Linear Algebra to Compute Syzygies and Gröbner Bases

Daniel Cabarcas*
Dept. of Mathematical Sciences
University of Cincinnati
2600 Clifton ave. Cincinnati, OH 45221
cabarcas@gmail.com

Jintai Ding†
Dept. of Mathematical Sciences
University of Cincinnati
2600 Clifton ave. Cincinnati, OH 45221
and South China University of Technology, China
jintai.ding@uc.edu

ABSTRACT

In this paper, we introduce a new method to avoid zero reductions in Gröbner basis computation. We call this method `LASyz`, which stands for **L**ineal **A**lgebra to compute **Syz**ygies. `LASyz` uses exhaustively the information of both principal syzygies and non-trivial syzygies to avoid zero reductions. All computation is done using linear algebra techniques. `LASyz` is easy to understand and implement. The method does not require to compute Gröbner bases of subsequences of generators incrementally and it imposes no restrictions on the reductions allowed. We provide a complete theoretical foundation for the `LASyz` method and we describe an algorithm to compute Gröbner bases for zero dimensional ideals based on this foundation. A qualitative comparison with similar algorithms is provided and the performance of the algorithm is illustrated with experimental data.

Categories and Subject Descriptors

I.1.2 [**Symbolic and Algebraic Manipulation**]: Algorithms—*Algebraic algorithms*; F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*Computations on discrete structures*

General Terms

Algorithms, Theory

Keywords

Gröbner Bases, Syzygy, Linear Algebra

1. INTRODUCTION

Gröbner bases have become the most important tool of applied algebraic geometry. Efficient computation of Gröbner bases has been the subject of abundant research, ever since the original algorithm was proposed by Buchberger in 1965 [3]. Progress has thrust applications, boosting attention, and subsequent progress.

In a polynomial ring R, a Gröbner basis for an ideal I is a particularly useful basis that can be computed from any set of generators for I. A Gröbner basis solves the ideal membership problem by providing standard representatives for the classes of the quotient ring R/I. Most algorithms to compute Gröbner bases start from any given basis for I and enlarge it with other elements from I until certain saturation condition is met.

Advancement in Gröbner basis computation has been driven, among others, by two ideas, reducing polynomials using linear algebra techniques, and avoiding zero reductions —linearly dependent polynomials. The relation between Linear algebra and Gröbner bases was first studied by Lazard [15] and later transformed into practical algorithms like $\mathtt{F}_4$ [11], `XL` [7] or `MGB` [5]. Zero reductions were first studied by Buchberger [4], who proposed criteria to identify s-polynomials that reduce to zero, and later by Möller, Mora and Traverso [18], who proposed computing simultaneously a basis for the module of syzygies —algebraic relations among polynomials. Although impractical, the latest approach laid the foundation for practical implementations like Faugère's $\mathtt{F}_5$ [12].

We propose a new method to avoid zero reductions by keeping a basis for the module of syzygies and using linear algebra techniques. We call this method `LASyz`, which stands for **L**ineal **A**lgebra to compute **Syz**ygies. `LASyz` uses known syzygies to avoid redundant computation in an exhaustive fashion. The use of linear algebra techniques for both polynomial reduction and syzygy reduction makes `LASyz` practical. `LASyz` procedes one degree at a time. Syzygies found at degree d are multiplied by monomials to predict syzygies at degree $d+1$. Principal syzygies are assembled and group together with other known syzygies. All known syzygies are row reduced to avoid redundancies and they are used to discard redundant polynomials. `LASyz` is easy to understand and implement, and its simplicity makes transparent the complexity of both reduction and syzygy bookkeeping. While some of the previous attempts to prevent zero reductions compute Gröbner bases incrementally by including one polynomial at a time, `LASyz` does not require incremental

*Daniel Cabarcas is partially supported by the Distinguished Dissertation Completion Fellowship awarded by the Graduate School of the University of Cincinnati.

†Jintai Ding would like to thank partial support of NSF China grant #60973131 and the Taft Research Center.

ISSAC'11, June 8–11, 2011, San Jose, California, USA.

computation and it imposes no restrictions on the reductions allowed, offering more flexibility.

The paper is organized as follows. In Section 2, we provide a complete theoretical foundation for `LASyz`, including a formal statement of the algorithm and results that prove its correctness and effectiveness. In Section 3, we present a toy example that illustrates `LASyz` proceeding. Then, in Section 4, we describe a Gröbner bases algorithm based on `LASyz`. In Section 5, `LASyz` is compared with previous work from a qualitative point of view, and in Section 6, experimental results are presented and analyzed. In Section 7, we state conclusions and propose future work.

2. THEORETICAL FOUNDATION

Let k be a field and $R = k[x_1, \dots, x_n]$ be the ring of polynomials over k on n variables. For $d \geq 0$ let R_d be the additive subgroup of homogeneous polynomials of degree d, so that $R = \oplus_{d=0}^{\infty} R_d$ is the usual gradation. Let M be the set of all monomials and M_d the set of monomials of degree d. Let P be a sequence of m polynomials. Let $\alpha \in R^m$ be an m-tuple of polynomials. We call *leading entry* of α with respect to P ($\mathrm{LE}_P(\alpha)$ for short) the polynomial in P corresponding to the first non-zero entry of α. For example, if $P = \{p_1, p_2, p_3\}$ and $\alpha = (0, xy - yz, xyz)$, then $\mathrm{LE}_P(\alpha) = p_2$. We often omit the reference to P when it is understood from the context.

Define the R-module homomorphism $v_P : R^m \to R$ by $v_P((\alpha_p)_{p \in P}) = \sum_{p \in P} \alpha_p p$. A *syzygy* of P is any m-tuple $\alpha = (\alpha_1, \dots, \alpha_m)$ in the kernel of v_P, and we denote by $\mathrm{Syz}(F)$ the R-module of all syzygies. For $f, g \in P$, with $f \neq g$, we denote by $\pi_{f,g}$ the syzygy $g\mathcal{E}_f - f\mathcal{E}_g$, where $\mathcal{E}_f$ denotes the *canonical unit vector* with a single non-zero entry 1 in the position corresponding to f. We call $\pi_{f,g}$ a *principal syzygy* of P (also known as trivial syzygy) and denote by $\mathrm{pSyz}(P)$ the R-module generated by all principal syzygies.

For the rest of this section, assume that for all $p \in P$, p is homogeneous of degree d_p. Then, R^m is a graded R-module with degree d elements defined by

$$R_d^m := \{(\alpha_p)_{p \in P} \in R^m \mid \text{ for } p \in P, \alpha_p = 0 \text{ or } \alpha_p \in R_{d-d_p}\} \ .$$

With $\mathrm{Syz}_d(P) := \mathrm{Syz}(P) \cap R_d^m$, $\mathrm{Syz}(P) = \oplus_{d=0}^{\infty} \mathrm{Syz}_d(P)$ is a graded module and with $\mathrm{pSyz}_d(P) := \mathrm{pSyz}(P) \cap R_d^m$, $\mathrm{pSyz}(P) = \oplus_{d=0}^{\infty} \mathrm{pSyz}_d(P)$ is also a graded module.

Syzygies have a relative nature that is exploited by the proposed `LASyz` method. For example, suppose $f_1, f_2, g_1, g_2, h_1, h_2$ are polynomials such that $f_1 g_1 h_1 + f_2 g_2 h_2 = 0$. We can say that $(f_1 g_1, f_2 g_2)$ is a syzygy of (h_1, h_2) or we can say that (f_1, f_2) is a syzygy of $(g_1 h_1, g_2 h_2)$. We are interested in a particular map between modules of syzygies. Consider the family of extension sets defined for $d \geq 0$ by

$$P_d := \{tp \mid t \in \mathrm{M}, p \in P, \deg(tp) = d\} \ ,$$

and

$$P_{(d)} := \bigcup_{j=0}^{d} P_j \ .$$

There is a natural surjective k-module homomorphism

$$\sigma_d : \mathrm{Syz}_d(P_{(d)}) \twoheadrightarrow \mathrm{Syz}_d(P_d) \ ,$$

defined by

$$\sum_{p \in P_{(d)}} \left(\sum_{t \in \mathrm{M}} a_{p,t} t \right) \mathcal{E}_p \mapsto \sum_{p \in P_{(d)}} \sum_{t \in \mathrm{M}} a_{p,t} \mathcal{E}_{tp} \ ,$$

where for $p \in P_{(d)}$ and $t \in \mathrm{M}$, $a_{p,t} \in k$ and $a_{p,t} \neq 0$ implies $\deg(tp) = d$. The homomorphisms σ_d provides a systematic way to transform syzygies between different extension sets.

Notice that degree d syzygies of P_d have scalar entries and they constitute a vector space. We are particularly interested in syzygies with only scalar entries, because linear algebra techniques can be used for computing with them. The set of all syzygies of P with scalar entries will be denoted by $\mathrm{Syz\text{-}S}(P)$. Using this notation, $\mathrm{Syz}_d(P_d) = \mathrm{Syz\text{-}S}(P_d)$.

The basic linear algebra procedure that we use can be described as follows. Let A be an $m \times n$ matrix in k. Consider the linear map associated with A, $L_A : k^m \to k^n$ defined by $L_A(X) = XA$. Suppose we are interested in finding a basis for the row space of A consisting of a subset of its rows. For this purpose, we can compute a triangular basis $\mathcal{B}$ for the kernel of L_A, and remove from A all rows that correspond to first non-zero entries of elements in $\mathcal{B}$. If we are given a-priori a finite subset B of the kernel of L_A, we can use it to reduce the problem, by first row reducing B to echelon form and removing from A all rows that correspond to first non-zero entries of elements in B.

Given $B \subseteq \mathrm{Syz\text{-}S}(P)$, we call C an *echelon form* of B if $0 \notin C$, $\mathrm{span}(B) = \mathrm{span}(C)$ and for $\alpha_1, \alpha_2 \in C$, $\alpha_1 \neq \alpha_2$ implies $\mathrm{LE}(\alpha_1) \neq \mathrm{LE}(\alpha_2)$.

We describe `LASyz` as a method to avoid zero reductions in an `XL` type algorithm [7]. Starting with a finite set of polynomials P, we generate, degree by degree, the extension sets P_d. We discard redundant elements from P_d by using the linear algebra procedure described above. Known kernel elements come from two sources, principal syzygies and redundancies found at previous degrees. We rely on the relativity of syzygies to obtain as many syzygies as possible from previous degrees.

Given a basis for the degree $d-1$ syzygies of P_{d-1}, Algorithm 1 computes a basis for the degree d syzygies of P_d.

```
Algorithm 1 LASyz($P, d, \mathcal{B}_{d-1}$)
Require: $P$ is a finite subset of homogeneous polynomials.
Require: $\mathcal{B}_{d-1}$ is a basis for $\mathrm{Syz}_{d-1}(P_{d-1})$
$A := \{\sigma_d(x \cdot \alpha) \mid x \in \{x_1, \dots, x_n\}, \alpha \in \mathcal{B}_{d-1}\}$
$B := \{\sigma_d(\pi_{f,g}) \mid f, g \in P, \deg(fg) = d\}$
$C :=$ an echelon form of $A \cup B$
$G := P_d \setminus \mathrm{LE}(C)$
$D :=$ a basis for $\mathrm{Syz\text{-}S}(G)$
$\mathcal{B}_d := C \cup D$
return $\mathcal{B}_d$
```

Note that for $f, g \in P$ and $s, t \in \mathrm{M}$ such that $\deg(stfg) = d$, $\sigma_d(\pi_{(tf,sg)}) = \sigma_d(st\pi_{(f,g)})$, hence, we only need to consider principal syzygies among elements of P and not among elements in extension sets.

The following lemma explains the use of the leading entries of C to discard elements from P_d.

Lemma 1. *If C is a subset of $\mathrm{Syz\text{-}S}(P)$ in echelon form, then the set $P \setminus \mathrm{LE}(C)$ spans the same space as P does. Moreover, if C spans $\mathrm{Syz\text{-}S}(P)$ then $P \setminus \mathrm{LE}(C)$ is a basis for $\mathrm{span}(P)$.*

Proof. Suppose that $P = \{p_1, \ldots, p_m\}$ and $C = \{\alpha_1, \ldots, \alpha_s\}$, $s \leq m$. By reordering P if necessary and multiplying by appropriate scalars, we can assume without lost of generality that for $i = 1, \ldots, s$, $\mathrm{LE}(\alpha_i) = p_i$ and that the i-th entry of α_i is 1. Then, for $i = 1, \ldots, s$, p_i belongs to $\mathrm{span}\{p_{i+1}, \ldots, p_m\}$. By induction it follows that for $i = 1, \ldots, s$, p_i belongs to $\mathrm{span}\{p_{s+1}, \ldots, p_m\}$. It follows that $P \setminus \mathrm{LE}(C) = \{p_{s+1}, \ldots, p_m\}$ spans the same space as P does.

Now suppose that C spans Syz-S(P). Since for $\alpha \neq \alpha' \in C$, $\mathrm{LE}(\alpha) \neq \mathrm{LE}(\alpha')$, C must be a basis. Further suppose that there exist $b_{s+1}, \ldots, b_m \in k$ such that $\sum_{i=s+1}^m b_i p_i = 0$. Then $\sum_{i=s+1}^m b_i e_i \in$ Syz-S(P) (e_i represents the i-canonical unit vector of R^m). Since C spans Syz-S(P) then there exist $c_1, \ldots, c_s \in k$ such that $\sum_{i=s+1}^m b_i e_i = \sum_{j=1}^s c_j \alpha_j$. Looking at the first entry of the equality we conclude that if $s > 0$

$$0 = \left(\sum_{i=s+1}^{m} b_i e_i\right)_1 = \left(\sum_{j=1}^{s} c_j \alpha_j\right)_1 = c_1 \; .$$

By induction on the entries, we can conclude that for $j = 1, \ldots, s$, $c_j = 0$ hence $\sum_{i=s+1}^m b_i e_i = \sum_{j=1}^s c_j \alpha_j = 0$ and therefore $b_{s+1} = \cdots = b_m = 0$. This shows that $P \setminus \mathrm{LE}(C)$ is a basis for $\mathrm{span}(P)$. □

We are now ready to prove the main result of this section, the correctness of `LASyz`.

Theorem 1. *Algorithm 1 computes a basis for the degree d syzygies of P_d.*

Proof. By definition of σ_d, $A \cup B$ is a subset of Syz-S(P_d). By definition of echelon form, $\mathrm{span}(C) = \mathrm{span}(A \cup B)$ and for $\alpha \neq \alpha' \in C$, $\mathrm{LE}(\alpha) \neq \mathrm{LE}(\alpha')$. Since Syz-S$(P_d)$ is a vector space over k, C is a subset of Syz-S(P_d). Then, by Lemma 1, $G = P_d \setminus \mathrm{LE}(C)$ spans the same space as P_d.

Let $\alpha \in \mathrm{Syz}_d(P_d)$. Because C is in echelon form, there exist $\beta \in \mathrm{span}(C)$ such that, for all $p \in \mathrm{LE}(C)$, the entry in $\alpha - \beta$ corresponding to p is zero. Then, $\alpha - \beta \in$ Syz-S$(G) = \mathrm{span}(D)$, and therefore, $\alpha \in \mathrm{span}(\mathcal{B}_d)$, proving that $\mathcal{B}_d$ spans $\mathrm{Syz}_d(P_d)$. C and D are bases so in order to show that $\mathcal{B}_d$ is a basis it suffices to show that for any $0 \neq \alpha \in C$, and $0 \neq \beta \in D$, $a, b \in k$, $a\alpha + b\beta = 0$ implies $a = 0 = b$. Indeed, because C is in echelon form, for all $p \in \mathrm{LE}(C)$, the entry in β corresponding to p is zero and the entry in α corresponding to p is non-zero, hence $a = 0$ and therefore $b = 0$. □

We conclude this section with two results that partially prove the effectiveness of `LASyz` in keeping track of syzygies.

In Algorithm 1, we look only at degree d syzygies of P_d, but it is always possible to track them back to the original set of generators P. For that purpose we define the surjective k-module homomorphism

$$\rho_d : \mathrm{Syz}_d(P_{(d)}) \twoheadrightarrow \mathrm{Syz}_d(P)$$

by

$$\sum_{\substack{s \in M, p \in P \\ sp \in P_{(d)}}} \alpha_{sp} \mathcal{E}_{sp} \mapsto \sum_{\substack{s \in M, p \in P \\ sp \in P_{(d)}}} \alpha_{sp} s \mathcal{E}_p .$$

Next theorem demonstrates that principal syzygies are used effectively by Algorithm 1, in the sense that, if P only possesses trivial syzygies, then all degree d syzygies of P_d are caught before Line 5 is executed. It is stated and proved for regular sequences but it can be adapted for semi-regular sequences with degree bounded by the degree of regularity.

Theorem 2. *If P is a regular sequence of polynomials then* Syz-S$(G) = 0$, *in Algorithm 1.*

Proof. It suffices to show that every syzygy of P_d with scalar entries belongs to the span of C. In such case, by Lemma 1, G is a basis for $\mathrm{span}(P_d)$ hence Syz-S$(G) = 0$. Since P is a regular sequence and the polynomials are homogeneous, $\mathrm{pSyz}(P) = \mathrm{Syz}(P)$.

Let $\alpha \in$ Syz-S(P_d). Since, $\mathrm{pSyz}(P) = \mathrm{Syz}(P)$, there exist $a_{t,f,g} \in k$ such that

$$\rho_d(\alpha) = \sum_{f \neq g \in P} \sum_{t \in M} a_{t,f,g} t \pi_{f,g} \; ,$$

where $a_{t,f,g} \neq 0$ implies $\deg(tfg) = d$. Then, applying σ_d to both sides of the equality we obtain

$$\alpha = \sigma_d(\rho_d(\alpha)) = \sum_{f \neq g \in P} \sum_{t \in M} a_{t,f,g} \sigma_d(t \pi_{f,g}) \; .$$

With A and B as defined in Algorithm 1, we can split the sum above into two parts, one coming from A and the other from B.

$$\alpha = \sum_{f \neq g \in P} \sum_{1 \neq t \in M} a_{t,f,g} \sigma_d(t \pi_{f,g}) + \sum_{f \neq g \in P} a_{1,f,g} \sigma_d(\pi_{f,g})$$

Note that for $a_{t,f,g} \neq 0$ with $t \neq 1$ there exist $x \in \{x_1, \ldots, x_n\}$ such that $\frac{t}{x} \in M$ thus $\sigma_{d-1}(\frac{t}{x}\pi_{f,g}) \in$ Syz-S(P_{d-1})= $\mathrm{span}(\mathcal{B}_{d-1})$ hence $\sigma_d(t\pi_{f,g}) = \sigma_d(x \sigma_{d-1}(\frac{t}{x}\pi_{f,g})) \in \mathrm{span}(A)$. Also, for $a_{1,f,g} \neq 0$, $\sigma_d(\pi_{f,g}) \in B$. Therefore $\alpha \in \mathrm{span}(A \cup B) = \mathrm{span}(C)$. □

The following theorem shows that syzygies of P_d with scalar entries account for all syzygies, which justifies focusing only on those.

Theorem 3. *If $\mathcal{B}_d$ is a basis for* Syz-S(P_d) *then $\rho_d(\mathcal{B}_d)$ is a basis for* $\mathrm{Syz}_d(P)$.

Proof. Let $\alpha = \sum_{p \in P} \alpha_p \mathcal{E}_p \in \mathrm{Syz}_d(P)$. Since α is a degree d syzygy, $\alpha_p = 0$ whenever $\deg(p) > d$. Hence α can be written as a syzygy of $P_{(d)}$, $\alpha' = \sum_{p \in P_{(d)}} \alpha'_p \mathcal{E}_p$ with $\alpha'_p = \alpha_p$ whenever $p \in P$ and $\alpha'_p = 0$ otherwise.

Consider $\sigma_d(\alpha') \in$ Syz-S(P_d). Because $\mathcal{B}_d$ is a basis for Syz-S(P_d), there exist $A_\beta \in k$ such that

$$\sigma_d(\alpha') = \sum_{\beta \in \mathcal{B}_d} A_\beta \beta \; .$$

Applying the homomorphism ρ_d on both sides of the equality we obtain

$$\alpha = \rho_d(\sigma_d(\alpha')) = \rho_d\left(\sum_{\beta \in \mathcal{B}_d} A_\beta \beta\right) = \sum_{\beta \in \mathcal{B}_d} A_\beta \rho_d(\beta) \; .$$

This shows that $\rho_d(\mathcal{B}_d)$ generates $\mathrm{Syz}_d(P)$.

Now suppose $\sum_{\beta \in \mathcal{B}_d} A_\beta \rho_d(\beta) = 0$ for some $A_\beta \in k$. Then

$$0 = \sigma\left(\sum_{\beta \in \mathcal{B}_d} A_\beta \rho_d(\beta)\right) = \sum_{\beta \in \mathcal{B}_d} A_\beta \sigma(\rho_d(\beta)) = \sum_{\beta \in \mathcal{B}_d} A_\beta \beta \; ,$$

and because $\mathcal{B}_d$ is a basis for Syz-S(P_d), it follows that $A_\beta = 0$ for all $\beta \in \mathcal{B}_d$ and therefore $\rho_d(\mathcal{B}_d)$ is a basis. □

3. TOY EXAMPLE

We illustrate how LASyz avoids reductions to zero by means of a simple example. Let

$$\begin{aligned} P = \{p_1 &= 22x^2 + 4xz + 20y^2 + 5yz + 14z^2 \\ p_2 &= 15x^2 + 17xy + 7xz + 12y^2 + 3yz + 10z^2 \\ p_3 &= x^2 + 4xy + 8xz + 16y^2 + 18yz + 18z^2 \\ p_4 &= x^2 + 7xy + 22xz + 11y^2 + 2yz + 10z^2\}, \end{aligned}$$

a set of degree 2 polynomials in the variables x, y, z with coefficients in GF(23). These polynomials are linearly independent. However, the polynomials in P_3 are not —

$$P_3 = \{xp_1, xp_2, xp_3, xp_4, yp_1, yp_2, yp_3, yp_4, zp_1, zp_2, zp_3, zp_4\} .$$

Using linear algebra over the matrix that represents P_3, we are able to find two linear relations among the polynomials in P_3,

$$\begin{aligned} \alpha &= \begin{pmatrix} 1 & 0 & 6 & 18 & 20 & 1 & 2 & 14 & 19 & 20 & 2 & 5 \end{pmatrix}, \\ \beta &= \begin{pmatrix} 0 & 1 & 0 & 8 & 6 & 15 & 12 & 18 & 11 & 11 & 3 & 5 \end{pmatrix} \in \mathrm{Syz}(P_3). \end{aligned}$$

From these, and by multiplying by x, y, z, we can obtain syzygies among the polynomials in P_4. For example, the syzygy α corresponds to the equation

$$xp_1 + 6xp_3 + 18xp_4 + 20yp_1 + yp_2 + 2yp_3 + \cdots + 5zp_4 = 0 .$$

By multiplying it by x we obtain

$$x^2p_1 + 6x^2p_3 + 18x^2p_4 + 20xyp_1 + xyp_2 + 2xyp_3 + \cdots + 5xzp_4 = 0 ,$$

which corresponds to the syzygy of P_4

$$\mathcal{E}_{x^2p_1} + 6\mathcal{E}_{x^2p_3} + 18\mathcal{E}_{x^2p_4} + 20\mathcal{E}_{xyp_1} + \mathcal{E}_{xyp_2} + 2\mathcal{E}_{xyp_3} + \cdots + 5\mathcal{E}_{xzp_4} ,$$

where $\mathcal{E}_p$ denotes the *canonical unit vector* with a single non-zero entry 1 in the position corresponding to p. In this fashion, we can obtain six elements of $\mathrm{Syz}(P_4)$ corresponding to $x\alpha, x\beta, y\alpha, y\beta, z\alpha, z\beta$.

We also know a-priori the principal syzygies of P. For example, $p_1\mathcal{E}_{p_2} - p_2\mathcal{E}_{p_1}$ which corresponds to the equation

$$(22x^2 + \cdots + 14z^2)p_2 - (15x^2 + \cdots + 10z^2)p_1 = 0 ,$$

or equivalently

$$22x^2p_2 + \cdots + 14z^2p_2 - 15x^2p_1 - \cdots - 10z^2p_1 = 0 ,$$

which corresponds to the syzygy of P_4

$$22\mathcal{E}_{x^2p_2} + \cdots + 14\mathcal{E}_{z^2p_2} - 15\mathcal{E}_{x^2p_1} - \cdots - 10\mathcal{E}_{z^2p_1} .$$

Overall, we have obtained twelve elements of $\mathrm{Syz}(P_4)$, six from multiplying elements of $\mathrm{Syz}(P_3)$ by variables, and six principal syzygies. Next, we put these twelve vectors on a matrix and row reduce it to obtain a row echelon form. Each pivot column corresponds to a redundant element of P_4 revealing redundant the polynomials

$$x^2p_1, x^2p_2, x^2p_3, xyp_1, xyp_2, xyp_3, xzp_1, xzp_2, xzp_3, xp_1, xp_2 .$$

We can safely remove them to form a smaller set of generators for the span of P_4. In this fashion we are using exhaustively the information of both principal syzygies of P and non-trivial syzygies found at degree three, to avoid redundancies at degree four.

4. NEW GRÖBNER BASES ALGORITHM

Next, we introduce a new algorithm to compute Gröbner bases of zero-dimensional ideals based on the mutantXL algorithm [8]. The description is for the ring of boolean functions

$$\mathfrak{B} := R/\langle x_1^2 - x_1, \ldots, x_n^2 - x_n\rangle ,$$

with $R = k[x_1, \ldots, x_n]$ and k the Galois field of order two. We decided to describe the algorithm for this ring due to its importance in cryptography and coding theory.

mutantXL can be summarized as follows. Assume a monomial order is fixed and let P be a finite set of elements in $\mathfrak{B}$ (usually not homogeneous). mutantXL constructs, one degree at a time, an extension set P_d, linearizes it, computes an echelon form, and searches for mutant polynomials, i.e. polynomials of a lower degree than d. If mutants are found, it extends mutants before constructing the next extension set P_{d+1}. The difference between the new algorithm and mutantXL is the use of LASyz to discard redundant polynomials.

We shall make some precisions in the notation for the particular ring $\mathfrak{B}$ and for working with non-homogeneous polynomials:

1. In the quotient ring $\mathfrak{B}$, the degree of a class is defined to be the minimum among the degrees of all representatives. In the case of a syzygy, the *degree* of $\alpha = (\alpha_p)_{p\in P} \in \mathrm{Syz}(P)$ is the maximum among the degrees of the $\alpha_p p$. We denote by $\mathrm{Syz}_{(d)}(P)$ the k-module of syzygies of P of degree up to d. Note that this notion of degree does not produce a gradation of $\mathrm{Syz}(P)$ but only a filtration.

2. Just as in Section 2, we are interested in syzygies with scalar entries, because they can be computed using linear algebra. We denote by $\text{Syz-S}(P)$ the k-module of all syzygies of P with scalar entries. There is a natural surjective k-module homomorphism
$$\sigma_d : \mathrm{Syz}_{(d)}(P_{(d)}) \twoheadrightarrow \text{Syz-S}(P_{(d)})$$
defined by
$$\sum_{p\in P_{(d)}} \left(\sum_{t\in \mathrm{M}} a_{p,t}t\right) \mathcal{E}_p \mapsto \sum_{p\in P_{(d)}} \sum_{t\in \mathrm{M}} a_{p,t}\mathcal{E}_{tp} ,$$
where for $p \in P_{(d)}$ and $t \in \mathrm{M}$, $a_{p,t} \in k$, and $a_{p,t} \neq 0$ implies $\deg(tp) \leq d$.

3. In the ring $\mathfrak{B}$, any $p \in \mathfrak{B}$ satisfies $p^2 = p$. We include this relations as principal syzygies by extending the notation $\pi_{f,g}$. We denote by $\pi_{p,p}$ the syzygy $(p-1)\mathcal{E}_p$.

4. For any $p \in \mathfrak{B}$ of degree d, we denote by p^h the *leading form* of p (the homogeneous part of p of degree d), and by p^{-h} the rest of the polynomial $p - p^h$.

In Algorithm 2, we spelled out the details of the new Gröbner bases algorithm which we call LASyzGB.

We now explain the algorithm. The **while** loop of the algorithm produces, one degree at a time, a set G that spans $P_{(d)}$. As termination condition, we can use the conditions in Proposition 3 from [16] which we state below for completeness. This proposition, together with Theorem 1 guarantee that upon termination LASyzGB returns a Gröbner basis.

Algorithm 2 `LASyzGB`(P)

```
Require: $P$ is a finite subset of polynomials in $\mathfrak{B}$.
 1: $d := 1$
 2: $\mathcal{B}_0 := \emptyset$
 3: while termination condition do
 4:   $A := \{\sigma_d(x \cdot \alpha) \mid x \in \{x_1, \ldots, x_n\}, \alpha \in \mathcal{B}_{d-1}\}$
 5:   $B := A \cup \{\sigma_d(\pi_{f,g}) \mid f, g \in P, \deg(fg) = d\}$
 6:   repeat
 7:     $C :=$ an echelon form of $B \cup \mathcal{B}_1 \cup \cdots \cup \mathcal{B}_{d-1}$
 8:     $G := P_{(d)} \setminus \mathrm{LE}(C)$
 9:     $D :=$ an echelon form of Syz-S$(G)$
10:     $G := G \setminus \mathrm{LE}(D)$
11:     $B := B \cup D$
12:     for $i = 1, \ldots, d$ do
13:       $\overline{G} := \{g \in G \mid \deg(g) = i\}$
14:       $E :=$ an echelon form of Syz-S$(\overline{G}^h)$
15:       if $E \neq \emptyset$ then
16:         mutants $:= v_{\overline{G}}(E)$
17:         $P := P \cup$ mutants
18:         $B := B \cup E$
19:         break
20:   until mutants $= \emptyset$
21:   $\mathcal{B}_d := B$
22:   $d := d + 1$
23: return $G$
```

Proposition 1 ([16]). *Let G be a finite subset of $\mathfrak{B}$ with highest degree D and suppose that the following holds:*

1. $\mathrm{LM}(G) \supseteq \mathrm{LM}(\mathrm{M}_D)$*, and*
2. *if $H := G \cup \{t \cdot g \mid g \in G,\ t \in \mathrm{M}$ and $\deg(t) + \deg(g) \leq D + 1\}$, there exists $\widetilde{H}$, a row echelon form of H, such that $\{h \in \widetilde{H} \mid \deg(h) \leq D\} = G$,*

then G is a Gröbner basis for the ideal I generated by G.

The set A in Algorithm 2, Line 4, groups syzygies obtained in previous degrees extended and interpreted as syzygies of $P_{(d)}$ with scalar entries. The set B in Line 5, appends to A all principal syzygies of degree d interpreted as syzygies of $P_{(d)}$ with scalar entries.

The **repeat-until** loop, constructs the set G and verifies whether there are any mutant polynomials, in which case it modifies the set P of original polynomials and reconstructs G accordingly. Lines 7 to 11 compute syzygies using `LASyz` as described in Algorithm 1.

The **for** loop checks for mutants at each degree i. Note that if $\alpha \in$ Syz-S$(\overline{G}^h)$ then $v_{\overline{G}}(\alpha)$ is a mutant polynomial of G or zero. In Line 17, the mutant polynomials found at degree i are appended to the original set of polynomials P, and in Line 18 the corresponding relations are appended to the set of syzygies B. If mutants are found the loop is broken.

5. QUALITATIVE COMPARISON

This is by no means the first attempt to avoid zero reductions in Gröbner basis computation. Remarkable work precedes us by Buchberger [4], Möller, Mora and Traverso [18], Faugère [12], among others. It is important to evaluate from a qualitative point of view where our new approach lies in this spectrum of algorithms.

Buchberger's criteria to discard s-polynomials are effective for avoiding zero reductions and can be efficiently implemented [14]. It has been shown that many more zero reductions can be avoided by a syzygy approach [18]. Möller, et. al. claim that their approach "covers both of Buchberger criteria" and "avoids more superfluous reductions". The algorithm proposed here is similar to Möller's in that it maintains a subset of the module of syzygies and uses it to avoid reductions to zero. However, Möller's algorithm is not practical, as the authors claim, "The first results show an ambiguous behavior: many useless pairs are discovered, but this involves a lot of extra computation, so the execution time is increased." `LASyz` overcomes this problem with a different way to maintain syzygies. Syzygies are kept in reference to the original set of generators and computation is purely based on linear algebra. In this way, all the burden of syzygy bookkeeping is carried by a sparse linear algebra package. As Faugère demonstrated with his `F`$_4$ algorithm, linear algebra can make a huge impact in the efficiency.

Faugère's `F`$_5$ relies on two criteria to avoid zero reductions, the rewritten criterion and Faugère's criterion [10]. The former uses non-trivial syzygies previously detected, but it is uncertain how effective it is. Faugère's criterion relies on principal syzygies and it has been proved to be effective, but the cost associated still awaits to be fully understood. The hidden cost of Faugère's criterion lies is its incremental nature, which restricts the reductions allowed. We explain this last point with the aid of `MatrixF`$_5$, a close relative of `F`$_5$ that was first mentioned in [1]. Let $P = \{p_1, \ldots, p_m\}$ be homogeneous polynomials of degrees $d_1, \ldots, d_m$ respectively. Define the sets

$$P_{d,i} := \{tp \mid t \in \mathrm{M}, p \in \{p_1, \ldots, p_i\}, \deg(tp) = d\} \, .$$

Faugère's criterion states that if G is an echelon form of $P_{d-d_i,i-1}$ and t is a leading monomial of G, then, tp_i is redundant in $P_{d,i}$. The problem with the implementation of this criterion is its incremental nature. It is necessary to obtain the leading monomials of an echelon form of $P_{d-d_i,i-1}$ in order to avoid syzygies in $P_{d,i}$, placing a burden on the linear algebra procedure. During the reduction of P_d a strict order must be enforced and only row operations "downward" are allowed. Such restriction may inhibit the choice of the most suitable sparse matrix algorithm to compute an echelon form.

In [13], Gao et. al. propose an incremental Gröbner bases algorithm called G2V, which offers another way to avoid zero reductions using information from the module of syzygies. Given $G = \{g_1, \ldots, g_m\}$ a Gröbner basis for an ideal I and any $g \in R$, G2V computes Gröbner bases for $\langle I, g\rangle$ and $(I : g) = \{u \in R \mid ug \in I\}$. Notice that $u \in (I : g)$ implies that there exist $h_1, \ldots, h_m \in R$ such that $ug = \sum_{i=1}^{m} h_i g_i$, so u can be regarded as some kind of signature for the syzygy $(-h_1, \ldots, -h_m, u)$ of $(g_1, \ldots, g_m, g)$.

The incremental nature of G2V also acts in detriment of its efficiency. Besides a restriction on the reductions allowed, G2V also imposes a strict order in the selection of the pairs. The paper announces a non-incremental version which shall be very interesting, given the simplicity and efficiency achieved already by G2V. It is also important to note that in the execution of G2V, the elements of $(I : g)$ are not reduced among each other, allowing redundancies and missing opportunities for discarding zero reductions.

In this context, our proposed non-incremental algorithm offers an alternative that is simple, easy to implement an analyze, that offers a comprehensive treatment of both trivial and non-trivial syzygies and that relies entirely on linear algebra procedures with no restrictions. The new algorithm also yields a basis for the module of syzygies.

6. EXPERIMENTAL RESULTS

We have tested LASyz' performance in avoiding zero reductions and we have compared it with other methods. In order to illustrate its capabilities and limitations, we present here details of some experiments. We have implemented the proposed algorithm for computing Gröbner bases in C++ and in Magma [2]. All experiments were run in a personal computer equipped with an Intel(R) core(TM) 2 Duo CPU E6550 @2.33GHz processor, with 2 GB of Ram, and running Windows XP. The first experiment illustrates the behavior of the algorithm in presence of non-trivial syzygies and the second one aims at evaluating the cost of keeping a basis of syzygies and detecting zero reductions. Two more experiments compare LASyz with Faugère's $\texttt{F}_5$ algorithm.

6.1 Non-trivial Syzygies Experiment

For the purpose of illustrating the behavior of the algorithm in presence of non-trivial syzygies, we chose polynomials coming from an HFE cryptosystem [19]. The cryptosystem is a random HFE with parameters: size of field $q = 2$, extension degree $n = 14$, degree bound $D = 16$. The polynomials are the homogeneous degree two part of the public key and computations were made modulo $\langle x_1^2, \ldots, x_n^2 \rangle$. Three equations were removed, making this an HFE minus system.

The results are presented in Table 6.1. The table shows the number of polynomials produced at each degree by the new algorithm and by Magma's implementation of Faugère's $\texttt{F}_4$ [2] for comparison.

LASyz				
Degree	2	3	4	5
Number of polynomials in P_d	11	154	1001	4004
Number of polynomials used	11	154	935	2436
Dimension of kernel	0	0	46	434
Number of syzygies			66	1568
Faugère's $\texttt{F}_4$				
Number of polynomials used	11	336	1958	4756

Table 1: Comparison in number of polynomials produced at each degree on an HFE minus system.

The use of less polynomials translates into less memory. The gain comes from two sources. At degree four, 66 trivial syzygies allow us to ignore 66 polynomials of the extension set P_4. Also, at this same degree, 46 non-trivial syzygies are spotted in the kernel of the set P_4. Then, we put together the 66 trivial plus the 46 non-trivial for a total of 112 syzygies, which are extended to degree five by multiplying by each variable to obtain 112(14) = 1568 syzygies of degree five. The 1568 syzygies allow us to ignore 1568 polynomials of the extension set P_5.

6.2 Performance Experiment

In order to illustrate the performance of LASyzGB, we used a random system of 22 polynomial equations in 14 variables with coefficients in GF(2^8). We computed a Gröbner basis for the system using Faugère's $\texttt{F}_4$ algorithm and the new proposed algorithm implemented in C++. Because the algorithm to compute row echelon forms is critical for performance, and in order to obtain comparable results, we ran both the proposed algorithm and our own home brew (HB) implementation of $\texttt{F}_4$ using the same row echelon form algorithm as described in [6]. We also run Magma's $\texttt{F}_4$ for reference.

The most time consuming task in both cases is the row reduction of a large matrix that represents degree five polynomials. Values for those matrices are presented in Table 6.2: number of rows and columns, number of non-zero entries before and after reduction, time and memory used.

	Magma $\texttt{F}_4$	HB $\texttt{F}_4$	LASyzGB	Syzygy
rows	12413	14011	9086	3234
cols	8184	9782	11508	14938
nnz before	6074177	13183492	1087961	774396
nnz after	14258890	16051461	12961466	12448862
time(sec)	57.109	606.5	424.5	96.86
mem(MB)	60.2	456.1	339.2	241.7

Table 2: Comparison in matrix size time and memory between new algorithm and $\texttt{F}_4$.

Note that LASyzGB produces less rows but more columns than $\texttt{F}_4$. The number of non-zero entries before the reduction takes place is significantly smaller, yet after the reduction it is comparable. Both the time and memory effort are lower for the new algorithm.

The right-most column of Table 6.2 shows the same measures for the matrix that represents the degree five syzygies. Note that the time and memory efforts for this were relatively small compared to the reduction of the matrix that represents degree five polynomials.

6.3 Comparison with Faugère's $\texttt{F}_5$

We chose a smaller system in order to compare LASyz with available implementations of Faugère's $\texttt{F}_5$ [12]. We used a random degree 2 homogeneous overdetermined system of 10 polynomials in 8 variables with coefficients in GF(2^8). We computed a Gröbner basis for the system using LASyz and two different implementations of Faugère's $\texttt{F}_5$, Stegers' [20] written for Magma and Eder and Perry's [9] written for Singular.

In summary, the execution of Stegers' $\texttt{F}_5$ reports that 506 polynomials were treated, 65314 pairs were avoided, the maximum degree of a critical pair was 16, the maximum degree of a polynomial was 12 and the total number of zero reductions was 138. The total running time was 44.80 seconds. Execution of Eder and Perry' $\texttt{F}_5$ run for more than 96 hours without terminating.

LASyz generated 1245 polynomials, the maximum degree of a polynomial was 5, the total number of zero reductions was 48 and 405 syzygies were used in avoiding the same number of zero reductions. Using the Magma implementation of LASyz, the total running time was 3.31. The C++ version was faster, terminating in 1.51 seconds.

6.4 A Standard Benchmark

Next we present results for Katsura 6 over GF(7). It is important to note that this system does not have a zero dimensional solution thus the algorithm proposed in Section 4 does not terminate. It was necessary therefore to halt the algorithm artificially at degree 6, where we knew a Gröbner basis was obtained. Also, in this case we used the Magma implementation of LASyz, instead of the C++ implementation used in previous cases.

In summary, the execution of Stegers' F_5 reports that 74 polynomials were treated, 2519 pairs were avoided, the maximum degree of a critical pair was 7, the maximum degree of a polynomial was 7 and the total number of zero reductions was zero. The total running time was 0.469 seconds.

LASyz generated 1572 polynomials, the maximum degree of a polynomial was 6 and the total number of zero reductions was zero. 820 syzygies were used in avoiding the same number of zero reductions. Using the Magma implementation of LASyz, the total running time was 7.047 seconds.

6.5 Analysis of Experimental Results

The Experiments show that LASyz is effective in avoiding zero reductions. The small number of polynomials used in the HFE minus example clearly shows that a significant amount of redundancy is being avoided. Also, in the random example, we can observe a small number of rows compared with F_4.

For overdetermined systems the performance of the proposed algorithm is comparable to Faugère's F_4 and much better than Faugère's F_5. In such cases the incremental nature of F_5 militates against its performance. This is evidenced by the high degree of the operation. A matrix version of F_5 may perform better with overdetermined systems but it is still unknown to the authors the impact in performance of the restrictions in the linear algebra procedures.

LASyz exhibits a poor performance in the katsura benchmark compared to F_5. This is due to lack of a selection strategy, that would filter the polynomials used. Examples of such strategies are s-polynomials [3], symbolic preprocessing [11] and partial enlargement [17]. Other examples show a similar behavior. In order to get a more throughout comparison, it is desirable to use a more efficient implementation of the linear algebra procedures. We are working on optimizing our implementation.

7. CONCLUSIONS AND FUTURE WORK

We have introduced a new method to avoid reductions to zero in Gröbner basis computation called LASyz. We have proved that LASyz works correctly and it effectively uses trivial and non-trivial syzygies to avoid zero reductions. LASyz provides the first mechanism to avoid zero reductions in XL type algorithms that does not require an incremental computation. A comparison with previous alternatives highlights the benefits of the new approach.

LASyz can be used to study syzygies, which is important in algebraic geometry to study geometric properties of algebraic varieties.

A Gröbner basis algorithm based on LASyz was described and tested. The Experiments ratify that using LASyz for avoiding zero reductions is effective and that the use of sparse linear algebra makes it efficient. For overdetermined systems, the performance of the proposed algorithm is comparable to Faugère's F_4 and much better than Faugère's F_5.

LASyz does not replace the need for a selection strategy. We envision that LASyz can be combined with s-polynomial strategy or with any other heuristic method for partial enlargement.

We are making progress in establishing complexity bounds for LASyz. A possible deficiency of LASyz stems from the lack of sparsity of non-trivial syzygies. LASyz can be adapted to overcome this issue by restricting the use of non-trivial syzygies to predict further syzygies. The resulting trade-off between accuracy and cost can be studied using the framework of this new method. The complexity can be studied thanks to the simplicity of the method. This direction shall be pursued in another paper.

Another possibility offered by the LASyz method is to track other level of syzygies. We can use the same strategy described in this paper, to manage syzygies of syzygies, and so on.

8. REFERENCES

[1] M. Bardet. *Étude des Systèmes Algébriques Surdéterminés. Applications aux Codes Correcteurs et à la Cryptographie.* PhD thesis, Université Paris VI, 2004.

[2] W. Bosma, J. Cannon, and C. Playoust. The Magma Algebra System. I. The User Language. *J. Symbolic Computation*, 24(3-4):235–265, 1997.

[3] B. Buchberger. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal (An Algorithm for Finding the Basis Elements in the Residue Class Ring Modulo a Zero Dimensional Polynomial Ideal).* PhD thesis, Mathematical Institute, University of Innsbruck, Austria, 1965. (English translation in Journal of Symbolic Computation, 2004).

[4] B. Buchberger. A Criterion for Detecting Unnecessary Reductions in the Construction of Gröbner Bases. In *Proceedings of the EUROSAM 79 Symposium on Symbolic and Algebraic Manipulation*, volume 72 of *Lecture Notes in Computer Science*, pages 3–21. Springer, Berlin - Heidelberg - New York, 1979.

[5] J. Buchmann, D. Cabarcas, J. Ding, and M. S. E. Mohamed. Flexible Partial Enlargement to Accelerate Gröbner Basis Computation over $\mathbb{F}_2$. In *Progress in Cryptology – AFRICACRYPT 2010*, Lecture Notes in Computer Science. Springer-Verlag, Berlin, 2010.

[6] D. Cabarcas. An Implementation of Faugère's F_4 Algorithm for Computing Gröbner Bases. Master's thesis, University of Cincinnati, 2010.

[7] N. Courtois, A. Klimov, J. Patarin, and A. Shamir. Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations. In *Advances in Cryptology – EUROCRYPT 2000*, Lecture Notes in Computer Science, pages 392–407. Springer-Verlag, Berlin / Heidelberg, 2000.

[8] J. Ding, J. Buchmann, M. S. E. Mohamed, W. S. A. Moahmed, and R.-P. Weinmann. MutantXL. In *Proceedings of the First International Conference on Symbolic Computation and Cryptography (SCC08)*, pages 16 – 22. LMIB, 2008.

[9] C. Eder and J. Perry. Singular Implementation of Faugère's F5 Algorithm. Available online at http://www.math.usm.edu/perry/Research/f5 library.lib. v 1.1 2009/01/26.

[10] C. Eder and J. Perry. F5C: A Variant of Faugère's F5 algorithm with Reduced Gröbner Bases. *Journal of Symbolic Computation*, 45(12):1442 – 1458, 2010.

[11] J.-C. Faugère. A New Efficient Algorithm for

Computing Gröbner Bases (F_4). *Pure and Applied Algebra*, 139(1-3):61–88, June 1999.

[12] J.-C. Faugère. A New Efficient Algorithm for Computing Gröbner Bases Without Reduction to Zero (F_5). In *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation (ISSAC)*, pages 75 – 83. ACM, July 2002.

[13] S. Gao, Y. Guan, and F. V. IV. A New Incremental Algorithm for Computing Groebner Bases. In *Proceedings of The 2010 International Symposium on Symbolic and Algebraic Computation (ISSAC)*. ACM, 2010.

[14] R. Gebauer and H. Möller. On an Installation of Buchberger's Algorithm. *Journal of Symbolic Computation*, 6(2-3):275–286, 1988.

[15] D. Lazard. Gröbner-Bases, Gaussian Elimination and Resolution of Systems of Algebraic Equations. In *EUROCAL '83: Proceedings of the European Computer Algebra Conference on Computer Algebra*, pages 146–156. Springer-Verlag, 1983.

[16] M. S. E. Mohamed, D. Cabarcas, J. Ding, J. Buchmann, and S. Bulygin. MXL3: An Efficient Algorithm for Computing Gröbner Bases of Zero-Dimensional Ideals. In *Proceedings of The 12th International Conference on Information Security and Cryptology, (ICISC 2009)*, Lecture Notes in Computer Science. Springer-Verlag, Berlin, 2009.

[17] M. S. E. Mohamed, W. S. A. E. Mohamed, J. Ding, and J. Buchmann. MXL2: Solving Polynomial Equations over GF(2) Using an Improved Mutant Strategy. In *Proceedings of The Second international Workshop on Post-Quantum Cryptography, (PQCrypto08)*, Lecture Notes in Computer Science, pages 203–215. Springer-Verlag, Berlin, 2008.

[18] H. Möller, F. Mora, and C. Traverso. Gröbner Bases Computation Using Syzygies. In *The 1992 International Symposium on Symbolic and Algebraic Computation (ISSAC)*, pages 320–328. ACM Press, 1992.

[19] J. Patarin. Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): Two New Families of Asymmetric Algorithms. In *Proceeding of International Conference on the Theory and Application of Cryptographic Techniques Advances in Cryptology- Eurocrypt*, volume 1070 of *Lecture Notes in Computer Science*, pages 33–48. Springer, 1996.

[20] T. Stegers. *Faugère's F5 Algorithm Revisited*. PhD thesis, Technische Universität Darmstadt, 2005.