Metode directe pentru sisteme de ecuaţii liniare

Eliminare gaussiană, descompunere LU, Cholesky

Radu T. Trîmbiţaş

Universitatea "Babeş-Bolyai"

March 19, 2009

Descompunerea LU

- Transformă $A \in \mathbb{C}^{m\times m}$ într-o matrice triunghiulară superior, U scăzând multiplii de linii
- Fiecare L_i introduce zerouri sub diagonală în coloana i:

$$\underbrace{L_{m-1}\dots L_2L_1}_{L^{-1}}A = U \Longrightarrow A = LU \text{ unde } L = L_1^{-1}L_2^{-1}\dots L_{m-1}^{-1}$$

$$\underset{A}{\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix}} \xrightarrow{L_1} \underset{L_1A}{\begin{bmatrix} \times & \times & \times & \times \\ \mathbf{0} & \times & \times & \times \\ \mathbf{0} & \times & \times & \times \\ \mathbf{0} & \times & \times & \times \end{bmatrix}} \xrightarrow{L_2} \underset{L_2L_1A}{\begin{bmatrix} \times & \times & \times & \times \\ & \times & \times & \times \\ \mathbf{0} & \times & \times & \\ \mathbf{0} & \times & \times & \end{bmatrix}} \xrightarrow{L_3} \underset{L_3L_2L_1A}{\begin{bmatrix} \times & \times & \times & \times \\ & \times & \times & \times \\ & & \times & \times \\ & & \mathbf{0} & \times \end{bmatrix}}$$

- "Triunghiularizare triunghiulară"

Matricele L_k

- La pasul k se elimină elementele de sub A_{kk}:

$$x_k = \begin{bmatrix} x_{11} & \cdots & x_{kk} & x_{k+1,k} & \cdots & x_{mk} \end{bmatrix}^T$$
$$L_k x_k = \begin{bmatrix} x_{11} & \cdots & x_{kk} & 0 & \cdots & 0 \end{bmatrix}^T$$

- Multiplicatorii $\ell_{jk} = x_{jk}/x_{kk}$ apar in L_k:

$$L_k = \begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & -\ell_{k+1,k} & 1 & & \\ & & \vdots & & \ddots & \\ & & -\ell_{mk} & & & 1 \end{bmatrix}$$

Construcția lui L

- Matricea L conține toți multiplicatorii într-o singură matrice (cu semne $+$)

$$L = L_1^{-1} L_2^{-1} \dots L_{m-1}^{-1} = \begin{bmatrix} 1 & & & & \\ \ell_{21} & 1 & & & \\ \ell_{31} & \ell_{32} & 1 & & \\ \vdots & \vdots & \ddots & \ddots & \\ \ell_{m1} & \ell_{m2} & \cdots & \ell_{m,m-1} & 1 \end{bmatrix}$$

- Definim $\ell_k = [0, \dots, 0, \ell_{k+1,k}, \dots, \ell_{m,k}]^T$. Atunci $L_k = I - \ell_k e_k^*$.
  - Avem $L_k^{-1} = I + \ell_k e_k^*$, deoarece $e_k^* \ell_k = 0$ și $\left(I - \ell_k e_k^*\right)\left(I + \ell_k e_k^*\right) = I - \ell_k e_k^* \ell_k e_k^* = I$
  - De asemenea, $L_k^{-1} L_{k+1}^{-1} = I + \ell_k e_k^* + \ell_{k+1} e_{k+1}^*$, deoarece $e_k^* \ell_{k+1} = 0$ și $\left(I + \ell_k e_k^*\right)\left(I + \ell_{k+1} e_{k+1}^*\right) = I + \ell_k e_k^* + \ell_{k+1} e_{k+1}^*$

Eliminare gaussiană fără pivotare

- Se factorizează $A \in \mathbb{C}^{m\times m}$ în $A = LU$

Eliminare gaussiană fără pivot

$U := A$; $L = I$;
for $k := 1$ **to** $m-1$ **do**
  for $j := k+1$ **to** m **do**
    $\ell_{jk} := u_{jk} / u_{kk}$;
    $u_{j,k:m} := u_{j,k:m} - \ell_{jk} u_{k,k:m}$;

- Ciclul interior poate fi scris utilizând operații matriciale în loc de cicluri for
- Număr de operații (complexitatea)
  $\sim \sum_{k=1}^{m} 2(m-k)(m-k) \sim 2\sum_{k=1}^{m} k^2 \sim \frac{2}{3}m^3$

Eliminare gaussiană cu produs exterior

- Ciclul interior poate fi scris cu operații matriciale în loc de for

Eliminare gaussiană cu produs exterior

for $k := 1$ **to** $m - 1$ **do**
$rows := k + 1 : m;$
$A_{rows,k} := A_{rows,k} / A_{k,k};$
$A_{rows,rows} := A_{rows,rows} - A_{rows,k} A_{k,rows};$

De ce este necesară pivotarea

- Să considerăm sistemul

$$\begin{bmatrix} 10 & -7 & 0 \\ -3 & 2.099 & 6 \\ 5 & -1 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 7 \\ 3.901 \\ 6 \end{bmatrix}$$

  cu soluția exactă $[0, -1, 1]^T$. Presupunem că lucrăm pe o mașină ipotetică cu AVF în baza 10, cu precizia de 5 cifre.

- Primul pas al eliminării gaussiene produce

$$\begin{bmatrix} 10 & -7 & 0 \\ 0 & -0.001 & 6 \\ 0 & 2.5 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 7 \\ 6.001 \\ 2.5 \end{bmatrix}$$

- Elementul (2,2) este mic in raport cu celelalte elemente ale matricei. Totuși, să realizăm eliminarea fără interschimbări.

Pivotare

- Pasul următor $2.5 \cdot 10^3 (E_2) + (E_3) \rightarrow (E_3)$

$$\left(5 + \left(2.5 \cdot 10^3\right)(6)\right) x_3 = \left(2.5 + \left(2.5 \cdot 10^3\right)(6.001)\right)$$

- Rezultatul lui $\left(2.5 \cdot 10^3\right)(6.001)$ este $1.50025 \cdot 10^4$ care nu este reprezentat exact pe mașina noastră ipotetică.
- Se rotunjește la $1.5002 \cdot 10^4$
- Rezultatul este adăugat la 2.5 și rotunjit din nou. Ambele cifre 5 marcate

$$\left(5 + 1.5000 \cdot 10^4\right) x_3 = \left(2.\mathbf{5} + 1.5002\mathbf{5} \cdot 10^4\right)$$

se pierd datorită erorilor de rotunjire.

Pivotare

- Pe mașina noastră ipotetică, ultima ecuație devine

$$1.5005 \cdot 10^4 x_3 = 1.5004 \cdot 10^4$$

Pivotare

- Pe mașina noastră ipotetică, ultima ecuație devine

$$1.5005 \cdot 10^4 x_3 = 1.5004 \cdot 10^4$$

- Substituția inversă începe cu

$$x_3 = \frac{1.5004 \cdot 10^4}{1.5005 \cdot 10^4} = 0.999\,93,$$

comparat cu soluția exacta $x_3 = 1$.

Pivotare

- Pe mașina noastră ipotetică, ultima ecuație devine

$$1.5005 \cdot 10^4 x_3 = 1.5004 \cdot 10^4$$

- Substituția inversă începe cu

$$x_3 = \frac{1.5004 \cdot 10^4}{1.5005 \cdot 10^4} = 0.999\,93,$$

comparat cu soluția exacta $x_3 = 1$.

- Ecuația a doua devine

$$-0.001 x_2 + 6 \cdot 0.99993 = 6.001$$

cu soluția

$$x_2 = \frac{1.5 \cdot 10^{-3}}{-1.0 \cdot 10^{-3}} = -1.5$$

Pivotare

- În fine, x_1 se obține din ecuația

$$10x_1 + (-7)(-1.5) = 7,$$

care dă

$$x_1 = -0.35.$$

Pivotare

- În fine, x_1 se obține din ecuația

$$10x_1 + (-7)(-1.5) = 7,$$

care dă

$$x_1 = -0.35.$$

- În loc de $[0, -1, 1]^T$, am obținut $[-0.35, -1.5, 0.99993]^T$. Inacceptabil!

Pivotare

- În fine, x_1 se obține din ecuația

$$10x_1 + (-7)(-1.5) = 7,$$

care dă

$$x_1 = -0.35.$$

- În loc de $[0, -1, 1]^T$, am obținut $[-0.35, -1.5, 0.99993]^T$. Inacceptabil!
- Nu am avut o acumulare a erorilor de rotunjire, iar matricea este bine condiționată.

Pivotare

- *Necazurile provin de la alegerea unui pivot mic la al doilea pas al eliminării.*

Pivotare

- *Necazurile provin de la alegerea unui pivot mic la al doilea pas al eliminării.*
- Pivotul conduce la un multiplicator de $2.5 \cdot 10^3$, iar la ultima ecuație se ajunge la coeficienți de 10^3 ori mai mari decât în problema originală.

Pivotare

- *Necazurile provin de la alegerea unui pivot mic la al doilea pas al eliminării.*
- Pivotul conduce la un multiplicator de $2.5 \cdot 10^3$, iar la ultima ecuație se ajunge la coeficienți de 10^3 ori mai mari decât în problema originală.
- Erorile de rotunjire mici în raport cu acești coeficienți sunt inacceptabile comparativ cu matricea originală și soluția exactă.

Pivotare

- *Necazurile provin de la alegerea unui pivot mic la al doilea pas al eliminării.*
- Pivotul conduce la un multiplicator de $2.5 \cdot 10^3$, iar la ultima ecuație se ajunge la coeficienți de 10^3 ori mai mari decât în problema originală.
- Erorile de rotunjire mici în raport cu acești coeficienți sunt inacceptabile comparativ cu matricea originală și soluția exactă.
- Eliminarea gaussiană fără pivotare este instabilă!

Pivotare

- *Necazurile provin de la alegerea unui pivot mic la al doilea pas al eliminării.*
- Pivotul conduce la un multiplicator de $2.5 \cdot 10^3$, iar la ultima ecuație se ajunge la coeficienți de 10^3 ori mai mari decât în problema originală.
- Erorile de rotunjire mici în raport cu acești coeficienți sunt inacceptabile comparativ cu matricea originală și soluția exactă.
- Eliminarea gaussiană fără pivotare este instabilă!
- Dacă efectuăm $(E_2) \longleftrightarrow (E_3)$, nu mai apar multiplicatori mari și rezultatul final este precis.

Pivotare

- *Necazurile provin de la alegerea unui pivot mic la al doilea pas al eliminării.*
- Pivotul conduce la un multiplicator de $2.5 \cdot 10^3$, iar la ultima ecuație se ajunge la coeficienți de 10^3 ori mai mari decât în problema originală.
- Erorile de rotunjire mici în raport cu acești coeficienți sunt inacceptabile comparativ cu matricea originală și soluția exactă.
- Eliminarea gaussiană fără pivotare este instabilă!
- Dacă efectuăm $(E_2) \longleftrightarrow (E_3)$, nu mai apar multiplicatori mari și rezultatul final este precis.
- Dacă multiplicatorii sunt < 1 în modul avem garanția că soluția este precisă.

Pivotare

- *Necazurile provin de la alegerea unui pivot mic la al doilea pas al eliminării.*
- Pivotul conduce la un multiplicator de $2.5 \cdot 10^3$, iar la ultima ecuație se ajunge la coeficienți de 10^3 ori mai mari decât în problema originală.
- Erorile de rotunjire mici în raport cu acești coeficienți sunt inacceptabile comparativ cu matricea originală și soluția exactă.
- Eliminarea gaussiană fără pivotare este instabilă!
- Dacă efectuăm $(E_2) \longleftrightarrow (E_3)$, nu mai apar multiplicatori mari și rezultatul final este precis.
- Dacă multiplicatorii sunt < 1 în modul avem garanția că soluția este precisă.
- Aceasta se asigură prin pivotare parțială.

Pivotare

- La pasul k, am utilizat elementul k, k al matricei ca pivot și am introdus zerouri în poziția k a liniilor rămase

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ & \mathbf{x}_{kk} & \times & \times & \times \\ & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & \times & \times & \times & \times \end{bmatrix} \longrightarrow \begin{bmatrix} \times & \times & \times & \times & \times \\ & x_{kk} & \times & \times & \times \\ & \mathbf{0} & \times & \times & \times \\ & \mathbf{0} & \times & \times & \times \\ & \mathbf{0} & \times & \times & \times \end{bmatrix}$$

- Dar, orice alt element $i \geq k$ din coloana k poate fi utilizat ca pivot:

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & \mathbf{x}_{ik} & \times & \times & \times \\ & \times & \times & \times & \times \end{bmatrix} \longrightarrow \begin{bmatrix} \times & \times & \times & \times & \times \\ & \mathbf{0} & \times & \times & \times \\ & \mathbf{0} & \times & \times & \times \\ & x_{ik} & \times & \times & \times \\ & \mathbf{0} & \times & \times & \times \end{bmatrix}$$

Pivotare

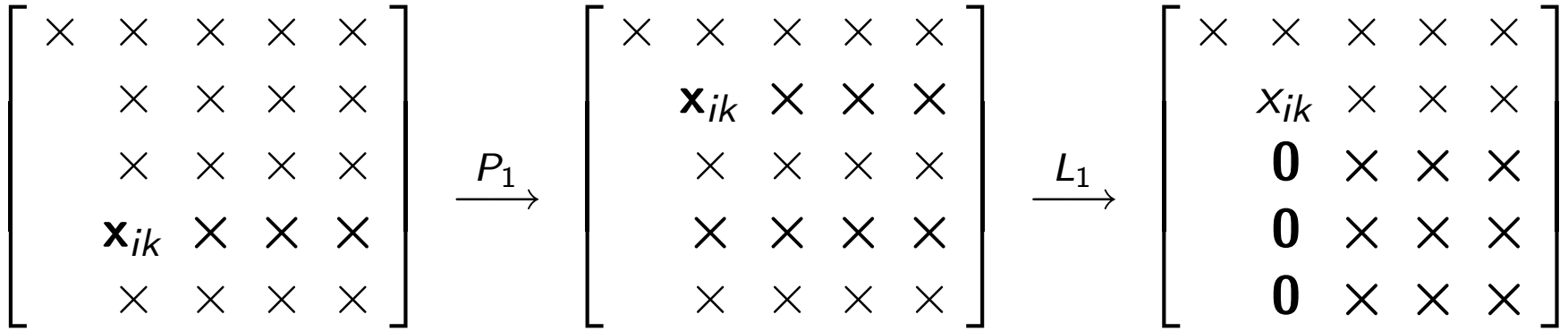
- De asemenea, se poate utiliza orice altă coloană $j \geq k$:

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & \times & \mathbf{x}_{ij} & \times & \times \\ & \times & \times & \times & \times \end{bmatrix} \longrightarrow \begin{bmatrix} \times & \times & \times & \times & \times \\ & \times & \mathbf{0} & \times & \times \\ & \times & \mathbf{0} & \times & \times \\ & \times & x_{ij} & \times & \times \\ & \times & \mathbf{0} & \times & \times \end{bmatrix}$$

- Alegând diferiți pivoți ne asigurăm că putem evita pivoții nuli sau foarte mici
- În loc să utilizăm pivoți în poziții diferite, putem interschimba linii sau coloane și să utilizăm algoritmul standard (pivotare)
- O implementare concretă poate face pivotarea indirect, fără a muta fizic datele

Pivotare parțială

- Alegerea pivoților dintre toți candidații valizi este costisitoare(pivotare completă)
- Considerăm doar pivoții din coloana k și interschimbăm liniile(pivotare parțială)



Selecție pivot | Interschimbare linii | Eliminare

- Cu operații matriceale:

$$L_{m-1}P_{m-1}\dots L_2P_2L_1P_1A = U$$

Factorizarea PA = LU

- Pentru a combina toți L_k și toți P_k în forma dorită de noi, rescriem factorizarea precedentă sub forma

$$L_{m-1}P_{m-1}\dots L_2P_2L_1P_1A = U$$
$$\left(L'_m\cdots L'_2L'_1\right)\left(P_{m-1}\cdots P_2P_1\right)A = U$$

unde

$$L'_k = P_{m-1}\cdots P_{k+1}L_kP_{k+1}^{-1}\cdots P_{m-1}^{-1}$$

- Aceasta ne dă factorizare (descompunerea) LU a lui A

$$PA = LU$$

Eliminarea gaussiană cu pivotare parțială

- Factorizează $A \in \mathbb{C}^{m \times m}$ în $PA = LU$

Eliminare gaussiană cu pivotare parțială

$U := A$; $L := I$; $P := I$;
for $k := 1$ **to** $m - 1$ **do**
  Alege $i \geq k$ care maximizează $|u_{ik}|$;
  $l_{k,1:k-1} \leftrightarrow l_{i,1:k-1}$;
  $p_{k,:} \leftrightarrow p_{i,:}$;
  for $j := k + 1$ **to** m **do**
    $\ell_{jk} := u_{jk} / u_{kk}$;
    $u_{j,k:m} := u_{j,k:m} - \ell_{jk} u_{k,k:m}$;

Exemplu

- Rezolvați sistemul

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 2 \\ 2 & 4 & 2 \end{bmatrix} x = \begin{bmatrix} 3 \\ 4 \\ 8 \end{bmatrix}$$

  prin descompunere LUP.
- **Soluție**: Avem

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & 1 & 2 \\ 3 & 2 & 4 & 2 \end{bmatrix} \sim \begin{bmatrix} 3 & 2 & 4 & 2 \\ 2 & 1 & 1 & 2 \\ 1 & 1 & 1 & 1 \end{bmatrix} \sim \begin{bmatrix} 3 & 2 & 4 & 2 \\ 2 & \frac{1}{2} & 1 & 2 \\ 1 & \frac{1}{2} & 1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 3 & 2 & 4 & 2 \\ 2 & \frac{1}{2} & -1 & 1 \\ 1 & \frac{1}{2} & -1 & 0 \end{bmatrix} \sim \begin{bmatrix} 3 & 2 & 4 & 2 \\ 2 & \frac{1}{2} & -1 & 1 \\ 1 & \frac{1}{2} & 1 & 0 \end{bmatrix} \sim \begin{bmatrix} 3 & 2 & 4 & 2 \\ 2 & \frac{1}{2} & -1 & 1 \\ 1 & \frac{1}{2} & 1 & -1 \end{bmatrix}.$$

Exemplu (continuare)

- Deci

$$L = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{2} & 1 & 0 \\ \frac{1}{2} & 1 & 1 \end{bmatrix}, \quad U = \begin{bmatrix} 2 & 4 & 2 \\ 0 & -1 & 1 \\ 0 & 0 & -1 \end{bmatrix}, \quad P = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}.$$

- Sistemele triunghiulare corespunzătoare sunt

$$\begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{2} & 1 & 0 \\ \frac{1}{2} & 1 & 1 \end{bmatrix} y = Pb = \begin{bmatrix} 8 \\ 4 \\ 3 \end{bmatrix},$$

cu soluția $y = [8, 0, -1]^T$

Exemplu (continuare)

- și

$$\begin{bmatrix} 2 & 4 & 2 \\ 0 & -1 & 1 \\ 0 & 0 & -1 \end{bmatrix} x = \begin{bmatrix} 8 \\ 0 \\ -1 \end{bmatrix},$$

  cu soluția $x = [1, 1, 1]^T$.

- Verificare

$$PA = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 2 \\ 2 & 4 & 2 \end{bmatrix} = \begin{bmatrix} 2 & 4 & 2 \\ 1 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix}$$

$$LU = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{2} & 1 & 0 \\ \frac{1}{2} & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 & 4 & 2 \\ 0 & -1 & 1 \\ 0 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 2 & 4 & 2 \\ 1 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix}.$$

Pivotare totală

- Dacă se selectează pivoți din coloane diferite, sunt necesare matrice de permutare la stânga Q_k:

$$L_{m-1}P_{m-1}\cdots L_2P_2L_1P_1AQ_1Q_2\cdots Q_{m-1} = U$$
$$(L'_{m-1}\cdots L'_2L'_1)(P_{m-1}\cdots P_2P_1)A(Q_1Q_2\cdots Q_{m-1}) = U$$

- Punem

$$L = (L'_{m-1}\cdots L'_2L'_1)^{-1}$$
$$P = P_{m-1}\cdots P_2P_1$$
$$Q = Q_1Q_2\cdots Q_{m-1}$$

pentru a obține

$$PAQ = LU$$

Liu Hui c. 220 –c. 280
Matematician chinez, a discutat eliminarea ,,gaussiană" în comentariile sale asupra lucrării ,,Cele nouă capitole ale artei matematice"
263 AD

Carl Friedrich Gauss 1777-1855
Matematică, astronomie, geodezie, magnetism
1809 GE
(Ca adolescent în Braunschweig a descoperit teorema binomială, reciprocitatea pătratică, media aritmetico-geometrică. . .)
1807-1855: Universitatea din Göttingen

Stabilitatea LU fără pivotare

- Pentru $A = LU$ calculată fără pivotare:

$$\widetilde{L}\widetilde{U} = A + \delta A, \qquad \frac{\|\delta A\|}{\|L\|\,\|U\|} = O(\text{eps})$$

- Eroare se referă la $\widetilde{L}\widetilde{U}$, nu la $\widetilde{L}$ sau $\widetilde{U}$
- Notă: la numitor apare $\|L\|\,\|U\|$, nu $\|A\|$
- $\|L\|$ și $\|U\|$ pot fi arbitrar de mari, de exemplu

$$A = \begin{bmatrix} 10^{-20} & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 10^{20} & 1 \end{bmatrix} \begin{bmatrix} 10^{-20} & 1 \\ 0 & 1 - 10^{20} \end{bmatrix}$$

- Deci, algoritmul este nestabil

Stabilitatea LU cu pivotare

- Daca se face pivotare, toate elementele lui L sunt ≤ 1 în modul, deci $\|L\| = O(1)$
- Pentru a măsura creșterea lui U, se introduce factorul de creștere

$$\rho = \frac{\max_{ij} |u_{ij}|}{\max_{ij} |a_{ij}|}$$

care implică $\|U\| = O(\rho \|A\|)$

- Pentru descompunerea $PA = LU$ calculată cu pivotare:

$$\widetilde{L}\widetilde{U} = PA + \delta A, \qquad \frac{\|\delta A\|}{\|A\|} = O(\rho \text{eps})$$

- Dacă $\rho = O(1)$, atunci algoritmul este regresiv stabil

Factorul de creștere

- Considerăm matricea

$$\begin{bmatrix} 1 & & & & 1 \\ -1 & 1 & & & 1 \\ -1 & -1 & 1 & & 1 \\ -1 & -1 & -1 & 1 & 1 \\ -1 & -1 & -1 & -1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & & & & \\ -1 & 1 & & & \\ -1 & -1 & 1 & & \\ -1 & -1 & -1 & 1 & \\ -1 & -1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 & & & & 1 \\ & 1 & & & 2 \\ & & 1 & & 4 \\ & & & 1 & 8 \\ & & & & 16 \end{bmatrix}$$

- Nu apare nici o pivotare, deci aceasta este o factorizare $PA = LU$
- Factorul de creștere $\rho = 16 = 2^{m-1}$ (se poate arăta că acesta este cazul cel mai nefavorabil)
- Deci, $\rho = 2^{m-1} = O(1)$, uniform, pentru toate matricele de dimensiune m
- Regresiv stabil conform definiției, dar rezultatul poate fi inutil
- Totuși, nu se știe exact de ce, factorii de creștere sunt mici în practică

Matrice SPD

- Reamintim:
  - $A \in \mathbb{R}^{m\times m}$ este simetrică dacă $a_{ij} = a_{ji}$, sau $A = A^T$
  - $A \in \mathbb{C}^{m\times m}$ este hermitiană dacă $a_{ij} = \bar{a}_{ji}$, sau $A = A^*$
- O matrice hermitiană A este hermitian pozitiv definită dacă $x^*Ax > 0$ pentru $x \neq 0$
  - x^*Ax este întotdeauna real deoarece $x^*Ay = \overline{y^*Ax}$
  - Simetric pozitiv definită, sau SPD, pentru matrice reale
- dacă A este $m \times m$ PD și X are rang maxim, atunci X^*AX este PD
  - Deoarece $(XAX)^* = X^*AX$, și dacă $x \neq 0$ atunci $Xx \neq 0$ și $x^*(X^*AX)x = (Xx)^*A(Xx) > 0$
  - Orice submatrice principală a lui A este PD, și orice element diagonal $a_{ii} > 0$
- matricele PD au valori proprii reale pozitive și vectori proprii ortonormali

Factorizarea Cholesky

- Se elimina sub pivot și la dreapta pivotului (datorită simetriei):

$$A = \begin{bmatrix} a_{11} & w^* \\ w & K \end{bmatrix} = \begin{bmatrix} \alpha & 0 \\ w/\alpha & I \end{bmatrix} \begin{bmatrix} \alpha & w^*/\alpha \\ 0 & K - ww^*/a_{11} \end{bmatrix}$$
$$= \begin{bmatrix} \alpha & 0 \\ w/\alpha & I \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & K - ww^*/a_{11} \end{bmatrix} \begin{bmatrix} \alpha & w^*/\alpha \\ 0 & I \end{bmatrix} = R_1^* A_1 R_1$$

  unde $\alpha = \sqrt{a_{11}}$

- $K - ww^*/a_{11}$ este o submatrice principală a matricei PD $R_1^{-*} A R_1^{-1}$, deci elementul ei din stânga sus este pozitiv

Factorizarea Cholesky

- Se aplică recursiv și se obține

$$A = (R_1^* R_2^* \dots R_m^*)(R_m \dots R_2 R_1) = R^* R, \quad r_{ii} > 0$$

- Existența și unicitatea: orice matrice HPD are o factorizare Cholesky unică
  - Algoritmul recursiv de pe folia precedentă nu eșuează niciodată
  - Rezultă și unicitatea, deoarece $\alpha = \sqrt{a_{11}}$ este determinat unic (dat) la fiecare pas și la fel, întreaga linie w/α

Algoritmul de factorizare Cholesky

- Factorizează matricea HPD $A \in \mathbb{C}^{m\times m}$ în $A = R^T R$:

Factorizare Cholesky

$R := A$;
for $k := 1$ **to** m **do**
  for $j := k+1$ **to** m **do**
    $R_{j,j:m} := R_{j,j:m} - R_{k,j:m} R_{k,j} / R_{k,k}$
  $R_{k,k:m} := R_{k,k:m} / \sqrt{R_{k,k}}$

- Complexitatea (număr de operații)

$$\sum_{k=1}^{m} \sum_{j=k+1}^{m} 2(m-j) \sim 2 \sum_{k=1}^{m} \sum_{j=1}^{k} j \sim \sum_{k=1}^{m} k^2 \sim \frac{m^3}{3}$$

Exemplu

- Să se rezolve sistemul

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 5 & 3 \\ 1 & 3 & 3 \end{bmatrix} x = \begin{bmatrix} 4 \\ 10 \\ 7 \end{bmatrix}$$

  folosind descompunerea Cholesky.
- **Soluție**: Calculând radicalii pivoților și complementele Schur se obține

$$B = \begin{bmatrix} 1 & 2 & 1 \\ & 5 & 3 \\ & & 3 \end{bmatrix} \sim \begin{bmatrix} 1 & 2 & 1 \\ & 1 & 1 \\ & & 2 \end{bmatrix} \sim \begin{bmatrix} 1 & 2 & 1 \\ & 1 & 1 \\ & & 1 \end{bmatrix}.$$

Exemplu

- Sistemele corespunzătoare sunt:

$$\begin{bmatrix} 1 & & \\ 2 & 1 & \\ 1 & 1 & 1 \end{bmatrix} y = \begin{bmatrix} 4 \\ 10 \\ 7 \end{bmatrix}$$

  cu soluția $y = \begin{bmatrix} 4 & 2 & 1 \end{bmatrix}^T$

- și

$$\begin{bmatrix} 1 & 2 & 1 \\ & 1 & 1 \\ & & 1 \end{bmatrix} x = \begin{bmatrix} 4 \\ 2 \\ 1 \end{bmatrix},$$

  cu soluția $x = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^T$.

Stabilitatea

- Factorul Cholesky calculat $\widetilde{R}$ satisface

$$\widetilde{R}^*\widetilde{R} = A + \delta A, \qquad \frac{\|\delta A\|}{\|A\|} = O(\text{eps})$$

  algoritmul este regresiv stabil
- Dar, eroarea în $\widetilde{R}$ poate fi mare ,

$$\left\|\widetilde{R} - R\right\| / \|R\| = O(\kappa(A)\text{eps})$$

- Rezolvare $Ax = b$ pentru HPD A și cu două substituții
  - Numărul de operații Cholesky $\sim m^3/3$
  - Algoritm regresiv stabil:

$$(A + \Delta A)\widetilde{x} = b, \qquad \frac{\|\Delta A\|}{\|A\|} = O(\text{eps})$$

Backslash în MATLAB

- `x=A\b` pentru `A` densă realizează următorii pași

1. Dacă `A` este triunghiulară superior sau inferior se rezolvă prin substituție inversă sau directă

Backslash în MATLAB

- `x=A\b` pentru `A` densă realizează următorii pași

1. Dacă `A` este triunghiulară superior sau inferior se rezolvă prin substituție inversă sau directă
2. Dacă `A` este o permutare a unei matrice triunghiulare, se rezolvă prin substituție (utilă pentru `[L,U]=lu(A)` căci `L` este permutată)

Backslash în MATLAB

- `x=A\b` pentru `A` densă realizează următorii pași

1. Dacă `A` este triunghiulară superior sau inferior se rezolvă prin substituție inversă sau directă
2. Dacă `A` este o permutare a unei matrice triunghiulare, se rezolvă prin substituție (utilă pentru `[L,U]=lu(A)` căci `L` este permutată)
3. Dacă `A` este simetrică sau hermitiană

Backslash în MATLAB

- `x=A\b` pentru `A` densă realizează următorii pași

1. Dacă `A` este triunghiulară superior sau inferior se rezolvă prin substituție inversă sau directă
2. Dacă `A` este o permutare a unei matrice triunghiulare, se rezolvă prin substituție (utilă pentru `[L,U]=lu(A)` căci `L` este permutată)
3. Dacă `A` este simetrică sau hermitiană
   - Se verifică dacă toate elementele digonale sunt pozitive

Backslash în MATLAB

- `x=A\b` pentru `A` densă realizează următorii pași

1. Dacă `A` este triunghiulară superior sau inferior se rezolvă prin substituție inversă sau directă
2. Dacă `A` este o permutare a unei matrice triunghiulare, se rezolvă prin substituție (utilă pentru `[L,U]=lu(A)` căci `L` este permutată)
3. Dacă `A` este simetrică sau hermitiană
   - Se verifică dacă toate elementele digonale sunt pozitive
   - Se încearca cu Cholesky; daca se termina cu succes se rezolvă prin substituție

Backslash în MATLAB

- `x=A\b` pentru `A` densă realizează următorii pași

1. Dacă `A` este triunghiulară superior sau inferior se rezolvă prin substituție inversă sau directă
2. Dacă `A` este o permutare a unei matrice triunghiulare, se rezolvă prin substituție (utilă pentru `[L,U]=lu(A)` căci `L` este permutată)
3. Dacă `A` este simetrică sau hermitiană
   - Se verifică dacă toate elementele digonale sunt pozitive
   - Se încearca cu Cholesky; daca se termina cu succes se rezolvă prin substituție
4. Dacă `A` este Hessenberg , se reduce la o matrice triunghiulară superior și apoi se rezolvă prin substituție inversă

Backslash în MATLAB

- `x=A\b` pentru `A` densă realizează următorii pași

1. Dacă `A` este triunghiulară superior sau inferior se rezolvă prin substituție inversă sau directă
2. Dacă `A` este o permutare a unei matrice triunghiulare, se rezolvă prin substituție (utilă pentru `[L,U]=lu(A)` căci `L` este permutată)
3. Dacă `A` este simetrică sau hermitiană
   - Se verifică dacă toate elementele digonale sunt pozitive
   - Se încearca cu Cholesky; daca se termina cu succes se rezolvă prin substituție
4. Dacă `A` este Hessenberg , se reduce la o matrice triunghiulară superior și apoi se rezolvă prin substituție inversă
5. Dacă `A` este pătratică, se factorizează `PA = LU` și se rezolvă prin substituție inversă

Backslash în MATLAB

- `x=A\b` pentru `A` densă realizează următorii pași

1. Dacă `A` este triunghiulară superior sau inferior se rezolvă prin substituție inversă sau directă
2. Dacă `A` este o permutare a unei matrice triunghiulare, se rezolvă prin substituție (utilă pentru `[L,U]=lu(A)` căci `L` este permutată)
3. Dacă `A` este simetrică sau hermitiană
   - Se verifică dacă toate elementele digonale sunt pozitive
   - Se încearca cu Cholesky; daca se termina cu succes se rezolvă prin substituție
4. Dacă `A` este Hessenberg , se reduce la o matrice triunghiulară superior și apoi se rezolvă prin substituție inversă
5. Dacă `A` este pătratică, se factorizează `PA = LU` și se rezolvă prin substituție inversă
6. Dacă `A` nu este pătratică, se face factorizare QR cu metoda Householder, și se rezolvă problema de aproximare în sensul celor mai mici pătrate

Descompunere QR

- Fie $A \in \mathbb{C}^{m \times n}$. Se numește descompunere QR a lui A perechea de matrice (Q, R) unde $Q \in \mathbb{C}^{m \times n}$ este unitară, $R \in \mathbb{C}^{n \times n}$ este triunghiulară superior și $A = QR$.

Triunghiularizare Householder

- Metoda lui Householder înmulțește cu matrice unitare pentru a transform matricea într-una triunghiulară; de exemplu la primul pas:

$$Q_1 A = \begin{bmatrix} r_{11} & \times & \cdots & \times \\ 0 & \times & \cdots & \times \\ 0 & \times & \cdots & \times \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \times & \cdots & \times \end{bmatrix}$$

- La sfârșit, am obținut un produs de matrice ortogonale

$$\underbrace{Q_n \ldots Q_2 Q_1}_{Q^*} A = R$$

- "Triunghiularizare ortogonală"

Introducerea de zerouri

- Q_k introduce zerouri sub diagonală în coloana k
- Păstrează zerourile introduse anterior

$$\begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix} \xrightarrow{Q_1} \begin{bmatrix} \times & \times & \times \\ \mathbf{0} & \times & \times \\ \mathbf{0} & \times & \times \\ \mathbf{0} & \times & \times \\ \mathbf{0} & \times & \times \end{bmatrix} \xrightarrow{Q_2} \begin{bmatrix} \times & \times & \times \\ & \times & \times \\ & \mathbf{0} & \times \\ & \mathbf{0} & \times \\ & \mathbf{0} & \times \end{bmatrix} \xrightarrow{Q3} \begin{bmatrix} \times & \times & \times \\ & \times & \times \\ & & \times \\ & & \mathbf{0} \\ & & \mathbf{0} \end{bmatrix}$$

$$A \qquad Q_1 A \qquad Q_2 Q_1 A \qquad Q_3 Q_2 Q_1 A$$

Reflectori Householder

- Fie Q_k de forma

$$Q_k = \begin{bmatrix} I & 0 \\ 0 & F \end{bmatrix}$$

unde I este $(k-1) \times (k-1)$ și F este $(m-k+1) \times (m-k+1)$

- Creăm reflectorul Householder F care introduce zerouri:

$$x = \begin{bmatrix} \times \\ \times \\ \vdots \\ \times \end{bmatrix} \qquad Fx = \begin{bmatrix} \|x\| \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \|x\| \, e_1$$

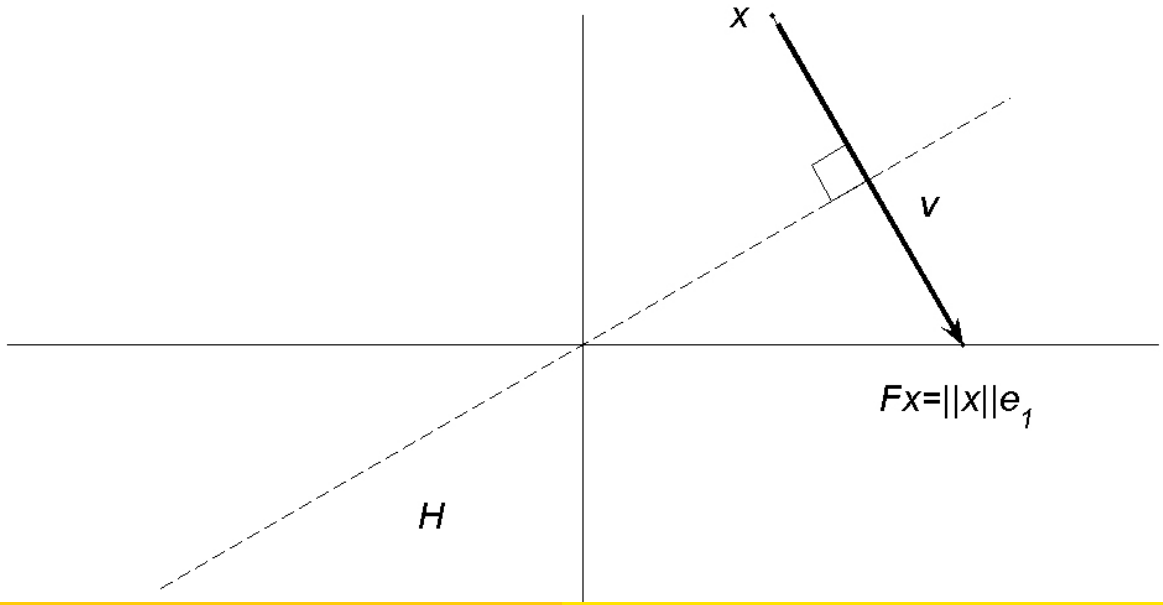
Reflectori Householder-Ideea

- Ideea: reflectăm în raport cu hiperplanul H, ortogonal pe $v = \|x\|_2 e_1 - x$, aplicând matricea unitară

$$F = I - 2\frac{vv^*}{v^*v}$$

- A se compara cu proiectorul

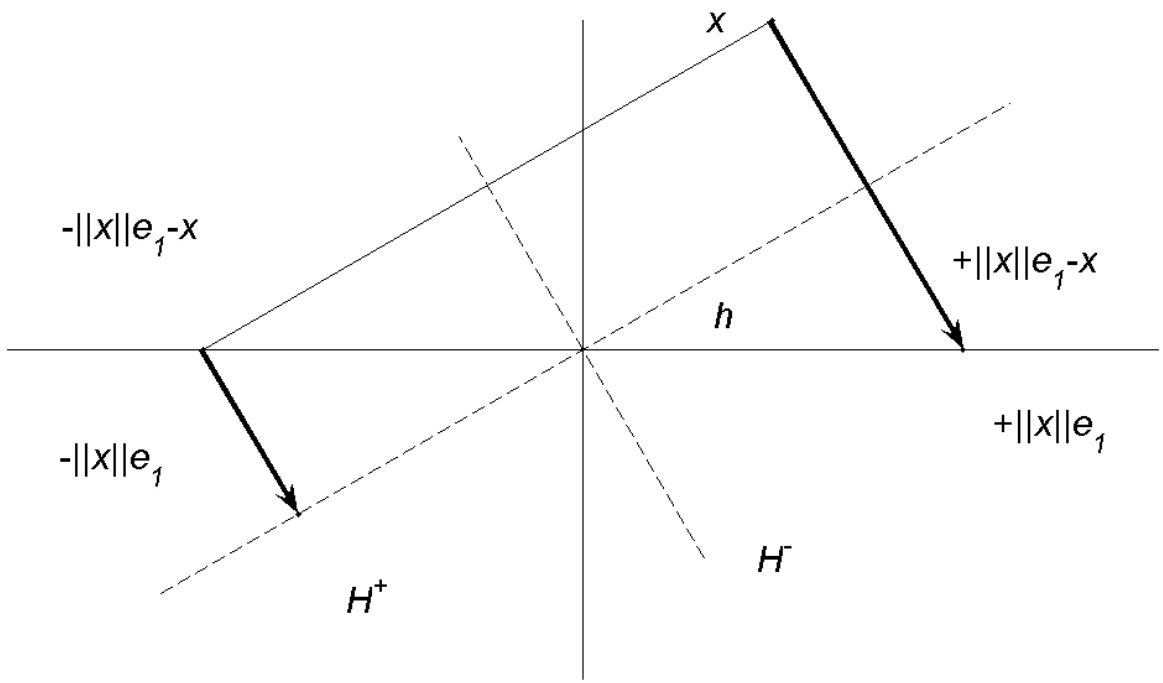
$$P_{\perp v} = I - \frac{vv^*}{v^*v}$$



Alegerea reflectorului

- Putem aplica reflexia oricărui multiplu z al lui $\|x\| e_1$ cu $|z| = 1$
- Proprietăți numerice mai bune pentru $\|v\|$ mare, de exemplu $v = \text{sign}(x_1) \|x\| e_1 + x$

- Notă: $\text{sign}(0) = 1$, dar în MATLAB, `sign(0)==0`



Algoritmul lui Householder

- Calculează factorul R al descompunerii QR a matricei $m \times n$ A $(m \geq n)$
- Lasă rezultatul în A, memorând vectorii de reflexie v_k pentru utilizare ulterioară

Factorizare QR prin metoda Householder

for $k := 1$ **to** n **do**

$x := A_{k:m,k}$;

$v_k := \text{sign}(x_1)\|x\|_2 e_1 + x$;

$v_k := v_k / \|v_k\|_2$;

$A_{k:m,k:n} = A_{k:m,k:n} - 2v_k \left(v_k^* A_{k:m,k:n}\right)$

Aplicarea sau obținerea lui Q

- Calculăm $Q^*b = Q_n \dots Q_2 Q_1 b$ și $Qx = Q_1 Q_2 \dots Q_n x$ implicit
- Pentru a crea Q explicit, aplicăm pentru $x = I$

Calculul implicit al lui Q^*b

for $k := 1$ **to** n **do**

$b_{k:m} = b_{k:m} - 2v_k \left(v_k^* b_{k:m}\right)$;

Calculul implicit al lui Qx

for $k := n$ **downto** 1 **do**

$x_{k:m} = x_{k:m} - 2v_k \left(v_k^* x_{k:m}\right)$;

Complexitatea -Householder QR

- Cea mai mare parte a efortului

$$A_{k:m,k:n} = A_{k:m,k:n} - 2v_k \left(v_k^* A_{k:m,k:n}\right)$$

- Operații pe iterație:
  - $2(m-k)(n-k)$ pentru produsele scalare $v_k^* A_{k:m,k:n}$
  - $(m-k)(n-k)$ pentru produsul exterior $2v_k(\cdots)$
  - $(m-k)(n-k)$ pentru scăderea $A_{k:m,k:n} - \cdots$
  - $4(m-k)(n-k)$ total
- Incluzând ciclul exterior, totalul devine

$$\sum_{k=1}^{n} 4(m-k)(n-k) = 4\sum_{k=1}^{n} \left(mn - k\left(m+n\right) + k^2\right)$$

$$\sim 4mn^2 - 4(m+n)n^2/2 + 4n^3/3 = 2mn^2 - 2n^3/3$$

Figure: Alston S. Householder (1904-1993), matematician american. Contribuții importante: biologie mtematică, algebră liniară numerică. Cartea sa "The theory of matrices in numerical analysis" a avut un mare impact asupra dezvoltării analizei numerice și a informaticii.

Figure: James Wallace Givens (1910-1993) Pionier al algebrei liniare numerice și informaticii