Sisteme liniare - metode iterative

Radu T. Trîmbiţaş

21 martie 2022

1 Metode iterative staţionare

Dorim să calculăm soluţia sistemului

$$Ax = b, \tag{1}$$

când A este inversabilă. Presupunem că am găsit o matrice T şi un vector c astfel încât $I - T$ este inversabilă şi punctul fix unic al ecuaţiei

$$x = Tx + c \tag{2}$$

coincide cu soluţia sistemului $Ax = b$. Fie x^* soluţia lui (1) sau, echivalent, a lui (2).

Iteraţia: $x^{(0)}$ dat; se defineşte $(x^{(k)})$ prin

$$x^{(k+1)} = Tx^{(k)} + c, \quad k \in \mathbb{N}. \tag{3}$$

Metoda (3) este convergentă dacă şi numai dacă $\rho(T) < 1$, unde $\rho(T)$ este raza spectrală a lui T.

Criteriul de oprire este

$$\|x^{(k)} - x^{(k-1)}\| \leq \frac{1 - \|T\|}{\|T\|}\varepsilon. \tag{4}$$

Presupunem că putem descompune A sub forma $A = M - N$. Dacă M este uşor de inversat(diagonală, triunghiulară, ş.a.m.d.) este mai uşor să realizăm calculele în modul următor

$$Ax = b \Leftrightarrow Mx = Nx + b \Leftrightarrow x = M^{-1}Nx + M^{-1}b$$

Ultima ecuaţie este de forma $x = Tx + c$, unde $T = M^{-1}N = I - M^{-1}A$. Se obţine şirul

$$x^{(k+1)} = M^{-1}Nx^{(k)} + M^{-1}b, \quad k \in \mathbb{N}, \tag{5}$$

unde $x^{(0)}$ este un vector arbitrar. Considerăm descompunerea $A = D - L - U$, unde

$$(D)_{ij} = a_{ij}\delta_{ij}, \quad (-L)_{ij} = \begin{cases} a_{ij}, & i > j \\ 0, & \text{altfel} \end{cases}$$

$$(-U)_{ij} = \begin{cases} a_{ij}, & i < j \\ 0, & \text{altfel} \end{cases}$$

Pentru diverse alegeri ale lui M şi N se obţine:

- metoda lui Jacobi: $M = D$, $N = L + U$. In acest caz, $T = D^{-1}(L + U)$, $c = D^{-1}b$. Scrisă pe componente, metoda are forma

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

- metoda Gauss-Seidel: $M = D - L$, $N = U$. In acest caz, $T = (D - L)^{-1}U$, $c = (D - L)^{-1}b$. Scrisă pe componente, metoda are forma

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

- metoda SOR (Successive OverRelaxation). In acest caz, $M = \frac{D}{\omega} - L$. Se obţine

$$T = (D - \omega L)^{-1}((1 - \omega)D + \omega U),$$
$$c = \omega(D - \omega L)^{-1}.$$

  Pe componente,

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

  Valoarea optimă a lui ω, valabilă doar pentru anumite tipuri de matrice (tridiagonale, tridiagonal pe blocuri, ordonate consistent, etc.) este

$$\omega_O = \frac{2}{1 + \sqrt{1 - \rho^2}}, \tag{6}$$

  unde ρ este raza spectrală a matricei metodei lui Jacobi.

2 Consideraţii de implementare

Dacă putem descompune A sub forma $A = M - N$, unde M este inversabilă, iteraţia (5) se scrie sub forma

$$x^{(k+1)} = M^{-1}\left(Nx^{(k)} + b\right), \quad k \in \mathbb{N}.$$

Folosind operatorul \ din MATLAB putem scrie iteraţia sub forma

$$\mathtt{xc} = \mathtt{M}\backslash\left(\mathtt{N} * \mathtt{xp} + \mathtt{N}\right)$$

unde xp este aproximaţia precedentă, iar xc aproximaţia curentă.

Dacă ε_a este eroarea absolută, iar ε_r eroarea relativă, criterul de oprire se poate scrie sub forma (criteriu mixt)

$$\|x^{(k)} - x^{(k-1)}\| \leq \frac{1 - \|T\|}{\|T\|}\varepsilon_a + \varepsilon_r \left\|x^{(k)}\right\|.$$

Pentru efectivitatea criteriului, cei doi termeni $\frac{1-\|T\|}{\|T\|}\varepsilon_a$ şi ε_r trebuie să aibă acelaşi ordin de mărime. La implementarea criteriului de oprire pute lua în considerare şi reziduul

$$r^{(k)} = b - Ax^{(k)}$$

şi norma sa $\left\|r^{(k)}\right\|$. Alegerea unui criteriu de oprire eneral este o problemă dificilă.

Metodele iterative dau rezultate bune dacă sunt aplicate sistemelor mari cu matrice rară sau structurată.

3 Probleme

Problema 1 *Implementaţi metoda lui Jacobi în MATLAB.*

Problema 2 *Implementaţi metoda SOR în MATLAB. Găsiţi ω optim utilizând (6). Atenţie: aceasta nu este o metodă practică pentru calculul lui ω_o; ea are numai scop didactic.*

Problema 3 *Rezolvaţi sistemele:*

$$
\begin{bmatrix}
5 & -1 & 0 & 0 & \dots & & 0 \\
-1 & 5 & -1 & 0 & \dots & & 0 \\
0 & -1 & 5 & \ddots & \ddots & & \vdots \\
\vdots & \ddots & \ddots & \ddots & -\ddots & 0 & \vdots \\
\vdots & & \ddots & -1 & 5 & -1 & 0 \\
0 & \dots & \dots & 0 & -1 & 5 & -1 \\
0 & \dots & \dots & \dots & 0 & -1 & 5
\end{bmatrix} x = \begin{bmatrix} 4 \\ 3 \\ 3 \\ \vdots \\ 3 \\ 3 \\ 4 \end{bmatrix}
$$

$$
\begin{bmatrix}
5 & -1 & 0 & -1 & \dots & & \dots & 0 \\
-1 & 5 & -1 & 0 & -1 & & & \vdots \\
0 & -1 & 5 & -1 & \ddots & \ddots & \vdots & \vdots \\
-1 & & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\
\vdots & -1 & \ddots & -1 & 5 & -1 & 0 & -1 \\
0 & \dots & \ddots & 0 & -1 & 5 & -1 & 0 \\
0 & \dots & & -1 & 0 & -1 & 5 & -1 \\
0 & \dots & & & -1 & 0 & -1 & 5
\end{bmatrix} x = \begin{bmatrix} 3 \\ 2 \\ 2 \\ 1 \\ \vdots \\ 2 \\ 2 \\ 3 \end{bmatrix}.
$$

cu toate metodele implementate.

Problema 4 *Generaţi sisteme cu matrice diagonal dominante aleatoare ce au soluţia* $[1, \dots, n]^T$ *şi rezolvaţi-le cu metodele Jacobi, Gauss-Seidel şi SOR.*

4 Probleme suplimentare

Problema 5 *Testaţi rutinele implementate pentru matrice rare de diverse dimensiuni şi comparaţi timpii de execuţie cu cei necesari pentru matrice dense.*

Problema 6 *Pentru rutinele implementate generaţi curbe de convergenţă, adică curbe semilogaritimice care au pe abscisă numărul pasului curent, iar pe ordonată logaritmul normei reziduului. (folosiţi funcţia MATLAB* `semilogy`*).*