Numerical Methods

Math 3338 – Spring 2022

Worksheet 12

LU-Decomposition

1 Reading

CP	6.1
NMEP	2.1, 2.2, 2.3, 2.5

Table 1: Sections Covered

2 LU-decomposition

As you should have already seen, solving a system that is triangular is quite easy. That's the basic idea behind LU-decomposition, write $A = LU$ where L is lower triangular and U is upper triangular. The idea is that if $Ax = b$ and $A = LU$, then $LUx = b$, so we solve $Ly = b$ then $Ux = y$.

How do we find L and U? Suppose

$$A = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{bmatrix}$$

Think about Gaussian elimination in terms of matrix multiplication. The first step of Gaussian elimination is to make the top left be 1, and make 0's below it. If

$$T_1 = \frac{1}{a_{00}} \begin{bmatrix} 1 & 0 & 0 \\ -a_{10} & a_{00} & 0 \\ -a_{20} & 0 & a_{00} \end{bmatrix}$$

then,

$$T_1 A = \begin{bmatrix} 1 & b_{01} & b_{02} \\ 0 & b_{11} & b_{12} \\ 0 & b_{21} & b_{22} \end{bmatrix}$$

The trick here was finding T_1. Do the multiplication $T_1 A$ on a piece of paper to see exactly what is happening. Try to justify to yourself where "Gaussian Elimination" comes into play.

Do this two more times to find,

$$T_2 = \frac{1}{b_{11}} \begin{bmatrix} b_{11} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -b_{21} & b_{11} \end{bmatrix} \qquad T_3 = \frac{1}{c_{22}} \begin{bmatrix} c_{22} & 0 & 0 \\ 0 & c_{22} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Combining these we see,

$$T_3 T_2 T_1 A = U$$

where U is upper triangular. Set $L = T^{-1}$, or $L_i = T_i^{-1}$, and we finally have

$$A = LU$$

where L is lower triangular.

It's easy to see,

$$T_1^{-1} = \begin{bmatrix} a_{00} & 0 & 0 \\ a_{10} & 1 & 0 \\ a_{20} & 0 & 1 \end{bmatrix} \qquad T_2^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & b_{11} & 0 \\ 0 & b_{21} & 1 \end{bmatrix} \qquad T_3^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & c_{22} \end{bmatrix}$$

Numerical Methods

Math 3338 – Spring 2022

Homework 12 (Due: Thursday, February 24)

Problem 1 (1 pt) Write a function called `forwardsub(A,b)` that takes an lower-triangular matrix A and a column vector b and returns a vector x so that $Ax = b$.

Problem 2 (1 pt) Write a program called `lu_decomp` that finds the LU-decomposition of a matrix A. Your input should be `A` where A is a matrix (as a nested array) and b is a column array. Return a tuple `L,U`.

Problem 3 (1 pt) Write a program called `lu_solve` that solves a system $Ax = b$ using LU-decomposition. You input should be `L,U,b` where $Ax = b$ and $A = LU$. The output is a column vector `x`.

Problem 4 (1 pt) LU-decomposition is a lot of work, it's computationally intensive. Why do we do this as opposed to Gaussian Elimination? The answer is when you need to solve a system more than once. You can cache (or save) the LU-decomposition and quickly solve follow up systems of equations. Let's test this. The file "LU.pickle" on Canvas contains a list of tuples of the form `[(A,[b])]`, where $[b]$ is a list of solutions. For each A, time how long it takes to

1. Use Gaussian Elimination to solve $Ax = b$ for each b
2. Find LU-decomposition and use that to solve $LUx = b$ for each b[1].

Make a table to compare the evaluation times, include a column with the number of b's for each A. Write a little paragraph discussing the results.[2]

[1] you should only find the LU-decomp once

[2] I didn't see any speed improvements. Do you?