează folosind met. pred-corector}}]$

Scanat cu CamScanner

• **Aproximare numerică**

Un spațiu prehilbertian este definit drept perechea (F, u), unde F este un spațiu vectorial și u este produsul scalar.

$u: F \times F \to \mathbb{R}$

$u(f_1, f_2) \overset{not}{=} \langle f_1, f_2 \rangle$

Dacă $F = \mathbb{R}^n$, $\langle x, y \rangle = x^T y = y^T x = \sum_{i=0}^{n} x_i y_i$, $x, y \in \mathbb{R}^n$

Dacă $F = C([a,b])$, $\langle f, g \rangle = \int_a^b f(x) g(x) w(x) dx$, $f, g \in F$

Proprietăți ale produsului scalar

1) $\langle f, f \rangle \geq 0$

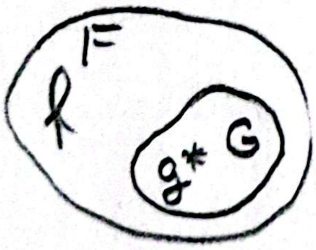
2) $\langle f_1 + f_2, f_3 \rangle = \langle f_1, f_3 \rangle + \langle f_2, f_3 \rangle$

3) $\langle f_1, f_2 \rangle = \langle f_2, f_1 \rangle$

4) $\langle c f_1, f_2 \rangle = c \langle f_1, f_2 \rangle$

$f, f_1, f_2, f_3 \in F$

Fie G un subspațiu finit al lui F



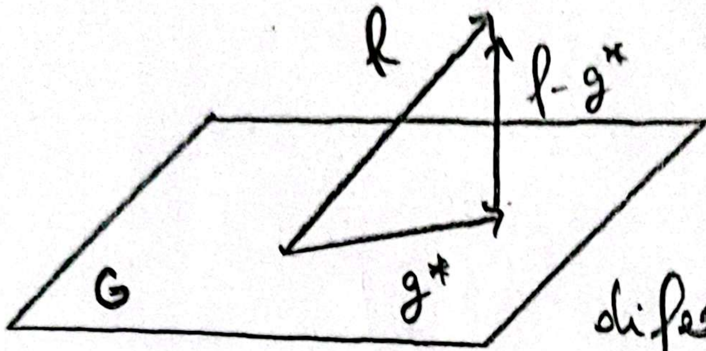
Fie elementul $f \in F$. Cel mai bun aproximant în sensul c.m.m.p pentru elementul $f \in F$ în subspațiul G este elementul $g^* \in G$ a.î.

$$\| f - g^* \|_2 = \min_{g \in G} \{ \| f - g \|_2 \}$$

Scanat cu CamScanner

Condiția necesară și suficientă ca g^* să fie cel mai bun aproximant în sensul c.m.m.p pentru f este:

$$\langle f-g^*, g\rangle = 0, \ \forall g \in G$$



Deci g^* este proiecția vectorului f pe subspațiul G, atunci vectorul diferență $f-g^*$ va fi perpendicular pe G adică va fi perpendicular (ortogonal) cu fiecare vector $g \in G \Rightarrow$

$$\langle f-g^*, g\rangle = 0, \ \forall g \in G$$

Fie o bază $\{u_0, u_1, \dots, u_m\}$ a subspațiului $G \Rightarrow$

$$g = \sum_{i=0}^{m} c_i u_i, \ \forall g \in G, \ c_i \in \mathbb{R}$$

$$g^* = \sum_{j=0}^{m} c_j^* u_j, \ c_j^* \in \mathbb{R}$$

Ne dorim să determinăm scalarii $c_j^*, j = \overline{0:m}$

$$\langle f-g^*, g\rangle = 0 \Leftrightarrow \langle f-g^*, \sum_{i=0}^{m} c_i u_i\rangle = 0 \Leftrightarrow$$

$$\langle f-g^*, c_0u_0 + c_1u_1 + \dots + c_m u_m\rangle = 0 \Leftrightarrow$$

$$\langle f-g^*, c_0u_0\rangle + \langle f-g^*, c_1u_1\rangle + \dots + \langle f-g^*, c_m u_m\rangle = 0 \Leftrightarrow$$

$$c_0\langle u_0, f-g^*\rangle + c_1\langle u_1, f-g^*\rangle + \dots + c_m\langle u_m, f-g^*\rangle = 0$$

Cum relația de mai sus este adevărată pentru orice combinații de scalari $(c_0, c_1, \dots, c_m)$ dacă ne

Scanat cu CamScanner

aleg combinațiile $(1,0,\dots,0),(0,1,\dots,0),\dots,(0,0,\dots,1)$

$$\Rightarrow \left.\begin{array}{l} \langle u_0, f-g^*\rangle = 0 \\ \langle u_1, f-g^*\rangle = 0 \\ \dots \\ \langle u_m, f-g^*\rangle = 0 \end{array}\right\} \Rightarrow \langle u_i, f-g^*\rangle = 0 \Leftrightarrow \langle u_i, f\rangle - \langle u_i, g^*\rangle = 0 \Leftrightarrow$$

$$\left.\begin{array}{l} \langle u_i, g^*\rangle = \langle u_i, f\rangle = \langle f, u_i\rangle \\ g^* = \sum_{j=0}^{m} c_j^* u_j \end{array}\right\} \Rightarrow$$

$$\langle u_i, \sum_{j=0}^{m} c_j^* u_j\rangle = \langle u_i, f\rangle \Leftrightarrow$$

$$\sum_{j=0}^{m} c_j^* \langle u_i, u_j\rangle = \langle u_i, f\rangle,\ i=\overline{0:m} \Leftrightarrow$$

$$c_0^*\langle u_i,u_0\rangle + c_1^*\langle u_i,u_1\rangle + \dots + c_m^*\langle u_i,u_m\rangle = \langle u_i, f\rangle,\quad i=\overline{0:m} \Leftrightarrow$$

$$\begin{cases} c_0^*\langle u_0,u_0\rangle + c_1^*\langle u_0,u_1\rangle + \dots + c_m^*\langle u_0,u_m\rangle = \langle u_0, f\rangle \\ c_0^*\langle u_1,u_0\rangle + c_1^*\langle u_1,u_1\rangle + \dots + c_m^*\langle u_1,u_m\rangle = \langle u_1, f\rangle \\ \dots \\ c_0^*\langle u_m,u_0\rangle + c_1^*\langle u_m,u_1\rangle + \dots + c_m^*\langle u_m,u_m\rangle = \langle u_m, f\rangle \end{cases}$$

$\Rightarrow$ Dacă rescriem sistemul de ecuații sub forma matriceală vom obține:

$$\begin{pmatrix} \langle u_0,u_0\rangle & \langle u_0,u_1\rangle & \dots & \langle u_0,u_m\rangle \\ \langle u_1,u_0\rangle & \langle u_1,u_1\rangle & \dots & \langle u_1,u_m\rangle \\ \dots & \dots & \dots & \dots \\ \langle u_m,u_0\rangle & \langle u_m,u_1\rangle & \dots & \langle u_m,u_m\rangle \end{pmatrix} \begin{pmatrix} c_0^* \\ c_1^* \\ \vdots \\ c_m^* \end{pmatrix} = \begin{pmatrix} \langle u_0, f\rangle \\ \langle u_1, f\rangle \\ \vdots \\ \langle u_m, f\rangle \end{pmatrix}$$

SEL de dimensiune $(m+1)\times(m+1)$, matricea sistemului este simetrică

Scanat cu CamScanner

Dacă $\{u_0, u_1, \dots, u_m\}$ este o bază ortogonală $\Rightarrow$

$$c_k^* = \frac{\langle u_k, f\rangle}{\langle u_k, u_k\rangle}, \quad k=\overline{0:m}$$

Dacă $\{u_0, u_1, \dots, u_m\}$ este o bază ortonormată $\Rightarrow$

$$c_k^* = \langle u_k, f\rangle, \quad k=\overline{0:m}$$

• <u>Aproximare continuă în sensul c.m.m.p</u>

Formula produsului scalar în cazul continuu este

$$\langle f, g\rangle = \int_a^b f(x)\, g(x)\, \omega(x)\, dx, \quad f, g \in F$$

<u>Bază polinomială</u>

$$\{u_0, u_1, \dots, u_m\} = \{1, x, \dots, x^m\}$$

$$g^*(x) = c_m^* x^m + c_{m-1}^* x^{m-1} + \dots + c_0^*$$

$$c_m^*, c_{m-1}^*, \dots, c_0^* = ?$$

<u>Bază trigonometrică</u>

$$\left\{\frac{1}{\sqrt{2}}, \cos x, \sin x, \cos 2x, \sin 2x, \dots, \cos mx, \sin mx\right\}$$

$$g^*(x) = \frac{1}{\sqrt{2}} a_0 + \sum_{p=1}^{m} (a_p \cos px + b_p \sin px)$$

$$a_0, a_p, b_p = ? \quad p=\overline{1:m}$$

$$\frac{1}{\pi}\int_0^{2\pi} u_q(x)\, u_r(x)\, dx = \begin{cases} 0, & q \neq r \\ 1, & q = r \end{cases}$$

$$a_0 = \left\langle \frac{1}{\sqrt{2}}, f\right\rangle = \int_0^{2\pi} \frac{1}{\sqrt{2}} f(x) \frac{1}{\pi}\, dx = \frac{1}{\pi\sqrt{2}} \int_0^{2\pi} f(x)\, dx$$

$$a_p = \langle \cos px, f\rangle = \frac{1}{\pi}\int_0^{2\pi} f(x) \cos px\, dx, \quad p=\overline{1:m}$$

$$b_p = \langle \sin px, f\rangle = \frac{1}{\pi}\int_0^{2\pi} f(x) \sin px\, dx, \quad p=\overline{1:m}$$

Scanat cu CamScanner

Bază Cebîșev

$\left\{ \frac{1}{\sqrt{2}}, T_1(x), T_2(x), \dots, T_m(x) \right\}$

Polinomul Cebîșev de ordinul m este:

$T_m(x) = \cos(m \arccos x), m \geq 0$

Notăm $\theta = \arccos x, \theta \in [0, \pi]$

$T_m(x) = \cos(m\theta)$

$T_{m+1}(x) = \cos(m\theta + \theta) = \cos m\theta \cos\theta - \sin m\theta \sin\theta$

$T_{m-1}(x) = \cos(m\theta - \theta) = \cos m\theta \cos\theta + \sin m\theta \sin\theta$

$$T_{m+1}(x) + T_{m-1}(x) = 2\underbrace{\cos m\theta}_{T_m(x)}\underbrace{\cos\theta}_{x}$$

$$\boxed{T_{m+1}(x) + T_{m-1}(x) = 2x\,T_m(x),\ T_0(x) = 1,\ T_1(x) = x}$$

$$\int_{-1}^{1} \frac{T_g(x) T_h(x)}{\sqrt{1-x^2}} dx = \begin{cases} 0, & g \neq h \\ \frac{\pi}{2}, & g = h \neq 0 \\ \pi, & g = h = 0 \end{cases}$$

$$g^*(x) = \frac{1}{\sqrt{2}} a_0 + \sum_{p=1}^{m} a_p T_p(x)$$

$a_0, a_p = ?, p = \overline{1:m}$

$$a_0 = \left\langle \frac{1}{\sqrt{2}}, f \right\rangle = \int_{-1}^{1} \frac{f(x)}{\pi\sqrt{1-x^2}} \cdot \frac{1}{\sqrt{2}} dx \Rightarrow$$

$$a_0 = \frac{1}{\pi\sqrt{2}} \int_{-1}^{1} \frac{f(x)}{\sqrt{1-x^2}} dx$$

$$a_p = \langle T_p, f \rangle = \frac{2}{\pi} \int_{-1}^{1} \frac{T_p(x) f(x)}{\sqrt{1-x^2}} dx, \ p = \overline{1:m}$$

Scanat cu CamScanner

- Aproximare discretă în sensul c.m.m.p

Formula produsului scalar în cazul discret este:

$$\langle f, g \rangle = \sum_{i=0}^{m} f(x_i)\, g(x_i)\, w(x_i), \quad f, g \in F$$

$$\| f - g^* \|_2 = \min_{g \in G} \{ \| f - g \|_2 \}$$

$$\| f - g \|_2 = \sqrt{\langle f-g, f-g \rangle} = \sqrt{\sum_{i=0}^{m} [f(x_i) - g(x_i)]^2 w(x_i)}$$

Scanat cu CamScanner