Polynomial Systems Solving by Fast Linear Algebra

J.-C. Faugère[1] P. Gaudry[2] L. Huot[1] G. Renault[1]

1: POLSYS Project INRIA Paris-Rocquencourt ; UPMC, Univ Paris 06, LIP6 ; CNRS, UMR 7606, LIP6
2: CARAMEL Project INRIA Grand-Est; Université de Lorraine; CNRS, UMR 7503; LORIA

Journées Nationales de Calcul Formel
Luminy, May 13-17th 2013

Context

Applications

Coding theory, cryptography, computational game theory, optimization *etc*

What means solving?

Depends on the context.

- find one solution;
- enumerate all the solutions in some field;
- find a certified approximation of the real solutions;
- $\cdots$

Problem: univariate polynomial representation (PoSSo)

Input: $S = \{f_1, \ldots, f_s\} \subset \mathbb{K}[x_1, \ldots, x_n]$ s.t. $\langle S \rangle$ **radical**, **zero-dimensional**.
Output: $h_1, \ldots, h_n \in \mathbb{K}[x_n]$ s.t. $S \equiv \{x_1 - h_1 = \cdots = x_{n-1} - h_{n-1} = h_n = 0\}$.

State of the art

D: number of solutions of S.

Particular case

$\mathbb{K}$ field of characteristic zero; $\delta \leq D$ number of real roots.

- (Mourrain, Pan 1998) Approximate all the real roots: $\widetilde{O}(12^n D^2)$ if $\delta = O(\log_2(D))$;
- (Bostan, Salvy, Schost 2003) RUR: $\widetilde{O}(n 2^n D^{\frac{5}{2}})$ if the multiplicative structure of the quotient ring is known.

$\langle S \rangle$ in *Shape Position* $\Rightarrow$ **univariate polynomial representation** $\equiv$ **LEX Gröbner basis**.

General case

In the *best case*, **LEX Gröbner basis**: $O(nD^3)$.

Our aim

Providing the **first algorithm** with sub-cubic complexity to compute a univariate polynomial representation of the solutions.

PoSSo and Gröbner basis

In our context PoSSo $\equiv$ computing a LEX Gröbner basis.

Usual algorithm to compute a LEX Gröbner basis

Input: $S \subset \mathbb{K}[x_1, \ldots, x_n]$.
Output: The LEX Gröbner basis of $\langle S \rangle$.

1. Compute the DRL Gröbner basis of $\langle S \rangle$;
2. Compute the LEX Gröbner basis of $\langle S \rangle$ by using a change of ordering algorithm.

- Gröbner basis algorithms:
  - Historical: (Buchberger 1965) Buchberger's algorithm;
  - Efficient: (Faugère 1999/2002) F_4 and F_5.
- Change of ordering algorithm:
  - (Faugère, Gianni, Lazard, Mora 1993) FGLM;
  - (Faugère, Mou 2011/2013) Sparse FGLM.

Gröbner basis and Complexity – State of the art

$(f_1, \ldots, f_n)$ regular sequence with $\deg(f_i) \leq d$.

$S = \{f_1, \ldots, f_n\}$

↓ F_4, F_5 (Bardet, Faugère, Salvy 2005) $O\left(d^{\omega n}\right)$

GB DRL

↓ Order-Change

- generic: (Faugère, Mou 2013) $O\left(\sqrt{\frac{6}{n\pi}} D^{2+\frac{n-1}{n}}\right)$
- non-generic: (Faugère,Gianni,Lazard,Mora 1993) $O(nD^3)$.

GB LEX

Gröbner basis and Complexity – State of the art

$(f_1, \dots, f_n)$ regular sequence with $\deg(f_i) \le d$.

$S = \{f_1, \dots, f_n\}$

F_4, F_5 ↓ (Bardet, Faugère, Salvy 2005) $O\left(d^{\omega n}\right)$

GB DRL

Order-Change ↓

Theoretical Bottleneck

- generic: (Faugère, Mou 2013) $O\left(\sqrt{\frac{6}{n\pi}} D^{2+\frac{n-1}{n}}\right)$
- non-generic: (Faugère,Gianni,Lazard,Mora 1993) $O(nD^3)$.

GB LEX

Gröbner basis and Complexity – Contributions

$(f_1, \ldots, f_n)$ regular sequence with $\deg(f_i) \leq d$ with $\langle f_1, \ldots, f_n \rangle$ a radical ideal.

$S = \{f_1, \ldots, f_n\}$

F_4, F_5 (Bardet, Faugère, Salvy 2005) $O\left(d^{\omega n}\right)$

GB DRL

Order-Change

Contributions

- d fixed integer:
  - deterministic (**Shape Position**): $O(\log(D)^{\omega+1} D^{\omega})$;
  - probabilistic: $O(\log(D) D^{\omega})$;
- d non fixed parameter:
  - probabilistic: $O(\log(D) D^{\omega})$;

GB LEX

PoSSo complexity

If the Bézout's bound is reached $\widetilde{O}(d^{\omega n} + D^{\omega}) = \widetilde{O}(D^{\omega})$

where $2 \leq \omega < 2.3727$ is the linear algebra constant.

Change of ordering algorithm

Input: $\mathcal{G}_{\mathsf{drl}}$ the DRL Gröbner basis of $I \subset \mathbb{K}[x_1, \ldots, x_n]$.

Finite number of solutions $D \Rightarrow V = \mathbb{K}[x_1, \ldots, x_n] / \langle \mathcal{G}_{\mathsf{drl}} \rangle$ is a $\mathbb{K}$-vector space of dimension D.

$B = \{1 = \epsilon_1 < \cdots < \epsilon_D\}$ the canonical basis of V.

(Sparse) FGLM

1. Multiplicative structure of V *i.e.* the multiplication matrices $T_1, \ldots, T_n$;

$$T_i = \begin{array}{c} \\ \\ \\ \end{array} \begin{array}{ccc} \mathsf{NF}_{\mathsf{drl}}(\epsilon_1 x_i) & \cdots & \mathsf{NF}_{\mathsf{drl}}(\epsilon_D x_i) \end{array}$$

$$T_i = \left(\begin{array}{ccc} \star & \cdots & \star \\ \vdots & \ddots & \vdots \\ \star & \cdots & \star \end{array} \right) \begin{array}{c} \epsilon_1 \\ \vdots \\ \epsilon_D \end{array}$$

2. From $T_1, \ldots, T_n$, recover the LEX Gröbner basis.
   I in Shape Position $\Rightarrow \mathcal{G}_{\mathsf{lex}} = \{x_1 - h_1(x_n), \ldots, x_{n-1} - h_{n-1}(x_n), h_n(x_n)\}$.

Key ideas

1 Multiplication matrices

FGLM	This work
nD normal forms $\equiv$ dependent matrix-vector products	$O(\log_2(D))$ row echelon form **Fast matrix multiplication**
$O(nD^3)$	$\widetilde{O}(D^\omega)$

Key ideas

1 Multiplication matrices

FGLM	This work
nD normal forms $\equiv$ dependent matrix-vector products	$O(\log_2(D))$ row echelon form **Fast matrix multiplication**
$O(nD^3)$	$\widetilde{O}(D^\omega)$

2 LEX Gröbner basis

$T = T_n^t$ matrix of size $D \times D$. $\#T$ number of nonzero entries in T.
$\mathbf{r}$ random column vector of size D or canonical vector.

	Sparse FGLM	This work
Step 1	$2D$ matrix-vector products $T^j\mathbf{r}$ for $j = 0, \ldots, 2D-1$ T dense $\rightsquigarrow O(D^3)$	
Step 2	Solving n Hankel systems $O(nD\log_2^2 D)$	
probabilistic	$O(D(\#T + n\log_2^2 D))$	
deterministic	(radical) $\widetilde{O}(D\#T + D^2 n)$	

Key ideas

1 Multiplication matrices

FGLM	This work
nD normal forms $\equiv$ dependent matrix-vector products	$O(\log_2(D))$ row echelon form
	Fast matrix multiplication
$O(nD^3)$	$\widetilde{O}(D^\omega)$

2 LEX Gröbner basis

$T = T_n^t$ matrix of size $D \times D$. $\#T$ number of nonzero entries in T.
$\mathbf{r}$ random column vector of size D or canonical vector.

	Sparse FGLM		This work
Step 1	$2D$ matrix-vector products		$2\log_2(D)$ matrix products
	$T^j\mathbf{r}$ for $j = 0, \ldots, 2D-1$	— Keller-Gehrig →	**Fast matrix multiplication**
	T dense $\rightsquigarrow O(D^3)$		$\widetilde{O}(D^\omega)$
Step 2	Solving n Hankel systems $O(nD\log_2^2 D)$		
probabilistic	$O(D(\#T + n\log_2^2 D))$		$O(\log_2 D(D^\omega + n\log_2 D))$
deterministic	(radical) $\widetilde{O}(D\#T + D^2 n)$		(radical) $\widetilde{O}(D^\omega + D^2 n)$

Key ideas

1 Multiplication matrices

FGLM	This work
nD normal forms $\equiv$ dependent matrix-vector products	$O(\log_2(D))$ row echelon form **Fast matrix multiplication**
$O(nD^3)$	$\widetilde{O}(D^\omega)$

2 LEX Gröbner basis

$T = T_n^t$ matrix of size $D \times D$. $\#T$ number of nonzero entries in T.
Input: $\mathcal{G}_{\mathsf{drl}}$ of an ideal in Shape Position and T_n.

	Sparse FGLM	This work
probabilistic	$O(D(\#T + n\log_2^2 D))$	$O(\log_2 D(D^\omega + n\log_2 D))$
deterministic	(radical) $\widetilde{O}(D\#T + D^2 n)$	(radical) $\widetilde{O}(D^\omega + D^2 n)$

Computing $T_1, \ldots, T_n$: the original algorithm

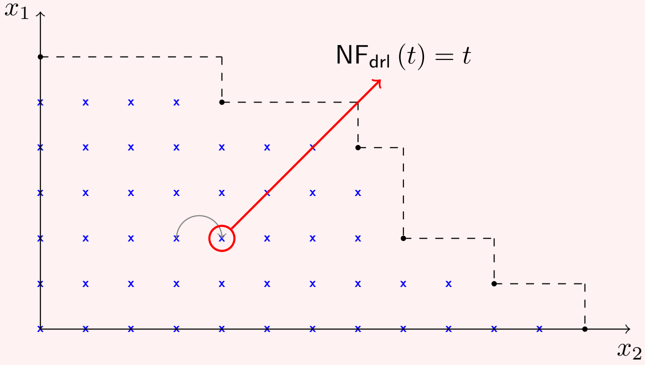
Computing $T_1, \ldots, T_n \Leftrightarrow$ computing $\mathsf{NF}_{\mathsf{drl}}(\epsilon_i x_j)$ $i = 1, \ldots, D$ and $j = 1, \ldots, n$.

$$T_i = \begin{array}{c} \\ \\ \end{array} \begin{array}{c} \begin{array}{ccc} \mathsf{NF}_{\mathsf{drl}}(\epsilon_1 x_i) & \cdots & \mathsf{NF}_{\mathsf{drl}}(\epsilon_D x_i) \end{array} \\ \left(\begin{array}{ccc} \star & \cdots & \star \\ \vdots & \ddots & \vdots \\ \star & \cdots & \star \end{array} \right) \end{array} \begin{array}{c} \epsilon_1 \\ \vdots \\ \epsilon_D \end{array}$$

Computing $T_1, \ldots, T_n$: the original algorithm

Computing $T_1, \ldots, T_n \Leftrightarrow$ computing $\mathsf{NF}_{\mathsf{drl}}(\epsilon_i x_j)$ $i = 1, \ldots, D$ and $j = 1, \ldots, n$.

Proposition (Faugère, Gianni, Lazard, Mora) – Three cases

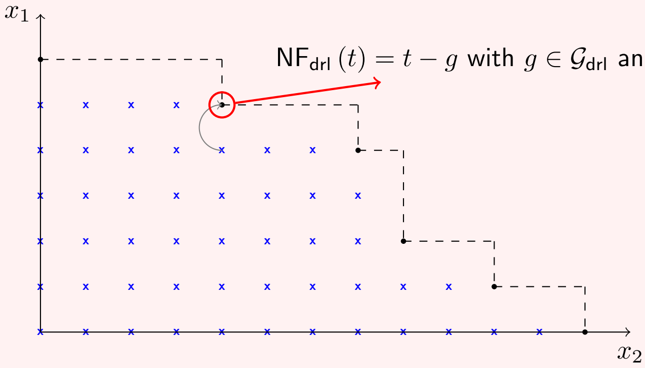


Case (1) $t = \epsilon_i x_j \in B$

Computing $T_1, \ldots, T_n$: the original algorithm

Computing $T_1, \ldots, T_n$ $\Leftrightarrow$ computing $\mathsf{NF}_{\mathsf{drl}}(\epsilon_i x_j)$ $i = 1, \ldots, D$ and $j = 1, \ldots, n$.

Proposition (Faugère, Gianni, Lazard, Mora) – Three cases



Case (2) $t = \epsilon_i x_j \in E(I) = \{\mathsf{LT}_{\mathsf{drl}}(g) \mid g \in \mathcal{G}_{\mathsf{drl}}\}$

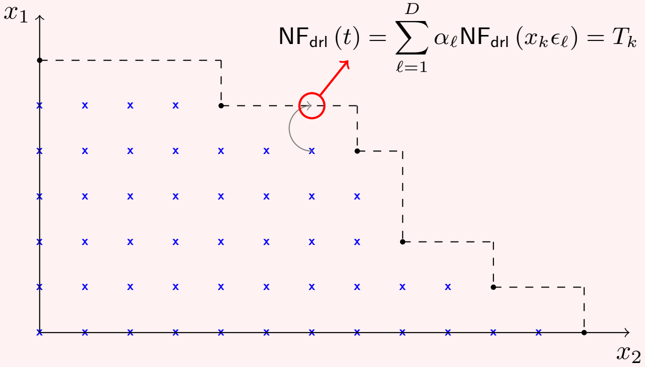
Computing $T_1, \ldots, T_n$: the original algorithm

Computing $T_1, \ldots, T_n \Leftrightarrow$ computing $\mathsf{NF}_{\mathsf{drl}}(\epsilon_i x_j)$ $i = 1, \ldots, D$ and $j = 1, \ldots, n$.

$F = \{\epsilon_i x_j \mid i = 1, \ldots, D$ and $j = 1, \ldots, n\} \setminus B$: frontier

Proposition (Faugère, Gianni, Lazard, Mora) – Three cases

$$\mathsf{NF}_{\mathsf{drl}}(t) = \sum_{\ell=1}^{D} \alpha_\ell \mathsf{NF}_{\mathsf{drl}}(x_k \epsilon_\ell) = T_k \cdot (\alpha_1, \ldots, \alpha_D)^t$$



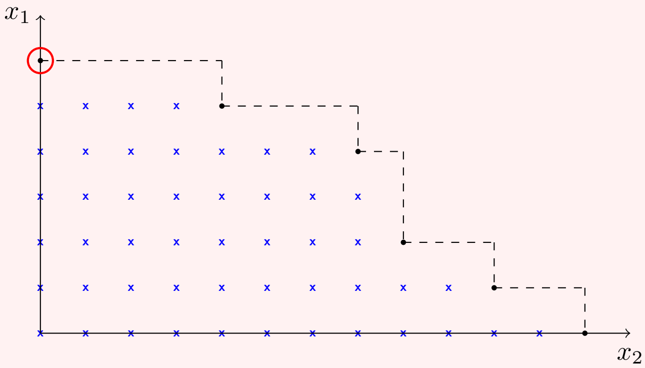
Case (3) $t = \epsilon_i x_i \in F \setminus E(I) \Rightarrow t = x_k t'$ with $t' \in F$ with $\mathsf{NF}_{\mathsf{drl}}(t') = \sum_{i=\ell}^{D} \alpha_\ell \epsilon_\ell$

Computing $T_1, \ldots, T_n$: the original algorithm

Computing $T_1, \ldots, T_n \Leftrightarrow$ computing $\mathsf{NF}_{\mathsf{drl}}(\epsilon_i x_j)$ $i = 1, \ldots, D$ and $j = 1, \ldots, n$.

$F = \{\epsilon_i x_j \mid i = 1, \ldots, D \text{ and } j = 1, \ldots, n\} \setminus B$: frontier

Proposition (Faugère, Gianni, Lazard, Mora) – Three cases

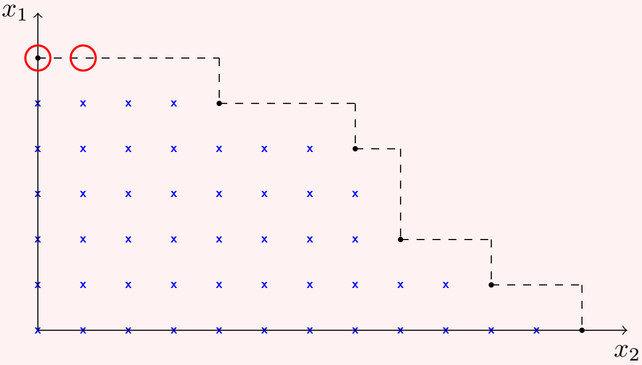


Computing $T_1, \ldots, T_n$: the original algorithm

Computing $T_1, \ldots, T_n \Leftrightarrow$ computing $\mathsf{NF}_{\mathsf{drl}}(\epsilon_i x_j)$ $i = 1, \ldots, D$ and $j = 1, \ldots, n$.

$F = \{\epsilon_i x_j \mid i = 1, \ldots, D \text{ and } j = 1, \ldots, n\} \setminus B$: frontier

Proposition (Faugère, Gianni, Lazard, Mora) – Three cases

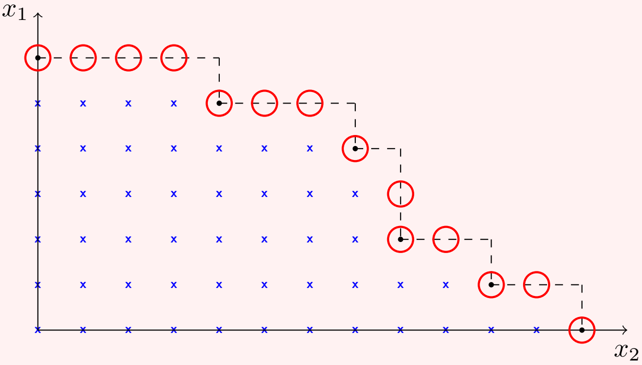


Computing $T_1, \ldots, T_n$: the original algorithm

Computing $T_1, \ldots, T_n \Leftrightarrow$ computing $\mathsf{NF}_{\mathsf{drl}}(\epsilon_i x_j)$ $i = 1, \ldots, D$ and $j = 1, \ldots, n$.

$F = \{\epsilon_i x_j \mid i = 1, \ldots, D \text{ and } j = 1, \ldots, n\} \setminus B$: frontier

Proposition (Faugère, Gianni, Lazard, Mora) – Three cases



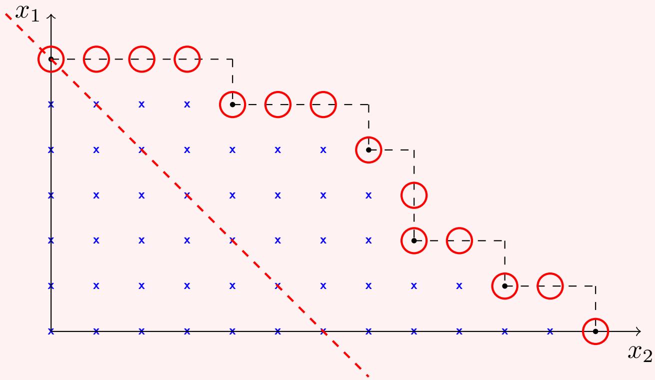
$\#F \leq nD \Rightarrow$ total complexity $O(nD^3)$

Computing $T_1, \ldots, T_n$

Computing $T_1, \ldots, T_n$ $\Leftrightarrow$ computing $\mathsf{NF}_{\mathsf{drl}}(\epsilon_i x_j)$ $i = 1, \ldots, D$ and $j = 1, \ldots, n$.

$F = \{\epsilon_i x_j \mid i = 1, \ldots, D \text{ and } j = 1, \ldots, n\} \setminus B$: frontier

This work

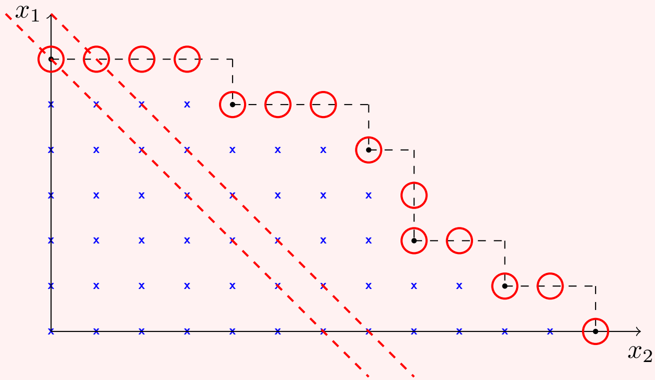


Computing $T_1, \ldots, T_n$

Computing $T_1, \ldots, T_n \Leftrightarrow$ computing $\mathsf{NF}_{\mathsf{drl}}(\epsilon_i x_j)$ $i = 1, \ldots, D$ and $j = 1, \ldots, n$.

$F = \{\epsilon_i x_j \mid i = 1, \ldots, D$ and $j = 1, \ldots, n\} \setminus B$: frontier

This work

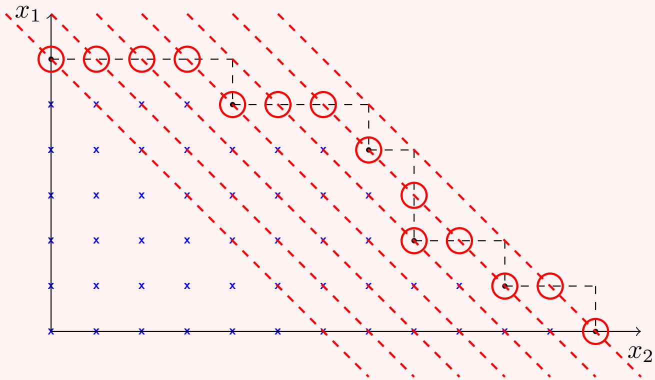


Computing $T_1, \ldots, T_n$

Computing $T_1, \ldots, T_n$ $\Leftrightarrow$ computing $\mathsf{NF}_{\mathsf{drl}}\,(\epsilon_i x_j)$ $i = 1, \ldots, D$ and $j = 1, \ldots, n$.

$F = \{\epsilon_i x_j \mid i = 1, \ldots, D \text{ and } j = 1, \ldots, n\} \setminus B$: frontier

This work



Computing $T_1, \ldots, T_n$ using fast linear algebra

Iterative algorithm: loop on the **degree** d

	$t_\ell \in F$ $\deg(t_\ell) = d$	$t_j \in F$ $\deg(t_j) < d$	$\epsilon_i \in B$ **Reading NF**
$t_j - \mathsf{NF}_{\mathsf{drl}}(t_j)$ $\forall t_j \in F$, $\deg(t_j) < d$	$\begin{matrix} 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{matrix}$	$\begin{matrix} 1 & \cdots & 0 \\ & \ddots & \\ 0 & \cdots & 1 \end{matrix}$	$\begin{matrix} \star & \cdots & \star \\ \vdots & \mathbf{C} & \vdots \\ \star & \cdots & \star \end{matrix}$

Computing $T_1, \ldots, T_n$ using fast linear algebra

Iterative algorithm: loop on the **degree** d

	$t_\ell \in F$, $\deg(t_\ell) = d$				$t_j \in F$, $\deg(t_j) < d$			$\epsilon_i \in B$		
	1	$\star$	$\cdots$	$\star$	$\star$	$\cdots$	$\star$	$\star$	$\cdots$	$\star$
$f_\ell \in \mathcal{I}$, $\mathsf{LT}_{\mathsf{drl}}(f_\ell) = t_\ell$	0	1		$\vdots$	$\star$	$\cdots$	$\star$	$\star$	$\cdots$	$\star$
$\forall t_\ell \in F$, $\deg(t_\ell) = d$	$\vdots$	**T**	$\ddots$	$\star$	$\vdots$	**A**	$\vdots$	$\vdots$	**B**	$\vdots$
	0	0	$\cdots$	1	$\star$	$\cdots$	$\star$	$\star$	$\cdots$	$\star$
$t_j - \mathsf{NF}_{\mathsf{drl}}(t_j)$	0	0	$\cdots$	0	1	$\cdots$	0	$\star$	$\cdots$	$\star$
$\forall t_j \in F$, $\deg(t_j) < d$	$\vdots$	$\vdots$	$\ddots$	$\vdots$		$\ddots$		$\vdots$	**C**	$\vdots$
	0	0	$\cdots$	0	0	$\cdots$	1	$\star$	$\cdots$	$\star$

- If $t_\ell \in E(I)$ then $f_\ell = g$ with $g \in \mathcal{G}_{\mathsf{drl}}$ st $\mathsf{LT}_{\mathsf{drl}}(g) = t_\ell$;
- Else $t_\ell \in F \setminus E(I) \Rightarrow t_\ell = x_k t_j$ and $f_\ell = x_k(t_j - \mathsf{NF}_{\mathsf{drl}}(t_j)) = t_\ell + \sum_{i=1}^{D} \alpha_i x_k \epsilon_i$.

Computing $T_1, \ldots, T_n$ using fast linear algebra

Iterative algorithm: loop on the **degree** d

	$t_\ell \in F$ $\deg(t_\ell) = d$	$t_j \in F$ $\deg(t_j) < d$	$\epsilon_i \in B$ **Reading NF**	
Reduced Row Echelon Form $\rightsquigarrow$	$\begin{matrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & 0 & \cdots & 1 \end{matrix}$	$\begin{matrix} 0 & \cdots & 0 \\ 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{matrix}$	$\begin{matrix} \star & \cdots & \star \\ \star & \cdots & \star \\ & \mathbf{T^{-1}(B - AC)} & \\ \star & \cdots & \star \end{matrix}$	$t_\ell - \mathsf{NF}_{\mathsf{drl}}(t_\ell)$ $\forall t_\ell \in F$, $\deg(t_\ell) = d$
	$\begin{matrix} 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{matrix}$	$\begin{matrix} 1 & \cdots & 0 \\ & \ddots & \\ 0 & \cdots & 1 \end{matrix}$	$\begin{matrix} \star & \cdots & \star \\ \vdots & \mathbf{C} & \vdots \\ \star & \cdots & \star \end{matrix}$	$t_j - \mathsf{NF}_{\mathsf{drl}}(t_j)$ $\forall t_j \in F$, $\deg(t_j) < d$

The normal forms of all the monomials of same degree can be computed simultaneously.

Computing $T_1, \ldots, T_n$ using fast linear algebra

Size of M at most $(nD \times (n+1)D)$.

Theorem

Given $\mathcal{G}_{\mathsf{drl}}$, computing all the multiplication matrices $T_1, \ldots, T_n$ can be done in

$$O(d_{\max} n^\omega D^\omega) \text{ arithmetic operations}$$

where $d_{\max} = \max\{\deg(t) \mid t \in F\} = \max\{\deg(g) \mid g \in \mathcal{G}_{\mathsf{drl}}\}$.

Regular System

Let $S = \{f_1, \ldots, f_n\}$ with $\deg(f_i) \leq d$ and $(f_1, \ldots, f_n)$ is a regular sequence.

- Macaulay's bound $\Rightarrow d_{\max} \leq n(d-1)+1$;
- Bézout's bound $\Rightarrow D \leq d^n$.

d *fixed integer* $\Rightarrow O(d_{\max} n^\omega D^\omega) = O(n^{\omega+1} D^\omega) = O(\log_2(D)^{\omega+1} D^\omega)$.

deterministic algorithm to compute $\mathcal{G}_{\mathsf{lex}}$ given $\mathcal{G}_{\mathsf{drl}}$ in
$O(n^{\omega+1} D^\omega + \log_2(D) D^\omega) = O(\log_2(D)^{\omega+1} D^\omega)$

Construction of T_n: (1) the generic case

To compute T_n we only need $\mathsf{NF}_{\mathsf{drl}}(\epsilon_i x_n)$ for $i = 1, \ldots, D$.

Proposition

For generic ideals, $\epsilon_i x_n \in B \cup E(I)$ for $i = 1, \ldots, D$.

Construction of T_n: (1) the generic case

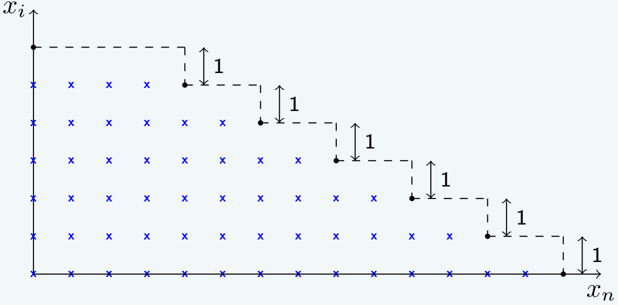
To compute T_n we only need $\mathsf{NF}_{\mathsf{drl}}(\epsilon_i x_n)$ for $i = 1, \ldots, D$.

Proposition

For generic ideals, $\epsilon_i x_n \in B \cup E(I)$ for $i = 1, \ldots, D$.

Moreno-Socias

For any instantiation of $\deg_{x_j}$ for $j \in \{1, \ldots, n-1\} \setminus \{i\}$



Construction of T_n: (1) the generic case

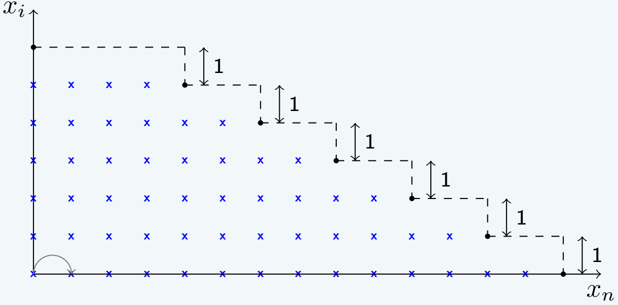
To compute T_n we only need $\mathsf{NF}_{\mathsf{drl}}(\epsilon_i x_n)$ for $i = 1, \ldots, D$.

Proposition

For generic ideals, $\epsilon_i x_n \in B \cup E(I)$ for $i = 1, \ldots, D$.

Moreno-Socias

For any instantiation of $\deg_{x_j}$ for $j \in \{1, \ldots, n-1\} \setminus \{i\}$



Construction of T_n: (1) the generic case

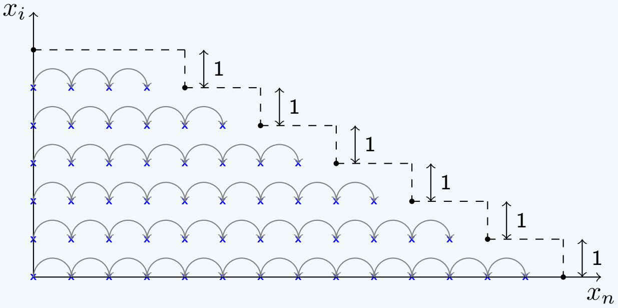
To compute T_n we only need $\mathsf{NF}_{\mathsf{drl}}(\epsilon_i x_n)$ for $i = 1, \ldots, D$.

Proposition

For generic ideals, $\epsilon_i x_n \in B \cup E(I)$ for $i = 1, \ldots, D$.

Moreno-Socias

For any instantiation of $\deg_{x_j}$ for $j \in \{1, \ldots, n-1\} \setminus \{i\}$



Construction of T_n: (1) the generic case

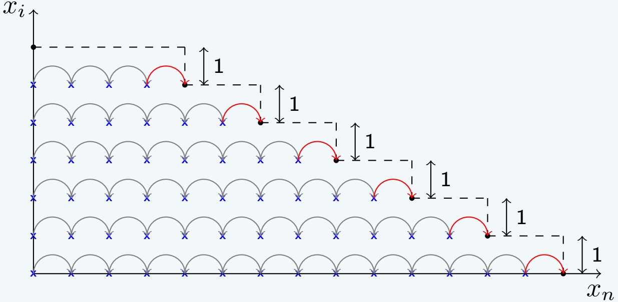
To compute T_n we only need $\mathsf{NF}_{\mathsf{drl}}(\epsilon_i x_n)$ for $i = 1, \ldots, D$.

Proposition

For generic ideals, $\epsilon_i x_n \in B \cup E(I)$ for $i = 1, \ldots, D$.

Moreno-Socias

For any instantiation of $\deg_{x_j}$ for $j \in \{1, \ldots, n-1\} \setminus \{i\}$



Construction of T_n: (2) the non-generic case

$p =$ characteristic of $\mathbb{K}$.
Hypothesis: $p = 0$ or p sufficiently large.

Galligo, Bayer and Stillman, Pardue

I an *homogeneous ideal*. There exists a Zariski open subset $U \subset \mathrm{GL}(\mathbb{K}, n)$ s.t. $\forall g \in U$, $g \cdot I$ has the structure of generic ideals.

Theorem

$I = \langle f_1, \ldots, f_n \rangle$ an *affine ideal* and $I^{(h)} = \langle f_1^{(h)}, \ldots, f_n^{(h)} \rangle$.

- $(f_1, \ldots, f_n)$ regular $\Rightarrow E(g \cdot I) = E\left(g \cdot I^{(h)}\right)$;
- **no arithmetic operation** to compute T_n of $g \cdot I$ where g is randomly chosen in $\mathrm{GL}(\mathbb{K}, n)$.

Shape Lemma

I a **radical** ideal. There exists a Zariski open subset $U' \subset \mathrm{GL}(\mathbb{K}, n)$ such that for all $g \in U'$, $g \cdot I$ is in *Shape Position*.

New algorithm for PoSSo

Let d such that $\deg(f_i) \leq d$.

Another algorithm for PoSSo.

Input : $S = \{f_1, \ldots, f_n\} \subset \mathbb{K}[x_1, \ldots, x_n]$ s.t. $\langle S \rangle$ is radical.
Output: g in GL$(\mathbb{K}, n)$ and the LEX Gröbner basis of $\langle g \cdot S \rangle$ or *fail*.
Randomly choose g in GL$(\mathbb{K}, n)$;
Compute $\mathcal{G}_{\mathsf{drl}}$ the DRL Gröbner basis of $g \cdot S$; $O(d^{\omega n})$
if *T_n can be read from $\mathcal{G}_{drl}$* **then**
  Extract T_n from $\mathcal{G}_{\mathsf{drl}}$; free
  if *$\langle g \cdot S \rangle$ is in Shape Position* **then**
    From T_n and $\mathcal{G}_{\mathsf{drl}}$ compute $\mathcal{G}_{\mathsf{lex}}$; $O(\log_2(D)(D^\omega + n\log_2(D)D))$
    return g and $\mathcal{G}_{\mathsf{lex}}$;
return *fail*;

Total complexity: $O(d^{\omega n} + D^\omega \log_2 D)$ arithmetic operations.

To summarize

$S = \{f_1, \ldots, f_n\}$ with $\deg(f_i) \leq d$ with $\langle f_1, \ldots, f_n \rangle$ is radical.

New complexity for PoSSo (simple roots)

- d **fixed** integer:
  - deterministic: (*Shape Position*) $O(d^{\omega n} + \log_2(D)^{\omega+1} D^{\omega})$ arithmetic operations;
  - probabilistic: $O(d^{\omega n} + \log_2(D) D^{\omega})$ arithmetic operations;
- d **non fixed** parameter:
  - probabilistic: $O(d^{\omega n} + \log_2(D) D^{\omega})$ arithmetic operations.

In practice

Probabilistic algorithm for PoSSo + Sparse FGLM $\Rightarrow$ **running time decreased**.

Example: $S = \{f_1, \ldots, f_n\}$ with $\mathrm{LT}_{\mathrm{drl}}(f_i) = x_i^2$. If $n = 11$:

- DRL Gröbner basis $\nearrow$: 0s $\to 5.02$s;
- Multiplication matrix $\searrow$: 7520.89s $\to 0.15$s ($31.93\% \to 21.53\%$);
- Univariate polynomial representation $\searrow$: 0.20s $\to 0.13$s;
- Total $\searrow$: 7521.09s $\to 5.30$s MAGMA: > 2 hours.