Journal of Symbolic Computation 80 (2017) 719–784

Contents lists available at ScienceDirect

Journal of Symbolic Computation

www.elsevier.com/locate/jsc

A survey on signature-based algorithms for computing Gröbner bases ☆

Christian Eder [a], Jean-Charles Faugère [b]

[a] University of Kaiserslautern, Department of Mathematics, PO Box 3049, 67653 Kaiserslautern, Germany
[b] Sorbonne Universités, UPMC Univ Paris 06, CNRS, INRIA, Laboratoire d'Informatique de Paris 6 (LIP6), Equipe PolSys, 4 place Jussieu, 75005 Paris, France

ARTICLE INFO

Article history:
Received 18 December 2015
Accepted 13 July 2016
Available online 26 July 2016

Keywords:
Gröbner bases
F5
GVW
Signature-based algorithms
Syzygies

ABSTRACT

In 1965 Buchberger introduced an algorithmic approach to compute Gröbner bases. Later on, he and many others presented various attempts to improve the computation by removing useless elements a priori. One approach, initiated by Gebauer, Möller, Mora and Traverso in the 1990s, is to keep track of the corresponding syzygies which is related to the topic of this survey: signature-based algorithms for Gröbner bases. This area was initiated by Faugère's **F5** algorithm in 2002. The general idea of signatures is to keep track of the history of the computation with a minimal overhead and to exploit this information to detect redundant elements. Here we give a summary of the literature on signature-based algorithms and show how to classify known algorithms by 3 different orderings. For this we give translations between different notations and show the relationships (differences and similarities) among many approaches. Moreover, we give a general description of how the idea of signatures is quite natural when performing the reduction process using linear algebra. We hope that this survey would help to outline this field of active research.

☆ The first author was supported by the EXACTA grant (ANR-09-BLAN-0371-01) of the French National Research Agency.
E-mail addresses: ederc@mathematik.uni-kl.de (C. Eder), Jean-Charles.Faugere@inria.fr (J.-C. Faugère).

http://dx.doi.org/10.1016/j.jsc.2016.07.031
0747-7171/© 2016 Elsevier Ltd. All rights reserved.

1. Introduction

In Buchberger (1965, 2006), Buchberger initiated the theory of Gröbner bases by which many fundamental problems in mathematics, science and engineering can be solved algorithmically. Specifically he introduced some key structural theory, and based on this theory, proposed the first algorithm for computing Gröbner bases. Buchberger's algorithm introduced the concept of critical pairs and repeatedly carries out a certain polynomial operation (called reduction).

Many of those reductions would be determined as "useless" (i.e. no contribution to the output of the algorithm), but only a posteriori, that is, after an (often expensive) reduction process. Thus intensive research was carried out, starting with Buchberger, to avoid the useless reductions via a priori criteria, for instance Buchberger (1979, 1985), Gebauer and Möller (1988).

There are many different attempts trying to avoid zero reductions, for example, the Gebauer–Möller implementation of Buchberger's Product and Chain criteria (Gebauer and Möller, 1988) or the Gröbner trace algorithms by Traverso (1988). This paper focuses on a different class of variants of Buchberger's algorithm: Those that try to remove all zero reductions under certain regularity conditions on the input.

Related is the work of Gebauer and Möller on staggered linear bases in Gebauer and Möller (1986) which was later on revised by Mora (2005) and Dellaca (2009). In 1992, Möller, Mora and Traverso presented a Gröbner basis algorithm that keeps track of all syzygies arising during the computation (Möller et al., 1992). Also being able to use this information for detecting useless elements the overhead of the module computation is often a burden.

In 2002, Faugère presented new criteria and a resulting algorithm called **F5**. It is based on *signatures* that are the leading monomials of the module representations of polynomials arising during the Gröbner basis computation. The key idea is to track most of the history of the computation with a minimal overhead of only one monomial. Moreover, Faugère proved that **F5** in fact avoids all useless reductions if the input is regular.

Beginning 2008, many researchers worked on understanding the new criteria behind **F5**, which lead to new insights, but also optimizations and generalizations of the signature-based approach (Eder, 2008a, 2008b; Eder and Perry, 2010; Arri and Perry, 2011).

While the question of **F5**'s termination was still an open one until recently (Galkin, 2012; Pan et al., 2012, 2013), many new signature-based algorithms were introduced, for example, **G2V** (Gao et al., 2010a) resp. **GVW** (Gao et al., 2010b, 2011, 2013; Volny, 2011) or **SB** (Roune and Stillman, 2012a, 2012b). Moreover, first papers trying to classify signature-based Gröbner basis algorithms came up (Huang, 2010; Eder and Perry, 2011; Sun and Wang, 2011a; Pan et al., 2012, 2013; Eder and Roune, 2013).

As a result, at the moment, the literature on signature-based algorithms is vast. In this paper, we try to give an organized survey on the literature. We will use uniform notations, in the hope that the readers would be able to quickly identify the main ideas and their relations (differences and similarities). Having studied existing algorithms in known literature we can construct a generic framework with the benefit of hindsight. This framework is used as a common core that can be specialized in various manners in order to become a link between the different known algorithms.

Since this is a survey, we do not give proofs if they are long, complex, or do not help in understanding the topic. We always explain the idea behind the proofs and refer to the related publication which includes a complete proof. There the reader is then, with our descriptions and explanations, able to understand the proof in the used notation and language. Table 1 gives the outline of this paper and can be used as an index for finding specific algorithms the reader might be interested in.

Fig. 1 gives a graphical overview on the connection for various algorithms that are explained in the following. Please note that **F5** is in the center of this figure due to be the first such algorithm in a chronological order. We tried to connect those algorithms that are in a relation to each other (and that are thus also grouped in the paper). The figure does explicitly not intend to prefer any algorithm over the other, but it tries to give a 2-dimensional representation on how this area of research evolved. All signature-based algorithms presented in this survey (besides **MatrixF5**) are variants of Buchberger's algorithm in the sense that they generate critical pairs and perform a polynomial reduction procedure.

Table 1
Various signature-based algorithms (in the order of appearance in this survey).

Name/case	Key features	Section	Reference
MatrixF5	uses Macaulay matrices and linear algebra for reduction purposes, does not build S-pairs but generates all multiples of the generators for a given degree step	3	Faugère and Joux (2003)
RB	generalized algorithm to compare signature-based algorithms; in Section 7 it is slightly generalized	7.2	Eder and Roune (2013)
F5	first signature-based algorithm	8	Faugère (2002)
F5'	homogenizes inhomogeneous input, interreduces intermediate Gröbner basis	8	Faugère (2002)
F5"	uses $<_{\text{d-pot}}$ instead of $<_{\text{pot}}$	8	Faugère (2002)
F5R	interreduces intermediate Gröbner basis, uses it only for reduction purposes	8.1	Stegers (2007)
F5C	interreduces intermediate Gröbner basis, uses it for reduction purposes and for creation of new S-pairs	8.2	Eder and Perry (2010)
F5A	variant of **F5C** directly using a zero reduction as signature for the syzygy module	8.2	Eder and Perry (2011)
iF5A	variant of **F5A** recomputing signatures after interreducing between two incremental steps, also **iG2V**, ...	8.2	Eder (2013b)
Extended **F5** criteria	uses different module monomial orders	8.3	Ars and Hashemi (2010)
F5/2	adds field equations to the input systems for computations over $\mathbb{F}_2$	9.1	Faugère and Joux (2003)
Bihomogeneous case	uses maximal minors of Jacobian matrices to enlarge system of syzygies	9.2	Faugère et al. (2011)
SAGBI Gröbner bases	uses the Reynolds operator on the syzygy criterion	9.3	Faugère and Rahmany (2009)
F5GEN	generalized algorithm for different rewrite orders, applicable with any compatible module monomial order	10.1	Pan et al. (2012, 2013)
F5t	uses the Macaulay bound, once it is exceeded the algorithm transforms to Buchberger's algorithm	10.2	Gash (2008, 2009)
F5B	uses two lists of S-pairs: one for usual **F5**, another one for computing a lower degree bound using Buchberger's chain criterion	10.2	Ars (2005)
F5+	distinguishes S-pairs needed for the Gröbner basis and those needed for **F5**'s correctness only, once only the later ones are left it uses the idea of **F5B**	10.2	Eder et al. (2011)
Arri & Perry's work	introduces rewrite order $\trianglelefteq_{\text{rat}}$, works for any compatible module monomial order, directly uses zero reduction as signature for syzygy module, also known by **AP**	11.1	Arri and Perry (2011)
TRB	generalized algorithm to compare **F5** and **GVW**, also introduces $\trianglelefteq_{\text{rat}}$ as rewrite order, applicable with any compatible module monomial order	11.2	Huang (2010)
GBGC	generalized algorithm, uses $\trianglelefteq_{\text{rat}}$ but also generalizes to use partial rewrite orders, applicable with any compatible module monomial order, later on further generalized to work on algebras of solvable type	11.3	Sun and Wang (2011a), Sun et al. (2012)
G2V	directly uses zero reduction as signature for syzygy module, rewriting is done implicitly w.r.t. $\trianglelefteq_{\text{add}}$, generalizes signatures	11.4	Gao et al. (2010a)
GVW	generalizes **G2V** to be applicable with any compatible module monomial order, gives a first easy characterization of signature Gröbner bases, uses $\trianglelefteq_{\text{rat}}$ since 2011 (and thus coincides with **AP**; also known as **GVWHS**)	11.5	Gao et al. (2010b, 2011, 2013, 2016)
mo-GVW	monomial-oriented **GVW** algorithm that uses Macaulay matrices like **MatrixF5**	11.5	Sun et al. (2014)
SB	coincides with **GVW** and **AP**	11.6	Roune and Stillman (2012a)
SSG	coincides with **SB**, **GVW** and **AP**	11.7	Galkin (2013)
ImpG2V	uses Buchberger's Product and Chain criterion in **G2V** (this is also introduced in the 2013 revision of **GVW**)	12	Gerdt and Hashemi (2013), Gao et al. (2013)
F4/5	uses **F4**-style $\mathfrak{s}$-reduction	13	Albrecht and Perry (2010)

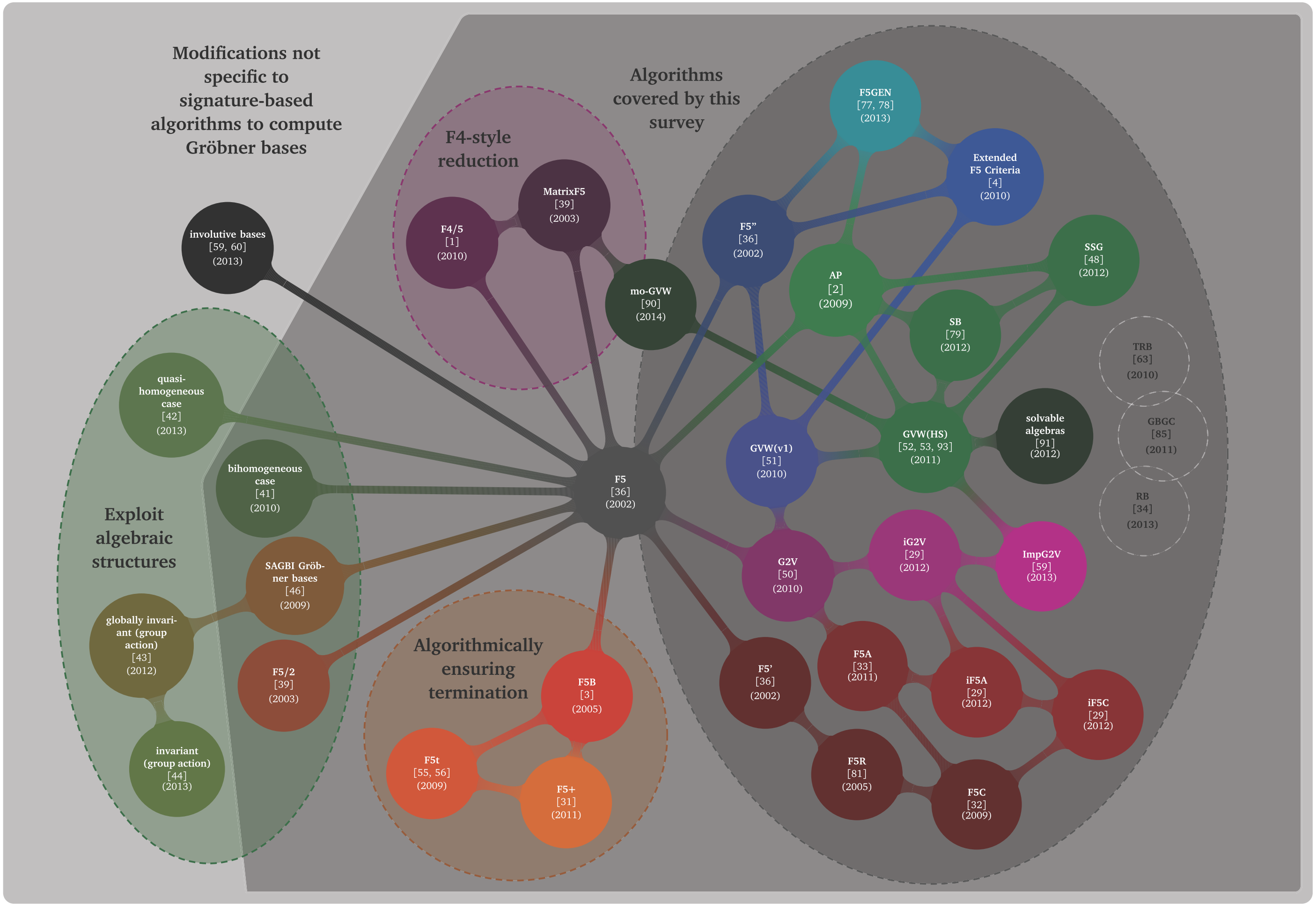

Modifications not specific to signature-based algorithms to compute Gröbner bases
Algorithms covered by this survey
F4-style reduction
Exploit algebraic structures
Algorithmically ensuring termination
involutive bases [59, 60] (2013)
F4/5 [1] (2010)
MatrixF5 [39] (2003)
mo-GVW [90] (2014)
F5GEN [77, 78] (2013)
Extended F5 Criteria [4] (2010)
F5” [36] (2002)
AP [2] (2009)
SSG [48] (2012)
SB [79] (2012)
TRB [63] (2010)
GBGC [85] (2011)
RB [34] (2013)
solvable algebras [91] (2012)
GVW(HS) [52, 53, 93] (2011)
GVW(v1) [51] (2010)
F5 [36] (2002)
quasi-homogeneous case [42] (2013)
bihomogeneous case [41] (2010)
SAGBI Gröbner bases [46] (2009)
globally invariant (group action) [43] (2012)
invariant (group action) [44] (2013)
F5/2 [39] (2003)
F5B [3] (2005)
F5t [55, 56] (2009)
F5+ [31] (2011)
G2V [50] (2010)
iG2V [29] (2012)
ImpG2V [59] (2013)
F5’ [36] (2002)
F5A [33] (2011)
iF5A [29] (2012)
iF5C [29] (2012)
F5R [81] (2005)
F5C [32] (2009)

Fig. 1. Relationship between various signature-based algorithms (status: January 2015).

In Section 10 we give the problem of proving **F5**'s termination an in-depth discussion, where we also explain how termination-ensuring algorithms as described in Ars (2005), Gash (2009), Eder et al. (2011) are still useful from an algorithmic point of view.

Moreover, we present descriptions of signature-based computation using linear algebra for the reduction process, see Sections 3 and 13. Besides (Albrecht and Perry, 2010) which is restricted to **F5** this is the first known discussion on this topic and shows how the ideas of signatures rather naturally arise in this setting.

Furthermore, we give in Section 14 detailed experimental results generated with various signature-based Gröbner basis algorithms presented in this survey. There we do not focus on timings, but on the characteristics of the different algorithms, like size of the resulting Gröbner basis, size of the recovered syzygy module, number of zero reductions and number of operations overall. The implementation of the various algorithms used for this is done in Singular (Decker et al., 2015) and available open-source. Thus the implementation is transparent and the reader is able to understand the different outcomes in the various algorithms.

We hope that this survey can be used as a useful handbook for researchers and students.

Next we give a short overview on the structure of this paper that can be used as a table of contents when using the survey as a reference.

Section 2 We review basic notations for polynomial rings, monomial orderings. Moreover we introduce the notion of signatures and Gröbner basis and state Buchberger's criterion.

Section 3 We discuss the algorithm **MatrixF5** as a first step into signature-based computations. Here the basic idea behind the exploitation of signatures is motivated. Using linear algebra instead of polynomial reduction it can provide an easy entry point to signature-based algorithms.

Section 4 We connect signatures and Gröbner bases, also introducing the term of signature Gröbner bases and show how they generalize the notion of Gröbner bases. Moreover, we discuss the changes to the polynomial reduction process due to taking care of signatures.

Section 5 We give a first generic algorithm for computing signature Gröbner bases. This algorithm is mainly a carryover of Buchberger's algorithm and aims to carefully introduce the reader to signature-based computations.

Section 6 This section discusses the idea of using signatures in order to detect redundant reductions. There we state the syzygy criterion and the singular criterion in general. Those are the criteria that are used, with different specifications, as pointed out later, in the efficient signature-based Gröbner basis algorithms known.

Section 7 Following this, the notion of rewrite bases is introduced. Rewrite bases generalize signature Gröbner bases. Using the signature-based criteria from the previous section we state a basic signature-based algorithm for computing rewrite bases, called **RB**. This generic framework is formulated retrospectively by studying existing algorithms in the literature. It covers all known signature-based algorithms for computing Gröbner bases discussed in this survey and is used as connecting link for the following instantiations. In the rest of the paper we refer to them a specializations of **RB**. Rewrite bases depend on a rewrite order that is defined in this section and used in the signature-based criteria to detect redundant reductions. Signature-based algorithms known up till now choose either $\trianglelefteq_{\text{add}}$ or $\trianglelefteq_{\text{rat}}$ as rewrite order. At the end of this section we are able to give a natural characterization of rewrite bases and signature Gröbner bases.

Section 8 This section is dedicated to a comparison of various signature-based algorithms that use $\trianglelefteq_{\text{add}}$ (**F5** et al.), whereas Section 11 discusses those incorporating $\trianglelefteq_{\text{rat}}$ (**GVW** et al.). Due to the fact that proving **F5**'s termination was an open problem for a rather long time, with many different attempts, this topic is discussed in more detail in Section 10.

Section 9 In given settings more algebraic structures are known, for example, if the input system is bihomogeneous or invariant under some group action. In this section we discuss several signature-based Gröbner basis algorithms that exploit these structures to improve the detection of useless computations.

Section 10 In this part we give an historical overview of the different approaches to proof the termination of Faugère's **F5** algorithm. One approach researcher used was to modify **F5** which lead to new algorithms that are discussed in this section, too.

Section 11 This section describes signature-based Gröbner basis algorithms that use $\trianglelefteq_{\text{rat}}$ as rewrite order. Here algorithms like **GVW**, **AP**, and **SB** are discussed.

Section 12 We give an overview on the connection between signature-based criteria and Buchberger's Product and Chain criterion to detect useless reductions.

Section 13 We explain how to use linear algebra for reductions considering signatures, here in the sense of considering S-pairs instead of full Macaulay matrices as done in **MatrixF5** (Section 3). At the end an **F4**-ish variant of **RB** which uses linear algebra instead of polynomial reduction is presented.

Section 14 We compare various signature-based algorithms for many different benchmarks (homogeneous and affine) in terms of zero reductions, size of resulting basis, size of syzygy module and number of operations.

Section 15 Here we give concluding remarks.

2. Notations and terminology

In this section we introduce notations and basic terminology used in this survey. Readers already familiar with signature-based algorithms might skip this section. Still note that notations itself play an important role in the following, especially when comparing different signature-based algorithms for Gröbner bases. We extend the notation introduced in Eder and Roune (2013). For a more extensive introduction to computational computer algebra we refer, for example, to Becker et al. (1993), Greuel and Pfister (2007), Kreuzer and Robbiano (2005, 2009).

Definition 2.1.

(a) A *set of terms* in n variables (or indeterminates) $x_1, \ldots, x_n$ over a field $\mathscr{K}$ is defined as

$$\mathscr{M} := \{\kappa_v x^v := \kappa_{v_1,\ldots,v_n} \prod_{i=1}^{n} x_i^{v_i} \mid v \in \mathbb{N}^n \text{ and } \kappa_v \in \mathscr{K}\}.$$

An element of $\mathscr{M}$ is called a *term*.

(b) A *polynomial f (over $\mathscr{K}$)* is a finite $\mathscr{K}$-linear combination of terms,

$$f = \sum_v \kappa_v x^v := \sum_{v\in\mathbb{N}^n}^{\text{finite}} \kappa_{v_1,\ldots,v_n} \prod_{i=1}^{n} x_i^{v_i}.$$

A *monomial* is a polynomial with exactly one term. A monomial with coefficient 1 is *monic*. Neither monomials nor terms of polynomials are necessarily monic. We write $f \simeq g$ for $f, g \in \mathscr{R}$ if there exists a non-zero $\kappa \in \mathscr{K}$ such that $f = \kappa g$.

(c) For $f = \kappa_v x^v \neq 0$ we call $\deg(f) := \max\{|v| \mid \kappa_v \neq 0\}$ where $|v| := v_1 + \cdots + v_n$ the *degree of f*. We set $\deg(f) = -1$ for $f = 0$.

(d) The *polynomial ring* $\mathscr{R} := \mathscr{K}[x] := \mathscr{K}[x_1, \ldots, x_n]$ in n variables over $\mathscr{K}$ is the set of all polynomials over $\mathscr{K}$ together with the usual addition and multiplication:

$$\sum_v \kappa_v x^v + \sum_v \xi_u x^v := \sum_v (\kappa_v + \xi_v) x^v,$$

$$\left(\sum_u \kappa_u x^u\right) \cdot \left(\sum_v \xi_v x^v\right) := \sum_w \left(\sum_{u+v=w} \kappa_u \xi_v\right) x^w.$$

(e) Let $\mathscr{R}^m$ be a free $\mathscr{R}$-module and let $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_m$ be the standard basis (unit vectors) in $\mathscr{R}^m$. A *set of module terms* in $\mathscr{R}^m$ $x_1, \ldots, x_n$ is defined as $\mathscr{N} := \{a\boldsymbol{e}_i \mid a \in \mathscr{M}\}$. An element of $\mathscr{N}$ is called a *module term*.

(f) A *module element* $\alpha \in \mathscr{R}^m$ can be written as a finite sum $\alpha = \sum_{a\boldsymbol{e}_i\in\mathscr{N}}^{\text{finite}} a\boldsymbol{e}_i$. The elements $a\boldsymbol{e}_i \in \mathscr{N}$ are the *terms* of α. A *module monomial* is an element of $\mathscr{R}^m$ with exactly one term. A module monomial with coefficient 1 is *monic*. Neither module monomials nor terms of module elements are necessarily monic. Let $\alpha \simeq \beta$ for $\alpha, \beta \in \mathscr{R}^m$ if $\alpha = \kappa\beta$ for some non-zero $\kappa \in \mathscr{K}$.

Next we try to connect a polynomial ring $\mathscr{R}$ with a special free module $\mathscr{R}^m$ lying over it.

Definition 2.2. Consider a finite sequence of polynomials $f_1, \ldots, f_m \in \mathscr{R}$ that we call the *input (polynomials)*.

(a) We call $f_1, \ldots, f_m$ a regular sequence if f_i is a non-zero-divisor of $\mathscr{R}/\langle f_1, \ldots, f_{i-1}\rangle$ for $i = 2, \ldots, m$.
(b) For $\alpha = \sum_{i=1}^m a_i \boldsymbol{e}_i \in \mathscr{R}^m$, $a_i \in \mathscr{R}$ we define the module homomorphism $\alpha \mapsto \overline{\alpha}$ from $\mathscr{R}^m$ to $\mathscr{R}$ by $\overline{\alpha} := \sum_{i=1}^m a_i f_i$. An element $\alpha \in \mathscr{R}^m$ with $\overline{\alpha} = 0$ is called a *syzygy*. The module of all syzygies of $f_1, \ldots, f_m$ is denoted by $\operatorname{syz}(f_1, \ldots, f_m)$.

Example 2.1. Let $f = xy - z^2, g = y^2 - z^2 \in \mathscr{R} = \mathscr{K}[xyz]$ be two polynomials. The finite sequence (f, g) is a regular sequence since g is a non-zero-divisor of $\mathscr{R}/\langle f\rangle$. Considering the module element $\alpha = f\boldsymbol{e}_2 - g\boldsymbol{e}_1 \in \mathscr{R}^m = \mathscr{R}^2$ together with the module homomorphism defined via $\overline{\boldsymbol{e}_1} = f$ and $\overline{\boldsymbol{e}_2} = g$ we see that $\overline{\alpha} = fg - gf = 0 \in \mathscr{R}$. Thus α is a syzygy, the principal or trivial syzygy of f and g.

Let $\leq$ denote two different orders – one for $\mathscr{R}$ and one for $\mathscr{R}^m$:

Definition 2.3.

(a) $\leq$ for $\mathscr{R}$ is a *monomial order*. A monomial order is a well-order on the set of monomials in $\mathscr{R}$ such that $a \leq b$ implies $ca \leq cb$ for all monomials $a, b, c \in \mathscr{R}$.
(b) $\leq$ for $\mathscr{R}^m$ is a *module monomial order*. A module monomial order is a well-order on the set of module monomials in $\mathscr{R}^m$ such that $S \leq T$ implies $cS \leq cT$ for all module monomials $S, T \in \mathscr{R}^m$ and monomials $c \in \mathscr{R}$.

Convention 2.1.

(a) *We require the two orders $\leq$ to be* compatible *in the sense that $a \leq b$ if and only if $a\boldsymbol{e}_i \leq b\boldsymbol{e}_i$ for all monomials $a, b \in \mathscr{R}$ and $i = 1, \ldots, m$.*
(b) *Given a monomial order $\leq$ for $\mathscr{R}$ we can write a polynomial $f \in \mathscr{R}$ in a unique ordered way via $f = \kappa_u x^u + \kappa_v x^v + \cdots + \kappa_w x^w$ such that $x^u > x^v > \cdots > x^w$ where no coefficient in the above representation is zero. Similarly we have a unique representation for elements $\alpha \in \mathscr{R}^m$ given a module monomial order $\leq$. In the following we always assume these unique representations.*

Next we introduce the notion of signatures together with related structures in the plain polynomial setting.

Definition 2.4.

(a) The *lead term* $\operatorname{lt}(f)$ of $f \in \mathscr{R} \setminus \{0\}$ is the $\leq$-maximal term of f. The *lead coefficient* $\operatorname{lc}(f)$ of f is the coefficient of $\operatorname{lt}(f)$. For a set $F \subset \mathscr{R}$ we define the *lead ideal of F* by $L(F) := \langle \operatorname{lt}(f) \mid f \in F\rangle$.
(b) The *signature* $\mathfrak{s}(\alpha)$ of $\alpha \in \mathscr{R}^m \setminus \{0\}$ denotes the $\leq$-maximal term of α. If $a\boldsymbol{e}_i = \mathfrak{s}(\alpha)$ then we call $\operatorname{ind}(\alpha) := i$ the *index* of α.
(c) For $\alpha \in \mathscr{R}^m$ we define the *sig-poly pair* of α by $(\mathfrak{s}(\alpha), \overline{\alpha}) \in \mathscr{R}^m \times \mathscr{R}$.
(d) $\alpha, \beta \in \mathscr{R}^m$ are *equal up to sig-poly pairs* if $\mathfrak{s}(\alpha) = \mathfrak{s}(\kappa\beta)$ and $\overline{\alpha} = \overline{\kappa\beta}$ for some non-zero $\kappa \in \mathscr{K}$. Correspondingly, α, β are said to be *equal up to sig-lead pairs* if $\mathfrak{s}(\alpha) = \mathfrak{s}(\kappa\beta)$ and $\operatorname{lt}(\overline{\alpha}) = \operatorname{lt}\left(\overline{\kappa\beta}\right)$ for some non-zero $\kappa \in \mathscr{K}$.

Remark 2.1. Note that Faugère initially introduced signatures *of polynomials* in Faugère (2002) by only taking the lead monomial into account. Moreover, in Faugère (2002) the signature is looked at from the polynomial point of view: Given a polynomial $f = \overline{\alpha}$ *the* signature is *uniquely defined* as

$$\min_{\leq}\{\mathfrak{s}(\beta) \mid \overline{\beta} = f\}.$$

At that time there was an overload of the term *signature* in the sense that there was a mathematical definition of it and there was the signature computed during the run of the algorithm. Here we define *a* signature and *a* sig-poly pair which means that for any element β such that $\overline{\beta} = f$, $\mathfrak{s}(\beta)$ is a signature for f. This more natural concept includes both above mentioned interpretations of the term *signature*. It corresponds to the definition of a signature that was first introduced by Gao, Guan, and Volny in **G2V** (Gao et al., 2010a), and later on it was used by Gao, Volny and Wang in **GVW** (Gao et al., 2016): There they define a signature not for a polynomial f but for an element $(\alpha, f) \in \mathscr{R}^m \times \mathscr{R}$. In our notation, $\mathfrak{s}(\alpha) \in \mathscr{R}^m$ is a signature for (α, f) such that $\overline{\alpha} = f$. See sections 11.4 and 11.5 for more information on this topic.

With these definitions every non-syzygy module element $\alpha \in \mathscr{R}^m$ has two main associated characteristics – the signature $\mathfrak{s}(\alpha) \in \mathscr{R}^m$ and the lead term $\operatorname{lt}(\overline{\alpha}) \in \mathscr{R}$ of its image $\overline{\alpha}$. Lead terms and signatures include a coefficient for mathematical convenience, though an implementation of an signature-based Gröbner Basis algorithm need not store the signature coefficients as we discuss in Sections 8 and 11.

We define some canonical module monomial orders that are useful in the following.

Definition 2.5. Let $<$ be a monomial order on $\mathscr{R}$ and let ae_i, be_j be two module monomials in $\mathscr{R}^m$.

(a) $ae_i <_{\text{pot}} be_j$ if and only if either $i < j$ or $i = j$ and $a < b$.
(b) $ae_i <_{\text{top}} be_j$ if and only if either $a < b$ or $a = b$ and $i < j$.

These two orders can be combined with either a weighted degree or a weighted leading monomial:

(a) $a\boldsymbol{e}_i <_{\text{d-pot}} b\boldsymbol{e}_j$ if and only if either $\deg\left(\overline{a\boldsymbol{e}_i}\right) < \deg\left(\overline{b\boldsymbol{e}_j}\right)$ or $\deg\left(\overline{a\boldsymbol{e}_i}\right) = \deg\left(\overline{b\boldsymbol{e}_j}\right)$ and $a\boldsymbol{e}_i <_{\text{pot}} b\boldsymbol{e}_j$. In the same way we define $a\boldsymbol{e}_i <_{\text{d-top}} b\boldsymbol{e}_j$.
(b) $a\boldsymbol{e}_i <_{\text{lt-pot}} b\boldsymbol{e}_j$ if and only if either $\operatorname{lt}\left(\overline{a\boldsymbol{e}_i}\right) < \operatorname{lt}\left(\overline{b\boldsymbol{e}_j}\right)$ or $\operatorname{lt}\left(\overline{a\boldsymbol{e}_i}\right) = \operatorname{lt}\left(\overline{b\boldsymbol{e}_j}\right)$ and $a\boldsymbol{e}_i <_{\text{pot}} b\boldsymbol{e}_j$. In the same way we define $a\boldsymbol{e}_i <_{\text{lt-top}} b\boldsymbol{e}_j$.

Note that $<_{\text{lt-pot}}$ is also known as Schreyer's order, for example, see Greuel and Pfister (2007).

$<_{\text{pot}}$ prefers the position in the module over the lead term in polynomial ring. On the other hand, $<_{\text{top}}$ first looks at the terms a and b in $\mathscr{M}$ and then breaks ties w.r.t. the position in the module.

Example 2.2. Note that a polynomial can have infinitely many different module representations with distinct signatures. Consider the three input polynomials $f_1 = x^2 - y^2$, $f_2 = xyz - z^3$, and $f_3 = yz^2 - xy$ in $\mathscr{R} = \mathbb{Q}[x, y, z]$ where $<$ denotes the graded reverse lexicographical monomial order. Moreover, assume $<$ to extend to $<_{\text{pot}}$ on the set of monomials of $\mathscr{R}^3$. For example, we can represent f_2 by $\boldsymbol{e}_2$. Since $\overline{f_1\boldsymbol{e}_3 - f_3\boldsymbol{e}_1} = 0$ another representation of f_2 might be $f_1\boldsymbol{e}_3 + \boldsymbol{e}_2 - f_3\boldsymbol{e}_1$. Note that the two representations of f_2 have two different signatures, $\boldsymbol{e}_2$ (corresponding to the algorithmic interpretation of the signature as mentioned in Remark 2.1) and $\operatorname{lt}(f_1)\boldsymbol{e}_3$, respectively. We also want to point out that $\operatorname{lt}\left(\overline{\mathfrak{s}(\alpha)}\right) \neq \operatorname{lt}(\overline{\alpha})$ is possible: In the above example $\operatorname{lt}\left(\overline{\mathfrak{s}(\boldsymbol{e}_2)}\right) = \operatorname{lt}(f_2)$, but $\operatorname{lt}\left(\overline{\mathfrak{s}(f_1\boldsymbol{e}_3 + \boldsymbol{e}_2 - f_3\boldsymbol{e}_1)}\right) = \operatorname{lt}(f_1)\operatorname{lt}(f_3) \neq \operatorname{lt}(f_2)$.

Finally, we review Buchberger's theory of Gröbner bases (Buchberger, 1965, 2006). For this, the reduction of polynomials is essential.

Definition 2.6 *(Reduction, Buchberger (1965, 2006))*. Let $f \in \mathscr{R}$ and let t be a term of f. Then we can *reduce* t by $g \in \mathscr{R}$ if there exists a monomial b such that $\operatorname{lt}(bg) = t$. The outcome of the reduction step is then $f - bg$ and g is called the *reducer*. When g reduces t we also say for convenience that bg reduces f. That way b is introduced implicitly instead of having to repeat the equation $\operatorname{lt}(bg) = t$.

The result of a reduction of $f \in \mathscr{R}$ is an element $h \in \mathscr{R}$ that has been calculated from f by a sequence of reduction steps. Thus, reductions can always be assumed to be done w.r.t. some finite subset $G \subset \mathscr{R}$.

Definition 2.7 *(Gröbner basis, Buchberger (1965, 2006)).* Let $I = \langle f_1, \ldots, f_m \rangle$ be an ideal in $\mathscr{R}$, $\leq$ a monomial order on $\mathscr{R}$.

(a) A finite subset G of $\mathscr{R}$ is a *Gröbner basis up to degree d* for I (w.r.t. $\leq$) if $G \subset I$ and for all $f \in I$ with $\deg(f) \leq d$ f reduces to zero w.r.t. G. G is a *Gröbner basis* for I (w.r.t. $\leq$) if G is a Gröbner basis in all degrees (w.r.t. $\leq$).
(b) A Gröbner basis G is called *minimal* if $0 \notin G$ and $\operatorname{lt}(f) \nmid \operatorname{lt}(g)$ for any two elements $f \neq g \in G$.
(c) A Gröbner basis G is called *reduced* if G is minimal, for all $f \in G$ $\operatorname{lc}(f) = 1$, and no monomial of $f - \operatorname{lt}(f)$ is contained in $L(G)$.

Clearly, for a given ideal I its reduced Gröbner basis w.r.t. $\leq$ is unique.

Example 2.3. Note that the monomial order chosen on $\mathscr{R}$ is crucial for a Gröbner basis: Assume $I = \langle f, g \rangle \subset \mathbb{Q}[x, y]$ with $f = y^2 + x$ and $g = x^2 + y$. Considering the degree reverse lexicographical order $\leq$ the monomials in f and g are already correctly sorted. The Gröbner basis for I w.r.t. the degree reverse lexicographical order is then trivially $G = \{f, g\}$. On the other hand, if we consider the lexicographical order on $\mathscr{R}$, $\operatorname{lt}(f) = x$ and no longer y^2. Thus the Gröbner basis for I in this setting is $G = \{x + y^2, y^4 + y\}$ since $\operatorname{lt}(f) \mid \operatorname{lt}(g)$: One can reduce $g - xf = -xy^2 + y$, which can then be further reduced with $y^2 f$ resulting in $y^4 + y$.

Analogously to Definition 2.7 one can define Gröbner bases with the notion of standard representations:

Definition 2.8. Let $f \in \mathscr{R}$ and $G \subset \mathscr{R}$ finite. A representation $f = \sum_{i=1}^{k} m_i g_i$ with monomials $m_i \neq 0$, $g_i \in G$ pairwise different is called a *standard representation* if

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

One can show that if for any $f \in \langle G \rangle$ with $f \neq 0$ f has a standard representation w.r.t. G and $\leq$ then G is a Gröbner basis for $\langle G \rangle$. Moreover, note that the existence of a standard representation does not imply reducibility to zero, see, for example, Exercise 5.63 in Becker et al. (1993).

In Buchberger (1965, 2006), Buchberger gave an key structural theorem on Gröbner basis theory. Using the notion of S-polynomials an algorithm naturally follows:

Definition 2.9 *(S-polynomial, Buchberger (1965, 2006)).* Let $f \neq 0, g \neq 0 \in \mathscr{R}$ and let $\lambda = \operatorname{lcm}(\operatorname{lt}(f), \operatorname{lt}(g))$ be the monic least common multiple of $\operatorname{lt}(f)$ and $\operatorname{lt}(g)$. The *S-polynomial* between f and g is given by

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

Theorem 2.1 *(Buchberger's criterion, Buchberger (1965, 2006)). Let $I = \langle f_1, \ldots, f_m \rangle$ be an ideal in $\mathscr{R}$. A finite subset G of $\mathscr{R}$ is a* Gröbner basis *for I if $G \subset I$ and for all $f, g \in G$* spol(f, g) *reduces to zero w.r.t. G.*

Algorithm 1 Buchberger's Gröbner basis algorithm (Buchberger, 1965, 2006).

```
Require: Ideal $I = \langle f_1, \dots, f_m \rangle \subset \mathscr{R}$, monomial order $\leq$ on $\mathscr{R}$
Ensure: Gröbner basis $G$ for $I$
 1: $G \leftarrow \emptyset$
 2: $\mathscr{P} \leftarrow \{f_1, \dots, f_m\}$
 3: while $\mathscr{P} \neq \emptyset$ do
 4:     $f \leftarrow$ Choose one element from $\mathscr{P}$
 5:     $\mathscr{P} \leftarrow \mathscr{P} \setminus \{f\}$
 6:     $g \leftarrow$ result of reducing $f$ w.r.t. $G$
 7:     if $g \neq 0$ then
 8:         $\mathscr{P} \leftarrow \mathscr{P} \cup \{\mathrm{spol}(g, h) \mid h \in G\}$
 9:         $G \leftarrow G \cup \{g\}$
10: return $G$
```

Example 2.4. Taking $f, g \in \mathbb{Q}[x, y]$ from Example 2.3 and the lexicographical order $\leq$ Algorithm 1 first adds f, g to $\mathscr{P}$ in Line 2. Let us assume to always choose the element of smallest lead term w.r.t. $\leq$ in Line 4. Thus we first take f from $\mathscr{P}$ and cannot reduce it further. We also cannot generate any new S-polynomial in Line 8 since $G = \emptyset$ at the moment. Afterwards f is added to G. In the next round g is chosen from $\mathscr{P}$. This time we can reduce g with xf, and further with $y^2 f$ (see Example 2.3). We end up with a new element, say, $h = y^4 + y$. Now we generate $\mathrm{spol}(h, f)$ and add it to $\mathscr{P}$. h is then added to G. In the next round we can only choose $\mathrm{spol}(h, f)$ from $\mathscr{P}$: $\mathrm{spol}(h, f) = xh - y^4 f = xy - y^6$. Further reducing with yf gives $-y^6 - y^3$. This can then be reduced with $y^2 h$ and we end up with zero. We do not generate a new S-polynomial, we also do not add any new element to G and $\mathscr{P} = \emptyset$. The algorithm terminates and returns a correct Gröbner basis $G = \{x + y^2, y^4 + y\}$.

Note that we could have seen $\mathrm{spol}(h, f)$ to reduce to zero w.r.t. G in Example 2.4 before even starting to reduce it. This is due to Buchberger's Product criterion, which we consider in more detail in Section 12. For more details on Buchberger's influential work on Gröbner bases we also refer to Buchberger (1970, 1984, 1987).

3. Matrix F5

Algebraic systems are solved by computing a Gröbner basis for a corresponding ideal (Buchberger, 1965, 1985). The link between solving such systems and linear algebra is already very old, see, for example, Macaulay (1916), Lazard (1983). In 1999 Faugère introduced the **F4** algorithm (Faugère, 1999). A simplified description of this algorithm using signature-based criteria is **MatrixF5** which we present here. The important fact is that polynomial reduction coincides with Gaussian elimination in **MatrixF5** and thus the process of computing the basis can be illustrated easily.

Before we approach signature-based Gröbner basis algorithms theoretically let us look at a small Gröbner basis computation. We start with a slightly simplified version of the **F5** algorithm, the **MatrixF5**. With this introduction to the topic we are able to give an easy description of the main ideas behind the classification of signature-based algorithms which is discussed in detail later on. In order to keep this section plain and easy we keep signature-based details at a minimum and focus on presenting their usefulness discarding useless elements from the computation.

Descriptions of **MatrixF5** can be also found, for example, in Bardet (2004a), Faugère and Rahmany (2009). It is first publicly mentioned in Faugère and Joux (2003) and known for breaking challenge 1 of the hidden field equations (HFE) crypto system.

Let $I = \langle f_1, \dots, f_m \rangle \subset \mathscr{R}$ be the *homogeneous* input ideal. We want to compute a Gröbner basis for I w.r.t. a given monomial order $<$. The idea is to incrementally construct *Macaulay matrices* M_d which are generalizations of the Sylvester matrix for finitely many (> 2 possible), multivariate polynomials. In the above setting the rows of M_d represent the polynomials $t_{j,k} f_k$ where $t_{j,k}$ are monomials in $\mathscr{R}$ such that $\deg(t_{j,k} f_k) \leq d$ for all $1 \leq k \leq m$. The columns of M_d are labeled by all possible terms x^v such that $\deg(x^v) \leq d$. Moreover, the columns are sorted, from left to right, by decreasing monomial order $<$. Thus a row of M_d labeled by $t_{j,k} f_k = \sum_{x^v \in \mathscr{M}, \deg(x^v) \leq d} \kappa_v x^v$ has in column x^v entry $\kappa_v \in \mathscr{K}$. Note that by the this representation of $t_{j,k} f_k$ $\kappa_v = 0$ is possible. Once M_d is generated, the row

$$M_2 = \begin{array}{c|cccccc} & x^2 & xy & y^2 & xz & yz & z^2 \\ e_3 & 3 & 4 & 2 & 0 & 0 & 0 \\ e_2 & 2 & 3 & 4 & 0 & 0 & 3 \\ e_1 & 0 & 0 & 1 & 0 & 4 & 0 \end{array} \longrightarrow \begin{array}{c|cccccc} & x^2 & xy & y^2 & xz & yz & z^2 \\ e_3 & 3 & 4 & 2 & 0 & 0 & 0 \\ \boxed{e_2+e_1} & 2 & 3 & 0 & 0 & 4 & 3 \\ e_1 & 0 & 0 & 1 & 0 & 4 & 0 \end{array}$$

$$\longrightarrow \begin{array}{c|cccccc} & x^2 & xy & y^2 & xz & yz & z^2 \\ \boxed{e_3+3e_1} & 3 & 4 & 0 & 0 & 2 & 0 \\ e_2+e_1 & 2 & 3 & 0 & 0 & 4 & 3 \\ e_1 & 0 & 0 & 1 & 0 & 4 & 0 \end{array} \longrightarrow \begin{array}{c|cccccc} & x^2 & xy & y^2 & xz & yz & z^2 \\ \boxed{e_3+e_2+4e_1} & 0 & 2 & 0 & 0 & 1 & 3 \\ e_2+e_1 & 2 & 3 & 0 & 0 & 4 & 3 \\ e_1 & 0 & 0 & 1 & 0 & 4 & 0 \end{array}$$

$$\longrightarrow \begin{array}{c|cccccc} & x^2 & xy & y^2 & xz & yz & z^2 \\ e_3+e_2+4e_1 & 0 & 2 & 0 & 0 & 1 & 3 \\ \boxed{e_3+2e_2+5e_1} & 2 & 0 & 0 & 0 & 0 & 1 \\ e_1 & 0 & 0 & 1 & 0 & 4 & 0 \end{array}$$

Fig. 2. Computing the row echelon form of M_2.

echelon form N_d of M_d is computed. The rows of N_d now correspond to polynomials in $\mathscr{R}$ that generate a Gröbner basis up to degree d for I. So, in contrast to Gröbner basis algorithms in the vein of Buchberger's description, **MatrixF5** needs another parameter, a degree bound D up to which the computations are carried out. We introduce the variant of this algorithm using signature-based criteria to improve computations by an example.

Consider the three homogeneous input polynomials $f_1 = y^2 + 4yz$, $f_2 = 2x^2 + 3xy + 4y^2 + 3z^2$, and $f_3 = 3x^2 + 4xy + 2y^2$ in $\mathscr{R} = \mathbb{F}_5[x, y, z]$ where $<$ denotes the graded reverse lexicographical monomial order. By the above description it is clear that the labels $t_{j,k} f_k$ of the rows coincide with the corresponding signatures $t_{j,k}\boldsymbol{e}_k$. We want to use these signatures to label the rows of the Macaulay matrices built in the following. Thus we need to extend $<$ on $\mathscr{R}^3$, say we use $<_{\text{pot}}$. Let us assume we want to compute a Gröbner basis up to degree $D = 4$.

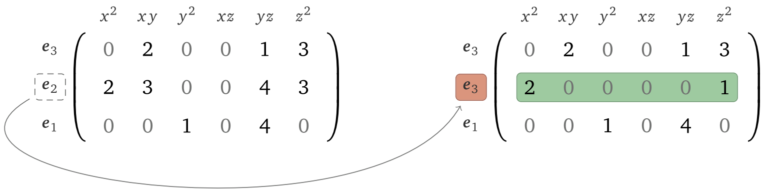
The main idea of using Macaulay matrices is now to calculate all possible elements in I for a given degree d. In Buchberger's attempt (Theorem 2.1) one considers S-polynomials of degree d and has to find reducers of these. Here we do not need to search for such elements, all possible reducers are already in M_d. So we can focus on the main question: How do signature-based criteria work to improve Gröbner basis computations?

Let us start with the lowest possible degree, $d = 2$. Building the Macaulay matrix M_2 in Fig. 2 we label the rows by the corresponding signatures. Throughout the steps of reducing M_2 we keep track in the label of the rows what computational steps have been done.

One of the reduction steps differs, the last step: Looking at the label of the second row after reducing it with the first row we see that there is a change:

$$\boldsymbol{e}_2 + \boldsymbol{e}_1 \quad \Longrightarrow \quad \boldsymbol{e}_3 + 2\boldsymbol{e}_2 + 5\boldsymbol{e}_1.$$

Since the labels change also in the other reduction steps, the question is, what is special in this step? Looking at the lead term of the module element we see the difference: Before the reduction the label of the row has a lead term of $\boldsymbol{e}_2$ w.r.t. $<_{\text{pot}}$, afterwards it is $\boldsymbol{e}_3$. In none of the other reduction steps above the lead term changed. And that is the general idea of the signature: We want to easily keep track of where the new rows are coming from. Storing the complete module representation as done above, the overhead of computing a Gröbner basis is too big (see also Möller et al., 1992). Thus,



$$\begin{array}{c|cccccc} & x^2 & xy & y^2 & xz & yz & z^2 \\ e_3 & 0 & 2 & 0 & 0 & 1 & 3 \\ e_2 & 2 & 3 & 0 & 0 & 4 & 3 \\ e_1 & 0 & 0 & 1 & 0 & 4 & 0 \end{array} \qquad \begin{array}{c|cccccc} & x^2 & xy & y^2 & xz & yz & z^2 \\ e_3 & 0 & 2 & 0 & 0 & 1 & 3 \\ e_3 & 2 & 0 & 0 & 0 & 0 & 1 \\ e_1 & 0 & 0 & 1 & 0 & 4 & 0 \end{array}$$

Fig. 3. Change of signature due to a reduction step.

instead of keeping the full module representation, we only store the lead term of it, the signature. Applied to our example above the last step would lead to the situation illustrated in Fig. 3.

In other words, we would loose the connection between the second row and $\boldsymbol{e}_2$ resp. f_2. As we see in following, to remember this connection is crucial for the strength of signature-based criteria to remove useless computations.

We agree to not do any such reduction. In terms of the Macaulay matrix this means that

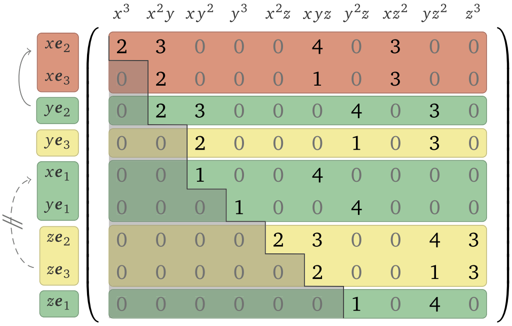
1. rows are sorted from top to bottom by decreasing signatures, and
2. the row we reduce with must be below the row to be reduced.

Thus for our purpose to keep the signatures, the row echelon form of M_2 received by restricting reductions is

$$N_2 = \begin{array}{c} \\ \boldsymbol{e}_3 \\ \boldsymbol{e}_2 \\ \boldsymbol{e}_1 \end{array} \begin{array}{c} \begin{array}{cccccc} x^2 & xy & y^2 & xz & yz & z^2 \end{array} \\ \begin{pmatrix} 0 & 2 & 0 & 0 & 1 & 3 \\ 2 & 3 & 0 & 0 & 4 & 3 \\ 0 & 0 & 1 & 0 & 4 & 0 \end{pmatrix} \end{array}$$

After computing the row echelon form N_2 of the Macaulay matrix M_2 we get two new polynomials, namely $f_4 = 2xy + yz + 3z^2$ and $f_5 = 2x^2 + 3xy + 4yz + 3z^2$, corresponding to the first and the second row of N_2. f_3 and f_4 have the same signature $\boldsymbol{e}_3$, thus we can say that there is a connection between them. The same holds for f_2 and f_5.

At this point we have not done any reduction in M_3 but just used the information stored in the signatures. Let us rearrange the rows of M_3 to see how near we are already to a row echelon form:



$$\begin{array}{c|cccccccccc} & x^3 & x^2y & xy^2 & y^3 & x^2z & xyz & y^2z & xz^2 & yz^2 & z^3 \\ xe_2 & 2 & 3 & 0 & 0 & 0 & 4 & 0 & 3 & 0 & 0 \\ xe_3 & 0 & 2 & 0 & 0 & 0 & 1 & 0 & 3 & 0 & 0 \\ ye_2 & 0 & 2 & 3 & 0 & 0 & 0 & 4 & 0 & 3 & 0 \\ ye_3 & 0 & 0 & 2 & 0 & 0 & 0 & 1 & 0 & 3 & 0 \\ xe_1 & 0 & 0 & 1 & 0 & 0 & 4 & 0 & 0 & 0 & 0 \\ ye_1 & 0 & 0 & 0 & 1 & 0 & 0 & 4 & 0 & 0 & 0 \\ ze_2 & 0 & 0 & 0 & 0 & 2 & 3 & 0 & 0 & 4 & 3 \\ ze_3 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 1 & 3 \\ ze_1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 4 & 0 \end{array}$$

Next we can go on with degree 3. Generating M_3 we get all multiples xf_i, yf_i and zf_i for $1 \leq i \leq 3$ as it is shown in Fig. 4. Looking at M_3 more closely we see some relation to M_2. The three steps

$$M_3 = \begin{array}{c|cccccccccc}
 & x^3 & x^2y & xy^2 & y^3 & x^2z & xyz & y^2z & xz^2 & yz^2 & z^3 \\
\hline
x\mathbf{e}_3 & 3 & 4 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
y\mathbf{e}_3 & 0 & 3 & 4 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\
z\mathbf{e}_3 & 0 & 0 & 0 & 0 & 3 & 4 & 2 & 0 & 0 & 0 \\
x\mathbf{e}_2 & 2 & 3 & \boxed{4} & 0 & 0 & 0 & 0 & 3 & 0 & 0 \\
y\mathbf{e}_2 & 0 & 2 & 3 & \boxed{4} & 0 & 0 & 0 & 0 & 3 & 0 \\
z\mathbf{e}_2 & 0 & 0 & 0 & 0 & 2 & 3 & \boxed{4} & 0 & 0 & 3 \\
x\mathbf{e}_1 & 0 & 0 & \boxed{1} & 0 & 0 & 4 & 0 & 0 & 0 & 0 \\
y\mathbf{e}_1 & 0 & 0 & 0 & \boxed{1} & 0 & 0 & 4 & 0 & 0 & 0 \\
z\mathbf{e}_1 & 0 & 0 & 0 & 0 & 0 & 0 & \boxed{1} & 0 & 4 & 0
\end{array}$$

Fig. 4. Initial Macaulay matrix M_3.

highlighted correspond to reduction steps that have already occurred in degree 2:

$$x\mathbf{e}_2 - x\mathbf{e}_1 = x(\mathbf{e}_2 - \mathbf{e}_1)$$
$$y\mathbf{e}_2 - y\mathbf{e}_1 = y(\mathbf{e}_2 - \mathbf{e}_1)$$
$$z\mathbf{e}_2 - z\mathbf{e}_1 = z(\mathbf{e}_2 - \mathbf{e}_1).$$

Since we have done these reductions already it makes sense to not redo them again, but use the information from M_2. We know that f_5 comes from f_2, both share the same signature. So we just *rewrite* xf_2, yf_2, zf_2 by xf_5, yf_5, zf_5 in M_3, respectively. The very same holds for f_3 and f_4. Fig. 5 illustrates this process.

Only rewriting f_2 and f_3 with "better" elements lead to this matrix in near row echelon form. Again, note that not all elements in the above picture are allowed to reduce freely: The rows highlighted in green can reduce any other row above them. So, for example, the row with signature $y\mathbf{e}_2$ can reduce the rows with signatures $x\mathbf{e}_3$ and $x\mathbf{e}_2$, respectively. In none of these reductions the signature of any row changes. On the other hand, the row labeled by signature $z\mathbf{e}_3$ is not allowed to reduce the row labeled by $x\mathbf{e}_1$. Otherwise the signature might change. Nevertheless, this row is still allowed to reduce the one labeled with $x\mathbf{e}_3$. Thus we highlighted this row in yellow to illustrate this restriction. The row labeled with $x\mathbf{e}_3$, and highlighted in red, is not allowed to reduce any other row. $x\mathbf{e}_3$ is the highest signature in degree 3 w.r.t. $<_{\text{pot}}$, thus any reduction of another row would lead to a change in signatures.

Executing all not signature changing reduction steps we end up with a Gröbner basis up to degree 3 represented by the row echelon form

$$N_3 = \begin{array}{c|cccccccccc}
 & x^3 & x^2y & xy^2 & y^3 & x^2z & xyz & y^2z & xz^2 & yz^2 & z^3 \\
\hline
x\mathbf{e}_2 & 2 & 0 & 0 & 0 & 0 & 4 & 0 & 3 & 3 & 3 \\
y\mathbf{e}_2 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 3 \\
x\mathbf{e}_1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 3 & 4 \\
y\mathbf{e}_1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 4 & 0 \\
z\mathbf{e}_2 & 0 & 0 & 0 & 0 & 2 & 3 & 0 & 0 & 4 & 3 \\
z\mathbf{e}_3 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 1 & 3 \\
z\mathbf{e}_1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 4 & 0 \\
x\mathbf{e}_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 2 \\
y\mathbf{e}_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 2
\end{array}$$

Next we are computing a Gröbner basis for $\langle f_1, f_2, f_3 \rangle$ up to degree 4. Again we generate the matrix M_4 building all combinations of monomials of degree 2 and f_i for $1 \leq i \leq 3$. This time we note that M_4 consists of $\binom{3}{2} \cdot 3 = 18$ rows and $\binom{3}{4} = 15$ columns. This means that when we are reducing M_4 we might end up with rows that reduced to zero (or rows that are multiples of others

$$M_3 = \begin{array}{c|cccccccccc}
 & x^3 & x^2y & xy^2 & y^3 & x^2z & xyz & y^2z & xz^2 & yz^2 & z^3 \\
\hline
xe_3 & 3 & 4 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
ye_3 & 0 & 3 & 4 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\
ze_3 & 0 & 0 & 0 & 0 & 3 & 4 & 2 & 0 & 0 & 0 \\
xe_2 & 2 & 3 & 0 & 0 & 0 & 4 & 0 & 3 & 0 & 0 \\
ye_2 & 0 & 2 & 3 & 0 & 0 & 0 & 4 & 0 & 3 & 0 \\
ze_2 & 0 & 0 & 0 & 0 & 2 & 3 & 0 & 0 & 4 & 3 \\
xe_1 & 0 & 0 & 1 & 0 & 0 & 4 & 0 & 0 & 0 & 0 \\
ye_1 & 0 & 0 & 0 & 1 & 0 & 0 & 4 & 0 & 0 & 0 \\
ze_1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 4 & 0
\end{array}$$

$$\downarrow$$

$$\begin{array}{c|cccccccccc}
 & x^3 & x^2y & xy^2 & y^3 & x^2z & xyz & y^2z & xz^2 & yz^2 & z^3 \\
\hline
xe_3 & 0 & 2 & 0 & 0 & 0 & 1 & 0 & 3 & 0 & 0 \\
ye_3 & 0 & 0 & 2 & 0 & 0 & 0 & 1 & 0 & 3 & 0 \\
ze_3 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 1 & 3 \\
xe_2 & 2 & 3 & 0 & 0 & 0 & 4 & 0 & 3 & 0 & 0 \\
ye_2 & 0 & 2 & 3 & 0 & 0 & 0 & 4 & 0 & 3 & 0 \\
ze_2 & 0 & 0 & 0 & 0 & 2 & 3 & 0 & 0 & 4 & 3 \\
xe_1 & 0 & 0 & 1 & 0 & 0 & 4 & 0 & 0 & 0 & 0 \\
ye_1 & 0 & 0 & 0 & 1 & 0 & 0 & 4 & 0 & 0 & 0 \\
ze_1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 4 & 0
\end{array}$$

Fig. 5. Rewriting rows: $f_2 \longrightarrow f_5$ (top), $f_3 \longrightarrow f_4$ (bottom).

due to the restricted reduction process). Nevertheless, these rows correspond to useless steps during a Gröbner basis algorithm. So how can we find out which to remove?

A polynomial reduction to zero corresponds to a syzygy in $\mathscr{R}^3$. There are principal (or trivial syzygies) we know already without any previous computations: $f_1\boldsymbol{e}_2 - f_2\boldsymbol{e}_1$, $f_1\boldsymbol{e}_3 - f_3\boldsymbol{e}_1$ and $f_2\boldsymbol{e}_3 - f_3\boldsymbol{e}_2$. Let us look at the signatures of these syzygies w.r.t. $<_{\mathrm{pot}}$:

$$\mathfrak{s}(f_1\boldsymbol{e}_2 - f_2\boldsymbol{e}_1) = \mathrm{lt}(f_1)\,\boldsymbol{e}_2 = y^2\boldsymbol{e}_2$$

$$\mathfrak{s}(f_1\boldsymbol{e}_3 - f_3\boldsymbol{e}_1) = \mathrm{lt}(f_1)\,\boldsymbol{e}_3 = y^2\boldsymbol{e}_3$$

$$\mathfrak{s}(f_2\boldsymbol{e}_3 - f_3\boldsymbol{e}_2) = \mathrm{lt}(f_2)\,\boldsymbol{e}_3 = x^2\boldsymbol{e}_3.$$

We have seen in the degree 3 case that we can rewrite elements with a given signature T by other elements that have the same signature. Thus for $T \in \{y^2\boldsymbol{e}_2, y^2\boldsymbol{e}_3, x^2\boldsymbol{e}_3\}$ we can just use the above syzygies resp. corresponding trivial relations in $\mathscr{R}$. So the following 3 elements are exchanged, respectively, in M_4

$$y^2 f_2 \longrightarrow f_1 f_2 - f_2 f_1,$$

$$y^2 f_3 \longrightarrow f_1 f_3 - f_3 f_1,$$

$$x^2 f_3 \longrightarrow f_2 f_3 - f_3 f_2.$$

This means that we would add rows that have only zero entries for $y^2\boldsymbol{e}_2$, $y^2\boldsymbol{e}_3$ and $x^2\boldsymbol{e}_3$. Those rows do not play any role during the reduction process of M_4, so we can remove them directly from the matrix. In the end we receive a matrix M_4 of dimensions 15 × 15, thus we know that when reducing M_4 to its row reduced echelon form N_4, all rows are useful. Clearly, as we have done for M_3, we try to rewrite the 15 rows remaining in M_4 that correspond to elements $x^j y^k z^\ell f_i$ with $1 \le i \le 3$ and $j + k + \ell = 2$ with elements from N_2 and N_3 in order to not repeat calculations already done at a lower degree. Computing the row echelon form of M_4 we then receive a Gröbner basis for $\langle f_1, f_2, f_3\rangle$ up to degree 4.

Let us try to summarize the main ideas behind using signatures when computing Gröbner bases:

- Try to rewrite data and reuse already done calculations.
- Keep track of this rewriting by not changing the signatures during the reduction process.
- If the rewritten data is trivial resp. corresponds to a syzygy (relations that are already known) then discard this data.

Remark 3.1. Note that building Macaulay matrices as done in **MatrixF5** is useful and efficient only if the corresponding polynomial system is dense. Otherwise it makes more sense to combine Buchberger's S-polynomial attempt with linear algebra. That means, one first searches for all S-polynomials in a given degree d and *all* needed reducers and generates a corresponding matrix afterwards. This is the main idea behind Faugère's **F4** algorithm (Faugère, 1999). In Section 13 we present an efficient way of combining signature-based criteria for discarding useless data with **F4**.

With this in mind we are able to give a more theoretical introduction to signature-based Gröbner basis computations in the vein of Buchberger's algorithm.

4. Gröbner bases with signatures

In this section we give an introduction to signature-based Gröbner basis algorithms from a mathematical point of view. Thus the content is dedicated to a complete and correct description of the algorithms' underlying ideas. Motivated by the specialized row echelon forms we presented in Section 3 the notion of a polynomial reduction process taking care of the signatures is introduced. Connections and differences to classic polynomial Gröbner basis theory are explained in detail.

Readers interested in optimized signature-based algorithms only might skip most of this section, but should at least consider notations introduced in Section 2 and here as we agree on those throughout the paper. We extend the notation introduced in Eder and Roune (2013).

4.1. Signature reduction

In order to keep track of the signatures when reducing corresponding polynomial data we need to adjust Definition 2.6. Sloppy speaking we get a classic polynomial reduction together with a further condition.

Definition 4.1. Let $\alpha \in \mathscr{R}^m$ and let t be a term of $\overline{\alpha}$. Then we can $\mathfrak{s}$-*reduce* t by $\beta \in \mathscr{R}^m$ if

(a) there exists a monomial b such that $\operatorname{lt}\left(\overline{b\beta}\right) = t$ and
(b) $\mathfrak{s}(b\beta) \leq \mathfrak{s}(\alpha)$.

The outcome of the $\mathfrak{s}$-reduction step is then $\alpha - b\beta$ and β is called the $\mathfrak{s}$-*reducer*. When β $\mathfrak{s}$-reduces t we also say for convenience that $b\beta$ $\mathfrak{s}$-reduces α. That way b is introduced implicitly instead of having to repeat the equation $\operatorname{lt}\left(\overline{b\beta}\right) = t$.

Remark 4.1. Note that Condition (a) from Definition 4.1 defines a classic polynomial reduction step (see 2.6). It implies that $\operatorname{lt}\left(\overline{b\beta}\right) \leq \operatorname{lt}(\overline{\alpha})$. Moreover, Condition (b) lifts the above implication to $\mathscr{R}^m$ so that it involves signatures. Since we are interested in computing Gröbner Bases in $\mathscr{R}$ one can interpret an $\mathfrak{s}$-reduction of α by β as classic polynomial reduction of $\overline{\alpha}$ by $\overline{\beta}$ together with Condition (b). Thus an $\mathfrak{s}$-reduction represents a connection between data in $\mathscr{R}$ and corresponding data in $\mathscr{R}^m$ when a polynomial reduction takes place.

Just as for classic polynomial reduction, if $\operatorname{lt}\left(\overline{b\beta}\right) \simeq \operatorname{lt}\left(\overline{\alpha}\right)$ then the $\mathfrak{s}$-reduction step is a *top $\mathfrak{s}$-reduction step* and otherwise it is a *tail $\mathfrak{s}$-reduction step*. Analogously we define the distinction for signatures: If $\mathfrak{s}\left(b\beta\right) \simeq \mathfrak{s}\left(\alpha\right)$ then the reduction step is a *singular $\mathfrak{s}$-reduction step* and otherwise it is a *regular $\mathfrak{s}$-reduction step*.

The result of $\mathfrak{s}$-reduction of $\alpha \in \mathscr{R}^m$ is a $\gamma \in \mathscr{R}^m$ that has been calculated from α through a sequence of $\mathfrak{s}$-reduction steps such that γ cannot be further $\mathfrak{s}$-reduced. The reduction is a *tail $\mathfrak{s}$-reduction* if only tail $\mathfrak{s}$-reduction steps are allowed and it is a *top $\mathfrak{s}$-reduction* if only top $\mathfrak{s}$-reduction steps are allowed. The reduction is a *regular $\mathfrak{s}$-reduction* if only regular $\mathfrak{s}$-reduction steps are allowed. A module element $\alpha \in \mathscr{R}^m$ is *$\mathfrak{s}$-reducible* if it can be $\mathfrak{s}$-reduced.

If α $\mathfrak{s}$-reduces to γ and γ is a syzygy then we say that α *$\mathfrak{s}$-reduces to zero* even if $\gamma \neq 0$.

Example 4.1. Assume an ideal $I = \langle f_1, f_2, f_3\rangle \subset \mathscr{R} = \mathbb{Q}[x, y, z, t]$ with $f_1 = xyz - z^2t$, $f_2 = x^2y - y^3$ and $f_3 = y^3 - zt^2 - t^3$. Furthermore, $<$ denotes the graded reverse lexicographical monomial order which we extend to $<_{\text{pot}}$ on the set of monomials of $\mathscr{R}^3$. Clearly, we have $\alpha_i = \boldsymbol{e}_i$ with $\overline{\alpha_i} = f_i$ for $i \in \{1, 2, 3\}$. We start with $\mathscr{G} = \{\alpha_1, \alpha_2, \alpha_3\}$.

Looking at $z\alpha_2$ we can regular top $\mathfrak{s}$-reduce $\operatorname{lt}\left(\overline{z\alpha_2}\right)$ with $x\alpha_1$ since $\operatorname{lt}\left(\overline{x\alpha_1}\right) = \operatorname{lt}\left(\overline{z\alpha_2}\right)$ and $\mathfrak{s}\left(x\alpha_1\right) <_{\text{pot}} \mathfrak{s}\left(z\alpha_2\right)$. Call the resulting element $\alpha_4 = z\alpha_2 - x\alpha_1$. We can see that we cannot further $\mathfrak{s}$-reduce $\overline{\alpha_4} = -y^3z + xz^2t$: The only possible candidate is α_3 but $\mathfrak{s}\left(z\alpha_3\right) = z\boldsymbol{e}_3 >_{\text{pot}} z\boldsymbol{e}_2 = \mathfrak{s}\left(\alpha_4\right)$. Note that $\overline{\alpha_4} + z\overline{\alpha_3}$ would be a correct classical polynomial reduction step, but it contradicts Condition (b) of an $\mathfrak{s}$-reduction. On the other hand, adding α_4 to $\mathscr{G}$ we are able to regular top $\mathfrak{s}$-reduce $z\alpha_3$ w.r.t. $\mathscr{G}$, namely by α_4. We see that whereas from a pure polynomial point of view reducing $\overline{\alpha_4} + z\overline{\alpha_3}$ is the same as $z\overline{\alpha_3} + \overline{\alpha_4}$ taking the signatures into account destroys this equality. Only the second operation is a valid $\mathfrak{s}$-reduction.

Again, we can regular top $\mathfrak{s}$-reduce $x\alpha_4$ with $y^2\alpha_1$. This gives a new element $\alpha_5 = x\alpha_4 + y^2\alpha_1$ whereas $\overline{\alpha_5} = x^2z^2t - y^2z^2t$.

Looking at $x^2\overline{\alpha_4} = -x^2y^3z + x^3z^2t$ one can use $\operatorname{lt}\left(\overline{x\alpha_5}\right)$ to tail $\mathfrak{s}$-reduce. Note that this $\mathfrak{s}$-reduction is singular due to $\mathfrak{s}\left(x\alpha_5\right) = x^2z\boldsymbol{e}_2 = \mathfrak{s}\left(x^2\alpha_4\right)$. In other words, $x^2\alpha_4 - x\alpha_5 = (x^2z\boldsymbol{e}_2 - x^3\boldsymbol{e}_1) - (x^2z\boldsymbol{e}_2 - x^3\boldsymbol{e}_1 + xy^2\boldsymbol{e}_1) = -xy^2\boldsymbol{e}_1$. Thus we see that $x^2\alpha_4$ $\mathfrak{s}$-reduces to a syzygy $\gamma = x^2\alpha_4 - x\alpha_5 - x^2y\alpha_1$.

Remark 4.2.

(a) The implied condition $\operatorname{lt}\left(\overline{b\beta}\right) \leq \operatorname{lt}\left(\overline{\alpha}\right)$ is equivalent to $\operatorname{lt}\left(\overline{\alpha - b\beta}\right) \leq \operatorname{lt}\left(\overline{\alpha}\right)$, so during $\mathfrak{s}$-reduction it is not allowed to increase the lead term. For tail $\mathfrak{s}$-reduction we perform only those $\mathfrak{s}$-reduction steps that do not change the lead term at all. Analogously, the condition $\mathfrak{s}\left(b\beta\right) \leq \mathfrak{s}\left(\alpha\right)$ is equivalent to $\mathfrak{s}\left(\alpha - b\beta\right) \leq \mathfrak{s}\left(\alpha\right)$, so during $\mathfrak{s}$-reduction it is not allowed to increase the signature. For regular $\mathfrak{s}$-reduction, we perform only those $\mathfrak{s}$-reduction steps that do not change the signature at all.

(b) Note that by Lemma 15 in Eder and Perry (2011) the notion of "being singular top $\mathfrak{s}$-reducible" is equivalent to what is sometimes in the literature also called "sig-redundant".

Note that analogously to the classic polynomial reduction $\mathfrak{s}$-reduction is always with respect to a finite *basis* $\mathscr{G} \subseteq \mathscr{R}^m$. The $\mathfrak{s}$-reducers in $\mathfrak{s}$-reduction are chosen from the basis $\mathscr{G}$.

4.2. Signature Gröbner bases

Having defined a polynomial reduction process taking signatures into account we are now able to define signature Gröbner bases analogously to classic polynomial Gröbner bases.

Definition 4.2. Let $I = \langle f_1, \ldots, f_m\rangle$ be an ideal in $\mathscr{R}$. A finite subset $\mathscr{G}$ of $\mathscr{R}^m$ is a *signature Gröbner basis in signature T* for I if all $\alpha \in \mathscr{R}^m$ with $\mathfrak{s}\left(\alpha\right) = T$ $\mathfrak{s}$-reduce to zero w.r.t. $\mathscr{G}$. $\mathscr{G}$ is a *signature Gröbner basis up to signature T* for I if $\mathscr{G}$ is a signature Gröbner basis in all signatures S such that $S < T$. $\mathscr{G}$ is a *signature Gröbner basis* for I if it is a signature Gröbner basis for I in all signatures.

We want to emphasize that in the above definition of a signature Gröbner basis we have the full module element $\alpha \in \mathscr{R}^m$ stored in $\mathscr{G}$, not only its leading term $\mathfrak{s}(\alpha)$. Thus we can reconstruct the corresponding polynomial Gröbner basis in $\mathscr{R}$ in the following way:

Lemma 4.1. *Let $I = \langle f_1, \ldots, f_m \rangle$ be an ideal in $\mathscr{R}$. If $\mathscr{G}$ is a signature Gröbner basis for I then $\{\overline{\alpha} \mid \alpha \in \mathscr{G}\}$ is a Gröbner basis for I.*

Proof. For example, see Section 2.2 in Roune and Stillman (2012a). □

Note that looking at the computed basis from the module point of view is a concept that was first introduced by Gao, Volny, and Wang in the **GVW** algorithm in Gao et al. (2016). There a slightly different basis, called *strong Gröbner basis* in $\mathscr{R}^m \times \mathscr{R}$ is defined (see also Definition 11.1). It stores not only a (polynomial) Gröbner basis for the input ideal $I = \langle f_1, \ldots, f_m \rangle$, but also a Gröbner basis for $\operatorname{syz}(f_1, \ldots, f_m)$. In this survey we keep both bases separate, one is the signature Gröbner basis $\mathscr{G}$ defined above, the other one is denoted $\mathscr{H}$ (see next section for more on the syzygies). We show in Vocabulary 5 how both concepts, the one in Gao et al. (2016) and the one used here, can be easily translated to each other.

Convention 4.1. *In the following, when denoting $\mathscr{G} \subseteq \mathscr{R}^m$ "a signature Gröbner basis (up to signature T)" we always mean "a signature Gröbner basis (up to signature T) for $I = \langle f_1, \ldots, f_m \rangle$". We omit the explicit notion of the input ideal whenever it is clear from the context.*

As in the classic polynomial setting we want to give an algorithmic description of signature Gröbner bases. For this we introduce the notion of S-pairs, similar to Definition 2.9.

Definition 4.3.

(a) Let $\alpha, \beta \in \mathscr{R}^m$ such that $\overline{\alpha} \neq 0$, $\overline{\beta} \neq 0$ and let $\lambda = \operatorname{lcm}\left(\operatorname{lt}(\overline{\alpha}), \operatorname{lt}\left(\overline{\beta}\right)\right)$ be the monic least common multiple of $\operatorname{lt}(\overline{\alpha})$ and $\operatorname{lt}\left(\overline{\beta}\right)$. The *S-pair* between α and β is given by

$$\operatorname{spair}(\alpha, \beta) := \frac{\lambda}{\operatorname{lt}(\overline{\alpha})}\alpha - \frac{\lambda}{\operatorname{lt}\left(\overline{\beta}\right)}\beta.$$

(b) $\operatorname{spair}(\alpha, \beta)$ is *singular* if $\mathfrak{s}\left(\frac{\lambda}{\operatorname{lt}(\overline{\alpha})}\alpha\right) \simeq \mathfrak{s}\left(\frac{\lambda}{\operatorname{lt}(\overline{\beta})}\beta\right)$. Otherwise it is *regular*.

Note that $\operatorname{spair}(\alpha, \beta) \in \mathscr{R}^m$ and $\overline{\operatorname{spair}(\alpha, \beta)} = \operatorname{spol}\left(\overline{\alpha}, \overline{\beta}\right)$.

Theorem 4.1. *Let T be a module monomial of $\mathscr{R}^m$ and let $\mathscr{G} \subseteq \mathscr{R}^m$ be a finite basis. Assume that all regular S-pairs $\operatorname{spair}(\alpha, \beta)$ with $\alpha, \beta \in \mathscr{G}$ and $\mathfrak{s}(\operatorname{spair}(\alpha, \beta)) < T$ $\mathfrak{s}$-reduce to zero and all $\boldsymbol{e}_i$ with $\boldsymbol{e}_i < T$ $\mathfrak{s}$-reduce to zero. Then $\mathscr{G}$ is a signature Gröbner basis up to signature T.*

Proof. For example, see Theorem 2 in Roune and Stillman (2012b). □

Note the similarity of Theorem 4.1 and Buchberger's Criterion for Gröbner bases (Theorem 2.1).

The outcome of classic polynomial reduction depends on the choice of reducer, so the choice of reducer can change what the intermediate bases are in the classic Buchberger algorithm. Lemma 4.2 implies that all S-pairs with the same signature yield the same regular $\mathfrak{s}$-reduced result as long as we process S-pairs in order of increasing signature.

Lemma 4.2. *Let $\alpha, \beta \in \mathscr{R}^m$ and let $\mathscr{G}$ be a signature Gröbner basis up to signature $\mathfrak{s}(\alpha) = \mathfrak{s}(\beta)$. If α and β are both regular top $\mathfrak{s}$-reduced then $\operatorname{lt}(\overline{\alpha}) = \operatorname{lt}\left(\overline{\beta}\right)$ or $\overline{\alpha} = \overline{\beta} = 0$. Moreover, if α and β are both regular $\mathfrak{s}$-reduced then $\overline{\alpha} = \overline{\beta}$.*

Proof. For example, see Lemma 3 in Roune and Stillman (2012b). □

Let us simplify our notations a bit using facts from the previous statements.

Notation 4.1.

(a) Due to Lemma 4.2 we assume in the following that $\mathscr{G}$ always denotes a finite subset of $\mathscr{R}^m$ with the property that for $\alpha, \beta \in \mathscr{G}$ with $\mathfrak{s}(\alpha) \simeq \mathfrak{s}(\beta)$ it follows that $\alpha = \beta$.
(b) Theorem 4.1 suggests to consider only regular S-pairs for the computation of signature Gröbner bases. Thus in the following “S-pair” always refers to “regular S-pair”.

Definition 4.4. A signature Gröbner basis is *minimal* if no basis element top $\mathfrak{s}$-reduces any other basis element.

Lemma 4.3 implies that the minimal signature Gröbner basis for an ideal $I \subset \mathscr{R}$ is unique and is contained in all signature Gröbner bases for I up to sig-lead pairs.

Lemma 4.3. *Let A be a minimal signature Gröbner basis and let B be a signature Gröbner basis for $\langle f_1, \ldots, f_m \rangle$. Then it holds for all $\alpha \in A$ that there exists a non-zero scalar $\kappa \in \mathscr{K}$ and a $\beta \in B$ such that $\mathfrak{s}(\alpha) = \kappa \mathfrak{s}(\beta)$ and $\operatorname{lt}(\overline{\alpha}) = \kappa \operatorname{lt}\left(\overline{\beta}\right)$.*

Proof. This is an easy corollary of Lemma 4.2. □

Let us close this introduction to signature Gröbner bases with a note on the precursors. The main idea of signatures is to keep track of syzygies during the computation of the Gröbner basis. This is idea is shared with the following other approaches:

Remark 1. In 1986, Gebauer and Möller introduced the notion of *staggered linear bases* (Gebauer and Möller, 1986). These can be seen as a first predecessor of the idea behind using signatures to track redundant reductions. In 1993, Mora presented a revised version of Gebauer and Möller's initial algorithm to compute such bases (Mora, 2005). Dellaca further revised this version in Dellaca (2009) fixing problems with termination and correctness. The main idea of staggered linear bases is to detect more useless reductions as the well-known Gebauer–Möller installation (Gebauer and Möller, 1988) (see also Section 12). In 1992, Möller, Mora and Traverso published an algorithm to compute Gröbner bases keeping track of all the syzygies (Möller et al., 1992). Clearly, keeping track of the operations not only in the Gröbner basis but also in the syzygy module adds an overhead to the computation. The main innovation in **F5** and all the other signature-based algorithms presented here is not to take the full module representations into account, but only the leading terms, the signatures. Thus the algorithms are no longer able to keep track of the full syzygy module, so it cannot interreduce the syzygies anymore (as Möller, Mora and Traverso's algorithm is doing), but the overhead of taking care only of a single module term instead is minimal. So signature-based algorithms are able to efficiently use information of known syzygies to detect useless steps in advance. For more details we refer to Section 6. For a more detailed discussion on the connection between these different attempts to exploit knowledge of known syzygies we also refer to Section 3 in Eder (2012).

5. Generic signature Gröbner basis computation

In the following we present a generic signature-based Gröbner basis algorithm **genSB** (Algorithm 2) mostly using notations and concepts already introduced in Eder and Roune (2013). This algorithm works the same way as the classic Gröbner basis algorithm presented by Buchberger (1965). The main difference is that in **genSB** the computations are lifted from $\mathscr{R}$ to $\mathscr{R}^m$ in the way presented in sections 4.1 and 4.2.

genSB should be understood as a generic description which does not aim on performance. We see in Section 7 how we can vary **genSB** to receive a template that can be used as a common basis from which all known efficient signature-based Gröbner basis algorithms can be derived from. Since **genSB** itself is a variant of Buchberger's algorithm (see Algorithm 1), we point out the main differences between those two in the following.

The classic Buchberger algorithm proceeds by reducing S-polynomials. If an S-polynomial reduces to a polynomial $h \in \mathscr{R}, h \neq 0$ then h is added to the basis so that the S-polynomial now reduces to zero by this larger basis. The classic Buchberger algorithm terminates once all S-polynomials between elements of the basis reduce to zero.

genSB does the very same with S-pairs using $\mathfrak{s}$-reductions. Based on Theorem 4.1, once all S-pairs $\mathfrak{s}$-reduced to zero w.r.t. $\mathscr{G}$, **genSB** terminates with a signature Gröbner basis.

Algorithm 2 Generic signature-based Gröbner basis algorithm **genSB**.

```
Require: Ideal $I = \langle f_1, \ldots, f_m \rangle \subset \mathscr{R}$, monomial order $\leq$ on $\mathscr{R}$ and a compatible extension on $\mathscr{R}^m$, total order $\preceq$ on the pairset $\mathscr{P}$ of S-pairs
Ensure: Signature Gröbner basis $\mathscr{G}$ for $I$, Gröbner basis $\mathscr{H}$ for $\operatorname{syz}(f_1, \ldots, f_m)$
 1: $\mathscr{G} \leftarrow \emptyset$, $\mathscr{H} \leftarrow \emptyset$
 2: $\mathscr{P} \leftarrow \{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_m\}$
 3: while $\mathscr{P} \neq \emptyset$ do
 4:     $\beta \leftarrow \min_{\preceq} \mathscr{P}$
 5:     $\mathscr{P} \leftarrow \mathscr{P} \setminus \{\beta\}$
 6:     $\gamma \leftarrow$ result of regular $\mathfrak{s}$-reducing $\beta$
 7:     if $\overline{\gamma} = 0$ then
 8:         $\mathscr{H} \leftarrow \mathscr{H} \cup \{\gamma\}$
 9:     else if $\gamma$ is not singular top $\mathfrak{s}$-reducible then
10:         $\mathscr{P} \leftarrow \mathscr{P} \cup \{\operatorname{spair}(\alpha, \gamma) \mid \alpha \in \mathscr{G}$ and $\operatorname{spair}(\alpha, \gamma)$ is regular$\}$
11:         $\mathscr{G} \leftarrow \mathscr{G} \cup \{\gamma\}$
12: return $(\mathscr{G}, \mathscr{H})$
```

Thinking about correctness and termination of Algorithm 2 Line 6 seems to be problematic: Only regular $\mathfrak{s}$-reductions are done in **genSB**. Moreover, if a reduction ends with an element γ that is singular top $\mathfrak{s}$-reducible w.r.t. $\mathscr{G}$, γ is not even added to $\mathscr{G}$. It turns out that singular top $\mathfrak{s}$-reductions are useless for the computation of signature Gröbner bases.

Lemma 5.1. *Let $\alpha \in \mathscr{R}^m$ and let $\mathscr{G}$ be a signature Gröbner basis up to $\mathfrak{s}(\alpha)$. If α is singular top $\mathfrak{s}$-reducible w.r.t. $\mathscr{G}$ then α $\mathfrak{s}$-reduces to zero w.r.t. $\mathscr{G}$.*

Proof. If α is singular top $\mathfrak{s}$-reducible w.r.t. $\mathscr{G}$ then there exists $\beta \in \mathscr{G}$ and $b \in \mathscr{R}$ such that $\mathfrak{s}(\alpha) = \mathfrak{s}(b\beta)$ and $\operatorname{lt}(\overline{\alpha}) = \operatorname{lt}\left(\overline{b\beta}\right)$. If γ denotes the result of the reduction of α by $b\beta$ then $\mathfrak{s}(\gamma) < \mathfrak{s}(\alpha)$. Since $\mathscr{G}$ is a signature Gröbner basis up to $\mathfrak{s}(\alpha)$ γ $\mathfrak{s}$-reduces to zero w.r.t. $\mathscr{G}$. □

Theorem 5.1. *Given $I = \langle f_1, \ldots, f_m \rangle \subset \mathscr{R}$ and a monomial order $\leq$ on $\mathscr{R}$ with a compatible extension on $\mathscr{R}^m$* **genSB** *is an algorithm that computes a signature Gröbner basis $\mathscr{G}$ for I and a module $\mathscr{H}$ generated by a Gröbner basis for* $\operatorname{syz}(f_1, \ldots, f_m)$.

Proof. Correctness of **genSB** computing signature Gröbner basis for I is an easy generalization of Theorem 14 in Eder and Perry (2011). Allowing any compatible module monomial order on $\mathscr{R}^m$ does not change the reasoning of the corresponding proof there. On the other hand, using Lemma 5.1 and the fact that **genSB** computes $\mathscr{G}$ by increasing signatures it is an easy exercise. $\mathscr{H}$ being a Gröbner basis for $\operatorname{syz}(f_1, \ldots, f_m)$ is clear by Theorem 5.2.

If $\preceq$ orders $\mathscr{P}$ by increasing signatures then termination of **genSB** follows by Theorem 20 in Eder and Roune (2013). Otherwise it is possible that **genSB** adds several elements to $\mathscr{G}$ with the same signature: Assume an intermediate state of $\mathscr{G}$ to consist of finitely many elements, thus $\mathscr{P}$ is finite, too. Next the S-pair $a\alpha - b\beta$ regular $\mathfrak{s}$-reduces to γ w.r.t. $\mathscr{G}$.

(a) γ is top singular $\mathfrak{s}$-reducible and thus not added to $\mathcal{G}$.
(b) There might exists $\delta \in \mathcal{G}$ such that $\mathfrak{s}(\gamma) = \mathfrak{s}(\delta)$ but $\operatorname{lt}(\overline{\gamma}) < \operatorname{lt}(\overline{\delta})$. Note that $\operatorname{lt}(\overline{\gamma}) \geq 0$ so for $\mathfrak{s}(\gamma)$ there are only finitely many elements in $\mathcal{G}$.

In the second situation there must be some element ε added to $\mathcal{G}$ inbetween γ and δ such that such that $\operatorname{lt}(\overline{e\varepsilon}) = \operatorname{lt}(\overline{\delta})$ and $\mathfrak{s}(e\varepsilon) < \mathfrak{s}(\delta)$ for some monomial e in $\mathcal{R}$. We need to show that there cannot be infinitely many steps between δ and γ. First of all only finitely many steps of lower signature can be done due to our above discussion: There are only finitely many elements in $\mathcal{G}$ per signature and there are only finitely many signatures below $\mathfrak{s}(\gamma)$ handled since $\mathcal{R}$ and $\mathcal{R}^m$ are Noetherian. On the other hand, at the moment δ was added to $\mathcal{G}$, there were only finitely many S-pairs of signature $> \mathfrak{s}(\delta)$ in $\mathcal{P}$. As in the situation above, in order to get a new element in a given signature $T > \mathfrak{s}(\delta)$ new elements of signature $< T$ must be added to $\mathcal{G}$. Also for T only finitely many sig-poly pairs are possible. Moreover, between $\mathfrak{s}(\delta)$ and T **genSB** handles only finitely many elements, again due to the Noetherianess of $\mathcal{R}$ and $\mathcal{R}^m$. All in all, between two elements of the same signature **genSB** executes finitely many steps. This completes the proof of **genSB**'s termination. □

The key to prove that **genSB** computes a Gröbner basis $\mathcal{H}$ for the syzygy module is Theorem 5.2 which implies that we can determine generators of the module of syzygies from looking at those S-pairs and $\boldsymbol{e}_i$ that regular $\mathfrak{s}$-reduce to zero.

Theorem 5.2. *Let $\alpha \in \mathcal{R}^m$ be a syzygy and let $\mathcal{G}$ be a signature Gröbner basis up to signature $\mathfrak{s}(\alpha)$. Then there exists a $\beta \in \mathcal{R}^m$ with $\mathfrak{s}(\beta) \,|\, \mathfrak{s}(\alpha)$ such that β is an S-pair or has the form $\boldsymbol{e}_i$ and such that β regular $\mathfrak{s}$-reduces to zero.*

Proof. The proof is clear by Definition 4.2. A variant of Theorem 5.2 is Proposition 2.2 in Gao et al. (2011). □

Remark 5.1. Note that in Sun and Wang (2011a) Sun and Wang were the first to introduce a description of signature-based Gröbner basis algorithms where the order in which S-pairs are handled does not matter. Clearly, the above description of **genSB** covers this, since we do not restrict $\preceq$. We refer the reader interested in a proof of Theorem 5.1 with an emphasis on the pair set order $\preceq$ to Theorem 2.2 in Sun and Wang (2011a).

Note that due to $\preceq$ $\mathcal{G}$ might not be a signature Gröbner basis up to signature T when **genSB** has just handled an S-pair in signature T. There might be S-pairs of signature $< T$ which are still in $\mathcal{P}$. Nevertheless, once **genSB** terminates $\mathcal{G}$ is a signature Gröbner basis for I and thus a signature Gröbner basis up to signature T for all T.

For the sake of efficiency one might choose $\preceq$ to order $\mathcal{P}$ by increasing signatures of the S-pairs. As we see in Section 6 such an order respects the criteria to remove useless S-pairs best.

Definition 5.1. $\preceq_{\mathfrak{s}}$ denotes the order $\preceq$ in a signature-based Gröbner basis algorithm which sorts $\mathcal{P}$ by increasing signature.

Lemma 5.2. *Let* **genSB** *with $\preceq_{\mathfrak{s}}$ pick the next S-pair α to be regular $\mathfrak{s}$-reduced such that $\mathfrak{s}(\alpha) = T$. Then $\mathcal{G}$ is a signature Gröbner basis up to signature T.*

Proof. Since **genSB** with $\preceq_{\mathfrak{s}}$ handles S-pairs by increasing signature this is clear by Definition 4.2. □

Corollary 5.1. **genSB** *with $\preceq_{\mathfrak{s}}$ computes a minimal signature Gröbner basis for the corresponding input.*

Proof. A new element γ with $\overline{\gamma} \neq 0$ is added to $\mathcal{G}$ only if γ is not singular top $\mathfrak{s}$-reducible w.r.t. $\mathcal{G}$. The minimality then follows by Lemma 5.2. □

Algorithmic Property 5.1.

(a) Note that Corollary 5.1 does not hold for arbitrary pair set orders $\preceq$: Assume S-pair α being regular $\mathfrak{s}$-reduced by **genSB** to γ and $\overline{\gamma} \neq 0$. W.l.o.g. we can assume that γ is not singular top $\mathfrak{s}$-reducible.[1] If, later on, **genSB** regular $\mathfrak{s}$-reduces an S-pair from $\mathscr{P}$ to β with $\mathfrak{s}(\beta) < \mathfrak{s}(\gamma)$ such that $\operatorname{lt}\left(\overline{\beta}\right) \mid \operatorname{lt}\left(\overline{\gamma}\right)$ then a new S-pair $\varepsilon = b\beta - \gamma$ is handled. Hereby $\mathfrak{s}(\varepsilon) = \mathfrak{s}(\gamma)$ but $\operatorname{lt}(\overline{\varepsilon}) < \operatorname{lt}(\overline{\gamma})$. So $\mathscr{G}$ can have several elements in the same signature T.
Still, Lemma 4.2 is valid and makes sense: Once all S-pairs of signature T are handled by **genSB** (in any given order $\preceq$) $\mathscr{G}$ is a signature Gröbner basis up to signature T and γ can be further regular $\mathfrak{s}$-reduced, namely to ε. Thus, in Section 4 and in the following we often consider signature Gröbner bases up to some signature T. Due to our considerations here, the second part of Remark 5.1 and Lemma 5.2 this makes sense.

(b) As one can easily see, Algorithm 2 does only rely on data provided by $\overline{\alpha}$ and $\mathfrak{s}(\alpha)$, but it does not need to store α completely. Thus instead of using α one can optimize an instantiation of **genSB** by using $(\mathfrak{s}(\alpha), \overline{\alpha})$.
Moreover, if one is only interested in a Gröbner basis for $f_1, \ldots, f_m$, **genSB** can be optimized in the sense that one can restrict $\mathscr{H}$ to store only the initial module of the corresponding syzygy module: Using only sig-poly pairs in Algorithm 2 we are no longer able to store the full module element γ in $\mathscr{H}$ at Line 8. Still one can compute at the same time the initial submodule $\mathscr{H}$ of the syzygy module of $f_1, \ldots, f_m$. In order to do so, one needs to exchange Line 8 with

```
8: $\mathscr{H} \leftarrow \mathscr{H} \cup \left\{\mathfrak{s}(\gamma)\right\}$
```

The fact that one can use signature-based Gröbner basis algorithms to compute the initial module of the module of corresponding syzygies was first mentioned in Gao et al. (2010a).

Signature-based Gröbner basis algorithms like **genSB** are in the vein of a bigger class of algorithms computing the image and the kernel of a module homomorphism at the same time: In our setting the image is the signature Gröbner basis $\mathscr{G}$ and the kernel is the syzygy module $\mathscr{H}$. Other well-known, Gröbner basis related algorithms of this type are, for example, the MMM algorithm by Marinari et al. (1993) and the FGLM algorithm by Faugère et al. (1993). Recently, Sun gave a detailed overview on the connections between those algorithms in Sun (2013).

6. S-pair elimination

Until now we have introduced signature Gröbner bases and their computation only to receive a Gröbner basis for some ideal $\langle f_1, \ldots, f_m \rangle$ and the initial module of $\operatorname{syz}(f_1, \ldots, f_m)$. As mentioned in Section 5 **genSB** should be understood as a template and common basis for all signature-based Gröbner basis algorithms. Thus, it is slow and not at all optimized. One main bottleneck of **genSB** is the high number of $\mathfrak{s}$-reductions to zero. As for the classic Buchberger algorithm (see Buchberger, 1965, 1985) we are searching for criteria to discard such useless computations in advance like we have used known syzygies in **MatrixF5** in Section 3. The notations and concepts used in this section are introduced in Eder and Roune (2013).

Assume that **genSB** regular $\mathfrak{s}$-reduces an S-pair in signature T to $\gamma \in \mathscr{R}^m$. Then three different situations can appear:

(a) If γ is a syzygy then γ is added to $\mathscr{H}$ in Line 8.
(b) If γ is not syzygy but singular top $\mathfrak{s}$-reducible then by Lemma 5.1 γ will $\mathfrak{s}$-reduce to zero. Thus it is discarded in Line 9.
(c) Otherwise γ is used to build new S-pairs with elements in $\mathscr{G}$ (Line 10) and later on itself added to $\mathscr{G}$ (Line 11).

[1] Otherwise there exists $\delta \in \mathscr{G}$ with $\mathfrak{s}(\delta) = c\,\mathfrak{s}(\gamma)$ and $\operatorname{lt}\left(\overline{\delta}\right) = c \operatorname{lt}(\overline{\gamma})$, thus we can just replace γ by δ in the above situation.

Definition 6.1. For the above three cases T is respectively a *syzygy*, *singular* or *basis signature*.

We are interested in the situations where elements are discarded. In the following we take a closer look at syzygy and singular signatures.

6.1. Eliminating S-pairs by known syzygies

Clearly, we receive syzygies by $\mathfrak{s}$-reductions to zero in **genSB**, but there are also syzygies immediately known without precomputations as we have already seen in the example computation of **MatrixF5** in Section 3.

Definition 6.2. The *Koszul syzygy* between $\alpha, \beta \in \mathscr{G}$ is $\operatorname{ksyz}(\alpha, \beta) := \overline{\beta}\alpha - \overline{\alpha}\beta$. If $\mathfrak{s}\left(\overline{\beta}\alpha\right) \not\simeq \mathfrak{s}\left(\overline{\alpha}\beta\right)$ then the Koszul syzygy is *regular*. By “Koszul syzygy” we always mean “regular Koszul syzygy”.

Trivial relations resp. principal syzygies are Koszul syzygies. Using those and already computed zero reductions we are able to flag a given signature being predictably syzygy.

Definition 6.3. A signature T is *predictably syzygy* if there exists

(a) a Koszul syzygy $\gamma \in \mathscr{R}^m$ such that $\mathfrak{s}(\gamma) = T$, or
(b) a syzygy $\sigma \in \mathscr{R}^m$ such that $\mathfrak{s}(\sigma) < T$ and $\mathfrak{s}(\sigma) \mid T$.

Being predictably syzygy gives us a useful characterization when computing Gröbner bases.

Lemma 6.1 *(Syzygy criterion). Let $\alpha, \beta \in \mathscr{G}$, $\gamma = \operatorname{spair}(\alpha, \beta)$ with $\mathfrak{s}(\gamma)$ being predictably syzygy, and let $\mathscr{G}$ be a signature Gröbner basis up to $\mathfrak{s}(\gamma)$. Then γ $\mathfrak{s}$-reduces to zero w.r.t. $\mathscr{G}$. Moreover, if S is a syzygy signature and $S \mid T$ then T is also a syzygy signature.*

Proof. If γ is predictably syzygy then there exists a syzygy $\sigma \in \mathscr{R}^m$ such that $\mathfrak{s}(\sigma) = \mathfrak{s}(\gamma)$. $\overline{\gamma - \sigma} = \overline{\gamma}$ but $\mathfrak{s}(\gamma - \sigma) < \mathfrak{s}(\gamma)$. By Definition 4.2 $\gamma - \sigma$ $\mathfrak{s}$-reduces to zero w.r.t. $\mathscr{G}$, thus also γ behaves in this way. □

The outcome of Lemma 6.1 is that whenever we handle an S-pair γ in a signature-based Gröbner basis algorithm like **genSB** whose signature is divisible by the signature of a syzygy we can discard γ. Or in more general: A predictably syzygy signature is a syzygy signature.

Remark 6.1. Restricting Lemma 6.1 to principal syzygies and the compatible module monomial order used to $<_{\text{pot}}$ we get a statement equivalent to the **F5** criterion presented in Theorem 1 in Faugère (2002).

Clearly, if we have knowledge of syzygies other than the Koszul ones a priori, we can directly use them to remove useless S-pairs, see also Section 9.

6.2. Uniqueness of S-pairs at a given signature

Next we are looking at the situation where the $\mathfrak{s}$-reduction of an S-pair ends with a non-syzygy element γ that is singular top $\mathfrak{s}$-reducible w.r.t. $\mathscr{G}$. We have already seen in Lemma 5.1 that we can discard such S-pairs in the computations. The remaining question is how to detect such a situation.

Being singular top $\mathfrak{s}$-reducible is a special case of the situation where there are two or more S-pairs in the same signature T. If so, we only have to regular $\mathfrak{s}$-reduce one of them as they all regular $\mathfrak{s}$-reduce to the same thing by Lemma 4.2. Since $\mathfrak{s}$-reduction proceeds by decreasing the lead term, we can for example try to speed up the process by choosing an S-pair γ in signature T whose lead term $\operatorname{lt}(\overline{\gamma})$ is minimal. If $\mathfrak{s}(\operatorname{spair}(\alpha,\beta)) = \mathfrak{s}(a\alpha)$, then we get the same result from regular $\mathfrak{s}$-reducing $\operatorname{spair}(\alpha,\beta)$ as for regular $\mathfrak{s}$-reducing $a\alpha$ by Notation 4.1 (b).

All in all we get the following description of the *singular criterion*:

Lemma 6.2 *(Singular criterion). For any signature T we need to handle exactly one $a\alpha \in \mathscr{R}^m$ from*

$$\mathscr{C}_T = \{a\alpha \mid \alpha \in \mathscr{G}, a \text{ is a monomial and } \mathfrak{s}(a\alpha) = T\} \tag{1}$$

computing a signature Gröbner basis.

Remark 6.2.

(a) Note that α might not be involved in any S-pair in signature T. In this situation at signature T no S-pair is computed resp. $\mathfrak{s}$-reduced at all.
(b) Note that when computing signature Gröbner bases by signature-based algorithms with an arbitrary pair set order $\preceq$ uniqueness of the elements in signature T is not guaranteed. A situation as pointed out in Property 5.1 (1) might appear and thus after having already chosen and regular $\mathfrak{s}$-reduced an element from $\mathscr{C}_T$ the algorithm might come back to signature T and makes a new choice from $\mathscr{C}_T$.
(c) Lemma 6.2 corresponds to rewriting rows in **MatrixF5** as done in Section 3. Choosing an element in signature T mirrors searching already reduced row echelon forms N_d for better representations of the row labeled by T.

What is now left to do is to make a good choice for $a\alpha$ from $\mathscr{C}_T$. For this we need to introduce the notion of a rewriter in the following.

7. Rewrite bases

In Section 6.2 we have seen that per signature T we only need to take care of one element. In order to make a choice of such an element we need to define an order on $\mathscr{C}_T$. For this the notion of so-called *rewriters* is introduced in the following. In this section we present a first signature-based Gröbner basis algorithm using S-pair elimination as presented in Section 6. This is then the fundamental algorithm we can derive all known, efficient instantiations from.

Similar attempts to achieve such a comprehensive representation of signature-based Gröbner basis algorithms are given, for example, in Huang (2010), Sun and Wang (2011a), Gao et al. (2016). The algorithms presented there, called **TRB** (see Section 11.2) and **GBGC** (see Section 11.3) are included in Algorithm 3, called **RB**. Note that in Eder and Roune (2013) the main concepts of this section and the algorithm **RB** are already introduced. ere we slightly generalize **RB** in order to allow different pair set orders. This is needed to cover also **GBGC** which would otherwise not be a special instantiation of **RB**. Besides this algorithmic construction, we give in Section 7.4 a characterization of rewrite bases.

7.1. Combining elimination criteria

Before we introduce the concept of rewriter, let us shortly recall the syzygy criterion: An element γ is discarded if there exists a syzygy σ such that $\mathfrak{s}(\sigma) \mid \mathfrak{s}(\gamma)$, or in other words, there exists a monomial $s \in \mathscr{R}$ such that $\mathfrak{s}(s\sigma) = \mathfrak{s}(\gamma)$. Thus we have again two elements of the same signature and need to decide which one to handle. Of course, by Remark 6.2 we take $s\sigma$ since we know that $\overline{s\sigma} = 0$ already, so no further computations need to be done in signature $\mathfrak{s}(s\sigma)$. But this is nothing else but a rewording of Lemma 6.1, the syzygy criterion. It follows that we can generalize the set $\mathscr{C}_T$ to

$$\mathscr{C}_T = \{a\alpha \mid \alpha \in \mathscr{G} \cup \mathscr{H}, a \text{ is a monomial and } \mathfrak{s}(a\alpha) = T\}. \tag{2}$$

The only difference between Equations (1) and (2) is that α is now allowed to be in $\mathscr{H}$, too. With this the two criteria from Sections 6.1 and 6.2 to find useless S-pairs unite to one single criterion. Furthermore, with this only one question remains to be answered when implementing signature-based Gröbner basis algorithms: How to choose the single element from $\mathscr{C}_T$?

Since we have seen that all elements from $\mathscr{C}_T$ "rewrite" the same information for the input ideal $I = \langle f_1, \ldots, f_m \rangle$ at signature T the following naming conventions are reasonable.

Definition 7.1.

(a) A *rewrite order* $\trianglelefteq$ is a partial order on $\mathscr{G} \cup \mathscr{H}$ such that $\trianglelefteq$ is a total order on $\mathscr{G}$

(b) An element $\alpha \in \mathscr{G}$ is a *rewriter in signature* T if $\mathfrak{s}(\alpha) \mid T$. If for a monomial $a \in \mathscr{R}$ $\mathfrak{s}(a\alpha) = T$ we also say for convenience that $a\alpha$ is a rewriter in signature T. The $\preceq$-maximal rewriter in signature T is the *canonical rewriter in signature* T. A multiple $a\alpha$ of a basis element α is *rewritable* if α is not the canonical rewriter in signature $\mathfrak{s}(a\alpha)$.

Remark 7.1. Of course, the definition of a rewrite order in Definition 7.1 is rather generic and not practical. For example, it does not even take care of the elements in $\mathscr{H}$. Clearly, for optimized computations one want $s\sigma$ be the canonical rewriter in signature $\mathfrak{s}(s\sigma)$ for $\sigma \in \mathscr{H}$. Still, in terms of correctness, one do not need to restrict Definition 7.1 (a) to this. We see in the following how explicitly defined rewrite orders can be used to reach efficient specializations of signature-based criteria to discard useless S-pairs.

Example 7.1. Looking again at Example 4.1 we see that $\mathfrak{s}(x\alpha_5) = \mathfrak{s}\left(x^2\alpha_4\right) = x^2 z\boldsymbol{e}_2$. Defining a rewrite order $\trianglelefteq$ by $\alpha \trianglelefteq \beta$ if $\mathfrak{s}(\alpha) \leq \mathfrak{s}(\beta)$ we can see that $x^2\alpha_4$ is rewritable since α_5 is the canonical rewriter in signature $x^2 z\boldsymbol{e}_2$ due to $\mathfrak{s}(\alpha_4) = z\boldsymbol{e}_2 < xz\boldsymbol{e}_2 = \mathfrak{s}(\alpha_5)$.

Definition 7.1 gives us a choice for $\mathscr{C}_T$, namely we can choose the canonical rewriter in signature T from $\mathscr{C}_T$. Of course, using Equation (2) to find the canonical rewriter w.r.t. $\trianglelefteq$ instead of using the syzygy criterion and the rewritable criterion independently from each other we need to explain the following: If a syzygy exists for signature T, then all S-pairs in signature T are removed. It turns out that in the general description of rewrite bases we are giving here this need not be true at all. Of course it makes sense to define $\alpha \trianglelefteq \beta$ whenever $\beta \in \mathscr{H}$. We come back to this fact once we are explicitly defining rewrite orders in Section 7.3.

Analogously to Section 3.2 in Eder and Roune (2013) we introduce next the important notion of a rewrite basis. Note that the combination of the syzygy and the singular criterion lead to a much easier notation.

Definition 7.2. $\mathscr{G}$ is a *rewrite basis in signature* T if the canonical rewriter in T is not regular top $\mathfrak{s}$-reducible. $\mathscr{G}$ is a *rewrite basis up to signature* T if $\mathscr{G}$ is a rewrite basis in all signatures $S < T$. $\mathscr{G}$ is a *rewrite basis* if $\mathscr{G}$ is a rewriter basis in all signatures.

In the following we show the strong connection between rewrite bases and signature Gröbner bases. Clearly, since a rewrite basis is parametrized by a rewrite order and a signature Gröbner basis is parametrized by an order on its signatures, we need to connect both orders in a natural way.

Convention 7.1. *If not otherwise stated we assume in the following that a rewrite order fulfills $\mathfrak{s}(\alpha) \mid \mathfrak{s}(\beta) \Longrightarrow \alpha \trianglelefteq \beta$. Note that such an order always exists due to our assumption that $\mathfrak{s}(\alpha) \simeq \mathfrak{s}(\beta) \Longrightarrow \alpha = \beta$.*

Lemma 7.1 *(Eder and Roune (2013)). If $\mathscr{G}$ is a rewrite basis up to signature T then $\mathscr{G}$ is also a signature Gröbner basis up to T.*

Proof. See Lemma 8 in Eder and Roune (2013). □

7.2. An algorithm computing rewrite bases

Next we present an algorithm quite similar to Algorithm 2 that implements the above mentioned S-pair elimination in the sense that it computes a rewrite basis. We show that depending on the chosen rewrite order the size of the rewrite basis varies.

Algorithm 3 Rewrite basis algorithm **RB**.

```
Require: Ideal $I = \langle f_1, \ldots, f_m \rangle \subset \mathscr{R}$, monomial order $\leq$ on $\mathscr{R}$ and a compatible extension on $\mathscr{R}^m$, total order $\preceq$ on the pairset
    $\mathscr{P}$ of S-pairs, a rewrite order $\trianglelefteq$ on $\mathscr{G} \cup \mathscr{H}$
Ensure: Rewrite basis $\mathscr{G}$ for $I$, Gröbner basis $\mathscr{H}$ for $\operatorname{syz}(f_1, \ldots, f_m)$
 1: $\mathscr{G} \leftarrow \emptyset$, $\mathscr{H} \leftarrow \emptyset$
 2: $\mathscr{P} \leftarrow \{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_m\}$
 3: $\mathscr{H} \leftarrow \left\{ f_i \boldsymbol{e}_j - f_j \boldsymbol{e}_i \mid 1 \leq i < j \leq m \right\} \subseteq \mathscr{R}^m$
 4: while $\mathscr{P} \neq \emptyset$ do
 5:     $\beta \leftarrow \min_{\preceq} \mathscr{P}$
 6:     $\mathscr{P} \leftarrow \mathscr{P} \setminus \{\beta\}$
 7:     if not Rewritable $(\beta, \mathscr{G} \cup \mathscr{H}, \trianglelefteq)$ then
 8:         $\gamma \leftarrow$ result of regular $\mathfrak{s}$-reducing $\beta$
 9:         if $\overline{\gamma} = 0$ then
10:             $\mathscr{H} \leftarrow \mathscr{H} + \{\gamma\}$
11:         else
12:             $\mathscr{P} \leftarrow \mathscr{P} \cup \left\{ \operatorname{spair}(\alpha, \gamma) \,\middle|\, \alpha \in \mathscr{G} \text{ and } \operatorname{spair}(\alpha, \gamma) \text{ is regular} \right\}$
13:             $\mathscr{G} \leftarrow \mathscr{G} \cup \{\gamma\}$
14: return $(\mathscr{G}, \mathscr{H})$
```

Algorithm 4 Rewritability check **Rewritable** for **RB**.

```
Require: S-pair $a\alpha - b\beta \in \mathscr{R}^m$, finite subset $\mathscr{G} \cup \mathscr{H} \in \mathscr{R}^m$, rewrite order $\trianglelefteq$ on $\mathscr{G} \cup \mathscr{H}$
Ensure: "True" if S-pair is rewritable; else "false"
if $a\alpha$ is rewritable then
    return true
if $b\beta$ is rewritable then
    return true
return false
```

Algorithm 3 differs from **genSB** in three points:

(a) In Line 3 **RB** directly adds the known Koszul syzygies to $\mathscr{H}$. This increases the number of possible canonical rewriters in $\mathscr{C}_T$ in a given signature T.

(b) In Line 7 **RB** uses Algorithm 4 to check if the S-pair β is rewritable or not. If so, **RB** discards β and chooses the next S-pair in $\mathscr{P}$. **genSB** does not provide any such check.

(c) In Lines 12 and 13 **RB** takes the currently regular $\mathfrak{s}$-reduced γ, generates new regular S-pairs with it and adds γ to $\mathscr{G}$. Whereas **genSB** handles only not singular top $\mathfrak{s}$-reducible γ, **RB** runs these steps on all non-syzygy γ.

Whereas the first two points are optimizations compared to **genSB**, the third change seems to be absurd. We have already seen that singular top $\mathfrak{s}$-reducible elements are not needed for $\mathscr{G}$, so why adding them? The reason is that **RB** computes rewrite bases, and in order to fulfill the definition it has to add all these elements to $\mathscr{G}$ nevertheless they are singular top $\mathfrak{s}$-reducible or not. Since **RB** depends on the chosen rewrite order $\trianglelefteq$ we need to store all elements, since they could lead to new canonical rewriters. We see in Section 7.3 how different rewrite orders can affect **RB** quite a lot.

Analogously to Theorem 5.1 we receive the following statement.

Theorem 7.1 *(Eder and Roune (2013)). Given $I = \langle f_1, \ldots, f_m \rangle \subset \mathscr{R}$, a monomial order $\leq$ on $\mathscr{R}$ with a compatible extension on $\mathscr{R}^m$, $\preceq_{\mathfrak{s}}$ on $\mathscr{P}$ and a rewrite order $\trianglelefteq$* **RB** *is an algorithm that computes a rewrite basis $\mathscr{G}$ for I and a module $\mathscr{H}$ generated by a Gröbner basis for* $\operatorname{syz}(f_1, \ldots, f_m)$.

Proof. See Eder and Roune (2013): Theorem 7 for correctness and Theorem 20 for termination. □

Algorithmic Property 7.1.

(a) In Eder and Roune (2013) algorithm **RB** is presented for the first time. Here **RB** is presented more general in the sense that different pair set orders are allowed. Moreover, generalizing the idea of rewritability to include the syzygy criterion is new in the current presentation.

(b) If $<_{\text{pot}}$ is used then **RB** computes $\mathscr{G}$ and $\mathscr{H}$ incremental by increasing indices. Thus it makes sense to optimize Algorithm 3 to recompute $\mathscr{H}$ once the computations in a new index k starts: At this point we have a Gröbner basis $\overline{\mathscr{G}} = \{\overline{\alpha_1}, \ldots, \overline{\alpha_{k-1}}\} \subset \mathscr{R}$ for $\langle f_1, \ldots, f_{i-1}\rangle$. Defining $\alpha_k = \boldsymbol{e}_k$ such that $\overline{\alpha_k} := f_i$ we can add for $j < k$ $\overline{\alpha_j}\alpha_k - \overline{\alpha_k}\alpha_j$ to $\mathscr{H}$.

(c) Note that in spite of Theorem 5.1 we have to limit Theorem 7.1 for **RB**: Whereas one can show that **genSB** terminates for any chosen pair set order $\preceq$ we restrict **RB** to $\preceq_{\mathfrak{s}}$. The problem is the interplay between $\preceq$ and $\trianglelefteq$: It is possible to choose both in a way such that **RB** adds the same sig-poly pair to $\mathscr{G}$. This is possible due to the fact that **RB** does not check for singular top $\mathfrak{s}$-reducibility when adding new elements to $\mathscr{G}$ (since this shall be handled by the more general and flexible rewritability criterion and thus $\trianglelefteq$).

By the ideas of Sun and Wang (2011a) it is noted in Gao et al. (2011) that GVW can compute Gröbner bases by handling S-pairs in any given order. This coincides with our descriptions of **genSB** and **RB**. Moreover, we show that not only **GVW** can do so, but all known efficient instantiations of **RB**, for example, also **F5**.

(d) Note that there is a strong connection between the signature and the so-called *sugar degree*. It is shown in Eder (2013a) that using $\preceq_{\mathfrak{s}}$ combined with a degree compatible monomial order $<$ a signature-based Gröbner basis algorithm refines the sugar degree order of critical pairs.

(e) Since all known signature-based Gröbner basis algorithms are special cases of **RB** their correctness and termination is clear with Theorem 7.1. Later on, we discuss the topic of termination further, especially for **F5** in Section 10. There we do not give full proofs, but refer the reader interested in more details on proving termination to the corresponding papers. A small selection might be already mentioned here:

- ▶ Galkin (2012), Pan et al. (2012, 2013), Eder and Roune (2013) for signature-based algorithms using $\trianglelefteq_{\text{add}}$.
- ▶ Arri and Perry (2011), Gao et al. (2011), Pan et al. (2012, 2013), Roune and Stillman (2012a), Eder and Roune (2013) for signature-based algorithms using $\trianglelefteq_{\text{rat}}$.

Note that due to Lemma 5.2, Corollary 5.1 as well as the definition of rewritability in 7.1 choosing $\preceq_{\mathfrak{s}}$ is the best possible choice for an efficient computation[2] of $\mathscr{G}$ and $\mathscr{H}$. Thus we restrict ourselves in Theorem 7.1 to this situation.

Moreover, let us agree in the remaining of the paper on the following:

Convention 7.2. *If not otherwise stated we assume* $\preceq = \preceq_{\mathfrak{s}}$.

If **RB** can make use of the rewritability checks, is the resulting rewrite basis, and thus signature Gröbner basis smaller?

Lemma 7.2. *Given* $I = \langle f_1, \ldots, f_m\rangle \subset \mathscr{R}$ *and a monomial order* $\leq$ *on* $\mathscr{R}$ *with a compatible extension on* $\mathscr{R}^m$ *the basis computed by* **genSB** *is always a subset of the one computed by* **RB** *up to sig-poly pairs.*

Proof. Due to $\preceq_{\mathfrak{s}}$ this follows directly from Corollary 5.1. □

[2] This already follows as a consequence from Section 3.3 in Eder and Roune (2013).

The optimization we achieve when switching from **genSB** to **RB** lies in the fact that **genSB** regular $\mathfrak{s}$-reduces many more elements to zero w.r.t. $\mathscr{G}$, whereas **RB** can detect, and thus discard, such an $\mathfrak{s}$-reduction in advance.

The following two lemmata are of importance when we compare different rewrite rules and specific implementations of **RB**.

Lemma 7.3 *(Slight variant of Lemma 11 in Eder and Roune (2013)). Let $\alpha \in \mathscr{R}^m$, let $\mathscr{G}$ be a rewrite basis up to signature $\mathfrak{s}(\alpha)$ and let t be a regular $\mathfrak{s}$-reducible term of $\overline{\alpha}$. Then there exists a regular $\mathfrak{s}$-reducer $b\beta$ which is*

- *not regular top $\mathfrak{s}$-reducible,*
- *not rewritable and*
- *not syzygy.*

Proof. Let M_t be the set of all regular $\mathfrak{s}$-reducers of t. Let $c\gamma \in M_t$ of minimal possible signature T, and let $b\beta$ be the canonical rewriter in signature T. By definition, $b\beta$ is not rewritable. Since $\mathfrak{s}(c\gamma) < \mathfrak{s}(\alpha)$ $b\beta$ is not regular top $\mathfrak{s}$-reducible.

Moreover, there cannot exist a $d\delta \in M_t$ such that $d\delta$ regular top $\mathfrak{s}$-reduces $c\gamma$ as otherwise $\mathfrak{s}(d\delta) < T$. By Lemma 4.2 $\operatorname{lt}\left(\overline{b\beta}\right) = \operatorname{lt}\left(\overline{c\gamma}\right)$ and thus $b\beta \in M_t$.

If there exists $\sigma \in \langle \mathscr{H} \rangle$ such that $\mathfrak{s}(\sigma) = T$ then $b\beta - \sigma \in M_t$ since $b\beta \in M_t$, but $\mathfrak{s}(b\beta - \sigma) < T$. This is a contradiction. □

Lemma 7.4. *Let $\mathscr{G}$ be a rewrite basis up to signature T, and let $a\alpha$ be the canonical rewriter in signature T. Then* **RB** *$\mathfrak{s}$-reduces an S-pair in signature T if and only if $a\alpha$ is regular top $\mathfrak{s}$-reducible and T is not predictably syzygy.*

Proof. See Lemma 12 in Eder and Roune (2013). □

Remark 7.2. Sun and Wang (2011a) explain a generalized criterion for signature-based Gröbner basis algorithms which is used in Gao et al. (2011) to generalize the original description of the GVW algorithm given in Gao et al. (2010b). For this a partial order on $\mathscr{R}^m \times \mathscr{R}$ is defined. Note that this is included in our combined criterion described in Section 7.1. This is very similar to the rewrite order we defined in 7.1. Still there are some slight differences: Sun and Wang call a partial order on $\mathscr{R}^m \times \mathscr{R}$ *admissible* if for any S-pair $a\alpha - b\beta$ that $\mathfrak{s}$-reduced to γ with $\mathfrak{s}(\gamma) = \mathfrak{s}(a\alpha)$ it holds that $\alpha \trianglelefteq \gamma$. Clearly, this is covered by our definition of a rewrite order. Still an admissible partial order could lead to several chains of ordered elements in $\mathscr{G}$ which are not connected to each other. This would mean that a possible canonical rewriter in signature T in chain C_i cannot be used to discard a useless S-pair which consists of a generator in chain C_j. So for each chain C_i we would receive an own set of rewriters in signature T:

$$\mathscr{C}_{T,C_i} = \{a\alpha \,|\alpha \in \mathscr{G} \cup \mathscr{H}, a \text{ is a monomial and } \mathfrak{s}(a\alpha) = T, \alpha \text{ is in chain } C_i\}.$$

Note that correctness and also termination of **RB** is not effected by this, but the criterion is not as efficient as it is using a total order $\trianglelefteq$ on $\mathscr{G}$.

All in all, the efficiency of **RB** depends on

(a) the order in which S-pairs are handled, and
(b) the strength of the detection of useless S-pairs.

We know already that $\preceq_{\mathfrak{s}}$ is the best possible order for $\mathscr{P}$ in terms of the size of the resulting signature Gröbner basis and the efficiency of the $\mathfrak{s}$-reduction steps. The second point, as well as the size of $\mathscr{G}$ also depend on the chosen rewrite order. So as a final step on our way understanding signature-based Gröbner basis algorithms we have to investigate the overall impact of rewrite orders.

7.3. *Choosing a rewrite order*

When thinking about a possible rewrite order to choose we should look again the set of all possible rewriters in signature T:

$$\mathscr{C}_T = \{a\alpha \,|\alpha \in \mathscr{G} \cup \mathscr{H}, a \text{ is a monomial and } \mathfrak{s}(a\alpha) = T\}.$$

We want to choose the canonical rewriter $a\alpha$ in T for further considerations in **RB** and discard all other elements. It is clear that we want to choose $a\alpha$ in terms of "being easier to $\mathfrak{s}$-reduce than the other elements in $\mathscr{C}_T$". From the point of view of Gröbner basis computations there are two canonical selections:

(a) α has been added to $\mathscr{G}$ latest for all $\beta \in \mathscr{G}$ such that $b\beta \in \mathscr{C}_T$. Here we hope that α is better $\mathfrak{s}$-reduced w.r.t. $\mathscr{G}$ and thus $a\alpha$ might be easier to handle in the following.

(b) Let $\operatorname{lt}\left(\overline{a\alpha}\right) \leq \operatorname{lt}\left(\overline{b\beta}\right)$ for any $b\beta \in \mathscr{C}_T$. Choosing $a\alpha$ as canonical rewriter in signature T we expect the fewest possible $\mathfrak{s}$-reduction steps.

It turns out that all signature-based Gröbner basis algorithms known until now choose one of the above options. Thus it makes sense to have a closer look at those.

Definition 7.3. Let $\alpha, \beta \in \mathscr{G} \cup \mathscr{H}$ during a computation of **RB**.

(a) We say that $\alpha \trianglelefteq_{\text{add}} \beta$ if $\beta \in \mathscr{H}$ or α has been added to $\mathscr{G}$ before β is added to $\mathscr{G}$.
(b) We say that $\alpha \trianglelefteq_{\text{rat}} \beta$ if $\mathfrak{s}(\alpha)\operatorname{lt}\left(\overline{\beta}\right) < \mathfrak{s}(\beta)\operatorname{lt}(\overline{\alpha})$ or if $\mathfrak{s}(\alpha)\operatorname{lt}\left(\overline{\beta}\right) = \mathfrak{s}(\beta)\operatorname{lt}(\overline{\alpha})$ and $\mathfrak{s}(\alpha) < \mathfrak{s}(\beta)$.

Remark 7.3.

(a) Using $\preceq_{\mathfrak{s}}$ in **RB** $\alpha \trianglelefteq_{\text{add}} \beta$ for $\alpha, \beta \in \mathscr{G}$ induces that $\mathfrak{s}(\alpha) < \mathfrak{s}(\beta)$.

(b) The suffix "rat" of $\trianglelefteq_{\text{rat}}$ refers to the usual notation of this rewrite order, for example, in Galkin (2012), Eder and Roune (2013). There the *ratios* of the signature and the polynomial lead term are compared:

$$\frac{\mathfrak{s}(\alpha)}{\operatorname{lt}(\overline{\alpha})} < \frac{\mathfrak{s}(\beta)}{\operatorname{lt}\left(\overline{\beta}\right)}.$$

Multiplying both sides of the inequality by $\operatorname{lt}(\overline{\alpha})\operatorname{lt}\left(\overline{\beta}\right)$ we get the representation of $\trianglelefteq_{\text{rat}}$ as in Definition 7.3 (b). We prefer the notation without ratios due to two facts: First of all we do not need to extend $<$ on the ratios and introduce negative exponents. Secondly, we can handle $\operatorname{lt}(\overline{\alpha}) = 0$ for elements $\alpha \in \mathscr{H}$.

Lemma 7.5. *If there exists $\gamma \in \mathscr{H}$ such that $\gamma \in \mathscr{C}_T$ then all S-pairs in signature T are discarded in* **RB** *using either $\trianglelefteq_{add}$ or $\trianglelefteq_{rat}$.*

Proof. If $\gamma \in \mathscr{H} \cap \mathscr{C}_T$ then for all α in $\mathscr{G} \cap \mathscr{C}_T$ it holds by definition that $\alpha \trianglelefteq_{\text{add}} \gamma$. Furthermore, $\alpha \trianglelefteq_{\text{rat}} \gamma$ due to $\mathfrak{s}(\alpha)\operatorname{lt}(\overline{\gamma}) < \mathfrak{s}(\gamma)\operatorname{lt}(\overline{\alpha})$ where $\operatorname{lt}(\overline{\gamma}) = 0$. Thus no S-pair in signature T is handled by **RB**. □

Corollary 7.1. *If $f_1, \ldots, f_m \in \mathscr{R}$ form a regular sequence then there is no $\mathfrak{s}$-reduction to zero while* **RB** *computes a signature Gröbner basis for $\langle f_1, \ldots, f_m\rangle$ using $<_{pot}$.*

Proof. The homology of the Koszul complex K^* associated to the regular sequence $(f_1, \ldots, f_m)$ has the property that $H_\ell(K^*) = 0$ for $\ell > 0$. Thus, there exist only Koszul syzygies of the form $\overline{\alpha_i}\alpha_j - \overline{\alpha_j}\alpha_i \in \mathscr{R}^m$ where $\overline{\mathscr{G}} = \{\overline{\alpha_1}, \ldots, \overline{\alpha_{k-1}}\}$ is the intermediate Gröbner basis for $\langle f_1, \ldots, f_{i-1}\rangle$ and $\alpha_k =$

$\boldsymbol{e}_k \in \mathscr{R}^m$ such that $\overline{\alpha_k} = f_i$. By Property (b) those syzygies are added in Line 3 of Algorithm 3. It follows that any zero reduction, corresponding to such a syzygy is detected in advance. □

Corollary 7.2. *If $f_1, \ldots, f_m \in \mathscr{R}$ form a homogeneous regular sequence then there is no $\mathfrak{s}$-reduction to zero while* **RB** *computes a signature Gröbner basis for $\langle f_1, \ldots, f_m \rangle$ using $<_{\text{d-pot}}$.*

Proof. This is clear by Corollary 7.1 and the fact that **RB** computes the signature Gröbner basis for the input ideal by increasing polynomial degree. Thus at each new degree step d $\mathscr{G}$ is already a d'-Gröbner basis for $\langle f_1, \ldots, f_m \rangle$ for all $d' < d$. □

Another question to answer is why **Rewritable** is allowed to check both generators of an S-pair and not only the one with higher signature.

Lemma 7.6. *Assume* **RB** *computing a signature Gröbner basis for $\langle f_1, \ldots, f_m \rangle$ using $\trianglelefteq_{add}$ or $\trianglelefteq_{rat}$. If* **Rewritable** *returns "true" for input S-pair $a\alpha - b\beta$, $\mathfrak{s}(a\alpha) > \mathfrak{s}(b\beta)$ due to $b\beta$ being rewritable then* **RB** *can discard $a\alpha - b\beta$.*

Proof. If $b\beta$ is rewritable then there exists $\gamma \in \mathscr{G} \cup \mathscr{H}$, $\gamma \neq \beta$ such that γ is the canonical rewriter in $\mathfrak{s}(b\beta)$. Let $\mathfrak{s}(c\gamma) = \mathfrak{s}(b\beta)$ for some monomial c. Since $\beta \trianglelefteq \gamma$ and $\trianglelefteq$ is either $\trianglelefteq_{\text{add}}$ or $\trianglelefteq_{\text{rat}}$ it follows from Definition 7.3 that $\operatorname{lt}\left(\overline{b\beta}\right) \geq \operatorname{lt}\left(\overline{c\gamma}\right)$. Two situations can happen:

(a) If $\operatorname{lt}\left(\overline{c\gamma}\right) = \operatorname{lt}\left(\overline{b\beta}\right)$ then **RB** handles the S-pair $a\alpha - c\gamma$.

(b) If $\operatorname{lt}\left(\overline{c\gamma}\right) < \operatorname{lt}\left(\overline{b\beta}\right)$ then there exists $\delta_i \in \mathscr{G}$ and monomials d_i such that $\overline{b\beta} = \sum_{i=1}^{\ell} \overline{d_i\delta_i} + \overline{c\gamma}$ and $\mathfrak{s}(d_i\delta_i) < \mathfrak{s}(b\beta)$ for all $i \in \{1, \ldots, \ell\}$ since $\mathscr{G}$ is a signature Gröbner basis up to $\mathfrak{s}(b\beta)$. Thus **RB** handles for some $k \in \{1, \ldots, k\}$ $a\alpha - d_k\delta_k = \lambda \operatorname{spair}(\alpha, \delta_k)$ for some monomial $\lambda \geq 1$. Note that this case includes $\gamma \in \mathscr{H}$. □

Note that whereas we have to handle elements in $\mathscr{H}$ explicitly for $\trianglelefteq_{\text{add}}$ there is no need to do so for $\trianglelefteq_{\text{rat}}$: If $\beta \in \mathscr{H}$ then for any $\alpha \in \mathscr{G}$ $\mathfrak{s}(\alpha) \operatorname{lt}\left(\overline{\beta}\right) = 0 \leq \mathfrak{s}(\beta) \operatorname{lt}(\overline{\alpha})$.

Lemma 7.7. *If* **RB** *uses $\trianglelefteq_{rat}$ as rewrite order then there exists no singular top $\mathfrak{s}$-reducible element in $\mathscr{G}$.*

Proof. **RB** only regular $\mathfrak{s}$-reduces an S-pair in a non-syzygy signature T if $\mathscr{G}$ is not already a rewrite basis in signature T (see Lemma 7.4), i.e. only if the canonical rewriter $a\alpha$ in T is regular top $\mathfrak{s}$-reducible. Let $b\beta$ be such a regular $\mathfrak{s}$-reducer of $a\alpha$. Let $a\alpha - b\beta$ regular $\mathfrak{s}$-reduce to γ. Assume there exists $\delta \in \mathscr{G}$ such that $\mathfrak{s}(d\delta) = T$ and $\operatorname{lt}\left(\overline{d\delta}\right) = \operatorname{lt}(\overline{\gamma})$. Since $a\alpha$ is the canonical rewriter in signature T w.r.t. $\trianglelefteq_{\text{rat}}$ it holds that

$$\operatorname{lt}\left(\overline{d\delta}\right) \geq \operatorname{lt}\left(\overline{a\alpha}\right) > \operatorname{lt}(\overline{\gamma}).$$

This contradicts the existence of such an element $\delta \in \mathscr{G}$. □

Corollary 7.3. *Using $\trianglelefteq_{rat}$ as rewrite order* **RB** *computes a minimal signature Gröbner basis.*

Proof. Clear by Lemma 7.7. See also Section 3.3 in Eder and Roune (2013) for more details. □

The question is now if there exist examples where **RB** using $\trianglelefteq_{\text{add}}$ computes a signature Gröbner basis with more elements than the one achieved by **RB** using $\trianglelefteq_{\text{rat}}$.

$\alpha_i \in \mathscr{G}$	reduced from	$\operatorname{lt}(\overline{\alpha_i})$	$\mathfrak{s}(\alpha_i)$
α_1	$\boldsymbol{e}_1$	yz	$\boldsymbol{e}_1$
α_2	$\boldsymbol{e}_2$	xy	$\boldsymbol{e}_2$
α_3	$\operatorname{spair}(\alpha_2, \alpha_1) = z\alpha_2 - x\alpha_1$	xt^2	$z\boldsymbol{e}_2$
α_4	$\boldsymbol{e}_3$	x^2z	$\boldsymbol{e}_3$
α_5	$\operatorname{spair}(\alpha_4, \alpha_2) = y\alpha_4 - xz\alpha_2$	y^2t^2	$y\boldsymbol{e}_3$
α_6	$\operatorname{spair}(\alpha_4, \alpha_3) = t^2\alpha_4 - xz\alpha_3$	z^3t^2	$t^2\boldsymbol{e}_3$
α_7	$\operatorname{spair}(\alpha_6, \alpha_1) = y\alpha_6 - z^2t^2\alpha_1$	y^2t^4	$yt^2\boldsymbol{e}_3$

Fig. 6. Computations for **RB** in Example 7.2.

Example 7.2. Let $\mathscr{K}$ be the finite field with 7 elements and let $\mathscr{R} = \mathscr{K}[x, y, z, t]$. Let $<$ be the graded reverse lexicographical monomial order which we extend to $<_{\text{pot}}$ on $\mathscr{R}^3$. Consider the input ideal I generated by $f_1 = yz - 2t^2$, $f_2 = xy + t^2$, and $f_3 = x^2z + 3xt^2 - 2yt^2$. We present the calculations done by **RB** using $\trianglelefteq_{\text{add}}$ in Fig. 6.

RB with $\trianglelefteq_{\text{rat}}$ regular $\mathfrak{s}$-reduce the same S-pairs except the last one: In signature $yt^2\boldsymbol{e}_3$ we have $y\alpha_6, t^2\alpha_5 \in \mathscr{C}_{yt^2\boldsymbol{e}_3}$. $\trianglelefteq_{\text{add}}$ prefers $y\alpha_6$ over $t^2\alpha_5$, thus the S-pair $y\alpha_6 - z^2t^2\alpha_1$ is handled. $\trianglelefteq_{\text{rat}}$ on the other hand has $t^2\alpha_5$ as canonical rewriter in signature $yt^2\boldsymbol{e}_3$ as $\operatorname{lt}\left(\overline{t^2\alpha_5}\right) = y^2t^4 < yz^3t^2 = \operatorname{lt}\left(\overline{y\alpha_6}\right)$. With this choice no S-pair in signature $yt^2\boldsymbol{e}_3$ is handled and thus **RB** terminates.

Note that the canonical rewriter in signature $yt^2\boldsymbol{e}_3$ w.r.t. $\trianglelefteq_{\text{rat}}$ is not regular top $\mathfrak{s}$-reducible. So by Lemma 7.4 **RB** does not reduce any S-pair in this signature. $\trianglelefteq_{\text{add}}$ chooses its canonical rewriter $y\alpha_6$ wrong in the sense that $y\alpha_6$ can be further reduced, but only until it reaches $t^2\alpha_5$. Whereas this computation is important for **RB** in order to compute a rewrite basis w.r.t. $\trianglelefteq_{\text{add}}$, it is not needed to achieve a signature Gröbner basis for I.

In particular, the following holds:

Lemma 7.8. *Let $\preceq = \preceq_{\mathfrak{s}}$. If $\mathscr{G}$ is a rewrite basis up to signature T w.r.t. $\trianglelefteq_{\text{add}}$ then $\mathscr{G}$ is a rewrite basis up to signature T w.r.t. $\trianglelefteq_{\text{rat}}$.*

Proof. Let $\mathscr{G}$ be a rewrite basis up to signature T w.r.t. $\trianglelefteq_{\text{add}}$. Assume $\mathscr{G}$ is not a rewrite basis up to signature T w.r.t. $\trianglelefteq_{\text{rat}}$. Then there exists some $\alpha \in \mathscr{G}$ and a term $a \in \mathscr{R}$ such that $a\alpha$ is the canonical rewriter in $\mathfrak{s}(a\alpha) < T$ and $a\alpha$ is regular top $\mathfrak{s}$-reducible w.r.t. $\mathscr{G}$. Thus there exists $\beta \in \mathscr{G}$ and a term $b \in \mathscr{R}$ such that $a\alpha - b\beta$ is an S-pair. This S-pair can possibly be further regular $\mathfrak{s}$-reduced to $\gamma \in \mathscr{R}^m$.

(a) If $\gamma \in \mathscr{H}$ then γ is the canonical rewriter in $\mathfrak{s}(a\alpha)$, a contradiction to our assumption.
(b) If $\operatorname{lt}(\overline{\gamma}) \neq 0$ then $\operatorname{lt}(\overline{\gamma}) < \operatorname{lt}\left(\overline{a\alpha}\right)$. If there exists $\delta \in \mathscr{G}$ and a term $t \in \mathscr{R}$ such that $t\delta = \gamma$ then $t\delta$ would be the canonical rewriter in $\mathfrak{s}(a\alpha)$. This is again a contradiction to our choice of $a\alpha$.

In particular, there exists no δ that has been added to $\mathscr{G}$ after α with $\mathfrak{s}(t\delta) = \mathfrak{s}(a\alpha)$ and $\operatorname{lt}\left(\overline{t\delta}\right) < \operatorname{lt}\left(\overline{a\alpha}\right)$. Since $\mathscr{G}$ is a signature Gröbner basis up to signature T (since we assume that $\mathscr{G}$ is a rewrite basis up to T w.r.t. $\trianglelefteq_{\text{add}}$) and since $\mathfrak{s}(t\delta) = \mathfrak{s}(a\alpha)$ it holds that $\operatorname{lt}\left(\overline{t\delta}\right) = \operatorname{lt}\left(\overline{a\alpha}\right)$ due to Lemma 4.2. But in this case $t\delta$ would also have been the canonical rewriter in $\mathfrak{s}(a\alpha)$ w.r.t. $\trianglelefteq_{\text{rat}}$ which again contradicts our assumption. Thus there exists no $\delta \in \mathscr{G}$ added after α to $\mathscr{G}$ with $\mathfrak{s}(t\delta) = \mathfrak{s}(a\alpha)$. Thus $a\alpha$ is the canonical rewriter in $\mathfrak{s}(a\alpha)$ w.r.t. $\trianglelefteq_{\text{add}}$. Since it is regular $\mathfrak{s}$-reducible by β this is a contradiction to the assumption that $\mathscr{G}$ is a rewrite basis up to signature T w.r.t. $\trianglelefteq_{\text{add}}$. □

7.4. Characterization of rewrite bases and signature Gröbner bases

As we have seen, different rewrite orders lead to different rewrite bases and thus signature Gröbner bases. In fact there is a quite natural description of rewrite bases.

In general, one can characterize rewrite basis by taking only S-pairs into account.

Theorem 7.2. *$\mathscr{G}$ is a rewrite basis if and only if it holds that for all S-pairs $a\alpha - b\beta$ with $\alpha, \beta \in \mathscr{G}$ and $\mathfrak{s}(a\alpha) > \mathfrak{s}(b\beta)$ $a\alpha$ is not the canonical rewriter in $\mathfrak{s}(a\alpha)$.*

Proof.

"⇒" Assume that there exists an S-pair $a\alpha - b\beta$ such that $\mathfrak{s}(a\alpha) > \mathfrak{s}(b\beta)$ and $a\alpha$ is the canonical rewriter in $\mathfrak{s}(a\alpha)$. Then the canonical rewriter in $\mathfrak{s}(a\alpha)$ is regular top $\mathfrak{s}$-reducible w.r.t. $\mathscr{G}$, namely by $b\beta$. Then $\mathscr{G}$ is not a rewrite basis in $\mathfrak{s}(a\alpha)$, thus $\mathscr{G}$ is not a rewrite basis.

"⇐" Assume that $\mathscr{G}$ is not a rewrite basis. Then there exists an $\alpha \in \mathscr{G}$ such that $a\alpha$ is the canonical rewriter in $\mathfrak{s}(a\alpha)$ and $a\alpha$ is regular top $\mathfrak{s}$-reducible. Thus there exists $\beta \in \mathscr{G}$ and $b \in \mathscr{R}$ such that $\operatorname{lt}\left(\overline{a\alpha}\right) = \operatorname{lt}\left(\overline{b\beta}\right)$ and $\mathfrak{s}(a\alpha) > \mathfrak{s}(b\beta)$, and $a\alpha - b\beta$ is an S-pair of elements of $\mathscr{G}$. □

Any rewrite basis is a signature Gröbner basis by Lemma 7.1. Still, we do not get the same statement as in Theorem 7.2 for signature Gröbner bases. There, we have to restrict ourselves to a specific rewrite order, namely $\trianglelefteq_{\text{rat}}$. Note that this characterization is due to Gao, Volny, and Wang. It was first presented for the **GVW** algorithm in Gao et al. (2016). There the authors use a different approach using strong Gröbner bases. Here we present the corresponding statement in our setting. In Section 11.5 we introduce strong Gröbner bases (Definition 11.1) and present the theorem from Gao et al. (2016) (Theorem 11.1) together with a translation to rewrite bases.

Theorem 7.3. *$\mathscr{G}$ is a signature Gröbner basis if and only if it holds that for all S-pairs $a\alpha - b\beta$ with $\alpha, \beta \in \mathscr{G}$ and $\mathfrak{s}(a\alpha) > \mathfrak{s}(b\beta)$ $a\alpha$ is not the canonical rewriter in $\mathfrak{s}(a\alpha)$ w.r.t. $\trianglelefteq_{rat}$.*

Proof.

"⇒" Let $a\alpha - b\beta$ be an S-pair of $\mathscr{G}$ such that $\mathfrak{s}(a\alpha) > \mathfrak{s}(b\beta)$. Since $\mathscr{G}$ is a signature Gröbner basis, $a\alpha$ $\mathfrak{s}$-reduces (and thus, top $\mathfrak{s}$-reduces) to zero. There exists at least one regular $\mathfrak{s}$-reduction, say with $b\beta$. Doing all possible further regular $\mathfrak{s}$-reductions we reach some $\gamma \in \mathscr{R}^m$ such that $\mathfrak{s}(\gamma) = \mathfrak{s}(a\alpha)$ and $\operatorname{lt}(\overline{\gamma}) < \operatorname{lt}\left(\overline{a\alpha}\right)$. If $\gamma \in \mathscr{H}$ then γ is the canonical rewriter in signature $\mathfrak{s}(a\alpha)$ w.r.t. $\trianglelefteq_{\text{rat}}$. If $\operatorname{lt}(\overline{\gamma}) \neq 0$ then $\delta \in \mathscr{G}$ has to exist such that there is some term $t \in \mathscr{R}$ such that $t\delta = \gamma$ since $\mathscr{G}$ is a signature Gröbner basis. Thus, $t\delta$ is then the canonical rewriter in signature $\mathfrak{s}(a\alpha)$ w.r.t. $\trianglelefteq_{\text{rat}}$.

"⇐" Assume that $\mathscr{G}$ is not a signature Gröbner basis. Then there exists some $\alpha \in \mathscr{R}^m$ such that α does not $\mathfrak{s}$-reduce to zero w.r.t. $\mathscr{G}$ and $\mathfrak{s}(\alpha) \neq 0$ is minimal with this property. Let $b\beta$ be the canonical rewriter in $\mathfrak{s}(\alpha)$. $b\beta$ is not regular top $\mathfrak{s}$-reducible: Otherwise there would exist $\gamma \in \mathscr{G}$ and $c \in \mathscr{R}$ such that $\mathfrak{s}(b\beta - c\gamma) = \mathfrak{s}(b\beta)$ and $\operatorname{lt}\left(\overline{b\beta - c\gamma}\right) < \operatorname{lt}\left(\overline{b\beta}\right)$, so that $b\beta - c\gamma$ would be the canonical rewriter in signature $\mathfrak{s}(b\beta)$. Now we compute $\alpha - b\beta$ and note that $\operatorname{lt}(\overline{\alpha}) \neq \operatorname{lt}\left(\overline{b\beta}\right)$ as otherwise α would be top $\mathfrak{s}$-reducible. Next we can subtract, if it exists, the multiple $c\gamma$ of $\gamma \in \mathscr{H}$ for some $c \in \mathscr{R}$ such that $\mathfrak{s}(c\gamma) = \mathfrak{s}(\alpha - b\beta)$. This step we can do inductively until no such $\gamma \in \mathscr{H}$ exists anymore. We call the resulting element δ. Since $\mathfrak{s}(\delta) < \mathfrak{s}(\alpha)$ there exist elements in $\mathscr{G}$ that top $\mathfrak{s}$-reduce δ. Let $\epsilon \in \mathscr{G}$ and $e \in \mathscr{R}$ such that $\operatorname{lt}\left(\overline{e\epsilon}\right) = \operatorname{lt}\left(\overline{\delta}\right)$. Two cases are possible:

(a) If $\operatorname{lt}(\overline{\alpha}) > \operatorname{lt}\left(\overline{b\beta}\right)$ then $e\epsilon$ is a regular top $\mathfrak{s}$-reducer of α, a contradiction.

(b) If $\operatorname{lt}(\overline{\alpha}) < \operatorname{lt}\left(\overline{b\beta}\right)$ then $e\epsilon$ is a regular top $\mathfrak{s}$-reducer of $b\beta$, also a contradiction.

Thus, such an element $\alpha \in \mathscr{R}^m$ cannot exist and $\mathscr{G}$ has to be a signature Gröbner basis. □

The question is now: What goes wrong assuming $\trianglelefteq_{\text{add}}$ instead of $\trianglelefteq_{\text{rat}}$ in the above proof? Let us look again at Example 7.2. $\mathscr{G}_1 = \{\alpha_1, \ldots, \alpha_6\}$ and $\mathscr{G}_2 = \{\alpha_1, \ldots, \alpha_7\}$ are both signature Gröbner bases, the first computed w.r.t. $\trianglelefteq_{\text{rat}}$, the second one w.r.t. $\trianglelefteq_{\text{add}}$. If we take $\mathscr{G}_1$ as signature Gröbner basis and try to prove the "⇒" direction of Theorem 7.3 while exchanging $\trianglelefteq_{\text{rat}}$ by $\trianglelefteq_{\text{add}}$ in the theorem's statement, we cannot conclude as intended: $y\alpha_6 - z^2t^2\alpha_1$ is an S-pair of elements of $\mathscr{G}_1$ and $\mathfrak{s}(y\alpha_6) > \mathfrak{s}(z^2t^2\alpha_1)$. Still, $y\alpha_6$ is the canonical rewriter in $\mathfrak{s}(y\alpha_6)$ w.r.t. $\trianglelefteq_{\text{add}}$ as $t^2\alpha_5 \trianglelefteq_{\text{add}} y\alpha_6$. So the statement of Theorem 7.3 does not hold. Clearly, assuming $\trianglelefteq_{\text{rat}}$ as rewrite order the statement stays true, since there $y\alpha_6 \trianglelefteq_{\text{rat}} t^2\alpha_5$.

We conclude this section with the following summary: As we have seen **RB** is mainly parametrized by three properties:

(a) the monomial order $<$ and its extension to $\mathscr{R}^m$,
(b) the pair set order $\preceq$ on $\mathscr{P}$, and
(c) the rewrite order $\trianglelefteq$.

We see that even though there are so many different notions of signature-based Gröbner algorithms in the literature, all those implementations boil down to variations of two of the above mentioned three orders: All known algorithms have in common to use $\preceq_{\mathfrak{s}}$ on $\mathscr{P}$.[3]

Remark 7.4. Note that such attempts of generalizing the description of signature-based Gröbner basis algorithms have already been done, for example, in Huang (2010), Sun and Wang (2011a), Pan et al. (2012, 2013). As we have already pointed out in the introduction of this section all of these characterizations are similar and included in our attempt using **RB**. The difference in notation are rather obvious (see also sections 8–11), thus we relinquish to give comparisons further than the ones depicted already in Sections 6 and 7.

Next we discuss known and efficient instantiations of signature-based Gröbner basis algorithms. **RB** can be seen as a common core these algorithms share that we will use for comparisons. Note that all algorithms described in the following can be implemented with any compatible extension to the monomial order. When algorithms were initially presented with a fixed module monomial order we take care of this. Still, the only real difference of the implements boils down to the rewrite orders used.

8. Signature-based Gröbner basis algorithms using $\trianglelefteq_{\text{add}}$

In 2002 Faugère presented the **F5** algorithm (Faugère, 2002). This was the first publication of a signature-based Gröbner basis algorithm and introduced the notion of a signature.

In Eder and Roune (2013) the connection between **RB** and **F5** is already given, so we give a short review and refer for details to that paper. **F5**, as presented in Faugère (2002) uses $<_{\text{pot}}$ as extension of the underlying monomial order $<$.[4]

Remark 8.1. In its initial presentation in Faugère (2002), **F5** uses $<_{\text{pot}}$, so it computes incrementally a Gröbner basis for $\langle f_1, \ldots, f_i \rangle$ for increasing i. For each such index i the algorithm stores a list of so-called “rewrite rules”: Rule$_i$. The S-pairs are first taken by minimal possible degree $d := \deg(\overline{a\alpha})$ for $a\alpha$ being a generator of an S-pair. Once this choice is done this list of S-pairs, denoted by $\mathscr{P}_d$ is handled by subalgorithm Spol. There S-pairs are checked by the criteria and new rewrite rules are added to the end of the list Rule$_i$. Once this step is done, the remaining S-pairs in $\mathscr{P}_d$ are handed to the subalgorithm Reduction. Not until this point the S-pairs in $\mathscr{P}_d$ are sorted by increasing signature. This leads to the following effects:

(a) If the input is homogeneous, **F5** reduces S-pairs by increasing signature, but the rewrite rules are *not* sorted by increasing signature.

[3] There is some slight difference in the original presentation of **F5** in Faugère (2002) which is discussed in Property 8.1 (b).

[4] Strictly speaking this is not completely true, **F5** as presented in Faugère (2002) uses $<_{\text{pot'}}$ defined by $a\boldsymbol{e}_i <_{\text{pot'}} b\boldsymbol{e}_j$ if and only if $i > j$ or $i = j$ and $a < b$. The only difference is to prefer the element of lower index instead of the one of higher index. In order to unify notations we assume in the following that **F5** means “**F5** uses $<_{\text{pot}}$ as module monomial order”.

(b) If the input is inhomogeneous then **F5** need not even reduce S-pairs by increasing signatures as it is pointed out in Eder (2013a). Note that this behavior is still covered by **RB** using a corresponding pair set order $\preceq \neq \preceq_{\mathfrak{s}}$. Still, as discussed in sections 5 and 7 the best possible pair set order is $\preceq_{\mathfrak{s}}$ and it is shown in Eder (2013a) that **F5** can easily be equipped with it. As pointed out already in Remark 1 in Eder and Roune (2013), **F5**'s presentation in Faugère (2002) is restricted to homogeneous regular sequences whereas benchmarks are also given for other systems.

The fact about not handling S-pairs by increasing signatures we describe in more detail in Property 8.1 (b). The problem of ordering the rewrite rules is more difficult: As described in Faugère (2002), **F5** might not use $\trianglelefteq_{\text{add}}$ as rewrite order: For **F5** the canonical rewriter in signature T is the element in Rule_i which was added last. But at the time of concatenation the S-pairs are not sorted by increasing signature! So the following situation can happen: Assume we have two S-pairs in degree d with signatures $z\boldsymbol{e}_i$ and $x\boldsymbol{e}_i$. We can assume that in $\mathscr{P}_d$ they are ordered like $[\ldots, x\boldsymbol{e}_i, \ldots, z\boldsymbol{e}_i, \ldots]$. Let us assume that both S-pairs are not rewritable, so we reduce both. Now, after $\mathscr{P}_d$ is sorted by increasing signature, **F5** first reduces the S-pair with signature $z\boldsymbol{e}_i$ to α, and later on the one with signature $x\boldsymbol{e}_i$ to β. Generating new S-pairs we could have two S-pairs in $\mathscr{P}_{d+2}$ with signature $xyz\boldsymbol{e}_i$: $\operatorname{spol}(\alpha, \gamma)$ and $\operatorname{spol}(\beta, \delta)$. In this situation, **F5** would remove $\operatorname{spol}(\beta, \delta)$ and keep $\operatorname{spol}(\alpha, \gamma)$ since the signature $z\boldsymbol{e}_i$ was added to Rule_i after $x\boldsymbol{e}_i$ had been added. So in our notation α is the canonical rewriter in signature $xyz\boldsymbol{e}_i$. Clearly, using $\trianglelefteq_{\text{add}}$ β is the canonical rewriter in $xyz\boldsymbol{e}_i$.

Since β was computed after α from the algorithm's point of view β might the better element. So it makes sense to optimize **F5** to use $\trianglelefteq_{\text{add}}$.

In the following we assume that **F5** uses $\trianglelefteq_{\text{add}}$ as rewrite order, then the only difference left from its original description is now the fact that **F5** checks the possible $\mathfrak{s}$-reducers $b\beta$ of an element α if they are not syzygy and not rewritable.

Lemma 8.1 *(Lemma 15 in Eder and Roune (2013)). Let $\alpha \in \mathscr{R}^m$, let t be a term of $\overline{\alpha}$ and let $\mathscr{G}$ be a rewrite basis up to signature $\mathfrak{s}(\alpha)$. Then t is regular $\mathfrak{s}$-reducible if and only if it is reducible in* **F5**.

Proof. Follows also from Lemma 7.3. □

So from Lemma 8.1 it follows that checking possible reducers by **Rewritable** in **RB** does not change the algorithm's behavior and is thus optional. In Section 13 we see that the idea of checking the $\mathfrak{s}$-reducers by the criteria comes from a linear algebra point of view.

Let us underline the following characteristics of **F5**.

Algorithmic Property 8.1.

(a) Note that, even so we assume the optimization of **F5**'s rewrite order as described in Remark 8.1, **F5** still does not completely implement $\trianglelefteq_{\text{add}}$ but a slightly different rewrite order: The requirement $\alpha \trianglelefteq \beta$ whenever $\beta \in \mathscr{H}$ from $\trianglelefteq_{\text{add}}$ is relaxed to $\beta = \operatorname{ksyz}\left(\boldsymbol{e}_i, \boldsymbol{e}_j\right)$ for $1 \leq i < j \leq m$. Thus non-Koszul syzygies in $\mathscr{H}$ have the same priority as elements in $\mathscr{G}$. The idea to improve computations by using zero reductions directly instead was introduced first in an arXiv preprint of Arri and Perry (2011) by Arri and Perry in 2009 as well as in Gao et al. (2010a) by Gao, Guan and Volny.

(b) Note that in Faugère (2002) the **F5** algorithm is described in the vein of using linear algebra for the reduction steps (see Section 13 for more details). Instead of ordering the pair set by increasing signatures it is ordered by increasing degree of the corresponding S-polynomial. A subset $\mathscr{P}_d$ of S-pairs at minimal given degree d is then handled by the Reduction procedure. There, all these S-pairs (corresponding to degree d polynomials) are sorted by increasing signature. As already discussed in Eder (2013a), for homogeneous input this coincides with using $\preceq_{\mathfrak{s}}$ since then the degree of the polynomial part and the degree of the signature are the same. For inhomogeneous input **F5**'s attempt might not coincide with $\preceq_{\mathfrak{s}}$. Galkin (2014) has given a proof for termination of **F5** taking care of this situation. Note that in such a situation one might either prefer to use $\preceq_{\mathfrak{s}}$ (as pointed out in Eder, 2013a) or saturate resp. desaturate the elements during the computation of the algorithm.

(c) Furthermore, thinking in terms of linear algebra also explains why in Faugère (2002) higher signature reductions lead to new S-pairs which are directly added to the ToDo list in subalgorithm TopReduction and not prolonged to the situation when a new element is added $\mathscr{G}$ as it is done in **RB**: Assuming homogeneous input, in a Macaulay matrix M_d (see, for example, Section 3) all corresponding rows are already stored. Thus a higher signature S-pair (in **RB** et al. due to single polynomial $\mathfrak{s}$-reduction prolonged to a later step) corresponds to a reduction of a row by some other one below. All possible S-pairs of degree d are handled at once thus one can directly execute the new S-pair without generating it later on.

Clearly, the **F5** criterion and the Rewritten criterion are just special cases of the syzygy criterion (Lemma 6.1) and the singular criterion (Lemma 6.2), respectively. For even more details on how to translate notions like “canonical rewriter” to **F5** equivalents like “rewrite rules” we refer to Eder and Roune (2013) Section 5.

Moreover, **F5** implements the $\mathfrak{s}$-reduction process different to the description in **RB**: Instead of prolonging an $\mathfrak{s}$-reduction $\alpha - b\beta$ with a reducer $b\beta$ of signature $\mathfrak{s}(b\beta) > \mathfrak{s}(\alpha)$ to the generation of the S-pair $b\beta - \alpha$ later on, **F5** directly adds $b\beta - \alpha$ to the todo list of the current degree in Reduction. Assuming homogeneous input this makes sense. Again, we see in Section 13 that this is coming from an **F4**-style implementation of the $\mathfrak{s}$-reduction process.

In the last decade several optimized algorithms were presented. Using **RB** we can easily categorize them.

Specifications 8.1. In Faugère (2002) three variations of **F5** are mentioned shortly without going into detail about their modifications:

(a) **F5’** denotes an algorithm similar to **F5R** (see Section 8.1) resp. **F5C** (see Section 8.2): For inhomogeneous input one can optimize computations by homogenizing the computations of the intermediate Gröbner basis G_i for $\langle f_1, \ldots, f_i \rangle$. Before adding the homogenized f_{i+1} one dehomogenizes G_i and interreduces G_i^{deh} to B_i. This B_i can then be used for checks with the syzygy criterion as well as for reduction purposes. We refer to sections 8.1 and 8.2 for details on signature handling in this situation.

(b) **F5”** denotes an algorithm using $<_{\text{d-pot}}$ as compatible module monomial order. Thus, instead of an incremental computation w.r.t. the initial generators $f_1, \ldots, f_m$ the algorithm handles elements by increasing degree. Note that in case of regular input **F5”** computes no zero reduction, whereas this is possible for $<_{\text{lt-pot}}$.

(c) **MatrixF5** is an algorithm (when a degree bound is given) which uses linear algebra for reduction purposes is described in Section 3 in detail.

Note that besides the algorithms presented in the following there are even more publications about optimizations and generalizations of the **F5** algorithm for computing Gröbner bases, for example, see Faugère and Svartz (2012, 2013), Faugère et al. (2013), Sun and Wang (2010, 2013a). Although the main results in these publications are presented for **F5**, they do not depend on the Gröbner basis algorithm used. Here we are giving a survey especially for signature-based Gröbner basis algorithms, thus taking care of not signature-based tailored research is out of scope of this publication.

Moreover, there are first works in using signature-based criteria for computing involutive bases (Gerdt and Hashemi, 2013, 2013).

8.1. *F5R – improved lower-index $\mathfrak{s}$-reduction*

In 2005 Stegers reviewed **F5** in Stegers (2007). There he introduced a new algorithm improving the reduction process. Due to the incremental structure of **RB** when using $<_{\text{pot}}$ one first computes a signature Gröbner basis for $\langle f_1, f_2\rangle$, then for $\langle f_1, f_2, f_3\rangle$, and so on. Since the intermediate bases need not be minimal Stegers suggested to use in step k of the algorithm not $\mathscr{G}_{k-1}$ for reduction purposes. Instead it is preferable to reduce the corresponding Gröbner basis $G_{k-1} = \left\{\overline{\alpha} \mid \alpha \in \mathscr{G}_{k-1}\right\}$ to the reduced Gröbner basis B_{k-1} for $\left\langle f_1, \ldots, f_{k-1}\right\rangle$. Since for all elements handled by **RB** in iteration step k the signature has an index k and all elements in $\mathscr{G}_{k-1}$ have signature index at most $k-1$ $\mathfrak{s}$-reductions are always allowed when using $<_{\text{pot}}$ and the signatures need not be checked.

Note that B_{k-1} is only used for the reduction purposes, new S-pairs are still generated using elements in $\mathscr{G}_{k-1}$ since otherwise the signatures would not be correct.

8.2. *F5C – improved S-pair generation*

Based on Stegers' idea, Eder and Perry presented in 2009 the **F5C** algorithm in Eder and Perry (2010). Whereas **F5R** uses the reduced Gröbner basis B_{k-1} for $\left\langle f_1, \ldots, f_{k-1}\right\rangle$ only for reduction purposes, **F5C** extends this to the generation of new S-pairs in iteration step k.

Once **RB** finishes computing $\mathscr{G}_{k-1}$ one reduces the corresponding Gröbner basis $\overline{\mathscr{G}_{k-1}}$ to B_{k-1} as described above. Let $B_{k-1} := \{g_1, \ldots, g_{m'}\}$, then one introduces $\mathscr{G}'_{k-1} = \{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_{m'}\}$. Moreover, one has to redefine the homomorphism $\alpha \mapsto \overline{\alpha}$ to go from $\mathscr{R}^{m'}$ to $\mathscr{R}$ by sending $\boldsymbol{e}_i$ to g_i for all $i \in \left\{1, \ldots, m'\right\}$.

Starting iteration step k, **RB** now computes the signature Gröbner basis for $\langle g_1, \ldots, g_{m'}, f_k\rangle$. Of course, at that point another extension of the homomorphism $\alpha \mapsto \overline{\alpha}$ has to be done, since now we are mapping $\mathscr{R}^{m'+1} \to \mathscr{R}$: We define that $\overline{\boldsymbol{e}_{m'+1}} := f_k$.

It is shown in Theorem 32 and Corollary 33 of Eder and Perry (2010) that with this resetting of the signatures the number of useless $\mathfrak{s}$-reductions is not increased, but instead the number of S-pairs generated in step k is decreased.

Specifications 8.2.

(a) Due to Property 8.1 (a) one also wants to implement **F5C** using $\trianglelefteq_{\text{add}}$ in order to use zero reductions directly. In 2011, Eder and Perry denoted this algorithm **F5A** in Eder and Perry (2011).

(b) In Eder (2013b) Eder improves the idea of **F5C** slightly: By symbolically generating S-pairs of elements in $\mathscr{G}'_{k-1}$ (they all already reduce to zero) signatures useful for discarding S-pairs in iteration step k can be made available a bit earlier. Thus, in terms of **RB**, $\mathscr{H}$ is initialized not only with the signatures of the Koszul syzygies but also with the signatures of other, already known syzygies. The idea presented there can be used in any incremental signature-based Gröbner basis algorithm. The corresponding algorithms are denoted, for example, **iF5C** and **iG2V**.

8.3. *Extended F5 criteria*

In 2010, Ars and Hashemi published (Ars and Hashemi, 2010) in which they generalized the **F5** criterion and the Rewritten criterion in the sense of using different extensions of the monomial order $<$ on $\mathscr{R}^m$. These algorithms are achieved by using **RB** not with $<_{\text{pot}}$ but one of the following two orders proposed in Ars and Hashemi (2010).

Definition 8.1. Let $<$ be a monomial order on $\mathscr{R}$ and let ae_i, be_j be two module monomials in $\mathscr{R}^m$.

(a) $a\boldsymbol{e}_i <_1 b\boldsymbol{e}_j$ if and only if[5] either

$$a \operatorname{lt}\left(\overline{\boldsymbol{e}_j}\right) < b \operatorname{lt}\left(\overline{\boldsymbol{e}_i}\right) \quad \text{or}$$
$$a \operatorname{lt}\left(\overline{\boldsymbol{e}_j}\right) = b \operatorname{lt}\left(\overline{\boldsymbol{e}_i}\right) \quad \text{and} \quad \operatorname{lt}\left(\overline{\boldsymbol{e}_j}\right) < \operatorname{lt}\left(\overline{\boldsymbol{e}_i}\right).$$

[5] Note that for $<_1$ to be a total order we need to ensure that $\operatorname{lt}\left(\overline{\boldsymbol{e}_i}\right) \neq \operatorname{lt}\left(\overline{\boldsymbol{e}_j}\right)$ whenever $i \neq j$. Having the input ideal $I = \langle f_1, \ldots, f_m\rangle$ this can be achieved by an interreduction of the f_is before entering **RB**.

(b) $ae_i <_2 be_j$ if and only if either

$$\begin{aligned} \deg\left(\overline{a\boldsymbol{e}_i}\right) &< \deg\left(\overline{b\boldsymbol{e}_j}\right) \quad \text{or} \\ \deg\left(\overline{a\boldsymbol{e}_i}\right) &= \deg\left(\overline{b\boldsymbol{e}_j}\right) \ \text{ and } \ a < b \quad \text{or} \\ \deg\left(\overline{a\boldsymbol{e}_i}\right) &= \deg\left(\overline{b\boldsymbol{e}_j}\right) \ \text{ and } \ a = b \ \text{ and } \ i < j. \end{aligned}$$

Ars and Hashemi implemented the original **F5** algorithm and their algorithms in the computer algebra system Magma and give timings for several Gröbner basis benchmarks. Their algorithms seem to be more efficient than the original **F5** algorithm in most of the examples. Still there exist input, for example the Schrans-Troost benchmark, for which $<_{\text{pot}}$ seems to be more efficient. Using a framework like **RB** such behavior can be tested easily.

Algorithmic Property 8.2. Note that the extended **F5** algorithm presented in Ars and Hashemi (2010) is still using $\trianglelefteq_{\text{add}}$: In Definition 2.4, Ars and Hashemi state the *Extended Rewritten Criterion*. There elements are compared via "…labeled polynomial $r'(\ldots)$ computed before r …".

9. Exploiting algebraic structures

In this section we present algorithms that use knowledge of underlying algebraic structures in order to improve the computations. Note that there exist more algorithms doing this besides the three we are discussing here, see, for example, Faugère and Svartz (2012, 2013), Faugère et al. (2013) (see also Fig. 1). The improvements in these algorithms are not specific to signature-based Gröbner basis algorithms, thus we waive to discuss them here.

It is clear that in the future a lot more improvements in this direction can be expected. Exploiting algebraic structures helps to find more syzygies on the one hand and to increase the independence of polynomials on the other hand. Both has a positive influence on the computation of (signature) Gröbner bases.

9.1. *F5/2 – improved computations over* $\mathbb{F}_2$

An easy way to improve **F5**'s performance over small finite fields is to add the field equations to $\mathscr{H}$. When breaking the first hidden field equations (HFE) challenge in 2003 (Faugère and Joux, 2003) the algorithm **F5/2** was used which adds to $f_1, \ldots, f_m$ the field equations $x_i^2 - x_i = 0$ in $\mathbb{F}_2$. With this the rewritable signature criterion is more powerful since Koszul syzygies generated by those supplementary equations have low signatures. The HFE challenge consists of 80 equations in degree 2. A Gröbner basis computation of such a system was intractable beforehand.

9.2. *An **F5** algorithm for bihomogeneous ideals generated by polynomials of bidegree* $(1, 1)$

In 2012 Faugère, Safey El-Din and Spaenlehauer published an algorithm dedicated to multihomogeneous, in particular, bihomogeneous systems generated by bilinear polynomials (Faugère et al., 2011). The main result is to exploit the algebraic structure of bilinear systems to enlarge $\mathscr{H}$.

In Corollary 7.1 we see that **RB** and thus also **F5** computes no reduction to zero if the input sequence is regular. Whereas a randomly chosen homogeneous polynomial system is regular, this is not the case for multihomogeneous polynomial systems. Those systems appear, for example, in cryptography or coding theory. Due to the non-regularity **F5** does not remove all zero reductions.

Let $f_1, \ldots, f_m \in \mathscr{K}[x_0, \ldots, x_{n_x}, y_0, \ldots, y_{n_y}]$ be bilinear polynomials, let F_i denote the sequence $f_1, \ldots, f_i$ and let I_i denote the ideal $\langle F_i \rangle$. The main result is that the kernel of the Jacobian matrices $\text{jac}_x(F_i)$ and $\text{jac}_y(F_i)$ w.r.t. $x_0, \ldots, x_{n_x}$ and $y_0, \ldots, y_{n_y}$, respectively, correspond to those reductions to zero **F5** does not detect. In general, all elements in these kernels are vectors of maximal minors of the corresponding Jacobian matrices.

Assuming the incremental structure of **F5** by using $<_{\mathrm{pot}}$ it is shown that the ideal $I_{i-1} : f_i$ is spanned by I_{i-1} and the maximal minors of $\mathrm{jac}_x(F_{i-1})$ (for $i-1 > n_y$) and $\mathrm{jac}_y(F_{i-1})$ (for $i-1 > n_x$). The lead ideal of $I_{i-1} : f_i$ corresponds to the zero reductions associated to f_i. In order to get rid of them one needs to get results for the ideals generated by the maximal minors of the Jacobian matrices. In Faugère et al. (2011) it is shown that in general Gröbner bases for these ideals w.r.t. the graded reverse lexicographical order are linear combinations of the generators. Thus, once a Gröbner basis of I_{i-1} is known (which we can assume due to the incremental structure of **F5**) one can efficiently compute a Gröbner basis of $I_{i-1} : f_i$. It follows that for generic bilinear systems this variant of **F5** does not compute any zero reduction.

It follows that for **RB** all one has to do is to add the computation of the maximal minors of the Jacobian matrices and add the corresponding syzygies resp. signatures to $\mathscr{H}$ in Line 3 of Algorithm 3.

9.3. An **F5** algorithm for SAGBI Gröbner bases

Faugère and Rahmany presented in 2009 an algorithm for computing so-called SAGBI Gröbner bases (Faugère and Rahmany, 2009). A SAGBI Gröbner basis is the analogon of a Gröbner basis for ideals in $\mathscr{K}$-subalgebras. We introduce notation as much as needed to explain the changes in **F5**, in particular, **MatrixF5**. For more details on the theory of SAGBI bases we refer, for example, to Kapur and Madlener (1989).

In this subsection let $G \subset \mathrm{GL}(n, \mathscr{K})$ be a subgroup of $n \times n$ invertible matrices over $\mathscr{K}$. Moreover, we assume that $\mathscr{K}$ has characteristic zero or p such that p and $|G|$ are coprime.

Definition 9.1.

(a) A polynomial $f \in \mathscr{R}$ is called *invariant (w.r.t. G)* if $f(Ax) = f(x)$ for all $A \in G$. The set of all polynomials of $\mathscr{R}$ invariant w.r.t. G is denoted R^G.

(b) For $|G| < \infty$ the *Reynolds operator (for G)* is the map $\mathfrak{R} : \mathscr{R} \to \mathscr{R}^G$ defined by $\mathfrak{R}(f) = \frac{1}{|G|}\sum_{A\in G} f(Ax)$.

Proposition 9.1 *(Cox et al. (2007)). Let $\mathfrak{R}$ be the Reynolds operator for a finite group $G \subset \mathrm{GL}(n, \mathscr{K})$. Then the following properties hold:*

(a) *$\mathfrak{R}$ is $\mathscr{K}$-linear.*

(b) *$f \in \mathscr{R} \Longrightarrow \mathfrak{R}(f) \in \mathscr{R}^G$.*

(c) *$f \in \mathscr{R}^G \Longrightarrow \mathfrak{R}(f) = f$.*

Even if $\mathscr{R}^G$ might not be finite dimensional as $\mathscr{K}$-vector space, there exists a decomposition in finite dimensional homogeneous components, $\mathscr{R}^G = \oplus_{d\geq 0}\mathscr{R}^G_d$.[6] For any term $t \in \mathscr{R}$ $\mathfrak{R}(t)$ is a homogeneous invariant, called *orbit sum*. Clearly, the set of orbit sums is a vector space basis for $\mathscr{R}^G$.

Here we assume that $f_1, \ldots, f_m$ are homogeneous, invariant polynomials in $\mathscr{R}$ and I resp. I^G represent the ideal generated by $f_1 \ldots, f_m$ in $\mathscr{R}$ resp. $\mathscr{R}^G$

Definition 9.2.

(a) A subset $F \subseteq I^G$ is a *SAGBI Gröbner basis for* I^G (up to degree d) if $\{\mathrm{lt}(f) \mid f \in F\}$ generates the lead ideal of I^G as an ideal over the algebra $\langle \mathrm{lt}(f) \mid f \in \mathscr{R}^G \rangle$ (up to degree d).

(b) Let $f, g, p \in \mathscr{R}^G$ such that $f \neq 0 \neq p$. f *SG-reduces* to g modulo p if there exists a term t of f such that there exists an s in the set of lead terms of $\mathscr{R}^G$ such that $s\,\mathrm{lt}(p) = t$ and $g = f - \frac{\mathrm{lc}(t)}{\mathrm{lc}(p)\mathrm{lc}(\mathfrak{R}(s))}\mathfrak{R}(s)p$.

[6] Note that $\mathscr{R} = \oplus_{d\geq 0}\mathscr{R}_d$ and that the action of G preserves the homogeneous components.

Clearly, one can speak of SG-reduction w.r.t. a finite subset $F \subseteq \mathscr{R}^G$. With this a SAGBI Gröbner basis can be defined similar to a usual Gröbner basis:

Proposition 9.2. *Let F be a subset of an ideal $I^G \subseteq \mathscr{R}^G$. The following are equivalent:*

(a) *F is a SAGBI Gröbner basis for I^G.*
(b) *Every $h \in I^G$ SG-reduces to zero w.r.t. F.*

Note that a SAGBI Gröbner basis might not be finite.

Instead of using elimination techniques in order to compute a SAGBI Gröbner basis for a given ideal $I^G \subseteq \mathscr{R}^G$ one can use the ideas of Thiéry who presented in Thiéry (2001) a variant of Buchberger's algorithm.

Faugère and Rahmany use in Faugère and Rahmany (2009) the **MatrixF5** description of **F5** to present the modifications: Let $f_1, \ldots, f_m \in \mathscr{R}^G$ be the homogeneous input elements. First one defines the so-called *invariant Macaulay matrix* $M_{d,i}$ generated by $\mathfrak{R}(t_{j,k}) f_k$ for $1 \leq k \leq i$ and terms $t_{j,k}$ such that $\deg(t_{j,k}) = d - \deg(f_k)$. Two modifications to the usual Macaulay matrix have to be made:

(a) Instead of labeling the rows of $M_{d,i}$ by $t_{j,k}\boldsymbol{e}_k$ one uses $\mathfrak{R}(t_j, k)\boldsymbol{e}_k$.
(b) Instead of labeling the columns by the usual monomials m_ℓ they are indexed by $\mathfrak{R}(m_\ell)$.

Besides this no further changes need to be done. The variant of **MatrixF5** presented here assumes $<_{\text{pot}}$ as module monomial order and $\trianglelefteq_{\text{add}}$ as rewrite order. One checks for any row labeled by $\mathfrak{R}(t_{j,k})\boldsymbol{e}_k$ if f_k is the canonical rewriter in signature $\mathfrak{s}\left(t_{j,k}\boldsymbol{e}_k\right)$ and removes the row otherwise. In the description of **MatrixF5** this is equivalent to the existence of a row with corresponding lead term $t_{j,k}$ in a matrix that was previously reduced to row echelon form.

10. F5 and the quest of termination

Until 2012 there was still no complete proof of **F5**'s termination given. Thus a lot of algorithms were published in the meantime which have small adjustments in order to ensure termination.

The main problem with the proof of **F5**'s termination given in Faugère (2002) is Theorem 2: It assumes that if the input of **F5** is a regular sequence of homogeneous elements then **F5** does enlarge the lead ideal after each call of the subalgorithm Reduction. In Section 8 of Faugère (2002) an example of **F5** computing a Gröbner basis for a regular sequence of three homogeneous elements is given. In the last call of Reduction only one element, r_{10}, is added to $\mathscr{G}$ with $\operatorname{lt}\left(\overline{r_{10}}\right) = y^6 t^2$. In degree $d = 7$ **F5** has already added element r_8 to $\mathscr{G}$ with $\operatorname{lt}\left(\overline{r_8}\right) = y^5 t^2$. Thus the statement of Theorem 2, on which the proof of termination of **F5** in Faugère (2002) is based on, is not true.

Note that in this section we discuss proofs of **F5**'s termination considering **F5**'s initial description from Faugère (2002) using $<_{\text{pot}}$, see Remark 8.1.

10.1. Proving *F5*'s termination

At least since Galkin's proof in Galkin (2014)[7] termination of **F5** is clear. Several other proofs of **F5**'s termination were independently published, see (Pan et al., 2012, 2013; Eder and Roune, 2013). All of these proofs share a common ansatz: The main idea is based on partitioning $\mathscr{G}$ into sets

$$R_r := \left\{ \alpha_i \;\middle|\; \frac{\mathfrak{s}\left(\alpha_i\right)}{\operatorname{lt}\left(\overline{\alpha_i}\right)} = r \right\}$$

for given ratios r. The proof of **F5**'s termination is then done in two steps:

[7] Preprint version published in Galkin (2012).

(a) One shows that there are only finitely many non-empty sets R_r.
(b) $\#R_r < \infty$ for any non-empty set R_r.

As one can easily see, this attempt can be used for any signature-based Gröbner basis algorithm related to **RB**, thus also termination of **GVW** and other algorithms discussed in Section 11 can be handled in the same way.

Pan et al. (2012, 2013) present another attempt in proving **F5**'s termination. For this they do not only focus on **F5** but give generalized algorithms in order to use known termination of algorithms like **GVW** (see Section 11.5). They give a generalized **F5** algorithm called **F5GEN** for which they can easily prove termination in the vein of Eder and Perry's proof of termination of general signature-based Gröbner basis algorithms given in Eder and Perry (2011). This proof originates from the ideas presented in Arri and Perry (2011) where termination of a new algorithm in the vein of **F5** was shown (see also Section 10.2.1 and 11.1). Both publications (Pan et al., 2012, 2013) use the notation introduced by **G2V** resp. **GVW** and then further adopted by Huang (2010). We refer to the corresponding sections (11.2 and 11.5, respectively) for a dictionary translating the notation given here to theirs. Moreover, note that Pan et al. (2013) takes care of the problem with the insertion of rewrite rules in the original **F5** algorithm discussed in Remark 8.1: Instead of using lists $\textsc{Rule}_i$ for rewrite rules they directly check rewritability by the order of elements in $\mathscr{G}$ as done in **RB**, too. **F5GEN** now has a generalized insertion strategy for new elements in $\mathscr{G}$, called INSERT_F5GEN. This mirrors the usage of different rewrite orders $\trianglelefteq$ as explained in Section 7. Whereas Pan et al. (2013) focuses on **F5**, Pan et al. (2012) covers also **GVW** and algorithms like those presented in Section 11.

In Eder and Roune (2013), Eder and Roune first introduced **RB**[8] and where able to give another proof for **F5**'s termination: In Section 4 of Eder and Roune (2013) the rewrite order $\trianglelefteq_5$[9] corresponding to **F5**'s initial representation is defined. Thus they show **F5**'s termination proving first **RB**'s termination (see also Theorem 7.1) and proving second the equivalence between **RB** using $\trianglelefteq_5$ and **F5** as presented in Faugère (2002) (see Section 5 in Eder and Roune, 2013).

10.2. Algorithms to ensure termination algorithmically

The following algorithms are still not deprecated, they generate lower degree bounds for an earlier termination of **F5**. All the changes presented here can easily be transfered to **RB**. Furthermore, note that all the following ideas for modifying **F5** to ensure termination assume homogeneous input. The main difference to proving **F5**'s termination directly as explained in Section 10.1 is that the algorithms presented next provide algorithmic, termination ensuring modifications to **F5**.

10.2.1. Arri and Perry's work to ensure termination

In Arri and Perry (2011), Arri and Perry presented a new signature-based algorithm that used several ideas for the first time: Besides introducing the rewrite order $\trianglelefteq_{\text{rat}}$, Arri and Perry (2011) also includes the first signature-based algorithm that implemented Lemma 5.1. For more information on this, see Section 11.1. Also their approach uses $\trianglelefteq_{\text{rat}}$ instead of $\trianglelefteq_{\text{add}}$, based on Lemma 5.1 Arri and Perry were the first ones that gave a correct proof of termination of their signature-based algorithm. This proof was based on the usage of monomodules.[10] Even more, the ideas from Arri and Perry (2011) where the origin for the proof of termination of generalized signature-based Gröbner basis algorithms in Eder and Perry (2011) (see also Section 10.1).

*10.2.2. **F5t** – using the Macaulay bound*

In Gash (2008) and Gash (2009), Gash presents algorithm **F5t** which makes use of the Macaulay bound M (see, for example, Macaulay, 1902; Lazard, 1983; Bardet et al., 2005) for regular sequences. Once the degree of the polynomials treated in the algorithm exceed $2M$ redundant elements (i.e.

[8] There **RB** is just slightly more restricted concerning pair set orders, see Remark 7.2.

[9] There denoted $\preceq_5$.

[10] See, for example, Kreuzer and Robbiano (2009), for more details on monomodules.

elements α such that $\operatorname{lt}(\overline{\alpha})$ is already in the lead ideal of the current partly computed Gröbner basis) are added to a different set D. Whenever **F5** returns such a redundant element α, $\overline{\alpha}$ is reduced (*not* $\mathfrak{s}$-reduced!) completely w.r.t. $\mathscr{G} \cup D$. All corresponding signatures and rewrite rules are marked to be invalid. Any newly computed S-pair with one generator out of D is handled without signature-based criteria checks and just completely reduced (again, *not* $\mathfrak{s}$-reduced!) w.r.t. $G \cup D$. Whereas termination and correctness are ensured in this approach, performance really becomes a problem. Depending on the input it often introduces an enormous number of zero reductions for elements generated out of D. Moreover, as for **F5B**, taking care of two different lists of elements at the same time, is a bottleneck, too.

10.2.3. Using Buchberger's chain criterion

In 2005, Ars defended his PhD thesis (Ars, 2005). There a different algorithm is presented which was later on denoted by **F5B** in Eder et al. (2011). In this algorithm a degree bound of the algorithm is computed with the help of Buchberger's chain criterion. Besides the usual pair set $\mathscr{P}$ a second set $\mathscr{P}^*$ is stored. Whereas $\mathscr{P}$ is still used for the actual computations with **F5** $\mathscr{P}^*$ has only the purpose to find a degree bound d for the algorithm. Whenever new S-pairs are computed the ones which are not detected by Buchberger's chain criterion are added to $\mathscr{P}^*$. After updating $\mathscr{P}^*$ d is set to the highest degree of any S-pair in $\mathscr{P}^*$. Once the degrees of all S-pairs in $\mathscr{P}$ exceed d then by Buchberger's chain criterion the polynomial part of the computed signature Gröbner basis up to degree d is already a Gröbner basis for the input ideal.

Algorithmic Property 10.1. **F5B** uses linear algebra instead of polynomial $\mathfrak{s}$-reduction. We refer to Section 13 for further details on such an implementation of the reduction process.

*10.2.4. **F5+** – keeping track of redundancy*

In 2011, as a last termination dedicated algorithm before Galkin's proof in Galkin (2014), Eder, Gash and Perry present **F5+** in Eder et al. (2011). The main contribution is the distinction between so-called "GB-critical pairs" and "F5-critical pairs". A GB-critical pair corresponds to an S-pair $a\alpha - b\beta$ whereas $\operatorname{lt}(\overline{\alpha})$ and $\operatorname{lt}(\overline{\beta})$ are not already in the lead ideal of the current state of the computed Gröbner basis. An F5-critical pair is an S-pair which does not correspond to a GB-critical pair, i.e. at least one generator is redundant. Whereas GB-critical pairs are needed to be checked for the resulting Gröbner basis, F5-critical pairs seem to be superfluous, but this is not always the case: Due to the rewritable signature criterion it might happen that an GB-critical pair is discarded and instead a corresponding F5-critical pair is $\mathfrak{s}$-reduced later on. Only since the F5-critical pair is taken care of the algorithm's correctness is ensured. This means that even if at a given degree d there is no GB-critical pair left, one might need to $\mathfrak{s}$-reduce corresponding F5-critical pairs in this degree. The idea is now to store all, by **F5**'s signature-based criteria discarded GB-critical pairs in a second list $\mathscr{P}^*$, and keep all usual critical pairs (resp. S-pairs) in $\mathscr{P}$. As long as the degree of the currently handled elements in $\mathscr{P}$ is smaller or equal to the maximal degree of elements in $\mathscr{P}^*$ the algorithm needs to carry on due to the above discussion. Once the degree exceeds the maximal degree of an element in $\mathscr{P}^*$ Buchberger's chain criterion is used: If all elements in $\mathscr{P}^*$ can be removed by it then the algorithm can terminate. This is due to the fact that in $\mathscr{P}^*$ all for the resulting Gröbner basis needed, but due to rewritings discarded GB-critical pairs are stored. Once it is ensured (by Buchberger's chain criterion) that those reduce to zero, we know that we already reached a Gröbner basis of the input.

Algorithmic Property 10.2. **F5+** starts checking $\mathscr{P}^*$ only once the degree of elements in $\mathscr{P}$ exceeds the maximal degree of all GB-critical pairs removed by **F5**'s signature-based criteria, not before. Since **F5B** does not take care of the connection between F5-critical pairs and GB-critical pairs, it has to check $\mathscr{P}^*$ in each step.

Moreover, **F5+** stores and checks in $\mathscr{P}^*$ only GB-critical pairs that are also discarded by **F5**'s signature-based criteria. Only for such a GB-critical pair a corresponding F5-critical pair might be necessary for the correctness of the algorithm.

Specifications 10.1. For a generic system **F5B** might find a lower degree bound than **F5+**. Moreover, note that both algorithms are able to terminate the algorithm once a constant is found: Due to checking $\mathscr{P}^*$ by Buchberger's chain criterion all other S-pairs are removed at this point.

11. Signature-based Gröbner basis algorithms using $\trianglelefteq_{\mathrm{rat}}$

Besides **F5** all other known signature-based Gröbner basis algorithms use $\trianglelefteq_{\mathrm{rat}}$.[11] We can easily see that those instantiations of **RB**, like **GVW** or **SB**, mostly coincide and just differ in notation.

Algorithmic Property 11.1. Note that using $\trianglelefteq_{\mathrm{rat}}$ in **RB** is optimal in the sense of the size of the output: For a given input system to **RB** let $\mathscr{G}_{\trianglelefteq_{\mathrm{rat}}}$ be the signature Gröbner basis computed w.r.t. $\trianglelefteq_{\mathrm{rat}}$. It is shown in Section 3.3 in Eder and Roune (2013) that using any rewrite rule other than $\trianglelefteq_{\mathrm{rat}}$, **RB** returns a signature Gröbner basis $\mathscr{G}$ that fulfills the relation: $\mathscr{G}_{\trianglelefteq_{\mathrm{rat}}} \subset \mathscr{G}$. Thus, instantiating **RB** with $\trianglelefteq_{\mathrm{rat}}$ is optimal in the sense of the size of the resulting basis.

*11.1. Arri and Perry's work – **AP***

Aberto Arri released in 2009 a first preprint of his paper with John Perry (Arri and Perry, 2011). There the first mention of $\trianglelefteq_{\mathrm{rat}}$ can be found. The paper reviews **F5**'s criteria given in Faugère (2002) and presents a signature-based Gröbner basis algorithm depending on one criterion only. There it is also called "**F5** criterion" but it is equivalent to choosing the canonical rewriter in signature T from $\mathscr{C}_T$ w.r.t. $\trianglelefteq_{\mathrm{rat}}$.

Vocabulary 1. The notions "$\mathscr{S}$-reduction" and "$\mathscr{S}$-Gröbner basis" coincide with $\mathfrak{s}$-reduction and signature Gröbner basis, respectively. The notion "primitive $\mathscr{S}$-irreducible polynomial" coincides in our notation to polynomials that are not singular top $\mathfrak{s}$-reducible (see also Remark 4.2).

Algorithmic Property 11.2.

(a) **AP** implements **RB** with $\trianglelefteq_{\mathrm{rat}}$ and can use any compatible module monomial order $<$.
(b) **AP** is (nearly simultaneously with **G2V**, see Section 11.4) the first signature Gröbner basis algorithm adding signatures of zero reductions directly to $\mathscr{H}$.
(c) **AP**'s $\mathscr{S}$-reduction is (also nearly simultaneously with **G2V**'s implementation of $\mathfrak{s}$-reduction, see Section 11.4) the first one without checking the reducers with the signature-based criteria (see also Lemma 7.4).
(d) **AP** is the first signature-based Gröbner basis algorithm that implemented Lemma 5.1. Realizing this fact, Arri and Perry were able to give a first proof of termination of their signature-based algorithm in Arri and Perry (2011) (see also Section 10.2.1).

Note that especially the last characteristic, the usage of Lemma 5.1, allowed a way easier termination condition for **AP** compared to the proofs of **F5**'s termination in Section 10. See Arri and Perry (2011), Eder and Perry (2011) for the corresponding proof of **AP**'s termination. Note that later on algorithms like **SB** (see Section 11.6) also implemented this.

*11.2. The **TRB** algorithm – top reductional basis*

Lei Huang was one of the first researchers comparing different signature-based Gröbner basis algorithms. In 2010 he presented his **TRB** algorithm in Huang (2010), where the name comes from the wording "top reductional basis".

[11] There are some minor exceptions we take care of in the following, too.

Vocabulary 2. A *top reductional prime element* coincides with the notion "not regular top $\mathfrak{s}$-reducible" given in Section 4.1 and a *top reductional basis* is just a signature Gröbner basis.

The **TRB** algorithm does not focus on efficiency, but is a more general algorithmic presentation of signature-based computations and included in **RB**: In Huang (2010) specializations of **TRB** are given that coincide with other known algorithms, like **TRB-F5**, **TRB-EF5**[12] and **TRB-GVW**.[13] Moreover, the most optimized variant **TRB-MJ** is presented which coincides with **RB** using $\trianglelefteq_{\text{rat}}$ and $\preceq_{\mathfrak{s}}$. Hereby "MJ" stands for "minimal joint multiplied pair" which corresponds to choose at a given signature T the canonical rewriter with minimal possible lead term, that means using $\trianglelefteq_{\text{rat}}$.

*11.3. The **GBGC** algorithm – generalized criteria*

In 2011, Sun and Wang presented the **GBGC** algorithm in Sun and Wang (2011a). This algorithm is also a general one and included in **RB**. This is, besides **RB**, the only signature-based Gröbner basis algorithm that considers different pair set orders $\preceq$. As already mentioned in Remark 7.2 **GBGC** is presented to use partial orders on $\mathcal{G}$ as rewrite orders which is not efficient for discarding useless S-pairs.

Vocabulary 3.

(a) The "generalized criterion" the algorithm's name comes from can be directly translated to choosing the canonical rewriter in signature T in $\mathcal{C}_T$ w.r.t. a given rewrite order $\trianglelefteq$.

(b) Note that whereas we decide to call the element *maximal* w.r.t. a rewrite order the canonical rewriter in a given signature, in Sun and Wang (2011a) the *minimum* is chosen. More particular, $\alpha \trianglelefteq \beta$ chosen there coindices with $\frac{1}{\alpha} \trianglelefteq_{\text{rat}} \frac{1}{\beta}$. So **GBGC** still implements **RB** with $\trianglelefteq_{\text{rat}}$, there is just a slight difference in notation.

Algorithmic Property 11.3. **GBGC** implements the test for regular $\mathfrak{s}$-reduction considering the coefficients of the signatures. Thus, a reduction of a term t of $\overline{\alpha}$ with some $b\beta$ such that $\mathfrak{s}(\alpha) = \mathfrak{s}(b\beta)$ is called *super-regular* if the coefficients of $\mathfrak{s}(\alpha)$ and $\mathfrak{s}(b\beta)$ differ. This definition comes initially from Gao et al. (2010a). By the definitions in Section 4.1 we call this a singular top $\mathfrak{s}$-reduction.

The following lemma shows that there is no need to consider coefficients of signatures at all, i.e. there cannot exist a super-regular top reduction without a regular top $\mathfrak{s}$-reduction.

Lemma 11.1. *In* **RB** *there cannot exist a super-regular reduction of a term t without a regular $\mathfrak{s}$-reduction of t.*

Proof. See Fact 24 in Eder and Perry (2011). □

Thus **GBGC** can be completely described by **RB**.

Specifications 11.1.

(a) Sun and Wang (2011c) use a signature Gröbner basis resulting from a computation of **RB** to decide the ideal membership problem for I. This is straightforward since the polynomial part of $\mathcal{G}$ is already a polynomial Gröbner basis. The other fact is that signatures can be used for the representation problem of an element in I. Also this is straightforward, since if you compute with the full module element $\alpha \in \mathcal{R}^m$ the signature Gröbner basis $\mathcal{G}$ stores already the full information. If one is using **RB** with sig-poly pairs (Sun and Wang, 2011c) proposes just an algorithm to recover the full module representation of elements in the Gröbner basis.

[12] See Section 8.3.

[13] See Section 11.5.

(b) In 2012, Sun, Wang, Ma and Zhang have presented the **SGB** algorithm in Sun et al. (2012). **SGB** is a signature-based Gröbner basis algorithm for computations in algebras of solvable type (for example, see Kandri-Rody and Weispfenning, 1990) like the Weyl algebra or quantum groups. As a rewrite order $\trianglelefteq_{\text{rat}}$ is used, which they denote as "GVW-order".[14] Besides adjusting the polynomial arithmetic for the corresponding algebras no changes with respect of the signature-based tools have to be made.

11.4. *The **G2V** algorithm*

The **G2V** algorithm refers to Gao, Guan and Volny and was first presented in 2010. Its description is published in Gao et al. (2010a). A high-level implementation in Singular is available under

http://www.math.clemson.edu/~sgao/code/g2v.sing.

G2V introduced several important concepts of signature-based Gröbner basis algorithms:

(a) As mentioned already in Property 8.1 (a) **G2V** was, after the description in Arri and Perry (2011), the first algorithm who used non-Koszul syzygies directly in the syzygy criterion. The algorithm is described in the vein of **F5**'s description in Faugère (2002) and thus based on using $<_{\text{pot}}$ as module monomial order, which leads to an incremental Gröbner basis algorithm.
(b) In Gao et al. (2010a) a generalized concept of signatures was introduced: Instead of defining the uniquely defined minimal (w.r.t. $<_{\text{pot}}$) signature for a polynomial f as suggested in Faugère (2002), they define a signature $\text{lm}(u)$ for any element $(u, f) \in \mathscr{R} \times \mathscr{R}$. Note that since **G2V** uses $<_{\text{pot}}$ as module monomial order it is enough to store an element $u \in \mathscr{R}$. This concept is further generalized in **GVW**, see Section 11.5.
(c) In Gao et al. (2010a) the authors describe for the first time how **G2V** and thus signature-based Gröbner basis algorithms in general can be used to compute a Gröbner basis for the syzygy module, by considering not only the signatures, but the full module representations. Still, this leads to the overhead of carrying out all computations in $\mathscr{R}^m$, too.

Algorithmic Property 11.4. Compared to **F5** as presented in Faugère (2002) **G2V** is the first signature-based algorithm that introduced the following ideas:

(a) Take coefficients into account for $\mathfrak{s}$-reductions.
(b) Implement no real rewrite rule as done in **F5**.

Whereas the first point enables so-called "super-regular reductions" that might be not possible in **F5** it turns out that this is not the case: As already mentioned in Section 11.3 and proven in Fact 24 in Eder and Perry (2011) resp. Lemma 11.1, whenever there exists a super-regular $\mathfrak{s}$-reduction then there exists also a regular $\mathfrak{s}$-reduction. It follows that when it comes to signatures, coefficients need not be taken into account at all.

In order to discuss the second difference, let us first introduce some vocabulary.

Vocabulary 4. In Gao et al. (2010a) notation is a bit different:

(a) Instead of considering sig-poly pairs $(\mathfrak{s}(\alpha), \overline{\alpha})$, pairs $(u, v) \in \mathscr{R}^2$ are considered. This is possible since **G2V** is presented only for $<_{\text{pot}}$, thus an incremental computation of $\mathscr{G}$ is achieved. So any signature $\mathfrak{s}(\alpha) \in \mathscr{R}^m$ is always of the type $\mathfrak{s}(\alpha) = s\boldsymbol{e}_k$ where $s \in \mathscr{M}$ and k is the currently highest index of an element considered. So one can remove $\boldsymbol{e}_k$ without any problem, since all signatures share this module generator for the current incremental step. So one gets a representation $(s, \overline{\alpha}) \in \mathscr{R}^2$ corresponding to (u, v).

[14] More details on the changes in **GVW**'s rewrite order over the years can be found in Section 11.5.

(b) Next pairs (u_1, v_1) and (u_2, v_2) are considered. Let $\lambda = \operatorname{lcm}(\operatorname{lt}(v_1), \operatorname{lt}(v_2))$ and define $t_i := \frac{\lambda}{\operatorname{lt}(v_i)}$. Then $(t_1(u_1, v_1), t_2(u_2, v_2))$ is called the *J-pair* of (u_1, v_1) and (u_2, v_2). This corresponds to the notion of our S-pairs. "J" denotes "joint", thus also parts of the J-pair have special notation: In the above setting $t_i v_i$ are called *J-polynomials* and $t_i \operatorname{lt}(u_i)$ are *J-signatures*.

The other difference to **F5** mentioned in Property 11.4 is not so obvious at the first look. Whenever a new $c\gamma$ may be added to $\mathscr{P}$ the authors state in the pseudo code of the algorithm to "store only one J-pair for each distinct J-signature". This clearly is a rewritable signature criterion, but no explicit statement on which element shall be kept and which shall be removed. Looking into the Singular code of **G2V** provided by the authors (see link above) one can see that in the procedure insertPairs the newly generated element by $c\gamma$ is taken whereas $a\alpha$, previously added to $\mathscr{P}$, is removed if $\mathfrak{s}(a\alpha) = \mathfrak{s}(c\gamma)$. Thus **G2V** implements $\trianglelefteq_{\text{add}}$ as rewrite rule and not $\trianglelefteq_{\text{rat}}$. The reason we keep **G2V** in this section is that it is the historical predecessor of **GVW** which uses (in its current version) $\trianglelefteq_{\text{rat}}$ (see below).

One difference left is the fact that in the provided code for **G2V** only one generator of an S-pair resp. J-pair is stored in $\mathscr{P}$. Thus an S-pair reduction $a\alpha - b\beta$ might not take place, but instead there might exist a better reducer $c\gamma$ instead of $b\beta$. This is an implicit statement of the rewritable criterion on the second generator of the S-pair.

Lemma 11.2. *After adding $a\alpha$ from the S-pair $a\alpha - b\beta$ to $\mathscr{P}$ in* **G2V***, if there exists another regular top $\mathfrak{s}$-reducer $c\gamma$ of $a\alpha$ which is not rewritable then $b\beta$ is rewritable.*

Proof. If there exists another regular $\mathfrak{s}$-reducer $\gamma \in \mathscr{G}$ which is, at the moment $a\alpha$ is started to be regular $\mathfrak{s}$-reduced, not rewritable, then instead of $a\alpha - b\beta$ the regular $\mathfrak{s}$-reduction $a\alpha - c\gamma$ takes place for some monomial c. Since $\operatorname{lt}\left(\overline{b\beta}\right) = \operatorname{lt}\left(\overline{c\gamma}\right)$ and $\mathfrak{s}(b\beta)$, $\mathfrak{s}(c\gamma) < \mathfrak{s}(a\alpha)$ three situations may happen:

(a) If $\mathfrak{s}(c\gamma) = \mathfrak{s}(b\beta)$ then we can assume w.l.o.g. that γ is the canonical rewriter in signature $\mathfrak{s}(b\beta)$. Thus $b\beta$ is rewritable.

(b) If $\mathfrak{s}(c\gamma) > \mathfrak{s}(b\beta)$ then the S-pair $c\gamma - b\beta$ has been already $\mathfrak{s}$-reduced to an element $\delta \in \mathscr{G}$. Since $\mathfrak{s}(\delta) = \mathfrak{s}(c\gamma)$ and **G2V** uses $\trianglelefteq_{\text{add}}$ δ is the canonical rewriter in signature $\mathfrak{s}(c\gamma)$ and thus $c\gamma$ is rewritable, a contradiction to our assumption.

(c) If $\mathfrak{s}(b\beta) > \mathfrak{s}(c\gamma)$ then the S-pair $b\beta - c\gamma$ has been already reduced to an element $\delta \in \mathscr{G}$, this time $\mathfrak{s}(\delta) = \mathfrak{s}(b\beta)$. By the same argument as above $b\beta$ is rewritable. □

All in all, **G2V** (as presented in Gao et al., 2010a) implements **RB** with $<_{\text{pot}}$ and $\trianglelefteq_{\text{add}}$.

Algorithmic Property 11.5. Lemma 11.2 might suggest that **G2V** as presented in Gao et al. (2010a) makes use of the rewritability. Looking at the Singular code provided it turns out that this is not the fact: In procedure findReductor a reducer of the same index is searched for in $\mathscr{G}$. This search starts from the initially added element of current index to $\mathscr{G}$. Thus the first possible regular top $\mathfrak{s}$-reducer found might not be a "better" choice, where "better" is meant in terms of the rewrite order $\trianglelefteq_{\text{add}}$.

11.5. The **GVW** algorithm

Later in 2010, Gao, Volny and Wang published (Gao et al., 2010b) in which they describe the algorithm **GVW**. There are different versions of the **GVW** paper which refer to Gao et al. (2010b, 2011, 2013), and Gao et al. (2016) respectively. Note that Gao et al. (2016) is the current version published in 2016, whereas we also refer to the other versions in order to show the historical development of **GVW**. In its first presentation in Gao et al. (2010b) **GVW** generalizes **G2V** in the sense that compatible module monomial orders can be used freely instead of restricting to only $<_{\text{pot}}$. Still, $\trianglelefteq_{\text{add}}$ is used as rewrite order in this version of **GVW**.

In Gao et al. (2010b) the generalized definition of a signature we use in this survey was stated for the first time, see Definition 2.4 and Remark 2.1. Note that **GVW** freely can use any pair set order

$\preceq$. If **GVW** uses $\preceq_{\mathfrak{s}}$, the minimality of the signature for each element added to the Gröbner basis is guaranteed.

The work of Huang (see Section 11.2) and Sun and Wang (see Section 11.3) resulted in an algorithm denoted **GVWHS** in Volny's PhD thesis (Volny, 2011). **GVWHS** uses $\trianglelefteq_{\text{rat}}$ as rewrite order, besides this fact it coincides with **GVW**.

In 2011 and later, the initial **GVW** paper Gao et al. (2010b) has been updated to Gao et al. (2011). There **GVW** already uses $\trianglelefteq_{\text{rat}}$ as rewrite order. Gao et al. (2011) was the first paper stating a simple characterization of signature Gröbner bases resp. *strong Gröbner bases* as defined in the following.

Definition 11.1 *(Gao et al. (2016)).* Let $M = \{(\alpha, g) \mid \overline{\alpha} = g\} \subset \mathscr{R}^m \times \mathscr{R}$. A subset $G \subset M$ is called a *strong Gröbner basis*[15] *of* M if every nonzero pair in M is top-reducible by some pair in G.

Vocabulary 5.

(a) With the above definition of a strong Gröbner basis G, $G = \{(\alpha_1, g_1), \ldots, (\alpha_k, g_k)\}$ consists of two different Gröbner bases:
  (i) For all $(\alpha_i, g_i) \in G$ with $g_i = 0$, the α_i generate a Gröbner basis for the syzygy module. In the notation of this survey we get $\mathscr{H} = \{\alpha_i \mid (\alpha_i, g_i) \in G, g_i = 0\} \subset \mathscr{R}^m$.
  (ii) For all $(\alpha_i, g_i) \in G$ the $\overline{\alpha_i} = g_i \neq 0$ generate the Gröbner basis for the input ideal $I = \langle f_1, \ldots, f_m \rangle \subset \mathscr{R}$. Thus, by Definition 4.2, $\mathscr{G} = \{\alpha_i \mid (\alpha_i, g_i) \in G, g_i \neq 0\} \subset \mathscr{R}^m$ is a signature Gröbner basis for I.

  In this survey we decided to keep both parts, $\mathscr{G}$ and $\mathscr{H}$ separate from each other, also in order to show differences between what we want to compute: only a polynomial Gröbner basis for I or also a basis for the syzygy module? Both ways of introducing Gröbner basis computation using signatures have advantages and disadvantages, both attempts are equivalent in the above sense. The reader should decide which one she/he prefers. What has to be pointed out is that **GVW** was the first algorithm with this general concept.

(b) When using the concept of strong Gröbner bases speaking of detecting all useless S-pairs resp. J-pairs the term "useless" needs to be taken with care: Clearly, a zero reduction of an S-pair is not useless in terms of a strong Gröbner basis since it leads to a new syzygy that is not a multiple of an element of $\mathscr{H}$ already. Thus one needs to be careful and should not mix this up with what is understood by "useless" in terms of an S-pair w.r.t. a usual polynomial Gröbner basis resp. a signature Gröbner basis $\mathscr{G}$.

(c) In Definition 11.1 "top-reducible by some pair in G" is equivalent to our notion of top $\mathfrak{s}$-reduction for elements in $\mathscr{R}^m$ which then naturally connects Definition 11.1 to Definition 4.2.

(d) In the same way "regular top-reduction by some pair in G" corresponds to our notion of regular top-$\mathfrak{s}$-reduction. What we call singular top-$\mathfrak{s}$-reduction is included in the notion "super top-reduction" in Gao et al. (2016).

(e) But since $G \subset M \subset \mathscr{R}^m \times \mathscr{R}$ and G contains a Gröbner basis for the corresponding syzygy module, in Gao et al. (2016) "super top-reduction" has also another meaning: Any top-reduction by a syzygy, that means an element $(\alpha, 0) \in M$, is also a super top-reduction. Since we are not assuming syzygies to be part of $\mathscr{G}$ in the definition of a signature Gröbner basis, in this survey such a super top-reduction by a syzygy is always handled by the rewrite orders where we always prefer elements from $\mathscr{H}$ to elements from $\mathscr{G}$.

(f) Moreover, the notion of "eventually super top-reducible by G" is introduced in Gao et al. (2016). An element $(\alpha, f) \in M$ is then said to be eventually super top-reducible if a finite sequence of regular top-reductions by elements in G exists such that the result of these reductions is an element (α', f') that is no longer regular top-reducible by G, but super top-reducible by G.

[15] Note that usually the term *strong Gröbner basis* denotes special Gröbner bases in polynomial rings over Euclidean domains like $\mathbb{Z}$.

The benefit of this notation is that Gao, Volny, and Wang were able to give a natural description of when a subset G of M is a strong Gröbner basis. In order to understand this characterization we first have to review the general definition of a *J-pair* we have already mentioned in Section 11.4.

Definition 11.2 *(Gao et al. (2016)).* Let $p_1 = (\alpha_1, f_1), p_2 = (\alpha_2, f_2) \in M$. Define $t_1 = \frac{\operatorname{lcm}(\operatorname{lm}(f_1),\operatorname{lm}(f_2))}{\operatorname{lt}(f_1)}$, and $t_2 = \frac{\operatorname{lcm}(\operatorname{lm}(f_1),\operatorname{lm}(f_2))}{\operatorname{lt}(f_2)}$. Without loss of generality, we assume that $t_1\mathfrak{s}(\alpha_1)$ is the maximum of $\{t_1\mathfrak{s}(\alpha_1), t_2\mathfrak{s}(\alpha_2)\}$. If $t_1\mathfrak{s}(\alpha_1) - \operatorname{lc}(f_1)t_2\mathfrak{s}(\alpha_2) = t_1\mathfrak{s}(\alpha_1)$ then $t_1 p_1$ denotes the *J-pair* of p_1 and p_2.

There are two points to be highlighted here:

Vocabulary 6.

(a) As already seen by Theorem 4.1 that we only need to consider regular S-pairs, **GVW** also only considers J-pairs that come from regular top-reductions.
(b) In contrast to S-pairs a J-pair between p_1 and p_2 does not cancel the leading terms of the multiplied polynomials f_1 and f_2, but keeps $\operatorname{lt}(t_1 f_1)$. Later on, when the J-pair $t_1 p_1$ is processed to be reduced in the algorithm its definition enforces that at least one regular top-reduction can take place.

Definition 11.3 *(Gao et al. (2016)).* Let $G \subset M \subset \mathscr{R}^m \times \mathscr{R}$. An element $(\alpha, f) \in M$ is called *covered by G* if there exist $(\beta, g) \in G$ and $t \in \mathscr{R}$ such that $\mathfrak{s}(\beta) \mid \mathfrak{s}(\alpha)$ with $t\mathfrak{s}(\beta) = \mathfrak{s}(\alpha)$ and $t \operatorname{lm}(g) < \operatorname{lm}(f)$.

Theorem 11.1 *(Main characterization of strong Gröbner bases in Gao et al. (2016)). Suppose that G is a subset of M such that for any monomial $T \in \mathscr{R}^m$ there is an element $(\alpha, f) \in G$ and a monomial $t \in \mathscr{R}$ such that $T = t\mathfrak{s}(\alpha)$. Then the following are equivalent:*

(a) *G is a strong Gröbner basis for M.*
(b) *Every J-pair of G is eventually super top-reducible by G.*
(c) *Every J-pair of G is covered by G.*

Note that Theorem 11.1 corresponds to Theorem 7.3: Criterion (c) in the above theorem corresponds to our notation of not being the canonical rewriter w.r.t. $\trianglelefteq_{\text{rat}}$. In this survey we used rewrite bases and rewrite orders to finally characterize signature Gröbner bases. Gao, Volny, and Wang used the notion of strong Gröbner bases.

Algorithmic Property 11.6. **GVW** stores only one J-pair (α, f) per signature, the one with minimal $\operatorname{lm}(f)$, and only if there is no syzygy whose lead term divides $\mathfrak{s}(\alpha)$ (Step 4*b* in Gao et al., 2016). This translates to α being the canonical rewriter in $\mathfrak{s}(\alpha)$ w.r.t. $\trianglelefteq_{\text{rat}}$ in the setting of Section 7.

Let us sum up the historic development of **GVW**: **G2V** implements **RB** with $<_{\text{pot}}$ and $\trianglelefteq_{\text{add}}$. **GVW** is introduced as **G2V** with the option to use different compatible module monomial orders, but still implementing $\trianglelefteq_{\text{add}}$. Due to the work of Huang (2010) and Sun and Wang (2011a), **GVW** nowadays is understood as the algorithm Volny denotes in his PhD thesis as **GVWHS**: **RB** with no restriction on the compatible module monomial order and $\trianglelefteq_{\text{rat}}$ as rewrite order.

Note that in Gao et al. (2013) the 2013 revision of **GVW** a new step in considering more principal syzygies is added. We discuss this in Section 12. In 2016, Gao, Volny and Wang published another revision of the paper (Gao et al., 2016). Moreover, in Sun and Wang (2013b) extending the **GVW** algorithm in terms of orderings and algebras over which one can compute. In 2014, Sun, Wang, Huang and Lin gave in Sun et al. (2014) a “monomial-oriented **GVW**” which is a signature-based algorithm using $\trianglelefteq_{\text{rat}}$ as rewrite order, not taking into account S-polynomials but constructing the full Macaulay matrices like **MatrixF5** (see Section 3). Thus this algorithm is dedicated to dense input systems.

Note that Theorem 2.3 in Gao et al. (2011) coincides with Lemma 7.7 (see also Vocabulary 5 (a)).

11.6. The **SB** algorithm

Roune and Stillman presented the **SB** algorithm at ISSAC'12 (Roune and Stillman, 2012a). Later on, Eder and Roune developed **RB** in Eder and Roune (2013). There they have also shown that **SB** coincides with **RB** using $\trianglelefteq_{\text{rat}}$ as rewrite order. Thus **SB** coincides with **AP** and **GVWHS**.

Remark 11.1. Note that Roune and Stillman lay an emphasis on implementational aspects and data structures. For this purpose an extended version of their ISSAC'12 paper is available (Roune and Stillman, 2012b) in which different data representations are compared and discussed extensively.

11.7. The **SSG** algorithm

In Galkin (2013), Galkin described the **SSG** algorithm, where "ssg" stands for "simple signature-based". Comparing **SSG** to **RB** both coincide once we choose $\trianglelefteq_{\text{rat}}$ as rewrite order. In Galkin (2013) defines a partial order $<_H$ on sig-poly pairs (H denotes the set of all sig-poly pairs) in the following way:

$$\left(\mathfrak{s}\left(\alpha\right), \overline{\alpha}\right) <_H \left(\mathfrak{s}\left(\beta\right), \overline{\beta}\right) \iff \mathfrak{s}\left(\beta\right) \operatorname{lt}\left(\overline{\alpha}\right) < \mathfrak{s}\left(\alpha\right) \operatorname{lt}\left(\overline{\beta}\right).$$

Moreover, syzygies are treated to be smaller w.r.t. $<_H$ then any non-syzygy. From this it follows that $\left(\mathfrak{s}\left(\alpha\right), \overline{\alpha}\right) <_H \left(\mathfrak{s}\left(\beta\right), \overline{\beta}\right)$ coincides with $\frac{1}{\alpha} \trianglelefteq_{\text{rat}} \frac{1}{\beta}$. In part 4 (b) of the pseudo code of the **SSG** algorithm the rewritable signature criterion is then implemented in the following way (adjusted to our notation):

$$\mathscr{P} \leftarrow \mathscr{P} \setminus \left\{\alpha \in \mathscr{P} \mid \exists \beta \in \mathscr{G} \text{ such that } \frac{1}{\beta} \trianglelefteq_{\text{rat}} \frac{1}{\alpha} \text{ and } \mathfrak{s}\left(\beta\right) \mid \mathfrak{s}\left(\alpha\right)\right\}$$

With the above described connection between $<_H$ and $\trianglelefteq_{\text{rat}}$ one directly sees that this is just **RB**'s rewrite procedure using $\trianglelefteq_{\text{rat}}$.

12. Using Buchberger's criteria in signature-based Gröbner basis algorithms

A natural question coming to one's mind is how **RB**'s rewrite criterion is related to Buchberger's Product and Chain criterion (Buchberger, 1965, 1979; Kollreider and Buchberger, 1978). Both predict useless computations in advance, but how do both attempts relate to each other? Does one include the other, or are there cases where one side is not able to cover the other side completely? It turns out that one can easily combine both classes of criteria, even more one can show that the rewrite criterion includes Buchberger's criteria "most of the time". It is more or less a question about how much overhead one wants to add to **RB** in order to track principal syzygies on the go. For a detailed discussion on the algebraic nature of this relation we refer to Eder (2014).

In Gash (2008) presented a first discussion on using Buchberger's Product and Chain criterion in signature-based algorithms. Moreover, Gerdt and Hashemi presented an improved version of **G2V** in Gerdt and Hashemi (2013) making use of these criteria. In 2013, Gao, Volny and Wang presented a revised version of **GVW** in Gao et al. (2013) that adds another step to store more principal syzygies. We shortly cover these algorithm in the following.

12.1. Buchberger's criteria

Let us give a short review of Buchberger's Product and Chain criterion:

Lemma 12.1 *(Product criterion, Buchberger (1965, 1979)). Let $f, g \in \mathscr{R}$ with $\operatorname{lcm}\left(\operatorname{lt}\left(f\right), \operatorname{lt}\left(g\right)\right) = \operatorname{lt}\left(f\right) \operatorname{lt}\left(g\right)$. Then $\operatorname{spol}\left(f, g\right)$ reduces to zero w.r.t. $\{f, g\}$.*

In the above situation we also say that the S-polynomial $\operatorname{spol}\left(f, g\right)$ *fulfills the Product criterion*.

Lemma 12.2 *(Chain criterion, Kollreider and Buchberger (1978) and Buchberger (1979)). Let $f, g, h \in \mathscr{R}$, and let $G \subset \mathscr{R}$ be a finite subset. If $\operatorname{lt}(h) \mid \operatorname{lcm}(\operatorname{lt}(f), \operatorname{lt}(g))$, and if $\operatorname{spol}(f, h)$ and $\operatorname{spol}(h, g)$ have a standard representation w.r.t. G resp., then $\operatorname{spol}(f, g)$ has a standard representation w.r.t. G.*

The question is now how do those criteria relate to the rewrite criterion in signature-based Gröbner basis algorithms. Gash gave a first proof that the Product criterion can be used in a signature-based algorithm without any problem due to the fact that the reductions w.r.t. $\{\alpha, \beta\}$ are regular $\mathfrak{s}$-reductions when considering $\overline{\alpha} = f$ and $\overline{\beta} = g$ in Lemma 12.1. Furthermore Gash proved that a version of Lemma 12.2 where the signatures corresponding to f, g, and h are restricted can be used in **RB**.

In Eder (2014) presented a proof that the Chain criterion is completely included in the rewrite criterion of **RB**, without any further restrictions. Moreover, the problem of being not able to predict all zero reductions that are found by the Product criterion is explained there in detail. A small counterexample for **RB** using $<_{\text{lt-pot}}$ is given. Furthermore, it is still an open question whether **RB** using $<_{\text{pot}}$ completely covers the Product criterion. So it seems that the relation between Buchberger's criteria and signature-based ones are depending on the chosen module monomial order.

Also the question of using Buchberger's criteria in **RB** is answered, there are two possible specializations of a combination of the criteria: The first one explicitly, the second one more subtle.

*12.2. **ImpG2V** – a Gebauer–Möller-like **G2V***

In Gerdt and Hashemi (2013) present **ImpG2V**, a new algorithm in the vein of **G2V**. In their algorithm they add 3 new conditions to be checked which coincides with the three steps in Gebauer–Möller's implementation of Buchberger's algorithm, see Gebauer and Möller (1988). Moreover, they show that adding these conditions can be done without corrupting signatures, thus **ImpG2V** is still a correct and terminating signature-based Gröbner basis algorithm with the rewrite criterion implemented as usual (see Theorem 4.1 in Gerdt and Hashemi, 2013).

The question whether the Chain criterion is already included by the rewrite criterion is investigated in Eder (2014).

*12.3. **GVW**'s 2013 revision*

In 2013 Gao, Volny and Wang revised **GVW** again, with the current status being presented in Gao et al. (2013). In this version of **GVW** a new step is inserted, namely an additional computation of principal syzygies even though the regular $\mathfrak{s}$-reduced γ might not fulfill $\overline{\gamma} = 0$. In Gao et al. (2013) this is Step 4*b* (*b*1) of Fig. 3.1. In our notation this would be after Line 11 of Algorithm 3. Even though $\overline{\gamma}$ is not zero, all new possible principal syzygies are generated and added to $\mathscr{H}$. Afterwards $\mathscr{H}$ is interreduced. This has two impacts:

(a) On the one hand new syzygies might be added such that more useless computations can be predicted and removed in advance. Clearly, with this attempt also all useless computations predicted by Buchberger's Product criterion (representing exactly some of these principal syzygies) are detected, too.
(b) On the other hand a lot of these new principal syzygies added to $\mathscr{H}$ may have signatures that are just multiples of signatures already available in $\mathscr{H}$. Thus the overhead might be rather high compared to the benefits.

Clearly, **GVW**'s attempt adding all possible principal syzygies does not give more information to the rewrite criterion than testing for Buchberger's Product criterion directly and adding the corresponding signature to $\mathscr{H}$ accordingly. In terms of efficient implementations it seems that checking the Product criterion explicitly introduces less overhead than generating new principal syzygies whenever a new element γ is added to $\mathscr{G}$.

Whereas the first variant adds 1 syzygy resp. signature to $\mathscr{H}$ when it is needed, the second one always tries to recover all such relations and afterwards checks, which ones can be removed from $\mathscr{H}$ being just multiples of each other.

13. $\mathfrak{s}$-Reductions using linear algebra

As already pointed out in Section 8 **F5** is presented in Faugère (2002) in the vein of implementing the $\mathfrak{s}$-reduction process using linear algebra. **MatrixF5**, presented in Section 3, is efficient once the system of polynomial equations is dense. Clearly, this is not always the case, and thus, selecting S-pairs to be reduced is more convenient compared to building full Macaulay matrices at a given degree d. The first presentation of such an S-pair generating algorithm using linear algebra for reduction purposes is the **F4** algorithm (Faugère, 1999). Here we present a variant of **F4** that uses signature-based criteria to detect reductions to zero resp. rows reducing to zero in advance. This leads to smaller changes in the implementation of some subalgorithms of **F4** corresponding to the switch from usual polynomial reduction to $\mathfrak{s}$-reduction. Albrecht and Perry describe a possible implementation of this, called **F4/5** in Albrecht and Perry (2010).

Algorithmic Property 13.1. Note that the algorithm **F4/5** described in Albrecht and Perry (2010) differs from **F5** by more than replacing the polynomial $\mathfrak{s}$-reduction by linear algebra:

(a) Instead of incrementally computing the Gröbner basis for $\langle f_1, \ldots, f_m \rangle$ computations are done by increasing degrees: Whereas **F5** proceeds by index first, **F4/5** prefers the degree of the polynomials over the index. This corresponds to switching from $<_{\text{pot}}$ to $<_{\text{d-pot}}$.
(b) Instead of sorting the generators by decreasing index, they are ordered by increasing index (see also Footnote 4).
(c) Due to the switch from $<_{\text{pot}}$ to $<_{\text{d-pot}}$ the rewrite rules $\textsc{Rule}_i$ might not be sorted by increasing degree when only appending new rules as done in Faugère (2002). Thus the subalgorithm Add Rule takes care of sorting $\textsc{Rule}_i$ by increasing degree. Note that as mentioned in Remark 8.1 this still need not ensure a sorting of $\textsc{Rule}_i$ by increasing signature.

Giving a full description of the ideas behind the **F4** algorithm out of scope of this survey, we refer the readers interested to Faugère (1999). Here we explain in detail an **F4**-style variant of **RB**. With this any known instantiation of signature-based Gröbner basis algorithms described in sections 8 and 11 can be modified in the same way to use linear algebra for reduction purposes.

The main difference between **RB** and **F4-RB** is the usage of linear algebra for the reduction process in the later one. Instead of fulfilling $\mathfrak{s}$-reductions on each new S-pair, **F4-RB** implements a variant of **F4**'s reduction process: In Line 6 we no longer need to choose only one single S-pair as done in **RB** but a subset of $\mathscr{P}$ can be taken at once. The generators of those symbolic S-pairs are then stored in $\mathcal{L}_d$ (Line 8). Subalgorithm **Symbolic Preprocessing** is then precomputing all possible reducers of the elements in $\mathcal{L}_d$. Due to the additional structure of the signatures one has to change this part slightly compared to an implementation in the **F4** Algorithm. This is discussed in Property 13.2. After all elements needed to execute in the dth reduction step of the algorithm are stored in $\mathcal{L}_d$ a corresponding matrix M_d w.r.t. $<$ is constructed: The rows of M_d represent the elements $\overline{a\alpha}$ for $a\alpha \in \mathcal{L}_d$, the columns represent the corresponding monomials in $\mathscr{R}$ ordered w.r.t. $<$. As in the **MatrixF5** Algorithm each row has a signature, namely $\mathfrak{s}(a\alpha)$. As mentioned already in Section 4.1 $\mathfrak{s}$-reductions on the polynomial side correspond to fixing an order on the rows in M_d. Thus the computation of the row echelon form of M_d in Line 11 is done without row swapping.

Specifications 13.1.

(a) As already mentioned in Property 5.1 (b) for an efficient implementation one would use $(\mathfrak{s}(\alpha), \overline{\alpha})$ instead of α in **F4-RB**. Algorithm 5 as presented here works with full module elements, that means when computing the row echelon form one needs to keep track of all corresponding

Algorithm 5 Rewrite basis algorithm using linear algebra **F4-RB**.

```
Require: Ideal $I = \langle f_1, \ldots, f_m \rangle \subset \mathscr{R}$, monomial order $\leq$ on $\mathscr{R}$ and a compatible extension on $\mathscr{R}^m$, total order $\preceq$ on the pairset $\mathscr{P}$ of S-pairs, a rewrite order $\trianglelefteq$ on $\mathscr{G} \cup \mathscr{H}$
Ensure: Rewrite basis $\mathscr{G}$ for $I$, Gröbner basis $\mathscr{H}$ for $\operatorname{syz}(f_1, \ldots, f_m)$
1: $\mathscr{G} \leftarrow \emptyset$, $\mathscr{H} \leftarrow \emptyset$, $d \leftarrow 0$
2: $\mathscr{P} \leftarrow \{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_m\}$
3: $\mathscr{H} \leftarrow \left\{ f_i \boldsymbol{e}_j - f_j \boldsymbol{e}_i \mid 1 \leq i < j \leq m \right\} \subseteq \mathscr{R}^m$
4: while $\mathscr{P} \neq \emptyset$ do
5:     $d \leftarrow d + 1$
6:     $\mathscr{P}_d \leftarrow$ Select $(\mathscr{P})$
7:     $\mathscr{P} \leftarrow \mathscr{P} \setminus \mathscr{P}_d$
8:     $\mathcal{L}_d \leftarrow \{a\alpha, b\beta \mid a\alpha - b\beta \in \mathscr{P}_d\}$
9:     $\mathcal{L}_d \leftarrow$ Symbolic Preprocessing$(\mathcal{L}_d, \mathscr{G})$
10:    $M_d \leftarrow$ matrix gen. by rows corr. to $\overline{a\alpha}$ for $a\alpha \in \mathcal{L}_d$ (sorted by signatures)
11:    $N_d \leftarrow$ row echelon form of $M_d$ computed without row swapping
12:    $\mathscr{G}_d \leftarrow \left\{ \gamma \mid \overline{\gamma} \text{ corresponding to a row in } N_d \right\}$
13:    $\mathscr{G}_d^+ \leftarrow \left\{ \gamma \in \mathscr{G}_d \mid \operatorname{lt}(\overline{\gamma}) \neq \operatorname{lt}\left(\overline{a\alpha}\right) \text{ for } a\alpha \in \mathcal{L}_d, \mathfrak{s}(\gamma) = \mathfrak{s}(a\alpha) \right\}$
14:    while $\mathscr{G}_d^+ \neq \emptyset$ do
15:        $\gamma \leftarrow \min_< \mathscr{G}_d^+$
16:        $\mathscr{G}_d^+ \leftarrow \mathscr{G}_d^+ \setminus \{\gamma\}$
17:        if $\overline{\gamma} = 0$ then
18:            $\mathscr{H} \leftarrow \mathscr{H} + \{\gamma\}$
19:        else
20:            $\mathscr{P} \leftarrow \mathscr{P} \cup \left\{ \operatorname{spair}(\alpha, \gamma) \,\middle|\, \alpha \in \mathscr{G} \text{ and } \operatorname{spair}(\alpha, \gamma) \text{ is regular} \right\}$
21:            $\mathscr{G} \leftarrow \mathscr{G} \cup \{\gamma\}$
22: return $(\mathscr{G}, \mathscr{H})$
```

module operations in $a\alpha$ for each such row in M_d. Focusing on the computation of a Gröbner basis and using only $\left(\mathfrak{s}(a\alpha), \overline{a\alpha}\right)$ this overhead disappears completely due to the fact that row swappings are not allowed and thus the signatures corresponding to rows in M_d do not change throughout the whole process.

(b) In **F4** all polynomials corresponding to rows in N_d are added to the Gröbner basis which lead term is not already included in the lead ideal. Signatures lead to $\mathfrak{s}$-reductions. We have seen already in Section 7 that elements γ might be added to $\mathscr{G}$ even so there exists some $\alpha \in \mathscr{G}$ such that $\operatorname{lt}(\overline{\alpha}) \mid \operatorname{lt}(\overline{\gamma})$. Thus we cannot discard those elements. In Line 13 we choose the elements γ that need to be added to $\mathscr{G}$ (or $\mathscr{H}$ if $\overline{\gamma} = 0$): If the polynomial lead term corresponding to a signature $\mathfrak{s}(a\alpha)$ has not changed during the computation of the row echelon form N_d of M_d then we do not need to add this element to $\mathscr{G}$. In any other case, we do so.

In Algorithm 6 we state the pseudo code of a signature respecting variant of **Symbolic Preprocessing** from Faugère (1999).

Algorithm 6 Symbolic Preprocessing respecting signatures.

```
Require: a finite subset $\mathscr{U}$ of $\mathscr{R}^m$, a finite subset $\mathscr{G}$ of $\mathscr{R}^m$
Ensure: a finite subset $\mathscr{U}$ of $\mathscr{R}^m$
$D \leftarrow \left\{ \operatorname{lt}\left(\overline{\beta}\right) \mid \beta \in \mathscr{U} \right\}$
$C \leftarrow \left\{ \text{monomials of } \overline{\beta} \mid \beta \in \mathscr{U} \right\}$
while $C \neq D$ do
    $m \leftarrow \max_< (C \setminus D)$
    $D \leftarrow D \cup \{m\}$
    $\mathscr{V} \leftarrow \emptyset$
    for $\gamma \in \mathscr{G}$ do
        if $\exists c \in \mathscr{M}$ such that $m = \operatorname{lt}\left(\overline{c\gamma}\right)$ and not Rewritable $(c\gamma, \mathscr{G} \cup \mathscr{H}, \trianglelefteq)$ then
            $\mathscr{V} \leftarrow \mathscr{V} \cup \{c\gamma\}$
    $e\varepsilon \leftarrow$ element of minimal signature in $\mathscr{V}$
    $\mathscr{U} \leftarrow \mathscr{U} \cup \{e\varepsilon\}$
    $C \leftarrow C \cup \left\{ \text{monomials of } \overline{e\varepsilon} \right\}$
return $\mathscr{U}$
```

Algorithmic Property 13.2. Algorithm 6 has undergone several small changes compared to the version presented in Faugère (1999):

(a) From lines 7 to 9 the algorithm loops over all elements $\gamma \in \mathscr{G}$ searching for a possible, not rewritable reducer of the monomial m. If successful we add the multiplied reducer to an intermediate set $\mathscr{V}$. Instead to the original **Symbolic Preprocessing** algorithm we do not stop after finding a first possible reducer her. The idea is to take in Line 10 the single reducer $e\varepsilon$ of minimal signature from $\mathcal{V}$. The smaller the signature of $e\varepsilon$ the bigger is the probability that $e\varepsilon$ might be an allowed reducer of some other row in M_d for term $\operatorname{lt}\left(\overline{e\varepsilon}\right)$.

(b) Let $a\alpha \in \mathscr{U}$ such that m is a monomial in $\overline{a\alpha}$ and $a\alpha$ is of maximal signature for all such elements in $\mathscr{U}$. Note that it is still possible that $\mathfrak{s}(e\varepsilon) > \mathfrak{s}(a\alpha)$. If $m = \operatorname{lt}\left(\overline{a\alpha}\right)$ this corresponds to the creation of a new S-pair $\operatorname{spair}(\varepsilon, \alpha) = e\varepsilon - a\alpha$. Note that in Algorithm 3 the generation of this S-pair is postponed: There only regular $\mathfrak{s}$-reductions are computed in Line 8, $\operatorname{spair}(\epsilon, \alpha)$ is generated in Line 12 first. Moreover, note that there does not exist another reducer $e'\varepsilon'$ such that $m - \overline{e'\varepsilon'}$ corresponds to a regular $\mathfrak{s}$-reduction since $e\varepsilon$ is chosen to be minimal w.r.t. its signature.

(c) Due to Lines 8 and 10 the reducer for m is uniquely defined. This choice depends on the chosen rewrite order $\trianglelefteq$ as well as the module monomial order $<$. Furthermore, one can exchange Line 10 by another choice, for example, the element in $\mathcal{V}$ which is most sparse or the one which has the lowest coefficient bound. Thus using the ideas of Brickenstein and Slimgb (2010) is possible. Note that such changes may put a penalty on the efficiency of the algorithm due to introducing many more S-pairs as the chosen reducer might not be of minimal possible signature. Still, correctness and termination are not affected.

An optimization of **F4** given in Faugère (1999) is the usage of the **Simplify** subalgorithm: **Simplify** tries to exchange generators of S-polynomials and found reducers in **Symbolic Preprocessing** with "better ones": Polynomial products $uf \in \mathscr{R}$ are tried to be exchanged by elements $\frac{u}{t}g$ where $\operatorname{lt}(g) = \operatorname{lt}(tf)$ for a divisor t of u. In Faugère (1999) the normal strategy for choosing critical pairs is used, that means, computations are done by increasing polynomial degree and thus g can be found in a previously constructed matrix M_d in degree $d := \deg(tf)$. g might not be added to the intermediate Gröbner basis as $\operatorname{lt}(f) \mid \operatorname{lt}(g)$. Still, g might be further reduced than f and thus one can prevent the algorithm in degree $\deg(uf)$ from redoing reduction steps already performed in degree d by exchanging uf by $\frac{u}{t}g$.

Due to the signatures this is not so easy in our setting: What if a simplification of $a\alpha$ by $\frac{a}{b}\beta$ leads to $\mathfrak{s}\left(\frac{a}{b}\beta\right) > \mathfrak{s}(a\alpha)$? In Property 13.2 (c) we have seen that the rewrite order $\trianglelefteq$ as well as the module monomial order $<$ uniquely define the reducer of a monomial m. This definition incorporates the ideas of **Simplify** in the signature-based world.

Specifications 13.2. Let us finish with the following notes on the idea of simplification in **F4**-like signature-based Gröbner basis algorithms.

(a) Besides the way **Simplify** is presented in Faugère (1999) other ways of choosing a better reducer are possible. In Brickenstein and Slimgb (2010), Brickenstein gives various choices. In the signature-based world this is reflected by the different implementations of the rewrite order $\trianglelefteq$ and the module monomial order $<$.

(b) If we assume $<_{\text{pot}}$ as module monomial order then we can make use of the incremental behavior of the computations: Assume that we are computing the Gröbner basis for $\langle f_1, \ldots, f_i\rangle$ having already computed one for $\langle f_1, \ldots, f_{i-1}\rangle$, say $\overline{\mathscr{G}_{i-1}}$. Now we can implement **F4**'s **Simplify** routine without any changes for elements in $\overline{\mathscr{G}_{i-1}}$: All reducers from $\overline{\mathscr{G}_{i-1}}$ have a lower signature due to its index $< i$. Thus, as already described in Faugère (2002) we do not need to check them by any criterion. Moreover, simplifying any such reducer by another element from a computation during a previous iteration step the corresponding signature still has index $< i$. Furthermore, assuming **F5C** (see Section 8.2) we can assume $\overline{\mathscr{G}_{i-1}}$ to be reduced to B_{i-1} which optimizes the choice of reducers even more. Since adding **Simplify** to **F4-RB** respectively **Symbolic Preprocessing** is straight forward in this situation we do not give explicit pseudo code for this.

(c) Moreover, exchanging $\mathscr{G}_d^+$ with $\mathscr{G}_d$ in the argument of the **while** loop in Line 14 of Algorithm 5 one can trigger a **Simplify**-like process: Since all non-zero elements are added to $\mathscr{G}$, only the S-pairs generated by the best reduced elements are not rewritten. Of course this feature is paid dearly for by generating all the useless S-pairs in first place due to the redundant elements in $\mathscr{G}$.

14. Experimental results

In the following we present experimental results of Gröbner basis benchmarks and random systems. All systems are computed over a field of characteristic 32003, with graded reverse lexicographical monomial order. The named benchmark systems can be found, for example, under Bini et al. (2012). The random systems are defined by 3 parameters on the input generators:

$$\begin{array}{ll}\text{HRandom}\,(\#\text{ vars}=\#\text{ equations, minimal degree, maximal degree}) & (\text{homogeneous})\\ \text{Random}\,(\#\text{ vars}=\#\text{ equations, minimal degree, maximal degree}) & (\text{affine})\end{array}$$

Polynomials are random dense in the corresponding number of variables. The systems are available under

https://github.com/ederc/singular-benchmarks.

The implementation is done in the computer algebra system Singular (Decker et al., 2015; Greuel and Pfister, 2007). Signature-based Gröbner basis algorithms are officially available in Singular starting version 4-0-0.[16]

We do not add timings since we do not want to start a fastest implementation contest. We are interested in presenting the size of the basis, the number of syzygies found and used, the number of reductions as well as the complete number of operations, that means, multiplications. Those are the numbers that are unique to the different signature-based Gröbner basis implementations. Any real new algorithm might compute numbers different to those presented in the following.

All algorithms using $<_{\text{pot}}$ are implemented with the ideas of **F5’** resp. **F5C**, that means, inbetween the incremental steps of computing the signature Gröbner basis the intermediate bases are reduced and new signatures are generated (see Section 8.2). This leads to three facts:

(a) The number of elements in $\mathscr{H}$ increases. The number is usually much higher than the ones for the computation w.r.t. $<_{\text{lt-pot}}$ or $<_{\text{d-pot}}$.

(b) The difference in the size of the resulting signature Gröbner basis between using $\trianglelefteq_{\text{add}}$ and $\trianglelefteq_{\text{rat}}$ diminishes: Since both computations are starting the last iteration step with the same number of elements (using the reduced Gröbner basis) only differences during the last incremental step are captured. Thus mostly the differences in the size of $\mathscr{G}$ are much bigger for the computations w.r.t. $<_{\text{lt-pot}}$.

(c) When counting the number of reduction steps as well as the number of overall operations, one needs to distinguish between the $\mathfrak{s}$-reductions done by **RB** and the number of usual reductions done inbetween two incremental steps when interreducing the intermediate Gröbner bases. In the tables below we give for computations w.r.t. $<_{\text{pot}}$ the values for $\mathfrak{s}$-reductions as well as the

[16] All examples in this survey are computed with the commit 5d25c42ce5a7cfe24a13632fa0f7cc6b85961ccb available under https://github.com/Singular/Sources.

values for all reductions including the interreduction steps. Clearly, for $<_{\text{lt-pot}}$ and $<_{\text{d-pot}}$ there is no interreduction due to non-incremental execution.

Remark 14.1.

(a) Note that the behavior for computations w.r.t. $<_{\text{d-top}}$ is not optimal. Choosing this module monomial order leads to very long running times in most of the cases. Thus we do not include the corresponding results.
(b) The differences when adding Buchberger's Product and Chain criterion to **RB** as described in Section 12 are subtle and do not change the overall behavior of **RB**. In order not to overload our tables with even more variations of **RB** we do not cover those differences here. With the information and discussions given in this survey the reader is able to understand the differences to experimental results given in Gerdt and Hashemi (2013), Gao et al. (2013), Eder (2014) which focus on this setting.

We have to distinguish different ways of computation in the following:

(a) **RB** can fulfill only top $\mathfrak{s}$-reductions or full $\mathfrak{s}$-reductions (including tail $\mathfrak{s}$-reductions).
(b) The examples can be affine or homogeneous.

Note that the differences between only top $\mathfrak{s}$-reductions and full $\mathfrak{s}$-reductions are only found in the number of $\mathfrak{s}$-reduction steps and the number of operations. Therefore the other tables do not include a differentiation between those two. Next we present the results for homogeneous respectively affine input. These values have two different ways of being used:

(a) The reader new to signature-based Gröbner basis algorithms can get a feeling for the behavior of this kind of algorithm. One can easily compare the results presented here with the outcome of SINGULAR's Gebauer–Möller implementation.
(b) For researchers trying to improve signature-based Gröbner basis algorithms those numbers are good reference points in order to see what kind of optimizations are achieved.

The corresponding figures after the corresponding tables give a graphical overview of the behavior of the different algorithms for the random systems w.r.t. increasing number of generators.

Moreover note that we stopped the computations for affine random systems at 12 resp. for homogeneous random systems at 13 generators for algorithms using only top $\mathfrak{s}$-reductions since running time was too long. For full $\mathfrak{s}$-reductions we could go on until 14 generators.

14.1. Experimental results for homogeneous systems

See Tables 2, 3, 4, 5 and 6.

14.2. Experimental results for affine systems

See Tables 7, 8, 9, 10 and 11.

14.3. Observations

From the experimental results stated here one can make several observations when it comes to signature-based Gröbner basis algorithms:

(a) For homogeneous input systems the number of zero reductions computed by **RB** using $<_{\text{pot}}$ and $<_{\text{d-pot}}$ is the very same: This is clear due to the fact that **RB** computes for each new degree d d-Gröbner bases step by step. Clearly, those numbers mostly differ for affine input systems, see Table 7. Even though the number of zero reductions for **RB** using $<_{\text{lt-pot}}$ is mostly higher

Table 2
# zero reductions (homogeneous).

Benchmark	$<_{\text{pot}}$		$<_{\text{lt-pot}}$		$<_{\text{d-pot}}$	
	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$
cyclic-7	36	36	145	145	36	36
cyclic-8	244	244	672	672	244	244
eco-10	247	247	367	367	247	247
eco-11	502	502	749	749	502	502
f-633	3	3	9	9	3	3
f-744	190	190	259	259	190	190
katsura-11	0	0	353	353	0	0
katsura-12	0	0	640	640	0	0
noon-8	0	0	294	294	0	0
noon-9	0	0	682	682	0	0
HRandom(6, 2, 2)	0	0	26	26	0	0
HRandom(7, 2, 2)	0	0	49	49	0	0
HRandom(7, 2, 4)	0	0	80	80	0	0
HRandom(7, 2, 6)	0	0	635	635	0	0
HRandom(8, 2, 2)	0	0	102	102	0	0
HRandom(8, 2, 4)	0	0	345	345	0	0
HRandom(9, 2, 2)	0	0	181	181	0	0
HRandom(10, 2, 2)	0	0	339	339	0	0
HRandom(11, 2, 2)	0	0	590	590	0	0
HRandom(12, 2, 2)	0	0	1083	1083	0	0
HRandom(13, 2, 2)	0	0	1867	1867	0	0
HRandom(14, 2, 2)	0	0	3403	3403	0	0

Table 3
Size of $\mathcal{G}$ (homogeneous).

Benchmark	$<_{\text{pot}}$		$<_{\text{lt-pot}}$		$<_{\text{d-pot}}$	
	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$
cyclic-7	758	658	871	848	949	751
cyclic-8	3402	2614	4074	3658	5534	3884
eco-10	677	508	541	478	934	567
eco-11	1423	1016	1092	965	2372	1168
f-633	60	58	61	59	61	57
f-744	616	465	377	348	745	573
katsura-11	700	700	553	553	2188	2161
katsura-12	1383	1384	1076	1076	6020	6020
noon-8	1384	1390	1384	1389	1384	1389
noon-9	3743	3750	3743	3749	3743	3749
HRandom(6, 2, 2)	52	52	39	39	62	62
HRandom(7, 2, 2)	101	101	67	67	124	124
HRandom(7, 2, 4)	333	333	249	249	349	349
HRandom(7, 2, 6)	4066	4066	2928	2928	4247	4247
HRandom(8, 2, 2)	185	185	128	128	242	242
HRandom(8, 2, 4)	1397	1397	997	997	1507	1507
HRandom(9, 2, 2)	365	365	223	223	479	479
HRandom(10, 2, 2)	676	676	426	426	932	932
HRandom(11, 2, 2)	1326	1326	767	767	1832	1832
HRandom(12, 2, 2)	2492	2492	1463	1463	3557	3557
HRandom(13, 2, 2)	4879	4879	2708	2708	6973	6973
HRandom(14, 2, 2)	9259	9259	5142	5142	13524	13524

Table 4
Size of $\mathscr{H}$ (homogeneous).

Benchmark	$<_{\text{pot}}$		$<_{\text{lt-pot}}$		$<_{\text{d-pot}}$	
	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$
cyclic-7	7260	7260	187	187	439	338
cyclic-8	103285	103285	761	761	3691	2599
eco-10	30508	30508	412	412	1111	848
eco-11	118110	118110	804	804	2978	1750
f-633	122	122	37	37	40	40
f-744	16616	16616	316	316	654	641
katsura-11	24976	24976	408	408	2728	2670
katsura-12	92235	92235	706	706	9065	9148
noon-8	406	406	322	322	84	84
noon-9	666	666	718	718	120	120
HRandom(6, 2, 2)	231	231	41	41	57	57
HRandom(7, 2, 2)	780	780	70	70	119	119
HRandom(7, 2, 4)	3160	3160	123	123	152	152
HRandom(7, 2, 6)	162735	162735	681	681	950	950
HRandom(8, 2, 2)	2278	2278	130	130	243	243
HRandom(8, 2, 4)	41328	41328	402	402	573	573
HRandom(9, 2, 2)	8256	8256	217	217	485	485
HRandom(10, 2, 2)	24976	24976	384	384	964	964
HRandom(11, 2, 2)	90951	90951	645	645	1896	1896
HRandom(12, 2, 2)	294528	294528	1149	1149	3728	3728
HRandom(13, 2, 2)	1070916	1070916	1945	1945	7285	7285
HRandom(14, 2, 2)	3667986	3667986	3494	3494	14258	14258

than those numbers for $<_{\text{pot}}$ resp. $<_{\text{d-pot}}$, due to $<_{\text{lt-pot}}$ **RB** can handle S-pairs more freely (not incremental, not degree-wise) and thus performs better with almost always less $\mathfrak{s}$-reductions and operations overall.

(b) Full $\mathfrak{s}$-reductions mostly behave better than top $\mathfrak{s}$-reductions. Performing tail $\mathfrak{s}$-reductions in earlier stages of the algorithm leads to fewer reduction steps overall. In Figs. 7 and 8 one can see this behavior quite good.

(c) As we can see also in Figs. 7 and 8 the differences between using $\trianglelefteq_{\text{add}}$ and $\trianglelefteq_{\text{rat}}$ vanish for bigger computations: In each figure 12 different implementations are presented, but we can only see 6 lines: both rewrite order behave the same. Even though we can see in the above tables that $\trianglelefteq_{\text{rat}}$ usually leads to smaller bases and generates less S-pairs, the choice of reducers w.r.t. $\trianglelefteq_{\text{add}}$ is better in terms of sparsity.

(d) Overall one can see the $<_{\text{lt-pot}}$ seems to be the best choice as module monomial order when it comes to the number $\mathfrak{s}$-reductions resp. the number of operations executed: Its numbers are always smaller than the ones for the corresponding setting of $\trianglelefteq$ with other module monomial orders.

Thus the chosen rewrite order $\trianglelefteq$ seems to be not as important as generally accepted. The main differences lay in the module monomial order.

15. Concluding remarks

In this survey we covered all known signature-based algorithms to compute Gröbner bases. We gave a complete classification based on a generic algorithmic framework called **RB**. This framework has been formulated retrospectively. The advantage of it is to parametrize all signature-based algorithms with three different order:

Table 5
$\mathfrak{s}$-reductions (incl. interreductions) (homogeneous).

Benchmark	only top $\mathfrak{s}$-reductions						full $\mathfrak{s}$-reductions					
	$<_{\text{pot}}$		$<_{\text{lt-pot}}$		$<_{\text{d-pot}}$		$<_{\text{pot}}$		$<_{\text{lt-pot}}$		$<_{\text{d-pot}}$	
	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$
cyclic-7	$2^{17.161}$	$2^{16.798}$	$2^{18.257}$	$2^{18.127}$	$2^{18.094}$	$2^{17.630}$	$2^{16.844}$	$2^{16.492}$	$2^{17.267}$	$2^{17.134}$	$2^{17.347}$	$2^{16.939}$
(incl. interred)	$2^{17.455}$	$2^{17.209}$					$2^{16.936}$	$2^{16.619}$				
cyclic-8	$2^{22.575}$	$2^{21.346}$	$2^{22.755}$	$2^{22.331}$	$2^{23.638}$	$2^{22.325}$	$2^{22.529}$	$2^{21.303}$	$2^{21.916}$	$2^{21.529}$	$2^{22.981}$	$2^{21.696}$
(incl. interred)	$2^{22.967}$	$2^{22.156}$					$2^{22.730}$	$2^{21.727}$				
eco-10	$2^{18.901}$	$2^{18.565}$	$2^{19.569}$	$2^{19.507}$	$2^{19.232}$	$2^{18.859}$	$2^{19.938}$	$2^{18.986}$	$2^{18.564}$	$2^{18.503}$	$2^{19.946}$	$2^{19.075}$
(incl. interred)	$2^{19.038}$	$2^{18.726}$					$2^{19.972}$	$2^{19.053}$				
eco-11	$2^{21.519}$	$2^{20.976}$	$2^{21.909}$	$2^{21.851}$	$2^{21.735}$	$2^{21.123}$	$2^{22.996}$	$2^{21.290}$	$2^{21.126}$	$2^{21.034}$	$2^{22.772}$	$2^{21.531}$
(incl. interred)	$2^{21.644}$	$2^{21.149}$					$2^{23.014}$	$2^{21.352}$				
f-633	$2^{9.767}$	$2^{9.604}$	$2^{10.321}$	$2^{10.236}$	$2^{9.658}$	$2^{9.397}$	$2^{9.484}$	$2^{9.522}$	$2^{10.086}$	$2^{9.801}$	$2^{9.433}$	$2^{9.044}$
(incl. interred)	$2^{9.864}$	$2^{9.713}$										
f-744	$2^{16.895}$	$2^{16.857}$	$2^{17.256}$	$2^{17.150}$	$2^{17.248}$	$2^{17.157}$	$2^{17.175}$	$2^{16.763}$	$2^{17.055}$	$2^{16.936}$	$2^{17.347}$	$2^{17.162}$
(incl. interred)	$2^{17.126}$	$2^{17.121}$					$2^{17.208}$	$2^{16.811}$				
katsura-11	$2^{18.953}$	$2^{18.708}$	$2^{22.403}$	$2^{22.384}$	$2^{20.638}$	$2^{20.527}$	$2^{22.393}$	$2^{22.257}$	$2^{22.018}$	$2^{22.067}$	$2^{22.040}$	$2^{21.985}$
(incl. interred)	$2^{22.556}$	$2^{22.514}$					$2^{22.815}$	$2^{22.712}$				
katsura-12	$2^{21.496}$	$2^{21.110}$	$2^{24.661}$	$2^{24.596}$	$2^{23.183}$	$2^{23.063}$	$2^{25.507}$	$2^{25.319}$	$2^{24.257}$	$2^{24.287}$	$2^{24.521}$	$2^{24.455}$
(incl. interred)	$2^{25.692}$	$2^{25.654}$					$2^{25.977}$	$2^{25.840}$				
noon-8	$2^{15.745}$	$2^{14.634}$	$2^{16.310}$	$2^{15.662}$	$2^{15.745}$	$2^{14.634}$	$2^{18.166}$	$2^{18.023}$	$2^{18.230}$	$2^{18.094}$	$2^{18.165}$	$2^{18.022}$
(incl. interred)	$2^{15.747}$	$2^{14.638}$										
noon-9	$2^{18.303}$	$2^{16.820}$	$2^{18.756}$	$2^{17.843}$	$2^{18.303}$	$2^{16.820}$	$2^{20.787}$	$2^{20.606}$	$2^{20.835}$	$2^{20.660}$	$2^{20.787}$	$2^{20.606}$
(incl. interred)		$2^{16.822}$										
HRandom(6, 2, 2)	$2^{10.039}$	$2^{10.229}$	$2^{11.126}$	$2^{11.126}$	$2^{10.764}$	$2^{10.966}$	$2^{10.137}$	$2^{10.364}$	$2^{10.537}$	$2^{10.537}$	$2^{10.707}$	$2^{10.880}$
(incl. interred)	$2^{10.982}$	$2^{11.094}$					$2^{10.408}$	$2^{10.599}$				
HRandom(7, 2, 2)	$2^{12.189}$	$2^{12.308}$	$2^{13.279}$	$2^{13.279}$	$2^{13.046}$	$2^{13.209}$	$2^{12.103}$	$2^{12.294}$	$2^{12.298}$	$2^{12.298}$	$2^{13.015}$	$2^{13.122}$
(incl. interred)	$2^{13.227}$	$2^{13.327}$					$2^{12.435}$	$2^{12.589}$				
HRandom(7, 2, 4)	$2^{16.664}$	$2^{16.701}$	$2^{16.782}$	$2^{16.782}$	$2^{17.093}$	$2^{17.202}$	$2^{15.190}$	$2^{15.298}$	$2^{14.911}$	$2^{14.911}$	$2^{16.557}$	$2^{16.796}$
(incl. interred)	$2^{17.314}$	$2^{17.365}$					$2^{15.268}$	$2^{15.370}$				
HRandom(7, 2, 6)	$2^{23.748}$	$2^{23.763}$	$2^{23.614}$	$2^{23.614}$	$2^{24.113}$	$2^{24.175}$	$2^{22.136}$	$2^{22.130}$	$2^{21.400}$	$2^{21.400}$	$2^{23.783}$	$2^{23.906}$
(incl. interred)	$2^{24.421}$	$2^{24.458}$					$2^{22.217}$	$2^{22.211}$				
HRandom(8, 2, 2)	$2^{14.104}$	$2^{14.290}$	$2^{15.300}$	$2^{15.300}$	$2^{15.342}$	$2^{15.462}$	$2^{14.073}$	$2^{14.232}$	$2^{14.127}$	$2^{14.127}$	$2^{15.306}$	$2^{15.408}$
(incl. interred)	$2^{15.431}$	$2^{15.538}$					$2^{14.534}$	$2^{14.652}$				
HRandom(8, 2, 4)	$2^{20.898}$	$2^{20.923}$	$2^{21.097}$	$2^{21.097}$	$2^{21.574}$	$2^{21.644}$	$2^{19.426}$	$2^{19.468}$	$2^{18.918}$	$2^{18.918}$	$2^{21.209}$	$2^{21.402}$
(incl. interred)	$2^{21.660}$	$2^{21.694}$					$2^{19.593}$	$2^{19.631}$				
HRandom(9, 2, 2)	$2^{16.135}$	$2^{16.239}$	$2^{17.331}$	$2^{17.331}$	$2^{17.719}$	$2^{17.804}$	$2^{16.200}$	$2^{16.315}$	$2^{15.758}$	$2^{15.758}$	$2^{17.685}$	$2^{17.745}$
(incl. interred)	$2^{17.586}$	$2^{17.679}$					$2^{16.703}$	$2^{16.785}$				
HRandom(10, 2, 2)	$2^{18.291}$	$2^{18.369}$	$2^{19.271}$	$2^{19.271}$	$2^{20.099}$	$2^{20.156}$	$2^{18.249}$	$2^{18.339}$	$2^{17.491}$	$2^{17.491}$	$2^{20.072}$	$2^{20.120}$
(incl. interred)	$2^{19.893}$	$2^{19.949}$					$2^{18.889}$	$2^{18.947}$				
HRandom(11, 2, 2)	$2^{20.217}$	$2^{20.299}$	$2^{21.169}$	$2^{21.169}$	$2^{22.524}$	$2^{22.563}$	$2^{20.448}$	$2^{20.510}$	$2^{19.105}$	$2^{19.105}$	$2^{22.490}$	$2^{22.507}$
(incl. interred)	$2^{22.013}$	$2^{22.057}$					$2^{21.078}$	$2^{21.118}$				
HRandom(12, 2, 2)	$2^{22.508}$	$2^{22.552}$	$2^{23.200}$	$2^{23.200}$	$2^{24.955}$	$2^{24.980}$	$2^{22.636}$	$2^{22.679}$	$2^{21.024}$	$2^{21.024}$	$2^{24.928}$	$2^{24.949}$
(incl. interred)	$2^{24.358}$	$2^{24.380}$					$2^{23.367}$	$2^{23.394}$				
HRandom(13, 2, 2)	$2^{24.684}$	$2^{24.805}$	$2^{25.310}$	$2^{25.310}$	$2^{27.409}$	$2^{27.427}$	$2^{24.937}$	$2^{24.966}$	$2^{22.506}$	$2^{22.506}$	$2^{27.370}$	$2^{27.373}$
(incl. interred)	$2^{26.562}$	$2^{26.606}$					$2^{25.651}$	$2^{25.669}$				
HRandom(14, 2, 2)							$2^{26.989}$	$2^{27.011}$	$2^{24.273}$	$2^{24.273}$	$2^{29.792}$	$2^{29.801}$
(incl. interred)							$2^{27.844}$	$2^{27.856}$				

Table 6
# multiplications (incl. interreductions) (homogeneous).

Benchmark	only top $\mathfrak{s}$-reductions						full $\mathfrak{s}$-reductions					
	$<_{\text{pot}}$		$<_{\text{lt-pot}}$		$<_{\text{d-pot}}$		$<_{\text{pot}}$		$<_{\text{lt-pot}}$		$<_{\text{d-pot}}$	
	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$
cyclic-7	$2^{24.276}$	$2^{23.871}$	$2^{24.024}$	$2^{23.996}$	$2^{23.851}$	$2^{23.467}$	$2^{24.050}$	$2^{23.598}$	$2^{24.257}$	$2^{24.133}$	$2^{24.319}$	$2^{23.866}$
(incl. interred)	$2^{24.429}$	$2^{24.076}$					$2^{24.130}$	$2^{23.709}$				
cyclic-8	$2^{31.011}$	$2^{29.800}$	$2^{29.796}$	$2^{29.543}$	$2^{30.427}$	$2^{29.361}$	$2^{30.890}$	$2^{29.641}$	$2^{30.162}$	$2^{29.738}$	$2^{31.363}$	$2^{30.127}$
(incl. interred)	$2^{31.287}$	$2^{30.374}$					$2^{31.086}$	$2^{30.066}$				
eco-10	$2^{24.459}$	$2^{24.148}$	$2^{24.120}$	$2^{24.102}$	$2^{23.676}$	$2^{23.394}$	$2^{25.484}$	$2^{24.560}$	$2^{24.046}$	$2^{23.975}$	$2^{25.292}$	$2^{24.480}$
(incl. interred)	$2^{24.582}$	$2^{24.294}$					$2^{25.514}$	$2^{24.619}$				
eco-11	$2^{27.765}$	$2^{27.229}$	$2^{26.904}$	$2^{26.895}$	$2^{26.556}$	$2^{26.111}$	$2^{29.186}$	$2^{27.564}$	$2^{27.216}$	$2^{27.098}$	$2^{28.630}$	$2^{27.505}$
(incl. interred)	$2^{27.881}$	$2^{27.392}$					$2^{29.203}$	$2^{27.619}$				
f-633	$2^{12.149}$	$2^{12.024}$	$2^{12.776}$	$2^{12.744}$	$2^{12.029}$	$2^{11.764}$	$2^{12.069}$	$2^{12.166}$	$2^{12.716}$	$2^{12.607}$	$2^{12.120}$	$2^{11.809}$
(incl. interred)	$2^{12.228}$	$2^{12.109}$										
f-744	$2^{21.443}$	$2^{21.480}$	$2^{21.888}$	$2^{21.799}$	$2^{21.654}$	$2^{21.656}$	$2^{21.767}$	$2^{21.414}$	$2^{21.798}$	$2^{21.695}$	$2^{21.842}$	$2^{21.683}$
(incl. interred)	$2^{21.613}$	$2^{21.664}$					$2^{21.795}$	$2^{21.452}$				
katsura-11	$2^{27.790}$	$2^{27.539}$	$2^{29.423}$	$2^{29.366}$	$2^{28.251}$	$2^{28.072}$	$2^{29.937}$	$2^{29.809}$	$2^{29.268}$	$2^{29.314}$	$2^{29.290}$	$2^{29.208}$
(incl. interred)	$2^{30.118}$	$2^{30.064}$					$2^{30.408}$	$2^{30.315}$				
katsura-12	$2^{30.965}$	$2^{30.650}$	$2^{32.378}$	$2^{32.238}$	$2^{31.579}$	$2^{31.352}$	$2^{33.522}$	$2^{33.337}$	$2^{32.214}$	$2^{32.240}$	$2^{32.537}$	$2^{32.421}$
(incl. interred)	$2^{33.729}$	$2^{33.679}$					$2^{34.073}$	$2^{33.947}$				
noon-8	$2^{20.100}$	$2^{19.657}$	$2^{20.365}$	$2^{19.983}$	$2^{20.142}$	$2^{19.681}$	$2^{22.212}$	$2^{22.292}$	$2^{22.249}$	$2^{22.327}$	$2^{22.212}$	$2^{22.292}$
(incl. interred)												
noon-9	$2^{22.901}$	$2^{22.234}$	$2^{23.042}$	$2^{22.455}$	$2^{22.901}$	$2^{22.234}$	$2^{25.142}$	$2^{25.212}$	$2^{25.165}$	$2^{25.234}$	$2^{25.142}$	$2^{25.212}$
(incl. interred)												
HRandom(6, 2, 2)	$2^{14.628}$	$2^{14.812}$	$2^{15.453}$	$2^{15.453}$	$2^{15.403}$	$2^{15.585}$	$2^{14.626}$	$2^{14.827}$	$2^{14.614}$	$2^{14.614}$	$2^{15.180}$	$2^{15.340}$
(incl. interred)	$2^{15.502}$	$2^{15.615}$					$2^{14.932}$	$2^{15.096}$				
HRandom(7, 2, 2)	$2^{17.592}$	$2^{17.729}$	$2^{18.409}$	$2^{18.409}$	$2^{18.430}$	$2^{18.566}$	$2^{17.448}$	$2^{17.602}$	$2^{17.275}$	$2^{17.275}$	$2^{18.217}$	$2^{18.316}$
(incl. interred)	$2^{18.505}$	$2^{18.608}$					$2^{17.806}$	$2^{17.927}$				
HRandom(7, 2, 4)	$2^{23.387}$	$2^{23.432}$	$2^{23.509}$	$2^{23.509}$	$2^{23.906}$	$2^{24.003}$	$2^{22.172}$	$2^{22.241}$	$2^{21.820}$	$2^{21.820}$	$2^{22.992}$	$2^{23.136}$
(incl. interred)	$2^{23.763}$	$2^{23.808}$					$2^{22.246}$	$2^{22.312}$				
HRandom(7, 2, 6)	$2^{33.815}$	$2^{33.837}$	$2^{33.676}$	$2^{33.676}$	$2^{34.197}$	$2^{34.256}$	$2^{32.492}$	$2^{32.495}$	$2^{31.905}$	$2^{31.905}$	$2^{33.185}$	$2^{33.239}$
(incl. interred)	$2^{34.120}$	$2^{34.142}$					$2^{32.541}$	$2^{32.544}$				
HRandom(8, 2, 2)	$2^{20.337}$	$2^{20.506}$	$2^{21.167}$	$2^{21.167}$	$2^{21.455}$	$2^{21.544}$	$2^{20.288}$	$2^{20.398}$	$2^{19.965}$	$2^{19.965}$	$2^{21.233}$	$2^{21.315}$
(incl. interred)	$2^{21.430}$	$2^{21.532}$					$2^{20.723}$	$2^{20.806}$				
HRandom(8, 2, 4)	$2^{29.458}$	$2^{29.498}$	$2^{29.617}$	$2^{29.617}$	$2^{30.086}$	$2^{30.171}$	$2^{28.285}$	$2^{28.307}$	$2^{27.830}$	$2^{27.830}$	$2^{29.250}$	$2^{29.361}$
(incl. interred)	$2^{29.869}$	$2^{29.904}$					$2^{28.397}$	$2^{28.418}$				
HRandom(9, 2, 2)	$2^{23.220}$	$2^{23.348}$	$2^{23.913}$	$2^{23.913}$	$2^{24.515}$	$2^{24.572}$	$2^{23.209}$	$2^{23.282}$	$2^{22.474}$	$2^{22.474}$	$2^{24.250}$	$2^{24.295}$
(incl. interred)	$2^{24.351}$	$2^{24.436}$					$2^{23.656}$	$2^{23.710}$				
HRandom(10, 2, 2)	$2^{26.107}$	$2^{26.187}$	$2^{26.596}$	$2^{26.596}$	$2^{27.576}$	$2^{27.609}$	$2^{26.082}$	$2^{26.133}$	$2^{25.097}$	$2^{25.097}$	$2^{27.301}$	$2^{27.332}$
(incl. interred)	$2^{27.328}$	$2^{27.377}$					$2^{26.589}$	$2^{26.625}$				
HRandom(11, 2, 2)	$2^{28.985}$	$2^{29.069}$	$2^{29.278}$	$2^{29.278}$	$2^{30.684}$	$2^{30.703}$	$2^{29.008}$	$2^{29.041}$	$2^{27.607}$	$2^{27.607}$	$2^{30.351}$	$2^{30.362}$
(incl. interred)	$2^{30.234}$	$2^{30.277}$					$2^{29.512}$	$2^{29.535}$				
HRandom(12, 2, 2)	$2^{31.888}$	$2^{31.953}$	$2^{32.059}$	$2^{32.059}$	$2^{33.791}$	$2^{33.802}$	$2^{31.914}$	$2^{31.936}$	$2^{30.326}$	$2^{30.326}$	$2^{33.462}$	$2^{33.472}$
(incl. interred)	$2^{33.203}$	$2^{33.232}$					$2^{32.468}$	$2^{32.483}$				
HRandom(13, 2, 2)	$2^{34.837}$	$2^{34.975}$	$2^{34.881}$	$2^{34.881}$	$2^{36.959}$	$2^{36.966}$	$2^{34.856}$	$2^{34.870}$	$2^{32.855}$	$2^{32.855}$	$2^{36.581}$	$2^{36.583}$
(incl. interred)	$2^{36.137}$	$2^{36.198}$					$2^{35.405}$	$2^{35.415}$				
HRandom(14, 2, 2)							$2^{37.720}$	$2^{37.729}$	$2^{35.562}$	$2^{35.562}$	$2^{39.738}$	$2^{39.741}$
(incl. interred)							$2^{38.315}$	$2^{38.321}$				

Table 7
# zero reductions (affine).

Benchmark	$<_{\text{pot}}$		$<_{\text{lt-pot}}$		$<_{\text{d-pot}}$	
	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$
cyclic-7	36	36	145	145	36	36
cyclic-8	244	244	672	672	244	244
eco-10	0	0	367	367	367	367
eco-11	0	0	749	749	749	749
f-633	1	1	9	9	3	3
f-744	1	1	259	259	188	188
katsura-11	0	0	353	353	0	0
katsura-12	0	0	640	640	0	0
noon-8	0	0	294	294	0	0
noon-9	0	0	682	682	0	0
Random(6, 2, 2)	0	0	26	26	0	0
Random(7, 2, 2)	0	0	49	49	0	0
Random(8, 2, 2)	0	0	102	102	0	0
Random(9, 2, 2)	0	0	181	181	0	0
Random(10, 2, 2)	0	0	339	339	0	0
Random(11, 2, 2)	0	0	590	590	0	0
Random(12, 2, 2)	0	0	1083	1083	0	0
Random(13, 2, 2)	0	0	1867	1867	0	0
Random(14, 2, 2)	0	0	3403	3403	0	0

Table 8
Size of $\mathcal{G}$ (affine).

Benchmark	$<_{\text{pot}}$		$<_{\text{lt-pot}}$		$<_{\text{d-pot}}$	
	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$
cyclic-7	779	679	871	848	949	751
cyclic-8	3559	2775	4074	3658	5534	3884
eco-10	522	405	541	478	782	671
eco-11	1055	774	1092	965	1717	1415
f-633	60	58	61	59	61	57
f-744	468	442	377	348	502	464
katsura-11	762	743	553	553	2188	2161
katsura-12	1473	1474	1076	1076	6020	6020
noon-8	1384	1390	1384	1389	1384	1389
noon-9	3743	3750	3743	3749	3743	3749
Random(6, 2, 2)	58	58	39	39	62	62
Random(7, 2, 2)	113	113	67	67	124	124
Random(8, 2, 2)	212	212	128	128	242	242
Random(9, 2, 2)	365	366	223	223	479	479
Random(10, 2, 2)	677	677	426	426	932	932
Random(11, 2, 2)	1327	1357	767	767	1832	1832
Random(12, 2, 2)	2502	2537	1463	1463	3557	3557
Random(13, 2, 2)	4879	4879	2708	2708	6973	6973
Random(14, 2, 2)	9259	9259	5142	5142	13924	13934

(a) $<$ denotes the monomial order as well as the compatible module monomial order. We have seen in Section 14 that this order has the biggest impact.

(b) $\trianglelefteq$ denotes the rewrite order. If **RB** handles various elements of the same signature, only one needs to be further $\mathfrak{s}$-reduced. The rewrite order give a unique choice which element is chosen

Table 9
Size of $\mathscr{H}$ (affine).

Benchmark	$<_{\text{pot}}$		$<_{\text{lt-pot}}$		$<_{\text{d-pot}}$	
	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$
cyclic-7	10011	10011	187	187	439	338
cyclic-8	186966	189420	761	761	3691	2599
eco-10	21528	21528	412	412	2809	2531
eco-11	73153	71253	804	804	6869	5914
f-633	120	120	37	37	40	40
f-744	20503	19701	316	316	641	628
katsura-11	40755	35511	408	408	2728	2670
katsura-12	134940	134940	706	706	9065	9148
noon-8	406	406	322	322	84	84
noon-9	666	666	718	718	120	120
Random$(6, 2, 2)$	378	378	41	41	57	57
Random$(7, 2, 2)$	1326	1326	70	70	119	119
Random$(8, 2, 2)$	4465	4465	130	130	243	243
Random$(9, 2, 2)$	8256	8385	217	217	485	485
Random$(10, 2, 2)$	25200	25200	384	384	964	964
Random$(11, 2, 2)$	91378	104653	645	645	1896	1896
Random$(12, 2, 2)$	302253	330078	1149	1149	3728	3728
Random$(13, 2, 2)$	1070916	1070916	1945	1945	7285	7285
Random$(14, 2, 2)$	3667986	3667986	3494	3494	15122	15122

and which are removed. In Section 14 we have seen that the outcomes of using different specializations of $\trianglelefteq$, namely $\trianglelefteq_{\text{add}}$ and $\trianglelefteq_{\text{rat}}$ are nearly equivalent when it comes to the number of operations.

(c) $\preceq$ denotes the order in which S-pairs are handled in **RB**. Nearly all known efficient instantiations use $\preceq_{\mathfrak{s}}$, so S-pairs are handled by increasing signature.

Thus every known algorithm, like **F5** or **GVW** can be implemented with any of the above 3 choices. Even so some of those algorithms are presented in a restricted setting, for example **G2V** for $<_{\text{pot}}$ only, they all can be seen as different, specialized implementations of **RB** and thus are connected to each other. We covered all attempts to signature-based algorithms known and gave a dictionary for translating different notations used in the corresponding publications. Thus this survey can also be used as a reference for researcher interested in this topic.

Important aspects when optimizing **RB** and further open questions are the following:

(a) Ensuring termination algorithmically as presented in Section 10.2 can lead to earlier termination and thus improved behavior of the algorithm by using different techniques to detect the completeness of $\mathscr{G}$.

(b) Exploiting algebraic structures is an area of high research at the moment (Section 9). Developments in this direction might have a huge impact on the computations of (signature) Gröbner bases in the near future and are promising in decreasing the complexity of computations.

(c) Using linear algebra for the reduction process as illustrated in Section 13 is another field where a lot more optimizations can be expected. At the moment, restrictions to $\mathfrak{s}$-reductions lead to restrictions swapping rows during the Gaussian Elimination. Getting more flexible and possibly able to use (at least some of) the ideas from Faugère and Lachartre (2010) is still an open problem.

(d) Even if we are only interested in computing a Gröbner basis (and not the complete syzygy) can one generalize the usage of signatures: Instead of choosing only the leading term we could take some of the higher terms of the module representation and find an intermediate representation between sig-poly pairs $(\mathfrak{s}(\alpha), \overline{\alpha}) \in \mathscr{R}^m \times \mathscr{R}$ and full module representations $\alpha \in \mathscr{R}^m$? This would relax the $\mathfrak{s}$-reduction allowing some same signature reductions. Where is the breaking point of

Table 10
$\mathfrak{s}$-reductions (incl. interreductions) (affine).

Benchmark	only top $\mathfrak{s}$-reductions						full $\mathfrak{s}$-reductions					
	$<_{\text{pot}}$		$<_{\text{lt-pot}}$		$<_{\text{d-pot}}$		$<_{\text{pot}}$		$<_{\text{lt-pot}}$		$<_{\text{d-pot}}$	
	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$
cyclic-7	$2^{17.136}$	$2^{16.774}$	$2^{18.247}$	$2^{18.117}$	$2^{18.093}$	$2^{17.630}$	$2^{16.792}$	$2^{16.492}$	$2^{16.977}$	$2^{16.881}$	$2^{16.984}$	$2^{16.829}$
(incl. interred)	$2^{17.383}$	$2^{17.133}$					$2^{16.854}$	$2^{16.569}$				
cyclic-8	$2^{22.533}$	$2^{21.343}$	$2^{22.742}$	$2^{22.312}$	$2^{23.598}$	$2^{22.286}$	$2^{22.454}$	$2^{21.273}$	$2^{21.810}$	$2^{21.439}$	$2^{22.693}$	$2^{21.504}$
(incl. interred)	$2^{22.890}$	$2^{22.093}$					$2^{22.669}$	$2^{21.730}$				
eco-10	$2^{17.827}$	$2^{16.397}$	$2^{20.148}$	$2^{20.124}$	$2^{20.251}$	$2^{20.114}$	$2^{19.858}$	$2^{17.368}$	$2^{19.964}$	$2^{19.807}$	$2^{20.017}$	$2^{19.773}$
(incl. interred)	$2^{18.181}$	$2^{17.170}$					$2^{19.911}$	$2^{17.633}$				
eco-11	$2^{20.872}$	$2^{18.652}$	$2^{22.605}$	$2^{22.567}$	$2^{22.720}$	$2^{22.563}$	$2^{23.620}$	$2^{19.822}$	$2^{22.497}$	$2^{22.307}$	$2^{22.511}$	$2^{22.290}$
(incl. interred)	$2^{21.208}$	$2^{19.755}$					$2^{23.651}$	$2^{20.177}$				
f-633	$2^{9.644}$	$2^{9.426}$	$2^{10.828}$	$2^{10.768}$	$2^{10.131}$	$2^{9.833}$	$2^{9.401}$	$2^{9.202}$	$2^{9.977}$	$2^{9.713}$	$2^{9.386}$	$2^{9.031}$
(incl. interred)	$2^{9.730}$	$2^{9.526}$										
f-744	$2^{15.037}$	$2^{14.422}$	$2^{17.500}$	$2^{17.526}$	$2^{17.500}$	$2^{17.579}$	$2^{15.305}$	$2^{14.829}$	$2^{16.996}$	$2^{16.872}$	$2^{17.123}$	$2^{17.089}$
(incl. interred)	$2^{15.535}$	$2^{15.361}$					$2^{15.601}$	$2^{15.100}$				
katsura-11	$2^{18.856}$	$2^{18.627}$	$2^{22.411}$	$2^{22.395}$	$2^{20.837}$	$2^{20.730}$	$2^{22.191}$	$2^{22.055}$	$2^{21.081}$	$2^{21.280}$	$2^{21.994}$	$2^{21.944}$
(incl. interred)	$2^{22.478}$	$2^{22.438}$					$2^{22.620}$	$2^{22.537}$				
katsura-12	$2^{21.302}$	$2^{21.254}$	$2^{24.540}$	$2^{24.528}$	$2^{23.412}$	$2^{23.288}$	$2^{24.980}$	$2^{24.867}$	$2^{23.689}$	$2^{23.741}$	$2^{24.488}$	$2^{24.437}$
(incl. interred)	$2^{25.391}$	$2^{25.372}$					$2^{25.648}$	$2^{25.575}$				
noon-8	$2^{15.693}$	$2^{14.519}$	$2^{16.097}$	$2^{15.308}$	$2^{15.691}$	$2^{14.529}$	$2^{17.602}$	$2^{17.408}$	$2^{17.758}$	$2^{17.582}$	$2^{17.667}$	$2^{17.479}$
(incl. interred)	$2^{15.694}$	$2^{14.523}$										
noon-9	$2^{18.197}$	$2^{16.651}$	$2^{18.523}$	$2^{17.422}$	$2^{18.202}$	$2^{16.650}$	$2^{20.203}$	$2^{19.962}$	$2^{20.312}$	$2^{20.084}$	$2^{20.243}$	$2^{20.003}$
(incl. interred)	$2^{18.198}$	$2^{16.652}$										
Random(6, 2, 2)	$2^{10.008}$	$2^{10.189}$	$2^{13.068}$	$2^{13.068}$	$2^{10.752}$	$2^{10.946}$	$2^{11.008}$	$2^{11.274}$	$2^{11.696}$	$2^{11.696}$	$2^{11.659}$	$2^{11.814}$
(incl. interred)	$2^{11.342}$	$2^{11.434}$					$2^{11.218}$	$2^{11.451}$				
Random(7, 2, 2)	$2^{12.137}$	$2^{12.267}$	$2^{15.076}$	$2^{15.076}$	$2^{13.018}$	$2^{13.169}$	$2^{13.110}$	$2^{13.322}$	$2^{13.393}$	$2^{13.393}$	$2^{14.010}$	$2^{14.110}$
(incl. interred)	$2^{13.632}$	$2^{13.752}$					$2^{13.382}$	$2^{13.559}$				
Random(8, 2, 2)	$2^{13.984}$	$2^{14.212}$	$2^{17.269}$	$2^{17.269}$	$2^{15.297}$	$2^{15.406}$	$2^{15.168}$	$2^{15.345}$	$2^{15.285}$	$2^{15.285}$	$2^{16.353}$	$2^{16.435}$
(incl. interred)	$2^{15.911}$	$2^{16.021}$					$2^{15.561}$	$2^{15.698}$				
Random(9, 2, 2)	$2^{16.106}$	$2^{16.210}$	$2^{19.294}$	$2^{19.294}$	$2^{17.656}$	$2^{17.729}$	$2^{17.438}$	$2^{17.562}$	$2^{16.906}$	$2^{16.906}$	$2^{18.755}$	$2^{18.805}$
(incl. interred)	$2^{18.019}$	$2^{18.225}$					$2^{17.876}$	$2^{17.968}$				
Random(10, 2, 2)	$2^{18.013}$	$2^{18.096}$	$2^{21.381}$	$2^{21.381}$	$2^{20.021}$	$2^{20.070}$	$2^{19.532}$	$2^{19.629}$	$2^{18.703}$	$2^{18.703}$	$2^{21.156}$	$2^{21.193}$
(incl. interred)	$2^{20.089}$	$2^{20.170}$					$2^{20.071}$	$2^{20.139}$				
Random(11, 2, 2)	$2^{20.117}$	$2^{20.165}$	$2^{23.397}$	$2^{23.397}$	$2^{22.418}$	$2^{22.450}$	$2^{21.859}$	$2^{21.923}$	$2^{20.377}$	$2^{20.377}$	$2^{23.591}$	$2^{23.605}$
(incl. interred)	$2^{22.409}$	$2^{22.476}$					$2^{22.391}$	$2^{22.436}$				
Random(12, 2, 2)	$2^{22.039}$	$2^{22.087}$	$2^{25.447}$	$2^{25.447}$	$2^{24.833}$	$2^{24.854}$	$2^{24.089}$	$2^{24.134}$	$2^{22.169}$	$2^{22.169}$	$2^{26.028}$	$2^{26.044}$
(incl. interred)	$2^{24.512}$	$2^{24.597}$					$2^{24.737}$	$2^{24.766}$				
Random(13, 2, 2)							$2^{26.472}$	$2^{26.501}$	$2^{23.917}$	$2^{23.917}$	$2^{28.463}$	$2^{28.465}$
(incl. interred)							$2^{27.079}$	$2^{27.097}$				
Random(14, 2, 2)							$2^{28.496}$	$2^{28.517}$	$2^{25.697}$	$2^{25.697}$	$2^{30.897}$	$2^{30.886}$
(incl. interred)							$2^{29.255}$	$2^{29.268}$				

Table 11
# multiplications (incl. interreductions) (affine).

Benchmark	only top $\mathfrak{s}$-reductions						full $\mathfrak{s}$-reductions					
	$<_{\text{pot}}$		$<_{\text{lt-pot}}$		$<_{\text{d-pot}}$		$<_{\text{pot}}$		$<_{\text{lt-pot}}$		$<_{\text{d-pot}}$	
	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$	$\trianglelefteq_{\text{add}}$	$\trianglelefteq_{\text{rat}}$
cyclic-7	$2^{24.253}$	$2^{23.830}$	$2^{24.008}$	$2^{23.989}$	$2^{23.866}$	$2^{23.482}$	$2^{24.065}$	$2^{23.612}$	$2^{24.220}$	$2^{24.105}$	$2^{24.229}$	$2^{23.832}$
(incl. interred)	$2^{24.366}$	$2^{23.986}$					$2^{24.122}$	$2^{23.679}$				
cyclic-8	$2^{30.906}$	$2^{29.738}$	$2^{29.788}$	$2^{29.534}$	$2^{30.435}$	$2^{29.367}$	$2^{30.885}$	$2^{29.649}$	$2^{30.261}$	$2^{29.837}$	$2^{31.273}$	$2^{30.076}$
(incl. interred)	$2^{31.169}$	$2^{30.276}$					$2^{31.091}$	$2^{30.092}$				
eco-10	$2^{22.815}$	$2^{21.728}$	$2^{24.428}$	$2^{24.509}$	$2^{24.608}$	$2^{24.492}$	$2^{25.262}$	$2^{22.866}$	$2^{25.001}$	$2^{24.889}$	$2^{24.996}$	$2^{24.758}$
(incl. interred)	$2^{23.141}$	$2^{22.336}$					$2^{25.302}$	$2^{23.062}$				
eco-11	$2^{26.502}$	$2^{24.807}$	$2^{27.357}$	$2^{27.421}$	$2^{27.574}$	$2^{27.433}$	$2^{29.367}$	$2^{25.900}$	$2^{28.161}$	$2^{27.961}$	$2^{28.087}$	$2^{27.867}$
(incl. interred)	$2^{26.823}$	$2^{25.662}$					$2^{29.398}$	$2^{26.194}$				
f-633	$2^{11.817}$	$2^{11.694}$	$2^{13.360}$	$2^{13.352}$	$2^{12.530}$	$2^{12.196}$	$2^{11.772}$	$2^{11.676}$	$2^{12.636}$	$2^{12.487}$	$2^{11.998}$	$2^{11.737}$
(incl. interred)	$2^{11.870}$	$2^{11.751}$										
f-744	$2^{19.292}$	$2^{18.824}$	$2^{22.257}$	$2^{22.351}$	$2^{22.040}$	$2^{22.231}$	$2^{19.776}$	$2^{19.329}$	$2^{21.737}$	$2^{21.626}$	$2^{21.839}$	$2^{21.774}$
(incl. interred)	$2^{19.701}$	$2^{19.571}$					$2^{20.018}$	$2^{19.574}$				
katsura-11	$2^{27.943}$	$2^{27.683}$	$2^{29.565}$	$2^{29.518}$	$2^{28.714}$	$2^{28.531}$	$2^{29.884}$	$2^{29.747}$	$2^{28.906}$	$2^{29.074}$	$2^{29.449}$	$2^{29.381}$
(incl. interred)	$2^{30.080}$	$2^{30.010}$					$2^{30.353}$	$2^{30.264}$				
katsura-12	$2^{31.121}$	$2^{30.819}$	$2^{32.451}$	$2^{32.365}$	$2^{32.092}$	$2^{31.867}$	$2^{33.503}$	$2^{33.263}$	$2^{32.052}$	$2^{32.173}$	$2^{32.712}$	$2^{32.620}$
(incl. interred)	$2^{33.642}$	$2^{33.586}$					$2^{34.102}$	$2^{33.946}$				
noon-8	$2^{20.056}$	$2^{19.653}$	$2^{20.264}$	$2^{19.887}$	$2^{20.114}$	$2^{19.689}$	$2^{21.588}$	$2^{21.714}$	$2^{21.678}$	$2^{21.798}$	$2^{21.623}$	$2^{21.747}$
(incl. interred)	$2^{20.057}$	$2^{19.654}$										
noon-9	$2^{22.844}$	$2^{22.216}$	$2^{22.945}$	$2^{22.356}$	$2^{22.854}$	$2^{22.218}$	$2^{24.490}$	$2^{24.615}$	$2^{24.546}$	$2^{24.667}$	$2^{24.511}$	$2^{24.634}$
(incl. interred)												
Random(6, 2, 2)	$2^{15.986}$	$2^{16.176}$	$2^{18.567}$	$2^{18.567}$	$2^{16.548}$	$2^{16.683}$	$2^{16.617}$	$2^{16.799}$	$2^{17.303}$	$2^{17.303}$	$2^{16.988}$	$2^{17.122}$
(incl. interred)	$2^{16.883}$	$2^{17.001}$					$2^{16.823}$	$2^{16.982}$				
Random(7, 2, 2)	$2^{18.938}$	$2^{19.090}$	$2^{21.285}$	$2^{21.285}$	$2^{19.618}$	$2^{19.708}$	$2^{19.548}$	$2^{19.674}$	$2^{19.885}$	$2^{19.885}$	$2^{19.970}$	$2^{20.057}$
(incl. interred)	$2^{19.896}$	$2^{20.017}$					$2^{19.792}$	$2^{19.900}$				
Random(8, 2, 2)	$2^{21.745}$	$2^{21.956}$	$2^{24.219}$	$2^{24.219}$	$2^{22.720}$	$2^{22.777}$	$2^{22.463}$	$2^{22.556}$	$2^{22.706}$	$2^{22.706}$	$2^{22.990}$	$2^{23.056}$
(incl. interred)	$2^{22.865}$	$2^{22.990}$					$2^{22.758}$	$2^{22.834}$				
Random(9, 2, 2)	$2^{24.760}$	$2^{24.885}$	$2^{26.994}$	$2^{26.994}$	$2^{25.853}$	$2^{25.887}$	$2^{25.443}$	$2^{25.505}$	$2^{25.278}$	$2^{25.278}$	$2^{25.952}$	$2^{25.990}$
(incl. interred)	$2^{25.888}$	$2^{26.020}$					$2^{25.756}$	$2^{25.807}$				
Random(10, 2, 2)	$2^{27.602}$	$2^{27.690}$	$2^{29.876}$	$2^{29.876}$	$2^{29.022}$	$2^{29.043}$	$2^{28.360}$	$2^{28.404}$	$2^{28.060}$	$2^{28.060}$	$2^{28.986}$	$2^{29.012}$
(incl. interred)	$2^{28.820}$	$2^{28.889}$					$2^{28.709}$	$2^{28.743}$				
Random(11, 2, 2)	$2^{30.564}$	$2^{30.645}$	$2^{32.708}$	$2^{32.708}$	$2^{32.214}$	$2^{32.226}$	$2^{31.334}$	$2^{31.362}$	$2^{30.725}$	$2^{30.725}$	$2^{31.979}$	$2^{31.989}$
(incl. interred)	$2^{31.836}$	$2^{31.880}$					$2^{31.692}$	$2^{31.715}$				
Random(12, 2, 2)	$2^{33.432}$	$2^{33.515}$	$2^{35.602}$	$2^{35.602}$	$2^{35.437}$	$2^{35.444}$	$2^{34.272}$	$2^{34.291}$	$2^{33.523}$	$2^{33.523}$	$2^{35.028}$	$2^{35.037}$
(incl. interred)	$2^{34.771}$	$2^{34.821}$					$2^{34.674}$	$2^{34.688}$				
Random(13, 2, 2)							$2^{37.253}$	$2^{37.266}$	$2^{36.278}$	$2^{36.278}$	$2^{38.084}$	$2^{38.085}$
(incl. interred)							$2^{37.651}$	$2^{37.661}$				
Random(14, 2, 2)							$2^{40.141}$	$2^{40.149}$	$2^{39.080}$	$2^{39.080}$	$2^{41.140}$	$2^{41.132}$
(incl. interred)							$2^{40.582}$	$2^{40.588}$				

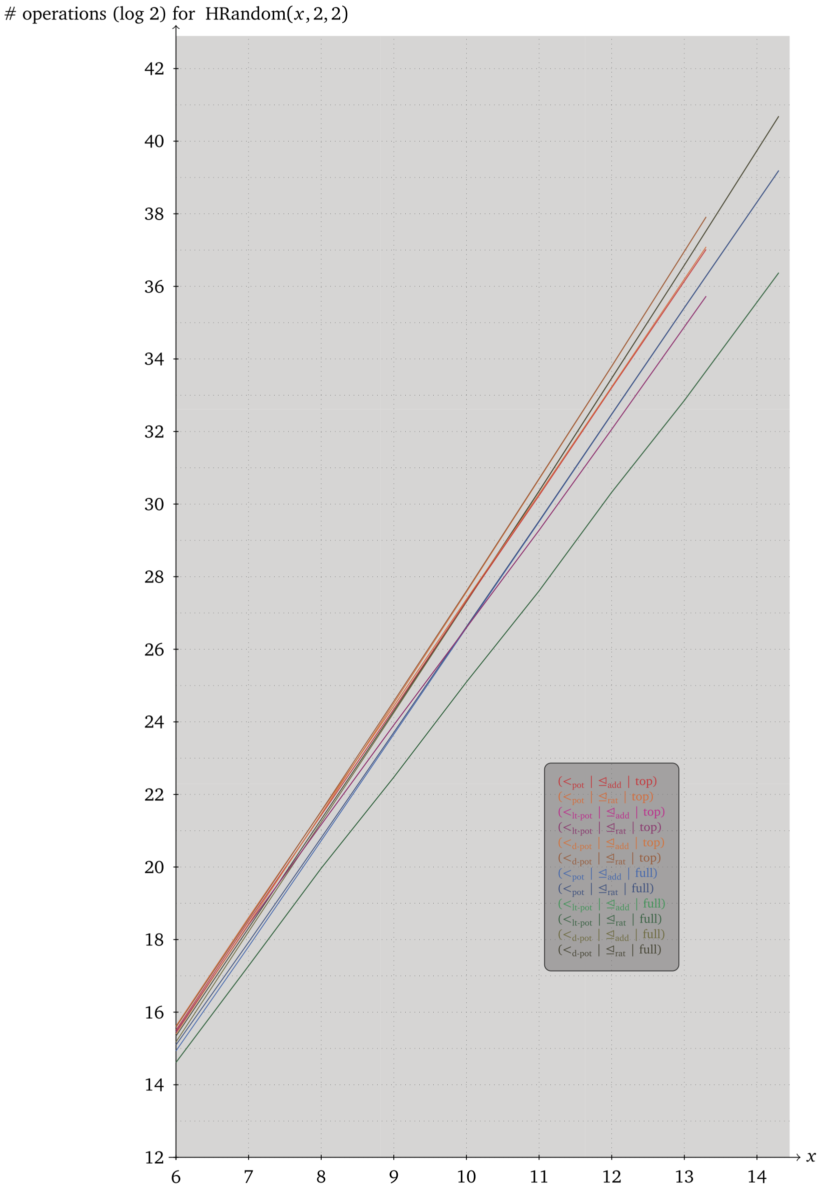


Fig. 7. Number of multiplications for homogeneous random examples by increasing number of generators, all of degree 2.

using more terms from the module representation in order to interreduce the syzygy elements even further and not adding too much overhead in time and memory?

We hope that this survey helps to give a better understanding on signature-based algorithms for Gröbner bases. Moreover, we would like to give researchers new to this area a guide to find their way through the enormous number of publications that have been released on this topic over the

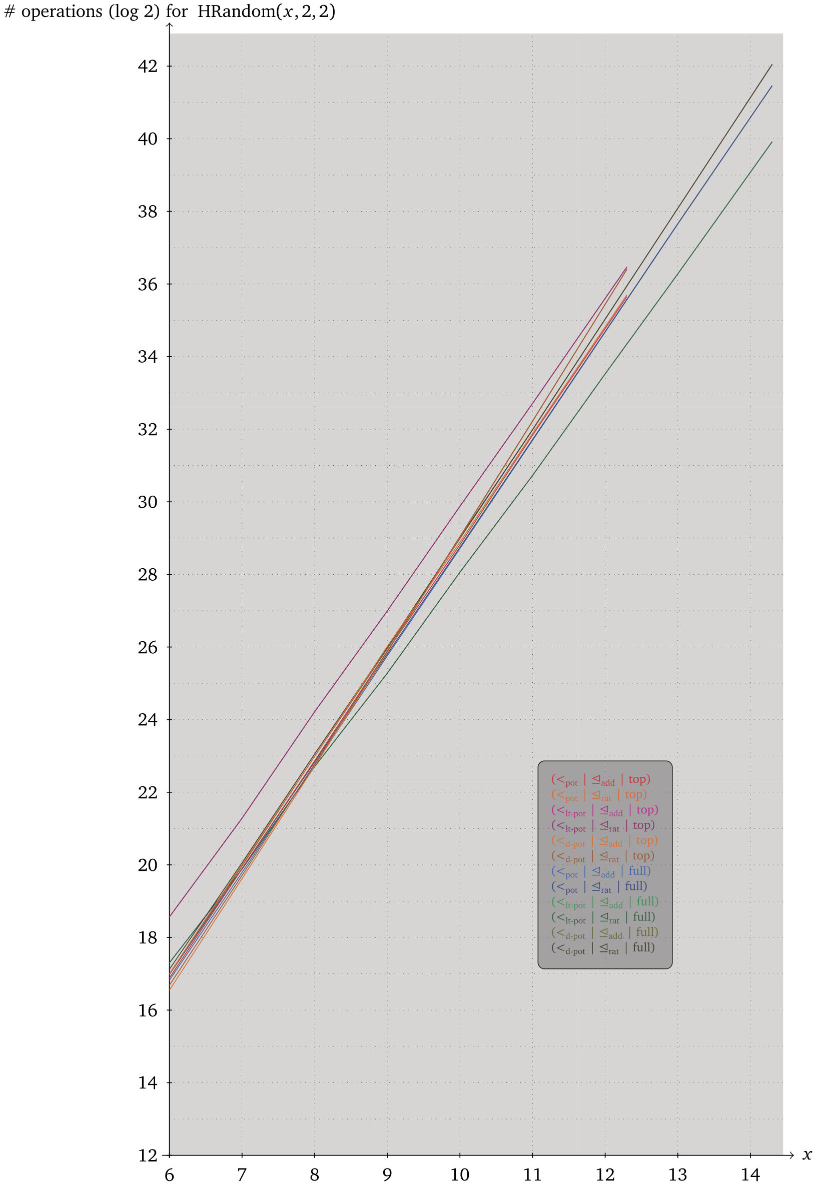


Fig. 8. Number of multiplications for affine random examples by increasing number of generators, all of degree 2.

last years. Even more, we hope that this survey could also facilitate collaboration among experts for pushing the field of Gröbner basis computations even further.

Acknowledgements

We would like to thank all the anonymous referees for their helpful suggestions that improved the survey quite a lot. In particular, we would like to thank John Perry for lots of supporting discussions.

References

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

Arri, A., Perry, J., 2011. The F5 Criterion revised. J. Symb. Comput. 46 (2), 1017–1029. Preprint online at http://arxiv.org/abs/1012.3664.

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

Ars, G., Hashemi, A., 2010. Extended F5 criteria. J. Symb. Comput. 45 (12), 1330–1340. MEGA 2009 special issue.

Bardet, M., 2004a. Étude des systèmes algébriques surdéterminés. Applications aux codes correcteurs et à la cryptographie. PhD thesis. Université Paris 6.

Bardet, M., Faugère, J.-C., Salvy, B., Yang, B.Y., 2005. Asymptotic expansion of the degree of regularity for semi-regular systems of equations. In: The Effective Methods in Algebraic Geometry Conference. Mega 2005, pp. 1–14.

Becker, T., Weispfenning, V., Kredel, H., 1993. Gröbner Bases – A Computational Approach to Commutative Algebra. Graduate Texts in Mathematics. Springer Verlag.

Bini, D.A., Mourrain, B., 2012. Polynomial test suite. http://www-sop.inria.fr/saga/POL/.

Brickenstein, M., 2010. Slimgb: Gröbner bases with slim polynomials. Rev. Mat. Complut. 23 (2), 453–466, the final publication is available at www.springerlink.com.

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

Buchberger, B., 1970. Ein algorithmisches Kriterium für die Lösbarkeit eines algebraischen Gleichungssystems. Aequ. Math. 4 (3), 374–383. English translation in: B. Buchberger, F. Winkler: Groebner Bases and Applications, Proc. of the International Conference "33 Years of Groebner Bases", 1998, RISC, Austria, London Math. Society Lecture Note Series 251, Cambridge Univ. Press, 1998, pp. 535–545.

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

Buchberger, B., 1984. A critical-pair/completion algorithm for finitely generated ideals in rings. Lecture Notes in Computer Science 171, 137–161.

Buchberger, B., 1985. Gröbner bases: an algorithmic method in polynomial ideal theory. In: Bose, N.K. (Ed.), Multidimensional Systems Theory – Progress, Directions and Open Problems in Multidimensional Systems. D. Reidel Publication Company, pp. 184–232.

Buchberger, B., 1987. History and basic features of the critical-pair/completion procedure. J. Symb. Comput. 3 (1/2), 3–38.

Buchberger, B., 2006. An algorithm for finding the basis elements of the residue class ring of zero dimensional polynomial ideal. J. Symb. Comput. 41 (3–4), 475–511 (English translation of Bruno Buchberger's PhD thesis).

Cox, D.A., Little, J., O'Shea, D.B., 2007. Ideals, Varieties, and Algorithms, 3rd edition. Undergraduate Texts in Mathematics. Springer.

Decker, Wolfram, Greuel, Gert-Martin, Pfister, Gerhard, Schönemann, Hans, 2015. SINGULAR 4-0-2 – a computer algebra system for polynomial computations. http://www.singular.uni-kl.de.

Dellaca, R.D., 2009. Gröbner basis algorithms. PhD thesis. California State University, Fullerton.

Eder, C., 2008a. A new attempt on the F5 criterion. Comput. Sci. J. Mold. 16, 4–14.

Eder, C., 2008b. On the criteria of the F5 algorithm. Preprint arXiv:0804.2033 [math.AC].

Eder, C., 2012. Signature-based algorithms to compute standard bases. PhD thesis. University of Kaiserslautern, Germany. https://kluedo.ub.uni-kl.de/frontdoor/index/index/docId/2975.

Eder, C., 2013a. An analysis of inhomogeneous signature-based Gröbner basis computations. J. Symb. Comput. 59, 21–35.

Eder, C., 2013b. Improving incremental signature-based Groebner bases algorithms. ACM SIGSAM Commun. Comput. Algebra 47 (1), 1–13. http://arxiv.org/abs/1201.6472.

Eder, C., 2014. Predicting zero reductions in Gröbner basis computations. Preprint at http://arxiv.org/abs/1404.0161.

Eder, C., Gash, J., Perry, J., 2011. Modifying Faugère's F5 Algorithm to ensure termination. ACM SIGSAM Commun. Comput. Algebra 45 (2), 70–89. http://arxiv.org/abs/1006.0318.

Eder, C., Perry, J., 2010. F5C: a variant of Faugère's F5 algorithm with reduced Gröbner bases. J. Symb. Comput. 45 (12), 1442–1458. http://dx.doi.org/10.1016/j.jsc.2010.06.019. MEGA 2009 special issue.

Eder, C., Perry, J., 2011. Signature-based algorithms to compute Gröbner bases. In: Proceedings of the 2011 International Symposium on Symbolic and Algebraic Computation. ISSAC 2011, pp. 99–106.

Eder, C., Roune, B.H., 2013. Signature rewriting in Gröbner basis computation. In: Proceedings of the 2013 International Symposium on Symbolic and Algebraic Computation. ISSAC 2013, pp. 331–338.

Faugère, J.-C., 1999. A new efficient algorithm for computing Gröbner bases (F4). J. Pure Appl. Algebra 139, 61–88. http://www-salsa.lip6.fr/~jcf/Papers/F99a.pdf.

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

Faugère, J.-C., Gianni, P.M., Lazard, D., Mora, T., 1993. Efficient computation of zero-dimensional Gröbner bases by change of ordering. J. Symb. Comput. 16 (4), 329–344.

Faugère, J.-C., Joux, A., 2003. Algebraic cryptanalysis of hidden field equation (HFE) cryptosystems using Gröbner bases. In: CRYPTO 2003. In: Advances in Cryptology, vol. 2729, pp. 44–60.

Faugère, J.-C., Safey El Din, M., Spaenlehauer, P.-J., 2011. Gröbner bases of bihomogeneous ideals generated by polynomials of bidegree $(1, 1)$: algorithms and complexity. J. Symb. Comput. 46 (4), 106–437.

Faugère, J.-C., Safey El Din, M., Verron, T., 2013. On the complexity of computing Gröbner bases for quasi-homogeneous systems. In: Proceedings of the 38th International Symposium on Symbolic and Algebraic Computation. ISSAC '13. ACM, New York, NY, USA, pp. 189–196.

Faugère, J.-C., Svartz, J., 2012. 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 Symbolic and Algebraic Computation. ISSAC '12. ACM, New York, NY, USA, pp. 170–178.

Faugère, J.-C., Svartz, J., 2013. Gröbner bases of ideals invariant under a commutative group: the non-modular case. In: Proceedings of the 38th International Symposium on Symbolic and Algebraic Computation. ISSAC '13. ACM, New York, NY, USA, pp. 347–354.

Faugère, J.-C., Lachartre, S., 2010. Parallel Gaussian elimination for Gröbner bases computations in finite fields. In: Moreno-Maza, M., Roch, J.L. (Eds.), Proceedings of the 4th International Workshop on Parallel and Symbolic Computation. PASCO '10. ACM, New York, NY, USA, pp. 89–97.

Faugère, J.-C., Rahmany, S., 2009. Solving systems of polynomial equations with symmetries using SAGBI-Gröbner bases. In: Proceedings of the 2009 International Symposium on Symbolic and Algebraic Computation. ISSAC '09. ACM, New York, NY, USA, pp. 151–158.

Galkin, V.V., 2012. Termination of original F5. http://arxiv.org/abs/1203.2402.

Galkin, V.V., 2013. Simple signature based iterative algorithm for calculation of Gröbner bases. Mosc. Univ. Math. Bull. 68 (5), 231–236.

Galkin, V.V., 2014. Termination of the F5 algorithm. Program. Comput. Softw. 40, 47–57 (translation from Programmirovanie 40, No. 2, 2014).

Gao, S., Guan, Y., Volny IV, F., 2010a. A new incremental algorithm for computing Gröbner bases. In: Proceedings of the 2010 International Symposium on Symbolic and Algebraic Computation. ISSAC '10. ACM, pp. 13–19.

Gao, S., Volny IV, F., Wang, M., 2010b. A new algorithm for computing Groebner bases. http://eprint.iacr.org/2010/641.

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

Gao, S., Volny IV, F., Wang, M., 2013. A new algorithm for computing Groebner bases (rev. 2013). http://www.math.clemson.edu/~sgao/papers/gvw_R130704.pdf.

Gao, S., Volny IV, F., Wang, M., 2016. A new framework for computing Gröbner bases. Math. Comput. 85, 449–465.

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

Gash, J.M., 2009. A provably terminating and speed-competitive variant of F5 – F5t. Unpublished preprint.

Gebauer, R., Möller, H.M., 1986. Buchberger's algorithm and staggered linear bases. In: Proceedings of the Fifth ACM Symposium on Symbolic and Algebraic Computation. SYMSAC '86. ACM, New York, NY, USA, pp. 218–221.

Gebauer, R., Möller, H.M., 1988. On an installation of Buchberger's algorithm. J. Symb. Comput. 6 (2–3), 275–286.

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

Gerdt, V.P., Hashemi, A., Alizadeh, M.B., 2013. Involutive bases algorithm incorporating F5 criterion. J. Symb. Comput. 59, 1–20.

Greuel, G.-M., Pfister, G., 2007. A Singular Introduction to Commutative Algebra, 2nd edition. Springer Verlag.

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

Kandri-Rody, A., Weispfenning, V., 1990. Non-commutative Gröbner bases in algebras of solvable type. J. Symb. Comput. 9 (1), 1–26.

Kapur, D., Madlener, K., 1989. A completion procedure for computing a canonical basis for a k-subalgebra. Comput. Math., 1–11.

Kollreider, C., Buchberger, B., 1978. An improved algorithmic construction of Gröbner-bases for polynomial ideals. SIGSAM Bull. 12, 27–36.

Kreuzer, M., Robbiano, L., 2005. Computational Commutative Algebra 2, 1st edition. Springer Verlag.

Kreuzer, M., Robbiano, L., 2009. Computational Commutative Algebra 1, 2nd edition. Springer Verlag.

Lazard, D., 1983. Gröbner bases, Gaussian elimination and resolution of systems of algebraic equations. In: van Hulzen, J.A. (Ed.), European Computer Algebra Conference. EUROCAL'83. In: Springer LNCS, vol. 162, pp. 146–156.

Macaulay, F.S., 1902. On some formulæ in elimination. Proc. Lond. Math. Soc. 33 (1), 3–27.

Macaulay, F.S., 1916. The Algebraic Theory of Modular Systems. Cambridge University Press.

Marinari, M.G., Möller, H.M., Mora, T., 1993. Gröbner bases of ideals defined by functionals with an application to ideals of projective points. Appl. Algebra Eng. Commun. Comput. 4, 103–145.

Möller, H.M., Mora, T., Traverso, C., 1992. Gröbner bases computation using syzygies. In: ISSAC 92: Papers from the International Symposium on Symbolic and Algebraic Computation, pp. 320–328.

Mora, T., 2005. Solving Polynomial Equation Systems II: Macaulay's Paradigm and Gröbner Technology. Encyclopedia of Mathematics and its Applications. Cambridge University Press.

Pan, S., Hu, Y., Wang, B., 2012. The termination of algorithms for computing Gröbner bases. http://arxiv.org/abs/1202.3524.

Pan, S., Hu, Y., Wang, B., 2013. The termination of the F5 algorithm revisited. In: Proceedings of the 2013 International Symposium on Symbolic and Algebraic Computation. ISSAC 2013, pp. 291–298.

Roune, B.H., Stillman, M., 2012a. Practical Gröbner basis computation. In: Proceedings of the 2012 International Symposium on Symbolic and Algebraic Computation. ISSAC 2012.

Roune, B.H., Stillman, M., 2012b. Practical Gröbner basis computation (extended version). http://arxiv.org/abs/1206.6940.

Stegers, T., 2007. Faugère's F5 algorithm revisited. Master's thesis. Technische Univerität Darmstadt.

Sun, Y., 2013. Signature-based Gröbner basis algorithms – extended MMM algorithm for computing Gröbner bases. http://arxiv.org/abs/1308.2371.

Sun, Y., Wang, D.K., 2010. A new proof for the correctness of the F5(F5-like) algorithm. http://arxiv.org/abs/1004.0084.

Sun, Y., Wang, D.K., 2011a. A generalized criterion for signature related Gröbner basis algorithms. In: Proceedings of the 2011 International Symposium on Symbolic and Algebraic Computation. ISSAC 2011, pp. 337–344.

Sun, Y., Wang, D.K., 2011c. The F5 algorithm in Buchberger's style. J. Syst. Sci. Complex. 24 (6), 1218–1231.

Sun, Y., Wang, D.K., 2013a. A new proof for the correctness of the F5 algorithm. Sci. China Math. 56 (4), 745–756.

Sun, Y., Wang, D.K., 2013b. Extending the GVW algorithm to compute Gröbner bases. Sci. China Math.. Submitted for publication.

Sun, Y., Wang, D.K., Huang, Z., Lin, D., 2014. A monomial-oriented GVW for computing Gröbner bases. http://arxiv.org/abs/1410.0105.

Sun, Y., Wang, D.K., Ma, D.X., Zhang, Y., 2012. A signature-based algorithm for computing Gröbner bases in solvable polynomial algebras. In: Proceedings of the 2011 International Symposium on Symbolic and Algebraic Computation. ISSAC 2012, pp. 351–358.

Thiéry, Nicolas M., 2001. Computing minimal generating sets of invariant rings of permutation groups with SAGBI-Gröbner basis. In: DM-CCG, pp. 315–328.

Traverso, C., 1988. Gröbner trace algorithms. In: ISSAC'88.

Volny, F., 2011. New algorithms for computing Gröbner bases. PhD thesis. Clemson University.

Further reading

Albrecht, Martin R., Cid, Carlos, Faugère, Jean-Charles, Perret, Ludovic, 2012. On the relation between the MXL family of algorithms and Gröbner basis algorithms. J. Symb. Comput. 47 (8), 926–941.

Bardet, M., 2004b. On the complexity of a Gröbner basis algorithm. In: INRIA Algorithms Seminar 2002–2004.

Bardet, M., Faugère, J.-C., Salvy, B., 2004. On the complexity of Gröbner basis computation of semi-regular overdetermined algebraic equations. In: International Conference on Polynomial System Solving. ICPSS, pp. 71–75.

Bigatti, A.M., Caboara, M., Robbiano, L., 2011. Computing inhomogeneous Gröbner bases. J. Symb. Comput. 46, 498–510.

Bigatti, A.M., La Scala, R., Robbiano, L., 1999. Computing toric ideals. J. Symb. Comput. 27, 351–365.

Collart, S., Kalkbrener, M., Mall, D., 1997. Converting bases with the Groebner walk. J. Symb. Comput. 24, 265–469.

Faugère, J.-C. Algebraic cryptanalysis of HFE using Gröbner bases. 2003. INRIA Research Report, No. 4738.

Faugère, J.-C., Safey El Din, M., Spaenlehauer, P.-J., 2010. Computing loci of rank defects of linear matrices using Grobner bases and applications to cryptology. In: Proceedings of the 2010 International Symposium on Symbolic and Algebraic Computation. ISSAC '10. ACM, New York, NY, USA, pp. 257–264. Best student paper award.

Hashemi, A., Benyamin, M.-A., Riahi, M., 2012. Invariant G2V algorithm for computing SAGBI-Gröbner bases. Sci. China Math., 1–15.

Sun, Y., Wang, D.K., 2009. A new proof of the F5 algorithm. http://www.mmrc.iss.ac.cn/mmpreprints/.

Sun, Y., Wang, D.K., 2011b. Solving detachability problem for the polynomial ring by signature-based Gröbner basis algorithms. http://arxiv.org/abs/1108.1301.

Wichmann, T., 1997. Der FGLM-Algorithmus: verallgemeinert und implementiert in Singular. Diploma thesis. University of Kaiserslautern.

Zobnin, A.I., 2010. Generalization of the F5 algorithm for calculating Gröbner bases for polynomial ideals. Program. Comput. Softw. 36, 75–82. http://dx.doi.org/10.1134/S0361768810020040.