Lecture 3: Solving linear systems of equations

Reading:  Ch. 1, 2.1-2.2  Ch. 2-3 (skip 3.8)

Outline: Systems of linear equations

- Examples, non-examples
- General problem, correspondence to matrices
- Solving by Gaussian elimination
  - complexity analysis
- Geometry & stability of solutions
- Homogeneous & non-homogeneous systems
- Solving with a computer: Matlab/Mathematica
  - timing
  - sparse matrices
  - 1D and 2D lattices
- When is there a solution?
- Solving repeatedly: LU decomposition
- Compressed sensing

Asymptotic complexity of matrix multiplication

http://en.wikipedia.org/wiki/Computational_complexity_of_mathematical_operations#Matrix_algebra

Matrix-vector multiplication

$$\begin{array}{cc} & n \\ m & \boxed{A} \end{array}$$

$$(A\vec{v})_i = \sum_{j=1}^{n} A_{ij} v_j$$

m entries to calculate

each term takes O(n) time

$\Rightarrow \boxed{O(m \cdot n)}$ time total

eg. $O(n^2)$ if $m=n$

$\Rightarrow$ doubling n, time increases by $\sim \times 4$

Matlab

```
>> n = 10000;
>> A = rand(n,n); v = rand(n,1);
>> start = cputime; for i = 1:10 A * v; end; cputime - start

ans =

    1.7813

>> n = 20000;
>> A = rand(n,n); v = rand(n,1);
>> start = cputime; for i = 1:10 A * v; end; cputime - start

ans =

    6.7969
```

×3.82

Sparse matrix-vector mult.

$$(A\vec{v})_i = \sum_{j=1}^{n} A_{ij} v_j$$

each term takes

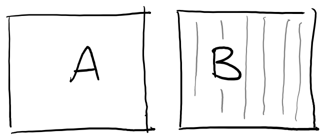
$O(\#$ of nonzero entries in row $i)$

$$\Rightarrow \boxed{O(\# \text{ nonzero entries in } A)}$$

×2.0

```
>> n = 100000; s = 10*n;
>> i = randi(n,s,1); j = randi(n,s,1); d = rand(s,1);
>> A = sparse(i,j,d,n,n); v = rand(n,1);
>> t = cputime; for i=1:10 A*v; end; cputime-t

ans =

  0.0625

>> n = 200000; s = 10*n;
>> i = randi(n,s,1); j = randi(n,s,1); d = rand(s,1);
>> A = sparse(i,j,d,n,n); v = rand(n,1);
>> t = cputime; for i=1:10 A*v; end; cputime-t

ans =

  0.1250
```

Matrix-matrix multiplication



$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

n^2 terms to calculate

each term takes $O(n)$ time

$$\Rightarrow \boxed{O(n^3)}$$

But Matlab is faster! (see HW 1)

Strassen's algorithm

http://en.wikipedia.org/wiki/Strassen_algorithm

$$A = \left(\begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array}\right) \begin{array}{l} n/2 \\ n/2 \end{array} \qquad B = \left(\begin{array}{c|c} B_{11} & B_{12} \\ \hline B_{21} & B_{22} \end{array}\right)$$

$$\begin{array}{cc} n/2 & n/2 \end{array}$$

$$AB = \left(\begin{array}{c|c} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ \hline A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{array}\right)$$

8 $n/2 \times n/2$ matrix multiplications

time $T(n) = 8T(n/2) + O(n^2)$

$$\sim n^2 + 8\left(\frac{n}{2}\right)^2 + 64\left(\frac{n}{4}\right)^2 + \cdots$$

log n levels

$$\sim n^2\left(1 + 2 + 4 + 8 + \cdots + 2^{\log_2 n}\right)$$

$$= \Theta(n^3)$$

Try computing

$M_1 = (A_{11} + A_{22})(B_{11} + B_{22})$

$M_2 = (A_{21} + A_{22})B_{11}$

$M_3 = A_{11}(B_{12} - B_{22})$

$M_4 = A_{22}(B_{21} - B_{11})$

$M_5 = (A_{11} + A_{12})B_{22}$

$M_6 = (A_{21} - A_{11})(B_{11} + B_{12})$

$M_7 = (A_{12} - A_{22})(B_{21} + B_{22})$

$$\Rightarrow AB = \begin{pmatrix} M_1 + M_4 - M_5 + M_7 & M_3 + M_5 \\ M_2 + M_4 & M_1 - M_2 + M_3 + M_6 \end{pmatrix}$$

$$M_2 + M_4 = (A_{21}B_{11} + \cancel{A_{22}B_{11}}) + (A_{22}B_{21} - \cancel{A_{22}B_{11}}) \checkmark$$

7 $n/2 \times n/2$ matrix multiplications

time $T(n) = 7T(n/2) + O(n^2)$

$$\sim n^2\left(1 + 7\frac{1}{2^2} + 7^2\frac{1}{4^2} + \cdots + \left(\frac{7}{4}\right)^{\log_2 n}\right)$$

$$\sim n^{2 + \log_2(7/4)}$$

$$= O(n^{\log_2 7 = 2.81\ldots})$$

$O(n^{2.376})$ [Coppersmith-Winograd '90]

but the constant factor is impractical

Stothers 2010: 2.374

Williams 2012: 2.3728642

Le Gall 2014: 2.3728639

Question: Is the correct exponent 2?

Remark: The asymptotic complexity of solving systems of linear equations is the same as matrix multiplication.

LINEAR EQUATIONS

Examples:

$\{\ x = 3$ ✓

$\{\ 2x - y = 3$ ✓

$\begin{cases} 2x - y = 3 \\ x - y = 2 \\ x + y = 0 \end{cases}$ ✓

$\begin{cases} x^2 + xy = 2 \\ x + y = 3 \end{cases}$ ✗ quadratic equation

$\begin{cases} x + y + z = 0 \\ x + \cos(y) + z = 2 \end{cases}$ ✗ transcendental equation

General problem:

Given: Linear equations

$$\text{equations}\begin{cases} \overbrace{a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n}^{n \text{ unknowns } x_1, \ldots, x_n} = b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \cdots + a_{2n}x_n = b_2 \\ \vdots \qquad\qquad\qquad\qquad \vdots \end{cases}$$

$$\underbrace{\begin{cases} a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \cdots + a_{2n}x_n = b_2 \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + a_{m3}x_3 + \cdots + a_{mn}x_n = b_m \end{cases}}_{m \text{ equations}}$$

$$\Updownarrow$$

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & & \cdots & a_{2n} \\ \vdots & & & & \vdots \\ a_{m1} & a_{m2} & & \cdots & a_{mn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}$$

$$\Updownarrow$$

$$\underset{m\times n}{A}\ \underset{n\times 1}{\vec{x}} = \underset{m\times 1}{\vec{b}}$$

Goal: Solve for $\vec{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$

(- If there isn't a solution, then say so.
- If there are multiple solutions, find them all!)

Note: We will see later that there is always either
* one unique solution,
* no solution, or
* infinitely many solutions

Example: Solve

$$\begin{aligned} 2x - y &= 3 \\ x - y &= 2 \\ x + y &= 0 \end{aligned} \iff \begin{pmatrix} 2 & -1 \\ 1 & -1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 3 \\ 2 \\ 0 \end{pmatrix}$$

shorthand notation:

$$\left(\begin{array}{cc|c} 2 & -1 & 3 \\ 1 & -1 & 2 \\ 1 & 1 & 0 \end{array}\right) \longrightarrow \left(\begin{array}{cc|c} 2 & -1 & 3 \\ 1 & -1 & 2 \\ 2 & 0 & 2 \end{array}\right)$$

Answer:
Adding $\begin{cases} x - y = 2 \\ x + y = 0 \end{cases} \Rightarrow 2x = 2 \Rightarrow x = 1 \Rightarrow y = -1$

$$\boxed{\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}}$$

Check your answer!

Key idea: This works by cancelling the coefficient of y.

Matlab

```
>> A = [2 -1; 1 -1; 1 1];
>> b = [3 2 0]';
>> x = A \ b

x =
```

Check:

Python

```
import numpy as np

A = [[2,-1],[1,-1],[1,1]]
b = [3,2,0]
x = np.linalg.solve(A[:2],b[:2])
print(x)
```

← using only

```
>> x = A \ b

x =

    1.0000
   -1.0000
```

Check:

```
>> A * x - b

ans =

   1.0e-15 *

         0
         0
   -0.2220
```

```
A = [[2,-1],[1,-1],[1,1]]
b = [3,2,0]
x = np.linalg.solve(A[:2],b[:2])
print(x)
print(np.dot(A,x) - b)   #check

[ 1. -1.]
[0. 0. 0.]
```

← using only first two equations

or:

```
x = np.linalg.lstsq(A, b, rcond=None)[0]
x

array([ 1., -1.])
```

Example: Polynomial interpolation

2 points determine a line

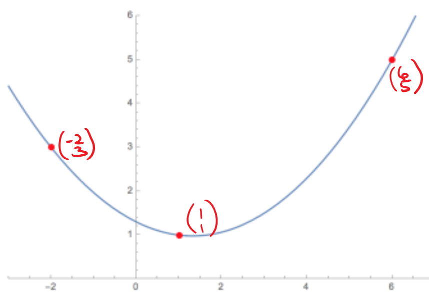
3 points determine a quadratic curve (parabola)

⋮

any $n+1$ points $(x_1,y_1),\dots,(x_{n+1},y_{n+1})$ determine a polynomial of degree $\leq n$.

Find the equation $y=ax^2+bx+c$ for the parabola that goes through

$(1,1)$
$(6,5)$
$(-2,3)$



Answer: Set up three equations:

$(1,1)$: $1 = a\cdot 1^2 + b\cdot 1 + c = a+b+c$

$(6,5)$: $5 = a\cdot 6^2 + b\cdot 6 + c = 36a+6b+c$ (-36)

$(-2,3)$: $3 = a\cdot(-2)^2 + b(-2)+c = 4a-2b+c$ (-4)

$$\left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 36 & 6 & 1 & 5 \\ 4 & -2 & 1 & 3 \end{array}\right) \xrightarrow{-36,\ -4} \left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 0 & -30 & -35 & -31 \\ 0 & -6 & -3 & -1 \end{array}\right) \xrightarrow{-5}$$

← this means $-6b-3c=-1$

$$\rightarrow \left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 0 & 6 & 3 & 1 \\ 0 & 0 & 20 & 26 \end{array}\right)$$

$\Downarrow$

$c=\frac{13}{10}$

Now work backward:

$$\left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 0 & 6 & 3 & 1 \\ 0 & 0 & 1 & \frac{13}{10} \end{array}\right) \xrightarrow{-1,\ -3} \left(\begin{array}{ccc|c} 1 & 1 & 0 & -3/10 \\ 0 & 6 & 0 & -29/10 \\ 0 & 0 & 1 & 13/10 \end{array}\right)$$

$$\rightarrow \left(\begin{array}{ccc|c} 1 & 1 & 0 & -3/10 \\ 0 & 1 & 0 & -29/60 \\ 0 & 0 & 1 & 13/10 \end{array}\right) \xrightarrow{-1} \left(\begin{array}{ccc|c} 1 & 0 & 0 & 11/60 \\ 0 & 1 & 0 & -29/60 \\ 0 & 0 & 1 & 13/10 \end{array}\right)$$

$$\rightarrow \left(\begin{array}{ccc|c} 0 & 1 & 0 & -29/60 \\ 0 & 0 & 1 & 13/10 \end{array}\right) \rightarrow \left(\begin{array}{ccc|c} 0 & 1 & 0 & -29/60 \\ 0 & 0 & 1 & 13/10 \end{array}\right)$$

$$\boxed{(a,b,c) = \left(\frac{11}{60}, -\frac{29}{60}, \frac{13}{10}\right)}$$

Check this!

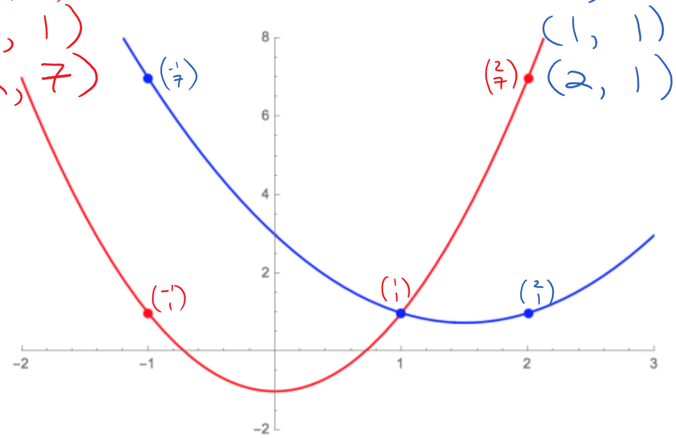
Matlab:

```
>> x = [1 6 -2]';
>> A = [x.^2 x ones(3,1)]

A =

     1     1     1
    36     6     1
     4    -2     1

>> y = [1 5 3]';
```

```
>> abc = A \ y

abc =

    0.1833
   -0.4833
    1.3000

>> format rat; abc

abc =

      11/60
     -29/60
      13/10
```

Example:

Find the parabola through $(-1, 1)$, $(1, 1)$, $(2, 7)$ AND Find the parabola through $(-1, 7)$, $(1, 1)$, $(2, 1)$



Answer: Set up the equations $x^2a + xa + 1 = y$

$$\begin{pmatrix} (-1)^2 & -1 & 1 \\ 1^2 & 1 & 1 \\ 2^2 & 2 & 1 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 7 \end{pmatrix} \qquad \begin{pmatrix} (-1)^2 & -1 & 1 \\ 1^2 & 1 & 1 \\ 2^2 & 2 & 1 \end{pmatrix} \begin{pmatrix} a' \\ b' \\ c' \end{pmatrix} = \begin{pmatrix} 7 \\ 1 \\ 1 \end{pmatrix}$$

same matrix!

$$\Rightarrow \begin{pmatrix} 1 & -1 & 1 \\ 1 & 1 & 1 \\ 4 & 2 & 1 \end{pmatrix} \begin{pmatrix} a & a' \\ b & b' \\ c & c' \end{pmatrix} = \begin{pmatrix} 1 & 7 \\ 1 & 1 \\ 7 & 1 \end{pmatrix}$$

We can solve them together

$$\left(\begin{array}{ccc|c} 1 & -1 & 1 & 7 \end{array}\right) \qquad \left(\begin{array}{ccc|cc} 1 & -1 & 1 & 1 & 7 \end{array}\right)$$

We can solve them together

$$\left(\begin{array}{ccc|cc} 1 & -1 & 1 & 1 & 7 \\ 1 & 1 & 1 & 1 & 1 \\ 4 & 2 & 1 & 7 & 1 \end{array}\right) \begin{matrix} \\ -1 \\ -4 \end{matrix} \longrightarrow \left(\begin{array}{ccc|cc} 1 & -1 & 1 & 1 & 7 \\ 0 & 2 & 0 & 0 & -6 \\ 0 & 6 & -3 & 3 & -27 \end{array}\right)$$

$$\longrightarrow \left(\begin{array}{ccc|cc} 1 & -1 & 1 & 1 & 7 \\ 0 & 1 & 0 & 0 & -3 \\ 0 & 2 & -1 & 1 & -9 \end{array}\right) \begin{matrix} +1 \\ \\ -2 \end{matrix} \longrightarrow \left(\begin{array}{ccc|cc} 1 & 0 & 1 & 1 & 4 \\ 0 & 1 & 0 & 0 & -3 \\ 0 & 0 & 1 & -1 & 3 \end{array}\right) -1$$

$$\Rightarrow I \begin{pmatrix} a & a' \\ b & b' \\ c & c' \end{pmatrix} = \boxed{\begin{pmatrix} a & a' \\ b & b' \\ c & c' \end{pmatrix} = \begin{pmatrix} 2 & 1 \\ 0 & -3 \\ -1 & 3 \end{pmatrix}}$$

Matlab:

```
>> x = [-1 1 2]';
>> ys = [1 1 7; 7 1 1]';
>> A = [ x.^2 x x.^0 ]

A =

     1    -1     1
     1     1     1
     4     2     1

>> abcs = A \ ys

abcs =

     2     1
     0    -3
    -1     3
```

Observe: Given equations

$$A\vec{x} = \vec{y}$$
$$A\vec{x}' = \vec{y}'$$
$$A\vec{x}'' = \vec{y}''$$
$$\vdots$$

with the same A

they are equivalent to

$$A\begin{pmatrix} | & | & | \\ \vec{x} & \vec{x}' & \vec{x}'' \\ | & | & | \end{pmatrix} = \begin{pmatrix} | & | & | \\ \vec{y} & \vec{y}' & \vec{y}'' \\ | & | & | \end{pmatrix}$$

and can be solved together.

Example: Invert the matrix $A = \begin{pmatrix} 1 & 2 & 0 \\ 3 & 2 & 1 \\ 0 & 1 & 0 \end{pmatrix}$

Answer: A^{-1} satisfies

$AA^{-1} = I$, so we can use Gaussian elimination to solve

$$\left(\begin{array}{c|c} A & I \end{array}\right) \quad \ldots$$

Matlab:

```
>> inv([1 2 0; 3 2 1; 0 1 0])

ans =

    1.0000    0.0000   -2.0000
         0         0    1.0000
   -3.0000    1.0000    4.0000
```

Python:

```
A = [[1,2,0],[3,2,1],[0,1,0]]
np.linalg.inv(A)

array([[ 1.,  0., -2.],
       [ 0.,  0.,  1.],
       [-3.,  1.,  4.]])
```

Solving linear equations on your computer

① Matlab/Octave:

```
>> A = [1 1 1; 36 6 1; 4 -2 1]
```

② Python

```
>>> import numpy as np
>>> A = np.array([[1,1,1],[36,6,1],[4,-2,1]])
```

① Matlab/Octave:

```
>> A = [1 1 1; 36 6 1; 4 -2 1]

A =

     1     1     1
    36     6     1
     4    -2     1

>> b = [1; 5; 3]

b =

     1
     5
     3

>> x = A \ b

x =

    0.1833
   -0.4833
    1.3000

>> A*x - b

ans =

   1.0e-15 *

         0
    0.8882
    0.4441
```

(see also "format rat")

You still need to check your answer!

(what if there's no solution?)

② Python

```
>>> import numpy as np
>>> A = np.array([[1,1,1],[36,6,1],[4,-2,1]])
>>> b = np.array([1,5,3])
>>> x = np.linalg.solve(A,b)
>>> x
array([ 0.18333333, -0.48333333,  1.3       ])
>>> np.dot(A,x) - b
array([  0.00000000e+00,   0.00000000e+00,   4.44089210e-16])

>>> n = 800; A = np.random.randn(n,n); b = np.dot(A, np.random.randn(n));
>>> start = time.time(); x = np.linalg.solve(A,b); end = time.time(); end-start
0.029742002487182617
>>> n = 1600; A = np.random.randn(n,n); b = np.dot(A, np.random.randn(n));
>>> start = time.time(); x = np.linalg.solve(A,b); end = time.time(); end-start
0.17376208305358887
>>> n = 3200; A = np.random.randn(n,n); b = np.dot(A, np.random.randn(n));
>>> start = time.time(); x = np.linalg.solve(A,b); end = time.time(); end-start
1.0949180126190186
>>> n = 6400; A = np.random.randn(n,n); b = np.dot(A, np.random.randn(n));
>>> start = time.time(); x = np.linalg.solve(A,b); end = time.time(); end-start
7.7281270027160645
```

③ Mathematica:

```
A = {{1, 1, 1}, {36, 6, 1}, {4, -2, 1}};
b = {1, 5, 3};
LinearSolve[A, b]
```

$\left\{\frac{11}{60}, -\frac{29}{60}, \frac{13}{10}\right\}$

Gaussian elimination

Observe:

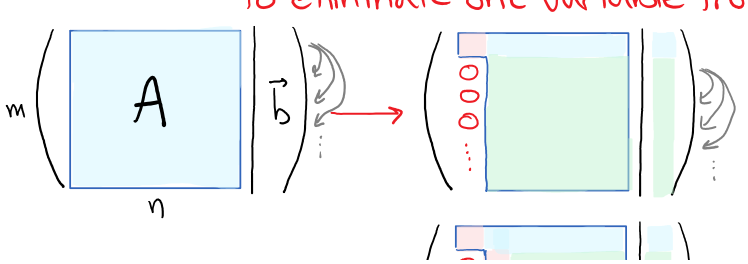
1) Multiplying an equation by $c \neq 0$
2) Adding one equation to another

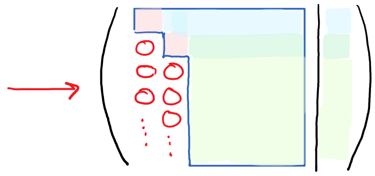
$\Big\}$ doesn't change the solutions

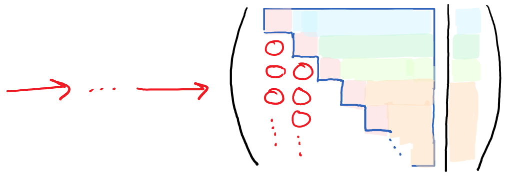
($\vec{x}$ solves eq. 1 and eq. 2 $\iff$ $\vec{x}$ solves eq. 1 and eq. 1 + eq. 2)

Therefore: Repeat:

Add multiples of one equation to the others to eliminate one variable from them.



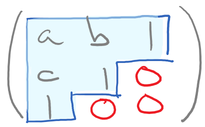


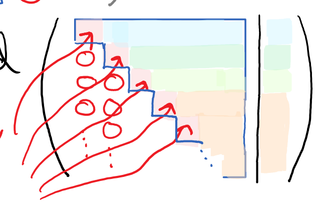


Notes: - You might have to rearrange the rows/equations

e.g., $\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \rightarrow \begin{pmatrix} 1 & & 0 \\ & 1 & \\ 0 & & 1 \end{pmatrix}$

- Reordering the columns/variables can be helpful

e.g. 

- These are called 

"pivots"

$$
\begin{aligned}
a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n &= b_1 \\
a_{22}'x_2 + a_{23}'x_3 + \cdots + a_{2n}'x_n &= b_2' \\
a_{33}'x_3 + \cdots + a_{3n}'x_n &= b_3' \\
\ddots \qquad &\vdots \\
a_{nn}'x_n &= b_n'
\end{aligned}
$$

Now solve for the last variable, backsubstitute.

last equation $\Rightarrow x_n = \dfrac{b_n'}{a_{nn}'}$

substitute into eq. $n-1$ $\Rightarrow a_{n-1,n-1}'x_{n-1} = b_{n-1}' - a_{n-1,n}' \dfrac{b_n'}{a_{nn}'}$

Keep on substituting backwards, solving for variables $x_n, x_{n-1}, x_{n-2}, \ldots, x_2, x_1$

Complexity: How many basic operations are needed?

$\boxed{O(n^3)}$ Why?

$O(n)$ time to add a multiple of a row to another

$O(n)$ time to add a multiple of a row to another
$\times\ O(n)$ rows you add into
$\times\ O(n)$ pivots (rows you add from)

Then backsubstitution takes $O(n^2)$ steps. (Check this!)

$\therefore$ doubling $n \Rightarrow 4\times$ the memory (to store A)
$8\times$ the running time!

Examples:

① Use Gaussian elimination to solve

$$\begin{cases} -x_1 + 3x_2 - 2x_3 = 1 \\ -x_1 + 4x_2 - 3x_3 = 0 \\ -x_1 + 5x_2 - 4x_3 = 0 \end{cases}$$

Answer:

$$\begin{pmatrix} -1 & 3 & -2 \\ -1 & 4 & -3 \\ -1 & 5 & -4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

$$\updownarrow$$

pivot

$$\left(\begin{array}{ccc|c} -1 & 3 & -2 & 1 \\ -1 & 4 & -3 & 0 \\ -1 & 5 & -4 & 0 \end{array}\right) \xrightarrow{-1,\ -1} \left(\begin{array}{ccc|c} -1 & 3 & -2 & 1 \\ 0 & 1 & -1 & -1 \\ 0 & 2 & -2 & -1 \end{array}\right) -2$$

$$\longrightarrow \left(\begin{array}{ccc|c} -1 & 3 & -2 & 1 \\ 0 & 1 & -1 & -1 \\ 0 & 0 & 0 & 1 \end{array}\right)$$

Stuck! 3^{rd} equation is $0x_1 + 0x_2 + 0x_3 = 1$
$\Rightarrow$ no solution.

② Solve

$$\begin{aligned} 2x_1 + x_2 \quad\quad &= 5 \\ x_1 + 2x_2 + 3x_3 &= 4 \\ -2x_1 + 3x_2 \quad\quad &= 7 \end{aligned}$$

Answer:

$$\left(\begin{array}{ccc|c} 2 & 1 & 0 & 5 \\ 1 & 2 & 3 & 4 \\ -2 & 3 & 0 & 7 \end{array}\right)$$

It is easiest to start with x_3!

$$\left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 2 & 1 & 0 & 5 \\ -2 & 3 & 0 & 7 \end{array}\right) \xrightarrow{1} \left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 2 & 1 & 0 & 5 \\ 0 & 4 & 0 & 12 \end{array}\right)$$

Now x_1?

Next we backsubstitute

$$\left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 2 & 1 & 0 & 5 \\ 0 & 1 & 0 & 3 \end{array}\right) \xrightarrow{-2,\ -1} \left(\begin{array}{ccc|c} 1 & 0 & 3 & -2 \\ 2 & 0 & 0 & 2 \\ 0 & 1 & 0 & 3 \end{array}\right) \xrightarrow{-1/2} \left(\begin{array}{ccc|c} 0 & 0 & 3 & -3 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 3 \end{array}\right)$$

$$\Rightarrow \vec{x} = \boxed{\begin{pmatrix} 1 \\ 3 \\ -1 \end{pmatrix}}$$

Check the answer!

$$\begin{pmatrix} 2 & 1 & 0 \\ 1 & 2 & 3 \\ -2 & 3 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 3 \\ -1 \end{pmatrix} = \begin{pmatrix} 5 \\ 4 \\ 7 \end{pmatrix} \quad \checkmark$$

LU DECOMPOSITION OF A MATRIX

Theorem: [Turing 1948]

Any $m \times n$ matrix A can be factored as

$$\underbrace{\begin{pmatrix} A \end{pmatrix}}_{m \times n} = \underbrace{\begin{pmatrix} P \end{pmatrix}}_{m \times m} \underbrace{\begin{pmatrix} L & 0 \end{pmatrix}}_{m \times m} \underbrace{\begin{pmatrix} 0 & U \end{pmatrix}}_{m \times n}$$

permutation — lower triangular — upper triangular

$\underbrace{\begin{pmatrix} 0 & \end{pmatrix}}_{m \times n}$ if $m \geq n$

Why? It is just Gaussian elimination!

e.g.,

$$\begin{pmatrix} 3 & 2 & 1 \\ 3/2 & 2 & 3/2 \\ -1 & 0 & 1 \end{pmatrix} \xrightarrow{-\frac{1}{2},\ +\frac{1}{3}} \begin{pmatrix} 3 & 2 & 1 \\ 0 & 1 & 1 \\ 0 & 2/3 & 4/3 \end{pmatrix} \xrightarrow{-\frac{2}{3}} \begin{pmatrix} 3 & 2 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 2/3 \end{pmatrix}$$

```
>> [L, U, P] = lu([3 2 1; 1.5 2 1.5; -1 0 1])

L =

    1.0000         0         0
    0.5000    1.0000         0
```

```
L =

    1.0000         0         0
    0.5000    1.0000         0
   -0.3333    0.6667    1.0000

U =

    3.0000    2.0000    1.0000
         0    1.0000    1.0000
         0         0    0.6667

P =

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

$\mathcal{U}$ is the result of Gaussian elimination

L and P keep track of the steps

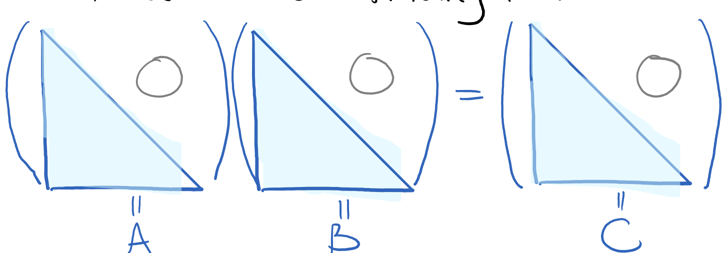
Proof sketch:

Claim 1: Multiplying on the left by

$$\begin{array}{c} \\ i \rightarrow \\ \\ \end{array}\begin{pmatrix} 1 & & & \bigcirc \\ \bigcirc & 1 & & \\ c & & 1 & \\ & \bigcirc & & 1 \end{pmatrix}, \quad c \text{ in column } \overset{\uparrow}{j}$$

adds c times row j to row i $(j<i)$

Claim 2: The product of lower-triangular matrices is also lower triangular.



Proof:

$A_{ik}=0$ if $i<k$

$B_{kj}=0$ if $k<j$

$$C_{ij}=\sum_{k=1}^{n} A_{ik}B_{kj}=\sum_{k=j}^{i} A_{ik}B_{kj}=0 \text{ if } i<j \quad \square$$

Matlab

```
>> rand('twister', 1)
>> A = rand(3, 5)

A =

    0.4170    0.3023    0.1863    0.5388    0.2045
    0.7203    0.1468    0.3456    0.4192    0.8781
    0.0001    0.0923    0.3968    0.6852    0.0274

>> [L, U, P] = lu(A);

>> P

P =

     0     0     1
     1     0     0
     0     1     0

>> L

L =

    1.0000         0         0
    0.5789    1.0000         0

>> U

U =

    0.7203    0.1468    0.3456    0.4192    0.8781
         0    0.2174   -0.0138    0.2961   -0.3039
         0         0    0.4026    0.5594    0.1563
```

Python

```python
import numpy as np

# doing it like this to get the same
# randoms as Matlab
np.random.seed(1)
A = np.random.rand(5,3).T
print(A)

from scipy.linalg import lu
lu(A)
```

```
[[4.17022005e-01 3.02332573e-01 1.86260211e-01 5.38816734e-01
  2.04452250e-01]
 [7.20324493e-01 1.46755891e-01 3.45560727e-01 4.19194514e-01
  8.78117436e-01]
 [1.14374817e-04 9.23385948e-02 3.96767474e-01 6.85219500e-01
  2.73875932e-02]]
```

```
(array([[0., 1., 0.],
        [1., 0., 0.],
        [0., 0., 1.]]),
 array([[1.00000000e+00, 0.00000000e+00, 0.00000000e+00],
        [5.78936311e-01, 1.00000000e+00, 0.00000000e+00],
        [1.58782352e-04, 4.24691460e-01, 1.00000000e+00]]),
 array([[ 0.72032449,  0.14675589,  0.34556073,  0.41919451,  0.87811744],
        [ 0.        ,  0.21737026, -0.01379744,  0.29612981, -0.30392182],
        [ 0.        ,  0.        ,  0.40257226,  0.55938914,  0.15632116]]))
```

$$\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1.0000 & 0 & 0 \\ 0.5789 & 1.0000 & 0 \\ 0.0002 & 0.4247 & 1.0000 \end{pmatrix} \begin{pmatrix} 0.7203 & 0.1468 & 0.3456 & 0.4192 & 0.8781 \\ 0 & 0.2174 & -0.0138 & 0.2961 & -0.3039 \\ 0 & 0 & 0.4026 & 0.5594 & 0.1563 \end{pmatrix}$$

```
>> P' * L * U - A

ans =
```

note the transpose!

```
   1.0e-16 *

        0         0         0         0   -0.5551
        0         0         0         0         0
        0         0         0         0         0
```

How to solve the SAME system of linear equations repeatedly (with different right-hand sides)

Problem: How can we solve the equations faster?

Given a matrix A,

and vectors $\vec{b}$, $\vec{b}'$, $\vec{b}''$, ...,

Goal: Solve quickly

$$A\vec{x} = \vec{b}$$
$$A\vec{x}' = \vec{b}'$$
$$A\vec{x}'' = \vec{b}''$$
$$\vdots$$

same matrix A

different vectors $\vec{b}$

Idea: Combine them, then use Gaussian elim.

$$A\begin{pmatrix} | & | & | \\ \vec{x} & \vec{x}' & \vec{x}'' \\ | & | & | \end{pmatrix} = \begin{pmatrix} | & | & | \\ \vec{b} & \vec{b}' & \vec{b}'' \\ | & | & | \end{pmatrix}$$

This only works if we know all $\vec{b}$s in advance

Bad idea: Precompute A^{-1} (if it exists).

Then just multiply

$\vec{x} = A^{-1}\vec{b}$, $\vec{x}' = A^{-1}\vec{b}'$, $\vec{x}'' = A^{-1}\vec{b}''$, ...

Don't compute inverses!

Slow

Numerically unstable

(sparse matrix)$^{-1}$ can be dense

Application: Quickly solving multiple sets of equations with the same A

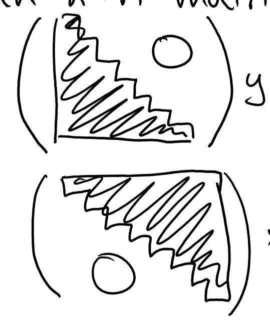
Goal: Solve $A\vec{x} = \vec{b}$, with $A = PL\mathcal{U}$

⓪ Precompute $P, L, \mathcal{U}$ — $O(n^3)$

① Let $\vec{y} = \mathcal{U}\vec{x}$. Solve $L\vec{y} = P^{-1}\vec{b}$ for $\vec{y}$.

② Solve $\mathcal{U}\vec{x} = \vec{y}$ for $\vec{x}$.

For an $n \times n$ matrix, this is fast! $O(n^2)$



$y = P^{-1}b$ by forward substitution (get y_1, substitute, $y_2, y_3, \ldots$)

$x = y$ by back substitution (get $x_n, x_{n-1}, x_{n-2}, \ldots$)

$\therefore$ Precomputing the $L\mathcal{U}$ decomposition of A allows for solving $Ax = b$, $Ax' = b'$, $Ax'' = b''$, … faster.

(In fact, if A is tridiagonal, L and $\mathcal{U}$ will be sparse while A^{-1} can be dense $\longrightarrow$ even given A^{-1} for free, it is faster to solve $Ax = b$ using the $L\mathcal{U}$ decomposition than to multiply by A^{-1}!)

More interesting Gaussian elimination examples

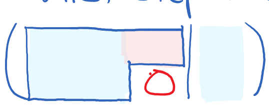
Example: Solve

$$x + 3y - z = 4$$
$$3x + 10y = 8$$

Answer:

$$\left(\begin{array}{ccc|c} 1 & 3 & -1 & 4 \\ 3 & 10 & 0 & 8 \end{array}\right)$$

The first step is already done!



Now backsubstitute.

$$\left(\begin{array}{ccc|c} 1 & 3 & -1 & 4 \\ 3 & 10 & 0 & 8 \end{array}\right) \xrightarrow{-\frac{1}{3}} \left(\begin{array}{ccc|c} 0 & -\frac{1}{3} & -1 & \frac{4}{3} \\ 3 & 10 & 0 & 8 \end{array}\right)$$

$$\Longrightarrow x = \frac{8}{3} - \frac{10}{3}y$$

$z = -\frac{4}{3} - \frac{1}{3}y$ where y is arbitrary!

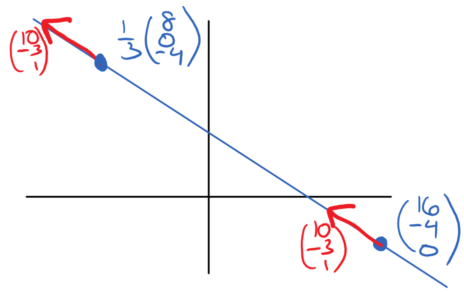
$$\Rightarrow \boxed{\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \frac{1}{3}\begin{pmatrix} 8 \\ 0 \\ -4 \end{pmatrix} + \begin{pmatrix} -\frac{10}{3} \\ 1 \\ -\frac{1}{3} \end{pmatrix} y, \quad y \in \mathbb{R}}$$

"free variable"

or $$\boxed{\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \frac{1}{3}\begin{pmatrix} 8 \\ 0 \\ -4 \end{pmatrix} + \begin{pmatrix} 10 \\ -3 \\ 1 \end{pmatrix} t, \quad t \in \mathbb{R}}$$ $t = -3y$

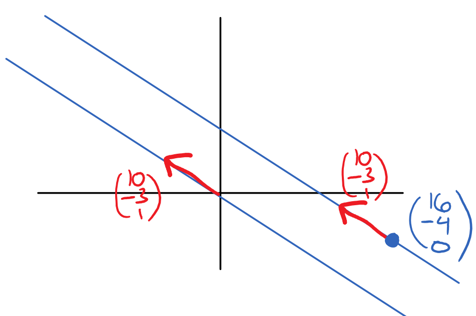
<u>Note</u>: You can also solve the equations with x or z free

e.g., $$\boxed{\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 16 \\ -4 \\ 0 \end{pmatrix} + \begin{pmatrix} 10 \\ -3 \\ 1 \end{pmatrix} z, \quad z \in \mathbb{R}}$$



<u>Observe</u>:

① The set $\{(10z, -3z, z) \mid z \in \mathbb{R}\}$ is a line through $\vec{0} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$.
Adding $(16, -4, 0)$ shifts this line.



② $\begin{pmatrix} 16 \\ -4 \\ 0 \end{pmatrix}$ solves $\begin{cases} x + 3y - z = 4 \\ 3x + 10y = 8 \end{cases}$

$\begin{pmatrix} 10 \\ -3 \\ 1 \end{pmatrix}$ solves $\begin{cases} x + 3y - z = 0 \\ 3x + 10y = 0 \end{cases}$ "homogeneous equations"

<u>Definition</u>: A system of equations $A\vec{x} = \vec{b}$ is
- **homogeneous** if $\vec{b} = \vec{0}$
- **non-homogeneous** if $\vec{b} \neq \vec{0}$

$\Rightarrow$ The general solution for a nonhomogeneous equation $A\vec{x} = \vec{b}$

$\Rightarrow$ The general solution for a nonhomogeneous equation $A\vec{x}=\vec{b}$ is a particular solution like $\begin{pmatrix}16\\-4\\0\end{pmatrix}$ or $\frac{1}{3}\begin{pmatrix}8\\0\\-4\end{pmatrix}$
plus the general solution to $A\vec{x}=\vec{0}$.
called the nullspace of A

This makes sense:
- If $A\vec{x}=\vec{b}$ and $A\vec{y}=\vec{b}$ $\Rightarrow A(\vec{x}-\vec{y})=\vec{b}-\vec{b}=\vec{0}$
- If $A\vec{x}=\vec{b}$ and $A\vec{y}=\vec{0}$ $\Rightarrow A(\vec{x}\pm\vec{y})=\vec{b}\pm\vec{0}=\vec{b}$

Example: Solve

$$\begin{cases} x_1+2x_2+2x_3+3x_4=0 \\ 2x_1+4x_2+x_3+3x_4=0 \\ 3x_1+6x_2+x_3+4x_4=0 \end{cases}$$

Note: Since the right-hand side is $\vec{0}$, $\vec{x}=\vec{0}$ is a solution.
Are there more solutions?

$$\left(\begin{array}{cccc|c} 1 & 2 & 2 & 3 & 0 \\ 2 & 4 & 1 & 3 & 0 \\ 3 & 6 & 1 & 4 & 0 \end{array}\right) \begin{array}{l} \\ -2 \\ -3 \end{array} \longrightarrow \left(\begin{array}{cccc|c} 1 & 2 & 2 & 3 & 0 \\ 0 & 0 & -3 & -3 & 0 \\ 0 & 0 & -5 & -5 & 0 \end{array}\right) -\tfrac{5}{3}$$

(Note: To minimize accumulation of floating point errors, it is often best to pivot on the largest magnitude entry in the column (3))

$$\rightarrow \left(\begin{array}{cccc|c} 1 & 2 & 2 & 3 & 0 \\ 0 & 0 & -3 & -3 & 0 \\ \cancel{0} & \cancel{0} & \cancel{0} & \cancel{0} & \cancel{0} \end{array}\right) \begin{array}{l} +1 \\ \\ \leftarrow 0=0 \end{array}$$

$$\rightarrow \left(\begin{array}{cccc|c} 1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -3 & -3 & 0 \end{array}\right)$$

$$x_1+2x_2-x_3=x_3+x_4=0$$
$$\Rightarrow x_3=-x_4$$
$$x_1=-2x_2-x_4$$

$$\Rightarrow \text{solutions} = \left\{(x_1,x_2,x_3,x_4)\in\mathbb{R}^4 : \begin{array}{l} x_1=-2x_2-x_4 \\ x_3=-x_4 \end{array}\right\}$$

x_2 & x_4 are <u>free</u> — can be arbitrary
— the solution set is a <u>2D plane</u> in $\mathbb{R}^4$

$$\begin{pmatrix}x_1\\x_2\\x_3\\x_4\end{pmatrix}=\begin{pmatrix}-2x_2-x_4\\x_2\\-x_4\\x_4\end{pmatrix}=x_2\begin{pmatrix}-2\\1\\0\\0\end{pmatrix}+x_4\begin{pmatrix}-1\\0\\-1\\1\end{pmatrix}$$

everything expressed

in the free variables

Example (continued): Solve

$$\begin{cases} x_1 + 2x_2 + 2x_3 + 3x_4 = 1 \\ 2x_1 + 4x_2 + x_3 + 3x_4 = -1 \\ 3x_1 + 6x_2 + x_3 + 4x_4 = \cancel{0}\ -2 \end{cases}$$

$$\underbrace{}_{\text{same } A}$$

$$\left(\begin{array}{cccc|c} 1 & 2 & 2 & 3 & 1 \\ 2 & 4 & 1 & 3 & -1 \\ 3 & 6 & 1 & 4 & \cancel{0}\ -2 \end{array}\right) \begin{matrix} -2 \\ -3 \end{matrix} \longrightarrow \left(\begin{array}{cccc|c} 1 & 2 & 2 & 3 & 1 \\ 0 & 0 & -3 & -3 & -3 \\ 0 & 0 & -5 & -5 & \cancel{-3}\ -5 \end{array}\right)$$

contradiction $\Rightarrow$ no solution!

$$x_1 + 2x_2 - x_3 = -2$$
$$x_3 + x_4 = 1$$

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} -2-2x_2+(1-x_4) \\ x_2 \\ 1-x_4 \\ x_4 \end{pmatrix} = \begin{pmatrix} -1 \\ 0 \\ 1 \\ 0 \end{pmatrix} + \underbrace{x_2 \begin{pmatrix} -2 \\ 1 \\ 0 \\ 0 \end{pmatrix} + x_4 \begin{pmatrix} -1 \\ 0 \\ -1 \\ 1 \end{pmatrix}}_{\text{this part is the same as for the homogeneous system}}$$

How to Solve Systems of Linear Equations

- ~~Inverting the matrix A (if it is invertible)~~
- Gaussian elimination
  - LU decomposition
- Iterative methods
  - preconditioners
  - conjugate gradient (for positive semidefinite A)
  - multigrid methods
- Methods for solving SDD systems (sparse diagonally dominant)

  ⋮

More goals: * Solve approximately, instead of exactly

* How good is the solution? (stability)
* Solve the same system multiple times (different r.h.s.s)
* Perturbed systems
* Systems with a special form
* Understand both how the algorithms work and how to use them! (and when to use them)

① Stability — very important

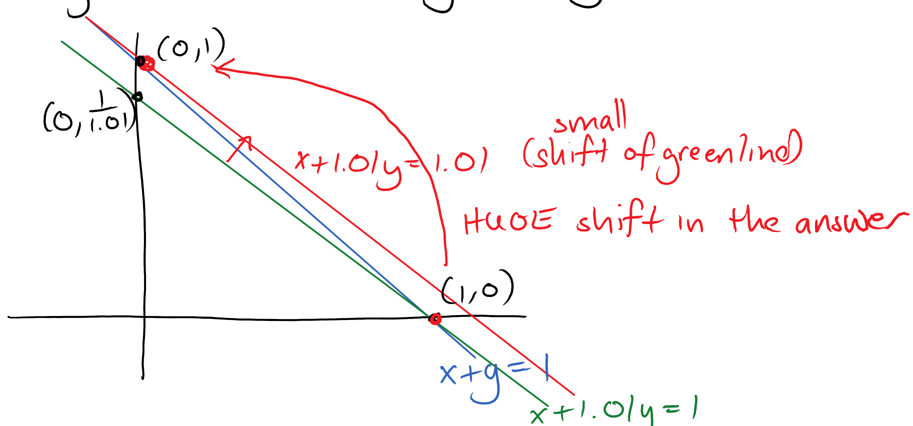
Example: Consider the equations

$$\begin{cases} x + y = 1 \\ x + 1.01y = 1 \end{cases} \Rightarrow (x,y) = (1,0)$$

$$\begin{cases} x + y = 1 \\ x + 1.01y = 1.01 \end{cases} \Rightarrow (x,y) = (0,1)$$

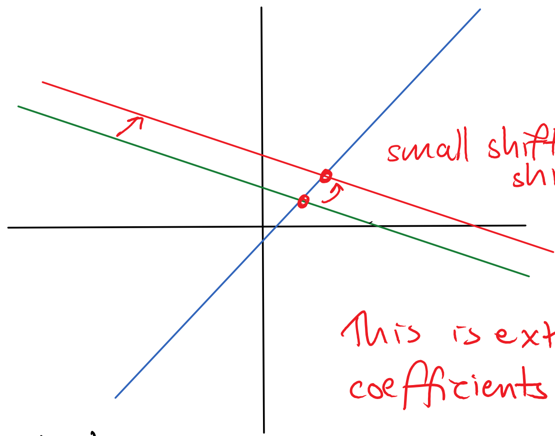
A small change to the equations led to a HUGE change in the solutions!

Why? Observe the geometry:



- because the lines are almost parallel

this wouldn't have happened if the lines were at a greater ∡



small shift of green line shifts intersection a small amount

This is extremely important when coefficients are given numerically.

<u>Note</u>: In <u>higher dimensions</u>, you have the <u>same problem</u> with planes that are nearly parallel.

But they don't have to be nearly parallel; it is more complicated than that. (Eg., think of above green and blue lines as intersections of planes in 3D with $\{z=0\}$ plane. They needn't be nearly parallel.)

We'll study this more later.

② Linear programming

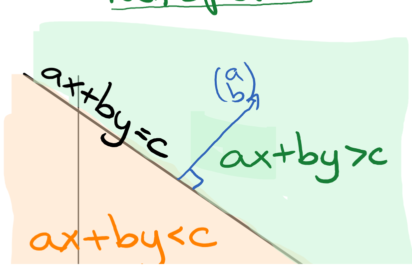
<u>Problem</u>: Solve a system of linear inequalities

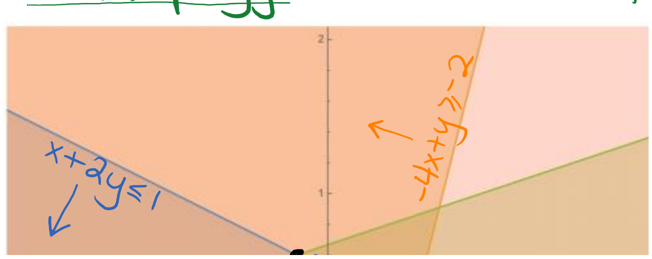
$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \leq b_1$$
$$a_{21}x_1 + a_{22}x_{2} + \cdots + a_{2n}x_n \leq b_2$$
$$\vdots$$

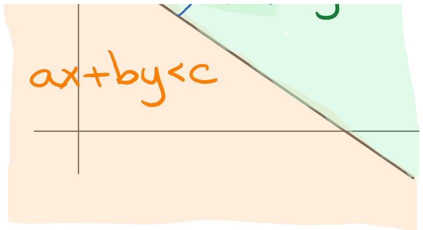
- Very useful in practice
  inequalities often express resource constraints
  eg. you have \$100
  rice is \$3/pound
  beans are \$5/pound
  $3r + 5b \leq 100$
- Generalizes linear systems of equations
  $ax+by \leq c$
  $ax+by \geq c$ $\iff$ $ax+by=c$

<u>Geometry</u>:

<u>Halfspaces</u>: <u>Convex polygon</u> = intersection of halfspaces







Another (less interesting) application: Computing Det(A)

$$\mathrm{Det}(A) = \mathrm{Det}(P)\cdot \mathrm{Det}(L)\cdot \mathrm{Det}(U)$$

Proof idea: Apply Gaussian elimination to A.

$$A = \begin{pmatrix} 2 & 2 & 2 \\ 4 & 7 & 7 \\ 6 & 18 & 22 \end{pmatrix} \quad -2$$

$$\rightarrow \underbrace{\begin{pmatrix} 2 & 2 & 2 \\ 0 & 3 & 3 \\ 6 & 18 & 22 \end{pmatrix}}_{= A_1} \; -3 \;\rightarrow\; \underbrace{\begin{pmatrix} 2 & 2 & 2 \\ 0 & 3 & 3 \\ 0 & 12 & 18 \end{pmatrix}}_{= A_2} \; -4$$

$$\longrightarrow \begin{pmatrix} 2 & 2 & 2 \\ 0 & 3 & 3 \\ 0 & 0 & 6 \end{pmatrix} \text{ done}$$

$\overset{\shortparallel}{A_1}$ $\overset{\shortparallel}{A_2}$ $\overset{\shortparallel}{A_3}$

Observe:

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} A_1$$

$$A_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 3 & 0 & 1 \end{pmatrix} A_2$$

$$A_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 4 & 1 \end{pmatrix} A_3$$

$$\Rightarrow A = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 3 & 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 4 & 1 \end{pmatrix}}_{\begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 4 & 1 \end{pmatrix} = L} \underbrace{\begin{pmatrix} 2 & 2 & 2 \\ 0 & 3 & 3 \\ 0 & 0 & 6 \end{pmatrix}}_{\shortparallel\ U\ \checkmark}$$

Why is P necessary?

$$A = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} a & 0 \\ b & c \end{pmatrix}\begin{pmatrix} d & e \\ 0 & f \end{pmatrix}$$

$$\Rightarrow 0 = a \cdot d$$

$\Rightarrow$ $\underline{a = 0}$ or $\underline{d = 0}$

$$\begin{pmatrix} 0 & 0 \\ b & c \end{pmatrix}\begin{pmatrix} d & e \\ 0 & f \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ u & u \end{pmatrix} \times \qquad \begin{pmatrix} a & 0 \\ b & c \end{pmatrix}\begin{pmatrix} 0 & e \\ 0 & f \end{pmatrix} = \begin{pmatrix} 0 & u \\ 0 & u \end{pmatrix} \times$$

Problem: You can't pivot on 0.
So switch the two rows with $P = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

$$PA = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} = \underset{L}{\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}} \underset{U}{\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}} \checkmark$$

Exercise ①: Find the equation $y = mx + b$ for the line through $(1,1)$ and $(6,5)$.

Answer: slope $m = \frac{5-1}{6-1} = \frac{4}{5}$
y-intercept $b = \frac{1}{5}$ (check this)

Example: Solve

$$\begin{aligned} x + 3y - z &= 4 \\ 3x + 10y &= 8 \end{aligned}$$

Answer: To eliminate x, subtract 3 times the first equation from the 2^{nd} one:

$$\Rightarrow \begin{cases} x + 3y - z = 4 \\ \quad\quad y + 3z = -4 \end{cases}$$

$\Rightarrow y = -4 - 3z$, where z is arbitrary

Substituting into the first equation,

$$\begin{aligned} x &= 4 - 3y + z \\ &= 4 - 3(-4 - 3z) + z \\ &= 16 + 10z \end{aligned}$$

$\Rightarrow$ Infinitely many solutions, all of the form

$$\boxed{x = 16 + 10z,\ y = -4 - 3z,\ z \in \mathbb{R} \text{ arbitrary}}$$

As a vector,

$$\boxed{\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \underbrace{\begin{pmatrix} 16 \\ -4 \\ 0 \end{pmatrix}}_{\text{constant coefficients}} + z\begin{pmatrix} 10 \\ -3 \\ 1 \end{pmatrix},\ \underbrace{z}_{\text{free variable}} \in \mathbb{R}}$$