Lab Session 2

MA423 : Matrix Computations July-November 2025 S. Bora

Important instructions:

(i) **Switch to `format long e` for all experiments.**

(ii) **Submit a single livescript program that contains all comments, answers and codes necessary to produce the required outputs. Ensure that the answers are correctly numbered and the file does not include any irrelevant material. The livescript program should be saved as MA423YourrollnumberLab2.mlx**

1. Write the following function programs to solve triangular systems of equations.

   (a) `x = colbackward(U,b)` to solve an upper triangular system $Ux = b$ by column oriented back substitution.

   (b) `x = rowforward(L,b)` to solve a lower triangular system $Lx = b$ by row oriented forward substitution.

2. Write a MATLAB function program $[\texttt{L},\texttt{U}] = \texttt{genp}(\texttt{A})$ which finds an LU factorization $A = LU$ of an n-by-n matrix A by performing Gaussian Elimination with no pivoting (GENP).

3. Use the MATLAB function program `[L,U] = genp(A)` to do the following:

   (a) Find the factors L and U of an LU decomposition of $A = \begin{bmatrix} 10^{-20} & 1 \\ 1 & 1 \end{bmatrix}$. What is $A - LU$?

   (b) Solve the system of equations $Ax = b$ where $b = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ by using the computed LU factorization from `genp` and the programs `rowforward` and `colbackward` for forward and backward substitution in the correct order. What is the difference of your answer with the correct solution in the 2-norm? Is it of the order of unit roundoff? To see this use $\texttt{norm}(\texttt{x}_\texttt{c} - \texttt{x})/\texttt{norm}(\texttt{x})$ where x_c is the computed solution and x is the exact solution via hand calculation.

   What can you conclude about GENP from the above algorithm? Can you identify the step at which things start to go wrong?

4. Write a function program `[L,U,p] = gepp(A)` to find a unit lower triangular matrix `L`, an upper triangular matrix `U` and a column vector `p` satisfying `A(p,:) = LU` via Gaussian Elimination with Partial Pivoting (GEPP) Your code should have the following features:

   (a) Your code should make only the most minimal changes to the `genp` code. In particular it should retain all important features of `genp` that ensure efficiency.

   (b) Unlike the `genp` code, `gepp` shoud not exit prematurely with an error message.

Then perform the following experiment:

The built in Matlab function program `lu` performs GEPP and GECP to find LU decompositions of appropriately permuted matrices, also giving the permutation matrices used in each case. Type `help lu` for details. Use the `norm` command to compare the output of your `gepp` code with the corresponding outputs of the `lu` program. The `norm` of the difference between the outputs should be `O(u)`. *The comparision should be performed for several different randomly generated matrices (use* `randn` *command for this).*

5. Write a function program `x = geppsolve(A,b)` to solve a system $Ax = b$ via GEPP. Your program should call the program `[L,U,p] = gepp(A)` and the `rowforward.m` and `colbackward.m` programs for solving upper and lower triangular systems. Then perform the following experiments.

   (a) Compare your answers with that of the MATLAB command $A\backslash b$ (which uses GEPP to solve the system) for several different choices of A and b that are randomly generated by using the `randn` command. If x and $\widehat{x}$ be the solutions from `geppsolve.m` and $A\backslash b$ respectively, you should use $\texttt{norm}(\texttt{x} - \widehat{\texttt{x}}/\texttt{norm}(\texttt{x}))$ to see the difference. If your code is correct, then $\texttt{norm}(\texttt{x} - \widehat{\texttt{x}})/\texttt{norm}(\texttt{x}) \approx \texttt{O}(\texttt{u})$.

   (b) Repeat the experiments in question 3 by using `gepp.m` and `geppsolve.m`. This time you should check $\texttt{A}(\texttt{p},:) - \texttt{LU}$ in part (a) and $\texttt{norm}(\texttt{x}_\texttt{c} - \texttt{x})$ in part (b) where x_c is the computed answer via `geppsolve.m` What is your conclusion about the performance of the two Gaussian elimination versions for this 2×2 system?

6. Given $A \in \mathbb{R}^{n \times n}$, write a function program `d = myinv(A)` that uses an *efficient* version of LU factorization to compute the inverse of A in $\frac{8}{3}n^3 + O(n^2)$ flops.