

Teme de laborator

Norme de matrici.

Metode iterative pentru sisteme liniare.

1. Fie $A \in \mathcal{M}_{n,n}(\mathbb{C})$ inversabilă. Se numește *număr de condiționare* (*conditioning number*) al matricii A relativ la norma operator $\|\cdot\|$, numărul

$$\mathcal{K}(A) := \mathcal{K}_{\|\cdot\|}(A) = \|A\| \|A^{-1}\|.$$

Dacă A nu este inversabilă vom scrie $\mathcal{K}(A) = \infty$.

- a) Să se arate că pentru orice normă operator $\mathcal{K}(A) \geq 1$.
b) Fie $A \in \mathcal{M}_{n,n}(\mathbb{C})$ o matrice unitară, i.e., $A^*A = I$. Arătați că

$$\mathcal{K}_2(A) = 1,$$

unde $\mathcal{K}_2(A)$ este calculată cu norma $\|\cdot\|_2$.

- c) (**Octave/MatLab**) Utilizați funcția predefinită `hilb` pentru a genera câteva matrici Hilbert. Scrieți un script Octave/MatLab care să calculeze numerele de condiționare relative la normele 2, 1 și ∞ pentru matrici Hilbert de dimensiuni $n = 3, \dots, 7$. Cum sunt aceste matrici din punct de vedere al condiționării?
2) (**Octave/MatLab**) Se consideră sistemul liniar:

$$\begin{pmatrix} 2.01 & 1.99 \\ 1.99 & 2.01 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 4 \\ 4 \end{pmatrix}$$

- i) Se observă că $x^* = (1, 1)^T$ este o soluție a sistemului de mai sus. Notăm cu A matricea sistemului și cu b membrul drept. Considerăm o perturbare \hat{b} a lui b ,

$$\hat{b} = \begin{pmatrix} 4.02 \\ 3.98 \end{pmatrix}.$$

Se observă că soluția sistemului $Ax = \hat{b}$ este $\hat{x} = (2, 0)^T$.

- ii) Verificați soluțiile sistemelor de mai sus folosind următoarele comenzi:

```
>> A= [2.01 1.99;1.99 2.01];  
>> b = [4;4];  
>> bhat = [4.02;3.98];  
>> xstar=A\b  
>> xhat =A\bhat
```

- iii) Calculați $\mathcal{K}_2(A)$, $\mathcal{K}_1(A)$ și $\mathcal{K}_\infty(A)$. Explicați diferența între x^* și \hat{x} .

2. (**Octave/MatLab**) Să se scrie o funcție cu antetul

```
function A = frank_matrix(n)
```

care să implementeze matricea Frank de dimensiune n . O matrice $A = (a_{ij})_{1 \leq i, j \leq n}$ este o matrice Frank dacă

$$a_{ij} = \begin{cases} 0, & j < i - 2, \\ n + 1 - i, & j = i - 1, \\ n + 1 - j, & j \geq i. \end{cases}$$

Printați numerele de condiționare pentru matricile Frank implementate folosind scriptul `test_frank_matrix.m`.

3. (**Octave/MatLab**) Să se scrie o funcție ce generează o matrice diagonal dominantă pe baza unei matrici date, după următoarea schemă: dacă $A = (a_{ij})_{1 \leq i, j \leq n}$ este matricea input, iar $B = (b_{ij})_{1 \leq i, j \leq n}$ este matricea output, atunci

$$b_{ij} = \begin{cases} a_{ij}, & i \neq j, \\ 1 + \sum_{k \neq i} |a_{ik}|, & i = j. \end{cases}$$

Funcția Octave/Matlab corespunzătoare va avea antetul

```
function B = create_dominant_matrix(A)
```

Scriptul de test pentru aceasta este `test_dominant_matrix.m`.

4. (**Octave/MatLab**) Folosind problema de mai sus să se genereze o matrice diagonal dominantă ce are elementele din afara diagonalei numere aleatoare, iar cele de pe diagonala principală să fie obținute în același mod ca mai sus. Funcția va avea antetul

```
function A = rand_dominant_matrix(n)
```

unde n este dimensiunea matricii. Folosiți *built-in function* `randn` pentru a genera componentele aleatorii.

5. (**Octave/MatLab**) Implementați și testați metoda lui Jacobi pentru rezolvarea sistemelor liniare. Folosiți descrierea pe componente a metodei. Mai precis, scrieți două funcții Octave/MatLab, după cum urmează:

- `function xk = iteratie_Jacobi(A,b,x0)`, unde A este matricea sistemului, b este vectorul (coloană) reprezentând membrul drept al sistemului linear, x_0 vectorul de start al iterației, iar x_k vectorul obținut după o **singură** iterație. Antetul funcției este precizat în porțiunea de cod de mai jos:

```

function xk = iteratie_Jacobi(A,b,x0)
% functie ce implementeaza o iteratie din metoda lui Jacobi
% Se foloseste forma metodei pe componente.
%
% Input:
%   - A,b: matricea si membrul drept al sistemului Ax = b
%   - x0:   valoarea de start a iteratiei (vector coloana)
%
% Output:
%   - xk:   valoarea de final a iteratiei
%
% ISA 2017-2018
- function [xk,iter,isTerminated] = metoda_Jacobi(A,b,x0,tol,maxIter)
unde A,b sunt matricea si membrul drept al sistemului liniar, x0
este vectorul inițial, maxIter este numărul maxim de iterații ad-
mis, xk este iterația finală, iter este numărul de iterații realizate,
iar isTerminated o valoare din {0,1} cu isTerminated = 0 dacă
maxIter a fost atins și isTerminated = 1 altfel. Parametrul de in-
put, tol este folosit în criteriul de terminare al iterațiilor, i.e., metoda
este oprită dacă:

```

$$\left\|x^{(k+1)} - x^{(k)}\right\|_2 \leq tol.$$

Antetul funcției este precizat în porțiunea de cod de mai jos:

```

function [xk,iter,isTerminated] = metoda_Jacobi(A,b,x0,tol,maxIter)
% functie ce implementeaza metoda lui Jacobi
% Se foloseste forma metodei pe componente.
%
% Input:
%   - A,b: matricea si membrul drept al sistemului Ax = b
%   - x0:   prima valoare in metoda iterativa
%   - tol:  valoare definita de utilizator pentru a fi folosita
%           pe post de criteriu de terminare al iteratiilor.
%   - maxIter: numarul maxim de iteratii
%
% Output:
%   - xk:   valoarea ultimei iteratii
%   - iter: numarul de iteratii pana la raspunsul final
%   - isTeminated: 0 daca s-a atins maxIter, 1 altfel
%
% ISA 2017-2018

```

În cazul în care matricea $B := D^{-1}B$ satisface

$$\|B\|_\infty < 1 \text{ sau } \|B\|_1 < 1$$

folosiți un criteriu de terminare alternativ care asigură utilizatorul că diferența între soluția aproximativă și cea exactă este mărginită de ceva cunoscut.

Folosiți scriptul `print_tabel_Jacobi.m` pentru a testa metoda lui Jacobi. Explicați pe baza cunoștințelor teoretice (analitice) valorile ultimei rubrici din tabel.