Feuille de TP N°03 – Non Triangular Systems

Exercise 01: Determinant Calculation

1. Write a recursive Matlab function "determinant" that takes a square matrix A as input and returns its determinant.
2. Test this function on a matrix of your choice and compare the result with the built-in Matlab function "det" to verify the accuracy of your result.

Reminder: The determinant of a square matrix $\boldsymbol{A}$ of order n can be calculated using the following formula (by choosing a row i):

$$\det(A) = \sum_{j=1}^{n} (-1)^{i+j} * a_{i,j} * \det(A_{i,j})$$

where $a_{i,j}$ is the element at row i and column j of matrix A, and $A_{i,j}$ is the matrix A without row i and column j.

For a square matrix $\boldsymbol{A}$ of size 2 we have:

$$\det(A) = a_{1,1} * a_{2,2} - a_{2,1} * a_{1,2}$$

Exercise 02: Gaussian Elimination

1. Write a function “**`gaussian_elimination`**” that takes as input a matrix of arbitrary coefficients $\boldsymbol{A}$ and a vector of second members $\boldsymbol{b}$, and applies Gaussian elimination to return an upper triangular matrix $\boldsymbol{A_2}$, and a column vector $\boldsymbol{b_2}$ such that.
    a. The system $\boldsymbol{A_2x=b_2}$ is equivalent to the initial system $\boldsymbol{Ax=b}$ (both systems have the same solution).
    b. The returned coefficient matrix $\boldsymbol{A_2}$ must be upper triangular.
2. Test the **“`gaussian_elimination`”** function on a random linear system of size 100:
    a. Use the function “**`matrix_type`**” to verify that the resulting matrix $\boldsymbol{A_2}$ is upper triangular.
    b. What do you observe? Explain.
3. Use your function to transform a random linear system of size 100 into an equivalent triangular system, then solve the resulting system using the function “**`solve_upper_triangular`**” (implemented in the second practical session):
    a. Verify that $\boldsymbol{b-Ax=0}$ (null vector) to confirm that your solution is correct.
    b. What do you observe? Explain.
    c. Compare your results with those obtained using Matlab's predefined solver (`x = A\b`) to confirm the correctness of your algorithm in another way.

Exercise 03: LU Factorization

1. Write a Matlab function "`lu_factorization`" that takes a matrix ***A*** as input and returns a lower triangular matrix ***L*** and an upper triangular matrix ***U***, such that ***A=LU***.
2. Write a function "`solve_lu`" that solves a linear system using the ***LU*** factorization of the coefficient matrix A. Then, test this function and compare the results with those obtained using Matlab's predefined solver to verify the correctness of your function.
3. Write a function "`determinant_lu`" that takes a matrix ***A*** as input and computes its determinant using ***LU*** factorization.
   - The computation is based on the property: ***det(A)=det(LU)=det(L)*det(U)***.
   - Use the function "`determinant_triangular`" from the second practical session to compute the determinants of matrices ***L*** and ***U***.
4. Compare the results and execution time of the "`determinant_lu`" function with those of the "`determinant`" function (from Exercise 01 in this session) by performing the calculation on a random square matrix of size 10.
   - What do you observe? Explain.

Exercise 04: Gaussian Elimination with Partial Pivoting

1. Write a function "`gauss_partial_pivot`" that takes as input a coefficient matrix ***A*** and a right hand side ***b***. The function should apply Gaussian elimination with partial pivoting to avoid zero pivots in matrices with non-zero determinants.
   a. The pivot must be chosen as the largest absolute value below the diagonal in the same column.
   b. The function should indicate if the coefficient matrix ***A*** is not invertible.
2. Consider the linear system defined by the coefficient matrix and second member vector, attempt to solve this system using:
   a. The Gaussian elimination method without pivoting (from Exercise 02).
   b. Gaussian elimination with partial pivoting.
   c. What do you observe? Explain.
3. Initialize a random linear system of size 15. Solve the system in two ways:
   a. Using Gaussian elimination without pivoting (store the solution in a variable x_1).
   b. Using Gaussian elimination with partial pivoting (store the solution in a variable x_2).
   c. Compare the two solutions and identify which is more accurate.

Hint: A solution is more accurate if it ensures a smaller residual $b-Ax$.