Gröbner Basis Computations

Christian Eder

May 22, 2014 $\Longrightarrow$ SKATEGOTT $\Longrightarrow$ June 05, 2014

Conventions

- $\mathscr{R} = \mathscr{K}[x_1, \ldots, x_n]$, $\mathscr{K}$ field, $<$ well-ordering on $\mathrm{Mon}(x_1, \ldots, x_n)$
- $f \in \mathscr{R}$ can be represented in a unique way by $<$.
  $\Rightarrow$ Definitions as $\mathrm{lc}(f)$, $\mathrm{lm}(f)$, and $\mathrm{lt}(f)$ make sense.
- An ideal I in $\mathscr{R}$ is an additive subgroup of $\mathscr{R}$ such that for $f \in I$, $g \in \mathscr{R}$ it holds that $fg \in I$.
- $G = \{g_1, \ldots, g_s\} \subset \mathscr{R}$ is a Gröbner basis for $I = \langle f_1, \ldots, f_m \rangle$ w.r.t. $<$

$$:\Longleftrightarrow$$

$$G \subset I \text{ and } L_<(G) = L_<(I)$$

Polynomial reduction

Definition

Let $f \in \mathscr{R}$ and let t be a term of f.

- We can **reduce** t by $g \in \mathscr{R}$ if $\exists\, b$ such that $\mathrm{lt}(bg) = t$.
  Outcome: $f - bg$.
- Reduce $f \in \mathscr{R}$ to $h \in \mathscr{R}$ by a sequence of reduction steps.
- Reductions always w.r.t. a finite subset $G \subset \mathscr{R}$

Notation: $f \xrightarrow{G} h$.

Polynomial reduction

Definition

Let $f \in \mathscr{R}$ and let t be a term of f.

- We can **reduce** t by $g \in \mathscr{R}$ if $\exists\, b$ such that $\mathrm{lt}(bg) = t$.
  Outcome: $f - bg$.
- Reduce $f \in \mathscr{R}$ to $h \in \mathscr{R}$ by a sequence of reduction steps.
- Reductions always w.r.t. a finite subset $G \subset \mathscr{R}$

Notation: $f \xrightarrow{G} h$.

$\Longrightarrow$ Another characterisation of Gröbner bases:

Polynomial reduction

Definition

Let $f \in \mathscr{R}$ and let t be a term of f.

- We can **reduce** t by $g \in \mathscr{R}$ if $\exists\, b$ such that $\mathrm{lt}(bg) = t$.
  Outcome: $f - bg$.
- Reduce $f \in \mathscr{R}$ to $h \in \mathscr{R}$ by a sequence of reduction steps.
- Reductions always w.r.t. a finite subset $G \subset \mathscr{R}$

$$\text{Notation: } f \xrightarrow{G} h.$$

$\Longrightarrow$ Another characterisation of Gröbner bases:

Gröbner basis (1)

$G \subset \mathscr{R}$ finite is

1. a **Gröbner basis up to degree** d **for** I if
$$G \subset I \text{ and for all } f \in I \text{ with } \deg(f) \leq d: \; f \xrightarrow{G} 0;$$
2. a **Gröbner basis for** I if G is a Gröbner basis in all degrees.

Even more characterizations!

Standard representation

Let $f \in \mathscr{R}$ and $G \subset \mathscr{R}$ finite. A representation

$$f = \sum_{i=1}^{k} m_i g_i$$

with $m_i \neq 0$, $g_i \in G$ pairwise different is called a **standard representation** if

$$\max_{\leq} \{\mathrm{lt}(m_i g_i) \mid 1 \leq i \leq k\} \leq \mathrm{lt}(f).$$

Even more characterizations!

Standard representation

Let $f \in \mathscr{R}$ and $G \subset \mathscr{R}$ finite. A representation

$$f = \sum_{i=1}^{k} m_i g_i$$

with $m_i \neq 0$, $g_i \in G$ pairwise different is called a **standard representation** if

$$\max_{\leq} \{\mathrm{lt}(m_i g_i) \mid 1 \leq i \leq k\} \leq \mathrm{lt}(f).$$

Gröbner basis (2)

$G \subset \mathscr{R}$ finite is

1. a **Gröbner basis up to degree** d **for** I if

   $G \subset I$ and for all $f \in I$ with $\deg(f) \leq d$: f has a standard representation w.r.t. G;

2. a **Gröbner basis for** I if G is a Gröbner basis in all degrees.

Buchberger's criterion

We are still not algorithmic!

Buchberger's criterion

We are still not algorithmic!

S-polynomials

Let $f \neq 0, g \neq 0 \in \mathscr{R}$ and let $\lambda = \operatorname{lcm}(\operatorname{lt}(f), \operatorname{lt}(g))$ be the least common multiple of $\operatorname{lt}(f)$ and $\operatorname{lt}(g)$. The **S-polynomial** between f and g is given by

$$\operatorname{spol}(f, g) := \frac{\lambda}{\operatorname{lt}(f)} f - \frac{\lambda}{\operatorname{lt}(g)} g.$$

Buchberger's criterion

We are still not algorithmic!

S-polynomials

Let $f \neq 0, g \neq 0 \in \mathscr{R}$ and let $\lambda = \operatorname{lcm}(\operatorname{lt}(f), \operatorname{lt}(g))$ be the least common multiple of $\operatorname{lt}(f)$ and $\operatorname{lt}(g)$. The **S-polynomial** between f and g is given by

$$\operatorname{spol}(f,g) := \frac{\lambda}{\operatorname{lt}(f)} f - \frac{\lambda}{\operatorname{lt}(g)} g.$$

Buchberger's criterion [7]

Let $I = \langle f_1, \ldots, f_m \rangle$ be an ideal in $\mathscr{R}$. A finite subset $G \subset \mathscr{R}$ is a **Gröbner basis up to degree d for I** if $G \subset I$ and for all $f, g \in G$ with $\deg(\operatorname{spol}(f,g)) \leq d$: $\operatorname{spol}(f,g) \xrightarrow{G} 0$.

Buchberger's algorithm

Input: Ideal $I = \langle f_1, \ldots, f_m \rangle$
Output: Gröbner basis G for I

1. $G \leftarrow \emptyset$
2. $G \leftarrow G \cup \{f_i\}$ for all $i \in \{1, \ldots, m\}$
3. Set $P \leftarrow \{\text{spol}(f_i, f_j) \mid f_i, f_j \in G, i > j\}$

Buchberger's algorithm

Input: Ideal $I = \langle f_1, \ldots, f_m \rangle$
Output: Gröbner basis G for I

1. $G \leftarrow \emptyset$
2. $G \leftarrow G \cup \{f_i\}$ for all $i \in \{1, \ldots, m\}$
3. Set $P \leftarrow \{\text{spol}(f_i, f_j) \mid f_i, f_j \in G, i > j\}$
4. Choose $p \in P$, $P \leftarrow P \setminus \{p\}$

Buchberger's algorithm

Input: Ideal $I = \langle f_1, \ldots, f_m \rangle$
Output: Gröbner basis G for I

1. $G \leftarrow \emptyset$
2. $G \leftarrow G \cup \{f_i\}$ for all $i \in \{1, \ldots, m\}$
3. Set $P \leftarrow \{\text{spol}(f_i, f_j) \mid f_i, f_j \in G, i > j\}$
4. Choose $p \in P$, $P \leftarrow P \setminus \{p\}$
   (a) If $p \xrightarrow{G} 0$ ▶▶ **no new information**
   Go on with the next element in P.
   (b) If $p \xrightarrow{G} h \neq 0$ ▶▶ **new information**
   Build new S-pair with h and add them to P.
   Add h to G.
   Go on with the next element in P.
5. When $P = \emptyset$ we are done and G is a Gröbner basis of I.

Complexity

In theory:

- Exponential in the number of variables for DRL
- Doubly exponential in the number of variables for LEX

Complexity

In theory:

- Exponential in the number of variables for DRL
- Doubly exponential in the number of variables for LEX

In practice:

- Quite OK

How to improve computations?

- Modular computations (modStd et al.)
- Predict zero reductions (Buchberger, Gebauer-Möller, Möller-Mora-Traverso, Faugère.)
- Sort pair set (Buchberger, Giovini et al., Möller et al.)
- Homogenize: d-Gröbner bases
- Change of ordering (FGLM, Gröbner Walk)
- Linear Algebra: Gauss Elimination (Lazard, Faugère)
- ...

- **Predicting zero reductions**
- **Pair set sorting and homogenization**
- **Finding better reducers**
- **Fast linear algebra for computing Gröbner bases**
- **Modular computations**
- **Change of ordering**

How to detect zero reductions in advance?

Let $I = \langle g_1, g_2 \rangle \in \mathbb{Q}[x, y, z]$ and let $<$ denote the reverse lexicographical ordering. Let

$$\mathbf{g_1 = xy - z^2}, \quad \mathbf{g_2 = y^2 - z^2}$$

How to detect zero reductions in advance?

Let $I = \langle g_1, g_2 \rangle \in \mathbb{Q}[x, y, z]$ and let $<$ denote the reverse lexicographical ordering. Let

$$\mathbf{g_1 = xy - z^2}, \quad \mathbf{g_2 = y^2 - z^2}$$

$$\begin{aligned} \text{spol}(g_2, g_1) &= xg_2 - yg_1 = \mathbf{xy^2} - xz^2 - \mathbf{xy^2} + yz^2 \\ &= -xz^2 + yz^2. \end{aligned}$$

$$\Longrightarrow \mathbf{g_3 = xz^2 - yz^2}.$$

How to detect zero reductions in advance?

Let $I = \langle g_1, g_2 \rangle \in \mathbb{Q}[x, y, z]$ and let $<$ denote the reverse lexicographical ordering. Let

$$\mathbf{g_1 = xy - z^2}, \quad \mathbf{g_2 = y^2 - z^2}$$

$$\begin{aligned} \operatorname{spol}(g_2, g_1) &= xg_2 - yg_1 = \mathbf{xy^2} - xz^2 - \mathbf{xy^2} + yz^2 \\ &= -xz^2 + yz^2. \end{aligned}$$

$$\Longrightarrow \mathbf{g_3 = xz^2 - yz^2}.$$

$$\operatorname{spol}(g_3, g_1) = \mathbf{xyz^2} - y^2z^2 - \mathbf{xyz^2} + z^4 = -y^2z^2 + z^4.$$

How to detect zero reductions in advance?

Let $I = \langle g_1, g_2 \rangle \in \mathbb{Q}[x, y, z]$ and let $<$ denote the reverse lexicographical ordering. Let

$$\mathbf{g_1 = xy - z^2}, \quad \mathbf{g_2 = y^2 - z^2}$$

$$\begin{aligned} \operatorname{spol}(g_2, g_1) &= xg_2 - yg_1 = \mathbf{xy^2} - xz^2 - \mathbf{xy^2} + yz^2 \\ &= -xz^2 + yz^2. \end{aligned}$$

$$\Longrightarrow \mathbf{g_3 = xz^2 - yz^2}.$$

$$\operatorname{spol}(g_3, g_1) = \mathbf{xyz^2} - y^2z^2 - \mathbf{xyz^2} + z^4 = -y^2z^2 + z^4.$$

We can reduce further using $z^2 g_2$:

$$-y^2z^2 + z^4 + y^2z^2 - z^4 = 0.$$

How to detect zero reductions in advance?

Can we see something? How are the generators of the S-polynomials related to each other?

How to detect zero reductions in advance?

Can we see something? How are the generators of the S-polynomials related to each other?

$$\begin{aligned}\operatorname{spol}(g_3, g_2) &= \mathbf{y^2}\left(xz^2 - yz^2\right) - \mathbf{xz^2}\left(y^2 - z^2\right) \\ &= \operatorname{lt}(\mathbf{g_2})g_3 - \operatorname{lt}(\mathbf{g_3})g_2 \\ &= \operatorname{lt}(\mathbf{g_2})\operatorname{lot}(g_3) - \operatorname{lt}(\mathbf{g_3})\operatorname{lot}(g_2)\end{aligned}$$

How to detect zero reductions in advance?

Can we see something? How are the generators of the S-polynomials related to each other?

$$\begin{aligned}\operatorname{spol}(g_3, g_2) &= \mathbf{y^2}\left(xz^2 - yz^2\right) - \mathbf{xz^2}\left(y^2 - z^2\right)\\ &= \operatorname{lt}(\mathbf{g_2})g_3 - \operatorname{lt}(\mathbf{g_3})g_2\\ &= \operatorname{lt}(\mathbf{g_2})\operatorname{lot}(g_3) - \operatorname{lt}(\mathbf{g_3})\operatorname{lot}(g_2)\end{aligned}$$

For all $u \in \operatorname{support}(\operatorname{lot}(g_3))$ we can reduce with ug_2:

$$\begin{aligned}\Longrightarrow &\operatorname{lt}(g_2)\operatorname{lot}(g_3) - \mathbf{g_2}\operatorname{lot}(\mathbf{g_3}) - \operatorname{lt}(g_3)\operatorname{lot}(g_2)\\ = &-\operatorname{lot}(g_2)\operatorname{lot}(g_3) - \operatorname{lt}(g_3)\operatorname{lot}(g_2)\\ = &-g_3\operatorname{lot}(g_2).\end{aligned}$$

How to detect zero reductions in advance?

Can we see something? How are the generators of the S-polynomials related to each other?

$$\begin{aligned}\operatorname{spol}(g_3, g_2) &= \mathbf{y^2}\left(xz^2 - yz^2\right) - \mathbf{xz^2}\left(y^2 - z^2\right)\\ &= \operatorname{lt}(\mathbf{g_2})g_3 - \operatorname{lt}(\mathbf{g_3})g_2\\ &= \operatorname{lt}(\mathbf{g_2})\operatorname{lot}(g_3) - \operatorname{lt}(\mathbf{g_3})\operatorname{lot}(g_2)\end{aligned}$$

For all $u \in \operatorname{support}(\operatorname{lot}(g_3))$ we can reduce with ug_2:

$$\begin{aligned}\Longrightarrow &\operatorname{lt}(g_2)\operatorname{lot}(g_3) - \mathbf{g_2}\operatorname{lot}(\mathbf{g_3}) - \operatorname{lt}(g_3)\operatorname{lot}(g_2)\\ = &-\operatorname{lot}(g_2)\operatorname{lot}(g_3) - \operatorname{lt}(g_3)\operatorname{lot}(g_2)\\ = &-g_3\operatorname{lot}(g_2).\end{aligned}$$

So we can reduce this to zero by vg_3 by all $v \in \operatorname{support}(\operatorname{lot}(g_2))$.

Product criterion [8, 9]

If $\operatorname{lcm}(\operatorname{lt}(f), \operatorname{lt}(g)) = \operatorname{lt}(f)\operatorname{lt}(g)$ then $\operatorname{spol}(f, g) \xrightarrow{\{f,g\}} 0$.

Product criterion [8, 9]

If $\operatorname{lcm}(\operatorname{lt}(f), \operatorname{lt}(g)) = \operatorname{lt}(f)\operatorname{lt}(g)$ then $\operatorname{spol}(f, g) \xrightarrow{\{f,g\}} 0$.

Couldn't we remove $\operatorname{spol}(g_3, g_2)$ in a different way?

Product criterion [8, 9]

If $\operatorname{lcm}(\operatorname{lt}(f), \operatorname{lt}(g)) = \operatorname{lt}(f)\operatorname{lt}(g)$ then $\operatorname{spol}(f, g) \xrightarrow{\{f,g\}} 0$.

Couldn't we remove $\operatorname{spol}(g_3, g_2)$ in a different way?

$$\operatorname{lt}(g_1) = xy \mid xy^2z^2 = \operatorname{lcm}(\operatorname{lt}(g_3), \operatorname{lt}(g_2))$$

Product criterion [8, 9]

If $\operatorname{lcm}(\operatorname{lt}(f), \operatorname{lt}(g)) = \operatorname{lt}(f)\operatorname{lt}(g)$ then $\operatorname{spol}(f, g) \xrightarrow{\{f,g\}} 0$.

Couldn't we remove $\operatorname{spol}(g_3, g_2)$ in a different way?

$$\operatorname{lt}(g_1) = xy \mid xy^2z^2 = \operatorname{lcm}(\operatorname{lt}(g_3), \operatorname{lt}(g_2))$$

$$\Longrightarrow \text{We can rewrite } \operatorname{spol}(g_3, g_2)\text{:}$$

$$\operatorname{spol}(g_3, g_2) = y \underbrace{\operatorname{spol}(g_3, g_1)}_{\xrightarrow{G} 0} - z^2 \underbrace{\operatorname{spol}(g_2, g_1)}_{\xrightarrow{G} -g_3}$$

Product criterion [8, 9]

If $\operatorname{lcm}(\operatorname{lt}(f), \operatorname{lt}(g)) = \operatorname{lt}(f)\operatorname{lt}(g)$ then $\operatorname{spol}(f, g) \xrightarrow{\{f,g\}} 0$.

Couldn't we remove $\operatorname{spol}(g_3, g_2)$ in a different way?

$$\operatorname{lt}(g_1) = xy \mid xy^2z^2 = \operatorname{lcm}(\operatorname{lt}(g_3), \operatorname{lt}(g_2))$$

$\Longrightarrow$ We can rewrite $\operatorname{spol}(g_3, g_2)$:

$$\operatorname{spol}(g_3, g_2) = y \underbrace{\operatorname{spol}(g_3, g_1)}_{\xrightarrow{G} 0} - z^2 \underbrace{\operatorname{spol}(g_2, g_1)}_{\xrightarrow{G} -g_3}$$

Standard representations of $\operatorname{spol}(g_2, g_1)$ and $\operatorname{spol}(g_3, g_1)$
$\Longrightarrow$ Standard representation of $\operatorname{spol}(g_3, g_2)$.

Buchberger's criteria

Chain criterion [10]

Let $f, g, h \in \mathscr{R}$, $G \subset \mathscr{R}$ finite. If

1. $\mathrm{lt}(h) \mid \mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(g))$, and
2. $\mathrm{spol}(f, h)$ and $\mathrm{spol}(h, g)$ have a standard representation w.r.t. G respectively,

then $\mathrm{spol}(f, g)$ has a standard representation w.r.t. G.

Chain criterion [10]

Let $f, g, h \in \mathscr{R}$, $G \subset \mathscr{R}$ finite. If

1. $\mathrm{lt}(h) \mid \mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(g))$, and
2. $\mathrm{spol}(f, h)$ and $\mathrm{spol}(h, g)$ have a standard representation w.r.t. G respectively,

then $\mathrm{spol}(f, g)$ has a standard representation w.r.t. G.

Note

Do not remove too much information! If $\lambda = 1$ and

$$\mathrm{spol}(f, g) = \lambda \mathrm{spol}(f, h) + \sigma \mathrm{spol}(h, g),$$

then we can remove $\mathrm{spol}(f, g)$ or $\mathrm{spol}(f, h)$ but not both!

Chain criterion [10]

Let $f, g, h \in \mathscr{R}$, $G \subset \mathscr{R}$ finite. If

1. $\mathrm{lt}(h) \mid \mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(g))$, and
2. $\mathrm{spol}(f, h)$ and $\mathrm{spol}(h, g)$ have a standard representation w.r.t. G respectively,

then $\mathrm{spol}(f, g)$ has a standard representation w.r.t. G.

Note

Do not remove too much information! If $\lambda = 1$ and

$$\mathrm{spol}(f, g) = \lambda \mathrm{spol}(f, h) + \sigma \mathrm{spol}(h, g),$$

then we can remove $\mathrm{spol}(f, g)$ or $\mathrm{spol}(f, h)$ but not both!

How to combine Product and Chain criterion?

Gebauer-Möller installation [33]

We add a new element h to G and generate new pairs $P' := \{(f, h) \mid f \in G\}$.

Gebauer-Möller installation [33]

We add a new element h to G and generate new pairs $P' := \{(f, h) \mid f \in G\}$.
We update the pairs in 4 steps:

Gebauer-Möller installation [33]

We add a new element h to G and generate new pairs $P' := \{(f, h) \mid f \in G\}$.
We update the pairs in 4 steps:

1. If $(f, g) \in P$ s.t.

- $\triangleright$ $\mathrm{lt}(h) \mid \mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(g))$,
- $\triangleright$ $\mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(h)) \neq \mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(g))$,
- $\triangleright$ $\mathrm{lcm}(\mathrm{lt}(g), \mathrm{lt}(h)) \neq \mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(g))$

$\Longrightarrow$ Remove (f, g) from P. [P done]

Gebauer-Möller installation [33]

We add a new element h to G and generate new pairs $P' := \{(f, h) \mid f \in G\}$.
We update the pairs in 4 steps:

1. If $(f, g) \in P$ s.t.
   - $\triangleright$ $\mathrm{lt}(h) \mid \mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(g))$,
   - $\triangleright$ $\mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(h)) \neq \mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(g))$,
   - $\triangleright$ $\mathrm{lcm}(\mathrm{lt}(g), \mathrm{lt}(h)) \neq \mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(g))$

   $\Longrightarrow$ Remove (f, g) from P. [P done]

2. Fix $(f, h) \in P'$. If $(g, h) \in P' \setminus \{(f, h)\}$ s.t.
   - $\triangleright$ $\exists \lambda > 1$ and $\mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(h)) = \lambda\, \mathrm{lcm}(\mathrm{lt}(g), \mathrm{lt}(h))$

   $\Longrightarrow$ Remove (g, h) from P'.

Gebauer-Möller installation [33]

We add a new element h to G and generate new pairs $P' := \{(f, h) \mid f \in G\}$.
We update the pairs in 4 steps:

1. If $(f, g) \in P$ s.t.
   - $\triangleright$ $\mathrm{lt}(h) \mid \mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(g))$,
   - $\triangleright$ $\mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(h)) \neq \mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(g))$,
   - $\triangleright$ $\mathrm{lcm}(\mathrm{lt}(g), \mathrm{lt}(h)) \neq \mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(g))$

   $\Longrightarrow$ Remove (f, g) from P. [P done]

2. Fix $(f, h) \in P'$. If $(g, h) \in P' \setminus \{(f, h)\}$ s.t.
   - $\triangleright$ $\exists \lambda > 1$ and $\mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(h)) = \lambda\, \mathrm{lcm}(\mathrm{lt}(g), \mathrm{lt}(h))$

   $\Longrightarrow$ Remove (g, h) from P'.

3. Fix $(f, h) \in P'$. If $(g, h) \in P' \setminus \{(f, h)\}$ s.t.
   - $\triangleright$ $\mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(h)) = \mathrm{lcm}(\mathrm{lt}(g), \mathrm{lt}(h))$

   $\Longrightarrow$ Remove (g, h) from P'. [Chain criterion done]

Gebauer-Möller installation [33]

We add a new element h to G and generate new pairs $P' := \{(f, h) \mid f \in G\}$.
We update the pairs in 4 steps:

1. If $(f, g) \in P$ s.t.
   - $\triangleright$ $\mathrm{lt}(h) \mid \mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(g))$,
   - $\triangleright$ $\mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(h)) \neq \mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(g))$,
   - $\triangleright$ $\mathrm{lcm}(\mathrm{lt}(g), \mathrm{lt}(h)) \neq \mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(g))$

   $\Longrightarrow$ Remove (f, g) from P. [P done]

2. Fix $(f, h) \in P'$. If $(g, h) \in P' \setminus \{(f, h)\}$ s.t.
   - $\triangleright$ $\exists \lambda > 1$ and $\mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(h)) = \lambda\, \mathrm{lcm}(\mathrm{lt}(g), \mathrm{lt}(h))$

   $\Longrightarrow$ Remove (g, h) from P'.

3. Fix $(f, h) \in P'$. If $(g, h) \in P' \setminus \{(f, h)\}$ s.t.
   - $\triangleright$ $\mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(h)) = \mathrm{lcm}(\mathrm{lt}(g), \mathrm{lt}(h))$

   $\Longrightarrow$ Remove (g, h) from P'. [Chain criterion done]

4. If $(f, h) \in P'$ s.t. $\mathrm{lcm}(\mathrm{lt}(f), \mathrm{lt}(h)) = \mathrm{lt}(f)\,\mathrm{lt}(h)$
   $\Longrightarrow$ Remove (f, h) from P'. [Product criterion done]

Can we do even better?

In our example we still need to consider

$$\operatorname{spol}(g_3, g_1) \xrightarrow{G} 0.$$

Can we do even better?

In our example we still need to consider

$$\operatorname{spol}(g_3, g_1) \xrightarrow{G} 0.$$

How to get rid of this useless computation?

Can we do even better?

In our example we still need to consider

$$\operatorname{spol}(g_3, g_1) \xrightarrow{G} 0.$$

How to get rid of this useless computation?

Use more structure of $I \Longrightarrow$ **Signatures**

Signatures

Let $I = \langle f_1, \ldots, f_m \rangle \subset \mathscr{R}$.
Idea: Give each $f \in I$ a bit more structure:

Signatures

Let $I = \langle f_1, \ldots, f_m \rangle \subset \mathscr{R}$.
Idea: Give each $f \in I$ a bit more structure:

1. Let $\mathscr{R}^m$ be generated by $e_1, \ldots, e_m$ and let $\prec$ be a compatible monomial order on the monomials of $\mathscr{R}^m$.

Signatures

Let $I = \langle f_1, \ldots, f_m \rangle \subset \mathscr{R}$.
Idea: Give each $f \in I$ a bit more structure:

1. Let $\mathscr{R}^m$ be generated by $e_1, \ldots, e_m$ and let $\prec$ be a compatible monomial order on the monomials of $\mathscr{R}^m$.

2. Let $\alpha \mapsto \overline{\alpha} : \mathscr{R}^m \to \mathscr{R}$ such that $\overline{e}_i = f_i$ for all i.

Signatures

Let $I = \langle f_1, \ldots, f_m \rangle \subset \mathscr{R}$.
Idea: Give each $f \in I$ a bit more structure:

1. Let $\mathscr{R}^m$ be generated by $e_1, \ldots, e_m$ and let $\prec$ be a compatible monomial order on the monomials of $\mathscr{R}^m$.

2. Let $\alpha \mapsto \overline{\alpha} : \mathscr{R}^m \to \mathscr{R}$ such that $\overline{e}_i = f_i$ for all i.

3. Each $f \in I$ can be represented via some $\alpha \in \mathscr{R}^m$: $f = \overline{\alpha}$

Signatures

Let $I = \langle f_1, \ldots, f_m \rangle \subset \mathscr{R}$.
Idea: Give each $f \in I$ a bit more structure:

1. Let $\mathscr{R}^m$ be generated by $e_1, \ldots, e_m$ and let $\prec$ be a compatible monomial order on the monomials of $\mathscr{R}^m$.

2. Let $\alpha \mapsto \overline{\alpha} : \mathscr{R}^m \to \mathscr{R}$ such that $\overline{e}_i = f_i$ for all i.

3. Each $f \in I$ can be represented via some $\alpha \in \mathscr{R}^m$: $f = \overline{\alpha}$

4. A **signature** of f is given by $\mathfrak{s}(f) = \mathrm{lt}_{\prec}(\alpha)$ where $f = \overline{\alpha}$.

Signatures

Let $I = \langle f_1, \ldots, f_m \rangle \subset \mathscr{R}$.
Idea: Give each $f \in I$ a bit more structure:

1. Let $\mathscr{R}^m$ be generated by $e_1, \ldots, e_m$ and let $\prec$ be a compatible monomial order on the monomials of $\mathscr{R}^m$.

2. Let $\alpha \mapsto \overline{\alpha} : \mathscr{R}^m \to \mathscr{R}$ such that $\overline{e}_i = f_i$ for all i.

3. Each $f \in I$ can be represented via some $\alpha \in \mathscr{R}^m$: $f = \overline{\alpha}$

4. A **signature** of f is given by $\mathfrak{s}(f) = \mathrm{lt}_{\prec}(\alpha)$ where $f = \overline{\alpha}$.

5. An element $\alpha \in \mathscr{R}^m$ such that $\overline{\alpha} = 0$ is called a **syzygy**.

Our example again – with signatures and $\prec_{\mathbf{pot}}$

$$g_1 = xy - z^2,\ \mathfrak{s}(g_1) = e_1,$$
$$g_2 = y^2 - z^2,\ \mathfrak{s}(g_2) = e_2.$$

Our example again – with signatures and $\prec_{\mathbf{pot}}$

$$g_1 = xy - z^2,\ \mathfrak{s}(g_1) = e_1,$$
$$g_2 = y^2 - z^2,\ \mathfrak{s}(g_2) = e_2.$$

$$g_3 = \mathrm{spol}(g_2, g_1) = xg_2 - yg_1$$
$$\Rightarrow \mathfrak{s}(g_3) = x\,\mathfrak{s}(g_2) = xe_2.$$

Our example again – with signatures and $\prec_{\mathbf{pot}}$

$$g_1 = xy - z^2,\ \mathfrak{s}(g_1) = e_1,$$
$$g_2 = y^2 - z^2,\ \mathfrak{s}(g_2) = e_2.$$

$$g_3 = \mathrm{spol}(g_2, g_1) = xg_2 - yg_1$$
$$\Rightarrow \mathfrak{s}(g_3) = x\,\mathfrak{s}(g_2) = xe_2.$$

$$\mathrm{spol}(g_3, g_1) = yg_3 - z^2 g_1$$
$$\Rightarrow \mathfrak{s}\left(\mathrm{spol}(g_3, g_1)\right) = y\,\mathfrak{s}(g_3) = xye_2.$$

Our example again – with signatures and $\prec_{\mathbf{pot}}$

$$g_1 = xy - z^2,\ \mathfrak{s}(g_1) = e_1,$$
$$g_2 = y^2 - z^2,\ \mathfrak{s}(g_2) = e_2.$$

$$g_3 = \mathrm{spol}(g_2, g_1) = xg_2 - yg_1$$
$$\Rightarrow \mathfrak{s}(g_3) = x\,\mathfrak{s}(g_2) = xe_2.$$

$$\mathrm{spol}(g_3, g_1) = yg_3 - z^2 g_1$$
$$\Rightarrow \mathfrak{s}(\mathrm{spol}(g_3, g_1)) = y\,\mathfrak{s}(g_3) = xye_2.$$

Note that $\mathfrak{s}(\mathrm{spol}(g_3, g_1)) = xye_2$ and $\mathrm{lm}(g_1) = xy$.

How to use signatures?

General idea: Only 1 element per signature.

How to use signatures?

General idea: Only 1 element per signature.

Several elements with the same signature?

How to use signatures?

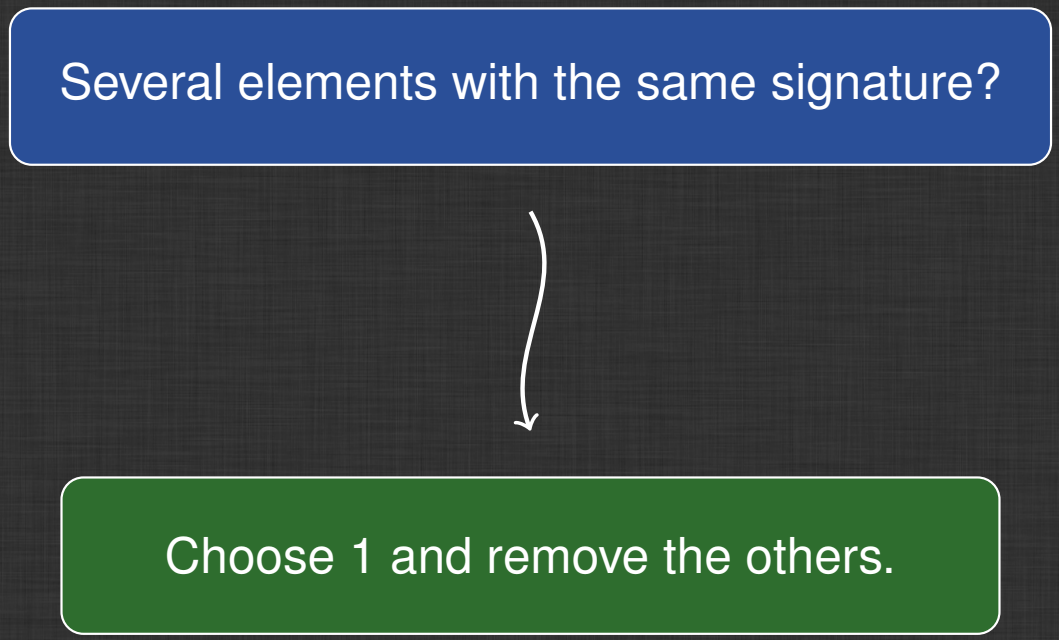
General idea: Only 1 element per signature.

Several elements with the same signature?

Choose 1 and remove the others.

How to use signatures?

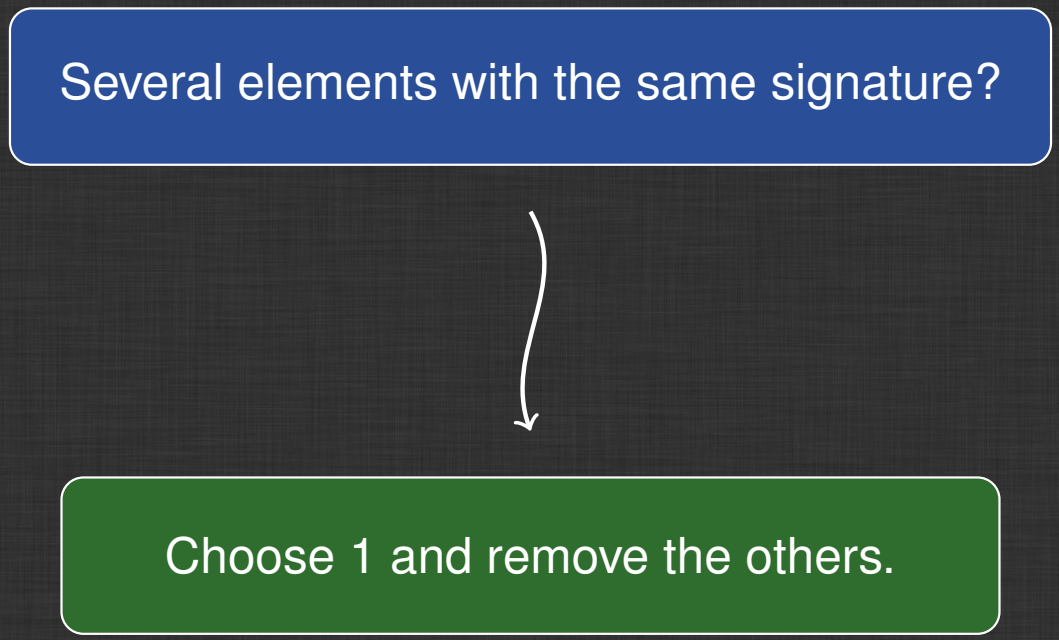
General idea: Only 1 element per signature.



Our goal: Make good choices.

How to use signatures?

General idea: Only 1 element per signature.



Our goal: Make good choices.

Our task: Keep signatures correct.

Think in the module

$\alpha \in \mathscr{R}^m \implies$ polynomial $\overline{\alpha}$ with $\mathrm{lt}(\overline{\alpha})$, signature $\mathfrak{s}(\alpha) = \mathrm{lt}(\alpha)$

Think in the module

$\alpha \in \mathscr{R}^m \implies$ polynomial $\overline{\alpha}$ with $\mathrm{lt}(\overline{\alpha})$, signature $\mathfrak{s}(\alpha) = \mathrm{lt}(\alpha)$

S-pairs/S-polynomials:

$$\mathrm{spol}\left(\overline{\alpha}, \overline{\beta}\right) = a\overline{\alpha} - b\overline{\beta} \implies \mathrm{spair}(\alpha, \beta) = a\alpha - b\beta$$

Think in the module

$\alpha \in \mathscr{R}^m \implies$ polynomial $\overline{\alpha}$ with $\mathrm{lt}(\overline{\alpha})$, signature $\mathfrak{s}(\alpha) = \mathrm{lt}(\alpha)$

S-pairs/S-polynomials:

$$\mathrm{spol}\left(\overline{\alpha}, \overline{\beta}\right) = a\overline{\alpha} - b\overline{\beta} \implies \mathrm{spair}(\alpha, \beta) = a\alpha - b\beta$$

$\mathfrak{s}$-reductions:

$$\overline{\gamma} - d\overline{\delta} \implies \gamma - d\delta$$

Think in the module

$$\alpha \in \mathscr{R}^m \implies \text{polynomial } \overline{\alpha} \text{ with } \mathrm{lt}(\overline{\alpha}), \text{ signature } \mathfrak{s}(\alpha) = \mathrm{lt}(\alpha)$$

S-pairs/S-polynomials:

$$\mathrm{spol}\left(\overline{\alpha}, \overline{\beta}\right) = a\overline{\alpha} - b\overline{\beta} \implies \mathrm{spair}(\alpha, \beta) = a\alpha - b\beta$$

$\mathfrak{s}$-reductions:

$$\overline{\gamma} - d\overline{\delta} \implies \gamma - d\delta$$

Remark

In the following we need one detail from signature-based Gröbner Basis computations:

We pick from P by increasing signature.

Signature-based criteria

$\mathfrak{s}(\alpha) = \mathfrak{s}(\beta) \implies$ Compute 1, remove 1.

Signature-based criteria

$\mathfrak{s}(\alpha) = \mathfrak{s}(\beta) \implies$ Compute 1, remove 1.

Sketch of proof

1. $\mathfrak{s}(\alpha - \beta) \prec \mathfrak{s}(\alpha), \mathfrak{s}(\beta)$.
2. All S-pairs are handled by increasing signature.
   $\Rightarrow$ All relatons $\prec \mathfrak{s}(\alpha)$ are known:

$$\alpha = \beta + \text{ elements of smaller signature}$$

□

Signature-based criteria

S-pairs in signature T

Signature-based criteria

S-pairs in signature T

What are all possible configurations to reach signature T?

Signature-based criteria

S-pairs in signature T

$$\mathfrak{R}_T = \left\{ a\alpha \mid \alpha \text{ handled by the algorithm and } \mathfrak{s}(a\alpha) = T \right\}$$

What are all possible configurations to reach signature T?

Signature-based criteria

S-pairs in signature T

$$\mathfrak{R}_T = \left\{ a\alpha \mid \alpha \text{ handled by the algorithm and } \mathfrak{s}(a\alpha) = T \right\}$$

What are all possible configurations to reach signature T?

Define an order on $\mathfrak{R}_T$ and choose the maximal element.

Special cases

$$\mathfrak{R}_T = \left\{ a\alpha \mid \alpha \text{ handled by the algorithm and } \mathfrak{s}(a\alpha) = T \right\}$$

Special cases

$$\mathfrak{R}_T = \left\{ a\alpha \mid \alpha \text{ handled by the algorithm and } \mathfrak{s}(a\alpha) = T \right\}$$

Choose $b\beta$ to be an element of $\mathfrak{R}_T$ maximal w.r.t. an order $\trianglelefteq$.

Special cases

$$\mathfrak{R}_T = \left\{ a\alpha \mid \alpha \text{ handled by the algorithm and } \mathfrak{s}(a\alpha) = T \right\}$$

Choose $b\beta$ to be an element of $\mathfrak{R}_T$ maximal w.r.t. an order $\trianglelefteq$.

1. If $b\beta$ is a syzygy $\Longrightarrow$ Go on to next signature.

Special cases

$$\mathfrak{R}_T = \left\{ a\alpha \mid \alpha \text{ handled by the algorithm and } \mathfrak{s}(a\alpha) = T \right\}$$

Choose $b\beta$ to be an element of $\mathfrak{R}_T$ maximal w.r.t. an order $\trianglelefteq$.

1. If $b\beta$ is a syzygy $\Longrightarrow$ Go on to next signature.
2. If $b\beta$ is not part of an S-pair $\Longrightarrow$ Go on to next signature.

Special cases

$$\mathfrak{R}_T = \left\{ a\alpha \mid \alpha \text{ handled by the algorithm and } \mathfrak{s}(a\alpha) = T \right\}$$

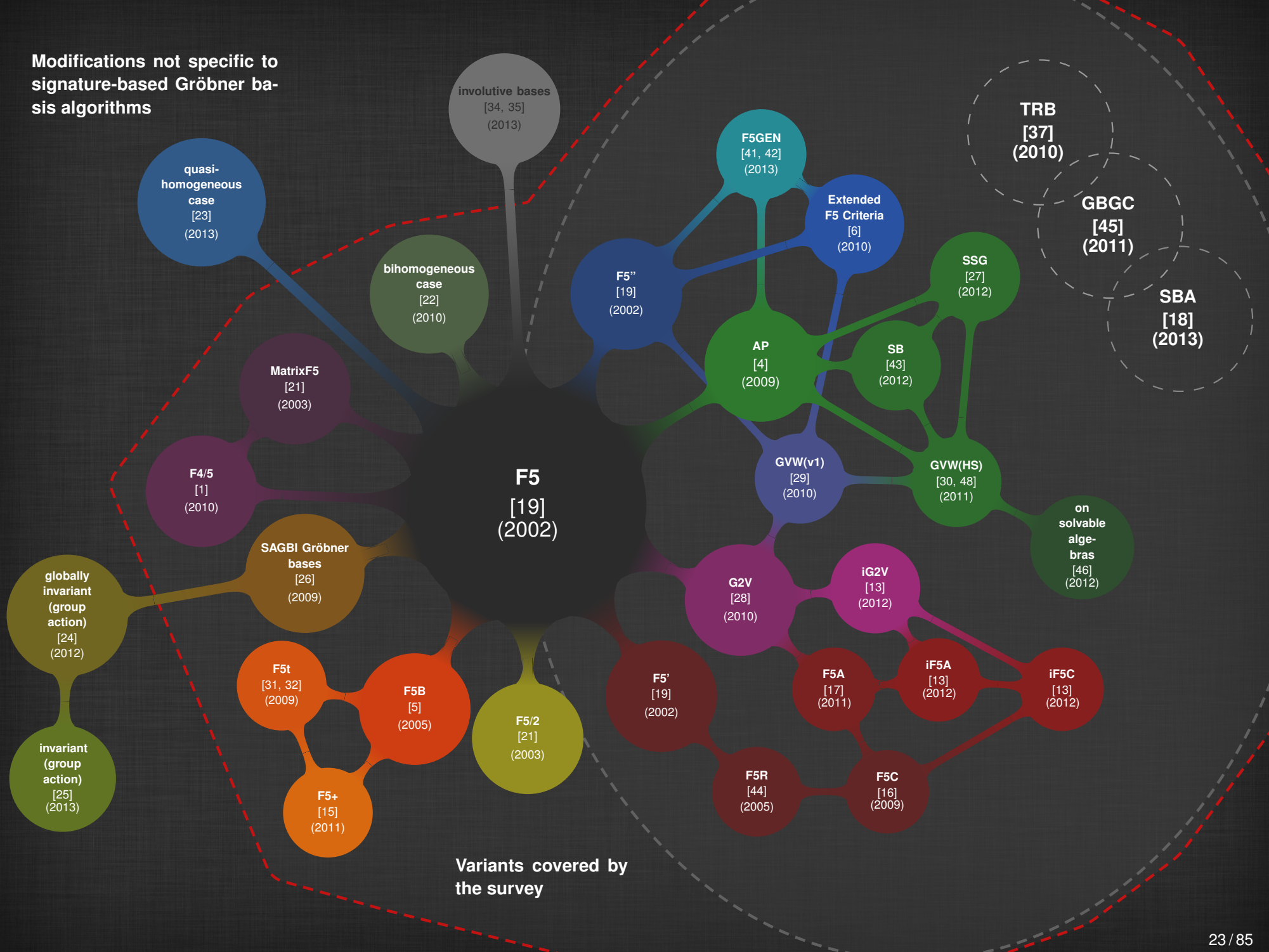
Choose $b\beta$ to be an element of $\mathfrak{R}_T$ maximal w.r.t. an order $\trianglelefteq$.

1. If $b\beta$ is a syzygy $\Longrightarrow$ Go on to next signature.
2. If $b\beta$ is not part of an S-pair $\Longrightarrow$ Go on to next signature.

Revisiting our example with $\prec_{\text{pot}}$

$$\mathfrak{s}\left(\text{spol}(g_3, g_1)\right) = xye_2$$

$$\left.\begin{aligned} g_1 &= xy - z^2 \\ g_2 &= y^2 - z^2 \end{aligned}\right\} \Rightarrow \text{psyz}(g_2, g_1) = g_1 e_2 - g_2 e_1 = xye_2 + \ldots$$


Modifications not specific to signature-based Gröbner basis algorithms
involutive bases [34, 35] (2013)
quasi-homogeneous case [23] (2013)
bihomogeneous case [22] (2010)
F5GEN [41, 42] (2013)
Extended F5 Criteria [6] (2010)
F5" [19] (2002)
SSG [27] (2012)
AP [4] (2009)
SB [43] (2012)
TRB [37] (2010)
GBGC [45] (2011)
SBA [18] (2013)
MatrixF5 [21] (2003)
F4/5 [1] (2010)
F5 [19] (2002)
GVW(v1) [29] (2010)
GVW(HS) [30, 48] (2011)
on solvable algebras [46] (2012)
SAGBI Gröbner bases [26] (2009)
globally invariant (group action) [24] (2012)
G2V [28] (2010)
iG2V [13] (2012)
F5t [31, 32] (2009)
F5B [5] (2005)
F5/2 [21] (2003)
F5' [19] (2002)
F5A [17] (2011)
iF5A [13] (2012)
iF5C [13] (2012)
invariant (group action) [25] (2013)
F5+ [15] (2011)
F5R [44] (2005)
F5C [16] (2009)
Variants covered by the survey

zero reductions (Singular-4-0-0, $\mathbb{F}_{32003}$)

Benchmark	STD	SBA $<_{\text{pot}}$	SBA $<_{\text{d-pot}}$	SBA $<_{\text{lt}}$
cyclic-8	4,284	243	243	671
cyclic-8-h	5,843	243	243	671
eco-11	3,476	0	749	749
eco-11-h	5,429	502	502	749
katsura-11	3,933	0	0	353
katsura-11-h	3,933	0	0	353
noon-9	25,508	0	0	682
noon-9-h	25,508	0	0	682
Random$(11,2,2)$	6,292	0	0	590
HRandom$(11,2,2)$	6,292	0	0	590
Random$(12,2,2)$	13,576	0	0	1,083
HRandom$(12,2,2)$	13,576	0	0	1,083

Time in seconds (Singular-4-0-0, $\mathbb{F}_{32003}$)

Benchmark	STD	SBA $<_{\text{pot}}$	SBA $<_{\text{d-pot}}$	SBA $<_{\text{lt}}$
cyclic-8	32.480	44.310	100.780	38.120
cyclic-8-h	38.300	35.770	98.440	32.640
eco-11	28.450	3.450	27.360	13.270
eco-11-h	20.630	11.600	14.840	7.960
katsura-11	54.780	35.720	31.010	11.790
katsura-11-h	51.260	34.080	32.590	17.230
noon-9	29.730	12.940	14.620	15.220
noon-9-h	34.410	17.850	20.090	20.510
Random(11,2,2)	267.810	77.430	130.400	28.640
HRandom(11,2,2)	22.970	14.060	39.320	3.540
Random(12,2,2)	2,069.890	537.340	1,062.390	176.920
HRandom(12,2,2)	172.910	112.420	331.680	22.060

Can we combine both attempts?

Yes, rather easily:

- Chain criterion included in Rewritten criterion
- Product criterion not completely included

Can we combine both attempts?

Yes, rather easily:

- Chain criterion included in Rewritten criterion
- Product criterion not completely included

1. Check Rewritten criterion (prefer signature-based stuff)
2. Check Product criterion (i.e. add new syzygy not found beforehand)

Can we combine both attempts?

Yes, rather easily:

- Chain criterion included in Rewritten criterion
- Product criterion not completely included

1. Check Rewritten criterion (prefer signature-based stuff)
2. Check Product criterion (i.e. add new syzygy not found beforehand)

Note

There is a conjecture that for $<_{\text{pot}}$ the Product criterion is also included in the Rewritten criterion [14].

- Predicting zero reductions
- Pair set sorting and homogenization
- Finding better reducers
- Fast linear algebra for computing Gröbner bases
- Modular computations
- Change of ordering

Why is homogeneous input nice?

If $\deg(\operatorname{spol}(f,g)) = d$ and $\operatorname{spol}(f,g) \xrightarrow{G} h \neq 0$

Why is homogeneous input nice?

$$\text{If } \deg(\text{spol}(f,g)) = d \text{ and } \text{spol}(f,g) \xrightarrow{G} h \neq 0$$

$$\Longrightarrow \deg(h) = d$$

Why is homogeneous input nice?

$$\text{If } \deg(\operatorname{spol}(f,g)) = d \text{ and } \operatorname{spol}(f,g) \xrightarrow{G} h \neq 0$$

$$\Longrightarrow \deg(h) = d$$

No degree drop, we can compute by increasing degree!

Why is homogeneous input nice?

$$\text{If } \deg(\operatorname{spol}(f,g)) = d \text{ and } \operatorname{spol}(f,g) \xrightarrow{G} h \neq 0$$

$$\Longrightarrow \deg(h) = d$$

No degree drop, we can compute by increasing degree!

Normal selection strategy

Take a subset $P_d \subset P$ of pairs of lowest possible degree d.
Reduce all elements in P_d, then go on with the rest in P.

Why is homogeneous input nice?

$$\text{If } \deg(\operatorname{spol}(f,g)) = d \text{ and } \operatorname{spol}(f,g) \xrightarrow{G} h \neq 0$$

$$\Longrightarrow \deg(h) = d$$

No degree drop, we can compute by increasing degree!

Normal selection strategy

Take a subset $P_d \subset P$ of pairs of lowest possible degree d.
Reduce all elements in P_d, then go on with the rest in P.

All newly generated S-polynomials have degree $> d$.
All possible lower-degree reducers are already in G.

And if the input is not homogeneous?

$$\text{If } \deg(\operatorname{spol}(f,g)) = d \text{ and } \operatorname{spol}(f,g) \xrightarrow{G} h \neq 0$$

And if the input is not homogeneous?

$$\text{If } \deg(\operatorname{spol}(f,g)) = d \text{ and } \operatorname{spol}(f,g) \xrightarrow{G} h \neq 0$$

$$\Longrightarrow \deg(h) \leq d$$

And if the input is not homogeneous?

$$\text{If } \deg(\operatorname{spol}(f,g)) = d \text{ and } \operatorname{spol}(f,g) \xrightarrow{G} h \neq 0$$

$$\Longrightarrow \deg(h) \leq d$$

If $\deg(h) < d$ then

1. $\deg(\operatorname{spol}(h,h'))$ might be $< d$,
2. h is a possible reducer of elements already in G.

And if the input is not homogeneous?

$$\text{If } \deg(\operatorname{spol}(f,g)) = d \text{ and } \operatorname{spol}(f,g) \xrightarrow{G} h \neq 0$$

$$\Longrightarrow \deg(h) \leq d$$

If $\deg(h) < d$ then

1. $\deg(\operatorname{spol}(h, h'))$ might be $< d$,
2. h is a possible reducer of elements already in G.

Still, choosing by the normal selection strategy is quite good.

How to improve the inhomogeneous situation?

1. Homogenize the input:
   - ▷ Better behaviour
   - ▷ Might add overhead due to solutions at infinity: $\#G^h > \#G$

How to improve the inhomogeneous situation?

1. Homogenize the input:
   - ▷ Better behaviour
   - ▷ Might add overhead due to solutions at infinity: $\#G^h > \#G$

2. Mimic homogeneous computations:
   - ▷ Pairs are sorted like being homogeneous
   - ▷ Computations are done with inhomogeneous elements

How to improve the inhomogeneous situation?

1. Homogenize the input:
   - ▷ Better behaviour
   - ▷ Might add overhead due to solutions at infinity: $\#G^h > \#G$

2. Mimic homogeneous computations:
   - ▷ Pairs are sorted like being homogeneous
   - ▷ Computations are done with inhomogeneous elements

Sugar-degree [36]
Let $I = \langle f_1, \ldots, f_m \rangle$.

$$\begin{aligned} \text{s-deg}(f_i) &:= \deg(f_i) \\ \text{s-deg}(tg) &:= \deg(t) + \deg(g) \text{ for } t \in M \text{ and } g \in G \\ \text{s-deg}(g+h) &:= \max\{\text{s-deg}(g), \text{s-deg}(h)\} \text{ for } g, h \in G. \end{aligned}$$

- Predicting zero reductions
- Pair set sorting and homogenization
- **Finding better reducers**
- Fast linear algebra for computing Gröbner bases
- Modular computations
- Change of ordering

Can we pick good reducers?

$$f - \lambda g$$

where f is an intermediate S-polynomial, $g \in G$.

Can we pick good reducers?

$$f - \lambda g$$

where f is an intermediate S-polynomial, $g \in G$.

What if $\lambda = 1$? After the reduction step we have 2 elements:

$f - g$ and g

But we could also take

$f - g$ and f

Can we pick good reducers?

$$f - \lambda g$$

where f is an intermediate S-polynomial, $g \in G$.

What if $\lambda = 1$? After the reduction step we have 2 elements:

$f - g$ and g

But we could also take

$f - g$ and f

If f has **better properties** then exchange g with f in G.

And if $\lambda > 1$?

If f has better properties than $\lambda g \Rightarrow f \in R$ (special reducer set)

And if $\lambda > 1$?

If f has better properties than $\lambda g \Rightarrow f \in R$ (special reducer set)

Example

$$\mathrm{lt}(f) = \mathrm{lt}(\lambda g)$$

Let h be another intermediate S-polynomial such that

- $\exists\, \lambda' \in \mathrm{Mon}$ with $\lambda' \mathrm{lt}(g) = \mathrm{lt}(h)$
- $\lambda \mid \lambda'$

Then we replace the reduction

$$h - \lambda' g = h - \frac{\lambda'}{\lambda} \lambda g$$

by

$$h - \frac{\lambda'}{\lambda} f$$

And if $\lambda > 1$?

If f has better properties than $\lambda g \Rightarrow f \in R$ (special reducer set)

Example

$$\mathrm{lt}(f) = \mathrm{lt}(\lambda g)$$

Let h be another intermediate S-polynomial such that

- $\exists\, \lambda' \in \mathrm{Mon}$ with $\lambda' \mathrm{lt}(g) = \mathrm{lt}(h)$
- $\lambda \mid \lambda'$

Then we replace the reduction

$$h - \lambda' g = h - \frac{\lambda'}{\lambda} \lambda g$$

by

$$h - \frac{\lambda'}{\lambda} f$$

In the same way:
Exchanging multiples of λg when generating S-polynomials.

How to implement this?

1. What is good in the given instance?
2. How often do we check?
3. How many special reducers do we store?

How to implement this?

1. What is good in the given instance?
2. How often do we check?
3. How many special reducers do we store?

There are three algorithms using this:

- **slimGB** (SINGULAR); to some extend also **std**
- Faugère's **F4** algorithm (everywhere but not in SINGULAR)
- Signature-based algorithms (**sba** in SINGULAR, **GVW**, **F5** in FGB)
  Checking reducers with the Rewritten criterion

- Predicting zero reductions
- Pair set sorting and homogenization
- Finding better reducers
- **Fast linear algebra for computing Gröbner bases**
- Modular computations
- Change of ordering

Buchberger's algorithm - revisited

Input: Ideal $I = \langle f_1, \ldots, f_m \rangle$
Output: Gröbner basis G for I

1. $G \leftarrow \emptyset$
2. $G \leftarrow G \cup \{f_i\}$ for all $i \in \{1, \ldots, m\}$
3. Set $P \leftarrow \{\text{spol}(f_i, f_j) \mid f_i, f_j \in G, i > j\}$
4. Choose $p \in P$, $P \leftarrow P \setminus \{p\}$
   (a) If $p \xrightarrow{G} 0$ ▶▶ **no new information**
   Go on with the next element in P.
   (b) If $p \xrightarrow{G} h \neq 0$ ▶▶ **new information**
   Build new S-pair with h and add them to P.
   Add h to G.
   Go on with the next element in P.
5. When $P = \emptyset$ we are done and G is a Gröbner basis of I.

Faugère's F4 algorithm

Input: Ideal $I = \langle f_1, \ldots, f_m \rangle$
Output: Gröbner basis G for I

1. $G \leftarrow \emptyset$
2. $G \leftarrow G \cup \{f_i\}$ for all $i \in \{1, \ldots, m\}$
3. Set $P \leftarrow \{(af, bg) \mid f, g \in G\}$
4. $d \leftarrow 0$
5. while $P \neq \emptyset$:

Faugère's F4 algorithm

Input: Ideal $I = \langle f_1, \ldots, f_m \rangle$
Output: Gröbner basis G for I

1. $G \leftarrow \emptyset$
2. $G \leftarrow G \cup \{f_i\}$ for all $i \in \{1, \ldots, m\}$
3. Set $P \leftarrow \{(af, bg) \mid f, g \in G\}$
4. $d \leftarrow 0$
5. while $P \neq \emptyset$:
   (a) $d \leftarrow d + 1$
   (b) $P_d \leftarrow \textbf{Select}(P)$, $P \leftarrow P \setminus P_d$
   (c) $L_d \leftarrow \{af, bg \mid (af, bg) \in P_d\}$
   (d) $L_d \leftarrow \textbf{Symbolic Preprocessing}(L_d, G)$
   (e) $F_d \leftarrow \textbf{Reduction}(L_d, G)$
   (f) for $h \in F_d$:
      - If $\mathrm{lt}(h) \notin L(G)$ (all other h are useless):
        - $P \leftarrow P \cup \{\text{new pairs with } h\}$
        - $G \leftarrow G \cup \{h\}$
6. Return G

Differences to Buchberger

1. Select a subset P_d of P, not only one element.
2. Do a symbolic preprocessing:
   Search and store reducers, but do not reduce.
3. Do a full reduction of P_d at once:
   Reduce a subset of $\mathscr{R}$ by a subset of $\mathscr{R}$

Differences to Buchberger

1. Select a subset P_d of P, not only one element.
2. Do a symbolic preprocessing:
   Search and store reducers, but do not reduce.
3. Do a full reduction of P_d at once:
   Reduce a subset of $\mathscr{R}$ by a subset of $\mathscr{R}$

If **Select**(P) selects only 1 pair F4 is just Buchberger's algorithm.
Usually one chooses the normal selection strategy,
i.e. all pairs of lowest degree.

Symbolic preprocessing

Input: L, G finite subsets of $\mathscr{R}$
Output: a finite subset of $\mathscr{R}$

1. $F \leftarrow L$
2. $D \leftarrow L(F)$ (S-pairs already reduce lead terms)
3. while $T(F) \neq D$:
   (a) Choose $m \in T(F) \setminus D$, $D \leftarrow D \cup \{m\}$.
   (b) If $m \in L(G) \Rightarrow \exists\, g \in G$ and $\lambda \in$ Mon such that $\lambda\, \mathrm{lt}(g) = m$
   $\triangleright\ F \leftarrow F \cup \{\lambda g\}$
4. Return F

Symbolic preprocessing

Input: L, G finite subsets of $\mathscr{R}$
Output: a finite subset of $\mathscr{R}$

1. $F \leftarrow L$
2. $D \leftarrow L(F)$ (S-pairs already reduce lead terms)
3. while $T(F) \neq D$:
   (a) Choose $m \in T(F) \setminus D$, $D \leftarrow D \cup \{m\}$.
   (b) If $m \in L(G) \Rightarrow \exists\, g \in G$ and $\lambda \in \mathrm{Mon}$ such that $\lambda\, \mathrm{lt}(g) = m$
   $\triangleright$ $F \leftarrow F \cup \{\lambda g\}$
4. Return F

We optimize this soon!

Reduction

Input: L, G finite subsets of $\mathscr{R}$
Output: a finite subset of $\mathscr{R}$

1. $M \leftarrow$ Macaulay matrix of L
2. $M \leftarrow$ Gaussian Elimination of M (Linear algebra)
3. $F \leftarrow$ polynomials from rows of M
4. Return F

Reduction

Input: L, G finite subsets of $\mathscr{R}$
Output: a finite subset of $\mathscr{R}$

1. $M \leftarrow$ Macaulay matrix of L
2. $M \leftarrow$ Gaussian Elimination of M (Linear algebra)
3. $F \leftarrow$ polynomials from rows of M
4. Return F

Macaulay matrix

columns $\hat{=}$ monomials (sorted by monomial order $<$)

rows $\hat{=}$ coeffs of polynomials in L

Example: Cyclic-4

$\mathscr{R} = \mathbb{Q}[a, b, c, d]$, $<$ denotes DRL and we use the normal selection strategy for **Select**(P). $I = \langle f_1, \ldots, f_4 \rangle$, where

$$
\begin{aligned}
f_1 &= abcd - 1, \\
f_2 &= abc + abd + acd + bcd, \\
f_3 &= ab + bc + ad + cd, \\
f_4 &= a + b + c + d.
\end{aligned}
$$

Example: Cyclic-4

$\mathscr{R} = \mathbb{Q}[a, b, c, d]$, $<$ denotes DRL and we use the normal selection strategy for **Select**(P). $I = \langle f_1, \ldots, f_4 \rangle$, where

$$\begin{aligned} f_1 &= abcd - 1, \\ f_2 &= abc + abd + acd + bcd, \\ f_3 &= ab + bc + ad + cd, \\ f_4 &= a + b + c + d. \end{aligned}$$

We start with $G = \{f_4\}$ and $P_1 = \{(f_3, bf_4)\}$, thus $L_1 = \{f_3, bf_4\}$.

Example: Cyclic-4

$\mathscr{R} = \mathbb{Q}[a,b,c,d]$, $<$ denotes DRL and we use the normal selection strategy for **Select**(P). $I = \langle f_1, \ldots, f_4 \rangle$, where

$$\begin{aligned} f_1 &= abcd - 1, \\ f_2 &= abc + abd + acd + bcd, \\ f_3 &= ab + bc + ad + cd, \\ f_4 &= a + b + c + d. \end{aligned}$$

We start with $G = \{f_4\}$ and $P_1 = \{(f_3, bf_4)\}$, thus $L_1 = \{f_3, bf_4\}$.
Let us do symbolic preprocessing:

$$\begin{aligned} T(L_1) &= \{ab, b^2, bc, ad, bd, cd \quad\} \\ L_1 &= \{f_3, bf_4 \quad\} \end{aligned}$$

Example: Cyclic-4

$\mathscr{R} = \mathbb{Q}[a,b,c,d]$, $<$ denotes DRL and we use the normal selection strategy for **Select**(P). $I = \langle f_1, \ldots, f_4 \rangle$, where

$$\begin{aligned} f_1 &= abcd - 1, \\ f_2 &= abc + abd + acd + bcd, \\ f_3 &= ab + bc + ad + cd, \\ f_4 &= a + b + c + d. \end{aligned}$$

We start with $G = \{f_4\}$ and $P_1 = \{(f_3, bf_4)\}$, thus $L_1 = \{f_3, bf_4\}$.
Let us do symbolic preprocessing:

$$\begin{aligned} T(L_1) &= \{ab, b^2, bc, ad, bd, cd \quad\} \\ L_1 &= \{f_3, bf_4 \quad\} \end{aligned}$$

$b^2 \notin L(G)$,

Example: Cyclic-4

$\mathscr{R} = \mathbb{Q}[a,b,c,d]$, $<$ denotes DRL and we use the normal selection strategy for **Select**(P). $I = \langle f_1, \ldots, f_4 \rangle$, where

$$\begin{aligned} f_1 &= abcd - 1, \\ f_2 &= abc + abd + acd + bcd, \\ f_3 &= ab + bc + ad + cd, \\ f_4 &= a + b + c + d. \end{aligned}$$

We start with $G = \{f_4\}$ and $P_1 = \{(f_3, bf_4)\}$, thus $L_1 = \{f_3, bf_4\}$.
Let us do symbolic preprocessing:

$$\begin{aligned} T(L_1) &= \{ab, b^2, bc, ad, bd, cd \quad\} \\ L_1 &= \{f_3, bf_4 \quad\} \end{aligned}$$

$b^2 \notin L(G)$, $bc \notin L(G)$,

Example: Cyclic-4

$\mathscr{R} = \mathbb{Q}[a,b,c,d]$, $<$ denotes DRL and we use the normal selection strategy for **Select**(P). $I = \langle f_1, \ldots, f_4 \rangle$, where

$$\begin{aligned} f_1 &= abcd - 1, \\ f_2 &= abc + abd + acd + bcd, \\ f_3 &= ab + bc + ad + cd, \\ f_4 &= a + b + c + d. \end{aligned}$$

We start with $G = \{f_4\}$ and $P_1 = \{(f_3, bf_4)\}$, thus $L_1 = \{f_3, bf_4\}$.
Let us do symbolic preprocessing:

$$\begin{aligned} T(L_1) &= \{ab, b^2, bc, ad, bd, cd, d^2\} \\ L_1 &= \{f_3, bf_4, df_4\} \end{aligned}$$

$b^2 \notin L(G)$, $bc \notin L(G)$, $d\,\mathrm{lt}(f_4) = ad$,

Example: Cyclic-4

$\mathscr{R} = \mathbb{Q}[a,b,c,d]$, $<$ denotes DRL and we use the normal selection strategy for **Select**(P). $I = \langle f_1, \ldots, f_4 \rangle$, where

$$\begin{aligned} f_1 &= abcd - 1, \\ f_2 &= abc + abd + acd + bcd, \\ f_3 &= ab + bc + ad + cd, \\ f_4 &= a + b + c + d. \end{aligned}$$

We start with $G = \{f_4\}$ and $P_1 = \{(f_3, bf_4)\}$, thus $L_1 = \{f_3, bf_4\}$.
Let us do symbolic preprocessing:

$$\begin{aligned} T(L_1) &= \{ab, b^2, bc, ad, bd, cd, d^2\} \\ L_1 &= \{f_3, bf_4, df_4\} \end{aligned}$$

$b^2 \notin L(G)$, $bc \notin L(G)$, $d\,\mathrm{lt}(f_4) = ad$, all others also $\notin L(G)$,

Example: Cyclic-4

Now reduction:
Convert polynomial data L_1 to Macaulay Matrix M_1

$$\begin{array}{c} \\ df_4 \\ f_3 \\ bf_4 \end{array} \begin{array}{c} \begin{array}{ccccccc} ab & b^2 & bc & ad & bd & cd & d^2 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

Example: Cyclic-4

Now reduction:
Convert polynomial data L_1 to Macaulay Matrix M_1

$$\begin{array}{c} \\ df_4 \\ f_3 \\ bf_4 \end{array} \begin{array}{c} \begin{array}{ccccccc} ab & b^2 & bc & ad & bd & cd & d^2 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

Gaussian Elimination of M_1:

$$\begin{array}{c} \\ df_4 \\ f_3 \\ bf_4 \end{array} \begin{array}{c} \begin{array}{ccccccc} ab & b^2 & bc & ad & bd & cd & d^2 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & -1 & 0 & -1 \\ 0 & 1 & 0 & 0 & 2 & 0 & 1 \end{pmatrix} \end{array}$$

Example: Cyclic-4

Convert matrix data back to polynomial structure F_1:

$$\begin{array}{c} \\ df_4 \\ f_3 \\ bf_4 \end{array} \begin{array}{c} \begin{array}{ccccccc} ab & b^2 & bc & ad & bd & cd & d^2 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & -1 & 0 & -1 \\ 0 & 1 & 0 & 0 & 2 & 0 & 1 \end{pmatrix} \end{array}$$

$$F_1 = \left\{ \underbrace{ad + bd + cd + d^2}_{f_5}, \underbrace{ab + bc - bd - d^2}_{f_6}, \underbrace{b^2 + 2bd + d^2}_{f_7} \right\}$$

Example: Cyclic-4

Convert matrix data back to polynomial structure F_1:

$$\begin{array}{c} \\ df_4 \\ f_3 \\ bf_4 \end{array} \begin{array}{c} \begin{array}{ccccccc} ab & b^2 & bc & ad & bd & cd & d^2 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & -1 & 0 & -1 \\ 0 & 1 & 0 & 0 & 2 & 0 & 1 \end{pmatrix} \end{array}$$

$$F_1 = \left\{ \underbrace{ad + bd + cd + d^2}_{f_5}, \underbrace{ab + bc - bd - d^2}_{f_6}, \underbrace{b^2 + 2bd + d^2}_{f_7} \right\}$$

$\mathrm{lt}(f_5), \mathrm{lt}(f_6) \in L(G)$, so

$\mathbf{G} \leftarrow \mathbf{G} \cup \{\mathbf{f_7}\}$.

Example: Cyclic-4

Next round:

$$G = \{f_4, f_7\},\ P_2 = \{(f_2, bcf_4)\},\ L_2 = \{f_2, bcf_4\}.$$

Example: Cyclic-4

Next round:

$$G = \{f_4, f_7\},\ P_2 = \{(f_2, bcf_4)\},\ L_2 = \{f_2, bcf_4\}.$$

We can simplify the computations:

$$\mathrm{lt}(bcf_4) = abc = \mathrm{lt}(cf_6).$$

f_6 possibly better reduced than f_4.(f_6 is not in G!)

$$\Longrightarrow\ L_2 = \{f_2, cf_6\}$$

Example: Cyclic-4

Next round:

$$G = \{f_4, f_7\},\ P_2 = \{(f_2, bcf_4)\},\ L_2 = \{f_2, bcf_4\}.$$

We can simplify the computations:

$$\mathrm{lt}(bcf_4) = abc = \mathrm{lt}(cf_6).$$

f_6 possibly better reduced than f_4.(f_6 is not in G!)

$$\Longrightarrow\ L_2 = \{f_2, cf_6\}$$

Symbolic preprocessing:

$$\begin{aligned} T(L_2) &= \{abc, bc^2, abd, acd, bcd, cd^2\} \\ L_2 &= \{f_2, cf_6, \quad\} \end{aligned}$$

Example: Cyclic-4

Next round:

$$G = \{f_4, f_7\},\ P_2 = \{(f_2, bcf_4)\},\ L_2 = \{f_2, bcf_4\}.$$

We can simplify the computations:

$$\mathrm{lt}(bcf_4) = abc = \mathrm{lt}(cf_6).$$

f_6 possibly better reduced than f_4.(f_6 is not in G!)

$$\Longrightarrow\ L_2 = \{f_2, cf_6\}$$

Symbolic preprocessing:

$$\begin{aligned} T(L_2) &= \{abc, bc^2, abd, acd, bcd, cd^2\} \\ L_2 &= \{f_2, cf_6, \quad\} \end{aligned}$$

$bc^2 \notin L(G)$,

Example: Cyclic-4

Next round:

$$G = \{f_4, f_7\},\ P_2 = \{(f_2, bcf_4)\},\ L_2 = \{f_2, bcf_4\}.$$

We can simplify the computations:

$$\mathrm{lt}(bcf_4) = abc = \mathrm{lt}(cf_6).$$

f_6 possibly better reduced than f_4.(f_6 is not in G!)

$$\Longrightarrow\ L_2 = \{f_2, cf_6\}$$

Symbolic preprocessing:

$$\begin{aligned} T(L_2) &= \{abc, bc^2, abd, acd, bcd, cd^2\} \\ L_2 &= \{f_2, cf_6, \quad\} \end{aligned}$$

$bc^2 \notin L(G)$, $abd = \mathrm{lt}(bdf_4)$, but also $abd = \mathrm{lt}(bf_5)$!

Example: Cyclic-4

Next round:

$$G = \{f_4, f_7\},\ P_2 = \{(f_2, bcf_4)\},\ L_2 = \{f_2, bcf_4\}.$$

We can simplify the computations:

$$\mathrm{lt}(bcf_4) = abc = \mathrm{lt}(cf_6).$$

f_6 possibly better reduced than f_4.(f_6 is not in G!)

$$\Longrightarrow\ L_2 = \{f_2, cf_6\}$$

Symbolic preprocessing:

$$\begin{aligned} T(L_2) &= \{abc, bc^2, abd, acd, bcd, cd^2\} \\ L_2 &= \{f_2, cf_6, \quad\} \end{aligned}$$

$bc^2 \notin L(G)$, $abd = \mathrm{lt}(bdf_4)$, but also $abd = \mathrm{lt}(bf_5)$!

Let us investigate this in more detail.

Interlude – Simplify

Idea

Try to replace $u \cdot f$ by a product $(wv) \cdot g$ where vg corresponds to an already computed row in the Gauss. Elim. of a previous matrix M_i.

$\Rightarrow$ Reuse rows that are reduced but not “in” G.

Interlude – Simplify

Idea

Try to replace $u \cdot f$ by a product $(wv) \cdot g$ where vg corresponds to an already computed row in the Gauss. Elim. of a previous matrix M_i.
$\Rightarrow$ Reuse rows that are reduced but not "in" G.

Input: monomial u, polynomial f, list $\mathscr{F}$ of old F_i (from M_i after Gauss. Elim.)
Output: product $v \cdot g$ replacing $u \cdot f$

Interlude – Simplify

Idea

Try to replace $u \cdot f$ by a product $(wv) \cdot g$ where vg corresponds to an already computed row in the Gauss. Elim. of a previous matrix M_i.
$\Rightarrow$ Reuse rows that are reduced but not "in" G.

Input: monomial u, polynomial f, list $\mathscr{F}$ of old F_i (from M_i after Gauss. Elim.)
Output: product $v \cdot g$ replacing $u \cdot f$

1. $d \leftarrow$ current index in the F4 algorithm
2. $D(u) \leftarrow \{\text{list of divisors of } u\}$
3. for $w \in D(u)$
   (a) if $\exists j \in \{1, \ldots, d-1\}$ such that $w \cdot f$ corresponds to row in M_j
      ▷ $\exists_1 g \in F_j$ such that $\mathrm{lt}(g) = \mathrm{lt}(w \cdot f)$
      ▷ if $w \neq u$: Return **Simplify** $\left(\frac{u}{w}, g, \mathscr{F}\right)$ (recursive call)
      ▷ else: Return $1 \cdot g$
4. else: Return $u \cdot f$

Interlude – Simplify

Note

- Tries to reuse all rows from old matrices.
  $\Rightarrow$ We need to keep them in memory.
- We also simplify generators of S-pairs, as we have done in our example: $(f_2, bcf_4) \Longrightarrow (f_2, cf_6)$.
- One can also choose “better” reducers by other properties, not only “last reduced one”.
- Without **Simplify** the F4 algorithm is rather slow.

Interlude – Simplify

Note

- Tries to reuse all rows from old matrices.
  $\Rightarrow$ We need to keep them in memory.
- We also simplify generators of S-pairs, as we have done in our example: $(f_2, bcf_4) \Longrightarrow (f_2, cf_6)$.
- One can also choose “better” reducers by other properties, not only “last reduced one”.
- Without **Simplify** the F4 algorithm is rather slow.

In our example:
Choose bf_5 as reducer, not bdf_4.

Example: Cyclic-4

Symbolic preprocessing - now with simplify:

$$\begin{aligned} T(L_2) &= \{abc, bc^2, abd, acd, bcd, cd^2 \} \\ L_2 &= \{f_2, cf_6 \} \end{aligned}$$

$bc^2 \notin L(G)$,

Example: Cyclic-4

Symbolic preprocessing - now with simplify:

$$\begin{aligned} T(L_2) &= \{abc, bc^2, abd, acd, bcd, cd^2 \} \\ L_2 &= \{f_2, cf_6 \} \end{aligned}$$

$bc^2 \notin L(G)$, $abd = \mathrm{lt}(bf_5)$,

Example: Cyclic-4

Symbolic preprocessing - now with simplify:

$$\begin{aligned} T(L_2) &= \{abc, bc^2, abd, acd, bcd, cd^2, b^2d, c^2d \quad\} \\ L_2 &= \{f_2, cf_6, bf_5 \quad\} \end{aligned}$$

$bc^2 \notin L(G)$, $abd = \mathrm{lt}(bf_5)$,

Example: Cyclic-4

Symbolic preprocessing - now with simplify:

$$\begin{aligned} T(L_2) &= \{abc, bc^2, abd, acd, bcd, cd^2, b^2d, c^2d, \dots\} \\ L_2 &= \{f_2, cf_6, bf_5, cf_5, df_7\} \end{aligned}$$

$bc^2 \notin L(G)$, $abd = \mathrm{lt}(bf_5)$, and so on.

Example: Cyclic-4

Symbolic preprocessing - now with simplify:

$$\begin{aligned} T(L_2) &= \{abc, bc^2, abd, acd, bcd, cd^2, b^2d, c^2d, \dots\} \\ L_2 &= \{f_2, cf_6, bf_5, cf_5, df_7\} \end{aligned}$$

$bc^2 \notin L(G)$, $abd = \mathrm{lt}(bf_5)$, and so on.

Now try to exploit the special structure of the Macaulay matrices.

Improve Gaussian Elimination

Use **Linear Algebra** for reduction steps in GB computations.

Improve Gaussian Elimination

Use **Linear Algebra** for reduction steps in GB computations.

$$\begin{matrix} 1 & 3 & 0 & 0 & 7 & 1 & 0 \\ 1 & 0 & 4 & 1 & 0 & 0 & 5 \\ 0 & 1 & 6 & 0 & 8 & 0 & 1 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 3 & 1 \end{matrix}$$

Improve Gaussian Elimination

Use **Linear Algebra** for reduction steps in GB computations.

$$\begin{matrix} 1 & 3 & 0 & 0 & 7 & 1 & 0 \\ 1 & 0 & 4 & 1 & 0 & 0 & 5 \\ 0 & 1 & 6 & 0 & 8 & 0 & 1 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 3 & 1 \end{matrix}$$

Knowledge of underlying GB structure

Improve Gaussian Elimination

Use **Linear Algebra** for reduction steps in GB computations.

Knowledge of underlying GB structure

Improve Gaussian Elimination

Use **Linear Algebra** for reduction steps in GB computations.

Knowledge of underlying GB structure

Improve Gaussian Elimination

Use **Linear Algebra** for reduction steps in GB computations.

Knowledge of underlying GB structure

Idea

Do a static **reordering before** the Gaussian Elimination to achieve a better initial shape. **Reorder afterwards**.

Faugère-Lachartre Idea

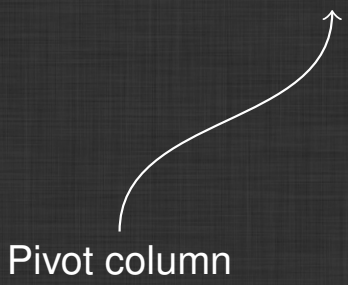
1st step: Sort pivot and non-pivot columns

$$\begin{matrix} 1 & 3 & 0 & 0 & 7 & 1 & 0 \\ 1 & 0 & 4 & 1 & 0 & 0 & 5 \\ 0 & 1 & 6 & 0 & 8 & 0 & 1 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 3 & 1 \end{matrix}$$

Faugère-Lachartre Idea

1st step: Sort pivot and non-pivot columns

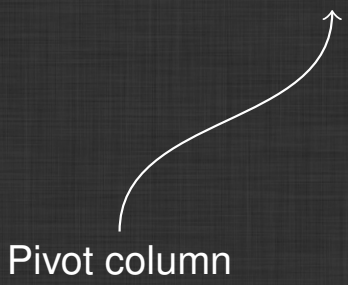
$$\begin{matrix} 1 & 3 & 0 & 0 & 7 & 1 & 0 \\ 1 & 0 & 4 & 1 & 0 & 0 & 5 \\ 0 & 1 & 6 & 0 & 8 & 0 & 1 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 3 & 1 \end{matrix}$$



Faugère-Lachartre Idea

1st step: Sort pivot and non-pivot columns

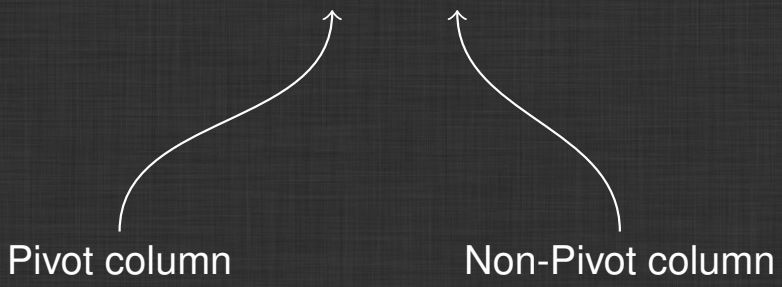
$$\begin{matrix} 1 & 3 & 0 & 0 & 7 & 1 & 0 \\ 1 & 0 & 4 & 1 & 0 & 0 & 5 \\ 0 & 1 & 6 & 0 & 8 & 0 & 1 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 3 & 1 \end{matrix}$$



Faugère-Lachartre Idea

1st step: Sort pivot and non-pivot columns

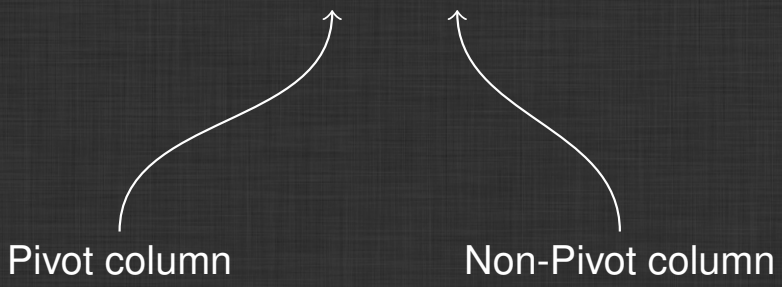
$$\begin{matrix} 1 & 3 & 0 & 0 & 7 & 1 & 0 \\ 1 & 0 & 4 & 1 & 0 & 0 & 5 \\ 0 & 1 & 6 & 0 & 8 & 0 & 1 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 3 & 1 \end{matrix}$$



Faugère-Lachartre Idea

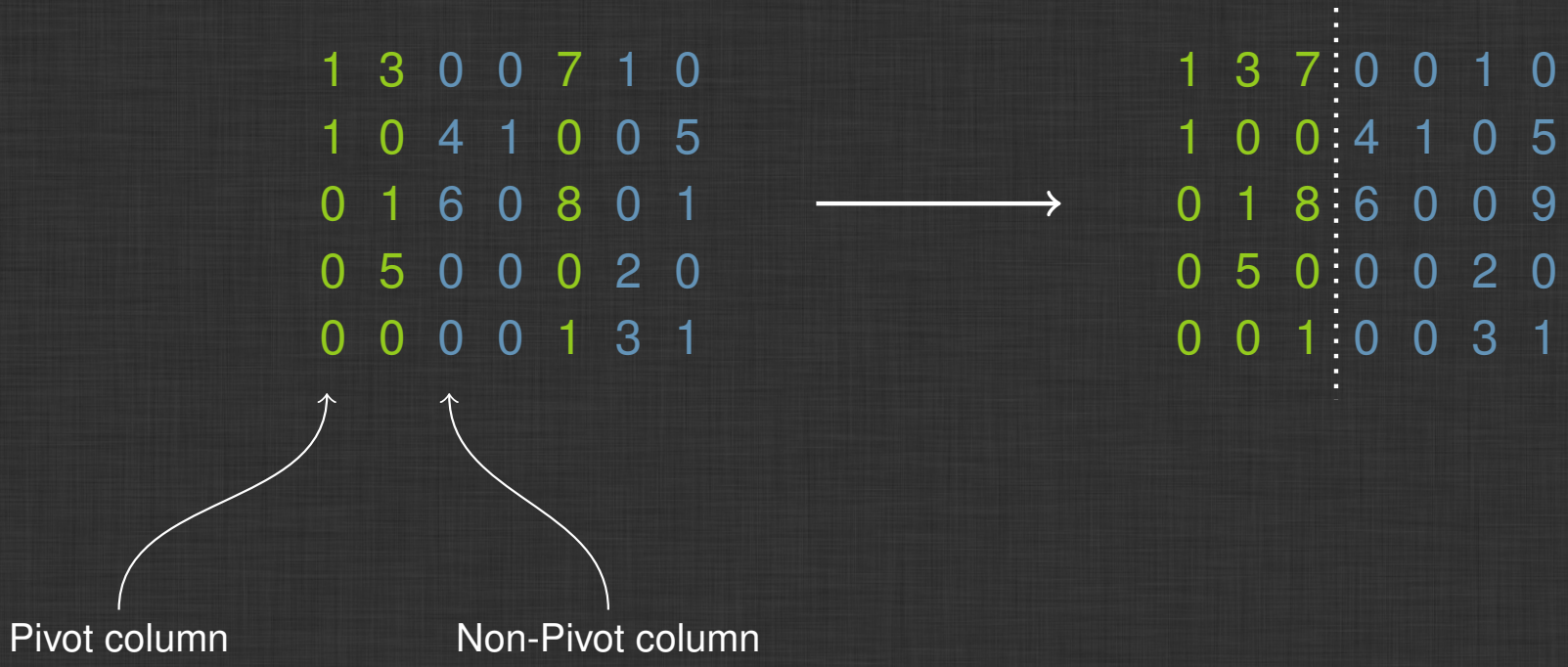
1st step: Sort pivot and non-pivot columns

$$\begin{matrix} 1 & 3 & 0 & 0 & 7 & 1 & 0 \\ 1 & 0 & 4 & 1 & 0 & 0 & 5 \\ 0 & 1 & 6 & 0 & 8 & 0 & 1 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 3 & 1 \end{matrix}$$



Faugère-Lachartre Idea

1st step: Sort pivot and non-pivot columns



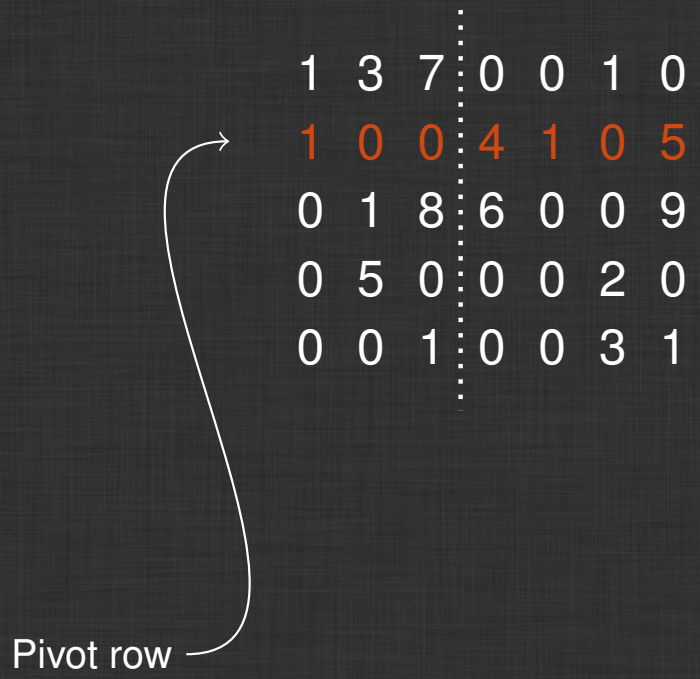
Faugère-Lachartre Idea

2nd step: Sort pivot and non-pivot rows

$$\left(\begin{array}{ccc:cccc} 1 & 3 & 7 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 4 & 1 & 0 & 5 \\ 0 & 1 & 8 & 6 & 0 & 0 & 9 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 1 & 0 & 0 & 3 & 1 \end{array}\right)$$

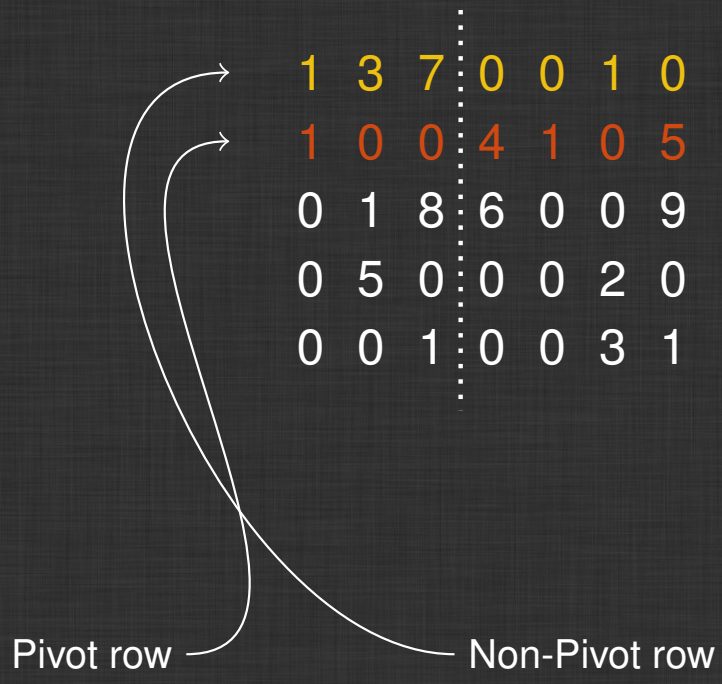
Faugère-Lachartre Idea

2nd step: Sort pivot and non-pivot rows



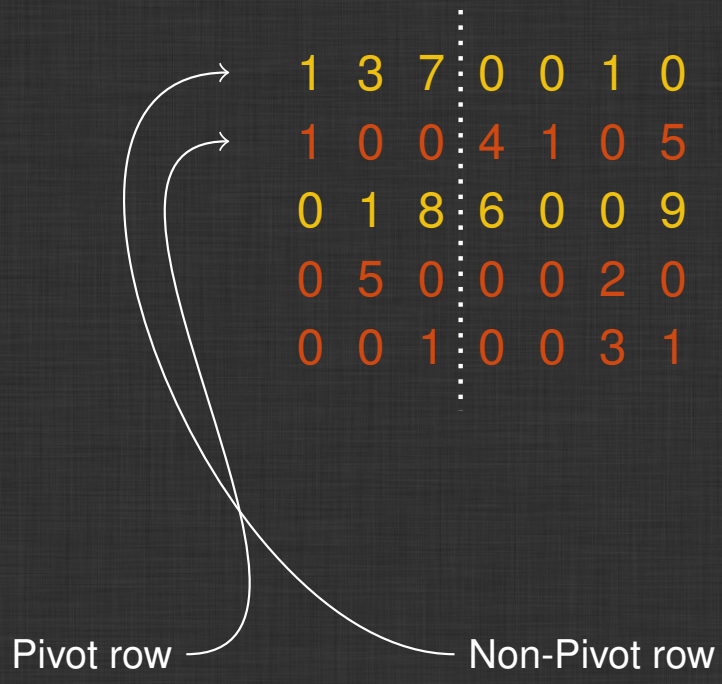
Faugère-Lachartre Idea

2nd step: Sort pivot and non-pivot rows



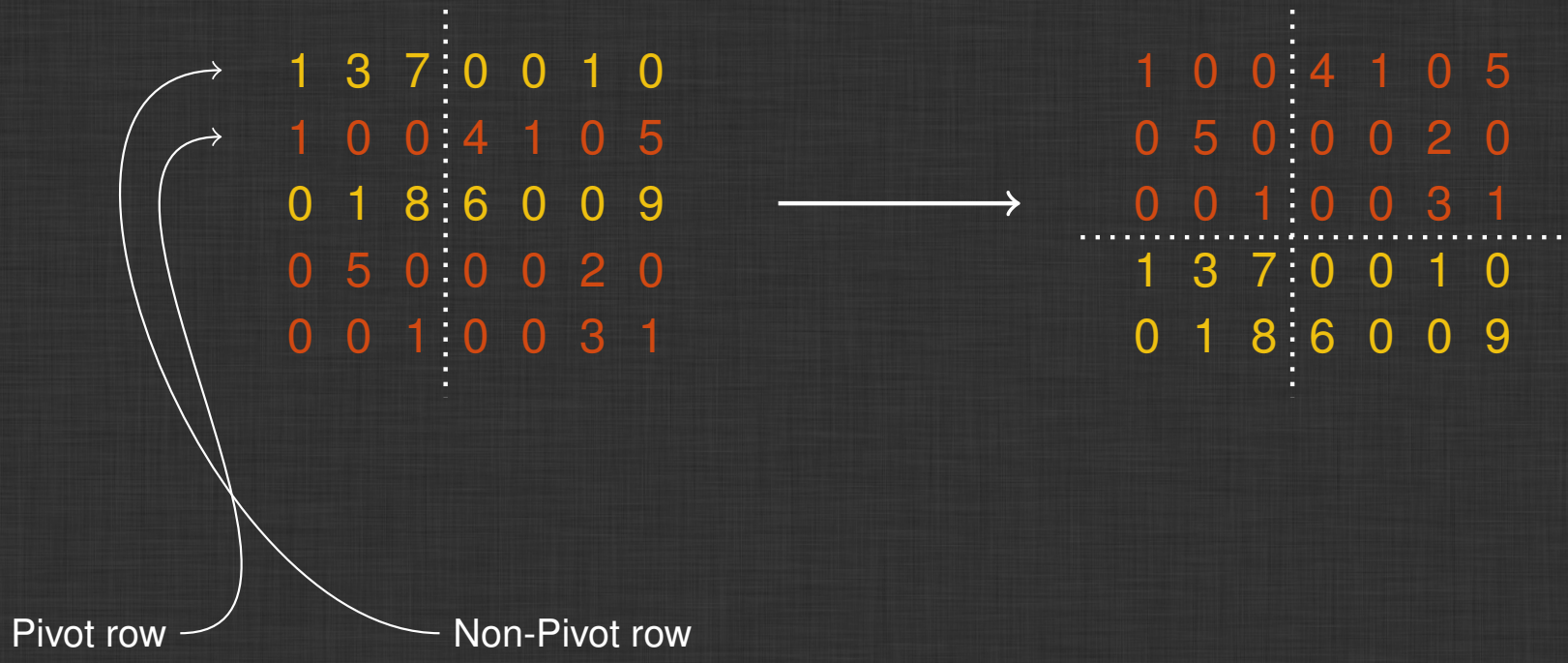
Faugère-Lachartre Idea

2nd step: Sort pivot and non-pivot rows



Faugère-Lachartre Idea

2nd step: Sort pivot and non-pivot rows



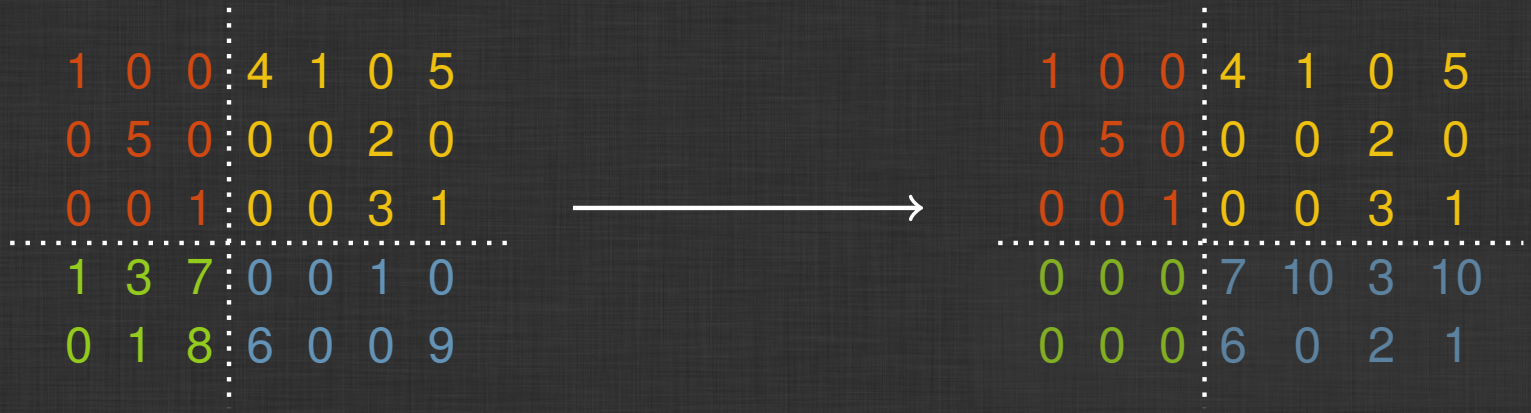
Faugère-Lachartre Idea

3rd step: Reduce lower left part to zero

$$\left(\begin{array}{ccc:cccc} 1 & 0 & 0 & 4 & 1 & 0 & 5 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 1 & 0 & 0 & 3 & 1 \\ \hdashline 1 & 3 & 7 & 0 & 0 & 1 & 0 \\ 0 & 1 & 8 & 6 & 0 & 0 & 9 \end{array}\right)$$

Faugère-Lachartre Idea

3rd step: Reduce lower left part to zero



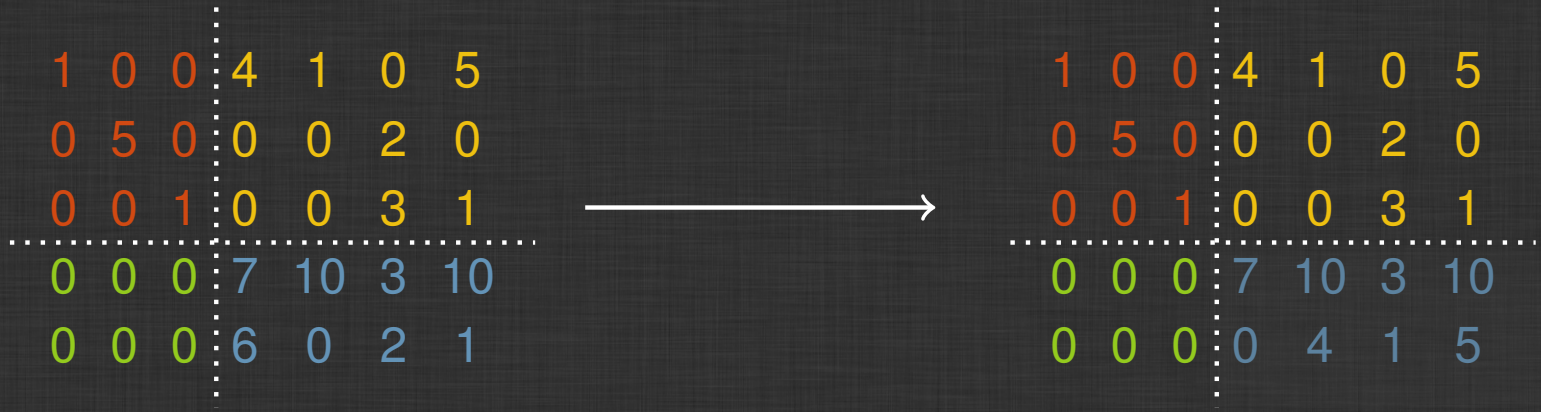
Faugère-Lachartre Idea

4th step: Reduce lower right part

$$\left(\begin{array}{ccc:cccc} 1 & 0 & 0 & 4 & 1 & 0 & 5 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 1 & 0 & 0 & 3 & 1 \\ \hdashline 0 & 0 & 0 & 7 & 10 & 3 & 10 \\ 0 & 0 & 0 & 6 & 0 & 2 & 1 \end{array}\right)$$

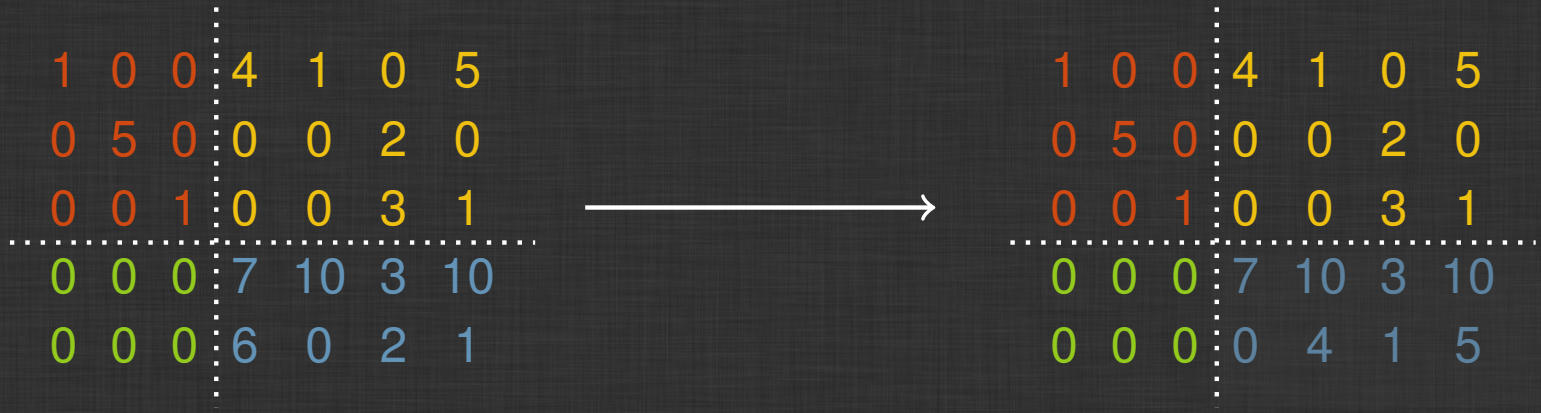
Faugère-Lachartre Idea

4th step: Reduce lower right part



Faugère-Lachartre Idea

4th step: Reduce lower right part



5th step: Remap columns of lower right part

How our matrices look like

How our matrices look like

How our matrices look like (2)

Some data about the matrix:

- F_4 computation of homogeneous KATSURA-12, degree 6 matrix

How our matrices look like (2)

Some data about the matrix:

- F_4 computation of homogeneous KATSURA-12, degree 6 matrix
- Size 137*MB*

How our matrices look like (2)

Some data about the matrix:

- F_4 computation of homogeneous KATSURA-12, degree 6 matrix
- Size 137*MB*
- $24,006,869$ nonzero elements (density: 5%)

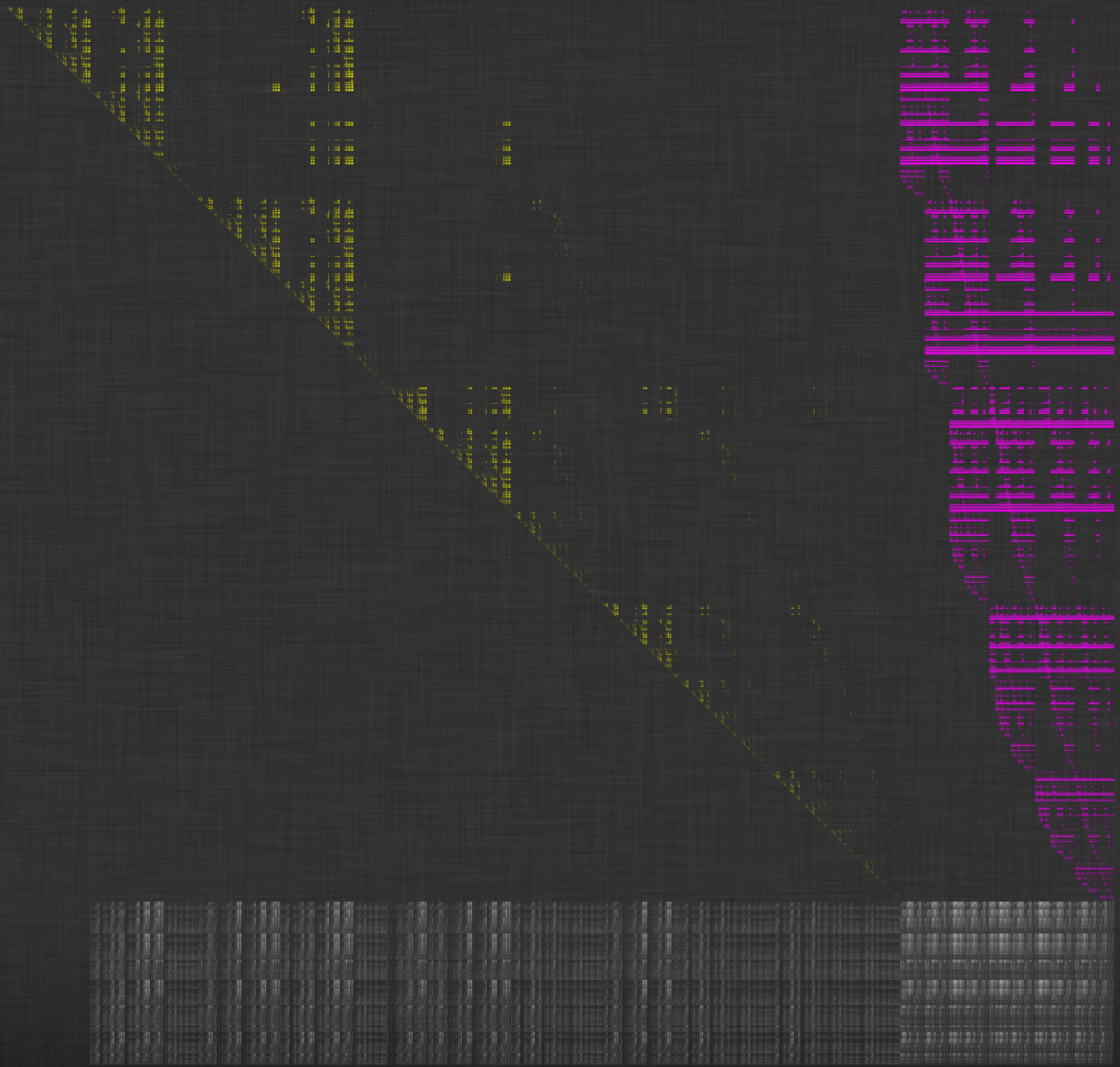
How our matrices look like (2)

Some data about the matrix:

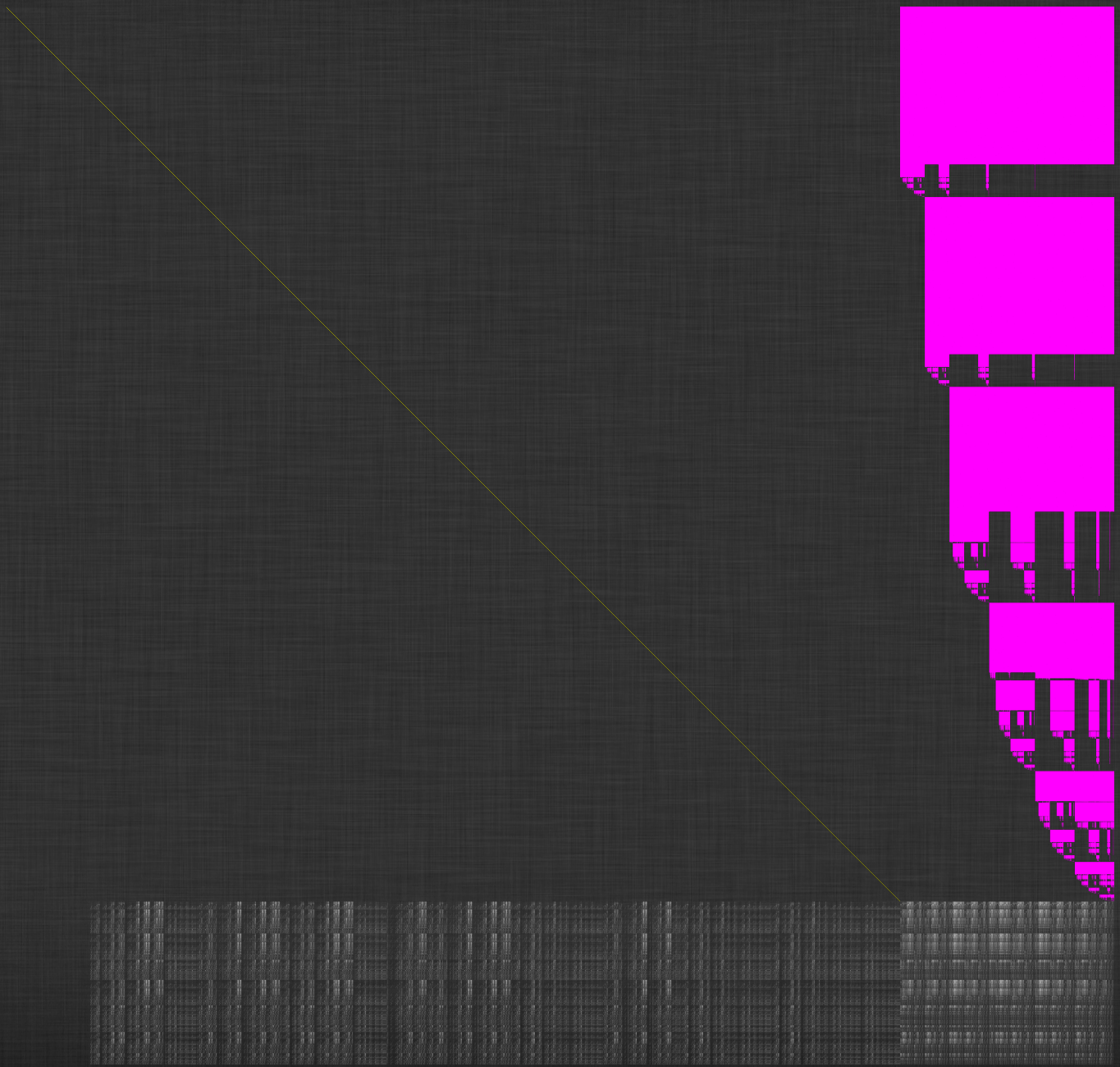
- F_4 computation of homogeneous KATSURA-12, degree 6 matrix
- Size 137*MB*
- $24,006,869$ nonzero elements (density: 5%)
- Dimensions:

full matrix:	$21,182$	$\times$	$22,207$
upper-left:	$17,915$	$\times$	$17,915$
lower-left:	$3,267$	$\times$	$17,915$
upper-right:	$17,915$	$\times$	$4,292$
lower-right:	$3,267$	$\times$	$4,292$

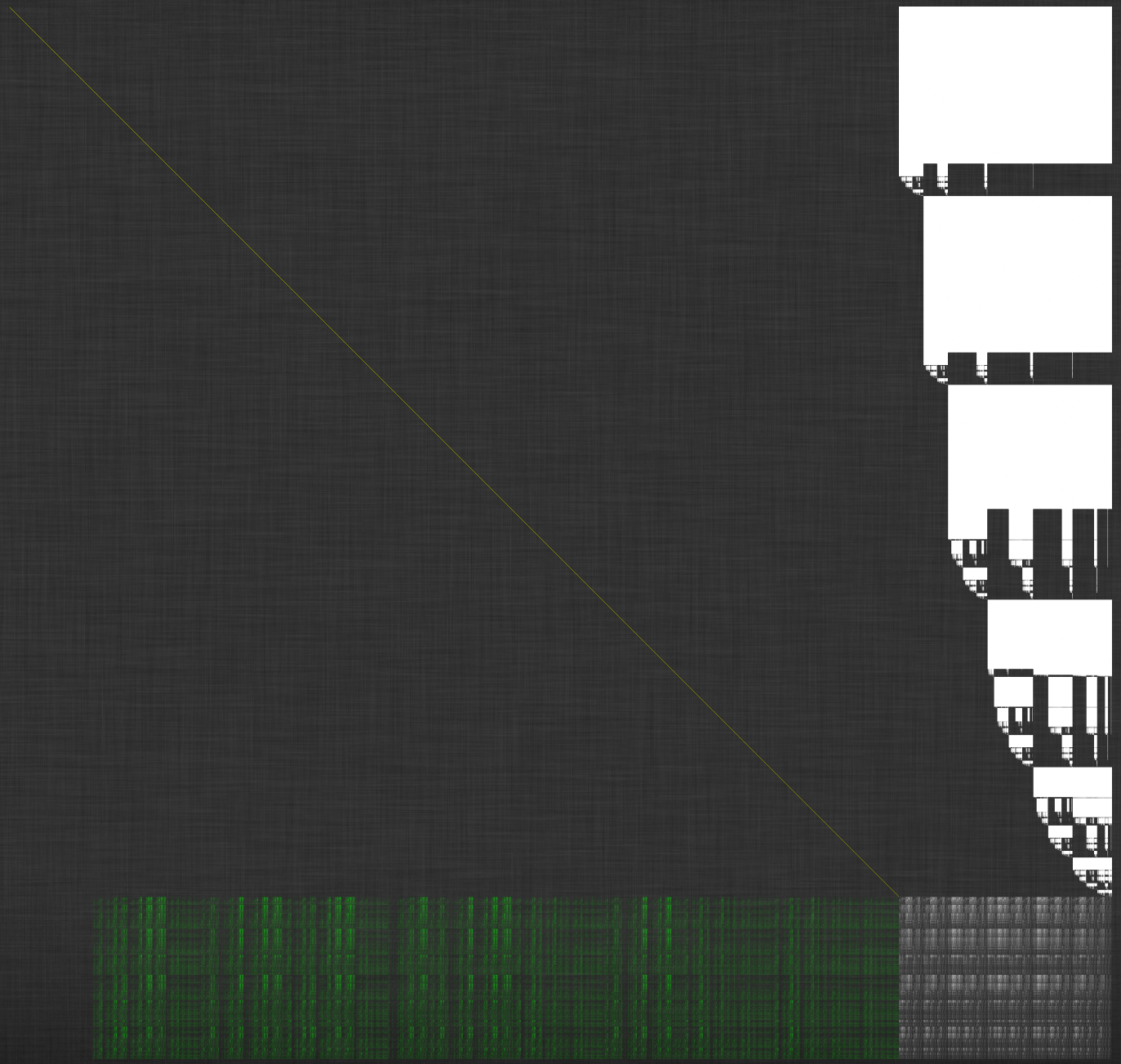
Hybrid Matrix Multiplication $A^{-1}B$



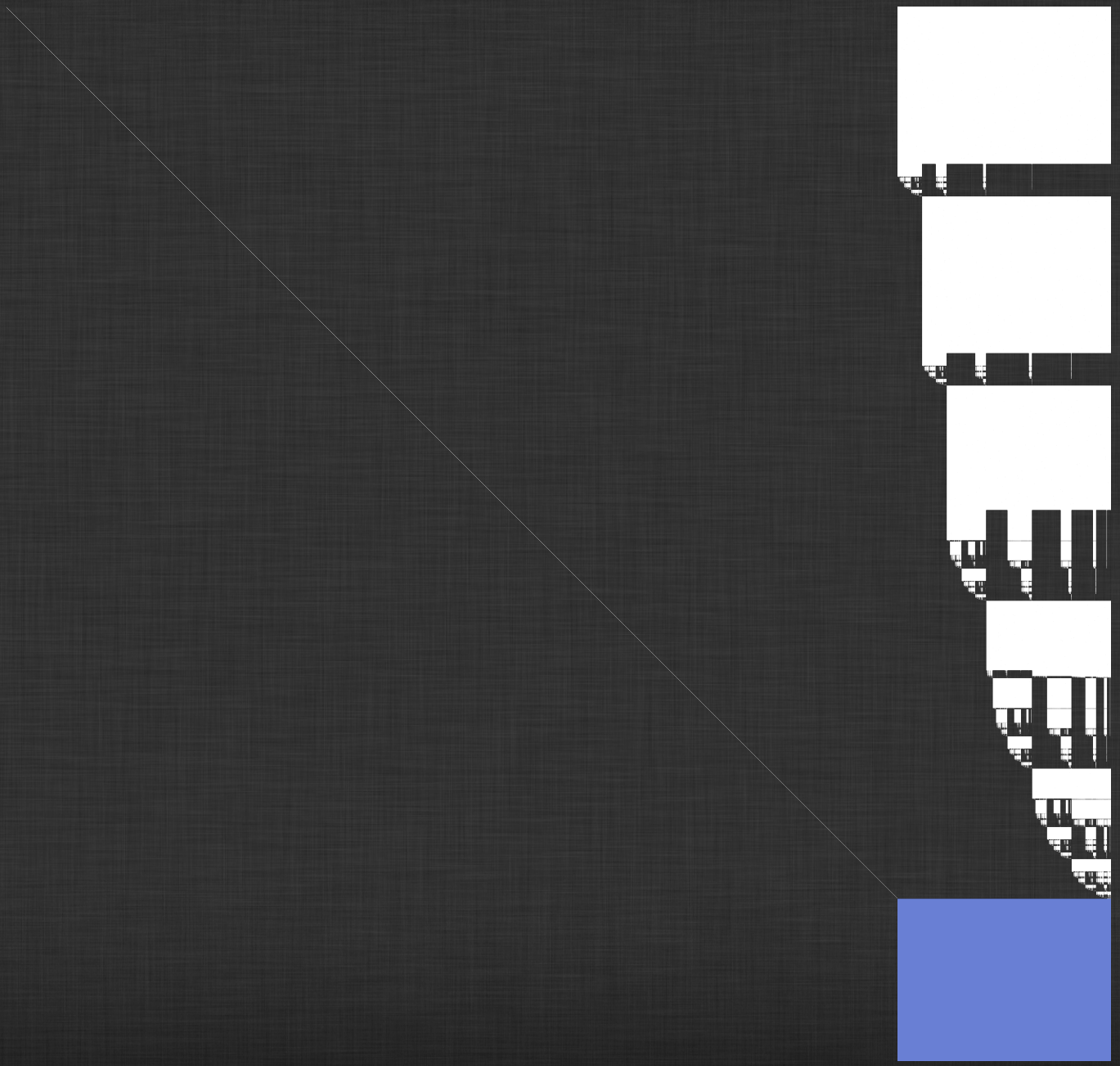
Hybrid Matrix Multiplication $A^{-1}B$



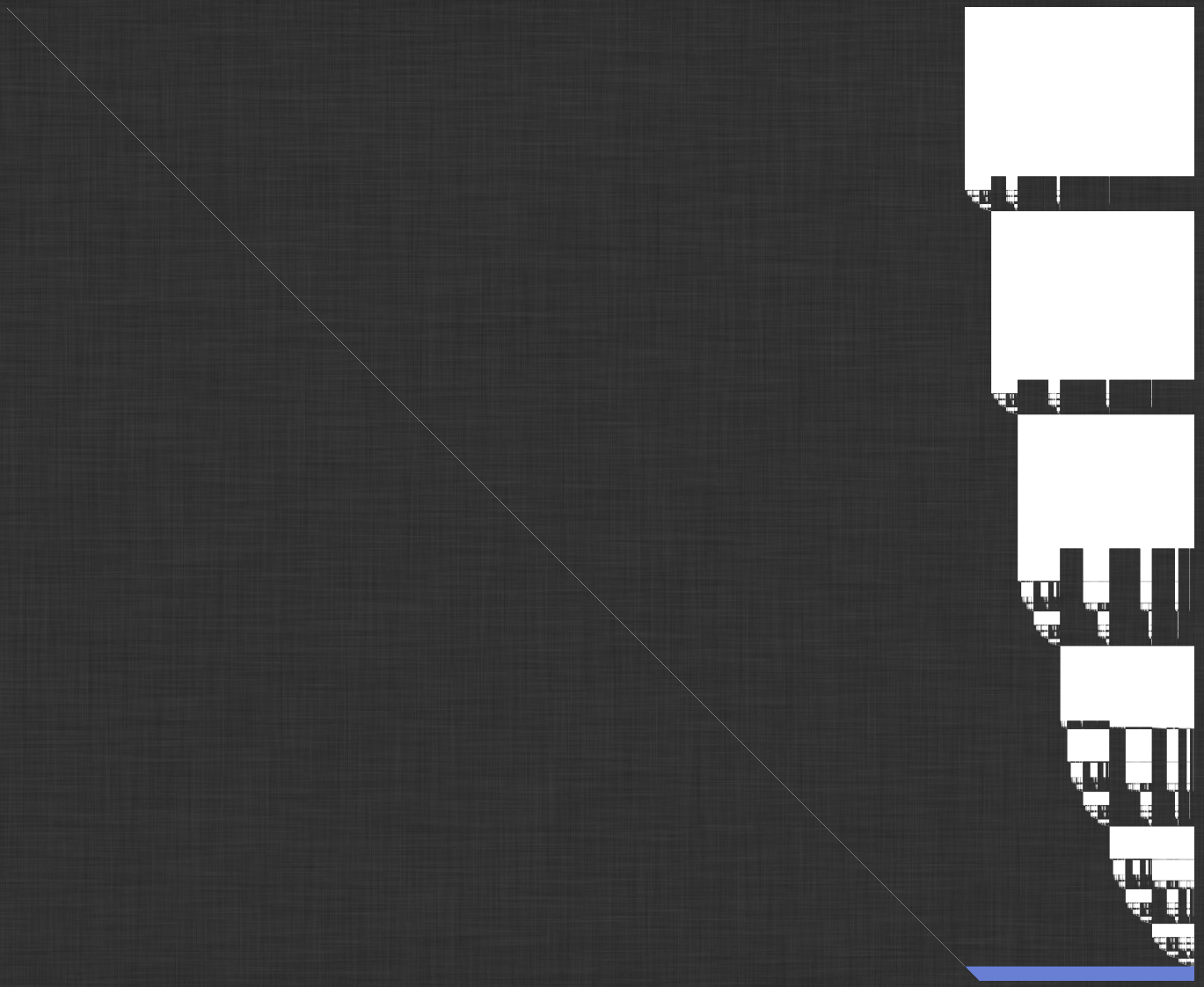
Reduce C to zero



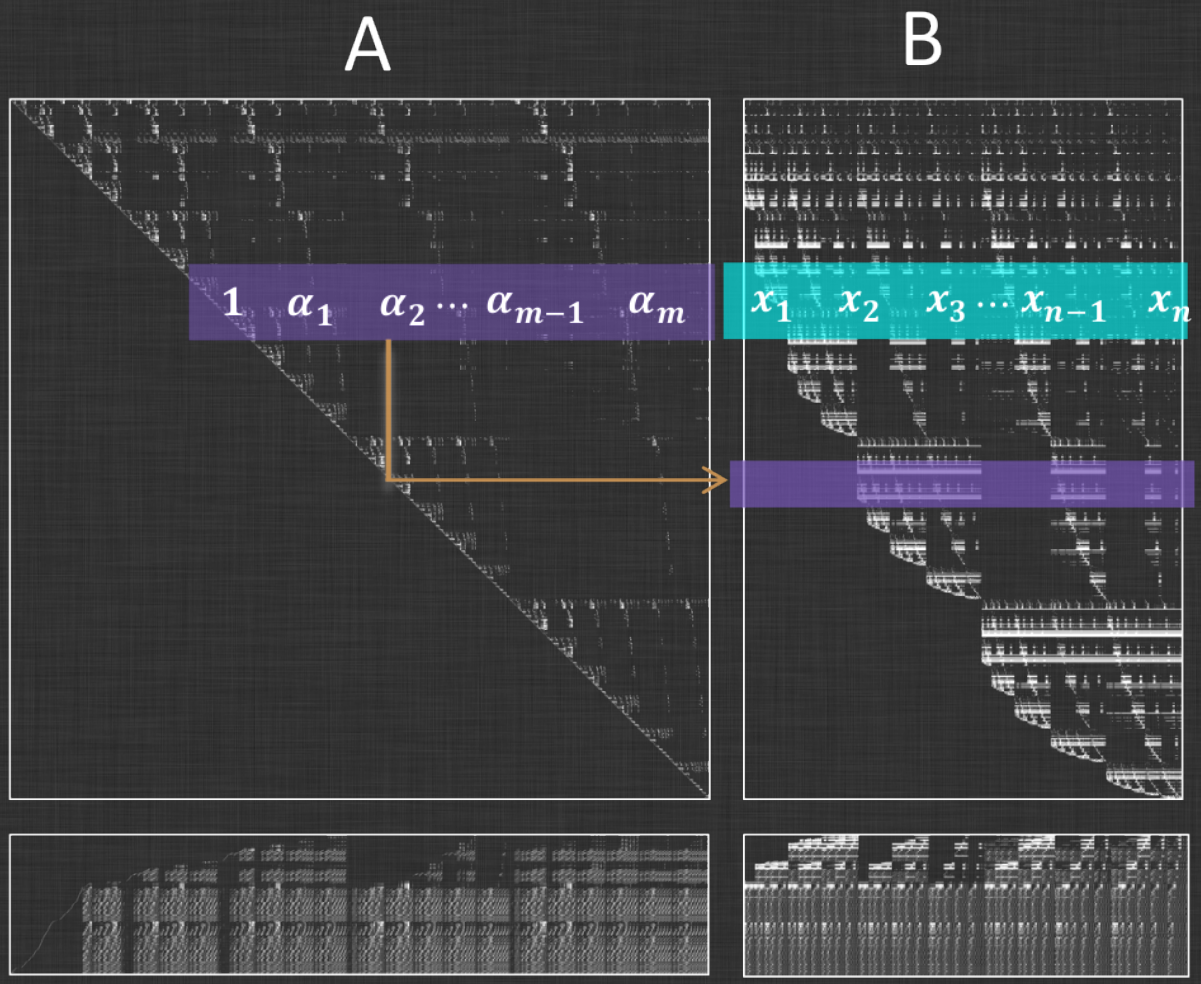
Gaussian Elimination on D



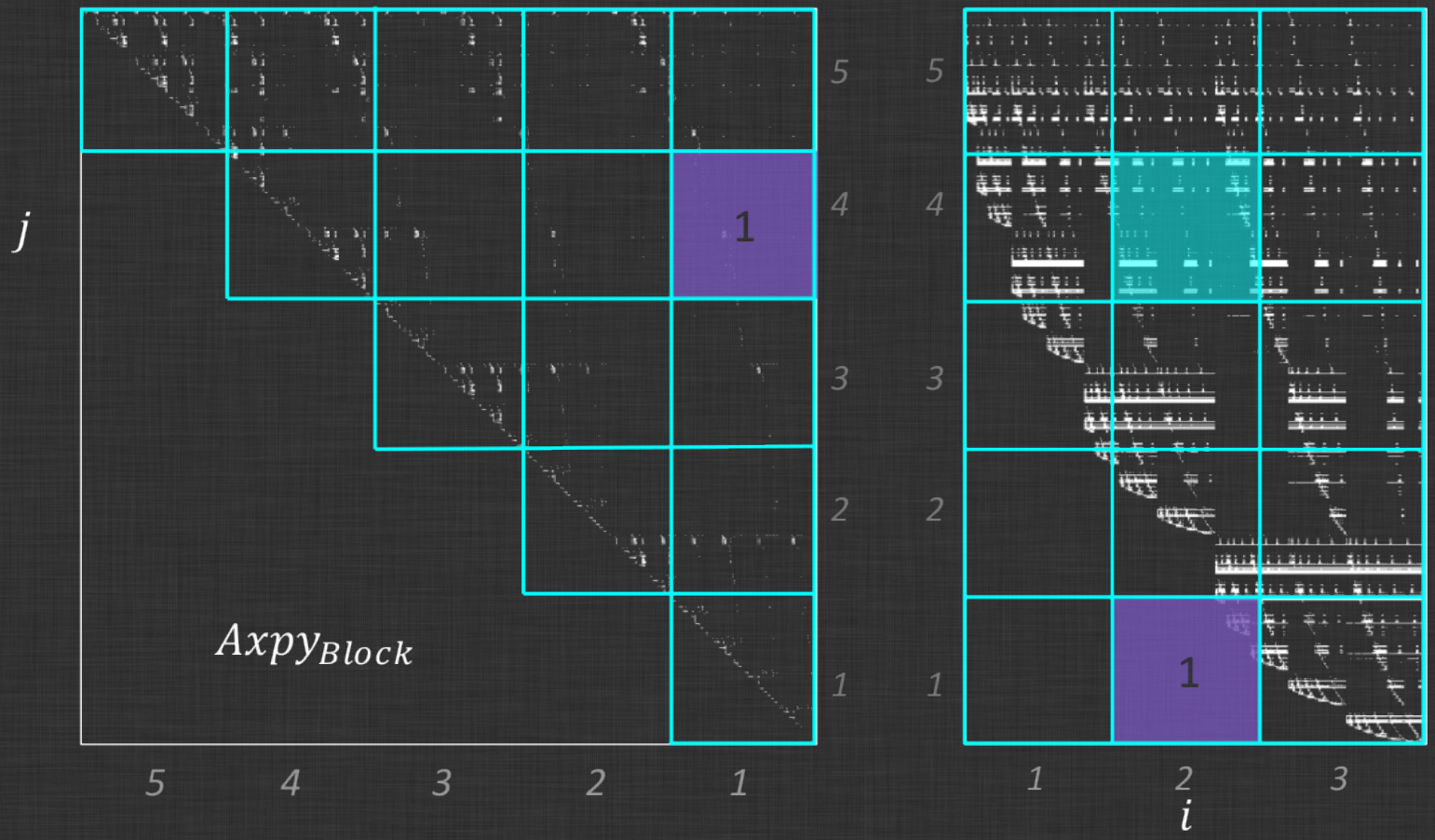
New information



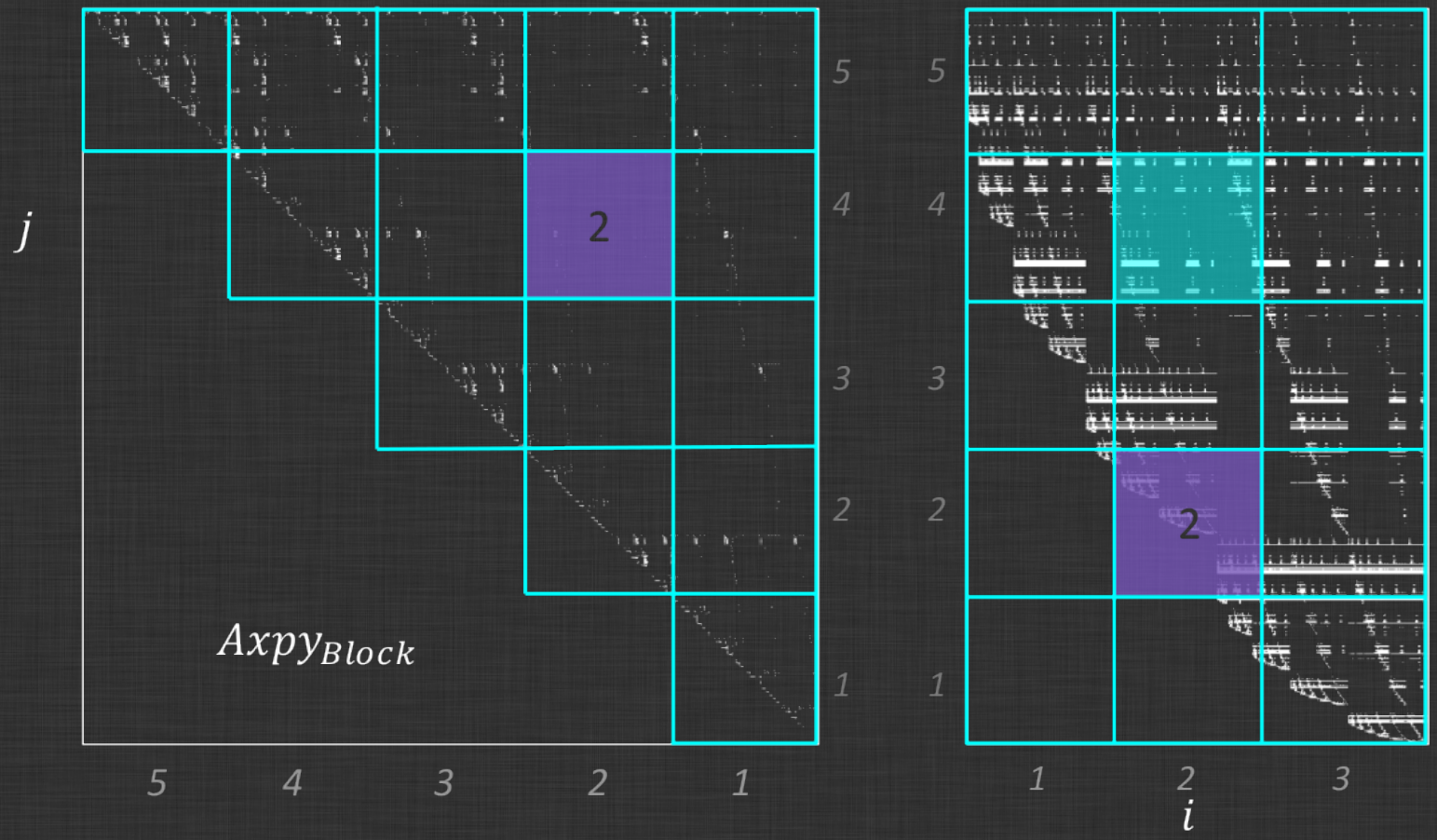
$$B \leftarrow A^{-1}B$$



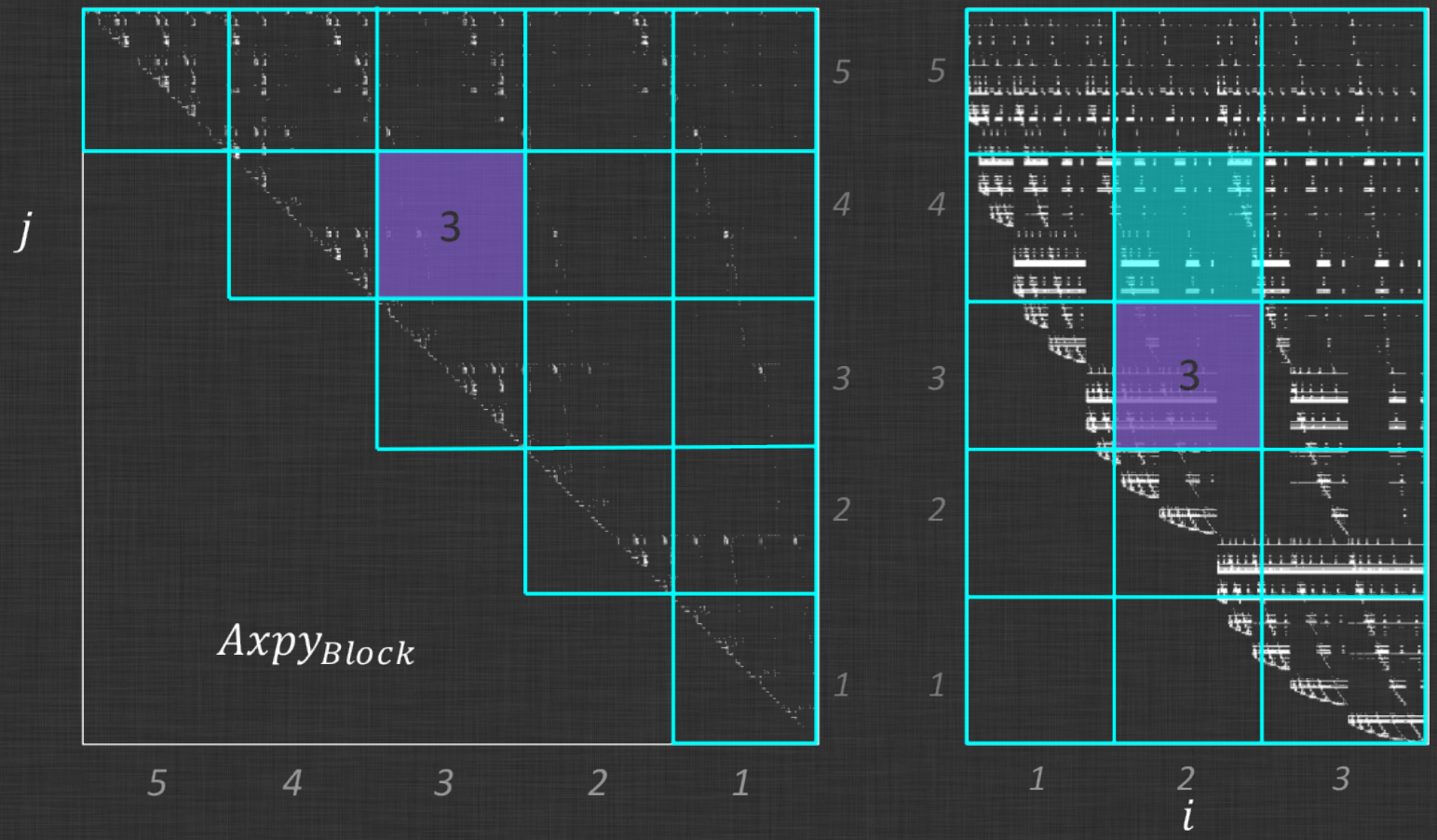
$B \leftarrow A^{-1}B$ – Block Version



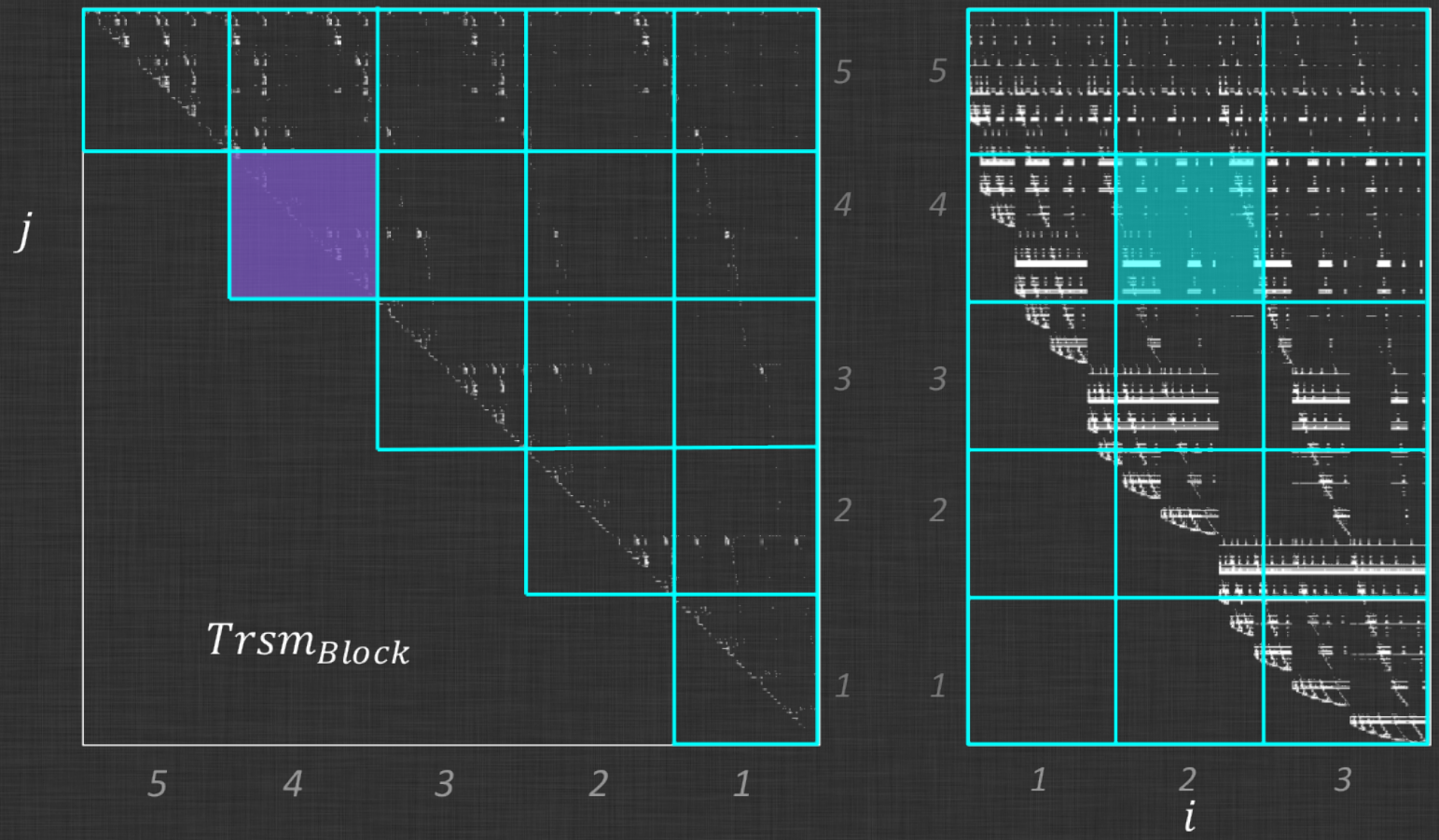
$B \leftarrow A^{-1}B$ – Block Version



$B \leftarrow A^{-1}B$ – Block Version



$B \leftarrow A^{-1}B$ – Block Version



First attempts

2011 – University of Kaiserslautern
Bradford Hovinen – LELA
`https://github.com/Singular/LELA`

2012 – UPMC Paris 6, INRIA PolSys Team
Fayssal Martani – new implementation in LELA
`https://github.com/martani/LELA`

2012-2013 – University of Kaiserslautern
Bjarke Hammersholt Roune – MathicGB
`https://github.com/broune/mathicgb`

2012-2014 – University of Passau
Severin Neumann – parallelGBC
`https://github.com/svrnm/parallelGBC`

- Predicting zero reductions
- Pair set sorting and homogenization
- Finding better reducers
- Fast linear algebra for computing Gröbner bases
- **Modular computations**
- Change of ordering

Computing over the rationals

Coefficient growth over the rationals is problematic.

In each reduction step $f - \lambda g$

1. $\mathrm{lc}(g)$ needs to be adjusted to meet $\mathrm{lc}(f)$
2. all coeffs in g have to be multiplied by $\frac{\mathrm{lc}(f)}{\mathrm{lc}(g)}$

Computing over the rationals

Coefficient growth over the rationals is problematic.

In each reduction step $f - \lambda g$

1. $\mathrm{lc}(g)$ needs to be adjusted to meet $\mathrm{lc}(f)$
2. all coeffs in g have to be multiplied by $\frac{\mathrm{lc}(f)}{\mathrm{lc}(g)}$

Once coeffs get really big this implies a tremendous slow down of the overall computations.

Computing over the rationals

Coefficient growth over the rationals is problematic.

In each reduction step $f - \lambda g$

1. $\mathrm{lc}(g)$ needs to be adjusted to meet $\mathrm{lc}(f)$
2. all coeffs in g have to be multiplied by $\frac{\mathrm{lc}(f)}{\mathrm{lc}(g)}$

Once coeffs get really big this implies a tremendous slow down of the overall computations.

1 GB computation over $\mathbb{Q}$ $\longleftrightarrow$ several GB computations over $\mathbb{F}_{p_i}$
[2, 3, 38]

(Simplified) Modular GB computations

Input: Ideal $I = \langle f_1, \ldots, f_m \rangle \subset \mathbb{Q}[x_1, \ldots, x_n]$
Output: Gröbner basis G for I in $\mathbb{Q}[x_1, \ldots, x_n]$

1. Generate set Q of primes not dividing the denoms of any coeff of any f_i.

(Simplified) Modular GB computations

Input: Ideal $I = \langle f_1, \ldots, f_m \rangle \subset \mathbb{Q}[x_1, \ldots, x_n]$
Output: Gröbner basis G for I in $\mathbb{Q}[x_1, \ldots, x_n]$

1. Generate set Q of primes not dividing the denoms of any coeff of any f_i.
2. For each $p \in Q$ compute GB G_p of I_p over $\mathbb{F}_p$.

(Simplified) Modular GB computations

Input: Ideal $I = \langle f_1, \ldots, f_m \rangle \subset \mathbb{Q}[x_1, \ldots, x_n]$
Output: Gröbner basis G for I in $\mathbb{Q}[x_1, \ldots, x_n]$

1. Generate set Q of primes not dividing the denoms of any coeff of any f_i.
2. For each $p \in Q$ compute GB G_p of I_p over $\mathbb{F}_p$.
3. Keep only $\{G_{p_1}, \ldots, G_{p_s}\}$ such that $L(G_{p_i}) = L(G_{p_j})$ and s maximal. (lucky primes, $H_G(d) \leq H_{G_{p_i}}(d)$)

(Simplified) Modular GB computations

Input: Ideal $I = \langle f_1, \ldots, f_m \rangle \subset \mathbb{Q}[x_1, \ldots, x_n]$
Output: Gröbner basis G for I in $\mathbb{Q}[x_1, \ldots, x_n]$

1. Generate set Q of primes not dividing the denoms of any coeff of any f_i.
2. For each $p \in Q$ compute GB G_p of I_p over $\mathbb{F}_p$.
3. Keep only $\{G_{p_1}, \ldots, G_{p_s}\}$ such that $L(G_{p_i}) = L(G_{p_j})$ and s maximal. (lucky primes, $H_G(d) \leq H_{G_{p_i}}(d)$)
4. Lift results back to $\mathbb{Q}[x_1, \ldots, x_n]$:
   - Chinese Remainder Theorem:

$$\prod_{i=1}^{s} \mathbb{Z}_{p_i}[x_1, \ldots, x_n] \longrightarrow \mathbb{Z}_N[x_1, \ldots, x_n]$$

$$G_{p_1} \times \cdots \times G_{p_s} \longmapsto G_N$$

   - Use Farey reconstruction to map G_N to G over $\mathbb{Q}$.

(Simplified) Modular GB computations

Input: Ideal $I = \langle f_1, \ldots, f_m \rangle \subset \mathbb{Q}[x_1, \ldots, x_n]$
Output: Gröbner basis G for I in $\mathbb{Q}[x_1, \ldots, x_n]$

1. Generate set Q of primes not dividing the denoms of any coeff of any f_i.
2. For each $p \in Q$ compute GB G_p of I_p over $\mathbb{F}_p$.
3. Keep only $\{G_{p_1}, \ldots, G_{p_s}\}$ such that $L(G_{p_i}) = L(G_{p_j})$ and s maximal. (lucky primes, $H_G(d) \leq H_{G_{p_i}}(d)$)
4. Lift results back to $\mathbb{Q}[x_1, \ldots, x_n]$:
   - ▷ Chinese Remainder Theorem:

$$\prod_{i=1}^{s} \mathbb{Z}_{p_i}[x_1, \ldots, x_n] \longrightarrow \mathbb{Z}_N[x_1, \ldots, x_n]$$

$$G_{p_1} \times \cdots \times G_{p_s} \longmapsto G_N$$

   - ▷ Use Farey reconstruction to map G_N to G over $\mathbb{Q}$.
5. Test G:
   - ▷ Make a fast test if G mod q is a GB for I_q, $q \notin Q$.
   - ▷ Check if $I \subset \langle G \rangle$ (over $\mathbb{Q}$).
   - ▷ Check if G is a GB for $\langle G \rangle$ (over $\mathbb{Q}$).

   If one test fails, go back to (1) and generate more primes $\notin Q$.

- Predicting zero reductions
- Pair set sorting and homogenization
- Finding better reducers
- Fast linear algebra for computing Gröbner bases
- Modular computations
- **Change of ordering**

Initial problem

One often needs a GB w.r.t. LEX for finding solutions (doubly exponential)

Initial problem

One often needs a GB w.r.t. LEX for finding solutions (doubly exponential)
We want to compute a GB w.r.t. DRL (exponential)

Initial problem

One often needs a GB w.r.t. LEX for finding solutions (doubly exponential)
We want to compute a GB w.r.t. DRL (exponential)

There are several different attempts to handle this problem:

- Using Hilbert functions, statically and dynamically
- Converting a GB G_1 w.r.t. $<_1$ to a GB G_2 w.r.t. $<_2$

1. Hilbert functions, statically

Let us assume homogeneous input $I = \langle f_1, \ldots, f_m \rangle$.

Let us assume homogeneous input $I = \langle f_1, \ldots, f_m \rangle$.
Let $G \subset I$. We know

$$H_{\mathscr{R}/I}(d) \leq H_{\mathscr{R}/L(G)}(d) \text{ for all } d,$$

$$H_{\mathscr{R}/I}(d) = H_{\mathscr{R}/L(G)}(d) \text{ for all } d \text{ if } G \text{ is a GB for } I.$$

Let us assume homogeneous input $I = \langle f_1, \ldots, f_m \rangle$.
Let $G \subset I$. We know

$$H_{\mathscr{R}/I}(d) \leq H_{\mathscr{R}/L(G)}(d) \text{ for all } d,$$
$$H_{\mathscr{R}/I}(d) = H_{\mathscr{R}/L(G)}(d) \text{ for all } d \text{ if } G \text{ is a GB for } I.$$

Even more: For two monomial orders $<_1$ and $<_2$ it holds that

$$H_{\mathscr{R}/L_{<_1}(G)}(d) = H_{\mathscr{R}/L_{<_2}(G)}(d) \text{ for all } d.$$

Let us assume homogeneous input $I = \langle f_1, \ldots, f_m \rangle$.
Let $G \subset I$. We know

$$H_{\mathscr{R}/I}(d) \leq H_{\mathscr{R}/L(G)}(d) \text{ for all } d,$$
$$H_{\mathscr{R}/I}(d) = H_{\mathscr{R}/L(G)}(d) \text{ for all } d \text{ if } G \text{ is a GB for } I.$$

Even more: For two monomial orders $<_1$ and $<_2$ it holds that

$$H_{\mathscr{R}/L_{<_1}(G)}(d) = H_{\mathscr{R}/L_{<_2}(G)}(d) \text{ for all } d.$$

Sloppy
$H_{\mathscr{R}/I}(d)$ measures how many elements of degree d a GB G for I has.
This number does not depend on the monomial order chosen.

We use this fact in the following way:

1. Get the Hilbert function $H(t) := H_{\mathscr{R}/I}(t)$
   - ▷ Either we have it
   - ▷ or we compute a GB G_1 for I w.r.t. an easy order $<_1$ (e.g. DRL).

We use this fact in the following way:

1. Get the Hilbert function $H(t) := H_{\mathscr{R}/I}(t)$
   - ▷ Either we have it
   - ▷ or we compute a GB G_1 for I w.r.t. an easy order $<_1$ (e.g. DRL).
2. Start usual GB computation for I w.r.t. an hard order $<_2$ (e.g. LEX).
   - ▷ Use normal selection strategy (increasing degree)
   - ▷ Whenever a new element f of $\deg(f) = d$ is added to G_2 we check if $H(t) = H_{\mathscr{R}/L(G_2)}(t)$ for all $t \geq d$.

We use this fact in the following way:

1. Get the Hilbert function $H(t) := H_{\mathscr{R}/I}(t)$
   - ▷ Either we have it
   - ▷ or we compute a GB G_1 for I w.r.t. an easy order $<_1$ (e.g. DRL).
2. Start usual GB computation for I w.r.t. an hard order $<_2$ (e.g. LEX).
   - ▷ Use normal selection strategy (increasing degree)
   - ▷ Whenever a new element f of $\deg(f) = d$ is added to G_2 we check if $H(t) = H_{\mathscr{R}/L(G_2)}(t)$ for all $t \geq d$.
     - ▷ If equal $\Longrightarrow$ Return G_2.
     - ▷ Else there exists $d' \geq d$ such that $H(d') < H_{\mathscr{R}/L(G_2)}(d')$: Remove all S-polynomials of degree $< d'$ from P and go on.

We use this fact in the following way:

1. Get the Hilbert function $H(t) := H_{\mathscr{R}/I}(t)$
   - ▷ Either we have it
   - ▷ or we compute a GB G_1 for I w.r.t. an easy order $<_1$ (e.g. DRL).
2. Start usual GB computation for I w.r.t. an hard order $<_2$ (e.g. LEX).
   - ▷ Use normal selection strategy (increasing degree)
   - ▷ Whenever a new element f of $\deg(f) = d$ is added to G_2 we check if $H(t) = H_{\mathscr{R}/L(G_2)}(t)$ for all $t \geq d$.
     - ▷ If equal $\Longrightarrow$ Return G_2.
     - ▷ Else there exists $d' \geq d$ such that $H(d') < H_{\mathscr{R}/L(G_2)}(d')$: Remove all S-polynomials of degree $< d'$ from P and go on.

Use information from G_1 during the computation of G_2.
Goes back to Traverso [47].

Implementation: **stdhilb** in SINGULAR

2. Hilbert functions, dynamically

Receive a GB for I w.r.t. some order $<$ ($<$ not known beforehand):

1. Get the Hilbert function $H(t) := H_{\mathscr{R}/I}(t)$
   - ▷ Either we have it
   - ▷ or we compute a GB G_1 for I w.r.t. an easy order $<_1$ (e.g. DRL).

2. Hilbert functions, dynamically

Receive a GB for I w.r.t. some order $<$ ($<$ not known beforehand):

1. Get the Hilbert function $H(t) := H_{\mathscr{R}/I}(t)$
   - ▷ Either we have it
   - ▷ or we compute a GB G_1 for I w.r.t. an easy order $<_1$ (e.g. DRL).
2. Start usual GB computation for I w.r.t. some initial order $<_{\text{initial}}$.
   - ▷ Use normal selection strategy (increasing degree)
   - ▷ Whenever a new element f of $\deg(f) = d$ is added to G_2 we check if $H(t) = H_{\mathscr{R}/L(G_2)}(t)$ for all $t \geq d$.

2. Hilbert functions, dynamically

Receive a GB for I w.r.t. some order $<$ ($<$ not known beforehand):

1. Get the Hilbert function $H(t) := H_{\mathscr{R}/I}(t)$
   - ▷ Either we have it
   - ▷ or we compute a GB G_1 for I w.r.t. an easy order $<_1$ (e.g. DRL).
2. Start usual GB computation for I w.r.t. some initial order $<_{\text{initial}}$.
   - ▷ Use normal selection strategy (increasing degree)
   - ▷ Whenever a new element f of $\deg(f) = d$ is added to G_2 we check if $H(t) = H_{\mathscr{R}/L(G_2)}(t)$ for all $t \geq d$.
     - ▷ If equal $\Longrightarrow$ Return G_2.
     - ▷ Else there exists $d' \geq d$ such that $H(d') < H_{\mathscr{R}/L(G_2)}(d')$: Remove all S-polynomials of degree $< d'$ from P.

2. Hilbert functions, dynamically

Receive a GB for I w.r.t. some order $<$ ($<$ not known beforehand):

1. Get the Hilbert function $H(t) := H_{\mathscr{R}/I}(t)$
   - ▷ Either we have it
   - ▷ or we compute a GB G_1 for I w.r.t. an easy order $<_1$ (e.g. DRL).
2. Start usual GB computation for I w.r.t. some initial order $<_{\text{initial}}$.
   - ▷ Use normal selection strategy (increasing degree)
   - ▷ Whenever a new element f of $\deg(f) = d$ is added to G_2 we check if $H(t) = H_{\mathscr{R}/L(G_2)}(t)$ for all $t \geq d$.
     - ▷ If equal $\Longrightarrow$ Return G_2.
     - ▷ Else there exists $d' \geq d$ such that $H(d') < H_{\mathscr{R}/L(G_2)}(d')$: Remove all S-polynomials of degree $< d'$ from P.
     - ▷ Find a better order $<_{\text{new}}$ such that $H_{\mathscr{R}/L_{<\text{new}}(G_2)}(t)$ is minimal.
     - ▷ Re-reduce G w.r.t. $<_{\text{new}}$ and go on.

2. Hilbert functions, dynamically

Receive a GB for I w.r.t. some order $<$ ($<$ not known beforehand):

1. Get the Hilbert function $H(t) := H_{\mathscr{R}/I}(t)$
   - ▷ Either we have it
   - ▷ or we compute a GB G_1 for I w.r.t. an easy order $<_1$ (e.g. DRL).
2. Start usual GB computation for I w.r.t. some initial order $<_{\text{initial}}$.
   - ▷ Use normal selection strategy (increasing degree)
   - ▷ Whenever a new element f of $\deg(f) = d$ is added to G_2 we check if $H(t) = H_{\mathscr{R}/L(G_2)}(t)$ for all $t \geq d$.
     - ▷ If equal $\Longrightarrow$ Return G_2.
     - ▷ Else there exists $d' \geq d$ such that $H(d') < H_{\mathscr{R}/L(G_2)}(d')$: Remove all S-polynomials of degree $< d'$ from P.
     - ▷ Find a better order $<_{\text{new}}$ such that $H_{\mathscr{R}/L_{<_{\text{new}}}(G_2)}(t)$ is minimal.
     - ▷ Re-reduce G w.r.t. $<_{\text{new}}$ and go on.

There are several questions to be answered.

OK, but. . . ?

1. What is the resulting order $<$? Should be near the order one wants, transform G with methods explained in the following.

OK, but. . . ?

1. What is the resulting order $<$? Should be near the order one wants, transform G with methods explained in the following.
2. What is meant by "minimal" $H_{\mathscr{R}/L(G)}(t)$? Can be understood in the sense of
   - ▷ lexicographically minimal considered as a function, or
   - ▷ of minimal degree considered as Hilbert polynomial.

   Caboara suggests a mix of both minimalizations in [11].

OK, but. . . ?

1. What is the resulting order $<$? Should be near the order one wants, transform G with methods explained in the following.
2. What is meant by "minimal" $H_{\mathscr{R}/L(G)}(t)$? Can be understood in the sense of
   - ▷ lexicographically minimal considered as a function, or
   - ▷ of minimal degree considered as Hilbert polynomial.

   Caboara suggests a mix of both minimalizations in [11].

1. It is hard to get the order changing right.

2. Caboara and Perry are improving the idea currently [12].

3. Converting a GB

Let $v \in \mathscr{R}^n$ be a weight vector, $I = \langle f_1, \ldots, f_m \rangle \subset \mathscr{R}$ an ideal, and let G be a GB for I w.r.t. some order $<$.

3. Converting a GB

Let $v \in \mathscr{R}^n$ be a weight vector, $I = \langle f_1, \ldots, f_m \rangle \subset \mathscr{R}$ an ideal, and let G be a GB for I w.r.t. some order $<$.

- $f \in I$, initial element of f w.r.t. v: $\mathrm{in}_v(f) = t$, $\deg_v(t)$ maximal for all $t \in \mathrm{support}(f)$.
- Initial ideal of I w.r.t. v: $\mathrm{in}_v(I) = \langle \mathrm{in}_v(f_1), \ldots, \mathrm{in}_v(f_m) \rangle$.
  (Not neccessarily a monomial ideal!)

3. Converting a GB

Let $v \in \mathscr{R}^n$ be a weight vector, $I = \langle f_1, \ldots, f_m \rangle \subset \mathscr{R}$ an ideal, and let G be a GB for I w.r.t. some order $<$.

- $f \in I$, initial element of f w.r.t. v: $\mathrm{in}_v(f) = t$, $\deg_v(t)$ maximal for all $t \in \mathrm{support}(f)$.
- Initial ideal of I w.r.t. v: $\mathrm{in}_v(I) = \langle \mathrm{in}_v(f_1), \ldots, \mathrm{in}_v(f_m) \rangle$. (Not neccessarily a monomial ideal!)
- Refinement of v:

$$x^\alpha (v,<) x^\beta \quad :\Leftrightarrow \quad \deg_v(x^\alpha) <_{\mathrm{nat}} \deg_v(x^\beta) \text{ or } \deg_v(x^\alpha) = \deg_v(x^\beta) \text{ and } x^\alpha < x^\beta .$$

3. Converting a GB

Let $v \in \mathscr{R}^n$ be a weight vector, $I = \langle f_1, \ldots, f_m \rangle \subset \mathscr{R}$ an ideal, and let G be a GB for I w.r.t. some order $<$.

- $f \in I$, initial element of f w.r.t. v: $\mathrm{in}_v(f) = t$, $\deg_v(t)$ maximal for all $t \in \mathrm{support}(f)$.
- Initial ideal of I w.r.t. v: $\mathrm{in}_v(I) = \langle \mathrm{in}_v(f_1), \ldots, \mathrm{in}_v(f_m) \rangle$. (Not neccessarily a monomial ideal!)
- Refinement of v:

$$x^\alpha (v,<) x^\beta \quad :\Leftrightarrow \quad \deg_v(x^\alpha) <_{\mathrm{nat}} \deg_v(x^\beta) \text{ or } \deg_v(x^\alpha) = \deg_v(x^\beta) \text{ and } x^\alpha < x^\beta .$$

- Gröbner cone: $C_<(G) = \{v \in \mathscr{R}^n \mid L_<(\mathrm{in}_v(f))\} = \mathrm{lt}_<(f)$ for all $f \in G\}$.

3. Converting a GB

Let $v \in \mathscr{R}^n$ be a weight vector, $I = \langle f_1, \ldots, f_m \rangle \subset \mathscr{R}$ an ideal, and let G be a GB for I w.r.t. some order $<$.

- $f \in I$, initial element of f w.r.t. v: $\mathrm{in}_v(f) = t$, $\deg_v(t)$ maximal for all $t \in \mathrm{support}(f)$.
- Initial ideal of I w.r.t. v: $\mathrm{in}_v(I) = \langle \mathrm{in}_v(f_1), \ldots, \mathrm{in}_v(f_m) \rangle$. (Not neccessarily a monomial ideal!)
- Refinement of v:

$$x^\alpha (v,<) x^\beta \quad :\Leftrightarrow \quad \deg_v(x^\alpha) <_{\mathrm{nat}} \deg_v(x^\beta) \text{ or } \deg_v(x^\alpha) = \deg_v(x^\beta) \text{ and } x^\alpha < x^\beta .$$

- Gröbner cone: $C_<(G) = \{v \in \mathscr{R}^n \mid L_<(\mathrm{in}_v(f))\} = \mathrm{lt}_<(f)$ for all $f \in G\}$.
- The Gröbner fan is the fan $\triangle_G$ consisting of all $C_<(G)$ where $<$ runs over all monomial orders on $\mathscr{R}$.

3. Converting a GB

Let $v \in \mathscr{R}^n$ be a weight vector, $I = \langle f_1, \ldots, f_m \rangle \subset \mathscr{R}$ an ideal, and let G be a GB for I w.r.t. some order $<$.

- $f \in I$, initial element of f w.r.t. v: $\mathrm{in}_v(f) = t$, $\deg_v(t)$ maximal for all $t \in \mathrm{support}(f)$.
- Initial ideal of I w.r.t. v: $\mathrm{in}_v(I) = \langle \mathrm{in}_v(f_1), \ldots, \mathrm{in}_v(f_m) \rangle$. (Not neccessarily a monomial ideal!)
- Refinement of v:
$$x^\alpha (v,<) x^\beta \quad :\Leftrightarrow \quad \deg_v(x^\alpha) <_{\mathrm{nat}} \deg_v(x^\beta) \text{ or } \deg_v(x^\alpha) = \deg_v(x^\beta) \text{ and } x^\alpha < x^\beta .$$
- Gröbner cone: $C_<(G) = \{v \in \mathscr{R}^n \mid L_<(\mathrm{in}_v(f))\} = \mathrm{lt}_<(f) \text{ for all } f \in G\}$.
- The Gröbner fan is the fan $\triangle_G$ consisting of all $C_<(G)$ where $<$ runs over all monomial orders on $\mathscr{R}$.

Note

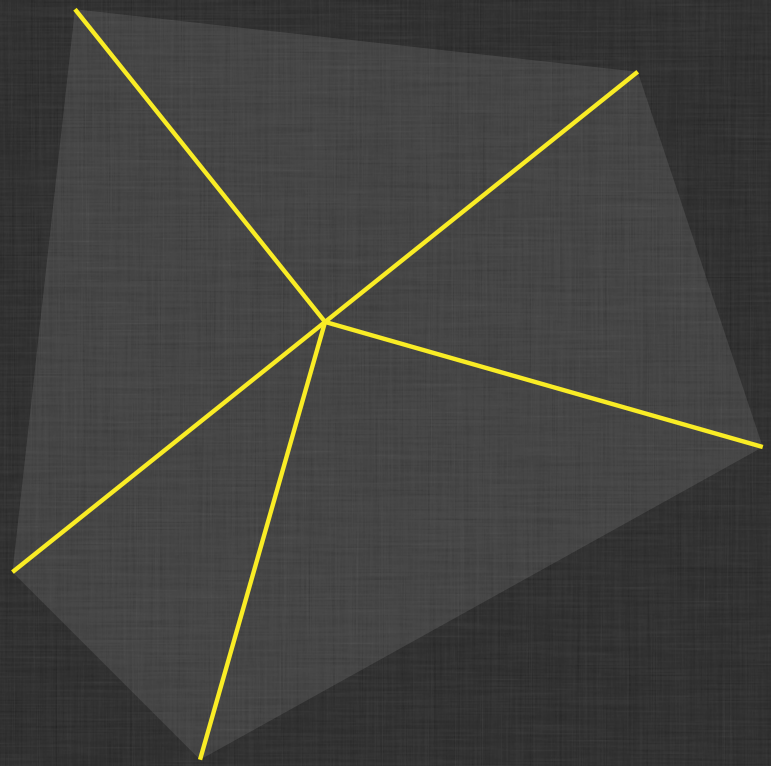
There are only finitely many not equivalent monomial orders on $\mathscr{R}$
$\Longrightarrow$ Gröbner cone is well-defined.

3. Converting a GB

How does such a fan look like?

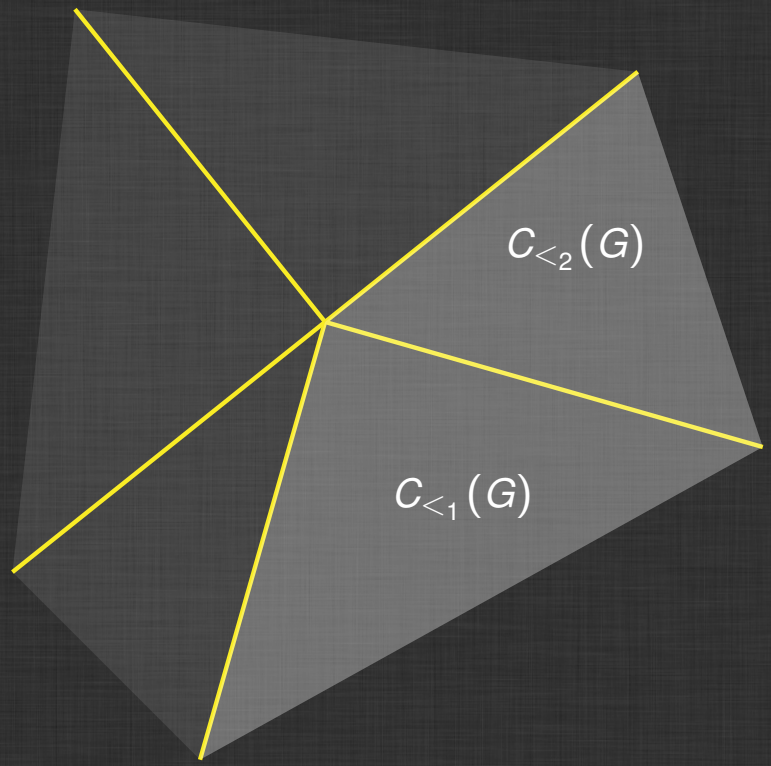
3. Converting a GB

How does such a fan look like?



3. Converting a GB

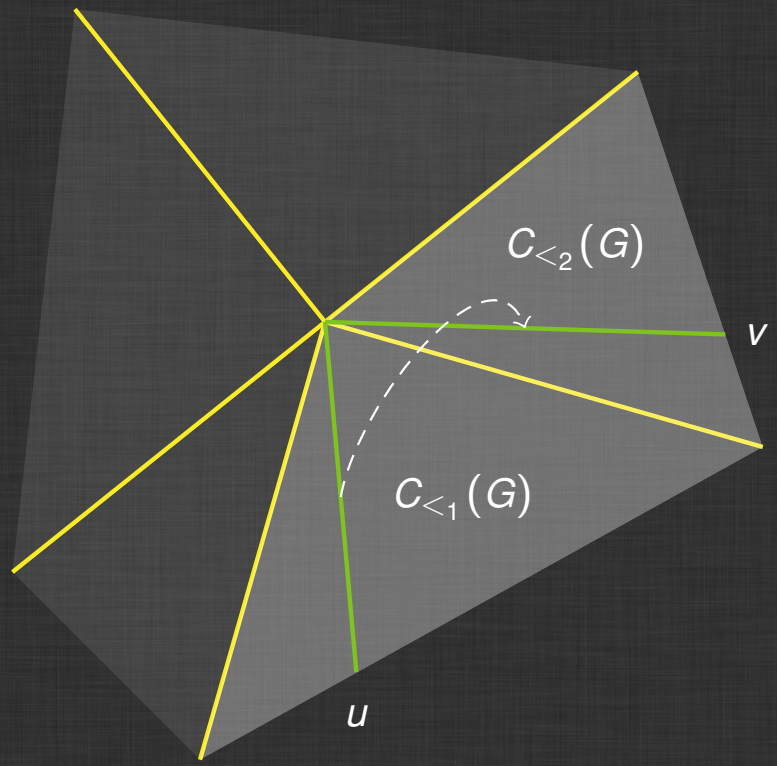
How does such a fan look like?



Main idea

Walk from one cone to an adjacent one until you reach the lovely monomial order w.r.t. which you want to get a GB.

How does such a fan look like?

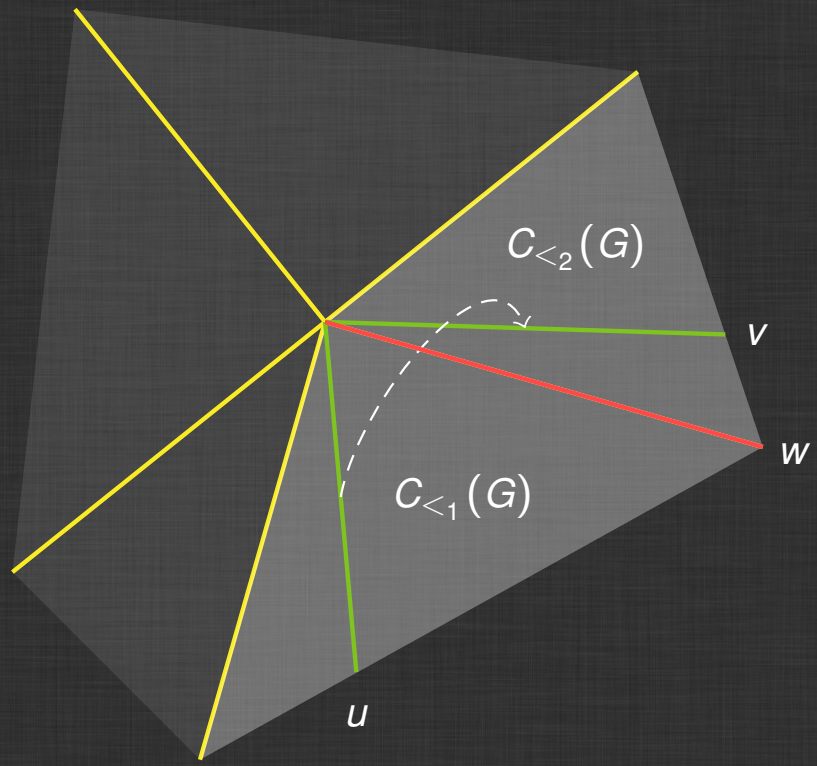


Main idea

Walk from one cone to an adjacent one until you reach the lovely monomial order w.r.t. which you want to get a GB.

3. Converting a GB

How does such a fan look like?



Main idea

Walk from one cone to an adjacent one until you reach the lovely monomial order w.r.t. which you want to get a GB.

Lemma

Let $<_1, <_2$ be two different orders such that $C_{<_1}(I) \cap C_{<_2}(I) \neq \emptyset$. Then there exists $w \in C_{<_1}(I) \cap C_{<_2}(I)$ such that $\mathrm{in}_w(I)$ is not monomial.

3. Converting a GB

Lemma

Let $<_1, <_2$ be two different orders such that $C_{<_1}(I) \cap C_{<_2}(I) \neq \emptyset$. Then there exists $w \in C_{<_1}(I) \cap C_{<_2}(I)$ such that $\text{in}_w(I)$ is not monomial.

Given a reduced $G_{<_1} = \{g_1, \ldots, g_r\}$ (and w), how to get now $G_{<_2}$?

1. Refine w by $<_2$: $(w, <_2)$
2. Compute reduced GB M of $\text{in}_w(G_{<_1})$ w.r.t. $(w, <_2)$ (Nearly monomial!) $m_j \in M$ represented by $m_j = \sum_{i=1}^{r} h_{ij} \text{in}_w(g_i)$.
3. $p_j \leftarrow$ Take m_j and replace $\text{in}_w(g_i)$ by g_i.
4. $G_{(w,<_2)} \leftarrow$ Reduce $\{p_1, \ldots, p_k\}$ w.r.t. $(w, <_2)$.
5. Convert $G_{(w,<_2)}$ to a reduced GB w.r.t. $<_2$.

Conclusions

- Often better complexity when computing GB in DRL and walk to LEX.
- Problem if conversion between not adjacent orders
  $\Longrightarrow$ Several steps from cone to cone
- Note that for more than 2 variables searching good paths is quite hard.
- Kalkbrener [40]: When converting from $G_{<_1}$ to an adjacent $G_{<_2}$ maximal degree of elements in $G_{<_2}$ bounded by

$$D(G_{<_2}) < 2D(G_{<_1})^2 + (n+1)D(G_{<_1}).$$

Way better than doubly exponential growth of degree for not adjacent transformation!

Conclusions

- Often better complexity when computing GB in DRL and walk to LEX.
- Problem if conversion between not adjacent orders
  $\Longrightarrow$ Several steps from cone to cone
- Note that for more than 2 variables searching good paths is quite hard.
- Kalkbrener [40]: When converting from $G_{<_1}$ to an adjacent $G_{<_2}$ maximal degree of elements in $G_{<_2}$ bounded by

$$D(G_{<_2}) < 2D(G_{<_1})^2 + (n+1)D(G_{<_1}).$$

Way better than doubly exponential growth of degree for not adjacent transformation!

Implementations: **GBWalk** in SINGULAR, **GFan** library [39]

3. Converting a GB - zero dimensional

If I is zero dimensional we can do even better – FGLM [20]:

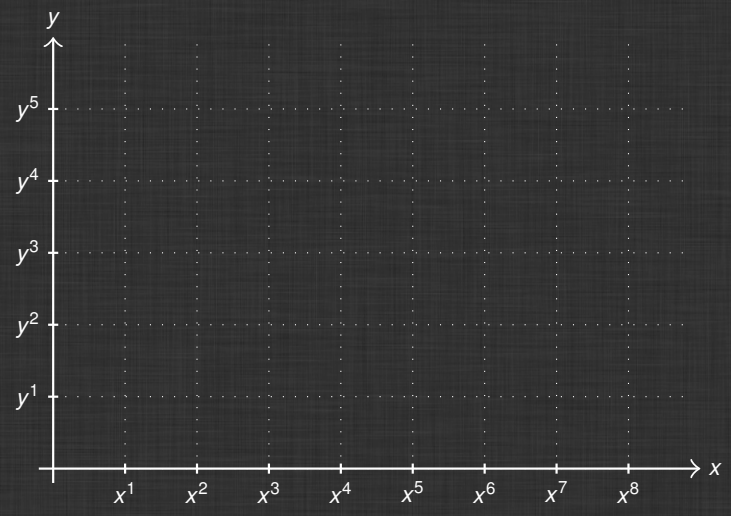
> Go directly from $G_{<_1}$ to $G_{<_2}$ even if $<_1$ and $<_2$ are not adjacent.

3. Converting a GB - zero dimensional

If I is zero dimensional we can do even better – FGLM [20]:

> Go directly from $G_{<_1}$ to $G_{<_2}$ even if $<_1$ and $<_2$ are not adjacent.

Let us illustrate some structures of $G_{<_1}$:

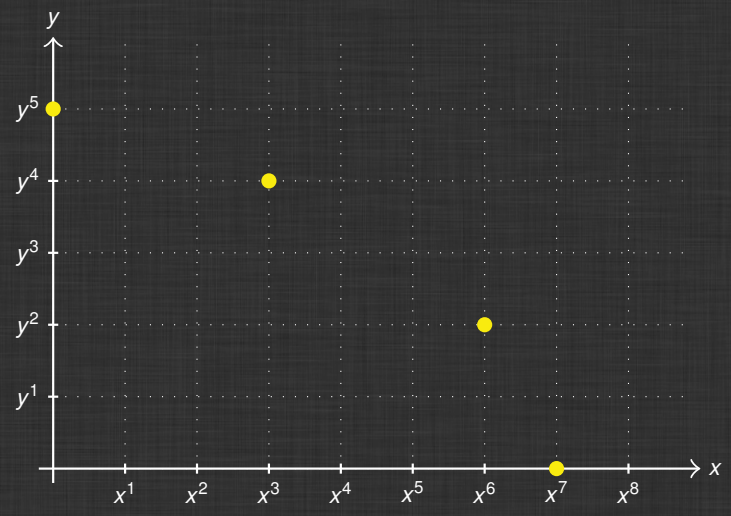


3. Converting a GB - zero dimensional

If I is zero dimensional we can do even better – FGLM [20]:

> Go directly from $G_{<_1}$ to $G_{<_2}$ even if $<_1$ and $<_2$ are not adjacent.

Let us illustrate some structures of $G_{<_1}$:

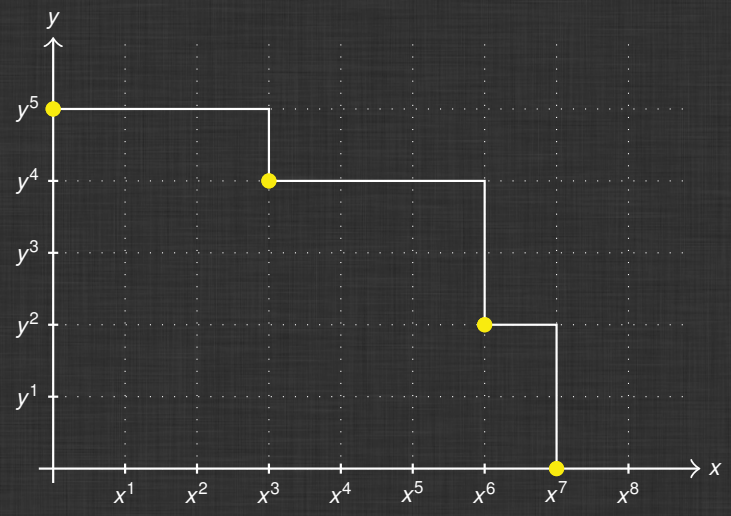


3. Converting a GB - zero dimensional

If I is zero dimensional we can do even better – FGLM [20]:

> Go directly from $G_{<_1}$ to $G_{<_2}$ even if $<_1$ and $<_2$ are not adjacent.

Let us illustrate some structures of $G_{<_1}$:

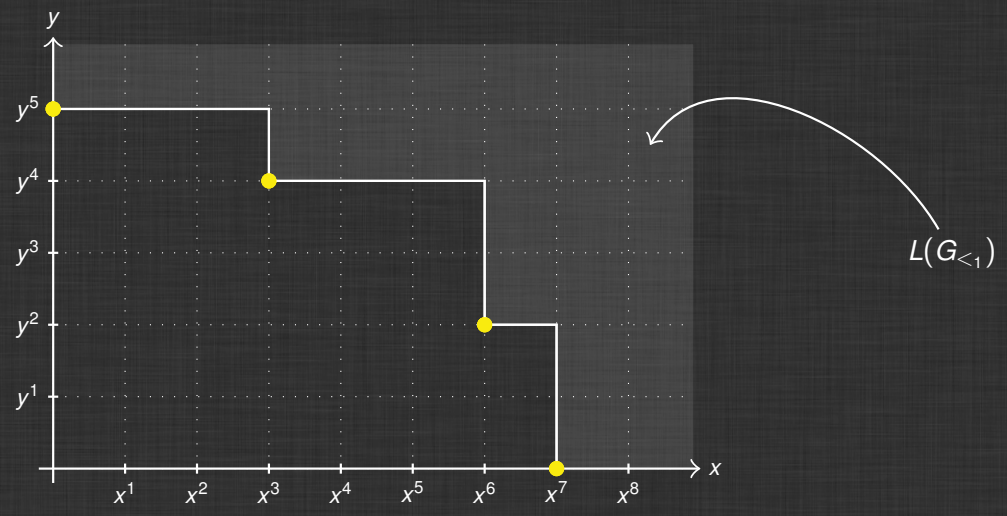


3. Converting a GB - zero dimensional

If I is zero dimensional we can do even better – FGLM [20]:

> Go directly from $G_{<_1}$ to $G_{<_2}$ even if $<_1$ and $<_2$ are not adjacent.

Let us illustrate some structures of $G_{<_1}$:

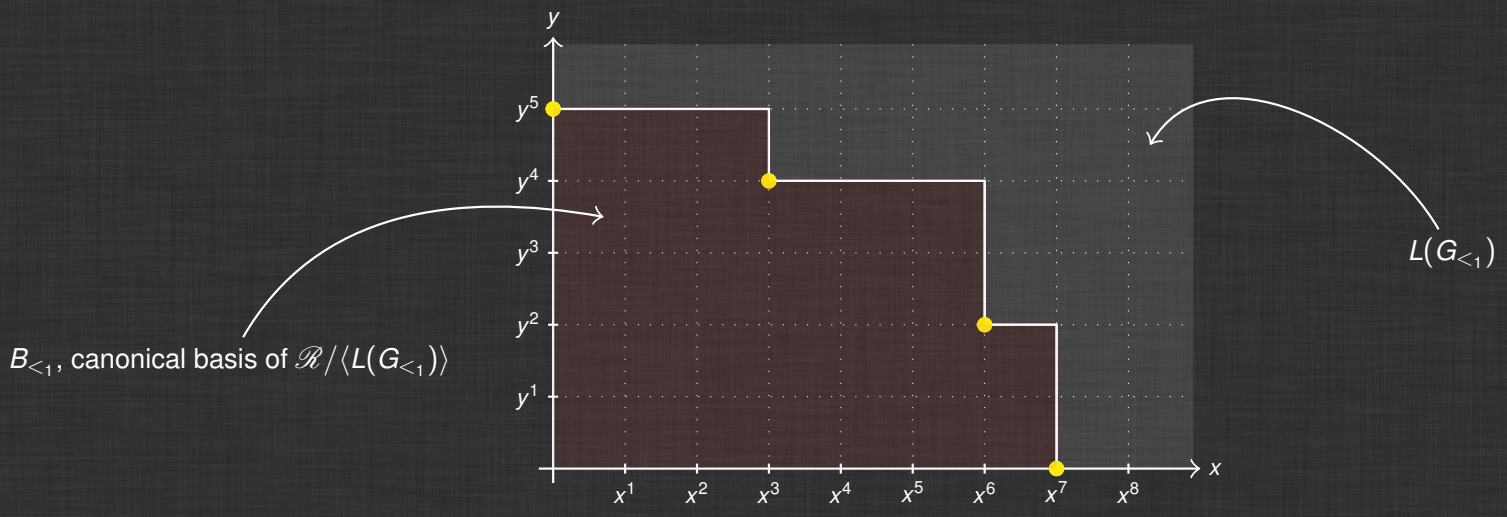


3. Converting a GB - zero dimensional

If I is zero dimensional we can do even better – FGLM [20]:

> Go directly from $G_{<_1}$ to $G_{<_2}$ even if $<_1$ and $<_2$ are not adjacent.

Let us illustrate some structures of $G_{<_1}$:



3. Converting a GB - zero dimensional

Given $G_{<_1} = \{g_1, \ldots, g_r\}$, how to get now $G_{<_2}$?

3. Converting a GB - zero dimensional

Given $G_{<_1} = \{g_1, \ldots, g_r\}$, how to get now $G_{<_2}$?

1. $B_{<_2} = \{1\}$, $G_{<_2} = \emptyset$
2. Take smallest monomial m w.r.t. $<_2$ such that $m \notin L(G_{<_2})$.

3. Converting a GB - zero dimensional

Given $G_{<_1} = \{g_1, \ldots, g_r\}$, how to get now $G_{<_2}$?

1. $B_{<_2} = \{1\}$, $G_{<_2} = \emptyset$
2. Take smallest monomial m w.r.t. $<_2$ such that $m \notin L(G_{<_2})$.
3. If $\mathrm{NF}_{<_1}(m)$ linearly independent w.r.t. $B_{<_2} \Rightarrow B_{<_2} \leftarrow B_{<_2} \cup \{\mathrm{NF}_{<_1}(m)\}$

3. Converting a GB - zero dimensional

Given $G_{<_1} = \{g_1, \ldots, g_r\}$, how to get now $G_{<_2}$?

1. $B_{<_2} = \{1\}$, $G_{<_2} = \emptyset$
2. Take smallest monomial m w.r.t. $<_2$ such that $m \notin L(G_{<_2})$.
3. If $\mathrm{NF}_{<_1}(m)$ linearly independent w.r.t. $B_{<_2} \Rightarrow B_{<_2} \leftarrow B_{<_2} \cup \{\mathrm{NF}_{<_1}(m)\}$
4. Else:

$$\mathrm{NF}_{<_1}(m) - \sum_{i=1}^{s} c_i \mathrm{NF}_{<_1}(b_i) = 0 \text{ for } c_i \in \mathscr{K}, b_i \in B_{<_2}$$

$$g := m - \sum_{i=1}^{s} c_i b_i \in I$$

$$\Rightarrow G_{<_2} \leftarrow G_{<_2} \cup \{g\}$$

3. Converting a GB - zero dimensional

Given $G_{<_1} = \{g_1, \ldots, g_r\}$, how to get now $G_{<_2}$?

1. $B_{<_2} = \{1\}$, $G_{<_2} = \emptyset$
2. Take smallest monomial m w.r.t. $<_2$ such that $m \notin L(G_{<_2})$.
3. If $\mathrm{NF}_{<_1}(m)$ linearly independent w.r.t. $B_{<_2} \Rightarrow B_{<_2} \leftarrow B_{<_2} \cup \{\mathrm{NF}_{<_1}(m)\}$
4. Else:

$$\mathrm{NF}_{<_1}(m) - \sum_{i=1}^{s} c_i \mathrm{NF}_{<_1}(b_i) = 0 \text{ for } c_i \in \mathscr{K}, b_i \in B_{<_2}$$

$$g := m - \sum_{i=1}^{s} c_i b_i \in I$$

$$\Rightarrow G_{<_2} \leftarrow G_{<_2} \cup \{g\}$$

5. Check if $\mathrm{lt}(g) = x_k^j$ for x_k largest variable w.r.t. $<_2$:
   - If $\mathrm{lt}(g) = x_k^j \Longrightarrow$ Terminate algorithm (Here we need "zero dimensional").
   - Else, take smallest monomial m' such that $m' >_2 m$ and $m' \notin L(G_{<_2})$. Go back to (1).

3. Converting a GB - zero dimensional

Optimize this process with linear algebra:

3. Converting a GB - zero dimensional

Optimize this process with linear algebra:
For $B_{<_1} = (b_1, \ldots, b_D)$ generate for each variable x_i multiplication matrices M_i (size $D \times D$):

$$\begin{array}{rccc} M_i: & B_{<_1} & \longrightarrow & B_{<_1} \\ & b_j & \longmapsto & \mathrm{NF}_{<_1}\left(x_i b_j\right). \end{array}$$

3. Converting a GB - zero dimensional

Optimize this process with linear algebra:
For $B_{<_1} = (b_1, \ldots, b_D)$ generate for each variable x_i multiplication matrices M_i (size $D \times D$):

$$\begin{aligned} M_i : \quad B_{<_1} &\longrightarrow B_{<_1} \\ b_j &\longmapsto \mathrm{NF}_{<_1}(x_i b_j). \end{aligned}$$

Change of order $\Longleftrightarrow$ Linear algebra
$\Rightarrow$ Complexity $O(nD^3)$ from Gaussian Elimination

3. Converting a GB - zero dimensional

Optimize this process with linear algebra:
For $B_{<_1} = (b_1, \ldots, b_D)$ generate for each variable x_i multiplication matrices M_i (size $D \times D$):

$$\begin{array}{rccc} M_i: & B_{<_1} & \longrightarrow & B_{<_1} \\ & b_j & \longmapsto & \mathrm{NF}_{<_1}(x_i b_j). \end{array}$$

Change of order $\Longleftrightarrow$ Linear algebra
$\Rightarrow$ Complexity $O(nD^3)$ from Gaussian Elimination

Implementations: **FGLM** in nearly all CAS

References I

[1] Albrecht, M. and Perry, J. F4/5. http://arxiv.org/abs/1006.4933, 2010.

[2] Arnold, E. A. *Computing Gröbner bases with Hilbert Lucky Primes*. PhD thesis, University of Maryland, College Park, MD, 2000.

[3] Arnold, E. A. Modular algorithms for computing Gröbner bases. *Journal of Symbolic Computation*, 35:403–419, April 2003.

[4] Arri, A. and Perry, J. The F5 Criterion revised. *Journal of Symbolic Computation*, 46(2):1017–1029, June 2011. Preprint online at arxiv.org/abs/1012.3664.

[5] Ars, G. *Applications des bases de Gröbner à la cryptographie*. PhD thesis, Université de Rennes I, 2005.

[6] Ars, G. and Hashemi, A. Extended F5 Criteria. *Journal of Symbolic Computation, MEGA 2009 special issue*, 45(12):1330–1340, 2010.

[7] Buchberger, B. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal*. PhD thesis, University of Innsbruck, 1965.

[8] Buchberger, B. Ein algorithmisches Kriterium für die Lösbarkeit eines algebraischen Gleichungssystems. *Aequ. Math.*, 4(3):374–383, 1970.

References II

[9] Buchberger, B. A criterion for detecting unnecessary reductions in the construction of Gröbner bases. In *EUROSAM '79, An International Symposium on Symbolic and Algebraic Manipulation*, volume 72 of *Lecture Notes in Computer Science*, pages 3–21. Springer, 1979.

[10] Buchberger, B. Gröbner Bases: An Algorithmic Method in Polynomial Ideal Theory. pages 184–232, 1985.

[11] Caboara, M. A Dynamic Algorithm for Gröbner basis computation. In *ISSAC'93*, 1993.

[12] Caboara, M. and Perry, J. Reducing the size and number of linear programs in a dynamic Gröbner basis algorithm. *Appl. Algebra Eng. Commun. Comput.*, 25(1-2):99–117, 2014.

[13] Eder, C. Improving incremental signature-based Groebner bases algorithms. *ACM SIGSAM Communications in Computer Algebra*, 47(1):1–13, 2013. http://arxiv.org/abs/1201.6472.

[14] Eder, C. Predicting zero reductions in Gröbner basis computations. submitted to Journal of Symbolc Computation, preprint at http://arxiv.org/abs/1404.0161, 2014.

[15] Eder, C., Gash, J., and Perry, J. Modifying Faugère's F5 Algorithm to ensure termination. *ACM SIGSAM Communications in Computer Algebra*, 45(2):70–89, 2011. http://arxiv.org/abs/1006.0318.

References III

[16] Eder, C. and Perry, J. F5C: A Variant of Faugère's F5 Algorithm with reduced Gröbner bases. *Journal of Symbolic Computation, MEGA 2009 special issue*, 45(12):1442–1458, 2010. dx.doi.org/10.1016/j.jsc.2010.06.019.

[17] Eder, C. and Perry, J. Signature-based Algorithms to Compute Gröbner Bases. In *ISSAC 2011: Proceedings of the 2011 international symposium on Symbolic and algebraic computation*, pages 99–106, 2011.

[18] Eder, C. and Roune, B. H. Signature Rewriting in Gröbner Basis Computation. In *ISSAC 2013: Proceedings of the 2013 international symposium on Symbolic and algebraic computation*, pages 331–338, 2013.

[19] Faugère, J.-C. A new efficient algorithm for computing Gröbner bases without reduction to zero F5. In *ISSAC'02, Villeneuve d'Ascq, France*, pages 75–82, July 2002. Revised version from http://fgbrs.lip6.fr/jcf/Publications/index.html.

[20] Faugère, J.-C., Gianni, P. M., Lazard, D., and Mora, T. Efficient Computation of Zero-Dimensional Gröbner Bases by Change of Ordering. *Journal of Symbolic Computation*, 16(4):329–344, 1993.

[21] Faugère, J.-C. and Joux, A. Algebraic Cryptanalysis of Hidden Field Equation (HFE) Cryptosystems Using Gröbner Bases. 2729:44–60, 2003.

References IV

[22] Faugère, J.-C., Safey El Din, M., and Spaenlehauer, P.-J. Gröbner Bases of Bihomogeneous Ideals Generated by Polynomials of Bidegree (1,1): Algorithms and Complexity. *Journal of Symbolic Computation*, 46(4):406–437, 2011. Available online 4 November 2010.

[23] Faugère, J.-C., Safey El Din, M., and Verron, T. On the complexity of Computing Gröbner Bases for Quasi-homogeneous Systems. In *Proceedings of the 38th international symposium on International symposium on symbolic and algebraic computation*, ISSAC '13, pages 189–196, New York, NY, USA, 2013. ACM.

[24] Faugère, J.-C. and Svartz, J. Solving polynomial systems globally invariant under an action of the symmetric group and application to the equilibria of n vertices in the plane. In *Proceedings of the 37th international symposium on International symposium on symbolic and algebraic computation*, ISSAC '12, pages 170–178, New York, NY, USA, 2012. ACM.

[25] Faugère, J.-C. and Svartz, J. Gröbner Bases of ideals invariant under a Commutative group : the Non-modular Case. In *Proceedings of the 38th international symposium on International symposium on symbolic and algebraic computation*, ISSAC '13, pages 347–354, New York, NY, USA, 2013. ACM.

[26] Faugère, J.-C. and Rahmany, S. Solving systems of polynomial equations with symmetries using SAGBI-Gröbner bases. In *ISSAC '09: Proceedings of the 2009 international symposium on Symbolic and algebraic computation*, ISSAC '09, pages 151–158, New York, NY, USA, 2009. ACM.

References V

[27] Galkin, V. Simple signature-based Groebner basis algorithm. http://arxiv.org/abs/1205.6050, 2012.

[28] Gao, S., Guan, Y., and Volny IV, F. A new incremental algorithm for computing Gröbner bases. In *ISSAC '10: Proceedings of the 2010 international symposium on Symbolic and algebraic computation*, pages 13–19. ACM, 2010.

[29] Gao, S., Volny IV, F., and Wang, D. A new algorithm for computing Groebner bases. http://eprint.iacr.org/2010/641, 2010.

[30] Gao, S., Volny IV, F., and Wang, D. A new algorithm for computing Groebner bases (rev. 2011). http://www.math.clemson.edu/~sgao/papers/gvw.pdf, 2011.

[31] Gash, J. M. *On efficient computation of Gröbner bases*. PhD thesis, University of Indiana, Bloomington, IN, 2008.

[32] Gash, J. M. A provably terminating and speed-competitive variant of F5 – F5t. *submitted to the Journal of Symbolic Computation*, 2009.

[33] Gebauer, R. and Möller, H. M. On an installation of Buchberger's algorithm. *Journal of Symbolic Computation*, 6(2-3):275–286, October/December 1988.

[34] Gerdt, V. P. and Hashemi, A. On the use of Buchberger criteria in G2V algorithm for calculating Gröbner bases. *Program. Comput. Softw.*, 39(2):81–90, March 2013.

References VI

[35] Gerdt, V. P., Hashemi, A., and M.-Alizadeh, B. Involutive Bases Algorithm Incorporating F5 Criterion. *J. Symb. Comput.*, 59:1–20, 2013.

[36] Giovini, A., Mora, T., Niesi, G., Robbiano, L., and Traverso, C. "One sugar cube, please" or selection strategies in the Buchberger algorithm. In *ISSAC'91*, pages 49–54, 1991.

[37] Huang, L. A new conception for computing Gröbner basis and its applications. `http://arxiv.org/abs/1012.5425`, 2010.

[38] Idrees, N., Pfister, G., and Steidel, S. Parallelization of Modular Algorithms. *Journal of Symbolic Computation*, 46:672–684, 2011.

[39] Jensen, A. N. Gfan, a software system for Gröbner fans and tropical varieties. Available at http://www.math.tu-berlin.de/ jensen/software/gfan/gfan.html.

[40] Kalkbrener, M. On the complexity of Gröbner Bases Conversion. *Journal of Symbolic Computation*, 28:265–273, 1999.

[41] Pan, S., Hu, Y., and Wang, B. The Termination of Algorithms for Computing Gröbner Bases. `http://arxiv.org/abs/1202.3524`, 2012.

[42] Pan, S., Hu, Y., and Wang, B. The Termination of the F5 Algorithm Revisited. In *ISSAC 2013: Proceedings of the 2013 international symposium on Symbolic and algebraic computation*, pages 291–298, 2013.

References VII

[43] Roune, B. H. and Stillman, M. Practical Gröbner Basis Computation. In *ISSAC 2012: Proceedings of the 2012 international symposium on Symbolic and algebraic computation*, 2012.

[44] Stegers, T. Faugère's F5 Algorithm revisited. Master's thesis, Technische Univerität Darmstadt, revised version 2007.

[45] Sun, Y. and Wang, D. K. A generalized criterion for signature related Gröbner basis algorithms. In *ISSAC 2011: Proceedings of the 2011 international symposium on Symbolic and algebraic computation*, pages 337–344, 2011.

[46] Sun, Y., Wang, D. K., Ma, D. X., and Zhang, Y. A signature-based algorithm for computing Gröbner bases in solvable polynomial algebras. In *ISSAC 2012: Proceedings of the 2011 international symposium on Symbolic and algebraic computation*, pages 351–358, 2012.

[47] Traverso, C. Hilbert Functions and the Buchberger Algorithm. *Journal of Symbolic Computation*, 22(4):355–376, 1996.

[48] Volny, F. *New algorithms for computing Gröbner bases*. PhD thesis, Clemson University, 2011.