The LU Factorization

Susanne Brenner and Li-Yeng Sung
(modified by Douglas B. Meade)
Department of Mathematics

Overview

The goal of this week's lab is to develop a `MATLAB` implementation of the **LU** factorization (without pivoting) of a general matrix **A**.

Part I

In this part Gaussian elimination — without pivoting — is performed to obtain the **LU** factorization of a 4×4 matrix. As with Lab 2, while this discussion addresses only 4×4 systems, you should be thinking about the changes required for a general $n \times n$ system.

Enter Problem
Create the matrix **A**:

$$\mathbf{A} = \begin{bmatrix} 3 & 2 & -1 & 2 \\ -3 & -4 & 2 & 1 \\ 6 & 2 & -2 & 11 \\ -6 & -10 & 6 & 2 \end{bmatrix}$$

Initialize the Matrix **L**

```
>> L = eye(4)                          % 4 x 4 identity matrix
>> U = A                               % start with the input matrix
```

Compute Multipliers for Column 1
This process creates the matrix **L** one component (multiplier) at a time and overwrites rows in **U**

```
>> L(2,1) = U(2,1)/U(1,1)              % multiplier to make U_{2,1} = 0
>> U(2,1:4) = U(2,1:4)-L(2,1)*U(1,1:4) % [row 2] - L(2,1) [row 1] → [row 2]
>> L(3,1) = U(3,1)/U(1,1)              % multiplier to make U_{3,1} = 0
>> U(3,1:4) = U(3,1:4)-L(3,1)*U(1,1:4) % [row 3] - L(3,1) [row 1] → [row 3]
>> L(4,1) = U(4,1)/U(1,1)              % multiplier to make U_{4,1} = 0
>> U(4,1:4) = U(4,1:4)-L(4,1)*U(1,1:4) % [row 4] - L(4,1) [row 1] → [row 4]
```

Note the patterns in these commands. Do not retype each command. Once you have entered one pair of commands, the others are easily obtained by changing the indices as necessary. *Be careful!*

Compute Multipliers for Column 2
Repeat the general process as for column 1. Note that there is no need to recompute any entries in column 1 (or row 1).

```
>> L(3,2) = U(3,2)/U(2,2)              % multiplier to make U_{3,2} = 0
>> U(3,2:4) = U(3,2:4)-L(3,2)*U(2,2:4) % [row 3] - L(3,2) [row 2] → [row 3]
>> L(4,2) = U(4,2)/U(2,2)              % multiplier to make U_{4,2} = 0
>> U(4,2:4) = U(4,2:4)-L(4,2)*U(2,2:4) % [row 4] - L(4,2) [row 2] → [row 4]
```

Compute Multipliers for Column 3

Repeat the general process as for column 1. Note that there is no need to recompute any entries in columns 1 or 2 (or rows 1 or 2).

≫ **L(4,3) = U(4,3)/U(3,3)**	**% multiplier to make U**$_{4,3} = 0$
≫ **U(4,3:4) = U(4,3:4)-L(4,3)*U(3,3:4)**	**% [row 4] - L(4,3) [row 3]** → **[row 4]**

Extract Upper and Lower Triangular Matrices

The matrix **U** is now upper triangular, and **L** is lower triangular.

≫ **L**	**% lower triangular matrix**
≫ **U**	**% upper triangular matrix**
≫ **L*U**	**% does this matrix look familiar?**

LU Factorization with `for` Loops

The next step towards the creation of a **LU** factorization for a general matrix **A** is to observe that loops can be used to simplify the previous steps.

Create an M-file containing the following:

```
% LU4 - The function in this M-file computes the L-U factorization
%       (without pivoting) of a 4x4 matrix. To use this function,
%       create a 4x4 matrix A and type [L,U]=LU4(A).

function [L,U]=LU4(A)
L = eye(4);
U = A;
for j=1:3
  for i=j+1:4
    L(i,j)=U(i,j)/U(j,j);
    U(i,j:4)=U(i,j:4) - L(i,j)*U(j,j:4);
  end
end
```

Now, return to the `MATLAB` window and type the following commands:

≫ **clear L U**	**% clear values of L and U**
≫ **help LU4**	**% display help information for** `LU4`
≫ **[L,U] = LU4(A)**	**% are these the same L and U as before?**

Clear all variables before you begin to work on Part II.