Gröbner Basis Computations

Christian Eder

May 22, 2014 $\Rightarrow$ SKATEGOTT $\Rightarrow$ June 05, 2014
Conventions

- \( \mathbb{R} = \mathbb{K}[x_1, \ldots, x_n] \), \( \mathbb{K} \) field, \( < \) well-ordering on \( \text{Mon}(x_1, \ldots, x_n) \)

- \( f \in \mathbb{R} \) can be represented in a unique way by \( < \).
  \( \Rightarrow \) Definitions as \( \text{lc}(f), \text{lm}(f), \) and \( \text{lt}(f) \) make sense.

- An ideal \( I \) in \( \mathbb{R} \) is an additive subgroup of \( \mathbb{R} \) such that for \( f \in I \), \( g \in \mathbb{R} \) it holds that \( fg \in I \).

- \( G = \{g_1, \ldots, g_s\} \subset \mathbb{R} \) is a Gröbner basis for \( I = \langle f_1, \ldots, f_m \rangle \) w.r.t. \( < \):
  \[
  \iff \quad G \subset I \text{ and } L_<(G) = L_<(I)
  \]
Definition
Let $f \in R$ and let $t$ be a term of $f$.

- We can reduce $t$ by $g \in R$ if $\exists b$ such that $\text{lt}(bg) = t$.
  Outcome: $f - bg$.

- Reduce $f \in R$ to $h \in R$ by a sequence of reduction steps.

- Reductions always w.r.t. a finite subset $G \subset R$

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

Polynomial reduction
Polynomial reduction

**Definition**

Let \( f \in \mathbb{R} \) and let \( t \) be a term of \( f \).

- We can **reduce** \( t \) by \( g \in \mathbb{R} \) if \( \exists \ b \) such that \( \text{lt} (bg) = t \).

  Outcome: \( f - bg \).

- Reduce \( f \in \mathbb{R} \) to \( h \in \mathbb{R} \) by a sequence of reduction steps.

- Reductions always w.r.t. a finite subset \( G \subset \mathbb{R} \)

Notation: \( f \stackrel{G}{\rightarrow} h \).

\( \iff \) Another characterisation of Gröbner bases:
Polynomial reduction

Definition
Let \( f \in \mathbb{R} \) and let \( t \) be a term of \( f \).

- We can **reduce** \( t \) by \( g \in \mathbb{R} \) if \( \exists b \) such that \( \text{lt}(bg) = t \).
  
  Outcome: \( f - bg \).

- Reduce \( f \in \mathbb{R} \) to \( h \in \mathbb{R} \) by a sequence of reduction steps.

- Reductions always w.r.t. a finite subset \( G \subset \mathbb{R} \)

  Notation: \( f \xrightarrow{G} h \).

\[\implies\] Another characterisation of Gröbner bases:

**Gröbner basis (1)**

\( G \subset \mathbb{R} \) 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 $f \in \mathbb{R}$ and $G \subset \mathbb{R}$ 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

$$\max \{ \text{lt} (m_i g_i) \mid 1 \leq i \leq k \} \leq \text{lt} (f).$$
Even more characterizations!

**Standard representation**
Let \( f \in \mathbb{R} \) and \( G \subset \mathbb{R} \) 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

\[
\max \{ \text{lt} (m_i g_i) \mid 1 \leq i \leq k \} \leq \text{lt} (f).
\]

**Gröbner basis (2)**
\( G \subset \mathbb{R} \) finite is

1. a **Gröbner basis up to degree \( d \)** for \( I \) if
   \[G \subset I \text{ and for all } f \in I \text{ with } \text{deg}(f) \leq d : f \text{ has a standard representation w.r.t. } G;\]

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 \( f \not= 0, g \not= 0 \in \mathbb{R} \) and let \( \lambda = \text{lcm} (\text{lt}(f), \text{lt}(g)) \) be the least common multiple of \( \text{lt}(f) \) and \( \text{lt}(g) \). The S-polynomial between \( f \) and \( g \) is given by

\[
\text{spol}(f, g) := \frac{\lambda}{\text{lt}(f)} f - \frac{\lambda}{\text{lt}(g)} g.
\]
Buchberger's criterion

We are still not algorithmic!

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

\[
\text{spol}(f, g) := \frac{\lambda}{\text{lt}(f)} f - \frac{\lambda}{\text{lt}(g)} g.
\]

**Buchberger’s criterion** [7]
Let \( I = \langle f_1, \ldots, f_m \rangle \) be an ideal in \( \mathbb{R} \). A finite subset \( G \subset \mathbb{R} \) is a **Gröbner basis up to degree** \( d \) for \( I \) if \( G \subset I \) and for all \( f, g \in G \) with \( \deg(\text{spol}(f, g)) \leq d \) : \( \text{spol}(f, g) \xrightarrow{G} 0 \).
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 $P \leftarrow \{ \text{spol}(f_i, f_j) \mid f_i, f_j \in G, i > j \}$

4. Choose $p \in P$

   - $P \leftarrow P \setminus \{ p \}$

   - If $p \in G \Rightarrow h \neq 0$ ▶ new information

     - Build new S-pair with $h$

     - Add $h$ to $G$

     - Go on with the next element in $P$

   - Else ▶ no new information

     - Go on with the next element in $P$

5. When $P = \emptyset$ we are done and $G$ is a Gröbner basis of $I$. 
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 $p \in P$, $P \leftarrow P \setminus \{p\}$

---

/ 85
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 \( p \xrightarrow{G} 0 \) \( \Rightarrow \) \textbf{no new information} \\
       Go on with the next element in \( P \).
   (b) If \( p \xrightarrow{G} h \neq 0 \) \( \Rightarrow \) \textbf{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 \( I \).
In theory:

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

\[
g_1 = xy - z^2, \quad g_2 = y^2 - z^2
\]
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

\[
g_1 = xy - z^2, \quad g_2 = y^2 - z^2
\]

\[
spol(g_2, g_1) = xg_2 - yg_1 = xy^2 - xz^2 - xy^2 + yz^2 = -xz^2 + yz^2.
\]

\[
\Rightarrow g_3 = xz^2 - yz^2.
\]
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

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

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

$$\implies g_3 = xz^2 - yz^2.$$ 

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

\[
\begin{align*}
g_1 &= xy - z^2, \\
g_2 &= y^2 - z^2
\end{align*}
\]

\[\text{spol}(g_2, g_1) = xg_2 - yg_1 = xy^2 - xz^2 - xy^2 + yz^2 = -xz^2 + yz^2.\]

\[\implies g_3 = xz^2 - yz^2.\]

\[\text{spol}(g_3, g_1) = xyz^2 - y^2 z^2 - xyz^2 + z^4 = -y^2 z^2 + z^4.\]

We can reduce further using \( z^2 g_2 \):

\[-y^2 z^2 + z^4 + y^2 z^2 - z^4 = 0.\]
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?

\[ \text{spol}(g_3, g_2) = y^2 (xz^2 - yz^2) - xz^2 (y^2 - z^2) \]
\[ = \text{lt}(g_2)g_3 - \text{lt}(g_3)g_2 \]
\[ = \text{lt}(g_2)\text{lot}(g_3) - \text{lt}(g_3)\text{lot}(g_2) \]
How to detect zero reductions in advance?

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

\[ \text{spol}(g_3, g_2) = y^2 (xz^2 - yz^2) - xz^2 (y^2 - z^2) \]
\[ = \text{lt}(g_2)g_3 - \text{lt}(g_3)g_2 \]
\[ = \text{lt}(g_2)\text{lot}(g_3) - \text{lt}(g_3)\text{lot}(g_2) \]

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

\[ \rightarrow \text{lt}(g_2)\text{lot}(g_3) - g_2\text{lot}(g_3) - \text{lt}(g_3)\text{lot}(g_2) \]
\[ = - \text{lot}(g_2)\text{lot}(g_3) - \text{lt}(g_3)\text{lot}(g_2) \]
\[ = - g_3\text{lot}(g_2). \]
How to detect zero reductions in advance?

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

\[ \text{spol}(g_3, g_2) = y^2(xz^2 - yz^2) - xz^2(y^2 - z^2) \]
\[ = \text{lt}(g_2)g_3 - \text{lt}(g_3)g_2 \]
\[ = \text{lt}(g_2)\text{lot}(g_3) - \text{lt}(g_3)\text{lot}(g_2) \]

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

\[ \implies \text{lt}(g_2)\text{lot}(g_3) - g_2\text{lot}(g_3) - \text{lt}(g_3)\text{lot}(g_2) \]
\[ = -\text{lot}(g_2)\text{lot}(g_3) - \text{lt}(g_3)\text{lot}(g_2) \]
\[ = -g_3\text{lot}(g_2). \]

So we can reduce this to zero by \( vg_3 \) by all \( v \in \text{support(}\text{lot}(g_2)) \).
Buchberger’s criteria

Product criterion [8, 9]

If \( \text{lcm}(\text{lt}(f), \text{lt}(g)) = \text{lt}(f) \text{lt}(g) \) then \( \text{spol}(f, g) \approx \{f, g\} \overset{\text{G}}{\rightarrow} 0 \).
Buchberger’s criteria

**Product criterion** [8, 9]

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

Couldn’t we remove \( \text{spol}(g_3, g_2) \) in a different way?
Buchberger’s criteria

Product criterion [8, 9]

If \( \text{lcm} \left( \text{lt}(f), \text{lt}(g) \right) = \text{lt}(f) \text{lt}(g) \) then \( \text{spol}(f, g) \rightarrow 0. \)

Couldn’t we remove \( \text{spol}(g_3, g_2) \) in a different way?

\[
\text{lt}(g_1) = xy \mid xy^2 z^2 = \text{lcm} \left( \text{lt}(g_3), \text{lt}(g_2) \right)
\]
Buchberger’s criteria

**Product criterion [8, 9]**

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

Could’t we remove \( \text{spol}(g_3, g_2) \) in a different way?

\[
\text{lt}(g_1) = xy \mid xy^2 z^2 = \text{lcm}(\text{lt}(g_3), \text{lt}(g_2))
\]

\( \implies \) We can rewrite \( \text{spol}(g_3, g_2) \):

\[
\text{spol}(g_3, g_2) = \underbrace{y \text{spol}(g_3, g_1)}_{\xrightarrow{G \to 0}} - \underbrace{z^2 \text{spol}(g_2, g_1)}_{\xrightarrow{G \to -g_3}}
\]
Buchberger’s criteria

Product criterion [8, 9]
If $lcm(\text{lt}(f), \text{lt}(g)) = \text{lt}(f) \text{lt}(g)$ then $\text{spol}(f, g) \rightarrow 0$.

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

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

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

$$\text{spol}(g_3, g_2) = y \text{spol}(g_3, g_1) - z^2 \text{spol}(g_2, g_1)$$

$$\xrightarrow{G \rightarrow 0} \text{spol}(g_2, g_1) \xrightarrow{G \rightarrow -g_3} \text{spol}(g_3, g_2).$$

Standard representations of $\text{spol}(g_2, g_1)$ and $\text{spol}(g_3, g_1)$

$$\implies \text{Standard representation of } \text{spol}(g_3, g_2).$$
Chain criterion [10]
Let \( f, g, h \in \mathbb{R} \), \( G \subset \mathbb{R} \) finite. If
1. \( \text{lt}(h) \mid \text{lcm} (\text{lt}(f), \text{lt}(g)) \), and
2. \( \text{spol}(f, h) \) and \( \text{spol}(h, g) \) have a standard representation w.r.t. \( G \) respectively,
then \( \text{spol}(f, g) \) has a standard representation w.r.t. \( G \).
Buchberger’s criteria

Chain criterion [10]
Let $f, g, h \in \mathbb{R}$, $G \subset \mathbb{R}$ finite. If
1. $\text{lt}(h) | \text{lcm}(\text{lt}(f), \text{lt}(g))$, and
2. $\text{spol}(f, h)$ and $\text{spol}(h, g)$ have a standard representation w.r.t. $G$ respectively,
then $\text{spol}(f, g)$ has a standard representation w.r.t. $G$.

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

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

then we can remove $\text{spol}(f, g)$ or $\text{spol}(f, h)$ but not both!
Chain criterion [10]

Let \( f, g, h \in \mathbb{R}, \ G \subset \mathbb{R} \) finite. If

1. \( \text{lt}(h) \mid \text{lcm}(\text{lt}(f), \text{lt}(g)) \), and

2. \( \text{spol}(f, h) \) and \( \text{spol}(h, g) \) have a standard representation w.r.t. \( G \) respectively,

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

**Note**

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

\[
\text{spol}(f, g) = \lambda \text{spol}(f, h) + \sigma \text{spol}(h, g),
\]

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

How to combine Product and Chain criterion?
We add a new element $h$ to $G$ and generate new pairs $P' := \{(f, h) \mid f \in G\}$. 
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:
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.
   - $\text{lcm}(\text{lt}(f), \text{lt}(h))$,
   - $\text{lcm}(\text{lt}(f), \text{lt}(h)) \neq \text{lcm}(\text{lt}(f), \text{lt}(g))$,
   - $\text{lcm}(\text{lt}(g), \text{lt}(h)) \neq \text{lcm}(\text{lt}(f), \text{lt}(g))$

   $\implies$ 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.
   - $\text{lt}(h) \mid \text{lcm} \left(\text{lt}(f), \text{lt}(g)\right)$,
   - $\text{lcm} \left(\text{lt}(f), \text{lt}(h)\right) \neq \text{lcm} \left(\text{lt}(f), \text{lt}(g)\right)$,
   - $\text{lcm} \left(\text{lt}(g), \text{lt}(h)\right) \neq \text{lcm} \left(\text{lt}(f), \text{lt}(g)\right)$

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

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

   $\implies$ 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.
   - $\text{lt}(h) | \text{lcm}(\text{lt}(f), \text{lt}(g))$,
   - $\text{lcm}(\text{lt}(f), \text{lt}(h)) \neq \text{lcm}(\text{lt}(f), \text{lt}(g))$,
   - $\text{lcm}(\text{lt}(g), \text{lt}(h)) \neq \text{lcm}(\text{lt}(f), \text{lt}(g))$

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

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

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

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

   $\implies$ 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.
   - \( \text{lt}(h) \mid \text{lcm}(\text{lt}(f), \text{lt}(g)) \),
   - \( \text{lcm}(\text{lt}(f), \text{lt}(h)) \neq \text{lcm}(\text{lt}(f), \text{lt}(g)) \),
   - \( \text{lcm}(\text{lt}(g), \text{lt}(h)) \neq \text{lcm}(\text{lt}(f), \text{lt}(g)) \)

   \[\implies \text{Remove } (f, g) \text{ from } P. \quad [P \text{ done}]\]

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

   \[\implies \text{Remove } (g, h) \text{ from } P'.\]

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

   \[\implies \text{Remove } (g, h) \text{ from } P'. \quad [\text{Chain criterion done}]\]

4. If \((f, h) \in P'\) s.t. \( \text{lcm}(\text{lt}(f), \text{lt}(h)) = \text{lt}(f) \text{lt}(h) \)

   \[\implies \text{Remove } (f, h) \text{ from } P'. \quad [\text{Product criterion done}]\]
Can we do even better?

In our example we still need to consider

$$\text{spol}(g_3, g_1) \xrightarrow{G} 0.$$
Can we do even better?

In our example we still need to consider

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

How to get rid of this useless computation?
Can we do even better?

In our example we still need to consider

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

How to get rid of this useless computation?

Use more structure of $I \xrightarrow{\text{Signatures}}$
Let \( I = \langle f_1, \ldots, f_m \rangle \subset \mathcal{R} \).

**Idea:** Give each \( f \in I \) a bit more structure:

1. Let \( \mathcal{R}_m \) be generated by \( e_1, \ldots, e_m \) and let \( \prec \) be a compatible monomial order on the monomials of \( \mathcal{R}_m \).
2. Let \( \alpha \mapsto \alpha : \mathcal{R}_m \rightarrow \mathcal{R} \) such that \( e_i = f_i \) for all \( i \).
3. Each \( f \in I \) can be represented via some \( \alpha \in \mathcal{R}_m \):
   \[ f = \alpha \]
4. A signature of \( f \) is given by \( s(f) = \text{lt}(\prec)(\alpha) \) where \( f = \alpha \).
5. An element \( \alpha \in \mathcal{R}_m \) such that \( \alpha = 0 \) is called a syzygy.
Signatures

Let \( l = \langle f_1, \ldots, f_m \rangle \subset \mathcal{R} \).

**Idea:** Give each \( f \in l \) a bit more structure:

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

Let \( l = \langle f_1, \ldots, f_m \rangle \subset R \).

**Idea:** Give each \( f \in I \) a bit more structure:

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

2. Let \( \alpha \mapsto \bar{\alpha} : R^m \to R \) such that \( \bar{e}_i = f_i \) for all \( i \).
Signatures

Let $l = \langle f_1, \ldots, f_m \rangle \subset \mathcal{R}$.

Idea: Give each $f \in l$ a bit more structure:

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

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

3. Each $f \in l$ can be represented via some $\alpha \in \mathcal{R}^m$: $f = \overline{\alpha}$.
Let $I = \langle f_1, \ldots, f_m \rangle \subset \mathcal{R}$.

**Idea:** Give each $f \in I$ a bit more structure:

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

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

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

4. A **signature** of $f$ is given by $s(f) = \text{lt}_\prec(\alpha)$ where $f = \overline{\alpha}$. 
Let \( l = \langle f_1, \ldots, f_m \rangle \subset \mathcal{R} \).

**Idea:** Give each \( f \in l \) a bit more structure:

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

2. Let \( \alpha \mapsto \overline{\alpha} : \mathcal{R}^m \rightarrow \mathcal{R} \) such that \( \overline{e_i} = f_i \) for all \( i \).

3. Each \( f \in l \) can be represented via some \( \alpha \in \mathcal{R}^m \): \( f = \overline{\alpha} \).

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

5. An element \( \alpha \in \mathcal{R}^m \) such that \( \overline{\alpha} = 0 \) is called a **syzygy**.
Our example again – with signatures and \( \preceq_{\text{pot}} \)

\[ g_1 = xy - z^2, \quad \sigma(g_1) = e_1, \]
\[ g_2 = y^2 - z^2, \quad \sigma(g_2) = e_2. \]
Our example again – with signatures and $\prec_{\text{pot}}$

\[ g_1 = xy - z^2, \ s(g_1) = e_1, \]
\[ g_2 = y^2 - z^2, \ s(g_2) = e_2. \]

\[ g_3 = \text{spol}(g_2, g_1) = xg_2 - yg_1 \]
\[ \Rightarrow s(g_3) = x \cdot s(g_2) = xe_2. \]
Our example again – with signatures and $\prec_{\text{pot}}$

\[ g_1 = xy - z^2, \quad s(g_1) = e_1, \]
\[ g_2 = y^2 - z^2, \quad s(g_2) = e_2. \]

\[ g_3 = \text{spol}(g_2, g_1) = xg_2 - yg_1 \]
\[ \Rightarrow s(g_3) = xs(g_2) = xe_2. \]

\[ \text{spol}(g_3, g_1) = yg_3 - z^2g_1 \]
\[ \Rightarrow s(\text{spol}(g_3, g_1)) = ys(g_3) = yxe_2. \]
Our example again – with signatures and $\prec_{\text{pot}}$

\[ g_1 = xy - z^2, \quad s(g_1) = e_1, \]
\[ g_2 = y^2 - z^2, \quad s(g_2) = e_2. \]

\[ g_3 = \text{spol}(g_2, g_1) = xg_2 - yg_1 \]
\[ \Rightarrow s(g_3) = x s(g_2) = xe_2. \]

\[ \text{spol}(g_3, g_1) = yg_3 - z^2 g_1 \]
\[ \Rightarrow s(\text{spol}(g_3, g_1)) = y s(g_3) = yxe_2. \]

Note that $s(\text{spol}(g_3, g_1)) = xy e_2$ and $\text{lm}(g_1) = xy$. 
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.

Several elements with the same signature?

Choose 1 and remove the others.

**Our goal**: Make good choices.
How to use signatures?

**General idea:** Only 1 element per signature.

Several elements with the same signature?

**Choose 1 and remove the others.**

**Our goal:** Make good choices.

**Our task:** Keep signatures correct.
Think in the module

\[ \alpha \in \mathbb{R}^m \rightarrow \text{polynomial } \overline{\alpha} \text{ with } \text{lt}(\overline{\alpha}), \text{signature } s(\alpha) = \text{lt}(\alpha) \]
Think in the module

\[ \alpha \in \mathbb{R}^m \implies \text{polynomial } \overline{\alpha} \text{ with } \text{lt}(\overline{\alpha}), \text{signature } s(\alpha) = \text{lt}(\alpha) \]

S-pairs/S-polynomials:

\[ \text{spol}(\overline{\alpha}, \overline{\beta}) = a\overline{\alpha} - b\overline{\beta} \implies \text{spair}(\alpha, \beta) = a\alpha - b\beta \]
Think in the module

\[ \alpha \in \mathbb{R}^m \rightarrow \text{polynomial } \overline{\alpha} \text{ with } \text{lt}(\overline{\alpha}), \text{signature } s(\alpha) = \text{lt}(\alpha) \]

S-pairs/S-polynomials:

\[ \text{spol}(\overline{\alpha}, \overline{\beta}) = a\overline{\alpha} - b\overline{\beta} \quad \Rightarrow \quad \text{spair}(\alpha, \beta) = a\alpha - b\beta \]

\( s \)-reductions:

\[ \overline{\gamma} - d\overline{\delta} \quad \Rightarrow \quad \gamma - d\delta \]
Think in the module

\[ \alpha \in \mathbb{R}^n \implies \text{polynomial } \overline{\alpha} \text{ with } \text{lt}(\overline{\alpha}), \text{signature } s(\alpha) = \text{lt}(\alpha) \]

**S-pairs/S-polynomials:**

\[ \text{spol}(\overline{\alpha}, \overline{\beta}) = a\overline{\alpha} - b\overline{\beta} \implies \text{spair}(\alpha, \beta) = a\alpha - b\beta \]

**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

\[ s(\alpha) = s(\beta) \implies \text{Compute 1, remove 1.} \]
Signature-based criteria

\[ s(\alpha) = s(\beta) \implies \text{Compute 1, remove 1.} \]

Sketch of proof

1. \( s(\alpha - \beta) \prec s(\alpha), s(\beta) \).
2. All S-pairs are handled by increasing signature.
   \( \Rightarrow \) All relations \( \prec s(\alpha) \) 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 \( T \)?
Definition:

**$\mathcal{R}_T$**

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

**Question:**

What are all possible configurations to reach signature $T$?
Signature-based criteria

S-pairs in signature $T$

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

What are all possible configurations to reach signature $T$?

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

\[ R_T = \{ a\alpha \mid \alpha \text{ handled by the algorithm and } s(a\alpha) = T \} \]
Special cases

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

Choose \( b\beta \) to be an element of \( \mathcal{R}_T \) maximal w.r.t. an order \( \preceq \).
Special cases

\[ \mathcal{R}_T = \{ a\alpha \mid \alpha \text{ handled by the algorithm and } s(a\alpha) = T \} \]

Choose \( b\beta \) to be an element of \( \mathcal{R}_T \) maximal w.r.t. an order \( \preceq \).

1. If \( b\beta \) is a syzygy \( \implies \) Go on to next signature.
Special cases

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

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

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

Revisiting our example with \( \prec \) potentials \( \text{spol}(g_3, g_1) = xye \)
\( g_1 = xy - z^2 \)
\( g_2 = y^2 - z^2 \)
Special cases

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

Choose \( b\beta \) to be an element of \( \mathcal{R}_T \) maximal w.r.t. an order \( \preceq \).

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

Revisiting our example with \( \prec_{\text{pot}} \)

\[ s(\text{spol}(g_3, g_1)) = xye_2 \]
\[ g_1 = xy - z^2 \]
\[ g_2 = y^2 - z^2 \]
\[ \Rightarrow \text{psyz}(g_2, g_1) = g_1 e_2 - g_2 e_1 = xye_2 + \ldots \]
Variants covered by the survey
# zero reductions ($\text{Singular-4-0-0, } F_{32003}$)

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>STD</th>
<th>SBA $\lessdot_{\text{pot}}$</th>
<th>SBA $\lessdot_{\text{d-pot}}$</th>
<th>SBA $\lessdot_{\text{lt}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>cyclic-8</td>
<td>4,284</td>
<td>243</td>
<td>243</td>
<td>671</td>
</tr>
<tr>
<td>cyclic-8-h</td>
<td>5,843</td>
<td>243</td>
<td>243</td>
<td>671</td>
</tr>
<tr>
<td>eco-11</td>
<td>3,476</td>
<td>0</td>
<td>749</td>
<td>749</td>
</tr>
<tr>
<td>eco-11-h</td>
<td>5,429</td>
<td>502</td>
<td>502</td>
<td>749</td>
</tr>
<tr>
<td>katsura-11</td>
<td>3,933</td>
<td>0</td>
<td>0</td>
<td>353</td>
</tr>
<tr>
<td>katsura-11-h</td>
<td>3,933</td>
<td>0</td>
<td>0</td>
<td>353</td>
</tr>
<tr>
<td>noon-9</td>
<td>25,508</td>
<td>0</td>
<td>0</td>
<td>682</td>
</tr>
<tr>
<td>noon-9-h</td>
<td>25,508</td>
<td>0</td>
<td>0</td>
<td>682</td>
</tr>
<tr>
<td>Random(11,2,2)</td>
<td>6,292</td>
<td>0</td>
<td>0</td>
<td>590</td>
</tr>
<tr>
<td>HRandom(11,2,2)</td>
<td>6,292</td>
<td>0</td>
<td>0</td>
<td>590</td>
</tr>
<tr>
<td>Random(12,2,2)</td>
<td>13,576</td>
<td>0</td>
<td>0</td>
<td>1,083</td>
</tr>
<tr>
<td>HRandom(12,2,2)</td>
<td>13,576</td>
<td>0</td>
<td>0</td>
<td>1,083</td>
</tr>
</tbody>
</table>
### Time in seconds (Singular-4-0-0, $\mathbb{F}_{32003}$)

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

Yes, rather easily:

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

Note: There is a conjecture that for \( \langle \text{pot} \rangle \) the Product criterion is also included in the Rewritten criterion [14].
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].
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 $\prec_{\text{pot}}$ 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 \( \text{deg} \left( \text{spol} \left( f, g \right) \right) = d \) and \( \text{spol} \left( f, g \right) \xrightarrow{G} h \neq 0 \)
Why is homogeneous input nice?

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

\[ \implies \deg(h) = d \]
Why is homogeneous input nice?

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

\[ \implies \text{deg}(h) = d \]

No degree drop, we can compute by increasing degree!
Why is homogeneous input nice?

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

\[ \implies \text{deg} (h) = d \]

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?

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

\[ \implies \text{deg} (h) = d \]

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?

If \( \text{deg}(\text{spol}(f, g)) = d \) and \( \text{spol}(f, g) \xrightarrow{G} h \neq 0 \)
And if the input is not homogeneous?

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

\[ \implies \deg(h) \leq d \]
And if the input is not homogeneous?

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

\[ \implies \deg(h) \leq d \]

If \( \deg(h) < d \) then

1. \( \deg(\text{spol}(h, h')) \) might be \(< d\),

2. \( h \) is a possible reducer of elements already in \( G \).
And if the input is not homogeneous?

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

\[ \implies \deg(h) \leq d \]

If \( \deg(h) < d \) then

1. \( \deg(\text{spol}(h, h')) \) 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 \( I = \langle f_1, \ldots, f_m \rangle \).

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

\[ f - g \text{ and } g \]

But we could also take

\[ f - g \text{ and } f \]
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**

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

Let $h$ be another intermediate S-polynomial such that

- $\exists \lambda' \in \text{Mon}$ with $\lambda' \text{lt}(g) = \text{lt}(h)$
- $\lambda | \lambda'$

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**

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

Let $h$ be another intermediate S-polynomial such that

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

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 $\lambda g$ 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?

There are three algorithms using this:
▶ slimGB (SINGULAR); to some extent also std
▶ Faugère’s F4 algorithm (everywhere but not in SINGULAR)
▶ Signature-based algorithms (sba in SINGULAR, GVW, F5 in FGB)
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 $P \neq \emptyset$:
   ▶ If $lt(h) \not\in L(G)$ (all other $h$ are useless):
   ▶ $P \leftarrow P \cup \{\text{new pairs with } h\}$
   ▶ $G \leftarrow G \cup \{h\}$
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) \( P_d \leftarrow \text{Select}(P), P \leftarrow P \setminus P_d \)
   (c) \( L_d \leftarrow \{af, bg \mid (af, bg) \in P_d\} \)
   (d) \( L_d \leftarrow \text{Symbolic Preprocessing}(L_d, G) \)
   (e) \( F_d \leftarrow \text{Reduction}(L_d, G) \)
   (f) for \( h \in F_d \):
      ▶ If \( \text{lt}(h) \notin L(G) \) (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 $\mathcal{R}$ by a subset of $\mathcal{R}$
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 $\mathcal{R}$ by a subset of $\mathcal{R}$

If $\textbf{Select}(P)$ 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 $\mathbb{R}$
Output: a finite subset of $\mathbb{R}$

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 $\lambda \in \text{Mon}$ such that $\lambda \text{lt}(g) = m$

4. Return $F$
Symbolic preprocessing

**Input:** $L, G$ finite subsets of $\mathbb{R}$

**Output:** a finite subset of $\mathbb{R}$

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 $\lambda \in \text{Mon}$ such that $\lambda \text{lt}(g) = m$
      $\triangleright F \leftarrow F \cup \{\lambda g\}$

4. Return $F$

We optimize this soon!
Reduction

**Input:** $L, G$ finite subsets of $\mathbb{R}$

**Output:** a finite subset of $\mathbb{R}$

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 $\mathbb{R}$

**Output:** a finite subset of $\mathbb{R}$

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{=} \text{ monomials (sorted by monomial order $<$)}$
- rows $\hat{=} \text{ coeffs of polynomials in } L$
Example: Cyclic-4

\[ R = \mathbb{Q}[a, b, c, d], < \text{ denotes DRL and we use the normal selection strategy for } \text{Select}(P). \]

\[
l = \langle f_1, \ldots, f_4 \rangle, \text{ where}
\]

\[
f_1 = abcd - 1,
\]

\[
f_2 = abc + abd + acd + bcd,
\]

\[
f_3 = ab + bc + ad + cd,
\]

\[
f_4 = a + b + c + d.
\]
Example: Cyclic-4

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

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

We start with \(G = \{f_4\}\) and \(P_1 = \{(f_3, bf_4)\}\), thus \(L_1 = \{f_3, bf_4\}\).
Example: Cyclic-4

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

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

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:

\[
\begin{align*}
  T(L_1) &= \{ ab, b^2, bc, ad, bd, cd \} \\
  L_1 &= \{ f_3, bf_4 \}
\end{align*}
\]
Example: Cyclic-4

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

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

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**:

\[
\begin{align*}
  T(L_1) &= \{ ab, b^2, bc, ad, bd, cd \} \\
  L_1 &= \{ f_3, bf_4 \}
\end{align*}
\]

\( b^2 \notin L(G) \),
Example: Cyclic-4

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

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

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:

\[
\begin{align*}
  T(L_1) &= \{ ab, b^2, bc, ad, bd, cd \} \\
  L_1 &= \{ f_3, bf_4 \}
\end{align*}
\]

\( b^2 \notin L(G), bc \notin L(G) \),
Example: Cyclic-4

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

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

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:

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

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

\[b^2 \notin L(G), bc \notin L(G), d \text{lt}(f_4) = ad,\]
Example: Cyclic-4

\[ R = \mathbb{Q}[a, b, c, d], \ < \text{denotes DRL and we use the normal selection strategy} \]

for \textbf{Select}(P). \ I = \langle f_1, \ldots, f_4 \rangle, \ where

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

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

Let us do \textbf{symbolic preprocessing}:

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

\( b^2 \notin L(G), \ bc \notin L(G), \ d \lt f_4 = ad, \) all others also \( \notin L(G), \)
Example: Cyclic-4

Now reduction:
Convert polynomial data $L_1$ to Macaulay Matrix $M_1$

\[
\begin{pmatrix}
0 & 0 & 0 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 0 & 1 & 0 \\
1 & 1 & 1 & 0 & 1 & 0 & 0 \\
\end{pmatrix}
\]
**Example: Cyclic-4**

Now **reduction**:  
Convert polynomial data $L_1$ to Macaulay Matrix $M_1$

\[
\begin{pmatrix}
ab & b^2 & bc & ad & bd & cd & d^2 \\
df_4 & \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
bf_4 & \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 1 & 0 \\
f_3 & \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\
\end{pmatrix}
\end{pmatrix}
\end{pmatrix}
\]

**Gaussian Elimination of $M_1$:**

\[
\begin{pmatrix}
ab & b^2 & bc & ad & bd & cd & d^2 \\
df_4 & \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
bf_4 & \begin{pmatrix} 0 & 1 & 0 & 0 & 2 & 0 & 1 \\
f_3 & \begin{pmatrix} 1 & 0 & 1 & 0 & -1 & 0 & -1 \\
\end{pmatrix}
\end{pmatrix}
\end{pmatrix}
\]
Example: Cyclic-4

Convert matrix data back to polynomial structure $F_1$:

\[
\begin{bmatrix}
ab & b^2 & bc & ad & bd & cd & d^2 \\
df_4 & 0 & 0 & 0 & 1 & 1 & 1 \\
f_3 & 1 & 0 & 1 & 0 & -1 & 0 & -1 \\
bf_4 & 0 & 1 & 0 & 0 & 2 & 0 & 1 \\
\end{bmatrix}
\]

\[
F_1 = \left\{ \begin{array}{l}
ad + bd + cd + d^2, \\
ab + bc - bd - d^2, \\
b^2 + 2bd + d^2 \\
\end{array} \right\}
\]
Example: Cyclic-4

Convert matrix data back to polynomial structure $F_1$:

$$
\begin{pmatrix}
  ab & b^2 & bc & ad & bd & cd & d^2 \\
 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
 1 & 0 & 1 & 0 & -1 & 0 & -1 \\
 0 & 1 & 0 & 0 & 2 & 0 & 1
\end{pmatrix}
$$

$$
F_1 = \begin{cases}
  ad + bd + cd + d^2, & f_5 \\
  ab + bc - bd - d^2, & f_6 \\
  b^2 + 2bd + d^2, & f_7
\end{cases}
$$

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

$$
G \leftarrow G \cup \{f_7\}.
$$
Example: Cyclic-4

Next round:

\[ G = \{ f_4, f_7 \}, \quad P_2 = \{ (f_2, bcf_4) \}, \quad L_2 = \{ f_2, bcf_4 \}. \]
Next round:

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

We can simplify the computations:

\[ \text{lt}(bcf_4) = abc = \text{lt}(cf_6). \]

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

\[ \implies L_2 = \{ f_2, cf_6 \} \]
Example: Cyclic-4

Next round:

\[ G = \{ f_4, f_7 \}, \quad P_2 = \{(f_2, bcf_4)\}, \quad L_2 = \{ f_2, bcf_4 \}. \]

We can simplify the computations:

\[ \text{lt}(bcf_4) = abc = \text{lt}(cf_6). \]

\( f_6 \) possibly better reduced than \( f_4 \). \((f_6 \text{ is not in } G!\)\)

\[ \implies L_2 = \{ f_2, cf_6 \} \]

Symbolic preprocessing:

\[ T(L_2) = \{ abc, bc^2, abd, acd, bcd, cd^2 \} \]

\[ L_2 = \{ f_2, cf_6, \} \]
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:

\[ \text{lt}(bcf_4) = abc = \text{lt}(cf_6). \]

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

\[ \implies L_2 = \{f_2, cf_6\} \]

Symbolic preprocessing:

\[ T(L_2) = \{abc, bc^2, abd, acd, bcd, cd^2\} \]

\[ L_2 = \{f_2, cf_6\} \]

\( bc^2 \notin L(G) \),
Example: Cyclic-4

Next round:

\[ G = \{f_4, f_7\}, \quad P_2 = \{ (f_2, bcf_4) \}, \quad L_2 = \{f_2, bcf_4\}. \]

We can simplify the computations:

\[ \text{lt}(bcf_4) = abc = \text{lt}(cf_6). \]

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

\[ \implies L_2 = \{f_2, cf_6\} \]

Symbolic preprocessing:

\[ T(L_2) = \{abc, bc^2, abd, acd, bcd, cd^2\} \]

\[ L_2 = \{f_2, cf_6, \} \]

\( bc^2 \notin L(G) \), \( abd = \text{lt}(bdf_4) \), but also \( abd = \text{lt}(bf_5) \)!
Example: Cyclic-4

Next round:

\[ G = \{ f_4, f_7 \}, \quad P_2 = \{(f_2, bcf_4)\}, \quad L_2 = \{ f_2, bcf_4 \}. \]

We can simplify the computations:

\[ \text{lt}(bcf_4) = abc = \text{lt}(cf_6). \]

\[ f_6 \text{ possibly better reduced than } f_4. (f_6 \text{ is not in } G!) \]

\[ \implies L_2 = \{ f_2, cf_6 \} \]

Symbolic preprocessing:

\[ T(L_2) = \{ abc, bc^2, abd, acd, bcd, cd^2 \} \]

\[ L_2 = \{ f_2, cf_6 \} \]

\[ bc^2 \notin L(G), \quad abd = \text{lt}(bdf_4), \text{ but also } abd = \text{lt}(bf_5)! \]

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 $M_i$.
⇒ 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 $M_i$.  
⇒ Reuse rows that are reduced but not “in” $G$.

**Input:** monomial $u$, polynomial $f$, list $F$ 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 \( M_i \).
⇒ Reuse rows that are reduced but not “in” \( G \).

**Input:** monomial \( u \), polynomial \( f \), list \( F \) 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 \( \text{lt}(g) = \text{lt}(w \cdot f) \)
      
      ▶ if \( w \neq u \): Return \textbf{Simplify} \( \left( \frac{u}{w}, g, F \right) \) (recursive call)
      
      ▶ else: Return \( 1 \cdot g \)

4. else: Return \( u \cdot f \)
Interlude – Simplify

Note

- Tries to reuse all rows from old matrices. ⇒ We need to keep them in memory.
- We also simplify generators of S-pairs, as we have done in our example: \((f_2, bcf_4) \Rightarrow (f_2, cf_6)\).
- 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.
  ⇒ We need to keep them in memory.
▶ We also simplify generators of S-pairs, as we have done in our example: \((f_2, bcf_4) \rightarrow (f_2, cf_6)\).
▶ 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:

\[ T(L_2) = \{ abc, bc^2, abd, acd, bcd, cd^2 \} \]
\[ L_2 = \{ f_2, cf_6 \} \]

\( bc^2 \notin L(G) \),
Symbolic preprocessing - now with simplify:

\[ T(L_2) = \{ abc, bc^2, abd, acd, bcd, cd^2 \} \]
\[ L_2 = \{ f_2, cf_6 \} \]

\( bc^2 \notin L(G) \), \( abd = \text{lt}(bf_5) \),
Symbolic preprocessing - now with simplify:

\[
T(L_2) = \{ abc, bc^2, abd, acd, bcd, cd^2, b^2d, c^2d \}
\]

\[
L_2 = \{ f_2, cf_6, bf_5 \}
\]

\( bc^2 \notin L(G), \ abd = \text{lt}(bf_5), \)
Example: Cyclic-4

Symbolic preprocessing - now with \texttt{simplify}:

\[
T(L_2) = \{ abc, bc^2, abd, acd, bcd, cd^2, b^2d, c^2d, \ldots \}
\]

\[
L_2 = \{ f_2, cf_6, bf_5, cf_5, df_7 \}
\]

\(bc^2 \not\in L(G)\), \(abd = \text{lt}(bf_5)\), and so on.
Symbolic preprocessing - now with **simplify**:

\[
T(L_2) = \{ abc, bc^2, abd, acd, bcd, cd^2, b^2d, c^2d, \ldots \}
\]

\[
L_2 = \{ f_2, cf_6, bf_5, cf_5, df_7 \}
\]

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

\[
\begin{align*}
1 & \ 3 & 0 & 0 & 7 & 1 & 0 \\
1 & 0 & 4 & 1 & 0 & 0 & 5 \\
0 & 1 & 6 & 0 & 8 & 0 & 1 \\
0 & 5 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 0 & 0 & 1 & 3 & 1 \\
\end{align*}
\]
Improve Gaussian Elimination

Use Linear Algebra for reduction steps in GB computations.

\[
\begin{pmatrix}
1 & 3 & 0 & 0 & 7 & 1 & 0 \\
1 & 0 & 4 & 1 & 0 & 0 & 5 \\
0 & 1 & 6 & 0 & 8 & 0 & 1 \\
0 & 5 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 0 & 0 & 1 & 3 & 1 \\
\end{pmatrix}
\]

Knowledge of underlying GB structure
Improve Gaussian Elimination

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

<table>
<thead>
<tr>
<th>S-pair</th>
<th>1 3 0 0 7 1 0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 0 4 1 0 0 5</td>
</tr>
<tr>
<td></td>
<td>0 1 6 0 8 0 1</td>
</tr>
<tr>
<td>S-pair</td>
<td>0 5 0 0 0 2 0</td>
</tr>
<tr>
<td>reducer</td>
<td>← 0 0 0 0 1 3 1</td>
</tr>
</tbody>
</table>

Knowledge of underlying GB structure
**Improve Gaussian Elimination**

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

<table>
<thead>
<tr>
<th>S-pair</th>
<th>1 3 0 0 7 1 0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 0 4 1 0 0 5</td>
</tr>
<tr>
<td></td>
<td>0 1 6 0 8 0 1</td>
</tr>
<tr>
<td>S-pair</td>
<td>0 5 0 0 0 2 0</td>
</tr>
<tr>
<td>reducer</td>
<td>0 0 0 0 1 3 1</td>
</tr>
</tbody>
</table>

Knowledge of underlying GB structure
**Improve Gaussian Elimination**

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

<table>
<thead>
<tr>
<th>S-pair</th>
<th>1 3 0 0 7 1 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-pair</td>
<td>1 0 4 1 0 0 5</td>
</tr>
<tr>
<td>reducer</td>
<td>0 1 6 0 8 0 1</td>
</tr>
<tr>
<td></td>
<td>0 5 0 0 0 2 0</td>
</tr>
<tr>
<td></td>
<td>0 0 0 0 1 3 1</td>
</tr>
</tbody>
</table>

Knowledge of underlying GB structure

**Idea**

Do a static **reordering before** the Gaussian Elimination to achieve a better initial shape. **Reorder afterwards.**
1st step: Sort pivot and non-pivot columns

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>3</th>
<th>0</th>
<th>0</th>
<th>7</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>2nd</td>
<td>0</td>
<td>1</td>
<td>6</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3rd</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>4th</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>
Faugère-Lachartre Idea

1st step: Sort pivot and non-pivot columns

<table>
<thead>
<tr>
<th>1</th>
<th>3</th>
<th>0</th>
<th>0</th>
<th>7</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>6</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

Pivot column
**Faugère-Lachartre Idea**

**1st step**: Sort pivot and non-pivot columns

```
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
```

Pivot column
**Faugère-Lachartre Idea**

**1st step:** Sort pivot and non-pivot columns

\[
\begin{align*}
1 & 3 & 0 & 0 & 7 & 1 & 0 \\
1 & 0 & 4 & 1 & 0 & 0 & 5 \\
0 & 1 & 6 & 0 & 8 & 0 & 1 \\
0 & 5 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 0 & 0 & 1 & 3 & 1
\end{align*}
\]
**1st step**: Sort pivot and non-pivot columns

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>6</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

Pivot column

Non-Pivot column
1st step: Sort pivot and non-pivot columns

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

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

Pivot column
Non-Pivot column
Faugère-Lachartre Idea

**2nd step**: Sort pivot and non-pivot rows

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>8</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>9</td>
</tr>
<tr>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>
2nd step: Sort pivot and non-pivot rows

<table>
<thead>
<tr>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>8</td>
<td>6</td>
<td>0</td>
<td>9</td>
</tr>
<tr>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
</tbody>
</table>
### Faugère-Lachartre Idea

**2nd step**: Sort pivot and non-pivot rows

<p>| | | | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>8</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Pivot row

Non-Pivot row
2nd step: Sort pivot and non-pivot rows

Faugère-Lachartre Idea
2nd step: Sort pivot and non-pivot rows

Faugère-Lachartre Idea

Pivot row

Non-Pivot row
Faugère-Lachartre Idea

3rd step: Reduce lower left part to zero

```
1 0 0 4 1 0 5
0 5 0 0 0 2 0
0 0 1 0 0 3 1
0 1 8 6 0 0 9
```
Faugère-Lachartre Idea

3rd step: Reduce lower left part to zero

\[
\begin{array}{cccc|cccc}
1 & 0 & 0 & 4 & 1 & 0 & 5 \\
0 & 5 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 1 & 0 & 0 & 3 & 1 \\
1 & 3 & 7 & 0 & 0 & 1 & 0 \\
0 & 1 & 8 & 6 & 0 & 0 & 9 \\
\end{array}
\]

\[
\begin{array}{cccc|cccc}
1 & 0 & 0 & 4 & 1 & 0 & 5 \\
0 & 5 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 1 & 0 & 0 & 3 & 1 \\
0 & 0 & 0 & 7 & 1 & 0 & 3 \\
0 & 0 & 0 & 6 & 0 & 2 & 1 \\
\end{array}
\]
4th step: Reduce lower right part
**Faugère-Lachartre Idea**

**4th step:** Reduce lower right part

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>2</td>
</tr>
</tbody>
</table>

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
</tbody>
</table>

Remap columns of 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 \textsc{katsura}-12, degree 6 matrix
- Size 137 MB
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 $A^{-1}B$
Reduce $C$ to zero
Gaussian Elimination on D
$B \leftarrow A^{-1} B$
$B \leftarrow A^{-1}B$ – Block Version

$A_{xpyBlock}$
$B \leftarrow A^{-1} B$ – 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. $\text{lcm}(g)$ needs to be adjusted to meet $\text{lcm}(f)$
2. all coeffs in $g$ have to be multiplied by $\frac{\text{lcm}(f)}{\text{lcm}(g)}$
Coefficient growth over the rationals is problematic. In each reduction step $f - \lambda g$

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

Once coeffs get really big this implies a tremendous slow down of the overall computations.
Coefficient growth over the rationals is problematic.

In each reduction step $f - \lambda g$

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

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 $\mathbb{F}_p$

[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:**
     \[
     \prod_{i=1}^{s} \mathbb{Z}_{p_i}[x_1, \ldots, x_n] \rightarrow \mathbb{Z}_N[x_1, \ldots, x_n]
     \]
     
     \[
     G_{p_1} \times \cdots \times G_{p_s} \rightarrow G_N
     \]
   - 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:
     \[
     \prod_{i=1}^{s} \mathbb{Z}_{p_i}[x_1, \ldots, x_n] \rightarrow \mathbb{Z}_N[x_1, \ldots, x_n]
     \]
     \[
     G_{p_1} \times \cdots \times G_{p_s} \mapsto G_N
     \]
     - 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 $I = \langle f_1, \ldots, f_m \rangle$. 

Sloppy $HR/I(d)$ measures how many elements of degree $d$ a GB $G$ for $I$ has. This number does not depend on the monomial order chosen.
1. Hilbert functions, statically

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

\[
H_{\mathbb{R}/I}(d) \leq H_{\mathbb{R}/L(G)}(d) \quad \text{for all} \quad d,
\]

\[
H_{\mathbb{R}/I}(d) = H_{\mathbb{R}/L(G)}(d) \quad \text{for all} \quad d \quad \text{if} \quad G \quad \text{is a GB for} \quad I.
\]
1. Hilbert functions, statically

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

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

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

\[ H_{\mathbb{R}} / L\langle_1(G)(d) = H_{\mathbb{R}} / L\langle_2(G)(d) \text{ for all } d. \]
1. Hilbert functions, statically

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

\[
H_{\mathbb{R}/I}(d) \leq H_{\mathbb{R}/L(G)}(d) \quad \text{for all } d,
\]

\[
H_{\mathbb{R}/I}(d) = H_{\mathbb{R}/L(G)}(d) \quad \text{for all } d \text{ if } G \text{ is a GB for } I.
\]

Even more: For two monomial orders \(<_1\) and \(<_2\) it holds that

\[
H_{\mathbb{R}/L<_1(G)}(d) = H_{\mathbb{R}/L<_2(G)}(d) \quad \text{for all } d.
\]

**Sloppy**

\( H_{\mathbb{R}/I}(d) \) measures how many elements of degree \( d \) a GB \( G \) for \( I \) has. This number does not depend on the monomial order chosen.
1. Hilbert functions, statically

We use this fact in the following way:

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

We use this fact in the following way:

1. Get the Hilbert function $H(t) := H_{\mathcal{R}/I}(t)$
   - 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 $\text{deg}(f) = d$ is added to $G_2$ we check if $H(t) = H_{\mathcal{R}/L(G_2)}(t)$ for all $t \geq d$. 

Implementation: \text{stdhilb} in SINGULAR
1. Hilbert functions, statically

We use this fact in the following way:

1. Get the Hilbert function $H(t) := H_{\mathcal{R}/I}(t)$
   - 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 $H(t) = H_{\mathcal{R}/L(G_2)}(t)$ for all $t \geq d$.
     - If equal $\implies$ Return $G_2$.
     - Else there exists $d' \geq d$ such that $H(d') < H_{\mathcal{R}/L(G_2)}(d')$: Remove all S-polynomials of degree $< d'$ from $P$ and go on.
1. Hilbert functions, statically

We use this fact in the following way:

1. Get the Hilbert function $H(t) := H_{R/I}(t)$
   - 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 $H(t) = H_{R/L(G_2)}(t)$ for all $t \geq d$.
     - If equal $\implies$ Return $G_2$.
     - Else there exists $d' \geq d$ such that $H(d') < H_{R/L(G_2)}(d')$: 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 \( H(t) := H_{\mathcal{R}/I}(t) \)
   - 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 \( H(t) := H_{\mathcal{R}/I}(t) \)
   - 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
     \( H(t) = H_{\mathcal{R}/L(G_2)}(t) \) 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 \( H(t) := H_{R/I}(t) \)
   - 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 \( H(t) = H_{R/I(L(G_2))}(t) \) for all \( t \geq d \).
     - If equal \( \implies \) Return \( G_2 \).
     - Else there exists \( d' \geq d \) such that \( H(d') < H_{R/I(L(G_2))}(d') \): 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 \( H(t) := H_{\mathcal{R}/I}(t) \)
   - 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 \( H(t) = H_{\mathcal{R}/L(G_2)}(t) \) for all \( t \geq d \).
     - If equal \( \Rightarrow \) Return \( G_2 \).
     - Else there exists \( d' \geq d \) such that \( H(d') < H_{\mathcal{R}/L(G_2)}(d') \): Remove all S-polynomials of degree \(< d' \) from \( P \).
     - Find a better order \(<_{\text{new}}\) such that \( H_{\mathcal{R}/L_{\text{new}}(G_2)}(t) \) 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 \( H(t) := H_{R \cap I}(t) \)
   - 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
     \( H(t) = H_{R / I(G_2)}(t) \) for all \( t \geq d \).
     - If equal \( \implies \) Return \( G_2 \).
     - Else there exists \( d' \geq d \) such that \( H(d') < H_{R / I(G_2)}(d') \): Remove all
       S-polynomials of degree \(< d'\) from \( P \).
     - Find a better order \(<_{\text{new}}\) such that \( H_{R / I(G_2)}(t) \) 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** $\prec$? Should be near the order one wants, transform $G$ with methods explained in the following.
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” $H_{R/L}(G)(t)$? 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. **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”** $H_{R/L(G)}(t)$? 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 $v \in \mathbb{R}^n$ be a weight vector, $I = \langle f_1, \ldots, f_m \rangle \subseteq \mathbb{R}$ an ideal, and let $G$ be a GB for $I$ w.r.t. some order $\prec$. 

▶ $f \in I$, initial element of $f$ w.r.t. $v$: in $v(f) = t$, deg $v(t)$ maximal for all $t \in \text{support}(f)$.

▶ Initial ideal of $I$ w.r.t. $v$: in $v(I) = \langle \text{in}_v(f_1), \ldots, \text{in}_v(f_m) \rangle$.

(Not necessarily a monomial ideal!)

▶ Refinement of $v$: $x^\alpha(v, \prec) x^\beta$: $\iff$ deg $v(x^\alpha)$ < nat deg $v(x^\beta)$ or deg $v(x^\alpha)$ = deg $v(x^\beta)$ and $x^\alpha < x^\beta$.

▶ Gröbner cone: $C_\prec(G) = \{ v \in \mathbb{R}^n | L_\prec(\text{in}_v(f)) \}$ for all $f \in G$.

▶ The Gröbner fan is the fan $\Delta_G$ consisting of all $C_\prec(G)$ where $\prec$ runs over all monomial orders on $\mathbb{R}$.

Note: There are only finitely many not equivalent monomial orders on $\mathbb{R} = \Rightarrow$ Gröbner cone is well-defined.
3. Converting a GB

Let $v \in \mathbb{R}^n$ be a weight vector, $I = \langle f_1, \ldots, f_m \rangle \subset \mathbb{R}$ 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. $v$: $\text{in}_v(f) = t$, $\text{deg}_v(t)$ maximal for all $t \in \text{support}(f)$.

- Initial ideal of $I$ w.r.t. $v$: $\text{in}_v(I) = \langle \text{in}_v(f_1), \ldots, \text{in}_v(f_m) \rangle$.
  (Not necessarily a monomial ideal!)
3. Converting a GB

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

- $f \in I$, initial element of $f$ w.r.t. $v$: $\text{in}_v(f) = t$, $\text{deg}_v(t)$ maximal for all $t \in \text{support}(f)$.

- Initial ideal of $I$ w.r.t. $v$: $\text{in}_v(I) = \langle \text{in}_v(f_1), \ldots, \text{in}_v(f_m) \rangle$.
  (Not necessarily a monomial ideal!)

- Refinement of $v$:
  
  $x^\alpha(v, \prec)x^\beta \iff \text{deg}_v(x^\alpha) \prec_{\text{nat}} \text{deg}_v(x^\beta)$ or
  
  $\text{deg}_v(x^\alpha) = \text{deg}_v(x^\beta)$ and $x^\alpha \prec x^\beta$. 

- Gröbner cone: $C_{\prec}(G) = \{ v \in \mathbb{R}^n | L_{\prec}(\text{in}_v(f)) \}$ for all $f \in G$.

- The Gröbner fan is the fan $\triangle_G$ consisting of all $C_{\prec}(G)$ where $\prec$ runs over all monomial orders on $\mathbb{R}$.

  Note: There are only finitely many not equivalent monomial orders on $\mathbb{R} = \Rightarrow$ Gröbner cone is well-defined.
3. Converting a GB

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

- $f \in I$, initial element of $f$ w.r.t. $\nu$: $\text{in}_\nu(f) = t$, $\deg_\nu(t)$ maximal for all $t \in \text{support}(f)$.

- Initial ideal of $I$ w.r.t. $\nu$: $\text{in}_\nu(I) = \langle \text{in}_\nu(f_1), \ldots, \text{in}_\nu(f_m) \rangle$.
  (Not necessarily a monomial ideal!)

- Refinement of $\nu$:
  
  \[ x^\alpha(\nu, \prec)x^\beta :\Leftrightarrow \deg_\nu(x^\alpha) <_{\text{nat}} \deg_\nu(x^\beta) \text{ or } \deg_\nu(x^\alpha) = \deg_\nu(x^\beta) \text{ and } x^\alpha < x^\beta. \]

- Gröbner cone: $C_{\prec}(G) = \{ \nu \in \mathbb{R}^n \mid L_{\prec}(\text{in}_\nu(f)) = \text{lt}_{\prec}(f) \text{ for all } f \in G \}$. 
3. Converting a GB

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

- $f \in I$, initial element of $f$ w.r.t. $v$: $\text{in}_v(f) = t$, $\text{deg}_v(t)$ maximal for all $t \in \text{support}(f)$.

- Initial ideal of $I$ w.r.t. $v$: $\text{in}_v(I) = \langle \text{in}_v(f_1), \ldots, \text{in}_v(f_m) \rangle$.
  (Not necessarily a monomial ideal!)

- Refinement of $v$:
  \[ x^\alpha (v, \lt) x^\beta : \iff \text{deg}_v(x^\alpha) <_{\text{nat}} \text{deg}_v(x^\beta) \text{ or } \text{deg}_v(x^\alpha) = \text{deg}_v(x^\beta) \text{ and } x^\alpha < x^\beta. \]

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

- The Gröbner fan is the fan $\Delta_G$ consisting of all $C_{\lt}(G)$ where $\lt$ runs over all monomial orders on $\mathbb{R}$. 
3. Converting a GB

Let $v \in \mathbb{R}^n$ be a weight vector, $I = \langle f_1, \ldots, f_m \rangle \subset \mathbb{R}$ 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. $v$: $\text{in}_v(f) = t$, $\text{deg}_v(t)$ maximal for all $t \in \text{support}(f)$.

- Initial ideal of $I$ w.r.t. $v$: $\text{in}_v(I) = \langle \text{in}_v(f_1), \ldots, \text{in}_v(f_m) \rangle$.
  (Not necessarily a monomial ideal!)

- Refinement of $v$:
  \[ x^\alpha(v, <)x^\beta :\Leftrightarrow \text{deg}_v(x^\alpha) <_{\text{nat}} \text{deg}_v(x^\beta) \text{ or } \text{deg}_v(x^\alpha) = \text{deg}_v(x^\beta) \text{ and } x^\alpha < x^\beta. \]

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

Note

There are only finitely many not equivalent monomial orders on $\mathbb{R} \implies$ Gröbner cone is well-defined.
How does such a fan look like?
How does such a fan look like?
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.
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.
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.
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. $<_2$. 
3. Converting a GB

Conclusions

► Often better complexity when computing GB in DRL and walk to LEX.

► Problem if conversion between not adjacent orders
  \[ \implies \text{Several steps from cone to cone} \]

► 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!
3. Converting a GB

Conclusions

- Often better complexity when computing GB in DRL and walk to LEX.
- Problem if conversion between not adjacent orders
  \[ \Rightarrow \] 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 \( G_{<1} \):

\[
\begin{align*}
\begin{array}{cccccccc}
& & & & & y^1 & & \\
& & & & & y^2 & & \\
& & & & & y^3 & & \\
& & & & & y^4 & & \\
& & & & & y^5 & & \\
\end{array}
\end{align*}
\]

\[
\begin{array}{cccccccc}
x^1 & x^2 & x^3 & x^4 & x^5 & x^6 & x^7 & x^8
\end{array}
\]
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 \( G_{<1} \):
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 $\text{NF}_{<1}(m)$ linearly independent w.r.t. $B_{<2}$ $\Rightarrow B_{<2} \leftarrow B_{<2} \cup \{\text{NF}_{<1}(m)\}$
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 \( \text{NF}_{<1}(m) \) linearly independent w.r.t. \( B_{<2} \) \( \Rightarrow \) \( B_{<2} \leftarrow B_{<2} \cup \{\text{NF}_{<1}(m)\} \)

4. Else:

\[
\text{NF}_{<1}(m) - \sum_{i=1}^{s} c_i \text{NF}_{<1}(b_i) = 0 \quad \text{for} \quad c_i \in \mathbb{K}, b_i \in B_{<2}
\]

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

\( \Rightarrow \ G_{<2} \leftarrow G_{<2} \cup \{g\} \)
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 $\text{NF}_{<1}(m)$ linearly independent w.r.t. $B_{<2} \Rightarrow B_{<2} \leftarrow B_{<2} \cup \{\text{NF}_{<1}(m)\}$

4. Else:

   \[ \text{NF}_{<1}(m) - \sum_{i=1}^{s} c_i \text{NF}_{<1}(b_i) = 0 \text{ for } c_i \in \mathcal{K}, b_i \in B_{<2} \]

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

   $\Rightarrow G_{<2} \leftarrow G_{<2} \cup \{g\}$

5. Check if $\text{lt}(g) = x_k^j$ for $x_k$ largest variable w.r.t. $<_2$:
   - If $\text{lt}(g) = x_k^j \Rightarrow$ 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 $D \times D$):

$$M_i : \quad B_{<1} \quad \rightarrow \quad B_{<1}$$

$\quad b_j \quad \rightarrow \quad NF_{<1} (x_i b_j)$.
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 $D \times D$):

$$M_i : \quad B_{<1} \quad \longrightarrow \quad B_{<1}$$

$$b_j \quad \longrightarrow \quad \text{NF}_{<1} (x_i b_j).$$

Change of order $\iff$ 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 $D \times D$):

$$M_i : \quad B_{<1} \quad \rightarrow \quad B_{<1}$$

$$b_j \quad \rightarrow \quad NF_{<1} (x_i b_j).$$

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

Implementations: **FGLM** in nearly all CAS
References I


References II


References IV


