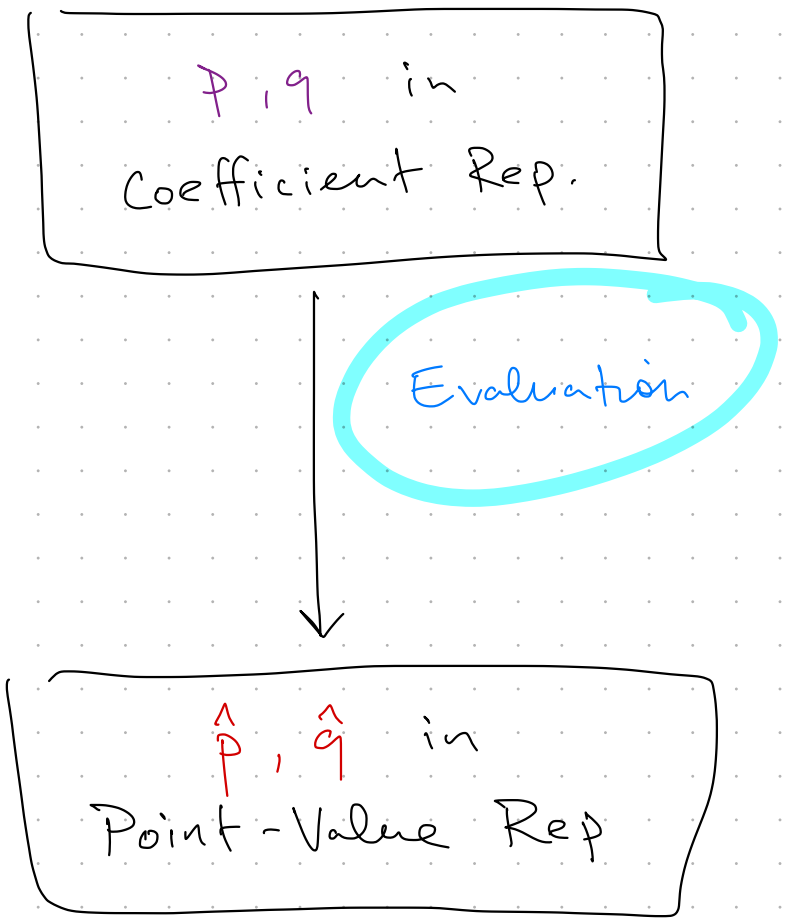
12 March 2025

Linear Systems

Plan

* Finish FFT
* Announcements
* Linear Systems



How quickly can we convert from

COEFFICIENT REPR.

↓ to

POINT-VALUE REPR. ?

Evaluation Problem

Given. degree-n polynomial P
in coefficient representation

$$P = p_n, p_{n-1}, \ldots p_1, p_0 \quad \text{s.t.} \quad p(x) = \sum_{i=0}^{n} p_i \cdot x^i$$

Return. degree-n polynomial $\hat{p}$
in point-value (PV) representation

That is:

canonical set of points $x_0, x_1, \ldots, x_{n-1}, x_n$

with evaluations $\underbrace{p(x_0), p(x_1), \ldots, p(x_{n-1}), p(x_n)}_{\hat{p}}$

Idea. Divide & Recurse Based on degree

$$p(x) = 5x^7 + x^6 - 3x^5 + 4x^4 - x^3 + 2x^2 + x - 1$$

$$p_{even}(x) = x^3 + 4x^2 + 2x - 1$$

$$p_{odd}(x) = 5x^3 - 3x^2 - x + 1$$

$$p(x) = p_{even}(x^2) + x \cdot p_{odd}(x^2)$$

$$x^2 = 1 \Rightarrow x = \begin{cases} \sqrt{1} \\ -\sqrt{1} \end{cases}$$

Idea. Divide & Recurse Based on degree

$$p(x) = 5x^7 + x^6 - 3x^5 + 4x^4 - x^3 + 2x^2 + x - 1$$

$$p_{even}(x) = x^3 + 4x^2 + 2x - 1$$

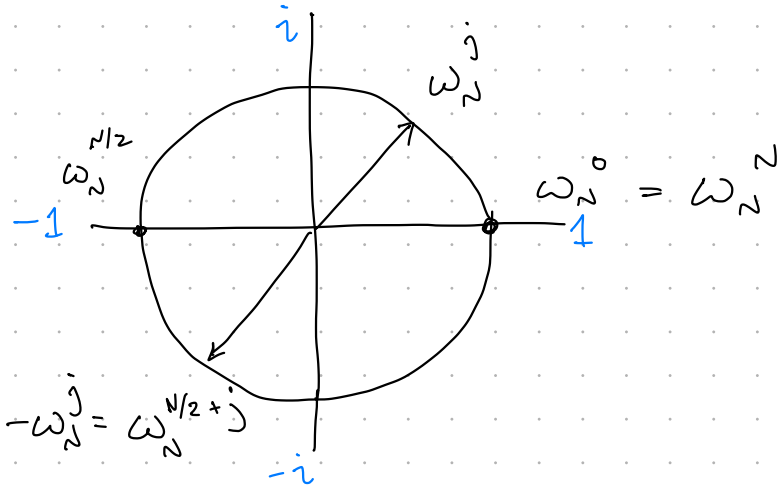
$$p_{odd}(x) = 5x^3 - 3x^2 - x + 1$$

$$p(x) = p_{even}(x^2) + x \cdot p_{odd}(x^2)$$

$$x^2 = 1 \implies x = \begin{cases} \sqrt{1} \\ -\sqrt{1} \end{cases}$$

$$x^4 = 1 \implies x^2 = \begin{cases} \sqrt{1} \\ -\sqrt{1} \end{cases} \implies x = \begin{cases} \sqrt{\sqrt{1}} = 1 \\ -\sqrt{\sqrt{1}} = -1 \\ \sqrt{-\sqrt{1}} = i \\ -\sqrt{-\sqrt{1}} = -i \end{cases}$$

Evaluate on "Roots of Unity"



$\omega_N \equiv N^{th}$ root of unity

Complex Number s.t. $\omega_N^N = 1$

$$\overrightarrow{\omega_N} = \omega_N^0 \; \omega_N^1 \; \omega_N^2 \; \cdots \; \omega_N^{n-2} \; \omega_N^{n-1} \; \omega_N^n$$

Key properties.

(1) $\omega_N^{N/2} = -1 \implies (\omega_N^j)^2 = (\omega_N^{j+N/2})^2$

$\implies$ 1 recursive call, 2 evaluations!

(2) $\omega_N^2 \equiv (N/2)^{th}$ root of unity

$\implies$ Root of unity structure preserved during Recursion

FFT: Algorithm for the Evaluation problem.

Given polynomial of degree n, P

Returns $\hat{P}$, evaluation at

N^{th} Roots of Unity

$$\vec{\omega}_N = \omega_N^0 \quad \omega_N^1 \quad \cdots \quad \omega_N^{N-1}$$

for $N = 2^d > n$.

FFT $\left(p = p_n p_{n-1} p_{n-2} \cdots p_2 p_1 p_0,\ \omega_N\right)$

Split p into

$p_{even} = p_n p_{n-2} \cdots p_2 p_0$ // Coefficients of even & odd degree monomials

$p_{odd} = p_{n-1} p_{n-3} \cdots p_3 p_1$

Goal:

$$p(x) = p_{even}(x^2) + x \cdot p_{odd}(x^2)$$

FFT $\left(p = p_n p_{n-1} p_{n-2} \cdots p_2 p_1 p_0,\ \omega_N\right)$

Split p into

$p_{even} = p_n p_{n-2} \cdots p_2 p_0$ // Coefficients of even & odd degree monomials

$p_{odd} = p_{n-1} p_{n-3} \cdots p_3 p_1$

$\hat{p}_{even} \leftarrow FFT\left(p_{even}, \omega_N^2\right)$

$\hat{p}_{odd} \leftarrow FFT\left(p_{odd}, \omega_N^2\right)$

// Evaluate p_{even} and p_{odd} on $(N/2)^{th}$ Roots BY Key Property (2)

Goal:

$$p(x) = p_{even}(x^2) + x \cdot p_{odd}(x^2)$$

FFT $\left(p = p_n p_{n-1} p_{n-2} \cdots p_2 p_1 p_0,\ \omega_N\right)$

Split p into

$p_{even} = p_n p_{n-2} \cdots p_2 p_0$ // Coefficients of even & odd degree monomials

$p_{odd} = p_{n-1} p_{n-3} \cdots p_3 p_1$

$\hat{p}_{even} \leftarrow FFT\left(p_{even}, \omega_N^2\right)$ // Evaluate p_{even} and p_{odd} on $(N/2)^{th}$ Roots

$\hat{p}_{odd} \leftarrow FFT\left(p_{odd}, \omega_N^2\right)$

For $j = 0, \cdots, N$.

$$\hat{p}_j = \left(\hat{p}_{even}\right)_{j \bmod N/2} + \omega_N^j \cdot \left(\hat{p}_{odd}\right)_{j \bmod N/2}$$

$$p_{even}\left(\left(\omega_N^j\right)^2\right) = p_{even}\left(\left(\omega_N^{j+N/2}\right)^2\right)$$

By Key Property (1)

FFT $\left(p = p_n p_{n-1} p_{n-2} \cdots p_2 p_1 p_0,\ \omega_N\right)$

Split p into

$p_{even} = p_n p_{n-2} \cdots p_2 p_0$ // Coefficients of even & odd degree monomials

$p_{odd} = p_{n-1} p_{n-3} \cdots p_3 p_1$

$\hat{p}_{even} \leftarrow FFT\left(p_{even}, \omega_N^2\right)$ // Evaluate p_{even} and p_{odd} on $(N/2)^{th}$ Roots

$\hat{p}_{odd} \leftarrow FFT\left(p_{odd}, \omega_N^2\right)$

For $j = 0, \ldots, N$.

$$\hat{p}_j = \left(\hat{p}_{even}\right)_{j \bmod N/2} + \omega_N^j \cdot \left(\hat{p}_{odd}\right)_{j \bmod N/2}$$

Return $\hat{p}$.

// $p(x) = p_{even}(x^2) + x \cdot p_{odd}(x^2)$

where $\left(\omega_N^j\right)^2 = \left(\omega_N^{j+N/2}\right)^2$

For $N = 2^d > n$

FFT $\left(p = p_n p_{n-1} p_{n-2} \cdots p_2 p_1 p_0, \ \omega_N \right)$

if $|p| = 1$ Return $\hat{p}_0 = p_0$ // Base case, constant deg-0 polynomial

Split p into

$p_{even} = p_n p_{n-2} \cdots p_2 p_0$

$p_{odd} = p_{n-1} p_{n-3} \cdots p_3 p_1$ // Coefficients of even & odd degree monomials

$\hat{p}_{even} \leftarrow FFT\left(p_{even}, \omega_N^2\right)$

$\hat{p}_{odd} \leftarrow FFT\left(p_{odd}, \omega_N^2\right)$ // Evaluate p_{even} and p_{odd} on $(N/2)^{th}$ Roots

For $j = 0, \cdots, N$.

$$\hat{p}_j = \left(\hat{p}_{even}\right)_{j \bmod N/2} + \omega_N^j \cdot \left(\hat{p}_{odd}\right)_{j \bmod N/2}$$

Return $\hat{p}$.

// $p(x) = p_{even}(x^2) + x \cdot p_{odd}(x^2)$

where $\left(\omega_N^j\right)^2 = \left(\omega_N^{j+N/2}\right)^2$

Running Time ?

* Each FFT makes 2 Recursive Calls.
* Each of size $N/2$
* Linear post-processing operations

$$T(N) \leq 2 \cdot T(N/2) + c \cdot N \quad \text{operations}$$

choose $N = \Theta(n)$

$\Rightarrow O(n \log n)$ basic operations

Announcements

* HW4 due
* HW3 grades available
* HW5 released after class
* Prelim #2 Conflict Survey Available on Ed

Systems of Equations

Common Problem: Many desiderata

↳ Encode each constraint as an equation

Systems of Equations

Common Problem: Many desiderata

↳ Encode each constraint as an equation

Goal: Find a setting of variables
s.t. ALL equations are satisfied simultaneously

Systems of Equations

Common Problem: Many desiderata

↳ Encode each constraint as an equation

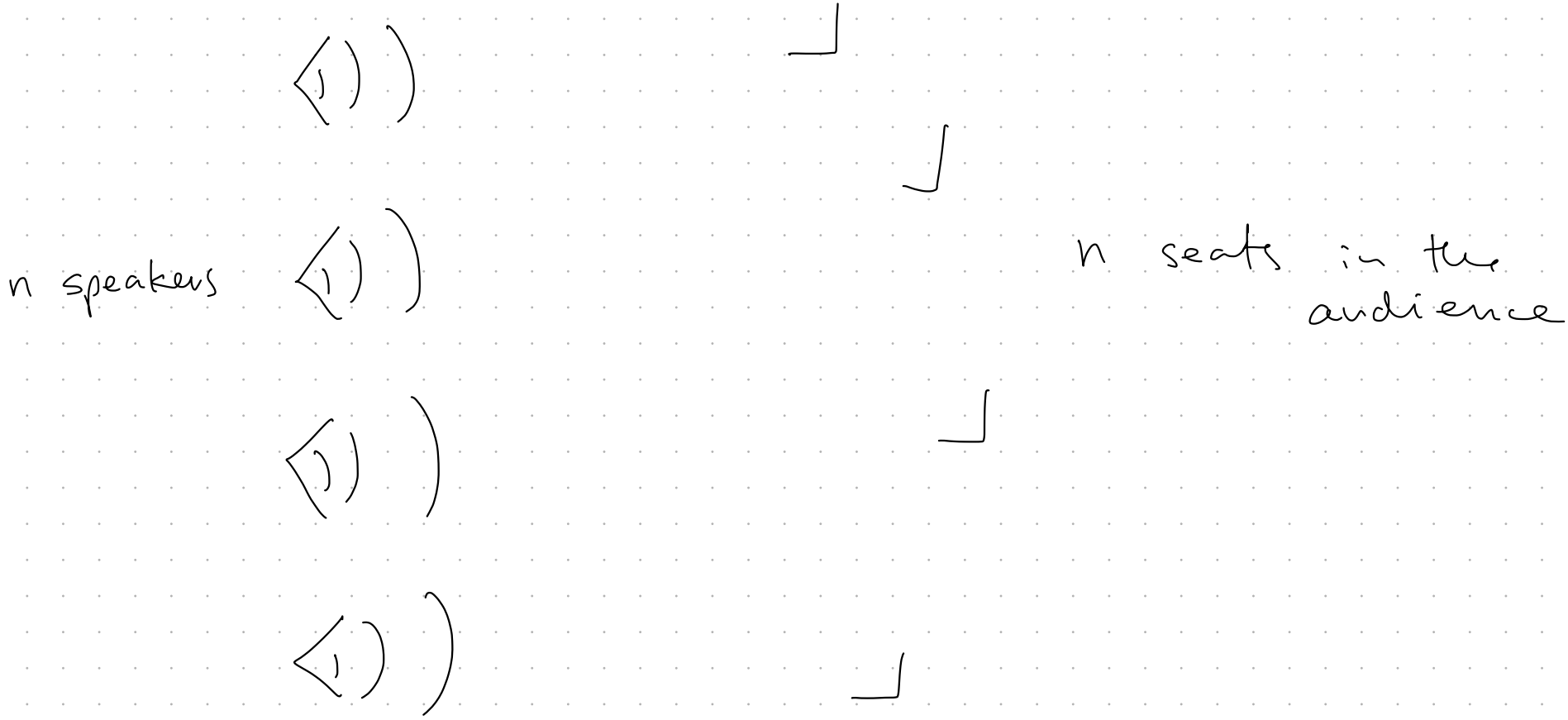
Goal: Find a setting of variables s.t. ALL equations are satisfied simultaneously

4820 Question.

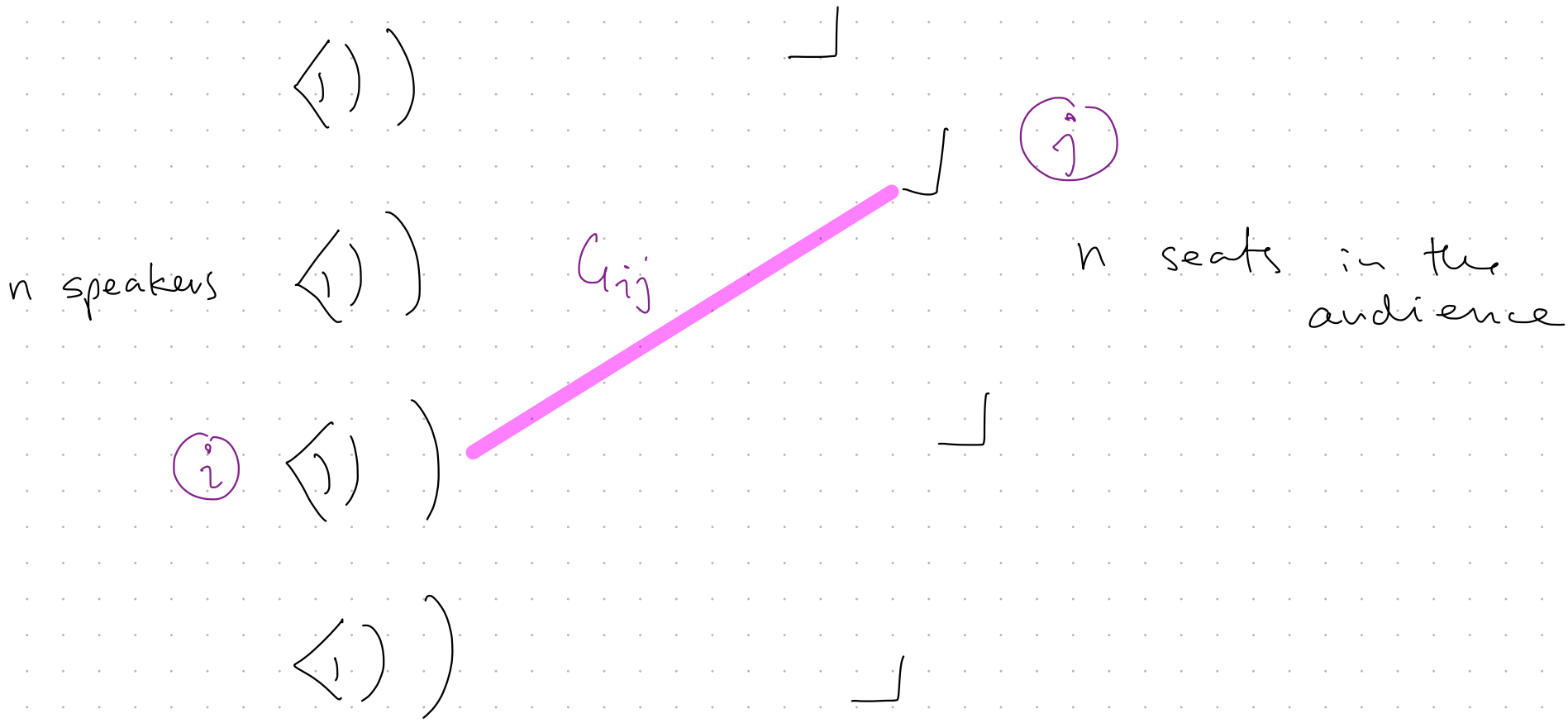
* Can we determine whether a system of eqns is satisfiable? Efficiently?

* How does the "type" of eqn change the hardness of the problem?

Toy Example: Sound Design

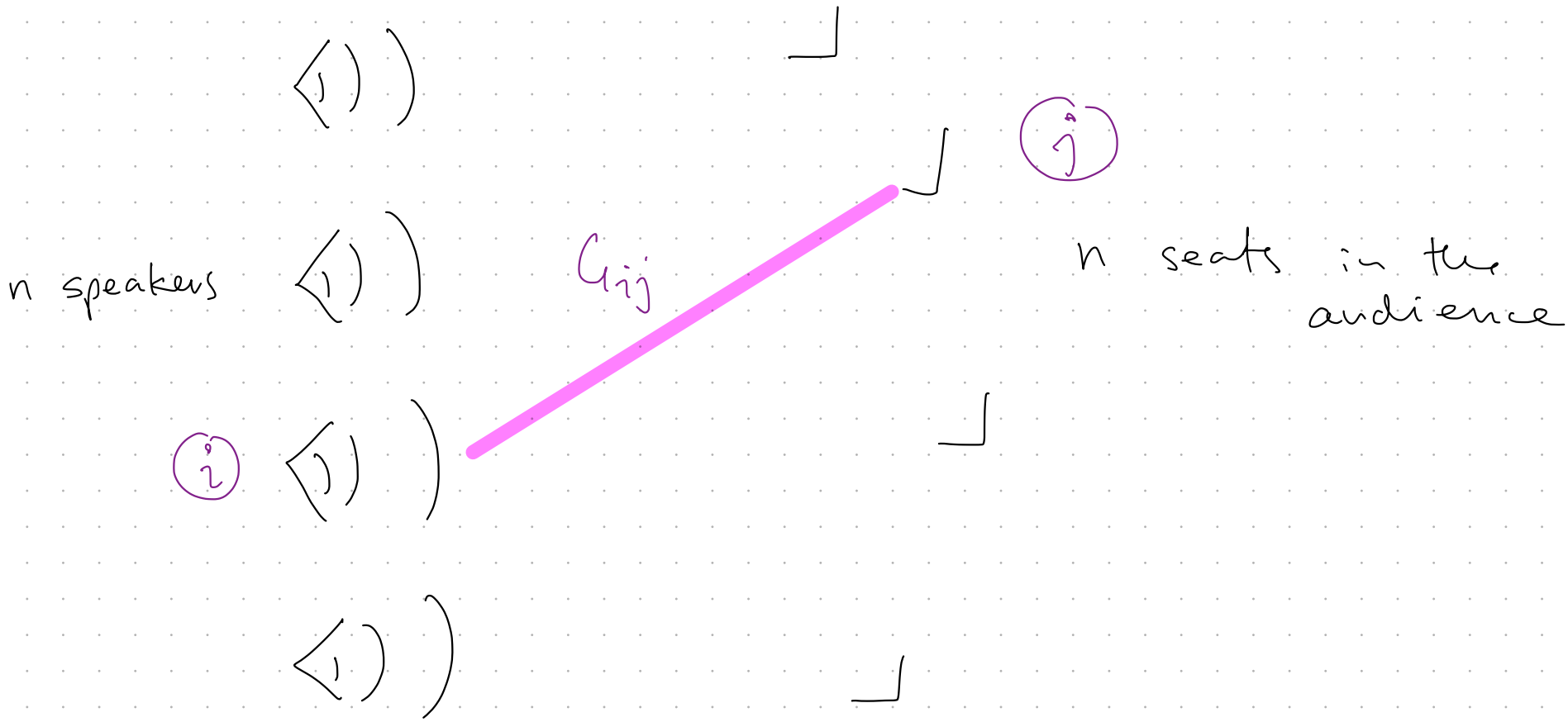


Toy Example: Sound Design



Gain G_{ji} : amplification from speaker i to seat j

Toy Example: Sound Design



Gain G_{ji} : amplification from speaker i to seat j

Goal: Personalized volumes

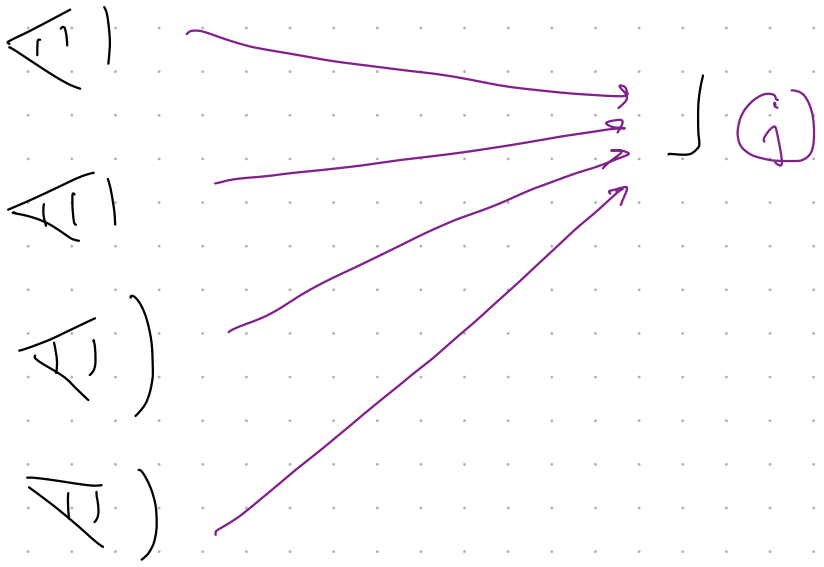
↳ Each audience member sets desired volume

Q: Can we achieve volumes?

l_i = level out of speaker i

s_j = Volume at seat j

Does there exist a setting of levels $l_1 \dots l_n$ such that for all seats, the volume equals s_j?



volume at seat j given by sum of levels at speakers, weighted by gain

l_i = level out of speaker i

s_j = Volume at seat j

Does there exist a setting of levels $l_1 \dots l_n$ such that for all seats, the volume equals s_j?

System of Linear Equations!

For all $j = 1 \dots n$ $\quad s_j = \sum_{i=1}^{n} G_{ji} \cdot l_i$

Solving Linear Systems

$$A x \stackrel{?}{=} b$$

Solving Linear Systems

$$A x \stackrel{?}{=} b$$

$$\downarrow$$

$$\left[\begin{array}{c|c} A & b \end{array}\right]$$

Gaussian Elimination $\longrightarrow$

Row-Echelon Form

$$\left[\begin{array}{ccc|c} 1 & & & \\ & 1 & & \\ & & 1 & \\ 0 & & & 1 \end{array} \begin{array}{c} A' \\ \\ \\ \end{array} \middle| \begin{array}{c} \\ \\ \\ b' \end{array}\right]$$

$\downarrow$ Substitution

$$x = b''$$

Gaussian Elimination (A, b)

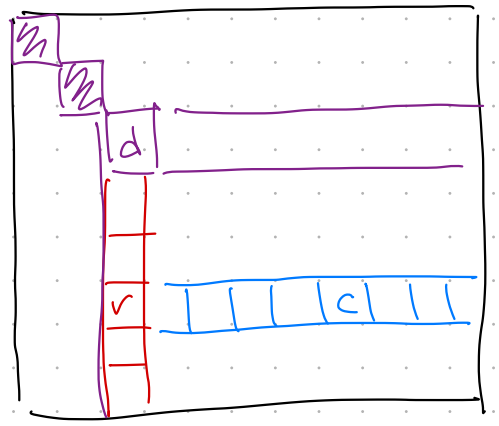
For each diagonal $d = 1, \ldots, n$.

$p \leftarrow$ Find row s.t. $A_{pd} \neq 0$. If no such row Return $\perp$.

Swap rows $A_{p:}$ and $A_{d:}$

& scale s.t. $A_{dd} = 1$.

Return A, b

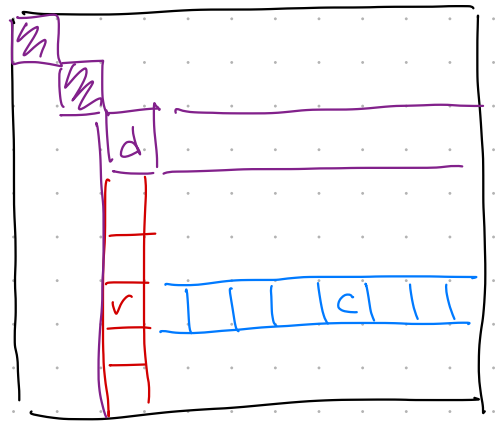


Gaussian Elimination (A, b)

For each diagonal $d = 1, \ldots, n$.

- $p \leftarrow$ Find row s.t. $A_{pd} \neq 0$. If no such row Return $\perp$.
- Swap rows $A_{p:}$ and $A_{d:}$ & scale s.t. $A_{dd} = 1$.
- For each row $r = d+1, \ldots, n$.
  - Zero out entries below A_{dd}

Return A, b

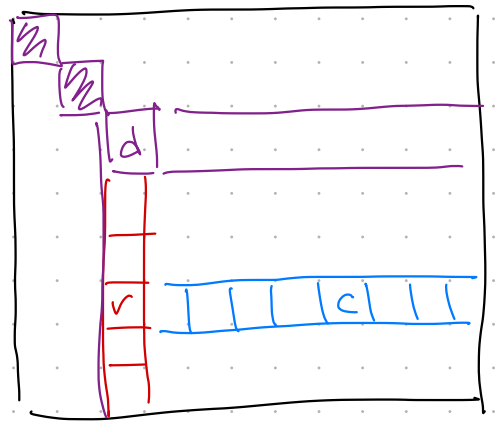


Gaussian Elimination (A, b)

For each diagonal $d = 1, \ldots, n$.

- $p \leftarrow$ Find row s.t. $A_{pd} \neq 0$. If no such row Return $\perp$.
- Swap rows $A_{p:}$ and $A_{d:}$ & scale s.t. $A_{dd} = 1$.
- For each row $r = d+1, \ldots, n$.
  - $s \leftarrow A_{rd}$
  - For each column $c = d, \ldots, n$.
    - $A_{rc} \leftarrow A_{rc} - s \cdot A_{dc}$
  - $b_r \leftarrow b_r - s \cdot b_d$

Return A, b



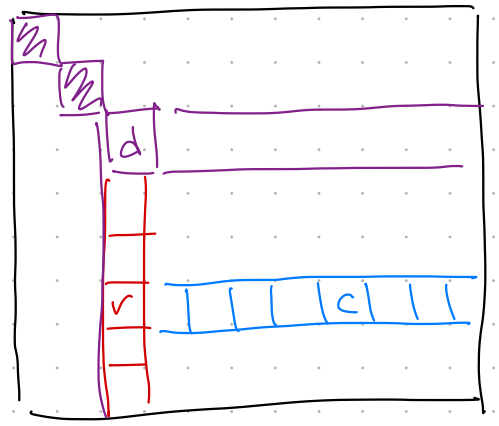
Gaussian Elimination (A, b)

$O(n) \cdot O(n) \cdot O(n)$

For each diagonal $d = 1, \ldots, n$.

- $p \leftarrow$ Find row s.t. $A_{pd} \neq 0$. If no such row Return $\perp$.
- Swap rows $A_{p:}$ and $A_{d:}$ & scale s.t. $A_{dd} = 1$.
- For each row $r = d+1, \ldots, n$.
  - $s \leftarrow A_{rd}$
  - For each column $c = d, \ldots, n$.
    - $A_{rc} \leftarrow A_{rc} - s \cdot A_{dc}$
  - $b_r \leftarrow b_r - s \cdot b_d$

Return A, b



Theorem Gaussian Elimination solves Linear Systems using $O(n^3)$ operations.

Many Problems Reduce to Linear System Solving!

Theorem Gaussian Elimination solves Linear Systems using $O(n^3)$ operations.

Note. For "real-valued" computation, need to worry about precision.

↳ Least-Squares / Numerical Stability

Theorem Gaussian Elimination solves Linear Systems using $O(n^3)$ operations.

Fact. Gaussian Elimination can solve Linear Systems over the integers mod p for prime p.

Ex. XOR equations

$$x \oplus y = (x + y) \bmod 2$$

x	y	$x \oplus y$	$x + y \bmod 2$
0	0	0	$0 \bmod 2 = 0$
1	0	1	$1 \bmod 2 = 1$
0	1	1	$1 \bmod 2 = 1$
1	1	0	$2 \bmod 2 = 0$

Ex. XOR equations

$$x_1 \oplus x_2 \oplus x_7 \oplus x_8 = 0$$

$$x_5 \oplus x_6 \oplus x_3 = 1$$

$$x_1 \oplus x_2 = 0$$

Ex. XOR equations

$$1 \oplus x_1 \oplus x_2 \oplus x_7 \oplus x_8 = 0 \quad 1$$

$$x_5 \oplus x_6 \oplus x_3 = 1$$

$$1 \oplus x_1 \oplus x_2 = 0 \quad 1$$

Ex. XOR equations

$$1 \oplus x_1 \oplus x_2 \oplus x_7 \oplus x_8 = \cancel{0}\ 1$$

$$x_5 \oplus x_6 \oplus x_3 = 1$$

$$1 \oplus x_1 \oplus x_2 = \cancel{0}\ 1$$

$\Updownarrow$ $\forall$ equations, XOR is 1.

$$(x_1 \oplus x_2 \oplus x_7 \oplus x_8 \oplus 1) \wedge (x_5 \oplus x_6 \oplus x_3) \wedge (x_1 \oplus x_2 \oplus 1) \equiv 1$$

Ex. XOR equations

$$1 \oplus x_1 \oplus x_2 \oplus x_7 \oplus x_8 = 1$$

$$x_5 \oplus x_6 \oplus x_3 = 1$$

$$1 \oplus x_1 \oplus x_2 = 1$$

$\Updownarrow$ $\forall$ equations, XOR is 1.

$$(x_1 \oplus x_2 \oplus x_7 \oplus x_8 \oplus 1) \wedge (x_5 \oplus x_6 \oplus x_3) \wedge (x_1 \oplus x_2 \oplus 1) \equiv 1$$

Corollary.

There exists a polynomial-time algorithm for determining whether a system of XOR equations is satisfiable.

↳ $\forall$ eqns. XOR is 1.

What about OR Equations?

$$x_1 \vee x_2 \vee x_5 = 1$$

$$x_3 \vee x_7 \vee \neg x_8 = 1$$

$$x_1 \vee \neg x_2 \vee x_6 = 1$$

$$(x_1 \vee x_2 \vee x_5) \wedge (x_3 \vee x_7 \vee \neg x_8) \wedge (x_1 \vee \neg x_2 \vee x_6)$$

Does there exist an efficient algorithm to determine if a system of OR-eqns is satisfiable?