Efficient Computation of (Finite) Pommaret Bases

Bentolhoda Binaei[1], Amir Hashemi[1,2], and Werner M. Seiler[3]

[1] Department of Mathematical Sciences, Isfahan University of Technology
Isfahan, 84156-83111, Iran;
[2] School of Mathematics, Institute for Research in Fundamental Sciences (IPM),
Tehran, 19395-5746, Iran
h.binaei@math.iut.ac.ir
Amir.Hashemi@cc.iut.ac.ir
[3] Institut für Mathematik, Universität Kassel
Heinrich-Plett-Straße 40, 34132 Kassel, Germany
seiler@mathematik.uni-kassel.de

Abstract. In this paper, we describe a new and efficient algorithm to compute Pommaret bases. To this end, based on the method proposed by Möller et al. [20], we present a more efficient variant of Gerdt's algorithm (than the algorithm presented in [16]) to compute *minimal* involutive bases. Further, by using the *involutive version* of Hilbert driven technique, along with the new variant of Gerdt's algorithm, we modify the algorithm, given in [23], to compute a linear change of coordinates for a given homogeneous ideal so that the new ideal (after performing this change) possesses a finite Pommaret basis. All the proposed algorithms have been implemented in MAPLE and their efficiency is discussed via a set of benchmark polynomials.

1 Introduction

Gröbner bases are one of the most important concepts in computer algebra for dealing with multivariate polynomials. A Gröbner basis is a special kind of generating set for an ideal which provides a computational framework to determine many properties of the ideal. The notion of Gröbner bases was originally introduced in 1965 by Buchberger in his Ph.D. thesis and he also gave the basic algorithm to compute it [2, 3]. Later on, he proposed two criteria for detecting superfluous reductions to improve his algorithm [1]. In 1983, Lazard [19] developed new approach by making connection between Gröbner bases and linear algebra. In 1988, Gebauer and Möller [10] reformulated Buchberger's criteria in an efficient way to improve Buchberger's algorithm. Furthermore, Möller et al. in [20] proposed an improved version of Buchberger's algorithm by using the syzygies of constructed polynomials to detect useless reductions (this algorithm may be considered as the first *signature-based* algorithm to compute Gröbner bases). Relying on the properties of the Hilbert series of an ideal, Traverso [26] described the so-called *Hilbert-driven Gröbner basis algorithm* to improve Buchberger's algorithm by discarding useless critical pairs. In 1999, Faugère [6] presented his

F_4 algorithm to compute Gröbner bases which stems from Lazard's approach [19] and uses fast linear algebra techniques on sparse matrices (this algorithm has been efficiently implemented in Maple and Magma). In 2002, Faugère presented the famous F_5 algorithm for computing Gröbner bases [7]. The efficiency of this signature-based algorithm benefits from an incremental structure and two new criteria, namely *F_5 and IsRewritten criteria* (nowadays known respectively as signature and syzygy criteria). We remark that several authors have studied signature-based algorithms to compute Gröbner bases and as the novel approaches in this directions we refer to e.g. [8, 9].

Involutive bases may be considered as an extension of Gröbner bases (w.r.t. a restricted monomial division) for polynomial ideals which include additional combinatorial properties. The origin of involutive bases theory must be traced back to the work of Janet [18] on a constructive approach to the analysis of linear and certain quasi-linear systems of partial differential equations. Then Janet's approach was generalized to arbitrary (polynomial) differential systems by Thomas [25]. Based on the related methods developed by Pommaret in his book [21], the notion of *involutive polynomial bases* was introduced by Zharkov and Blinkov in [27]. Gerdt and Blinkov [13] introduced a more general concept of *involutive division* and *involutive bases* for polynomial ideals, along with algorithmic methods for their construction. An efficient algorithm was devised by Gerdt [12] (see also [15]) for computing involutive and Gröbner bases using the involutive form of Buchberger's criteria (see `http://invo.jinr.ru` for the efficiency analysis of the implementation of this algorithm). In this paper, we refer to this algorithm as *Gerdt's algorithm*. Finally, Gerdt et al. [16] described a signature-based algorithm (with an incremental structure) to apply the F_5 criterion for deletion of unnecessary reductions. Although this algorithm is faster than Gerdt's algorithm (see [16]), however its drawbacks are as follows: Due to its incremental structure (in order to apply the F_5 criterion), the selection strategy should be the POT module monomial ordering (which may be not efficient in general). Further, to respect the signature of computed polynomials, the reduction process may be not accomplished and (that may increase the number of intermediate polynomials) that may significantly affect the efficiency of computation. Finally, the involutive basis that this algorithm returns may be not minimal.

The aim of this paper is to provide an effective method to calculate *Pommaret bases*. These bases are a particular form of involutive bases introduced by Zharkov and Blinkov in [27] and contain many combinatorial properties of the ideals they generate, see e.g. [22–24] for a comprehensive study of Pommaret bases. They are not only of interest in computational aspects of algebraic geometry (e.g. by providing deterministic approaches to transform a given ideal into some classes of generic positions [23]), but they also serve in theoretical aspects of algebraic geometry (e.g. by providing simple and explicit formulas to read off many invariants of an ideal like dimension, depth and Castelnuovo-Mumford regularity [23]).

Relying on the method developed by Möller et al. [20], we give a new signature-based variant of Gerdt's algorithm to compute *minimal* involutive bases. In particular, the experiments show that the new algorithm is more efficient than Gerdt et al. algorithm [16]. On the other hand, [23] proposes an algorithm to compute *deterministically* a linear change of coordinates for a given homogeneous ideal so that the changed ideal (after performing this change) possesses a finite Pommaret basis (note that in general a given ideal does not have a finite Pommaret basis). In doing so, one computes iteratively the Janet bases of certain polynomial ideals. By applying the involutive version of Hilbert driven technique on the new variant of Gerdt's algorithm, we modify this algorithm to compute Pommaret bases. We have implemented all the algorithms described in this article and we assess their performance on a number of test examples.

The rest of the paper is organized as follows. In the next section, we will review the basic definitions and notations which will be used throughout this paper. Section 3 is devoted to the description of the new variant of Gerdt's algorithm. In Section 4, we present the improved algorithm to compute a linear change of coordinates for a given homogeneous ideal so that the new ideal has a finite Pommaret basis. Finally, we analyze the performance of the proposed algorithms in the last section.

2 Preliminaries

In this section, we review the basic definitions and notations from the theory of Gröbner bases and involutive bases that will be used in the rest of the paper. Throughout this paper we assume that $\mathcal{P} = \mathrm{k}[x_1, \dots, x_n]$ is the polynomial ring (where k is an infinite field). We consider also *homogeneous* polynomials $f_1, \dots, f_k \in \mathcal{P}$ and the ideal $\mathcal{I} = \langle f_1, \dots, f_k \rangle$ generated by them. We denote the total degree of and the degree w.r.t. a variable x_i of a polynomial $f \in \mathcal{P}$ respectively by $\deg(f)$ and $\deg_i(f)$. Let $\mathcal{M} = \{x_1^{\alpha_1} \cdots x_n^{\alpha_n} \mid \alpha_i \geq 0, 1 \leq i \leq n\}$ be the monoid of all monomials in $\mathcal{P}$. A monomial ordering on $\mathcal{M}$ is denoted by $\prec$ and throughout this paper we shall assume that $x_n \prec \cdots \prec x_1$. The leading monomial of a given polynomial $f \in \mathcal{P}$ w.r.t. $\prec$ will be denoted by $\mathrm{LM}(f)$. If $F \subset \mathcal{P}$ is a finite set of polynomials, we denote by $\mathrm{LM}(F)$ the set $\{\mathrm{LM}(f) \mid f \in F\}$. The leading coefficient of f, denoted by $\mathrm{LC}(f)$, is the coefficient of $\mathrm{LM}(f)$. The leading term of f is defined to be $\mathrm{LT}(f) = \mathrm{LM}(f)\,\mathrm{LC}(f)$. A finite set $G = \{g_1, \dots, g_k\} \subset \mathcal{P}$ is called a *Gröbner basis* of $\mathcal{I}$ w.r.t $\prec$ if $\mathrm{LM}(\mathcal{I}) = \langle \mathrm{LM}(g_1), \dots, \mathrm{LM}(g_k) \rangle$ where $\mathrm{LM}(\mathcal{I}) = \langle \mathrm{LM}(f) \mid f \in \mathcal{I} \rangle$. We refer e.g. to [4] for more details on Gröbner bases.

Let us recall the definition of Hilbert function and Hilbert series of a homogeneous ideal. Let $X \subset \mathcal{P}$ and s a positive integer. We define the degree s part X_s of X to be the set of all homogeneous elements of X of degree s.

Definition 1. *The* Hilbert function *of $\mathcal{I}$ is defined by* $\mathrm{HF}_{\mathcal{I}}(s) = \dim_{\mathrm{k}}(\mathcal{P}_s/\mathcal{I}_s)$ *where the right-hand side denotes the dimension of $\mathcal{P}_s/\mathcal{I}_s$ as a* k*-linear space.*

It is well-known that the Hilbert function of $\mathcal{I}$ is the same as that of $\mathrm{LT}(\mathcal{I})$ (see e.g. [4, Prop. 4, page 458]) and therefore the set of monomials not contained in

$\mathrm{LT}(\mathcal{I})$ forms a basis for $\mathcal{P}_s/\mathcal{I}_s$ as a k-linear space (Macaulay's theorem). This observation is the key idea behind the Hilbert-driven Gröbner basis algorithm. Roughly speaking, suppose that $\mathcal{I}$ is a homogeneous ideal and we want to compute a Gröbner basis of $\mathcal{I}$ by Buchberger's algorithm in increasing order w.r.t. the total degree of the S-polynomials. Assume that we know beforehand $\mathrm{HF}_{\mathcal{I}}(s)$ for a positive integer s. Suppose that we are at the stage where we are looking at the critical pairs of degree s. Consider the set P of all critical pairs of degree s. Then, we compare $\mathrm{HF}_{\mathcal{I}}(s)$ with the Hilbert function at s of the ideal generated by the leading terms of all already computed polynomials. If they are equal, we can remove P.

Below, we review some definitions and relevant results on involutive bases theory (see [12] for more details). We recall first involutive divisions based on partitioning the variables into two subsets of the variables, the so-called *multiplicative* and *non-multiplicative variables.*

Definition 2. *An* involutive division $\mathcal{L}$ *is given on* $\mathcal{M}$ *if for any finite set* $U \subset \mathcal{M}$ *and any* $u \in U$*, the set of variables is partitioned into the subset of multiplicative* $M_{\mathcal{L}}(u, U)$ *and non-multiplicative variables* $NM_{\mathcal{L}}(u, U)$ *such that the following three conditions hold where* $\mathcal{L}(u, U)$ *denotes the monoid generated by* $M_{\mathcal{L}}(u, U)$*:*

1. $v, u \in U,\ u\mathcal{L}(u, U) \cap v\mathcal{L}(v, U) \neq \emptyset \Rightarrow u \in v\mathcal{L}(v, U)$ *or* $v \in u\mathcal{L}(u, U)$*,*
2. $v \in U,\ v \in u\mathcal{L}(u, U) \Rightarrow \mathcal{L}(v, U) \subset \mathcal{L}(u, U)$*,*
3. $V \subset U$ *and* $u \in V \Rightarrow \mathcal{L}(u, U) \subset \mathcal{L}(u, V)$*.*

We shall write $u \mid_{\mathcal{L}} w$ *if* $w \in u\mathcal{L}(u, U)$*. In this case,* u *is called an* $\mathcal{L}$*-involutive divisor of* w *and* w *an* $\mathcal{L}$*-involutive multiple of* u*.*

We recall the definitions of the Janet and the Pommaret division, respectively.

Example 3. Let $U \subset \mathcal{P}$ be a finite set of monomials. For each sequence $d_1, \ldots, d_n$ of non-negative integers and for each $1 \leq i \leq n$ we define the subsets

$$[d_1, \ldots, d_i] = \{u \in U \mid d_j = \deg_j(u),\ 1 \leq j \leq i\}.$$

The variable x_1 is Janet multiplicative (denoted by $\mathcal{J}$-multiplicative) for $u \in U$ if $\deg_1(u) = \max\{\deg_1(v) \mid v \in U\}$. For $i > 1$ the variable x_i is Janet multiplicative for $u \in [d_1, \ldots, d_{i-1}]$ if $\deg_i(u) = \max\{\deg_i(v) \mid v \in [d_1, \ldots, d_{i-1}]\}$.

Example 4. For $u = x_1^{d_1} \cdots x_k^{d_k}$ with $d_k > 0$ the variables $\{x_k, \ldots, x_n\}$ are considered as Pommaret multiplicative (denoted by $\mathcal{P}$-multiplicative) and the other variables as Pommaret non-multiplicative. For $u = 1$ all the variables are multiplicative. The integer k is called the *class* of u and is denoted by $\mathrm{cls}(u)$.

The Pommaret division is called a *global division*, because the assignment of the multiplicative variables is independent of the set U. In order to avoid repeating notations let $\mathcal{L}$ always denote an involutive division.

Definition 5. *The set* $F \subset \mathcal{P}$ *is called* involutively head autoreduced *if for each* $f \in F$ *there is no* $h \in F \setminus \{f\}$ *with* $\mathrm{LM}(h) \mid_{\mathcal{L}} \mathrm{LM}(f)$*.*

Definition 6. *Let $I \subset \mathcal{P}$ be an ideal. An $\mathcal{L}$-involutively head autoreduced subset $G \subset \mathcal{I}$ is an $\mathcal{L}$-*involutive basis *for $\mathcal{I}$ (or simply either an involutive basis or $\mathcal{L}$-basis) if for all $f \in \mathcal{I}$ there exists $g \in G$ so that $\mathrm{LM}(g) \mid_{\mathcal{L}} \mathrm{LM}(f)$.*

Example 7. Let $\mathcal{I} = \{x_1^2x_3, x_1x_2, x_1x_3^2\} \subset \mathrm{k}[x_1, x_2, x_3]$. Then, $\{x_1^2x_3, x_1x_2, x_1x_3^2, x_1^2x_2\}$ is a Janet basis for $\mathcal{I}$ and $\{x_1^2x_3, x_1x_2, x_1x_3^2, x_1^2x_2, x_1^{i+3}x_2, x_1^{i+3}x_3 \mid i \geq 0\}$ is a (infinite) Pommaret basis for $\mathcal{I}$. Indeed, Janet division is Noetherian, however Pommaret division is non-Noetherian (see [14] for more details).

Gerdt in [12] proposed an efficient algorithm to construct involutive bases based on a completion process where prolongations of generators by non-multiplicative variables are reduced. This process terminates in finitely many steps for any Noetherian division.

Definition 8. *Let $F \subset \mathcal{P}$ be a finite. Following the notations in [23], the* involutive span *generated by F is denoted by $\langle F \rangle_{\mathcal{L},\prec}$.*

Thus, a set $F \subset \mathcal{I}$ is an involutive basis for $\mathcal{I}$ if we have $\mathcal{I} = \langle F \rangle_{\mathcal{L},\prec}$.

Definition 9. *Let $F \subset \mathcal{I}$ be an involutively head autoreduced set of homogeneous polynomials. The* involutive Hilbert function *of F is defined by $\mathrm{IHF}_F(s) = \dim_{\mathrm{k}}(\mathcal{P}_s / (\langle F \rangle_{\mathcal{L},\prec})_s)$.*

Since F is involutively head autoreduced, one easily recognizes that $\langle F \rangle_{\mathcal{L},\prec} = \bigoplus_{f \in F} \mathrm{k}[M_{\mathcal{L}}(\mathrm{LM}(f), \mathrm{LM}(F))] \cdot f$. Thus using the well-known combinatorial formulas to count the number of monomials in certain variables, we get

$$\mathrm{IHF}_I(s) = \binom{n+s-1}{s} - \sum_{f \in F} \binom{s - \deg(f) + k_f - 1}{s - \deg(f)}$$

where k_f is the number of multiplicative variables of f (see e.g. [12]). We remark that an involutively head autoreduced subset $F \subset \mathcal{I}$ is an involutive basis for $\mathcal{I}$ if and only if $\mathrm{HF}_{\mathcal{I}}(s) = \mathrm{IHF}_F(s)$ for each s.

3 Using Syzygies to Compute Involutive Bases

We now propose a variant of Gerdt's algorithm [12] by using the intermediate computed syzygies to compute involutive bases and especially Janet bases. For this, we recall briefly the signature-based variant of Möller et al. algorithm [20] to compute Gröbner bases (the practical results are given in Section 5).

Definition 10. *Let us consider $F = (f_1, \ldots, f_k) \in \mathcal{P}^k$. The (first)* syzygy module *of F is defined to be $\mathrm{Syz}(F) = \{(h_1, \ldots, h_k) \mid h_i \in \mathcal{P}, \sum_{i=1}^k h_i f_i = 0\}$.*

Schreyer in his master thesis proposed a slight modification of Buchberger's algorithm to compute a Gröbner basis for the module of syzygies of a Gröbner basis. The construction of this basis relies on the following key observation (see [5]): Let $G = \{g_1, \ldots, g_s\}$ be a Gröbner basis. By tracing the dependency of each

$\mathrm{SPoly}(g_i, g_i)$ on G we can write $\mathrm{SPoly}(g_i, g_j) = \sum_{k=1}^{s} a_{ijk} g_k$ with $a_{ijk} \in \mathcal{P}$. Let $\mathbf{e}_1, \ldots, \mathbf{e}_s$ be the standard basis for $\mathcal{P}^s$ and $m_{ij} = lcm(\mathrm{LT}(g_i), \mathrm{LT}(g_j))$. Set

$$\mathbf{s}_{ij} = m_{i,j} / \mathrm{LT}(g_i).\mathbf{e}_i - m_{i,j} / \mathrm{LT}(g_j).\mathbf{e}_j - (a_{ij1}\mathbf{e}_1 + a_{ij2}\mathbf{e}_2 + \cdots + a_{ijs}\mathbf{e}_s).$$

Definition 11. *Let* $G = \{g_1, \ldots, g_s\} \subset \mathcal{P}$. Schreyer's module ordering *is defined as follows:* $x^\beta \mathbf{e}_j \prec_s x^\alpha \mathbf{e}_i$ *if* $\mathrm{LT}(x^\beta g_j) \prec \mathrm{LT}(x^\alpha g_i)$ *and breaks ties by* $i < j$.

Theorem 12 (Schreyer's Theorem). *For a Gröbner basis* $G = \{g_1, \ldots, g_s\}$ *the set* $\{\boldsymbol{s}_{ij} \mid 1 \leq i < j \leq s\}$ *forms a Gröbner basis for* $\mathrm{Syz}(g_1, \ldots, g_s)$ *w.r.t.* $\prec_s$.

Example 13. Let $F = \{xy - x, x^2 - y\} \subset \mathrm{k}[x, y]$. The Gröbner basis of F w.r.t. $x \prec_{dlex} y$ is $G = \{g_1 = xy - x, g_2 = x^2 - y, g_3 = y^2 - y\}$ and the Gröbner basis of $\mathrm{Syz}(g_1, g_2, g_3)$ is $\{(x, -y+1, -1), (-x, y^2-1, -x^2+y+1), (y, 0, -x)\}$.

According to this observation, Möller et al. [20] proposed a variant of Buchberger's algorithm by using the syzygies of constructed polynomials to remove superfluous reductions. Algorithm 1 below corresponds to it with a slight modification to derive a signature-based algorithm to compute Gröbner bases. We associate to each polynomial f, the two-tuple $p = (f, m\mathbf{e}_i)$ where $\mathrm{Poly}(p) = f$ is the polynomial part of f and $\mathrm{Sig}(p) = m\mathbf{e}_i$ is its signature. Further, the function NormalForm(f, G) returns a remainder of the division of f by G. Further, if $\mathrm{Sig}(p) = m\mathbf{e}_i$ in the first step of reduction process we must not use $f_i \in G$. We

Algorithm 1 GröbnerBasis

```
Input: A set of polynomials $F \subset \mathcal{P}$; a monomial ordering $\prec$
Output: A Gröbner basis $G$ for $\langle F \rangle$
$G := \{\}$ and $syz := \{\}$
$P := \{(F[i], \mathbf{e}_i) \mid i = 1, \ldots, |F|\}$
while  $P \neq \emptyset$ do
  select (using normal strategy) and remove $p \in P$
  if  $\nexists\ s \in syz$ s.t. $s \mid \mathrm{Sig}(p)$  then
    $f := \mathrm{Poly}(t)$
    $h :=$ NormalForm$(f, G)$
    $syz := syz \cup \{\mathrm{Sig}(t)\}$
    if $h \neq 0$  then
      $j := |G| + 1$
      for $g \in G$ do
        $P := P \cup \{(r.h, r.\mathbf{e}_j)\}$ s.t. $r.\mathrm{LM}(h) = \mathrm{LCM}(\mathrm{LM}(g), \mathrm{LM}(h))$
        $G := G \cup \{h\}$
        $syz := syz \cup \{\mathrm{LM}(g).\mathbf{e}_j \mid \mathrm{LM}(h)$ and $\mathrm{LM}(g)$ are coprime$\}$
      od
    fi
  fi
od
return $(G)$
```

show now how to apply this structure to improve Gerdt's algorithm [15].

Definition 14. *Let $F = (f_1, \ldots, f_k) \subset \mathcal{P}^k$ be a sequence of polynomials. The* involutive syzygy module $\mathrm{ISYZ}(F)$ *of F is the set of all $(h_1, \ldots, h_k) \in \mathcal{P}^k$ so that $\sum_{i=1}^{k} h_i f_i = 0$ where $h_i \in \mathrm{k}[M_{\mathcal{L}}(\mathrm{LM}(f_i), \mathrm{LM}(F))]$.*

[23, Thm. 5.10] contains an involutive version of Schreyer's theorem replacing S-polynomials by non-multiplicative prolongations and using involutive division. Algorithm 2 below represents the new variant of Gerdt's algorithm for computing involutive bases using involutive syzygies. For this purpose, we associate to each polynomial f, the quadruple $p = (f, g, V, m.\mathbf{e}_i)$ where $f = \mathrm{Poly}(p)$ is the polynomial itself, $g = \mathrm{Anc}(p)$ is its ancestor, $V = \mathrm{NM}(p)$ is the list of non-multiplicative variables of f which have been already processed in the algorithm and $m.\mathbf{e}_i = \mathrm{Sig}(p)$ is the signature of f. If P is a set of quadruple, we denote by $\mathrm{Poly}(P)$ the set $\{\mathrm{Poly}(p) \mid p \in P\}$.

Algorithm 2 InvolutiveBasis

```
Input: A finite set $F \subset \mathcal{P}$; an involutive division $\mathcal{L}$; a monomial ordering $\prec$
Output: A minimal $\mathcal{L}$-basis for $\langle F \rangle$
$F :=$sort$(F, \prec)$
$T := \{(F[1], F[1], \emptyset, \mathbf{e}_1)\}$
$Q := \{(F[i], F[i], \emptyset, \mathbf{e}_i) \mid i = 2, \ldots, |F|\}$
$syz := \{\}$
while $Q \neq \emptyset$ do
  $Q :=$sort$(Q, \prec_s)$
  $p := Q[1]$
  if $\nexists s \in syz$ s.t $s \mid \mathrm{Sig}(p)$ with non-constant quotient then
    $h :=$ InvolutiveNormalForm$(p, T, \mathcal{L}, \prec)$
    $syz := syz \cup \{h[2]\}$
    if $h = 0$ and $\mathrm{LM}(\mathrm{Poly}(p)) = \mathrm{LM}(\mathrm{Anc}(p))$ then
      $Q := \{q \in Q \mid \mathrm{Anc}(q) \neq \mathrm{Poly}(p)\}$
    fi
    if $h \neq 0$ and $\mathrm{LM}(\mathrm{Poly}(p)) \neq \mathrm{LM}(h)$ then
      for $q \in T$ with proper conventional division $\mathrm{LM}(\mathrm{Poly}(h)) \mid \mathrm{LM}(\mathrm{Poly}(q))$ do
        $Q := Q \cup \{q\}$
        $T := T \setminus \{q\}$
      od
      $j := |T| + 1$
      $T := T \cup \{(h, h, \emptyset, \mathbf{e}_j)\}$
    else
      $T := T \cup \{(h, \mathrm{Anc}(p), \mathrm{NM}(p), \mathrm{Sig}(p))\}$
    fi
    for $q \in T$ and $x \in NM_{\mathcal{L}}(\mathrm{LM}(\mathrm{Poly}(q)), \mathrm{LM}(\mathrm{Poly}(T)) \setminus \mathrm{NM}(q))$ do
      $Q := Q \cup \{(x.\mathrm{Poly}(q), \mathrm{Anc}(q), \emptyset, x.\mathrm{Sig}(q))\}$
      $\mathrm{NM}(q) := \mathrm{NM}(q) \cup NM_{\mathcal{L}}(\mathrm{LM}(\mathrm{Poly}(q)), \mathrm{LM}(\mathrm{Poly}(T))) \cup \{x\}$
    od
  fi
od
return $(\mathrm{Poly}(T))$
```

In this algorithm, the functions $\operatorname{sort}(X, \prec)$ and $\operatorname{sort}(X, \prec_s)$ sort X by increasing, respectively, $\operatorname{LM}(X)$ w.r.t. $\prec$ and $\{\operatorname{Sig}(p) \mid p \in X\}$ w.r.t. $\prec_s$. The involutive normal form algorithm is given in Algorithm 3.

Algorithm 3 InvolutiveNormalForm

```
Input: A quadruple $p$; a set of quadruples $T$; an involutive division $\mathcal{L}$; a monomial ordering $\prec$
Output: An $\mathcal{L}$-normal form of $p$ modulo $T$, and the corresponding signature, if any
$S := \{\}$ and $h := \operatorname{Poly}(p)$ and $G := \operatorname{Poly}(T)$
while $h$ has a monomial $m$ which is $\mathcal{L}$-divisible by $G$ do
  select $g \in G$ with $\operatorname{LM}(g) \mid_{\mathcal{L}} m$
  if $m = \operatorname{LM}(\operatorname{Poly}(p))$ and $(m/\operatorname{LM}(g).\operatorname{Sig}(g) = \operatorname{Sig}(p)$ or $\textsc{Criteria}(h, g))$ then
    return $(0, S)$
  fi
  if $m = \operatorname{LM}(\operatorname{Poly}(p))$ and $m/\operatorname{LM}(g).\operatorname{Sig}(g) \prec_s \operatorname{Sig}(p)$ then
    $S := S \cup \{\operatorname{Sig}(p)\}$
  fi
  $h := h - cm/\operatorname{LT}(g).g$ where $c$ is the coefficient of $m$ in $h$
od
return $(h, S)$
```

Furthermore, we apply the involutive form of Buchberger's criteria from [12]. We say that $\textsc{Criteria}(p, g)$ holds if either $C_1(p, g)$ or $C_2(p, g)$ holds where $C_1(p, g)$ is true if $\operatorname{LM}(\operatorname{Anc}(p)).\operatorname{LM}(\operatorname{Anc}(g)) = \operatorname{LM}(\operatorname{Poly}(p))$ and $C_2(p, g)$ is true if $\operatorname{LCM}(\operatorname{LM}(\operatorname{Anc}(p)), \operatorname{LM}(\operatorname{Anc}(g)))$ properly divides $\operatorname{LM}(\operatorname{Poly}(p))$.

Remark 15. We shall remark that, due to the second **if**-loop in Algorithm 3, if $m_i\mathbf{e}_i$ is added into syz then there exists an involutive representation of the form $m_i g_i = \sum_{j=1}^{\ell} h_j g_j + h$ where $T = \{g_1, \ldots, g_\ell\} \subset \mathcal{P}$ is the output of the algorithm, h is $\mathcal{L}$-normal form of p modulo T and $\operatorname{LM}(h_j)\mathbf{e}_j \prec_s m_i\mathbf{e}_i$ for each j.

In the next proof, by an abuse of notation, we refer to the signature of a quadruple as the signature of its polynomial part.

Theorem 16. InvolutiveBasis *terminates in finitely many steps (if $\mathcal{L}$ is a Noetherian division) and returns a minimal involutive basis for its input ideal.*

Proof. The termination and correctness of the algorithm are inherited from those of Gerdt's algorithm [12] provided that we show that any polynomial removed using syzygies is superfluous. This happens in both algorithms. Let us deal first with Algorithm 2. Now, suppose that for $p \in Q$ there exists $s \in syz$ so that $s \mid \operatorname{Sig}(p)$ with non-constant quotient. Suppose that $\operatorname{Sig}(p) = m_i\mathbf{e}_i$ and $s = m_i'\mathbf{e}_i$ where $m_i = um_i'$ with $u \neq 1$. Let $T = \{g_1, \ldots, g_\ell\} \subset \mathcal{P}$ be the output of the algorithm and $m_i' g_i = \sum_{j=1}^{\ell} h_j g_j + h$ be the representation of $m_i' g_i$ with $f, g_j \in T, h_j \in P$ and h the involutive remainder of the division of $m_i' g_i$ by T. Then, from the structure of both algorithms, it yields that $\mathrm{LM}(h_j g_j) \prec \mathrm{LM}(m'_i g_i)$. In particular, we have $\mathrm{LM}(h_j)\mathbf{e}_j \prec_s m'_i\mathbf{e}_i$ for each j. This follows that $\mathrm{LM}(uh_j)\mathbf{e}_j \prec_s um'_i\mathbf{e}_i = m_i\mathbf{e}_i$ for each j. On the other hand, if $h \neq 0$ then again by the structure of the algorithm uh has a signature less than $m_i\mathbf{e}_i$. For each j and for each term t in h_j we know that the signature of utg_j is less than $m_i\mathbf{e}_i$ and by the selection strategy used in the algorithm which is based on Schreyer's ordering, utg_j should be studied before $m'_i g_i$ and therefore it has an involutive representation in terms of T. Furthermore, the same holds also for uh provided that $h \neq 0$. These arguments show that $m'_i g_i$ is unnecessary and it can be omitted. Now we turn to Algorithm 3. Let $p \in Q$ and $g \in T$ so that $\mathrm{LM}(h) = u\,\mathrm{LT}(g)$ and $\mathrm{Sig}(p) = u\,\mathrm{Sig}(g)$ where $h = \mathrm{Poly}(p)$ and u is a monomial. Using the above notations, let $\mathrm{Sig}(p) = m_i\mathbf{e}_i$ and $\mathrm{Sig}(g) = m'_i\mathbf{e}_i$ where $m_i = um'_i$. Further, let $m'_i g_i = \sum_{j=1}^{\ell} h_j g_j + g$ be the representation of $m'_i g_i$ with $\mathrm{LM}(h_j)\mathbf{e}_j \prec_s m'_i\mathbf{e}_i$ for each j. It follows from the assumption that $\mathrm{LM}(h_j g_j) \prec \mathrm{LM}(m'_i g_i) = \mathrm{LM}(g)$ for each j. We can write $um'_i g_i = \sum_{j=1}^{\ell} uh_j g_j + ug$. Since $\mathrm{LM}(uh_j)\mathbf{e}_j \prec_s um'_i\mathbf{e}_i = m_i\mathbf{e}_i$ for each j then, by repeating the above argument, we deduce that $uh_j g_j$ for each j has an involutive representtaion. Therefore, $um'_i g_i$ has a representation using the fact that u is multiplicative for g. Thus h has a representation and it can be removed. $\square$

4 Hilbert Driven Pommaret Bases Computations

As we mentioned Pommaret division is not Noetherian and therefore, a given ideal may not have a finite Pommaret basis. However, if the ideal is in *quasi-stable position* it has a finite Pommaret basis. On the other hand, a generic linear change of variables transforms an ideal in such a position. Thus, one of the challenges in this direction is to find a linear change of variables so that the ideal after performing this change possesses a finite Pommaret basis. [23] proposes a deterministic algorithm to compute such a linear change by computing repeatedly the Janet basis of the last transformed ideal. In this section, by using the algorithm described in Section 3, we show how one can incorporate an involutive version of Hilbert driven strategy to improve this algorithm.

Algorithm 4 HDQuasiStable

```
Input: A finite set $F \subset \mathcal{P}$ and a monomial ordering $\prec$
Output: A linear change $\Phi$ so that $\langle \Phi(F) \rangle$ has a finite Pommaret basis
$\Phi := \emptyset$ and $J :=$InvolutiveBasis$(F, \mathcal{J}, \prec)$ and $A :=$test$(\mathrm{LM}(J), \prec)$
while $A \neq$ true do
    $G :=$ substitution of $\phi := A[3] \mapsto A[3] + cA[2]$ in $J$ for a random choice of $c \in K$
    $Temp :=$HDInvolutiveBasis$(G, \mathcal{J}, \prec)$
    $B :=$test$(\mathrm{LM}(Temp))$
    if $B \neq A$ then
        $\Phi := \Phi, \phi$ and $J := Temp$ and $A := B$
    fi
od
return $(\Phi)$
```

It is worth noting that in [23] it is proposed to perform a Pommaret head autoreduced process on the calculated Janet basis at each iteration. However, we do not need to perform this operation because each computed Janet basis is *minimal* and by [11, Cor. 15] each minimal Janet basis is Pommaret head autoreduced. All the used functions are described below. By the structure of the algorithm, we first compute a Janet basis for the input ideal using InvolutiveBasis algorithm. From this basis, one can read off easily the Hilbert function of the input ideal. Further, the Hilbert function of an ideal does not change after performing a linear change of variables. Thus we can apply this Hilbert function in the next Janet bases computations as follows. The algorithm has the same structure as the InvolutiveBasis algorithm and so we remove the similar lines. We add the next written lines in HDInvolutiveBasis algorithm between $p := Q[1]$ and the first **if**-loop in InvolutiveBasis algorithm.

Algorithm 5 HDInvolutiveBasis

```
Input: A set of monomials $F$; an involutive division $\mathcal{L}$ ; a monomial ordering $\prec$
Output: A minimal $\mathcal{L}$-involutive basis for $\langle F \rangle$
⋮
$d := \deg(p)$
while $\mathrm{HF}_{\langle F \rangle}(d) = \mathrm{IHF}_T(d)$ do
   remove from $Q$ all $q \in Q$ s.t. $\deg(\mathrm{Poly}(q)) = d$
   if $Q = \emptyset$ then
      return $(\mathrm{Poly}(T))$
   else
      $p := Q[1]$
      $d := \deg(p)$
   fi
od
⋮
```

Algorithm 6 test

```
Input: A finite set $U$ of monomials
Output: True if any element of $U$ has the same number of Pommaret and Janet multiplicative variables, and false otherwise
if $\exists u \in U$ s.t. $M_{\mathcal{P},\prec}(u,U) \neq M_{\mathcal{J},\prec}(u,U)$ then
   $V := M_{\mathcal{J},\prec}(u,F) \setminus M_{\mathcal{P},\prec}(u,F)$
   return$(false, V[1], x_{\mathrm{cls}(u)})$
fi
return $(true)$
```

Theorem 17. *The algorithm* HDQuasiStable *terminates in finitely many steps and it returns a linear change of variables for a given homogeneous ideal so that the changed ideal (after performing the change on the input ideal) possesses a finite Pommaret basis.*

Proof. Let $\mathcal{I}$ be the ideal generated by F; the input of HDQuasiStable algorithm. The termination of this algorithm follows, from one side, from the termination of the algorithms to compute Janet bases. From the other side, [23, Prop. 2.9] shows that there exists an open Zariski set U of $\mathrm{k}^{n\times n}$ so that for each linear change of variables, say Φ corresponding to an element of U we have $\Phi(\mathcal{I})$ has a finite Pommaret basis. Moreover, he proved that the process of finding such a linear change termintaes in finitely many steps (see [23, Rem. 9.11]). Taken together, these arguments show that HDQuasiStable algorithm terminates. To prove the correctness, using the notations of HDInvolutiveBasis algorithm, we shall prove that any $p \in Q$ removed by Hilbert driven strategy reduces to zero. In this direction, we recall that any change of variables is a linear automorphism of $\mathcal{P}$, [17, page 52]. Thus, for each i, the dimension over k of components of degree i of $\mathcal{I}$ and that of $\mathcal{I}$ after the change remains stable. This yields that the Hilbert function of $\mathcal{I}$ does not change after a linear change of variables. Let J be the Janet basis computed by InvolutiveBasis. One can readily observe that $\mathrm{HF}_{\mathcal{I}}(d) = \mathrm{IHF}_J(d)$ for each d, and therefore from the first Janet basis one can derive the Hilbert function of $\mathcal{I}$ and use it to improve the next Janet bases computations. Now, suppose that F is the input of HDInvolutiveBasis algorithm, $p \in Q$ and $\mathrm{HF}_{\mathcal{I}}(d) = \mathrm{IHF}_T(d)$ for $d = \deg(\mathrm{Poly}(p))$. It follows that $\dim_{\mathrm{k}}(\langle F\rangle_d) = \dim_{\mathrm{k}}(\langle \mathrm{Poly}(T)\rangle_d)$ and therefore the polynomials of $\mathrm{Poly}(T)$ generate involutively whole $\langle F\rangle_d$ and this shows that p is superfluous which ends the proof. □

Remark 18. We remark that we assumed that the input of InvolutiveBasis and HDQuasiStable algorithms should be homogeneous, however the former algorithm works also for non-homogeneous ideals. Further, the latter algorithm also may be applied for non-homogeneous ideals provided that we consider the affine Hilbert function for such an ideal; i.e. $\mathrm{HF}_{\mathcal{I}}(s) = \dim_{\mathrm{k}}(\mathcal{P}_{\leq s}/\mathcal{I}_{\leq s})$.

[23] provides a number of equivalent characterizations of the ideals which have finite Pommaret bases. Indeed, a given ideal has a finite Pommaret basis if only if the ideal is in *quasi stable position* (or equivalently if the coordinates are δ-regular) see [23, Prop. 4.4].

Definition 19. *A monomial ideal $\mathcal{I}$ is called* quasi stable *if for any monomial $m \in \mathcal{I}$ and all integers i, j, s with $1 \leq j < i \leq n$ and $s > 0$, if $x_i^s \mid m$ there exists an integer $t \geq 0$ such that $x_j^t m/x_i^s \in \mathcal{I}$. A homogeneous ideal $\mathcal{I}$ is in* quasi stable position *if* $\mathrm{LT}(\mathcal{I})$ *is quasi stable.*

Example 20. The ideal $\mathcal{I} = \langle x_2^2x_3, x_2^3, x_1^3\rangle \subset \mathrm{k}[x, y, z]$ is a quasi stable monomial ideal and its Pommaret basis is $\{x_2^2x_3, x_2^3, x_1^3, x_1x_2^2x_3, x_1x_2^3, x_1^2x_2^2x_3, x_1^2x_2^3\}$.

5 Experiments and Comparison

We have implemented both algorithms InvolutiveBasis and HDQuasiStable in Maple 17[4]. It is worth noting that, in the given paper, we are willing to

[4] The Maple code of the implementations of our algorithms and examples are available at `http://amirhashemi.iut.ac.ir/softwares`

compare behavior of InvolutiveBasis and HDQuasiStable algorithms with Gerdt et al. [16] and QuasiStable [23] algorithms, respectively (we shall remark that QuasiStable has the same structure as the HDQuasiStable, however to compute Janet bases we use Gerdt's algorithm). For this purpose, we used some well-known examples from computer algebra literature. All computations were done over Q, and for the input degree-reverse-lexicographical monomial ordering. The results are shown in the following tables where the time and memory columns indicate, respectively, the consumed CPU time in seconds and amount of megabytes of used memory. The C_1 and C_2 columns show, respectively, the number of polynomials removed by C_1 and C_2 criteria by the corresponding algorithm. The sixth column shows the number of polynomials eliminated by the new criterion related to syzygies applied in InvolutiveBasis and InvolutiveNormalForm algorithms. The F_5 and S columns show the number of polynomials removed, respectively, by F_5 and super-top-reduction criteria. Three last columns represent, respectively, the number of reductions to zero, the number and the maximum degree of polynomials in the final involutive basis (we note that for Gerdt et al. algorithm the number of polynomials is the size of the basis after the minimal process). The computations in this paper are performed on a personal computer with 2.70 GHz Intel(R) Core(TM) i7-2620M CPU, 8 GB of RAM, 64 bits under the Windows 7 operating system.

Liu	time	memory	C_1	C_2	Syz	F_5	S	redz	poly	deg
InvolutiveBasis	1.09	37.214	4	3	2	-	-	25	19	6
Gerdt et al.	2.901	41.189	7	39	-	25	0	1	19	7

Noon	time	memory	C_1	C_2	Syz	F_5	S	redz	poly	deg
InvolutiveBasis	3.822	43.790	4	15	6	-	-	69	51	10
Gerdt et al.	45.271	670.939	8	107	-	49	3	17	51	10

Haas3	time	memory	C_1	C_2	Syz	F_5	S	redz	poly	deg
InvolutiveBasis	8.424	95.172	0	20	24	-	-	203	73	13
Gerdt et al.	41.948	630.709	1	88	-	88	16	68	73	13

Sturmfels-Eisenbud	time	memory	C_1	C_2	Syz	F_5	S	redz	poly	deg
InvolutiveBasis	22.932	255.041	28	103	95	-	-	245	100	6
Gerdt et al.	2486.687	30194.406	29	1379	-	84	11	40	100	9

Lichtblau	time	memory	C_1	C_2	Syz	F_5	S	redz	poly	deg
InvolutiveBasis	24.804	391.3	0	5	6	-	-	19	35	11
Gerdt et al.	205.578	3647.537	0	351	-	18	0	31	35	19

Eco7	time	memory	C_1	C_2	Syz	F_5	S	redz	poly	deg
InvolutiveBasis	40.497	473.137	51	21	30	-	-	201	45	6
Gerdt et al.	1543.068	25971.714	63	1717	-	175	8	18	45	11

Katsura5	time	memory	C_1	C_2	Syz	F_5	S	redz	poly	deg
InvolutiveBasis	46.956	630.635	21	0	2	-	-	68	23	12
Gerdt et al.	42.416	621.551	62	73	-	114	1	21	23	8

Katsura6	time	memory	C_1	C_2	Syz	F_5	S	redz	poly	deg
InvolutiveBasis	81.526	992.071	43	0	4	-	-	171	43	8
Gerdt et al.	608.325	795.196	77	392	-	209	1	41	43	11

As one can observe InvolutiveBasis is a signature-based variant of Gerdt's algorithm which has a structure closer to Gerdt's algorithm and it is more efficient than Gerdt et al. algorithm. Moreover, we can see the detection of criteria and the number of reductions to zero by the algorithms are different. Indeed, this difference is due to the selection strategy used in each algorithm. More precisely, in teh Gerdt et al. algorithm the set of non-multiplicative prolongations is sorted by POT ordering however in InvolutiveBasis it is sortd using Schreyer ordering. However, one needs to implement it efficiently in C/C++ to be able to compare it with GINV software[5].

The next tables illustrate an experimental comparison of HDQuasiStable and QuasiStable algorithms. In these tables HD column shows the number of polynomials removed by Hilbert driven strategy in the corresponding algorithm. Further, the chen column shows the number of linear changes that one needs to transform the corresponding ideal into quasi stable position.

Liu	time	memory	C_1	C_2	Syz	HD	redz	chen	deg
HDQuasiStable	4.125	409.370	4	3	2	93	56	4	6
QuasiStable	9.56	1067.725	14	3	-	-	151	4	6

Katsura5	time	memory	C_1	C_2	Syz	HD	redz	chen	deg
HDQuasiStable	67.234	9191.288	44	3	6	185	168	2	8
QuasiStable	145.187	26154.263	86	29	-	-	359	2	8

Weispfenning94	time	memo	C_1	C_2	Syz	HD	reds	chen	deg
HDQuasiStable	110.339	6268.532	0	1	9	45	170	1	15
QuasiStable	243.798	16939.468	0	2	-	-	85	1	15

Noon	time	memory	C_1	C_2	Syz	HD	redz	chen	deg
HDQuasiStable	667.343	66697.995	4	25	6	325	119	4	11
QuasiStable	1210.921	205149.994	16	35	-	-	450	4	11

Sturmfels-Eisenbud	time	memory	C_1	C_2	Syz	HD	redz	poly	deg
HDQuasiStable	1507.640	125904.515	86	308	440	1370	1804	12	8
QuasiStable	843.171	96410.344	218	1051	-	-	3614	12	8

Eco7	time	memory	C_1	C_2	Syz	HD	redz	chen	deg
QuasiStable	2182.296	241501.340	298	98	373	1523	1993	8	11
QuasiStable	2740.734	500857.600	547	725	-	-	3889	8	11

Haas3	time	memory	C_1	C_2	Syz	HD	redz	chen	deg
HDQuasiStable	5505.375	906723.699	0	0	91	84	255	1	33
QuasiStable	10136.718	1610753.428	1	120	-	-	430	1	33

Lichtblau	time	memory	C_1	C_2	Syz	HD	redz	chen	deg
HDQuasiStable	16535.593	2051064.666	0	44	266	217	265	2	30
QuasiStable	18535.625	2522847.256	0	493	-	-	751	2	30

Acknowledgments.

The research of the second author was in part supported by a grant from IPM (No. 94550420).

References

1. Buchberger, B.: A criterion for detecting unnecessary reductions in the construction of Gröbner-bases. Symbolic and algebraic computation, EUROSAM '79, int. Symp., Marseille 1979, Lect. Notes Comput. Sci. 72, 3-21 (1979). (1979)

[5] See `http://invo.jinr.ru`

2. Buchberger, B.: Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal. Innsbruck: Univ. Innsbruck, Mathematisches Institut (Diss.) (1965)
3. Buchberger, B.: Bruno Buchberger's PhD thesis 1965: An algorithm for finding the basis elements of the residue class ring of a zero dimensional polynomial ideal. Translation from the German. J. Symb. Comput. 41(3-4), 475–511 (2006)
4. Cox, D., Little, J., O'Shea, D.: Ideals, varieties, and algorithms. An introduction to computational algebraic geometry and commutative algebra. 3rd ed. New York, NY: Springer, 3rd ed. edn. (2007)
5. Cox, D.A., Little, J., O'Shea, D.: Using algebraic geometry, Graduate Texts in Mathematics, vol. 185. Springer, New York, second edn. (2005)
6. Faugère, J.C.: A new efficient algorithm for computing Gröbner bases (F_4). J. Pure Appl. Algebra 139(1-3), 61–88 (1999)
7. Faugère, J.C.: A new efficient algorithm for computing Gröbner bases without reduction to zero (F_5). In: Proceedings of theinternational symposium on symbolic and algebraic computation, ISSAC'02. Lille, France, July 07–10, pp. 75–83 (2002)
8. Gao, S., Guan, Y., Volny, F.: A new incremental algorithm for computing Groebner bases. In: Proceedings of the international symposium on symbolic and algebraic computation, ISSAC'10, Munich, Germany, July 25–28, pp. 13–19 (2010)
9. Gao, S., Volny, F.I., Wang, M.: A new framework for computing Gröbner bases. Math. Comput. 85(297), 449–465 (2016)
10. Gebauer, R., Möller, H.: On an installation of Buchberger's algorithm. J. Symb. Comput. 6(2-3), 275–286 (1988)
11. Gerdt, V.P.: On the relation between Pommaret and Janet bases. In: Computer algebra in scientific computing. CASC 2000. Proceedings of the 3rd workshop, Samarkand, Uzbekistan, October 5–9, 2000, pp. 167–181. Berlin: Springer (2000)
12. Gerdt, V.P.: Involutive algorithms for computing Gröbner bases. In: Computational commutative and non-commutative algebraic geometry. Proceedings of the NATO Advanced Research Workshop, Chisinau, Republic of Moldova, June 6–11, 2004, pp. 199–225. Amsterdam: IOS Press (2005)
13. Gerdt, V.P., Blinkov, Y.A.: Involutive bases of polynomial ideals. Math. Comput. Simul. 45(5-6), 519–541 (1998)
14. Gerdt, V.P., Blinkov, Y.A.: Involutive bases of polynomial ideals. Math. Comput. Simul. 45(5-6), 519–541 (1998)
15. Gerdt, V.P., Hashemi, A., M.-Alizadeh, B.: A Variant of Gerdt's Algorithm for Computing Involutive Bases. Bulletin of PFUR. Series Mathematics. Information Sciences. Physics 2, 65–76 (2012)
16. Gerdt, V.P., Hashemi, A., M.-Alizadeh, B.: Involutive bases algorithm incorporating F_5 criterion. J. Symb. Comput. 59, 1–20 (2013)
17. Herzog, J., Hibi, T.: Monomial ideals. London: Springer (2011)
18. Janet, M.: Sur les systèmes d'équations aux dérivées partielles. C. R. Acad. Sci., Paris 170, 1101–1103 (1920)
19. Lazard, D.: Gröbner bases, Gaussian elimination and resolution of systems of algebraic equations. Computer algebra, EUROCAL '83, Proc. Conf., London 1983, Lect. Notes Comput. Sci. 162, 146-156. (1983)
20. Möller, H., Mora, T., Traverso, C.: Gröbner bases computation using syzygies. In: Proceedings of the international symposium on symbolic and algebraic computation, ISSAC'92. Berkeley, CA, USA, July 27–29, pp. 320–328 (1992)
21. Pommaret, J.: Systems of partial differential equations and Lie pseudogroups. With a preface by Andre Lichnerowicz. Mathematics and its Applications. Vol. 14. New York-London-Paris: Gordon and Breach Science Publishers. (1978)

22. Seiler, W.M.: A combinatorial approach to involution and δ-regularity. I: Involutive bases in polynomial algebras of solvable type. Appl. Algebra Eng. Commun. Comput. 20(3-4), 207–259 (2009)
23. Seiler, W.M.: A combinatorial approach to involution and δ-regularity. II: Structure analysis of polynomial modules with Pommaret bases. Appl. Algebra Eng. Commun. Comput. 20(3-4), 261–338 (2009)
24. Seiler, W.M.: Involution. The formal theory of differential equations and its applications in computer algebra. Berlin: Springer (2010)
25. Thomas, J.M.: Differential systems. New York: American Mathematical Society (AMS). IX. 118 p. (1937). (1937)
26. Traverso, C.: Hilbert functions and the Buchberger algorithm. J. Symb. Comput. 22(4), 355–376 (1996)
27. Zharkov, A., Blinkov, Y.: Involution approach to investigating polynomial systems. Math. Comput. Simul. 42(4), 323–332 (1996)