Eliminarea Gaussiană

Valentin-Ioan VINTILĂ

Facultatea de Automatică și Calculatoare - CTI
Universitatea POLITEHNICA București

14 martie 2023 (Lab. 3)

Cuprins

Transformări elementare - sisteme (1)

Ce sunt transformările elementare?

Considerăm transformările elementare acele operații pe care le putem aplica unui sistem fără a-i schimba soluția.

Transformări elementare - sisteme (2)

Să considerăm $\begin{cases} x + y + 2z &= 1 \\ 4x + 2y - 6z &= 14 \\ 2x + y &= 4 \end{cases}$. Ce putem face?

1. **Interschimbare** de ecuații: $\begin{cases} x + y + 2z &= 1 \\ 2x + y &= 4 \\ 4x + 2y - 6z &= 14 \end{cases}$
2. **Înmulțire** cu o constantă: $\begin{cases} 3x + 3y + 6z &= 3 \\ 2x + y - 3z &= 7 \\ 2x + y &= 4 \end{cases}$
3. **Adăugare (scalată)** de ecuații: $\begin{cases} 3x + 2y - z &= 8 \\ 4x + 2y - 6z &= 14 \\ 2x + y &= 4 \end{cases}$

Transformări elementare - matrice

Fie $A\mathbf{x} = \mathbf{b}$ un sistem de ecuații liniare. Sistemul poate fi văzut prin prisma unei singure matrice, $\overline{A} = \begin{bmatrix} A & \mathbf{b} \end{bmatrix}$.

Vectorul $\mathbf{x}$ nu se modifică atunci când executăm (pe $\overline{A}$):

1. **Interschimbare** de linii;
2. **Înmulțire** cu o constantă;
3. **Adăugare (scalată)** de ecuații.

Algoritmi clasici

Toți algoritmii de astăzi au la bază **algoritmul G**.

Algoritmii clasici sunt diverse variațiuni / îmbunătățiri ale algoritmului G.

Carl Friedrich Gauss

Carl Friedrich Gauss (1777-1885)

Algoritmul G - exemplu parțial (1)

Pornim de la un exemplu: $\begin{cases} a_{11}x_1 + \boxtimes x_2 + \boxtimes x_3 + \boxtimes x_4 = b_1 \\ a_{21}x_1 + \boxtimes x_2 + \boxtimes x_3 + \boxtimes x_4 = b_2 \\ a_{31}x_1 + \boxtimes x_2 + \boxtimes x_3 + \boxtimes x_4 = b_3 \\ a_{41}x_1 + \boxtimes x_2 + \boxtimes x_3 + \boxtimes x_4 = b_4 \end{cases}$

Cum ștergem x_1 de pe liniile 2, 3 și 4?

1. Scădem prima linie scalată cu $\frac{a_{21}}{a_{11}}$ din a doua;
2. Scădem prima linie scalată cu $\frac{a_{31}}{a_{11}}$ din a treia;
3. Scădem prima linie scalată cu $\frac{a_{41}}{a_{11}}$ din a patra.

Algoritmul G - exemplu parțial (2)

Rămânem cu: $\begin{cases} a_{11}x_1 & +\boxtimes x_2 + \boxtimes x_3 + \boxtimes x_4 = b_1 \\ 0 & +a_{22}x_2 + \boxtimes x_3 + \boxtimes x_4 = b_2' \\ 0 & +a_{32}x_2 + \boxtimes x_3 + \boxtimes x_4 = b_3' \\ 0 & +a_{42}x_2 + \boxtimes x_3 + \boxtimes x_4 = b_4' \end{cases}$

Cum ștergem x_2 de pe liniile 3 și 4?

1. Scădem a doua linie scalată cu $\frac{a_{32}}{a_{22}}$ din a treia;
2. Scădem a doua linie scalată cu $\frac{a_{42}}{a_{22}}$ din a patra.

Se elimină apoi x_3 de pe linia 4 prin scăderea liniei 3 (scalată) din a patra.

Algoritmul G - exemplu parțial (3)

Ce am obținut? **SST-ul:** $\begin{cases} \boxtimes x_1 + \boxtimes x_2 + \boxtimes x_3 + \boxtimes x_4 = b_1 \\ \boxtimes x_2 + \boxtimes x_3 + \boxtimes x_4 = b_2' \\ \boxtimes x_3 + \boxtimes x_4 = b_3'' \\ \boxtimes x_4 = b_4''' \end{cases}$

Algoritmul G - formal

Fie $A\mathbf{x} = \mathbf{b}$ un sistem de ecuații liniare, unde $A \in \mathbb{R}^{n\times n}$ este o matrice pătratică și $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$ sunt vectori coloană.

Pentru fiecare linie i cu excepția ultimei ($i \in \overline{1, n-1}$), vom transforma fiecare element aflat sub $\overline{A}_{ii}$ (anume $\overline{A}_{ji}$, $\forall j = \overline{i+1, n}$) în zero prin scăderea celei de-a i-a linie înmulțită cu $\frac{\overline{A}_{ji}}{\overline{A}_{ii}}$ din a j-a linie.

$$\boxed{\overline{A}(j,:) \to \overline{A}(j,:) - \frac{\overline{A}_{ji}}{\overline{A}_{ii}} \cdot \overline{A}(i,:)}, \ \forall j = \overline{i+1, n}$$

Algoritmul G - exemplu complet

Cine vine să rezolve la tablă:

$$\begin{bmatrix} 1 & -2 & 1 \\ 2 & 1 & -3 \\ 4 & -7 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 5 \\ 1 \end{bmatrix}$$

Ar trebui să obținem $\overline{A} = \begin{bmatrix} 1 & -2 & 1 & | & 0 \\ 0 & 5 & -5 & | & 5 \\ 0 & 0 & -2 & | & 0 \end{bmatrix} \Rightarrow \begin{cases} x_3 = 0 \\ x_2 = 1 \\ x_1 = 2 \end{cases}$

Algoritmul G - concluzii

Complexitate? $O(n^3)$

Îl folosim în practică? **NU!**

$$\begin{bmatrix} 0 & \boxtimes & \boxtimes \\ 1 & \boxtimes & \boxtimes \\ 0 & \boxtimes & \boxtimes \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Deoarece $A_{11} = 0$, prima coloană nu se modifică.

Algoritmul GPP - exemplu parțial

Soluția este **pivotarea parțială** (interschimbarea liniilor).

Să considerăm sistemul $A\mathbf{x} = \mathbf{b} \Rightarrow \begin{bmatrix} 0 & \boxtimes & \boxtimes \\ -8 & \boxtimes & \boxtimes \\ 4 & \boxtimes & \boxtimes \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$

Ca să rezolvăm problema $A_{11} = 0$, inversăm primele două ecuații:

$$\begin{bmatrix} -8 & \boxtimes & \boxtimes \\ 0 & \boxtimes & \boxtimes \\ 4 & \boxtimes & \boxtimes \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_2 \\ b_1 \\ b_3 \end{bmatrix}$$

Acum putem aplica algoritmul G.

Algoritmul GPP - formal (1)

Fie $A\mathbf{x} = \mathbf{b}$ un sistem de ecuații liniare, unde $A \in \mathbb{R}^{n\times n}$ este o matrice pătratică și $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$ sunt vectori coloană.

Pentru fiecare linie $i \in \overline{1, n-1}$ cu excepția ultimei, se va alege o valoare $k \in \overline{i, n}$ astfel încât $|A_{k,i}| = \max_{j=\overline{i,n}}\{|A_{j,i}|\}$, iar apoi se vor interschimba liniile i și k.

Apoi, **algoritmul G**: în noua matrice formată, se vor transforma toate elementele găsite pe a i-a coloană sub elementul de pe diagonala principală, $\overline{A}_{ii}$ (adică elementele $\overline{A}_{ji}$, $\forall j = \overline{i+1, n}$) în zerouri prin scăderea celei de-a i-a linie scalată cu $\frac{\overline{A}_{ji}}{\overline{A}_{ii}}$ din cea de-a j-a linie.

Algoritmul GPP - formal (2)

Matematic, interschimbarea celor două linii poate fi scrisă drept:

$$\exists\, k = \overline{i,n} : |A_{k,i}| = \max_{j=\overline{i,n}}\{|A_{j,i}|\} \Rightarrow \boxed{\overline{A}(i,:) \leftrightarrow \overline{A}(k,:)}$$

Iar apoi, se poate aplica algoritmul G clasic:

$$\boxed{\overline{A}(j,:) \to \overline{A}(j,:) - \frac{\overline{A}_{ji}}{\overline{A}_{ii}} \cdot \overline{A}(i,:)}, \ \forall j = \overline{i+1, n}$$

Algoritmul GPP - exemplu complet

Temă pentru acasă, același sistem dar cu GPP:

$$\begin{bmatrix}1 & -2 & 1\\ 2 & 1 & -3\\ 4 & -7 & 1\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix}=\begin{bmatrix}0\\ 5\\ 1\end{bmatrix}$$

Algoritmul GPP - concluzii

Complexitate? $O(n^3)$

Îl folosim în practică? Oarecum - avem ceva instabilitate numerică, motiv pentru care *preferăm* GPPS.

Algoritmul GPPS - ideea principală

Când ne alegem pivotul, avem grijă ca mai întâi să **scalăm opțiunile** pe care le avem și să o alegem pe **cea mai mare**.

Cum scalăm?

Generăm **doar la început** un vector $\mathbf{s} \in \mathbb{R}^n$ urmând această regulă:

$$s_i = \max_{j=\overline{i,n}} \left\{ \left| \overline{A}_{i,j} \right| \right\},\ i = \overline{1,n}$$

Algoritmul GPPS - formal (1)

Fie $A\mathbf{x} = \mathbf{b}$ un sistem de ecuații liniare, unde $A \in \mathbb{R}^{n\times n}$ este o matrice pătratică și $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$ sunt vectori coloană.

Calculăm mai întâi vectorul $\mathbf{s} \in \mathbb{R}^n$.

Pentru fiecare linie $i \in \overline{1, n-1}$ cu excepția ultimei, se va alege o valoare $k \in \overline{i,n}$ astfel încât $\left|\frac{A_{k,i}}{s_k}\right| = \max_{j=\overline{i,n}} \left\{\left|\frac{A_{j,i}}{s_j}\right|\right\}$ iar apoi se vor interschimba liniile i și k, respectiv s_i și s_k.

Apoi, **algoritmul G**: în noua matrice formată, se vor transforma toate elementele găsite pe a i-a coloană sub elementul de pe diagonala principală, $\overline{A}_{ii}$ (adică elementele $\overline{A}_{ji}$, $\forall j = \overline{i+1, n}$) în zerouri prin scăderea celei de-a i-a linie scalată cu $\frac{\overline{A}_{ji}}{\overline{A}_{ii}}$ din cea de-a j-a linie.

Algoritmul GPPS - formal (2)

Deci, generăm întâi vectorul de scalari $\mathbf{s}$ conform:

$$\boxed{s_i = \max_{j=\overline{i,n}} \left\{ \left| \overline{A}_{i,j} \right| \right\},\ i = \overline{1,n}}$$

Matematic, interschimbarea celor două linii poate fi scrisă drept:

$$\exists\, k = \overline{i,n} : \left|\frac{A_{k,i}}{s_k}\right| = \max_{j=\overline{i,n}} \left\{\left|\frac{A_{j,i}}{s_j}\right|\right\} \Rightarrow \boxed{\begin{cases}\overline{A}(i,:) \leftrightarrow \overline{A}(k,:) \quad \text{și} \\ s_i \leftrightarrow s_k\end{cases}}$$

Iar apoi, se poate aplica algoritmul G clasic:

$$\boxed{\overline{A}(j,:) \to \overline{A}(j,:) - \frac{\overline{A}_{ji}}{\overline{A}_{ii}} \cdot \overline{A}(i,:)},\ \forall j = \overline{i+1,n}$$

Algoritmul GPPS - exemplu (1)

Să aplicăm GPPS pe sistemul $\overline{A} = \begin{bmatrix}1 & -2 & 1 & | & 0\\ 2 & 1 & -3 & | & 5\\ 4 & -7 & 1 & | & 1\end{bmatrix}$.

Pasul 1. Calculăm vectorul de coeficienți:

$$S = \begin{bmatrix}\max\{|1|, |-2|, |1|\}\\ \max\{|2|, |1|, |-3|\}\\ \max\{|4|, |-7|, |1|\}\end{bmatrix}^T = \begin{bmatrix}2\\ 3\\ 7\end{bmatrix}^T$$

Algoritmul GPPS - exemplu (2)

GPPS pe sistemul $\overline{A} = \begin{bmatrix}1 & -2 & 1 & | & 0\\ 2 & 1 & -3 & | & 5\\ 4 & -7 & 1 & | & 1\end{bmatrix}$, $S = \begin{bmatrix}2 & 3 & 7\end{bmatrix}$.

Pasul 2. Deoarece $\left|\frac{\overline{A}_{21}}{s_2}\right| = \frac{2}{3} \geq \left|\frac{\overline{A}_{31}}{s_3}\right| = \frac{4}{7} \geq \left|\frac{\overline{A}_{11}}{s_1}\right| = \frac{1}{2}$, facem interschimbările de linie și de coeficient de scalare, adică $\overline{A}(1,:) \leftrightarrow \overline{A}(2,:)$ și $s_1 \leftrightarrow s_2$, deci: $\overline{A} = \begin{bmatrix}2 & 1 & -3 & | & 5\\ 1 & -2 & 1 & | & 0\\ 4 & -7 & 1 & | & 1\end{bmatrix}$ și $S = \begin{bmatrix}3 & 2 & 7\end{bmatrix}$.

Aplicăm apoi G pe prima coloană:

$$\begin{cases}\overline{A}(2,:) \to \overline{A}(2,:) - \frac{1}{2}\overline{A}(1,:)\\ \overline{A}(3,:) \to \overline{A}(3,:) - 2\overline{A}(1,:)\end{cases} \Rightarrow \overline{A} = \begin{bmatrix}2 & 1 & -3 & | & 5\\ 0 & -2.5 & 2.5 & | & -2.5\\ 0 & -9 & 7 & | & -9\end{bmatrix}$$

Algoritmul GPPS - exemplu (3)

GPPS pe sistemul $\begin{bmatrix}2 & 1 & -3 & | & 5\\ 0 & -2.5 & 2.5 & | & -2.5\\ 0 & -9 & 7 & | & -9\end{bmatrix}$, $S = \begin{bmatrix}3 & 2 & 7\end{bmatrix}$.

Pasul 3. Deoarece $\left|\frac{\overline{A}_{32}}{s_3}\right| = \frac{9}{7} \geq \left|\frac{\overline{A}_{22}}{s_2}\right| = \frac{5}{4}$, facem interschimbările de linie și de coeficient de scalare, adică $\overline{A}(2,:) \leftrightarrow \overline{A}(3,:)$ și $s_2 \leftrightarrow s_3$, deci:
$\overline{A} = \begin{bmatrix}2 & 1 & -3 & | & 5\\ 0 & -9 & 7 & | & -9\\ 0 & -2.5 & 2.5 & | & -2.5\end{bmatrix}$ și $S = \begin{bmatrix}3 & 7 & 2\end{bmatrix}$.

Aplicăm apoi G pe a doua coloană:

$$\overline{A}(3,:) \to \overline{A}(3,:) + \frac{5}{18}\overline{A}(2,:) \Rightarrow \overline{A} \approx \begin{bmatrix}2 & 1 & -3 & | & 5\\ 0 & -9 & 7 & | & -9\\ 0 & 0 & 0.5556 & | & 0\end{bmatrix}$$

Algoritmul GPPS - concluzii

Complexitate? $O(n^3)$

Îl folosim în practică? **DA!**

...și totuși mai există un algoritm clasic...?

Algoritmul GPT - ideea principală

Ne întoarcem la GPP și îl îmbunătățim altfel - în loc să interschimbăm doar linii, **vom interschimba și coloane**.

Cu alte cuvinte, în loc să căutăm pivotul de modul maxim de pe coloană, vom căuta pivotul de modul maxim din **submatrice**.

Algoritmul GPT - formal (1)

Fie $A\mathbf{x} = \mathbf{b}$ un sistem de ecuații liniare, unde $A \in \mathbb{R}^{n\times n}$ este o matrice pătratică și $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$ sunt vectori coloană.

Pentru fiecare linie $i \in \overline{1, n-1}$ cu excepția ultimei, se vor alege mai întâi două valori, $p, q \in \overline{i, n}$, astfel încât $|A_{p,q}| = \max_{j,k=\overline{i,n}}\{|A_{j,k}|\}$, iar apoi se vor interschimba liniile i și p, respectiv coloanele i și q.

Apoi, **algoritmul G**: în noua matrice formată, se vor transforma toate elementele găsite pe a i-a coloană sub elementul de pe diagonala principală, $\overline{A}_{ii}$ (adică elementele $\overline{A}_{ji}$, $\forall j = \overline{i+1, n}$) în zerouri prin scăderea celei de-a i-a linie scalată cu $\frac{\overline{A}_{ji}}{\overline{A}_{ii}}$ din cea de-a j-a linie.

Algoritmul GPT - formal (2)

Matematic, interschimbarea a două linii și a două coloane poate fi scrisă:

$$\exists\, p, q = \overline{i, n} : |A_{p,q}| = \max_{j,k=\overline{i,n}}\{|A_{j,k}|\} \Rightarrow \boxed{\begin{cases} \overline{A}(i,:) \leftrightarrow \overline{A}(p,:) & \text{și} \\ \overline{A}(:,i) \leftrightarrow \overline{A}(:,q) \end{cases}}$$

Iar apoi, se poate aplica algoritmul G clasic:

$$\boxed{\overline{A}(j,:) \rightarrow \overline{A}(j,:) - \frac{\overline{A}_{ji}}{\overline{A}_{ii}} \cdot \overline{A}(i,:)},\ \forall j = \overline{i+1, n}$$

Algoritmul GPT - exemplu complet

Cine vine să rezolve la tablă:

$$\overline{A} = \begin{bmatrix} 1 & -2 & -3 & | & 0 \\ 3 & 5 & 2 & | & 0 \\ 2 & 3 & -1 & | & -1 \end{bmatrix}$$

Ar trebui să obținem $\begin{cases} x_1 = 0.5 \\ x_2 = -0.5 \\ x_3 = 0.5 \end{cases}$

Algoritmul GPT - concluzii

Complexitate? $O(n^3)$

Îl folosim în practică? **Oarecum** - preferăm GPPS datorită performanței (chiar dacă au aceeași complexitate!).

Llewellyn Thomas

Llewellyn Thomas (1903-1992)

Algoritmul Thomas

La ce e util?

Rezolvă sisteme tridiagonale în **complexitate liniară**!

Algoritmul Thomas - intuiție (1)

Un sistem tridiagonal ia forma:

$$\begin{bmatrix} b_1 & c_1 & 0 & \dots & 0 & 0 \\ a_2 & b_2 & c_2 & \dots & 0 & 0 \\ 0 & a_3 & b_3 & & 0 & 0 \\ \vdots & \vdots & & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & b_{n-1} & c_{n-1} \\ 0 & 0 & 0 & \dots & a_n & b_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_{n-1} \\ d_n \end{bmatrix}$$

Algoritmul Thomas - intuiție (2)

Considerăm primele două ecuații: $\begin{cases} b_1x_1 + c_1x_2 & = d_1 \\ a_2x_1 + b_2x_2 + c_2x_3 & = d_2 \end{cases}$

Vrem să scăpăm de termenul x_1 din a doua ecuație. Putem așadar să aplicăm transformarea:

$$\text{Ec. } 2 \to b_1 \cdot (\text{Ec. } 2) - a_2 \cdot (\text{Ec. } 1)$$

Rămânem cu: $(b_2b_1 - c_1a_2)x_2 + c_2b_1x_3 = d_2b_1 - d_1a_2$

Algoritmul Thomas - intuiție (3)

Considerăm a doua și a treia ecuație (după transformarea inițială):

$$\begin{cases} (b_2b_1 - c_1a_2)x_2 + c_2b_1x_3 & = d_2b_1 - d_1a_2 \\ a_3x_2 + b_3x_3 + c_3x_4 & = d_3 \end{cases}$$

Vrem să scăpăm de termenul x_2 din a doua ecuație. Putem așadar să aplicăm transformarea:

$$\text{Ec. } 3 \to (b_2b_1 - c_1a_2) \cdot (\text{Ec. } 2) - a_3 \cdot (\text{Ec. } 1)$$

Rămânem cu rezultatul (greu de digerat):

$$\begin{aligned} &[b_3(b_2b_1 - c_1a_2) - c_2b_1a_3]x_3 + c_3(b_2b_1 - c_1a_2)x_4 \\ &\quad = d_3(b_2b_1 - c_1a_2) - (d_2b_1 - d_1a_2)a_3 \end{aligned}$$

Algoritmul Thomas - explicația algoritmică (1)

Ne amintim forma sistemului tridiagonal:

$$\begin{bmatrix} b_1 & c_1 & 0 & \dots & 0 & 0 \\ a_2 & b_2 & c_2 & \dots & 0 & 0 \\ 0 & a_3 & b_3 & & 0 & 0 \\ \vdots & \vdots & & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & b_{n-1} & c_{n-1} \\ 0 & 0 & 0 & \dots & a_n & b_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_{n-1} \\ d_n \end{bmatrix}$$

Fie $i \in \overline{1, n}$ pasul curent ce va transforma a i-a linie a matricei $A \in \mathbb{R}^{n \times n}$ astfel încât să conțină cel mult două intrări nenule, $n \in \mathbb{N}^*$ (cu alte cuvinte, se elimină $a_2, \dots, a_n$).

Chiar dacă pentru $i \in \{1, n\}$ nu trebuie făcut nimic, notațiile rămân valabile pentru celelalte variabile.

Algoritmul Thomas - explicația algoritmică (2)

Fie $\tilde{a}_i, \tilde{b}_i, \tilde{c}_i$ și $\tilde{d}_i$ valorile ce se vor găsi în matrice după al i-lea pas (spre exemplu, $\tilde{c}_3$ va fi calculat după a treia operație și va rămâne $\tilde{c}_3$ pentru iterațiile 4, 5 etc.).

Aceste valori pot fi calculate utilizând relațiile:

$$\begin{cases} \tilde{a}_i = 0 \\ \tilde{b}_i = \begin{cases} b_1 & , i = 1 \\ b_i\tilde{b}_{i-1} - \tilde{c}_{i-1}a_i & , i \geq 2 \end{cases} \\ \tilde{c}_i = \begin{cases} c_1 & , i = 1 \\ c_i\tilde{b}_{i-1} & , i \geq 2 \end{cases} \\ \tilde{d}_i = \begin{cases} d_1 & , i = 1 \\ d_i\tilde{b}_{i-1} - \tilde{d}_{i-1}a_i & , i \geq 2 \end{cases} \end{cases}$$

Algoritmul Thomas - explicația algoritmică (3)

Algoritmul original **normalizează** valorile $\tilde{b}_i$.

Fără explicații suplimentare, relațiile devin:

$$\boxed{\begin{cases} \tilde{a}_i = 0 \\ \tilde{b}_i = 1 \\ \tilde{c}_i = \begin{cases} \dfrac{c_1}{b_1} & , i = 1 \\ \dfrac{c_i}{b_i - \tilde{c}_{i-1}a_i} & , i \geq 2 \end{cases} \\ \tilde{d}_i = \begin{cases} \dfrac{d_1}{b_1} & , i = 1 \\ \dfrac{d_i - \tilde{d}_{i-1}a_i}{b_i - \tilde{c}_{i-1}a_i} & , i \geq 2 \end{cases} \end{cases}}$$

Algoritmul Thomas - explicația algoritmică (4)

Prin aplicarea algoritmului, se va obține următorul sistem:

$$\overline{A} = \begin{bmatrix} 1 & \tilde{c}_1 & 0 & \dots & 0 & 0 & | & \tilde{d}_1 \\ 0 & 1 & \tilde{c}_2 & \dots & 0 & 0 & | & \tilde{d}_2 \\ 0 & 0 & 1 & \dots & 0 & 0 & | & \tilde{d}_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \dots & 1 & \tilde{c}_{n-1} & | & \tilde{d}_{n-1} \\ 0 & 0 & 0 & \dots & 0 & 1 & | & \tilde{d}_n \end{bmatrix}$$

Pe scurt, se descrie sistemul:

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

Algoritmul Thomas - explicația algoritmică (5)

Rezolvarea sistemului se dovedește a fi trivială - necunoscutele primesc valori în funcție de aceste formule (când i descrește):

$$\boxed{\begin{cases} x_n = \tilde{d}_n \\ x_i = \tilde{d}_i - \tilde{c}_i \cdot x_{i+1}, \quad i = n-1, n-2, \dots, 1 \end{cases}}$$

Algoritmul Thomas - exemplu

Este evident că acest algoritm este greu de procesat. Cu toate acestea, este foarte ușor de aplicat *(cu formulele în față)*.

Să considerăm sistemul $\overline{A} = \left[\begin{array}{ccc|c} 3 & 1 & 0 & 5 \\ -1 & 3 & -2 & -7 \\ 0 & 4 & 3 & -1 \end{array}\right]$

Algoritmul Thomas - exemplu (2)

Din formule, obținem:

$$\begin{cases} \tilde{a}_2 = \tilde{a}_3 = 0 \\ \tilde{b}_1 = \tilde{b}_2 = \tilde{b}_3 = 1 \\ \tilde{c}_1 = \dfrac{1}{3} \\ \tilde{c}_2 = \dfrac{-2}{3 + \frac{1}{3}} = -\dfrac{3}{5} \\ \tilde{d}_1 = \dfrac{5}{3} \\ \tilde{d}_2 = \dfrac{-7 + \frac{5}{3}}{3 + \frac{1}{3}} = -\dfrac{8}{5} \\ \tilde{d}_3 = \dfrac{-1 + \frac{32}{5}}{3 + \frac{12}{5}} = 1 \end{cases} \Rightarrow \begin{bmatrix} 1 & 0.(3) & 0 \\ 0 & 1 & -0.6 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1.(6) \\ -1.6 \\ 1 \end{bmatrix}$$

Algoritmul Thomas - exemplu (3)

Am ajuns la: $\begin{bmatrix} 1 & 0.(3) & 0 \\ 0 & 1 & -0.6 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1.(6) \\ -1.6 \\ 1 \end{bmatrix}$

Calculăm, într-un final, $\mathbf{x}$: $\begin{cases} x_3 = 1 \\ x_2 = -1.6 + 0.6 = -1 \\ x_1 = 1.(6) + 0.(3) = 2 \end{cases}$

Algoritmul Thomas - concluzii

Complexitate? $O(n)$

Îl folosim în practică? **Adesea da** - este extrem de rapid și acceptăm că, pentru matrice special alese, poate deveni instabil numeric.

Wilhelm Jordan

Wilhelm Jordan (1842-1899)

Algoritmul Gauss-Jordan - introducere

Spre deosebire de algoritmii clasici de eliminare Gaussiană, algoritmul de eliminare Gauss-Jordan își propune transformarea matricei $A \in \mathbb{R}^{n \times n}$, $n \in \mathbb{N}^*$, într-o matrice identitate, nu doar superior triunghiulară.

Motivul pentru care ne interesează acest algoritm este pentru că poate fi modificat astfel încât **să inverseze matrice pătratice**!

Algoritmul Gauss-Jordan nemodificat - exemplu parțial (1)

Pornim de la un sistem general:

$$Ax = b \Leftrightarrow \begin{bmatrix} a_{11} & \boxtimes & \boxtimes & \boxtimes \\ \boxtimes & \boxtimes & \boxtimes & \boxtimes \\ \boxtimes & \boxtimes & \boxtimes & \boxtimes \\ \boxtimes & \boxtimes & \boxtimes & \boxtimes \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$$

De data aceasta, vrem să rămânem cu matricea identitate. Împărțim așadar prima ecuație la a_{11}:

$$\begin{bmatrix} 1 & \boxtimes & \boxtimes & \boxtimes \\ \boxtimes & \boxtimes & \boxtimes & \boxtimes \\ \boxtimes & \boxtimes & \boxtimes & \boxtimes \\ \boxtimes & \boxtimes & \boxtimes & \boxtimes \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1' \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$$

Algoritmul Gauss-Jordan nemodificat - exemplu parțial (2)

Aplicăm algoritmul G și obținem pe prima coloană:

$$\begin{bmatrix} 1 & \boxtimes & \boxtimes & \boxtimes \\ 0 & a_{22} & \boxtimes & \boxtimes \\ 0 & \boxtimes & \boxtimes & \boxtimes \\ 0 & \boxtimes & \boxtimes & \boxtimes \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1' \\ b_2' \\ b_3' \\ b_4' \end{bmatrix}$$

Continuăm același raționament, împărțind a doua ecuație la a_{22}:

$$\begin{bmatrix} 1 & \boxtimes & \boxtimes & \boxtimes \\ 0 & 1 & \boxtimes & \boxtimes \\ 0 & \boxtimes & \boxtimes & \boxtimes \\ 0 & \boxtimes & \boxtimes & \boxtimes \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1' \\ b_2'' \\ b_3' \\ b_4' \end{bmatrix}$$

Algoritmul Gauss-Jordan nemodificat - exemplu parțial (3)

De data aceasta însă, nu vom aplica algoritmul G doar în jos, ci vom aplica ideea și în sus. Cu alte cuvinte:

$$\begin{bmatrix} 1 & 0 & \boxtimes & \boxtimes \\ 0 & 1 & \boxtimes & \boxtimes \\ 0 & 0 & \boxtimes & \boxtimes \\ 0 & 0 & \boxtimes & \boxtimes \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1'' \\ b_2'' \\ b_3'' \\ b_4'' \end{bmatrix}$$

La fel pentru ultimele două ecuații. Așadar:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1^{(4)} \\ b_2^{(4)} \\ b_3^{(4)} \\ b_4^{(4)} \end{bmatrix} \Rightarrow \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1^{(4)} \\ b_2^{(4)} \\ b_3^{(4)} \\ b_4^{(4)} \end{bmatrix}$$

Algoritmul Gauss-Jordan nemodificat - sinteză

Ca să evităm problemele introduse de algoritmul G, folosim pivotarea de linii (ca la GPP sau doar alegând o valoare oarecare nenulă).

Pe scurt, la fiecare pas $i \in \overline{1, n}$ transformăm elementul de pe diagonala principală în 1, iar apoi aplicăm *GPP* atât sub, cât și deasupra elementului respectiv.

Ca temă, puteți formaliza acest algoritm.

Algoritmul Gauss-Jordan modificat - idee (1)

Privind matricea A drept o matrice de coeficienți a unui sistem de forma $A\mathbf{x} = \mathbf{b}$, unde $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$, cunoaștem deja că rezultatul sistemului nu se modifică în urma executării unor transformări elementare asupra lui A.

Așadar există transformările $E_1, \ldots, E_\mu$, $\mu \in \mathbb{N}$, alese astfel încât sistemul să se reducă la unul reprezentat de matricea identitate:

$$E_\mu \cdot E_{\mu-1} \ldots E_2 \cdot E_1 \cdot A = I_n$$

Știm însă că, prin aplicarea eliminării Gauss-Jordan, se reduce matricea A la matricea identitate, așadar cunoaștem algoritmul prin care să aflăm $E_1, \ldots, E_\mu$. Înmulțind cu A^{-1} în ambele părți, găsim următoarea formulă pentru inversa lui A:

$$E_\mu \cdot E_{\mu-1} \ldots E_2 \cdot E_1 \cdot I_n = A^{-1}$$

Algoritmul Gauss-Jordan modificat - idee (2)

Fie $A \in \mathbb{R}^{n \times n}$, $n \in \mathbb{N}^*$, o matrice pătratică nesingulară pe care vrem să o inversăm, adică vrem să calculăm A^{-1}. Dacă notăm $\overline{A} = \begin{bmatrix} A & I_n \end{bmatrix}$, putem refolosi deci întregul algoritm descris anterior pentru a compune în partea dreaptă a lui $\overline{A}$ matricea A^{-1}.

Algoritmul Gauss-Jordan modificat - exemplu (1)

Să calculăm inversa matricei $A = \begin{bmatrix} 2 & 5 & 10 \\ -1 & -2 & -3 \\ 5 & 6 & 0 \end{bmatrix}$

Înainte de a începe propriu-zis, definim $\overline{A} = \left[\begin{array}{ccc|ccc} 2 & 5 & 10 & 1 & 0 & 0 \\ -1 & -2 & -3 & 0 & 1 & 0 \\ 5 & 6 & 0 & 0 & 0 & 1 \end{array}\right]$

Algoritmul Gauss-Jordan modificat - exemplu (2)

Știm $\overline{A} = \left[\begin{array}{ccc|ccc} 2 & 5 & 10 & 1 & 0 & 0 \\ -1 & -2 & -3 & 0 & 1 & 0 \\ 5 & 6 & 0 & 0 & 0 & 1 \end{array}\right]$

Pasul 1. $\overline{A_{11}} \neq 0$, deci nu e nevoie de pivotare. Se împarte prima linie la $\overline{A_{11}} = 2$:

$$\overline{A}(1,:) \to \frac{1}{2}\overline{A}(1,:) \Rightarrow \overline{A} = \left[\begin{array}{ccc|ccc} 1 & 2.5 & 5 & 0.5 & 0 & 0 \\ -1 & -2 & -3 & 0 & 1 & 0 \\ 5 & 6 & 0 & 0 & 0 & 1 \end{array}\right]$$

Se golește prima coloană, adică se execută:

$$\begin{cases} \overline{A}(2,:) \to \overline{A}(2,:) + \overline{A}(1,:) \\ \overline{A}(3,:) \to \overline{A}(3,:) - 5\overline{A}(1,:) \end{cases} \Rightarrow \overline{A} = \left[\begin{array}{ccc|ccc} 1 & 2.5 & 5 & 0.5 & 0 & 0 \\ 0 & 0.5 & 2 & 0.5 & 1 & 0 \\ 0 & -6.5 & -25 & -2.5 & 0 & 1 \end{array}\right]$$

Algoritmul Gauss-Jordan modificat - exemplu (3)

Știm $\overline{A} = \left[\begin{array}{ccc|ccc} 1 & 2.5 & 5 & 0.5 & 0 & 0 \\ 0 & 0.5 & 2 & 0.5 & 1 & 0 \\ 0 & -6.5 & -25 & -2.5 & 0 & 1 \end{array}\right]$

Pasul 2. $\overline{A_{22}} \neq 0$, deci nu e nevoie de pivotare. Se împarte a doua linie la $\overline{A_{22}} = 0.5$:

$$\overline{A}(2,:) \to 2\overline{A}(2,:) \Rightarrow \overline{A} = \left[\begin{array}{ccc|ccc} 1 & 2.5 & 5 & 0.5 & 0 & 0 \\ 0 & 1 & 4 & 1 & 2 & 0 \\ 0 & -6.5 & -25 & -2.5 & 0 & 1 \end{array}\right]$$

Se golește a doua coloană, adică se execută:

$$\begin{cases} \overline{A}(1,:) \to \overline{A}(1,:) - 2.5\overline{A}(2,:) \\ \overline{A}(3,:) \to \overline{A}(3,:) + 6.5\overline{A}(2,:) \end{cases} \Rightarrow \overline{A} = \left[\begin{array}{ccc|ccc} 1 & 0 & -5 & -2 & -5 & 0 \\ 0 & 1 & 4 & 1 & 2 & 0 \\ 0 & 0 & 1 & 4 & 13 & 1 \end{array}\right]$$

Algoritmul Gauss-Jordan modificat - exemplu (4)

Știm $\overline{A} = \left[\begin{array}{ccc|ccc} 1 & 0 & -5 & -2 & -5 & 0 \\ 0 & 1 & 4 & 1 & 2 & 0 \\ 0 & 0 & 1 & 4 & 13 & 1 \end{array}\right]$

Pasul 3. $\overline{A_{33}} \neq 0$, deci nu e nevoie de pivotare. Se împarte a treia linie la $\overline{A_{33}} = 1$, adică nu modificăm nimic. Se golește a doua coloană:

$$\begin{cases} \overline{A}(1,:) \to \overline{A}(1,:) + 5\overline{A}(3,:) \\ \overline{A}(2,:) \to \overline{A}(2,:) - 4\overline{A}(3,:) \end{cases} \Rightarrow \overline{A} = \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 18 & 60 & 5 \\ 0 & 1 & 0 & -15 & -50 & -4 \\ 0 & 0 & 1 & 4 & 13 & 1 \end{array}\right]$$

Așadar, dacă $A = \begin{bmatrix} 2 & 5 & 10 \\ -1 & -2 & -3 \\ 5 & 6 & 0 \end{bmatrix}$, atunci $A^{-1} = \begin{bmatrix} 18 & 60 & 5 \\ -15 & -50 & -4 \\ 4 & 13 & 1 \end{bmatrix}$

Algoritmul Gauss-Jordan - concluzii

Complexitate? $O(n^3)$ - *foaaaarte rapid!*

Îl folosim în practică? **DA!**

Pentru data viitoare

Data viitoare vom avea timp suficient să discutăm.

1. **Aduceți-vă laptopurile!**
2. **OBLIGATORIU**, să aveți codul scris **ȘI ÎNȚELES** pentru G, GPPS și Gauss-Jordan modificat.
3. **VĂ VERIFIC** (...și depunctez...) (SĂ NU VĂ COPIAȚI CODUL DE LA UNUL LA ALTUL!)

Pentru bonus, am lăsat 2 exerciții în prezentare!

Bibliografie

Pentru aceste prezentări, am utilizat:

1. Cărțile *Matrix Decomposition and Applications*, respectiv *Numerical Matrix Decomposition and its Modern Applications: A Rigorous First Course* ale lui **Jun Lu**.

Sfârșit

Mulțumesc frumos pentru atenție!

Vă rog frumos să **completați formularul de feedback**!