# Matrix Rigidity

## Introduction

Sasha Golovnev

August 26, 2020

# Overview

- Classes: MW 9.30am–10.45am, Zoom

# Overview

- Classes: MW 9.30am–10.45am, Zoom
- Office Hours: M 10.45am–11.45am, Zoom

# Overview

- Classes: MW 9.30am–10.45am, Zoom
- Office Hours: M 10.45am–11.45am, Zoom
- Homework and Project

# Overview

- Classes: MW 9.30am–10.45am, Zoom
- Office Hours: M 10.45am–11.45am, Zoom
- Homework and Project
- Slides and Video

# Overview

- Classes: MW 9.30am–10.45am, Zoom
- Office Hours: M 10.45am–11.45am, Zoom
- Homework and Project
- Slides and Video
- email: `alex.golovnev@gmail.com`

# Outline

# Outline

- Definitions and Examples

# Outline

- Definitions and Examples
- Explicit Constructions

# Outline

- Definitions and Examples
- Explicit Constructions
- Semi-explicit Constructions

# Outline

- Definitions and Examples
- Explicit Constructions
- Semi-explicit Constructions
- Limitations

# Outline

- Definitions and Examples
- Explicit Constructions
- Semi-explicit Constructions
- Limitations
- Applications

# Definition and Examples

# Notation

$s = \|M\|_0 :=$ sparsity of $M$, i.e., the number of non-zero entries in $M$ $s$-sparse

# Notation

$\|M\|_0 :=$ sparsity of $M$, i.e., the number of non-zero entries in $M$

$$I_n = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$$

# Notation

$\|M\|_0 :=$ sparsity of $M$, i.e., the number of non-zero entries in $M$

$$I_n = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$$

$$0_n = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}$$

# Notation

$\|M\|_0 :=$ sparsity of $M$, i.e., the number of non-zero entries in $M$

$$I_n = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} \qquad J_n = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \ldots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{pmatrix}$$

$$0_n = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}$$

# Rigidity. Definition

## Definition

Let $\mathbb{F}$ be a field, $A \in \mathbb{F}^{n \times n}$ be a matrix, and $0 \leq r \leq n$. The rigidity of $A$ over $\mathbb{F}$, denoted by $\mathcal{R}_A^{\mathbb{F}}(r)$, is the Hamming distance between $A$ and the set of matrices of rank at most $r$.

# Rigidity. Definition

## Definition

Let $\mathbb{F}$ be a field, $A \in \mathbb{F}^{n \times n}$ be a matrix, and $0 \leq r \leq n$. The rigidity of $A$ over $\mathbb{F}$, denoted by $\mathcal{R}_A^{\mathbb{F}}(r)$, is the Hamming distance between $A$ and the set of matrices of rank at most $r$. Formally,

$$\mathcal{R}_A^{\mathbb{F}}(r) := \min_{\mathsf{rank}(A+C) \leq r} \|C\|_0 \ .$$

$A$ is $(R,S)$-Rigid $\Leftrightarrow R_A^{\mathbb{F}}(R) \geq S$

Non-rigid $=$ Sparse $+$ Low-Rank

Rigid $\neq$ Sparse $+$ Low-Rank

# Examples

Low-rank matrix $A \in \mathbb{F}^{n \times n}$, $\mathsf{rk}(A) = k$

$R^{\mathbb{F}}_A(R) = 0 \iff R \geq k$

$A = A' + O_n$

low-rank  sparse matrix

# Examples

Sparse matrix $A \in \mathbb{F}^{n \times n}$, $\|A\|_0 = s$

$\forall\ 0 \leq R \leq n$

$R_A^F(R) \leq S$

$A = A + O_n$

Sparse

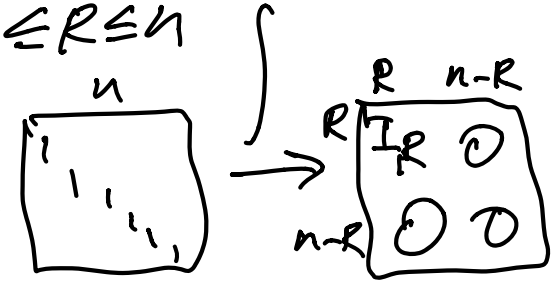
Low-rank

# Examples

Identityty matrix $I_n \in \mathbb{F}^{n \times n}$

$R^{\mathbb{F}}_{I_n}(R) = n - R$ $\quad 0 \le R \le n$

n-R changes




Pf. $R^{\mathbb{F}}_{I_n}(R) \le n - R$

$R_{I_n}(R) \ge n - R$

$rk(I_n + S) \ge rk(I_n) - rk(S) \ge$

$\ge n - (n - R - 1) = R + 1$

$\|S\|_0 < n - R$

$rk(S) \le \|S\|_0$

Sub-additivity of rank

# Examples

Let $n$ be a multiple of $2r$, and let

$$M_n = \begin{pmatrix} I_{2r} & I_{2r} & I_{2r} \\ \vdots & I_{2r} & \vdots \\ I_{2r} & \cdots & I_{2r} \end{pmatrix}.$$

$R^{\mathbb{F}}_{M_n}(R) \geq \frac{n^2}{4R}$ Actually tight.

Pf. Drop Rk of every $I_{2R}$ below R.

In every $I_{2R}$ you change $\geq R$ entries

$$\#\text{Matrices} = \left(\frac{n}{2R}\right)^2 \Rightarrow R^{\mathbb{F}}_{M_n}(R) \geq \left(\frac{n}{2R}\right)^2 \cdot R$$

$$= \frac{n^2}{4R} \quad \square$$

$$\begin{bmatrix} I_{2R} & I_{2R} \\ I_{2R} & I_{2R} \end{bmatrix} \Rightarrow \begin{bmatrix} I_R & 0_R & I_R & 0 \\ 0_R & 0_R & 0 & 0 \\ I_R & 0 & I_R & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

# Bounds on Rigidity

- Found a simple explicit matrix with rigidity

$$\mathcal{R}_{M_n}^{\mathbb{F}}(r) = \Omega\left(\frac{n^2}{r}\right).$$

# Bounds on Rigidity

- Found a simple explicit matrix with rigidity

$$\mathcal{R}^{\mathbb{F}}_{M_n}(r) = \Omega\left(\frac{n^2}{r}\right).$$

- The best known bound

$$\mathcal{R}^{\mathbb{F}}_{M_n}(r) = \Omega\left(\frac{n^2 \log(n/r)}{r}\right).$$

# Bounds on Rigidity

- Found a simple explicit matrix with rigidity

$$\mathcal{R}^{\mathbb{F}}_{M_n}(r) = \Omega\left(\frac{n^2}{r}\right).$$

If $R = \Omega(n)$ then $R^{\mathbb{F}}_{M_n}(R) \geq \Omega(n)$

- The best known bound

$$\mathcal{R}^{\mathbb{F}}_{M_n}(r) = \Omega\left(\frac{n^2 \log(n/r)}{r}\right).$$

- What we need (for circuit lower bounds) is

$$\mathcal{R}^{\mathbb{F}}_{M_n}(r) = n^{1+\delta} \text{ for } r = \Omega(n).$$

# Bounds on Rigidity

- Found a simple explicit matrix with rigidity
$$\mathcal{R}^{\mathbb{F}}_{M_n}(r) = \Omega\left(\frac{n^2}{r}\right).$$
- The best known bound
$$\mathcal{R}^{\mathbb{F}}_{M_n}(r) = \Omega\left(\frac{n^2 \log(n/r)}{r}\right).$$
- What we need (for circuit lower bounds) is
$$\mathcal{R}^{\mathbb{F}}_{M_n}(r) = n^{1+\delta} \text{ for } r = \Omega(n).$$
- (Even $\mathcal{R}^{\mathbb{F}}_{M_n}(r) = \omega(n)$ for $r = \Omega(n)$ would give new circuit lower bounds).

# Why Rigidity?

- Beautiful question between algebra and geometry

# Why Rigidity?

- Beautiful question between algebra and geometry
- Many applications to Communication Complexity, Data Structures, etc

# Why Rigidity?

- Beautiful question between algebra and geometry
- Many applications to Communication Complexity, Data Structures, etc
- One of the very few tools for Circuit Lower Bounds

# Circuit Complexity

# Boolean Circuits

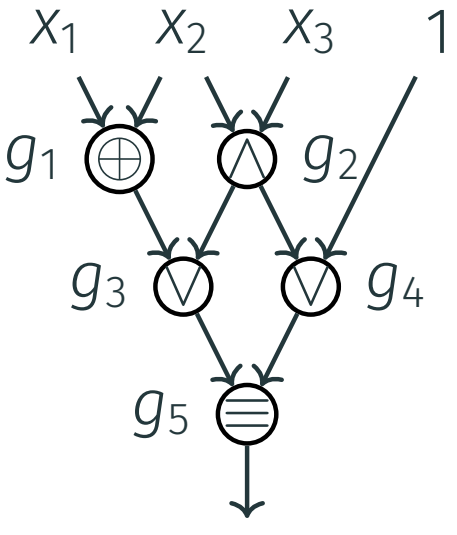
$$f\colon \{0,1\}^n \to \{0,1\}^n$$

$$\begin{aligned} g_1 &= x_1 \oplus x_2 \\ g_2 &= x_2 \wedge x_3 \\ g_3 &= g_1 \vee g_2 \\ g_4 &= g_2 \vee 1 \\ g_5 &= g_3 \equiv g_4 \end{aligned}$$

# Boolean Circuits

$$f\colon \{0,1\}^n \to \{0,1\}^n$$

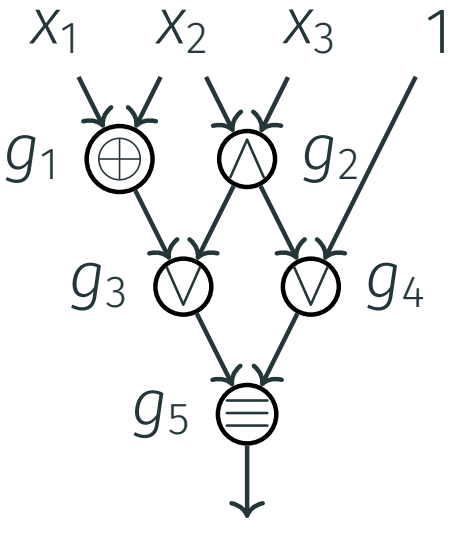
$$\begin{aligned} g_1 &= x_1 \oplus x_2 \\ g_2 &= x_2 \wedge x_3 \\ g_3 &= g_1 \vee g_2 \\ g_4 &= g_2 \vee 1 \\ g_5 &= g_3 \equiv g_4 \end{aligned}$$



# Boolean Circuits

$$f\colon \{0,1\}^n \to \{0,1\}^n$$

$$
\begin{aligned}
g_1 &= x_1 \oplus x_2 \\
g_2 &= x_2 \wedge x_3 \\
g_3 &= g_1 \vee g_2 \\
g_4 &= g_2 \vee 1 \\
g_5 &= g_3 \equiv g_4
\end{aligned}
$$




Inputs:
$x_1, \ldots, x_n, 0, 1$
Gates:
binary
functions
Fan-out:
unbounded

# Exponential Bounds

## Lower Bound [Sha1949]

Counting shows that almost all functions of $n$ variables have circuit size at least

$$2^n .$$

# Exponential Bounds

## Lower Bound [Sha1949]

Counting shows that almost all functions of $n$ variables have circuit size at least

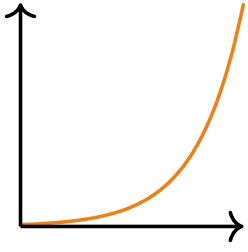
$$2^n .$$

## Upper Bound [Lup1958]

Every function can be computed by a circuit of size

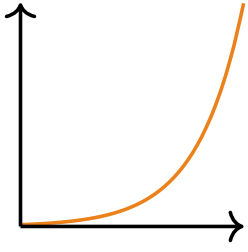
$$2^n .$$

# Explicit Bounds



Most functions have exponential circuit complexity

# Explicit Bounds

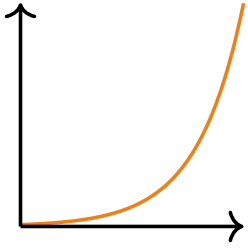


Most functions have exponential circuit complexity

$P \neq NP$

We want to prove super-polynomial lower bounds

# Explicit Bounds

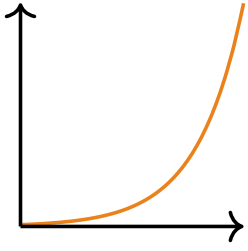


Most functions have exponential circuit complexity

$P \neq NP$

We want to prove super-polynomial lower bounds (for a function from **NP**)

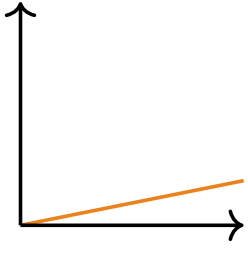
# Explicit Bounds



Most functions have exponential circuit complexity

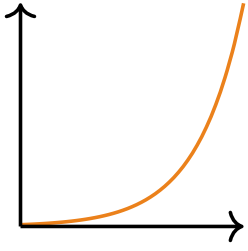
$P \neq NP$

We want to prove super-polynomial lower bounds (for a function from **NP**)



We can prove only $\approx 3n$ lower bounds

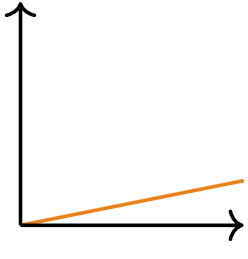
# Explicit Bounds



Most functions have exponential circuit complexity

$\mathbf{P} \neq \mathbf{NP}$

We want to prove super-polynomial lower bounds (for a function from **NP**)



We can prove only $\approx 3n$ lower bounds (even for a function from $\mathbf{E}^{\mathbf{NP}}$)

# Super-linear Circuit Lower Bounds?

- Two $n$-bit integers can be multiplied by a circuit of size $O(n \log n)$ [SS71,F07,HH19]

# Super-linear Circuit Lower Bounds?

- Two $n$-bit integers can be multiplied by a circuit of size $O(n \log n)$ [SS71,F07,HH19]
- Discrete Fourier Transform of a sequence of length $n$ can be computed by a circuit of size $O(n \log n)$

# Super-linear Circuit Lower Bounds?

- Two $n$-bit integers can be multiplied by a circuit of size $O(n \log n)$ [SS71,F07,HH19]
- Discrete Fourier Transform of a sequence of length $n$ can be computed by a circuit of size $O(n \log n)$
- Shifts, Permutations

# Super-linear Circuit Lower Bounds?

- Two $n$-bit integers can be multiplied by a circuit of size $O(n \log n)$ [SS71,F07,HH19]
- Discrete Fourier Transform of a sequence of length $n$ can be computed by a circuit of size $O(n \log n)$
- Shifts, Permutations
- **NP**-hard problems

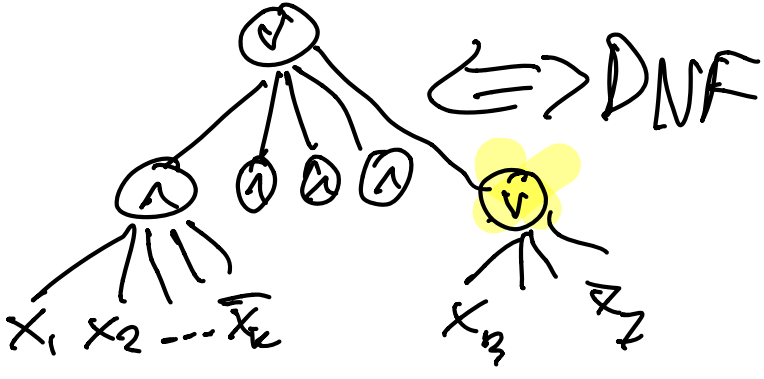
# Super-linear Circuit Lower Bounds?

- Two $n$-bit integers can be multiplied by a circuit of size $O(n \log n)$ [SS71,F07,HH19]
- Discrete Fourier Transform of a sequence of length $n$ can be computed by a circuit of size $O(n \log n)$
- Shifts, Permutations
- **NP**-hard problems
- ...

# What We Can Prove

# What We Can Prove

- Depth 2: CNF/DNF. Even $\oplus_n$ requires circuits of size $\Omega(2^n)$.



# What We Can Prove

- Depth 2: CNF/CNF. Even $\oplus_n$ requires circuits of size $\Omega(2^n)$.
- Constant depth $d$. Lower bounds $2^{n^{1/(d-1)}}$.

$d=3$ $\quad 2^{\Omega(\sqrt{n})}$

Fan-in 2.

$d = 0.9 \log n < \log n$

$x_1 \wedge \dots \wedge x_n$ $\quad \oplus_n$

# What We Can Prove

- Depth 2: CNF/CNF. Even $\oplus_n$ requires circuits of size $\Omega(2^n)$.
- Constant depth $d$. Lower bounds $2^{n^{1/(d-1)}}$.
- Depth $1.9 \log n$. Know functions that cannot be computed.

# What We Can Prove

- Depth 2: CNF/CNF. Even $\oplus_n$ requires circuits of size $\Omega(2^n)$.
- Constant depth $d$. Lower bounds $2^{n^{1/(d-1)}}$.
- Depth $1.9 \log n$. Know functions that cannot be computed.
- Depth $2 \log n$. Nothing better than $\approx 3n$.

# Problem on the Frontier

**Problem**

*Prove a lower bound of* $10n$ *against circuits of depth* $10 \log n$.

# Problem on the Frontier

**Problem**

*Prove a lower bound of* $10n$ *against circuits of depth* $10 \log n$.

*More generally, a lower bound of* $\omega(n)$ *against circuits of depth* $O(\log n)$.

# Problem on the Frontier

**Problem**

*Prove a lower bound of* $10n$ *against circuits of depth* $10 \log n$.

*More generally, a lower bound of* $\omega(n)$ *against circuits of depth* $O(\log n)$.

Valiant [Val77] gives us an amazing tool to study such circuits.

# Another Problem on the Frontier

**Problem**

*Prove a lower bound of $\omega(n)$ against linear circuits of depth $O(\log n)$.*

# Another Problem on the Frontier

**Problem**

*Prove a lower bound of $\omega(n)$ against linear circuits of depth $O(\log n)$.*

Valiant's [Val77] tool for these circuits is even nicer!

# Linear Circuits

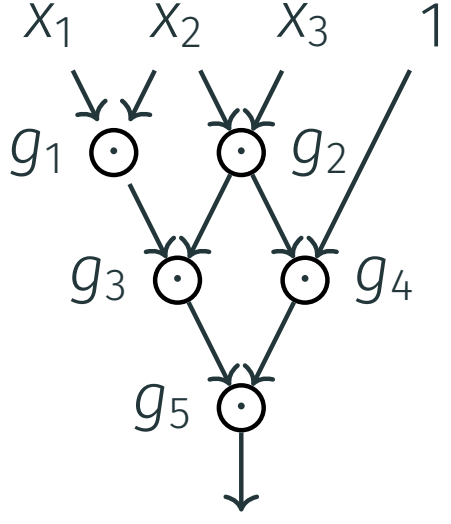
- A linear map computes $Mx$ for input $x \in \mathbb{F}^n$ where $M \in \mathbb{F}^{m \times n}$

# Linear Circuits

- A linear map computes $Mx$ for input $x \in \mathbb{F}^n$ where $M \in \mathbb{F}^{m \times n}$
- A linear circuit only contains gates that, for inputs $x$ and $y$, compute $\alpha x + \beta y$ for some $\alpha, \beta \in \mathbb{F}$

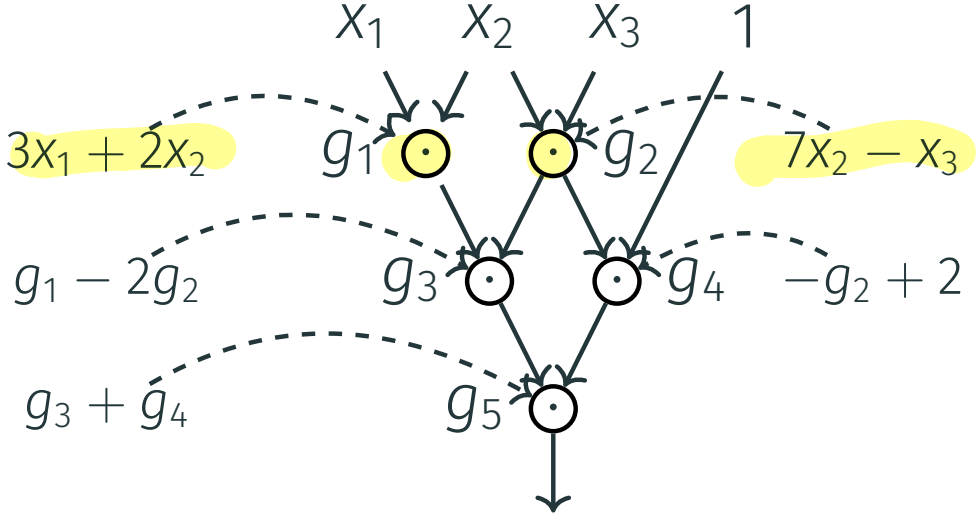
# Linear Circuits

- A linear map computes $Mx$ for input $x \in \mathbb{F}^n$ where $M \in \mathbb{F}^{m \times n}$
- A linear circuit only contains gates that, for inputs $x$ and $y$, compute $\alpha x + \beta y$ for some $\alpha, \beta \in \mathbb{F}$



# Linear Circuits

- A linear map computes $Mx$ for input $x \in \mathbb{F}^n$ where $M \in \mathbb{F}^{m \times n}$
- A linear circuit only contains gates that, for inputs $x$ and $y$, compute $\alpha x + \beta y$ for some $\alpha, \beta \in \mathbb{F}$



# Complexity of Linear Operators

- Linear circuits compute only linear functions

# Complexity of Linear Operators

- Linear circuits compute only linear functions
- We don't study linear functions with 1 output as they have circuit complexity $\leq n$ even in depth $\log n$

# Complexity of Linear Operators

- Linear circuits compute only linear functions
- We don't study linear functions with 1 output as they have circuit complexity $\leq n$ even in depth $\log n$
- A random linear map with $n$ outputs has complexity $n^2 / \log n$

# Complexity of Linear Operators

- Linear circuits compute only linear functions
- We don't study linear functions with 1 output as they have circuit complexity $\leq n$ even in depth $\log n$
- A random linear map with $n$ outputs has complexity $n^2/\log n$
- The best lower bound we can prove against linear circuits with $n$ outputs is $3n - o(n)$

# Another Problem on the Frontier

**Problem**

*Prove a lower bound of $\omega(n)$ against linear circuits of depth $O(\log n)$.*

# Another Problem on the Frontier

**Problem**

*Prove a lower bound of $\omega(n)$ against linear circuits of depth $O(\log n)$.*

- Incomparable to the previous problem (bounds against non-linear circuits):

# Another Problem on the Frontier

**Problem**

*Prove a lower bound of $\omega(n)$ against linear circuits of depth $O(\log n)$.*

- Incomparable to the previous problem (bounds against non-linear circuits):
- Weaker computational model

# Another Problem on the Frontier

**Problem**

*Prove a lower bound of $\omega(n)$ against linear circuits of depth $O(\log n)$.*

- Incomparable to the previous problem (bounds against non-linear circuits):
- Weaker computational model
- But fewer problems to prove lower bounds for.

# Circuits and Rigidity

# Rigidity Implies Circuit Lower Bounds

**Theorem (Val77)**

*Let $\mathbb{F}$ be a field, and $A \in \mathbb{F}^{n \times n}$ be a family of matrices for $n \in \mathbb{N}$.*

*If $\mathcal{R}_A^{\mathbb{F}}(\varepsilon n) > n^{1+\delta}$ for constant $\varepsilon, \delta > 0$, then any $O(\log n)$-depth linear circuit computing $x \to Ax$ must be of size $\omega(n)$.*

Rigidity for rank $n/100$ and sparsity $n^{1.01}$ implies super-linear circuit lower bounds

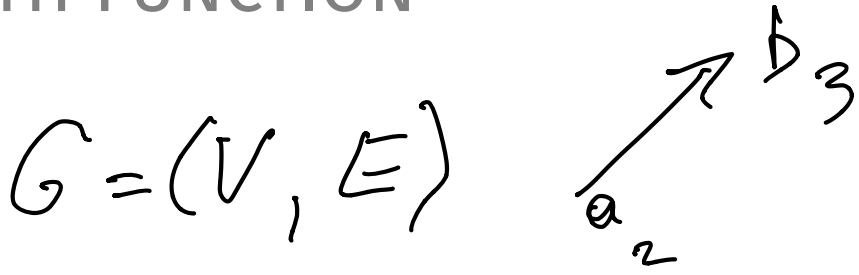
# Depth Reduction

**Lemma (EGS75)**

*Let $G$ be an acyclic digraph with $s$ edges and of depth $d = 2^k$.*

*There exists a set of $s/\log d$ edges in $G$ such that after their removal, the longest path in $G$ has length at most $d/2$.*

# Depth Function




$D : V \to \{0, 1, \ldots, d\}$ is a depth function for $G$ if for any $(a, b) \in E$, $D(a) < D(b)$.

**Claim**

Depth of $G \leq d$ iff there exists a depth function $D : V \to \{0, 1, \ldots, d-1\}$ for $G$.

for any $(a, b) \in E$, $D(a) < D(b)$.

**Claim**

Depth of $G \leq d$ iff there exists a depth function $D : V \to \{0, 1, \ldots, d-1\}$ for $G$.

Pf.:

1. Depth of $G \leq d \Rightarrow \exists$ depth fn

$D : V \to \{0, \ldots, d-1\}$

depth(v) = length of longest path from v to output

$D(v) = \text{depth}(v)$

$(a,b) \in E \Rightarrow D(a) < D(b)$

2. Depth of $> d \Rightarrow \exists$ path of length $\geq d$

cannot assign $0 \ldots d-1$ to the vertices of this path

□

# Depth Reduction. Proof

**Lemma (EGS75)**

*Let $G$ be an acyclic digraph with $s$ edges and of depth $d = 2^k$.*

*There exists a set of $s/\log d$ edges in $G$ such that after their removal, the longest path in $G$ has length at most $d/2$.*

**Lemma (EGS75)**

*Let $G$ be an acyclic digraph with $s$ edges and of depth $d = 2^k$.*

*There exists a set of $s/\log d$ edges in $G$ such that after their removal, the longest path in $G$ has length at most $d/2$.*

$D : V \to \{0, \dots, d-1\}$

$G = (V, E)$

$E_i = \{(a,b) \in E \mid D(a) < D(b)$ — Most significant bit where $D(a)$ & $D(b)$ differ is bit $\#i\}$

$1 \le i \le k \qquad d = 2^k$

$E_1 \dots E_k$ - partition $E$

By averaging, $\exists i$: $|E_i| \le \frac{|E|}{k} = \frac{s}{\log d}$

Remove $E_i$ $\Rightarrow$ new graph $G' = (V, E')$

Remains: longest path in $G' \le d/2$

$D' : V \to \{0, \dots, \frac{d}{2} - 1\}$ - depth fn for $G'$.

$D'(v) = D(v)$ without $i^{th}$ bit

$\forall (a,b) \in E'$ : $D'(a) < D'(b)$ ← WTS

$D(a) < D(b)$ $j$ is most sign bit $D(a)$ & $D(b)$ differ

Case 1. $j$ is more sign. than $i$ $D'(a) < D'(b)$

Case 2 $j = i \Rightarrow (a,b) \notin E'$

Case 3 $j$ is less sign. than $i$ $D'(a) < D'(b)$

□