

# An Introduction on Polynomial System Solving

**#1**

Setting and basic properties

System  $\mathcal{P}$  of polynomial equations

$$f_1 = 0, \dots, f_m = 0$$

where  $f_i \in k[x_1, \dots, x_n]$  for some field  $k$ .

Throughout the talk we (hopefully) assume  
 $m$  **equations** in  $n$  **variables**.

Solutions? Go to algebraic closure  $K$  of  $k$

Let  $I = \langle f_1, \dots, f_m \rangle \subset R := K[x_1, \dots, x_n]$ .

Compute

$$\begin{aligned} V(I) &= \{z \in K^n \mid f_i(z) = 0 \text{ for } i = 1, \dots, m\} \\ &= \{z \in K^n \mid f(z) = 0 \forall f \in I\}. \end{aligned}$$

$\mathcal{P}$  is **inconsistent** if it has no solution.

$\mathcal{P}$  is **overdetermined** if  $m > n$ .

Not all overdetermined systems are inconsistent, e.g.

$\mathcal{P} = (x^3 - 1 = 0, x^2 - 1 = 0)$  has solution  $x = 1$ .

$\mathcal{P}$  is **underdetermined** if  $m < n$ .

An underdetermined system is either inconsistent or it has infinitely many solutions.

$\mathcal{P}$  is **positive-dimensional** if it has infinitely many solutions.

$\mathcal{P}$  is **zero-dimensional** if it has finitely many solutions.

(Corresponds to  $\dim V(I) = 0$ .)

Let  $\mathcal{P}$  be zero-dimensional and  $m = n$ :

**Bézout's theorem** gives us:

$\deg f_i = d_i \Rightarrow$  at most  $\prod_{i=1}^m d_i$  solutions.

Bound is sharp and exponential in number of variables.

In general: **solving is difficult.**

What does solving mean?



If  $\mathcal{P}$  is **positive-dimensional** then counting solutions is meaningless.

Try to find a description of the solutions from which we can **easily** extract the **relevant data**.

Algebraic geometry, here we go!

*“Does  $\mathcal{P}$  over  $\mathbb{Q}$  has a finite number of **real** solutions?  
If so, compute them.”*

## Cylindrical algebraic decomposition (CAD):

Complexity: doubly exponential in  $n$ .

A **semi-algebraic set / cell** is a finite union of subsets of  $\mathcal{R}^n$  ( $\mathcal{R}$  is a real closed field) defined by a finite sequence of polynomial equations or inequalities.

A CAD is a decomposition of  $\mathcal{R}^n$  into connected semi-algebraic sets on which each polynomial has constant sign  $(+, -, 0)$ .

When projecting  $\pi: \mathcal{R}^n \rightarrow \mathcal{R}^{n-k}$  then for cells  $C$  and  $D$ , either  $\pi(C) = \pi(D)$  or  $\pi(C) \cap \pi(D) = \emptyset$ .

$\Rightarrow$  Images of  $\pi$  give CAD of  $\mathcal{R}^{n-k}$ .

## Algorithmic idea

Sequence of **projections**  $\mathcal{R}^n \rightarrow \mathcal{R}^{n-1} \rightarrow \dots \rightarrow \mathcal{R}$ .

Take  $f = \prod_{i=1}^m f_i$ , let  $g = \gcd(f, f')$  (w.r.t.  $x_n$ ).

Zeroes of  $g$  and intersections of  $f_i$  give cell boundaries  
(no local variation of  $f = 0$  when perturbing  $x_n$ ).

Zeroes of univariate polynomials provide critical points for  
cell decomposition of  $\mathcal{R}$  in zero- and one-dimensional cells.

**Lift** them back up, get a cylinder of cells from  $\mathcal{R}$  in  $\mathcal{R}^2$ .

Go on till  $\mathcal{R}^n$ .

Let's restrict to **zero-dimensional** systems in the following.

Solving means to compute all solutions.

There are two main ways to output the solutions:

## Numerical representation

For real/complex solutions one in general uses numeric approximations.

A **certified** solution provides a bound on the error of the approximations in order to separate the different solutions.

## Algebraic representation

Several different ways (we talk about them).

All boil down to a representation of the solution set by **univariate** equations.

Then compute a numerical approximation of the solutions by solving this univariate system.



#2

Numerical solving – quick & dirty

One can use general solvers for **non-linear** systems.

## Problems

- ▷ In general one cannot find all solutions.
- ▷ If the method does not find a solution there is no certificate that there really exists no solution.

## Notably mentions

- ▷ Newton's method (fast if we start near a solution)
- ▷ Optimization (meh)

## **Homotopy continuation method**

Semi-numerical, supposes  $m = n$ .

The algorithm consists of four main steps:

## Step 1

An **upper bound** on the number of solutions is computed.

This step is critical, the bound  $B$  should be as sharp as possible.

## Step 2

Another polynomial system

$$g_1 = 0, \dots, g_n = 0$$

is generated with exactly  $B$   
**easily computable** solutions.

### Step 3

We construct a **homotopy** between both systems:

$$(1 - t)g_1 + tf_1 = 0, \dots, (1 - t)g_n + tf_n = 0.$$

Not only straight lines, but also other paths, in order to avoid singularities and other trouble.

## Step 4

Now we **follow** the solutions of the  $g_i$ s ( $t = 0$ ) to the  $f_i$ s ( $t = 1$ ).

If  $t_k < t_l$  then we get the solutions for  $t = t_l$  from those for  $t_k$  using **Newton's** method.

**Difficult task:** How to choose  $t_l - t_k$ ?

- ▷ If **too large**, convergence is too slow, even jumps from one solution path to a different one is possible.
- ▷ If **too small**, then too many steps may slow down the computation.



There is a recent paper by **Verschelde** on using parallel approaches.

## **Main idea**

Different solution paths are independent of each other.

**#3**

Algebraic representations of solutions

A **triangular set** is a

non-empty set  $T = \{g_1, \dots, g_s\} \subset K[x_1, \dots, x_n]$  such that

- ▷ no  $g_i$  is constant,
- ▷ all main variables are different,
- ▷  $|T| \leq n$ .

(The main variable  $\text{mvar}(g)$  of a poly  $g$  is the greatest appearing variable.)

## A regular chain

$T = \{g_1, \dots, g_s\}$  is a triangular set such that

▷  $\text{mvar}(g_1) < \dots < \text{mvar}(g_s)$ .

▷ Let  $h = \prod_{i=1}^s \text{lm}(g_i)$ . Then  $\text{resultant}(h, T) \neq 0$  where each internal resultant is computed w.r.t. the main variable of  $g_i$ .

## **Main idea by Kalkbrenner**

Every irreducible variety is uniquely determined by one of its generic points.

Regular chains give us exactly these generic points.

## Example

Take  $R = \mathbb{Q}[x, y, z]$  such that  $x < y < z$ .

Then  $T = \{y^2 - x^2, y(z - x)\}$  is a triangular set and a regular chain.

Two generic points given by  $T$  are  $(t, t, t)$  and  $(t, -t, t)$  for  $t$  transcendental over  $\mathbb{Q}$ .

Thus we have two irreducible components:  
 $\{y - x, z - x\}$  and  $\{y + x, z - x\}$ .

## **Note**

$y$  is the content of the second polynomial of  $T$  and can be removed.

The dimension of each component is one, the number of free variables.

Let  $T$  be a regular chain.

The **quasi-component** of  $T$ :  $W(T) = V(T) \setminus V(\mathfrak{h})$ .

Corresponding algebraic structure:

The **saturated ideal**  $\text{sat}(T) = (T) : \mathfrak{h}^\infty$ .

We have  $\overline{W(\mathfrak{t})} = V(\text{sat}(T))$ .



Some properties of a regular chain  $T$ :

- ▷  $\text{sat}(T)$  is an unmixed ideal of dimension  $n - |T|$ .
- ▷  $\text{sat}(T \cap K[x_1, \dots, x_i]) = \text{sat}(T) \cap K[x_1, \dots, x_i]$ .
- ▷ A triangular set is a regular chain iff it is *Ritt characteristic set* of its saturated ideal.

**Triangular decomposition** of a polynomial system  $\mathcal{P}$ :

**Kalkbrenner** style, lazy decomposition:

$$\sqrt{(\mathcal{P})} = \cap_{i=1}^k \sqrt{\text{sat}(T_i)}.$$

**Lazard** style, describe all zeroes:

$$V(\mathcal{P}) = \cup_{i=1}^k W(T_i).$$

## Zero-dimensional **regular chains**

Sequence of polys  $g_1(x_1), g_2(x_1, x_2), \dots, g_n(x_1, \dots, x_n)$   
such that for all  $1 \leq i \leq n$

- ▷  $g_i$  poly in  $x_1, \dots, x_i$  such that  $d_{x_i} := \deg_{x_i} g_i > 0$ .
- ▷ Coefficient of  $x_i^{d_{x_i}}$  is a poly in  $x_1, \dots, x_{i-1}$   
that has no common zero with  $g_1, \dots, g_{i-1}$ .

Thus we have a **triangular system**

$$g_1(x_1) = 0$$

$$g_2(x_1, x_2) = 0$$

...

$$g_n(x_1, \dots, x_n) = 0.$$

Solve first equation, make thus second univariate, ...

Working over a finite field this is wonderful.

Over the rationals?

## Problem 1

Coefficients might explode.

### Idea

**Equiprojectable decomposition** by **Dahan** and **Schost**

- ▷ Bound on coefficients w.r.t. size of the input system.
- ▷ Depends only on choice of coordinates.
- ▷ Allows modular computation.

## Problem 2

Solving univariate polys with approximate coefficients is quite unstable.

## Ideas

Get regular chains in special form: **shape lemma**.

Use **rational univariate representation** starting from a general regular chain or a Gröbner basis.



## Shape lemma

Up to a linear change of coordinates any zero-dimensional radical ideal  $I$  has a LEX Gröbner basis in **shape position**, i.e.

$$G = \{x_1 - h_1(x_n), \dots, x_{n-1} - h_{n-1}(x_n), h_n(x_n)\},$$

such that

- ▷  $D = \dim_{\mathbb{K}} (\mathbb{R}/I)$ ,
- ▷  $\deg h_n = D$  and
- ▷  $\deg h_i < D$  for all  $1 \leq i < n$ .

## Rational univariate representation (RUR) by Rouillier

Connected to the shape lemma.

Uses **separating variable**  $t$ , a linear combination of the other variables.

We get a system

$$h(t) = 0,$$

$$x_1 = h_1(t)/q(t),$$

...

$$x_n = h_n(t)/q(t),$$

where  $D = \deg h$  and  $\deg q, \deg h_i < D$ .

## Example

Let  $\mathcal{P} = \{x^2 - 1, (x - 1)(y - 1), y^2 - 1\}$ .

Besides  $\lambda x$ ,  $\lambda y$ , and  $\lambda(x + y)$  we can use any linear combination of  $x$  and  $y$  as separating variable. For example, take  $t = \frac{x-y}{2}$ . Then we get as RUR

$$t^3 - t = 0, \quad x = \frac{t^2 + 2t - 1}{3t^2 - 1}, \quad y = \frac{t^2 - 2t - 1}{3t^2 - 1}.$$

## Properties of a RUR

- ▷ Only defined in the zero-dimensional case.
- ▷ Only finitely many linear combinations do **not** lead to a separating variable.
- ▷ Once a separating variable is chosen **the** corresponding RUR exists and is unique.
- ▷ 1-to-1 correspondence between roots of  $h$  and solutions of the system. (multiplicities coincide; triangular decompositions in general do not preserve this information.)
- ▷ If  $h$  has no multiple root then  $q = h'$ .

Factorizing  $\mathfrak{h}$  gives a RUR for each irreducible factor.

We get a **prime decomposition**

i.e. primary decomposition of the radical.

Especially if  $\mathcal{P}$  has a high multiplicity we thus get an output with much smaller coefficients.

Getting a **RUR** from a LEX Gröbner basis:

If  $I$  is **radical**, take smallest variable from LEX GB as separating variable  $t$ .

Check that  $h(t)$  is squarefree and get a RUR.

In the general case, there also exist algorithms.

If the separating variable is already known **and** if the multiplication matrices are already given then we can compute a RUR in  $O(D^3 + nD^2)$ .

#4

Numerical solving once having the RUR



Seems easy, but evaluating one poly at the roots of another one is highly unstable.

Compute roots of  $h$  with high precision.  
(This may change for different roots).

- ▷ **Aberth's** method,
- ▷ **Laguerre's** method (Singular),
- ▷ other improved algorithms by Rouillier, Zimmermann, etc.,
- ▷ other algorithms I know nothing about.

## Laguerre's method

Find approximation for **one** root of a polynomial  $f(x)$  of degree  $d$ :

Initial guess  $z_0$ .

For  $k = 0, 1, 2, \dots$ , some upper bound

If  $f(z_k)$  is small enough, exit loop.

$$G = f'(z_k)/f(z_k).$$

$$H = G^2 - f''(z_k)/f(z_k).$$

$$\alpha = d / \left( G \pm \sqrt{(d-1)(dH - G^2)} \right) \quad (\text{Choose sign to get bigger absolute value of denominator.})$$

$$z_{k+1} = z_k - \alpha.$$

## Aberth's method

Find approximation for **all** roots of a polynomial  $f(x) \in \mathbb{C}[x_1, \dots, x_n]$  of degree  $d$  simultaneously:

Compute upper and lower bounds of absolute values for the  $d$  roots from the coefficients of the polynomial.

Now pick randomly or evenly distributed **distinct** complex numbers  $z_1, \dots, z_d$  with absolute values within the same bounds.

For some number of iterations / until values are small enough do:

For current approximations  $z_1, \dots, z_d$  compute

$$w_k = -\frac{\frac{f(z_k)}{f'(z_k)}}{1 - \frac{f(z_k)}{f'(z_k)} \cdot \sum_{l \neq k} \frac{1}{z_k - z_l}}.$$

Calculate next approximations  $z'_k = z_k + w_k$  for all  $1 \leq k \leq d$ .

Both methods share the following properties:

If  $z$  is a **simple** root then convergence is **cubically**.

Over a finite field enumerating all the roots can be done in  $\tilde{O}(D)$ .

In characteristic 0 finding an approximation of all real roots can also be done in  $\tilde{O}(D)$ .

**Overall complexity** of multivariate solving lies in the computation of a LEX Gröbner basis resp. a RUR.

**#5**

How to get the RUR / LEX Gröbner basis in shape position

**F4 Algorithm** for computing DRL Gröbner basis

**gb** package, plain C code

**GB.jl** for **OSCAR**

Start your **julia** session. Then

```
//Load the GB.jl library, also loads Singular.jl.
using GB

// Next we define a ring R of characteristic  $2^{31}-1$ 
// with DRL order and the ideal I in R generated by the
// cyclic generators with 10 variables.
R,I = GB.cyclic_10(2^31-1, :degrevlex)

// Compute Groebner basis G for I using standard
// settings of GB's F4 implementation.
G = Gb.f4(I)

// Same computation, but with specialized setting:
// hash table size =  $2^{21}$ , 8 threads,
// max. 2500 s-polynomials, probabilistic linear algebra
G = Gb.f4(I, 21, 8, 2500, 42)

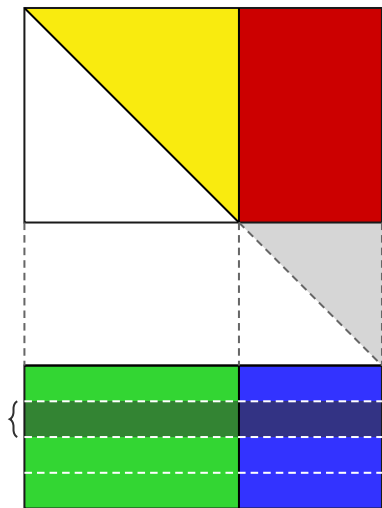
// Further process G using Singular stuff
H = Singular.fglm(G, :lex) // TODO as a first step?
```



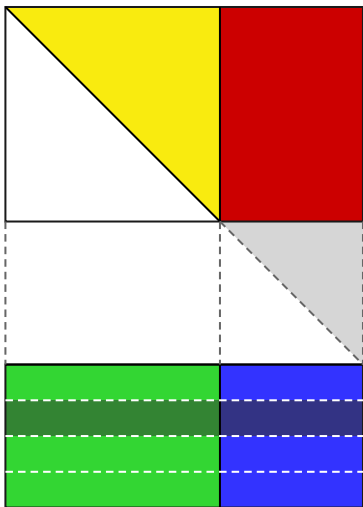
**Magma / Maple** performance for 31-bit prime fields  
using **probabilistic linear algebra** for reduction.

Case No.	Case Name	Case Type	Case Status	Case Date	Case Location	Case Description	Case Details	Case Notes	Case Actions	Case Comments	Case History	Case Attachments	Case Links	Case Tags	Case Filters	Case Sort	Case Search	Case Help	Case Logout
1	Case 1	Case Type 1	Case Status 1	Case Date 1	Case Location 1	Case Description 1	Case Details 1	Case Notes 1	Case Actions 1	Case Comments 1	Case History 1	Case Attachments 1	Case Links 1	Case Tags 1	Case Filters 1	Case Sort 1	Case Search 1	Case Help 1	Case Logout 1
2	Case 2	Case Type 2	Case Status 2	Case Date 2	Case Location 2	Case Description 2	Case Details 2	Case Notes 2	Case Actions 2	Case Comments 2	Case History 2	Case Attachments 2	Case Links 2	Case Tags 2	Case Filters 2	Case Sort 2	Case Search 2	Case Help 2	Case Logout 2
3	Case 3	Case Type 3	Case Status 3	Case Date 3	Case Location 3	Case Description 3	Case Details 3	Case Notes 3	Case Actions 3	Case Comments 3	Case History 3	Case Attachments 3	Case Links 3	Case Tags 3	Case Filters 3	Case Sort 3	Case Search 3	Case Help 3	Case Logout 3
4	Case 4	Case Type 4	Case Status 4	Case Date 4	Case Location 4	Case Description 4	Case Details 4	Case Notes 4	Case Actions 4	Case Comments 4	Case History 4	Case Attachments 4	Case Links 4	Case Tags 4	Case Filters 4	Case Sort 4	Case Search 4	Case Help 4	Case Logout 4
5	Case 5	Case Type 5	Case Status 5	Case Date 5	Case Location 5	Case Description 5	Case Details 5	Case Notes 5	Case Actions 5	Case Comments 5	Case History 5	Case Attachments 5	Case Links 5	Case Tags 5	Case Filters 5	Case Sort 5	Case Search 5	Case Help 5	Case Logout 5
6	Case 6	Case Type 6	Case Status 6	Case Date 6	Case Location 6	Case Description 6	Case Details 6	Case Notes 6	Case Actions 6	Case Comments 6	Case History 6	Case Attachments 6	Case Links 6	Case Tags 6	Case Filters 6	Case Sort 6	Case Search 6	Case Help 6	Case Logout 6
7	Case 7	Case Type 7	Case Status 7	Case Date 7	Case Location 7	Case Description 7	Case Details 7	Case Notes 7	Case Actions 7	Case Comments 7	Case History 7	Case Attachments 7	Case Links 7	Case Tags 7	Case Filters 7	Case Sort 7	Case Search 7	Case Help 7	Case Logout 7
8	Case 8	Case Type 8	Case Status 8	Case Date 8	Case Location 8	Case Description 8	Case Details 8	Case Notes 8	Case Actions 8	Case Comments 8	Case History 8	Case Attachments 8	Case Links 8	Case Tags 8	Case Filters 8	Case Sort 8	Case Search 8	Case Help 8	Case Logout 8
9	Case 9	Case Type 9	Case Status 9	Case Date 9	Case Location 9	Case Description 9	Case Details 9	Case Notes 9	Case Actions 9	Case Comments 9	Case History 9	Case Attachments 9	Case Links 9	Case Tags 9	Case Filters 9	Case Sort 9	Case Search 9	Case Help 9	Case Logout 9
10	Case 10	Case Type 10	Case Status 10	Case Date 10	Case Location 10	Case Description 10	Case Details 10	Case Notes 10	Case Actions 10	Case Comments 10	Case History 10	Case Attachments 10	Case Links 10	Case Tags 10	Case Filters 10	Case Sort 10	Case Search 10	Case Help 10	Case Logout 10

日期	时间	地点	事件	备注
2023-10-27	08:00	会议室	项目启动会	参会人员: 张三, 李四, 王五
2023-10-27	10:30	市场部	客户拜访	拜访客户: 赵六
2023-10-27	14:00	研发部	技术讨论	讨论主题: 新功能开发
2023-10-27	16:30	财务部	报表审核	审核报表: 销售数据
2023-10-28	09:00	市场部	客户拜访	拜访客户: 钱七
2023-10-28	11:30	人事部	招聘面试	面试职位: 销售经理
2023-10-28	13:00	研发部	代码审查	审查项目: 系统升级
2023-10-28	15:30	运营部	活动策划	策划活动: 双十一促销
2023-10-28	18:00	市场部	客户拜访	拜访客户: 孙八
2023-10-29	08:30	财务部	预算编制	编制部门: 市场部
2023-10-29	10:00	研发部	需求分析	分析需求: 用户反馈
2023-10-29	12:30	运营部	数据监控	监控指标: 网站流量
2023-10-29	14:00	市场部	客户拜访	拜访客户: 周九
2023-10-29	16:00	人事部	培训组织	组织培训: 新员工入职
2023-10-29	18:30	财务部	报表生成	生成报表: 月度总结
2023-10-30	09:30	运营部	活动策划	策划活动: 线上推广
2023-10-30	11:00	研发部	测试执行	执行测试: 新功能模块
2023-10-30	13:30	市场部	客户拜访	拜访客户: 吴十
2023-10-30	15:00	运营部	数据监控	监控指标: 转化率
2023-10-30	17:30	财务部	报表审核	审核报表: 运营数据
2023-10-31	08:00	市场部	客户拜访	拜访客户: 郑十一
2023-10-31	10:30	研发部	代码提交	提交代码: 新功能完成
2023-10-31	13:00	运营部	数据监控	监控指标: 用户留存
2023-10-31	15:30	市场部	客户拜访	拜访客户: 冯十二
2023-10-31	18:00	财务部	报表生成	生成报表: 季度总结

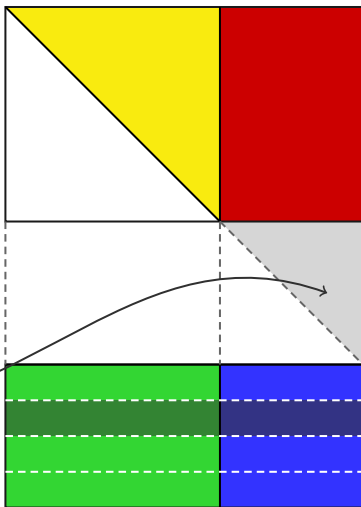


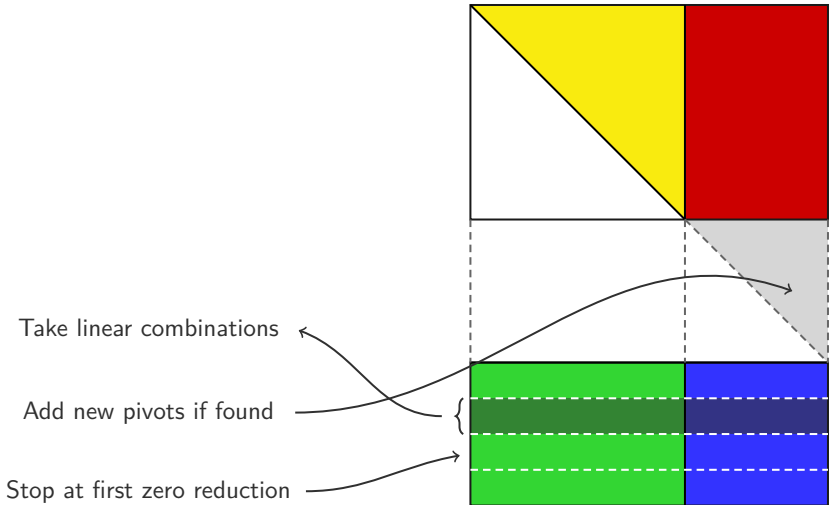
Take linear combinations



Take linear combinations

Add new pivots if found





# Todo

Implementation over  $\mathbb{Q}$

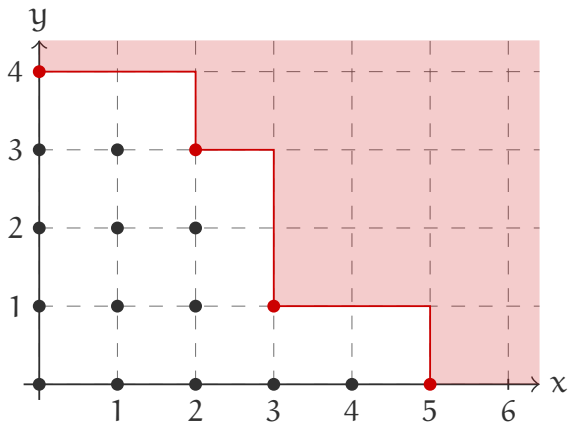
Multimodular implementation



**Conversion** from DRL to LEX Gröbner basis using the **FGLM** algorithm.

Complexity:  $O(nD^3)$

Use zero-dimensional structure of  $R/I$ .  
DRL Gröbner basis gives us a **finite basis**  $B$  for  $R/I$  as vector space.



## Step 1

Generate **multiplication matrices**

$$M_{x_i} : R/I \rightarrow R/I, \bar{p} \mapsto \overline{x_i p}$$

where reduction is done w.r.t. the DRL Gröbner basis.

We have  $O(nD)$  matrix-vector products of size  $D \times D$  times  $D \times 1$ , thus a complexity of  $O(nD^3)$  for this step.

## Step 2

Test **linear dependency** of  $O(nD)$  vectors of size  $D \times 1$ , done in  $O(nD^3)$  arithmetic operations:

Add 1 to  $B'$  and to  $C$ . Multiply 1 by all variables, add them to  $L$ .

Take  $m \in L$  minimal w.r.t. LEX and reduce  $m$  w.r.t.  $G$ .

▷ If  $\overline{m}$  is linearly independent w.r.t.  $C$

then add  $m$  to  $B'$ ,  $\overline{m}$  to  $C$  and add multiples of  $m$  to  $L$ .

▷ If  $\overline{m}$  is linearly dependent w.r.t.  $C$  then  $\overline{m - \sum_i \lambda_i b_i} = \overline{0}$ , i.e.  $m - \sum_i \lambda_i b_i \in I$ . Thus add  $m - \sum_i \lambda_i b_i$  to  $G'$ .

Method by **Mourrain, Telen** and **van Barel** (2018)

They propose a new method for constructing the multiplication matrices.

Allows **finite precision** computation.

Gröbner bases are unstable, border bases need a good initial choice of basis taking global numerical properties into account.

Idea of **truncated normal forms**.

## Setting

Let  $I = \langle f_1, \dots, f_m \rangle \subset R = \mathbb{C}[x_1, \dots, x_n]$  be zero-dimensional, say  $\dim_{\mathbb{C}}(R/I) = D$ .

Let  $V, W$  be finite dimensional vector spaces of  $R$  such that  $V \rightarrow \mathbb{C}^D$  is surjective and  $x_i W \subset V$  for all  $i$ .

Denote

$$\begin{aligned} \phi_I: V_1 \times \dots \times V_m &\rightarrow V, \\ (q_1, \dots, q_m) &\mapsto \sum_{i=1}^m q_i f_i. \end{aligned}$$

such that  $f_i V_i \subset V$  for all  $i$ .

## Algorithm

Compute  $\phi_I$  and let  $N \leftarrow \ker(\phi_I^T)^T$ .

Choose  $h = h_0 + \sum_{i=1}^n h_i x_i$  such that  $hW \subset V$ .

Set  $N_0 : W \mapsto N(hw)$  for  $w \in W$ .

$N' \leftarrow$  an invertible submatrix of  $N_0$ .

$B \leftarrow$  monomials corresponding to columns of  $N'$ .

(This gives us an isomorphism between the basis  $B$  and  $R/I$ .)

For  $i = 1, \dots, n$  do

$N_i \leftarrow$  columns of  $N$  corresponding to  $x_i B$ .

$M_{x_i} \leftarrow (N')^{-1} N_i$ .

Return  $(M_{x_1}, \dots, M_{x_n})$ .

$N'$  must be chosen such that an **inverse** (last step) can be computed **accurately**.

Use column pivoted **QR factorization** on  $N_0$ .

$\Rightarrow$  We get a monomial basis with good numerical properties.



## Comparison to GB approach

<b>Mourrain et al</b>	<b>Gröbner bases</b>
Construct $\phi_I$ and compute $N$ .	Compute reduced DRL GB $G$ with induced normal form NF.
QR factorization with pivoting on $N _W$ to get $N'$ corresponding to a basis $B$ for $R/I$ .	Find a normal set $B$ from $G$ .
Compute $N_i$ and set $M_{x_i} = (N')^{-1}N_i$ .	Compute multiplication matrices $M_{x_i}$ by applying NF to $x_i B$ .

Thanks

Questions? Remarks?