Contents lists available at SciVerse ScienceDirect

Journal of Symbolic Computation

journal homepage: www.elsevier.com/locate/jsc

An efficient algorithm for computing a comprehensive Gröbner system of a parametric polynomial system[☆]

Deepak Kapur[a], Yao Sun[b,1], Dingkang Wang[b]

[a] Department of Computer Science, University of New Mexico, Albuquerque, NM, USA

[b] KLMM, Academy of Mathematics and Systems Science, CAS, Beijing 100190, China

ARTICLE INFO

Article history:
Received 9 December 2010
Accepted 10 May 2011
Available online 22 December 2011

Keywords:
Gröbner basis
Comprehensive Gröbner system
Radical ideal membership
Probabilistic check

ABSTRACT

A new efficient algorithm for computing a comprehensive Gröbner system of a parametric polynomial ideal over $k[U][X]$ is presented. This algorithm generates fewer branches (segments) compared to previously proposed algorithms including Suzuki and Sato's algorithm as well as Nabeshima's algorithm. As a result, the algorithm is able to compute comprehensive Gröbner systems of parametric polynomial ideals arising from applications which have been beyond the reach of other well known algorithms. The starting point of the new algorithm is Weispfenning's algorithm with a key insight by Suzuki and Sato who proposed computing first a Gröbner basis of an ideal over $k[U, X]$ before performing any branches based on parametric constraints. The proposed algorithm exploits the result that along any branch in a tree corresponding to a comprehensive Gröbner system, it is only necessary to consider one polynomial for each nondivisible leading power product in $k(U)[X]$ with the condition that the product of their leading coefficients is not 0; other branches correspond to the cases where this product is 0. In addition, for dealing with a disequality parametric constraint, a probabilistic check is employed for radical membership test of an ideal of parametric constraints. This is in contrast to a general expensive check based on Rabinovitch's trick using a new variable as in Nabeshima's algorithm. The proposed algorithm has been implemented in Magma and Singular, and experimented with a number of examples from different applications. Its performance (the number of branches and execution time) has been compared with several other existing algorithms. A number of

[☆] This paper is an expanded version of the paper entitled "A new algorithm for computing comprehensive Gröbner systems," which was presented at ISSAC 2010 (Kapur et al., 2010).

E-mail addresses: kapur@cs.unm.edu (D. Kapur), sunyao@amss.ac.cn (Y. Sun), dwang@mmrc.iss.ac.cn (D. Wang).

[1] Tel.: +86 10 62651374; fax: +86 10 62630706.

0747-7171/$ – see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jsc.2011.12.015

heuristics and efficient checks have been incorporated into the Magma implementation, especially in the case when the ideal of parametric constraints is 0-dimensional. The algorithm has been successfully used to solve a special case of the famous P3P problem from computer vision.

1. Introduction

A new algorithm for computing a comprehensive Gröbner system (CGS), as defined by Weispfenning (1992, 2003) for parametric ideals (see also Kapur (1995) where a related concept of parametric Gröbner system was introduced) is proposed. The main advantage of the proposed algorithm is that it generates fewer branches (segments) compared to other related algorithms; as a result, the algorithm is able to compute comprehensive Gröbner systems for many problems from different application domains which could not be done previously. In the rest of this section, we provide some motivations for comprehensive Gröbner systems and approaches used for computing them.

Many engineering problems are parameterized and have to be repeatedly solved for different values of parameters (Donald et al., 1992). A case in point is the problem of finding solutions of a parameterized polynomial system. One is interested in finding parameter values for which the polynomial system has a common solution; more specifically, if there are solutions, one is also interested in finding out the structure of the solution space (finitely many, infinitely many, in which case, their dimension is of interest, etc.). One recent application of comprehensive Gröbner systems is in automated geometry theorem proving (Chen et al., 2005) and automated geometry theorem discovery (Montes and Recio, 2007). In the former, the goal is to consider all possible cases arising from an ambiguous problem formulation to determine whether the conjecture is generic enough to be valid in all cases, or certain cases have to be ruled out. In the latter, one is interested in identifying different relationship among geometric entities for different parameter values. Another recent application is in the automatic generation of loop invariants and inductive assertions of programs operating on numbers using quantifier elimination methods as proposed in Kapur (2006). The main idea is to hypothesize invariants/assertions to have a template like structure (such as a polynomial in which the degree of every variable is ≤ 2, or a polynomial with a predetermined support), in which the presence/coefficient of a power product is parameterized. Verification conditions from the program are then generated which are formulas involving parameterized polynomial equations. The objective is to generate conditions on parameters which make these verification conditions valid, (see Kapur (2006) for more details).

Let k be a field, R be the polynomial ring $k[U]$ in the parameters $U = \{u_1, \dots, u_m\}$, and $R[X]$ be the polynomial ring over the parameter ring R in the variables $X = \{x_1, \dots, x_n\}$; it is assumed that $X \cap U = \emptyset$, i.e., X and U are disjoint sets.

Given a polynomial set $F \subset R[X]$, we are interested in identifying conditions on parameters U such that the solution structure of the specialized polynomial system F for the values of U satisfying these conditions is different from other parameter values. One way to do this is to compute a comprehensive Gröbner system as introduced by Weispfenning, which is a finite set of triples of the form (E_i, N_i, G_i), where E_i, N_i are finite sets of polynomials in $k[U]$ and $\sigma(G_i)$ is a Gröbner basis of $\sigma(F)$ for every specialization σ such that for every $e_i \in E_i$, e_i vanishes and for at least one $n_i \in N_i$, n_i does not vanish; we will say that in that case σ satisfies the parametric constraints specified by E_i and N_i. Furthermore, for every specialization, there is at least one triple whose parametric constraints satisfy it. We will call each triple a *branch* (also called a segment) in a comprehensive Gröbner system.

In 1992, Weispfenning (1992) gave an algorithm for computing a comprehensive Gröbner system but it suffered from the problem of too many branches, many of which leading to the Gröbner basis $\{1\}$, implying the inconsistency of the corresponding parameterized polynomial systems for those specific parametric specializations.[2] Since then, many improvements have been made to improve

2 Kapur's algorithm for parametric Gröbner bases suffered from similar weaknesses.

these algorithms to make them useful for different applications; see Montes (2002), Suzuki and Sato (2003, 2004), Manubens and Montes (2006) and Wibmer (2007). A major breakthrough was an algorithm proposed by Suzuki and Sato (2006) (henceforth called the SS algorithm) in which they showed how traditional implementations of Gröbner basis algorithms for polynomial rings over a field could be exploited for computing a comprehensive Gröbner basis system. More recently, there is an interesting related concept of Gröbner cover introduced in Montes and Wibmer (2010). Parametric polynomial systems have also been investigated using parametric characteristic sets (Gao and Chou, 1992; Chen et al., 2007). Below, we discuss the main ideas of Suzuki and Sato's algorithm and Nabeshima's algorithm because of the close relationship with the proposed algorithm in this paper.

The main idea of the SS algorithm is to compute a Gröbner basis G from the parametric ideal basis in $k[U, X]$ using the block ordering in which $U \ll X$. In case G has polynomials purely in the parameters U, there are branches corresponding to each such polynomial being not equal to 0 in which case the Gröbner basis is $\{1\}$ for the specialization. For the branch when all these polynomials are 0, the Gröbner basis is G minus these polynomials under the additional condition that the leading coefficient of each polynomial is nonzero. In addition, there are branches corresponding to the cases when each of these leading coefficients is 0.

Nabeshima's speed-up algorithm (Nabeshima, 2007) improves upon the SS algorithm by using the fact that (i) for every leading power product, only one coefficient needs to be made nonzero, and (ii) Rabinovitch's trick of introducing a new variable can be used to make that polynomial monic. Nabeshima reported that these tricks led to fewer branches of the SS-algorithm for most examples.

First, let G be the the reduced Gröbner basis of a parametric ideal $\langle F\rangle \subset k[U, X]$ w.r.t. $\prec_{X,U}$, and let $G_r = G \cap k[U]$, the polynomials in parameters only in G. A minimal Dickson basis G_m, which is defined in Section 4, is extracted from $G \setminus G_r$, consisting only of polynomials with nondivisible power products in X in G. Let h be the product of the leading coefficients of the polynomials in G_m. $(G_r, \{h\}, G_m)$ is one of the branches of the comprehensive Gröbner system of F. Based on case analysis over the leading coefficients of the polynomials in G_m, it is possible to compute the remaining branches of a comprehensive Gröbner system.

For computing a Gröbner basis for specializations along many branches, it is useful to perform radical membership check of a parametric constraint in an ideal of other parametric constraints for checking consistency. Instead of using Rabinovitch's trick of introducing a new variable for radical membership check as proposed in Nabeshima's speed-up version of the SS algorithm, we have developed a collection of useful heuristics for this check based on case analysis on whether the ideal whose radical membership is being checked, is 0-dimensional or not. In case of a positive dimensional ideal, a probabilistic check is employed after randomly specializing the independent variables of the ideal. The general check is performed as a last resort.

The paper is organized as follows. Section 2 gives notations and definitions used. Section 3 briefly reviews the Suzuki–Sato algorithm. Section 4 is the discussion of the key insights needed for the proposed algorithm. This is followed by a high-level description of the algorithm and its termination proof. Section 5 discusses different methods for checking consistency of parametric constraints based on the dimension of the ideal generated from equality constraints. Section 6 illustrates the proposed algorithm on a simple example. Section 7 discusses optimizations for making the implementation of the proposed algorithm more efficient by processing parametric constraints using quotient ideals as well as by exploiting the dimensionality of the parametric equality constraints. Empirical data and comparison with several other existing algorithms are presented in Section 8, where the performance comparison of different heuristics for checking consistency of parametric constraints are discussed. Concluding remarks follow in Section 9.

2. Notations and definitions

Let k be a field, R be the polynomial ring $k[U]$ in the parameters $U = \{u_1, \ldots, u_m\}$, and $R[X]$ be the polynomial ring over R in the variables $X = \{x_1, \ldots, x_n\}$ and $X \cap U = \emptyset$.

Let $PP(X)$, $PP(U)$ and $PP(U, X)$ be the sets of power products of X, U and $U \cup X$ respectively. $\prec_{X,U}$ is an admissible block term order on $PP(U, X)$ such that $U \ll X$. $\prec_X$ and $\prec_U$ are restrictions of $\prec_{X,U}$ on $PP(X)$ and $PP(U)$, respectively.

For a polynomial $f \in R[X] = k[U][X]$, the leading power product, leading coefficient and leading monomial of f w.r.t. the order $\prec_X$ are denoted by $\mathrm{lpp}_X(f)$, $\mathrm{lc}_X(f)$ and $\mathrm{lm}_X(f)$ respectively. Since f can also be regarded as an element of $k[U, X]$, in this case, the leading power product, leading coefficient and leading monomial of f w.r.t. the order $\prec_{X,U}$ are denoted by $\mathrm{lpp}_{X,U}(f)$, $\mathrm{lc}_{X,U}(f)$ and $\mathrm{lm}_{X,U}(f)$ respectively.

Given a field L, a **specialization** of R is a homomorphism $\sigma : R \longrightarrow L$. In this paper, we assume L to be the algebraic closure of k, and consider the specializations induced by the elements in L^m. That is, for $\bar{a} \in L^m$, the induced homomorphism $\sigma_{\bar{a}}$ is denoted as $\sigma_{\bar{a}} : f \longrightarrow f(\bar{a})$, where $f \in R$. Every specialization $\sigma : R \longrightarrow L$ extends canonically to a homomorphism $\sigma : R[X] \longrightarrow L[X]$ by applying σ coefficient-wise. For an $F \subset R = k[U]$, the variety defined by F in L^m is denoted by $V(F)$. Parametric specializations can be grouped together using parametric constraints defined below and the associated algebraic constructible subsets.

Definition 2.1. For $E, N \subset R = k[U]$, a pair (E, N) is called a **parametric constraint**. A parametric constraint (E, N) is said to be **consistent** if the set $V(E) \setminus V(N)$ is not empty. Otherwise, (E, N) is called inconsistent.

It is easy to see that the consistency of (E, N) can be checked by ensuring that at least one $f \in N$ is not in the radical of $\langle E\rangle$. The above parametric constraint corresponds to a formula $\bigwedge_{e_i \in E}(e_i = 0) \wedge \neg(\bigwedge_{n_j \in N}(n_j = 0))$ over the parameters.

Definition 2.2. A constructible set A is defined as a pair of finite sets of polynomials (E, N) such that $A = V(E) \setminus V(N)$ where E, N are subsets of $k[U]$.

Definition 2.3. Let F be a subset of $R[X]$, $A_1, \ldots, A_l$ be algebraically constructible subsets of L^m and $G_1, \ldots, G_l$ be subsets of $R[X]$, and S be a subset of L^m such that $S \subseteq A_1 \cup \cdots \cup A_l$. A finite set $\mathcal{G} = \{(A_1, G_1), \ldots, (A_l, G_l)\}$ is called a **comprehensive Gröbner system** on S for F if $\sigma_{\bar{a}}(G_i)$ is a Gröbner basis of the ideal $\langle\sigma_{\bar{a}}(F)\rangle \subset L[X]$ for $\bar{a} \in A_i$ and $i = 1, \ldots, l$. Each (A_i, G_i) is called a branch of $\mathcal{G}$. Particularly, if $S = L^m$, then $\mathcal{G}$ is called, simply, a comprehensive Gröbner system for F.

Definition 2.4. A comprehensive Gröbner system $\mathcal{G} = \{(A_1, G_1), \ldots, (A_l, G_l)\}$ on S for F is said to be **minimal** if for every $i = 1, \ldots, l$,

1. $A_i \neq \emptyset$, and furthermore, for each $i, j = 1 \cdots l$, $A_i \cap A_j = \emptyset$ whenever $i \neq j$, and
2. $\sigma_{\bar{a}}(G_i)$ is a minimal Gröbner basis of the ideal $\langle\sigma_{\bar{a}}(F)\rangle \subset L[X]$ for $\bar{a} \in A_i$, and
3. for each $g \in G_i$, $\sigma_{\bar{a}}(\mathrm{lc}_X(g)) \neq 0$ for any $\bar{a} \in A_i$.

If $A_i = V(E_i) \setminus V(N_i)$ is empty, the branch (A_i, G_i) is redundant.

3. The Suzuki–Sato algorithm

In this section, we briefly review the key ideas of the Suzuki–Sato algorithm (Suzuki and Sato, 2006). The following two lemmas serve as the basis of the SS algorithm. The first lemma is a corollary of Theorem 3.1 given by Kalkbrener (1997).

Lemma 3.1. *Let G be a Gröbner basis of the ideal $\langle F\rangle \subset k[U, X]$ w.r.t. the order $\prec_{X,U}$. For any $\bar{a} \in L^m$, let $G_1 = \{g \in G \mid \sigma_{\bar{a}}(\mathrm{lc}_X(g)) \neq 0\}$. Then $\sigma_{\bar{a}}(G_1) = \{\sigma_{\bar{a}}(g) \mid g \in G_1\}$ is a Gröbner basis of $\langle\sigma_{\bar{a}}(F)\rangle$ in $L[X]$ w.r.t. $\prec_X$ if and only if $\sigma_{\bar{a}}(g)$ reduces to 0 modulo $\sigma_{\bar{a}}(G_1)$ for every $g \in G$.*

The next lemma, which follows from the first lemma, plays the key role in the design of the SS algorithm.

Lemma 3.2. *Let G be a Gröbner basis of the ideal $\langle F\rangle \subset k[U, X]$ w.r.t. the order $\prec_{X,U}$. If $\sigma_{\bar{a}}(\mathrm{lc}_X(g)) \neq 0$ for each $g \in G \setminus (G \cap R)$, then $\sigma_{\bar{a}}(G)$ is a Gröbner basis of $\langle\sigma_{\bar{a}}(F)\rangle$ in $L[X]$ w.r.t. $\prec_X$ for any $\bar{a} \in V(G \cap R)$.*

The main idea of the SS algorithm is to first compute a reduced Gröbner basis, say G, of $\langle F\rangle \subset k[U, X]$ w.r.t. $\prec_{X,U}$, which is also a Gröbner basis of the ideal $\langle F\rangle \subset k[U][X]$ w.r.t. $\prec_X$. Let $\{h_1, \dots, h_l\} = \{\mathrm{lc}_X(g) \mid g \in G \setminus R\} \subset R$. By the above lemma, $(G \cap k[U], V(h_1) \cup \dots \cup V(h_l), G)$ forms a branch of the comprehensive Gröbner system for F. That is, for any $\bar{a} \in V(G \cap k[U]) \setminus (V(h_1) \cup \dots \cup V(h_l))$, $\sigma_{\bar{a}}(G)$ is a Gröbner basis of $\langle \sigma_{\bar{a}}(F)\rangle$ in $L[X]$ w.r.t. $\prec_X$. To compute other branches corresponding to the specialization $\bar{a} \in V(h_1) \cup \dots \cup V(h_l)$, Lemma 3.2 is used for each $F \cup \{h_i\}$, the above steps are repeated. Since $h_i \notin \langle F\rangle$, the algorithm terminates in finitely many steps.

As stated earlier, this algorithm can be easily implemented in most of the computer algebra systems already supporting an efficient implementation of a Gröbner basis algorithm over a polynomial ring over a field. It has very good performance since it can take advantage of well-known fast implementations for computing Gröbner bases.

The algorithm however suffers from certain weaknesses. The algorithm does not check whether $V(G \cap R) \setminus V(h)$ is empty; as a result, many redundant/unnecessary branches may be produced. In Suzuki and Sato (2006), an improved version of the algorithm is reported which removes redundant branches. To reduce the number of branches generated from the SS algorithm, Nabeshima proposed a speed-up algorithm in Nabeshima (2007). The main idea of that algorithm is to exploit disequality parametric constraints for simplification. For every leading power product in $G \setminus R$ that is a nontrivial multiple of any other leading product in it, a branch is generated by asserting its leading coefficient h_i to be nonzero. The corresponding polynomial is made monic using Rabinovitch's trick of introducing a new variable to handle the disequality $h_i \neq 0$, and the Gröbner basis computation is performed again, simplifying polynomials whose leading power products are multiples, including their parametric coefficients.

4. The proposed algorithm

We present below a new algorithm for computing a comprehensive Gröbner system which avoids unnecessary branches in the SS algorithm. This is done using the radical ideal membership check for parametric constraints asserted to be nonzero. Heuristics are employed to do this check; when these heuristics fail, as exhibited by Table 2 in Section 8 on experimental results, only then the general check is performed by introducing a new variable, since this check is very inefficient because of the extra variable. Further, all parametric constraints leading to the specialized Gröbner basis being 1 are output as a single branch, leading to a compactified output.

Another major improvement of the proposed algorithm is that along any other branch for which the specialized Gröbner basis is different from 1, exactly one polynomial from $G \setminus R$ per minimal leading power product is selected. This is based on a generalization of Kalkbrener's Theorem 3.1. All these results are integrated into the proposed algorithm, resulting in considerable efficiency over the SS algorithm and Nabeshima's improved algorithm by avoiding expensive Gröbner basis computations along most branches.

The proposed algorithm is based on the following theorem. The definitions below are used in the theorem.

Definition 4.1 (*Minimal Dickson Basis*)**.** Given a set G of polynomials which are a subset of $k[U, X]$ and an admissible block order with $U \ll X$, we say $F \subset k[U, X]$, denoted as MDBasis(G), is a **minimal Dickson basis** of G, if

1. F is a subset of G,
2. for every polynomial $g \in G$, there is some polynomial $f \in F$ such that $\mathrm{lpp}_X(g)$ is a multiple of $\mathrm{lpp}_X(f)$, i.e. $\langle \mathrm{lpp}_X(F)\rangle = \langle \mathrm{lpp}_X(G)\rangle$, and
3. for any two distinct $f_1, f_2 \in F$, neither $\mathrm{lpp}_X(f_1)$ is a multiple of $\mathrm{lpp}_X(f_2)$ nor $\mathrm{lpp}_X(f_2)$ is a multiple of $\mathrm{lpp}_X(f_1)$.

The following simple example shows that given $G \subset k[U, X]$, MDBasis(G) may not be unique. Let $G = \{ax^2 - y, ay^2 - 1, ax - 1, (a + 1)x - y, (a + 1)y - a\} \subset \mathbb{Q}[a, x, y]$, with the lexicographic order on terms with $a < y < x$. Then $F = \{ax - 1, (a + 1)y - a\}$ and $F' = \{(a + 1)x - y, (a + 1)y - a\}$ are both MDBasis(G). It is easy to verify $\langle \mathrm{lpp}_X(F)\rangle = \langle \mathrm{lpp}_X(F')\rangle = \langle \mathrm{lpp}_X(G)\rangle = \langle x, y\rangle$.

Definition 4.2. Given $F \subset k[U,X]$ and $p \in k[U,X]$, p is said to be **divisible** by F if there exists an $f \in F$ such that some power product in X of p is divisible by $\mathrm{lpp}_X(f)$.

We should emphasize that "some power product" in the above definition may not be the leading power product.

Theorem 4.3. *Let G be a Gröbner basis of the ideal $\langle F\rangle \subset k[U,X]$ w.r.t. an admissible block order with $U \ll X$. Let $G_r = G \cap k[U]$ and $G_m = \mathrm{MDBasis}(G \setminus G_r)$. If σ is a specialization from $k[U]$ to L such that*

1. *$\sigma(g) = 0$ for $g \in G_r$, and*
2. *$\sigma(h) \neq 0$, where $h = \prod_{g\in G_m} \mathrm{lc}_X(g) \in k[U]$,*

then $\sigma(G_m)$ is a (minimal) Gröbner basis of $\langle\sigma(F)\rangle$ in $L[X]$ w.r.t. $\prec_X$.

Proof. Consider any $p \in G \setminus (G_r \cup G_m)$; p is divisible by G_m. p can be transformed by multiplying it with the leading coefficients of polynomials in G_m and then reduced using G_m, and then this process can be repeated on the result. Let r be the *remainder* of p w.r.t. G_m in X obtained by multiplying p by the leading coefficient of $g \in G_m$ such that r does not have any power product that is a multiple of any of the leading power products of polynomials in G_m (r could be different depending upon the order in which different polynomials in G_m are used to transform p). Thus,

$$(\mathrm{lc}_X(g_1))^{\alpha_1} \cdots (\mathrm{lc}_X(g_s))^{\alpha_s} p = q_1 g_1 + \cdots + q_s g_s + r, \tag{1}$$

where $g_i \in G_m$, $q_i \in k[U,X]$ for $i = 1,\ldots,s$, $r \in k[U,X]$ such that no power product of r in X is a multiple of any of the leading power products of G_m. Since $p \in \langle F\rangle$, $r \in \langle F\rangle$. Since G is a Gröbner basis of $\langle F\rangle$ in $k[U,X]$, r reduces to 0 by G. However, r is reduced (in normal form) w.r.t. G_m in X (and hence reduced w.r.t. $G \setminus G_r$ in X also, by the definition of G_m); so r reduces to 0 by G_r only and further no new power products in X can be introduced during the simplification of r by G_r. So $r \in \langle G_r\rangle \subset k[U,X]$. Additionally, $\mathrm{lpp}_X(p) \succeq \mathrm{lpp}_X(q_i g_i)$ since $\mathrm{lc}_X(g_i) \in k[U]$.

Let $c = (\mathrm{lc}_X(g_1))^{\alpha_1} \cdots (\mathrm{lc}_X(g_s))^{\alpha_s}$. Apply σ to the both sides of (1), then we have:

$$\sigma(c)\sigma(p) = \sigma(q_1)\sigma(g_1) + \cdots + \sigma(q_s)\sigma(g_s) + \sigma(r).$$

Since $\sigma(h) \neq 0$ by assumption, $\sigma(\mathrm{lc}_X(g)) \neq 0$ for $g \in G_m$; $\sigma(g) = 0$ for $g \in G_r$ which implies that $\sigma(r) = 0$. Notice $0 \neq \sigma(c) \in L$ and $\mathrm{lpp}_X(p) \succeq \mathrm{lpp}_X(q_i g_i)$, using Lemma 4.5, $\sigma(G_m)$ is a Gröbner basis of $\langle\sigma(G)\rangle = \langle\sigma(F)\rangle$. □

In the above theorem, if $G_r = \emptyset$, then G_m is actually a Gröbner basis of the ideal $\langle F\rangle$ generated in $k(U)[X]$.

We assume that the reader is familiar with the concept of t-representations which is often used to determine if a set of polynomials is a Gröbner basis; for details, the reader should consult (Becker et al., 1993). Here we only list the main results about t-representations.

Let $f \in k[X]$ and $G \subset k[X]$ be a finite subset. Given a power product t, we say f has a **t-representation** w.r.t. G, if there exist $p_1,\ldots,p_s \in k[X]$ such that

$$f = p_1 g_1 + \cdots + p_s g_s,$$

where $g_i \in G$ and $t \succeq_X \mathrm{lpp}_X(p_i g_i)$ for $i = 1,\ldots,s$.

Lemma 4.4. *Let $G \subset k[X]$ be a finite subset. The set G is a Gröbner basis for $\langle G\rangle \subset k[X]$, if and only if for any $g_i, g_j \in G$, the s-polynomial $\mathrm{spoly}_X(g_i,g_j)$ has a t-representation w.r.t. G, where $t \prec_X \mathrm{lcm}(\mathrm{lpp}_X(g_i), \mathrm{lpp}_X(g_j))$.*

The following lemma is used in the proof of Theorem 4.3.

Lemma 4.5. *Let G be a Gröbner basis of $\langle G\rangle \subset k[U,X]$ w.r.t. an admissible block order with $U \ll X$. Let $G_1 = \{g_1,\ldots,g_s\} \subset G$ and σ be a specialization from $k[U]$ to L such that $\sigma(\mathrm{lc}_X(g_i)) \neq 0$ for $i = 1,\ldots,s$. If for each $p \in G \setminus G_1$, there exist $p_1,\ldots,p_s \in L[X]$ such that:*

$$\sigma(p) = p_1\sigma(g_1) + \cdots + p_s\sigma(g_s),$$

where $\mathrm{lpp}_X(p) \succeq \mathrm{lpp}_X(p_i\sigma(g_i))$ for $i = 1,\ldots,s$, then $\sigma(G_1)$ is a Gröbner basis of $\langle\sigma(G)\rangle$ in $L[X]$ w.r.t. $\prec_X$.

Proof. By the hypothesis, it is easy to check $\sigma(G) \subset \langle \sigma(G_1) \rangle$ and hence $\sigma(G_1)$ is a basis of $\langle \sigma(G) \rangle$. So it remains to show $\sigma(G_1)$ is a Gröbner basis.

For each $g_j, g_k \in G_1$, we compute the s-polynomial of $\sigma(g_j)$ and $\sigma(g_k)$ in $L[X]$. Since $\sigma(\mathrm{lc}_X(g_j)) \neq 0$ and $\sigma(\mathrm{lc}_X(g_k)) \neq 0$, we have

$$\mathrm{spoly}(\sigma(g_j), \sigma(g_k)) = c\sigma(\mathrm{spoly}_X(g_j, g_k)), \tag{2}$$

where c is a nonzero constant in L and $\mathrm{spoly}_X(g_j, g_k) \in k[U][X]$ is the s-polynomial of g_j and g_k w.r.t. X.

Assume $G \setminus G_1 = \{g_{s+1}, \ldots, g_l\}$. Since G is a Gröbner basis of $\langle G \rangle \subset k[U, X]$ and $\mathrm{spoly}_X(g_j, g_k) \in \langle G \rangle \subset k[U, X]$, there exist $h_1, \ldots, h_l \in k[U, X]$ such that

$$\mathrm{spoly}_X(g_j, g_k) = h_1 g_1 + \cdots + h_l g_l,$$

where $\mathrm{lcm}(\mathrm{lpp}_X(g_j), \mathrm{lpp}_X(g_k)) \succ \mathrm{lpp}_X(h_i g_i)$ for $i = 1, \ldots, l$. Substitute back to (2), then obtain:

$$\mathrm{spoly}(\sigma(g_j), \sigma(g_k)) = c(\sigma(h_1)\sigma(g_1) + \cdots + \sigma(h_l)\sigma(g_l)), \tag{3}$$

where $\mathrm{lcm}(\mathrm{lpp}_X(\sigma(g_j)), \mathrm{lpp}_X(\sigma(g_k))) = \mathrm{lcm}(\mathrm{lpp}_X(g_j), \mathrm{lpp}_X(g_k)) \succ \mathrm{lpp}_X(h_i g_i) \succeq \mathrm{lpp}_X(\sigma(h_i))\mathrm{lpp}_X(g_i)$ for $i = 1, \ldots, l$. The next step is to use the hypothesis that for each $p \in G \setminus G_1$, there exist $p_1, \ldots, p_s \in L[X]$ such that: $\sigma(p) = p_1\sigma(g_1) + \cdots + p_s\sigma(g_s)$, where $\mathrm{lpp}_X(p) \succeq \mathrm{lpp}_X(p_i\sigma(g_i))$ for $i = 1, \ldots, s$. Substitute these representations back to (3), we get

$$\mathrm{spoly}(\sigma(g_j), \sigma(g_k)) = p'_1\sigma(g_1) + \cdots + p'_s\sigma(g_s), \tag{4}$$

where $p'_1, \ldots, p'_s \in L[X]$ and $\mathrm{lcm}(\mathrm{lpp}_X(\sigma(g_j)), \mathrm{lpp}_X(\sigma(g_k))) \succ \mathrm{lpp}_X(p'_i\sigma(g_i))$ for $i = 1, \ldots, s$. In fact, (4) is a t-representation of $\mathrm{spoly}(\sigma(g_j), \sigma(g_k))$ with $t \prec \mathrm{lcm}(\mathrm{lpp}_X(\sigma(g_j)), \mathrm{lpp}_X(\sigma(g_k)))$. Therefore, by Lemma 4.4, $\sigma(G_1)$ is a Gröbner basis. □

4.1. Algorithm

We are now ready to give the algorithm for computing a minimal comprehensive Gröbner system. Theorem 4.3 serves as the basis of its proof of correctness.

In order to keep the presentation simple so that the correctness and termination of the algorithm are evident, we have deliberately avoided tricks and optimizations such as factoring h below. All the tricks suggested in the SS algorithm can be used here as well. In fact, our implementations fully incorporate these optimizations.

Below we assume that all Gröbner basis computations are done using the ordering $\prec_{X,U}$.

Algorithm PGBMain(E, N, F)

Input: (E, N, F): E, N, finite subsets of $k[U]$; F, a finite subset of $k[U, X]$.

Output: a finite set of 3-tuples (E_i, N_i, G_i) such that $\{(V(E_i) \setminus V(N_i), G_i)\}$ constitutes a **minimal** comprehensive Gröbner system of F on $V(E) \setminus V(N)$.

```
1. If inconsistent$(E, N)$, then return $\emptyset$.
2. Otherwise, $G :=$ ReducedGröbnerBasis$(F \cup E)$.
3. If $1 \in G$, then return $\{(E, N, \{1\})\}$.
4. Let $G_r := G \cap k[U]$.
5. If inconsistent$(E, G_r \times N)$, then $\mathcal{PGB} := \emptyset$, else $\mathcal{PGB} := \{(E, G_r \times N, \{1\})\}$.
6. If inconsistent$(G_r, N)$, then return $\mathcal{PGB}$.
7. Otherwise, let $G_m :=$ MDBasis$(G \setminus G_r)$.
8. if consistent$(G_r, N \times \{h\})$, then $\mathcal{PGB} := \mathcal{PGB} \cup \{(G_r, N \times \{h\}, G_m)\}$, where $h = \mathrm{lcm}\{h_1, \ldots, h_k\}$,
   $h_i = \mathrm{lc}_X(g_i)$ and $g_i \in G_m$.
9. Return $\mathcal{PGB} \cup \bigcup_{h_i \in [h_1, \ldots, h_k]}$ PGBMain$(G_r \cup \{h_i\}, N \times \{h_1 h_2 \cdots h_{i-1}\}, G \setminus G_r)$.
```

In the above algorithm, $A \times B = \{fg \mid f \in A, g \in B\}$. Inconsistent (E, N) checks whether $V(E) \setminus V(N)$ is empty; below we discuss a number of algorithms to perform this check. If $i = 1$, then $N \times \{h_1 h_2 \cdots h_{i-1}\} = N$.

Theorem 4.6. *Algorithm PGBMain terminates.*

Proof. Using König's Lemma, it suffices to show that (1) in each step, the algorithm only creates finite branches; (2) each branch terminates after finite steps. (1) follows from the fact that in each call of PGBMain the number of polynomials in G_m is finite. Since $h_i = \mathrm{lc}_X(g) \in k[U]$ for some $g \in G$, where G is a *reduced* Gröbner basis for $\langle F \cup E \rangle \subset k[U, X]$ and $G_r = G \cap k[U]$, $h_i \notin \langle G_r \rangle \subset k[U]$, as otherwise, the polynomial $g \in G$ can be simplified further by G_r. In the next recursive call of PGBMain, the input $E' = G_r \cup \{h_i\}$ whose ideal is strictly larger than $\langle E \rangle$ from the previous call of PGBMain. So each branch terminates after finite steps. □

As should be evident from the discussion of the algorithm, a branch is never generated for the case when (E_i, N_i) is inconsistent. Further, the constructible sets are disjoint by construction. More importantly, branching is done only based on the leading coefficients of $G_m = \mathrm{MDBasis}(G \setminus G_r)$, instead of the whole $G \setminus G_r$, which is typically much smaller in size than $G \setminus G_r$. As a result, the number of branches generated by the above algorithm is strictly smaller than that of the branches in Suzuki–Sato's algorithm.

Efficient heuristics are employed to perform the consistency check; as a last resort only when other heuristics do not work, Rabinovitch's trick is employed for consistency check by introducing a new variable. As confirmed by experimental data discussed in Section 8, this general check is rarely needed. Because of these optimizations, the proposed algorithm has a much better performance than Suzuki–Sato's algorithm as well as Nabeshima's speed-up algorithm, as experimentally shown in Section 8.

4.2. Computing a comprehensive Gröbner Basis

Along with an algorithm for computing a comprehensive Gröbner system for a parametric polynomial system, Weispfenning (1992) also defined a comprehensive Gröbner basis for the polynomial system and gave an algorithm for computing it from the comprehensive Gröbner system.

Suzuki and Sato (2006) showed how a comprehensive Gröbner basis can be computed from a comprehensive Gröbner system using a new variable. The same idea can be used to adapt the above algorithm for computing a comprehensive Gröbner basis.

In Kapur et al. (2011), we have developed a modification of the algorithm proposed in this paper using which both a comprehensive Gröbner system as well as a faithful comprehensive Gröbner basis of a parametric polynomial system can be computed simultaneously. The ideas used in that paper for computing a comprehensive Gröbner basis can be adapted to other algorithms for computing comprehensive Gröbner systems as well.

5. Consistency of parametric constraints

As should be evident from the above description of the algorithm, there are two main computational steps which are being repeatedly performed: (i) Gröbner basis computations, and (ii) checking consistency of parametric constraints. As stated above, a parametric constraint (E, N), $E, N \subset k[U]$ is inconsistent if and only if for each $f \in N$, f is in the radical ideal of $\langle E \rangle$. This section discusses heuristics we have integrated into the implementation of the algorithm for the check whether $(E, \{f\})$ is inconsistent. In this section, we always assume that E itself is a Gröbner basis.

A general method to check whether $f \in \sqrt{\langle E \rangle}$ is to introduce a new variable y and compute the Gröbner basis G_y of $\langle E \cup \{fy - 1\} \rangle \subset k[U, y]$ for any admissible monomial order. If $G_y = \{1\}$, then $f \in \sqrt{\langle E \rangle}$ and $(E, \{f\})$ is inconsistent. Otherwise, $(E, \{f\})$ is consistent. However, this method can be, in general, very expensive partly because of introduction of a new variable. Consequently, this method is used only as a last resort when other heuristics fail.

The first heuristic is to check whether f is in the ideal generated by E; since in the algorithm, a Gröbner basis of E is already available, the normal form of f is computed; if it is 0, then f is in the ideal of E implying that $(E, \{f\})$ is inconsistent. This heuristic turns out to be quite effective as shown from experimental results in Section 8.

In case f is not in the ideal generated by E different heuristics are used for checking radical ideal membership, depending upon whether E is 0-dimensional or not. In case E is 0-dimensional, the method discussed in the next subsection for the radical membership check is complete, i.e., it decides whether f is in the radical ideal of E or not. In case E is of positive dimension, then independent variables of E are assigned random values, hopefully, resulting in a 0-dimensional ideal, for which the radical membership check can be done. However, this heuristic is not complete. If this heuristic cannot determine whether $(E, \{f\})$ is inconsistent, then another heuristic is employed that checks whether f^{2^k} is in the ideal of E for a suitably small value of k.

5.1. Ideal(E) is 0-dimensional

For the case when E is 0-dimensional, linear algebra techniques can be used to check the radical membership in E. The main idea is to compute the characteristic polynomial of the linear map associated with f, which can be efficiently done using a Gröbner basis of E.

Let $A = k[U]/\langle E\rangle$. Consider the map induced by $f \in k[U]$: $m_f : A \longrightarrow A$, $[g] \longmapsto [fg]$, where $g \in k[U]$ and $[g]$ is its equivalence class in A.

See Cox et al. (2005); Sun and Wang (2009) for the proofs of the following lemmas.

Lemma 5.1. *Assume that the map m_f is defined as above. Then,*

(1) *m_f is the zero map exactly when $f \in \langle E\rangle$.*
(2) *For a univariate polynomial q over k, $m_{q(f)} = q(m_f)$.*
(3) *$p_f(f) \in \langle E\rangle$, where p_f is the characteristic polynomial of m_f.*

Lemma 5.2. *Let $p_f \in k[\lambda]$ be the characteristic polynomial of m_f. Then for $\alpha \in L$, the following statements are equivalent.*

(1) *α is a root of the equation $p_f(\lambda) = 0$.*
(2) *α is a value of the function f on $V(E)$.*

Using these lemmas, we have:

Proposition 5.3. *Let $p_f \in k[\lambda]$ be the characteristic polynomial of m_f and $d = \deg(p_f)$.*

(1) *$p_f = \lambda^d$ if and only if $f \in \sqrt{\langle E\rangle}$.*
(2) *$p_f = q$ and $\lambda \nmid q$ if and only if there exists $g \in k[U]$ such that $gf \equiv 1 \bmod \langle E\rangle$.*
(3) *$p_f = \lambda^{d'}q$, where $0 < d' < d$ and $\lambda \nmid q$ if and only if $f \notin \sqrt{\langle E\rangle}$ and there exists $g \notin \sqrt{\langle E\rangle}$ such that $fg \in \sqrt{\langle E\rangle}$.*

Proof. (1) $\Rightarrow$) If $p_f = \lambda^d$, then $p_f(f) = f^d \in \langle E\rangle$ by Lemma 5.1, which shows $f \in \sqrt{\langle E\rangle}$. $\Leftarrow$) Since $f \in \sqrt{\langle E\rangle}$, 0 is the sole value of the function f on $V(E)$. By Lemma 5.2, $p_f = \lambda^d$.

(2) $\Rightarrow$) If $p_f = q$ and $\lambda \nmid q$, then there exist $a, b \in k[\lambda]$ such that $a\lambda + bp_f = 1$. Substitute λ by f. Then obtain $a(f)f + b(f)p_f(f) = 1$. $p_f(f) \in \langle E\rangle$ shows $a(f)f \equiv 1 \bmod \langle E\rangle$. $\Leftarrow$) If there exists $g \in k[U]$ such that $gf \equiv 1 \bmod \langle E\rangle$, then all the values of the function f on $V(\{f\})$ are not 0, which means the roots of $p_f(\lambda) = 0$ are not 0 as well by the above lemma. So $\lambda \nmid p_f$.

(3) $\Rightarrow$) If $p_f = \lambda^{d'}q$, where $0 < d' < d$ and $\lambda \nmid q$, then we have $f \notin \sqrt{\langle E\rangle}$ by (1). By Lemma 5.1, $p_f(f) = f^{d'}q(f) \in \langle E\rangle$, and hence, $fq(f) \in \sqrt{\langle E\rangle}$. It remains to show $q(f) \notin \sqrt{\langle E\rangle}$. We prove this by contradiction. If $q(f) \in \sqrt{\langle E\rangle}$, then there exists an integer $c > 0$ such that $q^c(f) \in \langle E\rangle$, which implies $m_{q^c(f)} = q^c(m_f) = 0$. Thus, q^c is a multiple of the minimal polynomial of m_f and hence all the irreducible factors of p_f should be factors of q^c. But this contradicts with $\lambda \nmid q$. $\Leftarrow$) Since $f, g \notin \sqrt{\langle E\rangle}$ and $fg \in \sqrt{\langle E\rangle}$, both f and g are nonzero functions on $V(E)$, but fg is a zero function on $V(E)$. This implies that f vanishes on some but not all points of $V(E)$. By Lemma 5.2, $p_f = \lambda^{d'}q$, where $0 < d' < d$ and $\lambda \nmid q$. □

For the case (2) of Proposition 5.3, clearly $V(E) \setminus V(\{f\}) = V(E)$ holds. For the case (3), it is easy to check $V(E) \setminus V(\{f\}) = V(E \cup \{q(f)\})$ by Lemma 5.2. So the parametric constraint $(E, \{f\})$ is equivalent to $(E \cup \{q(f)\}, \{1\})$, which converts the disequality constraint into equality constraint. Both (2) and (3) contribute to the speeding of the implementation of the new algorithm.

If E is zero-dimensional, then $k[U]/\langle E\rangle$ is a finite vector space and the characteristic polynomial of m_f can be generated in Cox et al. (2005). Since in our algorithm, E itself is a Gröbner basis, the complexity of doing radical membership check is of polynomial time, which is much more efficient than the general method based on Rabinovitch's trick.

The following algorithm is based on the above theory:

Algorithm 1 — Zero-DimCheck

```
Input:  $(E, \{f\})$: $E$ is the Gröbner basis of the zero dimensional ideal $\langle E\rangle$; $f$, a polynomial in
        $k[U]$.
Output: true (consistent) or false (inconsistent).

begin
    $p_f \leftarrow$ characteristic polynomial of $m_f$ defined on $k[U]/\langle E\rangle$
    $d \leftarrow \deg(p_f)$
    if $p_f \neq \lambda^d$ then return true else return false end if
end
```

5.2. Ideal(E) is of positive dimension

We discuss two heuristics, *CCheck* and *ICheck*, for radical membership check; neither one is complete.

A subset V of U is *independent* modulo the ideal I if $k[V] \cap I = \{0\}$. An independent subset of U is maximal if there is no independent subset containing V properly.

The following proposition is well-known.

Proposition 5.4. *Let $I \subset k[U]$ be an ideal and $\prec_U$ be a graded order on $k[U]$. If $k[V] \cap \mathrm{lpp}_U(I) = \emptyset$, then $k[V] \cap I = \emptyset$. Furthermore, the maximal independent subset modulo $\mathrm{lpp}_U(I)$ is also a maximal independent subset modulo I.*

A maximal independent subset modulo the monomial ideal of $\langle E\rangle$ can be easily computed; the above proposition thus provides a method to compute the maximal independent subset modulo an ideal.

The following theorem is obvious, so the proof is omitted.

Theorem 5.5. *Let $\langle E\rangle \subset k[U]$ with positive dimension, V be a maximal independent subset modulo $\langle E\rangle$, and $\bar{\alpha}$ be an element in k^l where l is the cardinality of V. If $f|_{V=\bar{\alpha}} \notin \sqrt{\langle E|_{V=\bar{\alpha}}\rangle}$, then $f \notin \sqrt{\langle E\rangle}$ i.e. $(E, \{f\})$ is consistent.*

Since V is a maximal independent subset modulo $\langle E\rangle$, the ideal $\langle E\rangle$ becomes a zero dimensional ideal in $k[U \setminus V]$ with probability 1 by setting V to a value in k^l randomly when the characteristic of k is 0. In this case, we can use the technique provided in the last subsection to check if $f|_{V=\bar{\alpha}} \notin \sqrt{\langle E|_{V=\bar{\alpha}}\rangle}$. If $(E|_{V=\bar{\alpha}}, f|_{V=\bar{\alpha}})$ is consistent, then $(E, \{f\})$ is consistent. This gives an algorithm for checking the consistence of $(E, \{f\})$. When $f \notin \sqrt{\langle E\rangle}$, this algorithm can detect it efficiently.

In the above algorithm, we only need to compute the Gröbner basis of $\langle E_{V=\bar{\alpha}}\rangle$ which is usually zero dimensional and has fewer variables. So *CCheck* is more efficient than the general method which needs to compute the Gröbner basis of $\langle E \cup \{fy-1\}\rangle$ whose dimension is positive.

If *CCheck*$(E, \{f\})$ returns true, then $(E, \{f\})$ is consistent. However, if *CCheck*$(E, \{f\})$ returns false, it need not be the case that $(E, \{f\})$ is inconsistent.

The following simple heuristic *ICheck* checks whether f^{2^k} is in the ideal generated by E by repeatedly squaring the normal form of f^{2^i} in an efficient way.

Clearly, if *ICheck*$(E, \{f\})$ returns true, then $(E, \{f\})$ is inconsistent.

5.3. Putting all together

The above discussed checks are done in the following order for checking the consistency of a parametric constraint $(E, \{f\})$. First check whether f is in the ideal of E; this check can be easily done by computing the normal form of f using a Gröbner basis of E which is readily available. If yes, then

Algorithm 2 — CCheck

```
Input:  $(E, \{f\})$: $E$ is the Gröbner basis of $\langle E\rangle$ w.r.t. a graded monomial order $\prec_U$; $f$, a
        polynomial in $k[U]$.
Output: true (consistent) or false .

begin
    $V \leftarrow$ independent variables of $\langle \mathrm{lpp}_U(E)\rangle$
    $\bar{\alpha} \leftarrow$ a random element in $k^l$
    $spE \leftarrow$ GröbnerBasis($E|_{V=\bar{\alpha}}, \prec_U$)
    if $\langle spE\rangle$ is zero dimensional in $k[U \setminus V]$ then
        if Zero-DimCheck($spE, f|_{V=\bar{\alpha}}$) =true then
            return true
        end if
    end if ;
    return false
end
```

Algorithm 3 — ICheck

```
Input:  $(E, \{f\})$: $E$ is the Gröbner basis of $\langle E\rangle$ w.r.t. a graded monomial order $\prec_U$; $f$, a
        polynomial in $k[U]$.
Output: true (inconsistent) or false .

begin
    $loops \leftarrow$ an integer given in advance
    $p \leftarrow f$
    for $i$ from 1 to $loops$ do
        $\{m_1, \ldots, m_l\} \leftarrow$ monomials of $p$
        $s \leftarrow 0$
        for $m \in \{m_1, \ldots, m_l\}$ do
            $s \leftarrow s +$NormalForm($p \cdot m, E$)
        end for
        if $s = 0$ then return true end if
        $p \leftarrow s$
    end for
    return false
end
```

the constraint is inconsistent. If no, then depending upon the dimension of the ideal of E, either *Zero-DimCheck* or *CCheck* is performed. If E is 0-dimensional, then the check is complete in that it decides whether the constraint is consistent or not. If E is of positive dimension then if *CCheck* returns true, the constraint is consistent; otherwise, *ICheck* is performed. If *ICheck* succeeds, then the constraint is inconsistent. Finally, the general check is performed by computing a Gröbner basis of $E \cup \{fy - 1 = 0\}$, where y is a new variable different from U.

6. An illustrative example

The proposed algorithm is illustrated on a simple example. We do not show some of the optimizations discussed in the next section so as not to complicate the discussion.

Example 6.1. Let $F = \{ax - b, by - a, cx^2 - y, cy^2 - x\} \subset \mathbb{Q}[a, b, c][x, y]$, with the block order $\prec_{X,U}$, $U \ll X$ where $U = \{a, b, c\}$ and $X = \{x, y\}$; within each block, $\prec_X$ and $\prec_U$ are graded reverse lexicographic orders with $y < x$ and $c < b < a$, respectively.

(1) We have $E = \emptyset$, $N = \{1\}$: the parametric constraint (E, N) is consistent. The reduced Gröbner basis of $\langle F\rangle$ w.r.t. $\prec_{X,U}$ is $G = \{x^3 - y^3, cx^2 - y, ay^2 - bc, cy^2 - x, ax - b, bx - acy, a^2y - b^2c, by - a, a^6 - b^6, a^3c - b^3, b^3c - a^3, ac^2 - a, bc^2 - b\}$; $G_r = G \cap \mathbb{Q}[a, b, c] = \{a^6 - b^6, a^3c - b^3, b^3c - a^3, ac^2 - a, bc^2 - b\}$. It is easy to see that (E, G_r) and $(E, G_r \times N)$ are consistent. This leads to the trivial branch of the comprehensive Gröbner system for F: $(\emptyset, G_r, \{1\})$.

(2) $G \setminus G_r = \{x^3 - y^3, cx^2 - y, ay^2 - bc, cy^2 - x, ax - b, bx - acy, a^2y - b^2c, by - a\}$; $G_m = \text{MDBasis}(G \setminus G_r) = \{bx - acy, by - a\}$. Further, $h = \text{lcm}\{\text{lc}_X(bx - acy), \text{lc}_X(by - a)\} = b$. This results in another branch of the comprehensive Gröbner system for F corresponding to the case when all polynomials in G_r are 0 and $b \neq 0$: $(G_r, \{b\}, G_m)$. Notice that $(G_r, \{b\})$ is consistent, which is detected using the *Zero-DimCheck*.

(3) The next case to consider is when $b = 0$. The Gröbner basis of $G_r \cup \{b\}$ is $\{a^3, ac^2 - a, b\}$. This is the input E' in the recursive call of PGBMain, with the other input being $N' = \{1\}$ and $F' = G \setminus G_r$. It is easy to see that (E', N') is consistent. The reduced Gröbner basis for $F' \cup E'$ is: $G' = \{x^3 - y^3, cx^2 - y, cy^2 - x, a, b\}$ of which $G'_r = \{a, b\}$. It is easy to check the parametric constraint (E', G'_r) is inconsistent: the check for a being in the radical ideal of E' is confirmed by *ICheck*; b is in the ideal of E'. So no branch is generated from this case.

$G'_m = \text{MDBasis}(G' \setminus G'_r) = \{cx^2 - y, cy^2 - x\}$ and $h' = \text{lcm}\{\text{lc}_X(cx^2 - y), \text{lc}_X(cy^2 - x)\} = c$. This results in another branch: $(G'_r, \{c\}, G'_m)$.

(4) For the case when $h' = c = 0$, $E'' = \{a, b, c\}$ is the Gröbner basis of $G'_r \cup \{c\}$ and $N'' = \{1\}$, $F'' = \{x^3 - y^3, cx^2 - y, cy^2 - x\}$. The Gröbner basis for $F'' \cup E''$ is $G'' = \{x, y, a, b, c\}$. Then $G''_r = \{a, b, c\}$ and $G''_m = \{x, y\}$. Since $h'' = \text{lcm}\{\text{lc}_X(x), \text{lc}_X(y)\} = 1$, this gives another branch: $(G''_r, \{1\}, G''_m)$. As $h'' = 1$, no other branches are created and the algorithm terminates.

The result is a comprehensive Gröbner system for F:

$$\begin{cases} \{1\}, & \text{if } (a^6 - b^6 \neq 0 \text{ or } a^3c - b^3 \neq 0 \text{ or } b^3c \\ & -a^3 \neq 0 \text{ or } ac^2 - a \neq 0 \text{ or } bc^2 - b \neq 0), \\ \{bx - acy, by - a\}, & \text{if } (a^6 - b^6 = a^3c - b^3 = b^3c - a^3 \\ & = ac^2 - a = bc^2 - b = 0) \text{ and } (b \neq 0), \\ \{cx^2 - y, cy^2 - x\} & \text{if } (a = b = 0) \text{ and } (c \neq 0), \\ \{x, y\} & \text{if } (a = b = c = 0). \end{cases}$$

7. Some optimizations

In this section, we briefly discuss some optimizations used in the implementation of the proposed algorithm in the Magma computer algebra system.

7.1. Choosing a minimal Dickson basis

As discussed above, Step 7 in the algorithm proposed in Section 4.1 chooses a minimal Dickson basis of $G \setminus G_r$; this subset is usually not unique. The leading coefficients of the polynomials in G_m thus chosen are subsequently used as equality constraint along other branches in the algorithm. We have experimented with a strategy in which a polynomial with the same leading power product but *"simpler"* leading coefficient is preferred. One notion of simpler we have experimented with is to minimize the degree of maximal factors of the leading coefficient. For example, let $G \setminus G_r = \{(a^3 + a + 1)x + b, (a^4 - 1)x - c, by + 1\}$ where x, y are variables and a, b, c are parameters. Notice $G_{m1} = \{(a^3 + a + 1)x + b, by + 1\}$ and $G_{m2} = \{(a^4 - 1)x - c, by + 1\}$ are both minimal Dickson bases of $G \setminus G_r$. Our strategy is to choose G_{m2} instead of G_{m1}, since among the factors of $a^4 - 1$, $a^2 + 1$ is of the highest degree and it is of lower degree than the irreducible coefficient $a^3 + a + 1$, even though the degree of the leading coefficient $a^4 - 1$ is higher.

Another heuristic we plan to experiment with is whether there is a common factor among the leading coefficients of polynomials which are candidates for inclusion in minimal Dickson basis. In the example discussed in Section 6, in Step 2, $\{bx - acy, by - a\}$ is selected from $\{x^3 - y^3, cx^2 - y, ay^2 - bc, cy^2 - x, ax - b, bx - acy, a^2y - b^2c, by - a\}$ instead of $\{ax - b, by - a\}$.

An optimization introduced in Manubens and Montes (2009) for reducing the number of branches in the output is to combine the branches corresponding to Gröbner bases which have the same leading power products, much like it has been done for the branch corresponding to the Gröbner basis $\{1\}$. This heuristic/optimization is likely to be more useful if quotient ideals are used to process parametric constraints by using disequality constraints as well (see discussion below). In the example discussed in Section 6, branches 2 and 4 will be combined since their Gröbner bases have the same leading power products $\{x, y\}$.

7.2. *Computing quotient ideals of parametric equality constraints with a parametric disequality constraint*

As should be evident from the algorithm in Section 4.1, it is often necessary to compute a Gröbner basis of the ideal $\langle F \cup E\rangle$; this computation is also the most time consuming in the algorithm. Because of the special role of parametric constraints, it is often possible to use simple algebraic functions to make constraints in E of lower degree. Some of the possibilities are replacing a parametric constraint in E by its square-free part and/or factoring the parametric constraint. Further, disequality constraints in N can be used to simplify E by computing *quotient ideals* since

$$V(E) \setminus V(N) = V(\langle E\rangle : \langle N\rangle) \setminus V(N) \quad \text{and} \quad V(E) \setminus V(\{f\}) = V(\langle E\rangle : f^{\infty}) \setminus V(\{f\}).$$

This technique is also used in Montes and Wibmer (2010); the computation of a saturation ideal is a special case of the quotient ideal computation when N contains only a single polynomial. It is hoped that the generators of $\langle E\rangle : \langle N\rangle$ are “simpler” than the elements in E. For example, Let $E = \{a^6 - b^6, a^3c - b^3, b^3c - a^3, ac^2 - a, bc^2 - b\}$ and $N = \{b\}$. Notice that $V(E) \setminus V(N)$ is the constructible set corresponding to the branch obtained in (2) of Example 6.1. The generators of the quotient ideal $\langle E\rangle : \langle N\rangle$ are $\{a^3 - b^3c, c^2 - 1\}$, which is simpler than the set E.

Computation of quotient ideals is most useful if the generators of the quotient ideal $\langle E\rangle : \langle N\rangle$ are simpler than E. In that case, the Gröbner basis computation for $(\langle E\rangle : \langle N\rangle) + \langle F\rangle$ is often more efficient than for $\langle E \cup F\rangle$. We however observed in many examples that particularly when E is complicated, the two ideals $\langle E\rangle : \langle N\rangle$ and $\langle E\rangle$ are identical; as a result, computing quotient ideals does not help at all. Because of these reasons, we have used a heuristic of computing quotient ideals only if the size of E is <5, or elements of E are of low degree (no more than the total degree 3). These heuristics need to be further experimented with on a larger set of examples and their effectiveness should be analyzed.

7.3. *Further exploiting 0-dimensionality of equality constraints*

When the equality constraints constitute a 0-dimensional ideal, a more efficient algorithm for computing comprehensive Gröbner systems which is a variation over the proposed algorithm is discussed below. This is based on the following proposition, which is a direct consequence of Theorem 3.3 in Kalkbrener (1997), which can be given that minimizes the number of Gröbner basis computations.

Proposition 7.1. *Let G be a Gröbner basis of the ideal $\langle F\rangle \subset k[U, X]$ w.r.t. an admissible block order with $U \ll X$. Let $G_r = G \cap k[U]$. If $\langle G_r\rangle \subset k[U]$ is a radical 0-dimensional ideal, then $\sigma(G \setminus G_r)$ is a Gröbner basis of $\langle \sigma(F)\rangle \subset L[X]$ for any specialization σ from $k[U]$ to L such that $\sigma(g) = 0$ for all $g \in G_r$.*[3]

This proposition is also mentioned in Suzuki and Sato (2006). Using this proposition and Theorem 4.3, an efficient algorithm to compute a minimal comprehensive Gröbner system is presented below.

Algorithm Zero-Dim(E, N, F)

Input: (E, N, F): E, N, finite subsets of $k[U]$ and $\langle E\rangle$ is a zero dimensional ideal in $k[U]$; F, a finite subset of $k[U, X]$.

Output: a finite set of 3-tuples (E_i, N_i, G_i) such that $\{(V(E_i) \setminus V(N_i), G_i)\}$ constitute a **minimal** comprehensive Gröbner system of F on $V(E) \setminus V(N)$.

[3] In fact, the set $G \setminus G_r$ is a comprehensive Gröbner basis for F on G_r.

```
1. If inconsistent($E, N$), then return $\emptyset$.
2. Let $G :=$ ReducedGröbnerBasis($F \cup radE$), where $radE$ is the Gröbner basis of the radical ideal of $\langle E \rangle$.
3. If $1 \in G$, then return $\{(radE, N, \{1\})\}$.
4. Let $G_r := G \cap k[U]$.
5. If inconsistent($radE, G_r \times N$), then $\mathcal{PGB} := \emptyset$, else $\mathcal{PGB} := \{(radE, G_r \times N, \{1\})\}$.
6. If inconsistent($G_r, N$), then return $\mathcal{PGB}$.
7. Return $\mathcal{PGB} \cup$ Split($G_r, N, G \setminus G_r$).
```

The reader would notice that Steps 1–6 of the above algorithm and *PGBMain* are identical. Steps 7–9 of *PGBMain* have been replaced by Step 7 in which *Split* is called, exploiting the 0-dimensionality of G_r. By Proposition 7.1, $\sigma(G \setminus G_r)$ is a Gröbner basis of $\langle \sigma(F) \rangle \subset L[X]$ for any specialization σ from $k[U]$ to L such that $\sigma(g) = 0$ for all $g \in G_r$. However, $(V(G_r) \setminus V(N), G \setminus G_r)$ is not a branch of minimal comprehensive Gröbner system for F, since the leading coefficients of polynomials in $G \setminus G_r$ may be zero under the specializations. Note that no extra Gröbner basis computations in $k[U, X]$ are needed in *Split* discussed below.

Algorithm Split(E, N, F)

Input: (E, N, F): E, N, finite subsets of $k[U]$ and $\langle E \rangle$ is a radical zero dimensional ideal in $k[U]$; F, a finite subset of $k[U, X]$ such that $\sigma(F)$ is a Gröbner basis in $L[X]$ for any specialization σ from $k[U]$ to L such that $\sigma(g) = 0$ for all $g \in E$.

Output: a finite set of 3-tuples (E_i, N_i, G_i) such that $\{(V(E_i) \setminus V(N_i), G_i)\}$ constitute a **minimal** comprehensive Gröbner system of F on $V(E) \setminus V(N)$.

```
1. If inconsistent($E, N$), then return $\emptyset$.
2. Otherwise, let $G := \{$the normal form of $f$ w.r.t. Gröbner basis of $\langle E \rangle \mid f \in F\}$.
3. Let $G_m :=$ MDBasis($G$).
4. If consistent($E, N \times \{h\}$), then $\mathcal{PGB} := \mathcal{PGB} \cup \{(E, N \times \{h\}, G_m)\}$, where $h = \mathrm{lcm}\{h_1, \ldots, h_k\}$,
   $h_i = \mathrm{lc}_X(g_i)$ and $g_i \in G_m$.
5. Return $\mathcal{PGB} \cup \bigcup_{h_i \in [h_1, \ldots, h_k]}$ Split($E \cup \{h_i\}, N \times \{h_1 h_2 \cdots h_{i-1}\}, G$).
```

For checking consistency in Step 1, techniques discussed in Section 5 can be used as well in the above algorithms. Since $\langle E \rangle$ is a 0-dimensional ideal, a disequality constraint f from N can be used to compute E' such that $V(E') = V(E) \setminus V(\{f\})$. As in Section 5, let $p_f \in k[\lambda]$ be the characteristic polynomial of the map m_f induced by f and defined on $k[U]/\langle E \rangle$. Using Proposition 5.3, if $p_f = \lambda^{d'} q$, $0 \le d' \le deg(p_f)$ and $\lambda \nmid q$, then $V(E \cup \{q(f)\}) = V(E) \setminus V(\{f\})$, even when $q = 1$ or $q = p_n$, where $q(f)$ is the polynomial obtained by substituting f for λ. The radical ideal of $\langle E \rangle$ can be computed similarly as follows: let $p_{x_i} \in k[\lambda]$ be the characteristic polynomial of m_{x_i} defined on $k[U]/\langle E \rangle$. Then $p_{x_i}(x_i) \in \langle E \rangle$. By Proposition 2.7 of Cox et al. (2005), $\langle E \cup \{q_{x_1}(x_1), \ldots, q_{x_n}(x_n)\} \rangle$ is the radical ideal of $\langle E \rangle$, where $q_{x_i}(x_i)$ is the square-free part of $p_{x_i}(x_i)$.

8. Implementation and comparative performance

The proposed algorithm has been implemented on the computer algebra system *Magma*.[4] The implementation has been experimented on a number of examples from different application domains including geometry theorem proving and computer vision, and it has been compared with implementations of other algorithms. Since the proposed algorithm is able to avoid most unnecessary branches and computations, it is efficient and can compute comprehensive Gröbner systems for most problems in a few seconds. In particular, we have been successful in solving a particular case of the famous P3P problem for pose-estimation from computer vision, which is investigated by Gao et al. (2003) using the characteristic set method; see the polynomial system below.

[4] Software is available at http://www.mmrc.iss.ac.cn/~dwang/. We also implemented a simple version of the proposed algorithm in Singular, in which Theorem 4.3 is applied, but the techniques for checking consistency introduced in Section 5 and optimizations in Section 7.3 are not supported.

The following table shows a comparison of our implementations on Magma with other existing algorithms for computing comprehensive Gröbner systems, including: Suzuki–Sato's algorithm and Nabeshima's speed-up version both implemented by Nabeshima (2007) (package PGB, ver20090915) in Risa/Asir, Suzuki–Sato algorithm's implemented by Suzuki (ver20091102) in Singular, Montes' function "cgsdr" which is the first step of the Gröbner cover algorithm (package grobcov.lib) in Singular[5] and the function "gsys" for computing comprehensive Gröbner system (package RedLog) in Reduce. The versions of Risa/Asir, Magma, Singular and Reduce are ver20090715, v2.12–16, ver3-1-2 and free CSL version, respectively.

We tried all the implementations on Examples F6 and F8 from Nabeshima (2007) and Examples E4 and E5 from Montes and Recio (2007). Many other examples that were tried could be solved in little time. To generate complex examples, we modified problems F3, F4, F5, F6 and F8 in Nabeshima (2007), and labeled them as S1, S2, S3, S4 and S5. As stated above, we also tried the famous P3P problem from computer vision. The polynomials for all these problems are given below:

F6: $F = \{x^4 + ax^3 + bx^2 + cx + d, 4x^3 + 3ax^2 + 2bx + c\}, X = \{x\}, U = \{a, b, c, d\}$;

F8: $F = \{ax^2 + by, cw^2 + z, (x - z)^2 + (y - w)^2, 2dxw - 2by\}, X = \{x, y, z, w\}, U = \{a, b, c, d\}$;

E4: $F = \{(a - 1)y2 - b(x2 - 1), (a - 1)(x2 + 1) + by2, (a + 1)y3 - b(x3 + 1), (a + 1)(x3 - 1) + by3, (x3 - a)^2 + y3^2 - (x2 - a)^2 - y2^2\}, X = \{x2, x3, y2, y3\}, U = \{a, b\}$;

E5: $F = \{(x1 - a)^2 + (y1 - 1)^2 - a^2 - 1, (x2 + b)^2 + (y2 - 1)^2 - b^2 - 1, a(x1 - a) + (y1 - 1) + (a^2 + 1)cv, -b(x2 + b) + (y2 - 1) + (1 + b^2)cw, a(y1 - 1) - (x1 - a) + (a^2 + 1)sv, -b(y2 - 1) - (x2 + b) + (b^2 + 1)sw, x1y2 - 2x1 - x2y1 + 2x2, cv^2 + sv^2 - 1, cw^2 + sw^2 - 1\}, X = \{x1, y1, x2, y2\}, U = \{a, b, sv, cv, sw, cw\}$;

S1: $F = \{ax^4 + cx^2 + y, bx^3 + x^2 + 2, cx^2 + dx + y\}, X = \{x, y\}, U = \{a, b, c, d\}$;

S2: $F = \{ax^3y + cxy^2 + bx + y, x^4y + 3dy, cx^2 + bxy, x^2y^2 + ax^2, x^5 + y^5\}, X = \{x, y\}, U = \{a, b, c, d\}$;

S3: $F = \{ax^2y + bx^2 + y^3, ax^2y + bxy + cy^2, ay^3 + bx^2y + cxy\}, X = \{x, y\}, U = \{a, b, c\}$;

S4: $F = \{x^4 + ax^3 + bx^2 + cxy + d, 4x^3 + 3ax^2y + 2bx + c + y\}, X = \{x\}, U = \{a, b, c, d\}$;

S5: $F = \{ax^2 + byz + czw, cw^2 + by + z, (x - z)^2 + (y - w)^2, 2dxw - 2byz\}, X = \{x, y, z, w\}, U = \{a, b, c, d\}$;

P3P: $F = \{(1-a)y^2 - ax^2 - py + arxy + 1, (1-b)x^2 - by^2 - qx + brxy + 1\}, X = \{x, y\}, U = \{p, q, r, a, b\}$.

For all the examples, the term orders used on X are graded reverse lexicographic orders.

In Table 1, the entry labeled with *New*(M) is the proposed algorithm implemented in Magma; (S), (R) and (A) stand for Singular, Reduce and Risa/Asir, respectively. The label "*error*" is included if an implementation ran out of memory or broke down. The timings were obtained by running the implementations on Core i5 4×2.8 GHz with 4 GB Memory running Windows 7.

As is evident from Table 1, the proposed algorithm usually generates fewer branches, which is an important reason for the better performance of the proposed algorithm in contrast to other algorithms.

An efficient check for the consistency of parametric constraints is important for the performance of the proposed algorithm as well. The role of various checks discussed in Section 5 was investigated in detail. This is reported in Table 2 and Fig. 1, where *Tri*, 0-*dim*, *C*, *I*, and *Gen* stand, respectively, for the *trivial check*, *Zero-DimCheck*, *CCheck*, *ICheck*, and the *general method*.

About 59% of the consistency check is settled by the trivial check that a polynomial is in the ideal; about the remaining 39% of the consistency check is resolved by the *Zero-DimCheck*, *CCheck* and *ICheck*. The general method for checking consistency using Rabinovitch's trick of introducing a new variable is rarely used (almost 2%). We also tested the above examples by using only the trivial check and the general method. Table 3 shows that the new checks are especially helpful for complex examples; typically, they are more efficient than the general method.

Even though the proposed algorithm performs quite well on a large class of parametric polynomial systems, there are many problems still beyond its reach. A case in point is the fully general version

[5] The package grobcov.lib (Beta version) is downloaded from http://www-ma2.upc.edu/~montes/.

Table 1
Timings.

Exa.	Algorithm	Br.	Time (s)	Exa.	Algorithm	Br.	Time (s)
F6	New(M)	8	0.062	F8	New(M)	18	0.140
	SuzukiSato(S)	–	error		SuzukiSato(S)	821	0.66
	Montes(S)	8	0.245		Montes(S)	18	5.110
	gsys(R)	8	0.105		gsys(R)	54	278.846
	SuzukiSato(A)	875	29.33		SuzukiSato(A)	–	>1 h
	Nabeshima(A)	17	0.078		Nabeshima(A)	–	>1 h
E4	New(M)	6	0.031	E5	New(M)	22	0.374
	SuzukiSato(S)	26	0.032		SuzukiSato(S)	181 555	160.652
	Montes(S)	12	0.310		Montes(S)	26	1.125
	gsys(R)	12	0.565		gsys(R)	41	0.219
	SuzukiSato(A)	15	0.078		SuzukiSato(A)	98	16.72
	Nabeshima(A)	24	0.640		Nabeshima(A)	102	8.923
S1	New(M)	11	0.031	S2	New(M)	15	7.301
	SuzukiSato(S)	24	0.020		SuzukiSato(S)	–	error
	Montes(S)	27	1.365		Montes(S)	–	error
	gsys(R)	41	0.175		gsys(R)	155	758.749
	SuzukiSato(A)	14	0.046		SuzukiSato(A)	–	error
	Nabeshima(A)	8	0.047		Nabeshima(A)	–	>1 h
S3	New(M)	18	1.217	S4	New(M)	24	0.920
	SuzukiSato(S)	–	error		SuzukiSato(S)	–	error
	Montes(S)	–	error		Montes(S)	–	error
	gsys(R)	48	315.340		gsys(R)	–	>1 h
	SuzukiSato(A)	–	>1 h		SuzukiSato(A)	–	>1 h
	Nabeshima(A)	–	>1h		Nabeshima(A)	–	>1 h
S5	New(M)	23	1.498	P3P	New(M)	36	2.527
	SuzukiSato(S)	–	error		SuzukiSato(S)	–	error
	Montes(S)	–	error		Montes(S)	30	2.660
	gsys(R)	–	> 1h		gsys(R)	–	>1 h
	SuzukiSato(A)	–	>1 h		SuzukiSato(A)	–	>1 h
	Nabeshima(A)	–	>1 h		Nabeshima(A)	–	error

Table 2
Info about various consistence checks.

Exa.	Tri.	0-dim	pos-dim		Gen.	Total
			C.	I.		
F6	20	0	7	3	4	34
F8	45	0	29	0	0	74
E4	8	2	3	0	0	13
E5	36	2	34	0	0	72
S1	15	0	17	0	0	32
S2	93	1	37	1	0	132
S3	44	0	27	0	4	75
S4	44	4	33	0	0	81
S5	63	0	39	1	0	103
P3P	84	2	53	0	9	148

of the P3P problem; its polynomial system is presented below; neither our implementation nor other implementations we experimented with are able to produce any output in an hour of execution time.

$$\text{P3P}\mathit{general}: \quad F = \{y^2 + z^2 - pyz - a, z^2 + x^2 - qxz - b, x^2 + y^2 - rxy - c\},$$

where $X = \{x, y, z\}$, $U = \{p, q, r, a, b, c\}$.

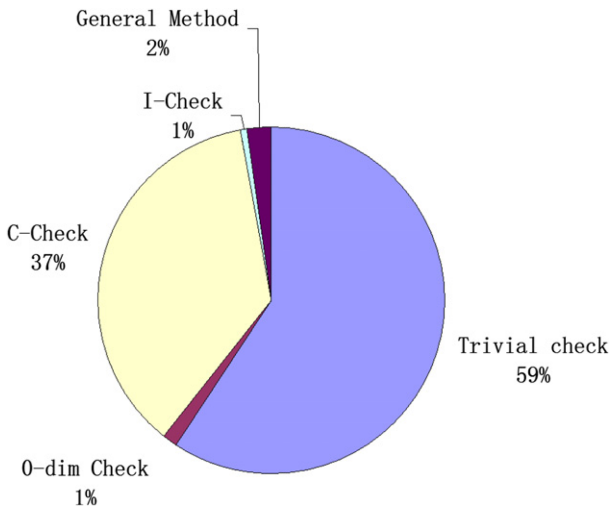

Fig. 1. Percentage of checks.

Table 3
Info about various consistence checks.

Exa.	F6	F8	E4	E5	S1	S2	S3	S4	S5	P3P
With New Checks	0.062	0.140	0.031	0.374	0.031	7.301	1.217	0.920	1.498	2.527
Gen. method only	0.062	0.156	0.031	0.406	0.031	8.377	1.264	1.014	2.590	4.836

9. Concluding remarks

A new algorithm for computing a comprehensive Gröbner system has been proposed using ideas from papers by Kalkbrener, Weispfenning, Suzuki and Sato. Preliminary experiments suggest that the algorithm is superior in practice in comparison to other existing algorithms; an experimental comparative analysis is reported in the paper. New techniques for checking the consistency of polynomial equality and disequality constraints are proposed as well. Experimental results show these new techniques are more efficient than the general method based on Rabinovitch's trick, especially in complicated examples. Particularly, we have been able to make substantial progress in attacking problems such as the famous P3P problem from computer vision, which have been found extremely difficult to solve using most symbolic computation algorithms.

We believe that the proposed algorithm can be further improved. We are exploring conditions under which the radical membership ideal check (i.e., consistency check) is either unwarranted as well as whenever needed, can be done efficiently. Using insights developed during this research, we are also exploring new methods for computing a comprehensive Gröbner basis of a parametric polynomial system.

Acknowledgement

The first author is supported by the National Science Foundation award CCF-0729097 and the last two authors are supported by NKBRPC 2011CB302400, NSFC 10971217 and 60821002/F02.

References

Becker, T., Weispfenning, V., Kredel, H., 1993. Gröbner Bases, a Computational Approach to Commutative Algebra. Springer-Verlag, London.

Chen, C., Golubitsky, O., Lemaire, F., Moreno Maza, M., Pan, W., 2007. Comprehensive triangular decomposition. In: Proceedings of CASC'07. In: Lect. Notes in Comp. Sci., vol. 4770. Springer, Berlin, pp. 73–101.
Chen, X.F., Li, P., Lin, L., Wang, D.K., 2004. Proving geometric theorems by partitioned-parametric Gröbner bases. In: Proceeding of Automated Deduction in Geometry (ADG). In: Lect. Notes in Comp. Sci., vol. 3763. Springer, Berlin, pp. 34–43.
Cox, D., Little, J., O'Shea, D., 2005. Using Algebraic Geometry, 2nd edition. Springer, New York.
Donald, B., Kapur, D., Mundy, J.L. (Eds.), 1992. Symbolic and numerical computation for artificial intelligence. In: Computational Mathematics and Applications. Academic Press Ltd, London.
Gao, X.S., Chou, S.C., 1992. Solving parametric algebraic systems. In: Proceedings of ISSAC'1992. ACM Press, New York, pp. 335–341.
Gao, X.S., Hou, X., Tang, J., Chen, H.F., 2003. Complete solution classification for the perspective-three-point problem. IEEE Trans. Pattern Anal. Mach. Intell. 25 (8), 930–943.
Kalkbrener, K., 1997. On the stability of Gröbner bases under specialization. J. Symbolic Comput. 24 (1), 51–58.
Kapur, D., 1995. An approach for solving systems of parametric polynomial equations. In: Saraswat, Van Hentenryck (Eds.), Principles and Practice of Constraint Programming. MIT Press, Cambridge.
Kapur, D., 2006. A quantifier-elimination based heuristic for automatically generating inductive assertions for programs. J. Syst. Sci. Complex. 19 (3), 307–330.
Kapur, D., Sun, Y., Wang, D.K., 2010. A new algorithm for computing comprehensive Gröbner systems. In: Proceedings of ISSAC'2010. ACM Press, New York, pp. 29–36.
Kapur, D., Sun, Y., Wang, D.K., 2011. Computing comprehensive Gröbner systems and comprehensive Gröbner bases simultaneously. Proceedings of ISSAC'2011 (in press).
Manubens, M., Montes, A., 2006. Improving DISPGB algorithm using the discriminant ideal. J. Symbolic. Comput. 41 (11), 1245–1263.
Manubens, M., Montes, A., 2009. Minimal canonical comprehensive Gröbner system. J. Symbolic. Comput. 44 (5), 463–478.
Montes, A., 2002. A new algorithm for discussing Gröbner basis with parameters. J. Symbolic. Comput. 33 (1–2), 183–208.
Montes, A., Recio, T., 2007. Automatic discovery of geometry theorems using minimal canonical comprehensive Gröbner systems. In: Proceeding of Automated Deduction in Geometry (ADG) 2006. In: Lecture Notes in Artificial Intelligence, vol. 4869. Springer, Berlin, Heidelberg, pp. 113–138.
Montes, A., Wibmer, M., 2010. Gröbner bases for polynomial systems with parameters. J. Symbolic. Comput. 45 (12), 1391–1425.
Nabeshima, K., 2007. A speed-up of the algorithm for computing comprehensive Gröbner systems. In: Proceedings of ISSAC'2007. ACM Press, New York, pp. 299–306.
Nabeshima, K., 2007. PGB: a package for computing parametric Gröbner bases and related objects. Conference posters of ISSAC 2007, pp. 104–105.
Sun, Y., Wang, D.K., 2009. An efficient algorithm for factoring polynomials over algebraic extension field. Preprint, arXiv:0907.2300v2 [cs.SC].
Suzuki, A., Sato, Y., 2003. An alternative approach to comprehensive Gröbner bases. J. Symbolic. Comput. 36 (3–4), 649–667.
Suzuki, A., Sato, Y., 2004. Comprehensive Gröbner bases via ACGB. In: Tran, Q-N. (Ed.), ACA2004. pp. 65–73.
Suzuki, A., Sato, Y., 2006. A simple algorithm to compute comprehensive Gröbner bases using Gröbner bases. In: Proceedings of ISSAC'2006. ACM Press, New York, pp. 326–331.
Weispfenning, V., 1992. Comprehensive Gröbner bases. J. Symbolic. Comput. 14 (1), 1–29.
Weispfenning, V., 2003. Canonical comprehensive Gröbner bases. J. Symbolic. Comput. 36 (3–4), 669–683.
Wibmer, M., 2007. Gröbner bases for families of affine or projective schemes. J. Symbolic. Comput. 42 (8), 803–834.