Approximating the (Commutative) Rank of Symbolic Matrices

joint work with **Vishwas Bhargava**, **Markus Bläser** and **Gorav Jindal**

March 27, 2019

WACT 2019
ICTS, Bangalore

Markus Bläser

Vishwas Bhargava

Gorav Jindal

Slides-skeleton : Vishwas.

Central Problem: Rank of Symbolic Matrices

Suppose you have,

$$Q = \begin{pmatrix} q_{11} & q_{12} & \dots & q_{1n} \\ q_{21} & q_{22} & \dots & q_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ q_{n1} & q_{n2} & \dots & q_{nn} \end{pmatrix}.$$

where q_{ij} are degree-d polynomials $\in \mathbb{F}[x_1, \dots, x_m]$. Compute the rank of Q (over $\mathbb{F}(x_1, \dots, x_m)$)

For this talk, d is a constant..

Many equivalent definitions

1. Cardinality of a maximal linearly independent subset of row vectors (over $\mathbb{F}(x_1, \ldots, x_m)$).

Many equivalent definitions

1. Cardinality of a maximal linearly independent subset of row vectors (over $\mathbb{F}(x_1, \ldots, x_m)$).
2. The maximum number r such that at least one of the $r \times r$ minor is a non-zero **polynomial**.

Many equivalent definitions

1. Cardinality of a maximal linearly independent subset of row vectors (over $\mathbb{F}(x_1, \ldots, x_m)$).
2. The maximum number r such that at least one of the $r \times r$ minor is a non-zero **polynomial**.
3. Over large enough fields: same as the maximum possible rank of the evaluated Q (i.e evaluating the entries by fixing the variables $x_1, \ldots, x_m$ to some constants from $\mathbb{F}$) over $\mathbb{F}$.

Many equivalent definitions

1. Cardinality of a maximal linearly independent subset of row vectors (over $\mathbb{F}(x_1, \ldots, x_m)$).
2. The maximum number r such that at least one of the $r \times r$ minor is a non-zero **polynomial**.
3. Over large enough fields: same as the maximum possible rank of the evaluated Q (i.e evaluating the entries by fixing the variables $x_1, \ldots, x_m$ to some constants from $\mathbb{F}$) over $\mathbb{F}$.
4. Matrix Factorization: the smallest integer r such that Q can be factored as $Q = L \times M$, where Q is an $n \times r$ matrix and M is a $r \times n$ matrix. (entries of L and M come from $\mathbb{F}(x_1, \ldots, x_m)$).

Examples

Motivation:

Subsumes many computational problems arising in algebra, geometry and combinatorics.

Motivation:

Subsumes many computational problems arising in algebra, geometry and combinatorics.

- Linear Case:
  1. Size of maximum matching in a graph (using the Tutte Matrix).
  2. More generally, PIT for Determinants (and ABPs).

Motivation:

Subsumes many computational problems arising in algebra, geometry and combinatorics.

- Linear Case:
  1. Size of maximum matching in a graph (using the Tutte Matrix).
  2. More generally, PIT for Determinants (and ABPs).
- Non-Linear Case
  1. Algebraic rank (transcendence degree) of polynomials over zero characteristic (using the Jacobian Matrix)
  2. Dimension of the dual varieties of hypersurfaces (using the Hessian Matrix)

Homogenization

Can restrict ourselves to the case when the entries are all homogeneous degree-d polynomials.

Homogenization

Can restrict ourselves to the case when the entries are all homogeneous degree-d polynomials.

$f \in \mathbb{F}[x_1, \ldots, x_m]$ of degree at most d, the homogenization f^H of f, $f^H := \sum_{i=0}^{d} \text{hom}\, fi \cdot y^{d-i}$., (i.e. $f^H \in \mathbb{F}[x_1, \ldots, x_m, y]$)

The homogenization $Q^H(x_1, \ldots, x_m, y)$ of $Q(x_1, \ldots, x_m)$ is defined as $(Q^H)_{ij} := (Q_{ij})^H$.

Homogenization

Can restrict ourselves to the case when the entries are all homogeneous degree-d polynomials.

$f \in \mathbb{F}[x_1, \ldots, x_m]$ of degree at most d, the homogenization f^H of f, $f^H := \sum_{i=0}^{d} \text{hom } fi \cdot y^{d-i}$., (i.e. $f^H \in \mathbb{F}[x_1, \ldots, x_m, y]$)

The homogenization $Q^H(x_1, \ldots, x_m, y)$ of $Q(x_1, \ldots, x_m)$ is defined as $(Q^H)_{ij} := (Q_{ij})^H$.

Lemma

If $Q(x_1, \ldots, x_m)$ is a matrix with its entries being polynomials of degree at most d in the variables $x_1, \ldots, x_m$ and $|\mathbb{F}| > dn + 1$ then $rank(Q) = rank(Q^H)$

Homogenization

Can restrict ourselves to the case when the entries are all homogeneous degree-d polynomials.

$f \in \mathbb{F}[x_1, \ldots, x_m]$ of degree at most d, the homogenization f^H of f, $f^H := \sum_{i=0}^{d} \text{hom } fi \cdot y^{d-i}$., (i.e. $f^H \in \mathbb{F}[x_1, \ldots, x_m, y]$)

The homogenization $Q^H(x_1, \ldots, x_m, y)$ of $Q(x_1, \ldots, x_m)$ is defined as $(Q^H)_{ij} := (Q_{ij})^H$.

Lemma

If $Q(x_1, \ldots, x_m)$ is a matrix with its entries being polynomials of degree at most d in the variables $x_1, \ldots, x_m$ and $|\mathbb{F}| > dn + 1$ then $rank(Q) = rank(Q^H)$

Complexity status

1. Field of constant size: NP complete
2. Large enough fields: simple randomized polynomial time algorithm (Schwarz-Zippel)
3. Finding Rank here is as hard as PIT for ABPs, even when q_{ij}-s are linear forms.

Complexity status

1. Field of constant size: NP complete
2. Large enough fields: simple randomized polynomial time algorithm (Schwarz-Zippel)
3. Finding Rank here is as hard as PIT for ABPs, even when q_{ij}-s are linear forms.
4. When $q_{ij} \in \mathbb{F}\langle x_1, x_2, \ldots x_m \rangle$ are given as polynomial sized Non-commutative formula then (non-commutative) Rank is in **P**. (GGOW'16; IQS'17)

Complexity status

1. Field of constant size: NP complete
2. Large enough fields: simple randomized polynomial time algorithm (Schwarz-Zippel)
3. Finding Rank here is as hard as PIT for ABPs, even when q_{ij}-s are linear forms.
4. When $q_{ij} \in \mathbb{F}\langle x_1, x_2, \dots x_m \rangle$ are given as polynomial sized Non-commutative formula then (non-commutative) Rank is in **P**. (GGOW'16; IQS'17)
5. When q_{ij} are linear forms then commutative rank $\leq$ non-commutative rank $\leq$ $2\cdot$ commutative-rank. Thus GGOW'16, IQS'17 give deterministic polynomial time algorithms for computing factor 2 approximation of the commutative rank.

Complexity status

1. Field of constant size: NP complete
2. Large enough fields: simple randomized polynomial time algorithm (Schwarz-Zippel)
3. Finding Rank here is as hard as PIT for ABPs, even when q_{ij}-s are linear forms.
4. When $q_{ij} \in \mathbb{F}\langle x_1, x_2, \ldots x_m \rangle$ are given as polynomial sized Non-commutative formula then (non-commutative) Rank is in **P**. (GGOW'16; IQS'17)
5. When q_{ij} are linear forms then commutative rank $\leq$ non-commutative rank $\leq 2\cdot$ commutative-rank. Thus GGOW'16, IQS'17 give deterministic polynomial time algorithms for computing factor 2 approximation of the commutative rank.
6. BJP'17 improved this to give a $(1 - \epsilon)$ approximation algorithm for commutative rank in deterministic polynomial time.

Complexity status

1. Field of constant size: NP complete
2. Large enough fields: simple randomized polynomial time algorithm (Schwarz-Zippel)
3. Finding Rank here is as hard as PIT for ABPs, even when q_{ij}-s are linear forms.
4. When $q_{ij} \in \mathbb{F}\langle x_1, x_2, \ldots x_m \rangle$ are given as polynomial sized Non-commutative formula then (non-commutative) Rank is in **P**. (GGOW'16; IQS'17)
5. When q_{ij} are linear forms then commutative rank $\leq$ non-commutative rank $\leq 2\cdot$ commutative-rank. Thus GGOW'16, IQS'17 give deterministic polynomial time algorithms for computing factor 2 approximation of the commutative rank.
6. BJP'17 improved this to give a $(1-\epsilon)$ approximation algorithm for commutative rank in deterministic polynomial time.

Q. What if the entries are higher degree forms?

Q. What if the entries are higher degree forms?
The connection between non-commutative-rank and commutative rank is not known!

This work: makes a progress

Gives an algorithm for constant factor approximation of the rank in the case of higher degree forms.

This work: makes a progress

Gives an algorithm for constant factor approximation of the rank in the case of higher degree forms.

Theorem (PTAS for RANK)

Given $Q = [q_{ij}]$ *and a constant* $0 < \epsilon < 1$, *there exists a deterministic algorithm which computes an assignment* $(\lambda_1, \lambda_2, \ldots, \lambda_m) \in \mathbb{F}^m$ *such that,*

$$rk(Q(\lambda_1, \lambda_2, \ldots, \lambda_m)) \geq (1 - \epsilon) rk(Q(x_1, \ldots, x_m)).$$

This work: makes a progress

Gives an algorithm for constant factor approximation of the rank in the case of higher degree forms.

Theorem (PTAS for RANK)

Given $Q = [q_{ij}]$ and a constant $0 < \epsilon < 1$, there exists a deterministic algorithm which computes an assignment $(\lambda_1, \lambda_2, \ldots, \lambda_m) \in \mathbb{F}^m$ such that,

$$rk(Q(\lambda_1, \lambda_2, \ldots, \lambda_m)) \geq (1 - \epsilon) rk(Q(x_1, \ldots, x_m)).$$

Time Complexity- poly $\left((nmd)^{O\left(\frac{d^2}{\epsilon}\right)}\right)$

This work: makes a progress

Gives an algorithm for constant factor approximation of the rank in the case of higher degree forms.

Theorem (PTAS for RANK)

Given $Q = [q_{ij}]$ *and a constant* $0 < \epsilon < 1$, *there exists a deterministic algorithm which computes an assignment* $(\lambda_1, \lambda_2, \ldots, \lambda_m) \in \mathbb{F}^m$ *such that,*

$$rk(Q(\lambda_1, \lambda_2, \ldots, \lambda_m)) \geq (1 - \epsilon) rk(Q(x_1, \ldots, x_m)).$$

Time Complexity- poly $\left((nmd)^{O\left(\frac{d^2}{\epsilon}\right)}\right)$

Clearly, the above running time is polynomial when d and ϵ are constants.

This work: makes a progress

Gives an algorithm for constant factor approximation of the rank in the case of higher degree forms.

Theorem (PTAS for RANK)

Given $Q = [q_{ij}]$ *and a constant* $0 < \epsilon < 1$, *there exists a deterministic algorithm which computes an assignment* $(\lambda_1, \lambda_2, \ldots, \lambda_m) \in \mathbb{F}^m$ *such that,*

$$rk(Q(\lambda_1, \lambda_2, \ldots, \lambda_m)) \geq (1 - \epsilon) rk(Q(x_1, \ldots, x_m)).$$

Time Complexity- poly $\left((nmd)^{O\left(\frac{d^2}{\epsilon}\right)} \right)$

Clearly, the above running time is polynomial when d and ϵ are constants.

Can also be seen as an attempt to bridge the knowledge gap between the commutative and the non-commutative world.

Algebraic rank approximation

Corollary (PTAS for AlgRank)

Given a set $\mathbf{f} := \{f_1, \ldots, f_n\} \subset \mathbb{F}[x_1, \ldots, x_m]$ *of polynomials of degrees bounded by a constant* d *with* $char(\mathbb{F}) = 0$*, and a constant* $0 < \epsilon < 1$*, there is a deterministic algorithm that outputs a number* r*, such that* $r \geq (1 - \epsilon) \cdot algRank(\mathbf{f})$*.*

Time Complexity- poly $\left((nmd)^{O\left(\frac{d^2}{\epsilon}\right)}\right)$

The Algorithm

High-level Approach– Greedily increase the rank!

The Algorithm

High-level Approach– Greedily increase the rank!
If we cannot, we are already good.

The Algorithm

High-level Approach– Greedily increase the rank!
If we cannot, we are already good.

- We begin with all variables being set to 0
- Suppose we have an assignment $(\lambda_1, \lambda_2, \ldots, \lambda_m)$ s.t. $rk(Q(\lambda_1, \lambda_2, \ldots, \lambda_m)) = r$. i.e. $(\lambda_1, \lambda_2, \ldots, \lambda_m)$ "hits" an $r \times r$ minor of $Q(x_1, \ldots, x_m)$.
- We try to increase the rank by updating the assignment. By finding an assignment which "hits" an $(r+1) \times (r+1)$ minor.

The Algorithm

High-level Approach– Greedily increase the rank!
If we cannot, we are already good.

- We begin with all variables being set to 0
- Suppose we have an assignment $(\lambda_1, \lambda_2, \ldots, \lambda_m)$ s.t. $rk(Q(\lambda_1, \lambda_2, \ldots, \lambda_m)) = r$. i.e. $(\lambda_1, \lambda_2, \ldots, \lambda_m)$ "hits" an $r \times r$ minor of $Q(x_1, \ldots, x_m)$.
- We try to increase the rank by updating the assignment. By finding an assignment which "hits" an $(r+1) \times (r+1)$ minor.
  We update it to $(\lambda_1 + y_1, \lambda_2 + y_2, \ldots, \lambda_m + y_m)$

The Algorithm

High-level Approach– Greedily increase the rank!
If we cannot, we are already good.

- We begin with all variables being set to 0
- Suppose we have an assignment $(\lambda_1, \lambda_2, \ldots, \lambda_m)$ s.t. $rk(Q(\lambda_1, \lambda_2, \ldots, \lambda_m)) = r$. i.e. $(\lambda_1, \lambda_2, \ldots, \lambda_m)$ "hits" an $r \times r$ minor of $Q(x_1, \ldots, x_m)$.
- We try to increase the rank by updating the assignment. By finding an assignment which "hits" an $(r+1) \times (r+1)$ minor.
  We update it to $(\lambda_1 + y_1, \lambda_2 + y_2, \ldots, \lambda_m + y_m)$
- repeat as long as you can increase the rank by doing this.
- If we cannot, conclude that the current assignment already gives a good enough approximation.

The Algorithm

INPUT: A matrix Q, with entries from $\mathbb{F}[x_1, \ldots, x_m]$, with degrees bounded by d, and the approximation parameter $0 < \epsilon < 1$ (think $d = 2, \epsilon = 2/3$)

The Algorithm

INPUT: A matrix Q, with entries from $\mathbb{F}[x_1, \ldots, x_m]$, with degrees bounded by d, and the approximation parameter $0 < \epsilon < 1$ (think $d = 2, \epsilon = 2/3$)
OUTPUT: $\lambda_1, \ldots, \lambda_m \in \mathbb{F}$ such that
$\text{rank}(Q(\lambda_1, \ldots, \lambda_m)) \geq (1 - \epsilon)\text{rank}(Q(x_1, \ldots, x_m))$

The Algorithm

INPUT: A matrix Q, with entries from $\mathbb{F}[x_1, \ldots, x_m]$, with degrees bounded by d, and the approximation parameter $0 < \epsilon < 1$ (think $d = 2, \epsilon = 2/3$)
OUTPUT: $\lambda_1, \ldots, \lambda_m \in \mathbb{F}$ such that
$\text{rank}(Q(\lambda_1, \ldots, \lambda_m)) \geq (1 - \epsilon)\text{rank}(Q(x_1, \ldots, x_m))$

1. Set $s = \left\lceil \frac{d}{\epsilon} - 1 \right\rceil$ (in the example, $s = 2$)
2. Construct the (hitting) set $\mathcal{H}_{m,nd,s}$ of size $O((m(nd + 1))^s)$
3. Find an assignment $(\lambda_1, \lambda_2, \ldots, \lambda_m)$ such that $\text{rank}(Q(\lambda_1, \ldots, \lambda_m)) \geq (2s + 2)$

The Algorithm

INPUT: A matrix Q, with entries from $\mathbb{F}[x_1, \ldots, x_m]$, with degrees bounded by d, and the approximation parameter $0 < \epsilon < 1$ (think $d = 2, \epsilon = 2/3$)
OUTPUT: $\lambda_1, \ldots, \lambda_m \in \mathbb{F}$ such that
$\mathrm{rank}(Q(\lambda_1, \ldots, \lambda_m)) \geq (1 - \epsilon)\mathrm{rank}(Q(x_1, \ldots, x_m))$

1. Set $s = \lceil \frac{d}{\epsilon} - 1 \rceil$ (in the example, $s = 2$)
2. Construct the (hitting) set $\mathcal{H}_{m,nd,s}$ of size $O((m(nd+1))^s)$
3. Find an assignment $(\lambda_1, \lambda_2, \ldots, \lambda_m)$ such that $\mathrm{rank}(Q(\lambda_1, \ldots, \lambda_m)) \geq (2s+2)$ Using brute force over all $\binom{n}{(2s+2)}$ minors.

The Algorithm

INPUT: A matrix Q, with entries from $\mathbb{F}[x_1, \ldots, x_m]$, with degrees bounded by d, and the approximation parameter $0 < \epsilon < 1$ (think $d = 2, \epsilon = 2/3$)
OUTPUT: $\lambda_1, \ldots, \lambda_m \in \mathbb{F}$ such that
$\text{rank}(Q(\lambda_1, \ldots, \lambda_m)) \geq (1 - \epsilon)\text{rank}(Q(x_1, \ldots, x_m))$

1. Set $s = \left\lceil \frac{d}{\epsilon} - 1 \right\rceil$ (in the example, $s = 2$)
2. Construct the (hitting) set $\mathcal{H}_{m,nd,s}$ of size $O((m(nd+1))^s)$
3. Find an assignment $(\lambda_1, \lambda_2, \ldots, \lambda_m)$ such that $\text{rank}(Q(\lambda_1, \ldots, \lambda_m)) \geq (2s + 2)$ Using brute force over all $\binom{n}{(2s+2)}$ minors. In the example, an assignment hitting a 6×6 minor

The Algorithm

INPUT: A matrix Q, with entries from $\mathbb{F}[x_1, \ldots, x_m]$, with degrees bounded by d, and the approximation parameter $0 < \epsilon < 1$ (think $d = 2, \epsilon = 2/3$)
OUTPUT: $\lambda_1, \ldots, \lambda_m \in \mathbb{F}$ such that
$\text{rank}(Q(\lambda_1, \ldots, \lambda_m)) \geq (1 - \epsilon)\text{rank}(Q(x_1, \ldots, x_m))$

1. Set $s = \lceil \frac{d}{\epsilon} - 1 \rceil$ (in the example, $s = 2$)
2. Construct the (hitting) set $\mathcal{H}_{m,nd,s}$ of size $O((m(nd+1))^s)$
3. Find an assignment $(\lambda_1, \lambda_2, \ldots, \lambda_m)$ such that $\text{rank}(Q(\lambda_1, \ldots, \lambda_m)) \geq (2s+2)$ Using brute force over all $\binom{n}{(2s+2)}$ minors. In the example, an assignment hitting a 6×6 minor
4. While the rank is increasing,

The Algorithm

INPUT: A matrix Q, with entries from $\mathbb{F}[x_1, \ldots, x_m]$, with degrees bounded by d, and the approximation parameter $0 < \epsilon < 1$ (think $d = 2, \epsilon = 2/3$)
OUTPUT: $\lambda_1, \ldots, \lambda_m \in \mathbb{F}$ such that
$\text{rank}(Q(\lambda_1, \ldots, \lambda_m)) \geq (1 - \epsilon)\text{rank}(Q(x_1, \ldots, x_m))$

1. Set $s = \left\lceil \frac{d}{\epsilon} - 1 \right\rceil$ (in the example, $s = 2$)
2. Construct the (hitting) set $\mathcal{H}_{m,nd,s}$ of size $O((m(nd + 1))^s)$
3. Find an assignment $(\lambda_1, \lambda_2, \ldots, \lambda_m)$ such that $\text{rank}(Q(\lambda_1, \ldots, \lambda_m)) \geq (2s + 2)$ Using brute force over all $\binom{n}{(2s+2)}$ minors. In the example, an assignment hitting a 6×6 minor
4. While the rank is increasing, Check if there exists some $(y_1, \ldots, y_m) \in \mathcal{H}_{m,nd,s}$, such that $\text{rank}(Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m)) > \text{rank}(Q(\lambda_1, \ldots, \lambda_m))$,
5. if $\text{rank}(Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m)) > \text{rank}(Q(\lambda_1, \ldots, \lambda_m))$,

The Algorithm

INPUT: A matrix Q, with entries from $\mathbb{F}[x_1, \ldots, x_m]$, with degrees bounded by d, and the approximation parameter $0 < \epsilon < 1$ (think $d = 2, \epsilon = 2/3$)
OUTPUT: $\lambda_1, \ldots, \lambda_m \in \mathbb{F}$ such that
$\text{rank}(Q(\lambda_1, \ldots, \lambda_m)) \geq (1 - \epsilon)\text{rank}(Q(x_1, \ldots, x_m))$

1. Set $s = \lceil \frac{d}{\epsilon} - 1 \rceil$ (in the example, $s = 2$)
2. Construct the (hitting) set $\mathcal{H}_{m,nd,s}$ of size $O((m(nd+1))^s)$
3. Find an assignment $(\lambda_1, \lambda_2, \ldots, \lambda_m)$ such that $\text{rank}(Q(\lambda_1, \ldots, \lambda_m)) \geq (2s + 2)$ Using brute force over all $\binom{n}{(2s+2)}$ minors. In the example, an assignment hitting a 6×6 minor
4. While the rank is increasing, Check if there exists some $(y_1, \ldots, y_m) \in \mathcal{H}_{m,nd,s}$, such that $\text{rank}(Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m)) > \text{rank}(Q(\lambda_1, \ldots, \lambda_m))$,
5. if $\text{rank}(Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m)) > \text{rank}(Q(\lambda_1, \ldots, \lambda_m))$, update $(\lambda_1, \ldots, \lambda_m)$ to $(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m)$
6. Finally return $(\lambda_1, \ldots, \lambda_m)$.

Understanding the working of the Algorithm

We have found $(\lambda_1, \ldots, \lambda_m)$ such that $\text{rank}(Q(\lambda_1, \ldots, \lambda_m)) = r$

Understanding the working of the Algorithm

We have found $(\lambda_1, \ldots, \lambda_m)$ such that $\text{rank}(Q(\lambda_1, \ldots, \lambda_m)) = r$
We want to find an assignment of the form
$(\lambda_1 + y_1, \lambda_2 + y_2, \ldots ., \lambda_m + y_m)$ such that
$rk(Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots ., \lambda_m + y_m)) > r$
After some preprocessing we can interpret this as,

$$Q(\lambda_1, \lambda_2, \ldots, \lambda_m) = \begin{bmatrix} I_r & 0 \\ 0 & 0 \end{bmatrix}_{n \times n} . \tag{1}$$

Understanding the working of the Algorithm

We have found $(\lambda_1, \ldots, \lambda_m)$ such that $\text{rank}(Q(\lambda_1, \ldots, \lambda_m)) = r$
We want to find an assignment of the form
$(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m)$ such that
$rk(Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m)) > r$
After some preprocessing we can interpret this as,

$$Q(\lambda_1, \lambda_2, \ldots, \lambda_m) = \begin{bmatrix} I_r & 0 \\ 0 & 0 \end{bmatrix}_{n \times n}. \tag{1}$$

And,

$$Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m) = \begin{bmatrix} I_r + L & B \\ A & C \end{bmatrix}_{n \times n}. \tag{2}$$

Here, L, A, B, C are matrices with entries being degree at-most d. None of them are homogeneous, but don't have constant terms.

Rank increasing step

We want to "hit" an $(r+1) \times (r+1)$ minor of $Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m)$.

Rank increasing step

We want to "hit" an $(r+1) \times (r+1)$ minor of $Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m)$.
Consider the minor formed by taking $I_r + L$, the k^{th} row of A, the ℓ^{th} column of B, and also the $(k, \ell)^{\text{th}}$entry of C.

Rank increasing step

We want to "hit" an $(r+1)\times(r+1)$ minor of $Q(\lambda_1+y_1,\lambda_2+y_2,\ldots,\lambda_m+y_m)$.
Consider the minor formed by taking I_r+L, the k^{th} row of A, the ℓ^{th} column of B, and also the $(k,\ell)^{\text{th}}$entry of C.
Denote this by $M_{k,\ell}$. Clearly, $M_{k,\ell}$ looks like below:

$$M_{k,\ell} = \begin{pmatrix} 1+l_{11} & l_{12} & \ldots & l_{1r} & b_1 \\ l_{21} & 1+l_{22} & \ldots & l_{2r} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ l_{r1} & l_{r2} & \ldots & 1+l_{rr} & b_r \\ a_1 & a_2 & \ldots & a_r & c \end{pmatrix}. \quad (3)$$

Rank increasing step

We want to “hit” the following minor

$$M_{k,\ell} = \begin{pmatrix} 1 + l_{11} & l_{12} & \dots & l_{1r} & b_1 \\ l_{21} & 1 + l_{22} & \dots & l_{2r} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ l_{r1} & l_{r2} & \dots & 1 + l_{rr} & b_r \\ a_1 & a_2 & \dots & a_r & c \end{pmatrix}. \tag{4}$$

Rank increasing step

We want to “hit” the following minor

$$M_{k,\ell} = \begin{pmatrix} 1 + l_{11} & l_{12} & \dots & l_{1r} & b_1 \\ l_{21} & 1 + l_{22} & \dots & l_{2r} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ l_{r1} & l_{r2} & \dots & 1 + l_{rr} & b_r \\ a_1 & a_2 & \dots & a_r & c \end{pmatrix}. \tag{4}$$

Perhaps seems as hard as the original problem

Rank increasing step

We want to "hit" the following minor

$$M_{k,\ell} = \begin{pmatrix} 1 + l_{11} & l_{12} & \dots & l_{1r} & b_1 \\ l_{21} & 1 + l_{22} & \dots & l_{2r} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ l_{r1} & l_{r2} & \dots & 1 + l_{rr} & b_r \\ a_1 & a_2 & \dots & a_r & c \end{pmatrix}. \quad (4)$$

Perhaps seems as hard as the original problem

We try to "hit" the low degree components of Determinant of $M_{k,\ell}$.
Concretely, $hom_s(Det(M_{k,\ell}))$, (recall $s \sim \frac{d}{\epsilon}$).

Hitting $hom_s(Det(M_{k,\ell}))$ is easy for small s.

Hitting the low degree components

Let $F_{m,d,s} := \{f \in \mathbb{F}[x_1, \ldots, x_m] \mid \deg(f) \leq d, \text{ord}(f) \leq s\}$, then there is a Hitting set $H_{m,d,s}$ of size $O(m(d+1)^s)$ against $F_{m,d,s}$

Proof sketch: Let $f \in \mathbb{F}_{m,d,s}$. Since $\text{ord}(f) \leq s$, there is a non-zero monomial $x_{i_1} \cdot x_{i_2} \cdots x_{i_s}$ in f.

Hitting the low degree components

Let $F_{m,d,s} := \{f \in \mathbb{F}[x_1, \dots, x_m] \mid \deg(f) \leq d, \text{ord}(f) \leq s\}$, then there is a Hitting set $H_{m,d,s}$ of size $O(m(d+1)^s)$ against $F_{m,d,s}$

Proof sketch: Let $f \in \mathbb{F}_{m,d,s}$. Since $\text{ord}(f) \leq s$, there is a non-zero monomial $x_{i_1} \cdot x_{i_2} \cdots x_{i_s}$ in f.
Brute force search for these s variables (not necessarily distinct) by setting all the other $m - s$ variables to zero.

Hitting the low degree components

Let $F_{m,d,s} := \{f \in \mathbb{F}[x_1, \ldots, x_m] \mid \deg(f) \leq d, \text{ord}(f) \leq s\}$, then there is a Hitting set $H_{m,d,s}$ of size $O(m(d+1)^s)$ against $F_{m,d,s}$

Proof sketch: Let $f \in \mathbb{F}_{m,d,s}$. Since $\text{ord}(f) \leq s$, there is a non-zero monomial $x_{i_1} \cdot x_{i_2} \cdots x_{i_s}$ in f.
Brute force search for these s variables (not necessarily distinct) by setting all the other $m - s$ variables to zero.
Uses $\binom{m}{s} = O(m^s)$ assignments.

Hitting the low degree components

Let $F_{m,d,s} := \{f \in \mathbb{F}[x_1, \ldots, x_m] \mid \deg(f) \leq d, \text{ord}(f) \leq s\}$, then there is a Hitting set $H_{m,d,s}$ of size $O(m(d+1)^s)$ against $F_{m,d,s}$

Proof sketch: Let $f \in \mathbb{F}_{m,d,s}$. Since $\text{ord}(f) \leq s$, there is a non-zero monomial $x_{i_1} \cdot x_{i_2} \cdots x_{i_s}$ in f.
Brute force search for these s variables (not necessarily distinct) by setting all the other $m - s$ variables to zero.
Uses $\binom{m}{s} = O(m^s)$ assignments.
We now have a f' which is a polynomial in s variables of degree at most d.
Can be hit using $(d+1)^s$ assignments (Schwarz Zippel Lemma)

Hitting the low degree components

Let $F_{m,d,s} := \{f \in \mathbb{F}[x_1, \ldots, x_m] \mid \deg(f) \leq d, \text{ord}(f) \leq s\}$, then there is a Hitting set $H_{m,d,s}$ of size $O(m(d+1)^s)$ against $F_{m,d,s}$

Proof sketch: Let $f \in \mathbb{F}_{m,d,s}$. Since $\text{ord}(f) \leq s$, there is a non-zero monomial $x_{i_1} \cdot x_{i_2} \cdots x_{i_s}$ in f.
Brute force search for these s variables (not necessarily distinct) by setting all the other $m - s$ variables to zero.
Uses $\binom{m}{s} = O(m^s)$ assignments.
We now have a f' which is a polynomial in s variables of degree at most d.
Can be hit using $(d+1)^s$ assignments (Schwarz Zippel Lemma)
Thus, a hitting set of size $O(m^s \cdot (d+1)^s) = O((m(d+1))^s)$

The essence of the algorithm

The overall scenario can be reformulated as below. One of the following always happens:

1. For an appropriately chosen s (depending upon d and ϵ), $\exists k, \ell \in [n-r]$ such that $\det(M_{k,\ell})$ has a non-zero monomial of degree at most s. In this case, we can increase the rank (and repeat)
2. $\forall k, \ell \in [n-r]$, $\det(M_{k,\ell})$ has no non-zero monomials of degree at most s. In this case, we show that $r \geq (1-\epsilon) \cdot rk(Q(x_1 \ldots x_m))$.

Heavy lifting

Condition 1

$\forall k, \ell \in [n-r]$, $\det(M_{k,\ell})$ has no non-zero monomials of degree at most s.

Want to show, Condition 1 $\implies r \geq (1-\epsilon) \cdot rk(Q(x_1 \ldots x_m))$.

Taste of analysis: a special case

We look at the case $d = 2, \epsilon = 2/3$

Taste of analysis: a special case

We look at the case $d = 2, \epsilon = 2/3$

We can decompose $Q(\lambda_1 + y_1, \lambda_2 + y_2 \ldots, \lambda_m + y_m)$ as

$$Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m) = \begin{bmatrix} I_r + Q_{11} + L_{11} & Q_{12} + L_{12} \\ Q_{21} + L_{21} & Q_{22} + L_{22} \end{bmatrix}_{n\times n}.$$

Minor of interest $M_{k\ell} =$

$$\begin{pmatrix} 1 + q_{11}(\mathbf{y}) + \ell_{11}(\mathbf{y}) & \ldots & q_{1r}(\mathbf{y}) + \ell_{1r}(\mathbf{y}) & t_1(\mathbf{y}) + b_1(\mathbf{y}) \\ q_{21}(\mathbf{y}) + \ell_{21}(\mathbf{y}) & \ldots & q_{2r}(\mathbf{y}) + \ell_{2r}(\mathbf{y}) & t_2(\mathbf{y}) + b_2(\mathbf{y}) \\ \vdots & \ddots & \vdots & \vdots \\ q_{r1}(\mathbf{y}) + \ell_{12}(\mathbf{y}) & \ldots & 1 + q_{rr}(\mathbf{y}) + \ell_{rr}(\mathbf{y}) & t_r(\mathbf{y}) + b_r(\mathbf{y}) \\ s_1(\mathbf{y}) + a_1(\mathbf{y}) & \ldots & s_r(\mathbf{y}) + a_r(\mathbf{y}) & q(\mathbf{y}) + \ell(\mathbf{y}) \end{pmatrix}.$$

where $q_{ij}(\mathbf{y}), s_i(\mathbf{y}), t_j(\mathbf{y}), q(\mathbf{y})$ are quadratic forms, while $\ell_{ij}(\mathbf{y}), a_i(\mathbf{y}), b_j(\mathbf{y}), \ell(\mathbf{y})$ are linear forms

Taste of analysis: a special case

Suppose $\ell(\mathbf{y}) \neq 0$, observe the following equality:

Taste of analysis: a special case

Suppose $\ell(\mathbf{y}) \neq 0$, observe the following equality:
$Det(M_{k,\ell}) = \ell(\mathbf{y}) +$ monomials of degree at least 2.

Taste of analysis: a special case

Suppose $\ell(\mathbf{y}) \neq 0$, observe the following equality:
$Det(M_{k,\ell}) = \ell(\mathbf{y})+$ monomials of degree at least 2.
Thus one can easily find an assignment $(y_1, \ldots, y_m)$'s such that $det(M_{k,\ell}(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m) \neq 0$ and this assignment increases the rank of Q.

Taste of analysis: a special case

Suppose $\ell(\mathbf{y}) \neq 0$, observe the following equality:
$Det(M_{k,\ell}) = \ell(\mathbf{y}) +$ monomials of degree at least 2.
Thus one can easily find an assignment $(y_1, \ldots, y_m)$'s such that $det(M_{k,\ell}(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m) \neq 0$ and this assignment increases the rank of Q.
Hence, we can assume $\ell(\mathbf{y}) = 0$.
In particular, the matrix $L_{22} = 0$.

Taste of analysis: a special case

Suppose $\ell(\mathbf{y}) \neq 0$, observe the following equality:
$Det(M_{k,\ell}) = \ell(\mathbf{y}) +$ monomials of degree at least 2.
Thus one can easily find an assignment $(y_1, \ldots, y_m)$'s such that $det(M_{k,\ell}(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m) \neq 0$ and this assignment increases the rank of Q.
Hence, we can assume $\ell(\mathbf{y}) = 0$.
In particular, the matrix $L_{22} = 0$.
In this case, observe the following equality.
$det(M_{k,\ell}) = q(\mathbf{y}) - \sum_i^r a_i(\mathbf{y}) \cdot b_i(\mathbf{y}) +$ monomials of degree at least 3.

Taste of analysis: a special case

Suppose $\ell(\mathbf{y}) \neq 0$, observe the following equality:
$Det(M_{k,\ell}) = \ell(\mathbf{y}) +$ monomials of degree at least 2.
Thus one can easily find an assignment $(y_1, \ldots, y_m)$'s such that $det(M_{k,\ell}(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m) \neq 0$ and this assignment increases the rank of Q.
Hence, we can assume $\ell(\mathbf{y}) = 0$.
In particular, the matrix $L_{22} = 0$.
In this case, observe the following equality.
$det(M_{k,\ell}) = q(\mathbf{y}) - \sum_i^r a_i(\mathbf{y}) \cdot b_i(\mathbf{y}) +$ monomials of degree at least 3.
If $q(\mathbf{y}) - \sum_i^r a_i(\mathbf{y}) \cdot b_i(\mathbf{y}) \neq 0$, we can increase the rank.

Taste of analysis: a special case

Suppose $\ell(\mathbf{y}) \neq 0$, observe the following equality:
$Det(M_{k,\ell}) = \ell(\mathbf{y}) +$ monomials of degree at least 2.
Thus one can easily find an assignment $(y_1, \ldots, y_m)$'s such that $det(M_{k,\ell}(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m) \neq 0$ and this assignment increases the rank of Q.
Hence, we can assume $\ell(\mathbf{y}) = 0$.
In particular, the matrix $L_{22} = 0$.
In this case, observe the following equality.
$det(M_{k,\ell}) = q(\mathbf{y}) - \sum_i^r a_i(\mathbf{y}) \cdot b_i(\mathbf{y}) +$ monomials of degree at least 3.
If $q(\mathbf{y}) - \sum_i^r a_i(\mathbf{y}) \cdot b_i(\mathbf{y}) \neq 0$, we can increase the rank.
So, we can assume $q(\mathbf{y}) = \sum_i^r a_i(\mathbf{y}) \cdot b_i(\mathbf{y})$

Taste of analysis: a special case

Suppose $\ell(\mathbf{y}) \neq 0$, observe the following equality:
$Det(M_{k,\ell}) = \ell(\mathbf{y}) +$ monomials of degree at least 2.
Thus one can easily find an assignment $(y_1, \ldots, y_m)$'s such that $det(M_{k,\ell}(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m) \neq 0$ and this assignment increases the rank of Q.
Hence, we can assume $\ell(\mathbf{y}) = 0$.
In particular, the matrix $L_{22} = 0$.
In this case, observe the following equality.
$det(M_{k,\ell}) = q(\mathbf{y}) - \sum_i^r a_i(\mathbf{y}) \cdot b_i(\mathbf{y}) +$ monomials of degree at least 3.
If $q(\mathbf{y}) - \sum_i^r a_i(\mathbf{y}) \cdot b_i(\mathbf{y}) \neq 0$, we can increase the rank.
So, we can assume $q(\mathbf{y}) = \sum_i^r a_i(\mathbf{y}) \cdot b_i(\mathbf{y})$
Thus, $Q_{22} = L_{21} L_{12}$.

1/3-rd approximation

Hence,

$$Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m) = \begin{bmatrix} I_r + Q_{11} + L_{11} & Q_{12} + L_{12} \\ Q_{21} + L_{21} & L_{21} L_{12} \end{bmatrix}_{n \times n}.$$

1/3-rd approximation

Hence,

$$Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m) = \begin{bmatrix} I_r + Q_{11} + L_{11} & Q_{12} + L_{12} \\ Q_{21} + L_{21} & L_{21} L_{12} \end{bmatrix}_{n \times n}.$$

Note that

rank $(Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m) = \text{rank}(Q(x_1, x_2, \ldots, x_m))$

1/3-rd approximation

Hence,

$$Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m) = \begin{bmatrix} I_r + Q_{11} + L_{11} & Q_{12} + L_{12} \\ Q_{21} + L_{21} & L_{21}L_{12} \end{bmatrix}_{n\times n}.$$

Note that

rank $(Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m) = \text{rank}(Q(x_1, x_2, \ldots, x_m))$

Now rank$(L_{21}L_{12}) \leq r$

1/3-rd approximation

Hence,

$$Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m) = \begin{bmatrix} I_r + Q_{11} + L_{11} & Q_{12} + L_{12} \\ Q_{21} + L_{21} & L_{21}L_{12} \end{bmatrix}_{n\times n}.$$

Note that

rank $(Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m) = \text{rank}(Q(x_1, x_2, \ldots, x_m))$

Now rank$(L_{21}L_{12}) \leq r$

We get that the rank$(Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m)) \leq 3r$

1/3-rd approximation

Hence,

$$Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m) = \begin{bmatrix} I_r + Q_{11} + L_{11} & Q_{12} + L_{12} \\ Q_{21} + L_{21} & L_{21}L_{12} \end{bmatrix}_{n \times n}.$$

Note that

rank $(Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m) = \text{rank}(Q(x_1, x_2, \ldots, x_m))$

Now rank$(L_{21}L_{12}) \leq r$

We get that the rank$(Q(\lambda_1 + y_1, \lambda_2 + y_2, \ldots., \lambda_m + y_m)) \leq 3r$

So rank$(Q(\lambda_1, \lambda_2, \ldots, \lambda_m))$ is already a 1/3-approximation of rank$(Q(x_1, x_2, \ldots, x_m))$.

General Case

$$M = \begin{bmatrix} \tilde{L} & B \\ A & C \end{bmatrix}_{n\times n}.$$

General Case

$$M = \begin{bmatrix} \tilde{L} & B \\ A & C \end{bmatrix}_{n \times n}.$$

$$Det(M) = det(C - A\tilde{L}^{-1}B)det(\tilde{L})$$

General Case

$$M = \begin{bmatrix} \tilde{L} & B \\ A & C \end{bmatrix}_{n \times n}.$$

$$Det(M) = det(C - A\tilde{L}^{-1}B)det(\tilde{L})$$

This directly gives,

$$\det(M_{k,\ell}) = -\mathbf{a} \cdot (adj(I_r + L)) \cdot \mathbf{b} + c \cdot (\det(I_r + L)).$$

General Case

$$M = \begin{bmatrix} \tilde{L} & B \\ A & C \end{bmatrix}_{n \times n}.$$

$$Det(M) = det(C - A\tilde{L}^{-1}B)det(\tilde{L})$$

This directly gives,

$$\det(M_{k,\ell}) = -\mathbf{a} \cdot (adj(I_r + L)) \cdot \mathbf{b} + c \cdot (\det(I_r + L)).$$

After staring for sometime,

$$W = -\mathbf{A} \cdot (adj(I_r + L)) \cdot \mathbf{B} + C \cdot (\det(I_r + L)).$$

Here W is the $(n-r) \times (n-r)$ matrix polynomial having the polynomial $\det(M_{u,v})$ as its $(u,v)^{\text{th}}$-entry for all $1 \leq u, v \leq n - r$.

$$W = -\mathbf{A} \cdot (adj(I_r + L)) \cdot \mathbf{B} + C \cdot (\det(I_r + L)).$$

Here W is the $(n-r) \times (n-r)$ matrix polynomial having the polynomial $\det(M_{u,v})$ as its $(u,v)^{\text{th}}$-entry for all $1 \leq u, v \leq n-r$.

$$W = -\mathbf{A} \cdot (adj(I_r + L)) \cdot \mathbf{B} + C \cdot (\det(I_r + L)).$$

Here W is the $(n-r) \times (n-r)$ matrix polynomial having the polynomial $\det(M_{u,v})$ as its $(u,v)^{\mathrm{th}}$-entry for all $1 \le u, v \le n-r$.

Recall that we wanted to analyze $\forall k, \ell \in [n-r]$, $hom_s(\det(M_{k,\ell})) = 0$.

$$W = -\mathbf{A}\cdot(adj(I_r + L)) \cdot \mathbf{B} + C \cdot (\det(I_r + L)).$$

Here W is the $(n-r) \times (n-r)$ matrix polynomial having the polynomial $\det(M_{u,v})$ as its $(u,v)^{\text{th}}$-entry for all $1 \le u, v \le n-r$.

Recall that we wanted to analyze $\forall k, \ell \in [n-r]$, $hom_s(\det(M_{k,\ell})) = 0$.

That is, $hom_s W = 0$!

$$W = -\mathbf{A}\cdot (adj(I_r + L)) \cdot \mathbf{B} + C \cdot (\det(I_r + L)).$$

Here W is the $(n-r) \times (n-r)$ matrix polynomial having the polynomial $\det(M_{u,v})$ as its $(u,v)^{\text{th}}$-entry for all $1 \leq u, v \leq n-r$.

Recall that we wanted to analyze $\forall k, \ell \in [n-r]$, $hom_s(\det(M_{k,\ell})) = 0$.

That is, $hom_s W = 0$!

We finally get

$$W = -A\cdot \left(\sum_{i=0}^{r-1} (-1)^i p_i \cdot \left(\sum_{j=0}^{r-i-1} (-L)^j \right) \right) \cdot B + (p_0 - p_1 + \cdots + (-1)^r p_r) \cdot C.$$

We have to study the $hom_s(W)$.

$$W = -\mathbf{A}\cdot (adj(I_r + L)) \cdot \mathbf{B} + C \cdot (\det(I_r + L)).$$

Here W is the $(n-r) \times (n-r)$ matrix polynomial having the polynomial $\det(M_{u,v})$ as its $(u,v)^{\text{th}}$-entry for all $1 \leq u, v \leq n-r$.

Recall that we wanted to analyze $\forall k, \ell \in [n-r]$, $hom_s(\det(M_{k,\ell})) = 0$.

That is, $hom_s W = 0$!

We finally get

$$W = -A\cdot \left(\sum_{i=0}^{r-1} (-1)^i p_i \cdot \left(\sum_{j=0}^{r-i-1} (-L)^j \right) \right) \cdot B + (p_0 - p_1 + \cdots + (-1)^r p_r) \cdot C.$$

We have to study the $hom_s(W)$.
This work does that!
Finally

Lemma

If $hom_i(W) = 0, \forall i \in [s]$, $rank(Q(x_1, x_2, \ldots, x_m)) \leq r(1 + \frac{s}{s-d+1})$

Conclusion and Open Problems

- Having Vishwas in BJP helps: yields stronger results. Enables us to solve non-linear problems too.

Conclusion and Open Problems

- Having Vishwas in BJP helps: yields stronger results. Enables us to solve non-linear problems too.
- We give a deterministic PTAS for the Commutative rank problem when the entries are polynomials with degrees bounded by a constant d.

Conclusion and Open Problems

- Having Vishwas in BJP helps: yields stronger results. Enables us to solve non-linear problems too.
- We give a deterministic PTAS for the Commutative rank problem when the entries are polynomials with degrees bounded by a constant d.
- Generalizing it to stronger models? - Eg: when entries are given as sparse polynomials, ROABP, formulas or circuits.

Conclusion and Open Problems

- Having Vishwas in BJP helps: yields stronger results. Enables us to solve non-linear problems too.
- We give a deterministic PTAS for the Commutative rank problem when the entries are polynomials with degrees bounded by a constant d.
- Generalizing it to stronger models? - Eg: when entries are given as sparse polynomials, ROABP, formulas or circuits.
- A limitation to this approach: we need PIT for the entries, at least.

Conclusion and Open Problems

- Having Vishwas in BJP helps: yields stronger results. Enables us to solve non-linear problems too.
- We give a deterministic PTAS for the Commutative rank problem when the entries are polynomials with degrees bounded by a constant d.
- Generalizing it to stronger models? - Eg: when entries are given as sparse polynomials, ROABP, formulas or circuits.
- A limitation to this approach: we need PIT for the entries, at least.
- Other models where we know PIT for the entries, eg: Sparse, ROABP? - OPEN?

Conclusion and Open Problems

- Having Vishwas in BJP helps: yields stronger results. Enables us to solve non-linear problems too.
- We give a deterministic PTAS for the Commutative rank problem when the entries are polynomials with degrees bounded by a constant d.
- Generalizing it to stronger models? - Eg: when entries are given as sparse polynomials, ROABP, formulas or circuits.
- A limitation to this approach: we need PIT for the entries, at least.
- Other models where we know PIT for the entries, eg: Sparse, ROABP? - OPEN?
- Approximating algebraic rank over fields of small characteristic?

Conclusion and Open Problems

- Having Vishwas in BJP helps: yields stronger results. Enables us to solve non-linear problems too.
- We give a deterministic PTAS for the Commutative rank problem when the entries are polynomials with degrees bounded by a constant d.
- Generalizing it to stronger models? - Eg: when entries are given as sparse polynomials, ROABP, formulas or circuits.
- A limitation to this approach: we need PIT for the entries, at least.
- Other models where we know PIT for the entries, eg: Sparse, ROABP? - OPEN?
- Approximating algebraic rank over fields of small characteristic?
- No Jacobian criterion! Constant inseparable degree (PSS'16)?

Thanks a lot for attending :)