Modern algorithms for one-block quantifier elimination over the reals

Mohab Safey El Din

SORBONNE UNIVERSITÉ

cnrs

Joint works with Louis Gaillard and Huu Phuoc Le

One block quantifier elimination

$$\exists \boldsymbol{x} \in \mathbb{R}^n \quad f_1 = \cdots = f_p = 0$$
$$g_1 > 0, \ldots, g_s > 0$$
$$\Updownarrow$$
$$\Phi(y) = \Phi_1(y) \vee \cdots \vee \Phi_\ell(y)$$

with

- $\boldsymbol{x} = (x_1, \ldots, x_n)$, $\boldsymbol{y} = (y_1, \ldots, y_t)$
- f_i and g_j in $\mathbb{Q}[\boldsymbol{x}, y]$
- d = max. degree of the input polynomials

One block quantifier elimination

$$\exists \boldsymbol{x} \in \mathbb{R}^n \quad f_1 = \cdots = f_p = 0$$
$$g_1 > 0, \ldots, g_s > 0$$
$$\Updownarrow$$
$$\Phi(\boldsymbol{y}) = \Phi_1(\boldsymbol{y}) \vee \cdots \vee \Phi_\ell(\boldsymbol{y})$$

with

- $\boldsymbol{x} = (x_1, \ldots, x_n)$, $\boldsymbol{y} = (y_1, \ldots, y_t)$
- f_i and g_j in $\mathbb{Q}[\boldsymbol{x}, \boldsymbol{y}]$
- d = max. degree of the input polynomials

$$\exists x \in \mathbb{R} \quad x^2 + bx + c = 0$$
$$\Updownarrow$$
$$b^2 - 4c \geq 0$$

One block quantifier elimination

$$\begin{array}{c} \exists \boldsymbol{x} \in \mathbb{R}^n \quad f_1 = \cdots = f_p = 0 \\ g_1 > 0, \ldots, g_s > 0 \\ \Updownarrow \\ \Phi(\boldsymbol{y}) = \Phi_1(\boldsymbol{y}) \vee \cdots \vee \Phi_\ell(\boldsymbol{y}) \end{array}$$

with

- $\boldsymbol{x} = (x_1, \ldots, x_n)$, $\boldsymbol{y} = (y_1, \ldots, y_t)$
- f_i and g_j in $\mathbb{Q}[\boldsymbol{x}, \boldsymbol{y}]$
- d = max. degree of the input polynomials

$$\begin{array}{c} \exists x \in \mathbb{R} \quad ax^2 + bx + c = 0 \\ \Updownarrow \\ b^2 - 4ac \geq 0 \wedge a \neq 0 \\ \vee \\ a = 0 \wedge b \neq 0 \\ \vee \\ a = 0 \wedge b = 0 \wedge c = 0 \end{array}$$

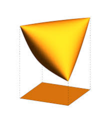
One block quantifier elimination

$$\exists \boldsymbol{x} \in \mathbb{R}^n \quad f_1 = \cdots = f_p = 0$$
$$g_1 > 0, \ldots, g_s > 0$$
$$\Updownarrow$$
$$\Phi(y) = \Phi_1(y) \vee \cdots \vee \Phi_\ell(y)$$

with

- ☛ $\boldsymbol{x} = (x_1, \ldots, x_n)$, $\boldsymbol{y} = (y_1, \ldots, y_t)$
- ☛ f_i and g_j in $\mathbb{Q}[\boldsymbol{x}, y]$
- ☛ d = max. degree of the input polynomials

$$\exists x \in \mathbb{R} \quad ax^2 + bx + c = 0$$
$$\Updownarrow$$
$$b^2 - 4ac \geq 0 \wedge a \neq 0$$
$$\vee$$
$$a = 0 \wedge b \neq 0$$
$$\vee$$
$$a = 0 \wedge b = 0 \wedge c = 0$$



One-block QE is equivalent to compute a description of some projection

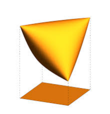
One block quantifier elimination

$$\exists \boldsymbol{x} \in \mathbb{R}^n \quad f_1 = \cdots = f_p = 0$$
$$g_1 > 0, \ldots, g_s > 0$$
$$\Updownarrow$$
$$\Phi(\boldsymbol{y}) = \Phi_1(\boldsymbol{y}) \vee \cdots \vee \Phi_\ell(\boldsymbol{y})$$

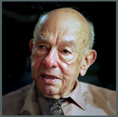
with

- $\boldsymbol{x} = (x_1, \ldots, x_n)$, $\boldsymbol{y} = (y_1, \ldots, y_t)$
- f_i and g_j in $\mathbb{Q}[\boldsymbol{x}, \boldsymbol{y}]$
- d = max. degree of the input polynomials

$$\exists x \in \mathbb{R} \quad ax^2 + bx + c = 0$$
$$\Updownarrow$$
$$b^2 - 4ac \geq 0 \wedge a \neq 0$$
$$\vee$$
$$a = 0 \wedge b \neq 0$$
$$\vee$$
$$a = 0 \wedge b = 0 \wedge c = 0$$



One-block QE is equivalent to compute a description of some projection



The projection of
a semi-algebraic set
is **semi-algebraic (Tarski)**

State of the art

n-variate quantified formula

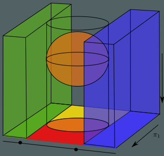
$n-1$-variate quantified formula

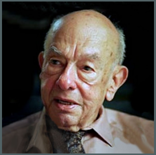
$n-2$-variate quantified formula

⋮

Unquantified formula

Univariate root counting ☛ $d \to d^2$





State of the art

n-variate quantified formula

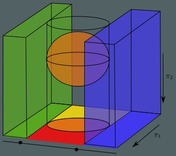
$n-1$-variate quantified formula

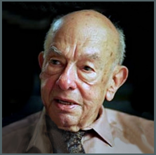
$n-2$-variate quantified formula

$\vdots$

Unquantified formula

Univariate root counting ☛ $d \to d^2$





Cylindrical algebraic decomposition

$$d^{O(2^{n+t})}$$

State of the art

n-variate quantified formula

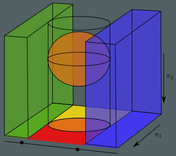
$n-1$-variate quantified formula

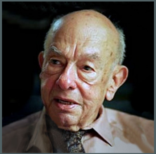
$n-2$-variate quantified formula

$\vdots$

Unquantified formula

Univariate root counting ☛ $d \to d^2$





Cylindrical algebraic decomposition

$$d^{O(2^{n+t})}$$

CAD is at the foundations of all general QE software

State of the art – Modern era starts

$$\exists \boldsymbol{x} \in \mathbb{R}^n \quad f_1 = \cdots = f_p = 0$$
$$g_1 > 0, \ldots, g_s > 0$$
$$\Updownarrow$$
$$\Phi(\boldsymbol{y}) = \Phi_1(\boldsymbol{y}) \vee \cdots \vee \Phi_\ell(\boldsymbol{y})$$

with

- ☛ $\boldsymbol{x} = (x_1, \ldots, x_n)$, $\boldsymbol{y} = (y_1, \ldots, y_t)$
- ☛ f_i and g_j in $\mathbb{Q}[\boldsymbol{x}, \boldsymbol{y}]$
- ☛ d = max. degree of the input polynomials

State of the art – Modern era starts

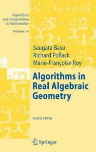
$$\exists \boldsymbol{x} \in \mathbb{R}^n \quad f_1 = \cdots = f_p = 0$$
$$g_1 > 0, \ldots, g_s > 0$$
$$\Updownarrow$$
$$\Phi(\boldsymbol{y}) = \Phi_1(\boldsymbol{y}) \vee \cdots \vee \Phi_\ell(\boldsymbol{y})$$

with

- $\boldsymbol{x} = (x_1, \ldots, x_n)$, $\boldsymbol{y} = (y_1, \ldots, y_t)$
- f_i and g_j in $\mathbb{Q}[\boldsymbol{x}, \boldsymbol{y}]$
- d = max. degree of the input polynomials

There exists an algorithm which performs one block QE using $s^{(n+1)(t+1)} d^{O(nt)}$ arithmetic operations and outputs polynomials of degree bounded by $d^{O(n)}$.

Basu/Pollack/Roy



State of the art – Modern era starts

$$\begin{aligned}\exists \boldsymbol{x} \in \mathbb{R}^n \quad & f_1 = \cdots = f_p = 0 \\ & g_1 > 0, \dots, g_s > 0\end{aligned}$$

$$\Updownarrow$$

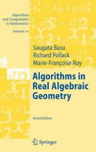
$$\Phi(y) = \Phi_1(y) \vee \cdots \vee \Phi_\ell(y)$$

with

- $\boldsymbol{x} = (x_1, \dots, x_n)$, $y = (y_1, \dots, y_t)$
- f_i and g_j in $\mathbb{Q}[\boldsymbol{x}, y]$
- d = max. degree of the input polynomials

There exists an algorithm which performs one block QE using $s^{(n+1)(t+1)} d^{O(nt)}$ arithmetic operations and outputs polynomials of degree bounded by $d^{O(n)}$.

Basu/Pollack/Roy



Input $\xrightarrow[\text{Algebraic algorithms}]{\text{Sample points}}$ **0-dimensional system** $\xrightarrow[\text{root counting}]{\text{Multivariate}}$ **Formulas**

$\rightsquigarrow \mathbb{Q}(y)$

What we want to do

There exists an algorithm which performs one block QE using $s^{(n+1)(t+1)} d^{O(nt)}$ arithmetic operations and outputs polynomials of degree bounded by $d^{O(n)}$.

What we want to do

There exists an algorithm which performs one block QE using $s^{(n+1)(t+1)}\,d^{O(nt)}$ arithmetic operations and outputs polynomials of degree bounded by $d^{O(n)}$.

? What is hidden by the **Landau notation**?

What we want to do

There exists an algorithm which performs one block QE using $s^{(n+1)(t+1)} d^{O(nt)}$ arithmetic operations and outputs polynomials of degree bounded by $d^{O(n)}$.

? What is hidden by the **Landau notation**?

? How to **better represent semi-algebraic sets**?

☛ Formulas should be easy to evaluate!

☛ We like determinantal representations

What we want to do

There exists an algorithm which performs one block QE using $s^{(n+1)(t+1)}\, d^{O(nt)}$ arithmetic operations and outputs polynomials of degree bounded by $d^{O(n)}$.

? What is hidden by the **Landau notation**?

? How to **better represent semi-algebraic sets**?

☛ Formulas should be easy to evaluate!

☛ We like determinantal representations

? Can we perform **faster in practice** than the best CAD implementations?

Main result

$$\exists \boldsymbol{x} \in \mathbb{R}^n \quad f_1 = \cdots = f_p = 0$$

$$\cancel{g_1 > 0, \ldots, g_s > 0}$$

$$\Updownarrow$$

$$\Phi(\boldsymbol{y}) = \Phi_1(\boldsymbol{y}) \vee \cdots \vee \Phi_\ell(\boldsymbol{y})$$

with

- $\boldsymbol{x} = (x_1, \ldots, x_n)$, $\boldsymbol{y} = (y_1, \ldots, y_t)$
- f_i and g_j in $\mathbb{Q}[\boldsymbol{x}, \boldsymbol{y}]$
- d = max. degree of the input polynomials

Main result

$$\begin{array}{c} \exists \boldsymbol{x} \in \mathbb{R}^n \quad f_1 = \cdots = f_p = 0 \\ \cancel{g_1 > 0, \ldots, g_s > 0} \\ \Updownarrow \\ \Phi(\boldsymbol{y}) = \Phi_1(\boldsymbol{y}) \vee \cdots \vee \Phi_\ell(\boldsymbol{y}) \end{array}$$

with

- $\boldsymbol{x} = (x_1, \ldots, x_n)$, $\boldsymbol{y} = (y_1, \ldots, y_t)$
- f_i and g_j in $\mathbb{Q}[\boldsymbol{x}, \boldsymbol{y}]$
- $d =$ max. degree of the input polynomials

- **regularity / transversality assumptions on** $f_1, \ldots, f_p$
- **output defines a Zariski dense subset of the projection of** $V_{\mathbb{R}}(f_1, \ldots, f_p)$

Main result

$$\begin{array}{c}\exists \boldsymbol{x} \in \mathbb{R}^n \quad f_1 = \cdots = f_p = 0 \\ \cancel{g_1 > 0, \ldots, g_s > 0} \\ \Updownarrow \\ \Phi(\boldsymbol{y}) = \Phi_1(\boldsymbol{y}) \vee \cdots \vee \Phi_\ell(\boldsymbol{y})\end{array}$$

with

- $\boldsymbol{x} = (x_1, \ldots, x_n)$, $\boldsymbol{y} = (y_1, \ldots, y_t)$
- f_i and g_j in $\mathbb{Q}[\boldsymbol{x}, \boldsymbol{y}]$
- $d =$ max. degree of the input polynomials

- **regularity / transversality assumptions on** $f_1, \ldots, f_p$
- **output defines a Zariski dense subset of the projection of** $V_{\mathbb{R}}(f_1, \ldots, f_p)$

There exists a **randomized** algorithm that performs (weak) one block QE using $O\left(8^t \mathscr{D}^{3t}\binom{t+\mathscr{D}}{t}\right)$ arithmetic operations with $\mathscr{D} = nd^p(d-1)^{n-p+1}\binom{n}{p-1}$ and with output polynomials of degree bounded by $\mathscr{D}$, for generic entries.

Main result

$$\exists \boldsymbol{x} \in \mathbb{R}^n \quad f_1 = \cdots = f_p = 0$$
$$\cancel{g_1 > 0, \ldots, g_s > 0}$$
$$\Updownarrow$$
$$\Phi(\boldsymbol{y}) = \Phi_1(\boldsymbol{y}) \vee \cdots \vee \Phi_\ell(\boldsymbol{y})$$

with

- ☛ $\boldsymbol{x} = (x_1, \ldots, x_n)$, $\boldsymbol{y} = (y_1, \ldots, y_t)$
- ☛ f_i and g_j in $\mathbb{Q}[\boldsymbol{x}, \boldsymbol{y}]$
- ☛ d = max. degree of the input polynomials

- **regularity / transversality assumptions on** $f_1, \ldots, f_p$
- **output defines a Zariski dense subset of the projection of** $V_{\mathbb{R}}(f_1, \ldots, f_p)$

There exists a **randomized** algorithm that performs (weak) one block QE using $O\left(8^t \mathscr{D}^{3t} \binom{t+\mathscr{D}}{t}\right)$ arithmetic operations with $\mathscr{D} = nd^p(d-1)^{n-p+1}\binom{n}{p-1}$ and with output polynomials of degree bounded by $\mathscr{D}$, for generic entries.

✓ output polynomials are given as **minors of some matrices**

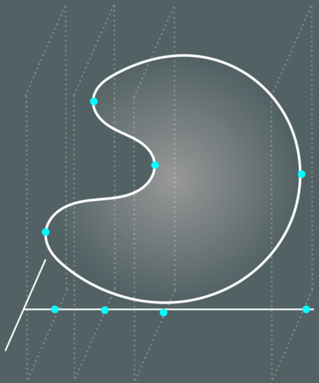
preliminary implementation → **solves problems previously unreachable**

First ingredients (I)

$$\textbf{Input} \xrightarrow[\text{Algebraic algorithms}]{\text{Sample points}} \boxed{\begin{array}{c}0\text{-}\textbf{dimensional}\\ \textbf{system}\end{array}} \xrightarrow[\text{root counting}]{\text{Multivariate}} \textbf{Formulas}$$

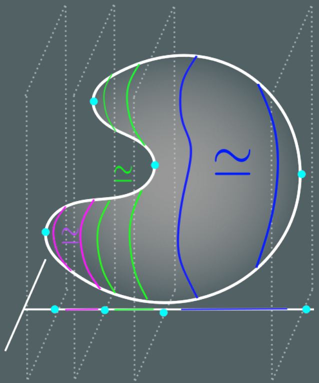
First ingredients (I)

Input $\xrightarrow[\text{Algebraic algorithms}]{\text{Sample points}}$ 0-dimensional system $\xrightarrow[\text{root counting}]{\text{Multivariate}}$ Formulas



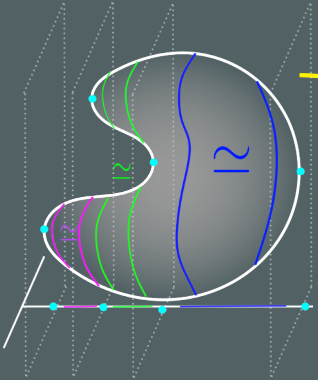
First ingredients (I)

Input $\xrightarrow[\text{Algebraic algorithms}]{\text{Sample points}}$ 0-dimensional system $\xrightarrow[\text{root counting}]{\text{Multivariate}}$ Formulas



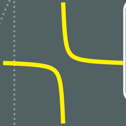
First ingredients (I)

Input $\xrightarrow[\text{Algebraic algorithms}]{\text{Sample points}}$ 0-dimensional system $\xrightarrow[\text{root counting}]{\text{Multivariate}}$ Formulas



S./Schost'03

Up to a **generic linear change of coordinates** the projection on the $(x_1, \ldots, x_i)$-subspace of $V_{\mathbb{R}}$ is **closed**.

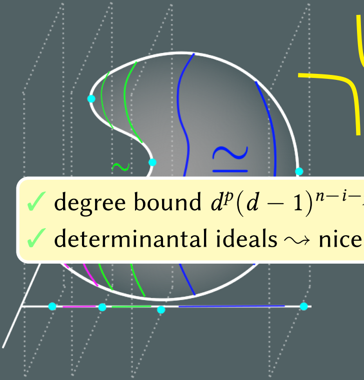


$$f_1 = \cdots = f_p = x_1 = \cdots = x_{i-1} = 0$$

$$\operatorname{rk} \begin{bmatrix} \frac{\partial f_1}{\partial x_{i+1}} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_p}{\partial x_{i+1}} & \cdots & \frac{\partial f_p}{\partial x_n} \end{bmatrix} < p$$

First ingredients (I)

$$\boxed{\text{Input}} \xrightarrow[\text{Algebraic algorithms}]{\text{Sample points}} \boxed{\text{0-dimensional system}} \xrightarrow[\text{root counting}]{\text{Multivariate}} \boxed{\text{Formulas}}$$



S./Schost'03

Up to a **generic linear change of coordinates** the projection on the $(x_1, \dots, x_i)$-subspace of $V_{\mathbb{R}}$ is **closed**.

$$f_1 = \cdots = f_p = x_1 = \cdots = x_{i-1} = 0$$

$$\operatorname{rk} \begin{bmatrix} \frac{\partial f_1}{\partial x_{i+1}} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_p}{\partial x_{i+1}} & \cdots & \frac{\partial f_p}{\partial x_n} \end{bmatrix} < p$$

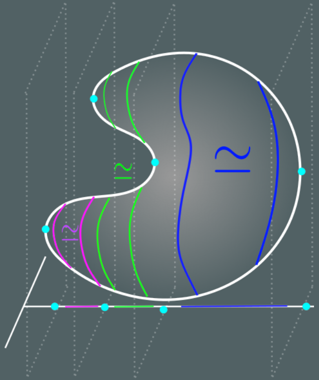
✓ degree bound $d^p(d-1)^{n-i-p}\binom{n-i-1}{p-1}$

✓ determinantal ideals $\rightsquigarrow$ nice properties

First ingredients (II)

Input $\xrightarrow[\text{Algebraic algorithms}]{\text{Sample points}}$ 0-dimensional system $\xrightarrow[\text{root counting}]{\text{Multivariate}}$ Formulas

$\rightsquigarrow \mathbb{Q}(y)$

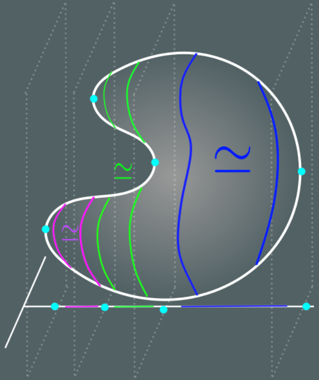


First ingredients (II)

Input $\xrightarrow[\text{Algebraic algorithms}]{\text{Sample points}}$ **0-dimensional system** $\xrightarrow[\text{root counting}]{\text{Multivariate}}$ **Formulas**

$\rightsquigarrow \mathbb{Q}(\boldsymbol{y})$

$$f_1 = \cdots = f_p = x_1 = \cdots = x_{i-1} = 0$$
$$\mathrm{rk}\begin{bmatrix} \frac{\partial f_1}{\partial x_{i+1}} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_p}{\partial x_{i+1}} & \cdots & \frac{\partial f_p}{\partial x_n} \end{bmatrix} < p$$



Parametrized systems of critical points
zero-dimenional + determinantal structure

With parameters, we need a specialization theorem

First ingredients (II)

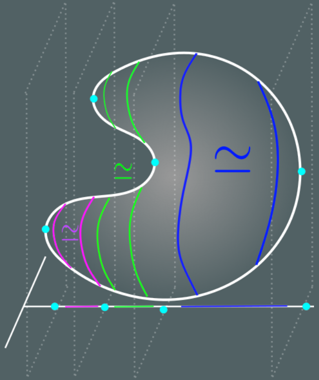
$$\begin{tikzcd}[column sep=huge] \boxed{\text{Input}} \arrow[r, "\text{Sample points}", "\text{Algebraic algorithms}"'] & \boxed{\text{0-dimensional system}} \arrow[r, "\text{Multivariate}", "\text{root counting}"'] & \boxed{\text{Formulas}} \end{tikzcd}$$

$\rightsquigarrow \mathbb{Q}(y)$

$$f_1 = \cdots = f_p = x_1 = \cdots = x_{i-1} = 0$$

$$\mathrm{rk} \begin{bmatrix} \frac{\partial f_1}{\partial x_{i+1}} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_p}{\partial x_{i+1}} & \cdots & \frac{\partial f_p}{\partial x_n} \end{bmatrix} < p$$

Parametrized systems of critical points
zero-dimenional + determinantal structure



With parameters, we need a specialization theorem

One generic linear change of variables is ok for almost all parameter's values.

Second ingredients

$$\boxed{\text{Input}} \xrightarrow[\text{Algebraic algorithms}]{\text{Sample points}} \boxed{0\text{-dimensional system}} \xrightarrow[\text{root counting}]{\text{Multivariate}} \boxed{\text{Formulas}}$$

$\rightsquigarrow \mathbb{Q}(y)$

Second ingredients

Input $\xrightarrow[\text{Algebraic algorithms}]{\text{Sample points}}$ 0-dimensional system $\xrightarrow[\text{root counting}]{\text{Multivariate}}$ Formulas

$\rightsquigarrow \mathbb{Q}(y)$

Parametrized system with finitely many solutions in $\overline{\mathbb{Q}(y)}$

Let I be the ideal generated by these equations

- ☛ The quotient ring $\mathbb{A} = \frac{\mathbb{Q}(y)[x]}{I}$ allows to compute "modulo" the solutions
- ☛ The quotient ring $\mathbb{A} = \frac{\mathbb{Q}(y)[x]}{I}$ is a finite dimensional vector space

Gröbner bases

A confluent rewriting system

Let $\mathscr{B} = \{b_1, \ldots, b_\delta\}$ be a basis of $\mathbb{A}$

Second ingredients

Input —Sample points / Algebraic algorithms $\rightsquigarrow \mathbb{Q}(y)$→ **0-dimensional system** —Multivariate root counting→ **Formulas**

Parametrized system with finitely many solutions in $\overline{\mathbb{Q}(y)}$

Let I be the ideal generated by these equations

☛ The quotient ring $\mathbb{A} = \frac{\mathbb{Q}(y)[x]}{I}$ allows to compute "modulo" the solutions

☛ The quotient ring $\mathbb{A} = \frac{\mathbb{Q}(y)[x]}{I}$ is a finite dimensional vector space

Gröbner bases

A confluent rewriting system

Let $\mathscr{B} = \{b_1, \ldots, b_\delta\}$ be a basis of $\mathbb{A}$

$$\mu_g : f \in \mathbb{A}_{\mathbb{Q}} \to f \times g \in \mathbb{A}_{\mathbb{Q}}$$

$$\mathscr{H}_{\mathbb{Q}} : (f, g) \in \mathbb{A}_{\mathbb{Q}}^2 \to \mathrm{Tr}(\mu_{f.g}) \rightsquigarrow \text{matrix representation } H_{\mathscr{B}}$$

☛ the rank of $\mathscr{H}_{\mathbb{Q}}$ is the number of complex roots

☛ the **signature** of $\mathscr{H}_{\mathbb{Q}}$ is the number of real roots

Second ingredients

$$\boxed{\text{Input}} \xrightarrow[\text{Algebraic algorithms}]{\text{Sample points}} \boxed{\begin{array}{c}0\text{-dimensional}\\ \text{system}\end{array}} \xrightarrow[\text{root counting}]{\text{Multivariate}} \boxed{\text{Formulas}}$$

$\rightsquigarrow \mathbb{Q}(\boldsymbol{y})$

$\boxed{\text{Hermite matrix } \mathscr{H}_{\mathbb{K}} \text{ computed over } \mathbb{K} = \overline{\mathbb{Q}(\boldsymbol{y})}}$

Second ingredients

Input $\xrightarrow[\text{Algebraic algorithms}]{\text{Sample points}}$ **0-dimensional system** $\xrightarrow[\text{root counting}]{\text{Multivariate}}$ **Formulas**

$\rightsquigarrow \mathbb{Q}(\boldsymbol{y})$

Hermite matrix $\mathscr{H}_{\mathbb{K}}$ computed over $\mathbb{K} = \overline{\mathbb{Q}(\boldsymbol{y})}$

Sample points in semi-algebraic sets

Second ingredients

Input $\xrightarrow[\text{Algebraic algorithms}]{\text{Sample points}}$ **0-dimensional system** $\xrightarrow[\text{root counting}]{\text{Multivariate}}$ **Formulas**

$\rightsquigarrow \mathbb{Q}(y)$

Hermite matrix $\mathscr{H}_{\mathbb{K}}$ computed over $\mathbb{K} = \overline{\mathbb{Q}(y)}$

Sample points in semi-algebraic sets

- **Specialization property** for $\mathscr{H}_{\mathbb{K}}$ and $H_{\mathbb{K}}$
- When I is generated by **generic** polynomials,
$$\deg(H_{\mathbb{K}}[i,j]) \leq \deg(b_i) + \deg(b_j)$$
- Hilbert series + graded orderings
  all principal minors have degree $n(d-1)d^n$

Second ingredients

Input $\xrightarrow[\text{Algebraic algorithms}]{\text{Sample points}}$ **0-dimensional system** $\xrightarrow[\text{root counting}]{\text{Multivariate}}$ **Formulas**

$\rightsquigarrow \mathbb{Q}(y)$

Hermite matrix $\mathscr{H}_{\mathbb{K}}$ computed over $\mathbb{K} = \overline{\mathbb{Q}(y)}$

Sample points in semi-algebraic sets

- ☛ **Specialization property** for $\mathscr{H}_{\mathbb{K}}$ and $H_{\mathbb{K}}$
- ☛ When I is generated by **generic** polynomials,

$$\deg(H_{\mathbb{K}}[i,j]) \leq \deg(b_i) + \deg(b_j)$$

- ☛ Hilbert series + graded orderings

all principal minors have degree $n(d-1)d^n$

Our ideals are **NOT** generic

Second ingredients

Input $\xrightarrow[\text{Algebraic algorithms}]{\text{Sample points}}$ **0-dimensional system** $\xrightarrow[\text{root counting}]{\text{Multivariate}}$ **Formulas**

$\rightsquigarrow \mathbb{Q}(y)$

Hermite matrix $\mathscr{H}_{\mathbb{K}}$ computed over $\mathbb{K} = \overline{\mathbb{Q}(y)}$

Sample points in semi-algebraic sets

- **Specialization property** for $\mathscr{H}_{\mathbb{K}}$ and $H_{\mathbb{K}}$
- When I is generated by **generic** polynomials,
  $$\deg(H_{\mathbb{K}}[i,j]) \leq \deg(b_i) + \deg(b_j)$$
- Hilbert series + graded orderings
  all principal minors have degree $n(d-1)d^n$

Our ideals are **NOT** generic

The combinatorics of determinantal ideals

Berthomieu/Bostan/Ferguson/S.'21

Hilbert series $\left(\sum_{i=0}^{d-1} z^i\right)^p \left(\sum_{i=0}^{d-2} z^i\right)^{n-p} \sum_{i=0}^{p-1} \binom{n-p-1+i}{i} z^{(d-1)i}$

Practical results **maple**, RAGLIB, **msolve**

t	n	s	D	MAPLE[QE]	MATHEMATICA[QE]	MAT+DET	SP	**total**	DEG
2	3	2	2	> 10d	> 10d				
2	4	2	2	> 10d	> 10d				
2	5	2	2	> 10d	> 10d				
2	6	2	2	> 10d	> 10d				
3	3	2	2	> 10d	> 10d				
3	4	2	2	> 10d	> 10d				
3	5	2	2	> 10d	> 10d				

Random dense systems

MAT +DET: compute Hermite matrices + minors DEG: highest degree in outputs
SP: compute sample points

Practical results **maple**, RAGlib, **msolve**

t	n	s	D	MAPLE[QE]	MATHEMATICA[QE]	MAT+DET	SP	**total**	DEG
2	3	2	2	> 10d	> 10d	1s	6s	**7s**	24
2	4	2	2	> 10d	> 10d	9s	2m	**2m**	40
2	5	2	2	> 10d	> 10d	2m	23m	**25m**	56
2	6	2	2	> 10d	> 10d	20m	6.5h	**7h**	72
3	3	2	2	> 10d	> 10d	6s	3m	**3m**	24
3	4	2	2	> 10d	> 10d	2m	43m	**45m**	40
3	5	2	2	> 10d	> 10d	1h	14h	**15h**	56

Random dense systems

MAT +DET: compute Hermite matrices + minors DEG: highest degree in outputs

SP: compute sample points

Practical results **maple**, RAGLIB, **msolve**

t	n	s	D	MAPLE[QE]	MATHEMATICA[QE]	MAT+DET	SP	**total**	DEG
2	3	2	2	> 10d	> 10d	1s	6s	**7s**	24
2	4	2	2	> 10d	> 10d	9s	2m	**2m**	40
2	5	2	2	> 10d	> 10d	2m	23m	**25m**	56
2	6	2	2	> 10d	> 10d	20m	6.5h	**7h**	72
3	3	2	2	> 10d	> 10d	6s	3m	**3m**	24
3	4	2	2	> 10d	> 10d	2m	43m	**45m**	40
3	5	2	2	> 10d	> 10d	1h	14h	**15h**	56

Random dense systems

t	n	s	D	MAPLE	MATHEMATICA	MAT+DET	SP	**total**	DEG
3	4	2	2	> 10d	> 10d	2 m.	10 m.	**12 m**	34
3	5	2	2	> 10d	> 10d	2 m.	10 m.	**12 m**	32
4	3	2	2	> 10d	> 10d	20 s.	20 m.	**21 m**	22
4	4	2	2	> 10d	> 10d	15 s.	18 m.	**19 m**	20

Random sparse systems

MAT +DET: compute Hermite matrices + minors DEG: highest degree in outputs

SP: compute sample points

Conclusions and perspectives

✓ new complexity results: probabilistic algorithm, genericity assumptions

✓ practical performances that reflect the complexity gains

✓ solves applications that were previously out of reach

✓ generalization to the case involving inequalities

Conclusions and perspectives

✓ new complexity results: probabilistic algorithm, genericity assumptions

✓ practical performances that reflect the complexity gains

✓ solves applications that were previously out of reach

✓ generalization to the case involving inequalities

✗ how to compute faster sample points in semi-algebraic sets defined by determinantal representations?

✗ how to remove the genericity assumptions?

✗ how to remove probabilistic aspects?