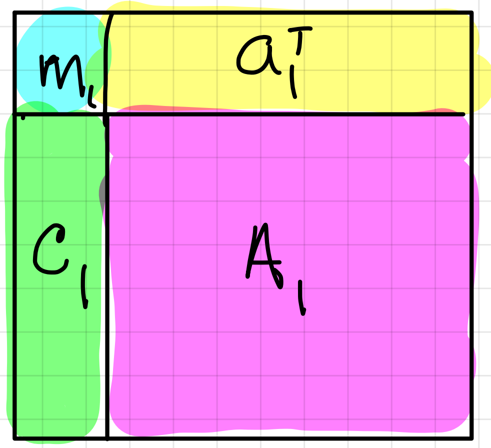
LU Decomposition Algorithm

LU Algorithm – $A \in \mathbb{R}^{4\times4}$

$c_1 = 3\times1$ column vector

a_1 3×1 column vector



$U_0 = A$

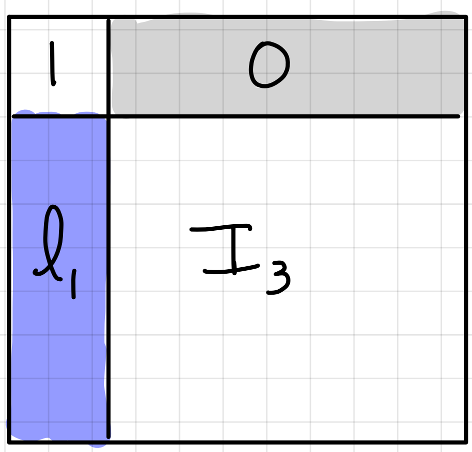
Pivot

Unchanged

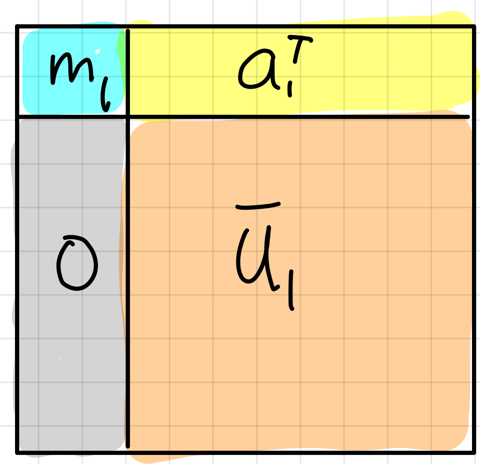
eliminated

modified

zero



L_1



U_1

l_1 multipliers $= \dfrac{c_1}{m_1}$

column vector

$$\bar{U}_1 = A_1 - \frac{1}{m_1} c_1 a_1^T$$

3x3 outer product.

Outer Product Step

$$\begin{bmatrix} m_1 & a_1^T \\ c_1 & A_1 \end{bmatrix}$$

$$U_0 = A$$

$$\bar{U}_1 = A_1 - \frac{1}{m_1} c_1 a_1^T = A_1 - l_1 a_1^T$$

$$\begin{array}{cccc} & & & \boxed{a_{12} \;\; a_{13} \;\; a_{14}} \quad \text{Pivot row} \\ \underbrace{\begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \end{bmatrix}}_{\bar{U}_1} = & \underbrace{\begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \end{bmatrix}}_{A_1} - & \underbrace{\begin{bmatrix} l_{21} \\ l_{31} \\ l_{41} \end{bmatrix}}_{l_1} & \underbrace{\begin{matrix} l_{21}a_{12} & l_{21}a_{13} & l_{21}a_{14} \\ l_{31}a_{12} & l_{31}a_{13} & l_{31}a_{14} \\ l_{41}a_{12} & l_{41}a_{13} & l_{41}a_{14} \end{matrix}}_{\substack{3 \times 3 \\ \text{outer product.}}} \end{array}$$

Apply the same procedure to the 3×3 submatrix $\bar{U}_1$

$$U_1 = \left[\begin{array}{c|c} m_1 & a_1^T \\ \hline 0 & \underbrace{\begin{array}{c|c} m_2 & a_2^T \\ \hline c_2 & A_2 \end{array}}_{\bar{U}_1} \end{array}\right]$$

$$L_2 = \left[\begin{array}{c|c|c} 1 & \multicolumn{2}{c}{0} \\ \hline \ell_1 & 1 & 0 \\ \cline{2-3} & \ell_2 & I_2 \end{array}\right] \qquad U_2 = \left[\begin{array}{c|c|c} m_1 & \multicolumn{2}{c}{a_1^T} \\ \hline 0 & m_2 & a_2^T \\ \cline{2-3} & 0 & \bar{U}_2 \end{array}\right]$$

$$\ell_2 = \frac{c_2}{m_2} \qquad \bar{U}_2 = A_2 - \frac{1}{m_2} c_2 a_2^T$$

2x2 outer product

Apply the same procedure to the 2×2 submatrix $\bar{U}_2$

$$U_2 = \left[\begin{array}{c|c|c|c} m_1 & \multicolumn{3}{c}{a_1^T} \\ \hline & m_2 & \multicolumn{2}{c}{a_2^T} \\ \cline{2-4} 0 & 0 & m_3 & a_3^T \\ \cline{3-4} & & c_3 & A_3 \end{array}\right] \begin{array}{l} \\ \\ \left.\begin{array}{l} \\ \\ \end{array}\right\} \bar{U}_2 \end{array}$$

$$\underbrace{\qquad\qquad}_{\bar{U}_2}$$

$$L_3 = \left[\begin{array}{c|c|c|c} 1 & \multicolumn{3}{c}{0} \\ \hline & 1 & \multicolumn{2}{c}{0} \\ \cline{2-4} l_1 & l_2 & 1 & 0 \\ \cline{3-4} & & l_3 & 1 \end{array}\right] \qquad U_3 = \left[\begin{array}{c|c|c|c} m_1 & \multicolumn{3}{c}{a_1^T} \\ \hline & m_2 & \multicolumn{2}{c}{a_2^T} \\ \cline{2-4} 0 & 0 & m_3 & a_3^T \\ \cline{3-4} & & 0 & \bar{u}_3 \end{array}\right]$$

$$l_3 = \frac{c_3}{m_3} \qquad \bar{u}_3 = A_3 - \frac{1}{m_3} c_3 a_3^T$$

1×1 outer product

Apply same procedure to the 1×1 submatrix $\bar{u}_3$

$$
U_3 = \begin{bmatrix} m_1 & a_1^T & & \\ 0 & m_2 & a_2^T & \\ 0 & 0 & m_3 & a_3^T \\ 0 & 0 & 0 & \underbrace{m_4}_{\bar{u}_3} \end{bmatrix} \begin{matrix} \\ \\ \\ \}\ \bar{u}_3 \end{matrix}
$$

$$
\underset{L}{\begin{bmatrix} 1 & 0 & 0 & 0 \\ l_1 & 1 & 0 & 0 \\ l_1 & l_2 & 1 & 0 \\ l_1 & l_2 & l_3 & 1 \end{bmatrix}}
\qquad
\underset{U}{\begin{bmatrix} m_1 & a_1^T & & \\ 0 & m_2 & a_2^T & \\ 0 & 0 & m_3 & a_3^T \\ 0 & 0 & 0 & m_4 \end{bmatrix}}
$$

$$A = LU$$

LU Decomposition

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

N = size(A,1);

U = A;
L = eye(N);     % Initialize using identity matrix

% Decomposition
for k = 1:N-1

    % Get multiplier, vectors and submatrix
    m  = U(k,k);                % Multiplier
    ck = U(k+1:end,k);          % column vector
    ak = U(k,k+1:end)';         % Use transpose to get a column vector
    Ak = U(k+1:end,k+1:end);    % Submatrix

    % Update L
    lk = ck/m;
    L(k+1:end,k) = lk;

    % Update U
    U(k+1:end,k) = 0;                       % Zero out variables
    U(k+1:end,k+1:end) = Ak - lk*ak';       % Outer product used
end

end
```

Results should satisfy

$$A = LU$$

Results

```
>> A = [3, -7, -2, 2; -3, 5, 1, 0; 6,-4,0,-5; -9,5,-5,12]; disp(A)
     3    -7    -2     2
    -3     5     1     0
     6    -4     0    -5
    -9     5    -5    12
```

A

```
>> [L,U] = lu_math(A);
>> disp(L); disp(U)
     1     0     0     0
    -1     1     0     0
     2    -5     1     0
    -3     8     3     1
```

L

```
     3    -7    -2     2
     0    -2    -1     2
     0     0    -1     1
     0     0     0    -1
```

U

```
>> disp(L*U)
     3    -7    -2     2
    -3     5     1     0
     6    -4     0    -5
    -9     5    -5    12
```

LU

Bonus question: What is the determinant of A?

Results using Matlab LU

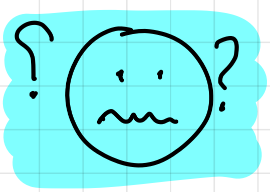
```
>> A = [3, -7, -2, 2; -3, 5, 1, 0; 6,-4,0,-5; -9,5,-5,12]; disp(A)
     3    -7    -2     2
    -3     5     1     0
     6    -4     0    -5
    -9     5    -5    12

>> [L,U] = lu(A);
>> disp(L); disp(U)
   -0.3333    1.0000         0         0
    0.3333   -0.6250   -0.1304    1.0000
   -0.6667    0.1250    1.0000         0
    1.0000         0         0         0

   -9.0000    5.0000   -5.0000   12.0000
         0   -5.3333   -3.6667    6.0000
         0         0   -2.8750    2.2500
         0         0         0    0.0435

>> disp(L*U)
    3.0000   -7.0000   -2.0000    2.0000
   -3.0000    5.0000    1.0000    0.0000
    6.0000   -4.0000         0   -5.0000
   -9.0000    5.0000   -5.0000   12.0000
```

The columns/rows of L have been permuted; the pivots in U are different from what our code produces.



Results using Matlab LU

```
>> A = [3, -7, -2, 2; -3, 5, 1, 0; 6,-4,0,-5; -9,5,-5,12]; disp(A)
     3    -7    -2     2
    -3     5     1     0
     6    -4     0    -5
    -9     5    -5    12
```

A

```
>> [L,U,P] = lu(A);
>> disp(L); disp(U); disp(P)
    1.0000         0         0         0
   -0.3333    1.0000         0         0
   -0.6667    0.1250    1.0000         0
    0.3333   -0.6250   -0.1304    1.0000
```

L

```
   -9.0000    5.0000   -5.0000   12.0000
         0   -5.3333   -3.6667    6.0000
         0         0   -2.8750    2.2500
         0         0         0    0.0435
```

U

```
     0     0     0     1
     1     0     0     0
     0     0     1     0
     0     1     0     0
```

P

Matlab permutes the rows of A.

```
>> disp(L*U)
   -9.0000    5.0000   -5.0000   12.0000
    3.0000   -7.0000   -2.0000    2.0000
    6.0000   -4.0000         0   -5.0000
   -3.0000    5.0000    1.0000    0.0000

>> disp(P*A)
    -9     5    -5    12
     3    -7    -2     2
     6    -4     0    -5
    -3     5     1     0
```

$R_1 \leftarrow R_4$
$R_2 \leftarrow R_1$
$R_3 \leftarrow R_3$
$R_4 \leftarrow R_2$

LU Decomposition Algorithm with Partial Pivoting

Partial Pivoting

Why does matlab permute the rows?

Reason #1: We hit a zero pivot.

Example:

$$A = \begin{bmatrix} 0 & -3 \\ 2 & 4 \end{bmatrix} \quad m_1 = 0 \quad ☹$$

We have a zero pivot, but the matrix is clearly invertible. $(\det(A) = 6 \neq 0)$

We need to swap rows before continuing

permute rows of A

already in upper triangular form

$$PA = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & -3 \\ 2 & 4 \end{bmatrix} = \begin{bmatrix} 2 & 4 \\ 0 & -3 \end{bmatrix}$$

The LU decomposition of PA is obvious

$$L = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad U = \begin{bmatrix} 2 & 4 \\ 0 & -3 \end{bmatrix} \quad PA = LU$$

2 non-zero pivots after swapping rows.

Or we hit a small pivot.

Or, we hit a very small pivot

$$A = \begin{bmatrix} 10^{-10} & 4 \\ 2 & 1 \end{bmatrix} \quad m_1 \ll 1$$

one row operation:

$$U = \begin{bmatrix} 10^{-10} & 4 \\ 0 & 1 - \left(\frac{2}{10^{-10}}\right)4 \end{bmatrix} \quad \begin{matrix} 24 \times 10^{10} \\ = 2.4 \times 10^{11} \end{matrix}$$

$$= \begin{bmatrix} 10^{-10} & 4 \\ 0 & -7.9999\ldots9 \times 10^{10} \end{bmatrix}$$

really large pivot.

General idea to avoid zero pivots and very small pivots:

Always choose largest entry in column as next pivot.

$$A = \begin{bmatrix} 10^{-10} & 4 \\ 2 & 1 \end{bmatrix} \quad m_1 = 10^{-10}$$

$$PA = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} 10^{-10} & 4 \\ 2 & 1 \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 10^{-10} & 4 \end{bmatrix}$$

rows swaped

$$L = \begin{bmatrix} 1 & 0 \\ 5\times 10^{-11} & 1 \end{bmatrix} \quad U = \begin{bmatrix} 2 & 1 \\ 0 & \underbrace{4 - 5\times 10^{-11}}_{\approx 4} \end{bmatrix}$$

Both zero pivot and small pivot can be fixed by "partial pivoting".

Choose largest pivot in the column.

Store permuted matrix.

LU Decomposition with Partial Pivoting.

$$\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} \left[\begin{array}{c|c} c_1 & A_1 \\ c_1 & A_1 \\ \hline m_1 & a_1^T \\ \hline c_1 & A_1 \end{array}\right]$$

m_1 = largest entry (in magnitude) in rows 1:4 in column 1

keep track of positions

$$\left[\begin{array}{c|c} 1 & 0 \\ \hline \ell_{21} & \\ \ell_{31} & I_3 \\ \ell_{41} & \end{array}\right] \qquad \left[\begin{array}{c|c} m_1 & a_1^T \\ \hline 0 & \bar{U}_1 \end{array}\right]$$

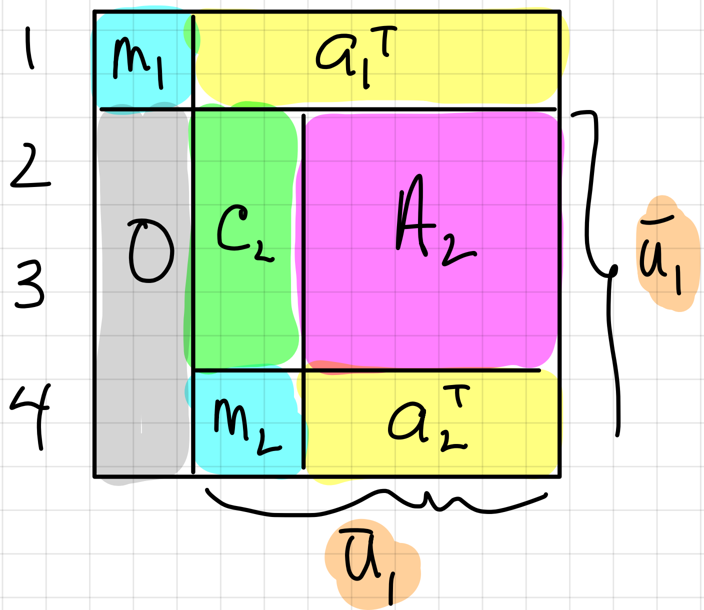
$L_1 \qquad \ell_1 = \dfrac{c_1}{m_1} \qquad\qquad U_1$

ℓ_1

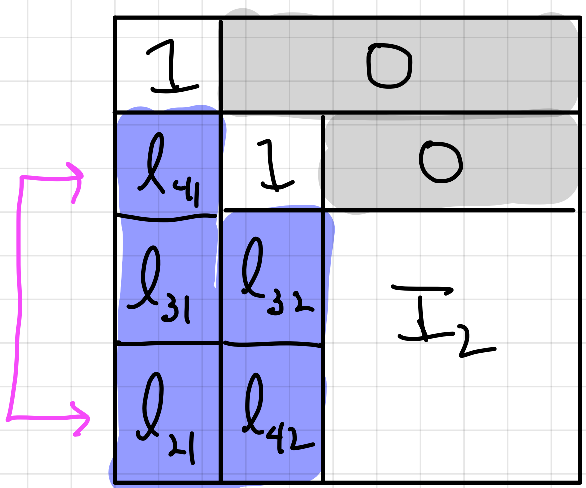
Record Permutations.

$$P = [3\ 2\ 1\ 4]$$

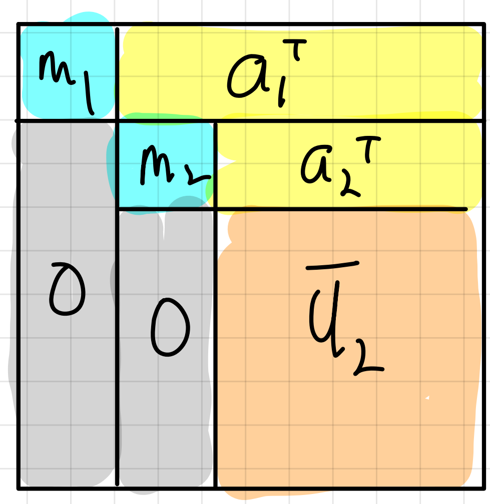
LU Decomposition with Partial Pivoting



m_2 = largest entry (in magnitude) in rows 2:4, column 2.



$= l_2 = \frac{c_2}{m_2}$



U_2

swap rows 2 and 4 in column 1.

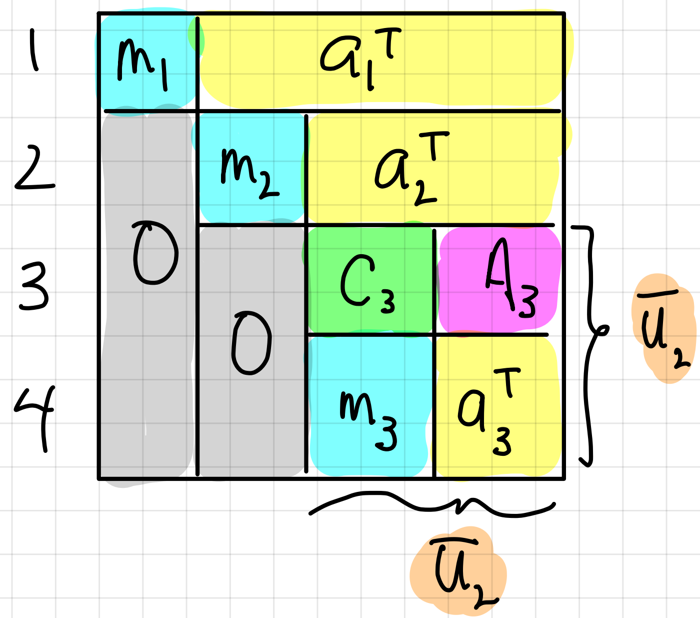
$$P_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

swap rows 2 and 4

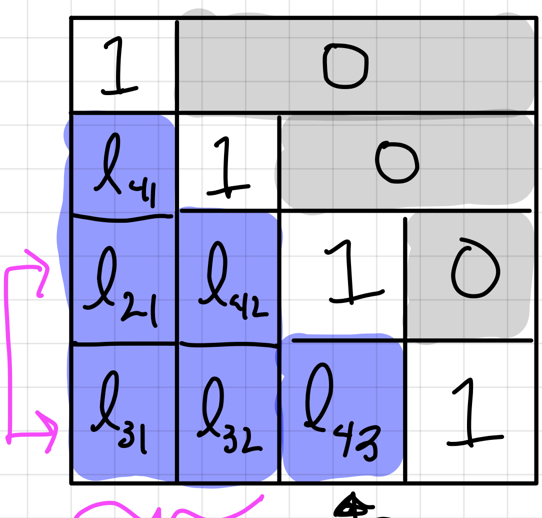
$$P_2^v = [3\ 4\ 1\ 2]$$

(vector version)

LU Decomposition with Partial Pivoting

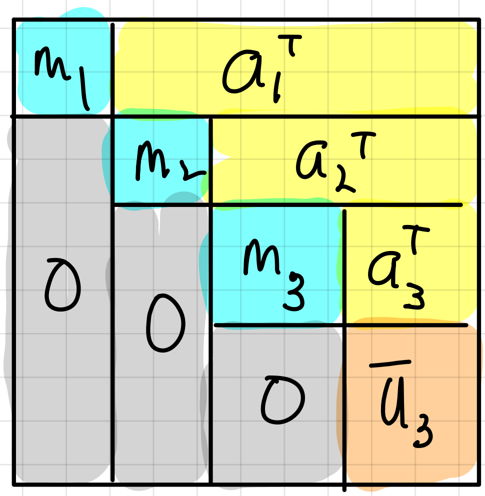


m_3 = largest entry in mag. in rows 3:4 of column 3



swap rows 3 and 4 in columns 1 and 2.

$$l_{43} = \frac{c_3}{m_3}$$



U_3

$$P = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

swap rows 2 and 4

$$P_2^v = [3\ 4\ 2\ 1]$$

(vector version)

LU decomposition with Partial Pivoting

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \ell_{41} & 1 & 0 & 0 \\ \ell_{21} & \ell_{42} & 1 & 0 \\ \ell_{31} & \ell_{32} & \ell_{43} & 1 \end{bmatrix} \qquad U = \begin{bmatrix} m_1 & & a_1^T & \\ 0 & m_2 & & a_2^T \\ 0 & 0 & m_3 & a_3^T \\ 0 & 0 & 0 & m_4 \end{bmatrix}$$

note order of multipliers

$$P = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$ matrix form of permutations

$$\vec{P}_2 = [3\ 4\ 2\ 1]$$ vector form

After partial pivoting, our LU decomposition satisfies:

$$\boxed{PA = LU}$$

Observation about L in LU when using partial pivoting.

Since we always choose the largest pivot in the column, we always have

$$|l_{ij}| \leq 1 , \quad i \geq j$$

This has important implications for the stability of the LU decomposition and limits the growth of errors when using L and U in forward and backward solves.

Partial Pivoting code – Find the bugs!

```
function [L,U,P,pv] = lu_bug_pp(A)

N = size(A,1);

U = A;
L = eye(N);     % Initialize using identity matrix
P = eye(N);
pv = 1:N;

% Decomposition
for k = 1:N-1
    % Find largest pivot in the columnx
    [m,p] = max(abs(U(k:end,k)));
    U([k,p],:) = U([p,k],:);   % Swap rows
    L([k,p],1:k) = L([p,k],1:k);

    % Store permutations
    pv([k,p]) = pv([p,k]);

    % Get multiplier, vectors and submatrix
    % m  = U(k,k);              % Multiplier
    ck = U(k+1:end,k);         % column vector
    ak = U(k,k+1:end)';        % Use transpose to get a column vector
    Ak = U(k+1:end,k+1:end);   % Submatrix

    % Update L
    lk = ck/m;
    L(k+1:end,k) = lk;

    % Update U
    U(k+1:end,k) = 0;                      % Zero out variables
    U(k+1:end,k+1:end) = Ak - lk*ak';      % Outer product used
end
P = P(pv,:);
end
```

Homework:

Fix 3 bugs in this code so that $LU = PA$.

Solving using the LU Decomposition

Once we have L and U, we can solve linear systems $Ax = b$.

Assuming we have L, U (obtained without pivoting) so that $A = LU$. Then

$$Ax = b$$

$$LUx = b$$

Solve $Ly=b$ using forward substitution

$$\underset{L}{\begin{bmatrix} 1 & 0 & & \\ \tilde{\ell}_1^T & 1 & 0 & \\ \tilde{\ell}_2^T & & 1 & 0 \\ \tilde{\ell}_3^T & & & 1 \end{bmatrix}} \underset{y}{\begin{bmatrix} y_1 \\ \\ \\ \\ \end{bmatrix}} = \underset{b}{\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}}$$

$$y_1 = b_1$$

$\tilde{y}_2$

$\tilde{\ell}_2 = 1 \times 1$ matrix

$$\underset{L}{\begin{bmatrix} 1 & 0 & & \\ \tilde{\ell}_2^T & 1 & 0 & \\ \tilde{\ell}_3^T & & 1 & 0 \\ \tilde{\ell}_4^T & & & 1 \end{bmatrix}} \underset{y}{\begin{bmatrix} y_1 \\ y_2 \\ \\ \\ \end{bmatrix}} = \underset{b}{\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}}$$

$$\tilde{\ell}_2^T \tilde{y}_1 + y_2 = b_2$$

$$\Longrightarrow \quad y_2 = b_2 - \tilde{\ell}_2^T \tilde{y}_2$$

$\tilde{y}_3$ and $\tilde{\ell}_3$ are 2×1

$\tilde{y}_3$

$$\underbrace{\begin{bmatrix} 1 & 0 & & \\ \tilde{\ell}_2^T & & 1 & 0 \\ \tilde{\ell}_3^T & & 1 & 0 \\ \tilde{\ell}_4^T & & & 1 \end{bmatrix}}_{L} \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \end{bmatrix}}_{y} = \underbrace{\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}}_{b}$$

$$y_3 = b_3 - \underbrace{\tilde{\ell}_3^T \tilde{y}_3}$$

dot product or "inner product"

$\tilde{y}_4$

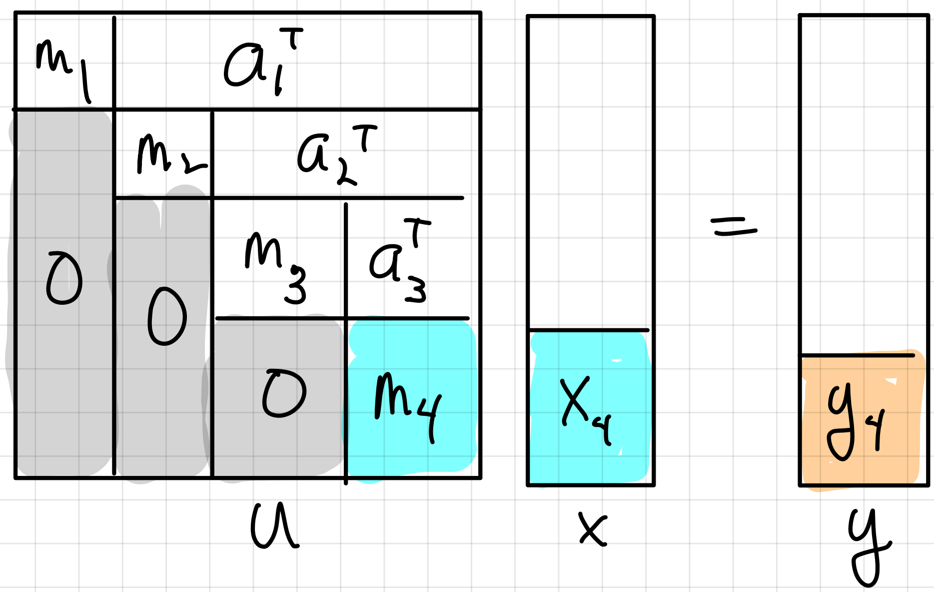
$\tilde{y}_4$ and $\tilde{\ell}_3$ are 3×1

$$\underbrace{\begin{bmatrix} 1 & 0 & & \\ \tilde{\ell}_2^T & & 1 & 0 \\ \tilde{\ell}_3^T & & 1 & 0 \\ \tilde{\ell}_4^T & & & 1 \end{bmatrix}}_{L} \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}}_{y} = \underbrace{\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}}_{b}$$

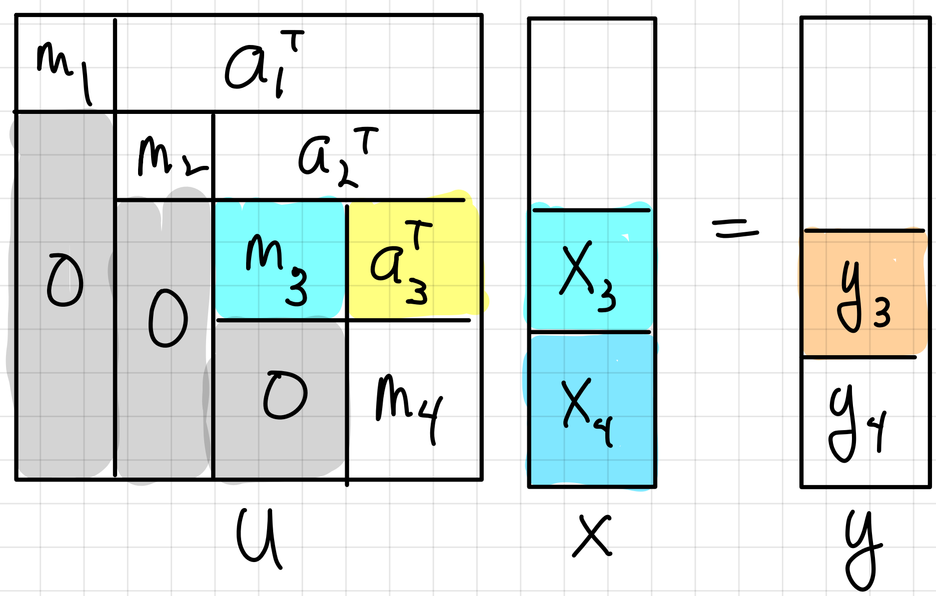
$$y_4 = b_4 - \underbrace{\tilde{\ell}_4^T \tilde{y}_4}$$

dot product or "inner product"

Solve $Ux = y$ using back substitution

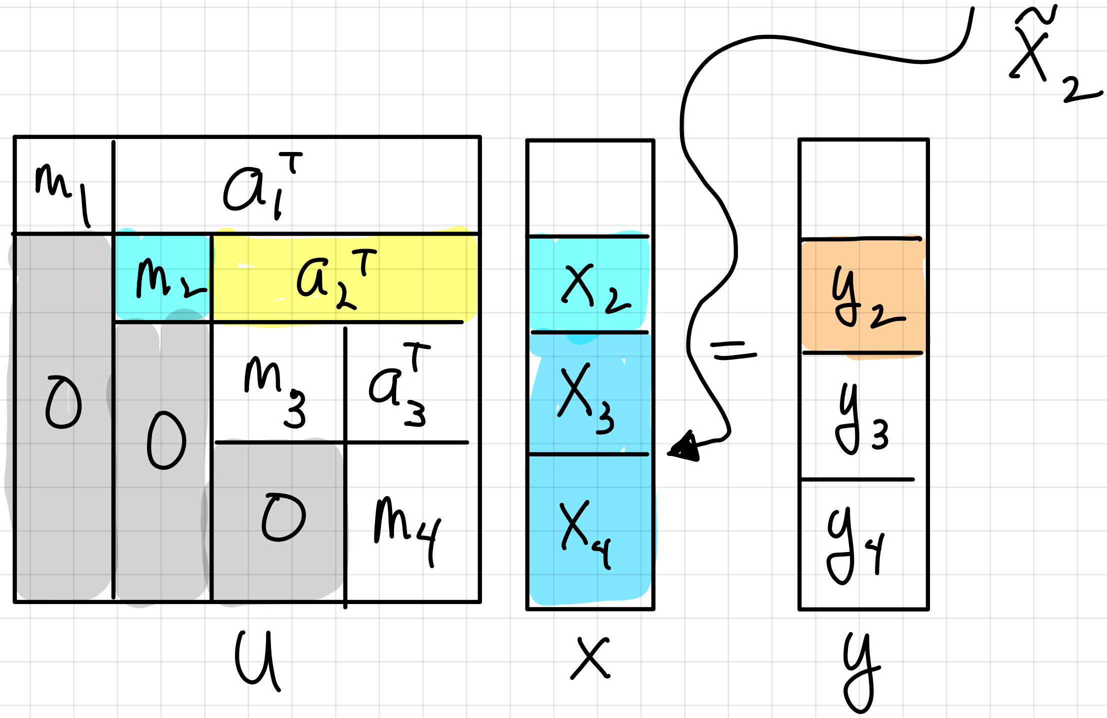


$$\Rightarrow \quad x_4 = (y_4)/m_4$$



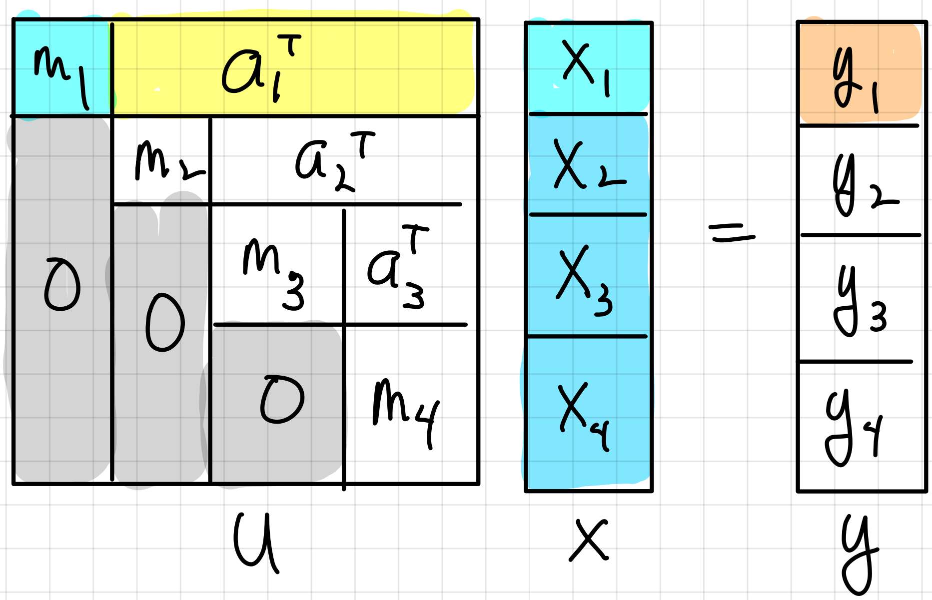
$$m_3 x_3 + a_3^T \tilde{x}_3 = y_3$$

$$\Rightarrow \quad x_3 = \left(y_3 - a_3^T \tilde{x}_3\right)/m_3$$



$$m_2 x_2 + a_2^T \tilde{x}_2 = y_2$$

$$\Rightarrow x_2 = (y_2 - a_2^T \tilde{x}_2)/m_2$$



$$m_1 x_1 + a_1^T \tilde{x}_1 = y_1$$

$$\Rightarrow x_1 = (y_1 - a_1^T \tilde{x}_1)/m_1$$

Forward and backward substitution

```
function x = solve_math(L,U,b)

N = size(L,1);

y = zeros(N,1);

% Forward Solve
for i = 1:N
    lk = L(i,1:i-1)';
    yk = y(1:i-1);
    y(i) = b(i) - lk'*yk;
end

% Backward Solve
x = zeros(N,1);
for i = N:-1:1
    m = U(i,i);
    xk = x(i+1:end);
    ak = U(i,i+1:end)';
    x(i) = (y(i) - ak'*xk)/m;
end

end
```

Homework : modify this to solve with partial pivoting.

$$PA = LU$$