

Metode iterative pentru rezolvarea sistemelor de ecuații liniare: Jacobi, Gauss-Seidel, Suprarelaxare

Cuprins

1	Obiective laborator	1
2	Noțiuni teoretice	1
2.1	Metode iterative pentru funcții	1
2.1.1	Puncte fixe	2
2.2	Metode iterative pentru sisteme de ecuații liniare	3
2.2.1	Metoda Jacobi	4
2.2.2	Metoda Gauss-Seidel	4
2.2.3	Metoda suprarelaxării	5
3	Probleme	6

1 Obiective laborator

În urma parcurgerii acestui laborator, studentul va fi capabil să rezolve sisteme de ecuații liniare utilizând metode iterative.

2 Noțiuni teoretice

Metodele exacte de rezolvare a sistemelor de ecuații liniare, având complexitate $O(n^3)$, au aplicabilitate limitată la sisteme de orid relativ mic. Pentru sisteme de dimensiuni mai mari se utilizează metode iterative, cu complexitate $O(n^2)$. Acestea utilizează relații de recurență, care prin aplicare repetată furnizează aproximații, cu precizie controlată, a soluției sistemului.

2.1 Metode iterative pentru funcții

Vom dezvolta ideea de *metode iterative* pornind de la un caz mai simplu, cel al metodelor iterative pentru funcții.

Exemplu grafic. Numărul de aur. Începem cu orice număr real pozitiv x și calculăm

$$x^{(k+1)} = \sqrt{1 + x^{(k)}}$$

La un moment dat, $|x^{(k+1)} - x^{(k)}| < \varepsilon$, unde ε este o valoare mică dată. Asta înseamnă $x^{(k+1)} \approx x^{(k)}$.

$$x = \sqrt{1 + x}$$

$$x^2 = 1 + x$$

$$x^2 - x - 1 = 0$$

$$x = \frac{1 \pm \sqrt{5}}{2}$$

$$\phi = \frac{1 + \sqrt{5}}{2} \approx 1.618033988749895$$

Soluția sistemului este dată de intersecția a două funcții: $f(x) = x$ și $g(x) = \sqrt{1+x}$. Dacă trasăm grafic aceste puncte, obținem un punct de intersecție, care este soluția sistemului.

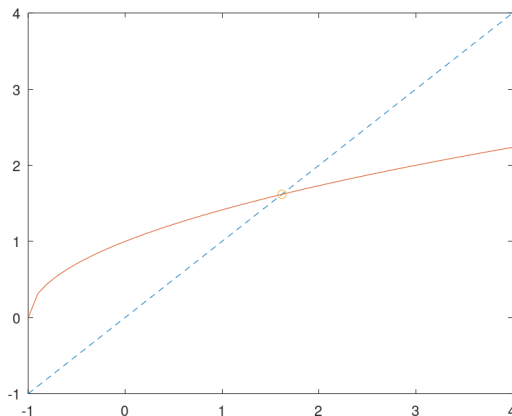


Figura 1: Numărul de aur

2.1.1 Puncte fixe

Definiție. Fie $f : M \rightarrow M$ o funcție continuă și un spațiu metric (M, d) . f este o contracție dacă există un număr real $k \in [0, 1)$ astfel încât

$$d(f(x), f(y)) \leq kd(x, y), \quad \forall x, y \in M$$

Cel mai mic număr k pentru care această relație este adevărată se numește *constantă Lipschitz*. Pentru că $k < 1$, putem spune despre f că este continuă.

Teorema lui Banach. Dacă $f : M \rightarrow M$ este o contracție pe un spațiu metric complet (M, d) , atunci există un unic punct fix $x^* \in M$ pentru care $f(x^*) = x^*$.

Pe \mathbb{R} , distanța o putem considera ca fiind $d(x, y) = |x - y|$.

$$\begin{aligned} |f(x) - f(y)| &\leq k|x - y| \\ \frac{|f(x) - f(y)|}{|x - y|} &\leq k \\ \left| \frac{f(x) - f(y)}{x - y} \right| &\leq k \end{aligned}$$

dar, din teorema lui Lagrange, știm că există un $c \in (a, b)$ astfel încât

$$|f'(c)| \leq k$$

Astfel, putem spune că iterația $x^{(k+1)} = f(x^{(k)})$ converge către un punct fix x^* dacă $|f'(c)| < 1$, pentru orice $c \in (a, b)$.

Exemplu. Fie $f(x) = \sqrt{1+x}$. Calculăm derivata funcției:

$$\begin{aligned} f'(x) &= \frac{1}{2}(1+x)^{-\frac{1}{2}} \\ f'(x) &= \frac{1}{2\sqrt{1+x}} \end{aligned}$$

Pentru orice $x \in \mathbb{R}$, $x \geq 0$, avem $|f'(x)| < 1$, deci funcția este o contracție pe \mathbb{R}_+ și are un punct fix unic.

2.2 Metode iterative pentru sisteme de ecuații liniare

Folosind teorema de mai sus, putem spune despre transformarea liniară $T : \mathbb{R}^n \rightarrow \mathbb{R}^n$ că este o contracție dacă "derivata" transformării este mai mică decât 1. Pentru că T este o funcție vectorială, derivata se calculează folosind *matricea Jacobiană*. Ne dorim ca această matrice să aibe norma subunitară, lucru care se întâmplă dacă matricea are raza spectrală mai mică decât 1.

Considerăm sistemul $Ax = b$. Ne dorim să îl rescriem pentru a ajunge la forma $x = T(x)$. Metoda prin care vom face asta e să descompunem matricea A în $A = M - N$.

$$\begin{aligned}(M - N)x &= b \\ Mx &= Nx + b \\ x &= M^{-1}Nx + M^{-1}b\end{aligned}$$

Astfel, $T(x) = M^{-1}Nx + M^{-1}b$. Notăm $G = M^{-1}N$ și $c = M^{-1}b$. G este *matricea de iterație*, iar c este *vectorul de iterație*. Jacobiana lui T este chiar matricea G , ceea ce înseamnă că, pentru convergență, avem nevoie ca raza spectrală a lui G să fie mai mică decât 1. Acest lucru se poate vedea și din analiza erorii:

$$\begin{aligned}x - x^{(k)} &= (Gx + c) - (Gx^{(k-1)} + c) = G(x - x^{(k-1)}) \\ e^{(k)} &= G^k e^{(0)}\end{aligned}$$

Pentru convergență, avem nevoie ca $\lim_{k \rightarrow \infty} e^{(k)} = 0 \equiv \lim_{k \rightarrow \infty} G^k = 0$. Pentru o demonstrație mai amplă, se poate consulta [1].

Pentru alegerea lui M și N , vom partiționa matricea A punând în evidență o matrice diagonală D , o matrice strict triunghiular inferioară L și o matrice strict triunghiular superioară U :

$$A = D - L - U$$

$$\begin{array}{c} \boxed{\begin{array}{ccc} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{array}} \\ A \end{array} = \begin{array}{c} \boxed{\begin{array}{ccc} a_{11} & & \\ & a_{22} & \\ & & a_{33} \end{array}} \\ D \end{array} - \begin{array}{c} \boxed{\begin{array}{cc} & \\ -a_{21} & \\ -a_{31} & -a_{32} \end{array}} \\ L \end{array} - \begin{array}{c} \boxed{\begin{array}{cc} & \\ & -a_{12} \quad -a_{13} \\ & & -a_{23} \end{array}} \\ U \end{array}$$

Diferența dintre următoarele metode constă în modul în care se asociază aceste matrice.

Condiția de oprire. Pentru a finaliza execuția metodelor iterative din cadrul acestui laborator vom introduce două condiții: **toleranța** (notată în continuare cu ϵ) și **numărul maxim de iterații** (notat în continuare cu N_{iter}). Toleranța ne asigură faptul ca algoritmul nu efectuează iterații mai mult decât este necesar, practic nu se continuă execuția dacă "diferența" soluțiilor a două iterații consecutive nu este semnificativă. Numărul maxim de pași garantează că algoritmul se încheie, indiferent dacă alegerea inițială pentru $x^{(0)}$ a fost una bună sau nu.

2.2.1 Metoda Jacobi

În metoda Jacobi se aleg:

$$\begin{aligned}M &= D \\N &= L + U \\G &= D^{-1}(L + U) \\c &= D^{-1}b\end{aligned}$$

Pentru că M este o matrice diagonală, inversa sa este foarte ușor de calculat. Pasul de iterație este:

$$\begin{aligned}x^{(k+1)} &= D^{-1}[(L + U)x^{(k)} + b] \\x_i^{(p+1)} &= \frac{b_i - \sum_{j \neq i} a_{ij}x_j^{(p)}}{a_{ii}}\end{aligned}$$

Algoritmul în MATLAB poate fi gândit în două moduri. Cel mai simplu este să folosim prima relație de mai sus.

Algorithm 1 Metoda Jacobi

<pre> 1: $D \leftarrow \text{diag}(\text{diag}(A))$ 2: $L \leftarrow -\text{tril}(A, -1)$ 3: $U \leftarrow -\text{triu}(A, 1)$ 4: $G \leftarrow D^{-1}(L + U)$ 5: $c \leftarrow D^{-1}b$ 6: $x \leftarrow \text{zeros}(\text{length}(b), 1)$ 7: $i \leftarrow 1$ 8: while $i \leq \text{max_iter}$ do 9: $x_{\text{prev}} \leftarrow x$ 10: $x \leftarrow G \cdot x + c$ 11: if $\ x - x_{\text{prev}}\ < \text{tol}$ then 12: break 13: end if 14: $i \leftarrow i + 1$ 15: end while </pre>	<pre> ▷ Extragem Matricea diagonală ▷ Extragem matricea inferior triunghiulară ▷ Extragem matricea superior triunghiulară ▷ Matricea de iterație ▷ Vectorul de iterație ▷ Inițializăm vectorul soluție </pre>
--	---

2.2.2 Metoda Gauss-Seidel

În metoda Gauss-Seidel se aleg:

$$\begin{aligned}M &= D - L \\N &= U \\G &= (D - L)^{-1}U \\c &= (D - L)^{-1}b\end{aligned}$$

În acest caz, matricea M este o matrice inferior triunghiulară, iar inversa nu mai este așa ușor de calculat.

Pasul de iterație este:

$$x^{(k+1)} = (D - L)^{-1}(Ux^{(k)} + b)$$

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

$$x_i = \frac{b_i - \sum_{j \neq i} a_{ij}x_j}{a_{ii}}$$

Diferența dintre metoda Jacobi și Gauss-Seidel constă în faptul că în metoda Gauss-Seidel, la calculul lui $x_i^{(p+1)}$ se folosesc valorile deja calculate pentru $x_j^{(p+1)}$ cu $j < i$.

Algorithm 2 Metoda Gauss-Seidel

```

1:  $x \leftarrow \text{zeros}(\text{length}(b), 1)$  ▷ Inițializăm vectorul soluție
2: for  $i = 1$  to  $\text{max\_iter}$  do
3:    $x_{\text{prev}} \leftarrow x$ 
4:   for  $j = 1$  to  $\text{length}(x)$  do
5:      $x[j] \leftarrow \frac{b[j] - \sum_{k \neq j} A[j,k]x[k]}{A[j,j]}$ 
6:   end for
7:   if  $\|x - x_{\text{prev}}\| < \text{tol}$  then
8:     break
9:   end if
10: end for

```

De observat la metodele Jacobi și Gauss-Seidel este că o condiție suficientă dar nu necesară pentru convergența acestora este ca matricea A să fie diagonal dominantă. Metodele Jacobi și Gauss-Seidel au proprietatea că ori sunt ambele convergente, ori niciuna nu este convergentă (teorema Stein-Rosenberg). Atunci când converg, Gauss-Seidel converge mai rapid decât Jacobi, $\rho(GS) < \rho(J) < 1$.

2.2.3 Metoda suprarelaxării

O variantă a metodei Gauss-Seidel este metoda suprarelaxării succesive (SOR). Se introduce un parametru de relaxare ω și se obține:

$$A = M - N$$

$$A = M - \omega M - N + \omega M$$

$$A = (1 - \omega)M - (N - \omega M)$$

$$A = M(\omega) - N(\omega)$$

Dacă notăm cu GS formula pentru $x_i^{(p+1)}$ de la Gauss-Seidel, atunci formula pentru SOR este:

$$x_i^{(p+1)} = (1 - \omega)x_i^{(p)} + \omega GS$$

Dacă A este simetrică și pozitiv definită, atunci pentru $\omega \in (0, 2)$ metoda SOR converge. Pentru $\omega = 1$ obținem metoda Gauss-Seidel.

Algorithm 3 Metoda SOR

```

1:  $x \leftarrow \text{zeros}(\text{length}(b), 1)$  ▷ Inițializăm vectorul soluție
2: for  $i = 1$  to  $\text{max\_iter}$  do
3:    $x_{\text{prev}} \leftarrow x$ 
4:   for  $j = 1$  to  $\text{length}(x)$  do
5:      $x[j] \leftarrow \frac{b[j] - \sum_{k \neq j} A[j,k]x[k]}{A[j,j]}$ 
6:   end for
7:    $x \leftarrow \omega \cdot x + (1 - \omega) \cdot x_{\text{prev}}$  ▷ Aplicăm relaxarea
8:   if  $\|x - x_{\text{prev}}\| < \text{tol}$  then
9:     break
10:  end if
11: end for

```

3 Probleme

1. Să se implementeze în MATLAB funcțiile pentru metodele iterative Jacobi, Gauss-Seidel și SOR.
2. Desenați grafic, pentru fiecare metodă, evoluția erorii în funcție de numărul de iterații. Testați parametrii diferiți pentru ω în metoda SOR.
3. Folosiți metoda Jacobi pentru a aproxima soluția sistemului:

$$10x_1 - 5x_2 + x_3 = 1$$

$$x_1 + 4x_2 + 3x_3 = 4$$

$$4x_1 - 3x_2 - 9x_3 = 6$$

4. Fie sistemul liniar:

$$2x + y + z = 4$$

$$x + 2y + z = 4$$

$$x + y + 2z = 4$$

Stabiliți convergența metodelor Jacobi și Gauss-Seidel și razele spectrale corespunzătoare. În caz de convergență, calculați soluția iterativă după trei pași. Alegeți voi aproximația inițială.

Referințe

- [1] Anders C. Hansen. Iterative methods for linear algebraic systems.