Illustration of Gaussian elimination to find $LU$ factorization.

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}$$
- Compute “multipliers”:

\[ m_{i1} = \frac{a_{i1}}{a_{11}}, \quad i = 2, 3, 4 \]

- Eliminate entries in first column:

\[ i\text{th row} = i\text{th row} - m_{i1} \times 1\text{st row} \]

\[
A = \begin{bmatrix}
a_{11} & a_{12} & a_{13} & a_{14} \\
a_{21} & a_{22} & a_{23} & a_{24} \\
a_{31} & a_{32} & a_{33} & a_{34} \\
a_{41} & a_{42} & a_{43} & a_{44}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
a_{11} & a_{12} & a_{13} & a_{14} \\
0 & a_{22}^{(1)} & a_{23}^{(1)} & a_{24}^{(1)} \\
0 & a_{32}^{(1)} & a_{33}^{(1)} & a_{34}^{(1)} \\
0 & a_{42}^{(1)} & a_{43}^{(1)} & a_{44}^{(1)}
\end{bmatrix}
\]

where \( a_{ij}^{(1)} = a_{ij} - m_{i1} \times a_{1j} \)
• Compute “multipliers”:

\[ m_{i2} = \frac{a_{i2}^{(1)}}{a_{22}^{(1)}}, \quad i = 3, 4 \]

• Eliminate entries in second column:

\[ \text{ith row} = \text{ith row} - m_{i2} \times \text{2nd row} \]

\[
A = \begin{bmatrix}
    a_{11} & a_{12} & a_{13} & a_{14} \\
    a_{21} & a_{22} & a_{23} & a_{24} \\
    a_{31} & a_{32} & a_{33} & a_{34} \\
    a_{41} & a_{42} & a_{43} & a_{44}
\end{bmatrix} \rightarrow \begin{bmatrix}
    a_{11} & a_{12} & a_{13} & a_{14} \\
    0 & a_{22}^{(1)} & a_{23}^{(1)} & a_{24}^{(1)} \\
    0 & a_{32}^{(1)} & a_{33}^{(1)} & a_{34}^{(1)} \\
    0 & a_{42}^{(1)} & a_{43}^{(1)} & a_{44}^{(1)}
\end{bmatrix}
\rightarrow \begin{bmatrix}
    a_{11} & a_{12}^{(1)} & a_{13}^{(1)} & a_{14}^{(1)} \\
    0 & a_{22}^{(1)} & a_{23}^{(1)} & a_{24}^{(1)} \\
    0 & a_{33}^{(2)} & a_{34}^{(2)} \\
    0 & a_{43}^{(2)} & a_{44}^{(2)}
\end{bmatrix}
\]

where \( a_{ij}^{(2)} = a_{ij}^{(1)} - m_{i2} \times a_{2j}^{(1)} \)
• Compute “multipliers”:

\[ m_{i3} = \frac{a_{i3}^{(2)}}{a_{33}^{(2)}}, \quad i = 4 \]

• Eliminate entries in third column:

\[ \text{ith row} = \text{ith row} - m_{i3} \times \text{3rd row} \]

\[
A = \begin{bmatrix}
a_{11} & a_{12} & a_{13} & a_{14} \\
a_{21} & a_{22} & a_{23} & a_{24} \\
a_{31} & a_{32} & a_{33} & a_{34} \\
a_{41} & a_{42} & a_{43} & a_{44}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
a_{11} & a_{12} & a_{13} & a_{14} \\
0 & a_{22}^{(1)} & a_{23}^{(1)} & a_{24}^{(1)} \\
0 & a_{32}^{(1)} & a_{33}^{(1)} & a_{34}^{(1)} \\
0 & a_{42}^{(1)} & a_{43}^{(1)} & a_{44}^{(1)}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
a_{11} & a_{12} & a_{13} & a_{14} \\
0 & a_{22}^{(1)} & a_{23}^{(1)} & a_{24}^{(1)} \\
0 & 0 & a_{33}^{(2)} & a_{34}^{(2)} \\
0 & 0 & a_{43}^{(2)} & a_{44}^{(2)}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
a_{11} & a_{12} & a_{13} & a_{14} \\
0 & a_{22}^{(1)} & a_{23}^{(1)} & a_{24}^{(1)} \\
0 & 0 & a_{33}^{(2)} & a_{34}^{(2)} \\
0 & 0 & 0 & a_{44}^{(3)}
\end{bmatrix}

\text{where } a_{ij}^{(3)} = a_{ij}^{(2)} - m_{i3} \times a_{3j}^{(2)}
Observation: Gaussian elimination reduces $A$ to an upper triangular matrix:

\[
A = \begin{bmatrix}
a_{11} & a_{12} & a_{13} & a_{14} \\
a_{21} & a_{22} & a_{23} & a_{24} \\
a_{31} & a_{32} & a_{33} & a_{34} \\
a_{41} & a_{42} & a_{43} & a_{44}
\end{bmatrix} \rightarrow \cdots \rightarrow 
\begin{bmatrix}
a_{11} & a_{12} & a_{13} & a_{14} \\
0 & a_{22}^{(1)} & a_{23}^{(1)} & a_{24}^{(1)} \\
0 & 0 & a_{33}^{(2)} & a_{34}^{(2)} \\
0 & 0 & 0 & a_{44}^{(3)}
\end{bmatrix}
\]

Guess: This probably gives the $U$ in the $LU$ factorization.

Question: If that’s true, where is $L$?
Consider the matrix:

\[
M_1 = \begin{bmatrix}
1 & 0 & 0 & 0 \\
-m_{21} & 1 & 0 & 0 \\
-m_{31} & 0 & 1 & 0 \\
-m_{41} & 0 & 0 & 1 \\
\end{bmatrix}, \quad m_i = \text{multipliers for 1st column} = \frac{a_{i1}}{a_{11}}
\]

Then observe that:

\[
A_1 = M_1A = \begin{bmatrix}
1 & 0 & 0 & 0 \\
-m_{21} & 1 & 0 & 0 \\
-m_{31} & 0 & 1 & 0 \\
-m_{41} & 0 & 0 & 1 \\
\end{bmatrix} \begin{bmatrix}
a_{11} & a_{12} & a_{13} & a_{14} \\
a_{21} & a_{22} & a_{23} & a_{24} \\
a_{31} & a_{32} & a_{33} & a_{34} \\
a_{41} & a_{42} & a_{43} & a_{44} \\
\end{bmatrix} = \begin{bmatrix}
a_{11} & a_{12} & a_{13} & a_{14} \\
0 & a_{22}^{(1)} & a_{23}^{(1)} & a_{24}^{(1)} \\
0 & a_{32}^{(1)} & a_{33}^{(1)} & a_{34}^{(1)} \\
0 & a_{42}^{(1)} & a_{43}^{(1)} & a_{44}^{(1)} \\
\end{bmatrix}
\]
Similarly, if

\[ M_2 = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & -m_{32} & 1 & 0 \\
0 & -m_{42} & 0 & 1
\end{bmatrix}, \quad m_{i2} = \text{multipliers for 2nd column} = \frac{a_{i2}}{a_{22}} \]

Then

\[
A_2 = M_2(M_1 A) = M_2 A_1
\]

\[
= \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & -m_{32} & 1 & 0 \\
0 & -m_{42} & 0 & 1
\end{bmatrix}
\begin{bmatrix}
a_{11} & a_{12} & a_{13} & a_{14} \\
0 & a_{22} & a_{23} & a_{24} \\
0 & a_{32} & a_{33} & a_{34} \\
0 & a_{42} & a_{43} & a_{44}
\end{bmatrix}
\]

\[
= \begin{bmatrix}
a_{11} & a_{12} & a_{13} & a_{14} \\
0 & a_{22} & a_{23} & a_{24} \\
0 & 0 & a_{33} & a_{34} \\
0 & 0 & a_{43} & a_{44}
\end{bmatrix}
\]
Finally, if

\[
M_3 = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & -m_{43} & 1
\end{bmatrix}, \quad m_{i3} = \text{multiplier for 3rd column} = \frac{a_{i3}}{a_{33}}
\]

Then

\[
A_3 = M_3(M_2M_1A) = M_3A_2
\]

\[
= \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & -m_{43} & 1
\end{bmatrix} \begin{bmatrix}
a_{11} & a_{12} & a_{13} & a_{14} \\
0 & a_{22} & a_{23} & a_{24} \\
0 & 0 & a_{33} & a_{34} \\
0 & 0 & a_{43} & a_{44}
\end{bmatrix} = \begin{bmatrix}
a_{11} & a_{12} & a_{13} & a_{14} \\
0 & a_{22} & a_{23} & a_{24} \\
0 & 0 & a_{33} & a_{34} \\
0 & 0 & 0 & a_{44}
\end{bmatrix}
\]
Therefore, we have:

\[ M_3M_2M_1A = U \]

Observations:

- Each \( M_i \) is nonsingular, so
  \[
  A = M_1^{-1}M_2^{-1}M_3^{-1}U
  \]

- Each \( M_i \) is unit lower triangular, so
  \( M_i^{-1} \) is unit lower triangular

- The product of unit lower triangular matrices is unit lower triangular, so
  \[
  L = M_1^{-1}M_2^{-1}M_3^{-1}
  \]
  is unit lower triangular.

Therefore, Gaussian elimination can be used to get

\[ A = LU \]

where \( L \) is unit lower triangular, and \( U \) is upper triangular.
One final observation: We do not need to form $M_i$ or $M_i^{-1}$ to get $L$. In fact, $L$ is simply the matrix with:

- Ones on the main diagonal
- Zeros above the main diagonal
- The entries below the main diagonal are the multipliers, $m_{ij}$. That is,

$$L = \begin{bmatrix}
1 & 0 & 0 & 0 \\
m_{21} & 1 & 0 & 0 \\
m_{31} & m_{32} & 1 & 0 \\
m_{41} & m_{42} & m_{43} & 1
\end{bmatrix}$$

This can be proved directly by verifying that

$$M_i = \begin{bmatrix}
1 & \cdots & 1 \\
-\ m_{i+1,i} & 1 & \ \cdots \\
\vdots & \vdots & \vdots \\
-\ m_{n,i} & \cdots & 1
\end{bmatrix} \Leftrightarrow M_i^{-1} = \begin{bmatrix}
1 & \cdots & 1 \\
\ \ m_{i+1,i} & 1 & \ \cdots \\
\vdots & \vdots & \vdots \\
\ m_{n,i} & \cdots & 1
\end{bmatrix}$$

Then verify that $L = M_1^{-1}M_2^{-1}\cdots M_{n-1}^{-1}$ has the above form.
An algorithm for computing the $LU$ factorization could look like:

Initialize $L$ to be an $n$-by-$n$ identity matrix

for $k = 1, 2, \ldots, n-1$

    for $i = k+1, \ldots, n$
        $L(i,k) = A(i,k) / A(k,k)$
    end

    for $j = k+1, \ldots, n$
        for $i = k+1, \ldots, n$
            $A(i,j) = A(i,j) - L(i,k)*A(k,j)$
        end
    end

end

Set $U =$ upper triangular part of $A$
One common improvement typically done is to conserve storage by observing:

- The entries below the main diagonal of $U$ are 0, so they do not need to be explicitly stored.

- The same thing can be said for the entries of $L$ above the main diagonal.

- Furthermore, the entries on the main diagonal of $L$ are always 1, so they don’t need to be explicitly stored.

- Therefore, we can conserve storage by “overwriting” $A$ with $U$ on and above the main diagonal, and $L$ below the main diagonal.

That is,

$$A = \begin{bmatrix}
  u_{11} & u_{12} & u_{13} & u_{14} \\
  l_{21} & u_{22} & u_{23} & u_{24} \\
  l_{31} & l_{32} & u_{33} & u_{34} \\
  l_{41} & l_{42} & l_{43} & u_{44}
\end{bmatrix}$$
An algorithm for computing the $LU$ factorization, overwriting $A$ with the factors, could look like:

for $k = 1, 2, \ldots, n-1$

for $i = k+1, \ldots, n$
    $A(i,k) = A(i,k) / A(k,k)$
end

for $j = k+1, \ldots, n$
    for $i = k+1, \ldots, n$
        $A(i,j) = A(i,j) - A(i,k)*A(k,j)$
    end
end

end
An Matlab function, exploiting the vector and array operations available in Matlab, could look like:

```matlab
function A = MyLU(A)

% Overwrite A with L and U, where A = LU.
%
% more comments ...
%
[m,n] = size(A);

if ( m ~= n)
    error('MyLU only works for square matrices.')
end

for k = 1:n-1
    A(k+1:n,k) = A(k+1:n,k) / A(k,k);
    A(k+1:n,k+1:n) = A(k+1:n,k+1:n) - A(k+1:n,k) * A(k,k+1:n);
end
```
An Matlab function, exploiting the vector and array operations available in Matlab, could look like:

```matlab
function [L, U] = MyLU(A)
%
% Explicitly compute L and U, where A = LU.
% more comments ...
%
[m,n] = size(A);

if ( m ~= n)
    error('MyLU only works for square matrices.‘)
end

for k = 1:n-1
    A(k+1:n,k) = A(k+1:n,k) / A(k,k);
    A(k+1:n,k+1:n) = A(k+1:n,k+1:n) - A(k+1:n,k) * A(k,k+1:n);
end

L = eye(n) + tril(A, -1);  
U = triu(A);
```
An Matlab function, exploiting the vector and array operations available in Matlab, could look like:

```matlab
function [L, U] = MyLU(A)

% Either overwrite A or explicitly compute L and U, where A = LU.

% more comments ...

[m,n] = size(A);

if ( m ~= n)
    error('MyLU only works for square matrices. ')
end

for k = 1:n-1
    A(k+1:n,k) = A(k+1:n,k) / A(k,k);
    A(k+1:n,k+1:n) = A(k+1:n,k+1:n) - A(k+1:n,k) * A(k,k+1:n);
end

if nargout == 2
    L = eye(n) + tril(A, -1);
    U = triu(A);
else
    L = A;
end
```

Using \texttt{nargout},

- If you want to overwrite \( A \) with \( L \) and \( U \):
  \[
  \texttt{>> A = MyLU(A)}; \\
  \]

- If you don’t want to overwrite \( A \), you can use it like:
  \[
  \texttt{>> [L, U] = MyLU(A)}; \\
  \]
Some final remarks on $A = LU$

- The computational cost of computing the $LU$ factorization is $2n^3/3$ flops. (Check!)

- Recall that forward and backward solves with the $L$ and $U$ factors costs $n^2$ flops. Thus, if we have several systems to solve, with the same matrix $A$, but different right hand side vectors, it is cheaper to compute $A = LU$ once and use the forward and backward solves, rather than to use Gaussian (or Gauss-Jordan) elimination on each system.

- Our approach to computing $A = LU$ only works if each of the pivots, $a_{kk}^{(k-1)}$, is nonzero.

**Theorem:** If all of the principal submatrices,

$$A_k = \begin{bmatrix} a_{11} & \cdots & a_{1k} \\
& & \\
& & \\
a_{k1} & \cdots & a_{kk} \end{bmatrix}$$

are nonsingular, then each pivot $a_{kk}^{(k-1)}$ is nonzero, and thus $A$ has an $LU$ factorization.
In our code, we could check for zero pivots:

```matlab
function [L, U] = MyLU(A)
    %
    %  some comments here ...
    %
    [m,n] = size(A);

    if ( m ~= n)
        error('MyLU only works for square matrices.‘)
    end

    for k = 1:n-1
        if A(k,k) == 0
            error('MyLU fails!‘)
        end
        A(k+1:n,k) = A(k+1:n,k) / A(k,k);
        A(k+1:n,k+1:n) = A(k+1:n,k+1:n) - A(k+1:n,k) * A(k,k+1:n);
    end

    if nargout == 2
        L = eye(n) + tril(A, -1);
        U = triu(A);
    else
        L = A;
    end
```
Remarks:

• Zero pivots are easy to recognize.

• The real danger is if a pivot is very small, but not zero.

• What does “very small” mean?