Journal of Symbolic Computation 47 (2012) 610–642

Contents lists available at SciVerse ScienceDirect

Journal of Symbolic Computation

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

Algorithms for computing triangular decomposition of polynomial systems☆

Changbo Chen, Marc Moreno Maza

The University of Western Ontario, London, Ontario, Canada N6A 5B7

ARTICLE INFO

Article history:
Received 12 September 2011
Accepted 15 October 2011
Available online 22 December 2011

In dedication to Professor Wen Tsün Wu

Keywords:
Characteristic set
Triangular decomposition
Regular chain
Resultant
GCD

ABSTRACT

We discuss algorithmic advances which have extended the pioneer work of Wu on triangular decompositions. We start with an overview of the key ideas which have led to either better implementation techniques or a better understanding of the underlying theory. We then present new techniques that we regard as essential to the recent success and for future research directions in the development of triangular decomposition methods.

© 2011 Elsevier Ltd. All rights reserved.

1. Introduction

The characteristic set method of Wu has freed Ritt's decomposition from polynomial factorization, opening the door to a variety of discoveries in polynomial and differential algebra. The landmark paper *A Zero Structure Theorem for Polynomial Equations Solving* (Wu, 1987) where the method is proposed, and subsequent articles, among them (Wu, 1989a,c,d,e, 1992), already suggest important directions for further development.

During the past 25 years, the work of Wu has been extended to allow for more powerful decomposition algorithms and applied to different types of polynomial systems or decompositions: parametric algebraic systems (Chou and Gao, 1991), differential systems (Gao and Chou, 1993; Boulier et al., 1995; Hubert, 2000), difference systems (Gao et al., 2009), unmixed decompositions (Kalkbrener, 1998) and primary decomposition (Shimoyama and Yokoyama, 1996) of polynomial ideals, cylindrical algebraic decomposition (Chen et al., 2009), parametric (Yang et al., 2001) and non-parametric (Chen et al., 2010) semi-algebraic systems. Today, triangular decomposition algorithms are available in

☆ This research was partly supported by the NSERC, Maplesoft and the MITACS of Canada.
E-mail addresses: cchen252@csd.uwo.ca (C. Chen), moreno@csd.uwo.ca (M. Moreno Maza).

0747-7171/$ – see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jsc.2011.12.023

several software packages (Chen et al., 2011; Lemaire et al., 2005; Wang, `Wsolve`; Wang, Epsilon 0.618). Moreover, they provide back-engines for computer algebra system front-end solvers, such as MAPLE's `solve` command Maplesoft Incorporation.

A first part of this paper, presented in Section 2, is an overview of the algorithmic advances which have extended the pioneer work of Wu on triangular decompositions. Our aim is to highlight those key ideas which have led to either better implementation techniques and practical performance, or a better understanding of the relations between the computed algebraic objects and the represented geometrical entities.

Algorithms for computing triangular decompositions of polynomial systems can be classified in several ways. One can first consider the relation between the input system S and the output triangular systems, say $S_1, \ldots, S_e$. From that perspective, two types of decomposition are essentially different: those for which $S_1, \ldots, S_e$ encode all the points of the zero set of S (over the algebraic closure of the coefficient field of S) and those for which $S_1, \ldots, S_e$ represent only the "generic zeros" of the irreducible components of S.

One can also classify triangular decomposition algorithms by the algorithmic principles on which they rely: those which proceed *by variable elimination*, that is, by reducing the solving of a system in n unknowns to that of a system in $n-1$ unknowns and those which proceed *incrementally*, that is, by reducing the solving of a system in m equations to that of a system in $m-1$ equations.

The characteristic set method and most of the decomposition algorithms proposed by Wu's students (see the book of Wang (2000) and the references therein) belong to the first type in each classification. Kalkbrener's algorithm (Kalkbrener, 1993), which is an elimination method representing the "generic zeros" (as defined in van der Waerden (1991)), has brought efficient techniques, based on the concept of a *regular chain*, introduced independently by Kalkbrener in his Ph.D. thesis and, by Yang and Zhang (1991). Other works on triangular decomposition algorithms focus on incremental solving, following an idea proposed by Lazard (1991).

In a second part of this paper, from Section 4 to Section 6, we discuss algorithmic techniques that we regard as essential to the recent success and for future research directions in the development of triangular decomposition methods. Though we present these ideas in the context of incremental solving, we believe that they could also apply to other triangular decomposition schemes. These ideas can be summarized as follows.

First, we believe that a decomposition scheme should rely on a routine which is geometrically meaningful while allowing efficient algebraic calculations (computing by homomorphic images as in Dahan et al. (2005) and taking advantage of fast polynomial arithmetic as in Li et al. (2007)). The notion of a *regular GCD* introduced in Moreno Maza (1999) was a first step in that direction for incremental triangular decomposition algorithms. Recently, we observed in Chen and Moreno Maza (2011) that this notion and the related algorithms could be greatly simplified, leading to significant practical improvements as reflected in the experimental results therein. One key to this progress is a *specialization property of subresultants* repurposed to compute regular GCDs, namely Theorems 4 and 6 in Section 4. While this property extends known results, it provides corner cases for which we could not find a reference in the literature and which helps us greatly simplifying our decomposition algorithms.

Secondly, we believe that a decomposition scheme should prevent from the recomputation of costly intermediate objects, at least on generic examples. By recomputation of objects, we mean not only identical objects but also objects which are the homomorphic images of another one. Thus the use of hash tables at the implementation level to record intermediate results cannot cope with this requirement. For the incremental algorithm proposed in Chen and Moreno Maza (2011), we observe, with Theorem 7 in Section 5, that the intersection of a hypersurface $V(p)$ and the quasi-component $W(T)$ of a regular chain T reduces to computing regular GCDs of p and a polynomial $t \in T$. Moreover, all those regular GCDs can be derived from the subresultant chain of p and t (with respect to the main variable of t) thanks to the aforementioned specialization property of subresultants. The effectiveness of this recycling strategy is, again, illustrated by the experimental results of Chen and Moreno Maza (2011), some of which appear in Appendix B, and also by the fact that the `Triangularize` command of the `RegularChains` library is one of the back engines of MAPLE's `solve` command. This was

implemented after benchmarking Triangularize against other MAPLE's polynomial system solvers on more than 3300 test problems coming from the literature and MAPLE's users.

Thirdly, we believe that a decomposition scheme should control expression swell in the sense that intermediate objects which do not contribute to the final result should not be computed, at least on generic examples. Expression swell is a traditional challenge of symbolic computation. Fraction-free elimination methods such as subresultant algorithms (Ducos, 2000) are already used by Wu (1987) as an optimization technique. Their use became more systematic in the 90's with Moreno Maza and Rioboo (1995), Moreno Maza (1999) and Wang (1998). However, subresultant-based regular GCDs still calculate intermediate objects which do not contribute anything to the solving process. To understand why, suppose that the Intersect operation of Chen and Moreno Maza (2011) is given a regular chain T and a polynomial f which is regular modulo the saturated ideal of T. In addition, suppose that Wu's CHARSET procedure Wu (1987) is given $F = T \cup \{f\}$ as input. Among other objects, both procedures will compute the iterated resultant of f w.r.t. T, denoted by $\mathrm{res}(T, f)$. If the initials of the polynomials in T are not all equal to 1, some potentially large factors of $\mathrm{res}(T, f)$ are likely to be superfluous. As argued in Section 6, this will be the case generically if the saturated ideal of T has dimension 1. Theorem 8 highlights this expression swell explicitly. The remainder of Section 6 explains how to deal with this problem, that is, how to compute only the factors of $\mathrm{res}(T, f)$ that are of interest. Examples illustrate the proposed techniques. For some test problems, reported in Section 6.5, these techniques reduce the size of the computed iterated resultants by a factor of 50, leading to a running time speedup of three orders of magnitude.

2. The characteristic set method and related works

The characteristic set method is the first *factorization-free* algorithm for decomposing an algebraic variety into equidimensional components. The author, Wu Wen Tsün, realized an implementation of this method and reported experimental data in Wu (1987), following a series of preliminary works, among them are the papers (Wu, 1984a,b, 1986). To put this work into context, let us recall what the common idea of an algebraic set decomposition was at the time the article (Wu, 1987) was written.

Let $\mathbf{K}$ be an algebraically closed field and $\mathbf{k}$ be a subfield of $\mathbf{K}$. A subset $V \subset \mathbf{K}^n$ is an *(affine) algebraic variety* over $\mathbf{k}$ if there exists a polynomial set $F \subset \mathbf{k}[x_1, \ldots, x_n]$ such that the zero set $V(F) \subset \mathbf{K}^n$ of F equals V. Recall that V is called *irreducible* if for all algebraic varieties $V_1, V_2 \subset \mathbf{K}^n$ the relation $V = V_1 \cup V_2$ implies either $V = V_1$ or $V = V_2$. A first algebraic variety decomposition result is the famous Lasker–Nöther Theorem (Lasker, 1905; Nöther, 1921) which states the following.

Theorem 1 (*Lasker–Nöther*). *For each algebraic variety $V \subset \mathbf{K}^n$ there exist finitely many irreducible algebraic varieties $V_1, \ldots, V_e \subset \mathbf{K}^n$ such that we have*

$$V = V_1 \cup \cdots \cup V_e. \tag{1}$$

Moreover, if $V_i \not\subseteq V_j$ holds for $1 \le i < j \le e$ then the set $\{V_1, \ldots, V_e\}$ is unique and forms the irreducible decomposition *of V.*

The varieties $V_1, \ldots, V_e$ in Theorem 1 are called the irreducible components of V and can be regarded as a natural output for a decomposition algorithm, or, in other words, for an algorithm solving a system of equations given by polynomials in $\mathbf{k}[x_1, \ldots, x_n]$. In order to be implemented as a computer program, this algorithm specification should stipulate how irreducible components are represented. One such encoding is introduced by Ritt (1932) through the following result, that we present here as Wu does it in Wu (1984a) p. 215.

Theorem 2 (*Ritt*). *If $V(F) \subset \mathbf{K}^n$ is a non-empty and irreducible variety then one can compute a* reduced triangular set *C contained in the ideal $\langle F \rangle$ generated by F in $\mathbf{k}[x_1, \ldots, x_n]$ and such that each polynomial $g \in \langle F \rangle$ reduces to zero by pseudo-division w.r.t. C.*

We call the set C in Theorem 2 a *Ritt characteristic set* of the ideal $\langle F \rangle$. The notions of triangular set and pseudo-division are reviewed in Section 3. Ritt (1950) describes a method for solving polynomial systems, which is based on polynomial factorization over field extensions and computation of characteristic sets of prime ideals. Deriving a practical implementation from this method, however,

was and remains a difficult problem. In the 80's, when the characteristic set method was introduced, polynomial factorization was an active research area and certain fundamental questions on this subject were only solved recently (Steel, 2005). Nowadays, decomposing an algebraic variety into irreducible components is not essential for most application problems, since weaker notions of decompositions, less costly to compute, are sufficient.

The characteristic set method relies on the following variant of Theorem 2.

Theorem 3 (*Wu*). *For any finite polynomial set $F \subset \mathbf{k}[x_1, \ldots, x_n]$, one can compute a reduced triangular set $C \subset \langle F \rangle$ such that each polynomial $g \in F$ reduces to zero by pseudo-division w.r.t. C.*

We call the set C in Theorem 3 a *Wu characteristic set* of the polynomial set F. From now on, we shall assume that variables are ordered as $x_1 < \cdots < x_n$. Then, Algorithm 1, called CHARSET in Wu (1987), computes a Wu characteristic set of F. In this pseudo-code, the function call MinimalAutoreducedSubset(F) returns a so-called *basic set*, that is, a triangular set with minimum rank (see Section 3 for this term) among the reduced triangular sets contained in F; the function call prem(A, B) returns the set of all prem(a, B) for $a \in A$, where prem(a, B) is the pseudo-remainder of a w.r.t. B.

Algorithm 1: CHARSET

```
Input: F ⊂ k[x_1 < ··· < x_n].
Output: C a Wu characteristic set of F.
begin
    repeat
        (S)  B := MinimalAutoreducedSubset(F);
        (R)  A := F \ B;
             R := prem(A, B);
        (I)  R := R \ {0};
             F := F ∪ R;
    until R = ∅ ;
    return B;
end
```

Algorithm 2: Buchberger's Algorithm

```
Input: F ⊂ k[x_1, ..., x_n] and an admissible term order ≤.
Output: G a reduced Gröbner basis w.r.t. ≤ of the ideal ⟨F⟩ generated by F.
begin
    repeat
        (S)  B := MinimalAutoreducedSubset(F, ≤);
        (R)  A := S_Polynomials(B, ≤) ∪ F;
             R := Reduce(A, B, ≤);
        (I)  R := R \ {0};
             F := F ∪ R;
    until R = ∅ ;
    return B;
end
```

After reformulating Buchberger's Algorithm for computing Gröbner bases into Algorithm 2, its structure appears to be similar to that of the CHARSET procedure, as observed by Golubitsky (2005). In this pseudo-code, the function call S_Polynomials(B) computes the S-polynomials of all pairs of elements in B w.r.t. $\leq$ while Reduce(A, B, $\leq$) returns the remainders of all elements in A w.r.t. B and $\leq$. The specification of MinimalAutoreducedSubset(F, $\leq$) is an adaptation to the term order $\leq$ of the specification of MinimalAutoreducedSubset(F) in Wu's CHARSET procedure; more precisely, this

adaptation is obtained by computing the rank of a polynomial f as its leading monomial (instead of v^d where v is the leading variable of v and d the degree of f w.r.t. v).

In the early developments of the CHARSET procedure and Buchberger's Algorithm, the *selection* step (S), the *reduction* step (R), and the *incremental* step (I) have been studied intensively and variants have been proposed in order to improve the practical efficiency of these algorithms. Such crucial tricks already appear in Wu's pioneer article (Wu, 1987) and in his note (Wu, 1989b). A nice survey of these aspects is given in Wang (1991).

During the mid 80's another important *factorization-free* decomposition technique was introduced by J. Della Dora, C. Dicrescenzo, and D. Duval: the *D5 Principle*. This permits one to compute over a simple algebraic extension of the field $\mathbf{k}$, say $\mathbf{k}[x]/\langle m(x)\rangle$, as if the univariate polynomial $m(x)$ was irreducible, while assuming only that $m(x)$ is squarefree. M. Kalkbrener, in his Ph.D. thesis (Kalkbrener, 1991), combined the characteristic set method and the D5 Principle into a novel triangular decomposition method. This approach outperforms the characteristic set method, on examples for which a Wu characteristic set C of the input system F can be "split" into several Wu characteristic sets of the same dimension as that of C. Without using the D5 Principle, Wang's Algorithm in Wang (1993) is also motivated by the same type of examples. These examples, however, were not generic in the applications studied by Wu (1984a, 1989e).

Another important idea was introduced independently by Kalkbrener (1991) and by Yang and Zhang in Yang and Zhang (1991): the notion of a *regular chain*. Its main purpose is to cope with the fact that the CHARSET procedure may not detect, in some cases, that an input polynomial system F has no solutions. For instance, with $x_1 < x_2 < x_3 < x_4$ and $F = \{x_2^2 - x_1, x_1x_3^2 - 2x_2x_3 + 1, (x_2x_3 - 1)x_4 + x_1\}$, the CHARSET procedure applied to F returns F, although F is inconsistent. However, F is not a regular chain since the initial of $(x_2x_3 - 1)x_4 + x_1$ is a zero-divisor modulo the (saturated) ideal generated by the other two polynomials. Again, this situation was not generic in the applications studied by Wu.

In Kalkbrener's decomposition algorithm, applied to an input polynomial system F, the output regular chains represent generic zeros (in the sense of van der Waerden (1991)) of the irreducible components of $V(F)$. The relation between characteristic sets and generic zeros were also studied by Wu (1989c, 2006).

Further algorithmic improvements are based on the principle of incremental solving. This principle is quite attractive, since it allows one to control the properties and the size (Dahan et al., in press) of the intermediate computed objects. This is crucial in view of designing modular methods such as that presented in Dahan et al. (2005).

Lazard (1991) proposed incremental solving for computing triangular decompositions. His work was extended by the authors in Moreno Maza (1999) and Chen and Moreno Maza (2011). This solving principle is used in other areas of polynomial system solving such as the probabilistic algorithm of Lecerf (2003) (based on lifting fibers) and the numerical method of Sommese et al. (2008), (based on diagonal homotopy).

Each of those incremental triangular decompositions algorithms rely on a procedure for computing the intersection of a hypersurface and the quasi-component of a regular chain. Thus, the input of this operation can be regarded as well-behaved geometrical objects. However, known algorithms, namely the one of Lazard (1991) and the one of the second author (Moreno Maza, 1999) are quite involved, thus difficult to analyze and optimize. In the rest of the present paper, we revisit this intersection operation, defined below, and extend our preliminary study reported in Chen and Moreno Maza (2011).

Let $R = \mathbf{k}[x_1, \ldots, x_n]$ be the ring of multivariate polynomials with coefficients in $\mathbf{k}$ and variables $\mathbf{x} = x_1 < \cdots < x_n$. For a polynomial $p \in R$ and a regular chain $T \subset R$, the function call $\mathsf{Intersect}(p, T)$ returns regular chains $T_1, \ldots, T_e \subset R$ such that we have:

$$V(p) \cap W(T) \subseteq W(T_1) \cup \cdots \cup W(T_e) \subseteq V(p) \cap \overline{W(T)}.$$

We refer the reader to Section 3 for the notion of a regular chain and related concepts. In Section 4, we discussed how the notion of *regular GCD* is used to perform the intersection operation. We also discuss its relation with the specialization property of subresultants.

One practical challenge with the intersection operation is that many cases may need to be handled while computing $\mathsf{Intersect}(p, T)$. These special cases often lead to recompute the same thing, namely

the subresultant chain of p and one polynomial of T. This difficulty was not dealt with in Lazard (1991) and Moreno Maza (1999) and a solution was only first proposed in Chen and Moreno Maza (2011). In Section 5, we propose a new result which formally explains how the algorithm of Chen and Moreno Maza (2011) recycles intermediate results, thus preventing potentially expensive recomputations of subresultant chains.

Another practical challenge with the intersection operation is that, when computing the subresultant chain of a p and one polynomial of T, the initials of the polynomials in T are "propagated" into the subresultants and create a potentially dramatic expression swell, as stated by Theorem 8. Dealing with this problem necessitates the reduction of the computations to a case where the initials of T are equal to 1. In Section 6, for the sake of simplicity, we explain how to do so for iterated resultant computation. The handling of the whole intersection operation through this technique will be reported in a future paper.

3. Regular chains

We review hereafter the notion of a regular chain and its related concepts. Then we state basic properties of regular chains (Propositions 1–5, and Corollaries 1 and 2) which will be used in the rest of this paper. Recall that $\mathbf{k}$, $\mathbf{K}$, $\mathbf{k}[\mathbf{x}]$ denote respectively a field, its algebraic closure and the ring of polynomials over $\mathbf{k}$, with ordered variables $\mathbf{x} = x_1 < \cdots < x_n$. Let $p \in \mathbf{k}[\mathbf{x}]$.

Notations for polynomials. If p is not constant, then the greatest variable appearing in p is called the *main variable* of p, denoted by $\mathrm{mvar}(p)$. Furthermore, the leading coefficient, the degree, the leading monomial, the leading term and the reductum of p, regarded as a univariate polynomial in $\mathrm{mvar}(p)$, are called respectively the *initial*, the *main degree*, the *rank*, the *head* and the *tail* of p; they are denoted by $\mathrm{init}(p)$, $\mathrm{mdeg}(p)$, $\mathrm{rank}(p)$, $\mathrm{head}(p)$ and $\mathrm{tail}(p)$ respectively. Let q be another polynomial of $\mathbf{k}[\mathbf{x}]$. If q is not constant, then we denote by $\mathrm{prem}(p, q)$ and $\mathrm{pquo}(p, q)$ the pseudo-remainder and the pseudo-quotient of p by q as univariate polynomials in $\mathrm{mvar}(q)$. We say that p is less than q and write $p \prec q$ if either $p \in \mathbf{k}$ and $q \notin \mathbf{k}$ or both are non-constant polynomials such that $\mathrm{mvar}(p) < \mathrm{mvar}(q)$ holds, or $\mathrm{mvar}(p) = \mathrm{mvar}(q)$ and $\mathrm{mdeg}(p) < \mathrm{mdeg}(q)$ both hold. We write $p \sim q$ if neither $p \prec q$ nor $q \prec p$ hold.

Notations for polynomial sets. Let $F \subset \mathbf{k}[\mathbf{x}]$. We denote by $\langle F \rangle$ the ideal generated by F in $\mathbf{k}[\mathbf{x}]$. For an ideal $\mathcal{I} \subset \mathbf{k}[\mathbf{x}]$, we denote by $\dim(\mathcal{I})$ its dimension. A polynomial is *regular* modulo $\mathcal{I}$ if it is neither zero, nor a zerodivisor modulo $\mathcal{I}$. Denote by $V(F)$ the *zero set* (or algebraic variety) of F in $\mathbf{K}^n$. Let $h \in \mathbf{k}[\mathbf{x}]$. The *saturated ideal* of $\mathcal{I}$ w.r.t. h, denoted by $\mathcal{I} : h^\infty$, is the ideal $\{q \in \mathbf{k}[\mathbf{x}] \mid \exists m \in \mathbb{N} \text{ s.t. } h^m q \in \mathcal{I}\}$.

Triangular set. Let $T \subset \mathbf{k}[\mathbf{x}]$ be a *triangular set*, that is, a set of non-constant polynomials with pairwise distinct main variables. The set of main variables and the set of ranks of the polynomials in T are denoted by $\mathrm{mvar}(T)$ and $\mathrm{rank}(T)$, respectively. A variable in $\mathbf{x}$ is called *algebraic* w.r.t. T if it belongs to $\mathrm{mvar}(T)$, otherwise it is said *free* w.r.t. T. For $v \in \mathrm{mvar}(T)$, denote by T_v the polynomial in T with main variable v. For $v \in \mathbf{x}$, we denote by $T_{<v}$ (resp. $T_{\geq v}$) the set of polynomials $t \in T$ such that $\mathrm{mvar}(t) < v$ (resp. $\mathrm{mvar}(t) \geq v$) holds. Let h_T be the product of the initials of the polynomials in T. We denote by $\mathrm{sat}(T)$ the *saturated ideal* of T defined as follows: if T is empty then $\mathrm{sat}(T)$ is the trivial ideal $\langle 0 \rangle$, otherwise it is the ideal $\langle T \rangle : h_T^\infty$. The *quasi-component* $W(T)$ of T is defined as $V(T) \setminus V(h_T)$. Denote $\overline{W(T)} = V(\mathrm{sat}(T))$ as the Zariski closure of $W(T)$. For $F \subset \mathbf{k}[\mathbf{x}]$, we write $Z(F, T) := V(F) \cap W(T)$.

Rank of a triangular set. Let $S \subset \mathbf{k}[\mathbf{x}]$ be another triangular set. We say that T has smaller rank than S and we write $T \prec S$ if there exists $v \in \mathrm{mvar}(T)$ such that $\mathrm{rank}(T_{<v}) = \mathrm{rank}(S_{<v})$ holds and: (i) either $v \notin \mathrm{mvar}(S)$; (ii) or $v \in \mathrm{mvar}(S)$ and $T_v \prec S_v$. We write $T \sim S$ if $\mathrm{rank}(T) = \mathrm{rank}(S)$.

Iterated resultant. Let again $p, q \in \mathbf{k}[\mathbf{x}]$ and $v \in \mathbf{x}$. If either p or q is not constant and has main variable v, then we define $\mathrm{res}(p, q, v)$ as the resultant of p and q w.r.t. v. Let $T \subset \mathbf{k}[\mathbf{x}]$ be a triangular set. We define $\mathrm{res}(p, T)$ (resp. $\mathrm{res}(T, p)$) inductively: if $T = \varnothing$, then $\mathrm{res}(p, T) = p$ (resp. $\mathrm{res}(T, p) = p$); otherwise let v be greatest variable appearing in T, then $\mathrm{res}(p, T) = \mathrm{res}(\mathrm{res}(p, T_v, v), T_{<v})$ (resp. $\mathrm{res}(T, p) = \mathrm{res}(\mathrm{res}(T_v, p, v), T_{<v})$).

Regular chain. A triangular set $T \subset \mathbf{k}[\mathbf{x}]$ is a *regular chain* if: (i) either T is empty; (ii) or $T \setminus \{T_{\max}\}$ is a regular chain, where $T_{\max}$ is the polynomial in T with maximum rank, and the initial of $T_{\max}$ is regular w.r.t. $\mathrm{sat}(T \setminus \{T_{\max}\})$.

Good specialization. Let $T \subset \mathbf{k}[\mathbf{x}]$ be a regular chain. Let $x_1, \ldots, x_d$ be the free variables of T. Let $z = (z_1, \ldots, z_d)$ be a point of $\mathbf{K}^d$. We say that T *specializes well* at z if (i) none of the initials of the polynomials in T vanishes modulo the ideal $\langle x_1 - z_1, \ldots, x_d - z_d \rangle$ of $\mathbf{K}[\mathbf{x}]$; (ii) the image of T modulo $\langle x_1 - z_1, \ldots, x_d - z_d \rangle$ is a regular chain of $\mathbf{K}[\mathbf{x}]$.

Triangular decomposition. Let $F \subset \mathbf{k}[\mathbf{x}]$ be finite. Let $\mathfrak{T} := \{T_1, \ldots, T_e\}$ be a finite set of regular chains of $\mathbf{k}[\mathbf{x}]$. We call $\mathfrak{T}$ a *Kalkbrener triangular decomposition* of $V(F)$ if we have $V(F) = \cup_{i=1}^{e} \overline{W(T_i)}$. We call $\mathfrak{T}$ a *Lazard–Wu triangular decomposition* of $V(F)$ if we have $V(F) = \cup_{i=1}^{e} W(T_i)$.

Proposition 1 (*Th. 6.1. in Aubry et al., 1999*)**.** *Let p and T be respectively a polynomial and a regular chain of $\mathbf{k}[\mathbf{x}]$. Then, $\mathrm{prem}(p, T) = 0$ holds if and only if $p \in \mathrm{sat}(T)$ holds.*

Proposition 2 (*Prop. 5 in Moreno Maza, 1999*)**.** *Let T and T' be two regular chains of $\mathbf{k}[\mathbf{x}]$ such that $\sqrt{\mathrm{sat}(T)} \subseteq \sqrt{\mathrm{sat}(T')}$ and $\dim(\mathrm{sat}(T)) = \dim(\mathrm{sat}(T'))$ hold. Let $p \in \mathbf{k}[\mathbf{x}]$ such that p is regular w.r.t. $\mathrm{sat}(T)$. Then p is also regular w.r.t. $\mathrm{sat}(T')$.*

Proposition 3. *Let $p \in \mathbf{k}[\mathbf{x}] \setminus \mathbf{k}$ and $T \subset \mathbf{k}[\mathbf{x}]$ be a regular chain. Let $v = \mathrm{mvar}(p)$ and $r = \mathrm{prem}(p, T_{\geq v})$ such that $r \in \sqrt{\mathrm{sat}(T_{<v})}$ holds. Then, we have $p \in \sqrt{\mathrm{sat}(T)}$.*

Proof. Since $r = \mathrm{prem}(p, T_{\geq v})$, there exists an integer $e_0 \geq 0$ and a polynomial $f \in \langle T_{\geq v} \rangle$ such that $\mathrm{init}(T_{\geq v})^{e_0} p = f + r$. On the other hand, $r \in \sqrt{\mathrm{sat}(T_{<v})}$, therefore there exists an integer $e_1 \geq 0$ such that $\mathrm{init}(T_{<v})^{e_1} (\mathrm{init}(T_{\geq v})^{e_0} p - f)^{e_1} \in \langle T_{<v} \rangle$, which implies that $p \in \sqrt{\mathrm{sat}(T)}$. □

Corollary 1. *Let T and T' be two regular chains of $\mathbf{k}[x_1, \ldots, x_k]$, where $1 \leq k < n$. Let $p \in \mathbf{k}[\mathbf{x}]$ with $\mathrm{mvar}(p) = x_{k+1}$ such that $\mathrm{init}(p)$ is regular w.r.t. both $\mathrm{sat}(T)$ and $\mathrm{sat}(T')$. Assume that $\sqrt{\mathrm{sat}(T)} \subseteq \sqrt{\mathrm{sat}(T')}$ holds. Then we also have $\sqrt{\mathrm{sat}(T \cup p)} \subseteq \sqrt{\mathrm{sat}(T' \cup p)}$.*

Proof. This follows easily from Proposition 1. □

Proposition 4 (*Lemma 4 in Chen et al., 2007*)**.** *Let $p \in \mathbf{k}[\mathbf{x}]$. Let $T \subset \mathbf{k}[\mathbf{x}]$ be a regular chain. Then the following statements are equivalent:*

(i) *the polynomial p is regular w.r.t. $\mathrm{sat}(T)$,*
(ii) *for each prime ideal $\mathfrak{p}$ associated with $\mathrm{sat}(T)$, we have $p \notin \mathfrak{p}$,*
(iii) *the iterated resultant $\mathrm{res}(p, T)$ is not zero.*

Corollary 2. *Let $p \in \mathbf{k}[\mathbf{x}] \setminus \mathbf{k}$ and $T \subset \mathbf{k}[\mathbf{x}]$ be a regular chain. Let $v := \mathrm{mvar}(p)$ and $r := \mathrm{res}(p, T_{\geq v})$. We have:*

(1) *the polynomial p is regular w.r.t. $\mathrm{sat}(T)$ if and only if r is regular w.r.t. $\mathrm{sat}(T_{<v})$;*
(2) *if $v \notin \mathrm{mvar}(T)$ and $\mathrm{init}(p)$ is regular w.r.t. $\mathrm{sat}(T)$, then p is regular w.r.t. $\mathrm{sat}(T)$.*

Proof. By Proposition 4, p is regular w.r.t. $\mathrm{sat}(T)$ if and only if $\mathrm{res}(p, T) \neq 0$, which is equivalent to $\mathrm{res}(r, T_{<v}) \neq 0$, that is r is regular w.r.t. $\mathrm{sat}(T_{<v})$. So (1) holds. Claim (2) is a consequence of the McCoy Theorem. We can also prove (2) directly. Since $\mathrm{res}(\mathrm{init}(p), T) = \mathrm{res}(\mathrm{init}(p), T_{<v})$, if $\mathrm{init}(p)$ is regular w.r.t. $\mathrm{sat}(T)$, then $\mathrm{init}(p)$ is also regular w.r.t. $\mathrm{sat}(T_{<v})$. We claim that p is regular w.r.t. $\mathrm{sat}(T_{<v})$. Otherwise by Proposition 4, there is an associated prime ideal $\mathfrak{p}$ of $\mathrm{sat}(T_{<v})$ such that $p \in \mathfrak{p}$, which implies that $\mathrm{init}(p) \in \mathfrak{p}$, a contradiction. Therefore p is regular w.r.t. $\mathrm{sat}(T_{<v})$. On the other hand, $v \notin \mathrm{mvar}(T)$, which implies that $p = r$ and therefore p is regular w.r.t. $\mathrm{sat}(T)$. □

Proposition 5 (*Theorem 1.6 Boulier et al., 2006*)**.** *Let $T \subset \mathbf{k}[\mathbf{x}]$ be a regular chain. Let $x_1, \ldots, x_d$ be all the free variables of T. Then $\mathrm{sat}(T)$ is unmixed of dimension d. Moreover we have $\mathrm{sat}(T) \cap \mathbf{k}[x_1, \ldots, x_d] = \langle 0 \rangle$.*

4. Subresultants and regular GCDs

Algorithms for triangular decomposition make use implicitly or explicitly of a notion of GCD for univariate polynomials over coefficient rings that are not necessarily fields. A formal definition for those GCDs was proposed in Moreno Maza (1999) (see Definition 1) and applied to residue class rings of the form $\mathbb{A} = \mathbf{k}[\mathbf{x}]/\mathrm{sat}(T)$ where $\mathrm{sat}(T)$ is the saturated ideal of a regular chain T. In Chen and Moreno Maza (2011), we propose to consider rings $\mathbb{A}$ of the form $\mathbf{k}[\mathbf{x}]/\sqrt{\mathrm{sat}(T)}$ instead and we show how to adapt, and improve, the algorithms of Moreno Maza (1999) without computing a basis nor a characteristic set of $\sqrt{\mathrm{sat}(T)}$.

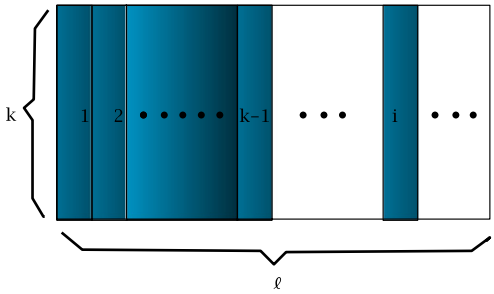
For the purpose of polynomial system solving (when retaining the multiplicities of zeros is not required) this weaker notion of a polynomial GCD is clearly sufficient. In addition, this yields a very simple procedure for computing such GCDs, see Theorem 6. To this end, we rely on two *specialization properties of subresultants*, namely Theorems 4 and 5. These technical results require the following brief review of subresultant theory.

4.1. Definition of subresultants

Throughout this section, we denote by $\mathbb{A}$ a commutative ring with unit elements. Let $f = a_m x^m + \cdots + a_0$ and $g = b_n x^n + \cdots + b_0$ be two polynomials of $\mathbb{A}[x]$ with positive degrees m and n. We call the following matrix the *Sylvester matrix* of f and g w.r.t. x.

$$L = \begin{pmatrix} a_m & a_{m-1} & \cdots & a_0 & & & \\ & a_m & a_{m-1} & \cdots & a_0 & & \\ & & \ddots & \ddots & & \ddots & \\ & & & a_m & a_{m-1} & \cdots & a_0 \\ b_n & b_{n-1} & \cdots & b_0 & & & \\ & b_n & b_{n-1} & \cdots & b_0 & & \\ & & \ddots & \ddots & & \ddots & \\ & & & b_n & b_{n-1} & \cdots & b_0 \end{pmatrix} \begin{array}{l} \left.\vphantom{\begin{matrix}a\\a\\a\\a\end{matrix}}\right\} n \\ \left.\vphantom{\begin{matrix}a\\a\\a\\a\end{matrix}}\right\} m \end{array}$$

Its determinant is called the *(Sylvester) resultant* of f and g w.r.t. x, denoted by $\mathrm{res}(f, g, x)$.



Let $\lambda = \min(m, n)$. For any $0 \le i < \lambda$, let L_i be the submatrix of L formed by removing the bottom i rows that include the coefficients of f and the bottom i rows that include the coefficients of g. Thus the j-th row of L_i is the j-th row of L for $j = 1 \cdots n - i$ and the $(i + j)$-th row of L for $j = n - i + 1 \cdots m + n - 2i$. Note that L_i is an $(m + n - 2i) \times (m + n)$ matrix. For $j = 0, \ldots, i$, let $L_{i,j}$ be the submatrix of L_i consisting of the first $m + n - 2i - 1$ columns and the $(m + n - 2i + j)$-th column. We call the polynomial $S_i(f, g) = \sum_{j=0}^{i} \det(L_{i,j}) x^{i-j}$ the i-th *subresultant* of f and g. Let $s_i(f, g) = \mathrm{coeff}(S_i(f, g), x^i)$ and call it the *principal subresultant coefficient* of S_i.

The previous construction can be described in the following more abstract way. Let $k \le \ell$ be two positive integers. Let M be an $k \times \ell$ matrix with coefficients in $\mathbb{A}$. Let M_j be the square submatrix

of M consisting of the first $k-1$ columns of M and the jth column of M, for $j = k \cdots \ell$. Let $\mathrm{dpol}(M) := \sum_{j=k}^{\ell} \det(M_j) x^{\ell-j}$; we call it the *determinant polynomial* of M.

Let $f_1(x), \ldots, f_k(x) \in \mathbb{A}[x]$. Let $\ell = 1 + \max(\deg(f_1(x)), \ldots, \deg(f_k(x)))$. The matrix M of $f_1, \ldots, f_k$ is a $k \times \ell$ matrix defined by $M_{ij} = \mathrm{coeff}(f_i, x^{\ell-j})$, for $1 \le i \le k$ and $1 \le j \le \ell$. We then define $\mathrm{dpol}(f_1, \ldots, f_k) = \mathrm{dpol}(M)$ and $\mathrm{mat}(f_1, \ldots, f_k) = M$.

Proposition 6. *Let $f = a_m x^m + \cdots + a_0$ and $g = b_n x^n + \cdots + b_0$ be two polynomials of $\mathbb{A}[x]$ with positive degrees m and n. Let $\lambda = \min(m, n)$. For $i = 0, \ldots, \lambda - 1$, we have*

$$S_i(f, g) = \mathrm{dpol}(x^{n-1-i} f, \ldots, xf, f, x^{m-1-i} g, \ldots, xg, g).$$

Proof. It follows directly from the definition of subresultants. □

We extend the definition of subresultants and principal subresultant coefficients in such a way that f and g are themselves subresultants. If $m \ge n$, we define $S_{\lambda+1} = f$, $S_\lambda = g$, $s_{\lambda+1} = a_m$ and $s_\lambda = b_n$. If $m < n$, we define $S_{\lambda+1} = g$, $S_\lambda = f$, $s_{\lambda+1} = b_n$ and $s_\lambda = a_m$.

4.2. Specialization properties of subresultants

In this section, we investigate the specialization property of subresultants. Although it is a well-known property, we did not find in the literature any result covering all the corner cases that we need to handle in the computation of regular GCDs. Therefore, we provide here such results together with self-contained proofs.

Let $\mathbb{B}$ be a field and let ϕ be a ring homomorphism from $\mathbb{A}$ to $\mathbb{B}$, which induces naturally also a ring homomorphism from $\mathbb{A}[x]$ to $\mathbb{B}[x]$. Define $m' = \deg(\phi(f))$, $n' = \deg(\phi(g))$ and $\lambda' = \min(m', n')$.

Lemma 1. *Let k be an integer such that $0 \le k < \lambda$. Assume that $\phi(s_k) \ne 0$ holds. Then either $\phi(a_m) \ne 0$ or $\phi(b_n) \ne 0$ holds. Moreover, we have both $\deg(\phi(f)) \ge k$ and $\deg(\phi(g)) \ge k$.*

Proof. Observe that

$$s_k = \begin{vmatrix} a_m & a_{m-1} & \cdots & a_0 & \\ & \cdots & & \cdots & \\ & a_m & a_{m-1} & \cdots & a_k \\ b_n & b_{n-1} & \cdots & b_0 & \\ & \cdots & & \cdots & \\ & b_n & b_{n-1} & \cdots & b_k \end{vmatrix}.$$

Therefore there exists $i \ge k, j \ge k$ such that $\phi(a_i) \ne 0$ and $\phi(b_j) \ne 0$. The conclusion follows. □

Lemma 2. *Assume that $\phi(s_0) = \cdots = \phi(s_{\lambda-1}) = 0$ hold. Then, if $m \le n$, we have*

(1) *If $\phi(a_m) \ne 0$ and $\phi(b_n) = \cdots = \phi(b_m) = 0$ hold, then $\phi(g) = 0$,*
(2) *If $\phi(a_m) = 0$ and $\phi(b_n) \ne 0$ hold, then $\phi(f) = 0$.*

Symmetrically, if $m > n$, we have

(3) *If $\phi(b_n) \ne 0$ and $\phi(a_m) = \cdots = \phi(a_n) = 0$ hold, then $\phi(f) = 0$,*
(4) *If $\phi(b_n) = 0$ and $\phi(a_m) \ne 0$ hold, then $\phi(g) = 0$.*

Proof. We prove only (1) and (2). The correctness of (3) and (4) follow by symmetry. We assume that $m \le n$ holds. To prove (1) and (2), it is clearly sufficient to prove respectively the following two statements.

(1*) If $\phi(a_m) \ne 0$ and $\phi(b_n) = \cdots = \phi(b_m) = 0$ hold, then $\phi(b_{m-i}) = 0$, for $i = 1, \ldots, m$.
(2*) If $\phi(a_m) = 0$ and $\phi(b_n) \ne 0$ hold, then $\phi(a_{m-i}) = 0$, for $i = 1, \ldots, m$.

Next we prove (1*) and (2*) simultaneously by induction on i. Firstly, for the base case $i = 1$, we have $S_{m-i} = S_{m-1} = \mathrm{dpol}(x^{n-m}f, \ldots, xf, f, g)$, which implies that we have

$$s_{m-1} = \begin{vmatrix} a_m & a_{m-1} & \cdots & & \\ & \ddots & \ddots & & \\ & & a_m & a_{m-1} \\ b_n & \cdots & b_m & b_{m-1} \end{vmatrix}. \tag{2}$$

If $\phi(a_m) \neq 0$ and $\phi(b_n) = \cdots = \phi(b_m) = 0$, then we have $\phi(s_{m-1}) = \phi(a_m)^{(n-m+1)}\phi(b_{m-1})$ by Eq. (2). Since $\phi(s_{m-1}) = 0$, we deduce that $\phi(b_{m-1}) = 0$, that is (1*) for $i = 1$. Similarly, if $\phi(a_m) = 0$ and $\phi(b_n) \neq 0$, then we have $\phi(s_{m-1}) = (-1)^{n-m+3}\phi(b_n)\phi(a_{m-1})^{(n-m+1)}$ by Eq. (2). Thus $\phi(a_{m-1}) = 0$ must hold, that is (2*) for $i = 1$. Now we assume that (1*) and (2*) hold for $< i$. By

$$S_i(f, g) = \mathrm{dpol}(x^{n-1-i}f, \ldots, xf, f, x^{m-1-i}g, \ldots, xg, g),$$

we have

$$s_{m-i} = \begin{vmatrix} a_m & a_{m-1} & \cdots & a_{m-(i-1)} & a_{m-i} & \cdots & & \\ & \ddots & & \ddots & & \ddots & & \\ & & a_m & a_{m-1} & \cdots & a_{m-(i-1)} & a_{m-i} \\ b_n & b_{n-1} & \cdots & b_{m-(i-1)} & b_{m-i} & \cdots & & \\ & \ddots & & \ddots & & \ddots & & \\ & & b_n & b_{n-1} & \cdots & b_{m-(i-1)} & b_{m-i} \end{vmatrix} \begin{matrix} \left.\vphantom{\begin{matrix}a\\a\\a\end{matrix}}\right\} n-m+i \\ \left.\vphantom{\begin{matrix}a\\a\\a\end{matrix}}\right\} i \end{matrix} \tag{3}$$

By induction, if $\phi(a_m) \neq 0$ and $\phi(b_n) = \cdots = \phi(b_m) = 0$ hold, then we have $\phi(b_n) = \cdots = \phi(b_{m-(i-1)}) = 0$. By Eq. (3), we deduce that $\phi(s_{m-i}) = \phi(a_m)^{(n-m+i)}\phi(b_{m-i})$ holds. Since $\phi(s_{m-i}) = 0$, we deduce $\phi(b_{m-i}) = 0$, that is (1*) for i. Similarly, if $\phi(a_m) = 0$ and $\phi(b_n) \neq 0$ hold, then by induction, we have $\phi(a_m) = \cdots = \phi(a_{m-(i-1)}) = 0$. By Eq. (3), we deduce that $\phi(s_{m-i}) = (-1)^{n-m+i+2}\phi(b_n)^i\phi(a_{m-i})^{(n-m+i)}$ holds. Since $\phi(s_{m-i}) = 0$, we deduce $\phi(a_{m-i}) = 0$, that is (2*) for i. Finally, both (1) and (2) hold. □

Lemma 3. *Let i be an integer such that $0 \leq i < \lambda$.*

(1) *If $m' = m$ and $n' \geq i$, then we have*

$$\phi(S_i) = \phi(a_m)^{n-n'}\mathrm{dpol}(x^{n'-1-i}\phi(f), \ldots, x\phi(f), \phi(f), x^{m-1-i}\phi(g), \ldots, x\phi(g), \phi(g)).$$

(2) *If $n' = n$ and $m' \geq i$, then we have*

$$\begin{aligned} \phi(S_i) = \quad & (-1)^{(m-m')(n-i+2)}\mathrm{dpol}(x^{n-1-i}\phi(f), \ldots, x\phi(f), \phi(f), \\ & x^{m'-1-i}\phi(g), \ldots, x\phi(g), \phi(g)). \end{aligned}$$

Proof. The matrix $M = \mathrm{mat}(x^{n-1-i}f, \ldots, xf, f, x^{m-1-i}g, \ldots, xg, g)$ is as follows

$$M = \begin{pmatrix} a_m & a_{m-1} & \cdots & a_0 & & & \\ & a_m & a_{m-1} & \cdots & a_0 & & \\ & & \ddots & \ddots & & \ddots & \\ & & & a_m & a_{m-1} & \cdots & a_0 \\ b_n & b_{n-1} & \cdots & b_0 & & & \\ & b_n & b_{n-1} & \cdots & b_0 & & \\ & & \ddots & \ddots & & \ddots & \\ & & & b_n & b_{n-1} & \cdots & b_0 \end{pmatrix} \begin{matrix} \left.\vphantom{\begin{matrix}a\\a\\a\\a\end{matrix}}\right\} n-i \\ \left.\vphantom{\begin{matrix}a\\a\\a\\a\end{matrix}}\right\} m-i \end{matrix}$$

We know that $S_i = \mathrm{dpol}(M)$. We first prove (1). We assume that $m' = m$ and $n' \geq i$ both hold. Thus, we have $n - n' \leq n - i$, which yields

$$\begin{aligned}\phi(S_i) &= \phi(\mathrm{dpol}(x^{n-1-i}f, \ldots, xf, f, x^{m-1-i}g, \ldots, xg, g)) \\ &= \mathrm{dpol}(x^{n-1-i}\phi(f), \ldots, x\phi(f), \phi(f), x^{m-1-i}\phi(g), \ldots, x\phi(g), \phi(g)) \\ &= \phi(a_m)^{n-n'}\mathrm{dpol}(x^{n'-1-i}\phi(f), \ldots, x\phi(f), \phi(f), x^{m-1-i}\phi(g), \ldots, x\phi(g), \phi(g)).\end{aligned}$$

This proves (1). Now, we prove (2). We assume that $n' = n$ and $m' \geq i$ hold. Thus, we have $m - m' \leq m - i$, which yields

$$\begin{aligned}\phi(S_i) &= \phi(\mathrm{dpol}(x^{n-1-i}f, \ldots, xf, f, x^{m-1-i}g, \ldots, xg, g)) \\ &= \mathrm{dpol}(x^{n-1-i}\phi(f), \ldots, x\phi(f), \phi(f), x^{m-1-i}\phi(g), \ldots, x\phi(g), \phi(g)) \\ &= (-1)^{(m-m')(n-i+2)}\mathrm{dpol}(x^{n-1-i}\phi(f), \ldots, x\phi(f), \phi(f), \\ &\quad x^{m'-1-i}\phi(g), \ldots, x\phi(g), \phi(g)).\end{aligned}$$

This proves (2). □

Theorem 4 (*Specialization Property of Subresultants*)**.** *Let i be an integer such that $0 \leq i < \lambda$.*

(1) *If $m' = m$ and $n' > i$, then we have $\phi(S_i(f, g)) = \phi(a_m)^{n-n'}S_i(\phi(f), \phi(g))$,*
(2) *If $m' = m$ and $n' = i$, then we have $\phi(S_i(f, g)) = \phi(a_m)^{n-n'}\phi(b_{n'})^{m-1-i}\phi(g)$,*
(3) *If $n' = n$ and $m' > i$, then we have*

$$\phi(S_i(f, g)) = (-1)^{(m-m')(n-i+2)}S_i(\phi(f), \phi(g)),$$

(4) *If $n' = n$ and $m' = i$, then we have*

$$\phi(S_i(f, g)) = (-1)^{(m-m')(n-i+2)}\phi(a_{m'})^{n-1-i}\phi(f).$$

Proof. It directly follows from Lemma 3. □

Remark 1. Theorem 4 provides corner cases which are not covered by other papers or textbooks in the literature, such as Mishra's book *Algorithmic Algebra* (Mishra, 1993). For example, the case where $m = n = m' = n'$ and $i = n' - 1$ both hold is not covered by Lemma 7.8.1 nor Corollary 7.8.2 in Mishra (1993). The case where $m = n = m' = n' + 1$ and $i = n'$ hold is not covered either. As we shall see with Theorem 5, these corner cases are needed in polynomial GCD computation.

Theorem 5. *We have the following relations between the subresultants of f and g and, the GCDs of $\phi(f)$ and $\phi(g)$:*

(1) *Let $0 \leq k < \lambda$ be an integer such that $\phi(s_k) \neq 0$ and $\phi(s_i) = 0$ for any $0 \leq i < k$. Then $\phi(S_k)$ is a GCD of $\phi(f)$ and $\phi(g)$.*
(2) *Assume that $\phi(s_i) = 0$ for all $0 \leq i < \lambda$. We have the following cases.*
   (2a) *If $m \leq n$ and $\phi(a_m) \neq 0$, then $\phi(f)$ is a GCD of $\phi(f)$ and $\phi(g)$; symmetrically, if $m > n$ and $\phi(b_n) \neq 0$, then $\phi(g)$ is a GCD of $\phi(f)$ and $\phi(g)$.*
   (2b) *If $m \leq n$ and $\phi(a_m) = 0$ but $\phi(b_n) \neq 0$, then $\phi(g)$ is a GCD of $\phi(f)$ and $\phi(g)$; if $m \geq n$ and $\phi(b_n) = 0$ but $\phi(a_m) \neq 0$, then $\phi(f)$ is a GCD of $\phi(f)$ and $\phi(g)$.*
   (2c) *If $\phi(a_m) = \phi(b_n) = 0$, then a GCD of $\phi(f)$ and $\phi(g)$ is also a GCD of $\phi(\mathrm{red}(f))$ and $\phi(\mathrm{red}(g))$ and vice versa, where $\mathrm{red}(f)$ and $\mathrm{red}(g)$ are the reductums of f and g respectively.*

Proof. Let us first prove (1). Since $\phi(s_k) \neq 0$, by Lemma 1, we know that either $\phi(a_m) \neq 0$ or $\phi(b_n) \neq 0$ holds; moreover, we have $k \leq m'$ and $k \leq n'$. W.l.o.g, we assume that $\phi(a_m) \neq 0$, that is $m = m'$. Since $k \leq n'$, for all $i < k$, we have $i < n'$. Applying (1) of Theorem 4, we deduce that $\phi(s_i(f, g)) = \phi(a_m)^{n-n'}s_i(\phi(f), \phi(g))$ holds. Since $\phi(s_i) = 0$ for any $0 \leq i < k$, we deduce that

$$s_i(\phi(f), \phi(g)) = 0, \quad i = 0, \ldots, k-1. \tag{4}$$

Similarly, if $k < n'$, applying (1) of Theorem 4, we deduce that

$$\phi(s_k(f, g)) = \phi(a_m)^{n-n'} s_k(\phi(f), \phi(g)).$$

Since $\phi(s_k) \neq 0$, we have

$$s_k(\phi(f), \phi(g)) \neq 0. \tag{5}$$

If $k = n'$, we have $s_k(\phi(f), \phi(g)) = \phi(b_{n'})$. Since $\phi(b_{n'}) \neq 0$, Eq. (5) still holds. By the Subresultant Chain Theorem (Theorem 7.10.5 of Mishra (1993), p. 279), we know that $S_k(\phi(f), \phi(g))$ is a GCD of $\phi(f)$ and $\phi(g)$. Applying (1) and (2) of Theorem 4, we conclude that $\phi(S_k)$ is a GCD of $\phi(f)$ and $\phi(g)$.

Next we prove (2a). By symmetry, we prove it for $m \leq n$. If $\phi(b_n) = \cdots = \phi(b_m) = 0$, it follows directly from Lemma 2. Otherwise, we have $n' \geq m$. Thus for all $i < m$, we have $i < n'$. By (1) of Theorem 4, we have $\phi(S_i) = \phi(a_m)^{n-n'} S_i(\phi(f), \phi(g))$, $i = 0, \ldots, m-1$. Thus $\phi(s_i) = 0$ implies that $s_i(\phi(f), \phi(g)) = 0$ holds, for $i = 0, \ldots, m-1$. Since $\phi(a_m) \neq 0$ holds, the Subresultant Chain Theorem implies that $\phi(f)$ is a GCD of $\phi(f)$ and $\phi(g)$.

Finally (2b) follows directly from Lemma 2 and (2c) is obviously true. □

4.3. Regular GCDs

Definition 1. Let $\mathbb{A}$ be a commutative ring with unit elements. Let $p, t, g \in \mathbb{A}[y]$ with $t \neq 0$ and $g \neq 0$. We say that $g \in \mathbb{A}[y]$ is a *regular GCD* of p, t if:

(R_1) the leading coefficient of g in y is a regular element of $\mathbb{A}$;
(R_2) g belongs to the ideal generated by p and t in $\mathbb{A}[y]$;
(R_3) if $\deg(g, y) > 0$, then g pseudo-divides both p and t, that is, we have $\mathrm{prem}(p, g) = \mathrm{prem}(t, g) = 0$.

Definition 1 was introduced in Moreno Maza (1999) as part of a formal framework for algorithms manipulating regular chains (Della Dora et al., 1985; Lazard, 1991; Chou and Gao, 1992; Kalkbrener, 1993; Yang and Zhang, 1991). In this section, the ring $\mathbb{A}$ will always be of the form $\mathbf{k}[\mathbf{x}]/\sqrt{\mathrm{sat}(T)}$. Thus, a regular GCD of p, t in $\mathbb{A}[y]$ is also called a regular GCD of p, t modulo $\sqrt{\mathrm{sat}(T)}$.

Proposition 7. *For $1 \leq k \leq n$, let $T \subset \mathbf{k}[x_1, \ldots, x_{k-1}]$ be a regular chain, possibly empty. Let $p, t, g \in \mathbf{k}[x_1, \ldots, x_k]$ be non-constant polynomials with main variable x_k. Let h_g be the initial of g. Assume $T \cup \{t\}$ is a regular chain and g is a regular GCD of p and t modulo $\sqrt{\mathrm{sat}(T)}$. Then, we have:*

(i) *if $\mathrm{mdeg}(g) = \mathrm{mdeg}(t)$, then $W(T \cup t) \subseteq Z(h_g, T \cup t) \cup W(T \cup g) \subseteq \overline{W(T \cup t)}$ and $\sqrt{\mathrm{sat}(T \cup t)} = \sqrt{\mathrm{sat}(T \cup g)}$ both hold,*
(ii) *if $\mathrm{mdeg}(g) < \mathrm{mdeg}(t)$, let $q = \mathrm{pquo}(t, g)$, then $T \cup q$ is a regular chain and the following two relations hold:*
  (ii.a) $\sqrt{\mathrm{sat}(T \cup t)} = \sqrt{\mathrm{sat}(T \cup g)} \cap \sqrt{\mathrm{sat}(T \cup q)}$,
  (ii.b) $W(T \cup t) \subseteq Z(h_g, T \cup t) \cup W(T \cup g) \cup W(T \cup q) \subseteq \overline{W(T \cup t)}$,
(iii) $W(T \cup g) \subseteq V(p)$,
(iv) $V(p) \cap W(T \cup t) \subseteq W(T \cup g) \cup V(p, h_g) \cap W(T \cup t) \subseteq V(p) \cap \overline{W(T \cup t)}$.

Proof. We first establish a relation between p, t and g. By definition of pseudo-division, there exist polynomials q, r and a nonnegative integer e_0 such that

$$h_g^{e_0} t = qg + r \quad \text{and} \quad r \in \sqrt{\mathrm{sat}(T)} \tag{6}$$

both hold. Hence, there exists an integer $e_1 \geq 0$ such that:

$$(h_T)^{e_1} (h_g^{e_0} t - qg)^{e_1} \in \langle T \rangle \tag{7}$$

holds, which implies: $t \in \sqrt{\mathrm{sat}(T \cup g)}$. We first prove (i). Since $\mathrm{mdeg}(t) = \mathrm{mdeg}(g)$ holds, we have $q \in \mathbf{k}[x_1, \ldots, x_{k-1}]$, and thus we have $h_g^{e_0} h_t = q h_g$. Since h_t and h_g are regular modulo

$\mathrm{sat}(T)$, the same property holds for q. Together with (7), we obtain $g \in \sqrt{\mathrm{sat}(T \cup t)}$. Therefore $\sqrt{\mathrm{sat}(T \cup t)} = \sqrt{\mathrm{sat}(T \cup g)}$. The inclusion relation in (i) follows from (6).

We prove (ii). Assume $\mathrm{mdeg}(t) > \mathrm{mdeg}(g)$. With (6) and (7), this hypothesis implies that $T \cup q$ is a regular chain and $t \in \sqrt{\mathrm{sat}(T \cup q)}$ holds. Since $t \in \sqrt{\mathrm{sat}(T \cup g)}$ also holds, $\sqrt{\mathrm{sat}(T \cup t)}$ is contained in $\sqrt{\mathrm{sat}(T \cup g)} \cap \sqrt{\mathrm{sat}(T \cup q)}$. Conversely, for any $f \in \sqrt{\mathrm{sat}(T \cup g)} \cap \sqrt{\mathrm{sat}(T \cup q)}$, there exists an integer $e_2 \geq 0$ and $a \in \mathbf{k}[\mathbf{x}]$ such that $(h_g h_q)^{e_2} f^{e_2} - aqg \in \mathrm{sat}(T)$ holds. With (6) we deduce that $f \in \sqrt{\mathrm{sat}(T \cup t)}$ holds and so does (ii.a). From (6), we also derive (ii.b).

We prove (iii) and (iv). Definition 1 implies: $\mathrm{prem}(p, g) \in \sqrt{\mathrm{sat}(T)}$. Thus $p \in \sqrt{\mathrm{sat}(T \cup g)}$ holds, that is, $\overline{W(T \cup g)} \subseteq V(p)$, which implies (iii). Moreover, since $g \in \langle p, t, \sqrt{\mathrm{sat}(T)} \rangle$, we have $Z(p, T \cup t) \subseteq V(g)$, so we deduce (iv). $\square$

Let p, t be two polynomials of $\mathbf{k}[x_1, \ldots, x_k]$, for $k \geq 1$. Let $m = \deg(p, x_k)$, $n = \mathrm{mdeg}(t, x_k)$. Assume that $m, n \geq 1$. Let $\lambda = \min(m, n)$. Let T be a regular chain of $\mathbf{k}[x_1, \ldots, x_{k-1}]$. Let $\mathbb{B} = \mathbf{k}[x_1, \ldots, x_{k-1}]$ and $\mathbb{A} = \mathbb{B}/\sqrt{\mathrm{sat}(T)}$. Let $S_0, \ldots, S_{\lambda+1}$ be the subresultant polynomials of p and t w.r.t. x_k in $\mathbb{B}[x_k]$. Let s_i be the principal subresultant coefficient of S_i, for $0 \leq i \leq \lambda + 1$. The following theorem provides sufficient conditions for S_j (with $1 \leq j \leq \lambda + 1$) to be a regular GCD of p and t in $\mathbb{A}[x_k]$.

Theorem 6. *Let j be an integer, with $1 \leq j \leq \lambda + 1$, such that s_j is a regular element of $\mathbb{A}$ and such that for any $0 \leq i < j$, we have $s_i = 0$ in $\mathbb{A}$. Then S_j is a regular GCD of p and t in $\mathbb{A}[x_k]$.*

Proof. By Definition 1, it suffices to show that $\mathrm{prem}(p, S_j, x_k) = 0$ and $\mathrm{prem}(t, S_j, x_k) = 0$ both hold in $\mathbb{A}$. We prove the former equality, the proof of the latter being similar.

Let $\mathfrak{p}$ be any prime ideal associated with $\mathrm{sat}(T)$. Define $\mathbb{D} = \mathbf{k}[x_1, \ldots, x_{k-1}]/\mathfrak{p}$ and let $\mathbb{L}$ be the fraction field of the integral domain $\mathbb{D}$. Let ϕ be the homomorphism from $\mathbb{B}$ to $\mathbb{L}$. By Theorem 5, we know that $\phi(S_j)$ is a GCD of $\phi(p)$ and $\phi(t)$ in $\mathbb{L}[x_k]$. Therefore there exists a polynomial q of $\mathbb{L}[x_k]$ such that $p = qS_j$ in $\mathbb{L}[x_k]$, which implies that there exists a nonzero element a of $\mathbb{D}$ and a polynomial q' of $\mathbb{D}[x_k]$ such that $ap = q'S_j$ in $\mathbb{D}[x_k]$. Therefore $\mathrm{prem}(ap, S_j) = 0$ in $\mathbb{D}[x_k]$, which implies that $\mathrm{prem}(p, S_j) = 0$ in $\mathbb{D}[x_k]$. Therefore $\mathrm{prem}(p, S_j)$ belongs to $\mathfrak{p}$ and thus to $\sqrt{\mathrm{sat}(T)}$. So $\mathrm{prem}(p, S_j, x_k) = 0$ in $\mathbb{A}$. $\square$

5. Recycling computations

Consider the intersection operation as defined at the end of Section 2. Up to technical details, if T consists of a single polynomial t whose main variable is the same as p, say v, computing $\mathsf{Intersect}(p, T)$ can be achieved by successively calculating:

(s_1) the resultant r of p and t w.r.t. v, and,
(s_2) a regular GCD of p and t modulo the squarefree part of r.

Observe that Steps (s_1) and (s_2) reduce essentially to computing the subresultant chain of p and t w.r.t. v. The algorithms listed in Appendix A and presented in Chen and Moreno Maza (2011) extend this simple observation for computing $\mathsf{Intersect}(p, T)$ with an arbitrary regular chain T. In broad terms, the intermediate polynomials computed during the "elimination phases" of $\mathsf{Intersect}(p, T)$ are recycled for performing the "extension phases" at essentially no cost.

In this section, we show that this recycling strategy leads to the following surprising result, which was unknown to us at the time of writing (Chen and Moreno Maza , 2011). Each regular chain in the output of $\mathsf{Intersect}(p, T)$ (as computed by the algorithms of Chen and Moreno Maza (2011)) is of the form $T_i \cup g_i$ where g_i is a regular GCD of p and t modulo $\sqrt{\mathrm{sat}(T_i)}$. Thanks to the results of Section 4, this implies all those GCDs can be obtained from the same subresultant chain, namely the one of p and t.

This result is formalized by Theorem 7, which is a property of the incremental algorithm presented in Chen and Moreno Maza (2011). Proving this property formally requires to follow the proof of the algorithm. Due to the mutual recursion of the algorithm's subprocedures, this is a non-trivial proof by

induction. (These subprocedures are presented in Appendix A.) It is not our purpose here to enter this aspect, which would imply to repeat the proof of Theorem 2 in Chen and Moreno Maza (2011). Our goal hereafter is just to highlight the algebraic construction which allows us to recycle intermediate computations. For this reason, the justification below is called a sketch of proof.

Theorem 7 (*Recycling Theorem*)**.** *For $1 \leq k \leq n$, let $T \subset \mathbf{k}[x_1, \ldots, x_{k-1}]$ be a regular chain. Let $p, t \in \mathbf{k}[x_1, \ldots, x_k]$ be polynomials with main variable x_k. Assume $T \cup \{t\}$ is a regular chain. Then there exist finitely many regular chains $T_1 \cup g_1, \ldots, T_e \cup g_e$ such that the following hold:*

(i) $V(p) \cap W(T \cup t) \subseteq \cup_{i=1}^{e} W(T_i \cup g_i) \subseteq V(p) \cap \overline{W(T \cup t)}$,
(ii) *each g_i is some subresultant polynomial of p and t,*
(iii) *g_i is a regular GCD of p and t modulo $\sqrt{\mathrm{sat}(T_i)}$.*

Sketch of Proof. Let $r := \mathrm{res}(p, t)$ be the resultant of p and t. By calling $\mathsf{Intersect}(r, T)$, one computes a family $\mathfrak{T}_0$ of regular chains in $\mathbf{k}[x_1, \ldots, x_{k-1}]$ such that $V(r) \cap W(T) \subseteq \cup_{C \in \mathfrak{T}_0} W(C) \subseteq V(r) \cap \overline{W(T)}$. Note that $W(C) \subseteq V(r)$ implies $r \in \sqrt{\mathrm{sat}(C)}$.

For each $C \in \mathfrak{T}_0$, by calling $\mathsf{Regularize}(\mathrm{init}(t), C)$, we can compute another family $\mathfrak{T}_1$ of regular chains such that we have

- $V(r) \cap W(T) \setminus V(\mathrm{init}(t)) \subseteq \cup_{D \in \mathfrak{T}_1} W(D) \setminus V(\mathrm{init}(t)) \subseteq V(r) \cap \overline{W(T)}$,
- for each $D \in \mathfrak{T}_1$, $\mathrm{init}(t)$ is regular modulo $\mathrm{sat}(D)$,
- for each $D \in \mathfrak{T}_1$, we have $r \in \sqrt{\mathrm{sat}(D)}$.

By Corollary 1, we deduce that

$$V(p) \cap W(T \cup t) \subseteq \bigcup_{D \in \mathfrak{T}_1} V(p) \cap W(D \cup t) \subseteq V(p) \cap \overline{W(T \cup t)}$$

holds. Moreover, for each $D \in \mathfrak{T}_1$, we have $\mathrm{res}(p, t) \in \sqrt{\mathrm{sat}(D)}$.

Let $\lambda = \min(\mathrm{mdeg}(p), \mathrm{mdeg}(t))$. Let m, $1 \leq m \leq \lambda + 1$ be the integer such that $S_m(p, t) = t$ and $s_m(p, t) = \mathrm{init}(t)$. For each $D \in \mathfrak{T}_1$, we call $\mathsf{Regularize}$ to split D w.r.t. the principal subresultant coefficients of p and t and obtain a family $\mathfrak{L}_D$ of regular chains such that

- $W(D) \setminus V(\mathrm{init}(t)) \subseteq \cup_{E \in \mathfrak{L}_D} W(E) \setminus V(\mathrm{init}(t)) \subseteq \overline{W(D)}$,
- for each $E \in \mathfrak{L}_D$, there exists a j_E, $1 \leq j_E \leq m$ such that s_{j_E} and s_m are regular modulo $\mathrm{sat}(E)$ and $s_i \in \sqrt{\mathrm{sat}(E)}$ for all $i < j_E$ hold.

By Theorem 6, we know that for each $E \in \mathfrak{L}_D$, S_{j_E} is a regular GCD of p and t modulo $\sqrt{\mathrm{sat}(E)}$.

Let $\mathfrak{T}_2$ be the union of all $\mathfrak{L}_D$. The following properties hold:

- $V(p) \cap W(T \cup t) \subseteq \cup_{E \in \mathfrak{T}_2} V(p) \cap W(E \cup t) \subseteq V(p) \cap \overline{W(T \cup t)}$,
- for each $E \in \mathfrak{T}_2$, there exists a polynomial g_E which is some subresultant polynomial of p and t,
- and g_E is a regular GCD of p and t modulo $\sqrt{\mathrm{sat}(E)}$.

For each $E \in \mathfrak{T}_2$, by (iv) of Proposition 7, we have

$$V(p) \cap W(E \cup t) \subseteq W(E \cup g_E) \cup V(p, \mathrm{init}(g_E)) \cap W(E \cup t) \subseteq V(p) \cap \overline{W(E \cup t)}.$$

Therefore we deduce that

$$V(p) \cap W(T \cup t) \subseteq \bigcup_{E \in \mathfrak{T}_2} W(E \cup g_E) \cup V(p, \mathrm{init}(g_E)) \cap W(E \cup t) \subseteq V(p) \cap \overline{W(T \cup t)}. \tag{8}$$

Since $\mathrm{init}(g_E)$ is regular modulo $\mathrm{sat}(E)$, by $\mathsf{Intersect}$ and $\mathsf{Regularize}$, we can compute a family $\mathfrak{L}_E$ of regular chains such that

- for each $F \in \mathfrak{L}_E$, the dimension of $\mathrm{sat}(F)$ is less than that of $\mathrm{sat}(E)$,
- $F \cup t$ is a regular chain.

By recursion, this theorem holds for p and $F \cup t$. Together with Relation (8), we deduce that the theorem holds for p and $T \cup t$. $\square$

Example 1. Let $p := z^3 + z^2 + w$, $t_z := z^3 - z + y$ and $t_y := y^2 + x$ be three polynomials in $\mathbb{Q}[w < x < y < z]$. Let $T := \{t_y\}$. Note that both T and $T \cup \{t_z\}$ are regular chains. Moreover, p and t_z have the same main variable z. The subresultants of p and t_z are the following three polynomials:

$$\begin{aligned} S_0(p, t_z) &= y^3 - 3\,wy^2 + \left(3w^2 + w\right) y - 2\,w^2 - w^3 \\ S_1(p, t_z) &= (y - w)\,z + w \\ S_2(p, t_z) &= -z^2 - w - z + y. \end{aligned}$$

Let

$$T_1 := \begin{cases} \left(-3\,w^2 - w + x\right) y + w^3 + 2\,w^2 - 3\,xw \\ x^3 + \left(3\,w^2 - 2\,w\right) x^2 + \left(-6\,w^3 + 3\,w^4 + w^2\right) x + w^6 + 4\,w^5 + 4\,w^4, \end{cases}$$

$T_2 := \{64\,y^2 - 5, 64\,x + 5, 8\,w + 1\}$ and $T_3 := \{y, x, w\}$, which are all regular chains in $\mathbb{Q}[w < x < y]$. Let $g_1 := S_1(p, t_z)$, $g_2 := S_1(p, t_z)$ and $g_3 := S_2(p, t_z)$. Then one can verify that $T_1 \cup \{g_1\}$, $T_2 \cup \{g_2\}$ and $T_3 \cup \{g_3\}$ satisfy the three conditions (i)–(iii) in Theorem 7. In other words, the regular chains $T_1 \cup \{g_1\}$, $T_2 \cup \{g_2\}$ and $T_3 \cup \{g_3\}$ form a valid output for the Intersect operation. Therefore, the function call Intersect$(p, \{t_y, t_z\})$ can be achieved essentially at the cost of computing the subresultants of p and t_z ONCE!

6. Controlling expression swell

It is a well known fact that the iterated resultant of a polynomial f w.r.t. a regular chain T may contain factors whose roots cannot be extended to points in the intersection of the hypersurface $V(f)$ and the quasi-component $W(T)$. Let us consider a simple example with two bivariate polynomials $t = ux^2 + (u + 1)x + 1$ and $f = x + u + 1$. The "iterated" resultant $\mathrm{res}(\{t\}, f)$ is $u^3 + u^2 - u$. Since u is the initial of t, this factor of $\mathrm{res}(\{t\}, f)$ does not lead to a common point of $V(f)$ and $W(T)$.

More generally, a root of a common factor of $\mathrm{res}(T, f)$ and $\mathrm{res}(T, h_T)$ may not lead to a common point of $V(f)$ and $W(T)$. Indeed, recall that $\mathrm{res}(T, h_T) = 0$ defines the locus of the values at which the regular chain T does not specialize well. Consider a simple example with three trivariate polynomials $t_2 = x_1x_2 + u$, $t_1 = x_1^2 + u$, $f = x_2 + u + 1$ for the variable ordering $u < x_1 < x_2$. Note that $T = \{t_2, t_1\}$ is a regular chain and that the iterated resultant $\mathrm{res}(T, f)$ is $u^3 + 3u^2 + u$, while the resultant of t_1 and the initial of t_2 is u. One can easily check that the projection of $V(f) \cap W(T)$ on the u-space is given by $u^2 + 3u + 1 = 0$, thus $u = 0$ does not lead to any points in this intersection.

However, some roots of a common factor of $\mathrm{res}(T, f)$ and $\mathrm{res}(T, h_T)$ may lead to a common point of $V(f)$ and $W(T)$. Consider now an example with three polynomials in four variables $u_1 < u_2 < x_1 < x_2$:

$$t_2 = x_1x_2^2 + x_2 + u_1, \quad t_1 = x_1(x_1 - 1) + u_1u_2, \quad \text{and} \quad f = x_2 + x_1 - 1.$$

The polynomials t_2, t_1 form a regular chain T and we have $\mathrm{res}(T, h_T) = u_1u_2$. Moreover, the iterated resultant $\mathrm{res}(T, f)$ is given by

$$\mathrm{res}(T, f) = u_1 \left({u_2}^3 {u_1}^2 + 2\,u_1\,{u_2}^2 + u_1\,u_2 + u_1 + u_2 + 1\right),$$

and one can check that the point of coordinates $(u_1, u_2, x_1, x_2) = (0, -1, 1, 0)$ belongs to $V(f) \cap W(T)$. Thus, the "bad specialization condition" $u_1 = 0$ leads in this case to a point of $V(f) \cap W(T)$.

Another feature of the common factors of $\mathrm{res}(T, f)$ and $\mathrm{res}(T, h_T)$ is that they may appear as large powers in the irreducible factorization of $\mathrm{res}(T, f)$, as we shall see with Theorem 8. In fact, they are the cause of expression swell in iterated resultant computations. In practice, roots of $\mathrm{res}(T, h_T)$ will often not extend to points of $V(f) \cap W(T)$ in situations where expression swell is a bottleneck. To be more precise, and giving a generic example, consider a regular sequence $f_1, \ldots, f_n$ of polynomials in $\mathbf{k}[x_1, \ldots, x_n]$. Assume that a triangular decomposition of the polynomial system $f_1 = \cdots = f_n = 0$ is being computed incrementally by one of the algorithms described in Lazard (1991),

Moreno Maza (1999) and Chen and Moreno Maza (2011). Let T be any one-dimensional regular chain obtained after solving $f_1 = \cdots = f_{n-1} = 0$. Since f_n is regular w.r.t. the ideal $\langle f_1, \ldots, f_{n-1}\rangle$, it is also regular w.r.t. the saturated ideal of T. Since h_T is also regular w.r.t. $\mathrm{sat}(T)$, roots of $r = \mathrm{res}(T, h_T)$ will not extend to points of $V(f_n) \cap W(T)$ unless the hypersurfaces $r = 0$ and $f_n = 0$ intersect on the quasi-component $W(T)$, which will not happen "generically". Unfortunately, in the process of solving $f_1 = \cdots = f_n = 0$, computing $V(f_n) \cap W(T)$ (for any one-dimensional regular chain T obtained after solving $f_1 = \cdots = f_{n-1} = 0$) is the most challenging step due to intermediate expression swell, in particular considering the degree of the polynomials in T w.r.t. the free variable of T. These observations lead us to the following problem.

Problem 1. Let T be a one-dimensional regular chain and a polynomial f regular w.r.t. $\mathrm{sat}(T)$. Assume that no zeros of $\mathrm{res}(T, h_T)$ extend to a point of $V(f) \cap W(T)$. Let us call *useful part* of $\mathrm{res}(T, f)$ its irreducible factors that are not factors of $\mathrm{res}(T, h_T)$. Then, the problem is how to compute the useful part of $\mathrm{res}(T, f)$ without computing the whole $\mathrm{res}(T, f)$, since this latter may have a much larger degree than the former.

To address this problem, we proceed in three steps. In Section 6.1, we start by establishing a *Poisson Product Formula* for the iterated resultant $\mathrm{res}(T, f)$, assuming that T is a zero-dimensional regular chain of $\mathbf{k}[\mathbf{x}]$ where $\mathbf{x}$ stands for n ordered variables $x_1 < x_2 < \cdots < x_n$. Two equivalent product formulas are, in fact, stated in Theorem 8. Similar formulas are well-known in the context of multipolynomial resultants of homogeneous polynomials, see Chapter 3 in the landmark textbook *Using Algebraic Geometry* by Cox et al. (1998). Our proofs are, however, based on repeated use of the elementary version of Poisson's Product Formula, that is, the one for the resultant of two univariate polynomials.

In Section 6.2, we move to the positive dimensional case by assuming that $\mathbf{k}$ is a field of rational functions. We associate T with two remarkable regular chains denoted by $\widehat{T}$ and $\widetilde{T}$. Proposition 10 implies that, under the assumption of Problem 1 the three regular chains T, $\widehat{T}$ and $\widetilde{T}$ play an equivalent role for the purpose of computing $V(f) \cap W(T)$. Applying to $\widehat{T}$ and $\widetilde{T}$ the results of Section 6.1 brings a better insight on the expression swell issue. Moreover, this suggests that working with $\widetilde{T}$ instead of T is the way for reducing useless expression swell and solving Problem 1.

In Section 6.3, we are now under the hypotheses of Problem 1. That is, T is a one-dimensional regular chain, with a free variable $\underline{u}$, and no zeros of $\mathrm{res}(T, h_T)$ extend to a point $V(f) \cap W(T)$. We explain how to take advantage of $\widetilde{T}$, without computing it, in order to obtain the projection on the u-space of $V(f) \cap W(T)$ at much better cost than through a direct computation of $\mathrm{res}(T, f)$.

Finally, Section 6.4 contains two detailed illustrative examples while Section 6.5 is an experimental report on a variety of test examples. As mentioned in the introduction, these show that the proposed techniques, on sufficiently large test cases, reduce the size of the computed iterated resultants by a factor of 50, leading to a running time speedup of three orders of magnitude.

6.1. *A Poisson product formula for iterated resultants*

Let T be a zero-dimensional regular chain of $\mathbf{k}[\mathbf{x}]$, for $\mathbf{x} = x_1 < \cdots < x_n$. We denote by $V_M(T)$ the multiset of the zeros of T, where each zero of T appears a number times equal to its local multiplicity as defined in Chapter 4 of Cox et al. (1998). For $i = 1 \cdots n$, we denote respectively by t_i, h_i, r_i, d_i

- the polynomial of T whose main variable is x_i,
- the initial of t_i,
- the iterated resultant $\mathrm{res}(\{t_1, \ldots, t_{i-1}\}, h_i)$,
- the total degree of t_i.

In particular, we have $r_1 = h_1$. The following concept is standard but appears under different names in the literature.

Definition 2. A zero dimensional regular chain $N \subset \mathbf{k}[\mathbf{x}]$ is called a *normalized form* of T if N and T generate the same ideal of $\mathbf{k}[\mathbf{x}]$ and if for each $f \in N$ we have $\mathrm{init}(f) = 1$. Observe that N is a lexicographic Gröbner basis, but not necessarily a minimal one.

Example 2. The existence of a normalized form of T follows easily from the fact that the h_i is invertible modulo the ideal $\langle t_1, \ldots, t_{i-1}\rangle$, for $i = 2 \cdots n$. Note that $h_1 \in \mathbf{k}$, so its invertibility is immediate. Computing the inverse of h_i modulo $\langle t_1, \ldots, t_{i-1}\rangle$, for $i = 2 \cdots n$, is achieved by computing an extended resultant of h_i and t_{i-1} modulo $\langle t_1, \ldots, t_{i-2}\rangle$, that is, by computing $a_i, b_i \in \mathbf{k}[x_1, \ldots, x_{i-1}]$ such that we have

$$a_i h_i + b_i t_{i-1} \equiv r_i \mod \langle t_1, \ldots, t_{i-2}\rangle$$

where $r_i = \mathrm{res}(\{t_1, \ldots, t_{i-1}\}, h_i)$. Then, we deduce

$$\frac{a_i}{r_i} h_i \equiv 1 \mod \langle t_1, \ldots, t_{i-1}\rangle. \tag{9}$$

We define $\widetilde{t_1} = t_1/\mathrm{init}(t_1)$. Then, for $i = 2 \cdots n$, we denote by $\widetilde{t_i}$ the normal form (in the sense of Gröbner bases) of $\frac{a_i}{r_i} t_i$ modulo the ideal $\langle t_1, \ldots, t_{i-1}\rangle$. It is easy to check that $\widetilde{T} = \{\widetilde{t_1}, \ldots, \widetilde{t_n}\}$ is a normal form of T in the sense of Definition 2. Moreover, it is a reduced minimal lexicographic Gröbner basis of $\langle T\rangle$.

From now on, we will denote by $\widetilde{T}$ a normal form of T and by $\mathrm{NF}(f, \widetilde{T})$ the normal form of a polynomial f w.r.t. $\widetilde{T}$. The following observation is a first version of *Poisson's Product Formula* for iterated resultants. We give a direct proof to keep our presentation self-contained. However, this result could be derived from those of Chapter 2 in Cox et al. (1998).

Proposition 8. *For every polynomial $f \in \mathbf{k}[\mathbf{x}]$, we have*

$$\mathrm{res}(\widetilde{T}, f) = \prod_{\alpha \in V_M(T)} f(\alpha). \tag{10}$$

Proof. If $n = 1$, this is the elementary Poisson's Product Formula for univariate polynomials over a field. Otherwise, we have

$$\begin{aligned}
r &= \mathrm{res}(\widetilde{T}_{<x_n}, \mathrm{res}(\widetilde{t_n}, f, x_n)) \\
&= \prod_{\beta \in \pi_{V_M(T)}} \mathrm{res}(\widetilde{t_n}, f, x_n)(\beta) && \text{By induction} \\
&= \prod_{\beta \in \pi_{V_M(T)}} \mathrm{res}(\widetilde{t_n}(\beta), f(\beta), x_n) && \text{By specialization property and } \mathrm{init}(\widetilde{t_i}) = 1 \\
&= \prod_{\beta \in \pi_{V_M(T)}} \prod_{\gamma \in V_M(t_n(\beta, x_n))} f(\beta, \gamma) && \text{By induction} \\
&= \prod_{\alpha \in V_M(T)} f(\alpha).
\end{aligned}$$

where π denotes the projection $(x_1, \ldots, x_n) \longmapsto (x_1, \ldots, x_{n-1})$ from $\mathbf{K}^n$ to $\mathbf{K}^{n-1}$. □

The next proposition is also not new and is certainly used in all implementations of iterated resultant computation. The point that we want to make here is that, by replacing f with $\mathrm{NF}(f, \widetilde{T})$ in computing $\mathrm{res}(\widetilde{T}, f)$ one can efficiently control monomial expression swell. More precisely, when computing $\mathrm{res}(\widetilde{T}, f)$, intermediate polynomials can be kept reduced w.r.t. $\widetilde{T}$, in the sense of Gröbner basis. Of course, when computing $\mathrm{res}(T, f)$, intermediate polynomials can be kept reduced w.r.t. T, in the sense of pseudo-division. But this is more tricky to implement and computationally more expensive to achieve, since the initials of T have to be handled.

Proposition 9. *For every polynomial $f \in \mathbf{k}[\mathbf{x}]$, we have*

$$\mathrm{res}(\widetilde{T}, f) = \mathrm{res}(\widetilde{T}, \mathrm{NF}(f, \widetilde{T})). \tag{11}$$

Proof. We denote $\mathrm{res}(\widetilde{T}, f)$ and $\mathrm{res}(\widetilde{T}, \mathrm{NF}(f, \widetilde{T}))$ respectively by r and $\widetilde{r}$. We start with the case $n = 1$ and denote by $\mathrm{rem}(f, t_1)$ the remainder in the Euclidean division of f by t_1. Then, we have

$$\begin{aligned}
\widetilde{r} &= \mathrm{res}(\widetilde{t_1}, \mathrm{NF}(f, \{\widetilde{t_1}\}), x_1) \\
&= \mathrm{res}(\widetilde{t_1}, \mathrm{rem}(f, \widetilde{t_1}), x_1) \\
&= \prod_{\alpha \in V_M(t_1)} \mathrm{rem}(f, \widetilde{t_1})(\alpha) && \text{By Proposition 8} \\
&= \prod_{\alpha \in V_M(t_1)} f(\alpha) && \text{By definition of } \widetilde{t_1} \\
&= r. && \text{By Proposition 8.}
\end{aligned}$$

For $n > 1$, we write $g = \mathrm{res}(\widetilde{t_n}, \mathrm{NF}(f, \widetilde{T}), x_n)$. Then, we have

$$\begin{aligned}
\widetilde{r} &= \mathrm{res}(\widetilde{T}, \mathrm{NF}(f, \widetilde{T})) \\
&= \mathrm{res}(\widetilde{t_1}, \ldots, \widetilde{t_{n-1}}, g) && \text{By definition of } \mathrm{res}(\widetilde{T}, \cdot) \\
&= \prod_{\alpha \in V_M(t_1, \ldots, t_{n-1})} g(\alpha) && \text{By Proposition 8} \\
&= \prod_{\alpha \in V_M(t_1, \ldots, t_{n-1})} \mathrm{res}(\widetilde{t_n}, \mathrm{NF}(f, \widetilde{T}), x_n)(\alpha) && \text{By definition of } g \\
&= \prod_{\alpha \in V_M(t_1, \ldots, t_{n-1})} \mathrm{res}(\widetilde{t_n}(\alpha), \mathrm{NF}(f, \widetilde{T})(\alpha), x_n) && \text{By specialization property and} \\
& && \mathrm{init}(\widetilde{t_n}) = 1 \\
&= \prod_{\alpha \in V_M(t_1, \ldots, t_{n-1})} \mathrm{res}(\widetilde{t_n}(\alpha), f(\alpha), x_n) && \text{From the case } n = 1 \\
&= \prod_{(\alpha, \beta) \in V_M(T)} f(\alpha, \beta) && \text{By Proposition 8} \\
&= r. \quad \square
\end{aligned}$$

Theorem 8 is a more general version of *Poisson's Product Formula* for iterated resultants. It is stated for T, which may have non-constant initials, instead of $\widetilde{T}$ whose initials are all 1. This latter product formula gives more insight on the expression swell occurring when computing $\mathrm{res}(T, f)$. From now on, we fix a polynomial $f \in \mathbf{k}[\mathbf{x}]$ and we define

- $e_n = \deg(f, x_n)$,
- $f_i = \mathrm{res}(t_{i+1}, \ldots, t_n, f)$, for $0 \le i \le n-1$,
- $e_i = \deg(f_i, x_i)$, for $1 \le i \le n-1$.

Theorem 8. *If $f \in \mathbf{k}[\mathbf{x}]$ is a non-constant polynomial whose initial is regular w.r.t. $\langle T \rangle$, then we have*

$$\mathrm{res}(T, f) = h_1^{e_1} \left(\prod_{\alpha_1 \in V_M(t_1)} h_2(\alpha_1) \right)^{e_2} \cdots \left(\prod_{\beta \in V_M(t_1, \ldots, t_{n-1})} h_n(\beta) \right)^{e_n} \left(\prod_{\alpha \in V_M(T)} f(\alpha) \right). \tag{12}$$

Equivalently, we have

$$\mathrm{res}(T, f) = h_1^{e_1} \left(\mathrm{res}(\{\widetilde{t_1}\}, h_2)\right)^{e_2} \cdots \left(\mathrm{res}(\{\widetilde{t_1}, \ldots, \widetilde{t_{n-1}}\}, h_n)\right)^{e_n} \mathrm{res}(\widetilde{T}, f). \tag{13}$$

Proof. We denote by r the iterated resultant $\mathrm{res}(T, f)$. If $n = 1$, the relation $r = \mathrm{res}(T, f, x_1) = h_1^{\deg(f)} \prod_{\alpha \in V_M(T)} f(\alpha)$, is the (less) elementary version of Poisson's Product Formula where the "left"

polynomial is not monic. Consider now the case $n = 2$. Recall that $e_1 = \deg(\mathrm{res}(t_2, f), x_1)$ and $e_2 = \deg(f, x_2)$. Then, we have

$$\begin{aligned}
r &= \mathrm{res}(t_1, t_2, f) \\
&= \mathrm{res}(t_1, \mathrm{res}(t_2, f)) \\
&= h_1^{e_1} \prod_{\alpha_1 \in V_M(t_1)} \mathrm{res}(t_2, f)(\alpha_1) && \text{From } n = 1 \\
&= h_1^{e_1} \prod_{\alpha_1 \in V_M(t_1)} \mathrm{res}(t_2(\alpha_1, x_2), f(\alpha_1, x_2)) && \text{Since } \mathrm{init}(f) \text{ is regular w.r.t. } \mathrm{sat}(T) \\
&= h_1^{e_1} \prod_{\alpha_1 \in V_M(t_1)} \left(h_2(\alpha_1)^{e_2} \prod_{\alpha_2 \in V_M(t_2(\alpha_1, x_2))} f(\alpha_1, \alpha_2) \right) && \text{From } n = 1 \\
&= h_1^{e_1} \left(\prod_{\alpha_1 \in V_M(t_1)} h_2(\alpha_1) \right)^{e_2} \left(\prod_{\alpha \in V_M(t_1, t_2)} f(\alpha) \right). && \text{Regrouping factors}
\end{aligned}$$

More generally, we have

$$\begin{aligned}
r &= \mathrm{res}(t_1, \ldots, t_{n-1}, t_n, f) \\
&= \mathrm{res}(t_1, \ldots, t_{n-1}, \mathrm{res}(t_n, f)) \\
&= h_1^{e_1} \left(\prod_{\alpha_1 \in V_M(t_1)} h_2(\alpha_1) \right)^{e_2} \cdots \left(\prod_{\gamma \in V_M(t_1, \ldots, t_{n-2})} h_{n-1}(\gamma) \right)^{e_{n-1}} \cdot \\
&\quad \left(\prod_{\beta \in V_M(t_1, \ldots, t_{n-1})} \mathrm{res}(t_n, f)(\beta) \right) \\
&= h_1^{e_1} \left(\prod_{\alpha_1 \in V_M(t_1)} h_2(\alpha_1) \right)^{e_2} \cdots \left(\prod_{\gamma \in V_M(t_1, \ldots, t_{n-2})} h_{n-1}(\gamma) \right)^{e_{n-1}} \cdot \\
&\quad \left(\prod_{\beta \in V_M(t_1, \ldots, t_{n-1})} h_n(\beta)^{e_n} \prod_{\alpha_n \in V_M(t_n(\beta, x_n))} f(\beta, \alpha_n) \right) \\
&= h_1^{e_1} \left(\prod_{\alpha_1 \in V_M(t_1)} h_2(\alpha_1) \right)^{e_2} \cdots \left(\prod_{\beta \in V_M(t_1, \ldots, t_{n-1})} h_n(\beta) \right)^{e_n} \left(\prod_{\alpha \in V_M(T)} f(\alpha) \right). \qquad \square
\end{aligned}$$

6.2. *Identifying the "useful" part of an iterated resultant*

From now on we assume that $\mathbf{k}$ is a field $\mathbf{c}(\mathbf{u})$ of rational functions in variables $\mathbf{u} = u_1, \ldots, u_d$ and with coefficients in a field $\mathbf{c}$. Let T be a zero-dimensional regular chain in $\mathbf{k}[\mathbf{x}]$. We also assume that $T \subset \mathbf{c}[\mathbf{u}][\mathbf{x}]$ holds, that is, all denominators of T are equal to 1. Since $\mathrm{sat}(T) \cap \mathbf{c}(\mathbf{u}) = \langle 0 \rangle$ holds (see Proposition 5) the regular chain T can be regarded both as an element of $\mathbf{c}[\mathbf{u}][\mathbf{x}]$ and $\mathbf{c}(\mathbf{u})[\mathbf{x}]$. Next we introduce two regular chains $\widehat{T}$ and $\widetilde{T}$ associated with T. The latter is a normal form of T (regarded as a regular chain of $\mathbf{c}(\mathbf{u})[\mathbf{x}]$) in the sense of Section 6.1 while $\widehat{T}$ will essentially be obtained from $\widetilde{T}$ by clearing out denominators. Thus $\widehat{T}$ is a regular chain of $\mathbf{c}[\mathbf{u}][\mathbf{x}]$.

Definition 3. Let $T \subset \mathbf{c}[\mathbf{u}][\mathbf{x}]$ be a regular chain. We define $\widehat{T} := \{\widehat{t_1}, \ldots, \widehat{t_n}\}$ as follows: (1) $\widehat{t_1} = t_1$; (2) for $i := 2, \ldots, n$, let $r_i = \mathrm{res}(\{t_1, \ldots, t_{i-1}\}, h_i)$ and compute $a_i, b_1, \ldots, b_{i-1}$ such

that $r_i = a_i h_i + b_1 t_1 + \cdots + b_{i-1} t_{i-1}$; let

$$\begin{aligned}\widehat{t_i} &= a_i t_i + \left(\sum_{j=1}^{i-1} b_j t_j\right) \mathrm{rank}(t_i)\\ &= a_i(h_i \mathrm{rank}(t_i) + \mathrm{tail}(t_i)) + \left(\sum_{j=1}^{i-1} b_j t_j\right) \mathrm{rank}(t_i)\\ &= r_i \cdot \mathrm{rank}(t_i) + a_i \mathrm{tail}(t_i).\end{aligned}$$

We define $\widetilde{T} := \{\widetilde{t_1}, \ldots, \widetilde{t_n}\}$ as follows: $\widetilde{t_i} = \frac{\widehat{t_i}}{r_i} = \mathrm{rank}(t_i) + \frac{a_i}{r_i}\mathrm{tail}(t_i)$, for $i = 1, \ldots, n$. Observe that $\widehat{T}$ and $\widetilde{T}$ are not uniquely defined, making the results below more general.

Lemma 4. *We have* $\mathrm{sat}(T) = \mathrm{sat}(\widehat{T})$ *and* $\widetilde{T}$ *is a normalized form of* T *over* $\mathbf{c}(\mathbf{u})$.

Proof. We first prove $\mathrm{sat}(T) = \mathrm{sat}(\widehat{T})$ by induction. It obviously holds for $n = 1$. Assume that $\mathrm{sat}(t_1, \ldots, t_{n-1}) = \mathrm{sat}(\widehat{t_1}, \ldots, \widehat{t_{n-1}})$ holds. From Definition 3, we have $\widehat{t_n} = a_n t_n + \left(\sum_{j=1}^{n-1} b_j t_j\right) \mathrm{rank}(t_n)$. Thus $\widehat{t_n}$ belongs to $\langle T\rangle$ and thus $\widehat{t_n} \in \mathrm{sat}(T)$ holds. By induction, we have $\langle\widehat{T}\rangle \subseteq \mathrm{sat}(T)$. Saturating both sides by $r_1 \cdots r_n$, we deduce $\mathrm{sat}(\widehat{T}) \subseteq \mathrm{sat}(T)$. Similarly, we have $a_n t_n \in \mathrm{sat}(\widehat{T})$. From the relation $r_n = a_n h_n + \sum_{j=1}^{n-1} b_j t_j$, we know that a_n is regular modulo $\mathrm{sat}(\widehat{t_1}, \ldots, \widehat{t_{n-1}})$ and thus regular modulo $\mathrm{sat}(\widehat{T})$. Thus we have $t_n \in \mathrm{sat}(\widehat{T})$. By induction, we have $\langle T\rangle \subseteq \mathrm{sat}(\widehat{T})$. Saturating both sides by $h_1 \cdots h_n$, we deduce $\mathrm{sat}(T) \subseteq \mathrm{sat}(\widehat{T})$. Therefore $\mathrm{sat}(T) = \mathrm{sat}(\widehat{T})$ holds. By Definitions 2 and 3, we conclude that $\widetilde{T}$ is a normalized form of T over $\mathbf{c}(\mathbf{u})$. $\square$

Let $\mathbf{C}$ be the algebraic closure of $\mathbf{c}$. In the sequel, algebraic varieties are taken in $\mathbf{C}^{d+n}$ and $\pi_{\mathbf{u}}$ denotes the projection $(u_1, \ldots, u_d, x_1, \ldots, x_n) \longmapsto (u_1, \ldots, u_d)$ from $\mathbf{C}^{d+n}$ to $\mathbf{C}^d$.

Proposition 10. *Let* $f \in \mathbf{c}[\mathbf{u}, \mathbf{x}]$. *Then, we have*

$$\pi_{\mathbf{u}}(V(\mathrm{res}(T, f)) \setminus V(r_1 \cdots r_n)) = \pi_{\mathbf{u}}(W(T) \cap V(f) \setminus V(r_1 \cdots r_n)). \tag{14}$$

Moreover, we have

$$\pi_{\mathbf{u}}(V(\mathrm{res}(\widehat{T}, f)) \setminus V(r_1 \cdots r_n)) = \pi_{\mathbf{u}}(V(\mathrm{res}(T, f)) \setminus V(r_1 \cdots r_n)). \tag{15}$$

Furthermore, we have

$$\pi_{\mathbf{u}}(V(\mathrm{res}(\widehat{T}, f)) \setminus V(r_1 \cdots r_n)) = \pi_{\mathbf{u}}(V(\mathrm{numer}(\mathrm{res}(\widetilde{T}, f))) \setminus V(r_1 \cdots r_n)). \tag{16}$$

Proof. We prove the first claim, that is, Formula (14). We denote $V(\mathrm{res}(T, f)) \setminus V(r_1 \cdots r_n)$ and $W(T) \cap V(f) \setminus V(r_1 \cdots r_n)$ by A and B, respectively. Observe that there exist polynomials $a_n, b_n, \ldots, b_1 \in \mathbf{c}[\mathbf{u}, \mathbf{x}]$ such that we have

$$a_n f + b_n t_n + \cdots + b_1 t_1 = \mathrm{res}(T, f).$$

Thus we have $W(T) \cap V(f) \subseteq V(\mathrm{res}(T, f))$ which implies that $\pi_{\mathbf{u}}(B) \subseteq \pi_{\mathbf{u}}(A)$ holds. The reversed inclusion follows from the *Extension Theorem*. Indeed, let $(\zeta_1, \ldots, \zeta_d) \in \mathbf{C}^d$ be a point of $\pi_{\mathbf{u}}(A)$. Since $(\zeta_1, \ldots, \zeta_d)$ is a zero of $\mathrm{res}(f_1, t_1)$ which does not cancel $r_1 = h_1$, this zero can be extended to a common zero of f_1 and t_1, say $(\zeta_1, \ldots, \zeta_d, \zeta_{d+1}) \in \mathbf{C}^{d+1}$. Similarly, since $(\zeta_1, \ldots, \zeta_d, \zeta_{d+1})$ is zero of $\mathrm{res}(f_2, t_2)$ which does not cancel r_2, and thus h_2, this zero can be extended to a common zero of f_2 and t_2, say $(\zeta_1, \ldots, \zeta_d, \zeta_{d+1}, \zeta_{d+2}) \in \mathbf{C}^{d+1}$. Continuing in this manner, we prove that $\pi_{\mathbf{u}}(A) \subseteq \pi_{\mathbf{u}}(B)$ holds.

The second claim, that is, Formula (15), is essentially a consequence of the definition of $\widehat{T}$ and Lemma 4. Indeed, by Lemma 4, we have $\mathrm{sat}(T) = \mathrm{sat}(\widehat{T})$. Now, recall that for an arbitrary

regular chain C we have $W(C) \setminus V(h_C) = \overline{W(C)} \setminus V(h_C) = V(\mathrm{sat}(C)) \setminus V(h_C)$. Therefore, we deduce

$$W(T) \setminus V(r_1 \cdots r_n) = W(\widehat{T}) \setminus V(r_1 \cdots r_n),$$

and the conclusion follows from Formula (14).

The third claim follows from Definition 3. Indeed, in this case, the definition yields

$$\mathrm{res}(\widehat{T}, f) = r_1^{m_1} \cdots r_1^{m_n} \mathrm{numer}(\mathrm{res}(\widetilde{T}, f)), \tag{17}$$

where $m_1, \ldots, m_n$ are integers, possibly negative, from which Formula (16) is easily derived. This completes the proof. □

6.3. Computing the "useful" part of an iterated resultant

We are now under the hypotheses of Problem 1. It follows from Section 6.2 that $\mathrm{numer}(\mathrm{res}(\widetilde{T}, f))$ can replace $\mathrm{res}(T, f)$ for the purpose of computing $\pi_{\mathbf{u}}(W(T) \cap V(f) \setminus V(r_1 \cdots r_n))$, which, under our hypotheses, is simply $\pi_{\mathbf{u}}(W(T) \cap V(f))$. Moreover, $\mathrm{numer}(\mathrm{res}(\widetilde{T}, f))$ is expected to have a smaller degree than $\mathrm{res}(\widehat{T}, f)$ and $\mathrm{res}(T, f)$.

To compute $\mathrm{numer}(\mathrm{res}(\widetilde{T}, f))$ we proceed by evaluation and interpolation. Indeed, by specializing T to a zero-dimensional regular chain, say $T(\alpha)$, we can compute $\widetilde{T(\alpha)}$ at a reasonable cost and then take advantage of Proposition 9 to efficiently compute $\mathrm{res}(\widetilde{T(\alpha)}, f(\alpha))$. Obtaining images of $\mathrm{numer}(\mathrm{res}(\widetilde{T}, f))$ at sufficiently many evaluation points α's will bring $\mathrm{numer}(\mathrm{res}(\widetilde{T}, f))$ (by rational function interpolation) without computing $\widetilde{T}$ itself. However, proceeding by evaluation and interpolation requires

- a "commutation diagram", which is provided by Proposition 11.
- a bound on the number of evaluations.

Let us discuss this second point. One could easily derive a bound for the degree of the numerator and a bound for the degree of the denominator of $\mathrm{res}(\widetilde{T}, f)$ from Formula (17) in the proof of Proposition 10. But this bound would be very pessimistic.

Instead, we recall that $W(T) \cap V(f)$ is meant to be part of a triangular decomposition of a regular sequence with finitely many solutions. For this reason, it is reasonable to use the Bézout bound of the input system (i.e. the product of the total degrees of the input polynomials) for each of the numerator and the denominator of $\mathrm{res}(\widetilde{T}, f)$.

Proposition 11. *Let $\alpha \in \mathbf{C}^d$ such that $\prod_{i=1}^n r_i(\alpha) \neq 0$. Then we have*

$$\mathrm{res}(\widetilde{T}, f)(\alpha) = \mathrm{res}(\widetilde{T}(\alpha), f(\alpha)) = \mathrm{res}(\widetilde{T(\alpha)}, f(\alpha)).$$

Proof. By Lemma 4, $\widetilde{T}$ is a normalized form of T over $\mathbf{c}(\mathbf{u})$. Therefore $\widetilde{T}(\alpha)$ is a normalized form of $T(\alpha)$ over $\mathbf{C}$. By Proposition 8, we deduce that $\mathrm{res}(\widetilde{T(\alpha)}, f(\alpha)) = \mathrm{res}(\widetilde{T}(\alpha), f(\alpha))$ holds. Now we prove by induction on n that $\mathrm{res}(\widetilde{T}, f)(\alpha) = \mathrm{res}(\widetilde{T}(\alpha), f(\alpha))$ holds. If $n = 1$, its correctness follows from Theorem 4. Otherwise, we have

$$\begin{aligned}
\mathrm{res}(\widetilde{T}, f)(\alpha) &= \mathrm{res}(\widetilde{t_1}, \ldots, \widetilde{t_{n-1}}, \widetilde{t_n}, f)(\alpha) \\
&= \mathrm{res}(\widetilde{t_1}, \ldots, \widetilde{t_{n-1}}, \mathrm{res}(\widetilde{t_n}, f))(\alpha) \\
&= \mathrm{res}(\widetilde{t_1}(\alpha), \ldots, \widetilde{t_{n-1}}(\alpha), \mathrm{res}(\widetilde{t_n}, f)(\alpha)) && \text{By induction} \\
&= \mathrm{res}(\widetilde{t_1}(\alpha), \ldots, \widetilde{t_{n-1}}(\alpha), \mathrm{res}(\widetilde{t_n}(\alpha), f(\alpha))) && \text{By specialization property and} \\
& && \mathrm{init}(\widetilde{t_n}) = 1 \\
&= \mathrm{res}(\widetilde{T}(\alpha), f(\alpha)). \quad \square
\end{aligned}$$

6.4. Examples

We provide two detailed examples illustrating the effectiveness of the techniques proposed in Section 6.3. They differ by the fact that, for the first example, but not for the second one, the regular chain T satisfies $T = \widehat{T}$.

Example 3. In MAPLE, we randomly generate three polynomials g_1, g_2 and g_3.

$$\begin{aligned} g_1 &:= y^2 - 2\,z^2x - 2\,z^2x^2 - y^4 + 1 \\ g_2 &:= 2\,z^2x^2 + 2\,z^2y - 2\,z^2 - z + 1 \\ g_3 &:= -y + y^3 + 2\,z^3 - z^2x^2 + z^4 + 1. \end{aligned}$$

For $z > y > x$, the `Triangularize` command of `RegularChains` library in MAPLE takes $\{g_1, g_2\}$ as input and returns a one-dimensional regular chain $T := \{t_z, t_y\}$, where

$$t_z := (x + x^2)z + y^5 + (-1 + x^2)y^4 - y^3 + (1 - x^2)y^2 - y - x - 2\,x^2 + 1,$$

and

$$\begin{aligned} t_y \;:=\; & 2\,y^{10} + (4\,x^2 - 4)y^9 + (-4\,x^2 - 2 + 2\,x^4)y^8 \\ & + (8 - 8\,x^2)y^7 + (-6 - 4\,x^4 + 8\,x^2)y^6 + (4 - 4\,x - 8\,x^2)y^5 \\ & + (2 + 5\,x - 4\,x^3 - 6\,x^4 + 9\,x^2)y^4 + (4\,x + 12\,x^2 - 8)y^3 \\ & + (-5\,x - 13\,x^2 + 8\,x^4 + 6 + 4\,x^3)y^2 + (4\,x + 8\,x^2 - 4)y \\ & + 2 - 7\,x^2 + 8\,x^3 - 5\,x + 8\,x^4. \end{aligned}$$

We first compute $\mathrm{res}(T, g_3)$, which is as follows

$$\begin{aligned} \mathrm{res}(T, g_3) \;:=\; & x^{36}(x+1)^{36}(160000\,x^{32} - 1996800\,x^{31} + 1865216\,x^{30} \\ & + 39076352\,x^{29} - 13755136\,x^{28} - 292989952\,x^{27} - 492288\,x^{26} \\ & + 1266265600\,x^{25} + 411825152\,x^{24} - 3352744704\,x^{23} - 2254328832\,x^{22} \\ & + 4741870720\,x^{21} + 5431924832\,x^{20} - 805462400\,x^{19} - 5314139328\,x^{18} \\ & - 9219790080\,x^{17} - 3910480928\,x^{16} + 14746844160\,x^{15} + 16366917424\,x^{14} \\ & - 4208599168\,x^{13} - 10656443328\,x^{12} - 2971537424\,x^{11} - 1632713152\,x^{10} \\ & + 674535336\,x^9 + 3977359985\,x^8 + 491023040\,x^7 - 1666879386\,x^6 \\ & + 24976342\,x^5 + 366817077\,x^4 - 61683960\,x^3 - 30489670\,x^2 \\ & + 9251686\,x - 661849). \end{aligned}$$

Proposition 11 suggests that we can compute $\mathrm{res}(\widetilde{T}, g_3)$ by evaluation and rational interpolation. Since the Bézout bound of the input system is 64, we evaluate T at $2 \times 64 + 1$ points $\alpha_1, \ldots, \alpha_{129}$ chosen such that T specializes well at each of them. Let $\beta_i := \mathrm{res}(\widetilde{T(\alpha_i)}, f(\alpha_i))$, $i = 1, \ldots, 129$. By applying rational interpolation to the (α_i, β_i)'s, we obtain $\mathrm{res}(\widetilde{T}, g_3)$:

$$\begin{aligned} \mathrm{res}(\widetilde{T}, g_3) \;:=\; & \frac{1}{1048576\,x^4(x+1)^4}(160000\,x^{32} - 1996800\,x^{31} + 1865216\,x^{30} \\ & + 39076352\,x^{29} - 13755136\,x^{28} - 292989952\,x^{27} - 492288\,x^{26} \\ & + 1266265600\,x^{25} + 411825152\,x^{24} - 3352744704\,x^{23} - 2254328832\,x^{22} \\ & + 4741870720\,x^{21} + 5431924832\,x^{20} - 805462400\,x^{19} - 5314139328\,x^{18} \\ & - 9219790080\,x^{17} - 3910480928\,x^{16} + 14746844160\,x^{15} + 16366917424\,x^{14} \\ & - 4208599168\,x^{13} - 10656443328\,x^{12} - 2971537424\,x^{11} - 1632713152\,x^{10} \\ & + 674535336\,x^9 + 3977359985\,x^8 + 491023040\,x^7 - 1666879386\,x^6 \\ & + 24976342\,x^5 + 366817077\,x^4 - 61683960\,x^3 - 30489670\,x^2 \\ & + 9251686\,x - 661849). \end{aligned}$$

Note that the numerator of $\mathrm{res}(\widetilde{T}, g_3)$ is the "useful" part that we expect.

Next we verify Formula (13) of Theorem 8 for this example. We have $h_1 = 2$, $e_1 = \deg(\mathrm{res}(t_z, g_3), y) = 20$, $h_2 = x + x^2$, $e_2 = \deg(g_3, z) = 4$ and $\deg(\widetilde{t_y}, y) = 10$, which implies $h_1^{e_1} = 1048576$ and $\left(\mathrm{res}(\{\widetilde{t_y}\}, h_2, y)\right)^{e_2} = (x + x^2)^{40}$. Thus Formula (13) is verified.

Example 4. Let g_1, g_2 and g_3 be another group of randomly generated polynomials.

$$\begin{aligned} g_1 &:= 2z^2yx - zy^3 - 2yx + 1 \\ g_2 &:= -z^2x^2 + 2zyx^2 + z^2y + 2zy^2 + 1 \\ g_3 &:= 2x^2z + y^2 + 3. \end{aligned}$$

For $z > y > x$, Triangularize takes $\{g_1, g_2\}$ as input and returns a one-dimensional regular chain $T := \{t_z, t_y\}$, where

$$t_z := \left(y^4 + \left(-x^2 + 4x\right)y^3 + 4y^2x^3\right)z + 2y^2x + \left(-2x^3 - 1 + 2x\right)y + x^2,$$

and

$$\begin{aligned} t_y := {} & (4x - 1)y^7 + \left(-4x + 17x^2 - 2\right)y^6 + \left(-4x^5 - 4x^3 + 32x^4 - 8x\right)y^5 \\ & + \left(16x^6 - 16x^3 - 4x^2 + 2x^4\right)y^4 + \left(-8x^2 - 8x^5 + 4x + 8x^4\right)y^3 \\ & + \left(-4x^6 + 4x - 1 + 8x^4 - 4x^2 - 8x^3\right)y^2 + \left(4x^5 + 2x^2 - 4x^3\right)y - x^4. \end{aligned}$$

We first compute $\mathrm{res}(T, g_3)$, which is as follows

$$\begin{aligned} \mathrm{res}(T, g_3) := {} & 4x^{14}\left(64x^5 - 8x^4 - 40x^3 + 41x^2 - 12x + 4\right)^2 \\ & (4096x^{18} + 2048x^{17} - 9216x^{16} - 23552x^{15} - 47616x^{14} - 110912x^{13} \\ & + 15136x^{12} + 48624x^{11} - 71288x^{10} + 146328x^9 + 48736x^8 - 122015x^7 \\ & + 82217x^6 - 65270x^5 - 74670x^4 + 49738x^3 - 32183x^2 \\ & + 9476x - 2718)\left(x^2 - 2\right). \end{aligned}$$

Secondly, we compute $\mathrm{res}(T, h_T)$, which is

$$4x^{14}\left(64x^5 - 8x^4 - 40x^3 + 41x^2 - 12x + 4\right)^2(4x - 1)^7.$$

Since the Bézout bound of the input system is 48, we evaluate T at $2 \times 48 + 1$ points $\alpha_1, \ldots, \alpha_{97}$ chosen such that T specializes well at each of them. Let $\beta_i := \mathrm{res}(\widetilde{T(\alpha_i)}, f(\alpha_i))$, $i = 1, \ldots, 97$. By applying rational interpolation to the (α_i, β_i)'s, we obtain $\mathrm{res}(\widetilde{T}, g_3)$:

$$\begin{aligned} \mathrm{res}(\widetilde{T}, g_3) := {} & \frac{1}{(4x - 1)^2}(4096x^{18} + 2048x^{17} - 9216x^{16} - 23552x^{15} - 47616x^{14} \\ & - 110912x^{13} + 15136x^{12} + 48624x^{11} - 71288x^{10} + 146328x^9 + 48736x^8 \\ & - 122015x^7 + 82217x^6 - 65270x^5 - 74670x^4 + 49738x^3 - 32183x^2 \\ & + 9476x - 2718)\left(x^2 - 2\right). \end{aligned}$$

We observe that the numerator of $\mathrm{res}(\widetilde{T}, g_3)$ is the "useful" part that we expect.

6.5. Experimental results

In this section, we report experimental results for computing the "useful part" of the iterated resultant when the regular chain has dimension one.

A function for computing the "useful part" of the iterated resultant has been implemented, with the name IteratedResultantDim1, in the module `FastArithmeticTools` of the `RegularChains` library. The kernel of this function is implemented in C within the MODPN library, using the FFT-based

Table 1
Same regular chain but different polynomials.

i	Computing time (seconds)			Degree of output		
	$\mathrm{res}(f_i, T)$	$\mathrm{res}(f_i, \widetilde{T})$ (default bound)	$\mathrm{res}(f_i, \widetilde{T})$ (Bézout bound)	$\mathrm{res}(f_i, T)$	$\mathrm{res}(f_i, \widetilde{T})$ (default bound)	$\mathrm{res}(f_i, \widetilde{T})$ (Bézout bound)
2	0.540	0.516	0.044	224	16	16
3	2.328	1.080	0.084	336	24	24
4	7.388	1.576	0.120	448	32	32
5	19.482	3.220	0.244	560	40	40
6	51.415	4.288	0.324	672	48	48
7	121.944	5.816	0.428	784	56	56
8	279.158	7.920	0.584	896	64	64
9	608.082	14.573	1.072	1008	72	72
10	1234.849	19.034	1.392	1120	80	80

polynomial arithmetic together with the evaluation–interpolation method described in Section 6.3. This function takes as input a one-dimensional regular chain T and a polynomial f, which is regular modulo $\mathrm{sat}(T)$, and returns the numerator of $\mathrm{res}(f, \widetilde{T})$. This function can take a bound as an extra argument. By default, it uses the bound calculated from the product of the degrees of the polynomials in T and the degree of f. We refer to this bound as the *default bound*. In this experimentation, T will be generated from an input set of $n - 1$ dense polynomials. The product of the degrees of those polynomials is referred hereafter as the *Bézout bound*.

We randomly generated two groups of test examples. For both groups, we pick an FFT prime number p of machine-word size and with large Fourier degree, for instance $p = 962592769$. Then we conduct the computations over the finite field $\mathbb{Z}/p\mathbb{Z}$.

In the first group, we generate a one-dimensional regular chain T and 9 polynomials f_i, for $i = 2, \ldots, 10$, as follows:

(1) we randomly generate three dense polynomials, g_1, g_2, g_3, of $\mathbb{Z}/p\mathbb{Z}[x_1, x_2, x_3, x_4]$, all with total degree 2;
(2) we call $\mathsf{Triangularize}(g_1, g_2, g_3)$ to compute a Kalkbrener triangular decomposition of the system $\{g_1, g_2, g_3\}$ and call T the unique one-dimensional regular chain in the output;
(3) for $i = 2, \ldots, 10$, we randomly generate a dense polynomial f_i in $\mathbb{Z}/p\mathbb{Z}[x_1, x_2, x_3, x_4]$ with total degree i.

For each $i = 2, \ldots, 10$, we run the following three different computations on f_i and T:

(a) compute $\mathrm{res}(f_i, T)$ by successively calling MAPLE's $\mathsf{Resultant}$ function;
(b) compute $\mathrm{res}(f_i, T)$ by calling the function $\mathsf{IteratedResultantDim1}$ with the default bound calculated from T and f_i;
(c) compute $\mathrm{res}(f_i, \widetilde{T})$ by calling the function $\mathsf{IteratedResultantDim1}$ with Bézout bound calculated from g_1, g_2, g_3 and f_i.

The experimental results of the above computations are reported in Table 1. For all i, $i = 2, \ldots, 10$, we see that the degree of $\mathrm{res}(f_i, \widetilde{T})$ is 14 times smaller than that of $\mathrm{res}(f_i, T)$. It is interesting to observe that the computing time of $\mathrm{res}(f_i, \widetilde{T})$ with Bézout bound is also approximately 14 times smaller than that with the default bound. An explanation for this is that the computing time of the FFT-based evaluation–interpolation method is quasi-linear w.r.t. the bound it uses, while the default bound (resp. Bézout bound) is linear w.r.t. the degree of $\mathrm{res}(f_i, T)$ (resp. $\mathrm{res}(f_i, \widetilde{T})$). Finally, we observe that as i increases, the timing for computing $\mathrm{res}(f_i, T)$ grows much faster than computing $\mathrm{res}(f_i, \widetilde{T})$. For the largest example, that is $i = 10$, the ratio between computing $\mathrm{res}(f_i, T)$ and $\mathrm{res}(f_i, \widetilde{T})$ (with Bézout bound) is about 1000.

In the second group, we generate one trivariate polynomial f and 9 one-dimensional trivariate regular chains T_i, for $i = 2, \ldots, 10$, as follows:

(1) we randomly generate a trivariate dense polynomial f of $\mathbb{Z}/p\mathbb{Z}[x_1, x_2, x_3]$, with total degree 4;
(2) for $i = 2, \ldots, 10$, we randomly generate a pair of dense polynomials, $g_{i,1}, g_{i,2}$ in $\mathbb{Z}/p\mathbb{Z}[x_1, x_2, x_3]$, with total degree i;

Table 2
Same polynomial but different regular chains.

i	Computing time (seconds)			Degree of output		
	$\mathrm{res}(f, T_i)$	$\mathrm{res}(f, \widetilde{T}_i)$ (default bound)	$\mathrm{res}(f, \widetilde{T}_i)$ (Bézout bound)	$\mathrm{res}(f, T_i)$	$\mathrm{res}(f, \widetilde{T}_i)$ (default bound)	$\mathrm{res}(f, \widetilde{T}_i)$ (Bézout bound)
2	0.020	0.060	0.016	32	16	16
3	0.388	0.460	0.088	180	36	36
4	4.992	2.292	0.212	640	64	64
5	41.623	10.581	0.640	1700	100	100
6	274.997	45.935	1.780	3744	144	144
7	1404.183	119.436	3.232	7252	196	196
8	5734.366	289.658	5.788	12 800	256	256
9	-	833.260	12.797	-	324	324
10	-	1738.141	21.101	-	400	400

(3) for $i = 2, \ldots, 10$, we call $\mathsf{Triangularize}(g_{i,1}, g_{i,2})$ to compute a Kalkbrener triangular decomposition of the system $\{g_{i,1}, g_{i,2}\}$ and call T_i the unique one-dimensional regular chain in the output.

For each $i = 2, \ldots, 10$, we run the following three different computations on T_i and f:

(a) compute $\mathrm{res}(f, T_i)$ by successively calling MAPLE's Resultant function;
(b) compute $\mathrm{res}(f, \widetilde{T}_i)$ by calling the function IteratedResultantDim1 with the default bound calculated from f and T_i;
(c) compute $\mathrm{res}(f, \widetilde{T}_i)$ by calling the function IteratedResultantDim1 with Bézout bound calculated from f and $g_{i,1}, g_{i,2}$.

The experimental results of the above computations are reported in Table 2. In the table, "-" means that the computation does not finish within 2 h. As we can see, as i increases, the degree of $\mathrm{res}(f, T_i)$ becomes much larger than that of $\mathrm{res}(f, \widetilde{T}_i)$. Meanwhile, the time for computing $\mathrm{res}(f, T_i)$ also grows much faster than for computing $\mathrm{res}(f, \widetilde{T}_i)$. For $i = 8$, the ratios for degree and computing time are respectively 50 and 1000.

7. Conclusion

In this paper, we have presented recent progress in computing triangular decomposition incrementally. For input polynomial systems forming regular sequences, this approach appear to very successful, as illustrated by the experimental results reported in Chen and Moreno Maza (2011) and in Section 6.5. The theoretical results proposed through Section 4 to Section 6 aim at explaining these empirical observations.

Nevertheless, triangular decomposition methods remain an active research area. For instance, over-constrained systems put incremental methods at challenge. We believe that revisiting Wu's characteristic set method for those systems, integrating techniques such as multipolynomial resultants as in Kapur (1996) and modular methods as in Li et al. (2009), is a promising direction for future research.

Appendix A. The algorithms

In this appendix, we present an algorithm to compute Lazard–Wu triangular decompositions in an incremental manner, see Chen and Moreno Maza (2011) for a proof. We recall the concepts of a *process* and a *regular (delayed) split*, which were introduced as Definition 9 and 11 in Moreno Maza (1999). To serve our purpose, we modify the original definitions as follows.

Definition 4. A *process* of $\mathbf{k}[\mathbf{x}]$ is a pair (p, T), where $p \in \mathbf{k}[\mathbf{x}]$ is a polynomial and $T \subset \mathbf{k}[\mathbf{x}]$ is a regular chain. The process $(0, T)$ is also written as T for short. Given two processes (p, T) and (p', T'), let v and v' be respectively the greatest variable appearing in (p, T) and (p', T'). We say $(p, T) \prec (p', T')$ if: (i) either $v < v'$; (ii) or $v = v'$ and $\dim T < \dim T'$; (iii) or $v = v'$, $\dim T = \dim T'$ and $T \prec T'$; (iv) or $v = v'$, $\dim T = \dim T'$, $T \sim T'$ and $p \prec p'$. We write $(p, T) \sim (p', T')$ if neither $(p, T) \prec (p', T')$ nor $(p', T') \prec (p, T)$ hold. Clearly any sequence of processes which is strictly decreasing w.r.t. $\prec$ is finite.

Definition 5. Let T_i, $1 \le i \le e$, be regular chains of $\mathbf{k}[\mathbf{x}]$. Let $p \in \mathbf{k}[\mathbf{x}]$. We call $T_1, \ldots, T_e$ a *regular split* of (p, T) and we write $(p, T) \longrightarrow T_1, \ldots, T_e$, whenever we have

(L_1) $\sqrt{\mathrm{sat}(T)} \subseteq \sqrt{\mathrm{sat}(T_i)}$,
(L_2) $W(T_i) \subseteq V(p)$ (or equivalently $p \in \sqrt{\mathrm{sat}(T_i)}$),
(L_3) $V(p) \cap W(T) \subseteq \cup_{i=1}^{e} W(T_i)$.

Observe that the above three conditions are equivalent to the following relation:

$$V(p) \cap W(T) \subseteq W(T_1) \cup \cdots \cup W(T_e) \subseteq V(p) \cap \overline{W(T)}.$$

Geometrically, this means that $W(T_1) \cup \cdots \cup W(T_e)$ is a "sharp" approximation of the intersection of $V(p)$ and $W(T)$. When $p = 0$, we simply write T instead of (p, T). Therefore the notation $T \longrightarrow T_1, \ldots, T_e$ stands for

$$W(T) \subseteq W(T_1) \cup \cdots \cup W(T_e) \subseteq \overline{W(T)}.$$

Next we list the specifications of our triangular decomposition algorithm and its subroutines. We denote by R the polynomial ring $\mathbf{k}[\mathbf{x}]$, where $\mathbf{x} = x_1 < \cdots < x_n$.

Triangularize(F)

- Input: F, a finite set of polynomials of R.
- Output: A Lazard–Wu triangular decomposition of $V(F)$.

Intersect(p, T)

- Input: p, a polynomial of R; T, a regular chain of R.
- Output: a set of regular chains $\{T_1, \ldots, T_e\}$ such that $(p, T) \longrightarrow T_1, \ldots, T_e$.

Regularize(p, T)

- Input: p, a polynomial of R; T, a regular chain of R.
- Output: a set of pairs $\{[p_1, T_1], \ldots, [p_e, T_e]\}$ such that for each i, $1 \le i \le e$: (1) T_i is a regular chain; (2) $p \equiv p_i \bmod \sqrt{\mathrm{sat}(T_i)}$; (3) if $p_i = 0$, then $p_i \in \sqrt{\mathrm{sat}(T_i)}$ otherwise p_i is regular modulo $\sqrt{\mathrm{sat}(T_i)}$; moreover we have $T \longrightarrow T_1, \ldots, T_e$.

SubresultantChain(p, q, v)

- Input: v, a variable of $\{x_1, \ldots, x_n\}$; p and q, polynomials of R, whose main variables are both v.
- Output: a list of polynomials $(S_0, \ldots, S_\lambda)$, where $\lambda = \min(\mathrm{mdeg}(p), \mathrm{mdeg}(q))$, such that S_i is the i-th subresultant of p and q w.r.t. v.

RegularGcd(p, q, v, S, T)

- Input: v, a variable of $\{x_1, \ldots, x_n\}$,
  - T, a regular chain of R such that $\mathrm{mvar}(T) < v$,
  - p and q, polynomials of R with the same main variable v such that: $\mathrm{init}(q)$ is regular modulo $\sqrt{\mathrm{sat}(T)}$; $\mathrm{res}(p, q, v)$ belongs to $\sqrt{\mathrm{sat}(T)}$,
  - S, the subresultant chain of p and q w.r.t. v.
- Output: a set of pairs $\{[g_1, T_1], \ldots, [g_e, T_e]\}$ such that $T \longrightarrow T_1, \ldots, T_e$ and for each T_i: if $\dim T = \dim T_i$, then g_i is a regular GCD of p and q modulo $\sqrt{\mathrm{sat}(T_i)}$; otherwise $g_i = 0$, which means undefined.

Algorithm 3: Intersect(p, T)

```
begin
    if prem(p, T) = 0 then return {T};
    if p ∈ k then return { };
    r := p; P := {r}; S := { };
    while mvar(r) ∈ mvar(T) do
        v := mvar(r); src := SubresultantChain(r, T_v, v);
        S := S ∪ {src}; r := resultant(src);
        if r = 0 then break;
        if r ∈ k then return { };
        P := P ∪ {r}
    𝔗 := {∅}; 𝔗′ := { }; i := 1;
    while i ≤ n do
        for C ∈ 𝔗 do
            if x_i ∉ mvar(P) and x_i ∉ mvar(T) then
                𝔗′ := 𝔗′ ∪ CleanChain(C, T, x_{i+1})
            else if x_i ∉ mvar(P) then
                𝔗′ := 𝔗′ ∪ CleanChain(C ∪ T_{x_i}, T, x_{i+1})
            else if x_i ∉ mvar(T) then
                for D ∈ IntersectFree(P_{x_i}, x_i, C) do
                    𝔗′ := 𝔗′ ∪ CleanChain(D, T, x_{i+1})
            else
                for D ∈ IntersectAlgebraic(P_{x_i}, T, x_i, S_{x_i}, C) do
                    𝔗′ := 𝔗′ ∪ CleanChain(D, T, x_{i+1})
        𝔗 := 𝔗′; 𝔗′ := { }; i := i + 1
    return 𝔗
end
```

Algorithm 4: RegularGcd(p, q, v, S, T)

```
begin
    𝔗 := {(T, 1)};
    while 𝔗 ≠ ∅ do
        let (C, i) ∈ 𝔗; 𝔗 := 𝔗 \ {(C, i)};
        for [f, D] ∈ Regularize(s_i, C) do
            if dim D < dim C then
                output [0, D]
            else if f = 0 then
                𝔗 := 𝔗 ∪ {(D, i + 1)}
            else
                output [S_i, D]
end
```

IntersectFree(p, x_i, C)

- Input: x_i, a variable of $\mathbf{x}$; p, a polynomial of R with main variable x_i; C, a regular chain of $\mathbf{k}[x_1, \ldots, x_{i-1}]$.
- Output: a set of regular chains $\{T_1, \ldots, T_e\}$ such that $(p, C) \longrightarrow T_1, \ldots, T_e$.

IntersectAlgebraic(p, T, x_i, S, C)

- Input: p, a polynomial of R with main variable x_i,

Algorithm 5: IntersectFree(p, x_i, C)

```
begin
    for [f, D] ∈ Regularize(init(p), C) do
        if f = 0 then
            output Intersect(tail(p), D)
        else
            output D ∪ p;
            for E ∈ Intersect(init(p), D) do
                output Intersect(tail(p), E)
end
```

Algorithm 6: IntersectAlgebraic(p, T, x_i, S, C)

```
begin
    for [g, D] ∈ RegularGcd(p, T_{x_i}, x_i, S, C) do
        if dim D < dim C then
            for E ∈ CleanChain(D, T, x_i) do
                output IntersectAlgebraic(p, T, x_i, S, E)
        else
            output D ∪ g;
            for E ∈ Intersect(init(g), D) do
                for F ∈ CleanChain(E, T, x_i) do
                    output IntersectAlgebraic(p, T, x_i, S, F)
end
```

. T, a regular chain of R, where $x_i \in \mathrm{mvar}(T)$,
. S, the subresultant chain of p and T_{x_i} w.r.t. x_i,
. C, a regular chain of $\mathbf{k}[x_1, \ldots, x_{i-1}]$, such that: $\mathrm{init}(T_{x_i})$ is regular modulo $\sqrt{\mathrm{sat}(C)}$; the resultant of p and T_{x_i}, which is S_0, belongs to $\sqrt{\mathrm{sat}(C)}$.

– Output: a set of regular chains $T_1, \ldots, T_e$ such that $(p, C \cup T_{x_i}) \longrightarrow T_1, \ldots, T_e$.

CleanChain(C, T, x_i)

– Input: T, a regular chain of R; C, a regular chain of $\mathbf{k}[x_1, \ldots, x_{i-1}]$ such that $\sqrt{\mathrm{sat}(T_{<x_i})} \subseteq \sqrt{\mathrm{sat}(C)}$.
– Output: if $x_i \notin \mathrm{mvar}(T)$, return C; otherwise return a set of regular chains $\{T_1, \ldots, T_e\}$ such that $\mathrm{init}(T_{x_i})$ is regular modulo each $\mathrm{sat}(T_j)$, $\sqrt{\mathrm{sat}(C)} \subseteq \sqrt{\mathrm{sat}(T_j)}$ and $W(C) \setminus V(\mathrm{init}(T_{x_i})) \subseteq \cup_{j=1}^{e} W(T_j)$.

Extend(C, T, x_i)

– Input: C, is a regular chain of $\mathbf{k}[x_1, \ldots, x_{i-1}]$; T, a regular chain of R such that $\sqrt{\mathrm{sat}(T_{<x_i})} \subseteq \sqrt{\mathrm{sat}(C)}$.
– Output: a set of regular chains $\{T_1, \ldots, T_e\}$ of R such that $W(C \cup T_{\geq x_i}) \subseteq \cup_{j=1}^{e} W(T_j)$ and $\sqrt{\mathrm{sat}(T)} \subseteq \sqrt{\mathrm{sat}(T_j)}$.

Algorithm SubresultantChain is standard, see Ducos (2000). The algorithm Triangularize is a *principle algorithm* which was first presented in Moreno Maza (1999). We use the following conventions in our pseudo-code: the keyword **return** yields a result and terminates the current function call while the keyword **output** yields a result and keeps executing the current function call.

Algorithm 7: Regularize(p, T)

```
begin
    if p ∈ k or T = ∅ then return [p, T];
    v := mvar(p);
    if v ∉ mvar(T) then
        for [f, C] ∈ Regularize(init(p), T) do
            if f = 0 then
                output Regularize(tail(p), C);
            else
                output [p, C];
    else
        src := SubresultantChain(p, T_v, v); r := resultant(src);
        for [f, C] ∈ Regularize(r, T_<v) do
            if dim C < dim T_<v then
                for D ∈ Extend(C, T, v) do
                    output Regularize(p, D)
            else if f ≠ 0 then
                output [p, C ∪ T_≥v]
            else
                for [g, D] ∈ RegularGcd(p, T_v, v, src, C) do
                    if dim D < dim C then
                        for E ∈ Extend(D, T, v) do
                            output Regularize(p, E);
                    else
                        if mdeg(g) = mdeg(T_v) then output [0, D ∪ T_≥v]; next;
                        output [0, D ∪ g ∪ T_>v];
                        q := pquo(T_v, g);
                        output Regularize(p, D ∪ q ∪ T_>v);
                        for E ∈ Intersect(h_g, D) do
                            for F ∈ Extend(E, T, v) do
                                output Regularize(p, F)
end
```

Algorithm 8: Extend(C, T, x_i)

```
begin
    if T_≥x_i = ∅ then return C;
    let p ∈ T with greatest main variable; T' := T \ {p};
    for D ∈ Extend(C, T', x_i) do
        for [f, E] ∈ Regularize(init(p), D) do
            if f ≠ 0 then output E ∪ p;
end
```

Appendix B. Experimentation

Part of the algorithms presented in this paper are implemented in Maple 15 while all of them are present in the current development version of Maple. Tables B.1 and B.3 report on our comparison between Triangularize and other Maple solvers. The notations used in these tables are defined below.

Algorithm 9: CleanChain(C, T, x_i)

```
begin
    if x_i ∉ mvar(T) or dim C = dim T_<x_i then return C;
    for [f, D] ∈ Regularize(init(T_x_i), C) do
        if f ≠ 0 then output D
end
```

Algorithm 10: Triangularize(F)

```
begin
    if F = { } then return {∅};
    Choose a polynomial p ∈ F with maximal rank;
    for T ∈ Triangularize(F \ {p}) do
        output Intersect(p, T)
end
```

Table B.1
The input and output sizes of systems.

sys	Input size				Output size				
	#v	#e	deg	dim	GL	GS	GD	TL16	TK16
4corps-1parameter-homog	4	3	8	1	–	–	21 863	–	30 738
8-3-config-Li	12	7	2	7	67 965	–	72 698	7 538	1 384
Alonso-Li	7	4	4	3	1 270	–	614	2 050	374
Bezier	5	3	6	2	–	–	32 054	–	114 109
Cheaters-homotopy-1	7	3	7	4	26 387 452	–	17 297	–	285
childDraw-2	10	10	2	0	938 846	–	157 765	–	–
Cinquin-Demongeot-3-3	4	3	4	1	1 652 062	–	680	2 065	895
Cinquin-Demongeot-3-4	4	3	5	1	–	–	690	–	2 322
collins-jsc02	5	4	3	1	–	–	28 720	2 770	1 290
f-744	12	12	3	1	102 082	–	83 559	4 509	4 510
Haas5	4	2	10	2	–	–	28	–	548
Lichtblau	3	2	11	1	6 600 095	–	224 647	110 332	5 243
Liu-Lorenz	5	4	2	1	47 688	123 965	712	2 339	938
Mehta2	11	8	3	3	–	–	1 374 931	5 347	5 097
Mehta3	13	10	3	3	–	–	–	25 951	25 537
Mehta4	15	12	3	3	–	–	–	71 675	71 239
p3p-isosceles	7	3	3	4	56 701	–	1 453	9 253	840
p3p	8	3	3	5	160 567	–	1 768	–	1 712
Pavelle	8	4	2	4	17 990	–	1 552	3 351	1 086
Solotareff-4b	5	4	3	1	2 903 124	–	14 810	2 438	872
Wang93	5	4	3	1	2 772	56 383	1 377	1 016	391
Xia	6	3	4	3	63 083	2 711	672	1 647	441
xy-5-7-2	6	3	3	3	12 750	–	599	–	3 267

Notation for Triangularize**.** We denote by TK16 and TL16 the latest implementation of Triangularize for computing, respectively, Kalkbrener and Lazard–Wu decompositions, in the current version of MAPLE. Denote by TK13, TL13 the implementation based on the algorithm of Moreno Maza (1999) in MAPLE 13. Finally, STK16 and STL16 are versions of TK16 and TL16 respectively, enforcing that all computed regular chains are squarefree.

Notation for the other solvers. Denote by GL, GS, GD, respectively the function Gröebner:-Basis (plex order), Gröebner:-Solve, Gröebner:-Basis (tdeg order) in the current beta version of MAPLE. Denote by WS the function wsolve of the package `Wsolve` Wang, `Wsolve` (0000a), which decomposes a variety as a union of quasi-components of Wu characteristic sets.

Table B.2
Timings of Triangularize of different versions.

sys	TK13	TK16	TL13	TL16	STK16	STL16
4corps-1parameter-homog	–	36.9	–	–	62.8	–
8-3-config-Li	8.7	5.9	29.7	25.8	6.0	26.6
Alonso-Li	0.3	0.4	14.0	2.1	0.4	2.2
Bezier	–	88.2	–	–	–	–
Cheaters-homotopy-1	0.4	0.7	–	–	451.8	–
childDraw-2	–	–	–	–	1326.8	1437.1
Cinquin-Demongeot-3-3	3.2	0.6	–	7.1	0.7	8.8
Cinquin-Demongeot-3-4	166.1	3.1	–	–	3.3	–
collins-jsc02	5.8	0.4	–	1.5	0.4	1.5
f-744	–	12.7	–	14.8	12.9	15.1
Haas5	452.3	0.3	–	–	0.3	–
Lichtblau	0.7	0.3	801.7	143.5	0.3	531.3
Liu-Lorenz	0.4	0.4	4.7	2.3	0.4	4.4
Mehta2	–	2.2	–	4.5	2.2	6.2
Mehta3	–	14.4	–	51.1	14.5	63.1
Mehta4	–	859.4	–	1756.3	859.2	1761.8
p3p-isosceles	1.2	0.3	–	352.5	0.3	–
p3p	168.8	0.3	–	–	0.3	–
Pavelle	0.8	0.5	–	7.0	0.4	12.6
Solotareff-4b	1.5	0.8	–	1.9	0.9	2.0
Wang93	0.5	0.7	0.6	0.8	0.8	0.9
Xia	0.2	0.4	4.0	1.9	0.5	2.7
xy-5-7-2	3.3	0.6	–	–	0.7	–

Table B.3
Timings of Triangularize versus other solvers.

sys	GL	TK16	GS	WS	TL16
4corps-1parameter-homog	–	36.9	–	–	–
8-3-config-Li	108.7	5.9	–	27.8	25.8
Alonso-Li	3.4	0.4	–	7.9	2.1
Bezier	–	88.2	–	–	–
Cheaters-homotopy-1	2609.5	0.7	–	–	–
childDraw-2	19.3	–	–	–	–
Cinquin-Demongeot-3-3	63.6	0.6	–	–	7.1
Cinquin-Demongeot-3-4	–	3.1	–	–	–
collins-jsc02	–	0.4	–	0.8	1.5
f-744	30.8	12.7	–	–	14.8
Haas5	–	0.3	–	–	–
Lichtblau	125.9	0.3	–	–	143.5
Liu-Lorenz	3.2	0.4	2160.1	40.2	2.3
Mehta2	–	2.2	–	5.7	4.5
Mehta3	–	14.4	–	–	51.1
Mehta4	–	859.4	–	–	1756.3
p3p-isosceles	6.2	0.3	–	792.8	352.5
p3p	33.6	0.3	–	–	–
Pavelle	1.8	0.5	–	–	7.0
Solotareff-4b	35.2	0.8	–	9.1	1.9
Wang93	0.2	0.7	1580.0	0.8	0.8
Xia	4.7	0.4	0.1	12.5	1.9
xy-5-7-2	0.3	0.6	–	–	

The tests were launched on a machine with Intel Core 2 Quad CPU (2.40GHz) and 3.0Gb total memory. The time-out is set as 3600 s. The memory usage is limited to 60% of total memory, using the UNIX command `ulimit`. In both Tables B.1 and B.3, the symbol "-" means either time or memory exceeds the limit we set.

The examples are mainly in positive dimension since other triangular decomposition algorithms are specialized to dimension zero (Dahan et al., 2005). All examples are in characteristic zero.

In Table B.1, we provide characteristics of the input systems and the sizes of the output obtained by different solvers. For each polynomial system $F \subset \mathbb{Q}[\mathbf{x}]$, the number of variables appearing in F, the number of polynomials in F, the maximum total degree of a polynomial in F, the dimension of the algebraic variety $V(F)$ are denoted respectively by $\#v$, $\#e$, deg, dim. For each solver, the size of its output is measured by the total number of characters in the output. To be precise, let "dec" and "gb" be respectively the output of the Triangularize and Gröebner functions. The MAPLE command we use are length(convert(map(Equations, dec, R), string)) and length(convert(gb, string)). From Table B.1, it is clear that Triangularize produces much smaller output than commands based on Gröbner basis computations.

TK16, TL16, GS, WS (and, to some extent, GL) can all be seen as polynomial system solvers in the sense of that they provide equidimensional decompositions where components are represented by triangular sets. Moreover, they are implemented in MAPLE (with the support of efficient C code in the case of GS and GL). The specification of TK16 are close to those of GS while TL16 is related to WS, though the triangular sets returned by WS are not necessarily regular chains.

In Table B.2, we provide the timings of different versions of Triangularize. From this table, it is clear that the implementations of Triangularize, based on the algorithms presented in this paper (that is TK16, TL16) outperform the previous versions (TK13, TL13), based on Moreno Maza (1999), by several orders of magnitude. In Table B.3, we provide the timings of Triangularize and other solvers. We observe that TK16 outperforms GS and GL while TL16 outperforms WS.

References

Aubry, P., Lazard, D., Moreno Maza, M., 1999. On the theories of triangular sets. J. Symbolic Comput. 28 (1–2), 105–124.

Boulier, F., Lazard, D., Ollivier, F., Petitot, M., 1995. Representation for the radical of a finitely generated differential ideal. In: Proc. of ISSAC'95. Montréal, Canada, pp. 158–166.

Boulier, F., Lemaire, F., Moreno Maza, M., 2006. Well known theorems on triangular systems and the D5 principle. In: Proc. of Transgressive Computing 2006, Universidad de Granda.

Chen, C., Davenport, J.H., May, J., Moreno Maza, M., Xia, B., Xiao, R., 2010. Triangular decomposition of semi-algebraic systems. In: Proc. of ISSAC'10. ACM Press, pp. 187–194.

Chen, C., Davenport, J.H., Lemaire, F., Moreno Maza, M., Phisanbut, N., Xia, B., Xiao, R., Xie, Y., 2011. Solving semi-algebraic systems with the RegularChains library in Maple. Ib Proc. of MACIS 2011, S. Ratschau Ed., pp. 38–51.

Chen, C., Golubitsky, O., Lemaire, F., Moreno Maza, M., Pan, W., 2007. Comprehensive triangular decomposition. In: Proc. of CASC'07. In: Lecture Notes in Computer Science, vol. 4770. pp. 73–101.

Chen, C., Moreno Maza, M., 2011. Algorithms for computing triangular decompositions of polynomial systems. In: Proc. of ISSAC'11, pp. 83–90.

Chen, C., Moreno Maza, M., Xia, B., Yang, L., 2009. Computing cylindrical algebraic decomposition via triangular decomposition. In: Proc. of ISSAC'09, pp. 95–102.

Chou, S.C., Gao, X.S., 1991. Computations with parametric equations. In: Proc. of ISSAC'91, Bonn, Germany, pp. 122–127.

Chou, S.C., Gao, X.S., 1992. Solving parametric algebraic systems. In: Proc. of ISSAC'92, pp. 335–341.

Cox, D., Little, J., O'Shea, D., 1998. Using Algebraic Geometry. In: Graduate Text in Mathematics, vol. 185. Springer-Verlag, New-York.

Dahan, X., Kadri, A., Schost, É, 2009. Bit-size estimates for triangular sets in positive dimension. Technical report, The University of Western Ontario, Journal of Complexity (in press).

Dahan, X., Moreno Maza, M., Schost, É, Wu, W., Xie, Y., 2005. Lifting techniques for triangular decompositions. In: Proc. of ISSAC'05, pp. 108–115.

Della Dora, J., Dicrescenzo, C., Duval, D., 1985. About a new method for computing in algebraic number fields. In: Proc. EUROCAL 85, Vol. 2. Springer-Verlag, pp. 289–290.

Ducos, L., 2000. Optimizations of the subresultant algorithm. J. Pure Appl. Algebra 145, 149–163.

Gao, X.S., Chou, S.C., 1993. A Zero Structure Theorem for Differential Parametric Systems. J. Symbolic Comput. 16 (6), 585–595.

Gao, X.S., van der Hoeven, J., Luo, Y., Yuan, C., 2009. Characteristic set method for differential-difference polynomial systems. J. Symbolic Comput. 44, 1137–1163.

Golubitsky, O., 2005. Private communication.

Hubert, É., 2000. Factorization free decomposition algorithms in differential algebra. J. Symbolic Comput. 29 (4–5), 641–662.

Kalkbrener, M., 1991. Three contributions to elimination theory. Ph.D. Thesis, Johannes Kepler University, Linz.

Kalkbrener, M., 1993. A generalized euclidean algorithm for computing triangular representations of algebraic varieties. J. Symbolic Comput. 15, 143–167.

Kalkbrener, M., 1998. Algorithmic properties of polynomial rings. J. Symbolic Comput. 26 (5), 525–581.

Kapur, D., 1996. Automated Geometric Reasoning: Dixon Resultants, Gröbner Bases, and Characteristic Sets. In: Wang, D.M. (Ed.), Automated Deduction in Geometry. In: Lecture notes in artificial intelligence, I360. Springer, pp. 1–36.

Lasker, E., 1905. Zur Theorie der Moduln und Ideale. Math. Ann. 60, 19–116.

Lazard, D., 1991. A new method for solving algebraic systems of positive dimension. Discrete Appl. Math. 33, 147–168.

Lemaire, F., Moreno Maza, M., Xie, Y., 2005. The `RegularChains` library. In: Proc. of Maple Conference 2005, pp. 355–368.
Li, X., Moreno Maza, M., Pan, W., 2009. Computations modulo regular chains. In: Proc. of ISSAC'09, pp. 239–246.
Li, X., Moreno Maza, M., Schost, É, 2007. Fast arithmetic for triangular sets: From theory to practice. In: ISSAC'07. ACM Press, pp. 269–276.
Moreno Maza, M., Rioboo, R., 1995. Polynomial gcd computations over towers of algebraic extensions. In: Proc. of AAECC-11. Springer, pp. 365–382.
Lecerf, G., 2003. Computing the equidimensional decomposition of an algebraic closed set by means of lifting fibers. J. Complexity 19 (4), 564–596.
Maplesoft Incorporation. Maple 15: The essential tool for mathematics and modeling. http://www.maplesoft.com/products/maple/.
Mishra, B., 1993. Algorithmic Algebra. Springer-Verlag.
Moreno Maza, M., On triangular decompositions of algebraic varieties. Technical Report TR 4/99, NAG Ltd, Oxford, UK, 1999. Presented at the MEGA-2000 Conference, Bath, England. http://www.csd.uwo.ca/~moreno/books-papers.html.
Nöther, E., 1921. Idealtheorie in Ringbereichen. Mathematische Annalen 83 (1), 24.
Ritt, J.F., 1932. Differential Equations from an Algebraic Standpoint, Vol. 14. American Mathematical Society, New York.
Ritt, J.F., 1950. Differential Algebra. Amer. Math. Soc, New York.
Shimoyama, T., Yokoyama, K., 1996. Localization and primary decomposition of polynomial ideals. J. Symbolic Comput. 22 (3), 247–277.
Sommese, A.J., Verschelde, J., Wampler, C.W., 2008. Solving polynomial systems equation by equation. In: Algorithms in Algebraic Geometry. Springer-Verlag, pp. 133–152.
Steel, A., 2005. Conquering inseparability: Primary decomposition and multivariate factorization over algebraic function fields of positive characteristic. J. Symbolic Comput. 40 (3), 1053–1075.
van der Waerden, B., 1991. Algebra, seventh edition. Springer-Verlag.
Wang, D.K., The `Wsolve` package. http://www.mmrc.iss.ac.cn/~dwang/wsolve.txt.
Wang, D.M., 1991. On Wu's method for solving systems of algebraic equations. Technical report RISC-LINZ Series no 91-52.0, Johannes Kepler University, Austria.
Wang, D.M., 1993. An elimination method for polynomial systems. J. Symbolic Comput. 16, 83–114.
Wang, D.M., Epsilon 0.618. http://www-calfor.lip6.fr/wang/epsilon.
Wang, D.M., 1998. Decomposing polynomial systems into simple systems. J. Symbolic Comput. 25 (3), 295–314.
Wang, D.M., 2000. Elimination Methods. Springer, New York.
Wu, W.T., 1984a. Basic principles of mechanical theorem proving in elementary geometries. J. Systems Sci. Math. Sci. 4, 207–235.
Wu, W.T., 1984b. Some recent advances in mechanical theorem-proving of geometries. Contemporary Math. 29, 235–241.
Wu, W.T., 1986. On zeros of algebraic equations — an application of Ritt Principle. Kexue Tongbao 1 (31), 1–5.
Wu, W.T., 1987. A zero structure theorem for polynomial equations solving. MM Research Preprints 1, 2–12.
Wu, W.T., 1989a. On the foundation of algebraic differential geometry. MM Research Preprints 3, 1–26.
Wu, W.T., 1989b. Some remarks on characteristic-set formation. MM Research Preprints 3, 27–29.
Wu, W.T., 1989c. On the generic zero and Chow basis of an irreducible ascending set. MM Research Preprints 4, 1–21.
Wu, W.T., 1989d. On a projection theorem of quasi-varieties in elimination theory. MM Research Preprints 4, 40–48.
Wu, W.T., 1989e. A mechanization method of geometry and its applications: solving inverse kinematic equation of PUMA-type robots. MM Research Preprints 4, 49–53.
Wu, W.T., 1992. On problems involving inequalities. MM Research Preprints 7, 1–13.
Wu, W.T., 2006. Memory of my first research teacher: The great geometer Chern Shiing-Shen. In: Griffiths, Phillip A. (Ed.), Inspired by S.S. Chern: A memorial volume in honor of a great mathematician. In: Nankai Tracts in Mathematics, Vol. 11.
Yang, L., Hou, X., Xia, B., 2001. A complete algorithm for automated discovering of a class of inequality-type theorems. Science in China, Series **F** 44 (1), 33–49.
Yang, L., Zhang, J., 1991. Searching dependency between algebraic equations: an algorithm applied to automated reasoning. Technical Report IC/89/263, International Atomic Energy Agency, Miramare, Trieste, Italy.