# Minimal Canonical Comprehensive Gröbner Systems*

Montserrat Manubens & Antonio Montes

Departament de Matemàtica Aplicada 2,
Universitat Politècnica de Catalunya, Spain.
e-mail: {montserrat.manubens, antonio.montes}@upc.edu
http://www-ma2.upc.edu/∼montes

May 9, 2007

**Abstract**

This is the continuation of Montes' paper "On the canonical discussion of polynomial systems with parameters". In this paper we define the Minimal Canonical Comprehensive Gröbner System (MCCGS) of a parametric ideal and fix under which hypothesis it exists and is computable. An algorithm to obtain a canonical description of the segments of the MCCGS is given, completing so the whole MCCGS algorithm (implemented in Maple). We show its high utility for applications, like automatic theorem proving and discovering, and compare it with other existing methods. A way to detect a counterexample is outlined, although the high number of tests done give evidence of the existence of the MCCGS.

*Keywords:* comprehensive Gröbner system, canonical, minimal, reduced specification, generalized canonical specification, constructible sets.

*MSC:* 68W30, 13P10, 13F10.

## 1 Introduction

In this paper we continue the task introduced in (Montes 2007). Let us briefly remember the basic features.

Given a parametric polynomial ideal $I \subset K[\overline{a}][\overline{x}]$ in the variables $\overline{x} = (x_1, \dots, x_n)$ and the parameters $\overline{a} = (a_1, \dots, a_m)$, and monomial order $\succ_{\overline{x}}$, our interest is to find the different types of solutions for the different values of the parameters. Let $K$ be a computable field and $\overline{K}$ an algebraically closed extension. A specialization is the homomorphism $\sigma_{\overline{\alpha}} : K[\overline{a}][\overline{x}] \to \overline{K}[\overline{x}]$, that corresponds to the substitution of the parameters by concrete values $\overline{\alpha} \in \overline{K}^m$. A comprehensive Gröbner system (CGS) is a set of pairs:

$$\begin{aligned}\mathrm{CGS}(I, \succ_{\overline{x}}) \;=\; & \{(S_i, B_i)\ : S_i \subseteq \overline{K}^m \text{constructible sets},\ B_i \subset A[\overline{x}], \\ & \sigma_{\overline{\alpha}}(B_i) = \mathrm{gb}(\sigma_{\overline{\alpha}}(I), \succ_{\overline{x}})\ \forall \overline{\alpha} \in S_i\ \text{, and } \textstyle\bigcup_i S_i = \overline{K}^m\},\end{aligned}$$

*This research was partly supported by the Ministerio de Ciencia y Tecnología under project MTM 2006-01267, and by the Generalitat de Catalunya under project 2005 SGR 00692.

where the $S_i$ are called "segments" and the $B_i$ "bases". Frequently the word "segment" is also used for the pair $(B_i, S_i)$ whenever the sense is clear from the context.

There are different known algorithms that provide Comprehensive Gröbner Bases and Systems for a given ideal:

CGB (Weispfenning 1992),

ACGB (Sato-Suzuki 2003; Sato 2005; Nabeshima 2005),

SACGB (Suzuki-Sato 2006),

HSGB (González-Traverso-Zanoni 2005),

BUILDTREE (Montes 2002; Manubens-Montes 2006; Montes 2007).

There are available implementations (Dolzman-Seidl-Sturm 2006) of Weispfenning's CGB algorithm in Reduce, of Suzuki-Sato's SACGB in Risa/Asir and in Maple (Suzuki-Sato 2006) and of Montes's BUILDTREE in Maple[1]. All these algorithms allow to build both Comprehensive Gröbner Bases and Systems, but they are differently oriented. A comparison of the most interesting among them is given in section 6.

In fact, comprehensive Gröbner systems are in general more effective to handle for their use in the applications than comprehensive Gröbner bases. But it is also convenient to require some more additional features to these Gröbner systems when looking for applications.

The first requirement is to have *disjoint* and *reduced* CGS. By disjoint we mean that the $S_i$ form a partition of $\overline{K}^m$, and by reduced that the bases $B_i$ specialize to the reduced Gröbner basis of $\sigma_{\overline{\alpha}}(I)$ preserving the leading power products (lpp), for every value $\overline{\alpha}$ of the parameters inside $S_i$. The algorithm BUILDTREE (introduced in (Montes 2002) as DISPGB and improved in (Manubens-Montes 2006)) already builds a disjoint, reduced CGS.

In (Montes 2007) the interest is focused in the improvement of BUILDTREE to obtain a simpler and canonical CGS. The method consists of grouping together all the segments with the same lpp that allow a same basis specializing well on all the grouped segments. A natural conjecture establishes the existence of an equivalence relation between the segments having the same lpp, and an algorithm is given to compute the basis corresponding to the grouped segments.

In order to obtain a truly canonical CGS we need to describe the segments in a canonical way. This is the objective of the present paper. In (Montes 2007) a canonical description of a segment determined by a diff-specification was already given, but it remained to obtain a canonical representation of the addition of such segments. The objective is thus to obtain the MCCGS (minimal canonical CGS).

**Definition 1.** We call Minimal Canonical CGS (MCCGS) a CGS with the following properties:

i) disjoint CGS, i.e. $S_i \cap S_j = \emptyset$ for $i \neq j$;

[1] The library DPGB 7.0 written in Maple 8 is available at the web http://www-ma2.upc.edu/∼montes, and is actualized with the MCCGS algorithm.

ii) reduced CGS, i.e. the polynomials in $B_i$ have content 1 w.r.t. $\overline{x}$, $B_i$ specializes to the reduced Gröbner basis of $\sigma_{\overline{\alpha}}(I)$ for every $\overline{\alpha} \in S_i$, their leading coefficients are non-null on $S_i$ and their lpp remain stable;

iii) the sets $S_1, \ldots, S_s$ are intrinsic for the given $I$ and $\succ_{\overline{x}}$ and are described in a canonical form.

iv) the number of segments of the CGS with the above properties is minimal.

The currently existing algorithms that can build comprehensive Gröbner systems, say BUILDTREE, CGB, CCGB, ACGB and SACGB, do not hold all these properties. BUILDTREE builds a comprehensive Gröbner system satisfying properties i) and ii). But CGB, ACGB and SACGB do not hold property i), at least. Finally, although the Gröbner system obtained within CCGB is canonically determined, does not hold properties i) nor ii) as for the obtention of a comprehensive Gröbner basis the algorithm needs the Gröbner systems to be faithful.

It must be emphasized that the existence of the MCCGS depends on the Conjecture formulated in (Montes 2007) about the existence of an equivalence relation between segments allowing a common basis.

If the Conjecture is true, then the computation using MCCGS algorithm proposed in (Montes 2007) and in this paper, already depends on the semi-algorithm GENIMAGE given there for computing pre-images, that uses arbitrary bounds.

With these restrictions, MCCGS algorithm builds a comprehensive Gröbner system satisfying all the properties in Definition 1. These properties will make the algorithm more suitable for the applications. In particular, they are very appropriate for automatic theorem proving and discovery (see (Montes-Recio 2007)) as well as to compute geometric loci as shown in example 9.

Furthermore, MCCGS also allows to restrict the parameter space to a constructible set and impose a-priori null and non-null conditions. This is also an interesting tool for applications when we want some degenerate cases to be avoided (see Section 5) or restrictions on the parameters to be given. For example, when the parameters involve angles, and the equations are given using the sine and cosine of the angles as parameters, it is important to restrict the solutions to $\cos^2 \varphi + \sin^2 \varphi - 1 = 0$.

The whole algorithm MCCGS is achieved by three steps:

i) BUILDTREE (described in (Manubens-Montes 2006)),

ii) grouping segments with common basis (described in (Montes 2007)),

iii) representing the subsets in canonical form. This part will be described in sections 3 and 4.

Although the algorithm requires two term orders (one for the variables $\succ_{\overline{x}}$ and another for the parameters $\succ_{\overline{a}}$), the result will not depend on $\succ_{\overline{a}}$, as the segments $(B_i, S_i)$ are intrinsic for the given ideal $I$ and the term order $\succ_{\overline{x}}$. Even though, $\succ_{\overline{a}}$ will be used to determine the reduced Gröbner bases of the ideals involved in the description of $S_i$.

The paper is structured as follows: section 2 is devoted to recalling some properties and results from (Montes 2007) which are used in the subsequent sections. The generalization of the canonical specification and its properties are given in Section 3. In Section 4 we give the algorithm which collects the corresponding segments into a generalized canonical specification and builds up the Minimal Canonical Comprehensive Gröbner System (MCCGS). In section 5 a practical application to automatic theorem proving is given. Finally, in section 6 we compare the main available CGS algorithms.

## 2 Preliminaries

We describe now briefly steps i) and ii) of the MCCGS before tackling the last step iii) that will be studied in this paper. The algorithm starts with a parametric ideal $I$ and a term-order $\succ_{\overline{x}}$ on the variables. An auxiliary term-order $\succ_{\overline{a}}$ over the parameters is needed to describe the subsets in $\overline{K}^m$ using Gröbner bases. It does not affect the segments themselves but only their description.

Step i) is performed by BUILDTREE algorithm, and was described for the first time in (Montes 2002) and improved in (Manubens-Montes 2006). The output is a disjoint reduced CGS, where the subsets $S_i$ are determined by *red-specifications*. A red-specification of a segment $S$ is described by the pair $(N, W)$, where $N$ is the radical null-conditions ideal, and $W$ is a set of irreducible (prime) polynomials on $K[\overline{a}]$ representing non-null conditions such that no prime component $N_i$ of the prime decomposition of $N$ does contain any of the polynomials in $W$. We have $S = \mathbb{V}(N) \setminus \mathbb{V}(h)$ with $h = \prod_{w \in W} w$. A red-specification determined by $(N, W)$ is easily transformed into a *diff-specification* $(N, M)$ with $N \subset M$ where $S = \mathbb{V}(N) \setminus \mathbb{V}(M)$, by considering the polynomial $h = \prod_{w \in W} w$ and taking $M = \langle h \rangle + N$.

Let us denote $\mathrm{CGS}_1$ the output of BUILDTREE that consists of a list of segments each represented by the three objects $(B_i, N_i, W_i)$. Remember that each of these segments have characteristic set of lpp of their bases $B_i$ that are preserved by specialization on $S_i$. We say that a basis $G$ *specializes well* to $(B, N, W)$, with $\mathrm{lpp}(G) = \mathrm{lpp}(B)$, if the polynomials of $\overline{G}^N$ are proportional to the polynomials of $B$, i.e. for each $g \in G$ there exist $f \in B$ and $\alpha, \beta \in W^*$ such that $\alpha\, \overline{g}^N = \beta\, f$, where $W^* = \{k \prod_{i=1}^{s} w_i^{\lambda_i} : k \in K, \lambda_i \in \mathbb{Z}_{\geq 0}, w_i \in W\}$.

Step ii), described in (Montes 2007), selects the segments of $\mathrm{CGS}_1$ with the same lpp that admit a common reduced basis specializing well to the reduced Gröbner basis for every specialization in the grouped segments. If Conjecture 7 in (Montes 2007) is true, the grouped segments form an intrinsic partition of the parameter space. To perform that task the algorithms DECIDE and GENIMAGE are used. The first one tests whether one from two segments with the same lpp has already a generic basis specializing to the other (this is the most frequent case) or a sheaf exists and is necessary or whether possibly a more generic basis must be found (by GENIMAGE). Whenever no pre-image nor sheaf is found then both segments are not equivalent and cannot be summarized. It can happen that instead of simple polynomials the basis $B_i$ contains also sheaves of polynomials. A sheaf $\{g_1, \ldots, g_k\}$ is accepted in a basis of a segment instead of a simple polynomial, whenever all the polynomials in the sheaf specialize to the corresponding polynomial of the reduced Gröbner basis of the specialized ideal or to 0, and some of the polynomials in the sheaf specialize to non-zero for every $\overline{\alpha} \in S_i$. As was shown in (Wibmer 2006), it is necessary to use sheaves for some over-determined systems if we want to group all the segments admitting a common basis with the same lpp. We must notice that DECIDE algorithm also depends on the semi-algorithm GENIMAGE to determine a polynomial $\tilde{f}$ that specializes well to $f_1$ over $(N_1, W_1)$ and to $f_2$ over $(N_2, W_2)$. Thus the canonicity of the results of the computation of a MCCGS relies on GENIMAGE and the truthfulness of the mentioned conjecture.

Let us denote the output of the second step $\mathrm{CGS}_2$. It will be described by segments with a common basis $B_i$ and a set of red-specifications:

$$(B_i, \{(N_{i1}, W_{i1}), \ldots, (N_{ij_i}, W_{ij_i})\}). \tag{1}$$

$S_i$ will now be the union of the segments determined by the red-specifications $(N_{ik}, W_{ik})$ for $k$ from 1 to $j_i$ .

Step iii) will be described in next sections. Its objective is to give a canonical description of the union of the grouped segments of step ii). In (Montes 2007) it was shown how a diff-specification can be transformed into a *can-specification*. Here we will prove that the union of red-specifications or their corresponding diff-specifications can be transformed into a generalized can-specification using what we call a $P$-tree. The idea is based on Theorem 12 in (Montes 2007). Let us give here a slightly different formulation of it, more appropriate for the current purposes.

**Theorem 2.**

i) *Every diff-specification* $S = \mathbb{V}(N) \setminus \mathbb{V}(M)$ *admits a unique can-specification*

$$S = \mathbb{V}(N) \setminus \mathbb{V}(M) = \bigcup_i \left(\mathbb{V}(N_i) \setminus (\cup_j \mathbb{V}(M_{ij})\right), \tag{2}$$

*where* $\mathcal{N} = \cap_i N_i$ *and* $\mathcal{M}_i = \cap_j M_{ij}$ *are the irredundant prime decompositions over* $A$ *of the radical ideals* $\mathcal{N}$ *and* $\mathcal{M}_i$ *respectively, where* $N_i \subsetneq M_{ij}$.

ii) *The Zariski closure over* $\overline{K}^m$ *verifies*

$$\overline{S} = \overline{\bigcup_i (\mathbb{V}(N_i) \setminus (\cup_j \mathbb{V}(M_{ij}))} = \bigcup_i \mathbb{V}(N_i) = \mathbb{V}(\mathcal{N}).$$

iii) *The can-specification verifies*

$$\mathbb{V}(N_i) \setminus (\cup_j \mathbb{V}(M_{ij})) = S \cap \mathbb{V}(N_i).$$

iv) *Given a diff-specification of* $S$ *the algorithm* DIFFTOCANSPEC (Montes 2007) *builds its can-specification.*

The need of having a canonical description of the intrinsic segments comes from the need of comparing different outputs for the same problem, and also to have a final simple description of the segments.

## 3 Adding segments

We tackle now the third step of MCCGS, i.e. the description of the union of the segments in a canonical form. We start with segments of the form (1). The red-specifications $(N, W)$ can be transformed into diff-specifications $(N, M)$, as explained in Section 2, so we are attained with the obtention of a canonical representation for the addition of diff-specifications. We cannot assume that the simple form given by formula (2) will be sufficient. A more complex constructible set will be formed grouping all the segments $S_{ik}$ for $1 \leq k \leq j_i$.

Thus we generalize the concept of canonical specification given in (Montes 2007):

**Definition 3** (P-tree)**.** A P-tree is a rooted directed tree such that

i) the nodes are prime ideals over $A$ except the root, denoted $r$,

ii) when $P \to Q$ is an arc then $P \subsetneq Q$,

iii) the children of a node are a set of irredundant prime ideals over A, (whose intersection form a radical ideal).

By definition the root level is 0.

**Definition 4** (C-tree)**.** To any P-tree we associate an isomorphic C-tree by changing every node $P$ to a subset of $\overline{K}^m$ denoted $C(P)$ by the following recursive procedure:

i) if $P$ is a leaf (terminal vertex) then $C(P) = \mathbb{V}(P)$,

ii) if $P$ is an inner node different from the root and $P_1, \dots, P_d$ are its children, then

$$C(P) = \mathbb{V}(P) \setminus (C(P_1) \cup \cdots \cup C(P_d)) \tag{3}$$

iii) if $P_1, \dots, P_d$ are the children of the root vertex $r$ then

$$C(r) = C(P_1) \cup \cdots \cup C(P_d).$$

Note that for $C(r)$ the parity of the vertex-level acts additively for odd level vertices and as a subtraction for even level vertices. (See example 6 below).

**Definition 5** (Generalized canonical specification)**.** A generalized canonical specification (GCS) of a set $S$ is a P-tree such that $S = C(r)$ satisfying, for every node $P$ at level $j$, the following condition:

$$C(P) = \mathbb{V}(P) \cap B \tag{4}$$

where $B = S$ for $j$ odd and $B = \overline{K}^m \setminus S$ for $j$ even.

**Example 6.** To clarify the definition suppose that we want to describe the set $S_1$ of the $\mathbb{R}^3$-space with coordinates $a, b, c$ consisting of the planes $a = 0$ and $b = -1$ except the lines $a = b = 0$ and $a = c = 0$ plus the point $O(0, 0, 0)$. We can express $S_1$ as

$$S_1 = ((\mathbb{V}(a) \cup \mathbb{V}(b+1)) \setminus (\mathbb{V}(a, b) \cup \mathbb{V}(a, c))) \cup \mathbb{V}(a, b, c)$$

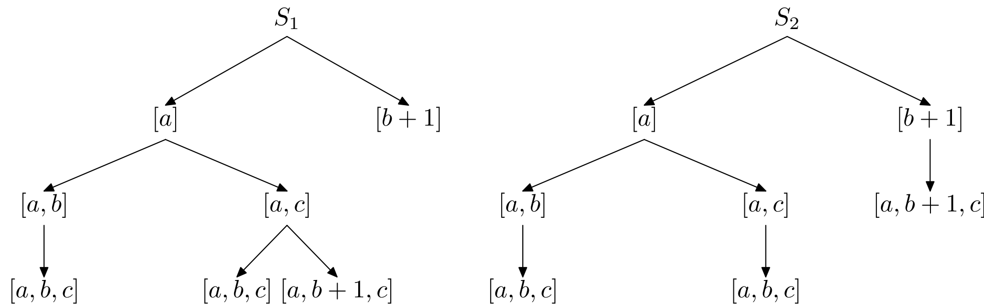


Figure 1: Trees representing the sets $S_1$ and $S_2$ in generalized can-specification.

But there exist many other possible determinations of this set. If we want to obtain the GCS of $S_1$ we must write $S_1$ in the form

$$S_1 = (\mathbb{V}(a) \setminus ((\mathbb{V}(a,b) \setminus \mathbb{V}(a,b,c)) \cup (\mathbb{V}(a,c) \setminus (\mathbb{V}(a,b,c) \cup \mathbb{V}(a,b+1,c))))) \cup \mathbb{V}(b+1)$$

This formula can be represented by the tree associated to $S_1$ shown in figure 1. Notice that we must include the point $\mathbb{V}(a,b+1,c)$ under the branch of $\mathbb{V}(a,c)$, as this point belongs to $S_1$ and condition (4) requires it to belong also to $C(a)$. The interest of that representation lies in the fact that it is unique as we prove in Theorem 7 below.

Consider now the set $S_2 = S_1 \setminus \mathbb{V}(a,b+1,c)$. In order to preserve property (4) of the GCS definition, the P-tree associated to $S_2$ will be modified from the P-tree associated to $S_1$ by eliminating the point under the variety $\mathbb{V}(a,c)$ and setting it under the variety $V(b+1)$. The new tree is also shown in Figure 1. These examples should clarify the definition of GCS to obtain canonicity of the description by preserving condition (4).

**Theorem 7.** *A subset $S \subset \overline{K}^m$ defined by a* GCS *has the following properties:*

i) *For every vertex $P$, except for the root,*

$$\overline{C(P)} = \mathbb{V}(P)$$

*where, as usual, the Zariski closure is taken over $\overline{K}^m$.*

ii) *For the root vertex $r$*

$$\overline{S} = \overline{C(r)} = \bigcup_{i=1}^{d} \mathbb{V}(P_i)$$

*where the $P_i$'s are the children vertices of $r$.*

iii) *$S$ has a unique* GCS *decomposition.*

*Proof.* i) The inclusion $\subseteq$ is obvious as $C(P) \subseteq \mathbb{V}(P)$. To prove the equality we have

$$C(P) = \mathbb{V}(P) \setminus \bigcup_{i=1}^{d} C(P_i) \supseteq \mathbb{V}(P) \setminus \bigcup_{i=1}^{d} \mathbb{V}(P_i).$$

Consider the closure of the above formula and apply Theorem 2 (ii). The result follows.

ii) Is an immediate consequence of i).

iii) To prove the uniqueness we proceed by induction on $d$. For $d = 1$, the tree is formed by the root $r$ and a set of children nodes forming an irredundant prime decomposition of the radical ideal defining $S$, by Definition 3 iii). Thus, in this segment the P-tree is unique.

Assume now by induction hypothesis the uniqueness of the GCS for every P-tree of maximum depth less than $d$ and let us prove, as a consequence, the uniqueness also for depth $d$. Let $S$ be defined by a P-tree of maximal depth $d$ representing a GCS. By part (ii) of the Theorem we have

$$\overline{S} = \bigcup_i \mathbb{V}(P_i) = \mathbb{V}(\cap_i P_i),$$

where the $P_i$'s form the unique irredundant prime decomposition over $A$ of the radical ideal $\cap_i P_i$ defining $\overline{S}$ by Definition 3 (iii). Thus they are uniquely determined. Denoting $P_{ij}$ the children of $P_i$, by (4), we have

$$C(P_i) = \mathbb{V}(P_i) \setminus \bigcup_{j=1}^{d_i} C(P_{ij}) = \mathbb{V}(P_i) \cap S \tag{5}$$

showing that $C(P_i)$ is also uniquely determined. Set $S_i$ for the subtracting set

$$S_i = \bigcup_{j=1}^{d_i} C(P_{ij}). \tag{6}$$

As $S_i \subseteq \mathbb{V}(P_i)$, $S_i$ is also uniquely defined by (5). By Definition 5, formula (4), we have

$$C(P_{ij}) = \mathbb{V}(P_{ij}) \cap \left(\overline{K}^m \setminus S\right),$$

Thus

$$S_i = \bigcup_{j=1}^{d_i} C(P_{ij}) = \left(\bigcup_{j=1}^{d_i} \mathbb{V}(P_{ij})\right) \cap \left(\overline{K}^m \setminus S\right)$$

and so

$$C(P_{ij}) = \mathbb{V}(P_{ij}) \cap S_i \tag{7}$$

By the ascending chain condition for the ideals in the branches and condition (4) for the P-tree of $S$, equation (7) ensures that condition (4) is also respected for the subtree of $S_i$, whose root vertex is given by (6). Thus the subtree of $S_i$ also forms a GCS of $S_i$ with depth less than $d$. By the induction hypothesis it is uniquely determined and so does the complete P-tree of $S$.

□

## 4 The MCCGS algorithm

Given an ideal $I$ and the monomial orders $\succ_{\overline{x}}$ for the variables and $\succ_{\overline{a}}$ for the parameters, the following sequence of algorithms build up the $P$-tree $T$ corresponding to the Minimal Canonical Comprehensive Gröbner System associated to $I$ and $\succ_{\overline{x}}$. We describe them in descendent design.

---

tree $T \leftarrow$ **MCCGS**$(B, \succ_{\overline{x}}, \succ_{\overline{a}})$
*Input:* $B$ a basis of the parametric polynomial ideal $I$ and monomial orders $\succ_{\overline{x}}, \succ_{\overline{a}}$.
*Output:* $T$ a tree containing the minimal canonical comprehensive Gröbner system associated to $I$.

$T_0$ :=**BUILDTREE**$(B, \succ_{\overline{x}}, \succ_{\overline{a}})$
$S$ :=**SELECTCASES**$(T_0)$
$T$ :=**GENCANTREE**$(S)$

---

MCCGS uses BUILDTREE (see (Montes 2002; Manubens-Montes 2006)) to build up the discussion tree $T_0$ containing a CGS whose segments are expressed as red-specifications. Then SELECTCASES takes $T_0$ as input and classifies the segments from the CGS associated to $T_0$ into pairs of the form $(B_i, l_i)$, where $l_i$ is a set of red-specifications $\{(N_{i1}, W_{i1}), \ldots (N_{ij_i}, W_{ij_i})\}$ whose corresponding bases have been generalized by the same basis $B_i$. Afterwards, MCCGS calls the new algorithm GENCANTREE to finally obtain the MCCGS associated to the initial ideal and term order.

GENCANTREE uses GCS algorithm to build the $P$-tree corresponding to the generalized canonical specification of the addition of segments. GCS algorithm begins by setting the ideal $\{0\}$ at the root of new tree $\overline{T}$ and calls iteratively the recursive algorithm ADDCASE. It must be noted that there are two kinds of nodes, namely odd level vertices and even level vertices, that are treated differently by ADDCASE. ADDCASE uses two auxiliary algorithms: DIFFTOCANTREE (a minor transformation of DIFFTOCANSPEC) converts a diff-specification into a $P$-tree containing the associated can-specification, and SIMPLIFYSONS just makes the suitable simplifications.

At the first iteration ADDCASE stores under root the $P$-tree of the unique canonical specification associated to $(N_{i1}, W_{i1})$. Then, to add each further red-specification $(N_{ik}, W_{ik})$, ADDCASE executes itself recurrently in post-order at the even level vertices $u \in \overline{T}$ and adds the can-specification associated to $(N_{ik}, W_{ik})$ contained in $\mathbb{V}(P_u)$. For example, in figure 2 it would act successively on the vertices

$$c,\ f,\ i,\ j,\ \ell,\ m,\ g,\ o,\ p,\ d,\ t,\ u,\ r,\ v,\ a.$$

Thus, before acting on an even vertex $u \in \overline{T}$, the algorithm must have acted on all its even descendants. Therefore, if an even level descendant $v$ verifies that $N_{ik} \supseteq P_v$, then the can-specification associated to $(N_{ik}, W_{ik})$ must have been completely hung under $v$. In this case the *test* variable will contain *false* and thus DIFFTOCANTREE for current $(N_{ik}, W_{ik})$ will not act on $P_u$ nor on any of its ascendant vertices. We must also remind that the

set of pairs $S \leftarrow$ **SELECTCASES**$(T_0)$
*Input:* $T_0$ a BUILDTREE discussion tree whose terminal vertices shape a CGS with red-specifications.
*Output:* $S$ a finite set of pairs of the form $(B_i, \{(N_{i1}, W_{i1}), \ldots, (N_{ij_i}, W_{ij_i})\})$ taken from the CGS associated to $T_0$.

$G := \{(B_1, N_1, W_1), \ldots, (B_r, N_r, W_r)\}$ {the CGS associated to $T_0$}
$S := \emptyset$
**while** $G \neq \emptyset$ **do**
  Let $(B, N, W)$ be the first element of $G$
  $B_0 := B;\ N_0 := N;\ W_0 := W;$
  $l := \{(N_0, W_0)\}$
  $G := G \setminus \{(B_0, N_0, W_0)\}$
  **for all** $(B', N', W') \in G$ such that $\text{lpp}(B) = \text{lpp}(B')$ **do**
    $p := 0$
    **for all** $f \in B$ **while** $p \neq$ **false do**
      Let $f' \in B'$ be such that $\text{lpp}(f) = \text{lpp}(f')$
      $p :=$ **DECIDE**$(f, N, W, f', N', W')$
      **if** $p \neq$ **false then**
        Substitute $f$ by $p$ in $B_0$
      **end if**
    **end for**
    **if** $p \neq$ **false then**
      $l := l \cup \{(N', W')\}$
      $B := B_0;\ N := N \cap N';\ W := W \cap W';$
    **end if**
  **end for**
  $S := S \cup \{(B, l)\}$
  $G := G \setminus \{(B', N', W') \in G$ such that $(N', W') \in l\}$
**end while**

ideals associated to the paths in $\overline{T}$ starting from root form ascending chains of prime ideals. Thus, whenever *test* is *false*, the condition cited above will also hold for all vertices placed between $u$ and $v$, even the odd level ones, i.e. for all $w \in \overline{T}$ descendent of $u$ and ascendant of $v$, $N_{ik} \supseteq P_w$.

This way, ADDCASE completes current P-tree $\overline{T}$ to a new tree such that for every odd level vertex $u$ with prime ideal $P_u$, all points in $\mathbb{V}(P_u) \cap (\mathbb{V}(N_{ik}) \setminus \mathbb{V}(h_{ik}))$ (where $h_{ik} = \prod_{w \in W_{ik}} w$) are in $C(P_u)$, as required.

Nevertheless in the new tree completed by ADDCASE it could happen that $P_u + N_{ik} = P_u$ for some even level vertex $u$, which would cause that $P_u$ and its unique child $P_{\text{child}(u)}$ coincide. If so, SIMPLIFYSONS takes the subtree under child$(u)$, slips it upwards hanging it from parent$(u)$ and eliminates both vertices $u$ and child$(u)$ from the tree. When this action is performed, it could also happen that some set of current even level siblings do not preserve the prime decomposition irredundancy property, as some lifted primes can contain

tree $T \leftarrow$ **GENCANTREE**$(S)$
*Input:* $S$ a finite set of pairs of the form $(B_i, \{(N_{i1}, W_{i1}), \dots, (N_{ij_i}, W_{ij_i})\})$.
*Output:* the canonical tree $T$ associated to $S$.

initialize $T$
**for** $1 \leq i \leq \sharp S$ **do**
  Create $u_i$ a new vertex in $T$ hanging from root
  store $B_i$ in $u_i$
  $l := \{(N_{i1}, W_{i1}), \dots, (N_{ij_i}, W_{ij_i})\}$ {red-specifications associated to $B_i$}
  $\overline{T}$ :=**GCS**$(l)$
  hang $\overline{T}$ from $u_i$
**end for**

tree $\overline{T} \leftarrow$ **GCS**$(l)$
*Input:* $l$ a finite set of red-specifications
*Output:* a tree containing the Generalized Can-Specification associated to the addition of segments in $l$.

initialize tree $\overline{T}$ with root $r$
set $P_r := \phi$
**for all** pairs $(N, W) \in l$ **do**
  $\overline{T}$ :=**ADDCASE**$((N, W), r, \overline{T})$
**end for**

some of their sibling vertices, i.e. $\exists v_1, v_2 \in \text{children}(u)$ such that $P_{v_1} \subseteq P_{v_2}$ for $u$ an even level vertex in $\overline{T}$. SIMPLIFYSONS algorithm also detects these cases and eliminates the subtrees hanging from $v_2$ as well as $v_2$. Though, the action of SIMPLIFYSONS will restore the GCS-condition property of the tree.

Note: For algorithmic reasons, all paths starting from root vertex in a $P$-tree will be of even length. Thus for odd length branches, the algorithm will add a new vertex [1] at the end.

The above described algorithms build the complete MCCGS of the initial ideal. The following theorem states that GCS algorithm builds the generalized can-specification (GCS) associated to the set of the corresponding diff-specifications:

**Theorem 8.** *Given a finite list of pairs $l = \{(N_{ik}, W_{ik}) : k = 1, \dots, M\}$ of red-specifications,* GCS$(l)$ *computes the $P$-tree associated to the generalized can-specification determining the constructible set*

$$\bigcup_{k=1}^{M} \mathbb{V}(N_{ik}) \setminus \mathbb{V}(\prod_{w \in W_{ik}} w).$$

*Proof.* Let $S = \bigcup_{k=1}^{M} \mathbb{V}(N_{ik}) \setminus \mathbb{V}(\prod_{w \in W_{ik}} w)$. The proof is done by induction on $M$, the number of red-specifications to be added.

For $M = 1$, GCS uses DIFFTOCANTREE just once and, by Theorem 1 (iv), it builds

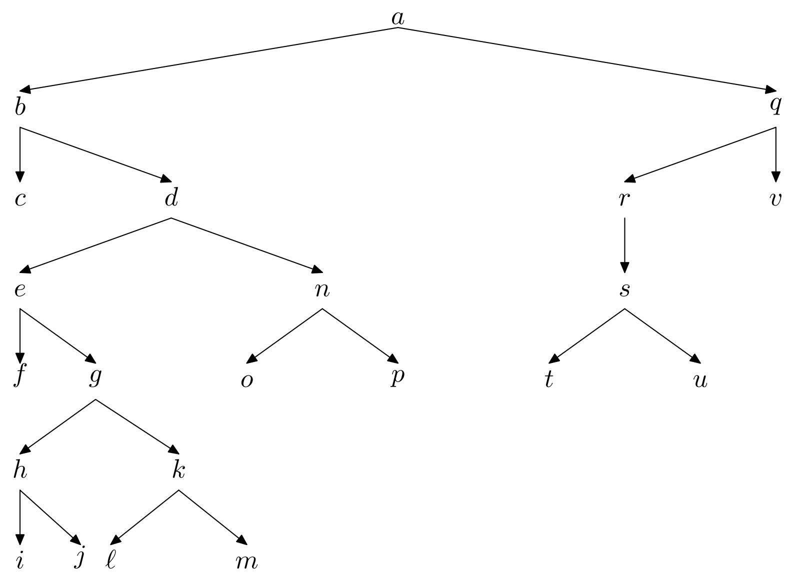


Figure 2: The action of ADDCASE.

up the unique can-specification in tree $\overline{T}$. Thus $\overline{T}$ is a $P$-tree such that $C(\overline{T}) = \mathbb{V}(N_{i1}) \setminus \mathbb{V}(\prod_{w\in W_{i1}} w)$.

By induction hypothesis, assume now that after the $M-1$ iteration of ADDCASE the GCS tree of the $M-1$ red-specifications has been built and let $\tilde{T}$ be this tree, which is a $P$-tree such that $C(\tilde{T}) = \bigcup_{k=1}^{M-1} \mathbb{V}(N_{ik}) \setminus \mathbb{V}(\prod_{w\in W_{ik}} w)$ and such that every vertex $u \in \tilde{T}$ holds that $C(P_u) = \mathbb{V}(P_u) \cap C(\tilde{T})$. We shall prove that the $M$-th iteration will build the GCS tree of $S$.

Let us describe how the recursive ADDCASE algorithm acts on $\tilde{T}$ adding $\mathbb{V}(N_{iM}) \setminus \mathbb{V}(\prod_{w\in W_{iM}} w)$. Denote by $\Lambda(u)$ the operation on an even level vertex $u$ that hangs to it the tree associated to the can-specification of $(N_{iM}, W_{iM})$ contained in $\mathbb{V}(P_u)$ (i.e. $\mathbb{V}(N_{iM}+P_u)\setminus \mathbb{V}(N_{iM} + P_u + \langle\prod_{w\in W_{iM}} w\rangle)$ whenever it can be hung and returns $false$ or $true$ depending on whether parent$(P) \subseteq N_{iM}$ or not, respectively. So it hangs the points $\mathbb{V}(P_u)\cap(\mathbb{V}(N_{iM})\setminus \mathbb{V}(N_{iM} + \langle\prod_{w\in W_{iM}} w\rangle))$, and thus $C(P_u) = \mathbb{V}(P_u) \cap S$.

$\Lambda(u)$ is applied recursively in post-order. If $\Lambda(u)$ returns $false$ at some even level vertex $u$, the whole set $\mathbb{V}(N_{iM}) \setminus \mathbb{V}(N_{iM} + \langle\prod_{w\in W_{iM}} w\rangle)$ has been hung under $u$ and thus, as $u$ is even, $C(\text{father}(u)) \supset \mathbb{V}(N_{iM}) \setminus \mathbb{V}(N_{iM} + \langle\prod_{w\in W_{iM}} w\rangle)$. Then $\Lambda$ will not be applied to any of its ascendant vertices because $C(\overline{T}) = C(\tilde{T}) \cup \Big(\mathbb{V}(N_{iM}) \setminus \mathbb{V}(N_{iM} + \langle\prod_{w\in W_{iM}} w\rangle)\Big)$.

If $\Lambda(u)$ returns $true$ for all $u \in \tilde{T}$, which means that $\mathbb{V}(N_{iM})\setminus\mathbb{V}(N_{iM}+\langle\prod_{w\in W_{iM}} w\rangle)$ has not completely been hung under root, then the $P$-tree corresponding to the red-specification $(N_{iM}, W_{iM})$ computed by DIFFTOCANTREE will be hung from root. Thus, we finally have that $C(\overline{T}) = C(\tilde{T}) \cup \Big(\mathbb{V}(N_{iM}) \setminus \mathbb{V}(N_{iM} + \langle\prod_{w\in W_{iM}} w\rangle)\Big)$.

This way, GCS algorithm obtains, as SIMPLIFYSONS ensures, a $P$-tree $\overline{T}$ such that for every node $v \in \overline{T}$ holds that $C(P_v) = \mathbb{V}(P_v) \cap C(\overline{T})$ and $C(\overline{T}) = S$. □

(bool, tree $\overline{T}$) $\leftarrow$ **ADDCASE**$((N,W),u,\overline{T})$
*Input:* $(N,W)$ a red-specification , $u$ the current vertex in $P$-tree $\overline{T}$.
*Output:* $false$ if $(N,W)$ is not to be added to parent vertices, $true$ otherwise. It also returns current tree $\overline{T}$.

```
if u is not terminal then
   test := treu
   for all v ∈ children(u) do
      for all w ∈ children(v) do
         if ADDCASE((N,W), w) = false then
            test := false
         end if
      end for
      T̄ := SIMPLIFYSONS(v, T̄)
   end for
else
   test := true
end if
if test = true then
   h := ∏_{w∈W} w
   (R,S) := (N + P_u, N + ⟨h⟩ + P_u) {diff-specification associated to (N,W) in V(P_u) }
   t := DIFFTOCANTREE(R,S)
   hang t from u
   if parent(u) exists and P_parent(u) ⊆ R then
      test := false
   end if
end if
```

Furthermore, GENCANTREE algorithm performs a GCS computation for each list of segments whose associated reduced Gröbner bases specialize properly, obtaining a tree for which the subtrees hanging from the root correspond to the generalized can-specifications of the lists configuring a partition of the parameter space. Thus, MCCGS algorithm performs the discussion and obtains the Minimal Canonical Comprehensive Gröbner System stored in the output tree $T$.

**Example 9.** [Singular points of a conic] The general equation of a conic can be reduced by a suitable change of variables to the form

$$f \equiv x^2 + by^2 + 2cxy + dx = 0.$$

To study its singular points consider the system of equations

$$S := \left[ f, \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right],$$

and apply MCCGS algorithm to $S$ using $\mathrm{lex}(x,y)$ and $\mathrm{lex}(b,c,d)$ for variables and parameters respectively. The result is shown in Figure 3. The interpretation of the output tree is

tree $\overline{T} \leftarrow$ **SIMPLIFYSONS**$(v, \overline{T})$
*Input:* $v$ a vertex at odd level of tree $\overline{T}$ where to start the simplifications.
*Input:* The tree after simplifications

*Description*:
SIMPLIFYSONS just simplifies the subtree under $v$ on the global $\overline{T}$ in order to not having cancellations nor inclusions between the children of $v$. Let $P$ be the prime stored in vertex $v$. The simplification is performed as follows:
Check that there is no $P_i$ child of $P$ such that $P_i = P_{ij}$. And if any, hang to $P$ all subtrees descendant from $P_{ij}$ and drop both $P_i$ and $P_{ij}$ from $\overline{T}$.
Then check whether there is any pair of children of $P$, $P \to P_i$, $P \to P_j$, such that $P_i \subseteq P_j$. If so, drop subtree hanging from $P_j$ and also vertex $P_j$.

```
for all v ∈ children(u) do
  if P_v = P_child(v) then
    hang from u all subtrees under child(v)
    drop v and child(v) from T̄
  end if
end for
if there ∃ v, w ∈children(u) such that P_v ⊆ P_w then
  drop subtree with root w from T̄
end if
```

the following.

There are three different segments: The generic case with lpp set [1] where the conic has no singular points, the segment with lpp set $[y, x]$ corresponding to a single singular point in the conic, and the segment with lpp set $[x]$ corresponding to a solution with one degree of freedom, where the conic is a double line. The conditions over the parameters given by the trees are to be interpreted in the following way:

| lpp | Basis | Description |
|---|---|---|
| $[1]$ | $[1]$ | $\mathbb{C}^3 \setminus ((\mathbb{V}(b) \setminus (\mathbb{V}(c,b) \setminus \mathbb{V}(d,c,b))) \cup \mathbb{V}(d))$ |
| $[y,x]$ | $[2cy+d,x]$ | $(\mathbb{V}(b) \setminus \mathbb{V}(c,b)) \cup \left(\mathbb{V}(d) \setminus \mathbb{V}(d,b-c^2)\right)$ |
| $[x]$ | $[x+cy]$ | $\mathbb{V}(d,b-c^2)$ |

Figure 4 shows the geometrical description of the partition of the parameter space provided by the three segments. The generic segment occurs in the whole 3-dimensional space except the two planes $\mathbb{V}(c)$ and $\mathbb{V}(d)$ plus the line $\mathbb{V}(c,b)$ except the point $(0,0,0)$. The one-singular point segment occurs in the two planes $\mathbb{V}(c)$ and $\mathbb{V}(d)$ except both the line $\mathbb{V}(c,b)$ and the parabola $\mathbb{V}(d,b-c^2)$. Finally the double line occurs on the parabola $\mathbb{V}(d,b-c^2)$.

tree $t \leftarrow$ **DIFFTOCANTREE**$((I, J))$
*Input:* $(I, J)$ a diff-specification.
*Output:* a tree structure containing the Can-Specification of $\mathbb{V}(I) \setminus \mathbb{V}(J)$.

initialize local tree $t$
$\{P_i\}$ :=**PRIMEDECOMP**$(I)$
**for all** $P_i$ **do**
  **if** $P_i \neq \sqrt{J + P_i}$ **then**
    store the $P_i$ as the children of root in $t$
    $\{P_{ij}\}$ :=**PRIMEDECOMP**$(J + P_i)$
    store the $P_{ij}$ as the children of $P_i$ in $t$
  **end if**
**end for**

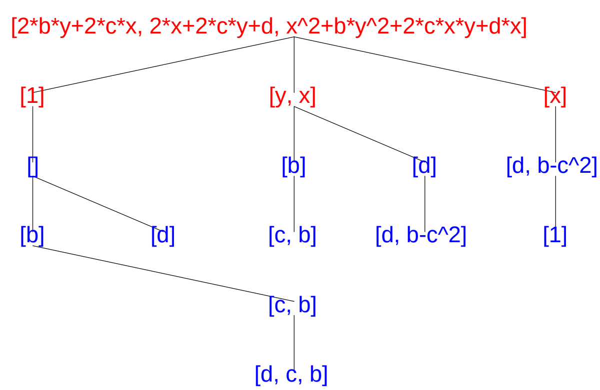


Figure 3: MCCGS for the singular points of a conic system.

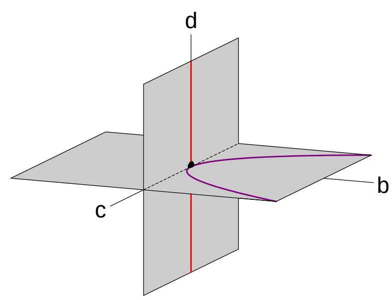


Figure 4: Geometrical description of the MCCGS for the singular points of a conic system.

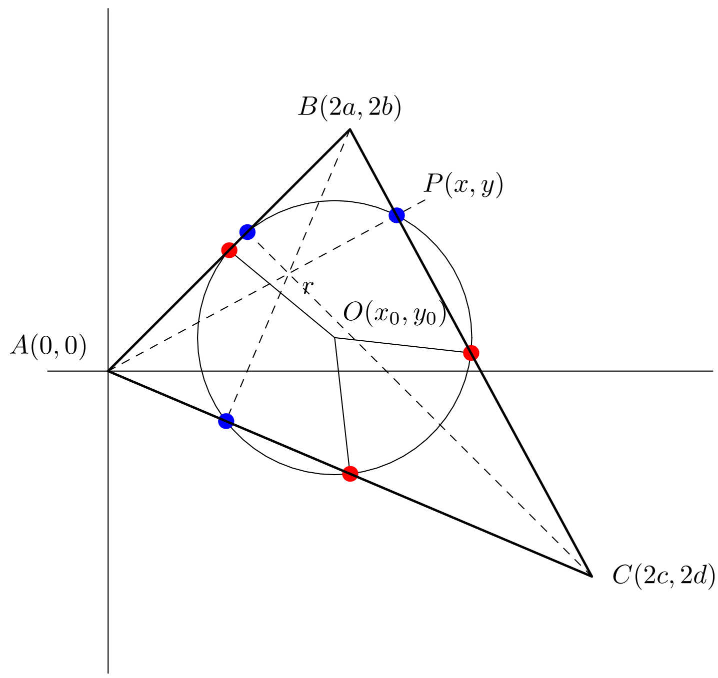


Figure 5: Nine points circle Theorem.

# 5 Applications

We use now the algorithm to prove part of the 9 points circle Theorem on a triangle. It states: *For every triangle, the circle through the three middle points of the sides is also incident with the height feet.* To prove it, and also to obtain supplementary hypotheses if needed, consider a triangle with vertices at the points $A(0,0)$, $B(2a, 2b)$ and $C(2c, 2d)$ and denote $P(x, y)$ the height foot from $A$ (see Figure 5). The first set of hypotheses are the equations of the side $BC$ and the height from $A$ defining the point $P(x, y)$:

$$\begin{aligned} h_1: &\quad (b-d)x + (c-a)y + 2ad - 2bc = 0 \\ h_2: &\quad (c-a)x + (b-d)y = 0 \end{aligned}$$

Denote $r$ and $(x_0, y_0)$ the radius and the center of the circle through the three middle points $(a, b)$, $(c, d)$ and $(a + c, b + d)$. Its equation will be $(x - x_0)^2 + (y - y_0)^2 - r^2 = 0$. So we have the three new hypotheses:

$$\begin{aligned} h_3: &\quad (a - x_0)^2 + (b - y_0)^2 - r^2 = 0 \\ h_4: &\quad (c - x_0)^2 + (d - y_0)^2 - r^2 = 0 \\ h_5: &\quad (a + c - x_0)^2 + (b + d - y_0)^2 - r^2 = 0 \end{aligned}$$

The thesis of the theorem is that the circle is incident with the point $P(x, y)$, thus that the polynomial

$$f = (x - x_0)^2 + (y - y_0)^2 - r^2$$

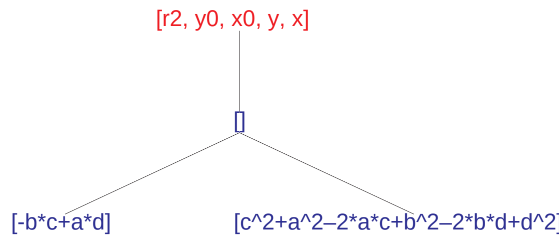


Figure 6: Generic case for $HT$ in the nine points circle Theorem.

is zero as a consequence of the hypotheses. The first to do is searching for the solutions of the system $HT = \langle h_1, h_2, h_3, h_4, h_5, f \rangle$. Thus we call

$$\text{mccgs}(HT, \text{grevlex}(x, y, x_0, y_0, r_2), \text{lex}(a, b, c, d)),$$

where we set $r_2 = r^2$. We obtain a canonical tree with nine cases. But only two cases are really interesting. The first one is the generic case (see Figure 6) for which the lpp are $[r_2, y_0, x_0, y, x]$ showing that for parameter values not in $\mathbb{V}(ad - bc) \cup \mathbb{V}((a-c)^2 + (b-d)^2)$ there exists a unique solution. For the real case it is sufficient to consider $ad - bc \neq 0$, as the real part of the second variety is inside the first one. The second interesting case is the case with basis [1] where no solution exists. The corresponding tree shows that it covers both varieties $\mathbb{V}(ad - bc) \cup \mathbb{V}((a-c)^2 + (b-d)^2)$ except for very special cases corresponding to degenerate triangles. Thus we have proved that the theorem is true whenever $ad - cb \neq 0$. We can also go further and ask if the thesis is a real consequence of the hypotheses, i.e. if $f$ belongs to the radical of the hypotheses ideal $H = \langle h_1, h_2, h_3, h_4, h_5 \rangle$ whenever $ad - bc \neq 0$. To test this we must have

$$HT_1 = \langle h_1, h_2, h_3, h_4, h_5, 1 - wf \rangle = \langle 1 \rangle$$

i.e. the Gröbner basis of $HT_1$ is [1]. We call now

$$\begin{aligned} \text{mccgs} \quad & ([h_1, h_2, h_3, h_4, h_5, 1 - wf], \text{grevlex}(w, x, y, x_0, y_0, r), \\ & \text{lex}(a, b, c, d), notnull = \{ad - bc\}), \end{aligned}$$

and the result is a unique case with basis [1]. Thus effectively $f$ belongs always to the ideal of the hypothesis whenever $ad - bc \neq 0$.

## 6 Comparison of algorithms

The CGS of a parametric ideal $I$ can have very different properties as commented in section 1. For example

i) the subsets $S_i$ of the parameter space $\overline{K}^m$ in which the CGS are divided can be very different, they can contain different number of segments, they can overlap, and so on;

ii) a CGS can contain incompatible segments;

iii) the basis $B_i$ can be reduced or not;

iv) even when a given algorithm does not theoretically ensure some property it can however hold it experimentally in most examples.

So it is quite difficult to make automatic comparisons of the outputs.

There are three available known implemented methods for obtaining a GCS:

i) Weispfenning CGB implemented by (Dolzman-Seidl-Sturm 2006) in Reduce.

ii) Suzuki-Sato SACGB implemented in Risa/Asir[2],

iii) Montes MCCGS implemented in Maple 8 by M. Manubens in the DPGB library 7.0.

Even though we use some criteria to evaluate them: correctness of the results, existence of incompatibilities, existence of overlaps, number of segments, whether the $S_i$ form a partition, whether or not specializations preserve the lpp's of the bases, reduction of the bases, theoretical canonicity ensured, theoretical minimality ensured, execution time.

Although it is not possible to evaluate Weispfenning's CCGB algorithm in practice because it has not been implemented, we can analyze its theoretical features. The canonicity of CCGB comes from the use of primary decompositions over the conditions, but the method is not dichotomic and so the segments are not disjoint. As its objective is to obtain a canonical CGB, the bases of the corresponding CGS are faithful and therefore not reduced, so specializations do not preserve their lpp. Furthermore as the segments are not disjoint, minimality does not hold.

For the comparisons with the implemented methods, we have used a Pentium(R) D CPU 3.00 GHz, 1.00 GB RAM for the computations and tested different examples using the above implementations.

We have not been able to obtain CGB Reduce in time for these comparisons, so we could only test some very simple executions with a demo version. To what we have experimentally observed, it gives a partition of the parameter space containing quite more segments than MCCGS. The bases are faithful, which is interesting to compute a CGB, but do not give direct information on the type of solutions, as these bases are not reduced. It seems to be very efficient but the provided results are difficult to be interpreted. In the future we will make a more precise analysis.

SACGB is a very simple and interesting algorithm based on Kalkbrenner's theorem for stabilization of polynomial ideals over rings (Kalkbrenner 1997) under specialization. The published algorithm provides a highly complex CGS, containing even incompatible segments, but the Risa/Asir implementation makes an initial reduction and gives a better output. We implemented an extra routine to further reduce the output by transforming specifications into red-specifications characterized by a pair $(N, W)$, where $N$ is the null-condition ideal and $W$ is a set of irreducible polynomials.

Among the tests we have done we explain four interesting ones.

**Example 10.** First we consider a very simple but illustrative example: the discussion of the singular points of a conic already studied in example 9.

[2] There exist also a preliminary Maple version but it is not yet fully developed.

Using the Risa/Asir implementation of SACGB together with the additional simplifications we obtain the following description of the CGS:

| lpp's | Basis | Description |
|---|---|---|
| $[1]$ | $[1]$ | $\mathbb{C}^3 \setminus \mathbb{V}(bcd)$ |
| $[1]$ | $[1]$ | $\mathbb{V}(c) \setminus \mathbb{V}(d)$ |
| $[x]$ | $[x+cy]$ | $\mathbb{V}(d, b-c^2)$ |
| $[y,x]$ | $[y,x]$ | $\mathbb{V}(d) \setminus \mathbb{V}((b-c^2)c)$ |
| $[y,x]$ | $[y, 2x+d]$ | $\mathbb{V}(d,c) \setminus \mathbb{V}(b)$ |
| $[y,x]$ | $[2cy+d, x]$ | $\mathbb{V}(b) \setminus \mathbb{V}(cd)$ |

There are two segments with basis [1], i.e. when the conic has no singular points. The first one corresponds to the whole $\mathbb{C}^3$ space except the three planes $\mathbb{V}(b)$, $\mathbb{V}(c)$ and $\mathbb{V}(d)$. The second one corresponds to the plane $\mathbb{V}(c)$ except the line $\mathbb{V}(c,d)$. They have empty intersection and its union describes the unique generic segment in MCCGS, namely the whole $\mathbb{C}^3$ space except the two planes $\mathbb{V}(b)$ and $\mathbb{V}(d)$ plus the line $\mathbb{V}(b,c)$ except the origin $(0,0,0)$.

The segment with lpp set $[x]$ (i.e. the conic is a double line of singular points) coincides with the one in MCCGS.

Finally, there are three segments with lpp set $[x,y]$, i.e. the conic has one single singular point. The first one corresponds to the plane $\mathbb{V}(d)$ minus the line $\mathbb{V}(c,d)$ and the parabola $\mathbb{V}(d, b-c^2)$. The second one corresponds to the line $\mathbb{V}(c,d)$ minus the origin $(0,0,0)$. The third one corresponds to the plane $\mathbb{V}(b)$ minus the lines $\mathbb{V}(b,c)$ and $\mathbb{V}(b,d)$. These three sets have no common intersection and their union describes the plane $\mathbb{V}(b)$ minus the line $\mathbb{V}(b,c)$ plus the plane $\mathbb{V}(d)$ minus the parabola $\mathbb{V}(d, b-c^2)$, which is the unique segment in MCCGS. Also the basis given by MCCGS for this segment specializes to the bases of the three segments provided by SACGB.

Using Reduce implementation of Weispfenning's CGB, we obtained the following CGS:

| Segment | Basis | Description |
|---|---|---|
| 1 | $[bd^2]$ | $b^2cd - bc^3d \neq 0$ |
| 2 | $[x^2+2cxy+dx+by^2, 2x+2cy+d,$ $cx+by, (2b-2c^2)y-cd]$ | $b-c^2 \neq 0, c \neq 0, bd = 0$ |
| 3 | $[2cdy+d^2]$ | $b \neq 0, d \neq 0, c = 0$ |
| 4 | $[x^2+2cxy+dx+by^2, 2x+2cy+d, cx+by]$ | $b \neq 0, c = 0, d = 0$ |
| 5 | $[(2b-2c^2)y-cd]$ | $c \neq 0, d \neq 0, b-c^2 = 0$ |
| 6 | $[x^2+2cxy+dx+by^2, 2x+2cy+d, cx+by]$ | $c \neq 0, d \neq 0, b-c^2 = 0$ |
| 7 | $[4cxy+4by^2-2cdy-d^2]$ | $d \neq 0, b = 0, c = 0$ |
| 8 | $[x^2+2cxy+dx+by^2, 2x+2cy+d]$ | $b = 0, c = 0, d = 0$ |

As it can be seen, the description of the segments is not very friendly. In order to interpret these CGS as a partition we have manually built the following binary table in

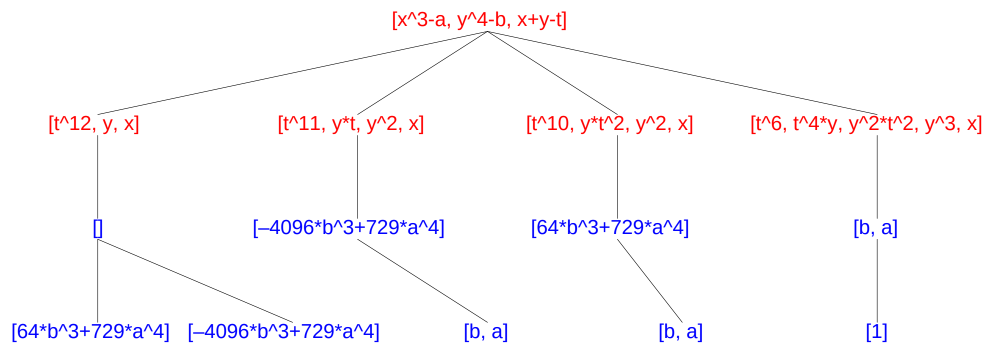


Figure 7: Canonical tree for $[x^3 - a, y^4 - b, x + y - t]$ wrt lex$(x, y, t)$.

which 0 represents "being equal to 0", and 1 "being different from 0". The last column matches each CGS segment with one of the three MCCGS segments identified by its lpp.

| Segment | $b$ | $c$ | $d$ | $b - c^2$ | MCCGS lpp |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | [1] |
| 2 | 0 | 1 | 0 | 1 | $[x, y]$ |
| | 0 | 1 | 1 | 1 | |
| | 1 | 1 | 0 | 1 | |
| 3 | 1 | 0 | 1 | 1 | [1] |
| 4 | 1 | 0 | 0 | 1 | $[x, y]$ |
| 5 | 1 | 1 | 1 | 0 | [1] |
| 6 | 1 | 1 | 0 | 0 | $[x]$ |
| 7 | 0 | 0 | 1 | 0 | [1] |
| 8 | 0 | 0 | 0 | 0 | $[x]$ |

The CPU times are 1.46 sec for MCCGS, 0.18 sec for SACGS and 0.05 sec for CGB.

We see that MCCGS outputs a simpler discussion, not only theoretically but also experimentally as all the segments corresponding to the same set of solutions are summarized in a single segment, while SACGS and CGB do not. Nevertheless, SACGS and CGB are both correct and faster than MCCGS, and although they do not ensure that the $S_i$ form a partition of the parameter space, in this example they do.

**Example 11.** We consider now an example proposed in (Suzuki-Sato 2006) for which they give the following comprehensive Gröbner basis wrt $lex(t, x, y)$

$$S := [x^3 - a, y^4 - b, x + y - t]$$

and ask for the CGS of $\langle S \rangle$ wrt lex$(x, y, t)$.

MCCGS provides in 632 sec. the canonical tree shown in Figure 7 with only 4 segments which takes 16 lines of a Maple worksheet.

On the other hand, the Risa/Asir SACGB with the respective simplifications produces in 1.62 sec. the following CGS which takes 22 lines of a Maple worksheet: (for space restrictions we do not print the bases)

| lpp's | Description |
|---|---|
| $[t^{12}, y, x]$ | $\mathbb{C}^2 \setminus \mathbb{V}(ab(729a^4 + 64b^3)(729a^4 - 4096b^3)(16767a^4 + 5632b^3))$ |
| $[t^{12}, y, x]$ | $\mathbb{V}(16767a^3 + 5632b^3) \setminus \mathbb{V}(ab)$ |
| $[t^{12}, y, x]$ | $\mathbb{V}(a) \setminus \mathbb{V}(b)$ |
| $[t^{12}, y, x]$ | $\mathbb{V}(b) \setminus \mathbb{V}(a)$ |
| $[t^{11}, ty, y^2, x]$ | $\mathbb{V}(729a^4 - 4096b^3) \setminus \mathbb{V}(ab)$ |
| $[t^{10}, t^2y, y^2, x]$ | $\mathbb{V}(729a^4 - 64b^3) \setminus \mathbb{V}(ab)$ |
| $[t^6, t^4y, t^2y^2, y^3, x]$ | $\mathbb{V}(b, a)$ |

The first segment is described by the whole $\mathbb{C}^2$ space minus the three curves $\mathbb{V}(729a^4 + 64b^3)$, $\mathbb{V}(729a^4 - 4096b^3)$, $\mathbb{V}(16767a^4 + 5632b^3)$ and the lines $\mathbb{V}(a)$ and $\mathbb{V}(b)$. The second one is described by the curve $\mathbb{V}(16767a^4 + 5632b^3)$ except the origin. The third one is the line $\mathbb{V}(a)$ minus the origin and the forth segment is described by the line $\mathbb{V}(b)$ minus the origin. These four segments have empty intersection and are associated to bases with lpp set $[t^{12}, y, x]$. Their union corresponds to the unique generic segment in MCCGS, namely the whole $\mathbb{C}^2$ space minus the two curves $\mathbb{V}(729a^4 + 64b^3)$ and $\mathbb{V}(729a^4 - 4096b^3)$.

The segment in SACGB with lpp set $[t^{11}, ty, y^2, x]$ is described by the curve $\mathbb{V}(729a^4 - 4096b^3)$ except the origin, which corresponds exactly to the segment associated to the same lpp set in MCCGS.

The segment with lpp set $[t^{10}, t^2y, y^2, x]$ and described by the curve $\mathbb{V}(729a^4 - 64b^3)$ minus the origin also coincides with the one in MCCGS associated to this lpp set.

And finally, the segment having basis with lpp set $[t^6, t^4y, t^2y^2, y^3, x]$ is described on the origin $\mathbb{V}(b, a)$, which agrees with the segment associated to the same lpp set in MCCGS.

All seven segments have no common intersection and thus they form a partition of the $\mathbb{C}^2$ space, even though SACGB does not ensure it.

**Example 12.** We also have tried to test SACGB with the systems of the nine points circle theorem explained in section 5 above. SACGB after 3 hours of computation went out of memory and had not yet reached an end, while MCCGS takes only 11.45 sec. for testing the compatibility of the hypotheses and 2.21 sec. for discussing the theorem thesis.

**Example 13.** The last test is the system of the Romin robot(González-Recio 1993):

$$R = [a + ds_1, b - dc_1, l_2c_2 + l_3c_3 - d, l_2s_2 + l_3s_3 - c, s_1^2 + c_1^2 - 1, s_2^2 + c_2^2 - 1, s_3^2 + c_3^2 - 1]$$

wrt lex$(c_3, s_3, c_2, s_2, c_1, s_1)$ and lex$(l_2, l_3, a, b, c, d)$. MCCGS takes 43.23 sec in discussing the system and provides 9 segments. SACGB also went out of memory.

# Conclusions

The interest of MCCGS relies, essentially, in the simplicity of the output for applications, and in the canonical character of it, conceding an easier interpretation of the results. We have also observed that the obtention of the MCCGS from the BUILDTREE CGS only increases the computation time in about 20-30%.

The existence of the MCCGS depends on the Conjecture formulated in (Montes 2007). The use of the algorithm will provide evidence of it or a counterexample. In almost all the high number of tests that we have done the algorithm has always obtained a unique segment for each different lpp set, confirming the conjecture. The only ideal for which the algorithm obtains two different segments with the same lpp is $= \langle u(ux+1), (ux+1)x \rangle$ proposed by (Wibmer 2006), and there both segments are clearly intrinsically different and cannot be merged nor summarized into a single one. Thus this example also provides evidence of the Conjecture. To give a counterexample proving the falsehood of the Conjecture, we must find an ideal for which the algorithm MCCGS obtains two or more segments with the same lpp which could be merged or summarized in a different way.

Although we have only made some very simple tests with CGB, we have observed that it seems faster than SACGB and MCCGS in those specific problems. It stands out for computing a CGS with faithful bases which are not always useful for applications. Experimentally, it seems to obtain a partition of the parameter space, even if there is no theoretic evidence. Nevertheless, the number of segments is much higher than MCCGS and are difficult to understand.

SACGB stands out for being in general very reliable to compute a CGS. Its efficiency depends on the type of system to be dealt with. It seems to behave faster than MCCGS in problems for which a low number of cases is expected. Furthermore, we must remind that the output of SACGB is very complex and also needs extra simplifications to be interpreted.

One can also adapt the MCCGS algorithm to the CGS obtained by other algorithms instead of BUILDTREE. To do this one needs to transform the output of the involved algorithm into a disjoint reduced CGS, and then apply step ii) and iii), i.e. SELECTCASES and MCCGS.

MCCGS takes, generally, more CPU time for simple problems. Nevertheless the simplifications inside MCCGS often allow to discuss systems of higher complexity, as seen in examples 12 and 13 above.

Finally, we have seen that MCCGS algorithm stands out for having the best features to be used for automatic theorem proving and discovering as well as for other applications.

## Ackowledgements

We are very grateful to Josep M. Brunat for his suggestions on the recursive handling of trees. We would also like to thank Professor Pelegrí Viader for his many helpful comments and his insightful perusal of our first draft. Finally, we also want to thank our referees for their very helpful comments.

## References

[Dolzman-Seidl-Sturm 2006] A. Dolzmann, A. Seidl, T. Sturm. (2006) REDLOG software in REDUCE http://staff.fim.uni-passau.de/∼ sturm/

[González-Recio 1993] González-López, M.J., Recio, T., 1993. The ROMIN inerse geometric model and the dynamic evaluation method. In: Computer Algebra in Industry, A.M. Cohen ed., John Wiley & Sons: 117–141 (1993).

[González-Traverso-Zanoni 2005] González-Vega, L., Traverso, C., Zanoni, A., 2005. Hilbert Stratification and Parametric Groebner Bases. CASC-2005, p. 220-235.

[Kalkbrenner 1997] Kalkbrenner, M., 1997. On the stability of Gröbner bases under specializations. *Jour. Symb. Comp.*, **24**(1): (1997), 51–58.

[Manubens-Montes 2006] Manubens, M., Montes, A., 2006. Improving DISPGB Algorithm Using the Discriminant Ideal. *Jour. Symb. Comp.* **41** (2006), 1245–1263.

[Montes 2002] Montes, A., 2002. New Algorithm for Discussing Gröbner Bases with Parameters. *Jour. Symb. Comp.* **33**:1-2 (2002), 183–208.

[Montes 2007] Montes, A., 2007. On the canonical discussion of polynomial systems with parameters. Preprint arXiv: AC/0601674.

[Montes-Recio 2007] Montes, A., Recio, T., 2006. Automatic discovery of geometry theorems using minimal canonical comprehensive Groebner systems. Preprint arXiv: AC/0703483.

[Nabeshima 2005] Nabeshima, K., 2005. A computation method for ACGB-V. Proceedings of A3L 2005 (Conference in Honour of the 60th Bithday of V. Weispfenning), eds. A. Dolzmann, A Seidl, T. Sturm. p 171-180. BOD Norderstedt.

[Nabeshima 2006] Nabeshima, K., 2006. Comprehensive Groebner Bases for Modules. Short communication, ACA-2006, Varna.

[Sato 2005] Sato, Y., 2005. Stability of Gröbner basis and ACGB. Proceedings of A3L 2005 (Conference in Honour of the 60th Birthday of V. Weispfenning), eds. A. Dolzmann, A Seidl, T. Sturm. p 223-228. BOD Norderstedt.

[Sato-Suzuki 2003] Sato, Y., Suzuki, A., 2003. An alternative approach to Comprehensive Gröbner bases. *Jour. Symb. Comp.* **36**:3-4 (2003), 649-667.

[Suzuki-Sato 2006] Suzuki, A., Sato, Y., 2006. A Simple Algorithm to compute Comprehensive Gröbner bases. Proceedings of ISSAC 2006, ACM. p 326-331. Implementation in Risa/Asir and Maple (http://kurt.scitec.kobe-u.ac.jp/∼sakira/).

[Weispfenning 1992] Weispfenning, V., 1992. Comprehensive Gröbner bases. *Jour. Symb. Comp.* **14**:1-1 (1992), 1-29.

[Weispfenning 2003] Weispfenning, V., 2003. Canonical Comprehensive Gröbner bases. *Jour. Symb. Comp.* **36**:3-4 (2003), 669-683.

[Wibmer 2006] M. Wibmer. Gröbner bases for families of affine schemes. arXiv. math/0608019 (2006).