Lecture 3: Solving linear systems of equations (class)

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

A is $m \times n$, $\vec{v} \in \mathbb{R}^n$, $(A\vec{v})_i = \sum_{j=1}^{n} A_{ij} v_j$

n multiplications, $n-1$ additions $\Rightarrow O(n) \approx c \cdot n$

$\Rightarrow O(m \cdot n)$ time to compute $A\vec{v}$

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 $i)$

$\Rightarrow O(\#$ nonzero entries in $A)$

```
>> 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
```

×2.0

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

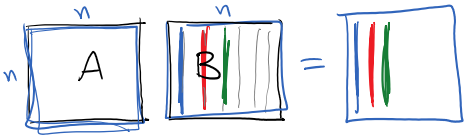
```
>> n = 200000; s = 10*n;
>> i = randi(n,s,1); j = randi(n,s,1); d = rand(s,1);
>> A = sparse(i,j,d,n,n); v = rand(n,1);
>> t = cputime; for i=1:10 A*v; end; cputime-t

ans =

    0.1250
```

×2.0

Matrix-matrix multiplication

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

terms to calculate — each term takes



n matrix-vector multips $= n \cdot O(n^2) = O(n^3)$

$(2n)^3 = 2^3 n^3 = 8n^3$

But Matlab is faster! (see HW 1)

Strassen's algorithm

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

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \quad B = \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}$$

$T(n)$ = time to multiply two $n \times n$ matrices

$$AB = \begin{pmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{pmatrix}$$

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

time $T(n) = 8T\left(\frac{n}{2}\right) + O(n^2)$

$$= c \cdot n^2 + 8\left(c\left(\frac{n}{2}\right)^2 + 8T\left(\frac{n}{4}\right)\right)$$

$$= c \cdot n^2 + c \cdot 8\left(\frac{n}{2}\right)^2 + 8^2\left(c \cdot \left(\frac{n}{4}\right)^2 + 8T\left(\frac{n}{8}\right)\right)$$

$$= c \cdot n^2 \cdot \left(1 + \frac{8}{2^2} + \frac{8^2}{4^2} + \cdots\right)$$

$$= c \cdot n^2 \cdot \left(1 + 2 + 2^2 + 2^3 + 2^4 + \cdots + 2^{\log_2 n}\right)$$

$$\approx 2c \cdot n^3 = O(n^3)$$

Try computing

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

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

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

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

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

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

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

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

$M_2 + M_4 = (A_{21}B_{11} + A_{22}B_{11}) + (A_{22}B_{21} - A_{22}B_{11})$ ✓

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

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

$$= n^2\left(7\frac{1}{2^2} + 7^2\frac{1}{2^4} + \frac{7^3}{2^6} + \cdots + \left(\frac{7}{4}\right)^{\log_2 n}\right)$$

$= n^{\log_2 7/4}$

$= n^2\left(1 + \frac{7}{2^2} + \frac{7^2}{2^4} + \frac{7^3}{2^6} + \cdots\right)$ $\quad (*) = n^{\log_2 7/4}$

$= n^{2+\log_2 \frac{7}{4}} = n^{\log_2 7} = \boxed{n^{2.81\ldots}}$

$O(n^{2.376})$ [Coppersmith-Winograd '90]
but the constant factor is impractical

Stothers 2010: 2.374
Williams 2012: 2.37328642
Le Gall 2014: 2.37286639 $\quad n^\omega$

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:

$\begin{cases} x = 3 \end{cases}$ ✓

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

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

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

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

$\left.\begin{aligned} x + 2\cos(y) &= 3 \\ 2x - \cos(y) &= 10 \end{aligned}\right\}$ linear in x and $\cos y$

let $z = \cos y$

$\Rightarrow \begin{cases} x + 2z = 3 \\ 2x - z = 10 \end{cases}$

General problem:

Given: Linear equations

n unknowns $x_1, \ldots, x_n$

m equations

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{m1}x_1 + a_{m2}x_2 + a_{m3}x_3 + \cdots + a_{mn}x_n &= b_m \end{aligned}$$

$$\Updownarrow$$

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

$$\Updownarrow$$

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

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

$$2x - y = 3$$
$$x - y = 2$$
$$x + y = 0$$

$$\Leftrightarrow \quad \begin{pmatrix} 2 & -1 \\ 1 & -1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 3 \\ 2 \\ 0 \end{pmatrix}$$

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

Answer:

$$2x + (1-1)y = 2 + 0$$
$$\Rightarrow x = 1$$
$$\Rightarrow y = -1$$

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 =

    1.0000
   -1.0000
```

Check:

```
>> A * x - b

ans =

   1.0e-15 *

         0
         0
   -0.2220
```

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)
print(np.dot(A,x) - b)  #check

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

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

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

← using only first two equations

Example: Polynomial interpolation

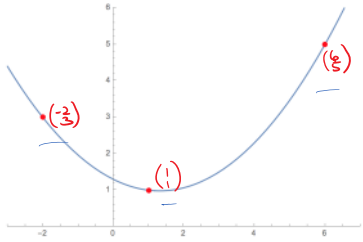
2 points determine a line

3 points determine a quadratic curve (parabola)

⋮

any $n+1$ points $(x_1, y_1), \ldots, (x_{n+1}, y_{n+1})$ 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:

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

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

$$\rightarrow \left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 36 & 6 & 1 & 5 \\ 4 & -2 & 1 & 3 \end{array}\right) \begin{matrix} \\ \cdot(-1) \\ \cdot(-1) \end{matrix}$$

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

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

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

$$\left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 36 & 6 & 1 & 5 \\ 4 & -2 & 1 & 3 \end{array}\right) \begin{matrix} \\ -36 \\ -4 \end{matrix} \rightarrow \left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 0 & -30 & -35 & -31 \\ 0 & -6 & -3 & -1 \end{array}\right) -5 \quad \leftarrow \text{this means } -6b-3c=-1$$

$$\rightarrow \left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 0 & 6 & 3 & 1 \\ 0 & 0 & 20 & 26 \end{array}\right) \Downarrow \quad c = \frac{13}{10}$$

Now work backward:

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

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

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

Check this!

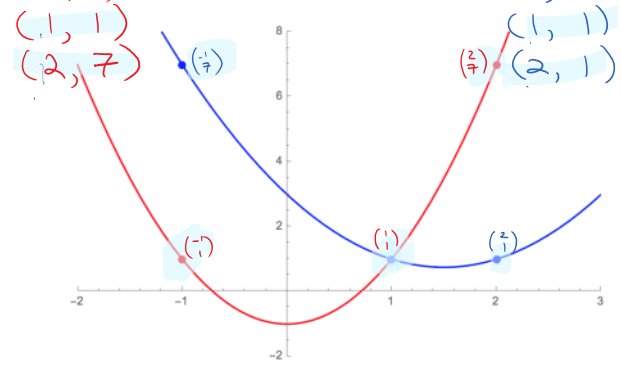
$x = \begin{pmatrix} 1 \\ 6 \\ -2 \end{pmatrix}$ $x.\hat{}2 = \begin{pmatrix} 1 \\ 36 \\ 4 \end{pmatrix}$

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 $x^2a + xa + 1 = y$

$$\left(\begin{array}{ccc|c} (-1)^2 & -1 & 1 & 1 \\ 1^2 & 1 & 1 & 1 \\ 2^2 & 2 & 1 & 7 \end{array}\right) \quad \left(\begin{array}{ccc|c} (-1)^2 & -1 & 1 & 7 \\ 1^2 & 1 & 1 & 1 \\ 2^2 & 2 & 1 & 1 \end{array}\right)$$

same matrix!

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

$$\left(\begin{array}{ccc|cc} 1 & -1 & 1 & 1 & 7 \\ 1 & 1 & 1 & 1 & 1 \\ 4 & 2 & 1 & 7 & 1 \end{array}\right) \begin{matrix} \\ -1 \\ -4 \end{matrix} \longrightarrow \left(\begin{array}{ccc|cc} 1 & -1 & 1 & 1 & 7 \\ 0 & 2 & 0 & 0 & -6 \\ 0 & 2 & -1 & 1 & -9 \end{array}\right) -1$$

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

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

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

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

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

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

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

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

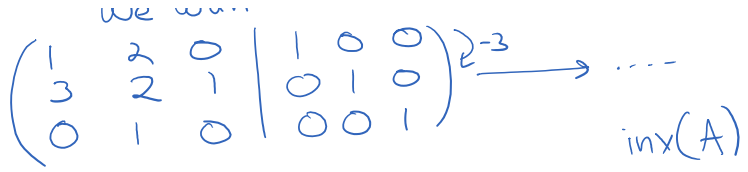
and can be solved together.

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

let $X = A^{-1}$

We want to solve $AX = I$

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



Answer: A^{-1} satisfies

$AA^{-1} = I$, so we can use Gaussian elimination to solve $\left(\begin{array}{c|c} A & I \end{array}\right)$ ...

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]

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:

Observe:

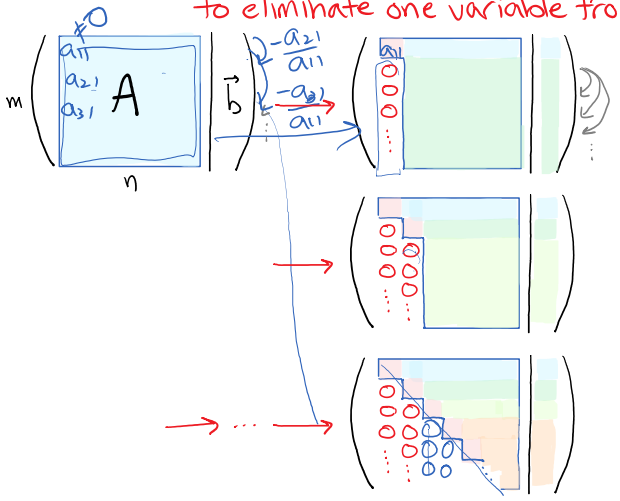
1) Multiplying an equation by $c \neq 0$
2) Adding one equation to another

} doesn't change the solutions

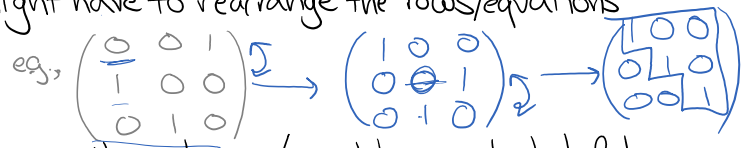
($\vec{x}$ solves eq. 1 and eq. 2 $\Leftrightarrow$ $\vec{x}$ solves eq. 1 and eq. 1 + eq. 2)

Therefore: Repeat:

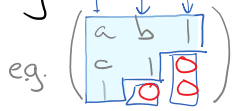
Add multiples of one equation to the others to eliminate one variable from them.



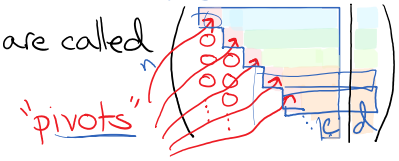
Notes: - You might have to rearrange the rows/equations



- Reordering the columns/variables can be helpful



- These are called "pivots"



$cx_n = d \Rightarrow x_n = \frac{d}{c}$

x_{n-1}

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n = b_1$$
$$a'_{22}x_2 + a'_{23}x_3 + \cdots + a'_{2n}x_n = b'_2$$
$$a'_{33}x_3 + \cdots + a'_{3n}x_n = b'_3$$
$$\vdots \qquad \ddots \qquad \vdots$$
$$a'_{nn}x_n = b'_n$$

Now solve for the last variable, backsubstitute.

last equation $\Rightarrow$ $x_n = \frac{b'_n}{a'_{nn}}$

substitute into eq. $n-1$ $\Rightarrow$ $a'_{n-1,n-1}x_{n-1} = b'_{n-1} - a'_{n-1,n}\frac{b'_n}{a'_{nn}}$

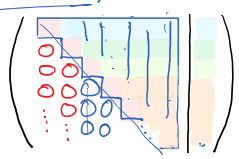
Keep on substituting backwards, solving for variables

$x_n, x_{n-1}, x_{n-2}, \ldots, x_2, x_1$

Complexity: How many basic operations are needed?

$O(n^3)$ Why?

n (rows to cancel) $\times$ $(n-1)$ (rows to add into) $\times$ $O(n)$ (to get to ↓) $= O(n^3)$



+ Then backsubstitution: $(n-1)+(n-2)+(n-3)+\cdots+2+1$

$= O(n^2)$

$O(n)$ 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× the memory (to store A)

8× 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:

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

pivot

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

$0x_1+0x_2+0x_3=1$

no solution!

② Solve

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

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

$x_2 = 3,\ x_1 = 1,\ x_3 = \text{something}$

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!

$$\left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 2 & 1 & 0 & 5 \\ -2 & 3 & 0 & 7 \end{array}\right) \quad \text{Now } x_1?$$

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

Next we backsubstitute

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

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

Check the answer!

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

LU DECOMPOSITION OF A MATRIX

Theorem: [Turing 1948]
Any $m \times n$ matrix A can be factored as

$$A_{m\times n} = \underbrace{P_{m\times m}}_{\text{permutation}} \underbrace{L_{m\times m}}_{\text{lower triangular}} \underbrace{U_{m\times n}}_{\text{upper triangular}}$$

U has the form $\begin{pmatrix} \text{upper triangular} \\ 0 \end{pmatrix}$ if $m \geq n$

Why? It is just Gaussian elimination!

eg.,

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

```
>> [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
   -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
```

U is the result of Gaussian elimination
L and P keep track of the steps

Proof sketch:

Claim 1: Multiplying on the left by

$$L = \begin{pmatrix} 1 & & & & \\ & 1 & & 0 & \\ & & \ddots & & \\ & c & & 1 & \\ & & & & \ddots \end{pmatrix} \quad \begin{matrix} \\ \\ \\ \leftarrow i \\ \\ \\ \uparrow j \end{matrix}$$

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

$$L = (I + C) = \begin{pmatrix} 1 & & & & \\ & 1 & & 0 & \\ & & \ddots & & \\ & c & & 1 & \\ & & & & \ddots \end{pmatrix}$$

$$LA = A + CA$$

$$= A + \begin{pmatrix} 0 & & & & \\ & \ddots & & & \\ & & 0 & & \\ & a_{j1} & a_{j2} & \cdots & a_{jn} \\ & & & & 0 \end{pmatrix} \leftarrow i$$

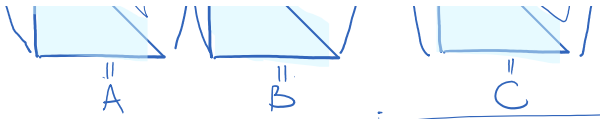
Claim 2: The product of lower-triangular matrices is also lower triangular.

$$\begin{pmatrix} & & \\ & \Delta & \\ & & \end{pmatrix}\begin{pmatrix} & & \\ & B & \\ & & \end{pmatrix} = \begin{pmatrix} & & \\ & C & \\ & & \end{pmatrix}$$

Proof:

$A_{ik} = 0$ if $k > i$
$B_{kj} = 0$ if $j > k$

Proof:



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

$$C_{ij} = \sum_{k=1}^{i} A_{ik} B_{kj} = \begin{cases} \\ \\ \end{cases} = 0 \text{ if } j > i$$

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     1     0
     1     0     0
     0     0     1

>> L

L =

    1.0000         0         0
    0.5789    1.0000         0
    0.0002    0.4247    1.0000

>> 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

>> P' * L * U - A

ans =

   1.0e-16 *

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

note the transpose!

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]]))
```

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 $\vec{b}, \vec{b}', \vec{b}'', \ldots,$

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). $O(n^3)$ once only in advance

Then just multiply

$\vec{x} = A^{-1}\vec{b}$, $\vec{x}' = A^{-1}\vec{b}'$, $\vec{x}'' = A^{-1}\vec{b}''$, ... — each takes $O(n^2)$ more time.

Don't compute inverses!

Slow ✓

Numerically unstable ✓

A^{-1} might not exist ✓

Slow ✓
Numerically unstable ✓ A^{-1} might not exist ✓
$(\text{sparse matrix})^{-1}$ can be dense ✓

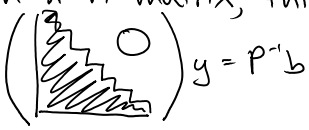
$A\vec{x} = \vec{b}$
$PLU\vec{x} = \vec{b}$
$\Downarrow$
$LU\vec{x} = P^T\vec{b}$

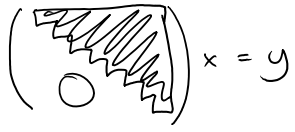
Application: Quickly solving multiple sets of equations with the same A

Goal: Solve $A\vec{x} = \vec{b}$
$A = PLU$

0. Precompute P, L, U — $O(n^3)$
1. Solve $L\vec{y} = P^T\vec{b}$ — $O(n^2)$ time (back subst)
2. Solve $\vec{y} = U\vec{x}$ for $\vec{x}$ — $O(n^2)$ time

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 $x_n, x_{n-1}, x_{n-2}, \ldots$)


$\vec{x} = \vec{y}$

∴ Precomputing the LU decomposition of A allows for solving $Ax = b$, $Ax' = b'$, $Ax'' = b''$, ... faster.

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

More interesting Gaussian elimination examples

Example: Solve

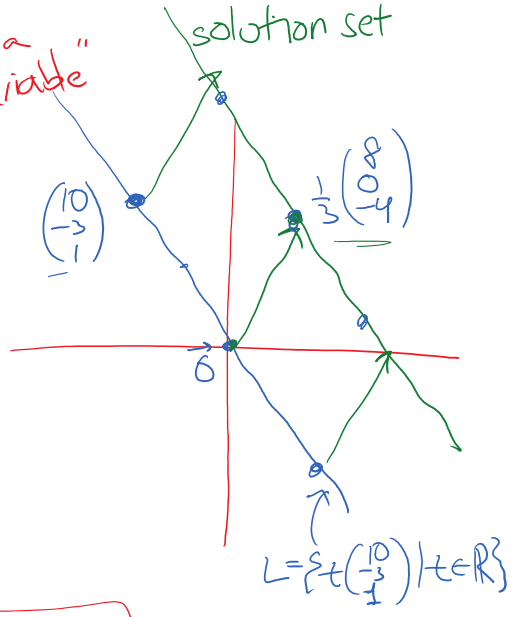
$$x + 3y - z = 4$$
$$3x + 10y = 8$$

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

A

y is called a "free variable"

$$-\frac{1}{3}y - z = \frac{4}{3}$$
$$3x + 10y = 8$$



⇒ For any choice of $y \in \mathbb{R}$,

$$x = \frac{8}{3} - \frac{10}{3}y$$
$$z = -\frac{4}{3} - \frac{1}{3}y$$

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

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \frac{1}{3}\begin{pmatrix} 0 \\ 0 \\ -4 \end{pmatrix} + \begin{pmatrix} 1 \\ -\frac{1}{3} \\ \end{pmatrix} t \quad \text{for any}$$

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

solution set

$A\begin{pmatrix} 10 \\ -3 \\ 1 \end{pmatrix} = \vec{0}$ ← homogeneous eq.

$A\begin{pmatrix} 8 \\ 0 \\ -4 \end{pmatrix} = \begin{pmatrix} 4 \\ 8 \end{pmatrix}$ ← inhomogeneous eq.

Answer:

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

The first step is already done!

$$\left(\begin{array}{ccc|c} \square & \square & \square & \square \\ 0 & \square & \square & \square \end{array}\right)$$

Now backsubstitute.

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

$$\Rightarrow x = \frac{8}{3} - \frac{10}{3}y$$
$$z = -\frac{4}{3} - \frac{1}{3}y \quad \text{where } y \text{ is arbitrary!}$$

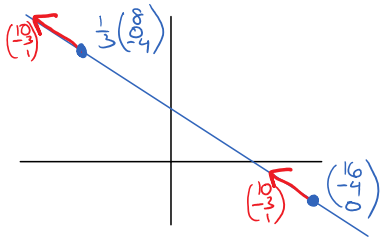
$$\Rightarrow \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \frac{1}{3}\begin{pmatrix} 8 \\ 0 \\ -4 \end{pmatrix} + \begin{pmatrix} -\frac{10}{3} \\ 1 \\ -\frac{1}{3} \end{pmatrix} y, \quad y \in \mathbb{R}$$

"free variable"

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

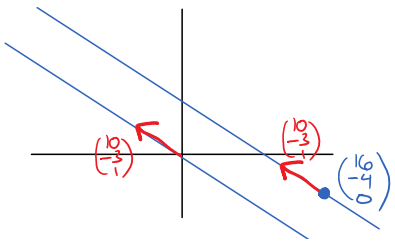
Note: You can also solve the equations with x or z free

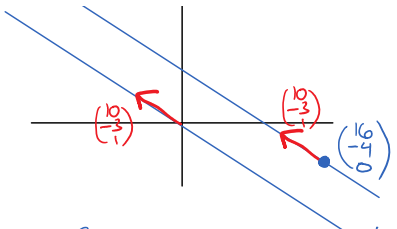
eg., $\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 16 \\ -4 \\ 0 \end{pmatrix} + \begin{pmatrix} 10 \\ -3 \\ 1 \end{pmatrix} z, \quad z \in \mathbb{R}$



Observe:

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"

Definition: 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 $A\vec{x}=\vec{b}\neq\vec{0}$ 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 $A\vec{y}=\vec{b}$ $\Rightarrow A(\vec{x}-\vec{y})=A\vec{x}-A\vec{y}=\vec{b}-\vec{b}=\vec{0}$
  $\Rightarrow \vec{x}-\vec{y}$ solves homog. syst.
- If $A\vec{x}=\vec{b}$ and $A\vec{y}=\vec{0}$
  $\Rightarrow A(\vec{x}+\vec{y})=A\vec{x}+A\vec{y}=\vec{b}+\vec{0}=\vec{b}$

Example: Solve

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

NEXT TIME

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

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

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

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

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

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

$\Rightarrow x_3=-x_4$

$x_1=-2x_2-x_4$

$\Rightarrow$ solutions $=\{(x_1,x_2,x_3,x_4)\in\mathbb{R}^4 : x_1=-2x_2-x_4\}$

$x_3 = -x_4$

x_2 & x_4 are free — can be arbitrary

— the solution set is a 2D plane 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

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

same A

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

contradiction ⇒ no solution!

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

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

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.)
* 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

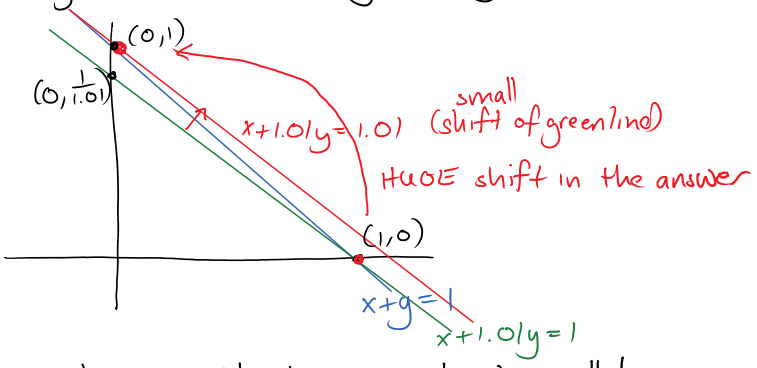
Example: Consider the equations

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

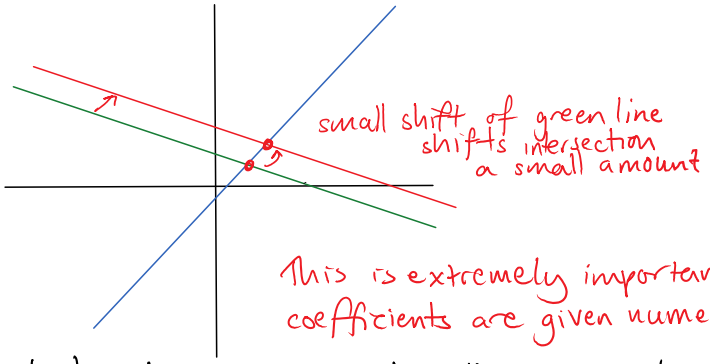
$$\begin{cases} x + y = 1 \\ x + 1.01y = 1.01 \end{cases} \Rightarrow (x,y) = (0,1)$$

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

Why? Observe the geometry:

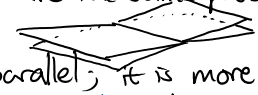


- because the lines are almost parallel this wouldn't have happened if the lines were at a greater $\measuredangle$



This is extremely important when coefficients are given numerically.

Note: In higher dimensions, you have the same problem 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

② Linear programming

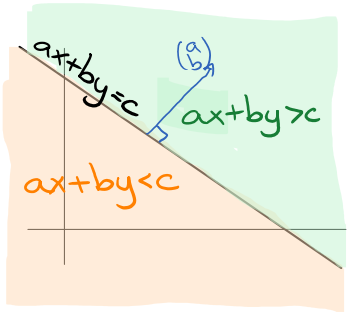
Problem: Solve a system of linear inequalities

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

- 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
  $ax+by \leq c$ $\quad\Leftrightarrow\quad$ $ax+by=c$
  $ax+by \geq c$

Geometry:

Halfspaces:



Convex polygon = intersection of halfspaces