Polynomial Systems Solving by Fast Linear Algebra

J.-C. Faugère\textsuperscript{1}  P. Gaudry\textsuperscript{2}  L. Huot\textsuperscript{1}  G. Renault\textsuperscript{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

PolSys Seminar, April 25\textsuperscript{th}
Context

Motivation

Many 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;
- ...

Problem: univariate polynomial representation (PoSSo)

Let $S = \{f_1, \ldots, f_s\} \subset \mathbb{K}[x_1, \ldots, x_n]$ be a set of polynomial equations with a finite number of solutions which are all simple. Find a univariate polynomial representation of the solutions of $S$ i.e. $h_1, \ldots, h_n \in \mathbb{K}[x_n]$ such that
\[
\{x_1 - h_1 = \cdots = x_{n-1} - h_{n-1} = h_n = 0\}
\]
has the same solutions as $S$. 

L. Huot

PoSSo by Fast Linear Algebra
State of the art

Let $D$ be the number of solutions of $S$.

Particular case

$K$: field of characteristic zero. Sub-cubic algorithms to:

- approximate all the real roots: $\tilde{O}(12^n D^2)$ (Mourrain, Pan 1998);
- compute a rational parametrization knowing the multiplicative structure of the quotient ring: $\tilde{O}(n2^n D^{5/2})$ (Bostan, Salvy, Schost 2003).

If the ideal is in Shape Position, the univariate polynomial representation is given by the LEX Gröbner basis.

General case

In the best case, the complexity of computing a LEX Gröbner basis is bounded by $O(nD^3)$.

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:** A polynomial system \( S \subset K[x_1, \ldots, x_n] \).

**Output:** The LEX Gröbner basis of \( \langle S \rangle \).

1. Compute the DRL Gröbner basis of \( \langle S \rangle \);
2. From the DRL Gröbner basis, by using a change of ordering algorithm, compute the LEX Gröbner basis of \( \langle S \rangle \).

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) \(O(d^{\omega n})\)

- **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

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

\[ S = \{f_1, \ldots, f_n\} \]

\[ F_4, F_5 \ (\text{Bardet, Faugère, Salvy 2005}) \quad O(d^{\omega n}) \]

\[ \text{GB DRL} \]

\[ \text{Order-Change} \]

\[ \text{Bottleneck} \]

- generic: \((\text{Faugère, Mou 2013}) \quad O \left( \sqrt{\frac{6}{n\pi}} D^2 + \frac{n-1}{n} \right) \)
- non-generic: \((\text{Faugère, Gianni, Lazard, Mora 1993}) \quad O(nD^3)\).
Gröbner basis and Complexity – Contributions

\((f_1, \ldots, f_n)\) regular sequence with \(\text{deg}(f_i) \leq d\).

\[ S = \{f_1, \ldots, f_n\} \]

\(F_4, F_5\) (Bardet, Faugère, Salvy 2005) \(O(d^\omega n)\)

Order-Change

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

**Shape Position case**

An ideal $I$ is said to be in *Shape Position* if its LEX Gröbner basis if of the form:

$$\{x_1 - h_1(x_n), \ldots, x_{n-1} - h_{n-1}(x_n), h_n(x_n)\}.$$

**Main results**

Let $S = \{f_1, \ldots, f_n\} \subset K[x_1, \ldots, x_n]$ with $\deg(f_i) \leq d$. If $(f_1, \ldots, f_n)$ is a regular sequence then

- if $d$ is a **fixed integer** and $\langle S \rangle$ is in *Shape Position* then there exists a deterministic algorithm solving the PoSSo problem in $\tilde{O}(d^{\omega n} + D^{\omega})$;
Contributions

**Shape Position case**

An ideal $I$ is said to be in *Shape Position* if its LEX Gröbner basis is of the form:

$$\{x_1 - h_1(x_n), \ldots, x_{n-1} - h_{n-1}(x_n), h_n(x_n)\}.$$ 

**Main results**

Let $S = \{f_1, \ldots, f_n\} \subset \mathbb{K}[x_1, \ldots, x_n]$ with $\deg(f_i) \leq d$. If $(f_1, \ldots, f_n)$ is a regular sequence then

- if $d$ is a **fixed integer** and $\langle S \rangle$ is in *Shape Position* then there exists a deterministic algorithm solving the PoSSo problem in $\tilde{O}(d^{\omega n} + D^\omega)$;
- if $d$ is a **fixed or non fixed parameter** then there exists a Las Vegas algorithm solving the PoSSo problem in $\tilde{O}(d^{\omega n} + D^\omega)$. 

If the Bézout's bound is reached $\tilde{O}(d^{\omega n} + D^\omega) = \tilde{O}(D^\omega)$ where $2 \leq \omega < 2.3727$ is the linear algebra constant.
Contributions

**Shape Position case**

An ideal $I$ is said to be in *Shape Position* if its LEX Gröbner basis is of the form:

$$\{x_1 - h_1(x_n), \ldots, x_{n-1} - h_{n-1}(x_n), h_n(x_n)\}.$$

**Main results**

Let $S = \{f_1, \ldots, f_n\} \subset K[x_1, \ldots, x_n]$ with $\deg(f_i) \leq d$. If $(f_1, \ldots, f_n)$ is a regular sequence then

- if $d$ is a fixed integer and $\langle S \rangle$ is in *Shape Position* then there exists a deterministic algorithm solving the PoSSo problem in $\tilde{O}(d^\omega n^2 + D^\omega)$;
- if $d$ is a fixed or non fixed parameter then there exists a Las Vegas algorithm solving the PoSSo problem in $\tilde{O}(d^\omega n^2 + D^\omega)$.

If the Bézout’s bound is reached $\tilde{O}(d^\omega n^2 + D^\omega) = \tilde{O}(D^\omega)$ where $2 \leq \omega < 2.3727$ is the linear algebra constant.
Change of ordering algorithm

Given $G_{drl}$ be the DRL Gröbner basis of an ideal $I \subset \mathbb{K}[x_1, \ldots, x_n]$.

$I$ has a finite number of solutions $D \Rightarrow V = \mathbb{K}[x_1, \ldots, x_n]/\langle G_{drl} \rangle$ is a $\mathbb{K}$-vector space of dimension $D$.

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

**Algorithm**

1. Compute the multiplicative structure of $V$ i.e. the multiplication matrices $T_1, \ldots, T_n$ with

   $T_i = \begin{pmatrix}
   \text{NF}_{drl}(\epsilon_1 x_i) & \cdots & \text{NF}_{drl}(\epsilon_D x_i) \\
   \star & \cdots & \star \\
   \vdots & \ddots & \vdots \\
   \star & \cdots & \star 
   \end{pmatrix}

   $\epsilon_1$

2. From this multiplicative structure, recover the LEX Gröbner basis.
Key ideas

(Sparse) FGLM

- Multiplication matrices
  \( nD \) normal forms \( \equiv \) dependent matrix-vector products
  \( O(nD^3) \) arithmetic operations

- LEX Gröbner basis
  \( 2D \) matrix-vector products
  \( T^j r \) for \( j = 0, \ldots, 2D - 1 \)
  \( O(D^3) \) arithmetic operations

This work
## Key ideas

### (Sparse) FGLM

- **Multiplication matrices**
  - $nD$ normal forms $\equiv$ dependent matrix-vector products
  - $O(nD^3)$ arithmetic operations

- **LEX Gröbner basis**
  - $2D$ matrix-vector products
  - $T^j r$ for $j = 0, \ldots, 2D - 1$
  - $O(D^3)$ arithmetic operations

### This work

- **Multiplication matrices**
  - $\log_2(D)$ row echelon form
  - Fast matrix multiplication
  - $\tilde{O}(D^\omega)$ arithmetic operations

- **LEX Gröbner basis**
  - $2 \log_2(D)$ matrix products
  - Fast matrix multiplication
  - $\tilde{O}(D^\omega)$ arithmetic operations
Fast Univariate Polynomial Representation
Hypothesis

$I$ is in *Shape Position* i.e.

$$G_{\text{lex}} = \{x_1 - h_1(x_n), \ldots, x_{n-1} - h_{n-1}(x_n), h_n(x_n)\}$$

where $\deg(h_n) = D$ and $\deg(h_i) = D - 1$ for $i = 1, \ldots, n - 1$.

Problem

Given $G_{\text{drl}}$ and the multiplication matrices $T_1, \ldots, T_n$ compute the polynomials $h_n$ and $h_i = \sum_{k=0}^{D-1} c_{i,k} x_n^k$ for $i = 1, \ldots, n - 1$. 
Sparse change of ordering: Part 1

Faugère, Mou 2011 and 2013

1. Find the univariate polynomial $h_n(x_n)$
   Let $r$ be a random vector.
   1. Construct $S = \langle r, T_n^i \mathbf{1} \rangle : i = 0, \ldots, 2D - 1$.
      Note that $\langle r, T_n^i \mathbf{1} \rangle = \langle (T_n^t)^i r, \mathbf{1} \rangle$
      $\Rightarrow O(ND)$
   2. Berlekamp-Massey $\Rightarrow$ minimal polynomial of $S = \mu$.
      If $\deg(\mu) = D$ then $h_n = \mu$
      $\Rightarrow O(\log_2(D)^2 D)$

2. Recover the coefficients $c_{i,k}$ by solving structured linear systems (Hankel matrices).
   $\Rightarrow O(n \log_2(D)^2 D)$
Sparse change of ordering

**Lemma**
From $G_{drl}$, $T_i \mathbf{1}$ can be computed without $T_i$.

**Theorem (Faugèere and Mou)**
Given $T_n$ and $G_{drl}$ of an ideal in *Shape Position*, the LEX Gröbner basis can be computed in

- $O(D(N + n \log_2(D)^2))$ (probabilistic)
- $O(D(N + D(n + \log_2(D) \log_2(\log_2(D)))))$ (deterministic)

arithmetic operations where $N$ is the number of non zero entries in $T_n$.

**Open issue**
If $T_n$ is assumed to be dense $\Rightarrow O(D^3)$ arithmetic operations.
How to compute efficiently $T^j \mathbf{r}$ for $j = 0, \ldots, 2D - 1$ with $T = T_n^t$?
Computing $T^j r$ for $j = 0, \ldots, 2D - 1$

1. Compute $T^2, T^4, \ldots, T^{2^\lceil \log_2(D) \rceil}$ with $\lceil \log_2(D) \rceil$ multiplication matrices.

   \[ O(\log_2(D)D^\omega) \text{ arithmetic operations.} \]

2. (Keller-Gehrig) Compute $\lceil \log_2(D) \rceil$ multiplication matrices of the form

   \[
   T^2(T^3 \mid r) = (T^3 r \mid T^2 r) \\
   T^4(T^3 r \mid T^2 r \mid T^r \mid r) = (T^7 r \mid T^6 r \mid T^5 r \mid T^4 r) \\
   \vdots \\
   T^{2^\lceil \log_2(D) \rceil}(T^{2^\lceil \log_2(D) \rceil - 1} \mid \ldots \mid r) = (T^{2D-1} r \mid T^{2D-2} r \mid \ldots \mid T^{2^\lceil \log_2(D) \rceil} r).
   \]

   \[ O(\log_2(D)D^\omega) \text{ arithmetic operations.} \]
Fast univariate polynomial representation

Theorem

Given \( T_n \) and \( G_{drl} \) of an ideal in \textit{Shape Position}, the LEX Gröbner basis can be computed in

- \( O(\log_2(D)(D^\omega + n \log_2(D)D)) \) (probabilistic);
- \( O(\log_2(D)D^\omega + D^2(n + \log_2(D)\log_2(\log_2(D)))) \) (deterministic).

Open issue

Compute \( T_n \) with less than \( O(nD^3) \) arithmetic operations.
Multiplication Matrices
Computing $T_1, \ldots, T_n$: the original algorithm

Computing $T_1, \ldots, T_n$ requires to compute $\text{NF}_{\text{drl}}(\epsilon_i x_j)$ for $i = 1, \ldots, D$ and $j = 1, \ldots, n$.

\[
T_i = \begin{pmatrix}
\text{NF}_{\text{drl}}(\epsilon_1 x_i) & \cdots & \text{NF}_{\text{drl}}(\epsilon_D x_i) \\
\star & \cdots & \star \\
\vdots & \ddots & \vdots \\
\star & \cdots & \star \\
\end{pmatrix} \epsilon_i
\]

Let denote,
- $E(I) = \{\text{LT}_{\text{drl}}(g) \mid g \in G_{\text{drl}}\}$;
- $F = \{\epsilon_i x_j \mid i = 1, \ldots, D \text{ and } j = 1, \ldots, n\} \setminus B$. 

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

Computing $T_1, \ldots, T_n$ requires to compute $\text{NF}_{drl}(\epsilon_i x_j)$ for $i = 1, \ldots, D$ and $j = 1, \ldots, n$.

Let denote,

- $E(I) = \{\text{LT}_{drl}(g) \mid g \in G_{drl}\}$;
- $F = \{\epsilon_i x_j \mid i = 1, \ldots, D \text{ and } j = 1, \ldots, n\} \setminus B$.

**Proposition (Faugère, Gianni, Lazard, Mora)**

Let $t = \epsilon_i x_j$; $t$ can be

1. either in $B$ i.e. $t = \epsilon_k$ for some $k > i$;
2. or in $E(I)$ i.e. $t = \text{LT}_{drl}(g)$ with $g \in G_{drl}$;
3. or in $F \setminus E(I)$ i.e. $t = x_k t'$ with $t' \in F$. 

Complexity $\#F \leq nD \Rightarrow$ at most $nD$ matrix-vector products.

$O(nD^3)$ arithmetic operations.
Computing $T_1, \ldots, T_n$: the original algorithm

Computing $T_1, \ldots, T_n$ requires to compute $\text{NF}_{\text{drl}}(\epsilon_i x_j)$ for $i = 1, \ldots, D$ and $j = 1, \ldots, n$.

Let denote,

- $E(I) = \{\text{LT}_{\text{drl}}(g) \mid g \in \mathcal{G}_{\text{drl}}\}$;
- $F = \{\epsilon_i x_j \mid i = 1, \ldots, D \text{ and } j = 1, \ldots, n\} \setminus B$.

**Proposition (Faugère, Gianni, Lazard, Mora)**

Let $t = \epsilon_i x_j$; $t$ can be

1. either in $B$ i.e. $t = \epsilon_k$ for some $k > i$;
2. or in $E(I)$ i.e. $t = \text{LT}_{\text{drl}}(g)$ with $g \in \mathcal{G}_{\text{drl}}$;
3. or in $F \setminus E(I)$ i.e. $t = x_k t'$ with $t' \in F$.

\[ \Rightarrow \text{NF}_{\text{drl}}(t) = t \]
Computing $T_1, \ldots, T_n$: the original algorithm

Computing $T_1, \ldots, T_n$ requires to compute $\text{NF}_\text{drl}(\epsilon_i x_j)$ for $i = 1, \ldots, D$ and $j = 1, \ldots, n$.

Let denote,

- $E(I) = \{\text{LT}_\text{drl}(g) \mid g \in \mathcal{G}_\text{drl}\}$;
- $F = \{\epsilon_i x_j \mid i = 1, \ldots, D \text{ and } j = 1, \ldots, n\} \setminus B$.

### Proposition (Faugère, Gianni, Lazard, Mora)

Let $t = \epsilon_i x_j$; $t$ can be

1. either in $B$ i.e. $t = \epsilon_k$ for some $k > i$;
2. or in $E(I)$ i.e. $t = \text{LT}_\text{drl}(g)$ with $g \in \mathcal{G}_\text{drl}$;
3. or in $F \setminus E(I)$ i.e. $t = x_k t'$ with $t' \in F$.

$\Rightarrow \text{NF}_\text{drl}(t) = t$

$\Rightarrow \text{NF}_\text{drl}(t) = t - g$
Computing $T_1, \ldots, T_n$: the original algorithm

Computing $T_1, \ldots, T_n$ requires to compute $\text{NF}_{\text{drl}}(\epsilon_{i}x_{j})$ for $i = 1, \ldots, D$ and $j = 1, \ldots, n$.

Let denote,

- $E(I) = \{\text{LT}_{\text{drl}}(g) \mid g \in \mathcal{G}_{\text{drl}}\}$;
- $F = \{\epsilon_{i}x_{j} \mid i = 1, \ldots, D \text{ and } j = 1, \ldots, n\} \setminus B$.

**Proposition (Faugère, Gianni, Lazard, Mora)**

Let $t = \epsilon_{i}x_{j}$; $t$ can be

- (I) either in $B$ i.e. $t = \epsilon_{k}$ for some $k > i$; $\Rightarrow \text{NF}_{\text{drl}}(t) = t$
- (II) or in $E(I)$ i.e. $t = \text{LT}_{\text{drl}}(g)$ with $g \in \mathcal{G}_{\text{drl}}$; $\Rightarrow \text{NF}_{\text{drl}}(t) = t - g$
- (III) or in $F \setminus E(I)$ i.e. $t = x_{k}t'$ with $t' \in F$. $\Rightarrow \text{NF}_{\text{drl}}(t) = \sum_{\ell=1}^{D} \alpha_{\ell}\text{NF}_{\text{drl}}(x_{k}\epsilon_{\ell})$ where $\text{NF}_{\text{drl}}(t') = \sum_{i=\ell}^{D} \alpha_{\ell}\epsilon_{\ell}$
Computing $T_1, \ldots, T_n$: the original algorithm

Computing $T_1, \ldots, T_n$ requires to compute $\text{NF}_{\text{drl}}(\epsilon_i x_j)$ for $i = 1, \ldots, D$ and $j = 1, \ldots, n$.

Let denote,

- $E(I) = \{\text{LT}_{\text{drl}}(g) \mid g \in \mathcal{G}_{\text{drl}}\}$;
- $F = \{\epsilon_i x_j \mid i = 1, \ldots, D$ and $j = 1, \ldots, n\} \setminus B$.

Proposition (Faugère, Gianni, Lazard, Mora)

Let $t = \epsilon_i x_j$; $t$ can be

(I) either in $B$ i.e. $t = \epsilon_k$ for some $k > i$;\[ \Rightarrow \text{NF}_{\text{drl}}(t) = t \]

(II) or in $E(I)$ i.e. $t = \text{LT}_{\text{drl}}(g)$ with $g \in \mathcal{G}_{\text{drl}}$;\[ \Rightarrow \text{NF}_{\text{drl}}(t) = t - g \]

(III) or in $F \setminus E(I)$ i.e. $t = x_k t'$ with $t' \in F$.\[ \Rightarrow \text{NF}_{\text{drl}}(t) = \sum_{\ell=1}^{D} \alpha_{\ell} \text{NF}_{\text{drl}}(x_k \epsilon_{\ell}) \text{ where } \text{NF}_{\text{drl}}(t') = \sum_{i=\ell}^{D} \alpha_{\ell} \epsilon_{\ell} \]

Complexity

$\#F \leq nD \Rightarrow$ at most $nD$ matrix-vector products.

$O(nD^3)$ arithmetic operations
Computing $T_1, \ldots, T_n$ using fast linear algebra

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

$$M = \begin{bmatrix}
\begin{array}{cccc}
1 & * & \cdots & * \\
0 & 1 & \cdots & * \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & 1 \\
\end{array}
\end{bmatrix}
\begin{bmatrix}
\begin{array}{c}
1 \\
* \\
\vdots \\
* \\
\end{array}
\end{bmatrix}
\begin{bmatrix}
\begin{array}{c}
* \\
* \\
\vdots \\
* \\
\end{array}
\end{bmatrix}
\begin{bmatrix}
\begin{array}{c}
* \\
* \\
\vdots \\
* \\
\end{array}
\end{bmatrix}
$$

where $F_d = \{ t \in F \mid \deg(t) = d \} = \{ t_i < \cdots < t_{i+s_d} \}$

- $\tilde{f}_i = t_i - \text{NF}_{drl}(t_i)$
- If $t_i \in E(I)$ then $f_i = \tilde{f}_i = g$ with $g \in G_{drl}$ st $\text{LT}_{drl}(g) = t_i$.
- Else $t_i \in F \setminus E(I) \Rightarrow t_i = x_k t_j$ and $f_i = x_k \tilde{f}_j = t_i + \sum_{i=1}^{D} \alpha_j x_k e_j$. 

where $F_d = \{ t \in F \mid \deg(t) = d \} = \{ t_i < \cdots < t_{i+s_d} \}$

- $\tilde{f}_i = t_i - \text{NF}_{drl}(t_i)$
- If $t_i \in E(I)$ then $f_i = \tilde{f}_i = g$ with $g \in G_{drl}$ st $\text{LT}_{drl}(g) = t_i$.
- Else $t_i \in F \setminus E(I) \Rightarrow t_i = x_k t_j$ and $f_i = x_k \tilde{f}_j = t_i + \sum_{i=1}^{D} \alpha_j x_k e_j$. 

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

$M$: matrix of size at most $(nD \times (n + 1)D)$.  
Computing the row echelon form of $M$ can be done in $O(n^\omega D^\omega)$ arithmetic operations.

**Theorem**

Given $G_{drl}$, computing all the multiplication matrices $T_1, \ldots, T_n$ can be done in

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

where $d_{\text{max}} = \max\{\deg(t) \mid t \in F\} = \max\{\deg(g) \mid g \in G_{drl}\}$.

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

If $d$ is a fixed integer then computing $T_1, \ldots, T_n$ given $G_{drl}$ can be done in

$$O(n^{\omega+1} D^\omega) = O(\log_2(D)^{\omega+1} D^\omega)$$ arithmetic operations.
Gröbner basis and Complexity – Contributions

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

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

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

GB DRL

Order-Change

If $d$ is a fixed integer and $\langle f_1, \ldots, f_n \rangle$ is in *Shape Position*: $O(n\omega^{+1}D^\omega + \log_2(D)D^\omega) = O(\log_2(D)^{\omega+1}D^\omega)$ (deterministic)

GB LEX
Gröbner basis and Complexity – Contributions

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

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

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

GB DRL

Order-Change

If $d$ is a fixed integer and $\langle f_1, \ldots, f_n \rangle$ is in \textit{Shape Position}:

$O(n^{\omega+1} D^\omega + \log_2(D)D^\omega) = O(\log_2(D)^{\omega+1} D^\omega)$ (deterministic)

GB LEX

Since we need only $T_n$, can we compute it more efficiently?
Construction of $T_n$: the generic case

To compute $T_n$ we only need $\text{NF}_{\text{drl}}(\epsilon_i x_n)$ 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 $T_n$: the generic case

To compute $T_n$ we only need $NF_{drl}(\epsilon_ix_n)$ for $i = 1, \ldots, D$.

**Proposition**

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

**Moreno-Socías**

For any instantiation of $\deg x_j$ for $j \in \{1, \ldots, n-1\} \setminus \{i\}$
Construction of $T_n$: the generic case

To compute $T_n$ we only need $\text{NF}_{\text{drl}}(\epsilon_i x_n)$ 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 $\text{deg}_{x_j}$ for $j \in \{1, \ldots, n - 1\} \setminus \{i\}$
Construction of $T_n$: the generic case

To compute $T_n$ we only need $\text{NF}_{\text{drl}}(\epsilon_i x_n)$ 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 $T_n$: the generic case

To compute $T_n$ we only need $\text{NF}_{\text{drl}}(\epsilon_i x_n)$ 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 $T_n$: the generic case

To compute $T_n$ we only need $\text{NF}_{drl}(\epsilon_i x_n)$ 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 $T_n$: the generic case

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

Proposition

For generic ideals, $\epsilon_ix_n \in B \cup E(I)$ 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 $T_n$: the non-generic case

Galligo, Bayer and Stillman, Pardue

Let $I$ be an homogeneous ideal. Let $p$ denotes the characteristic of $\mathbb{K}$. If $p = 0$ or if $p$ is sufficiently large then there exists a Zariski open subset $U \subset \text{GL}(\mathbb{K}, n)$ such that $\forall g \in U$, $g \cdot I$ has the structure of generic ideals.

Can be generalized to affine ideals.

Theorem

Let $p$ denotes the characteristic of $\mathbb{K}$. If $p = 0$ or if $p$ is sufficiently large and $\mathbb{K}$ too then no arithmetic operations are required to compute $T_n$ of $g \cdot I$ where $g$ is randomly chosen in $\text{GL}(\mathbb{K}, n)$.
New algorithm for PoSSo

Shape Lemma

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

Another algorithm for PoSSo.

**Input**: A polynomial system $S = \{f_1, \ldots, f_n\} \subset K[x_1, \ldots, x_n]$ generating a radical ideal.

**Output**: $g$ in $\text{GL}(K, n)$ and the LEX Gröbner basis of $\langle g \cdot S \rangle$ or **fail**.

1. Choose randomly $g$ in $\text{GL}(K, n)$;
2. Compute $G_{\text{drl}}$ the DRL Gröbner basis of $g \cdot S$;
3. **if** $T_n$ can be read from $G_{\text{drl}}$ **then**
   - Extract $T_n$ from $G_{\text{drl}}$;
   - **if** $\langle g \cdot S \rangle$ is in **Shape Position** **then**
     - From $T_n$ and $G_{\text{drl}}$ compute $G_{\text{lex}}$;
     - **return** $g$ and $G_{\text{lex}}$;
4. **return** **fail**;
New algorithm for PoSSo: Probability and Complexity

**Probability of success**

Let $p$ be the characteristic of $\mathbb{K}$ and $P$ the probability of success.

- If $p = 0$, then $P = 1$;
- If $p > \sum_{i=1}^{n} (\deg(f_i) - 1) + 1$, then
  
  $$P \geq 1 - \frac{1}{\#\mathbb{K}} \left( \frac{D(D-1)}{2} + (\sum_{i=1}^{n} (\deg(f_i) - 1) + 1) nD \right).$$

**Complexity**

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

- Compute $G_{\text{drl}}$ of $\langle g \cdot S \rangle$:
  $$O(d \omega n)$$ arithmetic operations;
- Compute $T_n$: no arithmetic operation;
- Compute $G_{\text{lex}}$ given $T_n$:
  $$O(\log_2(D)(D \omega + n \log_2(D) D))$$ arithmetic operations.

Total: $$O(d \omega n + \log_2(D)(D \omega))$$ arithmetic operations.
New algorithm for PoSSo: Probability and Complexity

**Probability of success**

Let \( p \) be the characteristic of \( \mathbb{K} \) and \( P \) the probability of success.

- If \( p = 0 \), then \( P = 1 \);
- If \( p > \sum_{i=1}^{n}(\deg(f_i) - 1) + 1 \), then
  \[
  P \geq 1 - \frac{1}{\#\mathbb{K}} \left( \frac{D(D-1)}{2} + (\sum_{i=1}^{n}(\deg(f_i) - 1) + 1) nD \right).
  \]

**Complexity**

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

- Compute \( G_{\text{drl}} \) of \( \langle g \cdot S \rangle \): \( O(d^{\omega n}) \) arithmetic operations;
- Compute \( T_n \): no arithmetic operation;
- Compute \( G_{\text{lex}} \) given \( T_n \): \( O(\log_2(D)(D^\omega + n \log_2(D)D)) \) arithmetic operations.

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

\[ \mathcal{I} = \langle f_1, \ldots, f_n \rangle \text{ where } f_i = x_i^2 + \sum_{k=1}^{n} \left( c_k x_k + \sum_{j > i \land j \neq k} c_{k,j} x_k x_j \right) \text{ with } c_k, c_{k,j} \in \mathbb{K}. \]

\[ \rightsquigarrow \mathcal{G}_{\text{drl}} = \{ f_1, \ldots, f_n \} \]

Normal forms to compute: \( x_n^2 \cdot m \) for all monomial \( m \) linear in each the \( n - 1 \) first variables \( \Rightarrow 2^{n-1} - 1 \).

<table>
<thead>
<tr>
<th>( n )</th>
<th>Algorithm</th>
<th>( D )</th>
<th>DRL</th>
<th>( T_n )</th>
<th>Univariate polynomial representation</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>( F_5 \rightarrow \text{FGLM} )</td>
<td>512</td>
<td>0.00s</td>
<td>255NF 12.81s</td>
<td>0.29s</td>
<td>13.10s</td>
</tr>
<tr>
<td>9</td>
<td>This work</td>
<td>512</td>
<td>0.00s</td>
<td>0.00s</td>
<td>0.01s</td>
<td>0.01s</td>
</tr>
<tr>
<td>11</td>
<td>( F_5 \rightarrow \text{FGLM} )</td>
<td>( 2^{11} )</td>
<td>0.00s</td>
<td>1023NF 7520.89s</td>
<td>( &gt; 2 \text{ days} )</td>
<td>( &gt; 2 \text{ days} )</td>
</tr>
<tr>
<td>11</td>
<td>This work</td>
<td>( 2^{11} )</td>
<td>5.02s</td>
<td>0.15s</td>
<td>0.13s</td>
<td>5.30s</td>
</tr>
<tr>
<td>13</td>
<td>( F_5 \rightarrow \text{FGLM} )</td>
<td>( 2^{13} )</td>
<td>0.00s</td>
<td>4095NF ( &gt; 2 \text{ days} )</td>
<td>( &gt; 2 \text{ days} )</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>This work</td>
<td>( 2^{13} )</td>
<td>156.67s</td>
<td>2.36s</td>
<td>25.80s</td>
<td>184.83s</td>
</tr>
</tbody>
</table>
To summarize

\[ S = \{f_1, \ldots, f_n\} \text{ with } \deg(f_i) \leq d. \]

New algorithms

- PoSSo
- Change of ordering
  - Multiplication matrices
  - Univariate polynomial representation

<table>
<thead>
<tr>
<th></th>
<th>Deterministic</th>
<th>Probabilistic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiplication matrices</td>
<td>(O(dn^{\omega+1}D^\omega))</td>
<td>Free</td>
</tr>
<tr>
<td>Univariate polynomial representation</td>
<td>(O(\log_2(D)D^\omega))</td>
<td>(O(\log_2(D)D^\omega))</td>
</tr>
</tbody>
</table>

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.