Numerical Analysis (089180) A.Y. 2022-2023
prof. Simona Perotto Luca Liverotti
simona.perotto@polimi.it luca.liverotti@polimi.it

Lab 5 – Solutions

October 28, 2022

This lab deals with the numerical resolution of a linear system

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}, \quad \text{for } \mathbb{A} \in \mathbb{R}^{n\times n}, \boldsymbol{b} \in \mathbb{R}^n$$

We will focus on direct methods (i.e. methods for which the solution is obtained in a **finite number** of operations) and, in particular on:

- LU factorization with the pivoting technique
- Cholesky decomposition
- Thomas factorization

1 LU with pivoting

When the standard LU factorization fails, dealing with non-singular matrices, we can resort to the **LU factorization with pivoting**. Pivoting is a technique which changes the order of the rows of a matrix in order to avoid null pivot elements during the LU factorization.

LU decomposition with pivoting

$$\left.\begin{array}{l} \mathbb{A}\boldsymbol{x} = \boldsymbol{b} \\ \mathbb{P}\mathbb{A} = \mathbb{L}\mathbb{U} \end{array}\right\} \longrightarrow \mathbb{L}\mathbb{U}\boldsymbol{x} = \mathbb{P}\boldsymbol{b} \longrightarrow \left\{\begin{array}{l} \mathbb{L}\boldsymbol{y} = \mathbb{P}\boldsymbol{b} \\ \mathbb{U}\boldsymbol{x} = \boldsymbol{y} \end{array}\right.$$

Notice that the rigth-hand side of the lower triangular system is modified with respect to the standard LU factorization

Exercise 5.1. Consider the linear system $\mathbb{E}\boldsymbol{x} = \boldsymbol{b}$ with

$$\mathbb{E} = \begin{bmatrix} 4 & 1 & 1 & 1 & 5 \\ 4 & 1 & 2 & 0 & 0 \\ 1 & 0 & 15 & 5 & 1 \\ 0 & 2 & 4 & 10 & 2 \\ 3 & 1 & 2 & 4 & 20 \end{bmatrix} \qquad \text{and} \qquad \boldsymbol{b} = \begin{bmatrix} 12 \\ 19 \\ 22 \\ 18 \\ 30 \end{bmatrix}$$

1. Verify with Matlab that the sufficient conditions for the existence and uniqueness of the LU decomposition without pivoting are not fulfilled by matrix $\mathbb{E}$.
2. Compute the LU of matrix $\mathbb{E}$ with the Matlab command `lu` and verify if pivoting takes place with the Matlab command `spy`.
3. Solve the system $\mathbb{E}\mathbf{x} = \mathbf{b}$ by using the LU decomposition at the previous item.

Solution Exercise 5.1.

ex_5_1.m

```
clc
clear all
close all

E = [4 1 1 1 5;
    4 1 2 0 0;
    1 0 15 5 1;
    0 2 4 10 2;
    3 1 2 4 20];

 b = [12 19 22 18 30]';

 n = length(E(1,:));

 % Let calculate the determinant of the all the minor fo the matrix E

 for i = 1:n-1
    det(E(1:i, 1:i))
 end

 % Since one of the minor determinants is equal to zero, the LU
 % factorization without pivoting does not exist!

 [L, U] = lu(E);
 detA = prod(diag(U))
 L % is not triangular!
 figure()
 spy(L)
 U

 [L, U, P] = lu(E);
 detA = prod(diag(U))
 L
 figure()
 spy(L)
 U
 P % is not the identity matrix => pivoting is needed

 y = L\(P*b);
 x = U\y

 xex = E\b
```

2 Cholesky decomposition

Let $\mathbb{A} \in \mathbb{R}^{n\times n}$ be a **symmetric** and **positive definite** matrix: then there exits a unique **lower triangular** matrix $\mathbb{H}$, with $h_{ii} > 0$, s.t.

$$\mathbb{A} = \mathbb{H}\mathbb{H}^T.$$

For any symmetric matrix $\mathbb{A}$, $\quad \mathbb{A}$ pos. def. $\iff$

- $\boldsymbol{x}^T \mathbb{A}\, \boldsymbol{x} > 0, \quad \forall \boldsymbol{x} \neq \boldsymbol{0}$
- $\lambda_i > 0, \quad \forall i = 1, \dots, n, \quad \lambda_i$ eigenvalue of $\mathbb{A}$
- $\det(\mathbb{A}_i) > 0, \quad \forall i = 1, \dots, n$

The elements of $\mathbb{H}$ can be computed by the following algorithm:

Cholesky decomposition
We set $h_{11} = \sqrt{a_{11}}$ and for $i = 2, ..., n$,

$$h_{ij} = \frac{1}{h_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} h_{ik} h_{jk} \right) \quad j = 1, \cdots i-1$$
$$h_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} h_{ik}^2}$$

Cholesky factorization is available in MATLAB via the command `R=chol(A)`, where `R` is the upper triangular matrix $\mathbb{H}^T$. Notice that the Matlab command `chol` **does not** check the symmetry of the matrix!

Cholesky decomposition

$$\left.\begin{array}{l} \mathbb{A}\boldsymbol{x} = \boldsymbol{b} \\ \mathbb{A} = \mathbb{H}\mathbb{H}^T \end{array}\right\} \longrightarrow \mathbb{H}\mathbb{H}^T \boldsymbol{x} = \boldsymbol{b} \longrightarrow \left\{\begin{array}{l} \mathbb{H}\boldsymbol{y} = \boldsymbol{b} \\ \mathbb{H}^T \boldsymbol{x} = \boldsymbol{y} \end{array}\right.$$

Exercise 5.2. Consider the matrix

$$\mathbb{A} = \begin{bmatrix} 44 & 15 & 29 & 26 & 119 \\ 15 & 33 & 32 & 18 & 15 \\ 29 & 32 & 252 & 112 & 73 \\ 26 & 18 & 112 & 124 & 90 \\ 119 & 15 & 73 & 90 & 430 \end{bmatrix}$$

1. Verify that the Cholesky decomposition can be applied to matrix $\mathbb{A}$ (the command `eig`);
2. Write a Matlab function with signature [`H`] = `MyChol(A)` to implement the Cholesky decomposition of the matrix $\mathbb{A}$;
3. Compute the Cholesky decomposition of the matrix $\mathbb{A}$ by means of both the Matlab command `chol` and the function `MyChol` and compare the results;
4. Solve the linear system with $\mathbb{A}\mathbf{x} = \mathbf{b}$, $\mathbf{b} = [1, 1, 1, 1, 1]^T$

Solution Exercise 5.2.

ex_5_1.m

```
clc
clear all
close all

A = [44 15 29 26 119;
    15 33 32 18 15;
    29 32 252 112 73;
    26 18 112 124 90;
    119 15 73 90 430];
b = [1 1 1 1 1]';

eig(A)>0
% All the eigenvalues are strictly positive, so A is positive definite. The
% simmetry is trivial.

MatH = chol(A);
MatH = MatH'
MyH = MyChol(A)

% Pay attention to the output of the functions: chol(A) returns H^T (upper
% triangular),
% whereas the function MyChol we wrote returns the matrix H (lower
% triangular).

y = MyH\b;
x = MyH'\y

% Let us compare the result obtained with the one furnished by the
% backslash implemented in Matlab.
xex = A\b
```

MyChol.m

```
function [H] = MyChol(A)

   n = length(A(1,:));
   H = zeros(n, n);

   H(1, 1) = sqrt(A(1, 1));

   for i = 2:n

      for j = 1: (i-1)
         H(i,j) = 1/H(j, j) * (A(i, j) - H(i, 1:j-1)*H(j, 1:j-1)');
      end

      H(i, i) = sqrt(A(i, i) - H(i, 1:i-1)*H(i, 1:i-1)');

   end

end
```

3 Thomas algorithm

Consider the *tridiagonal* matrix

$$\mathbb{A} = \begin{bmatrix} a_1 & c_1 & & 0 \\ e_2 & a_2 & \ddots & \\ & \ddots & \ddots & c_{n-1} \\ 0 & & e_n & a_n \end{bmatrix}$$

If the $\mathbb{LU}$ decomposition exists, then factors $\mathbb{L}$ and $\mathbb{U}$ are *bidiagonal*, namely

$$\mathbb{L} = \begin{bmatrix} 1 & & & 0 \\ \beta_2 & 1 & & \\ & \ddots & \ddots & \\ 0 & & \beta_n & 1 \end{bmatrix}, \quad \mathbb{U} = \begin{bmatrix} \alpha_1 & c_1 & & 0 \\ & \alpha_2 & \ddots & \\ & & \ddots & c_{n-1} \\ 0 & & & \alpha_n \end{bmatrix}$$

The unknown coefficients α_i and β_i can be determined by imposing the equality $\mathbb{LU} = \mathbb{A}$. This yields

$$\alpha_1 = a_1, \quad \beta_i = \frac{e_i}{\alpha_{i-1}}, \quad \alpha_i = a_i - \beta_i c_{i-1}, \quad i = 2, \ldots, n.$$

Moreover, due to the bidiagonal structure of $\mathbb{L}$ and $\mathbb{U}$, a special version of the forward and backward substitution algorithms can be applied, so that we obtain

$$(\mathbb{L}\boldsymbol{y} = \boldsymbol{b}) \quad y_1 = b_1, \quad y_i = b_i - \beta_i y_{i-1}, \quad i = 2, \ldots, n$$

$$(\mathbb{U}\boldsymbol{x} = \boldsymbol{y}) \quad x_n = \frac{y_n}{\alpha_n}, \quad x_i = \frac{y_i - c_i x_{i+1}}{\alpha_i}, \quad i = n-1, \ldots, 1$$

Exercise 5.3. Consider the tridiagonal matrix $\mathbb{A} \in \mathbb{R}^{10\times 10}$ defined as

$$\mathbb{A} = \begin{bmatrix} 1 & 11 & & & & \\ 102 & 2 & 12 & & & \\ & 103 & 3 & 13 & & \\ & \ldots & \ldots & \ldots & & \\ & & 109 & 9 & 19 & \\ & & & 110 & 10 & \end{bmatrix}.$$

Then consider the linear system $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$ such that $\boldsymbol{x}$ =`ones(10,1)`.

a. Use the Matlab commands `spdiags` and `sparse` to store the matrix in sparse format, and compare the effect of these commands with respect to the commands `diag` and `full`. Read carefully the documentation of the command `spdiags` using the `help` command.

b. Implement the Thomas algorithm and solve the linear system.

Solution Exercise 5.3.

ex_5_3.m

```
clc
clear all
close all

a = [1:10]';
c = [10:19]';
e = [102:111]';
n = length(a);
A = spdiags([e a c], [-1 0 1], n, n)
xex = ones(n,1);
b = A*xex

% Compare memory usage of the sparse and full format:
Afull = full(A)
whos % memory usage of the sparse format is less than the full format

clear Afull

[L,U,x] = thomas(A,b);
x
```

thomas.m

```
function [L,U,x] = thomas(A,b)
   %  [L,U] = lu_decomposition(A)
   %
   %  Inputs : A = input tridiagonal matrix in sparse format
   %          b = input rhs vector
   %  Outputs : L = lower triangular matrix
   %          U = upper triangular matrix
   %          x = solution of the linear system

   n = length(b);

   c = spdiags(A, 1); % of the form c = [0, c_1, c_2, ... c_{n-1}]
   a = spdiags(A, 0); % of the form a = [a_1, a_2, ... a_n]
   e = spdiags(A,-1); % of the form e = [e_2, e_3, ... e_n, 0]

   y = zeros(n,1);
   x = zeros(n,1);
   alpha = zeros(n,1); % of the form alpha = [alpha_1, alpha_2, ... alpha_n]
   beta = zeros(n,1); % of the form beta = [beta_2, beta_3, ... beta_n, 0]
   eyediag = ones(n,1);

   %% LU factorization %%
   alpha(1) = a(1);
   for i=2:n
      beta(i-1) = e(i-1)/alpha(i-1);
      alpha(i) = a(i) - beta(i-1)*c(i);
   end
   L = spdiags([beta eyediag], [-1 0], n, n);
   U = spdiags([alpha c], [0 1], n, n);

   %% Forward substitution %%
   y(1)=b(1);
   for i=2:n
      y(i) = b(i) - beta(i-1)*y(i-1);
   end

   %% Backward substitution %%
   x(n) = y(n) / alpha(n);
   for i = n-1:-1:1
      x(i) = (y(i) - c(i+1)*x(i+1))/alpha(i);
```

```
    end
end
```