Lab 7. Linear Algebra and Matrix Computations

1) Given a matrix $A = \begin{bmatrix} 5 & 4 & 3 & 2 & 1 \\ 4 & 8 & 6 & 4 & 2 \\ 3 & 6 & 9 & 6 & 3 \\ 2 & 4 & 6 & 8 & 4 \\ 1 & 2 & 3 & 4 & 5 \end{bmatrix}$, find its eigenvalues analytically (use symbolic computations) and numerically. Find the characteristic polynomial and its roots. Compare all results.

2) Given a random fourth-order real matrix, find its eigenvalues and eigenvectors. Check that each eigenvector u corresponding to an eigenvalue λ satisfies the equation $Au = \lambda u$ (substitute the eigenvalue and its corresponding eigenvector into the equation and calculate the residual $Au - \lambda u$ and the residual norm $\|Au - \lambda u\|$).

3) Given a random real matrix A of the 4-th order, find its LU-factorization. Show that $A = LU$. What happens if you use the **`isequal`** command: `isequal(A,L*U)`? Is the result true or false ? Explain the result . Hint. Compute the difference $A - LU$ in long format. Check that residual norm $\|A - LU\|$ is close to zero. Find its QRrfactorization. Show that $A = QR$. What happens if you apply the **`isequal`** command: `isequal(A,Q*R)`? Is the result true or false ? Check that the residual norm $\|A - QR\|$ is close to zero. Check that the matrix Q is orthogonal.

4) Computational costs. Carry out the following commands in Matlab :

```
n=5000;
A=rand(n); B = rand(n); x=ones(n,1);
tic, y1 = A*B*x; toc
tic, y2 = A*(B*x); toc
```

Do `y1` and `y2` calculate the same value or not? Which method is the fastest and why?

5) Well-conditioned and ill-conditioned problems. Which of the following two problems is better conditioned?

- Problem 1: solve linear system $Ax = b$ where A is random matrix, b is random vector.
- Problem 2: solve linear system $Ax = b$ where A is Hilbert matrix (**`hilb`** command), b is random vector.

Calculate condition numbers and residuals for both systems and explain what happens when the size of the matrix increases.

6) Existence of solution. Solve the following three linear systems in Matlab $Ax = b$, using symbolic computations (find analytical solution) and numerical computations (find

numerical solution). For both numerical and analytical solutions find the residual vector $r = Ax - b$ and residual norm $\|r\|_2$. For each system find the determinant and rank of the matrix A. Which of these systems has unique solution, no solutions, infinitely many solutions?

a)
$$\begin{cases} 3x_1 - 2x_2 - 5x_3 + x_4 = 3, \\ 2x_1 - 3x_2 + x_3 + 5x_4 = -3, \\ x_1 + 2x_2 - 4x_4 = -3, \\ x_1 - x_2 - 4x_3 + 9x_4 = 22. \end{cases}$$

b)
$$\begin{cases} 4x_1 - 3x_2 + 2x_3 - x_4 = 8, \\ 3x_1 - 2x_2 + x_3 - 3x_4 = 7, \\ 2x_1 - x_2 - x_4 = 6, \\ 5x_1 - 3x_2 + x_3 - 8x_4 = 1. \end{cases}$$

c)
$$\begin{cases} 2x_1 + 3x_2 - x_3 + x_4 = 1, \\ 8x_1 + 12x_2 - 9x_3 + 8x_4 = 3, \\ 4x_1 + 6x_2 + 3x_3 - 2x_4 = 3, \\ 2x_1 + 3x_2 + 9x_3 - 7x_4 = 3. \end{cases}$$

Example

```
A = [4 1 2; 3 7 1; 2 2 8];
b = [7; 11; 12];
% Numerical solution
x_n = A\b % find numerical solution
x_n2 = mldivide (A, b) % this is the same

% find residual vector and residual vector norm
r_n =b - A* x_n , norm( r_n ) %norm(v) is Eucledian norm
r_n2=b - A* x_n 2, norm(r_n2) %check it is the same as r_n

%Analytical solution using symbolic computations
A_s = sym(A) % symbolic matrix
b_s = sym(b) % symbolic vector
x_s = A_s\b_s % symbolic computations
% find residual vector and residual vector norm
r_s = b_s - A_s* x_s , norm( r_s )
```

7) **Solving linear system using \, linsolve , LU- and QR-factorizations.** Consider a linear algebraic equation in matrix form $Ax = b$, where A is a random real matrix of size n×n and b is a random vector. You can first consider a low-order system (n=4). Then, examine cases where the order of the system is n=1000, 2000, 3000, and higher. Find a solution to the system using the methods described below. For each solution method, calculate the residual norm and the solution time (**`tic`**, **`toc`** commands). Formulate

conclusions based on your results: which method is the fastest, which method is the slowest, and which method provides the best accuracy.

1) Solution using **`mldivide`** or **`\`**
2) Solution using **`linsolve`**
3) Solution by finding the inverse matrix.
4) Solution using LU factorization ($A = LU$): Factorize the matrix A using the procedure `[L,U]=lu(A)` and obtain the solution based on the factorized system $LUx = b$. When measuring the solution time, consider the cases where the time to find the factorization is taken into account (method 4a) and is not taken into account (method 4b) :

```
% A = LU ; Ax = b => x = U ^(- 1)* L ^( - 1)* b
% tic %if you measure the time to obtain factorization
[L,U]=lu(A);

tic %if you don't measure the time to obtain factorization

x4=U\(L\b);

toc
```

5) Solution using QR factorization ($A = QR$): Factorize the matrix A using the procedure `[Q,R]=qr(A)` and obtain the solution based on the factorized system $QRx = b$. When measuring the solution time, consider the cases where the time to find the factorization is taken into account (method 5a) and not taken into account (method 45):

```
% A=QR ; x=R^(-1)*Q^(-1)*b or x=R^(-1)*Q'* b,
% because Q^(-1)=Q'
% tic %if you measure the time to obtain factorization
[Q,R]=qr(A);

tic %if you don't measure the time to obtain factorization

x5=R\(Q'*b); % first method

%x55=R\(Q\b); % or the second method
toc

norm(bA*x5)

Which one is calculated faster, x5 or x55? Use the fastest
one.
```

8) Solving linear system with symmetric positive definite matrix using `\` , `linsolve`, and `chol`. Consider a linear algebraic equation in matrix form $Ax = b$, where A is a random symmetric positive-definite matrix of size n × n (the rand function) and b is a random vector. First, consider a low-order system (n = 4). Then, examine cases where the order of the system is n = 1000, 2000, 3000, and higher. Find a solution to the system using the methods described below. For each solution method, calculate the residual norm and the solution time (**`tic`**, **`toc`** commands). Draw conclusions based on the results: which method is the fastest, which is the slowest, and which method provides the best accuracy.

```
% real positive definite symmetric
A ran (n);
A=A*A'+eye(n)
```

1) Solution using **`mldivide`** or **`\`**
2) Solution using **`linsolve`** command (use `linsolve(A,b,opts)` with positive definiteness option `opts.SYM = true`, `opts.POSDEF = true`)
3) Solution using Cholesky decomposition ($A = LL^T$ for factorization with a lower triangular matrix or $A = R^T R$ for factorization with an upper triangular matrix): Factorize the matrix A using the function `R=chol(A)` and, obtain the solution based on the factorized system $LL^T x = b$ or $R^T Rx = b$. When measuring solution time, consider the cases where the time to find the factorization is taken into account (method 3a) and is not taken into account (method 3b) :

   % A=R'*R ; x=R^(-1)*R'^(-1)*b

```
% tic %if you measure the time to obtain factorization
[R,p =chol(A); % What does p show ? Check the help

tic %if you don't measure the time to obtain factorization

x2=R\(R'\b);

toc
```