Aristotle University of Thessaloniki
Department of Electrical and Computer Engineering
Course: Numerical Analysis

NUMERICAL ANALYSIS PROJECT
JUNE 2023

Part A:

Your initial assignment is to create implementations of the LU and QR decomposition methods, **using Python**. The LU method should take a square matrix A as input and compute the lower triangular matrix L and the upper triangular matrix U such that A = LU. For the QR method, you should utilize Gram-Schmidt's algorithm. To verify the correctness of your code, you can calculate the difference between A and QR using the function "np.linalg.norm(A - Q*R)". This code will also be useful to answer some of the questions below.

Hilbert Matrices:

A Hilbert matrix is a matrix of the form Hij = 1 / (i+j + 1), where both indices i and j start from 0. For example a 4 x 4 Hilbert matrix is the following:

$$H = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} \\ \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} \end{bmatrix}$$

Hilbert matrices are often used to test algorithms in numerical linear algebra because they frequently exhibit "peculiar behavior"... something you will discover shortly by following the steps below:

a. Write a piece of code that creates a n x n hilbert matrix named H. Try your code for different values of n to verify its correctness.

b. Create a vector with elements equal to 1 using the code "b = np.ones((n, 1))", and then solve the system Hx = b using the factorization mentioned in the previous section.

c. Modify the value of the first element of b by a very small amount (e.g., add a number like 10^{-15}) and name the resulting vector bnew. Solve the system Hxnew = bnew and then compute the maximum absolute difference using the following line of code: "np.max(np.abs(x - xnew))". What do you observe? Does the result agree with what you would expect?

d. Create a plot with the values of n on the horizontal axis and the values of the maximum absolute difference on the vertical axis. What conclusions can you draw from this plot regarding the solution of the system Hx = b?

e. You are aware that the result of HH^{-1} should be equal to the identity matrix. For different values of n, compute HH^{-1} (using "np.linalg.inv()", to directly calculate the inverse), and then calculate the 2-norm of the difference between the identity matrix and the computed HH^{-1}. Finally, create a plot with the values of n on the horizontal axis and the values of the 2-norm (as defined earlier) on the vertical axis. What do you observe? What does this mean for the process of inverting Hilbert matrices?

Part B:

Approximation problem:

You are asked to construct a 4th degree polynomial that optimally approximates the function:

$$y = \cos(4t) + 0, 1e(t)$$

at 50 equidistant points with t ranging from 0 to 1. In this problem, we assume that e(t) is a function that produces normally distributed white noise values. In Python, you can define y as follows:

y = np.cos(4 * t) + 0.1 * np.random.randn(t.shape[0]).

First, create the vectors t and y that will contain the data/observations. Next, find the degree-4 polynomial that best approximates the above function for the given observations using the method of least squares. Solve the resulting system using LU factorization (Method 1) and QR factorization (Method 2). Calculate the sum of squared errors resulting from the above approximation for each method and plot a graph that includes the data and the curve of the optimal approximation you obtained.

You should submit an .ipynb (jupyter notebook) file that contains the code and the answers to the questions above.