Cap2. Sisteme de ecuaţii algebrice liniare - metode iterative

Prof.dr.ing. Gabriela Ciuprina

Universitatea "Politehnica" Bucureşti, Facultatea de Inginerie Electrică,
Departamentul de Electrotehnică

Suport didactic pentru disciplina *Metode numerice*, 2017-2018

Notes

Cuprins

Notes

Formularea problemei

Sistem de n ecuaţii algebrice liniare cu n necunoscute:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1, \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2, \\ \cdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = b_n. \end{cases} \quad (1)$$

Notes

Formularea problemei

Se dă matricea coeficienţilor

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & & & \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \in \mathbb{R}^{n \times n} \quad (2)$$

şi vectorul termenilor liberi

$$\mathbf{b} = \begin{bmatrix} b_1 & b_2 & \cdots & b_n \end{bmatrix}^T \in \mathbb{R}^n, \quad (3)$$

se cere să se rezolve sistemul

$$\mathbf{Ax} = \mathbf{b}, \quad (4)$$

unde $\mathbf{x}$ este soluţia

$$\mathbf{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^T \in \mathbb{R}^n. \quad (5)$$

Notes

Buna formulare matematică

Problema este bine formulată din punct de vedere matematic (soluţia există şi este unică)
$\Leftrightarrow$
matricea **A** este nesingulară (are determinantul nenul).
Se scrie formal:

$$"\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}"$$

trebuie citită ca:
*"**x** este soluţia sistemului algebric liniar* $\mathbf{Ax} = \mathbf{b}$*"*
şi **NU** "*se calculează inversa matricei **A** care se înmulţeşte cu vectorul **b**"*.

Notes

Condiţionarea problemei

$$\kappa(\mathbf{A}) = \|\mathbf{A}\|\|\mathbf{A}^{-1}\| \qquad (6)$$

număr de condiţionare la inversare al matricei **A**.

$$\varepsilon_x \leq \kappa(\mathbf{A})\varepsilon_b, \qquad (7)$$

- $\kappa(\mathbf{A}) \geq 1$:
  Cazul cel mai favorabil: $n_A = 1$ şi $\varepsilon_x = \varepsilon_b$. (matrice ortogonală)
- Numărul de condiţionare este o proprietate a matricei şi nu are legătură nici cu metoda de rezolvare propriu-zisă, nici cu erorile de rotunjire care apar în mediul de calcul.
  În practică:
  Dacă $\kappa(\mathbf{A}) > 1/\texttt{eps}$ problema se consideră slab condiţionată.

Notes

Clasificarea metodelor

1. **Metode directe** - găsesc soluţia teoretică a problemei într-un număr finit de paşi. (Gauss, factorizare LU)
2. **Metode iterative** - generează un şir de aproximaţii ale soluţiei care se doreşte a fi convergent către soluţia exactă.
   - **staţionare:** Jacobi, Gauss-Seidel, SOR, SSOR
   - **nestaţionare (semiiterative):** gradienţi conjugaţi (GC), reziduu minim (MINRES), reziduu minim generalizat (GMRES), gradienţi biconjugaţi (BiGC), etc.

Notes

Ideea metodelor staţionare

$$\mathbf{Ax} = \mathbf{b} \tag{8}$$

se construieşte un şir de aproximaţii
$\mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(k)}, \ldots$

$$\lim_{k\to\infty} \mathbf{x}^{(k)} = \mathbf{x}^*, \quad \text{unde} \quad \mathbf{Ax}^* = \mathbf{b}. \tag{9}$$

$$\mathbf{x}^{(k)} = \begin{bmatrix} x_1^{(k)} \\ x_2^{(k)} \\ \vdots \\ x_n^{(k)} \end{bmatrix} \in \mathbb{R}^{n\times 1}.$$

Notes

Ideea metodelor staţionare

Algoritmul are nevoie de

1. o iniţializare $\mathbf{x}^{(0)}$;
2. un mod de generare a şirului de iteraţii;
3. un criteriu de oprire.

1. Iniţializarea

- în principiu, arbitrară;
- dacă este posibil, cât mai aproape de soluţie.

Notes

Ideea metodelor staţionare

2. Şirul de iteraţii se generează *recursiv*:

$$\mathbf{x}^{(k)} = F(\mathbf{x}^{(k-1)}), \tag{10}$$

$$\mathbf{x}^* = F(\mathbf{x}^*), \tag{11}$$

$\mathbf{x}^*$ este punct fix pentru aplicaţia F.
În concluzie, soluţia exactă a sistemului de ecuaţii este şi punct fix pentru F. Rezolvarea sistemului de ecuaţii algebrice liniare se face prin căutarea unui punct fix pentru F.

Notes

Ideea metodelor staţionare

$$\mathbf{A} = \mathbf{B} - \mathbf{C}. \quad (12)$$

$$\mathbf{B}\mathbf{x} = \mathbf{C}\mathbf{x} + \mathbf{b} \quad (13)$$

$$\mathbf{x} = \mathbf{M}\mathbf{x} + \mathbf{u}, \quad (14)$$

$$\begin{aligned} \mathbf{M} &= \mathbf{B}^{-1}\mathbf{C}, \\ \mathbf{u} &= \mathbf{B}^{-1}\mathbf{b}. \end{aligned} \quad (15)$$

$\mathbf{M} \in \mathbb{R}^{n\times n}$ se numeşte *matrice de iteraţie*.

$$F(\mathbf{x}) = \mathbf{M}\mathbf{x} + \mathbf{u}, \quad (16)$$

Notes

Ideea metodelor staţionare

$$\mathbf{x}^{(k)} = F(\mathbf{x}^{(k-1)}), \quad (17)$$

$$\mathbf{x}^{(k)} = \mathbf{M}\mathbf{x}^{(k-1)} + \mathbf{u}, \quad (18)$$

$$\mathbf{x}^{(k)} = \mathbf{B}^{-1}\mathbf{C}\mathbf{x}^{(k-1)} + \mathbf{B}^{-1}\mathbf{b}. \quad (19)$$

$$\mathbf{B}\mathbf{x}^{(k)} = \mathbf{C}\mathbf{x}^{(k-1)} + \mathbf{b}, \quad (20)$$

B are o structură particulară.

Notes

Ideea metodelor staţionare

3. Criteriul de oprire
Condiţie de oprire bazată de criteriul Cauchy de convergenţă:

$$\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\| \leq \varepsilon, \tag{21}$$

Se poate întâmpla însă ca şirul iteraţiilor să nu fie convergent.
Procedurile iterative vor avea ca parametri de intrare, pe lângă mărimile ce definesc sistemul:

- o eroare ce reprezintă criteriul dorit de oprire a iteraţiilor;
- un număr maxim de iteraţii, util pentru a asigura oprirea naturală a procedurii în caz de neconvergenţă.

Nu are sens ca $\varepsilon <$ eps $\|\mathbf{x}^{(k)}\|$.

Notes

Util: vectori si valori proprii

Definiţie: vectorii proprii $\mathbf{v}$ ai unei matrice pătrate reale $\mathbf{M}$, de dimensiune n sunt acei vectori nenuli, pentru care există un scalar λ astfel încât

$$\mathbf{M}\mathbf{v} = \lambda\mathbf{v}. \tag{22}$$

Obs:

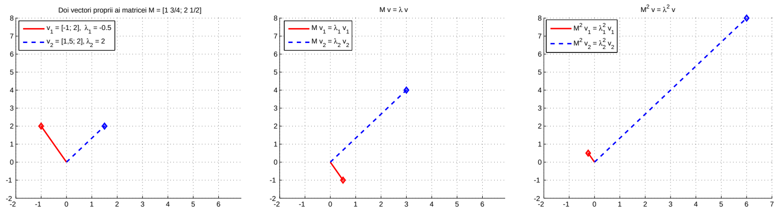
- Reprezentarea geometrică: prin aplicarea $\mathbf{M}$ asupra lui, vectorul nu se roteşte;
- Vectorii proprii ai unei matrice nu sunt unici. Dacă $\mathbf{v}$ este un vector propriu, atunci şi vectorul scalat $\alpha\mathbf{v}$ este de asemenea vector propriu;
- λ se numeşte valoare proprie a matricei $\mathbf{M}$ asociată vectorului propriu $\mathbf{v}$.

Notes

Util: vectori si valori proprii

$$\mathbf{M}^k \mathbf{v} = \lambda^k \mathbf{v}. \tag{23}$$

Dacă $|\lambda| < 1$ atunci $\lim_{k\to\infty} \|\mathbf{M}^k \mathbf{v}\| = 0$.
Dacă $|\lambda| > 1$ atunci $\lim_{k\to\infty} \|\mathbf{M}^k \mathbf{v}\| = \infty$.



Înmulţirea repetată dintre o matrice şi vectorii ei proprii.

Notes

Util: vectori si valori proprii

Dacă **M** este simetrică, atunci ea are n vectori proprii liniar independenţi:
$\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \ldots, \mathbf{v}^{(n)}$.
Această mulţime nu este unică, dar fiecare vector propriu are asociat o valoare proprie unică.
Dacă cei n vectori proprii ai lui **M** formează o bază, atunci orice vector $\mathbf{u} = \sum_{i=1}^{n} \alpha_i \mathbf{v}^{(i)}$.

$$\mathbf{M}^k \mathbf{u} = \alpha_1 \lambda_1^k \mathbf{v}^{(1)} + \cdots + \alpha_n \lambda_n^k \mathbf{v}^{(n)}. \tag{24}$$

Dacă toate valorile proprii sunt subunitare în modul, atunci norma vectorului rezultant va tinde către zero. E suficient ca o singură valoare proprie să fie în modul mai mare ca 1, ca norma vectorului rezultant să tindă către infinit.

Notes

Util: vectori si valori proprii

Raza spectrală a unei matrice: proprii

$$\rho(M) = \max_i |\lambda_i|. \tag{25}$$

Valorile proprii sunt rădăcinile polinomului caracteristic. Deoarece

$$(\lambda \mathbf{I} - \mathbf{M})\mathbf{v} = \mathbf{0} \tag{26}$$

rezultă in mod necesar anularea *polinomului caracteristic al matricei*:

$$\det(\lambda \mathbf{I} - \mathbf{M}) = 0. \tag{27}$$

Ecuaţie de gradul n în λ care, cf. teoremei fundamentale a algebrei, are exact n soluţii (reale sau în perechi complex conjugate), care sunt valorile proprii ale matricei.

Notes

Convergenţă

Teorema 1: Condiţia necesară şi suficientă

ca procesul iterativ să fie convergent este ca raza spectrală a matricei de iteraţie să fie strict subunitară:

$$\rho(M) < 1.$$

$$\mathbf{e}^{(k)} = \mathbf{x}^{(k)} - \mathbf{x}^* = F(\mathbf{x}^{(k-1)}) - F(\mathbf{x}^*) = \mathbf{M}\mathbf{x}^{(k-1)} + \mathbf{u} - \mathbf{M}\mathbf{x}^* - \mathbf{u} = \mathbf{M}\mathbf{e}^{(k-1)}, \tag{28}$$

$$\mathbf{e}^{(k)} = \mathbf{M}\mathbf{e}^{(k-1)} = \mathbf{M}^2\mathbf{e}^{(k-2)} = \cdots = \mathbf{M}^k\mathbf{e}^{(0)}. \tag{29}$$

Notes

Convergenţă

Teorema 2: O condiţie suficientă

ca procesul iterativ să fie convergent este ca norma matricei de iteraţie să fie strict subunitară:

$$\|\mathbf{M}\| < 1.$$

$$\rho(\mathbf{M}) \leq \|\mathbf{M}\|. \tag{30}$$

$$\|\mathbf{e}^{(k)}\| \leq \|\mathbf{M}\|^k \|\mathbf{e}^{(0)}\|. \tag{31}$$

Notes

Convergenţă

Mai mult

$$\begin{aligned}\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\| &= \|\mathbf{M}\mathbf{x}^{(k-1)} + \mathbf{u} - \mathbf{M}\mathbf{x}^{(k-2)} - \mathbf{u}\| = \\ &= \|\mathbf{M}(\mathbf{x}^{(k-1)} - \mathbf{x}^{(k-2)})\| \leq \\ &\leq \|\mathbf{M}\|\|\mathbf{x}^{(k-1)} - \mathbf{x}^{(k-2)}\|, \end{aligned} \tag{32}$$

$\Rightarrow$ Utilizarea unui criteriu de oprire Cauchy este pe deplin justificată.

Notes

Convergenţă

Fie o margine a erorii absolute notată cu a_k, unde $\|\mathbf{e}^{(k)}\| \leq a_k$.

$$a_k = \|\mathbf{M}\|^k a_0, \tag{33}$$

$$\log(a_k) = k \log \|\mathbf{M}\| + \log a_0. \tag{34}$$

$R(\mathbf{M}) = -\log \|\mathbf{M}\|$ se numeşte rata de convergenţă.

$$\log(a_k) = -kR(\mathbf{M}) + \log a_0. \tag{35}$$

$$R(\mathbf{M}) = \log(a_{k-1}) - \log(a_k), \tag{36}$$

Notes

Convergenţă

$$R(\mathbf{M}) = \log(a_{k-1}) - \log(a_k), \tag{37}$$

Rata de convergenţă = numărul de cifre semnificative corecte ce se câştigă la fiecare iteraţie.

Exemplu:

- $\|\mathbf{M}\| = 10^{-3}$, rata de convergenţă este 3, deci la fiecare iteraţie numărul de cifre semnificative corecte creşte cu 3.
- $\|\mathbf{M}\| = 10^{-1}$, la fiecare iteraţie se câştigă o cifră semnificativă.

OBS:

- Alegerea valorii iniţiale nu are nici o influenţă asupra convergenţei sau neconvergenţei procesului iterativ;
- În cazul unui proces iterativ convergent, valoarea iniţială afectează doar numărul de iteraţii necesar pentru atingerea unei erori impuse.

Notes

Algoritm general

```
procedură metodă_iterativă(n, B, C, b, x0, er, maxit, x)
...
xv = x0                         ; iniţializează şirul iteraţiilor
k = 0                           ; iniţializare contor iteraţii
repetă
      t = C * xv + b
      metodă_directă (n, B, t, x)
      d = ||xv − x||
      xv = x                    ; actualizează soluţia veche
      k = k + 1
cât timp d > er şi k ≤ maxit
retur
```


Notes

Algoritm general

Efortul de calcul

- poate fi făcut doar pentru o singură iteraţie;
- depinde de structura matricelor în care a fost descompusă matricea coeficienţilor;
- e consumat mai ales în calculul lui **t** şi în procedura de rezolvare directă, care în general are o complexitate liniară deoarece **B** are o structură rară, particulară.
- este de aşteptat ca procedeul iterativ să fie cu atât mai rapid convergent cu cât **B** conţine mai multă informaţie din **A**.

Notes

Metoda Jacobi: un exemplu simplu

A se descompune astfel încât **B** este diagonală.

$$\left\{\begin{array}{rcrcrcr} x & + & 2y & - & z & = & -1 \\ -2x & + & 3y & + & z & = & 0 \\ 4x & - & y & - & 3z & = & -2 \end{array}\right. \Leftrightarrow \left\{\begin{array}{rcl} x & = & -2y+z-1 \\ 3y & = & 2x-z \\ -3z & = & -4x+y-2 \end{array}\right. \tag{38}$$

$$\begin{array}{l} x^{(n)} = -2y^{(v)} + z^{(v)} - 1 \\ y^{(n)} = 2x^{(v)} - z^{(v)} \\ z^{(n)} = -4x^{(v)} + y^{(v)} - 2 \end{array} \tag{39}$$

$[0,0,0]^T$, $[-1,0,-2]^T$, $[-3,0,2]^T$, etc.

Notes

Algoritmul metodei Jacobi

Partiţionarea matricei în metodele iterative:

$$\mathbf{A} = \left[\begin{array}{cccc} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{array}\right] =$$

$$= \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \\ a_{21} & 0 & 0 & 0 \\ a_{31} & a_{32} & 0 & 0 \\ a_{41} & a_{42} & a_{43} & 0 \end{array}\right] + \left[\begin{array}{cccc} a_{11} & 0 & 0 & 0 \\ 0 & a_{22} & 0 & 0 \\ 0 & 0 & a_{33} & 0 \\ 0 & 0 & 0 & a_{44} \end{array}\right] + \left[\begin{array}{cccc} 0 & a_{12} & a_{13} & a_{14} \\ 0 & 0 & a_{23} & a_{24} \\ 0 & 0 & 0 & a_{34} \\ 0 & 0 & 0 & 0 \end{array}\right]$$

$$= \quad \mathbf{L} \quad + \quad \mathbf{D} \quad + \quad \mathbf{U}$$

$\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}$
$\mathbf{A} = \mathbf{B} - \mathbf{C}$ unde, în metoda Jacobi

$$\mathbf{B} = \mathbf{D}, \quad \mathbf{C} = -(\mathbf{L} + \mathbf{U}), \tag{40}$$

Notes

Algoritmul metodei Jacobi

Calculul recursiv al noii iteraţii

$$\mathbf{D}\mathbf{x}^{(k)} = -(\mathbf{L} + \mathbf{U})\mathbf{x}^{(k-1)} + \mathbf{b}. \tag{41}$$

Ecuaţia i:

$$a_{ii}\mathbf{x}_i^{(k)} = -\sum_{\substack{j=1 \\ j\neq i}}^{n} a_{ij}x_j^{(k-1)} + b_i. \tag{42}$$

$$\mathbf{x}_i^{(n)} = (b_i - \sum_{\substack{j=1 \\ j\neq i}}^{n} x_j^{(v)})/a_{ii} \quad i = 1, \ldots, n. \tag{43}$$

Obs: Fiecare componentă nouă poate fi calculată independent de celelalte componente noi, motiv pentru care metoda Jacobi se mai numeşte şi metoda deplasărilor simultane.

Notes

Algoritmul metodei Jacobi

```
procedură Jacobi(n, a, b, x0, er, maxit, x)
; rezolvă sistemul algebric liniar ax = b, de dimensiune n prin metoda Jacobi
întreg n                ; dimensiunea sistemului
tablou real a[n][n]     ; matricea coeficienţilor, indici de la 1
tablou real b[n]        ; vectorul termenilor liberi
; mărimi specifice procedurilor iterative
tablou real x0[n]       ; iniţializarea soluţiei
real er                 ; eroarea folosită de criteriul de oprire
întreg maxit            ; număr maxim de iteraţii admis
tablou real xv[n]       ; aproximaţia anterioară ("veche")
```


Notes

Algoritmul metodei Jacobi

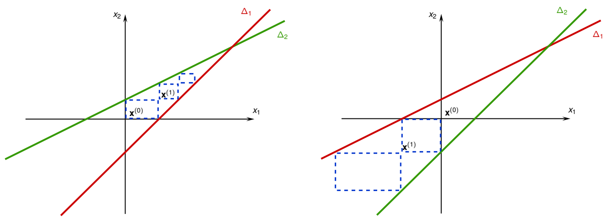
```
procedură Jacobi(n, a, b, x0, er, maxit, x)
.....
pentru i = 1, n
          xv_i = x0_i
•
k = 0                              ; iniţializare contor iteraţii
repetă
          d = 0
          pentru i = 1, n
                    s = 0
                    pentru j = 1, n
                              dacă j ≠ i
                                        s = s + a_ij * xv_j
                              •
                    •
                    x_i = (b_i − s)/a_ii
                    d = d + (x_i − xv_i)^2
          •
          d = √d
          pentru i = 1, n
                    xv_i = x_i     ; actualizează soluţia veche
          •
          k = k + 1
cât timp d > er şi k ≤ maxit
retur
```


Notes

Convergenţa metodei Jacobi

Matricea de iteraţie

$$\mathbf{M} = -\mathbf{D}^{-1}(\mathbf{L} + \mathbf{U}). \quad (44)$$



Schimbarea ordinii ecuaţiilor din sistem înseamnă schimbarea matricei de iteraţie, deci a proprietăţilor de convergenţă ale metodei Jacobi.

Rezultat util: Dacă matricea coeficienţilor este diagonal dominantă, atunci condiţia suficientă de convergenţă este satisfăcută şi algoritmul Jacobi este convergent.

Notes

Metoda Gauss-Seidel: un exemplu simplu

A se descompune astfel încât **B** este triunghiular inferioară.

$$\left\{\begin{array}{rcrcrcr} x & + & 2y & - & z & = & -1 \\ -2x & + & 3y & + & z & = & 0 \\ 4x & - & y & - & 3z & = & -2 \end{array}\right. \Leftrightarrow \left\{\begin{array}{rcrcrcl} x & & & & & = & -2y+z-1 \\ -2x & + & 3y & & & = & -z \\ 4x & - & y & - & 3z & = & -2 \end{array}\right. \tag{45}$$

$$\begin{array}{rcrcrcl} x^{(n)} & & & & & = & -2y^{(v)}+z^{(v)}-1 \\ -2x^{(n)} & + & 3y^{(n)} & & & = & -z^{(v)} \\ 4x^{(n)} & - & y^{(n)} & - & 3z^{(n)} & = & -2 \end{array} \tag{46}$$

$[0,0,0]^T$, $[-1,-2,4]^T$, $[7,10,-40]^T$, etc.

Notes

Algoritmul metodei Gauss-Seidel

Partiţionarea matricei în metodele iterative:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} =$$

$$= \begin{bmatrix} 0 & 0 & 0 & 0 \\ a_{21} & 0 & 0 & 0 \\ a_{31} & a_{32} & 0 & 0 \\ a_{41} & a_{42} & a_{43} & 0 \end{bmatrix} + \begin{bmatrix} a_{11} & 0 & 0 & 0 \\ 0 & a_{22} & 0 & 0 \\ 0 & 0 & a_{33} & 0 \\ 0 & 0 & 0 & a_{44} \end{bmatrix} + \begin{bmatrix} 0 & a_{12} & a_{13} & a_{14} \\ 0 & 0 & a_{23} & a_{24} \\ 0 & 0 & 0 & a_{34} \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$= \quad \mathbf{L} \quad + \quad \mathbf{D} \quad + \quad \mathbf{U}$$

$\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}$
$\mathbf{A} = \mathbf{B} - \mathbf{C}$, unde în metoda Gauss-Seidel

$$\mathbf{B} = \mathbf{L} + \mathbf{D}, \quad \mathbf{C} = -\mathbf{U}, \tag{47}$$

Notes

Algoritmul metodei Gauss-Seidel

Calculul recursiv al noii iteraţii

$$(\mathbf{L}+\mathbf{D})\mathbf{x}^{(k)} = -\mathbf{U}\mathbf{x}^{(k-1)} + \mathbf{b}. \quad (48)$$

Ecuaţia i:

$$\sum_{j=1}^{i-1} a_{ij} x_j^{(k)} + a_{ii} x_i^{(k)} = -\sum_{j=i+1}^{n} a_{ij} x_j^{(k-1)} + b_i. \quad (49)$$

$$\sum_{j=1}^{i-1} a_{ij} x_j^{(n)} + a_{ii} x_i^{(n)} = -\sum_{j=i+1}^{n} a_{ij} x_j^{(v)} + b_i, \quad (50)$$

Rezolvarea sistemului: prin substituţie progresivă, conform formulei:

$$\mathbf{x}_i^{(n)} = (b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(n)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(v)})/a_{ii}. \quad (51)$$

Notes

Algoritmul metodei Gauss-Seidel

Observaţii:

- Este respectat principiul lui Seidel, conform căruia o valoare nouă a unei necunoscute trebuie folosită imediat în calcule.
- O componentă nouă nu poate fi calculată independent de celelalte componente noi, motiv pentru care metoda Gauss-Seidel se mai numeşte şi metoda deplasărilor succesive

Notes

Algoritmul metodei Gauss-Seidel

```
procedură Gauss-Seidel(n, a, b, x0, er, maxit, x)
; rezolvă sistemul algebric liniar ax = b, de dimensiune n prin metoda Gauss-Seidel
; declaraţii ca la procedura Jacobi
. . .
pentru i = 1, n
      xv_i = x0_i
•
k = 0                      ; iniţializare contor iteraţii
repetă
      d = 0
      pentru i = 1, n
            s = b_i
            pentru j = 1, n
                  dacă j ≠ i
                        s = s + a_ij * xv_j
                  •
            •
            s = s/a_ii
            p = |x_i − s|
            dacă p > d
                  d = p
            •
            x_i = s
      •
      k = k + 1
cât timp d > er şi k ≤ maxit
retur
```

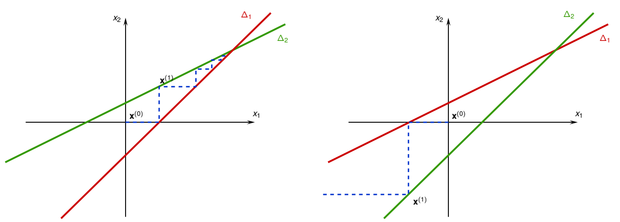

Nu este necesară memorarea soluţiei anterioare (ca la Jacobi). O dată calculată noua valoare, vechea valoare este folosită doar la calculul erorii.

Notes

Convergenţa metodei Gauss-Seidel

Matricea de iteraţie

$$\mathbf{M} = -(\mathbf{L} + \mathbf{D})^{-1}\mathbf{U}. \quad (52)$$



Schimbarea ordinii ecuaţiilor din sistem înseamnă schimbarea matricei de iteraţie, deci a proprietătilor de convergenţă ale metodei Gauss-Seidel.

Rezultate utile:

- Dacă matricea coeficienţilor este diagonal dominantă, atunci algoritmul Gauss-Seidel este convergent.
- În general, dacă metoda Jacobi este convergentă, metoda Gauss-Seidel este mai rapid convergentă.

Notes

Evaluarea algoritmilor Jacobi şi Gauss-Seidel

Efortul total de calcul depinde de numărul de iteraţii m (care depinde de matricea de iteraţie).

- Efortul de calcul pe iteraţie este $O(2n^2)$.
- Efortul total de calcul al algoritmilor Jacobi şi Gauss-Seidel *implementaţi cu matrice pline*: T = $O(2mn^2)$.
- Metoda Jacobi sau Gauss-Seidel este mai eficientă decât metoda Gauss dacă $2mn^2 < 2n^3/3$, deci dacă $m < n/3$.

Necesarul de memorie (matrice pline):

$$M_{GS} = O(n^2 + 2n) \approx O(n^2)$$

$$M_J = O(n^2 + 3n) \approx O(n^2)$$

diferenţa este nesemnificativă

Notes

Evaluarea algoritmilor Jacobi şi Gauss-Seidel

Observaţii:

1. Dacă matricea coeficienţilor este rară (memorată MM, CRS sau CCS), atunci efortul de calcul pe iteraţie se poate diminua.
2. Important: *Nu putem vorbi de umpleri ale matricei coeficienţilor* aşa cum se întâmplă în cazul algoritmului Gauss aplicat pentru matrice rare.

Metodele iterative sunt mai potrivite decât metodele directe pentru rezolvarea sistemelor cu matrice rare, într-un context hardware în care memoria disponibilă nu este suficientă rezolvării prin metode directe.

Notes

Metoda Suprarelaxării succesive (SOR)

Procedeu de accelerare a convergenţei:

$$x_i^{(n)} = x_i^{(v)} + \omega(x_i^{(n\_GS)} - x_i^{(v)}). \quad (53)$$

unde ω se numeşte factor de suprarelaxare
$\omega = 1$ corespunde metodei Gauss-Seidel.
Modificarea în pseudocodul algoritmului Gauss-Seidel:
instrucţiunea $x_i = s$ se înlocuieşte cu $x_i = x_i + \omega(s - x_i)$.

$$\begin{aligned} x_i^{(n)} &= \omega x_i^{(n\_GS)} + (1-\omega)x_i^{(v)} = \\ &= \omega(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(n)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(v)})/a_{ii} + (1-\omega)x_i^{(v)} \quad (54) \end{aligned}$$

$$(\mathbf{D} + \omega\mathbf{L})\mathbf{x}^{(n)} = [(1-\omega)\mathbf{D} - \omega\mathbf{U}]\,\mathbf{x}^{(v)} + \omega b. \quad (55)$$

Notes

Metoda Suprarelaxării succesive (SOR)

Matricea de iteraţie a metodei SOR:

$$\mathbf{M} = (\mathbf{D} + \omega\mathbf{L})^{-1}\,[(1-\omega)\mathbf{D} - \omega\mathbf{U}]\,, \quad (56)$$

- Metoda nu converge dacă $\omega \notin (0,2)$.
- Cazul $\omega \in (0,1)$ corespunde unei subrelaxări şi se foloseşte atunci când metoda Gauss-Seidel nu converge.
- Dacă matricea este simetrică şi pozitiv definită, SOR este garantat convergentă $(\forall)\ \omega \in (0,2)$.
- Alegerea valorii lui ω poate afecta în mod semnificativ rata de convergenţă.
- În general, este dificil să se calculeze apriori valoarea optimală a factorului de suprarelaxare.

Notes

Metoda Suprarelaxării succesive (SOR)

- Pentru matricile obţinute prin discretizarea unor ecuaţii cu derivate parţiale prin parcurgerea sistematică a nodurilor reţelei de discretizare se poate demonstra că

$$\omega_{opt} = \frac{2}{1 + \sqrt{1 - \rho^2}}, \tag{57}$$

unde ρ este raza spectrală a matricei de iteraţie Jacobi. În practică se folosesc tehnici euristice, ce iau în considerare dimensiunea gridului de discretizare al problemei [Templates].
- În cazul matricelor simetrice, o variantă a metodei SOR, numită SSOR, este folosită ca metodă de precondiţionare pentru metode nestaţionare.

Notes

Algoritmul general al metodelor staţionare

$$\mathbf{Ax} = \mathbf{b}$$

$$\mathbf{A} = \mathbf{B} - \mathbf{C}$$

$$\mathbf{Bx}^{(k+1)} = \mathbf{Cx}^{(k)} + \mathbf{b}. \tag{58}$$

$$\mathbf{B}(\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}) = \mathbf{b} - \mathbf{Ax}^{(k)}. \tag{59}$$

Reziduul la iteraţia k

$$\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{Ax}^{(k)}, \tag{60}$$

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \mathbf{B}^{-1}\mathbf{r}^{(k)}, \tag{61}$$

Notes

Algoritmul general al metodelor staţionare

$$\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}, \quad (62)$$

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \mathbf{B}^{-1}\mathbf{r}^{(k)}, \quad (63)$$

Staţionaritatea se referă la faptul că reziduu este întotdeauna înmulţit cu matricea $\mathbf{B}^{-1}$, aceeaşi pe parcursul tuturor iteraţiilor:

- Jacobi: $\mathbf{B} = \mathbf{D}$
- Gauss-Seidel: $\mathbf{B} = \mathbf{D} + \mathbf{L}$
- SOR: $\mathbf{B} = \omega^{-1}(\mathbf{D} + \omega\mathbf{L})$.

Nu se face inversarea propriu zisă, ci se rezolvă un sistem algebric liniar.

Notes

Algoritmul general al metodelor staţionare

```
procedură metodă_iterativă_v2(n, B, A, b, x0, er, maxit, x)
...
xv = x0                                ; iniţializează şirul iteraţiilor
k = 0                                  ; iniţializare contor iteraţii
repetă
    r = b − A · xv                     ; calculează reziduu
    metodă_directă (n, B, r, z)
    d = ||z||
    x = xv + z
    xv = x                             ; actualizează soluţia veche
    k = k + 1
cât timp d > er şi k ≤ maxit
retur
```


Notes

Algoritmul general al metodelor staţionare

- Metodele iterative sunt eficiente însă pentru matrici rare.
- În cazul matricilor pline, timpul de rezolvare cu metode iterative poate fi comparabil cu timpul de factorizare. Într-o astfel de situaţie, factorizarea este mai utilă deoarece, o dată ce factorii L şi U sunt calculaţi, rezolvarea sistemului poate fi făcută oricând pentru alt termen liber.
- De aceea, un pseudocod simplificat, în care sunt scrise operaţii cu matrice este mai general, putând fi adaptat unor matrice rare.

Notes

Lectura obligatorie pentru această săptămână

- Cap.4 din

  [1] Gabriela Ciuprina, Mihai Rebican, Daniel Ioan - Metode numerice in ingineria electrica - Indrumar de laborator pentru studentii facultatii de Inginerie electrica, Editura Printech, 2013, disponibil la

  http://mn.lmn.pub.ro/indrumar/IndrumarMN_Printech2013.pdf
- Facultativ

  [Templates] Richard Barrett, Michael Berry, Tony F. Chan, James Demmel, June M. Donato, Jack Dongarra, Victor Eijkhout,Roldan Pozo, Charles Romine, and Henk Van der Vorst, *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*, SIAM 1994. disponibil la

  http://www.netlib.org/templates/templates.pdf

Notes