Solving Polynomial Systems via Triangular Decomposition

(Spine title: Solving Polynomial Systems via Triangular Decomposition)

(Thesis format: Monograph)

by

Changbo Chen

Graduate Program
in
Computer Science

A thesis submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

School of Graduate and Postdoctoral Studies
The University of Western Ontario
London, Ontario, Canada
August, 2011

THE UNIVERSITY OF WESTERN ONTARIO
THE SCHOOL OF GRADUATE AND POSTDOCTORAL STUDIES

CERTIFICATE OF EXAMINATION

Supervisor:	Examination committee:
Dr. Marc Moreno Maza	Dr. John Barron
	Dr. Rob Corless
	Dr. Hoon Hong
	Dr. Pei Yu

The thesis by

Changbo Chen

entitled:

Solving Polynomial Systems via Triangular Decomposition

is accepted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Date ______________________

____
Chair of the Thesis Examination Board

Abstract

Finding the solutions of a polynomial system is a fundamental problem with numerous applications in both the academic and industrial world. In this thesis, we target on computing symbolically both the real and the complex solutions of nonlinear polynomial systems with or without parameters. To this end, we improve existing algorithms for computing triangular decompositions. Based on that, we develop various new tools for solving polynomial systems and illustrate their effectiveness by applications.

We propose new algorithms for computing triangular decompositions of polynomial systems incrementally. With respect to previous works, our improvements are based on a *weakened* notion of a polynomial GCD modulo a regular chain, which permits to greatly simplify and optimize the sub-algorithms. Extracting common work from similar expensive computations is also a key feature of our algorithms.

We adapt the concepts of regular chain and triangular decomposition, originally designed for studying the complex solutions of polynomial systems, to describing the solutions of semi-algebraic systems. We show that any such system can be decomposed into finitely many *regular semi-algebraic systems*. We propose two specifications (full and lazy) of such a decomposition and present corresponding algorithms. Under some assumptions, the lazy decomposition can be computed in singly exponential time w.r.t. the number of variables.

We introduce the concept of *comprehensive triangular decomposition* for solving parametric polynomial systems. It partitions the parametric space into disjoint cells such that the complex or real solutions of a polynomial system depend continuously on the parameters in each cell. In the real case, we rely on cylindrical algebraic decomposition (CAD) to decompose a cell into connected components. CAD itself is one of the most important tools for computing with semi-algebraic sets. We present a brand new algorithm for computing it based on triangular decomposition.

Keywords: Regular chain, triangular decomposition, polynomial system solving,

constructible set, cylindrical algebraic decomposition, semi-algebraic system, parametric polynomial system, comprehensive triangular decomposition, Regular GCD.

Acknowledgements

I would like to express my gratitude heartily to all those who gave me the possibility to complete this thesis.

First of all, I especially want to thank my adorable supervisor Professor Marc Moreno Maza, for his guidance during my research and study at The University of Western Ontario. His continuous inspiration, stimulating suggestions, encouragement, and all kinds of supports paved a smooth research road for me. I am the lucky beneficiary of his wide knowledge, kindness, graciousness and hard working.

I feel honoured to collaborate with my brilliant, insightful co-authors: François Boulier, James Davenport, Oleg Golubitsky, François Lemaire, Liyun Li, John May, Wei Pan, Bican Xia, Rong Xiao, Yuzhen Xie and Lu Yang. I would like to express my sincere appreciation to my colleagues at Maplesoft, in particular Jürgen Gerhard, John May and Clare So. I would also like to thank all the members from ORCCA lab and the Computer Science Department for their great help in the past five years.

Many thanks to the members of my committee Professor John Barron, Professor Rob Corless, Professor Hoon Hong and Professor Pei Yu for their inspiration, comments and questions.

My special gratitude goes to my family for their love and support. I dedicate this thesis to my wife Yan.

Contents

List of Algorithms

List of Figures

List of Tables

Chapter 1

Introduction

Solving a polynomial system, or computing its solutions, has been a fundamental topic in mathematics since ancient times. The meaning of "solving" does not have a single or simple definition. For example, considering the space of solutions, one may seek for integer solutions, rational number solutions, real solutions, complex solutions or even solutions in an arbitrary ring or field. Considering the form of the output, one may require numerical values or symbolic expressions. For nonlinear polynomial systems with rational number coefficients, this thesis aims to provide real or complex solutions which are encoded in the form of triangular systems akin to linear system solving.

This thesis is motivated by applications from biochemistry. In the field of biochemistry, many reaction networks are modelled by dynamical systems. The equilibria (or steady states) of a dynamical system are typically described by nonlinear parametric polynomial systems (a system of polynomial equations, inequations or inequalities with parameters), where a basic question is the stability of these equilibria when parameters vary. Traditionally, this question is answered by numerical simulation. In this thesis, we develop new symbolic tools and demonstrate how these tools can help answering the above question.

In our study, analyzing the stability of the equilibria of dynamical systems is treated as a particular case of solving nonlinear (parametric) polynomial systems. This is a central topic in the field of computer algebra. For polynomial system over a general coefficient field, the two basic tools are Gröbner basis and triangular decompositions. In the last decades, more attention was paid to the former tool due to its simple algebraic structure. However, both theory [47] and experimentation [33] indicate that the later one tends to produce smaller output. In addition, while the implementation techniques of the former one are already quite advanced, the latter one still has a large potential for improvement. All these factors motivate us to improve the efficiency of triangular decompositions and develop new theory and algorithms for supporting them.

In the last five years, we have developed step by the step the tools we needed. The theoretical and algorithmic results have been published or accepted in conference proceedings or journal articles [30, 35, 32, 12, 28, 36, 26, 33, 29, 34]. The implementation of these tools has been integrated into the computer algebra system MAPLE and are available in the `RegularChains` library of MAPLE releases 12, 13, 14 or 15.

In the rest of this introduction, we first introduce these new tools by an example from biochemistry in an informal manner. We then summarize the main results we have already obtained for this thesis.

1.1 An introductory example

In this section we present a complete process for analyzing the stability of a biochemistry network by means of the tools we developed in this thesis. Although not all our tools are directly involved in this process, this application example illustrates the results we have obtained.

1.1.1 A biochemical network

In [84], Laurent proposed a model for the dynamics of diseases of the central nervous system caused by prions, such as scrapie in sheep and goat, and "mad cow disease" or Creutzfeldt-Jacob disease in humans. The model is based on the protein-only hypothesis, which assumes that infection can be spread by particular proteins (prions) that can exist in two isomeric forms. The normal form PrP^C is harmless, while the infectious form PrP^{Sc} catalyzes a transformation from the normal form to itself. A natural question is: Can a *small amount of PrP^{Sc} cause prion disease?*

The generic kinetic scheme of prion diseases is illustrated as follows:

$$\begin{array}{l} \downarrow 1 \\ PrP^C \quad \xrightarrow{3} PrP^{Sc} \xrightarrow{4} \text{Aggregates.} \\ \downarrow 2 \end{array}$$

Denote by $\left[PrP^C\right]$ and $\left[PrP^{Sc}\right]$ the respective concentrations of PrP^C and PrP^{Sc}. Let ν_i be the rate of Step i for $i = 1, \ldots, 4$. In the above diagram, Step 1 corresponds to the synthesis of native PrP^C, which is considered in the present analysis as a zero-order kinetic process, that is $\nu_1 = k_1$ for some constant k_1. Output reactions (Steps 2 and 4, which correspond to the degradation of native PrP^C and to the formation of aggregates respectively) are taken as first-order rate equations: $\nu_2 = k_2 \left[PrP^C\right]$, $\nu_4 = k_4 \left[PrP^{S_C}\right]$. Step 3 corresponds to the transformation from PrP^C to PrP^{S_C}, which is a nonlinear process:

$$\nu_3 = \left[PrP^C\right] \frac{a\left(1 + b\left[PrP^{S_C}\right]^n\right)}{1 + c\left[PrP^{S_C}\right]^n}.$$

Hence we can describe the model by the following differential equations:

$$\frac{\mathrm{d}\left[PrP^C\right]}{\mathrm{d}t} = \nu_1 - \nu_2 - \nu_3$$
$$\frac{\mathrm{d}\left[PrP^{S_C}\right]}{\mathrm{d}t} = \nu_3 - \nu_4.$$

To simplify notation, we set $x = \left[PrP^C\right]$, $y = \left[PrP^{S_C}\right]$. The model is therefore described by the dynamical system:

$$\frac{\mathrm{d}x}{\mathrm{d}t} = k_1 - k_2 x - ax\frac{(1 + by^n)}{1 + cy^n}$$
$$\frac{\mathrm{d}y}{\mathrm{d}t} = ax\frac{(1 + by^n)}{1 + cy^n} - k_4 y,$$

where experiments in [84] suggest to set $b = 2$, $c = 1/20$, $n = 4$, $a = 1/10$, $k_4 = 50$ and $k_1 = 800$. Now we have:

$$\left\{ \begin{array}{lcl} \frac{\mathrm{d}x}{\mathrm{d}t} & = & f_1 \\ \frac{\mathrm{d}y}{\mathrm{d}t} & = & f_2 \end{array} \right. \quad \text{with} \quad \left\{ \begin{array}{lcl} f_1 & = & \frac{16000+800y^4-20k_2x-k_2xy^4-2x-4xy^4}{20+y^4} \\ f_2 & = & \frac{2(x+2xy^4-500y-25y^5)}{20+y^4} \end{array} \right. . \tag{1.1}$$

A constant solution of the above differential equations is called an *equilibrium*, that is a point $(x, y) \in \mathbb{R}^2$ at which the right hand side equations vanish for some $k_2 \in \mathbb{R}$. We say (x, y) is asymptotically stable if the solutions of differential equations starting out close to (x, y) become arbitrary close to it.

By Routh-Hurwitz criterion [62], the equilibrium (x, y) is asymptotically stable if

$$\Delta_1 := -\left(\frac{\partial f_1}{\partial x} + \frac{\partial f_2}{\partial y}\right) > 0 \text{ and } a_2 := \frac{\partial f_1}{\partial x} \cdot \frac{\partial f_2}{\partial y} - \frac{\partial f_1}{\partial y} \cdot \frac{\partial f_2}{\partial x} > 0.$$

In System (1.1), let p_1 and p_2 be respectively the numerators of f_1 and f_2. The parametric semi-algebraic systems $\mathcal{S}_1 : \{p_1 = p_2 = 0, x > 0, y > 0, k_2 > 0\}$ and $\mathcal{S}_2 : \{p_1 = p_2 = 0, x > 0, y > 0, k_2 > 0, \Delta_1 > 0, a_2 > 0\}$ encode respectively the equilibria and the asymptotically stable hyperbolic equilibria of System (1.1).

1.1.2 Describing the complex solutions

The previous section raises questions on how to compute the real solutions of two parametric polynomial systems $\mathcal{S}_1 : \{p_1 = p_2 = 0, x > 0, y > 0, k_2 > 0\}$ and $\mathcal{S}_2 : \{p_1 = p_2 = 0, k_2 > 0, x > 0, y > 0, \Delta_1 > 0, a_2 > 0\}$. Typically, before studying the real solutions of a polynomial system, one first wants to investigate its complex solutions. Let $\mathcal{C}_1 := \{p_1 = 0, p_2 = 0, x \neq 0, y \neq 0, k_2 \neq 0\}$. We first study the zero set of $\mathcal{C}_1$ in $\mathbb{C}^3$, denoted by $Z_{\mathbb{C}}(\mathcal{C}_1)$.

Under the order $x > y > k_2$, the zero set of $\mathcal{C}_1$ in $\mathbb{C}^3$ is a union of the zero sets of the following three subsystems.

$$R_1 := \left\{ \begin{array}{rcl} (2y^4+1)x - 500y - 25y^5 & = & 0 \\ (k_2+4)y^5 - 64y^4 + (20k_2+2)y - 32 & = & 0 \\ y & \neq & 0 \\ 2y^4+1 & \neq & 0 \\ 32y^4+39y+16 & \neq & 0 \\ k_2 & \neq & 0 \\ k_2+4 & \neq & 0 \end{array} \right. , \quad R_2 := \left\{ \begin{array}{rcl} 2x - 25y + 400 & = & 0 \\ 32y^4+39y+16 & = & 0 \\ k_2+4 & = & 0 \end{array} \right. . \tag{1.2}$$

Each subsystem is of triangular shape and has remarkable algebraic properties: we call them *regular systems*. The set of polynomials encoding the equations in each subsystem is called a *regular chain*. Such a decomposition is called a *triangular decomposition*. The first part of this thesis is dedicated to developing more efficient algorithms for computing such a decomposition.

1.1.3 Describing complex solutions as functions of parameters

In the previous section, all variables have the same status: they are all regarded as unknowns. Alternatively, one may wish to view some of the variables as parameters and investigate how the value of the other variables (let us call them the unknowns) change with the variation of parameter values. For our example, the unknowns are x, y while the only parameter is k_2. We would like to compute the following objects:

- a partition of parameter space into disjoint sets, called *cells*,
- above each connected component of any cell, functions describing the unknowns and depending continuously on the parameters.

We call such an object a *comprehensive triangular decomposition* (CTD). A CTD of $\mathcal{C}_1$ is given by the following piecewise definition:

$$\begin{cases} \{\,\} & k_2 = 0 \\ \{R_2\} & k_2 + 4 = 0 \\ \{R_1\} & k_2 \neq 0 \text{ and } k_2 + 4 \neq 0 \end{cases},$$

where R_1, R_2 are the systems defined by Relation (1.2). Sometimes, we further require that the graphs of the continuous functions defined above each cell are disjoint, which motivates a stronger notion of CTD.

Denote $t_x := (2y^4 + 1)x - 25y^5 - 500y$ and

$$\begin{aligned} r \;&:=\; 100000k_2^8 + 1250000k_2^7 + 5410000k_2^6 + 8921000k_2^5 - 9161219950k_2^4 \\ &-\; 5038824999k_2^3 - 1665203348k_2^2 - 882897744k_2 + 1099528405056. \end{aligned}$$

Let t_y be the following polynomial.

$$\begin{aligned}
t_y := {} & (23268734556450898419888092289684588240000k_2^7 + 88780850506496261345607404805520327377600 0k_2^6 \\
& -64275920104201045426092080735608473398637610 0k_2^5 + 798982465948689385180224786309623594746271260k_2^4 \\
& -755541969292212808074758347883749169568015348 1k_2^3 - 354490122054179307333155209793159741188459844 92k_2^2 \\
& -431875130060632180810654593775701709059288209 6k_2 - 327907507955945276712462277765503291468450043456)y^4 \\
& +(595041692603879832727686208640106565435559923 20 - 145515349655171850022515066001558204896000 00k_2^6 \\
& +554155115787515258964077274056246573122407566 20k_2^3 + 876847598754269841148937318213026162350958803520k_2 \\
& +317749599530866457124059591088318660732882314640k_2^2 - 854826288398480061771370481554049152352160 00k_2^5 \\
& -120352648770516635415131106557179868640000 0k_2^7 + 101785606088976258175525848622703391739538302 00k_2^4)y^3 \\
& +(525266951778505402027801480478861435200000 0k_2^7 - 167530270978266708856920671122396806455219200k_2^4 \\
& +115235109691639562654993861022218266571429229 120k_2^2 + 181667272408330520754764226895080840472636509696 0 \\
& +668319912100483042625432602606969870867763349 760k_2 + 112862573949811720414979561301565008985600 00k_2^6 \\
& -136191397343195728348723172154341170533120 00k_2^5 + 209062102331794345309905270593074607209227397 60k_2^3)y^2 \\
& +(305087509391280246850305169385511280140079029 520k_2 - 343356477061424268437820917723651218855443000k_2^5 \\
& +257371530074079023303501373503345352920980000k_2^6 + 322561009514594974832059146827403356061256455 95k_2^3 \\
& -445476939849013066022926875584021296050000k_2^7 + 294687389203168062136013553346702131219934495 40k_2^2 \\
& +112004292267797955734352101659152288598374293 4720 + 213642750647110707386272530916321910193129180 0k_2^4)y \\
& -163196051967222632295941353115340613924202875 9040 + 752923805329828287871807847129427549600000k_2^7 \\
& +116443127598064787316507772151330198618400 00k_2^6 + 737319470990393398599878903903678608444002400k_2^4 \\
& -314641696590549396895596270561712599814058672 640k_2 - 226733546531989363631975695021134672123615921280k_2^2 \\
& -720519375935590004833313923725484072420748670 40k_2^3 - 364594307740990294702210838952646256405464000k_2^5
\end{aligned}$$

Let R_3 be the regular system $[t_x = 0, t_y = 0, r = 0]$. Then the following piecewise definition describes a stronger CTD of $\mathcal{C}_1$:

$$\begin{cases} \{\,\} & k_2 = 0 \\ \{R_2\} & k_2 + 4 = 0 \\ \{R_3\} & r = 0 \\ \{R_1\} & k_2 \neq 0, k_2 + 4 \neq 0 \text{ and } r \neq 0 \end{cases}.$$

From such a CTD, one could easily count the number of complex solutions depending on parameters:

$$\begin{cases} 0 & k_2 = 0 \\ 4 & k_2 + 4 = 0 \text{ or } r = 0 \\ 5 & k_2 \neq 0, k_2 + 4 \neq 0 \text{ and } r \neq 0 \end{cases}.$$

The third part of this thesis is dedicated to provide such a tool for computing the complex solutions of a parametric polynomial system.

1.1.4 Describing the real solutions

We turn our attention to computing the real solutions of a polynomial system. The zero set of $\{p_1 = 0, p_2 = 0, k_2 > 0\}$ in $\mathbb{R}^3$ is a union of the zero sets of the following two subsystems

$$A_1 := \begin{cases} (2y^4+1)x - 25y^5 - 500y & = & 0 \\ (k_2+4)y^5 - 64y^4 + (2+20k_2)y - 32 & = & 0 \\ k_2 & > & 0 \\ r & \neq & 0 \end{cases}, \quad A_2 := \begin{cases} t_x & = & 0 \\ t_y & = & 0 \\ r & = & 0 \\ k_2 & > & 0 \end{cases}.$$

Each subsystem is called a *regular semi-algebraic system*. System A_1 describes segments of a space curve while system A_2 defines a finite set of points in the three-dimensional real space.

1.1.5 Describing the real solutions as functions of parameters

The CTD introduced in Section 1.1.3 provides a tool for computing the complex solutions of a polynomial system as functions of parameters. We generalize it to compute:

- a partition of the real parametric space into connected cells,

- above each cell, real valued functions describing the unknowns and depending continuously on the parameters, whose graphs are disjoint.

This is achieved by decomposing the intersection of a complex cell with the real space into connected semi-algebraic sets. Such a connected decomposition is obtained by computing a so-called *cylindrical algebraic decomposition* (CAD). For this task, we propose, in the fourth part of this thesis, a totally new algorithm based on triangular decomposition.

For this example, since there is only one parameter, computing a CAD degenerates into isolating the real roots of a univariate polynomial. The polynomial r has four real roots, two of them are positive, which we denote by $0 < \alpha_1 < \alpha_2$. The isolating intervals for α_1 and α_2 are respectively $[3.175933838, 3.175941467]$ and $[14.49724579, 14.49725342]$.

Let B_1 (resp. B_2) be the following two systems:

$$B_1 := \begin{cases} (2y^4+1)x - 25y^5 - 500y & = 0 \\ (k_2+4)y^5 - 64y^4 + (2+20k_2)y - 32 & = 0 \\ y & > 0 \end{cases}, \quad B_2 := \begin{cases} t_x & = 0 \\ t_y & = 0 \\ y & > 0 \end{cases}.$$

Then a CTD of $\mathcal{S}_1$ is given by the following piecewise definition:

$$\begin{cases} \{\,\} & k_2 \le 0 \\ \{B_1\} & 0 < k_2 < \alpha_1 \\ \{B_2\} & k_2 = \alpha_1 \\ \{B_1\} & \alpha_1 < k_2 < \alpha_2 \\ \{B_2\} & k_2 = \alpha_2 \\ \{B_1\} & k_2 > \alpha_2 \end{cases}$$

For each of the six cells, we can compute a sample point, substitute it into the corresponding B_i and count the number of real solutions of the specialized system:

0	1	2	3	2	1
$k_2 \le 0$	$0 < k_2 < \alpha_1$	$k_2 = \alpha_1$	$\alpha_1 < k_2 < \alpha_2$	$k_2 = \alpha_2$	$k_2 > \alpha_2$

Different cells having the same number of real solutions can be merged together

$$\begin{cases} 0 & k_2 \le 0 \\ 1 & k_2 > 0 \text{ and } r > 0 \\ 2 & k_2 > 0 \text{ and } r = 0 \\ 3 & k_2 > 0 \text{ and } r < 0 \end{cases}$$

Thus CTD provides a tool for counting the number of real solutions depending on the parameters.

1.1.6 Analyzing stability of the biochemical network

Since the real solutions of $\mathcal{S}_1$ are exactly the equilibria of System (1.1), we immediately have the following results.

Theorem 1.1. *If* $0 < k_2 < \alpha_1$ *or* $k_2 > \alpha_2$*, then System (1.1) has* 1 *equilibrium; if* $k_2 = \alpha_1$ *or* $k_2 = \alpha_2$*, then System (1.1) has* 2 *equilibria; if* $\alpha_1 < k_2 < \alpha_2$*, then System (1.1) has* 3 *equilibria.*

By a combination of the computation of CTDs of the following four semi-algebraic systems $\mathcal{S}_2 := \{p_1 = 0, p_2 = 0, x > 0, y > 0, k_2 > 0, \Delta_1 > 0, a_2 = 0\}$, $\mathcal{S}_3 := \{p_1 = 0, p_2 = 0, x > 0, y > 0, k_2 > 0, \Delta_1 = 0, a_2 = 0\}$, $\mathcal{S}_4 := \{p_1 = 0, p_2 = 0, k_2 > 0, x > 0, y > 0, \Delta_1 \neq 0, a_2 = 0\}$, and $\mathcal{S}_5 := \{p_1 = 0, p_2 = 0, k_2 > 0, x > 0, y > 0, \Delta_1 = 0, a_2 > 0\}$, we obtain the following theorem for the stability and bifurcation of System (1.1).

Theorem 1.2. *If* $k_2 > \alpha_2$*, see Figure 1.1, the system has* one *hyperbolic equilibrium, which is asymptotically stable. If* $0 < k_2 < \alpha_1$*, see Figure 1.3, the system also has* one *hyperbolic equilibrium, which is asymptotically stable. If* $k_2 = \alpha_1$ *or* $k_2 = \alpha_2$*, the system has* 2 *equilibria: one is nonhyperbolic and the other one is hyperbolic and asymptotically stable. Moreover, the system experiences* bifurcations *at both* $k_2 = \alpha_1$ *and* $k_2 = \alpha_2$*. If* $\alpha_1 < k_2 < \alpha_2$*, then the system has* three *hyperbolic equilibria, two of which are asymptotically stable and the other one is unstable.*

Remark 1.1. *This generalizes the illustrated results of Fig.1(c) in [84], where only concrete values of* k_2 *are given to make sure that System (1.1) is bistable. By symbolic methods presented here, we can give the precise condition.*

1.1.7 Explanation of the experimental results

From these figures, we also observe that: In Figure 1.1, the concentration of PrP^{Sc} (y-coordinate) finally becomes *low* and thus the system enters a *harmless* state. Conversely, in Figure 1.3 the concentration of PrP^{Sc} goes *high* and thus the systems enters a *pathogenic* state. In Figure 1.2, the system exhibits bistability, the *initial concentrations* of PrP^{Sc} determines whether the final state pathogenic or not. We thus deduce the following facts, as stated in paper [84]:

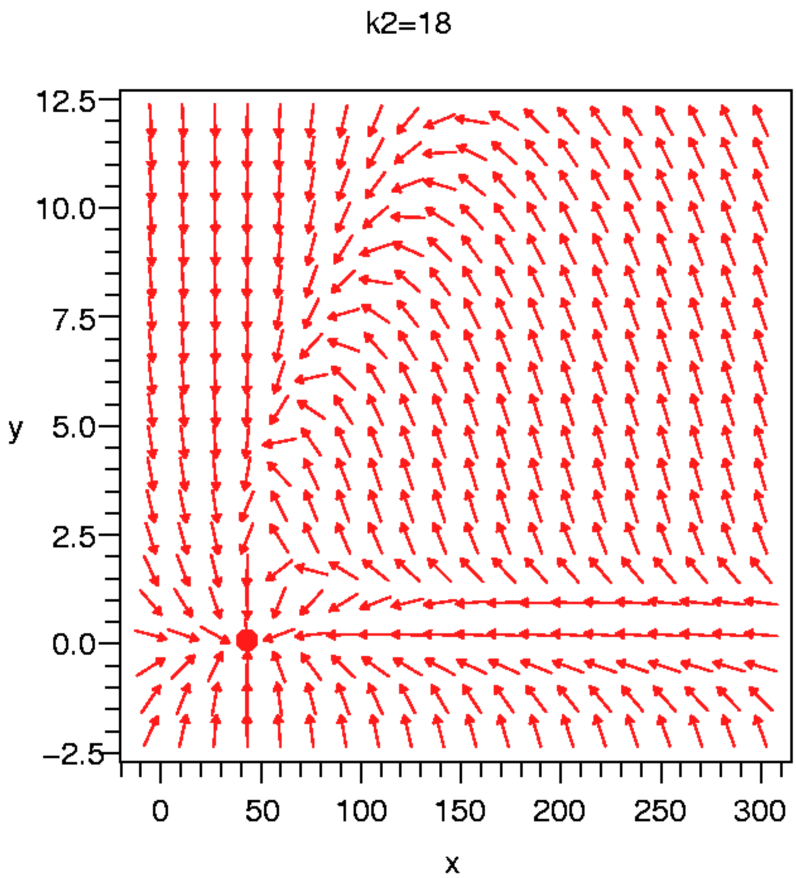


Figure 1.1: Vector field for $k_2 = 18$

- The turnover rate k_2 determines whether it is possible for a pathogenic state to occur.
- As an *answer* to our question, a small amount of PrP^{Sc} does not lead to a pathogenic state when k_2 is large enough.
- Compounds that inhibit addition of PrP^{Sc} can be seen as a possible therapy against prion diseases. However, compounds that *increase the turnover rate* k_2 would be the best therapeutic strategy against prion diseases.

1.2 Main results we have obtained

New algorithms for computing triangular decompositions. We propose new algorithms for computing triangular decompositions of polynomial systems incrementally. With respect to previous work, our improvements are based on a *weakened* no-

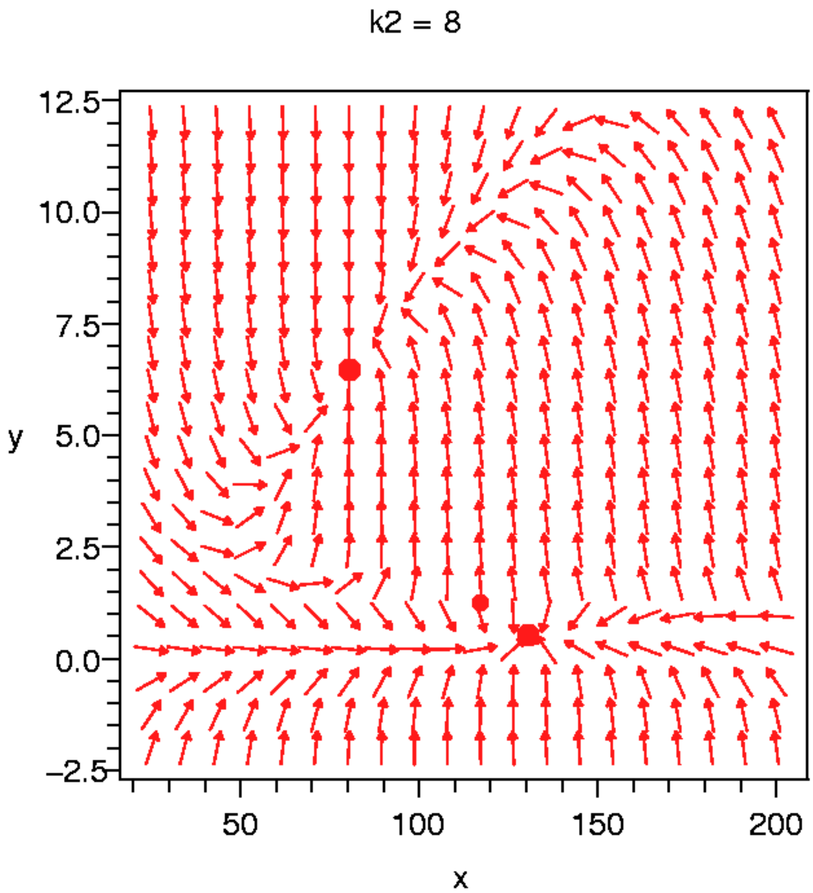


Figure 1.2: Vector field for $k_2 = 8$

tion of a polynomial GCD modulo a regular chain, which permits to greatly simplify and optimize the sub-algorithms. Extracting common work from similar expensive computations is also a key feature of our algorithms. In our experimental results the implementation of our new algorithms, realized with the `RegularChains` library in Maple, outperforms solvers with similar specifications by several orders of magnitude on sufficiently difficult problems. This joint work with Marc Moreno Maza is published in [33].

New approaches for verifying polynomial solvers. We discuss the verification of mathematical software solving polynomial systems symbolically by way of triangular decomposition. Standard verification techniques are highly resource consuming and apply only to polynomial systems that are easy to solve. We exhibit a new approach which manipulates constructible sets represented by regular systems. We provide comparative benchmarks of different verification procedures applied to four solvers on a large set of well-known polynomial systems. Our experimental results illustrate

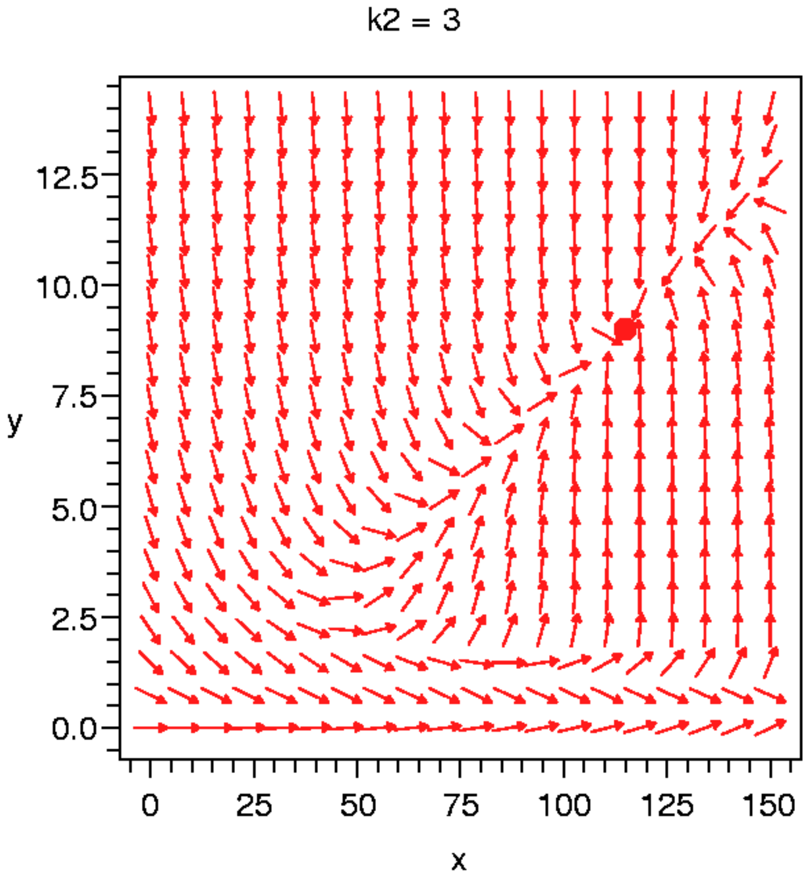


Figure 1.3: Vector field for $k_2 = 3$

the high efficiency of our new approach. In particular, we are able to verify triangular decompositions of polynomial systems which are not easy to solve. This joint work with Marc Moreno Maza, Wei Pan and Yuzhen Xie is published in [35] and the enhanced version is published in [32].

New tools for solving parametric systems. We introduce the concept of comprehensive triangular decomposition (CTD) for a parametric polynomial system F with coefficients in a field. In broad words, it is a finite partition of parameter space into cells such that each cell C is attached with a triangular decomposition of F which is "well-behaved" under specialization at any point of C. We propose several output specifications of CTD addressing different problems regarding the solutions of F as functions of the parameters. We present an algorithm for computing the CTD of F. It relies on a procedure for solving the following set theoretical instance of the coprime factorization problem. Given a family of constructible sets $A_1, \ldots, A_s$, compute a family $B_1, \ldots, B_t$ of pairwise disjoint constructible sets, such that for all

$1 \leq i \leq s$ the set A_i writes as a union of some of the $B_1, \ldots, B_t$. We report on an implementation of our algorithm computing CTDs, based on the `RegularChains` library in MAPLE. We provide comparative benchmarks with MAPLE implementations of related methods for solving parametric polynomial systems. Our results illustrate the good performances of our CTD code. This joint work with Oleg Golubitsky, François Lemaire, Marc Moreno Maza and Wei Pan is published in [30].

New tools for real solving. Regular chains and triangular decompositions are fundamental and well-developed tools for describing the complex solutions of polynomial systems. We propose adaptations of these tools focusing on solutions of the real analogue: semi-algebraic systems. We show that any such system can be decomposed into finitely many *regular semi-algebraic systems*. We propose two specifications (eager and lazy) of such a decomposition and present corresponding algorithms. Under some assumptions, the lazy decomposition can be computed in singly exponential time w.r.t. the number of variables. We have implemented our algorithms and present experimental results illustrating their effectiveness. This joint work with James H. Davenport, John P. May, Marc Moreno Maza, Bican Xia and Rong Xiao is published in [26] and its enhanced version [27].

Cylindrical algebraic decomposition is one of the most important tools for computing with semi-algebraic sets. For an arbitrary finite set $F \subset \mathbb{Q}[y_1, \ldots, y_n]$ we apply comprehensive triangular decomposition in order to obtain an F-invariant cylindrical decomposition of the n-dimensional complex space, from which we extract an F-invariant cylindrical algebraic decomposition of the n-dimensional real space. We report on an implementation of this new approach for constructing cylindrical algebraic decompositions. This joint work with Marc Moreno Maza, Bican Xia and Lu Yang is published in [36].

New tools for studying the equilibria of dynamical systems symbolically. We study continuous dynamical systems defined by autonomous ordinary differential equations, given by parametric polynomial equations. For such systems, we provide semi-algebraic description of their hyperbolic and non-hyperbolic equilibria, their asymptotically stable hyperbolic equilibria, their Hopf bifurcations. To this end, we revisit various criteria on sign conditions for the roots of a real parametric univariate polynomial. In addition, we introduce the notion of comprehensive triangular decomposition of a semi-algebraic system and demonstrate that it is well adapted for our study. This joint work with Marc Moreno Maza is published in [34].

Chapter 2

Background

In this chapter, we first introduce informally the notions of a regular chain and a triangular decomposition, which are the two fundamental concepts in this thesis. We then define formally the two notions and state some important properties. The latter and formal treatment relies on a few necessary notions, notations and results from commutative algebra and algebraic geometry, which are reviewed in Appendix A, p. 198 and Appendix B, p. 207.

2.1 An informal introduction to regular chains and triangular decompositions

In this section, we will not try to provide a precise definition of a regular chain and a triangular decomposition. Instead, we use examples to illustrate instances of regular chains and triangular decompositions.

Let $f(x) := x^2 - x - 1$ be a univariate polynomial in x. From high school mathematics, we know that it has two complex solutions and we can write down explicit formulas for each of the solutions as follows:

$$x = \frac{1+\sqrt{5}}{2} \text{ and } x = \frac{1-\sqrt{5}}{2}.$$

This seems to be a natural specification for the task "solving an equation symbolically". Now we slightly change the leading term of $f(x)$ and consider another polynomial $g(x) = x^5 - x - 1$. Then the roots of $g(x)$ cannot be represented by radicals anymore, as the reader may check, for instance, using the `solve` command in MAPLE[1].

[1] `http://en.wikipedia.org/wiki/Maple_(software)`

This phenomenon is not an exception. In fact, for any $d > 4$, by a deep theory initiated by Évariste Galois[2], there always exist polynomials of degree d whose roots cannot be represented by radicals.

Now consider a multivariate polynomial $f(x_1, \ldots, x_n)$. For a variable order $x_1 < \cdots < x_n$, we call the largest variable x_i appearing in f the *main variable* of f. Assume that x_n is the main variable, we can see f is a univariate polynomial in x_n.

$$f := a_d(x_1, \ldots, x_{n-1})x_n^d + \ldots + a_1(x_1, \ldots, x_{n-1})x_n + a_0(x_1, \ldots, x_{n-1})x_n.$$

By the fundamental theorem of algebra[3], for any $x_1, \ldots, x_{n-1}$, such that $a_d \neq 0$, f has exactly d complex solutions (counting multiplicities) in x_n. Thus, it is not a bad idea to use f itself as a representation of its solutions. In particular, any single nonconstant polynomial is a regular chain.

Let us consider a system of polynomials. We start from a system of linear equations,

$$E := \begin{cases} 2x + y + z - 1 = 0 \\ x + 2y + z - 1 = 0 \\ x + y + 2z - 1 = 0 \end{cases}.$$

Using Gaussian elimination[4], it can be transformed into the following equivalent simpler system

$$\begin{cases} z - \frac{1}{4} = 0 \\ y - \frac{1}{4} = 0 \\ x - \frac{1}{4} = 0 \end{cases}.$$

An interesting feature of this simpler system is that it is of a triangular shape, that is the polynomials appearing in it have different main variables, which is not true for the input system E. The polynomial set $\{x - 1/4, y - 1/4, z - 1/4\}$ is a regular chain while the set of polynomials in E is not a regular chain.

In general, we call a set of polynomials a *triangular set* if different polynomials in it have different main variables. The equations formed by such a triangular set is called a *triangular system*.

[2] http://en.wikipedia.org/wiki/Evariste_Galois
[3] http://en.wikipedia.org/wiki/Fundamental_theorem_of_algebra
[4] http://en.wikipedia.org/wiki/Gaussian_elimination

Let us replace the linear system E by the following nonlinear polynomial system

$$F := \begin{cases} x^2 + y + z - 1 = 0 \\ x + y^2 + z - 1 = 0 \\ x + y + z^2 - 1 = 0 \end{cases}.$$

By a so-called *Gröbner basis*[5] computation, which is a famous tool in computer algebra, under the lexicographic order $z > y > x$, we obtain the following equivalent system:

$$G := \begin{cases} z + y + x^2 - 1 & = & 0 \\ y^2 - y - x^2 + x & = & 0 \\ 2x^2y + x^4 - x^2 & = & 0 \\ x^6 - 4x^4 + 4x^3 - x^2 & = & 0. \end{cases}$$

We observe that the largest variable appearing in the four equations are respectively z, y, y, x. The system G is not a triangular system since y appears twice as a main variable.

Let us factorize the polynomials in G:

$$G := \begin{cases} z + y + x^2 - 1 & = & 0 \\ (y - x)(y + x - 1) & = & 0 \\ x^2(2y + x^2 - 1) & = & 0 \\ x^2(x - 1)^2(x^2 + 2x - 1) & = & 0 \end{cases}.$$

Performing elementary algebraic manipulations, the above system is equivalent to the disjunction of the four systems below, each of which is a triangular system. The equivalence is in the following sense: a tuple (x_0, y_0, z_0) of complex numbers is a solution of G if and only if it is a solution of one the four systems below.

$$\begin{cases} z - x = 0 \\ y - x = 0 \\ x^2 + 2x - 1 = 0 \end{cases}, \quad \begin{cases} z = 0 \\ y = 0 \\ x - 1 = 0 \end{cases}, \quad \begin{cases} z = 0 \\ y - 1 = 0 \\ x = 0 \end{cases}, \quad \begin{cases} z - 1 = 0 \\ y = 0 \\ x = 0 \end{cases}.$$

Moreover, the set of polynomials appearing in each subsystem is a regular chain. Such a decomposition is a *triangular decomposition*[6] of F.

Let us see some examples where triangular sets are not regular chains. The fol-

[5] http://en.wikipedia.org/wiki/Groebner_basis
[6] http://en.wikipedia.org/wiki/Triangular_decomposition

lowing triangular system clearly has no solutions.

$$\begin{cases} yz - 1 = 0 \\ y = 0 \\ x - 1 = 0 \end{cases}.$$

The triangular set $\{x-1, y, yz-1\}$ is not a regular chain. Consider another triangular system

$$\begin{cases} yz^2 + z - 1 = 0 \\ y(y-1) = 0 \\ x - 1 = 0 \end{cases}.$$

For $x = 1$ and $y = 1$, z has two complex solutions. But for $x = 1$ and $y = 0$, z has only one complex solution. In other words, this system is discontinuous w.r.t. back substitutions. The triangular set $\{x-1, y(y-1), yz^2 + z - 1\}$ is not a regular chain either.

Let us now consider a system having infinitely many solutions.

$$F := \begin{cases} z^2 + y^2 - x = 0 \\ zy - x = 0 \end{cases}.$$

Under the order $z > y > z$, the system F can be decomposed into the following two subsystems

$$T_1 := \begin{cases} yz - x = 0 \\ y^4 - xy^2 + x^2 = 0 \\ y \neq 0 \end{cases}, \quad T_2 := \begin{cases} z = 0 \\ y = 0 \\ x = 0 \end{cases}.$$

We verify now that any solution of F is a solution of T_1 or T_2 and vice versa. Firstly, assume that $y \neq 0$, from the second equation of F, we have $z = x/y$. Substitute it into the first equation we have $(x/y)^2 + y^2 - x = 0$. Eliminate the denominators, we obtain the second equation in T_1. Secondly, if $y = 0$, substitute $y = 0$ into both equations of F, we obtain $x = y = z = 0$, that is, T_2 is satisfied. Similarly, for any solution of T_1 or T_2, we can verify that F is satisfied.

Now we have a look at the triangular set T_1. It has several remarkable properties. Firstly, its solution set is nonempty. For example, when $y = 1$, its complex solutions are $\{x^2 - x + 1 = 0, y = 1, z = x\}$. Secondly, for almost all complex values of x (more precisely, except $x = 0$), T_1 has solutions and finitely many solutions in y, z. This suggests that the dimension of system T_1 is 1. Thirdly, for all values of $x \neq 0$, T_1 has four (counting multiplicities) complex solutions in y, z. The triangular set

$\{yz - x, y^4 - xy^2 + x^2\}$ is also an instance of a regular chain. Finally, the system T_1 and T_2 form a triangular decomposition of F.

2.2 A formal definition of regular chain and triangular decomposition

Throughout this thesis, we denote a field by $\mathbf{k}$. We say that a field $\mathbf{k}$ is *algebraically closed* if every nonconstant polynomial in $\mathbf{k}[x]$ has a root in $\mathbf{k}$. An *algebraic closure* of $\mathbf{k}$, denoted by $\mathbf{K}$, is an algebraic extension field of $\mathbf{k}$ which is algebraically closed. Up to an isomorphism that fixes every member of $\mathbf{k}$, an algebraic closure of $\mathbf{k}$ is unique. For example, the field $\mathbb{C}$ of complex numbers is the algebraic closure of the field $\mathbb{R}$ of the real numbers. Let $\mathbf{k}[\mathbf{x}]$ denote the ring of polynomials over $\mathbf{k}$, with ordered variables $\mathbf{x} = x_1 < \cdots < x_n$.

Notations for univariate polynomials. Let $\mathbb{A}$ be a commutative ring and let $\mathbb{A}[x]$ be the ring of the univariate polynomials over $\mathbb{A}$. Let $p = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$, with $a_n \neq 0$, be a polynomial in $\mathbb{A}[x]$. Then the nonnegative integer n is called the *degree* of p, denoted by $\deg(p, x)$; a_n is called the *leading coefficient* of p, denoted by $\mathrm{lc}(p, x)$. The monomial x^n, the term $a_n x^n$, the polynomial $a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$ are respectively called the leading monomial, the leading term and the reductum of p.

Pseudo division. Let f and g be polynomials in $\mathbb{A}[x]$ such that $\deg(g, x) > 0$ and $\mathrm{lc}(g, x)$ is regular (See Section A.2 for the meaning of regular) in $\mathbb{A}$. We define $e = \min(0, \deg(f, x) - \deg(g, x) + 1)$. Then there exists a unique couple (q, r) of polynomials in $\mathbb{A}[x]$ such that we have: $\mathrm{lc}(g, x)^e f = qg + r$ and $r = 0$ or $\deg(r, x) < \deg(g, x)$. The polynomial q (resp. r) is called the *pseudo-quotient* (resp. *pseudo-remainder*) of f by g and denoted by $\mathrm{pquo}(f, g)$ (resp. $\mathrm{prem}(f, g)$). The map $(f, g) \to (q, r)$ is called the *pseudo-division* of f by g.

Notations for polynomials. Let p be a polynomial in $\mathbf{k}[\mathbf{x}]$. 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. Denote by $\mathrm{der}(p)$ the derivative of p w.r.t. $\mathrm{mvar}(p)$, which is also called the *separant* of p w.r.t $\mathrm{mvar}(p)$, denoted by $\mathrm{sep}(p)$. Denote $\mathrm{discrim}(p)$ the discriminant of p w.r.t. $\mathrm{mvar}(p)$. The integer k such that $x_k = \mathrm{mvar}(p)$ is called the *level* of p.

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 to be *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 or $\mathrm{init}(T)$ 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 $\langle T \rangle : h_T^\infty$ (See Section A.2 for this notation).

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 $T \prec S$ or $\mathrm{rank}(T) < \mathrm{rank}(S)$ 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 as $T \sim S$ if neither $T \prec S$ nor $S \prec T$ holds.

Notations for zero sets. Let F and H be two sets of polynomials and T be a triangular set in $\mathbf{k}[\mathbf{x}]$. The *quasi-component* $W(T)$ of T is defined as $V(T) \setminus V(h_T)$. Denote by $\overline{W(T)}$ the Zariski closure (See Section A.2 for this notion) of $W(T)$. Denote by $\prod_{f \in H} f$ the product of polynomials in H. If H is empty, then $\prod_{f \in H} f$ is defined as 1. Let $h := \prod_{f \in H} f$. We define $Z(F, T, H) := (V(F) \cap W(T)) \setminus V(h)$. When F consists of a single polynomial p, we use $Z(p, T, H)$ instead of $Z(\{p\}, T, H)$; when F is empty we just write $Z(T, H)$. When H consists of a single polynomial h, we use $Z(F, T, h)$ instead of $Z(F, T, H)$; when H is empty, we just write $Z(F, T)$.

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 modulo $\mathrm{sat}(T \setminus \{T_{\max}\})$. The empty regular chain is simply denoted by $\varnothing$.

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 $V(F) = \cup_{i=1}^{e} W(T_i)$.

Next we recall some properties of triangular sets and regular chains. These properties will be used explicitly or implicitly in the following chapters.

Lemma 2.1. *Let T be a triangular set in $\mathbf{k}[\mathbf{x}]$. Then, we have*

$$\overline{W(T)} \setminus V(h_T) = W(T) \ \text{ and } \ \overline{W(T)} \setminus W(T) = V(h_T) \cap \overline{W(T)}.$$

Proof. Since $W(T) \subseteq \overline{W(T)}$, we have

$$W(T) = W(T) \setminus V(h_T) \subseteq \overline{W(T)} \setminus V(h_T).$$

On the other hand, $\overline{W(T)} \subseteq V(T)$ implies $\overline{W(T)} \setminus V(h_T) \subseteq V(T) \setminus V(h_T) = W(T)$. This proves the first claim. Observe that we have: $\overline{W(T)} = \left(\overline{W(T)} \setminus V(h_T)\right) \uplus \left(\overline{W(T)} \cap V(h_T)\right)$, where $\uplus$ denotes a disjoint union. We deduce the second one. □

Corollary 2.1. *Let T be a triangular set in $\mathbf{k}[\mathbf{x}]$ and $h \in \mathbf{k}[\mathbf{x}]$ a polynomial. Assume that h_T, the product of the initials of the polynomial in T, divides h. Then we have*

$$\overline{W(T)} \setminus V(h) = W(T) \setminus V(h).$$

Proof. This follows immediately from the identity $\overline{W(T)} \setminus V(h_T) = W(T)$. □

Lemma 2.2 ([6], [14])**.** *Let T be a triangular set in $\mathbf{k}[\mathbf{x}]$. Then the following properties hold:*

- *We have $V(\mathrm{sat}(T)) = \overline{W(T)}$.*
- *Let $\mathbf{u}$ be the free variables of T. Assume $W(T)$ is not empty. Then $\mathrm{sat}(T)$ is an unmixed ideal (See Section A.5 for the meaning of unmixed) with dimension $n - |T|$ such that $\mathrm{sat}(T) \cap \mathbf{k}[\mathbf{u}] = \{0\}$ holds.*

Proposition 2.1 ([6])**.** *If T is a regular chain of $\mathbf{k}[\mathbf{x}]$. Then $W(T)$ is a nonempty set in $\mathbf{K}^n$.*

Remark 2.1. *Let F be a set of polynomials in $\mathbf{k}[\mathbf{x}]$ and $\mathfrak{T}$ be a Kalkbrener or Lazard-Wu triangular decomposition of $V(F)$. Lemma 2.2 and Proposition 2.1 imply the following two important properties: (i) $V(F)$ is empty if and only if $\mathfrak{T}$ is empty; (ii) $\mathfrak{T}$ provides an equidimensional decomposition of $V(F)$.*

Remark 2.2. *Let* T *be a regular chain of* $\mathbf{k}[\mathbf{x}]$. *Let* x_i *be the largest variable appearing in* T. *Then* T *is also a regular chain in* $\mathbf{k}[x_1, \ldots, x_i]$. *We denote by* $\mathrm{sat}_i(T)$ *the saturated ideal of* T *defined in* $\mathbf{k}[x_1, \ldots, x_i]$. *By Proposition B.3, we have* $\mathrm{sat}_i(T)[x_{i+1}, \ldots, x_n] = \mathrm{sat}(T)$. *Let* p *be a polynomial in* $\mathbf{k}[x_1, \ldots, x_i]$. *By Proposition B.2,* p *is regular in* $\mathbf{k}[x_1, \ldots, x_i]/\mathrm{sat}_i(T)$ *if and only if* p *is regular in* $\mathbf{k}[\mathbf{x}]/\mathrm{sat}(T)$. *Thus, in the rest of this thesis, for both cases, we would simply say* p *is regular modulo* $\mathrm{sat}(T)$.

Remark 2.3. *Lemma 2.2 and Proposition 2.1 show that* $\mathrm{sat}(T)$ *is an unmixed ideal. Thus, by Proposition A.15,* p *is regular modulo* $\mathrm{sat}(T)$ *if and only if* p *is regular modulo* $\sqrt{\mathrm{sat}(T)}$.

Chapter 3

Subresultants and Regular GCDs

Calculating polynomial GCDs is a core operation in many algorithms of both symbolic and numeric computation. In the symbolic case, coefficients usually belong to a unique factorization domain (UFD) such as the ring of integers or a polynomial domain over a field. Computing over those domains generally lead to expression swell, which is a notorious problem that all students have observed, when solving on paper, linear systems over the integers.

The work-around is the use of the so-called *modular methods*. See the landmark books [67, 66] for an extensive presentation of those techniques. As an example, consider computing the GCD of two polynomials $f, g \in \mathbb{Z}[x]$, with $\deg(f) > \deg(g) > 0$. It is well known that the Euclidean Algorithm can compute such GCD but will suffer from intermediate expression swell. This phenomenon can be overcome as follows. Suppose for simplicity that g and all successive remainders computed in the Euclidean Algorithm are monic. Under this hypothesis, no divisions will occur during the computation and all coefficients of those polynomials remain integers. (This assumption does not hold in practice and we will relax it shortly.) Let B be the largest integer occurring among those remainders. Consider prime numbers $p_1, p_2, \ldots, p_e$ such that their product exceeds $2B$. (The factor 2 is there because coefficients can be positive or negative.) We compute polynomial GCDs of f and g modulo $p_1, p_2, \ldots, p_e$ successively obtaining polynomials $h_1, h_2, \ldots, h_e$. Using the Chinese Remaindering Theorem (CRT), one can reconstruct a GCD of f and g from $h_1, h_2, \ldots, h_e$. This strategy has at least two advantages. First, computing modulo one prime number limits the size of all coefficients to the size of that prime. If, moreover, that prime has machine word size, coefficient arithmetic is done directly by the hardware. Secondly, computing modulo prime numbers allow the use of fast polynomial arithmetic, such as techniques based on Fast Fourier Transforms. Let us relax now our assumptions that our intermediate remainders are monic. Since divisions are now occuring, our CRT strategy needs to be enhanced in order to recover the denominators of the coefficients in the output GCD. In addition, some prime numbers become ill-conditioned. As a simple example, if $f = (x-1)(x-8)$ and $g = (x-1)(x-5)$, modulo the prime number $p = 3$, the polynomials f and g become identical and thus their GCD, while over $\mathbb{Z}$ their GCD is $x - 1$. Indeed, the remainder of f by g is $-3x + 3$ over $\mathbb{Z}$.

The theory of subresultants helps understanding this difficulty. On the previous example, the resultant of $g/(x-1)$ and $f/(x-1)$ is 3 which, thanks to a well known theorem implies that 3 is ill-conditioned. Returning to the general case of arbitrary $f, g \in \mathbb{Z}[x]$, their subresultant of degree d (for $d < \deg(g)$) is proportional to the polynomial of degree d in the sequence of the Euclidean Algorithm remainders, while all the coefficients of this subresultant are in $\mathbb{Z}$.

More formally, one can say that an important feature of subresultants is their specialization property. In broad terms, and up to technical details which are handled in Section 3.2, the idea is as follows. Consider now f, g over an arbitrary commutative ring $\mathbb{A}$ with $\deg(f) \geq \deg(g) > 0$ and let $\mathcal{I}$ be an ideal of $\mathbb{A}$. Let $\overline{f}$ and $\overline{g}$ be the images of f, g modulo $\mathcal{I}$. Then, from the subresultants of f, g, one can deduce those of $\overline{f}$ and $\overline{g}$. This specialization property plays a central role in the algorithms computing triangular decompositions. Indeed, those algorithms often compute subresultants over some ring $\mathbb{A}$ and use them modulo an ideal $\mathcal{I}$ of $\mathbb{A}$. We can take great advantage of this in the algorithms presented in Chapter 4.

In this chapter, and after reviewing the definition of subresultants, we revisit the specialization property of subresultants in Section 3.2. In the literature, this property always appears with a few hypotheses. Those are not a limitation for most practical cases but they often lead to painful contortions in order to deal with these corner cases in actual algorithms and code. Theorem 3.2 states the specialization property without any hypotheses on the input polynomials. This has greatly helped simplifying the original subroutines of the Triade Algorithm [103].

This latter algorithm relies on a notion of univariate polynomial GCD which was introduced in [103]. It extends the usual notion in the sense that the ring needs not be a UFD. It is well suited to implement key operations such as testing the regularity of a polynomial modulo the saturated ideal of a regular chain. Theorem 32 in [103] and Proportion 3.2 show that it is a powerful tool for computing the intersection of a hypersurface and the quasi-component of a regular chain. In Section 3.3, we relax the original definition due to Marc Moreno Maza in a way that it is even better suited for polynomial system solving, while may be no longer appropriate for other purposes. This weaker definition helps simplifying further the algorithms of [103] in Chapter 4.

The present chapter is based on [33], co-authored with Marc Moreno Maza.

3.1 Definition of subresultants

Let $\mathbb{A}$ be a ring. 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.

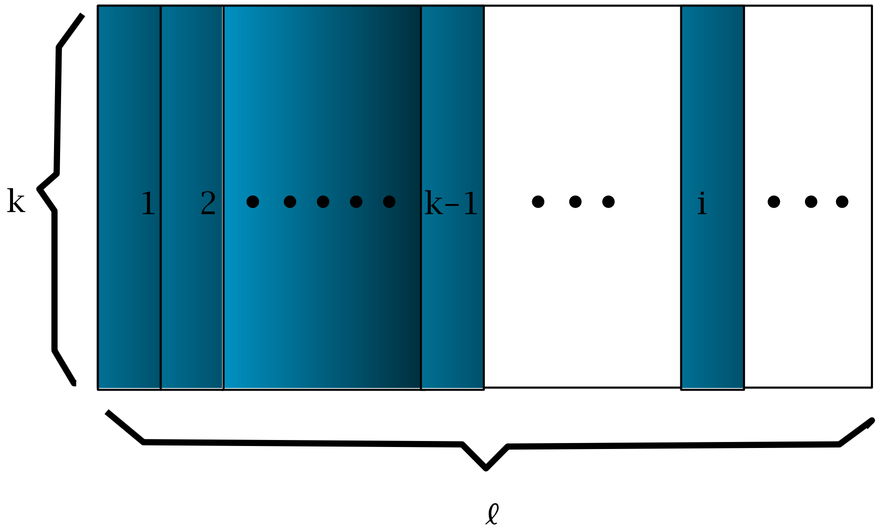
$$L = \left(\begin{array}{cccccccc} a_m & a_{m-1} & \cdots & a_0 & & & & \\ & a_m & a_{m-1} & \cdots & a_0 & & & \\ & & \ddots & \ddots & & & \ddots & \\ & & & a_m & a_{m-1} & & \cdots & a_0 \\ b_n & b_{n-1} & \cdots & b_0 & & & & \\ & b_n & b_{n-1} & \cdots & b_0 & & & \\ & & \ddots & \ddots & & & \ddots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

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 $0 \leq i < \lambda$, let L_i be the submatrix of S formed by removing the bottom i rows that include the coefficients of f and the bottom i rows that include the coefficients of g. 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 $\mathbb{A}$ be a ring and let $k \leq \ell$ 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 j_{th} column of M, for $j = k \cdots \ell$. Let $\mathrm{dpol}(M) :=$

$\sum_{j=k}^{\ell} \det(M_j)x^{\ell-j}$ and 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 matrix defined by $M_{ij} = \mathrm{coeff}(f_i, x^{\ell-j})$, for $1 \leq i \leq k$ and $1 \leq j \leq \ell$. We then define $\mathrm{dpol}(f_1, \ldots, f_k) = \mathrm{dpol}(M)$.

Proposition 3.1. *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 to cover f and g as follows. If $m \geq n$, 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 $s_\lambda = a_m$.

3.2 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 any literature that covers all the corner cases. Therefore, we provide here a self-contained proof.

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

Lemma 3.1. *Let k be an integer such that $0 \leq k < \lambda$. Assume that $\phi(s_k) \neq 0$ holds. Then either $\phi(a_m) \neq 0$ or $\phi(b_n) \neq 0$ holds. Moreover, we have both $\deg(\phi(f)) \geq k$ and $\deg(\phi(g)) \geq k$.*

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 $i \geq k, j \geq k$ such that $\phi(a_i) \neq 0$ and $\phi(b_j) \neq 0$. The conclusion follows. $\square$

Lemma 3.2. *Assume that $\phi(s_0) = \cdots = \phi(s_{\lambda-1}) = 0$ hold. Then, if $m \leq n$, we have*

(1) *if $\phi(a_m) \neq 0$ and $\phi(b_n) = \cdots = \phi(b_m) = 0$ hold, then $\phi(g) = 0$,*

(2) *if $\phi(a_m) = 0$ and $\phi(b_n) \neq 0$ hold, then $\phi(f) = 0$.*

Symmetrically, if $m > n$, we have

(3) *if $\phi(b_n) \neq 0$ and $\phi(a_m) = \cdots = \phi(a_n) = 0$ hold, then $\phi(f) = 0$,*

(4) *if $\phi(b_n) = 0$ and $\phi(a_m) \neq 0$ hold, then $\phi(g) = 0$.*

Proof. We prove (1) and (2), whose correctness implies (3) and (4) by symmetry. Let $i = \lambda - 1 = m - 1$, then we have

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

Therefore

$$s_{m-1} = \begin{vmatrix} a_m & \cdots & a_0 & \\ & \ddots & \ddots & \\ & & a_m & a_{m-1} \\ b_n & \cdots & b_m & b_{m-1} \end{vmatrix}.$$

So from $\phi(b_n) = \cdots = \phi(b_m) = 0$ and $\phi(s_{m-1}) = 0$, we conclude that $\phi(b_{m-1}) = 0$. On the other hand, if $\phi(a_m) = 0$ and $\phi(b_n) \neq 0$, then $\phi(a_{m-1}) = 0$.

Now let consider S_{m-2}. We have

$$s_{m-2} = \begin{vmatrix} a_m & a_{m-1} & \cdots & a_0 & & \\ & \ddots & & \ddots & & \\ & & a_m & a_{m-1} & a_{m-2} \\ b_n & \cdots & b_{m-1} & b_{m-2} & \\ & b_n & \cdots & b_{m-1} & b_{m-2} \end{vmatrix}.$$

From $\phi(b_{m-1}) = 0$, we conclude that $\phi(b_{m-2}) = 0$. From $\phi(a_{m-1}) = 0$, we conclude that $\phi(a_{m-2}) = 0$.

So on so forth, finally, if $\phi(a_m) \neq 0$ and $\phi(b_n) = \cdots = \phi(b_m) = 0$, we deduce that $\phi(b_i) = 0$, for all $0 \leq i \leq m-1$, which implies that $\phi(g) = 0$; if $\phi(a_m) = 0$ and $\phi(b_n) \neq 0$, we deduce that $\phi(a_{m-1}) = \cdots = \phi(a_0) = 0$, which implies that $\phi(f) = 0$. □

Lemma 3.3. *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

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

Proof. The matrix M of the polynomials $x^{n-1-i}f, \ldots, xf, f, x^{m-1-i}g, \ldots, xg, g$ is as follows

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

We know that $S_i = \mathrm{dpol}(M)$. If $m' = m$ and $n' \geq i$, then $n - n' \leq n - i$. Therefore we have

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

If $n' = n$ and $m' \geq i$, then $m - m' \leq m - i$. Therefore we have

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

□

Theorem 3.1 (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 3.3. □

Remark 3.1. *This theorem provides some corner cases which were not covered by other literatures, such as Mishra's book "Algorithmic Algebra" [101]. For example, the case $m = n = m' = n'$, $i = n' - 1$ is not covered by Lemma 7.8.1 nor Corollary 7.8.2 in [101]. The case $m = n = m' = n' + 1$, $i = n'$ is not covered either.*

On the other hand, this theorem covers all useful cases such as those needed for computing GCDs of specialized polynomials, see Theorem 3.2.

Theorem 3.2. *We have the following relations between the subresultants and the GCD 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 $\gcd(\phi(f), \phi(g)) = \phi(S_k)$.*

(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 $\gcd(\phi(f), \phi(g)) = \phi(f)$; symmetrically, if $m > n$ and $\phi(b_n) \neq 0$, then we have $\gcd(\phi(f), \phi(g)) = \phi(g)$.*

(2b) *If $m \leq n$ and $\phi(a_m) = 0$ but $\phi(b_n) \neq 0$, then we have $\gcd(\phi(f), \phi(g)) = \phi(g)$; symmetrically, if $m \geq n$ and $\phi(b_n) = 0$ but $\phi(a_m) \neq 0$, then we have $\gcd(\phi(f), \phi(g)) = \phi(f)$.*

(2c) *If $\phi(a_m) = \phi(b_n) = 0$, then*

$$\gcd(\phi(f), \phi(g)) = \gcd(\phi(\mathrm{red}(f)), \phi(\mathrm{red}(g))).$$

Proof. Let us first prove (1). W.l.o.g, we assume $\phi(a_m) \neq 0$. From Lemma 3.1, we know that $k \leq n'$. So for all $i < k$, we have $i < n'$. By Theorem 3.1, we have

- for $i < k$, $s_i(\phi(f), \phi(g)) = 0$,
- if $k < n'$, we have $s_k(\phi(f), \phi(g)) \neq 0$,
- if $k = n'$, we have $s_k(\phi(f), \phi(g)) = \phi(b_{m'}) = \mathrm{lc}(\phi(g)) \neq 0$.

Thus $\gcd(\phi(f), \phi(g)) = \phi(S_k)$.

Next we prove $(2a)$. By symmetry, we prove it when $m \leq n$. If $\phi(b_n) = \cdots = \phi(b_m) = 0$, it follows directly from Lemma 3.2. Otherwise, we have $n' \geq m$. Thus for all $i < m$, we have $i < n'$. By Theorem 3.1, we have $\phi(S_i) = \phi(a_m)^{n-n'} S_i(\phi(f), \phi(g))$. Thus $\phi(s_i) = 0$ implies that $s_i(\phi(f), \phi(g)) = 0$. Therefore we deduce that $\phi(f) = \gcd(\phi(f), \phi(g))$.

Finally $(2b)$ follows directly from Lemma 3.2 and $(2c)$ is obviously true. □

3.3 Regular GCDs

Definition 3.1. *Let $\mathbb{A}$ be a commutative ring with unity. 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;*

(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, $\operatorname{prem}(p, g) = \operatorname{prem}(t, g) = 0$.*

Example 3.1. *Let $p := y^4 + x + 1$ and $t := 2y^2 + x$ be two polynomial of $\mathbb{Q}[x, y]$. Let $\mathcal{I} := \langle (x + 2)^4 \rangle$, $\mathbb{A}_1 := \mathbb{Q}[x]/\mathcal{I}$ and $\mathbb{A}_2 := \mathbb{Q}[x]/\sqrt{\mathcal{I}}$. Next we show that $g := t$ is a regular GCD of p and t in $\mathbb{A}_2[y]$ but not in $\mathbb{A}_1[y]$.*

To see this, by Definition 3.1, it is enough to check if (R_3) holds or not. Note that we have $\operatorname{prem}(p, g) = 2(x + 2)^2$ in $\mathbb{Q}[x, y]$, which implies that $\operatorname{prem}(p, g) = 0$ holds in $\mathbb{A}_2[y]$ but not in $\mathbb{A}_1[y]$. Therefore $g := t$ is a regular GCD of p and t in $\mathbb{A}_2[y]$ but not in $\mathbb{A}_1[y]$.

Definition 3.1 was introduced in [104] as part of a formal framework for algorithms manipulating regular chains [51, 85, 43, 81, 141]. In this section, the ring $\mathbb{A}$ will always be of the form $\mathbf{k}[\mathbf{x}]/\sqrt{\operatorname{sat}(T)}$. Thus, a regular GCD of p, t in $\mathbb{A}[y]$ is also called a regular GCD of p, t modulo $\sqrt{\operatorname{sat}(T)}$.

Proposition 3.2. *For $1 \le k \le n$, 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 polynomials with main variable x_k. Assume $T \cup \{t\}$ is a regular chain and g is a regular GCD of p and t modulo $\sqrt{\operatorname{sat}(T)}$. We have:*

(i) if $\operatorname{mdeg}(g) = \operatorname{mdeg}(t)$, then $\sqrt{\operatorname{sat}(T \cup t)} = \sqrt{\operatorname{sat}(T \cup g)}$ and $W(T \cup t) \subseteq Z(h_g, T \cup t) \cup W(T \cup g) \subseteq \overline{W(T \cup t)}$ both hold,

(ii) if $\operatorname{mdeg}(g) < \operatorname{mdeg}(t)$, let $q = \operatorname{pquo}(t, g)$, then $T \cup q$ is a regular chain and the following two relations hold:

(ii.a) $\sqrt{\operatorname{sat}(T \cup t)} = \sqrt{\operatorname{sat}(T \cup g)} \cap \sqrt{\operatorname{sat}(T \cup q)}$,

(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

$$h_g^{e_0} t = qg + r \quad \text{and} \quad r \in \sqrt{\operatorname{sat}(T)} \tag{3.1}$$

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

$$(h_T)^{e_1}(h_g^{e_0}t - qg)^{e_1} \in \langle T \rangle \quad (3.2)$$

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 (3.2), 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 (3.1).

We prove (ii). Assume $\mathrm{mdeg}(t) > \mathrm{mdeg}(g)$. With (3.1) and (3.2), 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 (3.1) we deduce that $f \in \sqrt{\mathrm{sat}(T \cup t)}$ holds and so does $(ii.a)$. With (3.1), we have $(ii.b)$ holds.

We prove (iii) and (iv). Definition 3.1 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). □

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 subresulant 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 3.3. *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 3.1, it suffices to prove that both $\mathrm{prem}(p, S_j, x_k) = 0$ and $\mathrm{prem}(t, S_j, x_k) = 0$ hold in $\mathbb{A}$. By symmetry we only prove the former equality.

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 3.2, 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 $\mathbb{A}$. $\square$

Chapter 4

Algorithms for Computing Triangular Decompositions of Polynomial Systems

In this chapter, we propose new algorithms for computing triangular decompositions of polynomial systems incrementally. With respect to previous work, our improvements are based on a *weakened* notion of a polynomial GCD modulo a regular chain, which permits to greatly simplify and optimize the sub-algorithms. Extracting common work from similar expensive computations is also a key feature of our algorithms. In our experimental results the implementation of our new algorithms, realized with the `RegularChains` library in MAPLE, outperforms solvers with similar specifications by several orders of magnitude on sufficiently difficult problems.

4.1 Introduction

The Characteristic Set Method [132] of Wu has freed Ritt's decomposition from polynomial factorization, opening the door to a variety of discoveries in polynomial system solving. In the past two decades the work of Wu has been extended to more powerful decomposition algorithms and applied to different types of polynomial systems or decompositions: differential systems [13, 78], difference systems [63], real parametric systems [138], primary decomposition [112], cylindrical algebraic decomposition [36]. Today, triangular decomposition algorithms provide back-engines for computer algebra system front-end solvers, such as MAPLE's `solve` command.

Algorithms 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 $S_1, \ldots, S_e$. 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 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. From this other angle, two types of algorithms are essentially different: 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 the algorithms in [127] belong to the first type in each classification. Kalkbrener's algorithm [81], which is an elimination method solving in the sense of the "generic zeros", has brought efficient techniques, based on the concept of a *regular chain*. Other work [85, 104] on triangular decomposition algorithms focus on incremental solving. This principle is quite attractive, since it allows to control the properties and size of the intermediate computed objects. It is used in other areas of polynomial system solving such as the probabilistic algorithm of Lecerf [87] based on lifting fibers and the numerical method of Sommese, Verschelde, Wampler [114] based on diagonal homotopy.

Incremental algorithms for triangular decomposition 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 [85] and the one of Moreno Maza [104] are quite involved and difficult to analyze and optimize.

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

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

(See Section 2.2 for the notion of a regular chain and related concepts and notations.) Let us illustrate the geometrical meaning of this operation and its relation with incremental triangular decomposition by the following example.

Example 4.1. *Consider a polynomial system* $F := \{p_1, p_2, p_3\}$ *in* $\mathbb{Q}[x < y < z]$*, where*

- $p_1 := x^2 + y^2 + z^2 - 4$,
- $p_2 := x^2 + y^2 - z^2 - 1$,
- $p_3 := z^3 + xy - 1$.

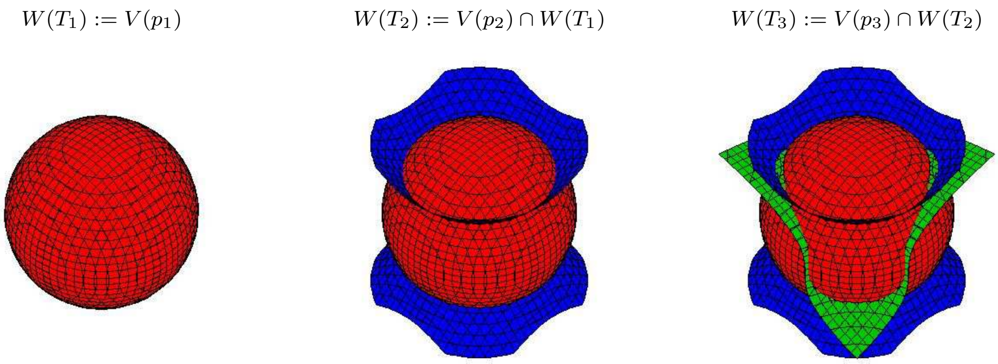
A triangular decomposition of F *can be computed incrementally as follows. We first compute a triangular decomposition of* p_1 *by calling* $\mathsf{Triangularize}(p_1)$*. (See Section 4.3 for the specification of this function.) The output simply consists of one regular chain* $T_1 := \{p_1\}$*. Next we compute a triangular decomposition of* $\{p_1, p_2\}$*, which is achieved by calling* $\mathsf{Intersect}(p_2, T_1)$*, whose output consists of one regular chain* T_2*, where*

$$T_2 := \begin{cases} 2z^2 - 3 \\ 2y^2 + 2x^2 - 5 \end{cases}$$

Finally we compute a triangular decomposition of F *by calling* $\mathsf{Intersect}(p_3, T_2)$*, which consists of a regular chain* T_3*, where*

$$T_3 := \begin{cases} 3z + 2xy - 2 \\ 16xy + 8x^4 - 20x^2 + 19 \\ 64x^8 - 320x^6 + 960x^4 - 1400x^2 + 361 \end{cases}$$

The geometrical meaning of the above process is illustrated by the following pictures.



In the first picture, the two-dimensional red ball depicts the variety of p_1 *(denoted by* $V(p_1)$*), which is also the zero set of* T_1 *(denoted by* $W(T_1)$*). In the second picture, the blue surface depicts the variety of* p_2 *(denoted by* $V(p_2)$*), whose intersection with* $W(T_1)$ *is the zero set of* T_2 *(denoted by* $W(T_2)$*), which is a union of two one-dimensional circles. In the third picture, the green surface describes the variety* *of p_3 (denoted by $V(p_3)$), whose intersection with the zero set of T_2 is the zero set of T_3 (denoted by $W(T_3)$), which is exactly the points in the zero set of F.*

In this work, we exhibit an algorithm for computing $\mathsf{Intersect}(p, T)$ which is conceptually simpler and practically much more efficient than those of [85, 104]. Our improvements result mainly from two new ideas.

Weakened notion of polynomial GCDs modulo regular chain. Modern algorithms for triangular decomposition rely implicitly or explicitly on a notion of GCD for univariate polynomials over an arbitrary commutative ring. A formal definition was proposed in [104] (see Definition 3.1) 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 the regular chain T. A modular algorithm for computing these GCDs appears in [89]: if $\mathrm{sat}(T)$ is known to be radical, the performance (both in theory and practice) of this algorithm are very satisfactory whereas if $\mathrm{sat}(T)$ is not radical, the complexity of the algorithm increases substantially w.r.t. the radical case. In this paper, the ring $\mathbb{A}$ will be of the form $\mathbf{k}[\mathbf{x}]/\sqrt{\mathrm{sat}(T)}$ while our algorithms will not need to compute 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 leads us to a very simple procedure for computing such GCDs, see Theorem 3.3. To this end, we rely on the *specialization property of subresultants.* Section 3.2 reviews this property and provides corner cases for which we could not find a reference in the literature.

Extracting common work from similar computations. 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 computing

(s_1) the resultant r of p and t w.r.t. v,

(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 of Section 4.3 extend this simple observation for computing $\mathsf{Intersect}(p, T)$ with an arbitrary regular chain. In broad terms, the intermediate polynomials computed during the "elimination phasis" of $\mathsf{Intersect}(p, T)$ are recycled for performing the "extension phasis" at essentially no cost.

The techniques developed for $\mathsf{Intersect}(p, T)$ are applied to other key sub-algorithms, such as:

- the regularity test of a polynomial modulo the saturated ideal of a regular chain, see Section 4.3,

- the squarefree part of a regular chain, see Section 4.7.

The primary application of the operation Intersect is to obtain triangular decomposition encoding all the points of the zero set of the input system. However, we also derive from it in Section 4.6 an algorithm for computing triangular decompositions in the sense of Kalkbrener.

Experimental results. We have implemented the algorithms presented in this thesis within the `RegularChains` library in MAPLE, leading to a new implementation of the Triangularize command. In Section 4.8, we report on various benchmarks. This new version of Triangularize outperforms the previous ones (based on [104]) by several orders of magnitude on sufficiently difficult problems. Other MAPLE commands or packages for solving polynomial systems (the `WSolve` package, the `Groebner:-Solve` command and the `Groebner:-Basis` command for a lexicographical term order) are also outperformed by the implementation of the algorithms presented in this paper both in terms of running time and, in the case of engines based on Gröbner bases, in terms of output size.

This chapter is based on paper [33], co-authored with Marc Moreno Maza.

4.2 Properties of regular chains

We review hereafter the notion of iterated resultants and state basic properties (Propositions 4.2,4.1, 4.3, 4.4, and Corollaries 4.2, 4.3) of regular chains, which are at the core of the proofs of the algorithms of Section 4.3.

Iterated resultant and iterated pseudo-remainder. Let $p, q \in \mathbf{k}[\mathbf{x}]$. Assume q is nonconstant and let $v = \operatorname{mvar}(q)$. We define $\operatorname{res}(p, q, v)$ as follows: if the degree $\deg(p, v)$ of p in v is null, then $\operatorname{res}(p, q, v) = p$; otherwise $\operatorname{res}(p, q, v)$ is the resultant of p and q w.r.t. v. Let T be a triangular set of $\mathbf{k}[\mathbf{x}]$. We define $\operatorname{res}(p, T)$ (resp. $\operatorname{prem}(p, T)$) by induction: if $T = \varnothing$, then $\operatorname{res}(p, T) = p$ (resp. $\operatorname{prem}(p, T) = p$); otherwise let v be greatest variable appearing in T, then $\operatorname{res}(p, T) = \operatorname{res}(\operatorname{res}(p, T_v, v), T_{<v})$ (reps. $\operatorname{prem}(p, T) = \operatorname{prem}(\operatorname{prem}(p, T_v, v), T_{<v})$).

Proposition 4.1 (Th. 6.1. in [6])**.** *Let p and T be respectively a polynomial and a regular chain of $\mathbf{k}[\mathbf{x}]$. Then,* $\operatorname{prem}(p, T) = 0$ *holds if and only if* $p \in \operatorname{sat}(T)$ *holds.*

Proof. Let $T = \{t_1, \ldots, t_s\}$ with $\mathrm{mvar}(t_i) < \mathrm{mvar}(t_{i+1})$ and let $h_i = \mathrm{init}(t_i)$. Let $r = \mathrm{prem}(p, T)$. Then there exists nonnegtive integers $e_1, \ldots, e_s$ and polynomials $q_1, \ldots, q_s$ of $\mathbf{k}[\mathbf{x}]$ such that $\prod_{i=1}^{s} h_i^{e_i} p = \sum_{i=1}^{s} q_i t_i + r$.

If $r = 0$, then obviously $p \in \mathrm{sat}(T)$ holds. Next we prove another direction by induction. If T is the empty regular chain, then $p = 0$ and thus $\mathrm{prem}(p, T) = 0$ trivially holds.

Now assume that the proposition holds for $i = s - 1$. Denote $T_i = \{t_1, \ldots, t_i\}$, for $i = 1, \ldots, s$. Since $\mathrm{prem}(p, T) = \mathrm{prem}(\mathrm{prem}(p, t_s), T_{s-1})$, to prove $\mathrm{prem}(p, T) = 0$, by induction it is enough to prove $\mathrm{prem}(p, t_s) \in \mathrm{sat}(T_{s-1})$. By Theorem B.1, we have $\mathrm{sat}(T) = \langle \mathrm{sat}(T_{s-1}), t_s \rangle : h_s^{\infty}$ hold. Then the conclusion directly follows from Proposition B.7.

□

Lemma 4.1. *Let $p \in \mathbf{k}[\mathbf{x}]$ be a polynomial and $T \subset \mathbf{k}[\mathbf{x}]$ be a zero-dimensional regular chain. Then the following statements are equivalent:*

(*i*) *The iterated resultant* $\mathrm{res}(p, T) \neq 0$.

(*ii*) *The polynomial p is regular modulo $\langle T \rangle$.*

(*iii*) *The polynomial p is invertible modulo $\langle T \rangle$.*

Proof. "(*i*) ⇒ (*ii*)" Let $r := \mathrm{res}(p, T)$. Then there exist polynomials $A_i \in \mathbf{k}[\mathbf{x}]$, $0 \leq i \leq n$, such that $r = A_0 p + \sum_{i=1}^{n} A_i T_i$. So $r \neq 0$ implies p is invertible modulo $\langle T \rangle$. Therefore, p is regular modulo $\langle T \rangle$.

"(*ii*) ⇒ (*iii*)" Since p is regular modulo $\langle T \rangle$ and T is a zero-dimensional regular chain, by Lemma A.1, we have $V(f, T) = \emptyset$. Thus f is invertible modulo $\langle T \rangle$.

"(*iii*) ⇒ (*i*)" Assume $\mathrm{res}(p, T) = 0$, then we claim that p and T have at least one common solution, which is a contradiction to (*iii*).

Let $T = \{t_1, \ldots, t_n\}$ with $\mathrm{mvar}(t_i) < \mathrm{mvar}(t_{i+1})$ and let $h_i = \mathrm{init}(t_i)$. Denote $T_i = \{t_1, \ldots, t_i\}$. We prove our claim by induction on i. If $i = 1$, the claim obviously holds. Now we assume that the claim holds for $i < n$.

(1) If $\mathrm{mvar}(p) < x_n$, then $\mathrm{res}(p, T) = \mathrm{res}(p, T_{n-1})$. By induction hypothesis, there exist $\xi_1, \xi_2, \cdots, \xi_{n-1} \in \mathbf{K}$, such that $\xi' = (\xi_1, \xi_2, \cdots, \xi_{n-1})$ is a common solution of p and T_{n-1}. Since T is a zero-dimensional regular chain, h_n is invertible modulo $\langle T_{n-1} \rangle$ (by "(*ii*) ⇒ (*iii*)"). So $h_n(\xi') \neq 0$, which implies that there exists a $\xi_n \in \mathbf{K}$, such that $\xi := (\xi_1, \xi_2, \cdots, \xi_{n-1}, \xi_n)$ is a solution of t_n. Therefore ξ is a common solution of p and T.

(2) If $\mathrm{mvar}(p) = x_n$, then $\mathrm{res}(p, T) = \mathrm{res}(\mathrm{res}(p, t_n, x_n), T_{n-1}) = 0$. By induction hypothesis, there exists $\xi' = (\xi_1, \xi_2, \cdots, \xi_{n-1})$, such that $\mathrm{res}(p, t_n, x_n)(\xi') = T_{n-1}(\xi') = 0$ and $h_n(\xi') \neq 0$. So by the specialization property of resultant, $\mathrm{res}(p(\xi'), t_n(\xi'), x_n) = 0$, which implies that there exists a $\xi_n \in \mathbf{K}$, such that $\xi := (\xi_1, \xi_2, \cdots, \xi_{n-1}, \xi_n)$ is a common solution of p and t_n. Therefore ξ is a common solution of p and T.

□

Proposition 4.2. *Let $p \in \mathbf{k}[\mathbf{x}]$. Let $T \subset \mathbf{k}[\mathbf{x}]$ be a regular chain. Then p is regular modulo* $\mathrm{sat}(T)$ *if and only if the iterated resultant* $\mathrm{res}(p, T)$ *is not zero.*

Proof. Let $T = \{t_1, \ldots, t_s\}$ with $\mathrm{mvar}(t_i) < \mathrm{mvar}(t_{i+1})$ and let $h_i = \mathrm{init}(t_i)$. Denote $T_i = \{t_1, \ldots, t_i\}$. Let $\mathbf{u} = u_1, \ldots, u_d$ and $\mathbf{y} = y_1, \ldots, y_m$ be respectively the free and the main variables of T. Let S be the set $\mathbf{k}[u_1, \ldots, u_d] \setminus \{0\}$. Let ϕ be the homomorphism $\mathbf{k}[\mathbf{x}] \to S^{-1}\mathbf{k}[\mathbf{x}]$. Note that $S^{-1}\mathbf{k}[\mathbf{x}]$ is the ring $\mathbf{k}(\mathbf{u})[\mathbf{y}]$.

Let $\mathrm{sat}(\phi(T_i))$ be the saturated ideal of $\phi(T_i)$ defined in $\mathbf{k}(\mathbf{u})[\mathbf{y}]$. By Theorem 1.1 of [14], for any polynomial of $f \in \mathbf{k}[\mathbf{x}]$, f is regular in $\mathbf{k}[\mathbf{x}]/\mathrm{sat}(T_i)$ if and only if $\phi(f)$ is regular in $\mathbf{k}(\mathbf{u})[\mathbf{y}]/\mathrm{sat}(\phi(\mathrm{sat}(T_i)))$. Thus $\phi(T)$ is a zero-dimensional regular chain in $\mathbf{k}(\mathbf{u})[\mathbf{y}]$. On the other hand we have $\mathrm{res}(\phi(p), \phi(T)) = \phi(\mathrm{res}(p, T))$. Thus, by Lemma 4.1, p is regular modulo $\mathrm{sat}(T)$ if and only if $\mathrm{res}(p, T) \neq 0$. □

Corollary 4.1. *Let $T \subset \mathbf{k}[\mathbf{x}]$ be a triangular set. Then T is a regular chain if and only if* $\mathrm{res}(h_T, T) \neq 0$.

Proof. It follows directly from the definition of regular chain and Proposition 4.2. □

Proposition 4.3 (Prop. 5 in [104])**.** *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 modulo* $\sqrt{\mathrm{sat}(T)}$. *Then p is also regular modulo* $\sqrt{\mathrm{sat}(T')}$.

Proof. By Proposition A.9, a radical ideal is an intersection of its associated prime ideals. By Proposition A.3, any associated prime ideal of $\sqrt{\mathrm{sat}(T')}$ contains an associated prime ideal of $\sqrt{\mathrm{sat}(T)}$. Since $\sqrt{\mathrm{sat}(T')}$ and $\sqrt{\mathrm{sat}(T)}$ are unmixed, we deduce that any associated prime of $\sqrt{\mathrm{sat}(T')}$ is an associated prime ideal of $\sqrt{\mathrm{sat}(T)}$. Since p is also regular modulo $\sqrt{\mathrm{sat}(T)}$, by Proposition A.10, p is regular modulo $\sqrt{\mathrm{sat}(T')}$. □

Proposition 4.4. *Let $p \in \mathbf{k}[\mathbf{x}]$ 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 $r = \text{prem}(p, T_{\geq v})$, there exists an integer $e_0 \geq 0$ and a polynomial $f \in \langle T_{\geq v} \rangle$ such that $\text{init}(T_{\geq v})^{e_0} p = f + r$. On the other hand, $r \in \sqrt{\text{sat}(T_{<v})}$, therefore there exists an integer $e_1 \geq 0$ such that $\text{init}(T_{<v})^{e_1} (\text{init}(T_{\geq v})^{e_0} p - f)^{e_1} \in \langle T_{<v} \rangle$, which implies that $p \in \sqrt{\text{sat}(T)}$. $\square$

Corollary 4.2. *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 $\text{mvar}(p) = x_{k+1}$ such that $\text{init}(p)$ is regular w.r.t. both $\text{sat}(T)$ and $\text{sat}(T')$. Assume that $\sqrt{\text{sat}(T)} \subseteq \sqrt{\text{sat}(T')}$ holds. Then we also have $\sqrt{\text{sat}(T \cup p)} \subseteq \sqrt{\text{sat}(T' \cup p)}$.*

Proof. This follows easily from Proposition 4.1. $\square$

Corollary 4.3. *Let $p \in \mathbf{k}[\mathbf{x}]$ and $T \subset \mathbf{k}[\mathbf{x}]$ be a regular chain. Let $v := \text{mvar}(p)$ and $r := \text{res}(p, T_{\geq v})$. We have:*

(1) *the polynomial p is regular w.r.t. $\text{sat}(T)$ if and only if r is regular w.r.t. $\text{sat}(T_{<v})$;*

(2) *if $v \notin \text{mvar}(T)$ and $\text{init}(p)$ is regular w.r.t. $\text{sat}(T)$, then p is regular w.r.t. $\text{sat}(T)$.*

Proof. By Proposition 4.2, p is regular w.r.t. $\text{sat}(T)$ if and only if $\text{res}(p, T) \neq 0$, which is equivalent as $\text{res}(r, T_{<v}) \neq 0$, that is r is regular w.r.t. $\text{sat}(T_{<v})$. So (1) holds. Claim (2) is a consequence of the McCoy Theorem. We can also prove (2) directly. Since $\text{res}(\text{init}(p), T) = \text{res}(\text{init}(p), T_{<v})$, if $\text{init}(p)$ is regular w.r.t. $\text{sat}(T)$, then $\text{init}(p)$ is also regular w.r.t. $\text{sat}(T_{<v})$. We claim that p is regular w.r.t. $\text{sat}(T_{<v})$. Otherwise by Proposition 4.2, there is an associated prime ideal $\mathfrak{p}$ of $\text{sat}(T_{<v})$ such that $p \in \mathfrak{p}$, which implies that $\text{init}(p) \in \mathfrak{p}$, contradiction. Therefore p is regular w.r.t. $\text{sat}(T_{<v})$. On the other hand, $v \notin \text{mvar}(T)$, which implies that $p = r$ and therefore p is regular w.r.t. $\text{sat}(T)$. $\square$

4.3 The incremental algorithm

In this section, we present an algorithm to compute Lazard-Wu triangular decompositions in an incremental manner. We recall the concepts of a *process* and a *regular (delayed) split*, which were introduced as Definitions 9 and 11 in [104]. To serve our purpose, we modify the definitions as below.

Definition 4.1. *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 4.2. *Let T_i, $1 \leq i \leq e$, 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) 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) $V(p) \cap W(T) \subseteq \cup_{i=1}^{e} W(T_i)$

We write as $(p, T) \longrightarrow T_1, \ldots, T_e$. 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 we may compute a little more than $V(p) \cap W(T)$; however, $W(T_1) \cup \cdots \cup W(T_e)$ 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 chat $(p, T) \longrightarrow T_1, \ldots, T_e$.

$\mathsf{Regularize}(p, T)$

- **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 $i, 1 \leq i \leq e$: (1) T_i is a regular chain; (2) $p = p_i \bmod \sqrt{\mathrm{sat}(T_i)}$; (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 $T \longrightarrow T_1, \ldots, T_e$.

$\mathsf{SubresultantChain}(p, q, v)$

- **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.

$\mathsf{RegularGcd}(p, q, v, S, T)$

- **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.

$\mathsf{IntersectFree}(p, x_i, C)$

- **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 $(p, C) \longrightarrow (T_1, \ldots, T_e)$.

$\mathsf{IntersectAlgebraic}(p, T, x_i, S, C)$

- **Input:** p, a polynomial of R with main variable x_i,
  - 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: $\text{init}(T_{x_i})$ is regular modulo $\sqrt{\text{sat}(C)}$; the resultant of p and T_{x_i}, which is S_0, belongs to $\sqrt{\text{sat}(C)}$.

- **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 $\sqrt{\text{sat}(T_{<x_i})} \subseteq \sqrt{\text{sat}(C)}$.
- **Output:** if $x_i \notin \text{mvar}(T)$, return C; otherwise return a set of regular chains $\{T_1, \ldots, T_e\}$ such that $\text{init}(T_{x_i})$ is regular modulo each $\text{sat}(T_j)$, $\sqrt{\text{sat}(C)} \subseteq \sqrt{\text{sat}(T_j)}$ and $W(C) \setminus V(\text{init}(T_{x_i})) \subseteq \cup_{j=1}^{e} W(T_j)$.

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 $\sqrt{\text{sat}(T_{<x_i})} \subseteq \sqrt{\text{sat}(C)}$.
- **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 $\sqrt{\text{sat}(T)} \subseteq \sqrt{\text{sat}(T_j)}$.

Algorithm SubresultantChain is standard, see [56]. The algorithm Triangularize is a *principle algorithm* which was first presented in [104]. 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.

4.4 Proof of the algorithms

Theorem 4.1. *All the algorithms in Figure 4.1 terminate.*

Proof. The key observation is that the flow graph of Figure 4.1 can be transformed into an equivalent flow graph satisfying the following properties: (1) the algorithms Intersect and Regularize only call each other or themselves; (2) all the other algorithms only call either Intersect or Regularize. Therefore, it suffices to show that Intersect and Regularize terminate.

Note that the input of both functions is a process, say (p, T). One can check that, while executing a call with (p, T) as input, any subsequent call to either functions Intersect or Regularize will take a process (p', T') as input such that $(p', T') \prec (p, T)$ holds. Since a descending chain of processes is necessarily finite, both algorithms terminate. $\square$

Algorithm 1: Intersect(p, T)

if $\mathrm{prem}(p, T) = 0$ **then** return $\{T\}$;
if $p \in \mathbf{k}$ **then** return $\{\ \}$;
$r := p$; $P := \{r\}$; $S := \{\ \}$;
while $\mathrm{mvar}(r) \in \mathrm{mvar}(T)$ **do**
    $v := \mathrm{mvar}(r)$; $src :=$ SubresultantChain(r, T_v, v);
    $S := S \cup \{src\}$; $r :=$ resultant(src);
    if $r = 0$ **then** break;
    if $r \in \mathbf{k}$ **then** return $\{\ \}$;
    $P := P \cup \{r\}$
$\mathfrak{T} := \{\varnothing\}$; $\mathfrak{T}' := \{\ \}$; $i := 1$;
while $i \leq n$ **do**
    for $C \in \mathfrak{T}$ **do**
        if $x_i \notin \mathrm{mvar}(P)$ **and** $x_i \notin \mathrm{mvar}(T)$ **then**
            $\mathfrak{T}' := \mathfrak{T}' \cup$ CleanChain(C, T, x_{i+1})
        else if $x_i \notin \mathrm{mvar}(P)$ **then**
            $\mathfrak{T}' := \mathfrak{T}' \cup$ CleanChain$(C \cup T_{x_i}, T, x_{i+1})$
        else if $x_i \notin \mathrm{mvar}(T)$ **then**
            for $D \in$ IntersectFree(P_{x_i}, x_i, C) **do**
                $\mathfrak{T}' := \mathfrak{T}' \cup$ CleanChain(D, T, x_{i+1})
        else
            for $D \in$ IntersectAlgebraic$(P_{x_i}, T, x_i, S_{x_i}, C)$ **do**
                $\mathfrak{T}' := \mathfrak{T}' \cup$ CleanChain(D, T, x_{i+1})
    $\mathfrak{T} := \mathfrak{T}'$; $\mathfrak{T}' := \{\ \}$; $i := i + 1$
return $\mathfrak{T}$

Algorithm 2: RegularGcd(p, q, v, S, T)

$\mathfrak{T} := \{(T, 1)\}$;
while $\mathfrak{T} \neq \emptyset$ **do**
    let $(C, i) \in \mathfrak{T}$; $\mathfrak{T} := \mathfrak{T} \setminus \{(C, i)\}$;
    for $[f, D] \in$ Regularize(s_i, C) **do**
        if $\dim D < \dim C$ **then** output $[0, D]$;
        else if $f = 0$ **then** $\mathfrak{T} := \mathfrak{T} \cup \{(D, i+1)\}$;
        else output $[S_i, D]$

Algorithm 3: IntersectFree(p, x_i, C)

for $[f, D] \in$ Regularize$(\text{init}(p), C)$ **do**
  if $f = 0$ **then** output Intersect$(\text{tail}(p), D)$;
  else
    output $D \cup p$;
    for $E \in$ Intersect$(\text{init}(p), D)$ **do**
      output Intersect$(\text{tail}(p), E)$

Algorithm 4: IntersectAlgebraic(p, T, x_i, S, C)

for $[g, D] \in$ RegularGcd(p, T_{x_i}, x_i, S, C) **do**
  if $\dim D < \dim C$ **then**
    for $E \in$ CleanChain(D, T, x_i) **do**
      output IntersectAlgebraic(p, T, x_i, S, E)
  else
    output $D \cup g$;
    for $E \in$ Intersect$(\text{init}(g), D)$ **do**
      for $F \in$ CleanChain(E, T, x_i) **do**
        output IntersectAlgebraic(p, T, x_i, S, F)

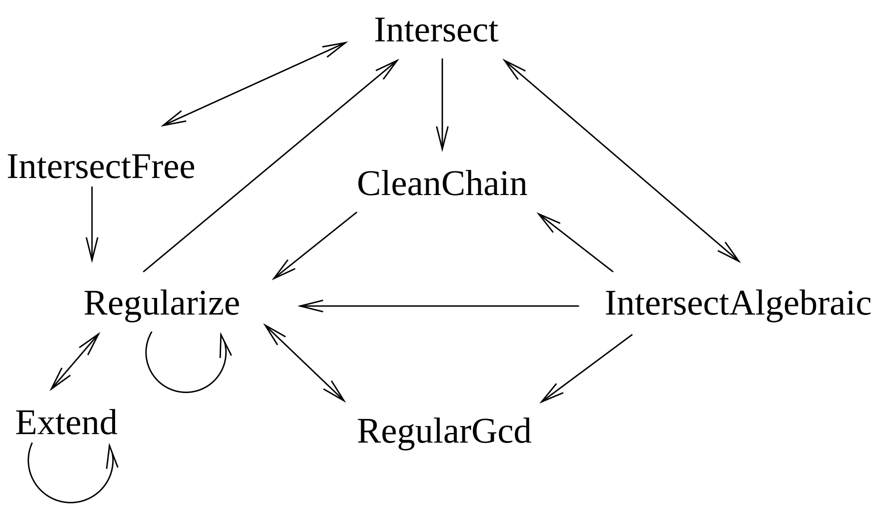


Figure 4.1: Flow graph of the Algorithms

Algorithm 5: Regularize(p, T)

```
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)
```

Algorithm 6: Extend(C, T, x_i)

```
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;
```

Algorithm 7: CleanChain(C, T, x_i)

if $x_i \notin \mathrm{mvar}(T)$ **or** $\dim C = \dim T_{<x_i}$ **then** return C;
for $[f, D] \in$ Regularize$(\mathrm{init}(T_{x_i}), C)$ **do**
  if $f \neq 0$ **then** output D

Algorithm 8: Triangularize(F)

if $F = \{\ \}$ **then** return $\{\varnothing\}$;
Choose a polynomial $p \in F$ with maximal rank;
for $T \in$ Triangularize$(F \setminus \{p\})$ **do**
  output Intersect(p, T)

Since all algorithms terminate, and following the flow graph of Figure 4.1, each call to one of our algorithms unfold to a finite dynamic acyclic graph (DAG) where each vertex is a call to one of our algorithms. Therefore, proving the correctness of these algorithms reduces to prove the following two points.

- *Base:* each algorithm call, which makes no subsequent calls to another algorithm or to itself, is correct.
- *Induction:* each algorithm call, which makes subsequent calls to another algorithm or to itself, is correct, as soon as all subsequent calls are themselves correct.

For all algorithms in Figure 4.1, proving the base cases is straightforward. Hence we focus on the induction steps.

Theorem 4.2. Triangularize *terminates and satisfies its specification.*

Proof. Its termination is obvious. It correctness can be proved by the following induction:

It holds clearly for the base case: $V(\{\ \}) = V(\varnothing) = \mathbf{K}^n$. Now we assume that the function call Triangularize$(F \setminus \{p\})$ returns a finite set of regular chains $T_1, \ldots, T_e$ such that $V(F \setminus \{p\}) = \cup_{i=1}^{e} W(T_i)$. By the specification of Intersect, for each T_i, there exists regular chains $T_{i,1}, \ldots, T_{i,i_s}$ s.t.

$$V(p) \cap W(T_i) \subseteq \bigcup_{j=1}^{i_s} W(T_{i,j}) \subseteq V(p) \cap \overline{W(T_i)}.$$

Therefore, we have

$$\begin{array}{rcl} V(F) & = & \bigcup_{i=1}^{e} V(p) \cap W(T_i) \subseteq \bigcup_{i=1}^{e} \bigcup_{j=1}^{i_s} W(T_{i,j}) \\ & \subseteq & \bigcup_{i=1}^{e} \left(V(p) \cap \overline{W(T_i)} \right) \subseteq \bigcup_{i=1}^{e} (V(p) \cap V(F \setminus \{p\})) \\ & = & V(F) \end{array}$$

That is $V(F) = \bigcup_{i=1}^{e} \bigcup_{j=1}^{i_s} W(T_{i,j})$, done. □

Proposition 4.5. IntersectFree *satisfies its specification.*

Proof. We have the following two key observations:

- $C \longrightarrow D_1, \ldots, D_s$, where D_i are the regular chains in the output of Regularize.
- $V(p) \cap W(D) = W(D, p) \cup V(\mathrm{init}(p), \mathrm{tail}(p)) \cap W(D)$.

Then it is not hard to conclude that $(p, C) \longrightarrow T_1, \ldots, T_e$. □

Proposition 4.6. IntersectAlgebraic *is correct.*

Proof. We need to prove: $(p, C \cup T_{x_i}) \longrightarrow T_1, \ldots, T_e$. Let us prove (L_1) now, that is, for each regular chain T_j in the output, we have $\sqrt{\mathrm{sat}(C \cup T_{x_i})} \subseteq \sqrt{\mathrm{sat}(T_j)}$. First by the specifications of the called functions, we have $\sqrt{\mathrm{sat}(C)} \subseteq \sqrt{\mathrm{sat}(D)} \subseteq \sqrt{\mathrm{sat}(E)}$, thus, $\sqrt{\mathrm{sat}(C \cup T_{x_i})} \subseteq \sqrt{\mathrm{sat}(E \cup T_{x_i})}$ by Corollary 4.2, since $\mathrm{init}(T_{x_i})$ is regular modulo both $\mathrm{sat}(C)$ and $\mathrm{sat}(E)$. Secondly, since g is a regular GCD of p and T_{x_i} modulo $\sqrt{\mathrm{sat}(D)}$, we have $\sqrt{\mathrm{sat}(C \cup T_{x_i})} \subseteq \sqrt{\mathrm{sat}(D \cup g)}$ by Corollaries 4.2 and Proposition 3.2.

Next we prove (L_2). It is enough to prove that $W(D \cup g) \subseteq V(p)$ holds. Since g is a regular GCD of p and T_{x_i} modulo $\sqrt{\mathrm{sat}(D)}$, the conclusion follows from point (iii) of Proposition 3.2.

Finally we prove (L_3), that is $Z(p, C \cup T_{x_i}) \subseteq \bigcup_{j=1}^{e} W(T_j)$. Let $D_1, \ldots, D_s$ be the regular chains returned from Algorithm RegularGcd. We have $C \longrightarrow D_1, \ldots, D_s$, which implies $Z(p, C \cup T_{x_i}) \subseteq \cup_{j=1}^{e} Z(p, D_j \cup T_{x_i})$. Next since g is a regular GCD of p and T_{x_i} modulo $\sqrt{\mathrm{sat}(D_j)}$, the conclusion follows from point (iv) of Proposition 3.2. □

Proposition 4.7. Intersect *satisfies its specification.*

Proof. The first while loop can be seen as a projection process. We claim that it produces a nonempty triangular set P such that $V(p) \cap W(T) = V(P) \cap W(T)$. The claim holds before staring the while loop. For each iteration, let P' be the set of polynomials obtained at the previous iteration. We then compute a polynomial r, which is the resultant of a polynomial in P' and a polynomial in T. So $r \in \langle P', T\rangle$. By induction, we have $\langle p, T\rangle = \langle P, T\rangle$. So the claim holds.

Next, we claim that the elements in $\mathfrak{T}$ satisfy the following invariants: at the beginning of the i-th iteration of the second while loop, we have

(1) each $C \in \mathfrak{T}$ is a regular chain; if T_{x_i} exists, then $\mathrm{init}(T_{x_i})$ is regular modulo $\mathrm{sat}(C)$,

(2) for each $C \in \mathfrak{T}$, we have $\sqrt{\mathrm{sat}(T_{<x_i})} \subseteq \sqrt{\mathrm{sat}(C)}$,

(3) for each $C \in \mathfrak{T}$, we have $\overline{W(C)} \subseteq V(P_{<x_i})$,

(4) $V(p) \cap W(T) \subseteq \bigcup_{C\in\mathfrak{T}} Z(P_{\geq x_i}, C \cup T_{\geq x_i})$.

When $i = n+1$, we then have $\sqrt{\mathrm{sat}(T)} \subseteq \sqrt{\mathrm{sat}(C)}$, $W(C) \subseteq V(P) \subseteq V(p)$ for each $C \in \mathfrak{T}$ and $V(p) \cap W(T) \subseteq \cup_{C\in\mathfrak{T}} W(C)$. So $(L_1), (L_2), (L_3)$ of Definition 4.2 all hold. This concludes the correctness of the algorithm.

Now we prove the above claims (1), (2), (3), (4) by induction. The claims clearly hold when $i = 1$ since $C = \varnothing$ and $V(p) \cap W(T) = V(P) \cap W(T)$. Now assume that the loop invariants hold at the beginning of the i-th iteration. We need to prove that it still holds at the beginning of the $(i+1)$-th iteration. Let $C \in \mathfrak{T}$ be an element picked up at the beginning of i-th iteration and let L be the set of the new elements of $\mathfrak{T}'$ generated from C.

Then for any $C' \in L$, claim (1) clearly holds by specification of CleanChain. Next we prove (2).

- if $x_i \notin \mathrm{mvar}(T)$, then $T_{<x_{i+1}} = T_{<x_i}$. By induction and specifications of called functions, we have

$$\sqrt{\mathrm{sat}(T_{<x_{i+1}})} \subseteq \sqrt{\mathrm{sat}(C)} \subseteq \sqrt{\mathrm{sat}(C')}.$$

- if $x_i \in \mathrm{mvar}(T)$, by induction we have $\sqrt{\mathrm{sat}(T_{<x_i})} \subseteq \sqrt{\mathrm{sat}(C)}$ and $\mathrm{init}(T_{x_i})$ is regular modulo both $\mathrm{sat}(C)$ and $\mathrm{sat}(T_{<x_i})$. By Corollary 4.2 we have

$$\sqrt{\mathrm{sat}(T_{<x_{i+1}})} \subseteq \sqrt{\mathrm{sat}(C \cup T_{x_i})} \subseteq \sqrt{\mathrm{sat}(C')}.$$

Therefore (2) holds. Next we prove claim (3). By induction and the specifications of called functions, we have $\overline{W(C')} \subseteq \overline{W(C \cup T_{x_i})} \subseteq V(P_{<x_i})$. Secondly, we have

$\overline{W(C')} \subseteq V(P_{x_i})$. Therefore $\overline{W(C')} \subseteq V(P_{<x_{i+1}})$, that is (3) holds. Finally, since $V(P_{x_i}) \cap W(C \cup T_{x_i}) \setminus V(\mathrm{init}(T_{x_{i+1}})) \subseteq \cup_{C' \in L} W(C')$, we have $Z(P_{\geq x_i}, C \cup T_{\geq x_i}) \subseteq \cup_{C' \in L} Z(P_{\geq x_{i+1}}, C' \cup T_{\geq x_{i+1}})$, which implies that (4) holds. This completes the proof. □

Proposition 4.8. Regularize *satisfies its specification.*

Proof. If $v \notin \mathrm{mvar}(T)$, the conclusion follows directly from point (2) of Corollary 4.3. From now on, assume $v \in \mathrm{mvar}(T)$. Let $\mathbf{L}$ be the set of pairs $[p', T']$ in the output. We aim to prove the following facts

(1) each T' is a regular chain,

(2) if $p' = 0$, then p is zero modulo $\sqrt{\mathrm{sat}(T')}$, otherwise p is regular modulo $\mathrm{sat}(T)$,

(3) we have $\sqrt{\mathrm{sat}(T)} \subseteq \sqrt{\mathrm{sat}(T')}$,

(4) we have $W(T) \subseteq \cup_{T' \in \mathbf{L}} W(T')$.

Statement (1) is due to Proposition 4.3. Next we prove (2). First, when there are recursive calls, the conclusion is obvious. Let $[f, C]$ be a pair in the output of Regularize$(r, T_{<v})$. If $f \neq 0$, the conclusion follows directly from point (1) of Corollary 4.3. Otherwise, let $[g, D]$ be a pair in the output of the algorithm RegularGcd(p, T_v, v, src, C). If $\mathrm{mdeg}(g) = \mathrm{mdeg}(T_v)$, then by the algorithm of RegularGcd, $g = T_v$. Therefore we have $\mathrm{prem}(p, T_v) \in \sqrt{\mathrm{sat}(C)}$, which implies that $p \in \sqrt{\mathrm{sat}(C \cup T_{\geq v})}$ by Proposition 4.4.

Next we prove (3). Whenever Extend is called, (3) holds immediately. Otherwise, let $[f, C]$ be a pair returned by Regularize$(r, T_{<v})$. When $f \neq 0$, since $\sqrt{\mathrm{sat}(T_{<v})} \subseteq \sqrt{\mathrm{sat}(C)}$ holds, we conclude $\sqrt{\mathrm{sat}(T)} \subseteq \sqrt{\mathrm{sat}(C \cup T_{\geq v})}$ by Corollary 4.2. Let $[g, D] \in$ RegularGcd(p, T_v, v, src, C). Corollary 4.2 and point (ii) of Proposition 3.2 imply that $\sqrt{\mathrm{sat}(T)} \subseteq \sqrt{\mathrm{sat}(D \cup T_{\geq v})}$, $\sqrt{\mathrm{sat}(T)} \subseteq \sqrt{\mathrm{sat}(D \cup g \cup T_{>v})}$ together with $\sqrt{\mathrm{sat}(T)} \subseteq \sqrt{\mathrm{sat}(D \cup q \cup T_{>v})}$ hold. Hence (3) holds.

Finally by point $(ii.b)$ of Proposition 3.2, we have $W(D \cup T_v) \subseteq Z(h_g, D \cup T_v) \cup W(D \cup g) \cup W(D \cup q)$. So (4) holds. □

Proposition 4.9. Extend *satisfies its specification.*

Proof. It clearly holds when $T_{\geq x_i} = \varnothing$, which is the base case. By induction and the specification of Regularize, we know that $\sqrt{\mathrm{sat}(T')} \subseteq \sqrt{\mathrm{sat}(E)}$. Since $\mathrm{init}(p)$ is regular modulo both $\mathrm{sat}(T')$ and $\mathrm{sat}(E)$, by Corollary 4.2, we have $\sqrt{\mathrm{sat}(T)} \subseteq \sqrt{\mathrm{sat}(E \cup p)}$. On the other hand, we have $W(C \cup T'_{\geq x_i}) \subseteq \cup W(D)$ and $W(D) \setminus V(h_p) \subseteq \cup\, W(E)$.

Therefore $W(C \cup T_{\geq x_i}) \subseteq \cup_{j=1}^{e} W(T_j)$, where $T_1, \ldots, T_e$ are the regular chains in the output. □

Proposition 4.10. CleanChain *satisfies its specification.*

Proof. It follows directly from Proposition 4.3. □

Proposition 4.11. RegularGcd *satisfies its specification.*

Proof. Let $[g_i, T_i]$, $i = 1, \ldots, e$, be the output. First from the specification of Regularize, we have $T \longrightarrow T_1, \ldots, T_e$. When $\dim T_i = \dim T$, by Proposition 4.3 and Theorem 3.3, g_i is a regular GCD of p and q modulo $\sqrt{\mathrm{sat}(T)}$. □

4.5 The recycling theorem

Theorem 3.3 in Section 3.3 indicates that for computing regular GCDs of two polynomials p and t modulo multiple regular chains one can re-use (or recycle) the subresultant chain of p and t (as soon as it is computed).

In this section, we present a result, that we call the *Recycling Theorem*, which extends this "subresultant chain re-using" strategy to the operation Intersect. In fact, this strategy is a fundamental property of Algorithm 1.

In broad terms, Theorem 4.3 states the following. Using the notations below, consider the subresultant chain S of the polynomials p and t. Then the intersection of the hypersurface $V(p)$ and the quasi-component $W(T \cup t)$ (in the sense of the operation Intersect) is obtained by computing regular GCDs of p and t modulo various regular chains. Moreover, the subresultant chain S can be recycled for each of these GCD computations. Therefore, being able in practice to recycle S for all those (thus avoiding recomputing S) is essential for performance issues.

Theorem 4.3 (Recycling Theorem)**.** *For* $1 \leq k \leq n$, *let* $T \subset \mathbf{k}[x_1, \ldots, x_{k-1}]$ *be a regular chain, possibly empty. Let* $p, t \in \mathbf{k}[x_1, \ldots, x_k]$ *be polynomials with main variable* x_k. *Assume* $T \cup \{t\}$ *is a regular chain. Then there exists 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* $\sqrt{\mathrm{sat}(T_i)}$.

Moreover, Algorithm Intersect$(p, T \cup t)$ *computes such regular chains.*

Proof. By the specification of Intersect, we have (i). Next we prove that (ii) and (iii) hold.

Firstly, if $\text{prem}(p, T \cup t) = 0$, then we have $\text{prem}(p, t) \in \text{sat}(T)$, which implies that $\text{prem}(p, t) \in \sqrt{\text{sat}(T)}$. Since $\text{prem}(t, t) = 0 \in \sqrt{\text{sat}(T)}$ and $\text{init}(t)$ is regular modulo $\text{sat}(T)$, we deduce that t is a regular GCD of p and t modulo $\sqrt{\text{sat}(T)}$. Thus the theorem holds.

Secondly, whenever $\text{Intersect}(p, T \cup t)$ returns an empty set of regular chains, the theorem obviously holds.

Thirdly, since $\text{mvar}(p) = \text{mvar}(t) = x_k < x_{k+1}$, all the regular chains in the output of Intersect are generated through IntersectAlgebraic at line 21 of algorithm Intersect. Then the conclusion follows from line 6 of IntersectAlgebraic and the specification of algorithm RegularGcd. ☐

4.6 Kalkbrener decomposition

In this section, we adapt the Algorithm Triangularize (Algorithm 8), in order to compute efficiently a Kalkbrener triangular decomposition. The basic technique we rely on follows from Krull's principle ideal theorem.

Theorem 4.4. *Let $F \subset \mathbf{k}[\mathbf{x}]$ be finite, with cardinality $\#(F)$. Assume F generates a proper ideal of $\mathbf{k}[\mathbf{x}]$. Then, for any minimal prime ideal $\mathfrak{p}$ associated with $\langle F \rangle$, the height of $\mathfrak{p}$ is less than or equal to $\#(F)$.*

Corollary 4.4. *Let $\mathfrak{T}$ be a Kalkbrener triangular decomposition of $V(F)$. Let T be a regular chain of $\mathfrak{T}$, the height of which is greater than $\#(F)$. Then $\mathfrak{T} \setminus \{T\}$ is also a Kalkbrener triangular decomposition of $V(F)$.*

Based on this corollary, we prune the decomposition tree generated during the computation of a Lazard-Wu triangular decomposition and remove the computation branches in which the height of every generated regular chain is greater than the number of polynomials in F.

Next we explain how to implement this tree pruning technique to the algorithms of Section 4.3. Inside Triangularize, define $A = \#(F)$ and pass it to every call to Intersect in order to signal Intersect to output only regular chains with height no greater than A. Next, in the second while loop of Intersect, for the i-th iteration, we pass the height $A - \#(T_{\geq x_{i+1}})$ to CleanChain, IntersectFree and IntersectAlgebraic.

In IntersectFree, we pass its input height A to every function call. Besides, Lines 5 to 6 are executed only if the height of D is strictly less than A, since otherwise we would obtain regular chains of height greater than A. In other algorithms, we apply similar strategies as in Intersect and IntersectFree.

4.7 Squarefree decomposition

Throughout this section, we assume that the coefficient field $\mathbf{k}$ is of characteristic zero. We propose two strategies for computing a squarefree triangular decomposition. The first one is a post-processing which applies Algorithm 11 to every regular chain returned by Algorithm 8. The second consists of ensuring that, each output or intermediate regular chain generated during the execution of Algorithm 8 is squarefree.

To implement the second strategy, we add an *squarefree option* to Algorithm 8 and each of its subalgorithms. If the option is set to *true*, this option requires that each output regular chain is squarefree. This is achieved by using Algorithm 9 whenever we need to construct new regular chains from a previous regular chain T and a polynomial p such that $T \cup p$ is known to be a regular chain.

Algorithm 9: Squarefree(p, x_i, T)

Input: a polynomial ring $R = \mathbf{k}[x_1, \ldots, x_n]$, a variable x_i of R, a squarefree regular chain T of $\mathbf{k}[x_1, \ldots, x_{i-1}]$, a polynomial p of R with main variable x_i such that $T \cup p$ is a regular chain.

Output: a set of squarefree regular chains $T_1, \ldots, T_e$ such that $p \cup T \longrightarrow T_1, \ldots, T_e$.

```
p := SquarefreePart(p);
if mdeg(p) = 1 then return T ∪ p;
else
    src := SubresultantChain(p, der(p), x_i);
    return Squarefree(p, x_i, src, T);
```

4.8 Experimentation

Part of the algorithms presented in this paper are implemented in MAPLE14 while all of them are present in the current development version of MAPLE. Tables 4.1 and 4.2 report on our comparison between Triangularize and other MAPLE solvers. The notations used in these tables are defined below.

Notation for Triangularize. We denote by TK and TL the latest implementation of Triangularize for computing, respectively, Kalkbrener and Lazard-Wu decompositions,

Algorithm 10: Squarefree(p, x_i, src, T)

Input: a polynomial ring $R = \mathbf{k}[x_1, \ldots, x_n]$, a variable x_i of R, a squarefree regular chain T of $\mathbf{k}[x_1, \ldots, x_{i-1}]$, a squarefree polynomial p of R with main variable x_i such that $T \cup p$ is a regular chain, the sub-resultant chain src of p and $\mathrm{der}(p)$ w.r.t x_i.

Output: a set of squarefree regular chains $T_1, \ldots, T_e$ such that $p \cup T \longrightarrow T_1, \ldots, T_e$.

$r :=$ resultant(src);
$\mathfrak{T} := \{\ \}$;
for $[f, C] \in$ Regularize(r, T) **do**
    if $f \neq 0$ **then** output $C \cup p$; next;
    else
        if $\dim C = \dim T$ **then**
            $\mathfrak{T} := \mathfrak{T} \cup \{C\}$; next;
        else
            for $[g, D] \in$ Regularize$(\mathrm{init}(p), C)$ **do**
                if $g \neq 0$ **then** $\mathfrak{T} := \mathfrak{T} \cup \{D\}$;
while $\mathfrak{T} \neq \{\ \}$ **do**
    let $C \in \mathfrak{T}$; $\mathfrak{T} := \mathfrak{T} \setminus \{C\}$;
    for $[g, D] \in$ RegularGcd$(p, \mathrm{der}(p), x_i, src, C)$ **do**
        if $\dim D = \dim C$ **then**
            output $D \cup \mathrm{pquo}(p, g)$;
            for $E \in$ Intersect$(\mathrm{init}(g), D)$ **do**
                for $[f, F] \in$ Regularize$(\mathrm{init}(p), E)$ **do**
                    if $f \neq 0$ **then** $\mathfrak{T} := \mathfrak{T} \cup \{F\}$;
        else
            for $[f, E] \in$ Regularize$(\mathrm{init}(p), D)$ **do**
                if $f \neq 0$ **then** $\mathfrak{T} := \mathfrak{T} \cup \{E\}$;

Algorithm 11: Squarefree(T)

Input: a polynomial ring $R = \mathbf{k}[x_1, \ldots, x_n]$, a regular chain T of R.
Output: a set of squarefree regular chains $T_1, \ldots, T_e$ such that $T \longrightarrow T_1, \ldots, T_e$.

$T :=$ {SquarefreePart(p) $\mid p \in T$};
$S := \{\ \}$;
for $p \in T$ **do**
  if $\mathrm{mdeg}(p) > 1$ **then**
    $S := S \cup$ {SubresultantChain($p, \mathrm{der}(p), \mathrm{mvar}(p), R$)};
$\mathfrak{T} := \{\varnothing\}$; $\mathfrak{T}' := \{\ \}$; $i := 1$;
while $i \leq n$ **do**
  for $C \in \mathfrak{T}$ **do**
    if $x_i \notin \mathrm{mvar}(T)$ **then**
      $\mathfrak{T}' := \mathfrak{T}' \cup$ CleanChain(C, T, x_{i+1})
    else
      if $\mathrm{mdeg}(T_{x_i}) = 1$ **then**
        $\mathfrak{T}' := \mathfrak{T}' \cup$ CleanChain($C \cup \{T_{x_i}\}, T, x_{i+1}$)
      else
        for $D \in$ Squarefree(T_{x_i}, x_i, S_{x_i}, C) **do**
          $\mathfrak{T}' := \mathfrak{T}' \cup$ CleanChain(D, T, x_{i+1})
  $\mathfrak{T} := \mathfrak{T}'$; $\mathfrak{T}' := \{\ \}$; $i := i + 1$;
return $\mathfrak{T}$

in the current version of Maple. Denote by TK14 and TL14 the corresponding implementation in Maple14. Denote by TK13, TL13 the implementation based on the algorithm of [104] in Maple13. Finally, STK and STL are versions of TK and TL respectively, enforcing that all computed regular chains are squarefree, by means of the algorithms in Section 4.7.

Notation for the other solvers. Denote by GL, GS, GD, respectively the function Groebner:-Basis (plex order), Groebner:-Solve, Groebner:-Basis (tdeg order) in current beta version of Maple. Denote by WS the function wsolve of the package `Wsolve` [123], which decomposes a variety as a union of quasi-components of Wu Characteristic Sets.

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 seconds. The memory usage is limited to 60% of total memory. In both Table 4.1 and 4.2, 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 [48]. All examples are in characteristic zero.

In Table 4.1, 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 Groebner functions. The MAPLE command we use are length(convert(map(Equations, dec, R), string)) and length(convert(gb, string)). From Table 4.1, it is clear that Triangularize produces much smaller output than commands based on Gröbner basis computations.

	sys	Input size				Output size				
		#v	#e	deg	dim	GL	GS	GD	TL	TK
1	4corps-1parameter-homog	4	3	8	1	-	-	21863	-	30738
2	8-3-config-Li	12	7	2	7	67965	-	72698	7538	1384
3	Alonso-Li	7	4	4	3	1270	-	614	2050	374
4	Bezier	5	3	6	2	-	-	32054	-	114109
5	Cheaters-homotopy-1	7	3	7	4	26387452	-	17297	-	285
7	childDraw-2	10	10	2	0	938846	-	157765	-	-
8	Cinquin-Demongeot-3-3	4	3	4	1	1652062	-	680	2065	895
9	Cinquin-Demongeot-3-4	4	3	5	1	-	-	690	-	2322
10	collins-jsc02	5	4	3	1	-	-	28720	2770	1290
11	f-744	12	12	3	1	102082	-	83559	4509	4510
12	Haas5	4	2	10	2	-	-	28	-	548
14	Lichtblau	3	2	11	1	6600095	-	224647	110332	5243
16	Liu-Lorenz	5	4	2	1	47688	123965	712	2339	938
17	Mehta2	11	8	3	3	-	-	1374931	5347	5097
18	Mehta3	13	10	3	3	-	-	-	25951	25537
19	Mehta4	15	12	3	3	-	-	-	71675	71239
21	p3p-isosceles	7	3	3	4	56701	-	1453	9253	840
22	p3p	8	3	3	5	160567	-	1768	-	1712
23	Pavelle	8	4	2	4	17990	-	1552	3351	1086
24	Solotareff-4b	5	4	3	1	2903124	-	14810	2438	872
25	Wang93	5	4	3	1	2772	56383	1377	1016	391
26	Xia	6	3	4	3	63083	2711	672	1647	441
27	xy-5-7-2	6	3	3	3	12750	-	599	-	3267

Table 4.1: The input and output sizes of systems

TK, TL, 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 TK are close to those of GS while TL is related to WS, though the triangular sets returned by WS are not necessarily regular chains.

In Table 4.2, we provide the timings of different versions of Triangularize and other solvers. From this table, it is clear that the implementations of Triangularize, based on the algorithms presented in this paper (that is TK14, TL14, TK, TL) outperform the previous versions (TK13, TL13), based on [104], by several orders of magnitude. We observe also that TK outperforms GS and GL while TL outperforms WS.

sys	Triangularize								Triangularize versus other solvers				
	TK13	TK14	TK	TL13	TL14	TL	STK	STL	GL	GS	WS	TL	TK
1	-	241.7	36.9	-	-	-	62.8	-	-	-	-	-	36.9
2	8.7	5.3	5.9	29.7	24.1	25.8	6.0	26.6	108.7	-	27.8	25.8	5.9
3	0.3	0.3	0.4	14.0	2.4	2.1	0.4	2.2	3.4	-	7.9	2.1	0.4
4	-	-	88.2	-	-	-	-	-	-	-	-	-	88.2
5	0.4	0.5	0.7	-	-	-	451.8	-	2609.5	-	-	-	0.7
7	-	-	-	-	-	-	1326.8	1437.1	19.3	-	-	-	-
8	3.2	0.7	0.6	-	55.9	7.1	0.7	8.8	63.6	-	-	7.1	0.6
9	166.1	5.0	3.1	-	-	-	3.3	-	-	-	-	-	3.1
10	5.8	0.4	0.4	-	1.5	1.5	0.4	1.5	-	-	0.8	1.5	0.4
11	-	29.1	12.7	-	27.7	14.8	12.9	15.1	30.8	-	-	14.8	12.7
12	452.3	454.1	0.3	-	-	-	0.3	-	-	-	-	-	0.3
14	0.7	0.7	0.3	801.7	226.5	143.5	0.3	531.3	125.9	-	-	143.5	0.3
16	0.4	0.4	0.4	4.7	2.6	2.3	0.4	4.4	3.2	2160.1	40.2	2.3	0.4
17	-	2.1	2.2	-	4.5	4.5	2.2	6.2	-	-	5.7	4.5	2.2
18	-	15.6	14.4	-	126.2	51.1	14.5	63.1	-	-	-	51.1	14.4
19	-	871.1	859.4	-	1987.5	1756.3	859.2	1761.8	-	-	-	1756.3	859.4
21	1.2	0.6	0.3	-	1303.1	352.5	0.3	-	6.2	-	792.8	352.5	0.3
22	168.8	5.5	0.3	-	-	-	0.3	-	33.6	-	-	-	0.3
23	0.8	0.9	0.5	-	10.3	7.0	0.4	12.6	1.8	-	-	7.0	0.5
24	1.5	0.7	0.8	-	1.9	1.9	0.9	2.0	35.2	-	9.1	1.9	0.8
25	0.5	0.6	0.7	0.6	0.8	0.8	0.8	0.9	0.2	1580.0	0.8	0.8	0.7
26	0.2	0.3	0.4	4.0	1.9	1.9	0.5	2.7	4.7	0.1	12.5	1.9	0.4
27	3.3	0.9	0.6	-	-	-	0.7	-	0.3	-	-	-	0.6

Table 4.2: Timings of Triangularize versus other solvers

4.9 Extra operations

In this section, we present some operations, which are not the core routines in the incremental triangular decomposition algorithm, but are very useful due to its specifications. Some of them are used as subroutines of algorithms in other chapters. The termination and correctness of these algorithms can be proved by similar arguments used in Section 4.4. We denote by R the polynomial ring $\mathbf{k}[\mathbf{x}]$, where $\mathbf{x} = x_1 < \cdots < x_n$. The specifications of the algorithms are as follows.

$\mathsf{Triangularize}(F, T)$

- **Input:** F, a finite polynomial set of R; T, a regular chain of R.
- **Output:** a set of regular chains $T_1, \ldots, T_e$ such that we have $V(F) \cap W(T) \subseteq \cup_{i=1}^{e} W(T_i) \subseteq V(F) \cap \overline{W(T)}$ holds.

$\mathsf{RegularOnly}(T, H)$

- **Input:** T, a regular chain of R; H, a finite polynomial set of R.

- **Output:** a set of regular chains $T_1, \ldots, T_e$ such that we have $Z(T, H) = \cup_{i=1}^{e} Z(T_i, H)$ and all polynomials in H are regular modulo $\mathrm{sat}(T_i)$, for $i = 1, \ldots, e$.

StrongRegularize(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 $i, 1 \leq i \leq e$: T_i is a regular chain; $p = p_i \bmod \mathrm{sat}(T_i)$; if $p_i = 0$, then $p_i \in \mathrm{sat}(T_i)$ and otherwise p_i is regular modulo $\mathrm{sat}(T_i)$; moreover we have $T \longrightarrow T_1, \ldots, T_e$.

StrongRegularGcd(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 w.r.t $\mathrm{sat}(T)$; the resultant of p and q w.r.t v belongs to $\mathrm{sat}(T)$
  - 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 $\mathrm{sat}(T_i)$; otherwise $g_i = 0$, which means undefined.

GCD(p, q, v, 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 w.r.t $\mathrm{sat}(T)$
- **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.

Now we describe the above algorithms. Firstly, if in the algorithm Regularize, we replace everywhere Regularize by StrongRegularize and RegularGcd by StrongRegularGcd, we then obtain an implementation of the algorithm StrongRegularize.

Algorithm 12: StrongRegularGcd(p, q, v, S, T)

```
for [g, C] ∈ RegularGcd(p, q, v, S) do
    if dim C = dim T then
        // prem(p, g) and prem(q, g) belongs to √sat(C) now
        for D ∈ StrongRegularize(prem(p, g), C) do
            for E ∈ StrongRegularize(prem(q, g), D) do
                if dim E = dim T then
                    output [g, E]
                else
                    output [0, E]
    else
        output [g_i, T_i]
```

Algorithm 13: GCD(p, q, v, T)

```
src := SubresultantChain(p, q, v); r := resultant(src)
for [f, C] ∈ Regularize(r, T) do
    if dim C < dim T then
        output [0, C]
    else if f ≠ 0 then
        output [r, C]
    else
        output RegularGcd(p, q, v, src, C)
```

Algorithm 14: Triangularize(F, T)

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

Algorithm 15: Regularize(T, H)

```
if H = { } then return {T}
for [f, C] ∈ Regularize(∏_{h∈H} h, T) do
    if f ≠ 0 then
        output C
```

Chapter 5

Set-theoretic Operations on Constructible Sets

Polynomial systems arising from applications often involve inequations, which typically exclude degenerated configurations. The solution set of a system of polynomial equations, say $f(\mathbf{x}) = 0$, and inequations, say $h(\mathbf{x}) \neq 0$, is called a *constructible set*. This chapter introduces the concept of a *regular system* which extends the notion of regular chains (used in Chapter 4 for encoding algebraic varieties) so as to represent constructible sets. Based on this representation, we present highly efficient algorithms for computing the set-theoretic difference of two constructible sets and apply it to verifying polynomial system solvers implementing triangular decompositions.

5.1 Introduction

Constructible sets, which are solution sets of polynomial systems involving equations and inequations, arise naturally in applications. For example in Chapter 1, the solution set of $\mathcal{C}_1 := \{p_1 = 0, p_2 = 0, k_2 \neq 0\}$ in $\mathbb{C}^3$ is a constructible set. Constructible sets are also generated naturally in triangular decomposition. Indeed, for a regular chain T of $\mathbf{k}[x_1, \ldots, x_n]$, its quasi-components $W(T)$ is the set $V(T) \setminus V(h_T)$, which is again a constructible set. A formal definition of a constructible set is given in Section 5.2.

Given a polynomial system Σ of $\mathbf{k}[x_1, \ldots, x_n]$, the first question one may want to answer is whether the constructible set *cs* defined by Σ is empty or not in $\mathbf{K}^n$. To this end, we introduce the concept of a *regular system* and prove that any constructible set decomposes as the union of the zero sets of finitely many regular systems. Then, testing the emptiness of cs reduces to checking whether it decomposes into an empty set of regular systems.

A regular system of $\mathbf{k}[x_1, \ldots, x_n]$ is a pair $[T, H]$, where T is a regular chain and H is a set of polynomials each of which is regular modulo $\mathrm{sat}(T)$. The name of regular system first appears in the paper [126] with much stronger properties than those that we impose. The motivation of our definition is to mimic the role that regular chains play for algebraic varieties.

Algorithms computing triangular decompositions of algebraic varieties, and more generally constructible sets, do not produce canonical output. In fact, due to different implementation choices, two implementations of the same triangular decomposition algorithm may produce different output (both valid if both implementations are correct) for the same input polynomial system. Deciding whether these two output decompositions represent the same constructible set is a fundamental verification problem for polynomial system solvers. In Section 5.5, we discuss this verification problem in great detail. If we assume that both outputs are represented by regular systems, the question boils down to computing the difference of the zero sets of two regular systems. In Section 5.4, we provide a highly efficient algorithm to do this task. The basic idea there is to exploit the structural properties of regular systems and extract their common zeros by performing GCD computations.

This chapter is based on paper [35] and its enhanced version [32], co-authored with Marc Moreno Maza, Wei Pan and Yuzhen Xie.

5.2 Representation of constructible sets

Definition 5.1 (Constructible set). *Let $F = \{f_1, \ldots, f_s\}$ and $H = \{h_1, \ldots, h_\ell\}$ be two sets of polynomials in $\mathbf{k}[\mathbf{x}]$. We call the conjunction of the following constraints $f_1 = 0, \ldots, f_s = 0$ and $h_1 \neq 0, \ldots, h_\ell \neq 0$ a* constructible system *in $\mathbf{k}[\mathbf{x}]$, denoted by $[F, H]$. Its zero set in $\mathbf{K}^n$ is called a* basic constructible set *of $\mathbf{k}[\mathbf{x}]$. A* constructible set *of $\mathbf{k}[\mathbf{x}]$ is a finite union of basic constructible sets of $\mathbf{k}[\mathbf{x}]$.*

Definition 5.2. *Let T be a regular chain and H be a set of polynomials in $\mathbf{k}[\mathbf{x}]$. If every polynomial in H is regular modulo $\mathrm{sat}(T)$, we call $[T, H]$ a* regular system. *If H consists of a single polynomial h, we simply write $[T, H]$ as $[T, h]$. It is easy to prove that $[T, H]$ is a regular system if and only if $[T, \prod_{f \in H} h]$ is a regular system. The rank of $[T, H]$, denoted by $\mathrm{rank}([T, H])$, is defined as $\mathrm{rank}(T)$. For a finite set $\mathcal{R}$ of regular systems, we define $\mathrm{rank}(\mathcal{R}) := \max\{\mathrm{rank}(R) \mid R \in \mathcal{R}\}$.*

Proposition 5.1. *For every regular system* $[T, h]$ *we have* $Z(T, h) \neq \emptyset$.

Proof. Since T is a regular chain, by Lemma 2.2 we have $V(\text{sat}(T)) \neq \emptyset$. By definition of a regular system, the polynomial hh_T is regular modulo $\text{sat}(T)$. Hence, by Lemma A.1, the set $V(hh_T) \cap V(\text{sat}(T))$ either is empty, or has lower dimension than $V(\text{sat}(T))$. Therefore, the set $V(\text{sat}(T)) \setminus V(hh_T) = V(\text{sat}(T)) \setminus (V(hh_T) \cap V(\text{sat}(T)))$ is not empty. Finally, by Corollary 2.1, the set

$$Z(T, h) = W(T) \setminus V(h) = \overline{W(T)} \setminus V(hh_T) = V(\text{sat}(T)) \setminus V(hh_T)$$

is not empty. □

Lemma 5.1. *Let* T *be a regular chain and* f *be a polynomial in* $\mathbf{k}[\mathbf{x}]$. *Then there exists finitely many regular systems* $[T_1, h_1], \ldots, [T_e, h_e]$ *in* $\mathbf{k}[\mathbf{x}]$ *such that* $Z(T, f) = \cup_{i=1} Z(T_i, h_i)$.

Proof. Let h_T be the initial of T and $h := fh_T$. Let $T_1, \ldots, T_s$ be the regular chains in the output of $\mathsf{Regularize}(T, h)$. Then we have $W(T) \subseteq \cup_{i=1}^{s} W(T_i) \subseteq \overline{W(T)}$, which implies that $Z(T, h) = \cup_{i=1}^{s} Z(T_i, h)$, thanks to Corollary 2.1. Moreover, by specification of $\mathsf{Regularize}$, h is either regular or zero modulo $\sqrt{\text{sat}(T_i)}$. W.l.o.g., we assume that there exists an integer e such that h is regular modulo T_i for $1 \leq i \leq e$ and zero modulo T_i for $e+1 \leq i \leq s$. If h is zero modulo $\text{sat}(T_i)$, then we have $\overline{W(T_i)} \subseteq V(h)$, which implies that $Z(T_i, h) = \emptyset$. Therefore we have $Z(T, h) = \cup_{i=1}^{s} Z(T_i, h)$, where each $[T_i, h]$ is a regular system. This completes the proof. □

Lemma 5.2. *Let* cs *be a constructible set of* $\mathbf{K}^n$ *defined by a constructible system of* $\mathbf{k}[\mathbf{x}]$. *Then, there exists finitely many regular systems* $[T_i, h_i]$ *of* $\mathbf{k}[\mathbf{x}]$, *with* $i = 1, \ldots, e$, *such that* $cs = \cup_{i=1}^{e} Z(T_i, h_i)$. *We call* $([T_i, h_i], i = 1, \ldots, e)$ *a* triangular decomposition *of* cs.

Proof. Since a constructible set of $\mathbf{k}[\mathbf{x}]$ is a finite union of basic constructible sets of $\mathbf{k}[\mathbf{x}]$, there exists finitely many polynomial sets E_i and polynomials f_i in $\mathbf{k}[\mathbf{x}]$ such that $cs = \cup_i (V(E_i) \setminus V(f_i))$. By applying $\mathsf{Triangularize}$ to each E_i, we obtain finitely many regular chains $T_{i,1}, \ldots, T_{i,e_i}$ in $\mathbf{k}[\mathbf{x}]$ such that we have $cs = \cup_i \cup_j Z(T_{i,j}, f_i)$. The conclusion follows from Lemma 5.1. □

5.3 A straightforward Difference algorithm

In this section, we give a naive method to realize the Difference algorithm for computing the set-theoretic difference of the zero sets of two regular systems. We first state a technical lemma.

Lemma 5.3. *Let p and h be polynomials and T be a regular chain of $\mathbf{k}[\mathbf{x}]$. Then there exists an operation* $\mathsf{Intersect}(p, T, h)$ *returning a set of regular chains $\{T_1, \ldots, T_e\}$ such that*

(*i*) *h is regular w.r.t.* $\mathrm{sat}(T_i)$ *for all i;*

(*ii*) *$T_i \prec T$, if $p \notin \mathrm{sat}(T)$;*

(*iii*) *$Z(p, T, h) \subseteq \cup_{i=1}^{e} Z(T_i, h) \subseteq (V(p) \cap \overline{W(T)}) \setminus V(h)$;*

(*iv*) *Moreover, if the product h_T of the initials of T divides h, then*

$$Z(p, T, h) = \bigcup_{i=1}^{e} Z(T_i, h)$$

holds.

Proof. Define

$$\mathcal{D} = \bigcup_{C \in \mathsf{Intersect}(p,T)} \{[f, D] \mid [f, D] \in \mathsf{Regularize}(h, C)\}$$

We then have

$$V(p) \cap W(T) \subseteq \bigcup_{[f,D] \in \mathcal{D}} D \subseteq V(p) \cap \overline{W(T)}.$$

Rename the regular chains $\{D \mid [f, D] \in \mathcal{D},\ f \neq 0\}$ as $\{T_1, \ldots, T_e\}$. We then have

$$Z(p, T, h) \subseteq \bigcup_{i=1}^{e} Z(T_i, h) \subseteq (V(p) \cap \overline{W(T)}) \setminus V(h).$$

Therefore (*i*) and (*iii*) hold. Since $p \notin \mathrm{sat}(T)$, by the specification of Intersect, (*ii*) holds. Finally, Corollary 2.1 implies (*iv*). □

For two regular systems $[T_1, h_1]$ and $[T_2, h_2]$, the following formula,

$$
\begin{aligned}
Z(T_1, h_1) \setminus Z(T_2, h_2) &= \Big(Z(T_1, h_1) \bigcap V(T_2)^c\Big) \bigcup \Big(Z(T_1, h_1) \bigcap V(h_2 h_{T_2})\Big) \\
&= \underbrace{\Big(\bigcup_{f \in T_2} Z(T_1, h_1) \setminus V(f)\Big)}_{\text{Task A}} \bigcup \underbrace{\Big(Z(T_1, h_1) \bigcap V(h_2 h_{T_2})\Big)}_{\text{Task B}} \qquad (5.1)
\end{aligned}
$$

provides a method to compute the difference of the zero sets of two regular systems. Indeed, Task A is achieved by calling $\mathsf{Intersect}(0, T_1, f h_1 h_{T_1})$ for each polynomial $f \in T_2$ and Task B is achieved by calling $\mathsf{Intersect}(h_2 h_{T_2}, T_1, h_1 h_{T_1})$. However, this method completely ignores the structure of $[T_2, h_2]$ (a regular system).

In the next section, we provide an algorithm which exploits the structure of $[T_2, h_2]$. In broad words, the procedure proceeds as follows.

(1) If $\mathrm{sat}(T_1) = \mathrm{sat}(T_2)$ holds, computations reduce to elementary manipulations of zero sets.

(2) Otherwise, let v be the largest variable such that $\mathrm{sat}(T_{1,<v}) = \mathrm{sat}(T_{2,<v})$ holds. Let G be a regular GCD of $T_{1,v}$ and $T_{2,v}$ modulo $\sqrt{\mathrm{sat}(T_{1,<v})}$. If G is not constant and has main variable v, computations split into cases where either one can conclude easily or where a recursive call to the procedure can be made. If G is constant and or has main variable less than v, one can also easily conclude.

5.4 An efficient Difference algorithm

In this section, we present an algorithm to compute the set-theoretic difference of two constructible sets given by regular systems. As mentioned in the last section, a naive approach appears to be very inefficient in practice. Here we contribute a more sophisticated algorithm, which carefully exploits the structure and properties of regular chains.

Two procedures, Difference and DifferenceLR, are involved in order to achieve this goal. Their specifications and pseudo-codes can be found below. The rest of this section is dedicated to proving the correctness and termination of these algorithms.

Algorithm 1 Difference($[T, h], [T', h']$)

Input Two regular systems $[T, h]$ and $[T', h']$.

Output A family of regular systems $[T_i, h_i]$, $i = 1, \ldots, e$, such that: (i) $Z(T, h) \setminus Z(T', h') = \bigcup_{i=1}^{e} Z(T_i, h_i)$; (ii) $\mathrm{rank}([T_i, h_i]) \le \mathrm{rank}([T, h])$; (iii) there are at most one $[T_i, h_i]$ with the same rank as $[T, h]$.

Algorithm 2 DifferenceLR($\mathcal{L}, \mathcal{R}$)

Input Two lists of regular systems $\mathcal{L} := \{[L_i, f_i] \mid i = 1 \cdots r\}$ and $\mathcal{R} := \{[R_j, g_j] \mid j = 1 \ldots s\}$.

Output A family $\mathcal{S}$ of regular systems $[T_i, h_i]$, $i = 1, \ldots, e$, such that

$$\left(\bigcup_{i=1}^{r} Z(L_i, f_i) \right) \setminus \left(\bigcup_{j=1}^{s} Z(R_j, g_j) \right) = \bigcup_{i=1}^{e} Z(T_i, h_i),$$

$\mathrm{rank}(\mathcal{S}) \le \mathrm{rank}(\mathcal{L})$, and the number of regular systems in $\mathcal{S}$ with $\mathrm{rank}(\mathcal{L})$ is no greater than the number of regular systems in $\mathcal{L}$ with $\mathrm{rank}(\mathcal{L})$.

To prove the termination and correctness of above two algorithms, we present a series of technical lemmas.

Lemma 5.4. *Let $[T, h]$ and $[T', h']$ be two regular systems. If $\mathrm{sat}(T) = \mathrm{sat}(T')$, then $h' h_{T'}$ is regular w.r.t. $\mathrm{sat}(T)$ and*

$$Z(T, h) \setminus Z(T', h') = Z(h' h_{T'}, T, h) \text{ and } Z(T, h) \cap Z(T', h') = Z(T, h h' h_{T'}).$$

Proof. Since $\mathrm{sat}(T) = \mathrm{sat}(T')$ and $h' h_{T'}$ is regular w.r.t. $\mathrm{sat}(T')$, $h' h_{T'}$ is regular w.r.t. $\mathrm{sat}(T)$. By Lemma 2.1 and Lemma 2.2, we have $Z(T, h h' h_{T'}) = Z(T', h h' h_T)$. Note that we can decompose $Z(T, h)$ into the disjoint union

$$Z(T, h) = Z(T, h h' h_{T'}) \uplus Z(h' h_{T'}, T, h).$$

Similarly, we have $Z(T', h') = Z(T', h h' h_T) \uplus Z(h h_T, T', h')$. Hence, the conclusion holds. □

Lemma 5.5. *Assume that $\mathrm{sat}(T_{<v}) = \mathrm{sat}(T'_{<v})$. We have*

(i) *If $p' := T'_v$ is defined but not T_v, then the following properties hold:*

$(i.a)$ *p' is regular w.r.t. $\mathrm{sat}(T)$,*

$(i.b)$ $Z(T, h) \setminus Z(T', h') = Z(T, hp') \uplus \ (Z(p', T, h) \setminus Z(T', h'))$,

Algorithm 16: Difference$([T,h],[T',h'])$

```
begin
    if sat(T) = sat(T') then
        output Intersect(h' h_{T'}, T, h h_T)
    else
        Let v be the largest variable s.t. sat(T_{<v}) = sat(T'_{<v})
        if v ∈ mvar(T') and v ∉ mvar(T) then
            p' ← T'_v
            output [T, h p']
            output DifferenceLR(Intersect(p', T, h h_T), [T', h'])
        else if v ∉ mvar(T') and v ∈ mvar(T) then
            p ← T_v
            output DifferenceLR([T, h], Intersect(p, T', h' h_{T'}))
        else
            p ← T_v
            𝒢 ← GCD(T_v, T'_v, v, T_{<v})
            if |𝒢| = 1 then
                Let [g, C] ∈ 𝒢
                if g ∈ k then  output [T, h]
                else if mvar(g) < v then
                    output [T, gh]
                    output DifferenceLR(Intersect(g, T, h h_T), [T', h'])
                else if mvar(g) = v then
                    if mdeg(g) = mdeg(p) then
                        D'_p ← T'_{<v} ∪ {p} ∪ T'_{>v}
                        output Difference([T, h], [D'_p, h' h_{T'}])
                    else if mdeg(g) < mdeg(p) then
                        q ← pquo(p, g, v)
                        D_g ← T_{<v} ∪ {g} ∪ T_{>v}
                        D_q ← T_{<v} ∪ {q} ∪ T_{>v}
                        output Difference([D_g, h h_T], [T', h'])
                        output Difference([D_q, h h_T], [T', h'])
                        output DifferenceLR(Intersect(h_g, T, h h_T), [T', h'])
            else if |𝒢| ≥ 2 then
                for [g, C] ∈ 𝒢 do
                    if |C| > |T_{<v}| then
                        for D ∈ Extend(C, T, v) do
                            for [f, E] ∈ Regularize(h h_T, D) do
                                if f ≠ 0 then  output Difference([E, h h_T], [T', h'])
                    else  output Difference([C ∪ T_{≥v}, h h_T], [T', h'])
end
```

Algorithm 17: DifferenceLR(L, R)

```
begin
    if L = ∅ then
        output ∅
    else if R = ∅ then
        output L
    else
        while R ≠ ∅ do
            Let [T′, h′] ∈ R, R ← R \ { [T′, h′] }
            S ← ∅
            for [T, h] ∈ L do
                S ← S ∪ Difference([T, h], [T′, h′])
            L ← S
        output L
end
```

$(i.c)$ $Z(T,h) \cap Z(T',h') = Z(p',T,h) \cap Z(T',h')$.

(ii) *If* $p := T_v$ *is defined but not* T'_v*, then the following properties hold:*

$(ii.a)$ p *is regular w.r.t.* $\mathrm{sat}(T')$,

$(ii.b)$ $Z(T,h) \setminus Z(T',h') = Z(T,h) \setminus Z(p,T',h')$,

$(ii.c)$ $Z(T,h) \cap Z(T',h') = Z(T,h) \cap Z(p,T',h')$.

Proof. We first prove (i). As $\mathrm{init}(p')$ is regular w.r.t. $\mathrm{sat}(T'_{<v})$, it is also regular w.r.t. $\mathrm{sat}(T_{<v})$. Recall that T_v not defined means that $v \notin \mathrm{mvar}(T)$ holds. Therefore, the polynomial p' is also regular w.r.t. $\mathrm{sat}(T)$ and $[T, hp']$ is a regular system. On the other hand, we have the following disjoint decomposition

$$Z(T,h) = Z(T,hp') \uplus Z(p',T,h).$$

Observe that $Z(T,hp') \cap Z(T',h') = \emptyset$ holds. Therefore, we have

$$Z(T,h) \setminus Z(T',h') = Z(T,hp') \uplus \ (Z(p',T,h) \setminus Z(T',h'))$$

and we have $Z(T,h) \cap Z(T',h') = Z(p',T,h) \cap Z(T',h')$. This proves (i).

Now we prove (ii). Similarly to what we did in the proof of (i), we observe that

p is regular w.r.t. $\operatorname{sat}(T')$. Moreover, the following disjoint decomposition

$$Z(T', h') = Z(T', h'p) \cup Z(p, T', h'h_{T'}),$$

and the relation $Z(T, h) \cap Z(T', h'p) = \emptyset$ lead to the conclusion that (ii) holds. □

Lemma 5.6. *Assume that* $\operatorname{sat}(T_{<v}) = \operatorname{sat}(T'_{<v})$ *and* $\operatorname{sat}(T_{\leq v}) \neq \operatorname{sat}(T'_{\leq v})$ *both hold. Assume that v is algebraic w.r.t. both T and T'. Define*

$$\begin{aligned} \mathcal{G} &= \mathsf{GCD}(T_v, T'_v, v, T_{<v}); \\ \mathcal{D} &= \bigcup_{[g,C]\in\mathcal{G},\ |C|>|T_{<v}|} \mathsf{Extend}(C, T, v); \\ \mathcal{E} &= \{E \mid [f, E] \in \bigcup_{D\in\mathcal{D}} \mathsf{Regularize}(hh_T, D)\}. \end{aligned}$$

Then we have

(i)

$$Z(T, h) = \left(\bigcup_{[g,C]\in\mathcal{G},\ |C|=|T_{<v}|} Z(C \cup T_{\geq v}, hh_T) \right) \cup \left(\bigcup_{E\in\mathcal{E}, hh_T \notin \sqrt{\operatorname{sat}(E)}} Z(E, hh_T) \right).$$

(ii) $E \prec T$, *for all* $[f, E] \in \mathcal{E}$.

(iii) *for all* $[g, C] \in \mathcal{G}$ *such that* $|C| = |T_{<v}|$, *we have:*

$(iii.a)$ $C \cup T_{\geq v}$ *is a regular chain and* hh_T *is regular w.r.t.* $\operatorname{sat}(T_{\geq v})$,

$(iii.b)$ *if* $|\mathcal{G}| > 1$, *then* $C \cup T_{\geq v} \prec T$ *holds.*

Proof. W.l.o.g, we assume that the pairs in $\mathcal{G}$ are numbered $[g_1, C_1], \ldots, [g_e, C_e]$, $[g_{e+1}, C_{e+1}], \ldots, [g_s, C_s]$ such that:

- for all $1 \leq i \leq e$, we have $|C| = |T_{<v}|$,
- for all $e + 1 \leq i \leq s$, we have $|C| > |T_{<v}|$.

By the specification of the operation GCD we have $T_{<v} \longrightarrow C_1, \ldots, C_s$. For each C_i, with $e+1 \leq i \leq s$, the operation Extend computes a family of regular chains $\mathcal{D}_i$, such that

- $W(C_i \cup T_{\geq v}) \subseteq \cup_{D\in\mathcal{D}_i} W(D)$ holds, and

- $\sqrt{\mathrm{sat}(T)} \subseteq \sqrt{\mathrm{sat}(D)}$ holds, for each $D \in \mathcal{D}_i$.

Note that $\mathcal{D} = \cup_{i=e+1}^{s} \mathcal{D}_i$. For each $D \in \mathcal{D}$, the operation Regularize outputs a family of regular chains $\mathcal{E}_D$ such that we have the following regular split $D \longrightarrow (E \mid E \in E_D)$. Note that we have $\mathcal{E} = \cup_{D \in \mathcal{D}} E_D$. From the definition of a *regular split* (Definition 4.2, p. 40) we have

$$
\begin{aligned}
W(T) &= W(T_{<v} \cup T_{\geq v}) \\
&\subseteq \cup_{i=1}^{s} W(C_i \cup T_{\geq v}) \\
&= \cup_{i=1}^{e} W(C_i \cup T_{\geq v}) \ \cup \ \cup_{i=e+1}^{s} W(C_i \cup T_{\geq v}) \\
&\subseteq \left(\bigcup_{(g,C) \in \mathcal{G}, \ |C| = |T_{<v}|} W(C \cup T_{\geq v}) \right) \cup \left(\bigcup_{E \in \mathcal{E}} W(E) \right) \\
&\subseteq \overline{W(T)},
\end{aligned}
$$

which implies,

$$
\begin{aligned}
& Z(T,h) = Z(T, hh_T) \\
& \subseteq \left(\bigcup_{[g,C] \in \mathcal{G}, \ |C| = |T_{<v}|} Z(C \cup T_{\geq v}, hh_T) \right) \cup \left(\bigcup_{E \in \mathcal{E}, hh_T \notin \sqrt{\mathrm{sat}(E)}} Z(E, hh_T) \right) \\
& \subseteq \overline{W(T)} \setminus V(hh_T) = Z(T,h).
\end{aligned}
$$

This proves (i). We prove (iii). Consider $[g, C] \in \mathcal{G}$ such that $|C| = |T_{<v}|$ holds. Properties $(iii.a)$ and $(iii.b)$ follow immediately from Proposition 5 of [104]. This proves (ii). Similarly, (ii) follows from the same Proposition 5 of [104]. □

Lemma 5.7. *Assume that* $\mathrm{sat}(T_{<v}) = \mathrm{sat}(T'_{<v})$ *and* $\mathrm{sat}(T_{\leq v}) \neq \mathrm{sat}(T'_{\leq v})$ *both hold. Assume that* v *is algebraic w.r.t. both* T *and* T'*. Define* $p = T_v$, $p' = T'_v$ *and*

$$\mathcal{G} = \mathsf{GCD}(p, p', v, T_{<v}).$$

We assume that $\mathcal{G}$ *consists of a single pair* (g, C)*. Then the following properties hold.*

(i) *We have* $\overline{W(C)} = \overline{W(T_{<v})}$.

(ii) *If* $g \in \mathbf{k}$*, then*

$$Z(T,h) \setminus Z(T',h') = Z(T,h).$$

(iii) *If* $g \notin \mathbf{k}$ *and* $\operatorname{mvar}(g) < v$*, then* g *is regular w.r.t.* $\operatorname{sat}(T)$ *and we have*

$$Z(T,h) \setminus Z(T',h') = Z(T,gh) \cup (Z(g,T,hh_T) \setminus Z(T',h')) .$$

(iv) *Assume that* $\operatorname{mvar}(g) = v$*. Then the following properties hold.*

(iv.a) *If* $\operatorname{mdeg}(g) = \operatorname{mdeg}(p)$*, defining* $D'_p := T'_{<v} \cup \{p\} \cup T'_{>v}$*, then* $h'h_{T'}$ *is regular w.r.t.* $\operatorname{sat}(D'_p)$ *and we have*

$$Z(T,h) \setminus Z(T',h') = Z(T,h) \setminus Z(D'_p, h'h_{T'}).$$

Moreover, if $\operatorname{mdeg}(p) < \operatorname{mdeg}(p')$*, we have* $D'_p \prec T'$*.*

(iv.b) *If* $\operatorname{mdeg}(g) < \operatorname{mdeg}(p)$*, defining*

$$q := \operatorname{pquo}(p, g, v),\ \ D_g := T_{<v} \cup \{g\} \cup T_{>v} \ \text{ and } \ D_q := T_{<v} \cup \{q\} \cup T_{>v},$$

then the following properties hold:

(iv.b.1) *both* D_g *and* D_q *are regular chains,*

(iv.b.2) hh_T *is regular w.r.t. both* $\operatorname{sat}(D_g)$ *and* $\operatorname{sat}(D_q)$*,*

(iv.b.3) $D_g \prec T$ *and* $D_q \prec T$ *both hold,*

(iv.b.4) $Z(T,h) = Z(D_g, hh_T) \bigcup Z(D_q, hh_T) \bigcup Z(h_g, T, hh_T)$ *holds.*

Proof. Since $|\mathcal{G}| = 1$, by the specification of the operation GCD and the definition of a regular split (Definition 4.2, p. 40) we deduce property (i). Thus we have

$$\sqrt{\operatorname{sat}(C)} = \sqrt{\operatorname{sat}(T_{<v})} = \sqrt{\operatorname{sat}(T'_{<v})}. \tag{5.2}$$

Moreover from the specification of the operation GCD, there exist polynomials A and B such that

$$g \equiv Ap + Bp' \quad \bmod \quad \sqrt{\operatorname{sat}(C)}. \tag{5.3}$$

From (5.3), we have

$$V(\operatorname{sat}(C)) \subseteq V(g - Ap - Bp') \tag{5.4}$$

Therefore, we deduce

$$\begin{aligned}
& W(T) \cap W(T') \\
& = W(T_{<v} \cup p \cup T_{>v}) \cap W(T'_{<v} \cup p' \cup T'_{>v}) \\
& \subseteq (W(T_{<v}) \cap V(p)) \ \cap \ (W(T'_{<v}) \cap V(p')) \\
& \subseteq V(\mathrm{sat}(T_{<v})) \cap V(p) \cap V(p') \qquad \text{by (5.2)} \\
& \subseteq V(g - Ap - Bp') \cap V(p) \cap V(p') \qquad \text{by (5.4)} \\
& \subseteq V(g).
\end{aligned}$$

that is

$$W(T) \cap W(T') \subseteq V(g). \tag{5.5}$$

Now we prove (ii). When $g \in \mathbf{k}$, $g \neq 0$, from (5.5) we deduce $W(T) \cap W(T') = \emptyset$. Now we prove (iii). Since $\sqrt{\mathrm{sat}(T_{<v})} = \sqrt{\mathrm{sat}(C)}$ and since $\mathrm{mvar}(g)$ is smaller than v, by the specification of GCD, the polynomial g is regular w.r.t. $\mathrm{sat}(T)$. Moreover, we have following decompositions

$$\begin{aligned}
Z(T, h) &= Z(T, gh) \uplus Z(g, T, hh_T), \\
Z(T', h') &= Z(T', gh') \uplus Z(g, T', h'h_{T'}).
\end{aligned}$$

On the other hand,

$$Z(T, gh) \cap Z(T', gh') \subseteq (W(T) \cap W(T')) \setminus V(g) = \emptyset \qquad \text{by (5.5)}.$$

Therefore,

$$\begin{aligned}
& Z(T, h) \setminus Z(T', h') \\
& = (Z(T, gh) \setminus Z(T', h')) \ \uplus \ (Z(g, T, hh_T) \setminus Z(T', h')) \\
& = Z(T, gh) \uplus \ (Z(g, T, hh_T) \setminus Z(T', h'))\,.
\end{aligned}$$

This proves (iii). Now we prove $(iv.a)$. We distinguish two cases: $\mathrm{mdeg}(g) = \mathrm{mdeg}(p) = \mathrm{mdeg}(p')$ and $\mathrm{mdeg}(g) = \mathrm{mdeg}(p) < \mathrm{mdeg}(p')$. Assume first that $\mathrm{mdeg}(g) = \mathrm{mdeg}(p) = \mathrm{mdeg}(p')$ holds. By Proposition 3.2, we have $\sqrt{\mathrm{sat}(T'_{<v} \cup p)} = \sqrt{\mathrm{sat}(T'_{<v} \cup p')}$, which implies that $\sqrt{\mathrm{sat}(T'_{<v} \cup p \cup T'_{>v})} = \sqrt{\mathrm{sat}(T'_{<v} \cup p' \cup T'_{>v})}$ by Corollary 4.2. So we have $Z(T', h') = Z(h_p, T', h') \cup Z(D'_p, h'h_{p'})$. Therefore $Z(T, h) \setminus Z(T', h') = Z(T, h) \setminus Z(D'_p, h'h_{T'})$ holds.

Now we assume that $\mathrm{mdeg}(g) = \mathrm{mdeg}(p) < \mathrm{mdeg}(p')$ holds. In this case, p is also a GCD of p and p' w.r.t. $T_{<v}$. (This fact is clear on the algorithm of GCD see Algorithm 15, in Chapter 4.) Let $q' := \text{pquo}(p', p, v)$. Define $D'_{q'} := T'_{<v} \cup \{q'\} \cup T'_{>v}$ and $D'_p := T'_{<v} \cup \{p\} \cup T'_{>v}$. By Proposition 3.2, we have

$$\begin{aligned} Z(T', h') &\subseteq Z(D'_p, h') \cup Z(D'_{q'}, h') \cup Z(h_p, T', h') \\ &\subseteq \overline{W(T')} \setminus V(h'). \end{aligned}$$

With Corollary 2.1, we deduce

$$Z(T', h') = Z(D'_p, h'h_{T'}) \cup Z(D'_{q'}, h'h_{T'}) \cup Z(h_p, T', h'h_{T'}). \tag{5.6}$$

In the other hand, we have

$$\begin{aligned} Z(D'_{q'}, h'h_{T'}) &= Z(D'_{q'}, h_p h' h_{T'}) \uplus Z(h_p, D'_{q'}, h'h'_T) \\ &= Z(D'_{q'}, p h_p h' h_{T'}) \uplus Z(p, D'_{q'}, h_p h' h'_T) \uplus Z(h_p, D'_{q'}, h'h'_T) \end{aligned}$$

and

$$Z(p, D'_{q'}, h_p h' h'_T) \subseteq Z(D'_p, h'h_{T'}) \quad \text{and} \quad Z(h_p, D'_{q'}, h'h'_T) \subseteq Z(h_p, T', h'h_{T'}).$$

Combined with (5.6) we obtain

$$Z(T', h') = Z(D'_p, h'h_{T'}) \uplus Z(D'_{q'}, p h_p h' h_{T'}) \uplus Z(h_p, T', h'h_{T'}).$$

Now observe that

$$\begin{aligned} &Z(T, h) \cap Z(D'_{q'}, p h_p h' h_{T'}) = \emptyset, \text{ and} \\ &Z(T, h) \cap Z(h_p, T', h'h_{T'}) = \emptyset. \end{aligned}$$

We deduce

$$Z(T, h) \setminus Z(T', h') = Z(T, h) \setminus Z(D'_p, h'h_{T'}).$$

This completes the proof of $(iv.a)$. Finally property $(iv.b)$ follows from Proposition 3.2. This completes the whole proof. □

Theorem 5.1. *Algorithms* Difference *and* DifferenceLR *terminate and satisfy their specifications.*

Proof. Let ($R_1 = [T_1, h_1], R'_1 = [T'_1, h'_1]$) be the initial input of Difference. Let ($R_2 = [T_2, h_2], R'_2 = [T'_2, h'_2]$) be the input of Difference when a recursive call is made. Let v_1 be the largest variable v such that $\mathrm{sat}(T_{1<v}) = \mathrm{sat}(T'_{1<v})$ holds. Let v_2 be the largest variable v such that $\mathrm{sat}(T_{2<v}) = \mathrm{sat}(T'_{2<v})$ holds. We observe that only the following three cases may arise:

- the rank of R_2 is less than that of R_1 (Lines 9, 21, 30, 31, 32, 38, 39)
- the rank of R'_2 is less than that of R'_1 (Line 12 and Line 25 if $\mathrm{mdeg}(p) < \mathrm{mdeg}(p')$)
- the ranks of R_1 and R_2 are the same; the ranks of R'_1 and R'_2 are also the same; however, v_2 is strictly larger than v_1 (Line 25 if $\mathrm{mdeg}(p) = \mathrm{mdeg}(p')$)

Therefore the algorithm Difference terminates. Its correctness follows directly from the previous lemmas. Finally the termination and correctness of DifferenceLR are implied by those of Difference. □

5.5 Application to the verification of polynomial system solvers

Given a polynomial system F and a set of components $C_1, \ldots, C_e$, it is hard, in general, to tell whether the union of $C_1, \ldots, C_e$ corresponds exactly to the solution set $V(F)$ or not. Actually, solving this verification problem is generally (at least) as hard as solving the system F itself.

Because of the high complexity of symbolic solvers, developing both verification algorithms and reliable verification software tools is a clear need. However, this verification problem has received little attention in the literature. In this section, we present new techniques for verifying a large class of symbolic solvers. We also report on intensive experimentation illustrating the high efficiency of our approach w.r.t. known techniques.

We assume that each component of the solution set $V(F)$ is given by a regular system. Recall that, in broad words, a regular system consists of several polynomial equations with a triangular shape

$$p_1(x_1) = p_2(x_1, x_2) = \cdots = p_i(x_1, x_2, \ldots, x_n) = 0$$

and a polynomial inequation

$$h(x_1, \ldots, x_n) \neq 0$$

such that there exists (at least) one point $(a_1, \ldots, a_n)$ satisfying the above equations and inequation. Note that these polynomials may contain parameters.

Let us consider the following well-known system F taken from [55].

$$\begin{cases} x^{31} - x^6 - x - y &= 0 \\ x^8 - z &= 0 \\ x^{10} - t &= 0 \end{cases}$$

We aim at solving this system for $x > y > z > t$, that is, expressing x as a function of y, z, t, then y as a function of z, t and z as a function of t. One possible decomposition is given by the three regular systems below:

$$\begin{cases} (t^4 - t)\, x - ty - z^2 &= 0 \\ t^3y^2 + 2t^2z^2y + (-t^6 + 2t^3 + t - 1)\, z^4 &= 0 \\ z^5 - t^4 &= 0 \\ t^4 - t &\neq 0 \end{cases}, \quad \begin{cases} x^2 - z^4 &= 0 \\ y + t^2z^2 &= 0 \\ z^5 - t &= 0 \\ t^3 - 1 &= 0 \end{cases}, \quad \begin{cases} x &= 0 \\ y &= 0 \\ z &= 0 \\ t &= 0 \end{cases}$$

Another decomposition is given by these other three regular systems:

$$\begin{cases} (t^4 - t)\, x - ty - z^2 &= 0 \\ tzy^2 + 2z^3y - t^8 + 2t^5 + t^3 - t^2 &= 0 \\ z^5 - t^4 &= 0 \\ z\,(t^4 - t) &\neq 0 \end{cases}, \quad \begin{cases} zx^2 - t &= 0 \\ ty + z^2 &= 0 \\ z^5 - t &= 0 \\ t^3 - 1 &= 0 \\ tz &\neq 0 \end{cases}, \quad \begin{cases} x &= 0 \\ y &= 0 \\ z &= 0 \\ t &= 0 \end{cases}$$

These two decompositions look slightly different (in particular, the second components) and one could think that, if each of them was produced by a different solver, then at least one of these solvers has a bug. In fact, both decompositions are valid, but proving that they both encode the solution set $V(F)$ is not feasible without computer assistance. However, proving that they define the same set of points can be achieved by an expert hand without computer assistance. This is an important observation that will guide us in this work.

Let us consider now an arbitrary input system F and a set of components $C_1, \ldots, C_e$ encoded by regular systems $S_1, \ldots, S_e$ respectively. The usual approach for verifying that $C_1, \ldots, C_e$ correspond exactly to the solution set $V(F)$ is as follows.

(1) First, one checks that each candidate component C_i is actually contained in $V(F)$. This essentially reduces to substitute the coordinates of the points given by C_i into the polynomials of F: if all these polynomials vanish at these points,

then C_i is a component of $V(F)$, otherwise, (and up to technical details that we will skip in this overview) C_i is not a component of $V(F)$.

(2) Secondly, one checks that $V(F)$ is contained in the union of the candidate components $C_1, \ldots, C_e$ by:

(2.1) computing a polynomial system G such that $V(G)$ corresponds exactly to $C_1, \ldots, C_e$, and

(2.2) checking that every solution of $V(F)$ cancels the polynomials of G.

Steps (2.1) and (2.2) can be performed using standard techniques based on computations of Gröbner bases, as we discuss in Section 5.5.3. These calculations are very expensive in practice, as shown by our experimentation, reported in Section 5.5.5.

In this work, we propose a different approach, summarized in non-technical language in Section 5.5.1. The main idea is as follows. Instead of comparing a candidate set of components $C_1, \ldots, C_e$ against the input system F, we compare it against the output $D_1, \ldots, D_f$ produced by another solver. Both this solver and the comparison process are assumed to be validated. Hence, the candidate set of components $C_1, \ldots, C_e$ corresponds exactly to the solution set $V(F)$ if and only if the comparison process shows that $D_1, \ldots, D_f$ and $C_1, \ldots, C_e$ define the same solution set.

The solvers we consider in this study are those solving polynomial systems by means of *triangular decompositions* in the so-called *sense of Lazard*. This choice is motivated by the following reasons. First, the case of decompositions in the sense of Kalkbrener was treated in [6], via Gröbner basis computations. Secondly, most algorithms computing triangular decompositions use the sense of Lazard and no verification tool for those has been reported prior to our work. We leave for future research the verification of Kalkbrener's decompositions by means of more efficient techniques than those reported in [6].

5.5.1 Methodology

Let us consider again an arbitrary input polynomial system F and a set of components $C_1, \ldots, C_e$ encoded by regular systems $S_1, \ldots, S_e$ respectively. As mentioned in the Introduction, checking whether $C_1, \ldots, C_e$ corresponds exactly to the solution set $V(F)$ of F can be done by means of Gröbner bases computations. This verification process is quite simple, see Section 5.5.2, and its implementation is straightforward, Thus, if the underlying Gröbner bases engine is *reliable*, such verification tool can be regarded as safe. See [7] for details.

Unfortunately, this verification process is highly expensive. Even worse, as shown by our experimental results in Section 5.5.5, this verification process is unable to check many triangular decompositions that are easy to compute.

We propose a new approach in order to overcome this limitation. Assume that we have at hand a reliable solver computing triangular decompositions of polynomial systems. We believe that this reliability can be acquired over time by combining several features.

- Checking the solver with a verification tool based on Gröbner bases for input systems of moderate difficulty.
- Using the solver for input systems of higher difficulty where the output can be verified by theoretical arguments, see [8] for an example of such input system.
- Involving the library supporting the solver in other applications.
- Making the solver widely available to potential users.

Suppose that we are currently developing a new solver computing triangular decompositions. In order to verify the output of this new solver, we can take advantage of the reliable solver.

This may sound natural and easy in the first place, but this is actually not. Indeed, as shown in the Introductory Chapter, two different solvers can produce two different, but valid, triangular decompositions for the same input system. Checking that these two triangular decompositions encode the same solution set boils down to compute the differences of two constructible sets. This is a non-trivial operation, see the survey paper [113].

The first contribution of our work is to provide a relatively simple, but efficient, procedure for computing the set theoretical differences between two constructible sets, see Section 5.4. Such procedure can be used to develop a verification tool for our new solver by means of our reliable solver. Moreover, this procedure is sufficiently straightforward to implement such that it can be trusted after a relatively short period of testing, as the case for the verification tool based on Gröbner bases computations.

The second contribution of our work is to illustrate the high efficiency of this new verification tool. In Section 5.5.5, we consider four solvers computing triangular decomposition of polynomial systems:

- the command *Triangularize* of the *RegularChains* library [88] in MAPLE
- the Triade solver of the *BasicMath* library [70] in ALDOR

- the commands *RegSer* and *SimSer* of the *Epsilon* library [124] in Maple.

We have run these four solvers on a large set of well-known input systems from the data base [96, 118, 127]. For those systems for which this is feasible, we have verified their computed triangular decompositions with a verification tool based on Gröbner bases computations. Then, for each input system, we have compared all its computed triangular decompositions by means of our new verification tool.

Based on our experimentation data reported in Section 5.5.5 we make the following observations.

- All computed triangular decompositions, that could be checked via Gröbner bases computations, are correct.
- However, the verification tool based on Gröbner bases computations failed to check many examples by running out of computer memory.
- For each input system F, all pairs of triangular decompositions of F could be compared successfully by our new verification tool.
- Moreover, for any system F to which all verification tools could be applied, our new approach runs much faster.

This suggests that the four solvers and our new verification tool have a good level of reliability. Moreover, it allows to process cases that were previously out of reach.

5.5.2 Verification of triangular decompositions

In this section, we describe how to verify the output from a triangular decomposition solver. For verification of triangular decomposition in Kalkbrener's sense, it is still unknown whether we can circumvent Gröbner basis computations. However, in Lazard's sense, we present two methods, based on Gröbner bases and regular systems, respectively.

5.5.3 Verification with Gröbner bases

Given a set of polynomials F and a polynomial f in $\mathbf{k}[\mathbf{x}]$, we denote $D(F, f)$ the difference of $V(F)$ and $V(f)$. If F is the empty set, then we write $D(f)$ for short. The following two lemmas state the Gröbner basis methods to verify whether two basic constructible sets are equal or not.

Lemma 5.8. *Let $F, G_0, G_1, \ldots, G_r$ be finite polynomial sets of $\mathbf{k}[\mathbf{x}]$ and $f, g_0, g_1, \ldots, g_r$ be polynomials of $\mathbf{k}[\mathbf{x}]$. The following statements are equivalent*

1. $D(F, f) \setminus D(G_0, g_0) \subseteq \bigcup_{i=1}^{r} D(G_i, g_i)$.

2. *For every integer s such that $0 \leq s \leq r$, for every subset $\{i_1, \ldots, i_s\} \subseteq \{0, \ldots, r\}$, we have*

$$\sqrt{\langle F \cup \{g_{i_1}, \ldots, g_{i_s}\}\rangle} \supseteq \prod_{k \in \{0,\ldots,r\} \setminus \{i_1,\ldots,i_s\}} \langle f \rangle \langle G_k \rangle. \tag{5.7}$$

Proof. (1) is equivalent to $D(F, f) \subseteq \bigcup_{i=0}^{r} D(G_i, g_i)$, that is

$$D(F, f) \bigcap \left(\bigcap_{i=0}^{r} D(G_i, g_i)^c \right) = \emptyset.$$

By distributivity, we deduce that (1) is equivalent to

$$\Big(D(F, f) \bigcap V(g_{i_1}, \ldots, g_{i_s}) \Big) \bigcap \left(\bigcap_{k \in \{0,\ldots,r\} \setminus \{i_1,\ldots,i_s\}} V(G_k)^c \right) = \emptyset,$$

for all subsets $\{i_1, \ldots, i_s\}$ of $\{0, \ldots, r\}$. The proof easily follows. □

Lemma 5.9. *Let $F, G_0, G_1, \ldots, G_r$ be finite polynomial sets of $\mathbf{k}[\mathbf{x}]$ and $f, g_0, g_1, \ldots, g_r$ be polynomials of $\mathbf{k}[\mathbf{x}]$. The following statements are equivalent*

1. $D(F, f) \setminus D(G_0, g_0) \supseteq \bigcup_{i=1}^{r} D(G_i, g_i)$.

2. *For all $1 \leq i \leq r$, we have*

$$g_i g_0 \in \sqrt{\langle G_i \cup G_0 \rangle}, g_i \in \sqrt{\langle G_i, f \rangle}, \text{ and } \langle g_i \rangle \langle F \rangle \subset \sqrt{\langle G_i \rangle}. \tag{5.8}$$

Proof. (1) holds if and only if for each $1 \leq i \leq r$ we have

$$\begin{cases} D(G_i, g_i) \bigcap D(F, f)^c &= \emptyset, \\ D(G_i, g_i) \bigcap D(G_0, g_0) &= \emptyset, \end{cases}$$

which holds if and only if

$$\begin{cases} V(G_i) \bigcap V(g_i)^c \bigcap V(F)^c &= \emptyset, \\ V(G_i) \bigcap V(g_i)^c \bigcap V(f) &= \emptyset, \\ V(G_i) \bigcap V(g_i)^c \bigcap V(G_0) \bigcap V(g_0)^c &= \emptyset. \end{cases}$$

The proof easily follows. □

The above general lemmas can be used to check if the output from the algorithm Difference is correct or not. In particular, they can be applied to check if a triangular decomposition is valid or not by comparing the input system and one triangular decomposition. We naively implement them using MAPLE package *PolynomialIdeals*.

5.5.4 Verification with the Difference algorithm

Given two Lazard-Wu's triangular decompositions $\{T_i \mid i = 1 \ldots e\}$ and $\{S_j \mid j = 1 \ldots f\}$. Checking $\cup_{i=1}^{e} W(T_i) = \cup_{j=1}^{f} W(S_j)$ amounts to checking both

$$\left(\bigcup_{i=1}^{e} Z(T_i, 1)\right) \setminus \left(\bigcup_{j=1}^{f} Z(S_j, 1)\right) \text{ and } \left(\bigcup_{j=1}^{f} Z(S_j, 1)\right) \setminus \left(\bigcup_{i=1}^{e} Z(T_i, 1)\right)$$

being empty, where $[T_i, 1]$ and $[S_j, 1]$ are all regular systems. This is equivalent to check whether DifferenceLR returns the empty set for both differences.

5.5.5 Experimentation

We have implemented a verifier, named *Diff-verifier*, according to the DifferenceLR algorithm proposed in Section 5.4, and it has been implemented in MAPLE 11 based on the `RegularChains` library. To verify the effectiveness of our Diff-verifier, we have also implemented another verifier, named *GB-verifier*, applying Lemma 5.8 and Lemma 5.9, on top of the *PolynomialIdeals* package in MAPLE 11.

We use these two verifiers to examine four polynomial system solvers herein. They are the *Triangularize* function in the *RegularChains* library [88], the Triade server in *Aldor* on top of the *BasicMath library* [70], the *RegSer* function and the *SimSer* function in *Epsilon* [124] implemented in MAPLE. The first two solvers solve a polynomial system into regular chains by means of the Triade algorithm [104]. They can work in both *Lazard's* sense and *Kalkbrener's* sense. In this work, we use the options for solving in *Lazard's* sense. The *RegSer* function decomposes a polynomial system into regular systems in a strong sense, and the *SimSer* function decomposes a polynomial system into simple systems. They adopt the elimination methods in [127].

The problems used in this benchmark are chosen from [96, 118, 127]. In Table 5.1, for each system, we give the *dimension* sequence of the triangular decomposition computed in *Kalkbrener's* sense by the Triade algorithm. The number of variables is denoted by n, and d is the maximum degree of a monomial. We also give the number of components in the solution set for each of the methods we are studying.

Table 5.2 gives the timing of each problem solved by the four methods. In this study, due to the current availability of *Epsilon*, the timings obtained by the *RegSer* and the *SimSer* are performed in Maple 8 on Intel Pentium 4 machines (1.60GHz CPU, 513MB memory and Red Hat Linux 3.2.2-5). All the other timings are run on Intel Pentium 4 (3.20GHz CPU, 2.0GB total memory, and Red Hat 4.0.0-9), and the MAPLE version used is 11. The Triade server is a stand-alone executable program compiled from a program in `Aldor`.

Table 5.3 summarizes the timings of GB-verifier for verifying the solutions of the four methods. Table 3 illustrates the timings of Diff-verifier for checking the solutions byMAPLE *Triangularize* against Aldor Triade server, MAPLE *Triangularize* against *Epsilon RegSer*, and *Epsilon RegSer* against *Epsilon SimSer*. For the case where there is a time, the verifying result is also true. The $'-'$ denotes the case where the test stalls by either reaching the time limit of 43200 seconds or causing a memory failure.

Based on (5.1) in Section 5.4, we implement the Difference operation naively, and we call it *Naive-diff-verifier*. From the Table 5.4 we can see clearly that, for most problems, the Diff-verifier performs much better than Naive-diff-verifier, especially for hard problems.

This experimentation results illustrate that verifying a polynomial solver is a truly difficult task. The GB-verifier is very costly in terms of CPU time and memory. It only succeeds for some easy examples. Assuming that the GB-verifier is reliable, for the examples it succeeds, the Diff-verifier agrees with its results by pairwise checking, while it takes much less time. This shows the efficiency of our Diff-verifier. Moreover, the Diff-verifier succeeds in computing the difference for any pair of output of the four solvers. (The comparison between GB-verifier and Diff-verifer is a bit unfair, since Gröbner basis method has to keep all information like multiplicities, whereas Difference does not.) Therefore, our new approach can verify the solution set of all test polynomial systems that at least two of our four solvers can solve, which serves well for our purpose.

Furthermore, the tests also show that the Diff-verifier can verify quite difficult problems. Therefore, the tests indicate that all four solvers are solving tools with a high probability of correctness, since the checking results would not agree to each other otherwise.

					Number of Components			
Sys	Name	n	d	Dimension	Maple 11 *Triangularize*	`Aldor` Triade server	*Epsilon RegSer*	*Epsilon SimSer*
1	Montes S1	4	2	[2,2,1]	3	3	3	3
2	Montes S2	4	3	[0]	1	1	1	1
3	Montes S3	3	3	[1,1]	2	2	2	3
4	Montes S4	4	2	[0]	1	1	1	1
5	Montes S6	4	3	[2,2,2]	3	3	3	3
6	Montes S7	4	3	[1]	2	2	3	6
7	Montes S8	4	12	[2,1]	2	2	6	6
8	Alonso	7	4	[3]	3	3	3	4
9	Raksanyi	8	3	[4]	4	4	4	10
10	YangBaxter Rosso	6	3	[4,3,3,1,1,1,1]	7	7	4	13
11	l-3	4	3	[0,0,0,0,0,0, 0,0,0,0,0,0,0]	25	13	8	8
12	Caprasse	4	4	[0,0,0,0,0]	15	5	4	4
13	Reif	16	3	[ ]	0	0	0	0
14	Buchberger WuWang	5	3	[2]	3	3	3	4
15	DonatiTraverso	4	31	[1]	6	3	3	3
16	Wu-Wang.2	13	3	[1,1,1,1,1]	5	5	5	5
17	Hairer-2-BGK	13	4	[2]	4	4	5	6
18	Montes S5	8	3	[4]	4	4	4	10
19	Bronstein	4	3	[1]	4	2	4	9
20	Butcher	8	4	[3,3,3,2,2,0]	7	6	6	6
21	genLinSyst-2-2	8	2	[6]	11	11	11	11
22	genLinSyst-3-2	11	2	[8]	17	18	18	18
23	Gerdt	7	4	[3,2,2,2,1,1]	7	6	10	10
24	Wang93	5	3	[1]	5	4	6	7
25	Vermeer	5	5	[1]	5	4	12	14
26	Gonnet	5	2	[3,3,3]	3	3	9	9
27	Neural	4	3	[1,1]	4	3	–	–
28	Noonburg	4	3	[1,1]	4	3	–	–
29	KdV	26	3	[12,12,11, 11,11,11,11]	7	7	–	–
30	Montes S12	8	2	[4]	22	17	23	–
31	Pappus	12	2	[6,6,6,6,6, 6,6,6,6,6]	124	129	156	–

Table 5.1: Features of the polynomial systems

Sys	Maple 11 *Triangularize*	`Aldor` Triade server	*Epsilon* *RegSer*	*Epsilon* *SimSer*
1	0.104	0.164	0.01	0.03
2	0.039	0.204	0.03	0.02
3	0.069	0.06	0.019	0.111
4	0.510	0.072	0.049	0.03
5	0.052	0.096	0.03	0.03
6	0.150	0.06	0.09	5.14
7	0.376	0.072	0.2	1.229
8	0.204	0.065	0.109	0.16
9	0.460	0.066	0.141	0.481
10	1.252	0.108	0.069	0.21
11	5.965	0.587	1.53	2.91
12	2.426	0.167	1.209	2.32
13	123.823	1.886	1.979	2.36
14	0.2	0.101	0.049	0.109
15	2.641	0.08	0.439	0.7
16	105.835	1.429	5.49	6.14
17	23.453	0.688	1.76	1.679
18	0.484	0.078	0.13	0.471
19	0.482	0.071	0.24	1.000
20	9.325	0.442	1.689	2.091
21	0.557	0.096	0.13	0.21
22	1.985	0.173	0.431	0.411
23	4.733	0.499	3.5	4.1
24	7.814	5.353	2.18	30.24
25	26.533	0.580	4.339	60.65
26	3.983	0.354	2.18	2.48
27	15.879	1.567	–	–
28	15.696	1.642	–	–
29	9245.442	49.573	–	–
30	17.001	0.526	2.829	–
31	79.663	4.429	11.78	–

Table 5.2: Solving timings in sec. of the four methods

	GB-verifier timing(s)				Diff-verifier timing(s)		
sys	Maple 11 *Triangularize* (M.T.)	`Aldor` TRIADE server (A.T.)	*Epsilon RegSer* (E.R.)	*Epsilon SimSer* (E.S.)	M.T. vs A.T.	M.T. vs E.R.	E.R. vs E.S.
1	0.556	0.526	0.518	0.543	0.188	0.238	0.217
2	0.128	0.127	0.129	0.131	0.012	0.010	0.010
3	0.584	0.575	0.585	2.874	0.067	0.088	0.326
4	0.104	0.133	0.139	0.137	0.018	0.017	0.018
5	1.484	1.472	1.457	1.469	0.198	0.178	0.190
6	76.596	72.374	71.853	–	2.010	2.390	12.591
7	0.616	0.601	4.501	4.536	0.191	0.404	0.492
8	–	–	–	–	0.571	0.677	0.925
9	–	–	–	–	4.257	4.454	7.884
10	–	–	–	–	6.555	8.824	9.037
11	–	–	–	–	5.341	3.564	1.997
12	–	58.332	33.469	35.213	1.506	1.657	2.354
13	–	–	–	–	0.000	0.000	0.000
14	1.96	1.937	2.165	5.739	0.617	0.661	0.722
15	330.317	–	–	–	1.689	3.095	2.870
16	10466.587	–	–	–	1.340	0.795	0.773
17	–	–	–	–	1.883	2.272	4.903
18	–	–	–	–	4.450	4.596	8.063
19	1.544	0.717	5.046	–	2.162	6.382	41.374
20	–	–	–	–	5.683	5.113	5.949
21	–	–	–	–	6.595	6.621	4.441
22	–	–	–	–	21.689	17.943	11.503
23	–	–	–	–	4.073	5.071	5.775
24	–	–	–	–	1064.127	636.221	707.668
25	–	–	–	–	817.499	1519.858	1585.095
26	–	–	–	–	0.554	1.276	1.741
27	11383.335	–	–	–	1072.199	–	–
28	–	–	–	–	1248.353	–	–
29	–	–	–	–	5.418	–	–
30	–	–	–	–	428.503	706.854	–
31	–	–	–	–	8071.055	9800.086	–

Table 5.3: Timings of GB-verifier and Diff-verifier

Sys	Naive-diff-verifier timing(s)	Diff-verifier timing(s)	Sys	Naive-diff-verifier timing(s)	Diff-verifier timing(s)
1	0.027	0.188	17	10876.470	1.883
2	0.003	0.012	18	5.498	4.450
3	0.075	0.067	19	7.491	2.162
4	0.010	0.018	20	450.342	5.683
5	0.049	0.198	21	158.879	6.595
6	2.146	2.010	22	4450.023	21.689
7	0.111	0.191	23	11.415	4.073
8	1.815	0.571	24	25047.768	1064.127
9	5.342	4.257	25	–	817.499
10	58.938	6.555	26	0.373	0.554
11	–	5.341	27	2466.459	1072.199
12	–	1.506	28	2464.389	1248.353
13	0.000	0.000	29	316.925	5.418
14	3.254	0.617	30	–	428.503
15	11.813	1.689	31	–	8071.055
16	11.374	1.340			

Table 5.4: Timings of Naive-diff-verifier and Diff-verifier for M.T. vs A.T.

Chapter 6

Comprehensive Triangular Decomposition

We introduce the concept of comprehensive triangular decomposition (CTD) for a parametric polynomial system P with coefficients in an arbitrary field $\mathbf{k}$. In broad words, it is a finite partition of the parameter space into cells such that each cell C is associated with a triangular decomposition of P that is "well-behaved" under specialization at any point of C. We propose several output specifications of CTD addressing different problems regarding the solutions of P as functions of the parameters. We also compare our algorithms, both theoretically and in practice, with other tools for solving parametric polynomial systems.

6.1 Introduction

Solving polynomial systems with parameters has become an increasing need in several applied areas such as robotics, geometric modeling, stability analysis of dynamical systems and others. For a given parametric polynomial system P, the following problems are of interest:

(1) Compute the values of the parameters for which P has solutions or finitely many solutions, or satisfies certain properties such as continuity. Determine the number of solutions or the dimension of solution set depending on parameters.

(2) Compute the solutions of P as functions of the parameters.

These questions have been approached by various techniques including Gröbner bases [86], comprehensive Gröbner bases (CGB) [130, 131, 102, 96, 117], cylindrical algebraic decomposition (CAD) [24] and triangular decompositions [132, 133, 42, 43, 68, 57, 125, 126, 138, 37, 136]. Methods based on CGB, or more generally Gröbner bases, are powerful tools for solving Problem (1), that is, determining the values u of the parameters such that, the specialized system $F(u)$ satisfies a given property. Methods based on CAD or triangular decompositions are naturally well designed for solving Problem (2).

In this paper, we introduce the concept of *comprehensive triangular decomposition* for a parametric polynomial system with coefficients in a field. This notion plays the role for triangular decompositions that CGB does for Gröbner bases. With this concept at hand, we show that Problems (1) and (2) can be completely answered by means of triangular decompositions.

We first consider parametric polynomial systems involving only equations. Let F be a finite set of polynomials with coefficients in a field $\mathbf{k}$, parameters $\mathbf{u} = u_1, \ldots, u_d$, and unknowns $\mathbf{y} = y_1, \ldots, y_m$, that is, $F \subset \mathbf{k}[u_1, \ldots, u_d, y_1, \ldots, y_m]$. Let $\mathbf{K}$ be the algebraic closure of $\mathbf{k}$, and let $V(F) \subset \mathbf{K}^{d+m}$ be the zero set of F . Let also $\pi_{\mathbf{u}}$ be the projection from $\mathbf{K}^{d+m}$ on the parameter space $\mathbf{K}^d$. For all $u \in \mathbf{K}^d$ we define $V(F(u)) \subseteq \mathbf{K}^m$ the zero set defined by F after specializing $\mathbf{u}$ at u.

Our first contribution is to show how to compute a finite partition $\mathcal{C}$ of $\pi_{\mathbf{u}}(V(F))$ and a family of triangular decompositions $(\mathcal{T}_C, C \in \mathcal{C})$ in $\mathbf{k}[\mathbf{u}, \mathbf{y}]$ such that for each $C \in \mathcal{C}$ and for each parameter value $u \in C$ the triangular decomposition $\mathcal{T}_C$ specializes at u into a triangular decomposition $\mathcal{T}_C(u)$ of $V(F(u))$ given by regular chains. Moreover, each "cell" $C \in \mathcal{C}$ is a constructible set given by a family of regular systems in $\mathbf{k}[\mathbf{u}]$. We call the pair $(\mathcal{C}, (\mathcal{T}_C, C \in \mathcal{C}))$ a *comprehensive triangular decomposition* of $V(F)$, see Section 6.3.

This is a natural definition inspired by that of a comprehensive Gröbner basis [130] introduced by Weispfenning with the additional requirements proposed by Montes in [102]. From each pair $(C, \mathcal{T}_C)$, we can read geometrical information, such as for which parameter values $u \in C$ the set $V(F(u))$ is finite; we also obtain a "generic" equidimensional decomposition of $V(F(u))$, for all $u \in C$. The notion of CTD is also related to the border polynomial of a polynomial system in [138] and the minimal discriminant variety of $V(F)$ as defined in [86] for the case where $\mathbf{K}$ is the field of complex numbers. See Section 6.6 for detailed discussions.

Example 6.1. *Let $F = \{vxy + ux^2 + x, uy^2 + x^2\}$ be a parametric polynomial system with parameters $u > v$ and unknowns $x > y$. Then a comprehensive triangular* *decomposition of* $V(F)$ *is:*

$$\begin{aligned} C_1 = \{u(u^3+v^2) \neq 0\}: &\quad \mathcal{T}_{C_1} = \{T_3, T_4\} \\ C_2 = \{u = 0\}: &\quad \mathcal{T}_{C_2} = \{T_2, T_3\} \\ C_3 = \{u^3+v^2 = 0, v \neq 0\}: &\quad \mathcal{T}_{C_3} = \{T_1, T_3\} \end{aligned}$$

where

$$\begin{aligned} T_1 &= \{vxy + x - u^2y^2, 2vy + 1, u^3 + v^2\} \\ T_2 &= \{x, u\} \\ T_3 &= \{x, y\} \\ T_4 &= \{vxy + x - u^2y^2, u^3y^2 + v^2y^2 + 2vy + 1\} \end{aligned}$$

Here , C_1, C_2, C_3 *is a partition of* $\pi_{\mathbf{u}}(V(F))$ *and* $\mathcal{T}_{C_i}$ *is a triangular decomposition of* $V(F)$ *above* C_i*, for* $i = 1, 2, 3$*. For different parameter values* u*, we can directly read geometrical information, such as the dimension of* $V(F(u))$*.*

By RegSer *[126],* $V(F)$ *can be decomposed into a set of regular systems:*

$$R_1 = \begin{cases} ux + vy + 1 &= 0 \\ (u^3+v^2)y^2 + 2vy + 1 &= 0 \\ u(u^3+v^2) &\neq 0 \end{cases}, \quad R_2 = \begin{cases} x &= 0 \\ y &= 0 \\ u &\neq 0 \end{cases},$$

$$R_3 = \begin{cases} x &= 0 \\ vy + 1 &= 0 \\ u &= 0 \\ v &\neq 0 \end{cases}, \quad R_4 = \begin{cases} 2ux + 1 &= 0 \\ 2vy + 1 &= 0 \\ u^3 + v^2 &= 0 \\ v &\neq 0 \end{cases}, \quad R_5 = \begin{cases} x &= 0 \\ u &= 0 \end{cases}.$$

For each regular system, one can directly read its dimension when parameters take corresponding values. However, the dimension of the input system could not be obtained immediately, since a partition of the parameter space is not provided.

By DISPGB [102], one can obtain all the cases over the parameters leading to different reduced Gröbner bases with parameters:

$$\begin{aligned} u(u^3+v^2) \neq 0: &\quad \{ux + (u^3v + v^3)y^3 + (-u^3 + v^2)y^2, (u^3+v^2)y^4 + 2vy^3 + y^2\} \\ u(u^3+v^2) = 0, u \neq 0: &\quad \{ux + 2v^2y^2, 2vy^3 + y^2\} \\ u = 0, v \neq 0: &\quad \{x^2, vxy + x\} \\ u = 0, v = 0: &\quad \{x\} \end{aligned}$$

Here for each parameter value, the input system specializes into a Gröbner basis. Since Gröbner bases do not necessarily have a triangular shape, the dimension may not be read immediately either. For example, when $u = 0, v \neq 0$, $\{x^2, vxy + x\}$ is not a triangular set.

In Section 6.3 we also propose an algorithm for computing the CTD of any parametric polynomial system. It relies on a procedure for solving the following problem. Given a family of constructible sets $A_1, \ldots, A_s$, compute a family $B_1, \ldots, B_t$ of pairwise disjoint constructible sets, such that for all $1 \leq i \leq s$ the set A_i writes as a union of some of the $B_1, \ldots, B_t$. This can be seen as the set theoretical version of the *coprime factorization* problem, see [11, 49] for other variants of this problem. Our solution is presented in Section 6.2 based on the Difference algorithm presented in Chapter 5 for computing the difference of the zero sets of two regular systems.

For a polynomial system involving inequations, or more generally a parametric constructible set, one can decompose it into regular systems by the triangular decomposition algorithm presented in Chapter 5. This suggests us to generalize the previous definition of CTD to the case of a parametric constructible set. This is done in Section 6.4. Moreover, in the same section we introduce the concept of disjoint squarefree comprehensive triangular decomposition (DSCTD) in order to classify the number of solutions depending on parameters. This is our second contribution.

Our third contribution is an implementation report of our algorithm computing CTDs, within the `RegularChains` library in Maple. We provide comparative benchmarks with Maple implementations of related methods for solving parametric polynomial systems, namely: *decompositions into regular systems* by Wang [126] and *discussing parametric Gröbner bases* by Montes [102]. We use a large set of well-known test-problems from the literature. Our implementation of the CTD algorithm can solve all problems which can be solved by the other methods. In addition, our CTD code can solve problems which are out of reach of the other two methods, generally due to memory consumption.

This chapter is based on paper [30], co-authored with Oleg Golubitsky, François Lemaire, Marc Moreno Maza and Wei Pan.

6.2 Decomposition into pairwise disjoint constructible sets

In this section we present two operations which decompose a list of regular systems into another list of regular systems whose zero sets are disjoint. In addition, the second operation computes an "intersection free basis" of a list of regular systems, which is applied to computing comprehensive triangular decompositions in the next section. The specification of the two operations are as follows.

The operation MPD. Given a list of regular systems $\mathcal{S}$ of $\mathbf{k}[\mathbf{x}]$, the operation MakePairwiseDisjoint (MPD for short) computes another list of regular systems $\mathcal{D}$ of $\mathbf{k}[\mathbf{x}]$ such that $\cup_{R\in\mathcal{S}}Z(R) = \uplus_{R\in\mathcal{D}}Z(R)$ hold, which $\uplus$ denotes a disjoint union.

The operation SMPD. Given a list of regular systems $R_1,\ldots,R_e$ of $\mathbf{k}[\mathbf{x}]$, the operation SymmetricallyMakePairwiseDisjoint (SMPD for short) computes another list of regular systems $S_1,\ldots,S_f$ of $\mathbf{k}[\mathbf{x}]$ such that the following hold:

- $\cup_{i=1}^{e}Z(R_i) = \uplus_{i=1}^{f}Z(S_i)$,
- each $Z(R_i)$ is a finite union of some of the $Z(S_j)$.

We call $S_1,\ldots,S_f$ an *intersection free basis* of $R_1,\ldots,R_e$.

Algorithm 18: MPD($\mathcal{S}$)

begin
  if $|\mathcal{S}| \le 1$ **then**
    output $\mathcal{S}$
  sort the regular systems in $\mathcal{S}$ by increasing rank
  let $\mathcal{S} = \mathcal{L} + \mathcal{R}$, where $|\mathcal{L}| = |\mathcal{R}|$ or $|\mathcal{L}| = |\mathcal{R}| + 1$
  $\mathcal{L}$:= DifferenceLR($\mathcal{L}, \mathcal{R}$)
  sort the regular systems in $\mathcal{L}$ by increasing rank
  output MPD($\mathcal{L}$)
  output MPD($\mathcal{R}$)
end

Definition 6.1. *Let $\mathcal{S}$ be a non-empty list of regular systems. Define $\mathcal{S}_r$ as the subset of regular systems of maximal rank in $\mathcal{S}$. Let $\phi(\mathcal{S}) = (\mathrm{rank}(\mathcal{S}), |\mathcal{S}_r|)$. Let $\mathcal{S}'$ be another non-empty list of regular systems. Let $\mathcal{S} \prec \mathcal{S}'$ if $\phi(\mathcal{S}) <_{\mathrm{lex}} \phi(\mathcal{S}')$, where $<_{\mathrm{lex}}$ is the lexicographic order. For the empty list [ ] and any non-empty list $\mathcal{S}$, we define $\phi([\ ]) \prec \phi(\mathcal{S})$. Clearly any sequence of $\phi(\mathcal{S})$ which is strictly decreasing w.r.t. $\prec$ is finite.*

Proposition 6.1. *Algorithm 18 terminates and satisfies its specifications.*

Proof. For empty and singleton lists $\mathcal{S}$, the proposition clearly holds. Let $\mathcal{S} = \mathcal{L} + \mathcal{R}$. By Theorem 5.1, we have $\phi(\mathsf{DifferenceLR}(\mathcal{L}, \mathcal{R})) \prec \phi(\mathcal{S})$ holds. On the other hand, $\phi(\mathcal{R}) \prec \phi(\mathcal{S})$ clearly holds. Thus the algorithm terminates. Its correctness is obvious. □

Algorithm 19: $\mathsf{SMPD}(\mathcal{S})$

begin
  if $|\mathcal{S}| \leq 1$ **then**
    output $\mathcal{S}$
  Let $[T_0, h_0] \in \mathcal{S}$, $\mathcal{S} \leftarrow \mathcal{S} \setminus \{[T_0, h_0]\}$
  $\mathcal{S} \leftarrow \mathsf{SMPD}(\mathcal{S})$
  for $[T, h] \in \mathcal{S}$ **do**
    $\mathcal{A} \leftarrow \mathsf{Difference}([T, h], [T_0, h_0])$
    $\mathcal{B} \leftarrow \mathsf{DifferenceLR}([T, h], \mathcal{A})$
    output $\mathsf{MPD}(\mathcal{A})$
    output $\mathsf{MPD}(\mathcal{B})$
  $\mathcal{C} \leftarrow \mathsf{DifferenceLR}([T_0, h_0], \mathcal{S})$
  output $\mathsf{MPD}(\mathcal{C})$
end

Proposition 6.2. *The Algorithm* SMPD *terminates and is correct.*

Proof. It follows directly from the termination and correctness of algorithms Difference, DifferenceLR and MPD. □

Remark 6.1. *In the rest of this thesis, we also use* SMPD *to denote an operation for computing intersection free basis of a set of constructible sets. More precisely, given a set of constructible sets $A_1, \cdots, A_s$,* SMPD *computes another set of constructible sets $B_1, \cdots, B_t$ whose zero sets are pairwise disjoint, such that each $Z(A_i)$ writes as a union of some of the $Z(B_1), \cdots, Z(B_t)$. This operation can be implemented by a similar algorithm as Algorithm 19.*

6.3 Comprehensive triangular decomposition of a parametric algebraic variety

In this section we introduce the concept of comprehensive triangular decomposition of an algebraic variety. We propose an algorithm for computing this decomposition and apply it to computing the set of all parameter values at which a given parametric system has an empty or an infinite set of solutions.

From now on, we assume that $n = m + d$, the variables $x_1, \ldots, x_d$ are renamed $u_1, \ldots, u_d$ and viewed as parameters, whereas $x_{d+1}, \ldots, x_n$ are renamed $y_1, \ldots, y_m$ and regarded as unknowns.

Let $T_{\mathbf{u}}$ (resp. $T_{\mathbf{y}}$) denote the set of polynomials in T whose main variables belong to $\mathbf{u}$ (resp. $\mathbf{y}$). That is $T_{\mathbf{u}} = T \cap \mathbf{k}[\mathbf{u}]$ and $T_{\mathbf{y}} = T \setminus T_{\mathbf{u}}$. Let $W^{\mathbf{u}}(T_{\mathbf{u}})$ be the quasi-component of $T_{\mathbf{u}}$ in $\mathbf{K}^d$.

Let $p \in \mathbf{k}[\mathbf{u}, \mathbf{y}]$. Denote by $\text{coeffs}(p, \mathbf{y})$ the set of coefficients of p w.r.t. the variables $\mathbf{y}$. Let $V^{\mathbf{u}}(\text{coeffs}(p, \mathbf{y}))$ be the algebraic variety of $\text{coeffs}(p, \mathbf{y})$ in $\mathbf{K}^d$. For $u \in \mathbf{K}^d$, we denote by $p(u)$ the polynomial of $\mathbf{K}[\mathbf{y}]$ obtained by evaluating p at $\mathbf{u} = u$. Clearly, for all $u \in \mathbf{K}^d$, the polynomial $p(u)$ is identically null iff $u \in V^{\mathbf{u}}(\text{coeffs}(p, \mathbf{y}))$. Let $F \subset \mathbf{k}[\mathbf{u}, \mathbf{y}]$. Then, we denote by $F(u)$ the set of all non-zero $p(u)$ for $p \in F$.

Defining set. Let $T \subset \mathbf{k}[\mathbf{u}, \mathbf{y}]$ be a regular chain. Let $u \in \mathbf{K}^d$. We say that T *specializes well* at u if $T(u)$ is a regular chain of $\mathbf{K}[\mathbf{y}]$ and $h_T(u) \neq 0$. The union of all these parameter values is called the *defining set* of T w.r.t. $\mathbf{u}$, denoted by $D^{\mathbf{u}}(T)$.

Lemma 6.1. *Let $T \subset \mathbf{k}[\mathbf{u}, \mathbf{y}]$ be a regular chain with* $\text{mvar}(T) \subseteq \mathbf{y}$ *and let $u \in \mathbf{K}^d$. We have*

$$u \notin V^{\mathbf{u}}(\text{res}(h_T, T)) \iff h_T(u) \neq 0 \text{ and } \text{res}(h_{T(u)}, T(u)) \neq 0.$$

Proof. We prove the lemma by induction. If $|T| = 1$, we have $\text{res}(h_T, T) = h_T$. So $u \notin V^{\mathbf{u}}(\text{res}(h_T, T))$ implies $h_T(u) \neq 0$ and therefore $\text{res}(h_{T(u)}, T(u)) = h_{T(u)} = h_T(u) \neq 0$. The other direction is obvious.

Now we assume that the conclusion holds for $|T| = s - 1$. If $|T| = s$, let v be the largest variable in $\text{mvar}(T)$. If $u \notin V^{\mathbf{u}}(\text{res}(h_T, T))$, we have $\text{res}(h_T, T)(u) = \text{res}(h_T, T_{<v})(u) \neq 0$. Therefore, $\text{res}(h_{T_{<v}}, T_{<v})(u) \neq 0$. By induction hypothesis, we know $h_{T_{<v}}(u) \neq 0$. By the specialization property of subresultants, one can deduce that $\text{res}(h_T, T_{<v})(u)$ and $\text{res}(h_T(u), T_{<v}(u))$ differ by a nonzero polynomial in $\mathbf{K}[\mathbf{y}]$. Thus we have $\text{res}(h_T(u), T_{<v}(u)) \neq 0$ holds. So $h_T(u) \neq 0$ holds. Thus we have $h_{T(u)} = h_T(u)$. Therefore $\text{res}(h_{T(u)}, T(u)) = \text{res}(h_{T(u)}, T_{<v}(u)) = \text{res}(h_T(u), T_{<v}(u)) \neq 0$ also holds. Another direction follows from similar arguments. □

Proposition 6.3. *Let $T \subset \mathbf{k}[\mathbf{u}, \mathbf{y}]$ be a regular chain. Let $D^{\mathbf{u}}(T)$ be the defining set of T w.r.t. $\mathbf{u}$. Then we have $D^{\mathbf{u}}(T) = W^{\mathbf{u}}(T_{\mathbf{u}}) \setminus V^{\mathbf{u}}(\text{res}(h_{T_{\mathbf{y}}}, T_{\mathbf{y}}))$.*

Proof. Assume that $u \in W^{\mathbf{u}}(T_{\mathbf{u}}) \setminus V^{\mathbf{u}}(\text{res}(h_{T_{\mathbf{y}}}, T_{\mathbf{y}}))$. We prove that T specializes well at u. From Lemma 6.1 we have $\text{res}(h_{T_{\mathbf{y}}(u)}, T_{\mathbf{y}}(u)) \neq 0$ and $h_{T_{\mathbf{y}}}(u) \neq 0$. Since

$u \in W^{\mathbf{u}}(T_{\mathbf{u}})$, we have $T_{\mathbf{u}}(u) = \emptyset$ and $h_{T_{\mathbf{u}}}(u) \neq 0$. So we have $h_T(u) \neq 0$. Moreover, by Proposition 4.2, $T(u) = T_{\mathbf{y}}(u)$ is a regular chain. Therefore, the regular chain T specializes well at u. The converse implication is proved similarly. □

Remark 6.2. *Since $D^{\mathbf{u}}(T)$ is a constructible set, by Lemma 5.2, there exists an algorithm to compute a set of regular systems $\mathcal{R}^{\mathbf{u}}(T)$, such that $D^{\mathbf{u}}(T) = Z(\mathcal{R}^{\mathbf{u}}(T))$.*

Definition 6.2. *Let $T \subset \mathbf{k}[\mathbf{u}, \mathbf{y}]$ be a regular chain. The* comprehensive quasi-component of T w.r.t. $\mathbf{u}$*, denoted by $W_C(T)$, is defined by $W_C(T) = W(T) \cap \pi_{\mathbf{u}}^{-1}(D^{\mathbf{u}}(T))$.*

Proposition 6.4. *Let $T \subset \mathbf{k}[\mathbf{u}, \mathbf{y}]$ be a regular chain. The following properties hold:*

(1) *We have: $W_C(T) = W(T) \setminus \pi_{\mathbf{u}}^{-1}(V^{\mathbf{u}}(\mathrm{res}(h_{T_{\mathbf{y}}}, T_{\mathbf{y}})))$.*

(2) *We have: $\pi_{\mathbf{u}}(W_C(T)) = D^{\mathbf{u}}(T)$.*

Proof. It follows directly from Proposition 6.3. □

Definition 6.3. *Let $F \subset \mathbf{k}[\mathbf{u}, \mathbf{y}]$ be a finite polynomial set. A* comprehensive triangular decomposition *(CTD) of $V(F)$ is given by :*

1. *a finite partition $\mathcal{C}$ of $\pi_{\mathbf{u}}(V(F))$,*
2. *for each $C \in \mathcal{C}$ a set of regular chains $\mathcal{T}_C$ of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$ such that for $u \in C$ each of the regular chains $T \in \mathcal{T}_C$ specializes well at u and we have for all $u \in C$*

$$V(F(u)) = \bigcup_{T \in \mathcal{T}_C} W(T(u)).$$

We will compute the above comprehensive triangular decomposition with the help of the following auxiliary concept:

Definition 6.4. *Let $F \subset \mathbf{k}[\mathbf{u}, \mathbf{y}]$ be a finite polynomial set. A* pre-comprehensive triangular decomposition *(PCTD) of $V(F)$ is a family of regular chains $\mathcal{T}$ satisfying the following property: for each $u \in \mathbf{K}^d$, let $\mathcal{T}_u$ be the subfamily of all regular chains in $\mathcal{T}$ that specialize well at u; then $V(F(u)) = \bigcup_{T \in \mathcal{T}_u} W(T(u))$.*

Proposition 6.5. *Let $F \subset \mathbf{k}[\mathbf{u}, \mathbf{y}]$ be a finite polynomial set. A triangular decomposition $\mathcal{T}$ of $V(F)$ is a pre-comprehensive triangular decomposition if and only if $V(F) = \bigcup_{T \in \mathcal{T}} W_C(T)$.*

Algorithm 20: $\mathsf{PCTD}(F)$

Input: A finite set $F \subset \mathbf{k}[\mathbf{u}, \mathbf{y}]$.
Output: A PCTD of $V(F)$.

```
1 𝒯 ← Triangularize(F)
2 while 𝒯 ≠ ∅ do
3     let T ∈ 𝒯,  𝒯 ← 𝒯 \ {T}
4     output T
5     G ← COEFFICIENTS(res(h_{T_y}, T_y), u)
6     𝒯 ← 𝒯 ∪ Triangularize(G, T)
```

Proof. It follows from the definition of $W_C(T)$, Proposition 6.3 and the definition of pre-comprehensive triangular decomposition. □

Proposition 6.6. *Algorithm 20 computes a pre-comprehensive triangular decomposition of $V(F)$.*

Proof. The loop satisfies the following invariant: the union of all $W(T)$, where T ranges over $\mathcal{T}$, and of the $W(T')$, where T' ranges over the current output, equals $V(F)$. Indeed, the invariant holds at the beginning, when the output is empty; and for the regular chain T taken from $\mathcal{T}$ at the current iteration, we have $W(T) \setminus W_C(T) = V(G) \cap W(T)$ by Proposition 6.4 (1). Then, correctness of the algorithm follows from Proposition 6.5 and the fact that at the end $\mathcal{T} = \varnothing$.

Since polynomials in G do not involve the main variables of T, by Lemma 2.2 they are regular w.r.t $\operatorname{sat}(T)$. Then by Lemma A.1, either the output of $\mathsf{Triangularize}(G, T)$ is empty or the dimensions of the regular chains computed by $\mathsf{Triangularize}(G, T)$ are strictly less than that of T. Therefore, the algorithm terminates. □

Proposition 6.7. *Algorithm 21 computes a comprehensive triangular decomposition of $F \subset \mathbf{k}[\mathbf{u}, \mathbf{y}]$.*

Proof. Let $\mathcal{T}$ be the output of $\mathsf{PCTD(F)}$. By Proposition 6.5 and Proposition 6.4 (2), we have

$$\pi_{\mathbf{u}}(V(F)) = \bigcup_{T \in \mathcal{T}} D^{\mathbf{u}}(T).$$

Then the conclusion follows from the definition of comprehensive triangular decomposition, Proposition 6.2, 6.6 and Remark 6.2. □

Given a polynomial set $F \subset \mathbf{k}[\mathbf{u}, \mathbf{y}]$, a natural question is to describe the points u of $\mathbf{K}^d$ for which the specialized system $F(u)$ admits solutions, finitely many or infinitely many solutions.

Algorithm 21: CTD(F)

Input: A finite set $F \subset \mathbf{k}[\mathbf{u}, \mathbf{y}]$.
Output: A CTD of $V(F)$.

$\mathcal{T} \leftarrow$ PCTD(F);
$\mathcal{S} \leftarrow \emptyset$;
for $T \in \mathcal{T}$ **do**
  $\mathcal{S} \leftarrow \mathcal{S} \cup \mathcal{R}^{\mathbf{u}}(T)$;
$\mathcal{S} \leftarrow$ SMPD($\mathcal{S}$);
while $\mathcal{S} \neq \emptyset$ **do**
  let $C \in \mathcal{S}$, $\mathcal{S} \leftarrow \mathcal{S} \setminus C$;
  $\mathcal{T}_C \leftarrow$ regular chains in $\mathcal{T}$ associated to C;
  output $(C, \mathcal{T}_C)$;

Theorem 6.1. *Let $\mathcal{T}$ is a pre-comprehensive triangular decomposition of $V(F)$. Denote by $\mathcal{T}_0 \subseteq \mathcal{T}$ and $\mathcal{T}_1 \subseteq \mathcal{T}$ respectively the set of regular chains T with $\mathbf{y} \subseteq \mathrm{mvar}(T)$ and $\mathbf{y} \not\subseteq \mathrm{mvar}(T)$. Then for any $u \in \mathbf{K}^d$, we have*

(i) The system $F(u)$ has solutions in $\mathbf{K}^m$ if and only if $u \in \cup_{T \in \mathcal{T}} D^{\mathbf{u}}(T)$.

(ii) The system $F(u)$ has infinitely many solutions in $\mathbf{K}^m$ if and only if $u \in \cup_{T \in \mathcal{T}_1} D^{\mathbf{u}}(T)$.

(iii) The system $F(u)$ has finitely many solutions in $\mathbf{K}^m$ if and only if $u \in \cup_{T \in \mathcal{T}_0} D^{\mathbf{u}}(T) \setminus \cup_{T \in \mathcal{T}_1} D^{\mathbf{u}}(T)$.

Proof. It follows directly from Proposition 6.4 and the definition of a pre-comprehensive triangular decomposition. □

Definition 6.5. *The* discriminant set *of F is defined as the set of all points $u \in \mathbf{K}^d$ for which $V(F(u))$ is empty or infinite.*

Remark 6.3. *By Theorem 6.1, the discriminant set of F can be computed directly from a pre-comprehensive triangular decomposition of $V(F)$.*

Proposition 6.8. *Let T be a regular chain of $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$ such that $\mathrm{mvar}(T) = \mathbf{y}$. Let C be a connected subset of $\mathbb{C}^d$ such that T specializes well at every $u \in C$. Then the roots of T in $\mathbf{y}$ are continuous functions of $\mathbf{u}$ above C.*

Proof. We prove the proposition by induction on m. If $m = 1$, for any $u \in C$, since T specializes well at u, we have $h_T(u) \neq 0$. By Theorem $(1, 4)$ in [97], the roots of T in $\mathbf{y}$ are continuous functions of $\mathbf{u}$ above C.

Now assume the proposition holds for $m-1$. Since T specializes well at $u \in C$, we know that for any $(u, \alpha_1, \ldots, \alpha_{m-1})$ such that $T_{<y_m}(u, \alpha_1, \ldots, \alpha_{m-1}) = 0$, we have $\mathrm{init}(T_{y_m})(u, \alpha_1, \ldots, \alpha_{m-1}) \neq 0$ holds. Applying Theorem $(1, 4)$ in [97] again, the root of T_{y_m} in y_m are continuous functions $\mathbf{u}$ and $y_1, \ldots, y_{m-1}$ above $C \times (T_{<y_m} = 0)$. By induction and composition properties of continuous functions, we conclude that the roots of T in $\mathbf{y}$ are continuous functions of $\mathbf{u}$ above C. □

6.4 Comprehensive triangular decomposition of a parametric constructible set

In this section, we assume that a constructible set cs is represented by finitely many regular systems in $\mathbf{k}[\mathbf{u}, \mathbf{y}]$, where $\mathbf{u}$ are parameters and $\mathbf{y}$ are unknowns.

Let $R := [T, h]$ be a regular system of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$. Let $u \in \mathbf{K}^d$. We say that R *specializes well* at u if $R(u)$ is a regular system of $\mathbf{K}[\mathbf{y}]$ and $h_T(u) \neq 0$. This is equivalent to say that T specializes well at u and $h(u)$ is regular w.r.t. $\mathrm{sat}(T(u))$.

Definition 6.6. *Let cs be a constructible set of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$. A* comprehensive triangular decomposition *of cs is given by : (i) a finite partition $\mathcal{C}$ of $\pi_{\mathbf{u}}(cs)$; (ii) for each $C \in \mathcal{C}$ a set of regular systems $\mathcal{R}_C$ of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$ such that for $u \in C$ each of the regular systems $R \in \mathcal{R}_C$ specializes well at u and we have for all $u \in C$ $cs(u) = \bigcup_{R \in \mathcal{R}_C} Z(R(u))$.*

Similarly, we can define the defining set of regular system $R = [T, h]$ as the set of all parameter values u in $\mathbf{K}^d$ such that R specializes well at u.

Proposition 6.9. *We have $D^{\mathbf{u}}(R) := W^{\mathbf{u}}(T_{\mathbf{u}}) \setminus V^{\mathbf{u}}(\mathrm{coeffs}(\mathrm{res}(h_{T_{\mathbf{y}}} h, T), \mathbf{y}))$.*

Based on this proposition, one could easily derive similar algorithms for computing the CTD of a constructible sets. We omit here the details.

Definition 6.7. *Let $R := [T, h]$ be a squarefree regular system of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$. Let $u \in \mathbf{K}^d$. We say that R* specializes well *at u if $R(u)$ is a squarefree regular system of $\mathbf{K}[\mathbf{y}]$ and $h_T(u) \neq 0$. The set of all parameters $u \in \mathbf{K}^d$ such that R specializes well at u is called the* defining set *of R, denoted by $D^{\mathbf{u}}(R)$. Let $\mathcal{R} = \{R_1, \ldots, R_e\}$ be a finite set of regular systems of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$. We say that $\mathcal{R}$* specializes disjointly well *at u, if: (i) each $R \in \mathcal{R}$ specializes well at u and (ii) the zero sets of $R_i(u)$ in $\mathbf{K}^m$ are pairwise disjoint.*

Let $\uplus$ denote the disjoint union of two sets.

Definition 6.8. *Let cs be a constructible set of* $\mathbf{k}[\mathbf{u}, \mathbf{y}]$. *A* disjoint squarefree comprehensive triangular decomposition *(DSCTD) of cs is a pair* $(\mathcal{C}, (\mathcal{R}_C, C \in \mathcal{C}))$, *where* $\mathcal{C}$ *is a finite partition of* $\pi_{\mathbf{u}}(cs)$ *into nonempty constructible sets, and, for each* $C \in \mathcal{C}$, $\mathcal{R}_C$ *is a finite set of regular systems of* $\mathbf{k}[\mathbf{u}, \mathbf{y}]$ *such that for each point* $u \in C$ *the following conditions hold:*

(i) $\mathcal{R}_C$ *specializes disjointly well at* u*;*

(ii) *we have* $cs(u) = \uplus_{R \in \mathcal{R}_C} Z(R(u))$

Lemma 6.2. *Let* $R := [T, h]$ *be a squarefree regular system of* $\mathbf{k}[\mathbf{u}, \mathbf{y}]$. *Then we have* $D^{\mathbf{u}}(R) := W^{\mathbf{u}}(T_{\mathbf{u}}) \setminus V^{\mathbf{u}}(\mathrm{coeffs}(\mathrm{res}(\mathrm{sep}(T_{\mathbf{y}})h, T), \mathbf{y}))$.

The computation of DSCTD relies on the following concept.

Definition 6.9. *Let cs be a constructible set of* $\mathbf{k}[\mathbf{u}, \mathbf{y}]$. *A* disjoint squarefree pre-comprehensive triangular decomposition *(DSPCTD) of cs is a family of squarefree regular systems* $\mathcal{R}$ *satisfying the following property: for each* $u \in \mathbf{K}^d$, *let* $\mathcal{R}_u$ *be the subfamily of all regular systems in* $\mathcal{R}$ *that specialize well at* u*; then* $cs(u) = \uplus_{R \in \mathcal{R}_u} Z(R(u))$.

Algorithm 22 computes a DSPCTD of a constructible set. Algorithm 23 computes a DSCTD of a constructible set. The proof of the termination and correctness of the two algorithms are similar to that of the algorithm PCTD and CTD. The implementation of the algorithm DSCTD is available in the `RegularChains` library since `Maple13`. It sits inside the `ParametricSystemTool` module and is implemented as the command `ComprehensiveTriangularize` with option the `'disjoint'='yes'`.

Algorithm 22: DSPCTD(cs)

Input: A constructible set cs of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$.
Output: A DSPCTD of cs.
let $\mathcal{R}$ be the set of regular systems representing cs
$\mathcal{R} := \mathsf{MPD}(\mathcal{R})$; $\mathcal{R}' := \{\ \}$
while $\mathcal{R} \neq \{\ \}$ **do**
  let $R := [T, h] \in \mathcal{R}$; $\mathcal{R} := \mathcal{R} \setminus \{R\}$
  $\mathcal{R}' := \mathcal{R}' \cup \{R\}$
  $G := \mathrm{coeffs}(\mathrm{res}(\mathrm{sep}(T_{\mathbf{y}})h, T_{\mathbf{y}}), \mathbf{y})$
  $\mathcal{R} := \mathcal{R} \cup \mathsf{MPD}(\mathsf{Intersect}(G, R))$
return $\mathcal{R}'$;

Let cs be a constructible set of $\mathbf{K}^n$. Often, we only need to partition the parameter space into constructible sets, called cells, such that above each cell:

Algorithm 23: DSCTD(cs)

Input: A constructible set cs of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$.

Output: A DSCTD of cs.

$\mathcal{R} :=$ DSPCTD(cs)
$\mathcal{C} := \{ \}$
for $R \in \mathcal{R}$ **do**
  $\mathcal{C} := \mathcal{C} \cup \{D^{\mathbf{u}}(R)\}$
$\mathcal{C} :=$ SMPD($\mathcal{C}$)
for $C \in \mathcal{C}$ **do**
  let $\mathcal{R}_C$ be the set of regular systems $R \in \mathcal{R}$ with $C \subseteq D^{\mathbf{u}}(R)$
return $(\mathcal{C}, (\mathcal{R}_C, C \in \mathcal{C}))$

1. either cs has no solutions;
2. or cs has infinitely many solutions;
3. or cs has a constant number of solutions and such that the solutions are continuous functions of the parameters above the connected component of each cell.

A precise definition of this idea is stated in Definition 6.10.

Definition 6.10. *Let cs be a constructible set of $\mathbf{K}^n$. A weak DSCTD (WDSCTD) of cs is a pair $(\mathcal{C}, (\mathcal{T}_C, C \in \mathcal{C}))$, where*

- *$\mathcal{C}$ is a finite partition of $\mathbf{K}^d$ into nonempty constructible sets,*
- *for each $C \in \mathcal{C}$, $\mathcal{T}_C$ is a finite set of regular chains of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$ such that:*
  - *(i) either $\mathcal{T}_C$ is empty, which means that $cs(u)$ is empty for each $u \in C$*
  - *(ii) or $\mathcal{T}_C = \{\varnothing\}$, which means that $cs(u)$ is infinite for each $u \in C$;*
  - *(iii) or each $T \in \mathcal{T}_C$ satisfies $\mathrm{mvar}(T) = \mathbf{y}$ and for each $u \in C$, $\mathcal{T}_C$ specializes disjointly well at u and $cs(u) = \cup_{T \in \mathcal{T}_C} Z(T(u))$.*

Algorithm 24 computes a WDSCTD of cs. It is not difficult to prove the termination and the correctness of this algorithm.

6.5 Complex root classification

We restrict now to the case where $\mathbf{k}$ (and thus $\mathbf{K}$) is the field $\mathbb{C}$ of complex numbers. The algorithm WDSCTD immediately suggests a solution to the following complex root classification problem. Let cs be a parametric constructible set of $\mathbb{C}[\mathbf{u}, \mathbf{y}]$. A *complex root classification* of cs is a finite set of pairs $\{(C_1, n_1), \ldots, (C_s, n_s)\}$ such that

Algorithm 24: WDSCTD(cs)

Input: A constructible set cs of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$.
Output: A WDSCTD of cs.

let $\mathcal{R}$ be the set of regular systems representing cs
let $\mathcal{R}_0$ (resp. $\mathcal{R}_1$) be the set of regular systems $[T, h]$ in $\mathcal{R}$ such that $\mathbf{y} \subseteq \text{mvar}(T)$ (resp. $\mathbf{y} \not\subseteq \text{mvar}(T)$)
let $(\mathcal{C}, (\mathcal{R}_C, C \in \mathcal{C}))$ be a DSCTD of $\mathcal{R}_0$
let $\mathcal{E}_1$ be the projection of the constructible set $\mathcal{R}_1$ on $\mathbf{K}^d$
$\mathcal{D} := \{\ \}$
if $\mathcal{E}_1$ *is not empty* **then**
    $D := \mathcal{E}_1$; $\mathcal{T}_D := \{\varnothing\}$; $\mathcal{D} := \mathcal{D} \cup \{D\}$
for $C \in \mathcal{C}$ **do**
    $D :=$ Difference$(C, \mathcal{E}_1)$
    if D *is not empty* **then**
        $\mathcal{T}_D := \{T_\mathbf{y} \mid [T, h] \in \mathcal{R}_C\}$; $\mathcal{D} := \mathcal{D} \cup \{D\}$
$D :=$ Difference$(\mathbf{K}^m, \cup_{D \in \mathcal{D}} D)$
if D *is not empty* **then**
    $\mathcal{D} := \mathcal{D} \cup \{D\}$; $\mathcal{T}_D := \{\ \}$
return $(\mathcal{D}, (\mathcal{T}_D, D \in \mathcal{D}))$

(i) each C_i is a non-empty constructible set of $\mathbb{C}[\mathbf{u}]$ and $\mathbb{C}^d = \uplus_{i=1}^s C_i$ holds,

(ii) each n_i is either ∞ or a nonnegative integer and the n_i's are pairwise distinct,

(iii) for any $u \in C_i$, the distinct number of complex solutions of $cs(u)$ in $\mathbb{C}^m$ is n_i.

Notation 6.1. *For a squarefree regular chain T of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$, define*

$$\deg(T) = \begin{cases} \prod_{v \in \mathbf{y}} \text{mdeg}(T_v), & \text{if } \ \mathbf{y} \subseteq \text{mvar}(T) \\ \infty, & \text{otherwise.} \end{cases}$$

For a collection of squarefree regular chains $\mathcal{T}$, define $\deg(\mathcal{T}) = \sum_{T \in \mathcal{T}} \deg(T)$.

Proposition 6.10. *Let T be a squarefree regular chain of* $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$, *with* $\text{mvar}(T) = \mathbf{y}$. *Let C be a connected subset of in $\mathbb{C}^d$. Let $k = \deg(T)$. Assume that T specializes well at any $\alpha \in C$. Then there exist k continuous functions $\psi_1(\mathbf{u}), \ldots, \psi_k(\mathbf{u})$ defined on C, such that $W(T) = \uplus_{i=1}^k \{(\alpha, \psi_i(\alpha)) \mid \alpha \in C\}$ holds, where $\uplus$ denotes a disjoint union. In particular, for each $\alpha \in C$, we have $W(T(\alpha)) = \{\psi_1(\alpha), \ldots, \psi_k(\alpha)\}$, which is a set of k points in $\mathbb{C}^m$.*

Proof. It follows from Proposition 6.8 and the fact that $T(u)$ has k distinct roots in $\mathbb{C}^m$ for any $u \in C$. $\square$

The following algorithm computes a complex root classification of cs w.r.t. $\mathbf{u}$, whose correctness is easily derived from the specification of WDSCTD.

Algorithm 25: ComplexRootClassificaition(cs)

Input: A parametric constructible set cs of $\mathbb{Q}[\mathbf{u}, \mathbf{x}]$
Output: A complex root classification of cs w.r.t. $\mathbf{u}$

begin
  $(\mathcal{C}, (\mathcal{T}_C, C \in \mathcal{C})) :=$ WDSCTD(cs);
  for $C \in \mathcal{C}$ **do** $n_C := \deg(\mathcal{T}_C)$;
  for *each distinct* n_i *in* $\{n_C \mid C \in \mathcal{C}\}$ **do**
    let C_i be the union of C such that $n_C = n_i$;
    output (C_i, n_i)
end

6.6 Defining sets, border polynomials, discriminant sets and discriminant varieties

In this section we investigate the relations between the notions of defining set, border polynomial, discriminant set and minimal discriminant variety. In this section, we fix $\mathbf{k} = \mathbb{Q}$.

Discriminant variety [86]. Let cs be a basic constructible set of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$. Let δ be the dimension of $\overline{\pi_{\mathbf{u}}(cs)}$. An algebraic variety W is a discriminant variety of cs w.r.t $\pi_{\mathbf{u}}$ if and only if:

- W is contained in $\overline{\pi_{\mathbf{u}}(cs)}$,
- $W = \overline{\pi_{\mathbf{u}}(cs)}$ if and only $cs(u)$ is infinite for almost all $u \in \overline{\pi_{\mathbf{u}}(cs)}$,
- the connected components $C_1, \cdots, C_k$ of $\overline{\pi_{\mathbf{u}}(cs)} \setminus W$ are analytic submanifolds of dimension δ,
- $(\pi_{\mathbf{u}}^{-1}(C_i) \cap cs, \pi_{\mathbf{u}})$ is an analytic covering of C_i, for $i = 1, \cdots, k$. In another words, for each connected component C_i, there exist a finite set of indexes $\mathcal{I}$ and disjoint connected subsets $(\mathcal{V}_j)_{j \in \mathcal{I}}$ of cs such that $\pi_{\mathbf{u}}^{-1}(C_i) \cap cs = \cup_{j \in \mathcal{I}} \mathcal{V}_j$. Moreover $\pi_{\mathbf{u}}$ is a local diffeomorphism from $\mathcal{V}_i$ onto C_i.

Proposition 6.11. *Given a basic constructible set cs of* $\mathbf{k}[\mathbf{u}, \mathbf{y}]$*, denote by D be the discriminant set of cs. Then for any discriminant variety W of cs, we have*

$$D \cap \overline{\pi_{\mathbf{u}}(cs)} \subseteq W.$$

Proof. If $W = \overline{\pi_{\mathbf{u}}(cs)}$, the conclusion holds immediately. Otherwise, we have $W \subsetneq \overline{\pi_{\mathbf{u}}(cs)}$. By the definition of discriminant variety, for any $u \in \overline{\pi_{\mathbf{u}}(cs)} \setminus W$, the set $cs(u)$ is finite and nonempty. By the definition of discriminant set, for any $u \in D$, the set $cs(u)$ is infinite or empty. Therefore $D \cap (cs(u) \setminus W) = \varnothing$, which implies that $D \cap \overline{\pi_{\mathbf{u}}(cs)} \subseteq W$. □

Border polynomial [138]. Let $rs := [T, h]$ be a squarefree regular system of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$ with $\mathrm{mvar}(T) = \mathbf{y}$. Let bp be the primitive and square free part of the product of all $\mathrm{res}(\mathrm{der}(t), T)$ and $\mathrm{res}(h, T)$. We call bp the *border polynomial* of $[T, h]$.

Proposition 6.12. *Let $rs := [T, h]$ be a squarefree regular system of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$ with $\mathrm{mvar}(T) = \mathbf{y}$. Let bp be the border polynomial of rs. Let $D^{\mathbf{u}}(rs)$ be the defining set of rs. We have $\mathbf{K}^d \setminus V^{\mathbf{u}}(bp) = D^{\mathbf{u}}(rs)$.*

Proof. It follows directly from Lemma 6.2. □

Proposition 6.13. *Let $rs := [T, h]$ be a squarefree regular system of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$ with $\mathrm{mvar}(T) = \mathbf{y}$. Let bp be the border polynomial of rs. Then $V^{\mathbf{u}}(bp)$ is a discriminant variety of both $Z(rs)$ and $V(T) \setminus V(h)$.*

Proof. In the following, we first prove that $V^{\mathbf{u}}(bp)$ is a discriminant variety of $Z(rs)$. Then the fact that $V^{\mathbf{u}}(bp)$ is a discriminant variety of $V(T) \setminus V(h)$ follows directly from the fact that $Z(rs) \setminus V(bp) = V(T) \setminus V(h) \setminus V(bp)$.

Since $V^{\mathbf{u}}(bp)$ is a hypersurface in $\mathbf{K}^d$, we have $\overline{D^{\mathbf{u}}(rs)} = \overline{\mathbf{K}^d \setminus V^{\mathbf{u}}(bp)} = \mathbf{K}^d$. On the other hand $D^{\mathbf{u}}(rs) \subseteq \pi_{\mathbf{u}}(Z(rs))$, which implies that $\overline{\pi_{\mathbf{u}}(Z(rs))} = \mathbf{K}^d$ and thus $\dim\left(\overline{\pi_{\mathbf{u}}(Z(rs))}\right) = d$.

Let $C_1, \ldots, C_k$ be the connected components of $\mathbf{K}^d \setminus V^{\mathbf{u}}(bp)$, clearly they are analytic submanifolds of dimension d. Next we prove that $(\pi_{\mathbf{u}}^{-1}(C_i) \cap Z(rs), \pi_{\mathbf{u}})$ is an analytic covering of C_i, for $i = 1, \ldots, k$. Or equivalently we prove that for each connected component C:

- (i) there exists a finite set of indexes $\mathcal{I}$ and disjoint connected subsets $(\mathcal{V}_i)_{i \in \mathcal{I}}$ of $Z(rs)$ such that $\pi_{\mathbf{u}}^{-1}(C) \cap Z(rs) = \bigcup_{i \in \mathcal{I}} \mathcal{V}_i$,
- (ii) $\pi_{\mathbf{u}}$ is a local diffeomorphism from $\mathcal{V}_i$ onto C.

Since rs specializes well at all $u \in C$, $T(u)$ is a zero-dimensional squarefree regular chain, which implies that $\pi_{\mathbf{u}}^{-1}(C) \cap Z(rs) = \pi_{\mathbf{u}}^{-1}(C) \cap W(T)$ holds. Since the number of distinct complex roots of T are constant above C and they are continuous functions of $\mathbf{u}$ above C with disjoint graphs, (i) holds. On the other hand, the Jacobian determinant $\mathrm{D}(T) = \prod_{t \in T} \mathrm{der}(t)$, therefore $\mathrm{D}(T)$ does not vanish above C, which implies that (ii) holds. In conclusion, $V^{\mathbf{u}}(bp)$ is a discriminant variety of $Z(rs)$. □

Theorem 6.2. *The variety $V^{\mathbf{u}}(bp)$ is the minimal discriminant variety of $Z(rs)$ and $V(T) \setminus V(h)$.*

Proof. We first prove that $V^{\mathbf{u}}(bp)$ is the minimal discriminant variety of $Z(rs)$. We prove by contradiction. Assume that $V^{\mathbf{u}}(bp)$ is not a minimal discriminant variety of $Z(rs)$. Let W be the minimal discriminant variety of $Z(rs)$. Then there exists $\alpha \in \mathbf{K}^d$ such that $\alpha \notin W$ and $\alpha \in V^{\mathbf{u}}(bp)$.

Since $\alpha \in V^{\mathbf{u}}(bp)$, rs does not specializes well at u, which leads to the following case discussion

(i) If $h_T(\alpha) = 0$, then $Z(rs(\alpha)) = \varnothing$.

(ii) If $h_T(\alpha) \neq 0$, but $rs(\alpha)$ is not a squarefree regular system, then $\#(R(\alpha)) < \deg(T)$.

On the other hand, let $\beta \notin V^{\mathbf{u}}(bp)$, we have $\#(rs(\beta)) = \deg(T)$. This is a contradiction to the fact that $\mathbf{K}^d \setminus W$ has only one connected component (This is true because all open Zariski sets in $\mathbf{K}^d$ have intersection).

The above proof can be also applied to $V(T) \setminus V(h)$, but noticing that in (i), if $h_T(\alpha) = 0$, then $V(T(\alpha))$ is either empty or infinite. □

Proposition 6.13 and Theorem 6.2 were also independently established in [136].

6.7 Implementation

We have implemented the algorithm for computing comprehensive triangular decompositions (CTD) based on *RegularChains* library in Maple 11. Our main function CTD calls essentially three functions

- Triangularize, computing a triangular decomposition of the input system F,
- PCTD, deducing a pre-comprehensive triangular decomposition of F,
- SMPD, obtaining a comprehensive triangular decomposition of F.

We provide comparative benchmarks with Maple implementations of related methods for solving parametric polynomial systems, namely: *decomposition into regular systems* by Wang [126] and *discussing parametric Gröbner bases* by Montes [102]. Corresponding Maple functions are RegSer and DISPGB, respectively.

Note that the specifications of these three methods are different. The outputs of CTD and DISPGB depend on the choice of the parameter sets, whereas RegSer does not require to specify parameters. RegSer decomposes the input system into pairwise disjoint constructible sets given by regular systems. CTD computes a comprehensive triangular decomposition, and thus a family of triangular decompositions with a partition of the parameter space. DISPGB computes a family of comprehensive Gröbner bases with a partition of the parameter space.

We run CTD in Maple 11 using an Intel Pentium 4 processor (3.20GHz CPU, 2.0GB total memory, and Red Hat 4.0.0-9); we set the time-out to 1 hour. Due to the current availability of RegSer and DISPGB, the timings obtained by these two functions are performed in Maple 8 on Intel Pentium 4 machines (1.60GHz CPU, 513MB memory and Red Hat Linux 3.2.2-5); and the time-out is 2 hours. The 30 test-systems used in our experimentation are chosen from [96, 118, 127].

As shown in the following three tables, our implementation of the CTD algorithm can solve all problems which can be solved by the other methods. In addition, the CTD in Maple 11 (Maple 15) can solve 4 (6) test-systems which are out of reach of the other two methods, generally due to memory consumption.

6.8 Conclusion

Comprehensive triangular decomposition is a powerful tool for the analysis of parametric polynomial systems: its purpose is to partition the parameter space into regions, so that within each region the "geometry" of the algebraic variety of the specialized system is the same for all values of the parameters.

As one of the main technical tools, we proposed an algorithmic solution for a set theoretical instance of the coprime factorization problem: refining a family of constructible sets into a family of pairwise disjoint constructible sets.

We have reported on an implementation of our algorithm computing CTDs, based on the `RegularChains` library in MAPLE. Our comparative benchmarks, with MAPLE implementations of related methods for solving parametric polynomial systems, illustrate the good performances of our CTD code.

Sys	Name	Triangularize	PCTD	SMPD	CTD	#Cells
1	MontesS1	0.089	0.002	0.031	0.122	3
2	MontesS2	0.031	0.002	0	0.033	1
3	MontesS3	0.103	0.006	0.005	0.114	2
4	MontesS4	0.101	0.016	0	0.117	1
5	MontesS5	0.383	0.022	0.465	0.870	11
6	MontesS6	0.395	0.019	0.121	0.535	4
7	MontesS7	0.416	0.215	0.108	0.739	4
8	MontesS8	0.729	0.001	0.016	0.746	2
9	MontesS9	0.945	0.116	3.817	4.878	23
10	MontesS10	5.325	0.684	1.138	7.147	10
11	MontesS11	0.757	0.208	12.302	13.267	28
12	MontesS12	14.199	2.419	10.114	26.732	10
13	MontesS13	0.415	0.143	1.268	1.826	9
14	MontesS14	41.167	31.510	0.303	72.980	4
15	MontesS15	6.919	0.579	1.123	8.621	5
16	MontesS16	6.963	0.083	2.407	9.453	21
17	AlkashiSinus	0.716	0.191	0.574	1.481	6
18	Bronstein	2.526	0.017	0.548	3.091	6
19	Gerdt	3.863	0.006	0.733	4.602	5
20	Hereman-2	1.826	0.019	0.020	1.865	2
21	Lanconelli	2.056	0.336	3.430	5.822	14
22	genLinSyst-3-2	1.624	0.275	25.413	27.312	32
23	genLinSyst-3-3	9.571	1.824	1097.291	1108.686	116
24	Wang93	6.795	37.232	11.828	55.855	8
25	Maclane	12.955	0.403	54.197	67.555	21
26	Neural	15.279	19.313	0.530	35.122	4
27	Leykin-1	1261.751	86.460	27.180	1375.391	57
28	Lazard-ascm2001	60.698	2817.801	–	–	–
29	Pavelle	–	–	–	–	–
30	Cheaters-homotopy	–	–	–	–	–

Table 6.1: Solving timings and number of cells of CTD (Maple 11)

Sys	Name	Triangularize	PCTD	SMPD	CTD	#Cells
1	MontesS1	0.360	0.104	0.108	0.572	2
2	MontesS2	0.268	0.080	0.004	0.352	1
3	MontesS3	0.376	0.096	0.100	0.572	2
4	MontesS4	0.356	0.104	0.004	0.464	1
5	MontesS5	0.580	0.124	0.296	1.000	8
6	MontesS6	0.464	0.144	0.144	0.752	3
7	MontesS7	0.616	0.184	0.148	0.948	4
8	MontesS8	0.584	0.084	0.092	0.760	2
9	MontesS9	0.840	0.160	0.656	1.656	13
10	MontesS10	1.196	0.232	0.204	1.632	6
11	MontesS11	0.660	0.256	0.424	1.340	11
12	MontesS12	3.316	0.752	0.408	4.476	5
13	MontesS13	0.612	0.164	0.328	1.104	9
14	MontesS14	1.096	0.208	0.144	1.448	2
15	MontesS15	2.284	0.288	0.220	2.792	5
16	MontesS16	4.172	0.188	0.524	4.884	8
17	AlkashiSinus	0.848	0.148	0.204	1.200	6
18	Bronstein	0.660	0.116	0.260	1.036	7
19	Gerdt	2.292	0.092	0.172	2.556	4
20	Hereman-2	1.192	0.108	0.132	1.432	2
21	Lanconelli	0.716	0.124	0.424	1.264	11
22	genLinSyst-3-2	1.424	0.200	5.172	6.796	28
23	genLinSyst-3-3	6.352	0.608	47.339	54.299	70
24	Wang93	1.248	0.612	0.336	2.196	5
25	Maclane	7.468	0.492	1.352	9.312	9
26	Neural	0.664	0.164	0.116	0.944	2
27	Leykin-1	8.916	0.172	2.472	11.560	20
28	Lazard-ascm2001	19.721	5.868	86.490	112.079	71
29	Pavelle	9.816	56.020	820.143	885.979	171
30	Cheaters-homotopy	–	–	–	–	–

Table 6.2: Solving timings and number of cells of CTD (Maple 15)

	DISPGB (Maple 8)		RegSer (Maple 8)		CTD (Maple 11)		CTD (Maple 15)	
Sys	Time (s)	# Cells	Time (s)	# components	Time (s)	# Cells	Time (s)	# Cells
1	0.509	2	0.021	3	0.122	3	0.572	2
2	0.410	2	0.021	1	0.033	1	0.352	1
3	0.550	2	0.060	3	0.114	2	0.572	2
4	1.511	2	0.070	1	0.117	1	0.464	1
5	1.030	3	0.099	4	0.870	11	1.000	8
6	1.350	4	0.049	5	0.535	4	0.752	3
7	1.609	2	0.180	4	0.739	4	0.948	4
8	2.181	3	0.150	4	0.746	2	0.760	2
9	10.710	5	0.171	7	4.878	23	1.656	13
10	9.659	5	0.329	5	7.147	10	1.632	6
11	0.489	3	0.260	9	13.267	28	1.340	11
12	259.730	5	2.381	23	26.732	10	4.476	5
13	5.830	9	0.199	9	1.826	9	1.104	9
14	–	–	–	–	72.980	4	1.448	2
15	30.470	7	0.640	10	8.621	5	2.792	5
16	61.831	7	6.060	22	9.453	21	4.884	8
17	4.619	6	0.150	5	1.481	6	1.200	6
18	8.791	5	0.319	6	3.091	6	1.036	7
19	20.739	5	3.019	10	4.602	5	2.556	4
20	101.251	2	0.371	7	1.865	2	1.432	2
21	43.441	4	0.330	7	5.822	14	1.264	11
22	–	–	0.350	18	27.312	32	6.796	28
23	–	–	2.031	61	1108.686	116	54.299	70
24	–	–	4.040	6	55.855	8	2.196	5
25	83.210	11	–	–	67.555	21	9.312	9
26	–	–	–	–	35.122	4	0.944	2
27	–	–	–	–	1375.391	57	11.560	20
28	–	–	–	–	–	–	112.079	71
29	–	–	–	–	–	–	885.979	171
30	–	–	–	–	–	–	–	–

Table 6.3: Solving timings and number of components/cells in three algorithms

Chapter 7

Computing Cylindrical Algebraic Decomposition via Triangular Decomposition

Cylindrical algebraic decomposition is one of the most important tools for computing with semi-algebraic sets, while triangular decomposition is among the most important approaches for manipulating constructible sets. In this chapter, for an arbitrary finite set $F \subset \mathbb{R}[y_1, \ldots, y_n]$ we apply comprehensive triangular decomposition in order to obtain an F-invariant cylindrical decomposition of the n-dimensional complex space, from which we extract an F-invariant cylindrical algebraic decomposition of the n-dimensional real space. We report on an implementation of this new approach for constructing cylindrical algebraic decompositions.

7.1 Introduction

Cylindrical algebraic decomposition (CAD) is a fundamental and powerful tool in real algebraic geometry. The original algorithm introduced by Collins in 1973 [44] has been followed by many substantial ameliorations, including adjacency and clustering techniques [4], improved projection methods [98, 73, 24, 17], partially built CADs [45, 99, 116], improved stack construction [46] and efficient projection orders [53].

The main application of CAD is quantifier elimination (QE) for which other approaches are also available. Some of them have more attractive complexity results [9] than CAD. However, as pointed out by Brown and Davenport [20], "there is the issue of whether the asymptotic cross-over points between CAD and those other QE algorithms actually occur in the range of problems that are even close to accessible with current machines". In addition, these authors observe that CAD can help solving certain QE problems [18, 74] that other QE algorithms can not.

For a finite set $F_n \subset \mathbb{R}[y_1, \ldots, y_n]$ the CAD algorithm [44] decomposes the real n-dimensional space into disjoint cells $C_1, \ldots, C_e$ together with one *sample point* $S_i \in C_i$, for all $1 \leq i \leq e$, such that the sign of each $f \in F_n$ does not change in C_i and can be determined at S_i. Besides, this decomposition is *cylindrical* in the following sense: For all $1 \leq j < n$ the projections on the first j coordinates $(y_1, \ldots, y_j)$ of any two cells are either disjoint or equal. We will make use of this notion of "cylindrical" decomposition in $\mathbb{C}^n$.

The algorithm of Collins is based on a *projection and lifting* procedure which computes from F_n a finite set $F_{n-1} \subset \mathbb{R}[y_1, \ldots, y_{n-1}]$ such that an F_n-invariant CAD of $\mathbb{R}^n$ can be constructed from an F_{n-1}-invariant CAD of $\mathbb{R}^{n-1}$. This construction and the base case $n = 1$ rely on real root isolation of univariate polynomials.

In this thesis, we propose a different approach for computing CAD, which proceeds by successive transformation of an initial decomposition of the complex n-dimensional space. Our algorithm consists of three main steps:

Initial Partition: we decompose $\mathbb{C}^n$ into disjoint constructible sets $C_1, \ldots, C_e$ such that for all $1 \leq i \leq e$, for each $f \in F_n$ either f is identically zero in C_i or f vanishes at no points of C_i.

Make Cylindrical: we transform the initial partition and obtain another decomposition of $\mathbb{C}^n$ into disjoint constructible sets such that this second decomposition is cylindrical in the above sense.

Make Semi-Algebraic: from the previous decomposition we produce an F_n-invariant CAD of $\mathbb{R}^n$.

Our first motivation is to understand the relation and possible interaction between cylindrical algebraic decompositions and triangular decompositions of polynomial systems. The primary goal of triangular decompositions is to provide unmixed decompositions of algebraic varieties. However, the authors in [138] have initiated the use of triangular decompositions in real algebraic geometry [138]. Moreover, real root isolation of zero-dimensional polynomial systems can be achieved via triangular decompositions [134, 94, 135, 41, 12].

A second motivation of this work is to investigate the possibility of improving the practical efficiency of CAD implementation by means of modular methods and fast polynomial arithmetic. Such techniques have been successfully introduced into triangular decomposition methods [48, 92, 90]. Each of the three main steps of the algorithm proposed in this thesis relies on existing sub-algorithms for triangular decompositions taken from [103, 30, 135] and for which efficient implementation in the `RegularChains` library [88] is work in progress based on the highly optimized low-level routines of the MODPN library [91].

Our third motivation is to extend to real algebraic geometry the concept of *Comprehensive Triangular Decomposition* (CTD) introduced in [30]. The relation between CAD and parametric polynomial system solving is natural as pointed in [54] and the presentation therein of Weispfenning's approach [24] for QE based on comprehensive Gröbner bases. This suggests that the algorithm proposed in this thesis could support a similar QE method.

This chapter is organized as follows. Section 7.2 and Section 7.3 are dedicated to the first two main steps of our algorithm whereas Sections 7.4 presents the last one. In Section 7.5 we report on a preliminary experimentation of our new algorithm. No modular methods or fast polynomial arithmetic are being used yet and our code is just high-level MAPLE interpreted code. However our code can already process well-known examples from the literature. We also analyze the performances of the different main steps and subroutines of our algorithm and implementation. This suggests that there is a large potential for improvement by means of modular methods, for instance for the computation of GCDs, resultants (and the discriminants) of polynomials modulo regular chains.

This chapter is based on paper [36], co-authored with Marc Moreno Maza, Bican Xia and Lu Yang.

7.2 Zero separation

In this section, we assume $n \geq 2$ and regard the variables $y_1 < \cdots < y_{n-1}$ as parameters, denoted by $\mathbf{u}$. Let $\pi_{\mathbf{u}}$ be the projection function which sends a point $(\bar{\mathbf{u}}, \bar{y_n})$ of $\mathbf{K}^n$ to the point $\bar{\mathbf{u}}$ of the parameter space $\mathbf{K}^{n-1}$. Let $\bar{\mathbf{u}} \in \mathbf{K}^{n-1}$. We write $\pi_{\mathbf{u}}^{-1}(\bar{\mathbf{u}})$ for the set of all points $(\bar{\mathbf{u}}, \bar{y_n})$ in $\mathbf{K}^n$ such that $\pi_{\mathbf{u}}(\bar{\mathbf{u}}, \bar{y_n}) = \bar{\mathbf{u}}$.

Let $p \in \mathbf{k}[\mathbf{u}, y_n]$ be a polynomial of level n. In broad terms, the goal of this section is to decompose the parameter space $\mathbf{K}^{n-1}$ into finitely many cells such that above each cell the "root structure" of p (number of roots, their multiplicity, ...) does not change. After some notations, we define in Definition 7.1 the object to be computed by the algorithm devised in this section. It can be seen as a specialization of the comprehensive triangular decomposition (CTD) to the case where the input system is a regular system and all variables but one are regarded as parameters. This algorithm is stated in Section 7.2.1 after two lemmas.

Notations. Let $rs = [T, h]$ be a regular system of $\mathbf{k}[\mathbf{u}, y_n]$. If y_n does not appear in rs, we denote by $Z_{\mathbf{u}}(rs)$ the zero set of rs in $\mathbf{K}^{n-1}$. If y_n does not appear in T, we write $W_{\mathbf{u}}(T)$ for the quasi-component of T in $\mathbf{K}^{n-1}$. If $\mathrm{mvar}(h) = y_n$ holds, we denote by $\mathrm{coeff}(h)$ be the set of coefficients of h when h is regarded as a polynomial in y_n with coefficients in $\mathbf{k}[\mathbf{u}]$ and by $V_{\mathbf{u}}(\mathrm{coeff}(h))$ the variety of $\mathrm{coeff}(h)$ in $\mathbf{K}^{n-1}$. Finally, if y_n is algebraic in T, letting t_n be the polynomial in T with main variable y_n, we write $T_{\mathbf{u}} = T \setminus \{t_n\}$ and $rs_{\mathbf{u}} = [T_{\mathbf{u}}, r]$, where $r = \mathrm{res}(h \cdot \mathrm{sep}(t_n), t_n)$ is the resultant of $h \cdot \mathrm{sep}(t_n)$ and t_n w.r.t y_n.

Definition 7.1. *Let C be a constructible set of $\mathbf{K}^{n-1}$. A finite set of level n polynomials $\mathcal{P} \subset \mathbf{k}[\mathbf{u}, y_n]$* separates above *$C$ if for each $\alpha \in C$: (1) the initial of any $p \in \mathcal{P}$ does not vanish at α; (2) the polynomials $p(\alpha, y_n) \in \mathbf{K}[y_n]$, $p \in \mathcal{P}$, are squarefree and coprime.*

Let $\mathcal{C}$ be a finite collection of pairwise disjoint constructible sets of $\mathbf{K}^{n-1}$, and, for each $C \in \mathcal{C}$, let $\mathcal{P}_C \subset \mathbf{k}[\mathbf{u}, y_n]$ be a finite set of level n polynomials. Let $rs_* = [T_*, h_*]$ be a regular system of $\mathbf{k}[\mathbf{u}, y_n]$, where $n \geq 2$ and y_n is algebraic w.r.t T. We say that the family $\{(C, \mathcal{P}_C) \mid C \in \mathcal{C}\}$ separates *$Z(rs_*)$ if the following conditions hold:*

(1) *$\mathcal{C}$ is a partition of $\pi_{\mathbf{u}}(Z(rs_*))$,*

(2) *for each $C \in \mathcal{C}$, $\mathcal{P}_C$ separates above C,*

(3) *$Z(rs_*) = \bigcup_{C \in \mathcal{C}} \bigcup_{p \in \mathcal{P}_C} V(p) \cap \pi_{\mathbf{u}}^{-1}(C)$.*

More generally, let cs be a constructible set of $\mathbf{K}^n$ such that there exist regular systems $rs_1, \ldots, rs_r$ of $\mathbf{k}[\mathbf{u}, y_n]$ whose zero sets form a partition of cs and such that y_n is algebraic w.r.t. the regular chain of rs_i, for all $1 \leq i \leq r$. Then, we say that the family $\{(C, \mathcal{P}_C) \mid C \in \mathcal{C}\}$ separates *cs if $\mathcal{C}$ is a partition of $\pi_{\mathbf{u}}(cs)$ and if for all $1 \leq i \leq r$ there exists a non-empty subset $\mathcal{C}_i$ of $\mathcal{C}$ and for each $C \in \mathcal{C}_i$ a non-empty subset $\mathcal{P}_{C,i} \subseteq \mathcal{P}_C$ such that $\{(C, \mathcal{P}_{C,i}) \mid C \in \mathcal{C}_i\}$ separates $Z(rs_i)$. In this case, we have: $cs = \bigcup_{C \in \mathcal{C}} \bigcup_{p \in \mathcal{P}_C} V(p) \cap \pi_{\mathbf{u}}^{-1}(C)$.*

Example 7.1. *Consider the polynomials in $\mathbf{k}[x > b > a]$*

$$p_1 = ax^2 - b \text{ and } p_2 = ax^2 + 2x + b,$$

and the constructible set $C = \{(a, b) \in \mathbf{K}^2 \mid ab(ab-1) \neq 0\}$*. For any point* (a, b) *of* C*, the two polynomials* $p_1(a, b)$ *and* $p_2(a, b)$ *of* $\mathbf{K}[x]$ *are squarefree and coprime. So the polynomial set* $\{p_1, p_2\}$ *separates above* C*.*

Consider the regular system $rs_* = [\{p_1\}, 1]$ *and the constructible sets*

$$C_1 = \{(a, b) \in \mathbf{K}^2 \mid ab \neq 0\}$$
$$C_2 = \{(a, b) \in \mathbf{K}^2 \mid a \neq 0 \ \&\ b = 0\}$$

Note that the zero set of rs_* *is* $\{p_1 = 0 \ \&\ a \neq 0\}$*. So the family* $\{\ (C_1, \{p_1\}),\ (C_2, \{ax\})\ \}$ *separates* $Z(rs_*)$*.*

Given two regular systems

$$rs_1 = [\{p_1\}, b] \text{ and } rs_2 = [\{p_2, b\}, 1].$$

Consider the constructible set

$$cs = Z(rs_1) \cup Z(rs_2) = (V(p_1) \setminus V(ab)) \cup (V(p_2, b) \setminus V(a)).$$

The family $\{\ (C_1, \{p_1\}),\ (C_2, \{p_2\})\ \}$ *separates* cs*.*

Lemma 7.1. *Let* $p \in \mathbf{k}[\mathbf{u}, y_n]$ *be a level* n *polynomial. Let* $r = \mathrm{res}(\mathrm{sep}(p), p)$ *be the resultant of* $\mathrm{sep}(p)$ *and* p *w.r.t* y_n*. Then, the polynomial* $p(\bar{\mathbf{u}})$ *of* $\mathbf{K}[y_n]$ *is squarefree and* $\mathrm{init}(p)$ *does not vanish at* $\bar{\mathbf{u}} \in \mathbf{K}^{n-1}$*, if and only if,* $r(\bar{\mathbf{u}}) \neq 0$ *holds.*

Observe that $\mathrm{init}(p)$ is a factor of r. So the conclusion follows directly from the specialization property of subresultants.

Lemma 7.2. *We have the following properties:*

(1) *If* y_n *does not appear in* rs*, then* $\pi_{\mathbf{u}}(Z(rs)) = Z_{\mathbf{u}}(rs)$*.*

(2) *If* y_n *does not appear in* T *and if* $\mathrm{mvar}(h) = y_n$ *holds, then we have* $\pi_{\mathbf{u}}(Z(rs)) = W_{\mathbf{u}}(T) \setminus V_{\mathbf{u}}(\mathrm{coeff}(h))$*.*

(3) *If* y_n *is algebraic w.r.t* T *and if the regular system* rs *is squarefree, then* $rs_{\mathbf{u}}$ *is a squarefree regular system of* $\mathbf{k}[\mathbf{u}]$*; moreover there exists a family* $\mathcal{R}'$ *of squarefree regular systems of* $\mathbf{k}[\mathbf{u}, y_n]$ *such that:*

   (a) *the rank of each* $rs' \in \mathcal{R}'$ *is less than that of* rs*,*

   (b) *for each* $[T', h'] \in \mathcal{R}'$*,* y_n *is algebraic w.r.t* T'*,* (*b*) *the zero sets* $Z(rs')$, $rs' \in \mathcal{R}'$ *and the zero set* $V(t_n) \cap Z(rs_{\mathbf{u}})$ *are pairwise disjoint, and we have*

(*d*) $Z(rs) = V(t_n) \cap Z(rs_{\mathbf{u}}) \ \cup \ \bigcup_{rs' \in \mathcal{R}'} Z(rs')$.

Proof. Property (1) is clear and proving (2) is routine. We prove (3). Since rs is squarefree, using the above notations, we have

$$\mathrm{res}(r, T) = \mathrm{res}(r, T_{\mathbf{u}}) = \mathrm{res}(h \cdot \mathrm{sep}(t_n), T) \neq 0.$$

This implies that r is regular w.r.t $\mathrm{sat}(T)$ and that $rs_{\mathbf{u}} = [T_{\mathbf{u}}, r]$ is a squarefree regular system of $\mathbf{k}[\mathbf{u}]$. Observe now that the zero set of rs decomposes in two disjoint parts:

$$Z(rs) = (Z(rs) \setminus V(r)) \cup (Z(rs) \cap V(r)).$$

For the first part, we have

$$Z(rs) \setminus V(r) = V(t_n) \cap Z(rs_{\mathbf{u}}).$$

For the second part, since r is regular w.r.t $\mathrm{sat}(T)$, by calling operation Intersect, we obtain a family $\mathcal{R}$ of squarefree regular systems of $\mathbf{k}[\mathbf{u}, y_n]$ such that

$$Z(rs) \cap V(r) = \bigcup_{rs' \in \mathcal{R}} Z(rs'),$$

where the rank of each $rs' \in \mathcal{R}$ is less than that of rs. Finally, applying the operation MPD to $\mathcal{R}$ we obtain a family $\mathcal{R}'$ satisfying the properties (a), (b), (c) and (d). □

7.2.1 The Algorithm SeparateZeros

We present now an algorithm "solving" a regular system in the sense of Definition 7.1. Precise specifications and algorithm steps follow.

Calling sequence. SeparateZeros($rs_*, \mathbf{u}, n$)

Input. A (squarefree) regular system $rs_* = [T_*, h_*]$ of $\mathbf{k}[\mathbf{u}, y_n]$, where $n \geq 2$ and y_n is algebraic w.r.t T_*.

Output. A finite family $\{(C, \mathcal{P}_C) \mid C \in \mathcal{C}\}$, where $\mathcal{C}$ is a finite collection of constructible sets of $\mathbf{K}^{n-1}$, and for each $C \in \mathcal{C}$, $\mathcal{P}_C \subset \mathbf{k}[y_1, \ldots, y_n]$ is a finite set of level n polynomials, such that $\{(C, \mathcal{P}_C) \mid C \in \mathcal{C}\}$ separates the zero set of rs_*. (See Definition 7.1.)

Step (1). Initialize $\mathcal{R} = \{rs_*\}$ and $\mathcal{P} = \emptyset$.

Step (2)**.** If $\mathcal{R} = \emptyset$, go to **Step** (3)**.** Otherwise arbitrarily choose one regular system $rs = [T, h]$ from $\mathcal{R}$ and let $\mathcal{R} = \mathcal{R} \setminus \{rs\}$. Using the above notations, let $\mathcal{R}'$ be as in Property (3) of Lemma 7.2. Set $\mathcal{P} = \mathcal{P} \cup \{(rs_{\mathbf{u}}, t_n)\}$, set $\mathcal{R} = \mathcal{R} \cup \mathcal{R}'$ and repeat **Step** (2)**.**

Comment. Observe that Step (2) will finally terminate since each newly added regular system into $\mathcal{R}$ has a rank less than that of the one removed from $\mathcal{R}$. When Step (2) terminates, we obtain a family $\mathcal{P}$ of pairs such that

$$Z(rs_*) = \bigcup_{(rs_{\mathbf{u}}, t_n) \in \mathcal{P}} V(t_n) \cap \pi_u^{-1}(Z_{\mathbf{u}}(rs_{\mathbf{u}})),$$

and the union is disjoint. Next, observe that for each pair $(rs_{\mathbf{u}}, t_n) \in \mathcal{P}$, the polynomial $\mathrm{init}(t_n)$ does not vanish at any point of $Z_{\mathbf{u}}(rs_{\mathbf{u}})$, by virtue of Lemma 7.1. Therefore, the union of all $Z_{\mathbf{u}}(rs_{\mathbf{u}})$ is equal to $\pi_{\mathbf{u}}(Z(rs_*))$.

Step (3)**.** By means of the operation SMPD we compute an intersection-free basis of all $Z_{\mathbf{u}}(rs_{\mathbf{u}})$. Hence we obtain a partition $\mathcal{C}$ of $\pi_{\mathbf{u}}(Z(rs_*))$. Then, for each $C \in \mathcal{C}$ we define $\mathcal{P}_C$ as the set of the polynomials t_n such that there exists a regular system $rs_{\mathbf{u}}$ satisfying $(rs_{\mathbf{u}}, t_n) \in \mathcal{P}$ and $C \subseteq Z_{\mathbf{u}}(rs_{\mathbf{u}})$. Clearly $\{(C, \mathcal{P}_C) \mid C \in \mathcal{C}\}$ is a valid output.

Finally, we generalize this algorithm in order to apply it to a constructible set represented by regular systems.

Calling sequence. SeparateZeros$(\{rs_1, \ldots, rs_r\}, \mathbf{u}, n)$

Input. Regular systems $rs_1, \ldots, rs_r$ of $\mathbf{k}[\mathbf{u}, y_n]$, $n \geq 2$, whose zero sets are pairwise disjoint and such that y_n is algebraic w.r.t. the regular chain of rs_i, for all $1 \leq i \leq r$; let cs be the constructible set represented by $rs_1, \ldots, rs_r$.

Output. A finite family $\{(C, \mathcal{P}_C) \mid C \in \mathcal{C}\}$, where $\mathcal{C}$ is a finite collection of constructible sets of $\mathbf{K}^{n-1}$, and for each $C \in \mathcal{C}$, $\mathcal{P}_C \subset \mathbf{k}[y_1, \ldots, y_n]$ is a finite set of level n polynomials, such that $\{(C, \mathcal{P}_C) \mid C \in \mathcal{C}\}$ separates cs. (See Definition 7.1.)

Step (1)**.** For each $1 \leq i \leq r$, call SeparateZeros$(rs_i, \mathbf{u}, n)$ obtaining $\{(C, \mathcal{P}_C) \mid C \in \mathcal{C}_i\}$ where $\mathcal{C}_i$ is a partition of $\pi_{\mathbf{u}}(Z(rs_i))$.

Step (2)**.** By means of the operation SMPD, compute an intersection-free basis $\mathcal{D}$ of the union of the $\mathcal{C}_i$, for $1 \leq i \leq r$.

Step (3)**.** For each $D \in \mathcal{D}$, let $\mathcal{P}_D$ be the union of the $\mathcal{P}_C$ such that $D \subseteq C$ holds. Return $\{(D, \mathcal{P}_D) \mid D \in \mathcal{D}\}$.

7.3 Cylindrical decomposition

In this section, we propose the notion of an *F-invariant cylindrical decomposition* of $\mathbf{K}^n$, generalizing ideas that are well-known in the case of real fields. The main algorithm and its subroutines for computing such a decomposition are stated in three subsections.

Definition 7.2. *We state the definition by induction on n. For $n = 1$, a cylindrical decomposition of $\mathbf{K}$ is a finite collection of sets $\{D_1, \ldots, D_{r+1}\}$, where either $r = 0$ and $D_1 = \mathbf{K}$, or $r > 0$ and there exists r nonconstant coprime squarefree polynomials $p_1, \ldots, p_r$ of $\mathbf{k}[y_1]$ such that*

$$D_i = \{y_1 \in \mathbf{K} \mid p_i(y_1) = 0\}, 1 \leq i \leq r,$$

and $D_{r+1} = \{y_1 \in \mathbf{K} \mid p_1(y_1) \cdots p_r(y_1) \neq 0\}$. Note that all D_i, $1 \leq i \leq r+1$ form a partition of $\mathbf{K}$. Now let $n > 1$, and let $\mathcal{D}' = \{D_1, \ldots, D_s\}$ be any cylindrical decomposition of $\mathbf{K}^{n-1}$. For each D_i, let $\{p_{i,1}, \ldots, p_{i,r_i}\}$, $r_i \geq 0$, be a set of polynomials which separates above D_i. (See Definition 7.1.) If $r_i = 0$, set $D_{i,1} = D_i \times \mathbf{K}$. If $r_i > 0$, set

$$D_{i,j} = \{(\alpha, y_n) \in \mathbf{K}^n \mid \alpha \in D_i \ \&\ p_{i,j}(\alpha, y_n) = 0\},$$

for $1 \leq j \leq r_i$ and set

$$D_{i,r_i+1} = \{(\alpha, y_n) \in \mathbf{K}^n \mid \alpha \in D_i \ \&\ \left(\prod_{j=1}^{r_i} p_{i,j}(\alpha, y_n)\right) \neq 0\}.$$

The collection $\mathcal{D} = \{D_{i,j} \mid 1 \leq i \leq s, 1 \leq j \leq r_i + 1\}$ is called a cylindrical decomposition of $\mathbf{K}^n$. Moreover, we say that $\mathcal{D}$ induces $\mathcal{D}'$.

*Let $F = \{f_1, \ldots, f_s\}$ be a finite set of polynomials of $\mathbf{k}[y_1 < \cdots < y_n]$. A cylindrical decomposition $\mathcal{D}$ of $\mathbf{K}^n$ is called F-*invariant *if $\mathcal{D}$ is an intersection-free basis of the $s+1$ constructible sets $V(f_i)$, $1 \leq i \leq s$ and $\{y \in \mathbf{K}^n \mid f_1(y) \cdots f_s(y) \neq 0\}$.*

Lemma 7.3. *Let $rs_1, \ldots, rs_{r+1}$, with $r \geq 1$, be regular systems of $\mathbf{k}[y_1]$ such that their zero sets form a partition of $\mathbf{K}^1$. Then, up to renumbering, there exist polynomials $p_1, \ldots, p_r, h_1, \ldots, h_r, h_{r+1} \in \mathbf{k}[y_1]$ such that $rs_i = [\{p_i\}, h_i]$ for $1 \leq i \leq r$ and $rs_{r+1} = [\varnothing, h_{r+1}]$. Moreover, setting $D_i = V(p_i)$ for $1 \leq i \leq r$ and $D_{r+1} = \{y_1 \in \mathbf{K} \mid p_1(y_1) \cdots p_r(y_1) \neq 0\}$, the sets $D_1, \ldots, D_{r+1}$ form a cylindrical decomposition of $\mathbf{K}$.*

Proof. Observe that for $1 \leq i \leq r$ we have $Z(rs_i) = V(p_i)$, as h_i and p_i have no common roots. Since the zero sets $Z(rs_1), \ldots, Z(rs_{r+1})$ form a partition of $\mathbf{K}^1$, we must have $V(h_{r+1}) = V(p_1 \cdots p_r)$. The conclusion follows. $\square$

7.3.1 The Algorithm MakeCylindrical

<u>**Calling sequence.**</u> MakeCylindrical($\mathcal{R}, n$)

Input. $\mathcal{R}$, a finite family of regular systems such that the zero sets $Z(rs)$, for all $rs \in \mathcal{R}$, form a partition of $\mathbf{K}^n$.

Output. $\mathcal{D}$, a cylindrical decomposition of $\mathbf{K}^n$ such that the zero set of each regular system in $\mathcal{R}$ is a union of some cells in $\mathcal{D}$.

Step (1)**: Base case.** If $n > 1$, go to (2). If $\mathcal{R}$ has only one element, return $\mathcal{D} = \mathbf{K}$ otherwise use the construction of Lemma 7.3 to return a cylindrical decomposition $\mathcal{D}$.

Step (2)**: Initialization.** Set to $\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3$ the subset of $\mathcal{R}$ consisting of regular systems $rs = [T, h]$ such that, y_n is algebraic w.r.t T, y_n appears in h but not in T, y_n does not appear in T nor in h, respectively.

Step (3)**: Processing $\mathcal{R}_1$.** Call SeparateZeros($\mathcal{R}_1, \mathbf{u}, n$) (see Section 7.2) obtaining $\{(C, \mathcal{P}_C) \mid C \in \mathcal{C}_1\}$ where $\mathcal{C}_1$ is a partition of $\pi_{\mathbf{u}}(cs_1)$, where cs_1 is the constructible set represented by $\mathcal{R}_1$. By adding a "1" in each pair, we obtain a collection of triples $\mathcal{T}_1 = \{(C, \mathcal{P}_C, 1) \mid C \in \mathcal{C}_1\}$.

Step (4)**: Processing $\mathcal{R}_2$.** For each $rs \in \mathcal{R}_2$, compute the projection $\pi_{\mathbf{u}}(Z(rs))$ by Property (2) of Lemma 7.2. Set $\mathcal{C}_2 = \{\pi_{\mathbf{u}}(Z(rs)) \mid rs \in \mathcal{R}_2\}$ and $\mathcal{T}_2 = \{(C, \emptyset, 2) \mid C \in \mathcal{C}_2\}$.

Step (5)**: Processing $\mathcal{R}_3$.** For each $rs \in \mathcal{R}_3$, compute the projection $\pi_{\mathbf{u}}(Z(rs))$ by Property (1) of Lemma 7.2. Set $\mathcal{C}_3 = \{\pi_{\mathbf{u}}(Z(rs)) \mid rs \in \mathcal{R}_3\}$ and $\mathcal{T}_3 = \{(C, \emptyset, 3) \mid C \in \mathcal{C}_3\}$.

Comment. Since the zero sets of regular systems in $\mathcal{R}$ are pairwise disjoint, after step (3), (4), (5), we know that the element in $\mathcal{C}_3$ has no intersection with any element in $\mathcal{C}_1$ or $\mathcal{C}_2$. Note that it is possible that an element in $\mathcal{C}_1$ has intersection with some element of $\mathcal{C}_2$. So we need the following step to remove the common part between them.

Step (6)**: Merging.** Set $\mathcal{C} = \mathcal{C}_1 \cup \mathcal{C}_2 \cup \mathcal{C}_3$ and $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2 \cup \mathcal{T}_3$. Note that each element in $\mathcal{T}$ is a triple $(C, \mathcal{P}_C, \mathcal{I}_C)$, with $C \in \mathcal{C}$ and where $\mathcal{I}_C$ is an integer of value $1, 2$ or 3. By means of the operation SMPD, compute an intersection-free basis $\mathcal{C}'$ of $\mathcal{C}$. For each $C' \in \mathcal{C}'$, compute $\mathcal{Q}_{C'}$ (resp. $\mathcal{J}_{C'}$) the union of the $\mathcal{P}_C$ (resp. $\mathcal{I}_C$) such that $C' \subseteq C$ holds. Set $\mathcal{T}' = \{(C, \mathcal{Q}_C, \mathcal{J}_C) \mid C \in \mathcal{C}'\}$.

Step (7)**: Refinement.** To each $C \in \mathcal{C}'$, apply operation MPD to the family of regular systems representing C, so as to obtain another family $\mathcal{R}_C$ of regular systems representing C and whose zero sets are pairwise disjoint. For each $rs \in \mathcal{R}_C$, set $\mathcal{P}_{rs} = \mathcal{Q}_C$ and $\mathcal{I}_{rs} = \mathcal{J}_C$. Let $\mathcal{R}'$ be the union of the $\mathcal{R}_C$, for all $C \in \mathcal{C}'$. Set $\mathcal{T}'' = \{(Z(rs), \mathcal{P}_{rs}, \mathcal{I}_{rs}) \mid rs \in \mathcal{R}'\}$.

Comment. Recall that the union of zero sets of the $Z(rs)$, for all $rs \in \mathcal{R}$ equals $\mathbf{K}^n$. Therefore, it follows from Steps (6) and (7), that $\{Z(rs) \mid rs \in \mathcal{R}'\}$ is a partition of $\mathbf{K}^{n-1}$.

Step (8)**: Recursive call.** Call $\mathsf{MakeCylindrical}(\mathcal{R}', n-1)$ to compute a cylindrical decomposition $\mathcal{D}'$ of $\mathbf{K}^{n-1}$ such that $Z(rs)$, for each $rs \in \mathcal{R}'$, is a union of some cells of $\mathcal{D}'$. For each $D' \in \mathcal{D}'$, observe that there exists a unique $rs \in \mathcal{R}'$ such that $D' \subseteq Z(rs)$, so set $\mathcal{P}_{D'} = \mathcal{P}_{rs}$ and $\mathcal{I}_{D'} = \mathcal{I}_{rs}$. Then, set $\mathcal{T}''' = \{(D', \mathcal{P}_{D'}, \mathcal{I}_{D'}) \mid D' \in \mathcal{D}'\}$.

Comment. By the comment below Step (5), we know that for each triple $(D', \mathcal{P}_{D'}, \mathcal{I}_{D'})$ of $\mathcal{T}'''$, the values of $\mathcal{I}_{D'}$ can only be $\{1, 2\}$, $\{2\}$ or $\{3\}$. Next, observe that for each $D' \in \mathcal{D}'$ such that $\mathcal{I}_{D'} = \{2\}$ or $\mathcal{I}_{D'} = \{3\}$ holds, we have $\mathcal{P}_{D'} = \emptyset$, whereas for each $D' \in \mathcal{D}'$ such that $\mathcal{I}_{D'} = \{1, 2\}$ the set $\mathcal{P}_{D'}$ is a nonempty finite family of level n polynomials in $\mathbf{k}[y_1, \ldots, y_n]$ such that $\mathcal{P}_{D'}$ separates above $\mathcal{D}'$. In Step (9) below, we lift the cylindrical decomposition $\mathcal{D}'$ of $\mathbf{K}^{n-1}$ to a cylindrical decomposition $\mathcal{D}$ of $\mathbf{K}^n$.

Step (9)**: Lifting.** Initialize $\mathcal{D}$ to the empty set. For each $D' \in \mathcal{D}'$ such that $\mathcal{I}_{D'} = \{2\}$ or $\mathcal{I}_{D'} = \{3\}$ holds, let $\mathcal{D} := \mathcal{D} \cup \{D' \times \mathbf{K}\}$. For each $D' \in \mathcal{D}'$ such that $\mathcal{I}_{D'} = \{1, 2\}$ holds, let $\mathcal{D} = \mathcal{D} \cup \{D_p\}$, where

$$D_p = \{(\alpha, y_n) \in \mathbf{K}^n \mid \alpha \in D' \text{ and } \quad p(\alpha, y_n) = 0\},$$

for each $p \in \mathcal{P}_{D'}$ and let $\mathcal{D} = \mathcal{D} \cup \{D_*\}$, where

$$D_* = \{(\alpha, y_n) \in \mathbf{K}^n \mid \alpha \in D' \;\&\; \left(\prod_{p \in \mathcal{P}_{D'}} p(\alpha, y_n) \right) \neq 0\},$$

Finally, return $\mathcal{D}$. The correctness of the algorithm follows from all the comments and Definition 7.2.

7.3.2 The Algorithm InitialPartition

<u>**Calling sequence.**</u> $\mathsf{InitialPartition}(F, n)$

Input. $F = \{f_1, \ldots, f_s\}$, a finite subset of $\mathbf{k}[y_1 < \cdots < y_n]$.

Output. A family $\mathcal{R}$ of regular systems, the zero sets of which form an intersection-free basis of the $s+1$ constructible sets $V(f_1), \ldots, V(f_s)$ and $\{y \in \mathbf{K}^n \mid (\prod_{i=1}^s f_i(y)) \neq 0\}$.

Step (1)**:** Let $\mathcal{B} = \mathsf{SMPD}(V(f_1), \ldots, V(f_s))$ be an intersection free basis of the s constructible sets $V(f_1), \ldots, V(f_s)$. For each element B of $\mathcal{B}$, we apply operation MPD to the family of regular systems representing B to compute another family $\mathcal{R}_B$ of squarefree regular systems such that the zero sets of regular systems in $\mathcal{R}_B$ are pairwise disjoint and their union is B. Let $\mathcal{R}$ be the union of all $\mathcal{R}_B$, $B \in \mathcal{B}$. Clearly the set $\{Z(rs) \mid rs \in \mathcal{R}\}$ is an intersection-free basis of the s constructible sets $V(f_1), \ldots, V(f_s)$.

Step (2)**:** Let $f = \prod_{f_i \in F} f_i$ and $rs_* = [\varnothing, f]$. Set $\mathcal{R} = \mathcal{R} \cup \{rs_*\}$. Obviously $\mathcal{R}$ is the valid output.

7.3.3 The Algorithm $\mathsf{CylindricalDecompose}$

Calling sequence. $\mathsf{CylindricalDecompose}(F, n)$

Input. F, a finite subset of $\mathbf{k}[y_1 < \cdots < y_n]$.

Output. an F-invariant cylindrical decomposition of $\mathbf{K}^n$.

Step (1)**:** If $n > 1$, go to step (2). Otherwise let $\{p_1, \ldots, p_r\}$, $r \geq 0$, be the set of irreducible divisors of non-constant elements of F. If $r = 0$, set $\mathcal{D} = \mathbf{K}$ and exit. Otherwise set

$$D_i = \{y_1 \in \mathbf{K} \mid p_i(y_1) = 0\}, 1 \leq i \leq r,$$

and $D_{r+1} = \{y_1 \in \mathbf{K} \mid p_1(y_1) \cdots p_r(y_1) \neq 0\}$. Clearly $\mathcal{D} = \{D_i \mid 1 \leq i \leq r+1\}$ is an F-invariant cylindrical decomposition of $\mathbf{K}$.

Step (2)**:** Let $\mathcal{R}$ be the output of $\mathsf{InitialPartition}(F, n)$.

Step (3)**:** Call algorithm $\mathsf{MakeCylindrical}(\mathcal{R}, n)$, to compute a cylindrical decomposition $\mathcal{D}$ of $\mathbf{K}^n$ such that the zero set of each regular system in $\mathcal{R}$ is a union of some cells in $\mathcal{D}$. Clearly, $\mathcal{D}$ is an intersection-free basis of the set $\{Z(rs) \mid rs \in \mathcal{R}\}$, which implies $\mathcal{D}$ is an intersection-free basis of the $s+1$ constructible sets $V(f_1), \ldots, V(f_s)$ and $\{y \in \mathbf{K}^n \mid (\prod_{i=1}^s f_i(y)) \neq 0\}$. Therefore, $\mathcal{D}$ is an F-invariant cylindrical decomposition of $\mathbf{K}^n$.

7.3.4 Relation with simple systems

Let $\mathcal{D}$ be a cylindrical decomposition of $\mathbf{K}^n$. As stated in the definition, each $D \in \mathcal{D}$ is described by the common zeros of a family of polynomial equations and inequations.

Let A and B be respectively the set of those polynomials appearing as equations and inequations in D. Observe that A and B have the following properties.

(a) $A \cap B = \emptyset$ and $A \cup B$ is a triangular set of $\mathbf{k}[y_1, \ldots, y_n]$.

(b) for any $1 \leq k \leq n$, let $A^{(k-1)}$ and $B^{(k-1)}$ be respectively the subset of A and B in which the level of each polynomial is less than k. Let α be a point of $\mathbf{K}^{k-1}$ which is a zero of each polynomial of $A^{(k-1)}$ and not a zero of any polynomial of $B^{(k-1)}$. Let $p_k \in A \cup B$ be a polynomial of level k. If p_k exists, then the initial of p_k does not vanish at α and $p_k(\alpha)$ is squarefree polynomial of $\mathbf{K}[y_k]$.

A pair $[A, B]$ satisfying the above two properties is called a *simple system* in [125], which was first introduced by Thomas in 1937 [120]. A simple system has many nice properties. For example, if $[A, B]$ is a simple system, then the pair $[A, \prod_{p \in B} p]$ is a squarefree regular system [125, 126].

7.4 Cylindrical algebraic decomposition

In this section, we show how to compute a CAD of $\mathbb{R}^n$ from a cylindrical decomposition of $\mathbb{C}^n$. This section starts with reviewing basic notions for CAD [3]. A theorem (Theorem 7.1) due to Collins [44] is then reviewed, where the relation between complex and real roots of a polynomial with real coefficients is shown. The bridge from cylindrical decomposition to CAD is built in Corollary 7.1, which can be directly obtained from Collins' theorem. The main algorithm TCAD (short name for CAD based on triangular decompositions), and its subroutines are stated in four subsections.

A *semi-algebraic set* [9] of $\mathbb{R}^n$ is a subset of $\mathbb{R}^n$ which can be written as a finite union of sets of the form:

$$\{y \in \mathbb{R}^n \mid \forall f \in F, f(y) = 0 \text{ and } \forall g \in G, g(y) > 0\},$$

where both F and G are finite subsets of the polynomial ring $\mathbb{R}[y_1, \ldots, y_n]$.

Given an n-dimensional real space $\mathbb{R}^n$, a nonempty connected subset of $\mathbb{R}^n$ is called a *region*. For any subset S of $\mathbb{R}^n$, a *decomposition* of S is a finite collection of disjoint regions whose union is S. For a region R, the *cylinder* over R, written $Z(R)$, is $R \times \mathbb{R}^1$. Let $f_1 < \cdots < f_r, r \geq 0$ be continuous, real-valued functions defined on R. Let $f_0 = -\infty$ and $f_{r+1} = +\infty$. For any f_i, $1 \leq i \leq r$, we call the set of points $\{(a, f_i(a)) \mid a \in R\}$ the f_i-*section* of $Z(R)$. For any two functions f_i, f_{i+1}, $0 \leq i \leq r$, the set of points (a, b), where a ranges over R and $f_i(a) < b < f_{i+1}(a)$, is called the (f_i, f_{i+1})-*sector* of $Z(R)$. All the sections and sectors of $Z(R)$ can be ordered as

$$(f_0, f_1) < f_1 < \cdots < f_r < (f_r, f_{r+1}).$$

Clearly they form a decomposition of $Z(R)$, which is called a *stack* over R.

A decomposition $\mathcal{E}$ of $\mathbb{R}^n$ is *cylindrical* if either (1) $n = 1$ and $\mathcal{E}$ is a stack over $\mathbb{R}^0$, or (2) $n > 1$, and there is a cylindrical decomposition $\mathcal{E}'$ of $\mathbb{R}^{n-1}$ such that for each region R in $\mathcal{E}'$, some subset of $\mathcal{E}$ is a stack over R. Moreover, We say that $\mathcal{E}$ induces $\mathcal{E}'$. A decomposition is *algebraic* if each of its regions is a semi-algebraic set. A *cylindrical algebraic decomposition* of $\mathbb{R}^n$ is a decomposition which is both cylindrical and algebraic.

Let p be a polynomial of $\mathbb{R}[y_1, \ldots, y_n]$, and let S be a subset of $\mathbb{R}^n$. The polynomial p is *invariant* on S (and S is p-invariant), if the sign of $p(\alpha)$ does not change when α ranges over S. Let $F \subset \mathbb{R}[y_1, \ldots, y_n]$ be a finite polynomial set. We say S is F-invariant if each $p \in F$ is invariant on S. A cylindrical algebraic decomposition $\mathcal{E}$ is F-invariant if F is invariant on each region of $\mathcal{E}$.

Let p be a polynomial of $\mathbb{R}[y_1, \ldots, y_n]$, and let R be a region in $\mathbb{R}^{n-1}$. p is *delineable* on R if the real zeros of p define continuous real-valued functions $\theta_1, \ldots, \theta_s$ such that, for all $\alpha \in R$, $\theta_1(\alpha) < \cdots < \theta_s(\alpha)$. Note that if $k = 0$, $V(p)$ has no intersection with $Z(R)$. Clearly when p is delineable on R, its real zeros naturally determine a stack over R.

Let $\mathcal{E}$ be a CAD of $\mathbb{R}^n$. As suggested in [3], each region $e \in \mathcal{E}$ can be represented by a pair (I, S), where I is the *index* of e and S is a *sample point* for e. The index I and the sample point S of e are defined as follows. If $n = 1$, let

$$e_1 < e_2 < \cdots < e_{2m} < e_{2m+1}, m \geq 0$$

be the elements of $\mathcal{E}$. For each e_i, the index of e_i is defined as (i). For each e_i, its sample point is any algebraic point belonging to e_i. Let $\mathcal{E}'$ be the CAD of $\mathbb{R}^{n-1}$ induced by $\mathcal{E}$. Suppose that region indices and sample points have been defined for $\mathcal{E}'$. Let

$$e_{i,1} < e_{i,2} < \cdots < e_{i,2m_i} < e_{i,2m_i+1}, m_i \geq 0$$

be the elements of $\mathcal{E}$ which form a stack over the region e_i of $\mathcal{E}'$. Let $(i_1, \ldots, i_{n-1})$ be the index of e_i. Then the index of $e_{i,j}$ is defined as $(i_1, \ldots, i_{n-1}, j)$. Let S' be a sample point of e_i. Then the sample point of $e_{i,j}$ is an algebraic point belonging to $e_{i,j}$ such that its first $n-1$ coordinates are the same as that of S'.

Theorem 7.1 (Collins)**.** *Let p be a polynomial of ring $\mathbb{R}[y_1 < \cdots < y_n]$ and R be a region of $\mathbb{R}^{n-1}$. If $init(p) \neq 0$ on R and the number of distinct complex roots of p is invariant on R, then p is delineable on R.*

Corollary 7.1. *Let $F = \{p_1, \ldots, p_r\}$ be a finite set of polynomials in $\mathbb{R}[y_1 < \cdots < y_n]$ of level n. Let R be a region of $\mathbb{R}^{n-1}$. Assume that for every $\alpha \in R$,* (1) *the initial of each p_i does not vanish at α;* (2) *all $p_i(\alpha, y_n)$, $1 \leq i \leq r$, as polynomials of $\mathbb{R}[y_n]$, are squarefree and coprime. Then each p_i is delineable on R and the sections of $Z(R)$ belonging to different p_i and p_j are disjoint.*

Let R and F be defined as in the above corollary. Then clearly the real roots of all $p \in F$ are continuous functions on R and they together determine a stack over R. The algorithm GenerateStack, described in Section 7.4.2, is a direct application of the above corollary.

7.4.1 Real root isolation

Let $\alpha = (\alpha_1, \ldots, \alpha_n)$ be an algebraic point of $\mathbb{R}^n$. Each α_i as an algebraic number is a zero of a nonconstant squarefree polynomial $t_i(y_i)$ of $\mathbb{Q}[y_i]$. Let T be the set of all $t_i(y_i)$. Clearly T is a zero dimensional squarefree regular chain of $\mathbb{Q}[\mathbf{y}]$. On the other hand, if T is a zero-dimensional regular chain of $\mathbb{Q}[\mathbf{y}]$, any real zero of T is an algebraic point of $\mathbb{R}^n$. Therefore any algebraic point α of $\mathbb{R}^n$ can be represented by a pair (T, L), where T is a zero-dimensional squarefree regular chain of $\mathbb{Q}[\mathbf{y}]$ such that $T(\alpha) = 0$ and L is an isolating cube containing α but not other zeros of T. The pair (T, L) is called a *regular chain representation* of α, which will be used to represent a sample point of CAD.

Next we provide the specification of an algorithm called IsolateZeros for isolating real zeros of univariate polynomials with real algebraic number coefficients. It is a subroutine of the algorithm NREALZERO proposed in [135] for isolating the real roots of a zero-dimensional regular chain.

Calling sequence. IsolateZeros($\alpha^{(n-1)}, F, n$)

Input. $\alpha^{(n-1)}$ is a point of $\mathbb{R}^{n-1}$, $n \geq 1$, with a regular chain representation (T', L'). If $n = 1$, $T' = \varnothing$ and $L' = \emptyset$. $F = \{p_1, \ldots, p_r\}$ is a list of non-constant polynomials of $\mathbb{Q}[y_1, \cdots, y_n]$ of level n satisfying that (1) for $p_i \in F$, $T' \cup \{p_i\}$ is a squarefree regular chain of $\mathbb{Q}[y_1, \ldots, y_n]$; (2) all $p_i(\alpha^{(n-1)}, y_n)$, $1 \leq i \leq r$, as polynomials of $\mathbb{R}[y_n]$, are squarefree and coprime.

Output. A pair (N, ν). Let $p = \prod_{i=1}^{r} p_i$. $N = (N_1, \ldots, N_m)$ is a list of intervals with rational endpoints with $N_1 < \cdots < N_m$ such that each N_j contains exactly one real zero of $p(\alpha^{(n-1)}, y_n)$. $\nu = (\nu_1, \ldots, \nu_m)$ is list of integers, where $1 \leq \nu_i \leq r$, such that the zero of $p(\alpha^{(n-1)}, y_n)$ in N_j is a zero of $p_{\nu_j}(\alpha^{(n-1)}, y_n)$.

7.4.2 The Algorithm GenerateStack

Calling sequence. GenerateStack(e', F, n)

Input. e' is a region of a CAD $\mathcal{E}'$ of $\mathbb{R}^{n-1}$, $n \geq 1$, and e' is represented by its index I' and its sample point S'. Let (T', L') be the regular chain representation of S'. If $n = 1$, $T' = \varnothing$, $I' = \emptyset$ and $L' = \emptyset$. F is a finite set of polynomials in $\mathbb{Q}[y_1, \ldots, y_n]$ of level n. The region e' and the polynomial set F satisfy the conditions specified in Corollary 7.1.

Output. A stack $\mathcal{S}$ over e'.

Step (1). If $F = \emptyset$, go to step (2). Otherwise call algorithm IsolateZeros(S', F, n) to isolate the real roots of polynomials in F w.r.t y_n at the sample point S' of e'. Let (N, ν) be the output. If $N \neq \emptyset$, go to step (3).

Step (2). Let $I = (I', 1)$. Let $T = T' \cup \{y_n\}$, $L = L' \times [0, 0]$, $S = (T, L)$ and return $\mathcal{S} = ((I, S))$.

Step (3). Let $N_1 = [a_1, b_1], \ldots, N_m = [a_m, b_m]$, $m > 0$ be the elements of N. For $1 \leq i \leq 2m + 1$, set $I_i = (I', i)$. Let s_1 be the greatest integer less than a_1. Let s_{2m+1} be the smallest integer greater than b_m. For $1 \leq i \leq m - 1$, let $s_{2i+1} = \frac{b_i + a_{i+1}}{2}$. For $0 \leq i \leq m$, Let $T_{2i+1} = T' \cup \{y_n - s_{2i+1}\}$, $L_{2i+1} = L' \times [s_{2i+1}, s_{2i+1}]$ and set $S_{2i+1} = (T_{2i+1}, L_{2i+1})$. For $1 \leq i \leq m$, let $T_{2i} = T' \cup p_{\nu_i}$, $L_{2i} = L' \times N_i$ and set $S_{2i} = (T_{2i}, L_{2i})$. Finally, set $\mathcal{S}$ be the list of all (I_i, S_i), $1 \leq i \leq 2m + 1$. Then $\mathcal{S}$ is the stack over e'.

7.4.3 The Algorithm MakeSemiAlgebraic

Calling sequence. MakeSemiAlgebraic$(\mathcal{D}, n)$

Input. $\mathcal{D}$ is a cylindrical decomposition of $\mathbb{C}^n$, $n \geq 1$.

Output. A CAD $\mathcal{E}$ of $\mathbb{R}^n$ such that, for each element D of $\mathcal{D}$, the set $D \cap \mathbb{R}^n$ is a union of some regions in $\mathcal{E}$.

Step (1). If $n > 1$ go to (2). Otherwise let $D_1, \ldots, D_r, D_{r+1}$, $r \geq 0$ be the elements of $\mathcal{D}$. For each $1 \leq i \leq r$, let p_i be the polynomial such that $D_i = \{y_1 \mid p_i(y_1) = 0\}$. Let $\mathcal{E}$ be the output of GenerateStack$(\varnothing, \{p_1, \ldots, p_r\}, 1)$. Clearly $\mathcal{E}$ is a CAD of $\mathbb{R}^1$.

Step (2). Let $\mathcal{D}'$ be the cylindrical decomposition of $\mathbb{C}^{n-1}$ induced by $\mathcal{D}$. Call MakeSemiAlgebraic recursively to compute a CAD $\mathcal{E}'$ of $\mathbb{R}^{n-1}$.

Step (3). In this step we lift the CAD $\mathcal{E}'$ of $\mathbb{R}^{n-1}$ to $\mathcal{E}$. Initialize $\mathcal{E} = (\ )$. For each region e' of $\mathcal{E}'$, let D' be the cell of $\mathcal{D}'$ such that $e' \subset D' \cap \mathbb{R}^n$. Let $D_1, \ldots, D_r, D_{r+1}$, $r \geq 0$ be the cells of $\mathcal{D}$ such that $D' \times \mathbb{C} = \cup_{j=1}^{r+1} D_j$. For each $1 \leq j \leq r$, let p_j be the polynomial such that $D_j = \{(\alpha, y_n) \mid \alpha \in D' \ \&\ p_j(\alpha, y_n) = 0\}$. Add output of GenerateStack$(e', \{p_1, \ldots, p_r\}, n)$ into $\mathcal{E}$. Clearly $\mathcal{E}$ is a CAD of $\mathbb{R}^n$ and for each $D \in \mathcal{D}$, the set $D \cap \mathbb{R}^n$ is a union of some regions in $\mathcal{E}$.

7.4.4 The Algorithm TCAD

Calling sequence. TCAD(F, n)

Input. F is a finite subset of $\mathbb{Q}[y_1 < \cdots < y_n]$, $n \geq 1$.

Output. An F-invariant CAD $\mathcal{E}$ of $\mathbb{R}^n$.

Step (1). Let $\mathcal{D} =$ CylindricalDecompose(F, n) be an F-invariant cylindrical decomposition of $\mathbb{C}^n$.

Step (2). Call algorithm MakeSemiAlgebraic to compute a CAD $\mathcal{E}$ of $\mathbb{R}^n$ such that, for each element D of $\mathcal{D}$, the set $D \cap \mathbb{R}^n$ is a union of some regions in $\mathcal{E}$. Since $\mathcal{D}$ is an intersection-free basis of the $s+1$ constructible sets $V_{\mathbb{C}}(f_1), \ldots, V_{\mathbb{C}}(f_s)$ and $\{y \in \mathbb{C}^n \mid (\prod_{i=1}^{s} f_i(y)) \neq 0\}$, $\mathcal{E}$ is an intersection-free basis of the $s+1$ semi-algebraic sets $V_{\mathbb{R}}(f_1), \ldots, V_{\mathbb{R}}(f_s)$ and $\{y \in \mathbb{R}^n \mid (\prod_{i=1}^{s} f_i(y)) \neq 0\}$. Note that each element in $\mathcal{E}$ is connected. Therefore $\mathcal{E}$ is an F-invariant cylindrical algebraic decomposition of $\mathbb{R}^n$.

7.5 Examples and experimentation

7.5.1 An example

Let us illustrate our method by a simple and classical example. Consider the parametric parabola $p = ax^2 + bx + c$. Set the order of variables as $x > c > b > a$. The first step InitialPartition generates four regular systems, whose zero sets form a partition of $\mathbb{C}^4$.

$$r_1 := \begin{cases} c &= 0 \\ b &= 0 \\ a &= 0 \end{cases}, \quad r_2 := \begin{cases} bx + c &= 0 \\ b &\neq 0 \\ a &= 0 \end{cases},$$

$$r_3 := \begin{cases} ax^2+bx+c &= 0 \\ a &\neq 0 \end{cases}, \quad r_4 := \left\{ ax^2+bx+c \neq 0 \right. .$$

Next we trace the algorithm MakeCylindrical. Initialize the sets $\mathcal{R}_1 := \{r_2, r_3\}$, $\mathcal{R}_2 := \{r_4\}$ and $\mathcal{R}_3 := \{r_1\}$. Since x appears in the equations of r_2 and r_3, SeparateZeros($\mathcal{R}_1$) is called to obtain a family of pairs

$$\{(C_1, \{t\}), (C_2, \{p\}), (C_3, \{q\})\},$$

defined as follows, which separates $Z(r_2) \cup Z(r_3)$.

$$\begin{array}{ll} C_1 : \{a = 0, b \neq 0\} & \to \quad \{t\} : \{bx + c\} \\ C_2 : \{a(4ac - b^2) \neq 0\} & \to \quad \{p\} : \{ax^2 + bx + c\} \\ C_3 : \{4ac - b^2 = 0, a \neq 0\} & \to \quad \{q\} : \{2ax + b\} \end{array}$$

The projection of $Z(r_4)$ is the values such that a, b, c do not vanish simultaneously, denoted by C_4. The projection of $Z(r_1)$ is the set $\{a = b = c = 0\}$, denoted by C_5.

Note that C_1, C_2, C_3 are all subsets of C_4. In the **Merging** step, by calling SMPD, we get another set $C_6 := \{a = b = 0, c \neq 0\}$ such that C_1, C_2, C_3, C_5 and C_6 are pairwise disjoint and their union is $\mathbb{C}^3$. Moreover, for each C_i, there is a family of polynomials and indices associated to it.

C_1	C_2	C_3	C_5	C_6
$\{t\}$	$\{p\}$	$\{q\}$	$\emptyset$	$\emptyset$
$\{1,2\}$	$\{1,2\}$	$\{1,2\}$	$\{3\}$	$\{2\}$

Since each C_i is already the zero set of some regular system,

$$\mathsf{MakeCylindrical}(\{C_1, C_2, C_3, C_5, C_6\}, 3)$$

is called recursively to compute a cylindrical decomposition of $\mathbb{C}^3$. By the **Lifting** step, we finally obtain a p-invariant cylindrical decomposition of $\mathbb{C}^4$. Let $r = 4ac - b^2$, the decomposition can be described by the following tree.

From the above tree, the algorithm MakeSemiAlgebraic finally produces a CAD of $\mathbb{R}^4$ with 27 cells. As pointed out in [17], by Collins-Hong or McCallum projection operator, one computes the following polynomials during the projection phase: $ax^2 + bx + c, b^2 - 4ac, c, b, a$. In the lifting phase, one then obtains a CAD of $\mathbb{R}^4$ with 115 cells! A CAD with 27 cells is obtained by McCallum-Brown projection operator. However, this latter operator fails in some (rare) cases.

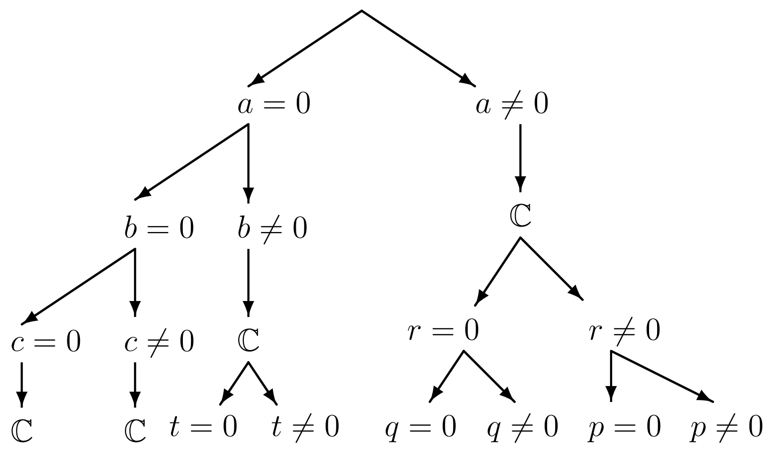


Figure 7.1: A cylindrical decomposition of $\mathbb{C}^4$ induced by $ax^2 + bx + c$

7.5.2 Experimental results

In this section, we present experimental results obtained with an implementation of the algorithms presented in this chapter. Our code is in MAPLE 12 running on a computer with Intel Core 2 Quad CPU (2.40GHz) and 3.0GB total memory. The test examples are available at `www.csd.uwo.ca/People/gradstudents/cchen252/CMXY09/examples.pdf`. They are taken from diverse papers [53, 3, 45, 98, 17, 46, 24] on CAD. The time-out for a test run is set to 2 hours.

In Table 7.1, we show the total computation time of TCAD and the time spent on three main phases of it, which are InitialPartition, (Partition for short), MakeCylindrical, (M.C. for short) and MakeSemiAlgebraic. (M.S.A. for short). We also report the number of elements ($N_{\mathbb{R}}$) in the CAD. Aborted computations due to time-out are marked with "-". From the table, one can see that, except examples 14 and 16, the steps of the algorithm dedicated to computations in complex space dominate the step taking place in the real space.

In Table 7.2, we show the total computation time of the algorithm CylindricalDecompose and the time spent on three main operations of it, which are respectively, MPD and SMPD. We can see that the cost of algorithm CylindricalDecompose is dominated by SMPD. The number of elements ($N_{\mathbb{C}}$) in the cylindrical decomposition of $\mathbb{C}^n$ is also reported.

The data reported in two tables shows that SMPD is the dominant operation, which computes intensively GCDs of polynomials modulo regular chains. This suggests that the modular methods and efficient implementation techniques in [48, 92, 90]

	Sys	Partition	M.C.	M.S.A.	Total	$N_{\mathbb{R}}$
1	Parabola	0.024	0.096	0.024	0.144	27
2	Whitney-umbrella	1.184	2.856	1.048	5.088	895
3	Quartic	0.004	7.512	0.704	8.220	233
4	Sphere-catastrophe	0.264	1.368	1.080	2.716	421
5	Arnon-84	0.016	0.052	0.116	0.184	55
6	Arnon-84-2	0.108	0.156	0.120	0.384	41
7	Real-implicitization	2.704	3.600	1.360	7.664	893
8	Ball-cylindar	0.380	1.608	1.196	3.184	365
9	Termination-term-rewrite	0.288	0.532	0.264	1.084	209
10	Collins-Johnson	5.668	48.079	18.833	72.640	3677
11	Range-lower-bounds	0.252	1.192	0.620	2.068	563
12	X-axis-ellipse	2.664	135.028	88.142	225.862	20143
13	Davenport-Heintz	10.576	35.846	6.905	53.335	4949
14	Hong-90	5.728	71.760	2520.354	2597.878	27547
15	Solotareff-3	690.731	2513.817	299.250	3503.954	66675
16	Collision	895.435	2064.469	-	-	-
17	McCallum-random	0.052	-	-	-	-
18	Ellipse-cad	-	-	-	-	-

Table 7.1: Timing (s) and number of cells for TCAD

(use of FFT-based polynomial arithmetic, ...) have a large potential for improving the implementation of our CAD algorithm.

In Table 7.3, we compare the timings and number of cells in the output with Qepcad b. The following is a sample calling sequence of Qepcad b for the example Parabola.

```
[]
(a, b, c, x)
4
[a x^2 + b x + c = 0].
full-cad:
go
go
go
d-fpc-stat:
finish:
```

We have the following observations.

- For systems 1, 4, 11, 12 and 14, TCAD outputs much fewer cells than Qepcad

	Sys	SeparateZeros	MPD	SMPD	Total	$N_{\mathbb{C}}$
1	Parabola	0.020	0.012	0.084	0.156	8
2	Whitney-umbrella	0.508	0.252	2.268	4.052	63
3	Quartic	3.856	0.836	2.460	7.880	24
4	Sphere-catastrophe	0.280	0.088	1.036	1.648	65
5	Arnon-84	0.032	0.008	0.012	0.064	7
6	Arnon-84-2	0.036	0.012	0.092	0.268	13
7	Real-implicitization	1.100	0.652	2.416	6.320	58
8	Ball-cylindar	0.536	0.144	1.040	2.008	55
9	Termination-term-rewrite	0.120	0.032	0.384	0.816	26
10	Collins-Johnson	3.204	0.756	49.031	54.119	594
11	Range-lower-bounds	0.128	0.032	0.960	1.416	49
12	X-axis-ellipse	8.508	2.024	125.104	138.188	856
13	Davenport-Heintz	2.040	1.784	42.578	47.002	407
14	Hong-90	5.741	2.092	64.875	76.956	983
15	Solotareff-3	83.469	62.736	3066.071	3232.073	2974
16	Collision	66.516	377.664	2501.947	2959.904	5877

Table 7.2: Timing (s) and number of cells for CylindricalDecompose

B. For the other 10 examples, where both software can compute, the cells in the output are either exact or nearly the same.

- Among the 15 systems that both solvers can compute, for 5 of them, QEPCAD B prints error or warning message [1] during the execution, which indicates the output CAD may not be a valid one.
- Among the 18 test examples, QEPCAD B could solve 17[2] while TCAD succeeds on 15 of them. In terms of timing, TCAD is currently slower than QEPCAD B.

The previous observation indicates that TCAD tends to produce much less cells while consumes much more time than QEPCAD B. We next provide some preliminary partial explanations and leave a complete explanation as future work.

The algorithm of QEPCAD B is based on a projection-lifting scheme. Let P_k be a set of polynomials with main variable y_k. To compute a P_k sign invariant CAD $\mathcal{C}_k$ of

[1]For system Quartic, the error message is: "Error! Delineating polynomial should be added over cell(2,2)!". For system Real-implicitization, 4 warning messages are generated with two types. The first one is "Warning! Some 3-level projection factor is acting as a delineating polynomial for another! CAD Simplification does not take this into account!". The second one is "A projection factor is everywhere zero in the cylinder over the cell (2,2,1) of positive dimension. The McCallum projection may not be valid." For system Range-lower-bounds, it generates 9 warning messages with the above two types. For system X-axis-ellipse, it generates 2 warning messages with the first type. For system Hong-90, it generates 5 warning messages with the above two types.

[2]Note that QEPCAD B solves only 13 if the default memory option "$+N20000000$" is used. We increase the memory usage by a factor of 10.

	Sys	TCAD		QEPCAD B	
1	Parabola	0.144	**27**	0.02	**115**
2	Whitney-umbrella	5.088	895	0.048	895
3	Quartic	8.220	233	0.052	223 (with error)
4	Sphere-catastrophe	2.716	**421**	0.048	**509**
5	Arnon-84	0.184	55	0.024	55
6	Arnon-84-2	0.384	41	0.02	41
7	Real-implicitization	7.664	893	0.052	889 (with warning)
8	Ball-cylindar	3.184	365	0.068	365
9	Termination-term-rewrite	1.084	209	0.02	207
10	Collins-Johnson	72.640	3677	0.32	3673
11	Range-lower-bounds	2.068	**563**	0.184	**4199** (with warning)
12	X-axis-ellipse	225.862	**20143**	3.156	**64625** (with warning)
13	Davenport-Heintz	53.335	4949	0.148	4949
14	Hong-90	2597.878	**27547**	13.852	**79289** (with warning)
15	Solotareff-3	3503.954	66675	4.188	66675
16	Collision	-	-	2.076	45979
17	McCallum-random	-	-	21.797	877
18	Ellipse-cad	-	-	-	-

Table 7.3: Timing (s) and number of cells for TCAD and QEPCAD B

$\mathbb{R}^k$, it firstly constructs a P_{k-1} sign invariant CAD $\mathcal{C}_{k-1}$ of $\mathbb{R}^{k-1}$, such that above each cell of $\mathcal{C}_{k-1}$, the polynomials of P_k are delineable. By do we really need polynomials in P_{k-1} are sign invariant? The answer is no! We only require the polynomials in P_k are sign invariant and the cells are cylindrically arranged. These two requirements corresponds exactly to the IntialPartition and MakeCylindrical steps of TCAD. Thus the algorithm of TCAD computes CAD in a more geometrically intrinsic way and avoids those unnecessary steps in QEPCAD B.

But why TCAD is slower and how can we make it faster? It is slower because there are potentially lots of hidden repeated computations, which are caused by the calling of operations MPD and SMPD. These two operations are originally designed for making constructible sets pairwise disjoint, which however does not guarantee the projections of them onto lower dimensional space are disjoint and thus does not satisfy the cylindricity requirements directly. In the current algorithm of MakeCylindrical, cylindricity is achieved by a repeated "projecting and making disjoint" process, whose side effect is that disjoint lower dimensional cells might by made pairwise disjoint again for many times. A potential solution, which will appear in the future, is to develop an algorithm which not only makes two constructible sets disjoint but also makes the projections of them disjoint. We believe that such an algorithm will greatly improve the efficiency of IntialPartition and MakeCylindrical. Finally, the efficiency of MakeSemiAlgebraic can be improved by simplifying the polynomials appearing in the cylindrical decomposition of a complex space and developing faster algorithms for isolating the real roots of regular chains.

7.6 Application to simplifying elementary functions

Elementary functions, like $\log z$ and $\sqrt{z}$, can be seen as both multi-valued and single-valued functions. Regarding them as single-valued functions often causes problems when one tries to simplify formulas involving those functions [15]. For example, a simplification of $\sqrt{x}\sqrt{y}$ as $\sqrt{xy}$ is invalid since $\sqrt{x}\sqrt{y} \neq \sqrt{xy}$ at $x = y = -1$.

More generally, given an elementary function $f(z)$, we say that a function $g(z)$ is a valid simplification of $f(z)$ if and only if $f(z) - g(z) = 0$ holds for all $z \in \mathbb{C}$. Deciding whether $g(z)$ is a simplification of $f(z)$ is an undecidable problem in its full generality. In [10], the authors propose a method which first computes the branch cuts of elementary functions and then decomposes branch cuts into connected components with CAD and finally tests whether $f(z) = g(z)$ holds at a sample point of each of these connected components. A detailed discussion of their method is beyond the scope of this thesis. We would instead illustrate their idea using the following example: do the following equations hold for all $z \in \mathbb{C}$?

- $\sqrt{z-1}\sqrt{z+1} = \sqrt{z^2-1}$
- $\sqrt{1-z}\sqrt{1+z} = \sqrt{1-z^2}$

To answer this question, one first needs to describe the branch cut of elementary functions. The branch cut of $\sqrt{z}$ is conventionally:

$$\{z \in \mathbb{C} \mid \Re(z) < 0 \ \wedge \ \Im(z) = 0\}. \tag{7.1}$$

If we write z as $x + iy$, the branch cut is the semi-algebraic set $\{(x, y) \in \mathbb{R}^2 \mid x < 0 \wedge y = 0\}$. Applying Equation (7.1), the branch cut of $\sqrt{z-1}$ is $\{z \in \mathbb{C} \mid \Re(z-1) < 0 \ \wedge \ \Im(z-1) = 0\}$. Writing z as $x + iy$, the branch cut is the semi-algebraic set $\mathcal{S}_1 := \{(x, y) \in \mathbb{R}^2 \mid x - 1 < 0 \wedge y = 0\}$. Similarly, we calculate the branch cuts of

$\sqrt{z+1}$, $\sqrt{z^2-1}$ $\sqrt{1-z}$, $\sqrt{1+z}$ and $\sqrt{1-z^2}$. These are respectively

$$\begin{aligned}
\mathcal{S}_2 &:= \{(x, y) \in \mathbb{R}^2 \mid x + 1 < 0 \wedge y = 0\}, \\
\mathcal{S}_3 &:= \{(x, y) \in \mathbb{R}^2 \mid 2xy = 0 \wedge x^2 - y^2 - 1 < 0\}, \\
\mathcal{S}_4 &:= \{(x, y) \in \mathbb{R}^2 \mid x + 1 < 0 \wedge y = 0\}, \\
\mathcal{S}_5 &:= \{(x, y) \in \mathbb{R}^2 \mid -x + 1 < 0 \wedge y = 0\}, \text{ and} \\
\mathcal{S}_6 &:= \{(x, y) \in \mathbb{R}^2 \mid 2xy = 0 \wedge -x^2 + y^2 + 1 < 0\}.
\end{aligned}$$

We collect polynomials appearing in $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$ and form a set $F := \{x+1, x-1, y, 2xy, x^2 - y^2 - 1\}$. By algorithm TCAD, we compute an F-invariant CAD of $\mathbb{R}^2$, which consists of 29 connected cells with a sample point per cell. By evaluating the polynomials in F at these sample points, we obtain 7 cells $C_1, \ldots, C_7$ whose sample points belongs to $\mathcal{S}_1$, $\mathcal{S}_2$ or $\mathcal{S}_3$. Thus the 7 cells form an intersection-free basis of $\mathcal{S}_1$, $\mathcal{S}_2$, $\mathcal{S}_3$. The seven sample points are $(-2, 0)$, $(-1, 0)$, $(-1/2, 0)$, $(0, -1)$, $(0, 0)$, $(0, 1)$ and $(1/2, 0)$.

By virtue of the *Monodromy Theorem* [10], it is sufficient to check whether the formula holds at these sample points. By the subs and simplify commands of Maple, we found that $\sqrt{z-1}\sqrt{z+1} - \sqrt{z^2-1} \neq 0$ at the first and fourth sample points. Thus $\sqrt{z^2-1}$ is not always a valid simplification of $\sqrt{z-1}\sqrt{z+1}$.

Running a similar procedure, we obtain two cells forming an intersection-free basis of $\mathcal{S}_4$, $\mathcal{S}_5$, $\mathcal{S}_6$. The sample points attached to each of them are respectively $(-2, 0)$ and $(2, 0)$. Then it is easy to check that $\sqrt{1-z}\sqrt{1+z} - \sqrt{1-z^2} = 0$ holds at both sample points. Thus $\sqrt{1-z^2}$ is a always a valid simplification of $\sqrt{1-z}\sqrt{1+z}$.

7.7 Conclusion

We have presented a new approach for computing cylindrical algebraic decompositions. Our main motivation is to understand the relations between CADs and triangular decompositions, studying how the efficient techniques developed for the latter ones can benefit to the former ones.

Our method can be applied for solving QE problems directly. However, to solve practical problems efficiently, our method needs to be equipped with existing techniques, like partially built CADs, for utilizing the specific feature of input problems. Such issues will be addressed in future work.

Chapter 8

Triangular Decomposition of Semi-algebraic Systems

Regular chains and triangular decompositions are fundamental and well-developed tools for describing the complex solutions of polynomial systems. This chapter proposes adaptations of these tools focusing on solutions of the real analogue: semi-algebraic systems. We show that any such system can be decomposed into finitely many *regular semi-algebraic systems*. We propose two specifications (full and lazy) of such a decomposition and present corresponding algorithms. Under some simplifying assumptions, the lazy decomposition can be computed in singly exponential time w.r.t. the number of variables. We have implemented our algorithms and the experimental results illustrate their effectiveness.

8.1 Introduction

Regular chains, the output of triangular decompositions of systems of polynomial equations, enjoy remarkable properties. Size estimates play in their favor [47] and permit the design of modular [48] and fast [89] methods for computing triangular decompositions. These features stimulate the development of algorithms and software for solving polynomial systems via triangular decompositions.

For the fundamental case of semi-algebraic systems with rational number coefficients, to which this work is devoted, several algorithms for studying the real solutions of such systems take advantage of the structure of a regular chain. Some are specialized to isolating the real solutions of systems with finitely many complex solutions [135, 41, 12]. Other algorithms deal with parametric polynomial systems via real root classification (RRC) [138] or with arbitrary systems via cylindrical algebraic decompositions (CAD) [36].

In this work, we introduce the notion of a *regular semi-algebraic system*, which in broad terms is the "real" counterpart of the notion of a regular chain. Then we define two notions of a *decomposition of a semi-algebraic system*: one that we call *lazy triangular decomposition*, where the analysis of components of strictly smaller (complex) dimension is deferred, and one that we call *full triangular decomposition* where all cases are worked out. These decompositions are obtained by combining triangular decompositions of algebraic sets over the complex field with a special Quantifier Elimination (QE) method based on RRC techniques.

Definition 8.1. *Let $T \subset \mathbb{Q}[\mathbf{x}]$ be a squarefree regular chain for an ordering of the variables $\mathbf{x} = x_1, \ldots, x_n$. Let $\mathbf{u} = u_1, \ldots, u_d$ and $\mathbf{y} = y_1, \ldots, y_{n-d}$ designate respectively the variables of $\mathbf{x}$ that are free and algebraic w.r.t. T. Let $P \subset \mathbb{Q}[\mathbf{x}]$ be finite and such that each polynomial in P is regular w.r.t. the saturated ideal of T. Define $\mathcal{P} := \{p > 0 \mid p \in P\}$. Let $\mathcal{Q}$ be a quantifier-free formula over $\mathbb{Q}[\mathbf{x}]$ involving only the $\mathbf{u}$ variables. Let S be the semi-algebraic subset of $\mathbb{R}^d$ defined by $\mathcal{Q}$. When $d = 0$, the 0-ary Cartesian product $\mathbb{R}^d$ is treated as a singleton set. We say that $R := [\mathcal{Q}, T, \mathcal{P}]$ (also written as $[R^Q, R^T, R^P]$) is a* regular semi-algebraic system *if:*

(i) S is a non-empty open subset in $\mathbb{R}^d$,

(ii) the regular system $[T, P]$ specializes well at every point u of S (see Section 8.2 for this notion),

(iii) at each point u of S, the specialized system $[T(u), P(u)_>]$ admits real solutions.

The zero set *of R, denoted by $Z_{\mathbb{R}}(R)$, is the set of points $(u, y) \in \mathbb{R}^d \times \mathbb{R}^{n-d}$ such that $\mathcal{Q}(u)$ holds and $t(u, y) = 0$, $p(u, y) > 0$, for all $t \in T$ and all $p \in P$.*

Using the notations of Definition 8.1, Let $R = [\mathcal{Q}, T, \mathcal{P}]$ be a regular semi-algebraic system. Since $\mathcal{Q}$ is open, each connected component C of $\mathcal{Q}$ is locally homeomorphic to the hypercube $(0, 1)^d$. From Property *(ii)*, the zero set $Z_{\mathbb{R}}(R)$ consists of disjoint graphs of continuous semi-algebraic functions defined on each such C. Moreover, from Property *(iii)*, there is at least one such graph. For these reasons, which are formally stated in Theorem 8.1, the regular semi-algebraic system R can be understood as a parameterization of the set $Z_{\mathbb{R}}(R)$. Clearly, the dimension of $Z_{\mathbb{R}}(R)$ is d.

Example 8.1. *For the variables $z > y > x$, we consider two classical surfaces (from* *the Algebraic Surface Gallery*[1]*) called* Sofa *and* Cylinder *with equations:*

$$x^2 + y^3 + z^5 = 0 \text{ and } x^4 + y^2 = 1.$$

The common points of these surfaces with real coordinates can be described as the union of the zero sets of the following 5 *regular semi-algebraic systems* R_1 *to* R_5 *(unspecified* R_i^P *are empty and unspecified* R_i^Q *are "true"):*

$$R_1^T = \begin{cases} z^5 + (1 - x^4)y + x^2 \\ y^2 + x^4 - 1 \end{cases} \quad R_1^Q = \begin{cases} -1 < x < 1 \\ x^{12} - 3x^8 + 4x^4 - 1 \neq 0, \end{cases} \quad R_2^T = \begin{cases} z + 1 \\ y \\ x - 1 \end{cases} \quad R_3^T = \begin{cases} z + 1 \\ y \\ x + 1 \end{cases}$$

$$R_4^T = \begin{cases} z^5 + (1 - x^4)y + x^2 \\ (x^4 - 1)y + x^2 \\ x^{12} - 3x^8 + 4x^4 - 1 \end{cases} \quad R_5^T = \begin{cases} z \\ (x^4 - 1)y - x^2 \\ x^{12} - 3x^8 + 4x^4 - 1 \end{cases}$$

This decomposition is obtained by the algorithms of Section 8.6. The fact that R_2 *to* R_5 *are regular semi-algebraic systems is clear, since each of them consists only of a zero-dimensional squarefree regular chain. For* R_1*, we observe that*

$$(-1 < x < 1) \quad \wedge \quad (x^{12} - 3x^8 + 4x^4 - 1 \neq 0)$$

is a quantifier-free formula[2] *defining an open set* S*; moreover* $p_y := y^2+x^4-1$*, regarded as a univariate polynomial in* y*, admits two distinct real roots for each* $x \in S$ *while* $p_z := z^5 + (1 - x^4)y + x^2$*, as a univariate polynomial in* z*, is squarefree and admits (exactly) one real root for any* $x \in S$ *and any* y *defined by* $y^2 + x^4 - 1 = 0$*. Indeed, the discriminant of* p_z *in* z *is* $3125\,(-y + yx^4 - x^2)^4$ *and the resultant w.r.t.* y *of this latter polynomial and* p_y *is* $9765625\,(x^{12} - 3\,x^8 + 4\,x^4 - 1)^4$.

In Section 8.2 we show that the zero set of any semi-algebraic system $\mathfrak{S}$ can be decomposed as a finite union of zero sets of regular semi-algebraic systems. We call such a decomposition a *full triangular decomposition* (or simply *triangular decomposition* when clear from context) of $\mathfrak{S}$, and denote by RealTriangularize an algorithm to compute it.

[1] `www1-c703.uibk.ac.at/mathematik/project/bildergalerie/gallery.html`

[2] We said 'involving only strict inequalities', but we are using the shorthand $f \neq 0$ for $f > 0 \vee f < 0$.

The existence of such a triangular decomposition can be understood in terms of CAD. Indeed, consider a CAD of the polynomials defining $\mathfrak{S}$ and a cell C where all constraints of $\mathfrak{S}$ are satisfied. The cell C is a connected semi-algebraic set homeomorphic to hypercube $(0,1)^d$, for some d, and from the CAD data (see for instance [36]) one can extract a regular semi-algebraic system R whose zero set is C. However, we should stress the fact that a triangular decomposition of $\mathfrak{S}$ has much less information and structure than a CAD of the polynomials defining $\mathfrak{S}$. For instance, the zero sets of the regular semi-algebraic systems in a triangular decomposition of $\mathfrak{S}$ need not be cylindrically arranged.

Our motivations in introducing this concept of triangular decomposition are threefold. First, we aim at proposing an encoding of the solutions of an arbitrary semi-algebraic system which, as much as possible, is both explicit (thus using "triangular representation" of the components) and compact (thus trying to keep the size of output under control). Secondly, we aim at developing algorithms that are capable of producing either a full description of the solution set, or partial answers (such as dimension information or sample points) at a lower cost than a full description. Thirdly, we aim at proposing an encoding of semi-algebraic sets that can support efficient algorithms for the set theoretical operations on such sets.

Triangular decomposition of algebraic sets come in two flavors (see Section 2.2 of Chapter 2). The first one, proposed by Kalkbrener in [81], focuses on representing the generic points of the irreducible components of the input algebraic set. In [119], Szántó establishes that this representation is computable in singly exponential time w.r.t. the number of variables.

The second one, introduced by Wu [132] and studied by many authors (see [33] and the references therein) represents all the points of the input algebraic set. Our proposed algorithm, RealTriangularize, leads to triangular decompositions of this second type for which it is not known whether or not they can be computed in singly exponential time w.r.t. the number of variables. Meanwhile, we are hoping to obtain an algorithm for decomposing semi-algebraic systems (certainly under some genericity assumptions) that would fit in that complexity class. Moreover, we observe that, in practice, full triangular decompositions are not always necessary and providing information about the components of maximum dimension is often sufficient. These theoretical and practical considerations yield a weaker notion of a decomposition of a semi-algebraic system.

Definition 8.2. *Let $\mathfrak{S} = [F, N_{\geq}, P_{>}, H_{\neq}]$ (see Section 8.2 for this notation) be a semi-algebraic system of $\mathbb{Q}[\mathbf{x}]$ and $Z_{\mathbb{R}}(\mathfrak{S}) \subseteq \mathbb{R}^n$ be its zero set. Denote by d the dimension* *of the constructible set* $\{x \in \mathbb{C}^n \mid f(x) = 0, g(x) \neq 0, \text{ for all } f \in F, g \in P \cup H\}$. *A finite set of regular semi-algebraic systems* $\{R_i \mid i = 1 \cdots t\}$ *is called a* lazy triangular decomposition *of* $\mathfrak{S}$ *if*

- $\cup_{i=1}^{t} Z_{\mathbb{R}}(R_i) \subseteq Z_{\mathbb{R}}(\mathfrak{S})$ *holds, and*
- *there exists* $G \subset \mathbb{Q}[\mathbf{x}]$ *such that the real-zero set* $Z_{\mathbb{R}}(G) \subset \mathbb{R}^n$ *contains* $Z_{\mathbb{R}}(\mathfrak{S}) \setminus (\cup_{i=1}^{t} Z_{\mathbb{R}}(R_i))$ *and the complex-zero set* $V(G) \subset \mathbb{C}^n$ *either is empty or has dimension less than* d.

We denote by LazyRealTriangularize *an algorithm computing such a decomposition. In our software implementation presented hereafter,* LazyRealTriangularize *outputs additional information in order to continue the computations and obtain a full triangular decomposition, if needed. This additional information appears in the form of* unevaluated recursive calls, *explaining the usage of the adjective* lazy *in this type of decompositions.*

Complexity results for lazy triangular decomposition. In Section 8.3, we provide a running time estimate for computing a lazy triangular decomposition of the semi-algebraic system $\mathfrak{S}$ when $\mathfrak{S}$ has no inequations nor inequalities, (that is, when $N_{\geq} = P_{>} = H_{\neq} = \emptyset$ holds) and when F generates a strongly equidimensional ideal of dimension d. We show that one can compute such a decomposition in time singly exponential w.r.t. n. Our estimates are not sharp and are just meant to reach a singly exponential bound. We rely on the work of J. Renagar [109] for quantifier elimination. In Sections 8.4, 8.5 and 8.6 we turn our attention to algorithms that are more suitable for implementation even though they rely on sub-algorithms with a doubly exponential running time w.r.t. d.

A special case of quantifier elimination. By means of triangular decomposition of algebraic sets over $\mathbb{C}$, triangular decomposition of semi-algebraic systems (both full and lazy) reduces to a special case of QE. In Section 8.4, we perform this latter step via the concept of a *fingerprint polynomial set*, which is inspired by that of a *discrimination polynomial set* used for RRC in [138, 136].

Complexity results for fingerprint polynomial set. In Section 8.5, we show that the fingerprint polynomial set of a pre-regular semi-algebraic system R (See Section 8.2 for this notion) can be computed in singly exponential time w.r.t. the number of variables as long as the regular chain part of R is in generic position. The advantage of this result, compared to that of Section 8.3, is that its proof leads to a practical algorithm, actually used in our software implementation. Despite its stronger assumptions, this latter result is practically important since regular chains are often in generic position.

Implementation and experimental results. In Section 8.6 we describe the algorithms that we have implemented for computing triangular decompositions of semi-algebraic systems. Our MAPLE code is part of the `RegularChains` library. We provide experimental data for two groups of well-known problems. In the first group, each input semi-algebraic system consists of equations only while the second group is a collection of semi-algebraic systems from QE problems. To illustrate the difficulty of our test problems, and only for this purpose, we provide timings obtained with other well-known polynomial system solvers which are based on algorithms whose running time estimates are comparable to ours. For this first group we use MAPLE's `Groebner:-Basis` command for computing lexicographical Gröbner bases. For the second group we use a general purpose QE software, QEPCAD B (in non-interactive mode) [19], on the respective QE problems. Our results show that LazyRealTriangularize code solves most of our test problems and more problems than the tools it is compared to, though these solving tools have different specifications.

We conclude this introduction by computing a triangular decomposition of a particular semi-algebraic system taken from [21]. Consider the following question: when does $p(z) = z^3+az+b$ have a non-real root $x+iy$ satisfying $xy < 1$? This problem can be expressed as $(\exists x)(\exists y)[f = g = 0 \wedge y \neq 0 \wedge xy - 1 < 0]$, where $f = \mathrm{Re}(p(x+iy)) = x^3 - 3xy^2 + ax + b$ and $g = \mathrm{Im}(p(x+iy))/y = 3x^2 - y^2 + a$. We call our LazyRealTriangularize command on the semi-algebraic system $f = 0, g = 0, y \neq 0, xy - 1 < 0$ with the variable order $y > x > b > a$. Its first step is to call the Triangularize command of the `RegularChains` library on the algebraic system $f = g = 0$. We obtain one squarefree regular chain $T = [t_1, t_2]$, where $t_1 = g$ and $t_2 = 8x^3 + 2ax - b$, satisfying $V(f, g) = V(T)$. The second step of LazyRealTriangularize is to check whether the polynomials defining inequalities and inequations are regular w.r.t. the saturated ideal of T, which is the case here. The third step is to compute the so called *border polynomial set* (see Section 8.2) which is $B = [h_1, h_2]$ with $h_1 = 4a^3 + 27b^2$ and $h_2 = -4a^3b^2 - 27b^4 + 16a^4 + 512a^2 + 4096$. One can check that the regular system $[T, \{y, xy - 1\}]$ specializes well outside of the hypersurface $h_1h_2 = 0$. The fourth step is to compute the fingerprint polynomial set which yields the quantifier-free formula $\mathcal{Q} = h_1 > 0 \wedge h_2 \neq 0$ telling us that $[\mathcal{Q}, T, 1 - xy > 0]$ is a regular semi-algebraic system. After performing these four steps, (based on Algorithm 30, Section 8.6) the function call LazyRealTriangularize$([f, g, y \neq 0, xy - 1 < 0], [y, x, b, a])$ in our implementation returns the following:

$$\begin{cases} [[t_1 = 0, t_2 = 0, 1 - xy > 0]] & h_1 > 0 \wedge h_2 \neq 0 \\ \texttt{\%LazyRealTriangularize}([t_1 = 0, t_2 = 0, f = 0, & \\ h_1 = 0, 1 - xy > 0, y \neq 0], [y, x, b, a]) & h_1 = 0 \\ \texttt{\%LazyRealTriangularize}([t_1 = 0, t_2 = 0, f = 0, & \\ h_2 = 0, 1 - xy > 0, y \neq 0], [y, x, b, a]) & h_2 = 0 \\ [\,] & \text{otherwise} \end{cases}$$

The above output shows that $\{[\mathcal{Q}, T, 1 - xy > 0]\}$ forms a lazy triangular decomposition of the input semi-algebraic system. Moreover, together with the output of the recursive calls, one obtains a full triangular decomposition. Note that the cases of the two recursive calls correspond to $h_1 = 0$ and $h_2 = 0$. Since LazyRealTriangularize uses the Maple piecewise structure for output format, one simply needs to evaluate the recursive calls with the `value` command, yielding the same result as directly calling RealTriangularize

$$\begin{cases} [[t_1 = 0, t_2 = 0, 1 - xy > 0]] & h_1 > 0 \wedge h_2 \neq 0 \\ [\,] & h_1 = 0 \\ [[t_3 = 0, t_4 = 0, h_2 = 0]] & h_2 = 0 \\ [\,] & \text{otherwise} \end{cases}$$

where $t_3 = xy + 1$ and $t_4 = 2a^3x - a^2b + 32ax - 48b + 18xb^2$.

From this output, after some simplification, one could obtain the equivalent quantifier-free formula, $4a^3 + 27b^2 > 0$, of the original QE problem.

This chapter is based on paper [26] and its enhanced version [27], co-authored with James Davenport, John May, Marc Moreno Maza, Bican Xia and Rong Xiao.

8.2 Triangular decomposition of semi-algebraic systems

In this section, we prove that any semi-algebraic system decomposes into finitely many regular semi-algebraic systems. This latter notion was defined in the introduction.

Semi-algebraic system. Let us consider four finite polynomial subsets $F = \{f_1, \ldots, f_s\}$, $N = \{n_1, \ldots, n_t\}$, $P = \{p_1, \ldots, p_r\}$ and $H = \{h_1, \ldots, h_\ell\}$ of $\mathbb{Q}[x_1, \ldots, x_n]$. Let $N_\geq$ denote the set of the inequalities $\{n_1 \geq 0, \ldots, n_t \geq 0\}$. Let

$P_>$ denote the set of the inequalities $\{p_1 > 0, \ldots, p_r > 0\}$. Let $H_{\neq}$ denote the set of inequations $\{h_1 \neq 0, \ldots, h_\ell \neq 0\}$. We will denote by $[F, P_>]$ the *basic semi-algebraic system* $\{f_1 = 0, \ldots, f_s = 0, p_1 > 0, \ldots, p_r > 0\}$. We denote by $\mathfrak{S} = [F, N_\geq, P_>, H_{\neq}]$ the semi-algebraic system (SAS) which is the conjunction of the following conditions: $f_1 = 0, \ldots, f_s = 0$, $n_1 \geq 0, \ldots, n_t \geq 0$, $p_1 > 0, \ldots, p_r > 0$ and $h_1 \neq 0, \ldots, h_\ell \neq 0$.

Notations for zero sets. In this paper, we use "Z" to denote the zero set in $\mathbb{C}^n$ of a polynomial system, involving equations and inequations, and "$Z_\mathbb{R}$" to denote the zero set in $\mathbb{R}^n$ of a semi-algebraic system.

Good specialization (Definition 6.7 in Section 6.4). Consider a squarefree regular system $[T, H]$ of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$. Recall that $\mathbf{y}$ and $\mathbf{u} = u_1, \ldots, u_d$ stand respectively for $\mathrm{mvar}(T)$ and $\mathbf{x} \setminus \mathbf{y}$. Let $z = (z_1, \ldots, z_d)$ be a point of $\mathbf{K}^d$. We recall that $[T, H]$ *specializes well* at z if: (i) none of the initials of the polynomials in T vanishes modulo the ideal $\langle z_1 - u_1, \ldots, z_d - u_d \rangle$; (ii) the image of $[T, H]$ modulo $\langle z_1 - u_1, \ldots, z_d - u_d \rangle$ is a squarefree regular system.

Border polynomial [138]. Let $[T, H]$ be a squarefree regular system of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$. Let bp be the primitive and square free part of the product of all $\mathrm{res}(\mathrm{der}(t), T)$ and all $\mathrm{res}(h, T)$ for $h \in H$ and $t \in T$. We call bp the *border polynomial* of $[T, H]$ and denote by $\mathsf{BorderPolynomial}(T, H)$ an algorithm to compute it. We call the set of irreducible factors of bp the *border polynomial set* of $[T, H]$. Denote by $\mathsf{BorderPolynomialSet}(T, H)$ an algorithm to compute it. Proposition 8.1, which is an immediate corollary of Lemma 6.2 in Section 6.4, follows from the specialization property of subresultants and states a fundamental property of border polynomials.

Proposition 8.1. *The system $[T, H]$ specializes well at $u \in \mathbf{K}^d$ if and only if the border polynomial $bp(u) \neq 0$.*

Corollary 8.1. *Let $[T, H]$ be a squarefree regular system of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$ and B be its border polynomial set. Let $D \subset \mathbf{k}[\mathbf{u}]$ such that $B \subseteq D$. Then we have*

$$V(\mathrm{sat}(T)) \setminus V(\prod_{h \in H} h) \setminus V(\prod_{f \in D} f) = W(T) \setminus V(\prod_{f \in D} f)$$

and $V(\mathrm{sat}(T)) \cap V(\prod_{h \in H} h) \setminus V(\prod_{f \in D} f) = \emptyset$ hold.

Pre-regular semi-algebraic system. Let $[T, P]$ be a squarefree regular system of $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$. Let bp be the border polynomial of $[T, P]$. Let $B \subset \mathbb{Q}[\mathbf{u}]$ be a polynomial set such that bp divides the product of polynomials in B. We call the triple $[B_{\neq}, T, P_>]$ a *pre-regular semi-algebraic system* of $\mathbb{Q}[\mathbf{x}]$. Its zero set, written as $Z_\mathbb{R}(B_{\neq}, T, P_>)$, is the set $(u, y) \in \mathbb{R}^n$ such that $b(u) \neq 0$, $t(u, y) = 0$, $p(u, y) > 0$, for all $b \in B$, $t \in T$, $p \in P$. Lemma 8.1 and Theorem 8.1 are fundamental properties of pre-regular semi-algebraic systems.

Lemma 8.1. *Let $\mathfrak{S}$ be a semi-algebraic system of $\mathbb{Q}[\mathbf{x}]$. Then there exists finitely many pre-regular semi-algebraic systems $[B_{i\neq}, T_i, P_{i>}]$, $i = 1 \cdots e$, s.t. $Z_{\mathbb{R}}(\mathfrak{S}) = \cup_{i=1}^{e} Z_{\mathbb{R}}(B_{i\neq}, T_i, P_{i>})$.*

Proof. The semi-algebraic system $\mathfrak{S}$ decomposes into basic semi-algebraic systems, by rewriting inequality of type $n \geq 0$ as: $n > 0 \vee n = 0$. Let $[F, P_>]$ be one of those basic semi-algebraic systems. If F is empty, then the triple $[\emptyset, \varnothing, P_>]$, is a pre-regular semi-algebraic system. If F is not empty, by Proposition 8.1 and the specifications of Triangularize and Regularize, one can compute finitely many squarefree regular systems $[T_i, H]$ such that $V(F) \cap Z(P_{\neq}) = \cup_{i=1}^{e} \left(V(T_i) \cap Z(B_{i\neq})\right)$ holds and where B_i is the border polynomial set of the regular system $[T_i, H]$. Hence, we have $Z_{\mathbb{R}}(F, P_>) = \cup_{i=1}^{e} Z_{\mathbb{R}}(B_{i\neq}, T_i, P_>)$, where each $[B_{i\neq}, T_i, P_>]$ is a pre-regular semi-algebraic system. $\square$

Next, we exhibit properties of pre-regular semi-algebraic systems. To this end, we recall the notion of delineability [44]. Assume $n > 1$. Let C be a connected cell in $\mathbb{R}^{n-1}$. A polynomial $p \in \mathbb{R}[x_1, \ldots, x_n]$ is *delineable* on C if the real zeros of p define continuous real-valued functions $\theta_1, \ldots, \theta_s$ such that, for all $\alpha \in C$ we have $\theta_1(\alpha) < \cdots < \theta_s(\alpha)$.

Lemma 8.2 (Theorem 1 in [44])**.** *Let p be a polynomial of $\mathbb{R}[y_1 < \cdots < y_n]$ and C be a connected semi-algebraic subset of $\mathbb{R}^{n-1}$. If init(p) $\neq 0$ on C and the number of distinct complex roots of p is invariant on C, then p is delineable on C.*

Theorem 8.1. *Let $[B_{\neq}, T, P_>]$ be a pre-regular semi-algebraic system of $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$, with T non-empty. Let h be the product of the polynomials in B. Let C be a connected subset of the complement of $h = 0$ in $\mathbb{R}^d$. Then there exist finitely many, say k, continuous semi-algebraic functions $\psi_1(\mathbf{u}), \ldots, \psi_k(\mathbf{u})$ defined on C, such that $Z_{\mathbb{R}}([T, P_>]) = \uplus_{i=1}^{k} \{(\alpha, \psi_i(\alpha)) \mid \alpha \in C\}$ holds, where $\uplus$ denotes a disjoint union. In particular, for each $\alpha \in C$, we have $Z_{\mathbb{R}}([T(\alpha), P_>(\alpha)]) = \{\psi_1(\alpha), \ldots, \psi_k(\alpha)\}$, which is a set of k points.*

Proof. We prove by induction on m, the number of variables in $\mathbf{y} = y_1 < \cdots < y_m$. For $1 \leq i \leq m$, let $P_i = \{p \in P \mid \text{mvar}(p) \leq y_i\}$. Write $T = \{t_1, \ldots, t_m\}$, where polynomials are sorted by main variables.

Case $m = 1$. For any $\alpha \in C$, the regular system $[\{t_1\}, P_1]$ specializes well at α by Proposition 8.1, which implies that $\mathrm{init}(t_1)(\alpha) \neq 0$ and $t_1(\alpha, y_1)$ is a squarefree polynomial in $\mathbb{R}[y_1]$. Therefore, the polynomial t_1 is delineable on C by Lemma 8.2, which implies that the real zero set of t_1 over C consists of finitely many (possibly none) disjoint graphs of continuous functions. Let $\psi_1(\mathbf{u}), \ldots, \psi_{k'}(\mathbf{u})$ be these functions. For $i = 1, \ldots, k'$, the graph of ψ_i over C, denoted by G_i, is a connected semi-algebraic set. Moreover, since $[\{t_1\}, P_1]$ specializes well above C, we deduce that the sign of each $p \in P_1$ does not change above G_i. We pick those ψ_i such that $G_i \cap Z_{\mathbb{R}}(P_{1>}) \neq \emptyset$ holds and renumber them as $\psi_1(\mathbf{u}), \ldots, \psi_k(\mathbf{u})$. Clearly we have $Z_{\mathbb{R}}([t_1, P_{1>}]) = \cup_{i=1}^{k} \{(\alpha, \psi_i(\alpha)) \mid \alpha \in C)\}$ holds.

Case $m > 1$. Assume that the conclusion holds for the pre-regular semi-algebraic system $[B_{\neq}, \{t_1, \ldots, t_{m-1}\}, P_{m-1>}]$, that is, there exist k continuous semi-algebraic functions $\psi_1(\mathbf{u}), \ldots, \psi_k(\mathbf{u})$ defined on C such that

$$Z_{\mathbb{R}}([\{t_1, \ldots, t_{m-1}\}, P_{m-1>}]) = \cup_{i=1}^{k} \{(\alpha, \psi_i(\alpha)) \mid \alpha \in C\}$$

holds. For $i = 1, \ldots, k$, let $G_i := \{(\alpha, \psi_i(\alpha)) \mid \alpha \in C\}$. Then each G_i is a connected semi-algebraic set. Moreover, by Proposition 8.1, $[T, P]$ specializes well above $Z_{\mathbb{R}}(B_{\neq})$, which implies that $[\{t_m\}, P_m]$ specializes well above G_i. By similar arguments as in the proof of the case $m = 1$, we deduce that for each $i = 1, \ldots, k$, there exists $n_i \geq 0$ continuous semi-algebraic functions $\psi_{i,1}(\mathbf{u}, y_1, \ldots, y_{m-1}), \ldots, \psi_{i,n_i}(\mathbf{u}, y_1, \ldots, y_{m-1})$ defined on G_i such that $\{(\gamma, \beta) \in \mathbb{R}^{d+m-1} \times \mathbb{R} \mid \gamma \in G_i, t_m(\gamma, \beta) = 0, p(\gamma, \beta) > 0 \text{ for all } p \in P_m\}$ equals to $\cup_{j=1}^{n_i} \{(\gamma, \psi_{i,j}(\gamma)) \mid \gamma \in G_i\}$, which implies that

$$Z_{\mathbb{R}}([T, P_>]) = \cup_{i=1}^{k} \cup_{j=1}^{n_i} \{(\alpha, \psi_i(\alpha), \psi_{i,j}(\alpha, \psi_i(\alpha))) \mid \alpha \in C\}$$

holds. Clearly $(\psi_i(\mathbf{u}), \psi_{i,j}(\mathbf{u}, \psi_i(\mathbf{u})))$, where $i = 1, \ldots, k$, $j = 1, \ldots, n_i$, are continuous semi-algebraic functions defined on C, so the conclusion holds. □

Lemma 8.3. *Let* $[B_{\neq}, T, P_>]$ *be a pre-regular semi-algebraic system of* $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$. *One can decide whether its zero set is empty or not. If it is not empty, then one can compute a regular semi-algebraic system* $[\mathcal{Q}, T, P_>]$ *whose zero set is the same as that of* $[B_{\neq}, T, P_>]$.

Proof. If $T = \varnothing$, we can test whether the zero set of $[B_{\neq}, P_>]$ is empty or not, for instance using CAD. If it is empty, we are done. Otherwise, defining $\mathcal{Q} = B_{\neq} \wedge P_>$, $[\mathcal{Q}, T, P_>]$ is a regular semi-algebraic system whose zero set equals that of $[B_{\neq}, T, P_>]$. If T is not empty, we solve the quantifier elimination problem $\exists \mathbf{y}(B(\mathbf{u}) \neq 0, T(\mathbf{u}, \mathbf{y}) =$

$0, P(\mathbf{u}, \mathbf{y}) > 0)$ and let $\mathcal{Q}$ be the resulting formula. By Theorem 8.1, above each connected component of $B(\mathbf{u}) \neq 0$, the number of real zeros of the system $[B_{\neq}, T, P_{>}]$ is constant. Hence, we claim that the zero set defined by $\mathcal{Q}$ is the union of the connected components of $B(\mathbf{u}) \neq 0$ above which $[B_{\neq}, T, P_{>}]$ possesses at least one solution. If $\mathcal{Q}$ is false, we are done. Otherwise, $\mathcal{Q}$ defines a nonempty open set of $\mathbb{R}^d$ and $[\mathcal{Q}, T, P_{>}]$ is a regular semi-algebraic system whose zero set equals that of $[B_{\neq}, T, P_{>}]$. □

Theorem 8.2. *Let $\mathfrak{S}$ be a semi-algebraic system of $\mathbb{Q}[\mathbf{x}]$. Then one can compute a (full) triangular decomposition of $\mathfrak{S}$, that is, as defined in the introduction, finitely many regular semi-algebraic systems such that the union of their zero sets is the zero set of $\mathfrak{S}$.*

Proof. This follows from Lemma 8.1 and 8.3. □

8.3 Complexity results for computing a lazy triangular decomposition: a theoretical perspective

We prove that, under some genericity assumptions, a lazy triangular decomposition of a polynomial system is computed in singly exponential time w.r.t. the number of variables. First, we state complexity estimates for basic multivariate polynomial operations.

Complexity of basic polynomial operations. Let $p, q \in \mathbb{Q}[\mathbf{x}]$ be polynomials with respective total degrees δ_p, δ_q, and let $x \in \mathbf{x}$. Let $\hbar_p, \hbar_q, \hbar_{pq}$ and $\hbar_r$ be the *height* (that is, the bit size of the maximum absolute value of the numerator or denominator of a coefficient) of p, q, the product pq and the resultant $\mathrm{res}(p, q, x)$, respectively; let $\delta := \max(\delta_p, \delta_q)$ and $\hbar := \max(\hbar_p, \hbar_q)$. In [50], it is proved that $\gcd(p, q)$ can be computed within $O(n^{2\delta+1}\hbar^3)$ bit operations. It is easy to establish that $\hbar_{pq}$ and $\hbar_r$ are respectively upper bounded by $\hbar_p + \hbar_q + n\log(\min(\delta_p, \delta_q) + 1)$ and $\delta_q\hbar_p + \delta_p\hbar_q + n\delta_q\log(\delta_p + 1) + n\delta_p\log(\delta_q + 1) + \log((\delta_p + \delta_q)!)$. Finally, according to [76], the bit operations of p pseudo-dividing q w.r.t. x is $O((\delta + 1)^{3n}\hbar^2)$; let M be a $k \times k$ matrix over $\mathbb{Q}[\mathbf{x}]$, δ (resp. $\hbar$) be the maximum total degree (resp. height) of an element of M, then $\det(M)$ can be computed within $O(k^{2n+5}(\delta + 1)^{2n}\hbar^2)$ bit operations.

We turn now to the main subject of this section, that is, complexity estimates for a lazy triangular decomposition of a polynomial system under some genericity assumptions. Let $F \subset \mathbb{Q}[\mathbf{x}]$. A lazy triangular decomposition (defined in the Introduction) of the semi-algebraic system $\mathfrak{S} = [F, \emptyset, \emptyset, \emptyset]$, involving only equations, is obtained by Algorithm 26.

Algorithm 26: $\mathsf{LazyRealTriangularize}(\mathfrak{S})$

Input: a semi-algebraic system $\mathfrak{S} = [F, \emptyset, \emptyset, \emptyset]$
Output: a lazy triangular decomposition of $\mathfrak{S}$

```
𝔗 := Triangularize(F, mode = Kalkbrener)
for T_i ∈ 𝔗 do
    bp_i := BorderPolynomial(T_i, ∅)
    solve ∃y(bp_i(u) ≠ 0, T_i(u, y) = 0); let Q_i be the resulting quantifier-free
    formula
    if Q_i ≠ false then  output [Q_i, T_i, ∅]
```

Proof of Algorithm 26. The termination of the algorithm is obvious. Let us prove its correctness. Let $R_i = [\mathcal{Q}_i, T_i, \emptyset]$, for $i = 1 \cdots t$ be the output of Algorithm 26 and let T_j for $j = t+1 \cdots s$ be the regular chains such that $\mathcal{Q}_j = false$. By Lemma 8.3, each R_i is a regular semi-algebraic system. For $i = 1 \cdots s$, define $F_i = \mathrm{sat}(T_i)$. Then we have $V(F) = \cup_{i=1}^{s} V(F_i)$, where each F_i is equidimensional. For each $i = 1 \cdots s$, by Proposition 8.1, we have $V(F_i) \setminus V(bp_i) = V(T_i) \setminus V(bp_i)$. Moreover, we have $V(F_i) = (V(F_i) \setminus V(bp_i)) \cup V(F_i \cup \{bp_i\})$. Hence, $Z_{\mathbb{R}}(R_i) = Z_{\mathbb{R}}(T_i) \setminus Z_{\mathbb{R}}(bp_i) \subseteq Z_{\mathbb{R}}(F_i) \subseteq Z_{\mathbb{R}}(F)$ holds. In addition, since bp_i is regular modulo F_i, we have

$$
\begin{array}{rcl}
Z_{\mathbb{R}}(F) \setminus \cup_{i=1}^{t} Z_{\mathbb{R}}(R_i) & = & \cup_{i=1}^{s} Z_{\mathbb{R}}(F_i) \setminus \cup_{i=1}^{t} Z_{\mathbb{R}}(R_i) \\
& \subseteq & \cup_{i=1}^{s} Z_{\mathbb{R}}(F_i) \setminus (Z_{\mathbb{R}}(T_i) \setminus Z_{\mathbb{R}}(bp_i)) \\
& \subseteq & \cup_{i=1}^{s} Z_{\mathbb{R}}(F_i \cup \{bp_i\}),
\end{array}
$$

and $\dim\left(\cup_{i=1}^{s} V(F_i \cup \{bp_i\})\right) < \dim(V(F))$. So the R_i, for $i = 1 \cdots t$, form a lazy triangular decomposition of $\mathfrak{S}$. $\square$

In this section, under some genericity assumptions for F, we establish running time estimates for Algorithm 26, see Theorem 8.4. This is achieved through Proposition 8.2 (which gives running time and output size estimates for a Kalkbrener triangular decomposition of an algebraic set) and Theorem 8.3 (which states running time and output size estimates for a border polynomial computation). Our assumptions for these results are the following:

($\mathbf{H_0}$) $V(F)$ is equidimensional of dimension d,

($\mathbf{H_1}$) $x_1, \ldots, x_d$ are algebraically independent modulo each associated prime ideal of the ideal generated by F in $\mathbb{Q}[\mathbf{x}]$,

($\mathbf{H_2}$) F consists of $m := n - d$ polynomials, $f_1, \ldots, f_m$.

Hypotheses ($\mathbf{H_0}$) and ($\mathbf{H_1}$) are equivalent to the existence of regular chains $T_1, \ldots, T_e$ of $\mathbb{Q}[x_1, \ldots, x_n]$ such that $x_1, \ldots, x_d$ are free w.r.t. each of $T_1, \ldots, T_e$ and such that we have $V(F) = \overline{W(T_1)} \cup \cdots \cup \overline{W(T_e)}$.

Denote by δ, $\hbar$ respectively the maximum total degree and height of $f_1, \ldots, f_m$. In her PhD Thesis [119], Á. Szántó describes an algorithm which computes a Kalkbrener triangular decomposition, $T_1, \ldots, T_e$, of $V(F)$. Under hypotheses ($\mathbf{H_0}$) to ($\mathbf{H_2}$), this algorithm runs in time $m^{O(1)}(\delta^{O(n^2)})^{d+1}$ counting operations in $\mathbb{Q}$, while the total degrees of the polynomials in the output are bounded by $n\delta^{O(m^2)}$. In addition, $T_1, \ldots, T_e$ are square free, *strongly normalized* [103] and *reduced* [6].

From $T_1, \ldots, T_e$, we obtain regular chains $E_1, \ldots, E_e$ forming another Kalkbrener triangular decomposition of $V(F)$, as follows. Let $i = 1 \cdots e$ and $j = (d+1) \cdots n$. Let $t_{i,j}$ be the polynomial of T_i with x_j as main variable. Let $e_{i,j}$ be the primitive part of $t_{i,j}$ regarded as a polynomial in $\mathbb{Q}[x_1, \ldots, x_d][x_{d+1}, \ldots, x_n]$. Define $E_i = \{e_{i,d+1}, \ldots, e_{i,n}\}$. According to the complexity results for polynomial operations stated at the beginning of this section, this transformation can be done within $\delta^{O(m^4)O(n)}$ operations in $\mathbb{Q}$.

Dividing $e_{i,j}$ by its initial we obtain a monic polynomial $d_{i,j}$ of the polynomial ring $\mathbb{Q}(x_1, \ldots, x_d)[x_{d+1}, \ldots, x_n]$. Denote by D_i the regular chain $\{d_{i,d+1}, \ldots, d_{i,n}\}$. Observe that D_i is the reduced lexicographic Gröbner basis of the radical ideal it generates in $\mathbb{Q}(x_1, \ldots, x_d)[x_{d+1}, \ldots, x_n]$. So Theorem 1 in [47] applies to each regular chain D_i. For each polynomial $d_{i,j}$, this theorem provides height and total degree estimates expressed as functions of the *degree* [22] and the *height* [108, 82] of the algebraic set $\overline{W(D_i)}$. Note that the degree and height of $\overline{W(D_i)}$ are upper bounded by those of $V(F)$. Write $d_{i,j} = \Sigma_\mu \frac{\alpha_\mu}{\beta_\mu} \mu$ where each $\mu \in \mathbb{Q}[x_{d+1}, \ldots, x_n]$ is a monomial and α_μ, β_μ are in $\mathbb{Q}[x_1, \ldots, x_d]$ such that $\gcd(\alpha_\mu, \beta_\mu) = 1$ holds. Let γ be the lcm of the β_μ's. Then for γ and each α_μ:

- the total degree is bounded by $2\delta^{2m}$ and,
- the height by $O(\delta^{2m}(m\hbar + dm \log(\delta) + n\log(n)))$.

Multiplying $d_{i,j}$ by γ brings $e_{i,j}$ back. We deduce the height and total degree estimates for each $e_{i,j}$ below.

Proposition 8.2. *Under the hypotheses* ($\mathbf{H_0}$), ($\mathbf{H_1}$), ($\mathbf{H_2}$), *the Kalkbrener triangular decomposition $E_1, \ldots, E_e$ of $V(F)$ can be computed in $\delta^{O(m^4)O(n)}$ operations in $\mathbb{Q}$. In addition, every polynomial $e_{i,j}$ has total degree upper bounded by $4\delta^{2m} + \delta^m$, and has height upper bounded by $O(\delta^{2m}(m\hbar + dm\log(\delta) + n\log(n)))$.*

Next we estimate running time and output size for a border polynomial computation.

Theorem 8.3. *Let $R = [T, P]$ be a squarefree regular system of $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$, with $m = \#T$ and $\ell = \#P$. Let bp be the border polynomial of R. Denote by δ_R, $\hbar_R$ respectively the maximum total degree and height of a polynomial in R. Then the total degree of bp is upper bounded by $(\ell + m)2^{m-1}{\delta_R}^m$, and bp can be computed within $(n\ell + nm)^{O(n)}(2\delta_R)^{O(n)O(m)}{\hbar_R}^3$ bit operations.*

Proof. Define $G := P \cup \{\mathrm{der}(t) \mid t \in T\}$. We need to compute the $\ell + m$ iterated resultants $\mathrm{res}(g, T)$, for all $g \in G$. Let $g \in G$. Observe that the total degree and height of g are bounded by δ_R and $\hbar_R + \log(\delta_R)$ respectively. Define $r_{m+1} := g$, $\ldots$, $r_i := \mathrm{res}(t_i, r_{i+1}, y_i)$, $\ldots$, $r_1 := \mathrm{res}(t_1, r_2, y_1)$. Let $i \in \{1, \ldots, m\}$. Denote by δ_i and $\hbar_i$ the total degree and height of r_i, respectively. Using the complexity estimates stated at the beginning of this section, we have $\delta_i \le 2^{m-i+1}{\delta_R}^{m-i+2}$ and $\hbar_i \le 2\delta_{i+1}(\hbar_{i+1} + n\log(\delta_{i+1} + 1))$. Therefore, we have $\hbar_i \le (2\delta_R)^{O(m^2)}n^{O(m)}\hbar_R$. From these size estimates, one can deduce that each resultant r_i (thus the iterated resultants) can be computed within $(2\delta_R)^{O(mn)+O(m^2)}n^{O(m)}{\hbar_R}^2$ bit operations, by the complexity of computing a determinant stated at the beginning of this section. Hence, the product of all iterated resultants has total degree and height bounded by $(\ell+m)2^{m-1}{\delta_R}^m$ and $(\ell+m)(2\delta_R)^{O(m^2)}n^{O(m)}\hbar_R$, respectively. Thus, the primitive and squarefree part of this product can be computed within $(n\ell+nm)^{O(n)}(2\delta_R)^{O(n)O(m)}{\hbar_R}^3$ bit operations, based on the complexity of a polynomial gcd computation stated at the beginning of this section. $\square$

Theorem 8.4. *From the Kalkbrener triangular decomposition $E_1, \ldots, E_e$ of Proposition 8.2, a lazy triangular decomposition of $f_1 = \cdots = f_m = 0$ can be computed in $\left(\delta^{n^2} n 4^n\right)^{O(n^2)} \hbar^{O(1)}$ bit operations. Thus, under the hypotheses $(\mathbf{H_0})$, $(\mathbf{H_1})$ and $(\mathbf{H_2})$, a lazy triangular decomposition of this system is computed from the input polynomials in singly exponential time w.r.t. n, counting operations in $\mathbb{Q}$.*

Proof. For each $i \in \{1 \cdots e\}$, let bp_i be the border polynomial of $[E_i, \emptyset]$ and let $\hbar_{R_i}$ (resp. δ_{R_i}) be the height (resp. the total degree) bound of the polynomials in the pre-regular semi-algebraic system $R_i = [\{bp_i\}_{\neq}, E_i, \emptyset]$. According to Algorithm 26, the remaining task is to solve the QE problem $\exists \mathbf{y}(bp_i(\mathbf{u}) \neq 0, E_i(\mathbf{u}, \mathbf{y}) = 0)$ for each $i \in \{1 \cdots e\}$, which can be solved within $((m+1)\delta_{R_i})^{O(dm)}\, \hbar_{R_i}^{O(1)}$ bit operations, based on the results of [109]. The conclusion follows from the size estimates in Proposition 8.2 and Theorem 8.3. $\square$

8.4 Quantifier elimination via real root classification

In Section 8.3, we saw that in order to compute a triangular decomposition of a semi-algebraic system, a key step was to solve the following quantifier elimination problem:

$$\exists \mathbf{y}(B(\mathbf{u}) \neq 0, T(\mathbf{u}, \mathbf{y}) = 0, P(\mathbf{u}, \mathbf{y}) > 0), \tag{8.1}$$

where $[B_{\neq}, T, P_{>}]$ is a pre-regular semi-algebraic system of $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$. This problem is an instance of the so-called *real root classification* (RRC) [139]. In this section, we show how to solve this problem when B is what we call a *fingerprint polynomial set*.

Definition 8.3. *Let $R := [B_{\neq}, T, P_{>}]$ be a pre-regular semi-algebraic system of $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$. Let $D \subset \mathbb{Q}[\mathbf{u}]$. Let dp be the product of all polynomials in D. We call D a* fingerprint polynomial set *(FPS) of R if:*

(*i*) *for all $\alpha \in \mathbb{R}^d$, for all $b \in B$ we have: $dp(\alpha) \neq 0 \Longrightarrow b(\alpha) \neq 0$,*

(*ii*) *for all $\alpha, \beta \in \mathbb{R}^d$ with $\alpha \neq \beta$, $dp(\alpha) \neq 0$ and $dp(\beta) \neq 0$: if $p(\alpha)$ and $p(\beta)$ have the same sign for all $p \in D$, then $R(\alpha)$ has real solutions if and only if $R(\beta)$ does.*

Now, we present a method for constructing an FPS based on CAD projection operators.

Open projection operator [116, 17]. Hereafter in this section, we let $\mathbf{u} = u_1 < \cdots < u_d$ be ordered variables. Let $p \in \mathbb{Q}[\mathbf{u}]$ be non-constant. We denote by $\text{factor}(p)$ the set of the non-constant irreducible factors of p. For $A \subset \mathbb{Q}[\mathbf{u}]$, we define $\text{factor}(A) = \cup_{p \in A} \text{factor}(p)$. Let C_d (resp. C_0) be the set of the polynomials in $\text{factor}(p)$ with main variable equal to (resp. less than) u_d. The *open projection operator* (oproj) w.r.t. variable u_d maps p to a set of polynomials of $\mathbb{Q}[u_1, \ldots, u_{d-1}]$ defined below:

$$\begin{array}{rcl} \text{oproj}(p, u_d) & := & C_0 \ \cup \ \bigcup_{f, g \in C_d,\, f \neq g} \ \text{factor}(\text{res}(f, g, u_d)) \\ & & \cup \ \bigcup_{f \in C_d} \ \text{factor}(\text{init}(f, u_d) \cdot \text{discrim}(f, u_d)). \end{array}$$

Then, we define: $\text{oproj}(A, u_d) := \text{oproj}(\Pi_{p \in A}\, p, u_d)$.

Augmentation. Let $A \subset \mathbb{Q}[\mathbf{u}]$ and $x \in \{u_1, \ldots, u_d\}$. Denote by $\text{der}(A, x)$ the *derivative closure* of A w.r.t. x, that is, $\text{der}(A, x) := \cup_{p \in A} \{\text{der}^{(i)}(p, x) \mid 0 \leq i < \deg(p, x)\}$. The *open augmented projected factors* of A is denoted by $\text{oaf}(A)$ and defined as follows. Let k be the smallest positive integer such that $A \subset \mathbb{Q}[u_1, \ldots, u_k]$ holds. Denote by C the set $\text{factor}(\text{der}(A, u_k))$; we have

- if $k = 1$, then $\mathrm{oaf}(A) := C$;
- if $k > 1$, then $\mathrm{oaf}(A) := C \cup \mathrm{oaf}(\mathrm{oproj}(C, u_k))$.

Proposition 8.3. *Let* $A \subset \mathbb{Q}[\mathbf{u}]$ *be finite and let* σ *be an arbitrary map from* $\mathrm{oaf}(A)$ *to the set of signs* $\{-1, +1\}$*. We define:*

$$S_d := \bigcap\nolimits_{p \in \mathrm{oaf}(A)} \{u \in \mathbb{R}^d \mid p(u)\, \sigma(p) > 0\}.$$

Then the set S_d *is either empty or a connected open set in* $\mathbb{R}^d$*.*

Proof. By induction on d. When $d = 1$, the conclusion follows from Thom's Lemma [9]. Assume $d > 1$. If d is not the smallest positive integer k such that $A \subset \mathbb{Q}[u_1, \ldots, u_k]$ holds, then S_d writes $S_{d-1} \times \mathbb{R}$ and the conclusion follows by induction. Otherwise, write $\mathrm{oaf}(A)$ as $C \cup E$, where $C = \mathrm{factor}(\mathrm{der}(A, u_d))$ and $E = \mathrm{oaf}(\mathrm{oproj}(C, u_d))$. We have: $E \subset \mathbb{Q}[u_1, \ldots, u_{d-1}]$. Let $\mathrm{M} = \cap_{p \in E} \{u \in \mathbb{R}^{d-1} \mid p(u)\sigma(p) > 0\}$. If M is empty then so is S_d and the conclusion is clear. From now on assume M not empty. Then, by induction hypothesis, M is a connected open set in $\mathbb{R}^{d-1}$. By the definition of the operator oproj and Lemma 8.2, the product of the polynomials in C is delineable over M w.r.t. u_d. Moreover, C is derivative closed (may be empty) w.r.t. u_d. Therefore $\cap_{p \in \mathrm{oaf}(A)} \{u \in \mathbb{R}^d \mid p(u)\, \sigma(p) > 0\} \subset M \times \mathbb{R}$ is either empty or a connected open set by Thom's Lemma. □

Theorem 8.5. *Let* $R := [B_{\neq}, T, P_{>}]$ *be a pre-regular semi-algebraic system of* $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$*. The polynomial set* $\mathrm{oaf}(B)$ *is a fingerprint polynomial set of* R*.*

Proof. Recall that the border polynomial bp of $[T, P]$ divides the product of the polynomials in B. We have $\mathrm{factor}(B) \subseteq \mathrm{oaf}(B)$. So $\mathrm{oaf}(B)$ clearly satisfies (i) in Definition 8.3. Let us prove (ii). Let dp be the product of the polynomials in $\mathrm{oaf}(B)$. Let $\alpha, \beta \in \mathbb{R}^d$ such that both $dp(\alpha) \neq 0$, $dp(\beta) \neq 0$ hold and the signs of $p(\alpha)$ and $p(\beta)$ are equal for all $p \in \mathrm{oaf}(B)$. Then, by Proposition 8.3, α and β belong to the same connected component of $dp(\mathbf{u}) \neq 0$, and thus to the same connected component of $B(\mathbf{u}) \neq 0$. Therefore the number of real solutions of $R(\alpha)$ and that of $R(\beta)$ are the same by Theorem 8.1. □

From now on, let us assume that the set B in the pre-regular semi-algebraic system $R = [B_{\neq}, T, P_{>}]$ is an FPS of R. We solve the quantifier elimination problem (8.1) in three steps: (s_1) compute at least one sample point in each connected component of the semi-algebraic set defined by $B(\mathbf{u}) \neq 0$; (s_2) for each sample point α such that the specialized system $R(\alpha)$ possesses real solutions, compute the sign of $b(\alpha)$ for each $b \in B$; (s_3) generate the corresponding quantifier-free formulas.

In practice, when the set B is not an FPS, one adds some polynomials from $\mathrm{oaf}(B)$, using a heuristic procedure (for instance one by one) until Property (ii) of the definition of an FPS is satisfied. This strategy is implemented in Algorithm 28 of Section 8.6.

8.5 Complexity results for computing a fingerprint polynomial set: a practical perspective

Let $R := [B_{\neq}, T, P_{>}]$ be a pre-regular semi-algebraic system of $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$, where $\mathbf{u}$ stands for the free variables of T and $\mathbf{y} = y_1 < \cdots < y_m$ are the main variables of T. We write $P = \{p_1, \ldots, p_\ell\}$ and $T = \{t_1, \ldots, t_m\}$. In this section, we always assume that T is in *generic position*, that is, the main degree of t_i is 1 for $1 < i \leq m$. Under such an assumption, we show that a fingerprint polynomial set of R can be computed in singly exponential time w.r.t. the number of variables. Note that the construction in Section 8.4 is doubly exponential [20]. Since a regular chain is often in generic position and detecting this shape is easy, this new construction leads to a practical and more effective way for computing fingerprint polynomial set, which has been integrated in our tools.

To achieve this, we present an alternative way (w.r.t. the one presented in last section) to construct a fingerprint polynomial set of R. This new method relies on a tool called *generalized discriminant sequence* (GDS) for counting the number of real solutions of a univariate polynomial with parametric coefficients, which we review as follows.

Definition 8.4 ([138, 140]). *Let $p, q \in \mathbb{R}[x]$. We denote by p' the derivative of p w.r.t. x. Let $r := \mathrm{rem}(p'q, p, x)$ be the Euclidean remainder of $p'q$ divided by p. Let $s := \deg(p, x)$ and write $p = a_0x^s + \cdots + a_s$, $r = c_0x^{s-1} + \cdots + c_{s-1}$. The following $2s \times 2s$ matrix*

$$(m_{ij}) = \begin{pmatrix} a_0 & a_1 & a_2 & \cdots & a_s & & & & \\ 0 & c_0 & c_1 & \cdots & c_{s-1} & & & & \\ & & \ddots & \ddots & \ddots & & & & \\ & & a_0 & a_1 & a_2 & \cdots & a_s & & \\ & & 0 & c_0 & c_1 & \cdots & c_{s-1} & & \end{pmatrix}$$

is called the generalized discrimination matrix *of p w.r.t. q. For $i = 1 \cdots s$, we denote by* $\mathrm{gds}_i(p, q, x)$*, the $2i$-th leading principal minor of the above matrix and call* $\mathrm{gds}_1(p, q, x), \ldots, \mathrm{gds}_s(p, q, x)$ *the* generalized discriminant sequence *of p w.r.t. q, denoted by* $\mathrm{gds}(p, q, x)$*. We write* $\{\mathrm{gds}(p, q, x)\}$ *the set consisting of the elements of* $\mathrm{gds}(p, q, x)$*.*

Notation 8.1. *Let p and q be two polynomials in $\mathbb{R}[x]$. Denote by* $\mathrm{TaQ}(p, q)$ *the number* $\#\{x \mid p = 0,\ q > 0\} - \#\{x \mid p = 0,\ q < 0\}$*, the* Tarski query *[9] of p w.r.t. q.*

Remark 8.1. *The elements in the generalized discriminant sequence of p w.r.t. q are in one-to-one correspondence (up to a power of a_0 and a power of -1) with the* signed subresultant coefficients *[9] of p and r. One can compute* $\mathrm{TaQ}(p, q)$ *merely from the signs of the elements in* $\mathrm{gds}(p, q, x)$*, see Theorem 4.32 in [9] or Theorem 3.2.1 in [140]: given two pairs of polynomials (p_i, q_i) $(i = 1, 2)$ with* $\deg(p_1) = \deg(p_2) = s$*, if* $\mathrm{sign}(\mathrm{gds}_j(p_1, q_1, x)) = \mathrm{sign}(\mathrm{gds}_j(p_2, q_2, x))$ *holds for all $j = 1, \ldots, s$, then* $\mathrm{TaQ}(p_1, q_1) = \mathrm{TaQ}(p_2, q_2)$ *holds.*

We prove Lemmas 8.4 and 8.5 for completness; similar results appear in [138, 140]. Let $k > 0$ be an integer. Let p and $q_1, q_2, \ldots, q_k$ be polynomials from $\mathbb{R}[x]$ with $\gcd(p, q_j) = 1$ for each $j = 1, \ldots, k$. Lemma 8.4 shows that in this case, the numbers $\#\{x \mid p = 0,\ q_1\sigma_1 0, \ldots, q_k\sigma_k 0\}$ with $\sigma_1, \ldots, \sigma_k \in \{>, <\}$ can be computed from the numbers in $\{\mathrm{TaQ}(p, \prod_{j=1}^{k} q_j^{e_j}) \mid e_1, \ldots, e_k \in \{0, 1\}\}$ by solving a linear system with fixed coefficients.

Denote $\mathbf{M} := \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$ and let $\mathbf{M}_1 := \mathbf{M}$. For $i = 1, \ldots, k-1$, denote by $\mathbf{M}_{i+1}$ the $2^i \times 2^i$ matrix obtained by replacing each element e of $\mathbf{M}$ with $e\mathbf{M}_i$. It is easy to deduce that $\det(\mathbf{M}_{i+1}) = 2^{2^i} \det(\mathbf{M}_i)^2$ from its block structure, which implies that all $\mathbf{M}_i$ $(i = 1, \ldots, k)$ are nonsingular.

Denote by $\mathbf{S}_1$ the list of constraints $[q_1 > 0, q_1 < 0]$, by $\mathbf{P}_1$ the polynomial list $[1, q_1]$. For $i = 1, \ldots, k-1$, denote by $\mathbf{S}_{i+1}$ the list of constraints

$$[\mathbf{S}_i[1] \wedge q_{i+1} > 0, \ldots, \mathbf{S}_i[2^i] \wedge q_{i+1} > 0, \mathbf{S}_i[1] \wedge q_{i+1} < 0, \ldots, \mathbf{S}_i[2^i] \wedge q_{i+1} < 0],$$

by $\mathbf{P}_{i+1}$ the polynomial list $[\mathbf{P}_i[1], \ldots, \mathbf{P}_i[2^i], \mathbf{P}_i[1] \cdot q_{i+1}, \ldots, \mathbf{P}_i[2^i] \cdot q_{i+1}]$. It is easy to deduce that $\mathbf{S}_i$ and $\mathbf{P}_i$ are of length 2^i.

Let $\mathbf{T}_k$ be $[\mathrm{TaQ}(p, \mathbf{P}_k[1]), \mathrm{TaQ}(p, \mathbf{P}_k[2]), \ldots, \mathrm{TaQ}(p, \mathbf{P}_k[2^k])]$. Let $\mathbf{N}_k$ be $[\#\{x \mid p = 0,\ \mathbf{S}_k[1]\}, \#\{x \mid p = 0,\ \mathbf{S}_k[2]\}, \ldots, \#\{x \mid p = 0,\ \mathbf{S}_k[2^k]\}]$. We observe that each of $\mathbf{T}_k$ and $\mathbf{N}_k$ is a list of 2^k non-negative integers.

Lemma 8.4 ([140]). *Using the above notations* $\mathbf{M}_k$, $\mathbf{T}_k$, $\mathbf{N}_k$ *and viewing* $\mathbf{T}_k$ *and* $\mathbf{N}_k$ *as vectors, we have* $\mathbf{N}_k = \mathbf{M}_k^{-1} \times \mathbf{T}_k$.

Proof. Consider the system of linear equations $\mathbf{M}_k \times \mathbf{X} = \mathbf{T}_k$ with $\mathbf{X}$ as unknown vector, one can verify that $\mathbf{X} = \mathbf{N}_k$ is the solution. Here, we only verify the base case, namely $k = 1$. Since $\gcd(p, q_1) = 1$, we have

$$\#\{x \mid p = 0,\, q_1 > 0\} + \#\{x \mid p = 0,\, q_1 < 0\} = \#\{x \mid p = 0\} = \mathrm{TaQ}(p, 1).$$

Moreover $\#\{x \mid p = 0,\, q_1 > 0\} - \#\{x \mid p = 0,\, q_1 < 0\}$ equals $\mathrm{TaQ}(p, q_1)$ by definition. □

Let p and q be two univariate polynomials of x with coefficients in $\mathbb{Q}[\mathbf{u}]$. The *signed pseudo-remainder* (see [9]) of p divided by q, denoted by $\mathrm{sPrem}(p, q, x)$, is the polynomial r satisfying $\mathrm{lc}(q)^e p = aq + r$, where $\deg(r, x) < \deg(q, x)$ and e is the smallest non-negative even integer greater than or equal to $\deg(p, x) - \deg(q, x) + 1$. In Definition 8.4, we reviewed the concepts of "generalized discriminant matrix (sequence)" of two univariate polynomials with real coefficients. We extend the definition to cover the case of two univariate polynomials p and q with coefficients in $\mathbb{Q}[\mathbf{u}]$ by replacing $r := \mathrm{rem}(p'q, p, x)$ with $r := \mathrm{sPrem}(p'q, p, x)$.

Lemma 8.5. *Let* p *and* q *be two polynomials of* x *with coefficients in* $\mathbb{Q}[\mathbf{u}]$. *Let* $p = a_0 x^s + \cdots + a_s$, *where* $a_0 \neq 0$. *Suppose* α_1 *and* α_2 *are two points of* $\mathbb{R}^d$ *such that both* $a_0(\alpha_1) \neq 0$ *and* $a_0(\alpha_2) \neq 0$ *hold. If* $\mathrm{sign}(\mathrm{gds}_j(p, q, x)(\alpha_1)) = \mathrm{sign}(\mathrm{gds}_j(p, q, x)(\alpha_2))$ *hold for all* $j = 1, \ldots, s$, *then we have* $\mathrm{TaQ}(p(\alpha_1), q(\alpha_1)) = \mathrm{TaQ}(p(\alpha_2), q(\alpha_2))$ *holds.*

Proof. Let $r := \mathrm{sPrem}(p'q, p, x)$. Then there exists a non-negative even integer e and a polynomial b such that $a_0^e p'q = bp + r$ holds. Therefore for any $\alpha \in \mathbb{R}^d$ such that $a_0(\alpha) \neq 0$, we have

$$p(\alpha)'q(\alpha) = \frac{b}{a_0^e}(\alpha)p(\alpha) + \frac{r}{a_0^e}(\alpha).$$

For $i = 1, 2$, denote $r_i := \mathrm{rem}(p(\alpha_i), q(\alpha_i))$. By the uniqueness of Euclidean reminder, we deduce that $r_i = \frac{r}{a_0^e}(\alpha_i)$ for $i = 1, 2$. By the specialization properties of computing the determinant of a polynomial matrix and the fact that e is an even number, we deduce that $\mathrm{sign}(\mathrm{gds}_j(p, q, x)(\alpha_i)) = \mathrm{sign}(\mathrm{gds}_j(p(\alpha_i), q(\alpha_i), x))$ holds for $i = 1, 2$ and $j = 1, \ldots, s$. Then the conclusion follows from (ii) of Remark 8.1. □

Lemma 8.6. *Let* p *and* $q_1, q_2, \ldots, q_k$ *be polynomials of* x *with coefficients in* $\mathbb{Q}[\mathbf{u}]$ *and* $\deg(p, x) = s$. *Assume that* p *is squarefree and that* p *has no common factors with each* *of* $q_1, q_2, \ldots, q_k$*. Let* D *be the polynomial set consisting of the non-zero polynomials in* $\bigcup_{e_1,\ldots,e_k\in\{0,1\}}\{\mathrm{gds}(p, \prod_{j=1}^{k} q_j^{e_j}, x)\}$*. Suppose* α_1, α_2 *are two values of* $\mathbf{u}$ *such that* $\mathrm{sign}(f(\alpha_1)) = \mathrm{sign}(f(\alpha_2)) \neq 0$ *for each* $f \in D$*. Then the numbers* $\#\{x|p(\alpha_1) = 0, q_1(\alpha_1) > 0, \ldots, q_k(\alpha_1) > 0\}$ *and* $\#\{x|p(\alpha_2) = 0, q_1(\alpha_2) > 0, \ldots, q_k(\alpha_2) > 0\}$ *are equal.*

Proof. Let q be any polynomial in $\{1, q_1, q_2, \ldots, q_k\}$. Then there exists a non-negative even integer e and polynomial b such that $\mathrm{lc}(p)^e p'q = bp + r$, where $\deg(r, x) < \deg(p, x)$. Since p is squarefree and p has no common factors with q, we deduce that $\gcd(p, r) = \gcd(p'q, p) = 1$ in $\mathbb{Q}(\mathbf{u})[x]$, which implies that $\mathrm{gds}_s(p, q) \neq 0$ and therefore belongs to D.

From $\mathrm{sign}(f(\alpha_1)) = \mathrm{sign}(f(\alpha_2)) \neq 0$ holds for each $f \in D$, we deduce

1. According to Definition 8.4, $\mathrm{lc}(p)$ is a factor of each polynomial in D. Therefore, $\mathrm{lc}(p)(\alpha_i) \neq 0$ holds.

2. For each $q \in \{q_1, \ldots, q_k\}$, we have $\mathrm{gds}_s(p, q, x)(\alpha_i) \neq 0$ holds, which implies that $\gcd(p(\alpha_i), \mathrm{sPrem}(p'q, p, x)(\alpha_i)) = 1$ by (1) and the specialization properties of computing the determinant of a polynomial matrix. So $\gcd(p(\alpha_i), p'(\alpha_i)q(\alpha_i)) = 1$, which implies that $\gcd(p(\alpha_i), q(\alpha_i)) = 1$.

3. For all $e_1, \ldots, e_k \in \{0, 1\}$, $\mathrm{TaQ}(p(\alpha_1), \prod_{j=1}^{k} q_j^{e_j}(\alpha_1)) = \mathrm{TaQ}(p(\alpha_2), \prod_{j=1}^{k} q_j^{e_j}(\alpha_2))$ by Lemma 8.5.

For $i = 1, 2$, let $\mathbf{N}_{\alpha_i}$, $\mathbf{T}_{\alpha_i}$ be the $\mathbf{N}_k$ and $\mathbf{T}_k$ constructed as in Lemma 8.4 for the polynomials $p(\alpha_i), q_1(\alpha_i), \cdots, q_k(\alpha_i)$. Then we have $\mathbf{T}_{\alpha_1} = \mathbf{T}_{\alpha_2}$ by the above item (3). Therefore, we have $\mathbf{N}_{\alpha_1} = \mathbf{N}_{\alpha_2}$ by Lemma 8.4. Then the conclusion follows, since the two numbers are the first element of $\mathbf{N}_{\alpha_1}$ and $\mathbf{N}_{\alpha_2}$ respectively. □

We return to the pre-regular semi-algebraic system $[B_{\neq}, T, P_{>}]$ introduced at the beginning of this section. Recall that m and ℓ are the numbers of polynomials in T and P respectively. Let $\mathfrak{P}_{m+1} := P$ and $\mathfrak{P}_i := \{\mathrm{sPrem}(p, t_i, y_i) \mid p \in \mathfrak{P}_{i+1}\}$ for $i = m, \ldots, 2$. Note $\mathfrak{P}_i$ $(i = m+1, \ldots, 2)$ has at most ℓ elements and suppose that $\mathfrak{P}_2 = \{b_1, \ldots, b_k\}$ $(k \leq \ell)$. Let

$$\mathfrak{P}_1 := \bigcup_{(\alpha_1,\alpha_2,\ldots,\alpha_k)\in\{0,1\}^k} \{\mathrm{gds}(t_1, \prod_{i=1}^{k} b_i^{\alpha_i}, y_1)\} \setminus \{0\}.$$

Proposition 8.4. *Assume that T is in generic position and let $D := B \cup \mathfrak{P}_1$. Then the set D is a fingerprint polynomial set of the pre-regular semi-algebraic system $[B_{\neq}, T, P_{>}]$.*

Proof. First since the main degree of t_i, $2 \leq i \leq m$, is 1, by the relation between pseudo remainder and resultant, we conclude that $\mathfrak{P}_2$ only have variables $\mathbf{u}$ and y_1.

Let dp be the product of polynomials in D. By the definition of D, we know that the border polynomial of $[T, P]$ divides dp. By Proposition 8.1, for any $\alpha \in \mathbb{R}^d$ such that $dp(\alpha) \neq 0$, the regular system $[T, P]$ specializes well at α. On the other hand, by the definition of signed pseudo remainder, there exists even integers δ_i, $1 \leq i \leq \ell$, and polynomials q_{ij}, $1 \leq i \leq \ell, 2 \leq j \leq m$, such that $h_{T_{\geq y_2}}^{\delta_i} p_i = \sum_{j=2}^{m} q_{ij} t_j + b_i$ $(*)$.

Hence, for any $\beta = (\beta_1, \ldots, \beta_m)$ such that $T(\alpha, \beta) = 0$ and $P(\alpha, \beta) > 0$, we have $t_1(\alpha, \beta_1) = 0$ and $b_1(\alpha, \beta_1) > 0$. Similarly, for all β_1 such that $t_1(\alpha, \beta_1) = 0$ and $b_1(\alpha, \beta_1) > 0$, there exists a unique $\beta = (\beta_1, \ldots, \beta_m)$ with $T(\alpha, \beta) = 0$ and $P(\alpha, \beta) > 0$.

Therefore, for any $\alpha \in \mathbb{R}^d$ such that $dp(\alpha) \neq 0$, there is a 1-to-1 correspondence between the real solutions of $t_1(\alpha) = 0, \mathfrak{P}_2(\alpha) > 0$ and those of $[T(\alpha), P(\alpha)_{>}]$. On the other hand, since for any β such that $T(\alpha, \beta) = 0$, we have $p(\alpha, \beta) \neq 0$ for any $p \in P$, by relation $(*)$ we deduce that $t_1(\alpha)$ has no common factors with any $p(\alpha)$, where $p \in P$. The polynomial $t_1(\alpha)$ is clearly squarefree since $[T, P]$ specializes well at α. Thus it follows from Lemma 8.6 that the number of real solutions of $t_1 = 0, \mathfrak{P}_2 > 0$ is determined by signs of polynomials in D. Therefore, the number of real solutions of $[B_{\neq}, T, P_{>}]$ is also determined by signs of polynomials in D. Finally, D is an FPS of $[B_{\neq}, T, P_{>}]$. $\square$

Theorem 8.6. *Let δ and $\hbar$ be respectively the maximum total degree and the maximum coefficient size among all polynomials in P or T. Recall that ℓ and m denote the number of polynomials in P and T respectively. Then the following three properties hold:*

1. *$\mathfrak{P}_1$ has at most $\delta 2^{\ell}$ polynomials,*

2. *the total degree and, the coefficient bit-size of any polynomials in $\mathfrak{P}_1$ are upper bounded by $2\ell(\delta + 1)^{m+3}$ and $\ell^3 \delta^{\mathcal{O}(m^2)} n\hbar$ respectively;*

3. *each polynomial in $\mathfrak{P}_1$ can be computed within $2^{\mathcal{O}(n)} \ell^{\mathcal{O}(n)} \delta^{\mathcal{O}(n)\mathcal{O}(m^2)} \hbar^2$ bit-operations.*

Proof. Denote the total degree and coefficient bit-size of any polynomials in $\mathfrak{P}_{i+1}$ by Δ_i and $\bar{H}_i$ respectively, for $i = 2, \ldots, m$. Combining the estimates for pseudo-reminder and polynomial product recalled in Section 8.3, we have

$$\Delta_i \leq \delta(\Delta_{i+1} + 1) \text{ and } \bar{H}_i \leq (\Delta_{i+1} + 1)\left(\bar{H}_{i+1} + n\log(\Delta_{i+1})\right),$$

where $\Delta_{m+1} = \delta$, $\bar{H}_{m+1} = \hbar$. Therefore, for $i = 0, \ldots, m-1$, we have

$$\Delta_{m-i+1} \leq (\delta+1)^{i+1} \text{ and } H_{m-i+1} < (\delta+1)^{\frac{i^2-i}{2}}(\hbar + i2n\log(\delta+1)).$$

Thus, the total degree and coefficient size of polynomials in $\mathfrak{P}_2$ are upper bounded by $(\delta+1)^m$ and $(\delta+1)^{\frac{m^2}{2}}(\hbar + nm\log(\delta+1))$. Applying the estimates of polynomial product, the total degrees and coefficient sizes of a product of k ($k \leq \ell$) polynomials from $\mathfrak{P}_2$ are bounded over respectively by $\ell(\delta+1)^m$ and $\ell^2(\delta+1)^{\frac{m^2}{2}}(\hbar + mn\log(\delta+1))$. Since $\mathfrak{P}_2$ has k ($k \leq \ell$) polynomials and $\deg(t_1) < \delta$, the set $\mathfrak{P}_1$ has at most $\delta 2^\ell$ polynomials.

Applying the estimates for the determinant of a matrix of multivariate polynomials in Section 8.3, each polynomial in $\mathfrak{P}_1$ has total degree and coefficient size upper bound $2\ell(\delta+1)^{m+3}$ and $\ell^3\delta^{\mathcal{O}(m^2)}n\hbar$ respectively, and can be computed in $2^{\mathcal{O}(n)}\ell^{\mathcal{O}(n)}\delta^{\mathcal{O}(n)\mathcal{O}(m^2)}\hbar^2$ bit operations starting from $\mathfrak{P}_2$.

Note that a pseudo-remainder can be computed as a determinant of a matrix of multivariate polynomials. So the computation of of each polynomial in $\mathfrak{P}_i$ ($i = m, \ldots, 2$) is dominated by the above estimates on computing a polynomial of $\mathfrak{P}_1$ from $\mathfrak{P}_2$. Therefore, each polynomial in $\mathfrak{P}_1$ is computed within $2^{\mathcal{O}(n)}\ell^{\mathcal{O}(n)}\delta^{\mathcal{O}(n)\mathcal{O}(m^2)}\hbar^2$ bit operations. □

8.6 Algorithms

In this section, we present algorithms for LazyRealTriangularize and RealTriangularize that we have implemented. As a byproduct of RealTriangularize, we obtain an algorithm called SamplePoints which computes at least one sample point per connected component of a semi-algebraic set.

Basic subroutines. The algorithms stated in this section rely on a few subroutines that we specify hereafter. For a zero-dimensional squarefree regular system $[T, P]$, the function call RealRootIsolate(T, P) [135] returns all the isolated real zeros of $[T, P_>]$. For $A \subset \mathbb{Q}[u_1, \ldots, u_d]$ and a point s of $\mathbb{Q}^d$ such that $p(s) \neq 0$ for all $p \in A$, the

Algorithm 27: GeneratePreRegularSas($\mathfrak{S}$)

Input: a semi-algebraic system $\mathfrak{S} = [F, N_{\geq}, P_{>}, H_{\neq}]$

Output: a set of pre-regular semi-algebraic systems $[B_{i\neq}, T_i, P_{i>}]$, $i = 1 \ldots e$, such that
$Z_{\mathbb{R}}(\mathfrak{S}) = \cup_{i=1}^{e} Z_{\mathbb{R}}(B_{i\neq}, T_i, P_{i>}) \cup_{i=1}^{e} Z_{\mathbb{R}}(\mathrm{sat}(T_i) \cup \{\Pi_{b \in B_i} b\}, N_{\geq}, P_{>}, H_{\neq})$.

1 $\mathfrak{T} :=$ Triangularize(F, mode = Kalkbrener); $\mathfrak{T}' := \emptyset$
2 **for** $p \in P \cup H$ **do**
3   **for** $T \in \mathfrak{T}$ **do**
4   **for** $C \in$ Regularize(p, T) **do**
5   **if** $p \notin \mathrm{sat}(C)$ **then** $\mathfrak{T}' := \mathfrak{T}' \cup \{C\}$
6     $\mathfrak{T} := \mathfrak{T}'$; $\mathfrak{T}' := \emptyset$
7 $\mathfrak{T} := \{[T, \emptyset] \mid T \in \mathfrak{T}\}$; $\mathfrak{T}' := \emptyset$
8 **for** $p \in N$ **do**
9   **for** $[T, N'] \in \mathfrak{T}$ **do**
10   **for** $C \in$ Regularize(p, T) **do**
11   **if** $p \in \mathrm{sat}(C)$ **then**
12                 $\mathfrak{T}' := \mathfrak{T}' \cup \{[C, N']\}$
13   **else**
14                 $\mathfrak{T}' := \mathfrak{T}' \cup \{[C, N' \cup \{p\}]\}$
15     $\mathfrak{T} := \mathfrak{T}'$; $\mathfrak{T}' := \emptyset$
16 $\mathfrak{T} := \{[T, N', P, H] \mid [T, N'] \in \mathfrak{T}\}$
17 **for** $[T, N', P, H] \in \mathfrak{T}$ **do**
18     $B :=$ BorderPolynomialSet($T, N' \cup P \cup H$)
19   output $[B, T, N' \cup P]$

function call GenerateFormula(A, s) computes a formula $\wedge_{p \in A} \; (p \, \sigma_{p,s} > 0)$, where $\sigma_{p,s}$ is defined as $+1$ if $p(s) > 0$ and -1 otherwise. For a set of formulas G, the function call Disjunction(G) computes a logic formula Φ equivalent to the disjunction of the formulas in G.

Proof of Algorithm 27. Its termination is obvious. We prove its correctness. By the specification of Triangularize and Regularize, at line 16, we have

$$Z(F, P_{\neq} \cup H_{\neq}) \;=\; \cup_{[T,N',P,H] \in \mathfrak{T}} \; Z(\mathrm{sat}(T), P_{\neq} \cup H_{\neq}).$$

Write $\cup_{[T,N',P,H] \in \mathfrak{T}}$ as $\cup_T$. Then we deduce that

$$Z_{\mathbb{R}}(F, N_{\geq}, P_{>}, H_{\neq}) \;=\; \cup_T \; Z_{\mathbb{R}}(\mathrm{sat}(T), N_{\geq}, P_{>}, H_{\neq}).$$

Algorithm 28: GenerateRegularSas(B, T, P)

Input: $\mathfrak{S} = [B_{\neq}, T, P_{>}]$, a pre-regular semi-algebraic system of $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$, where $\mathbf{u} = u_1, \ldots, u_d$ and $\mathbf{y} = y_1, \ldots, y_{n-d}$.

Output: A pair $(D, \mathcal{R})$ satisfying:

(1) $D \subset \mathbb{Q}[\mathbf{u}]$ such that $\text{factor}(B) \subseteq D$;

(2) $\mathcal{R}$ is a finite set of regular semi-algebraic systems, such that we have: $\cup_{R \in \mathcal{R}} Z_{\mathbb{R}}(R) = Z_{\mathbb{R}}(D_{\neq}, T, P_{>})$.

$D := \text{factor}(B \setminus \mathbb{Q})$
if $d = 0$ **then**
    if RealRootIsolate$(T, P) = [\ ]$ **then** return $(D, \emptyset)$; **else** return $(D, \{[true, T, P]\})$
while $true$ **do**
    $S :=$ SampleOutHypersurface(D, d); $G_0 := \emptyset$; $G_1 := \emptyset$
    for $s \in S$ **do**
        if RealRootIsolate$(T(s), P(s)) = [\ ]$ **then**
            $G_0 := G_0 \cup \{$GenerateFormula$(D, s)\}$
        else
            $G_1 := G_1 \cup \{$GenerateFormula$(D, s)\}$
    if $G_0 \cap G_1 = \emptyset$ **then**
        $\mathcal{Q} :=$ Disjunction(G_1)
        if $\mathcal{Q} = false$ **then** return $(D, \emptyset)$; **else** return $(D, \{[\mathcal{Q}, T, P]\})$
    else
        select a subset $D' \subseteq \text{oaf}(B) \setminus D$ by some heuristic method
        $D := D \cup D'$

Between lines 17 and 19, for each $[T, N', P, H]$, we generate a pre-regular semi-algebraic system $[\mathcal{B}, T, N'_{>} \cup P_{>}]$. By Corollary 8.1, we have

$$\begin{aligned} Z_{\mathbb{R}}(\text{sat}(T), N_{\geq}, P_{>}, H_{\neq}) &= Z_{\mathbb{R}}(\text{sat}(T), N'_{\geq}, P_{>}, H_{\neq}) \\ &= Z_{\mathbb{R}}(B_{\neq}, T, N'_{>} \cup P_{>}) \cup Z_{\mathbb{R}}\ (\text{sat}(T) \cup \{\Pi_{b \in B} b\}, N_{\geq}, P_{>}, H_{\neq}), \end{aligned}$$

which implies that

$$Z_{\mathbb{R}}(\mathfrak{S}) = \cup_T \ (Z_{\mathbb{R}}(B_{\neq}, T, N'_{>} \cup P_{>}) \cup Z_{\mathbb{R}}(\text{sat}(T) \cup \{\Pi_{b \in B} b\}, N_{\geq}, P_{>}, H_{\neq}))$$

holds. Therefore, Algorithm 27 satisfies its specification.

Proof of Algorithms 28 and 29. By the definition of oproj, Algorithm 29 terminates and satisfies its specification. By Theorem 8.5, $\text{oaf}(B)$ is an FPS. Thus, by the definition of an FPS, Algorithm 28 terminates and satisfies its specification.

Algorithm 29: $\mathsf{SampleOutHypersurface}(A, k)$

Input: $A \subset \mathbb{Q}[x_1, \ldots, x_k]$ is a finite set of non-zero polynomials
Output: A finite subset of $\mathbb{Q}^k$ contained in $(\Pi_{p \in A}\, p) \neq 0$ and having a non-empty intersection with each connected component of $(\Pi_{p \in A}\, p) \neq 0$.

if $k = 1$ **then**
    return one rational point from each connected component of $\Pi_{p \in A}\, p \neq 0$
else
    $A_k := \{p \in A \mid \mathrm{mvar}(p) = x_k\}$; $A' := \mathrm{oproj}(A, x_k)$
    for $s \in \mathsf{SampleOutHypersurface}(A', k-1)$ **do**
        Collect in a set S one rational point from each connected component of $\Pi_{p \in A_k} p(s, x_k) \neq 0$;
        for $\alpha \in S$ **do** output (s, α)

Algorithm 30: $\mathsf{LazyRealTriangularize}(\mathfrak{S})$

Input: a semi-algebraic system $\mathfrak{S} = [F, N_{\geq}, P_{>}, H_{\neq}]$
Output: a lazy triangular decomposition of $\mathfrak{S}$

$\mathfrak{T} := \mathsf{GeneratePreRegularSas}(F, N, P, H)$
for $[B, T, P'] \in \mathfrak{T}$ **do**
    $(D, \mathcal{R}) = \mathsf{GenerateRegularSas}(B, T, P')$
    if $\mathcal{R} \neq \emptyset$ **then** output $\mathcal{R}$

Algorithm 31: $\mathsf{RealTriangularize}(\mathfrak{S})$

Input: a semi-algebraic system $\mathfrak{S} = [F, N_{\geq}, P_{>}, H_{\neq}]$
Output: a triangular decomposition of $\mathfrak{S}$

$\mathfrak{T} := \mathsf{GeneratePreRegularSas}(F, N, P, H)$
for $[B, T, P'] \in \mathfrak{T}$ **do**
    $(D, \mathcal{R}) = \mathsf{GenerateRegularSas}(B, T, P')$
    if $\mathcal{R} \neq \emptyset$ **then** output $\mathcal{R}$
    for $p \in D$ **do**
        output $\mathsf{RealTriangularize}(F \cup \{p\}, N, P, H)$

Proof of Algorithm 30. Its termination is obvious; we prove it is correct. Let R_i, $i = 1 \cdots t$ be the output. By the specification of each sub-algorithm, each R_i is a regular semi-algebraic system and we have $\cup_{i=1}^t Z_{\mathbb{R}}(R_i) \subseteq Z_{\mathbb{R}}(\mathfrak{S})$. Next we show that there exists an ideal $\mathcal{I} \subseteq \mathbb{Q}[\mathbf{x}]$, whose dimension is less than $\dim(Z(F, P_{\neq} \cup H_{\neq}))$ and such that $Z_{\mathbb{R}}(\mathfrak{S}) \setminus \cup_{i=1}^t Z_{\mathbb{R}}(R_i) \subseteq Z_{\mathbb{R}}(\mathcal{I})$ holds. At line 1, the specification of Algorithm 27 imply:

$$Z_{\mathbb{R}}(\mathfrak{S}) = \cup_T Z_{\mathbb{R}}(B_{\neq}, T, P'_{>}) \ \cup \ \cup_T Z_{\mathbb{R}}(\mathrm{sat}(T) \cup \{\Pi_{b \in B}\ b\}, N_{\geq}, P_{>}, H_{\neq}).$$

At line 3, by the specification of Algorithm 28, for each B, we compute a set D such that factor$(B) \subseteq D$ and

$$\cup_T Z_{\mathbb{R}}(D_{\neq}, T, P'_{>}) = \cup_{i=1}^t Z_{\mathbb{R}}(R_i) \tag{8.2}$$

both hold. Following the strategy used in Algorithm 27, based on Corollary 8.1, we have

$$Z_{\mathbb{R}}(\mathfrak{S}) = \cup_T Z_{\mathbb{R}}(D_{\neq}, T, P'_{>}) \ \cup \ \cup_T Z_{\mathbb{R}}(\mathrm{sat}(T) \cup \{\Pi_{p \in D}\ p\}, N_{\geq}, P_{>}, H_{\neq}). \tag{8.3}$$

Combining the relations (8.2) and (8.3) together, we obtain

$$Z_{\mathbb{R}}(\mathfrak{S}) = \cup_T Z_{\mathbb{R}}(R_i) \ \cup \ \cup_T Z_{\mathbb{R}}(\mathrm{sat}(T) \cup \{\Pi_{p \in D}\, p\}, N_{\geq}, P_{>}, H_{\neq}).$$

Therefore, the following relations hold

$$\begin{array}{rcl} Z_{\mathbb{R}}(\mathfrak{S}) \setminus \cup_{i=1}^t Z_{\mathbb{R}}(R_i) & \subseteq & \cup_T Z_{\mathbb{R}}(\mathrm{sat}(T) \cup \{\Pi_{p \in D}\, p\}, N_{\geq}, P_{>}, H_{\neq}) \\ & \subseteq & Z_{\mathbb{R}}\left(\cap_T \left(\mathrm{sat}(T) \cup \{\Pi_{p \in D}\, p\}\right)\right). \end{array}$$

Define $\mathcal{I} = \cap_T \left(\mathrm{sat}(T) \cup \{\Pi_{p \in D}\, p\}\right)$. Since each $p \in D$ is regular modulo $\mathrm{sat}(T)$, we have $\dim(\mathcal{I}) < \dim\left(\cap_T \mathrm{sat}(T)\right) \leq \dim(Z(F, P_{\neq} \cup H_{\neq}))$. So all R_i form a lazy triangular decomposition of $\mathfrak{S}$. □

Proof of Algorithm 31. For its termination, it is sufficient to prove that there are only finitely many recursive calls to RealTriangularize. Indeed, if $[F, N, P, H]$ is the input of a call to RealTriangularize then each of the immediate recursive calls takes $[F \cup \{p\}, N, P, H]$ as input, where p belongs to the set D of some pre-regular semi-algebraic system $[D_{\neq}, T, P_{>}]$. Since p is regular (and non-zero) modulo $\mathrm{sat}(T)$ we have: $\langle F \rangle \subsetneq \langle F \cup \{p\} \rangle$. Therefore, the algorithm terminates by the ascending chain condition on ideals of $\mathbb{Q}[\mathbf{x}]$. The correctness of Algorithm 31 follows from that of its sub-algorithms. □

Implementation remark for LazyRealTriangularize. Our software implementation (within the `RegularChains` library in Maple) of Algorithm 30 returns the necessary information for completing a full triangular decomposition of the input semi-algebraic system $\mathfrak{S}$. This is achieved simply by returning $[F \cup \{p\}, N, P, H]$ for each $p \in D$, for each D.

For an input semi-algebraic system $\mathfrak{S}$ Algorithm 32 computes a sample point set of $\mathfrak{S}$, see Definition 8.5, thus producing at least one point per connected component of $Z_{\mathbb{R}}(\mathfrak{S})$.

Definition 8.5. *Let S be a semi-algebraic set of $\mathbb{R}^n$. A finite subset A of $\mathbb{R}^n$ is called a* sample point set *of S if the following conditions hold:*

(i) *every point of A belongs to some connected component of S,*

(2) *every connected component of S has a nonempty intersection with A.*

Algorithm 32: SamplePoints($\mathfrak{S}$)

Input: a semi-algebraic system $\mathfrak{S} = [F, N_{\geq}, P_{>}, H_{\neq}]$
Output: A sample point set of $\mathfrak{S}$.

$\mathfrak{T} :=$ GeneratePreRegularSas(F, N, P, H)
for $[B, T, P'] \in \mathfrak{T}$ **do**
  for $s \in$ SampleOutHypersurface(B) **do**
    for $\alpha \in$ RealRootIsolate$(T(s), P'(s))$ **do**
      output (s, α)
  for $p \in B$ **do**
    output SamplePoints$(F \cup \{p\}, N, P, H)$

Lemma 8.7. *Let S, S_1 and S_2 be nonempty semi-algebraic sets of $\mathbb{R}^n$. Assume that $S = S_1 \cup S_2$. Let A_1 (resp. A_2) be a sample point set of S_1 (resp. S_2). Then $A_1 \cup A_2$ is a sample point set of S.*

Proof. First, any point of $A_1 \cup A_2$ obviously belongs to S and therefore belongs to some connected component of S. Secondly, we want to prove that each connected component of S contains at least one point of $A_1 \cup A_2$. We prove this by contradiction. Suppose C is a connected component of S that does not contain any point of $A_1 \cup A_2$ ($*$). Let $p \in C$. Then p belongs to some connected component D of S_1 or S_2. Let q be a point of $A_1 \cup A_2$ such that q belongs to D. Then there exists a path $L(p, q)$

connecting p and q, which is contained in D and hence contained in S. So p and q belongs to the same connected component of S, which implies that $q \in C$ holds. This is a contradiction to $(*)$. □

Proof of Algorithm 32. The proof of its termination is exactly the same as that of RealTriangularize. It correctness follows from Lemma 8.7 and Theorem 8.1.

8.7 Experimentation

We have implemented our algorithms on top of the `RegularChains` library in MAPLE. Hereafter, we report on experimental results using well known benchmark examples from the literature. The test examples are available at `www.orcca.on.ca/~cchen/issac10.txt`.

Table 8.1. Table 8.1 summarizes the notations used in Tables 8.2, 8.3 and 9.1. These tables demonstrate benchmarks running in MAPLE 15, using an Intel Core 2 Quad CPU (2.40GHz) with 3.0GB memory. The timings are in seconds and the time-out is 1 hour.

symbol	meaning
#e	number of equations in the input system
#v	number of variables in the input equations
d	maximum total degree of an input equation
G	Groebner:-Basis (plex order) in MAPLE
T	Triangularize in `RegularChains` library of MAPLE
ST	Squarefree Triangularize in `RegularChains` library of MAPLE
LR	LazyRealTriangularize implemented in MAPLE
R_{re}	The recursive implementation of RealTriangularize in MAPLE
S	SamplePoints implemented in MAPLE
Q	QEPCAD B 1.61
$> 1h$	computation does not complete within 1 hour
FAIL	QEPCAD B failed due to prime list exhausted

Table 8.1: Notations

Table 8.2. The systems in this group involve equations only. We list the running times for computing a triangular decomposition of the input algebraic variety as well as a lazy and a full triangular decomposition of the corresponding real variety. We also provide the running times for computing lexicographical Gröbner bases with the MAPLE function `Groebner:-Basis`. The data illustrate the performance of LazyRealTriangularize, RealTriangularize and SamplePoints.

system	#v/#e/d	G	T	ST	LR	R_{re}	S
Hairer-2-BGK	13/11/4	24.64	2.05	2.08	2.96	4.20	5.55
Collins-jsc02	5/4/3	$> 1h$	0.52	0.52	1.81	560.92	10.82
Leykin-1	8/6/4	101.44	4.00	4.02	4.39	5.46	5.72
8-3-config-Li	12/7/2	110.24	5.96	6.01	7.38	417.90	446.29
Lichtblau	3/2/11	126.35	0.31	0.32	3.55	$> 1h$	$> 1h$
Cinquin-3-3	4/3/4	64.84	0.70	0.76	2.34	$> 1h$	57.23
Cinquin-3-4	4/3/5	$> 1h$	3.47	3.43	15.19	$> 1h$	$> 1h$
DonatiTraverso-rev	4/3/8	159.95	1.89	2.23	3.34	3.02	2.98
Cheaters-homotopy-1	7/3/7	2498.78	0.65	451.33	$> 1h$	$> 1h$	$> 1h$
hereman-8.8	8/6/6	$> 1h$	12.92	22.24	$> 1h$	$> 1h$	110.34
L	12/4/3	$> 1h$	0.79	0.80	1.12	14.94	18.16
dgp6	17/19/2	27.38	48.62	49.62	51.75	62.99	70.74
dgp29	5/4/15	85.70	0.20	0.20	0.37	0.38	0.33

Table 8.2: Timings for varieties

Table 8.3. The systems in this table are from quantifier elimination problems. Most of them involve both equations and inequalities. We provide the timings for computing (1) a lazy triangular decomposition, (2) a full triangular decomposition and (3) sample points of the corresponding semi-algebraic systems as well as the timings for solving the quantifier elimination problem via Qepcad b [19] (in non-interactive mode). Our tools complete the computations for most of the systems. However, one should note that the output of our tools is not a solution to the posed quantifier elimination probem. We note also that our tools are more effective for systems counting more equations than inequalities.

We conclude this section by reporting on an experimental comparison of `SamplePoints` versus related software tools. Among the software that we can access, we could find only one software function with the same specifications as `SamplePoints`, that is, a function computing a sample point set, see Definition 8.5, for an arbitrary semi-algebraic system. This function is the `SemialgebraicComponentInstances` command in Mathematica. We have tested the function `SemialgebraicComponentInstances` in Mathematica 8 for the systems (26 in total) listed in Table 8.2 and Table 8.3. We have found that this command succeeded for 9 of them, within the same resource limit and the same machine as described above, while `SamplePoints` could solve 19 of those systems. Among the 9 systems that `SemialgebraicComponentInstances` could solve, `SamplePoints` failed on 3 of them.

system	#v/#e/d	T	ST	LR	R_{re}	S	Q
BM05-1	4/2/3	0.28	0.28	0.65	1.15	1.19	8.16
BM05-2	4/2/4	0.29	0.29	3.50	$> 1h$	$> 1h$	FAIL
Solotareff-4b	5/4/3	0.91	0.93	1.98	881.15	14.42	$> 1h$
Solotareff-4a	5/4/3	0.71	0.74	1.63	4.00	3.12	FAIL
putnam	6/4/2	0.27	0.30	0.76	1.65	1.70	$> 1h$
MPV89	6/3/4	0.23	0.29	0.89	2.75	2.42	$> 1h$
IBVP	8/5/2	0.58	0.62	1.26	14.23	13.89	$> 1h$
Lafferriere37	3/3/4	0.33	0.38	0.69	0.72	0.62	2.3
Xia	6/3/4	0.46	0.46	2.20	209.65	168.49	$> 1h$
SEIT	11/4/3	0.70	0.71	32.67	$> 1h$	1355.81	$> 1h$
p3p-isosceles	7/3/3	0.35	0.35	$> 1h$	$> 1h$	$> 1h$	$> 1h$
p3p	8/3/3	0.37	0.40	$> 1h$	$> 1h$	$> 1h$	FAIL
Ellipse	6/1/3	0.18	0.19	0.96	$> 1h$	$> 1h$	$> 1h$

Table 8.3: Timings for semi-algebraic systems

8.8 Applications in program verification

We consider and example arising in the study of program verifications. We apply the `RegularChains` library implementation of the algorithms of Section 8.6.

Recent advances in program verification indicate that various problems, for instance, termination analysis of linear programs [121], reachability computation of linear hybrid systems [61], and invariant generation [95, 110] can be reduced to solving semi-algebraic systems. Tools for real algebraic computation such as REDLOG [52], QEPCAD [45, 77, 19], and DISCOVERER [140] have therefore been applied to program verification [95, 61].

We consider here Example 3.5 from [61]. This problem reduces to determine the set

$$\{(y_1, y_2) \in \mathbb{R}^2 \ \mid \ (\exists a \in \mathbb{R})(\exists z \in \mathbb{R}) \ (0 \leq a) \wedge (z \geq 1) \wedge (h_1 = 0) \wedge (h_2 = 0)\}$$

where $h_1 = 3\,y_1 - 2\,a(-z^4 + z)$ and $h_2 = 2\,y_2 z^2 - a(z^4 - 1)$. In order words, one wishes to compute the projection of the semi-algebraic set defined by $(0 \leq a) \wedge (z \geq 1) \wedge (h_1 = 0) \wedge (h_2 = 0)$ onto the (y_1, y_2)-plane. This question can be answered by running the RealTriangularize command on the semi-algebraic set for the variable ordering $a > z > y_1 > y_2$. We obtain the five following regular semi-algebraic systems

R_1 to R_5 (unspecified R_i^P and R_i^Q are empty):

$$R_1^T = \begin{cases} (z^4-1)\,a - 2\,z^2 y_2 \\ 4\,y_2\,z^5 + 4\,y_2\,z^4 + (3\,y_1 + 4\,y_2)\,z^3 + 3\,y_1\,z^2 + 3\,y_1\,z + 3\,y_1 \end{cases}$$

$$R_1^Q = \begin{cases} (y_1+y_2<0) \wedge (y_1<0) \wedge (0<y_2) \\ 3y_1^5 - 6y_2y_1^4 - 63y_2^2y_1^3 + 192y_2^3y_1^2 + 112y_2^4y_1 + 16y_2^5 \neq 0 \end{cases} \qquad R_1^P = \begin{cases} z>1 \end{cases}$$

$$R_2^T = \begin{cases} a \\ y_1 \\ y_2 \end{cases} \qquad R_3^T = \begin{cases} z-1 \\ y_1 \\ y_2 \end{cases} \qquad R_4^T = \begin{cases} a \\ z-1 \\ y_1 \\ y_2 \end{cases}$$

$$R_2^P = \begin{cases} z>1 \end{cases} \qquad R_3^P = \begin{cases} 0<a \end{cases}$$

$$R_5^T = \begin{cases} (z^4-1)\,a - 2\,z^2 y_2 \\ t_z \\ 3\,{y_1}^5 - 6\,y_2\,{y_1}^4 - 63\,{y_2}^2{y_1}^3 + 192\,{y_2}^3{y_1}^2 + 112\,{y_2}^4 y_1 + 16\,{y_2}^5 \end{cases}$$

$$R_5^Q = \begin{cases} 0<y_2 \end{cases} \qquad R_5^P = \begin{cases} z>1 \end{cases}$$

where

$$\begin{aligned} t_z = {} & (369252163868\,{y_1}^4 - 2508200686544\,y_2\,{y_1}^3 + 4300300820416\,{y_2}^2{y_1}^2 + 2761812320448\,{y_2}^3 y_1 \\ & + 406754520832\,{y_2}^4)z^4 + (-180672905280\,{y_2}^4 - 1228579249664\,{y_2}^3 y_1 - 1922937082240\,{y_2}^2{y_1}^2 \\ & + 1092105551100\,y_2\,{y_1}^3 - 157082832940\,{y_1}^4)z^3 + (-815128066608\,{y_2}^4 - 5538434025360\,{y_2}^3 y_1 \\ & - 8644620182000\,{y_2}^2{y_1}^2 + 4979116186797\,y_2\,{y_1}^3 - 728379335938\,{y_1}^4)z^2 + (-316725331280\,{y_2}^4 \\ & - 276096356865\,{y_1}^4 + 1914148321163\,y_2\,{y_1}^3 - 3371008535808\,{y_2}^2{y_1}^2 - 2153737071904\,{y_2}^3 y_1)z \\ & - 1030979306368\,{y_2}^4 - 10923966861712\,{y_2}^2{y_1}^2 + 6315633355800\,y_2\,{y_1}^3 - 7003676730320\,{y_2}^3 y_1 \\ & - 923425115541\,{y_1}^4. \end{aligned}$$

The projection on the (y_1, y_2)-plane of $Z_{\mathbb{R}}(R_2) \cup Z_{\mathbb{R}}(R_3) \cup Z_{\mathbb{R}}(R_4)$ is clearly equal to the $(y_1, y_2) = (0, 0)$ point. Properties (iii) of Definition 8.1 implies that the projection on the (y_1, y_2)-plane of $Z_{\mathbb{R}}(R_1)$ is given by $Z_{\mathbb{R}}(R_1^Q)$. For R_5, we observe that the polynomial of R_5^T with main variable y_1, say t_{y_1} is delineable above $0 < y_2$ (By Theorem 8.1). Using a sample point we check that t_{y_1} admits a single real root. It follows that the projection on the (y_1, y_2)-plane of $Z_{\mathbb{R}}(R_5)$ is given by:

$$(0<y_2) \wedge (3\,{y_1}^5 - 6\,y_2\,{y_1}^4 - 63\,{y_2}^2{y_1}^3 + 192\,{y_2}^3{y_1}^2 + 112\,{y_2}^4 y_1 + 16\,{y_2}^5 = 0).$$

To conclude, we have completed the projection of the semi-algebraic set onto the (y_1, y_2)-plane, which can be simplified as $(y_1 < 0 \wedge y_2 > 0 \wedge y_1 + y_2 < 0) \vee (y_1 = 0 \wedge y_2 = 0)$.

8.9 Discussion and concluding remarks

Given a semi-algebraic system $\mathfrak{S}$ the algorithm RealTriangularize (resp. LazyRealTriangularize), as stated in Section 8.6, returns a full (resp. lazy) triangular decomposition of $\mathfrak{S}$. Consider $R = [\mathcal{Q}, T, P_>]$ an output regular semi-algebraic system and assume that T admits $x_1 < \cdots < x_d$ as free variables, for $d > 0$. Let C be a connected component of the semi-algebraic set defined by $\mathcal{Q}$ in $\mathbb{R}^d$. Theorem 8.1 states that, above C, the set $Z_{\mathbb{R}}(R)$ consists of finitely many disjoint graphs of continuous functions where each of these graphs is locally homeomorphic to the hypercube $(0,1)^d$. Therefore R can be regarded as a parameterization of $Z_{\mathbb{R}}(R)$.

This situation is similar to that of triangular decomposition of algebraic sets. Indeed, consider an input polynomial system $F \in \mathbf{k}[\mathbf{x}]$, for a field $\mathbf{k}$, to which the algorithm Triangularize is applied. Consider also an output regular chain T with $x_1 < \cdots < x_d$ as free variables, for $d > 0$. Then T represents a generic zero for each irreducible component of $V(\mathrm{sat}(T))$; moreover each of these irreducible components has dimension d.

The complexity results of Sections 8.3 and 8.5 together with the experimental results of Section 8.7 suggest that the notions and algorithms presented in this work are promising tools for manipulating semi-algebraic sets symbolically. In the sequel of this section, we would like to address the following natural question: would there be an alternative and competitive algorithm implementing the specifications of LazyRealTriangularize while relying on existing tools from the literature?

One direct approach for computing a lazy triangular decomposition of the semi-algebraic system $\mathfrak{S}$ could be the following.

(i) Decompose $\mathfrak{S}$ into pre-regular semi-algebraic systems, using Algorithm 27.

(ii) For each output pre-regular semi-algebraic system $[B_{\neq}, T, P_>]$ compute a CAD of the complement of the hypersurface defined by B in the parameter space, where this CAD produces for each cell a sample point s and a Tarski formula Φ defining that cell.

(iii) For each $[B_{\neq}, T, P_>]$ for each (s, Φ) associated with $[B_{\neq}, T, P_>]$, if the specialized system $[T(s), P_>(s)]$ has real solutions then output $[\Phi, T, P_>]$.

In our approach we modify Step (ii) (and Step (iii)) and avoid the computation of a full CAD by reducing to the following quantifier elimination problem:

$$\exists \mathbf{y} (B(\mathbf{u}) \neq 0, T(\mathbf{u}, \mathbf{y}) = 0, P(\mathbf{u}, \mathbf{y}) > 0).$$

See Section 8.4 for details. When B is a fingerprint polynomial set, we solve this problem by computing (at least) one sample point in each connected component of the complement of the hypersurface defined by B in the parameter space. Then, the properties of an FPS yield the Tarski formulas from the polynomials in the FPS. When B is not a fingerprint polynomial set, we replace B by a superset D of B, which is an FPS.

A first advantage of our approach is that the concept of an FPS is independent of the elimination procedure (CAD or other). Actually, we have described two strategies for FPS construction: one based on open augmented projection (Section 8.4) and one based on generalized discriminant sequences (Section 8.5). A second advantage is that when T is in generic position, an FPS of $[B_{\neq}, T, P_{>}]$ can be computed in singly exponential time w.r.t. the number of variables. It is worth noticing that this case occurs very frequently in practice. Another important practical observation is the fact that, often, a fairly small subset of the theoretical FPS (the set $\mathrm{oaf}(B)$ in Theorem 8.5 and the set D in Proposition 8.4) is already an FPS. We take advantage of this latter observation in our implementation.

Regarding the construction and the use of an FPS, we conclude with two remarks. First, in our implementation and as suggested by Algorithm 28, an FPS is constructed by an incremental process starting from B. A related procedure appears in [16] where a CAD augmented projection is computed incrementally so as to produce a projection-definable CAD. One difference is that, in the FPS construction based on open augmented projection, the considered cells (in the space of the free variables of T) are all open. In the case of the augmented projection construction [16] cells of lower dimension may need to be considered as well. Secondly, we observe that, in principle, Algorithm 29 may be replaced by any procedure computing at least one rational point per connected component of the complement of a hypersurface. Despite of its doubly exponential running time, we have verified experimentally that our implementation of Algorithm 29 is competitive with other tools, such as MAPLE's command `RootFinding:- WitnessPoints`.

Chapter 9

Set-theoretic Operations on Semi-algebraic Sets

This chapter presents algorithms for performing set-theoretic operations on semi-algebraic sets based on the triangular decomposition representation of semi-algebraic sets. We illustrate the effectiveness of our algorithms by applying them to removing redundant components appearing in a triangular decomposition of semi-algebraic systems.

9.1 Introduction

Performing set-theoretic operations on semi-algebraic sets is a fundamental question with many applications. For two semi-algebraic sets S_1 and S_2, it includes computing their union $S_1 \cup S_2$, their intersection $S_1 \cap S_2$, the differences $S_1 \setminus S_2$ and $S_2 \setminus S_1$. For instance, we can apply the verification techniques developed in Chapter 5 for algebraic system solvers to semi-algebraic system solvers such that those implementing the algorithms of Chapter 8.

Another application is the removal superfluous components in the computation of triangular decomposition of semi-algebraic systems. Indeed, it is well known that decomposition algorithms for polynomial systems, whether they are symbolic [31] or numeric [115] tend to generate components which are contained in others within the same decomposition. This phenomenon happens also with the algorithm RealTriangularize for computing triangular decompositions of semi-algebraic systems, presented in Chapter 8. More precisely, the algorithm RealTriangularize can produce redundant components, that is, regular semi-algebraic systems S for which there exists another regular semi-algebraic system S' in the same decomposition and such that

$Z_{\mathbb{R}}(S) \subseteq Z_{\mathbb{R}}(S')$ holds. The relation $Z_{\mathbb{R}}(S) \subseteq Z_{\mathbb{R}}(S')$ holds if and only if the set-theoretic difference $Z_{\mathbb{R}}(S) \setminus Z_{\mathbb{R}}(S')$ is empty. Thus the inclusion test problem is a particular case of computing the set-theoretic difference of two semi-algebraic sets.

In Section 9.2, we provide procedures for set-theoretic operations on semi-algebraic sets represented by triangular decomposition. Those procedures rely on a new algorithm for computing triangular decomposition of semi-algebraic systems in an incremental manner. Presented in Section 9.3 this algorithm is a natural adaptation of the ideas of Chapter 4 for computing triangular decomposition of algebraic systems incrementally. Section 9.5 provides experimental results on the removal of redundant components in triangular decomposition of polynomial systems.

This chapter is based on paper [29], co-authored with James H. Davenport, Marc Moreno Maza, Bican Xia and Rong Xiao.

9.2 Set theoretic operations

In Chapter 8, we proved that every semi-algebraic set can be represented as the union of zero sets of finitely many regular semi-algebraic systems. It is natural to ask how to perform set theoretic operations, such as union, intersection, complement and difference of semi-algebraic sets, based on such a representation.

Note that each (regular) semi-algebraic system can also be seen as a quantifier free formula. So one can implement the set operations naively based on the algorithm RealTriangularize and logic operations. However, an obvious drawback of such an implementation is that it totally neglects the structure of a regular semi-algebraic system.

Indeed, if the structure of the computed object can be exploited, it is possible to obtain more efficient algorithms. One good example of this is the Difference algorithm, which computes the difference of the zero sets of two regular systems, presented in Chapter 5. This algorithm exploits the structure of a regular chain and outperforms the naive implementation by several orders of magnitude.

Apart from the algebraic computations, the idea behind the Difference algorithm in Chapter 5 is to compute the difference $(A_1 \cap A_2) \setminus (B_1 \cap B_2)$ in the following way:

$$(A_1 \cap B_1) \cap (A_2 \setminus B_2) \bigcup (A_1 \setminus B_1) \cap A_2. \tag{9.1}$$

Observe that if $A_1 \cap B_1 = \emptyset$, then the difference is $(A_1 \cap A_2)$. Moreover, computing $\cap_{i=1}^{s} A_i \setminus \cap_{i=1}^{t} B_i$ $(s, t \geq 2)$ can be reduced to the above base case.

In this section, we present algorithms (Algorithm 33 and 34) which take advantage of the algorithm Difference (also an algorithm Intersection derived from it) and the idea presented above for computing the intersection and difference of semi-algebraic sets represented by regular semi-algebraic systems.

We provide proofs of the termination and correctness of Algorithm 36 and 37. The termination and correctness of the other algorithms can be easily derived from them by relation 9.1 and other logic arguments.

Algorithm 33: DifferenceRsas(R, R')

Input: two regular semi-algebraic systems $R = [\mathcal{Q}, T, P_>]$ and $R' = [\mathcal{Q}', T', P'_>]$

Output: a set of regular semi-algebraic systems R_i, $i = 1, \ldots, e$, such that $Z_{\mathbb{R}}(R) \setminus Z_{\mathbb{R}}(R') = \cup_{i=1}^{e} Z_{\mathbb{R}}(R_i)$.

begin
  $\mathcal{Q} := \mathcal{Q} \wedge P_>$;
  $\mathcal{Q}' := \mathcal{Q}' \wedge P'_>$;
  $\mathfrak{T} :=$ Difference(T, T');
  $\mathfrak{T}' :=$ Intersection(T, T');
  if $\mathfrak{T}' = \emptyset$ **then** return R;
  for $[T^*, H^*] \in \mathfrak{T}'$ **do**
    $\mathcal{Q}^* = \mathcal{Q} \setminus \mathcal{Q}' \wedge H^*_{\neq}$;
    output RealTriangularize$(T^*, \mathcal{Q}^*)$
  for $[T^*, H^*] \in \mathfrak{T}$ **do**
    $\mathcal{Q}^* = \mathcal{Q} \wedge H^*_{\neq}$;
    output RealTriangularize$(T^*, \mathcal{Q}^*)$
end

Algorithm 34: IntersectionRsas(R, R')

Input: two regular semi-algebraic systems $R = [\mathcal{Q}, T, P_>]$ and $R' = [\mathcal{Q}', T', P'_>]$

Output: a set of regular semi-algebraic systems R_i, $i = 1, \ldots, e$, such that $Z_{\mathbb{R}}(R) \cap Z_{\mathbb{R}}(R') = \cup_{i=1}^{e} Z_{\mathbb{R}}(R_i)$.

$\mathcal{Q}^* := \mathcal{Q} \wedge P_> \wedge \mathcal{Q}' \wedge P'_>$;
for $[T^*, H^*] \in$ Intersection(T, T') **do**
  output RealTriangularize$(T^*, \mathcal{Q}^* \wedge H^*_{\neq})$

Proposition 9.1. *Algorithm 36 terminates and satisfies its specification.*

Algorithm 35: RealTriangularize$(T, \mathcal{Q})$

Input: T, a regular chain; $\mathcal{Q}$, a quantifier free formula
Output: a set of regular semi-algebraic systems R_i, $i = 1, \ldots, e$, such that $W_{\mathbb{R}}(T) \cap Z_{\mathbb{R}}(\mathcal{Q}) = \cup_{i=1}^{e} Z_{\mathbb{R}}(R_i)$.

for *each conjunctive formula* $F \wedge N_{\geq} \wedge P_{>} \wedge H_{\neq}$ **do**
  output RealTriangularize$(T, F, N_{\geq}, P_{>}, H_{\neq})$;

Algorithm 36: RealTriangularize$(T, F, N_{\geq}, P_{>}, H_{\neq})$

Input: a regular chain T and a semi-algebraic system $\mathfrak{S} = [F, N_{\geq}, P_{>}, H_{\neq}]$
Output: a set of regular semi-algebraic systems R_i, $i = 1 \cdots e$, such that $W_{\mathbb{R}}(T) \cap Z_{\mathbb{R}}(\mathfrak{S}) = \cup_{i=1}^{e} Z_{\mathbb{R}}(R_i)$.

$\mathfrak{T} :=$ Triangularize(F, T);
for $C \in \mathfrak{T}$ **do**
  output RealTriangularize$(C, N_{\geq}, P_{>}, H_{\neq} \cup \mathrm{init}(T)_{\neq})$;

Algorithm 37: RealTriangularize$(T, N_{\geq}, P_{>}, H_{\neq})$

Input: a regular chain T and a semi-algebraic system $\mathfrak{S} = [\emptyset, N_{\geq}, P_{>}, H_{\neq}]$
Output: a set of regular semi-algebraic systems R_i, $i = 1, \ldots, e$, such that $W_{\mathbb{R}}(T) \cap Z_{\mathbb{R}}(\mathfrak{S}) = \cup_{i=1}^{e} Z_{\mathbb{R}}(R_i)$.

$H' := \mathrm{init}(T) \cup H$;
$\mathfrak{T} := \{[T, \emptyset]\}$; $\mathfrak{T}' := \emptyset$;
for $p \in N$ **do**
  for $[T', N'] \in \mathfrak{T}$ **do**
    $\mathfrak{T}' := \mathfrak{T}' \cup \{[C, N'] \mid C \in$ Intersect$(p, T')\}$;
    $\mathfrak{T}' := \mathfrak{T}' \cup \{[T', N' \cup \{p\}]\}$
  $\mathfrak{T} := \mathfrak{T}'$; $\mathfrak{T}' := \emptyset$;
$\mathfrak{T} := \{[T', N' \cup P, H'] \mid [T', N'] \in \mathfrak{T}\}$;
while $\mathfrak{T} \neq \emptyset$ **do**
  let $[T', P', H'] \in \mathfrak{T}$; $\mathfrak{T} := \mathfrak{T} \setminus \{[T', P', H']\}$;
  for $C \in$ RegularOnly$(T', P' \cup H')$ **do**
    $BP :=$ BorderPolynomialSet$(C, P' \cup H')$;
    $(DP, \mathcal{R}) =$ GenerateRegularSas(BP, C, P');
    if $\mathcal{R} \neq \emptyset$ **then** output $\mathcal{R}$;
    for $f \in DP \setminus (P' \cup H')$ **do**
      $\mathfrak{T} := \mathfrak{T} \cup \{[D, P', H'] \mid D \in$ Intersect$(f, C)\}$;

Proof. By the specification of Triangularize, we have $V(F) \cap W(T) \subseteq \cup_{C \in \mathfrak{T}} W(C) \subseteq V(F) \cap \overline{W(T)}$. Therefore $V(F) \cap W(T) = \cup_{C \in \mathfrak{T}} W(C) \setminus V(\mathrm{init}(T))$. Thus its termination and correctness follows directly from that of Algorithm 37. □

Proposition 9.2. *Algorithm 37 terminates and satisfies its specification.*

Proof. The terminations follows from the fact that at line 16 of this algorithm, the polynomial f is regular modulo $\mathrm{sat}(C)$, which guarantees that each new generated regular chain has smaller dimension than C. Next we prove the correctness.

Firstly, after line 8, we claim that the following three relations hold.

(i) For any $[T', P', H'] \in \mathfrak{T}$, we have $W(T') \subseteq \overline{W(T)}$.

(ii) We have $Z(T, P \cup H) = \cup_{[T',P',H'] \in \mathfrak{T}} Z(T', P' \cup H')$.

(iii) We have $W_{\mathbb{R}}(T) \cap Z_{\mathbb{R}}(N_{\geq}, P_{>}, H_{\neq}) = \cup_{[T',P',H'] \in \mathfrak{T}} W_{\mathbb{R}}(T') \cap Z_{\mathbb{R}}(P'_{>}, H'_{\neq})$.

Now we prove the claims by induction on the number of polynomials in N. If $N = \emptyset$, then $T = T'$, $P = P'$ and $H' = \mathrm{init}(T) \cup H$. So the claims clearly hold.

Let $N = N' \cup \{p\}$ and we assume that the claims hold for N'. Let $\mathfrak{T}'$ be all the set of triples $[T', P', H']$ such that $Z(T, P \cup H) = \cup_{[T',P',H'] \in \mathfrak{T}'} Z(T', P' \cup H')$, $W_{\mathbb{R}}(T) \cap Z_{\mathbb{R}}(N'_{\geq}, P_{>}, H_{\neq}) = \cup_{[T',P',H'] \in \mathfrak{T}'} W_{\mathbb{R}}(T') \cap Z_{\mathbb{R}}(P'_{>}, H'_{\neq})$ and $W(T') \subseteq \overline{W(T)}$ for any $[T', P', H'] \in \mathfrak{T}'$ hold.

Consider the loop from line 3 to 7. Let $\mathfrak{T}$ be the set of all $[T', P', H']$ after executing the final iteration of this loop and line 8. For each triple $[T', P', H'] \in \mathfrak{T}'$, the following relation hold:

$$Z(T', P' \cup H') = Z(T', P' \cup H') \cap V(p) \cup Z(T', P' \cup H') \setminus V(p). \tag{9.2}$$

By line 6, the triple $[T', P' \cup \{p\}, H']$ belongs to $\mathfrak{T}$. Let $T_1, \ldots, T_s$ be the output of Intersect. We have

$$V(p) \cap W(T') \subseteq \cup_{i=1}^{s} W(T_i) \subseteq V(p) \cap \overline{W(T')}. \tag{9.3}$$

By line 5, $[T_i, P', H']$ belongs to $\mathfrak{T}$. By induction hypothesis, we have $W(T') \subseteq \overline{W(T)}$, together with relation 9.3, we deduce that $W(T_i) \subseteq \overline{W(T)}$ and

$$V(p) \cap Z(T', H') \subseteq \cup_{i=1}^{s} Z(T_i, H') \subseteq V(p) \cap Z(T, H'). \tag{9.4}$$

Combining relations 9.2, 9.4 and induction hypothesis, we deduce that claims (i), (ii) and (iii) hold for $\mathfrak{T}$ and thus for N.

Secondly, we prove that the while loop from line 9 to 16 generates regular semi-algebraic systems R_i, $i = 1, \ldots, e$, such that $W_{\mathbb{R}}(T) \cap Z_{\mathbb{R}}(\mathfrak{S}) = \cup_{i=1}^{e} Z_{\mathbb{R}}(R_i)$. To this end, it is enough to prove the following loop invariants. For a given iteration, let $\mathcal{R}$ be the set of regular semi-algebraic systems in the current output and let $\mathfrak{T}$ be the set of unprocessed tasks $[T', P', H']$. The invariant we shall prove is

$$W_{\mathbb{R}}(T) \cap Z_{\mathbb{R}}(N_{\geq}, P_{>}, H_{\neq}) = \cup_{R\in\mathcal{R}} Z_{\mathbb{R}}(R) \cup_{[T',P',H']\in\mathfrak{T}} W_{\mathbb{R}}(T') \cap Z_{\mathbb{R}}(P'_{>}, H'_{\neq}). \quad (9.5)$$

The invariant clearly hold at the beginning by claim (*iii*). For a given iteration, we assume that the invariant holds at the beginning of it and we would like to prove that the invariant stills holds at the end of it.

For a given iteration, let $[T', P', H']$ be the task picked from $\mathfrak{T}$. Let $C_1, \ldots, C_s$ be output of RegularOnly, we have $Z(T', P' \cup H') \subseteq \cup_{i=1}^{s} Z(C_i, P' \cup H') \subseteq W(T)$, which implies that $W_{\mathbb{R}}(C_i) \cap Z_{\mathbb{R}}(P'_{>}, H'_{\neq}) \subseteq W_{\mathbb{R}}(T) \cap Z_{\mathbb{R}}(N_{\geq}, P_{>}, H_{\neq})$. Rename $\mathfrak{T}$ as the set of all $[C_i, P', H']$. We thus deduce that relation 9.5 holds for the new $\mathfrak{T}$.

Moreover, each $[C_i, P' \cup H']$ is a regular system. By the specifications of BorderPolynomialSet and GenerateRegularRsas, for each $[C, P', H'] \in \mathfrak{T}$, there exists finitely many regular semi-algebraic systems $R_1, \ldots, R_s$ and a set D such that

$$W_{\mathbb{R}}(C) \cap Z_{\mathbb{R}}(P'_{>}, H'_{\neq}) = \cup_{i=1}^{s} Z_{\mathbb{R}}(R_i) \cup_{f\in D} V_{\mathbb{R}}(f) \cap W_{\mathbb{R}}(C) \cap Z_{\mathbb{R}}(P'_{>}, H'_{\neq}).$$

For a given f and C, Intersect computes regular chains $T_1, \ldots, T_t$ such that $V(f) \cap W(C) \subseteq \cup_{i=1}^{t} W(T_i) \subseteq V(f) \cap \overline{W(C)}$. Note that $\overline{W(C)} \subseteq \overline{W(T)}$, which implies that $Z(T_i, H') \subseteq W(T)$ and therefore $W_{\mathbb{R}}(T_i) \cap Z_{\mathbb{R}}(P'_{>}, H'_{\neq}) \subseteq W_{\mathbb{R}}(T) \cap Z_{\mathbb{R}}(N_{\geq}, P_{>}, H_{\neq})$. Hence, we deduce that the invariant 9.5 stills holds at end of the iteration. □

9.3 Incremental RealTriangularize

Given a semi-algebraic system $\mathfrak{S} := [F, N_{\geq}, P_{>}, H_{\neq}]$, by passing the empty regular chain $\varnothing$ and $\mathfrak{S}$ to Algorithm 36, we obtain another algorithm for computing a full triangular decomposition of $\mathfrak{S}$. We call this algorithm an incremental one since its subroutine Triangularize computes a Lazard-Wu triangular decomposition by solving equations one by one. This incremental algorithms serves as a counterpart of the recursive algorithm in the previous chapter.

9.4 Verification of real solvers

In this section, we present an application of the set theoretic operations on the verification of polynomial system solvers computing symbolic description of the real solutions.

On a given input polynomial system, two solving tools may produce correct results that look fairly different. Proving that these two results are equivalent can be a very complex task. Here's an example. Given a triangle with edge lengths a, b, c (denoting the respective edges a, b, c too) the following two conditions C_1, C_2 are both characterizing the fact that the external bisector of the angle of a, c intersects with b on the other side of a than the triangle: $C_1 = a > 0 \wedge b > 0 \wedge c > 0 \wedge a < b+c \wedge b < a+c \wedge c < a+b \wedge (b^2+a^2-c^2 \leq 0 \vee c(b^2+a^2-c^2)^2 < ab^2(2ac-(c^2+a^2-b^2)))$, $C_2 = a > 0 \wedge b > 0 \wedge c > 0 \wedge a < b+c \wedge b < a+c \wedge c < a+b \wedge c-a > 0$. We verify the equivalence of C_1 and C_2 by computing the set-theoretical differences $C_1 \setminus C_2$ and $C_2 \setminus C_1$. The algorithm DifferenceRsas implemented as the command `Difference` of the `SemiAlgebraicSetTools` module of the `RegularChains` library can be used for this purpose. Figure 9.1 shows how the computations are conducted.

We set S1 and S2 up first. S1 is the disjuction of C1 and C2.

```
> C1:=[a > 0 , b > 0 , c > 0 , a < b + c , b < a + c , c < a + b,  b^2 + a^2 - c^2 <= 0 ]:
  C2:=[a > 0 , b > 0 , c > 0 , a < b + c , b < a + c , c < a + b,  c*(b^2 + a^2 - c^2)^2 < a*b^2*
  (2*a*c - (c^2 + a^2 - b^2))]:
  S1:=[C1, C2]; S2 := [a-c<0,  a > 0 , b > 0 , c > 0 , a < b + c , b < a + c , c < a + b];
```

$$S1 := [[0 < a, 0 < b, 0 < c, a < b+c, b < a+c, c < a+b, b^2+a^2-c^2 \leq 0], [0 < a, 0 < b, 0 < c, a < b+c, b < a+c, c < a+b, c(b^2+a^2-c^2)^2 < a b^2 (2ac - c^2 - a^2 + b^2)]]$$
$$S2 := [a-c < 0, 0 < a, 0 < b, 0 < c, a < b+c, b < a+c, c < a+b] \quad \textbf{(1)}$$

Compute regular semi-algebraic system representations dec1 (resp. dec2) for S1 (resp. S2)

```
> R := PolynomialRing([a,b,c]): dec1 := map(op,  map(RealTriangularize, S1,R));
  dec2:= RealTriangularize(S2, R);
```

dec1 := [regular_semi_algebraic_system, regular_semi_algebraic_system, regular_semi_algebraic_system]

dec2 := [regular_semi_algebraic_system] **(2)**

Compute the differences: S1 \ S2 and S2 \ S1

```
> Difference(dec1,dec2,R);
```

[ ] **(3)**

```
> Difference(dec2,dec1,R);
```

[ ] **(4)**

Figure 9.1: Testing the equivalence of two formulas by `Difference`

9.5 Experimentation

In Table 9.1, R denotes RealTriangularize. The subscripts *re* and *inc* denote respectively the recursive and incremental algorithms of RealTriangularize. The symbol RR, short name for RemoveRedundantComponents, is an algorithm for removing the redundant components in the output of R_{re} and R_{inc}. Its implementation is based on the algorithm DifferenceRsas (Algorithm 33). For each algorithm, the left column records the time (in seconds) while the right one records the number of components in the output.

This table illustrates the effectiveness of the incremental RealTriangularize and that of the tool RemoveRedundantComponents based on set-theoretic operations of semi-algebraic sets. Consider for instance the system 8-3-config-Li: R_{inc} greatly outperforms R_{re}. Moreover, RR helps reduce the number of output components of R_{re} from 203 to 45.

sys	R_{re}		RR		R_{inc}		RR	
8-3-config-Li	418.6	203	1727	45	30.5	47	129.5	47
dgp6	65.17	20	17.44	15	47.73	19	22.38	17
Leykin-1	4.9	28	20.1	18	6.5	19	13.9	19
L	14.9	69	94.3	20	2.6	19	11.7	19
Mehta0	1294	21	$> 1h$	$> 1h$	1558	20	$> 1h$	$> 1h$
EdgeSquare	247.7	116	$> 1h$	$> 1h$	116.8	43	$> 1h$	$> 1h$
Enneper	6.1	18	12.4	13	4.9	17	12.7	12
IBVP	14.1	8	$> 1h$	$> 1h$	2.5	8	$> 1h$	$> 1h$
MPV89	2.7	6	84.1	6	2.1	7	73.4	6
Xia	223.7	12	$> 1h$	$> 1h$	21.4	9	$> 1h$	$> 1h$
Lanconelli	1.1	7	2.4	6	1.0	7	2.2	6
MacLane	17.4	79	240.5	28	5.8	27	35.8	27
MontesS12	197.8	163	346.5	62	49.9	85	413.9	61
MontesS14	3.4	23	14.1	13	2.8	15	11.0	13
Pappus	750.5	409	$> 1h$	$> 1h$	29.1	119	1127.6	119
Wang168	7.0	16	8.4	10	3.4	11	5.6	10
xia-issac07-1	2.7	13	$> 1h$	$> 1h$	2.2	12	$> 1h$	$> 1h$

Table 9.1: The timing and number of output components for different algorithms

Chapter 10

Comprehensive Triangular Decomposition of Semi-algebraic Systems

Typical problems on parametric dynamical systems, such as the stability analysis of equilibria, require to decompose the parameter space into connected semi-algebraic sets above which the qualitative behavior of the dynamical system is essentially constant. Taking also into consideration the fact that certain degenerated behaviors (for instance, infinitely many complex solutions) have no practical interest, we introduce in this chapter the notion of a *comprehensive triangular decomposition of a parametric semi-algebraic system*, together with an algorithm for computing it.

10.1 Introduction

As mentioned in Chapter 1, this thesis is motivated by applications from biochemistry. In the field of biochemistry, many reaction networks are modeled by dynamical systems. The equilibria (or steady states) of a dynamical system are typically described by nonlinear parametric polynomial systems (a system of polynomial equations, inequations or inequalities with parameters), where a fundamental question is the study of the stability of these equilibria when parameters vary.

The notion of a *comprehensive triangular decomposition of a parametric semi-algebraic system* (RCTD) is introduced in Section 10.2. We propose an algorithm for computing it based on the routines presented in Chapter 6 and Chapter 7. Since this work is quite recent, several natural questions are still a work in progress and not discussed here. We observe, however, that this new type of decomposition can be used to implement fundamental operations such as *complete real root classification* of a parametric semi-algebraic system and *projection* of a semi-algebraic set.

In Section 10.3, we return to the introductory example of Chapter 1 and explain how the tool proposed in this chapter helps studying this application from biochemistry.

The notion of RCTD is related to and was encouraged by several other tools in the literature, such as the notion of cylindrical algebraic decomposition [44], the notion of border polynomial [138] and discriminant variety [86]. We remark that there are several differences between RCTD and those other tools.

Cylindrical algebraic decomposition decomposes the whose space, say $\mathbb{R}^n$, into cylindrically arranged cells $C_1, \ldots, C_e$. Recall that this implies that the projections of any two cells C_i, C_j for $1 \leq i < j \leq e$ on a $\mathbb{R}^k$, for any k with $1 \leq k < n$, are either disjoint or equal.

In contrast, RCTD decomposes part of the whole space (actually $\Pi_d^{-1}(\Pi_d(\Sigma))$, where $\Sigma \subset \mathbb{R}^n$ is the semi-algebraic set under study and $\Pi_d : \mathbb{R}^n \longrightarrow \mathbb{R}^d$ the canonical projection on the parameter space) into cells $C_1, \ldots C_e$ such that the projections of any two cells C_i, C_j for $1 \leq i < j \leq e$ on $\mathbb{R}^d$ are either disjoint or equal.

Border polynomial and discriminant variety are objects of the parameter space designed for solving parametric systems in a lazy manner. They do not provide a complete partition of $\Pi_d(\Sigma)$ even if Σ is restricted to its components that are generically zero-dimensional over $\mathbb{R}^d$. Moreover, computing the border polynomial and the discriminant variety of Σ over $\mathbb{R}^d$ does not produce a description of the solutions as functions on parameters. The notion of RCTD meets all these requirements.

This chapter is based on paper [34], co-authored with Marc Moreno Maza.

10.2 Comprehensive triangular decomposition of parametric semi-algebraic systems

Let d, m, n be positive integers such that we have $n = d + m$ and $d, m \geq 1$. Let $\mathbf{x} = x_1 < \cdots < x_n$ be ordered variables, which are divided into two groups $x_1 < \cdots < x_d$ and $x_{d+1} < \cdots < x_n$. We rename x_i as u_i for $1 \leq i \leq d$ and see $\mathbf{u} = u_1, \ldots, u_d$ as parameters. We rename x_i as y_{i-d} for $d + 1 \leq i \leq n$ and see $\mathbf{y} = y_1, \ldots, y_m$ as unknowns. In this section, we introduce the concept of a *comprehensive triangular decomposition of a parametric semi-algebraic system* $\mathfrak{S}$ (RCTD) of $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$.

We first recall some notations on semi-algebraic systems introduced in the previous two chapters.

Semi-algebraic system. Let us consider four finite polynomial subsets $F = \{f_1, \ldots, f_s\}$, $N = \{n_1, \ldots, n_t\}$, $P = \{p_1, \ldots, p_r\}$ and $H = \{h_1, \ldots, h_\ell\}$ of $\mathbb{Q}[x_1, \ldots, x_n]$. Let $N_\geq$ denote the set of the inequalities $\{n_1 \geq 0, \ldots, n_t \geq 0\}$. Let $P_>$ denote the set of the inequalities $\{p_1 > 0, \ldots, p_r > 0\}$. Let $H_\neq$ denote the set of inequations $\{h_1 \neq 0, \ldots, h_\ell \neq 0\}$. We denote by $\mathfrak{S} = [F, N_\geq, P_>, H_\neq]$ the semi-algebraic system which is the conjunction of the following conditions: $f_1 = 0, \ldots, f_s = 0$, $n_1 \geq 0, \ldots, n_t \geq 0$, $p_1 > 0, \ldots, p_r > 0$ and $h_1 \neq 0, \ldots, h_\ell \neq 0$. The system $f_1 = 0, \ldots, f_s = 0$, $p_1 \neq 0, \ldots, p_r \neq 0$ and $h_1 \neq 0, \ldots, h_\ell \neq 0$ is called the *associated constructible system* of $\mathfrak{S}$. Its zero set in $\mathbb{C}^n$ is called the *associated constructible set* of $\mathfrak{S}$. We call $[F, \emptyset, P_>, H_\neq]$, written as $[F, P_>, H_\neq]$ or $[F, P_>]$ when $H_\neq$ is empty, a *basic semi-algebraic system*.

Squarefree semi-algebraic system. Let $R := [T, P]$ be a squarefree regular system of $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$. We call the pair $A := [T, P_>]$ a *squarefree semi-algebraic system* (SFSAS). The system R is called the *associated regular system* of A.

Definition 10.1. *Let $\mathfrak{S}$ be a semi-algebraic system of $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$. Let cs be the associated constructible system of $\mathfrak{S}$. A* comprehensive triangular decomposition *of $\mathfrak{S}$ (RCTD)*[1] *is a pair $(\mathcal{C}, (\mathcal{A}_C, C \in \mathcal{C}))$, where*

- *$\mathcal{C}$ is a finite partition of $\mathbb{R}^d$ into nonempty semi-algebraic sets,*
- *for each $C \in \mathcal{C}$, $\mathcal{A}_C$ is a finite set of SFSASes of $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$ such that:*
  - *(i) either $\mathcal{A}_C$ is empty, which means that $\mathfrak{S}(u)$ is empty for each $u \in C$;*
  - *(ii) or $\mathcal{A}_C = \{[\varnothing, \{\ \}]\}$, which implies that $cs(u)$ is infinite for each $u \in C$;*
  - *(iii) or each $A = [T, P_>] \in \mathcal{A}_C$ satisfies* $\mathrm{mvar}(T) = \mathbf{y}$, $\mathrm{mvar}(P) = \mathbf{y}$ *and for each $u \in C$ we have:*
    - *the associated regular systems of $\mathcal{A}_C$ specialize well at u,*
    - *for each $A \in \mathcal{A}_C$, $Z_\mathbb{R}(A(u))$ is not empty,*
    - *$\mathfrak{S}(u) = \uplus_{A \in \mathcal{A}_C} Z_\mathbb{R}(A(u))$.*

If we further require in (iii) that C is a connected semi-algebraic set, then we call $(\mathcal{C}, (\mathcal{A}_C, C \in \mathcal{C}))$ a RCTD with connected cells.

Remark 10.1. *The RCTD we proposed in paper [34] is essentially a RCTD with connected cells as defined above. Algorithm 38 and 40 presented hereafter also compute*

[1]The "R" in the term RCTD emphasizes the fact this tool focuses on the real solutions of the input parametric polynomial system.

a RCTD with connected cells. Nevertheless, a default specification not requiring connectivity provides more flexibility for some applications such as real root classification and projection.

Remark 10.2. *By Theorem 8.1 of Section 8.2 in Chapter 8, the condition* (*iii*) *in the definition of RCTD implies that above each connected component of the cell C, the solutions of $\mathfrak{S}$ w.r.t.* $\mathbf{y}$ *are finitely many continuous functions of the parameters* $\mathbf{u}$. *Moreover, the graphs of these functions are disjoint above each connected component of C.*

In the rest of this section, we provide an algorithm for computing a RCTD. It relies on an operation called CAD for decomposing real constructible sets into connected and cylindrically arranged cells of $\mathbb{R}^n$. The operation CAD can be easily described via the three subroutines MPD, MakeCylindrical and MakeSemiAlgebraic presented in Chapter 7. The correctness of the operations CAD follow immediately from those of its three subroutines.

Calling sequence. CAD($\mathcal{C}$)

Input. $\mathcal{C} := \{C_1, \ldots, C_e\}$ is a set of pairwise disjoint constructible sets of $\mathbb{C}^n$ given by polynomials in $\mathbb{Q}[\mathbf{x}]$ such that $\mathbb{C}^n = \cup_{i=1}^{e} C_i$.

Output. A CAD[2] $\mathcal{E}$ of $\mathbb{R}^n$ such that for each element C of $\mathcal{C}$, the set $C \cap \mathbb{R}^n$ is a union of some cells in $\mathcal{E}$.

Step (1)**.** For $1 \leq i \leq e$, apply operation MPD to the family of regular systems representing C_i, so as to obtain another family $\mathcal{R}_i$ of regular systems representing C_i and whose zero sets are pairwise disjoint.

Step (2)**.** Let $\mathcal{R} := \cup_{i=1}^{e} \mathcal{R}_i$. Call algorithm MakeCylindrical($\mathcal{R}, n$), to compute a cylindrical decomposition $\mathcal{D}$ of $\mathbb{C}^n$ such that the zero set of each regular system in $\mathcal{R}$ is a union of some cells in $\mathcal{D}$.

Step (3)**.** Call algorithm MakeSemiAlgebraic to compute a CAD $\mathcal{E}$ of $\mathbb{R}^n$ such that, for each element D of $\mathcal{D}$, the set $D \cap \mathbb{R}^n$ is a union of some cells in $\mathcal{E}$.

Next we describe algorithms for computing CTD of a semi-algebraic system.

- We start by describing an algorithm for computing CTD of a *basic semi-algebraic system*, see Algorithm 38.

- We then present a general algorithm for computing CTD of an *arbitrary semi-algebraic system*, see Algorithm 40.

These two steps should help the reader understanding the underlying principles

Algorithm 38: $\mathsf{RCTD}(\mathfrak{S})$

Input: A basic semi-algebraic system $\mathfrak{S} := [F, P_>, H_{\neq}]$ of $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$.
Output: A CTD of $\mathfrak{S}$.

begin
  let $(\mathcal{C}, (\mathcal{T}_C, C \in \mathcal{C}))$ be a WDSCTD of the associated constructible set of $\mathfrak{S}$
  $\mathcal{D} := \mathsf{CAD}(\mathcal{C})$; $\mathcal{E} := \{ \}$
  for each $C \in \mathcal{C}$, for each $D \in \mathcal{D}$ such that $D \subseteq C$, let $\mathcal{T}_D = \mathcal{T}_C$
  for $D \in \mathcal{D}$ **do**
    if $\mathcal{T}_D = \{ \}$ **then**
      $E := D$; $\mathcal{A}_E := \{ \}$; $\mathcal{E} := \mathcal{E} \cup \{E\}$
    else if $\mathcal{T}_D = \{\varnothing\}$ **then**
      $E := D$; $\mathcal{A}_E := \{[\varnothing, \{ \}]\}$; $\mathcal{E} := \mathcal{E} \cup \{E\}$
    else
      let s be a sample point of D
      $E := D$; $\mathcal{E} := \mathcal{E} \cup \{E\}$; $\mathcal{A}_E := \{ \}$
      let $P_{\mathbf{y}}$ be the set of polynomials in P such that $\mathrm{mvar}(p) \in \mathbf{y}$
      if $(P \setminus P_{\mathbf{y}})_>$ *is true after evaluating at* s **then**
        for $T \in \mathcal{T}_D$ **do**
          if $\mathsf{RealRootIsolate}(T, P_{\mathbf{y}}) \neq [\ ]$ **then**
            $\mathcal{A}_E := \mathcal{A}_E \cup \{[T, P_{\mathbf{y}>}]\}$
  return $(\mathcal{E}, (\mathcal{A}_E, E \in \mathcal{E}))$
end

Algorithm 39: $\mathsf{RegularizeInequalities}(\mathfrak{S})$

Input: A semi-algebraic system $\mathfrak{S} = [F, N_{\geq}, P_>, H_{\neq}]$ of a polynomial ring R
Output: A set $\mathcal{L}$ of triples $[\mathfrak{T}', P', H']$, where $\mathfrak{T}'$ is a set of regular chains of R, P' and H' are set of polynomials in R, such that: each polynomial in $P' \cup H'$ is regular w.r.t. every regular chain in $\mathfrak{T}'$; $\cup_{[\mathfrak{T}', P', H'] \in \mathcal{L}} \cup_{T' \in \mathfrak{T}'} Z(T', H')$ is the associated constructible set of $\mathfrak{S}$; and $Z_{\mathbb{R}}(\mathfrak{S}) = \cup_{[\mathfrak{T}', P', H'] \in \mathcal{L}} \cup_{T' \in \mathfrak{T}'} W_{\mathbb{R}}(T') \cap Z_{\mathbb{R}}(P'_>, H'_{\neq})$.

begin
  $\mathfrak{T} := \mathsf{Triangularize}(F)$;
  $\mathcal{L} := \{ [\mathfrak{T}, \emptyset] \}$;
  for $p \in N$ **do**
    for $[\mathfrak{T}', P'] \in \mathcal{L}$ **do**
      $\mathcal{L} := \mathcal{L} \cup \{ [\cup_{T \in \mathfrak{T}'} \mathsf{Intersect}(p, T), P'] \}$;
      $\mathcal{L} := \mathcal{L} \cup \{ [\mathfrak{T}', P' \cup \{p\}] \}$;
  $\mathcal{L} := \{ [\mathfrak{T}', P' \cup P, H \cup P \cup P'] \mid [\mathfrak{T}', P'] \in \mathcal{L} \}$;
  $\mathcal{L} := \{ [\cup_{T' \in \mathfrak{T}'} \mathsf{RegularOnly}(T', H'), P', H'] \mid [\mathfrak{T}', P', H'] \in \mathcal{L} \}$;
  return $\mathcal{L}$
end

Algorithm 40: $\mathsf{RCTD}(\mathfrak{S})$

Input: A semi-algebraic system $\mathfrak{S} = [F, N_{\geq}, P_{>}, H_{\neq}]$ of a polynomial ring R
Output: A comprehensive triangular decomposition of $\mathfrak{S}$

begin
  $\mathcal{L} := \mathsf{RegularizeInequalities}(\mathfrak{S})$; $\mathcal{L}_0 := \{\ \}$; $\mathcal{L}_1 := \{\ \}$;
  for $[\mathfrak{T}', P', H'] \in \mathcal{L}$ **do**
    $\mathcal{R} := \{\ \}$;
    for $T' \in \mathfrak{T}'$ **do**
      if $\mathbf{y} \subseteq \mathrm{mvar}(T')$ **then** $\mathcal{R} := \mathcal{R} \cup \{[T', H']\}$;
      else $\mathcal{L}_1 := \mathcal{L}_1 \cup \{[T', H']\}$;
    $\mathcal{L}_0 := \mathcal{L}_0 \cup \{[\mathcal{R}, P']\}$;
  $\mathcal{L}_0 := \{[\mathsf{DSPCTD}(\mathcal{R}), P'] \mid [\mathcal{R}, P'] \in \mathcal{L}_0\}$; $\mathcal{L}_0 := \cup_{[\mathcal{R},P'] \in \mathcal{L}_0} \cup_{rs \in \mathcal{R}} [rs, P']$;
  let cs_1 be the projection of the constructible set $\mathcal{L}_1$ on $\mathbb{C}^d$;
  $\mathcal{C} := \emptyset$;
  for $[rs, P'] \in \mathcal{L}_0$ **do**
    $C := \mathsf{Difference}(D^{\mathbf{u}}(rs), cs_1)$; **if** $C \neq \emptyset$ **then** $\mathcal{C} := \mathcal{C} \cup \{C\}$;
  $\mathcal{C} := \mathsf{SMPD}(\mathcal{C})$;
  for $C \in \mathcal{C}$ **do**
    if C *is not empty* **then**
      let $\mathcal{A}_C$ be the set of $[rs, P'] \in \mathcal{L}_0$ with $C \subseteq D^{\mathbf{u}}(rs)$;
      $\mathcal{A}_C := \{[T_{\mathbf{y}}, P'] \mid [[T, H'], P'] \in \mathcal{L}_0\}$
  $C := cs_1$; $\mathcal{C} := \mathcal{C} \cup \{C\}$; $\mathcal{A}_C := \{[\varnothing, \{\ \}]\}$;
  $C := \mathbb{C}^d \setminus \cup_{C \in \mathcal{C}} C$; $\mathcal{C} := \mathcal{C} \cup \{C\}$; $\mathcal{A}_C := \{\ \}$;
  $\mathcal{D} := \mathsf{CAD}(\mathcal{C})$;
  for each $C \in \mathcal{C}$, for each $D \in \mathcal{D}$ such that $D \subseteq C$, let $\mathcal{A}_D = \mathcal{A}_C$;
  $\mathcal{E} := \{\ \}$;
  for $D \in \mathcal{D}$ **do**
    if $\mathcal{A}_D = \{\ \}$ *or* $\mathcal{A}_D := \{[\varnothing, \{\ \}]\}$ **then**
      $E := D$; $\mathcal{A}_E := \{\ \}$; $\mathcal{E} := \mathcal{E} \cup \{E\}$;
    else
      let s be a sample point of D;
      $E := D$; $\mathcal{E} := \mathcal{E} \cup \{E\}$; $\mathcal{A}_E := \{\ \}$;
      for $[T', P'] \in \mathcal{A}_D$ **do**
        let $P'_{\mathbf{y}}$ be the set of polynomials in P' such that $\mathrm{mvar}(p) \in \mathbf{y}$;
        if $(P' \setminus P'_{\mathbf{y}})(s)_{>}$ *is true and* $\mathsf{RealRootIsolate}(T'(s), P'_{\mathbf{y}}(s)) \neq [\ ]$ **then**
          $\mathcal{A}_E := \mathcal{A}_E \cup \{[T', P'_{\mathbf{y}}]\}$;
  return $(\mathcal{E}, (\mathcal{A}_E, E \in \mathcal{E}))$
end

To prove the correctness of the algorithms, we first establish the following lemma.

Lemma 10.1. *Let* $\mathbf{k}[\mathbf{u}, \mathbf{y}]$ *be a polynomial ring. Let* T *be a regular chain of* $\mathbf{k}[\mathbf{u}, \mathbf{y}]$ *such that* $\mathrm{mvar}(T) = \mathbf{y}$. *Let* p *be a polynomial of* $\mathbf{k}[\mathbf{u}, \mathbf{y}]$. *Let* $u \in \mathbf{K}^d$ *such that* T *specializes well at* u *and such that* $p(u, y) \neq 0$ *holds for any* $(u, y) \in W(T)$. *Then* p *is regular modulo* $\mathrm{sat}(T)$.

Proof. Since T specializes well at u and $p(u, y) \neq 0$ for any $(u, y) \in W(T)$, the polynomial $p(u, \mathbf{y})$ is invertible (and thus regular) modulo $\langle T(u, \mathbf{y}) \rangle$. Thus, the regular system $[T, p]$ specializes well at u. Proposition 6.9 implies that $\mathrm{res}(p, T)(u) \neq 0$ holds. Therefore $\mathrm{res}(p, T) \neq 0$ holds too and p is regular modulo $\mathrm{sat}(T)$. □

Proposition 10.1. *Algorithm 38 terminates and satisfies its specification.*

Proof. By the specification of WDSCTD and Lemma 10.1, it is easy to deduce that each element of $\mathcal{A}_E$ is an SFSAS. Then the termination and the correctness of the algorithm follow directly from the specifications of its subroutines and the definition of a RCTD. □

Proposition 10.2. *Algorithm 39 terminates and satisfies its specification.*

Proof. We have the following observations

- Algorithm Triangularize compute a set of regular chains $\mathfrak{T}$ such that $V(F) = \cup_{T \in \mathfrak{T}} W(T)$.
- For each $p \in N$, line 6 and 7 consider respectively the case $p = 0$ and $p \neq 0$.
- For a regular chain $T \in \mathfrak{T}'$ and a polynomial $p \in N$, algorithm Intersect compute regular chains $T_1, \ldots, T_s$ such that $V(p) \cap W(T) \subseteq \cup_{i=1}^{s} W(T_i) \subseteq V(p) \cap \overline{W(T)}$, moreover $W(T_i) \subseteq \overline{W(T)} \subseteq V(F)$.
- For a regular chain $T \in \mathfrak{T}'$ and a set of polynomials H', algorithm RegularOnly computes regular chains $T_1, \ldots, T_t$ such that $Z(T, H') \subseteq \cup_{i=1}^{t} Z(T_i, H') \subseteq Z(F, H')$.

From the above arguments, we can easily deduce the conclusion. □

Proposition 10.3. *Algorithm 40 terminates and satisfies its specification.*

Proof. Firstly, similar to the proof of algorithm 38, each element of $\mathcal{A}_E$ is an SFSAS. Secondly, algorithm 39 decomposes the input system as disjoint systems. Then the termination and correctness of the algorithm follows easily from the specifications of DSPCTD and other subroutines. □

[2]That is, finitely many constructible sets of $\mathbb{R}^n$ which are connected and cylindrically arranged.

10.3 Example

In this section we revisit the biochemistry network presented in the introductory Chapter and explain more formally how to describe its equilibria with CTD. The notions related to dynamical systems are defined in next Chapter.

The dynamical system governing the biochemistry network is

$$\left\{ \begin{array}{lcl} \frac{\mathrm{d}x}{\mathrm{d}t} & = & f_1 \\ \frac{\mathrm{d}y}{\mathrm{d}t} & = & f_2 \end{array} \right. \quad \text{with} \quad \left\{ \begin{array}{lcl} f_1 & = & \frac{16000+800y^4-20k_2x-k_2xy^4-2x-4xy^4}{20+y^4} \\ f_2 & = & \frac{2(x+2xy^4-500y-25y^5)}{20+y^4} \end{array} \right. . \tag{10.1}$$

Let $x, y \in \mathbb{R}^2$ be an equilibrium of it. By Routh-Hurwitz criterion (x, y) is asymptotically stable if

$$\Delta_1 = -(\frac{\partial f_1}{\partial x} + \frac{\partial f_2}{\partial y}) > 0 \quad \text{and} \quad a_2 = \frac{\partial f_1}{\partial x} \cdot \frac{\partial f_2}{\partial y} - \frac{\partial f_1}{\partial y} \cdot \frac{\partial f_2}{\partial x} > 0.$$

In System (10.1), let p_1 and p_2 be respectively the numerators of f_1 and f_2. We have

$$\begin{aligned} p_1 &= 16000 + 800y^4 - 20k_2x - k_2xy^4 - 2x - 4xy^4 \\ p_2 &= 2x + 4xy^4 - 1000y - 50y^5 \end{aligned}$$

The Hurwitz determinants Δ_1 and a_2 are rational functions with the same denominator $(y^4 + 20)^2$, which is always positive. So we can safely set Δ_1 (resp. a_2) to the value of its numerator, and then have

$$\begin{aligned} \Delta_1 &= 400k_2 + 40k_2y^4 + k_2y^8 + 20040 + 2082y^4 + 54y^8 - 312xy^3 \\ a_2 &= 50k_2y^8 + 200y^8 + 2000k_2y^4 + 4100y^4 - 312k_2xy^3 + 2000 + 20000k_2 \end{aligned}$$

The parametric semi-algebraic systems $\mathcal{S}_1 : \{p_1 = p_2 = 0, x > 0, y > 0, k_2 > 0\}$ and $\mathcal{S}_2 : \{p_1 = p_2 = 0, k_2 > 0, x > 0, y > 0, \Delta_1 > 0, a_2 > 0\}$ encode respectively the equilibria and the asymptotically stable hyperbolic equilibria of System (10.1).

Next we take $\mathcal{S}_1$ as an example and show how to compute a RCTD of it. Let $\mathcal{C}_1 := \{p_1 = 0, p_2 = 0, x \neq 0, y \neq 0, k_2 \neq 0\}$ be the associated constructible set of $\mathcal{C}_1$. Under the order $x > y > k_2$, the zero set of $\mathcal{C}_1$ in $\mathbb{C}^3$ is a union of the zero sets of the following two regular systems.

$$R_1 := \begin{cases} (2y^4+1)x - 500y - 25y^5 &= 0 \\ (k_2+4)y^5 - 64y^4 + (20k_2+2)y - 32 &= 0 \\ y &\neq 0 \\ 2y^4+1 &\neq 0 \\ 32y^4+39y+16 &\neq 0 \\ k_2 &\neq 0 \\ k_2+4 &\neq 0 \end{cases}, \quad R_2 := \begin{cases} 2x - 25y + 400 &= 0 \\ 32y^4+39y+16 &= 0 \\ k_2+4 &= 0 \end{cases}.$$

A WDSCTD of $\mathcal{C}_1$ is given by the following piecewise definition: Denote $t_x := (2y^4 + 1)x - 25y^5 - 500y$ and

$$\begin{aligned} r \;&:=\; 100000k_2^8 + 1250000k_2^7 + 5410000k_2^6 + 8921000k_2^5 - 9161219950k_2^4 \\ &-\; 5038824999k_2^3 - 1665203348k_2^2 - 882897744k_2 + 1099528405056. \end{aligned}$$

Let t_y be the following polynomial.

$$\begin{aligned}
t_y :=\; &(2326873455645089841988809228968458824000 0k_2^7 + 88780850506496261345607404805520327377600 0k_2^6 \\
&-642759201042010454260920807356084733986376100k_2^5 + 7989824659486893851802247863096235947462712 60k_2^4 \\
&-755541969292212808074758347883749169568015348 1k_2^3 - 354490122054179307333155209793159741188459844 92k_2^2 \\
&-43187513006063218081065459377570170905928820 96k_2 - 327907507955945276712462277765503291468450043456)y^4 \\
&+(595041692603879832727686208640106565435559923 20 - 14551534965517185002251506600155820489600000k_2^6 \\
&+554155115787515258964077274056246573122407566 20k_2^3 + 876847598754269841148937318213026162350958803520k_2 \\
&+3177495995308664571240595910883186607328823146 40k_2^2 - 85482628839848006177137048155404915235216000k_2^5 \\
&-1203526487705166354151311065571798686400000k_2^7 + 10178560608897625817552584862270339173953830200k_2^4)y^3 \\
&+(5252669517785054020278014804788614352000000k_2^7 - 16753027097826670885692067112239680645521920 0k_2^4 \\
&+11523510969163956265499386102221826657142922912 0k_2^2 + 1816672724083305207547642268950808404726365096960 \\
&+66831991210048304262543260260696987086776334976 0k_2 + 11286257394981172041497956130156500898560000k_2^6 \\
&-136191397343195728348723172154341170533120 00k_2^5 + 20906210233179434530990527059307460720922739760k_2^3)y^2 \\
&+(30508750939128024685030516938551128014007902952 0k_2 - 34335647706142426843782091772365121885544300 0k_2^5 \\
&+2573715300740790233035013735033453529209800 00k_2^6 + 32256100951459497483205914682740335606125645595k_2^3 \\
&-44547693984901306602292687558402129605000 0k_2^7 + 29468738920316806213601355334670213121993449540k_2^2 \\
&+112004292267797955734352101659152288598374293472 0 + 21364275064711070738627253091632191019312918 00k_2^4)y \\
&-163196051967222632295941353115340613924202875904 0 + 75292380532982828787180784712942754960000 0k_2^7 \\
&+116443127598064787316507772151330198618400 00k_2^6 + 73731947099039339859987890390367860844400240 0k_2^4 \\
&-31464169659054939689559627056171259981405867264 0k_2 - 226733546531989363631975695021134672123615921280k_2^2 \\
&-72051937593559000483331392372548407242074867040k_2^3 - 36459430774099029470221083895264625640546400 0k_2^5
\end{aligned}$$

Let R_3 be the regular system $[t_x = 0, t_y = 0, r = 0]$. Then the following piecewise definition describes a WDSCTD (also a DSCTD) of $\mathcal{C}_1$:

$$\begin{cases} \{\ \} & k_2 = 0 \\ \{R_2\} & k_2 + 4 = 0 \\ \{R_3\} & r = 0 \\ \{R_1\} & k_2 \neq 0, k_2 + 4 \neq 0 \text{ and } r \neq 0 \end{cases}.$$

The polynomial r has four real roots, two of them are positive, which we denote by $0 < \alpha_1 < \alpha_2$. Let B_1 and B_2 be the squarefree semi-algebraic systems:

$$B_1 := \begin{cases} (2y^4+1)x - 25y^5 - 500y & = & 0 \\ (k_2+4)y^5 - 64y^4 + (2+20k_2)y - 32 & = & 0 \\ y & > & 0 \end{cases}, \quad B_2 := \begin{cases} t_x & = & 0 \\ t_y & = & 0 \\ y & > & 0 \end{cases}.$$

Then a RCTD of $\mathcal{S}_1$ is given by the following piecewise definition:

$$\begin{cases} \{\ \} & k_2 \leq 0 \\ \{B_1\} & 0 < k_2 < \alpha_1 \\ \{B_2\} & k_2 = \alpha_1 \\ \{B_1\} & \alpha_1 < k_2 < \alpha_2 \\ \{B_2\} & k_2 = \alpha_2 \\ \{B_1\} & k_2 > \alpha_2 \end{cases}$$

Here each cell is either a single point or an open interval in $\mathbb{R}$, and thus is connected. Above each cell, the solutions of the regular chain B_1 (or B_2) in x, y are the equilibria of the biochemistry network. They are continuous functions of k_2 and the graphs of the functions are disjoint above each cell.

Chapter 11

Semi-algebraic Description of the Equilibria of Dynamical Systems

In this chapter, we study continuous dynamical systems defined by autonomous ordinary differential equations, themselves given by parametric rational functions. For such systems, we provide semi-algebraic descriptions of their hyperbolic and non-hyperbolic equilibria, their asymptotically stable hyperbolic equilibria, their Hopf bifurcations. To this end, we revisit various criteria on sign conditions for the roots of a real parametric univariate polynomial. In addition, we demonstrate the notion of *comprehensive triangular decomposition* of a semi-algebraic system, introduced in last chapter, is well adapted for our study.

11.1 Introduction

The study of polynomial dynamical systems by means of symbolic computation is one of the most popular application of computer algebra. Equilibria, limit cycles, center manifolds, normal forms and bifurcation analysis are the main notions used in the study of dynamical systems [105, 23, 72, 111]. These objects can be manipulated by a variety of symbolic methods [40, 38, 39, 142, 59, 122, 93, 79, 80, 129, 71, 65, 64, 75, 128, 25, 106]. Among these notions, those which have received the greatest attention by the computer algebra community are equilibria and bifurcation analysis. Studying them for polynomial dynamical systems typically consists of: (1) setting up a (parametric) semi-algebraic system $\mathcal{S}$, (2) extracting from $\mathcal{S}$ some particular information.

The aim of this work is twofold. Our first objective is to revisit the results that are practically useful for finding equilibria and bifurcation by means of symbolic computation. These results are gathered in Sections 11.2 and 11.3. They are generally stated in terms of the coefficients of a univariate polynomial and translate into semi-algebraic systems. A prototype of such results is the Routh-Hurwitz's criterion. While many of these criteria appear in the literature (for instance in [79, 80]) we also provide some new criteria, like Theorem 11.9, as well as new interpretation of classical results, like Theorem 11.13.

Our second objective is to exhibit tools that are well adapted for solving the semi-algebraic systems implementing the above mentioned results. Typical problems on parametric dynamical systems (see Problems 1, 2, 3) require to decompose the parameter space into connected semi-algebraic sets above which the qualitative behavior of the dynamical system is essentially constant. Taking also into consideration the fact that certain degenerated behaviors have no practical interest, we introduce, in Section 10.2, the notion of a *comprehensive triangular decomposition of a parametric semi-algebraic system* (CTD), together with an algorithm for computing it. In Section 11.4 of this chapter, we exhibit that CTD is indeed a very useful tool in analyzing the stability of dynamical systems.

We dedicate the rest of this introduction to identify problems arising in the study of dynamical systems which are eligible to solutions based on semi-algebraic system solving. Some of these problems, namely Problems 1, 2, 3, are directly formulated in terms of dynamical systems. For a sake of clarity, the other problems, namely Problems 4 and 5, are stated in terms of conditions on the roots of a parametric univariate polynomial, which is meant to be the characteristic polynomial of the Jacobian matrix of the dynamical system under study.

We consider continuous dynamical systems defined by autonomous ordinary differential system of the following shape:

$$\left\{\begin{array}{lcl} \dot{y_1} & = & F_1(u_1, \ldots, u_d, y_1, \ldots, y_m), \\ \dot{y_2} & = & F_2(u_1, \ldots, u_d, y_1, \ldots, y_m), \\ \vdots & & \vdots \\ \dot{y_m} & = & F_m(u_1, \ldots, u_d, y_1, \ldots, y_m). \end{array}\right. \tag{11.1}$$

where $F_1, \ldots, F_m$ are polynomials of $\mathbb{Q}[u_1, \ldots, u_d, y_1, \ldots, y_m]$. The variables $\mathbf{u} = (u_1, \ldots, u_d)$ are considered as parameters and the variables $\mathbf{y} = (y_1, \ldots, y_m)$ are seen as unknowns. In addition, we have $y_i = y_i(t)$ and $\dot{y_i} = \mathrm{d}y_i/\mathrm{d}t$ while the parameters $u_1, \ldots, u_d$ are independent of the derivation variable t. In the sequel, we simply write (11.1) as

$$\dot{\mathbf{y}} = F(\mathbf{u}, \mathbf{y}) \tag{11.2}$$

where $F(\mathbf{u}, \mathbf{y}) = (F_1(\mathbf{u}, \mathbf{y}), \ldots, F_m(\mathbf{u}, \mathbf{y}))$ is called the *vector field* of the system.

For any given parameter value $u \in \mathbb{R}^d$, one may notice that any $y \in \mathbb{R}^m$ such that $F_1(u, y) = \cdots = F_m(u, y) = 0$ holds, is a constant solution of System (11.1), which is called an *equilibrium* (or a *steady state*, or a *fixed point*). We are interested in the following problem regarding the equilibria of the given dynamical system.

Problem 1. *For a fixed parameter value u (or in absence of parameters) determine the number of equilibria of (11.1) and compute each of them (for instance, by means of isolation intervals). In presence of parameters, partition the parameter space into connected semi-algebraic sets, such that above each of them, the number of equilibria is constant and each equilibrium is a continuous function of the parameters.*

Problems 1 is a particular instance of the solving of semi-algebraic systems. Section 10.2 is dedicated to this more general question, with a view toward Problem 1.

We consider now a fixed parameter value u and a particular equilibrium y of System (11.1) at u. An important problem concerning the equilibrium y is to analyze its stability. We say y is *stable* if any solution of System (11.1) starting out close to y remains close to it. We say y is *asymptotically stable* if y is *stable* and if the solutions of System (11.1) starting out close to y become arbitrary close to it. If y is not stable, it is said to be *unstable*. The above discussion leads to enhance Problem 1 into the following ones, which deals with the number of asymptotically stable equilibria of System (11.1) depending or not on parameters.

Problem 2. *For a fixed parameter value u (or in absence of parameters) determine the number of asymptotically stable hyperbolic equilibria of (11.1) and compute each of them. In presence of parameters, partition the parameter space into connected semi-algebraic sets, such that above each of them, the number of asymptotically stable hyperbolic equilibria is constant and each of these equilibria is a continuous function of the parameters.*

The study of the system near the particular equilibrium y is usually done using the linear system

$$\dot{\mathbf{y}} = J(u, y)(\mathbf{y} - y), \tag{11.3}$$

where J is the *Jacobian matrix* of F:

$$J = \begin{pmatrix} \frac{\partial F_1}{\partial y_1} & \frac{\partial F_1}{\partial y_2} & \cdots & \frac{\partial F_1}{\partial y_m} \\ \frac{\partial F_2}{\partial y_1} & \frac{\partial F_2}{\partial y_2} & \cdots & \frac{\partial F_2}{\partial y_m} \\ \vdots & \vdots & & \vdots \\ \frac{\partial F_m}{\partial y_1} & \frac{\partial F_m}{\partial y_2} & \cdots & \frac{\partial F_m}{\partial y_m} \end{pmatrix}$$

We denote by

$$f(\lambda) = a_0\lambda^m + a_1\lambda^{m-1} + a_2\lambda^{m-2} + \cdots + a_{m-1}\lambda + a_m,$$

where $a_0 = 1$, the characteristic polynomial of J. If the matrix $J(u, y)$ has no eigenvalues with zero real parts, that is, if $f(u, y, \lambda)$ has no roots with zero real parts, then y is called a *hyperbolic equilibrium* at u; otherwise y is a *non-hyperbolic equilibrium* at u. In [107], Hartman and Grobman proved the following result: if y is a hyperbolic equilibrium, then near y, the phase portrait of the dynamical system (11.1) is topologically equivalent to that of the linearized dynamical system (11.3). The results imply that, for a hyperbolic equilibrium y, the phase flow of (11.1) is asymptotically stable near y if and only if the phase flow of (11.3) is asymptotically stable near y. Therefore, using standard results on linear differential systems [2], the phase flow of (11.1) is asymptotically stable near y if and only if all the complex roots of $f(u, y, \lambda)$ have negative real parts. This reduces Problem 2 to the following problem.

Problem 2' *For a univariate polynomial $f(x) \in \mathbb{R}[x]$, determine whether all the complex roots of $f(x)$ have negative real parts or not.*

In the above analysis, we assume the equilibrium y is hyperbolic, so a natural question is how to determine whether y is hyperbolic or not. In other words, we want to solve the following problem:

Problem 3. *For a fixed parameter value u, determine whether each equilibrium of (11.1) is hyperbolic or not. In presence of parameters, partition the parameter space into connected semi-algebraic sets, such that above each of them, an equilibrium is always either hyperbolic or non-hyperbolic.*

This problem is equivalent to determine whether all the complex roots of the characteristic polynomial $f(u, y, \lambda)$ have nonzero real parts, which leads to the following general problem.

Problem 3' *For a univariate polynomial $f(x) \in \mathbb{R}[x]$, determine whether $f(x)$ has complex roots with zero real parts or not.*

When y is a non-hyperbolic equilibrium of (11.1), if the characteristic polynomial $f(u, y, \lambda)$ has at least one complex root with positive real part, then y is an unstable equilibrium. Otherwise, the stability of y depends also on the higher order terms of the Taylor expansion of F near the point y. In this situation, one usually needs to apply the *Centre Manifold Theorem* [23] to reduce the original system to a low dimensional dynamical system defined on a centre manifold and further simplify it by computing its normal form. Finally, the normal form can be further reduced by removing terms that do not affect the stability of the equilibrium. Therefore, the first step towards stability analysis of non-hyperbolic equilibria of (11.1) is to determine when the characteristic polynomial has at least one complex root with positive real part or, equivalently, determine when $f(u, y, \lambda)$ has only complex roots with non-positive real parts, which leads to the following problem.

Problem 4. *For a univariate polynomial $f(x)$ with parametric coefficients, determine whether $f(x)$ has at least one complex root with positive real part. Equivalently, given two integers k_1 and k_2, determine whether $f(x)$ has zero as a root of multiplicity k_1 and k_2 pairs of purely imaginary roots while all the other complex roots have negative real parts.*

When non-hyperbolic equilibria are present, another more interesting phenomenon is the appearance of bifurcation. For the dynamical system (11.1), a *bifurcation* occurs at a parameter α_0 if there are parameter values α_1 arbitrarily close to α_0 with dynamics topologically non-equivalent to those at α_0. For example, the number or stability of equilibria or periodic orbits of (11.1) may change with perturbations of u from α_0. For a general dynamical system, such as (11.1), a systematic study is difficult. However, given an equilibrium y of (11.1) at u, necessary conditions for bifurcation can be obtained as follows. If a bifurcation of an equilibrium occurs near (u, y), then either or both conditions below are met:

- the characteristic polynomial f has zero as a root of multiplicity k, for some $k > 0$,
- the characteristic polynomial f has k pairs of purely imaginary roots, for some $k > 0$.

Therefore, the last problem we want to answer in this paper is as follows:

Problem 5. *Given non-negative integers k_1, k_2 and a polynomial $f(x)$ with parametric coefficients, determine whether $f(x)$ has zero as a root of multiplicity k_1 and k_2 pairs of purely imaginary roots while no other roots have zero real parts.*

A particular case of the above problem is $(k_1, k_2) = (0, 1)$. In this case, thus if the characteristic polynomial $f(u, y, \lambda)$ has a pair of purely imaginary roots and no other roots with zero real part, the limit cycle bifurcation that may occur at (u, y) is called a *Hopf bifurcation*. Such bifurcation has attracted the interest of many authors. In [71], the authors presented sufficient conditions for the appearance of Hopf bifurcations. In [79], the authors give sufficient and necessary conditions on Hopf bifurcations by further demanding that all the other eigenvalues have negative real roots, which is convenient for applying *Centre Manifold Theory* in order to reduce the dimension of dynamical systems. In [80], the authors present a framework for solving Problem 5.

This chapter is based on paper [34], co-authored with Marc Moreno Maza.

11.2 On the complex roots of a univariate polynomial

As we have seen in the previous section, many problems related to dynamical systems reduce to studying the complex roots of a univariate polynomial with real coefficients. In particular, Problems 2', 3', 4 and 5 will be completely answered in the present section.

This section is firmly rooted in the papers [79, 80]. With respect to [79, 80] our main contribution in this section is Theorem 11.9, from which the main result of [79] (that is, Theorem 3.6 in [79] and Corollary 11.3 in this section), dedicated to Hopf bifurcation, can easily be derived. Theorem 11.9 provides two equivalent conditions for a polynomial with real coefficients to have only complex roots with non-positive real parts.

The proof of the first condition relies on several results of [79, 80], which are reviewed hereafter for the reader's convenience. To prove the second condition, we introduce Corollary 11.2 and Theorem 11.7. It should be pointed out that to deduce Corollary 11.3 from Theorem 11.9, this second condition is really needed. We also correct the error of sign difference in Theorem 3.1 of [79] (Theorem 1 in [80]) and revise it as Theorem 11.5 hereafter.

Let $f(x) \in \mathbb{R}[x]$ be a polynomial of degree m, and let us write

$$f(x) = a_0x^m + a_1x^{m-1} + \cdots + a_m.$$

After recalling the definition and standard properties (Lemma 11.1, Theorems 11.1, 11.3, 11.2, 11.4) of Hurwitz determinants, we discuss their relations with subresultant sequences in Section 11.2.2 and their use in the study of symmetric roots in Section 11.2.3.

Definition 11.1 (Hurwitz matrix)**.** *We call* Hurwitz matrix *of f the $m \times m$ matrix $H = (H_{\mu\nu})$ defined by $H_{\mu\nu} = a_{2\nu-\mu}$ for $\nu = 1, \ldots, m$ and $\mu = 1, \ldots, m$, with the convention that $a_i = 0$ holds as soon as $i < 0$ or $i > m$ holds. For $i = 1, \ldots, m$, we denote by Δ_i the leading principal minors of H, which are called the* Hurwitz determinants *of H:*

$$\Delta_1 = a_1,\ \Delta_2 = \begin{vmatrix} a_1 & a_3 \\ a_0 & a_2 \end{vmatrix}, \ldots, \Delta_m = \begin{vmatrix} a_1 & a_3 & a_5 & \cdots & \cdots \\ a_0 & a_2 & a_4 & \cdots & \cdots \\ 0 & a_1 & a_3 & a_5 & \cdots \\ 0 & a_0 & a_2 & a_4 & \cdots \\ & & & & \ddots \end{vmatrix}.$$

It is easy to see that we have $\Delta_m = a_m \Delta_{m-1}$.

The following criterion provides a sufficient and necessary condition for a polynomial f to have only roots with negative real parts, which is therefore an answer to Problem 11.1.

Theorem 11.1 (Routh-Hurwitz's criterion [62])**.** *The real parts of all the zeros of $f(\lambda)$ are negative if and only if $\Delta_1 > 0$, $\Delta_2 > 0$, ..., $\Delta_{m-1} > 0$, $a_m > 0$.*

There is also another famous criterion equivalent to the above one, which is called Liénard-Chipart's Criterion.

Theorem 11.2 (Liénard-Chipart's criterion [62])**.** *The real parts of all the zeros of $f(\lambda)$ are negative if and only if we have:*

(1) *If m is odd, then all the below inequalities hold:*

$$a_m > 0,\ a_2 > 0,\ a_4 > 0,\ \ldots,\ a_{m-1} > 0,\ \Delta_2 > 0,\ \Delta_4 > 0,\ \ldots,\ \Delta_{m-1} > 0.$$

(2) *If m is even, then all the below inequalities hold:*

$$a_m > 0,\ a_1 > 0,\ a_3 > 0,\ \ldots,\ a_{m-1} > 0,\ \Delta_1 > 0,\ \Delta_3 > 0,\ \ldots,\ \Delta_{m-1} > 0.$$

11.2.1 Hurwitz determinants and stability of hyperbolic equilibria of dynamical system

In this section, for a fixed parameter value $u \in \mathbb{R}^d$, let $y \in \mathbb{R}^m$ be an equilibrium of dynamical system (11.1).

Lemma 11.1 (Orlando's formula [60]). *Let λ_i, $i = 1, \ldots, m$, be the eigenvalues of $J(u, y)$ and Δ_{m-1} be the $(m-1)$-th Hurwitz determinant of its characteristic polynomial. Then we have:*

$$\Delta_{m-1} = (-1)^{\frac{1}{2}m(m-1)} \prod_{1 \le i < j \le m} (\lambda_i + \lambda_j).$$

Corollary 11.1 (Hyperbolic equilibrium criterion). *The following three properties hold.*

(1) *$J(u, y)$ have no zero eigenvalues if and only if $|J(u, y)| = (-1)^m a_m \neq 0$.*

(2) *If $\Delta_{m-1} \neq 0$, then $J(u, y)$ has no pure imaginary eigenvalues.*

(3) *If $\Delta_m = a_m \Delta_{m-1} \neq 0$, then y is a hyperbolic equilibrium.*

Proof. Property (1) is clear. Property (2) is an immediate consequence of Orlando's Formula. Property (3) follows from $|J(u, y)| = \lambda_1 \lambda_2 \cdots \lambda_m$. □

Remark 11.1. *Necessary and sufficient conditions for $J(u, y)$ to have no pure imaginary eigenvalues (resp. y to be hyperbolic equilibrium) will be provided in Section 11.2.3.*

Theorem 11.3 (Lyapunov's first method on stability [100]). *The following properties hold.*

(*i*) *If $J(u, y)$ has at least one eigenvalue with positive real parts, then y is unstable.*

(*ii*) *Assume that y is a hyperbolic equilibrium. If all the eigenvalues of $J(u, y)$ have negative real parts, then y is asymptotically stable.*

Theorem 11.4 (Stability criterion for hyperbolic equilibria). *Let y be an equilibrium of System (11.1), we have:*

(1) *y is an asymptotically stable hyperbolic equilibrium if and only if*

$$\Delta_1 > 0,\ \Delta_2 > 0,\ \ldots,\ \Delta_{m-1} > 0,\ a_m > 0.$$

(2) *If y is hyperbolic, then y is unstable if and only if there exists some i, $1 \leq i \leq n$, such that $\Delta_i \leq 0$.*

Proof. Directly by Theorem 11.3 and Routh-Hurwitz Criterion. □

11.2.2 Hurwitz determinants and subresultant sequences

Let $\mathbb{A} = \mathbb{Q}[a_0, \ldots, a_m]$ and $f \in \mathbb{A}[x] = a_0x^m + a_1x^{m-1} + \cdots + a_{m-1}x + a_m$ be a polynomial of degree m. We write $f(x) = f_1(x^2) + xf_2(x^2)$. If $m = 2\ell + 1$, we have $f_1(y) = a_1y^\ell + a_3y^{\ell-1} + \cdots + a_{2\ell+1}$ and $f_2(y) = a_0y^\ell + a_2y^{\ell-1} + \cdots + a_{2\ell}$. If $m = 2\ell$, we have $f_1(y) = a_0y^\ell + a_2y^{\ell-1} + \cdots + a_{2\ell}$ and $f_2(y) = a_1y^{\ell-1} + a_3y^{\ell-2} + \cdots + a_{2\ell-1}$.

Theorem 11.5. *Let $\Delta_1, \Delta_2, \ldots, \Delta_m$ be the Hurwitz determinants sequence of f. Then the following conclusion holds:*

(*i*) *If $m = 2\ell + 1$, we have $\Delta_{m-1-2i} = \Delta_{2\ell-2i} = (-1)^{\frac{(\ell-i)(\ell-i-1)}{2}} s_i(f_1, \ell, f_2, \ell, y)$ hold, for $i = 0, 1, \ldots, \ell - 1$.*

(*ii*) *If $m = 2\ell$, we have $\Delta_{m-1-2i} = \Delta_{2\ell-1-2i} = (-1)^{\frac{(\ell-i)(\ell-i-1)}{2}} s_i(f_1, \ell, f_2, \ell-1, y)$, for $i = 0, 1, \ldots, \ell - 1$.*

(*iii*) *If $m = 2\ell + 1$, for $i = 0, 1, \ldots, \ell$, we have*

$$\begin{aligned} \Delta_{m-2i} = \Delta_{2\ell+1-2i} &= (-1)^{\frac{(\ell-i)(\ell-i+1)}{2}} s_i(f_1, \ell, yf_2, \ell + 1, y) \\ &= (-1)^{\frac{3(\ell-i)(\ell-i+1)}{2}} s_i(yf_2, \ell + 1, f_1, \ell, y). \end{aligned}$$

(*iv*) *If $m = 2\ell$, we have $\Delta_{m-2i} = \Delta_{2\ell-2i} = (-1)^{\frac{(\ell-i)(\ell-i+1)}{2}} s_i(f_1, \ell, yf_2, \ell, y)$ hold, for $i = 0, 1, \ldots, \ell - 1$.*

Proof. Here, we only prove (i) holds and leave the other cases for exercise.

When $m = 2\ell+1$, we have $f_1(y) = a_1y^\ell + a_3y^{\ell-1} \cdots + a_m$, $f_2(y) = a_0y^\ell + a_2y^{\ell-1} \cdots + a_{m-1}$. So the Sylvester matrix M formed by the coefficients of f_1 and f_2 is an $2\ell \times 2\ell$

matrix of the form:

$$M = \begin{bmatrix} a_1 & a_3 & a_5 & \cdots & a_m & & & & \\ & a_1 & a_3 & a_5 & \cdots & a_m & & & \\ & & \ddots & \ddots & & & \ddots & & \\ & & & a_1 & a_3 & a_5 & \cdots & a_m & \\ a_0 & a_2 & a_4 & \cdots & a_{m-1} & & & & \\ & a_0 & a_2 & a_4 & \cdots & a_{m-1} & & & \\ & & \ddots & \ddots & & & \ddots & & \\ & & & a_0 & a_2 & a_4 & \cdots & a_{m-1} & \end{bmatrix} \tag{11.4}$$

On the other hand, the Hurwitz matrix H of f is an $(2\ell+1) \times (2\ell+1)$ matrix whose elements are arranged like this:

$$H = \begin{bmatrix} a_1 & a_3 & a_5 & \cdots & a_m & & & & \\ a_0 & a_2 & a_4 & \cdots & a_{m-1} & & & & \\ & a_1 & a_3 & a_5 & \cdots & a_m & & & \\ & a_0 & a_2 & a_4 & \cdots & a_{m-1} & & & \\ & & & \cdots & \cdots & & & & \\ & & a_1 & a_3 & a_5 & \cdots & a_m & & \\ & & a_0 & a_2 & a_4 & \cdots & a_{m-1} & & \\ & & & a_1 & a_3 & \cdots & a_{m-2} & a_m & \end{bmatrix} \tag{11.5}$$

Let H^* be the sub-matrix composed by the first 2ℓ rows and 2ℓ columns of H. We denote by H_{2i} the sub-matrix of H^*, formed by the first $2i$ rows and $2i$ columns, for $i = 1, 2, \ldots, \ell$. We denote by M_i the sub-matrix of M, formed by deleting the last i rows composed by the coefficients of $f_1(y)$ and the last i rows composed by the coefficients of $f_2(y)$ and then deleting the last $2i$ columns for $i = 0, 1, \ldots, \ell - 1$. Then it's easy to see that if we make the odd rows of $H_{2\ell-2i}$ "float up" one by one, we finally get the matrix M_i. So the number of row exchanges for $H_{2\ell-2i}$ is: $0 + 1 + 2 + \cdots + (\ell - i - 1) = \frac{(\ell-i)(\ell-i-1)}{2}$. Therefore, we have $\Delta_{2\ell-2i} = |H_{2\ell-2i}| = (-1)^{\frac{(\ell-i)(\ell-i-1)}{2}} |M_i| = (-1)^{\frac{(\ell-i)(\ell-i-1)}{2}} s_i(f_1, \ell, f_2, \ell, y)$, for $i = 0, 1, \ldots, \ell - 1$.

□

Remark 11.2. *This theorem is a corrected version of Theorem 1 in [80], where the sign differences between Δ_i and s_i are wrong.*

11.2.3 Hurwitz determinants and symmetric roots

The following result is taken from [79]. Corollary 11.2 is a direct consequence.

Lemma 11.2 ([79]). *Given a univariate polynomial* $f(x) = a_0x^m + a_1x^{m-1} + \cdots + a_m$ *of* $\mathbb{R}[x]$*, where* $a_0 \neq 0$*. We write* $f(x)$ *into the form:* $f(x) = f_1(x^2) + xf_2(x^2)$*. Then* $f(x)$ *has a pair of symmetric zeros* z *and* $-z$ *in* $\mathbb{C}$ *if and only if* z^2 *is a common zero of* $f_1(y)$ *and* $f_2(y)$*.*

Corollary 11.2. *Assume that* $a_m \neq 0$*, then* $f(x)$ *has a pair of symmetric zeros* z *and* $-z$ *in* $\mathbb{C}$ *if and only if* z^2 *is a common zero of* $f_1(y)$ *and* $yf_2(y)$*.*

Theorem 11.6 ([79]). *Let* $f(x) = a_0x^m + a_1x^{m-1} + \cdots + a_m \in \mathbb{R}[x]$ *be a polynomial of degree* m*. Then* $f(x)$ *has exactly* k *pairs of symmetric roots* z_i *and* $-z_i$ *in* $\mathbb{C}$ *if and only if* $\Delta_{m-1} = 0, \ldots, \Delta_{m-2k+1} = 0, \Delta_{m-2k-1} \neq 0$*.*

Theorem 11.7. *Notation as above, if* $a_m \neq 0$*, then* f *has exactly* k *pairs of symmetric roots* z_i *and* $-z_i$ *if and only if* $\Delta_m = 0, \ldots, \Delta_{m-2k+2} = 0, \Delta_{m-2k} \neq 0$*.*

Proof. If $a_m \neq 0$, by Corollary 11.2, the number of symmetric roots, counted with multiplicities, of the polynomial f is equal to the number of common roots, counted with multiplicities, of the two polynomials $f_1(y)$ and $yf_2(y)$. According to the elementary properties of subresultant sequences the polynomials $f_1(y)$ and $f_2(y)$ have k common roots if and only if

$$s_0(f_1, yf_2, y) = 0, \ldots, s_{k-1}(f_1, yf_2, y) = 0, s_k(f_1, yf_2, y) \neq 0.$$

So by Theorem 11.5 and specialization property of subresultants presented in Chapter 3, f has exactly k pairs of symmetric roots if and only if $\Delta_m = 0, \ldots, \Delta_{m-2k+2} = 0, \Delta_{m-2k} \neq 0$. □

Lemma 11.3 ([79]). *Let* $f(x) \in \mathbb{R}[x]$ *be a polynomial of degree* m *and* $z_1, \ldots, z_k$ *be arbitrary complex numbers. Let* $f^*(x) = f(x)(x^2 - z_1^2)\cdots(x^2 - z_k^2)$*. If* Δ_i^* *is the Hurwitz determinants of order* i *of the polynomial* $f^*(x)$*, then* $\Delta_i = \Delta_i^*$*, for* $i = 1, \ldots, m$*. Similarly, let* $f^*(x) = f(x)x^k$*, then we also have* $\Delta_i = \Delta_i^*$ *hold.*

Theorem 11.8. *The polynomial* $f(x)$ *has zero as root of multiplicity* k *and all the other roots in the left half-plane if and only if* $a_{m-k+1} = \cdots = a_m = 0$ *and* $\Delta_1 > 0, \Delta_2 > 0, \ldots, \Delta_{m-k} > 0$*.*

Proof. It follows directly from Routh-Hurwitz criterion and Lemma 11.3. □

Theorem 11.9. *Let $f(x) \in \mathbb{R}[x]$ be a polynomial of degree m and $f(x) = a_0x^m + a_1x^{m-1} + \cdots + a_m = f_1(x^2) + xf_2(x^2)$. Let $\Delta_1, \Delta_2, \ldots, \Delta_m$ be the Hurwitz determinants sequence of f. Then the following statements are equivalent:*

(i) *$f(x)$ has k pairs of pure imaginary roots and all the other roots are in the left half-plane.*

(ii) *$S_k(f_1, f_2, y)$ has k negative real roots and $\Delta_{m-1} = \Delta_{m-3} = \cdots = \Delta_{m-2k+1} = 0$, $\Delta_{m-2k} > 0, \Delta_{m-2k-1} > 0, \ldots, \Delta_1 > 0$.*

(iii) *$S_k(f_1, yf_2, y)$ has k negative real roots and $a_m \neq 0, \Delta_m = \Delta_{m-2} = \cdots = \Delta_{m-2k+2} = 0$, $\Delta_{m-2k} > 0, \Delta_{m-2k-1} > 0, \ldots, \Delta_1 > 0$.*

Proof. "(*i*) $\Rightarrow$ (*ii*)". Assume that $f(x)$ has k pairs of pure imaginary roots and all the other roots are in the left half-plane. Let $\pm i\omega_1, \ldots, \pm i\omega_k$ be the k pairs of pure imaginary roots, then we can write $f(x)$ as $f(x) = f^*(x)(x^2 + \omega_1^2) \cdots (x^2 + \omega_k^2)$, where $\omega_1^2 > 0, \ldots, \omega_k^2 > 0$ and $f^*(x)$ has only roots in the left half-plane. By Routh-Hurwitz criterion, we know that $\Delta_1^* > 0, \Delta_2^* > 0, \ldots, \Delta_{m-2k}^* > 0$. According to the Lemma 11.3, we know that $\Delta_i^* = \Delta_i$. Therefore, we have $\Delta_{m-2k} > 0, \Delta_{m-2k-1} > 0, \ldots, \Delta_1 > 0$ hold.

Moreover, by assumption we know the k pairs of pure imaginary roots are the only symmetric roots of $f(x)$, which implies $\Delta_{m-1} = \Delta_{m-3} = \cdots = \Delta_{m-2k+1} = 0, \Delta_{m-2k-1} \neq 0$. Therefore, by Theorem 11.5 we have $s_0(f_1, f_2, y) = 0, \ldots, s_{k-1}(f_1, f_2, y) = 0, s_k(f_1, f_2, y) \neq 0$, which implies that $S_k(f_1, f_2, y) = \gcd(f_1, f_2, y)$. On the other hand, since $\pm i\omega_1, \ldots, \pm i\omega_k$ are the symmetric roots of $f(x)$, by Lemma 11.2, $-\omega_1^2, \ldots, -\omega_k^2$ are the common roots of $f_1(y)$ and $f_2(y)$, that is, they are the real roots of $S_k(f_1, f_2, y)$. Therefore $S_k(f_1, f_2, y)$ has k negative real roots.

"(*ii*) $\Rightarrow$ (*i*)" By the assumption, we have $\Delta_{m-1} = \Delta_{m-3} = \cdots = \Delta_{m-2k+1} = 0, \Delta_{m-2k-1} \neq 0$, which implies that

$$s_0(f_1, f_2, y) = s_1(f_1, f_2, y) = \cdots = s_{k-1}(f_1, f_2, y) = 0, s_k(f_1, f_2, y) \neq 0.$$

Therefore the degree of $S_k(f_1, f_2, y)$ is k and $S_k(f_1, f_2, y) = \gcd(f_1, f_2, y)$. Since $S_k(f_1, f_2, y)$ has k negative real roots, we know that $f_1(y)$ and $f_2(y)$ has k common negative real roots and no other common roots. So by Lemma 11.2, $f(x)$ has exactly k pairs of pure imaginary roots and no other symmetric roots. Let us write $f(x) = f^*(x)(x^2 + \omega_1^2) \cdots (x^2 + \omega_k^2)$, according to $\Delta_{m-2k} > 0, \Delta_{m-2k-1} > 0, \ldots, \Delta_1 > 0$ and Lemma 11.3, we know that all the roots of $f^*(x)$ are in the left half-plane. Therefore

$f(x)$ has k pairs of pure imaginary eigenvalues and all the other roots are in the left half-plane.

The proof of equivalence of (i) and (iii) are similar. The only difference is that during the proof we need to use Theorem 11.7 instead of Theorem 11.6 and Corollary 11.2 instead of Lemma 11.2. □

By the above theorem, we get the following corollary, which is the main theorem on Hopf bifurcation in [79, 80].

Corollary 11.3 (Theorem 4 [80])**.** *Let $f(x) \in \mathbb{R}[x]$ be a degree m polynomial and write $f(x) = a_0x^m + a_1x^{m-1} + \cdots + a_m = f_1(x^2) + xf_2(x^2)$ with $a_0 > 0$. Let $\Delta_1, \Delta_2, \ldots, \Delta_m$ be the Hurwitz determinants sequence of f. Then $f(x)$ has a pair of distinct roots, $i\omega$ and $-i\omega$, on the imaginary and all the other roots in the left half-plane if and only if $a_m > 0, \Delta_{m-1} = 0, \Delta_{m-2} > 0, \ldots, \Delta_1 > 0$.*

Proof. By the equivalence of (i) and (iii) in Theorem 11.9, we only need to prove that $a_m > 0, \Delta_{m-1} = 0, \Delta_{m-2} > 0, \ldots, \Delta_1 > 0$ if and only if $S_1(f_1, yf_2, y)$ has one negative real root and $a_m \neq 0, \Delta_m = 0, \Delta_{m-2} > 0, \ldots, \Delta_1 > 0$. By Theorem 11.5, we have $S_1(f_1, yf_2, y) = (-1)^{\frac{\ell(\ell-1)}{2}}(\Delta_{m-2}y + a_m\Delta_{m-3})$.

" $\Rightarrow$ " Since $a_m > 0, \Delta_{m-1} = 0$, we have $a_m \neq 0$ and $\Delta_m = a_m\Delta_{m-1} = 0$. Moreover, as $a_m > 0$ and $\Delta_{m-2} > 0, \Delta_{m-3} > 0$, we know that $S_1(f_1, yf_2, y)$ has one negative real root.

" $\Leftarrow$ " Since $S_1(f_1, yf_2, y)$ has one negative real root and $\Delta_{m-2} > 0, \Delta_{m-3} > 0$, we have $-\Delta_{m-2}a_m\Delta_{m-3} < 0$, which implies that $a_m > 0$. Moreover, by $\Delta_m = 0$, we have $\Delta_{m-1} = 0$.

□

Combining the result of Theorem 11.8 and Theorem 11.9, we get the answer to Problem 4. The answer to Problem 5 was first briefly mentioned in [80], which we summarize as the following Theorem.

Theorem 11.10. *Let $f(x) = a_0x^m + a_1x^{m-1} + \cdots + a_m$ be a univariate polynomial of $\mathbb{R}[x]$. Then $f(x)$ has a root 0 of multiplicity k_1 and has k_2 pairs of pure imaginary roots while no other roots have zero real parts if and only if the following holds:*

- *The coefficients of $f(x)$ satisfy $a_m = \cdots = a_{m-k_1+1} = 0, a_{m-k_1} \neq 0$.*
- *Denote $a_0x^{m-k_1} + a_1x^{m-k_1-1} + \cdots + a_{m-k_1} = f_1(x^2) + xf_2(x^2)$. Then there exists an integer $k \geq k_2$ such that $S_k(f_1, f_2, y)$ has k_2 negative real roots and*

$$\Delta_{m-k_1-1} = \Delta_{m-k_1-3} = \cdots = \Delta_{m-k_1-2k+1} = 0, \Delta_{m-k_1-2k-1} \neq 0.$$

Proof. It directly follows from Lemma 11.2, Lemma 11.3 and Theorem 11.6. □

Remark 11.3. *In the above theorem, if both $k_1 = 0$ and $k_2 = 0$, then we get an answer to Problem $3'$. If $k_1 = 0$ and $k_2 = 1$, then we get the necessary and sufficient condition on Hopf bifurcation.*

The reader may notice that in [79, 80] there is also a theorem to provide sufficient and necessary conditions on Hopf bifurcation. More precisely, it is Theorem 3.5 in [79] and Theorem 3 in [80]. However, we find that (also noticed by the author) the condition provided there is only a sufficient condition.

In Theorem 11.10, we need to determine when a univariate polynomial S_k of degree k with parametric coefficients has k_2, $0 < k_2 \leq k$, negative real zeros. This problem can be reduced to an exhaustive case discussion on the signs of polynomials whose variables are the coefficients of S_k, by Sturm-Habicht sequence [69] or negative root discriminant sequence [137].

In Theorem 11.9, rather we want to determine when all the complex roots of a univariate polynomial with parametric real coefficients are real and negative. In the rest of this section, we provide a relatively simple answer by virtue of Descartes criterion and discriminant sequence [137, 138].

Lemma 11.4 (Descartes criterion)**.** *Let $f(x) \in \mathbb{R}[x]$ be a polynomial of degree n. Let ν be the number of sign variations of its coefficients sequence. Then there exists $m \geq 0$ such that the number of positive real roots of $f(x)$ equals $\nu - 2m$.*

Corollary 11.4. *Let $f(x) = a_0x^n + \cdots + a_{n-1}x + a_n$ be a polynomial of degree n. If $f(x)$ has n negative real roots, then we have $a_ia_{i+1} > 0$ for all $0 \leq i \leq n-1$.*

Proof. Since $f(x)$ has n negative real roots, $f(-x)$ has n positive real roots. By Descartes criterion, we have $a_i \neq 0$. On the other hand, since $f(x)$ has no positive real roots, we know that a_i have the same sign. Done. □

Definition 11.2 (Discrimination matrix)**.** *Given a polynomial with general symbolic coefficients, $f(x) = a_0x^n + a_1x^{n-1} + \cdots + a_n$, the following $2n \times 2n$ matrix in terms* *of the coefficients,*

$$\begin{bmatrix} a_0 & a_1 & a_2 & \cdots & a_n & & & & \\ 0 & na_0 & (n-1)a_1 & \cdots & a_{n-1} & & & & \\ 0 & a_0 & a_1 & \cdots & a_{n-1} & a_n & & & \\ 0 & 0 & na_0 & \cdots & 2a_{n-2} & a_{n-1} & & & \\ & & & \cdots & \cdots & & & & \\ & & & \cdots & \cdots & & & & \\ & & & a_0 & a_1 & a_2 & \cdots & a_n \\ & & & 0 & na_0 & (n-1)a_1 & \cdots & a_{n-1} \end{bmatrix}$$

is called the discrimination matrix of $f(x)$, *and denoted by* $Discr(f)$. *By* d_k *or* $d_k(f)$ *denote the determinant of the submatrix of* $Discr(f)$, *formed by the first* k *rows and the first* k *columns for* $k = 1, 2, \ldots, 2n$.

Definition 11.3 (Discriminant sequence)**.** *Let* $D_k = d_{2k}, k = 1, \ldots, n$. *We call the sequence* $[D_1, D_2, \ldots, D_n]$ *the discriminant sequence of* $f(x)$, *and denote it by* $DiscrList(f)$. *The last term* D_n *is just the discriminant of* f.

Definition 11.4 (Sign list)**.** *We call the list* $[sign(A_1), sign(A_2), \ldots, sign(A_n)]$ *the sign list of a given sequence* $A_1, A_2, \ldots, A_n$, *where*

$$sign(A_i) = \begin{cases} 1, & A_i > 0 \\ 0, & A_i = 0 \\ -1, & A_i < 0 \end{cases}$$

Definition 11.5 (Revised sign list)**.** *Given a sign list* $[s_1, s_2, \ldots, s_n]$, *we construct a new list* $[t_1, t_2, \ldots, t_n]$ *as follows: (which is called the revised sign list)*

- *If* $[s_i, s_{i+1}, \ldots, s_{i+j}]$ *is a section of the given list, where* $s_i \neq 0, s_{i+1} = \cdots = s_{i+j-1} = 0, s_{i+j} \neq 0$, *then, we replace the subsection* $[s_{i+1}, \ldots, s_{i+j-1}]$ *by the first* $j-1$ *terms of* $[-s_i, -s_i, s_i, s_i, -s_i, -s_i, s_i, s_i, \ldots]$.

- *Otherwise, let* $t_k = s_k$, *i.e. no changes for other terms.*

Theorem 11.11. *Given a polynomial* $f(x) = a_0x^n + a_1x^{n-1} + \cdots + a_n$, *where* $a_0 \neq 0$ *of* $\mathbb{R}[x]$. *If the number of sign changes of the revised sign list of* $D_1, D_2, \ldots, D_n$ *is* ν, *the number of non-vanishing members of the revised sign list is* l, *then we have: the number of distinct real roots of* $f(x)$ *equals* $l - 2\nu$; *the number of distinct pairs of conjugate imaginary roots of* $f(x)$ *is* ν.

Example 11.1. *Let* $f = (x-1)(x^2+1)$*, whose discriminant sequence is* $[3, -4, -16]$*. The sign list of it is:* $[1, -1, 1]$*. Its revised is the same to the sign list. So the number of distinct real roots of* f *is* $3 - 2 = 1$*.*

Theorem 11.12. *Let* $f(x) \in \mathbb{R}[x]$ *be a polynomial of degree* n *and* $[D_1, D_2, \ldots, D_n]$ *be its discriminant sequence. Then* $f(x)$ *has* n *negative real roots if and only if all its coefficients have the same nonzero sign and there exists* k*,* $1 \leq k \leq n$*, such that* $\forall i \leq k$*,* $D_i > 0$ *and for other* i*, we have* $D_i = 0$*.*

Proof. " $\Rightarrow$ " By Corollary 11.4, we know that all the coefficients of $f(x)$ have the same nonzero sign. On the other hand, since $f(x)$ has no imaginary real roots, the revised sign list of $[D_1, D_2, \ldots, D_n]$ has no sign changes according to Theorem 11.11. By the rule on constructing the revised sign list, we conclude that there exists k, $1 \leq k \leq n$, such that $\forall i \leq k$, $D_i > 0$ and for all $i > k$, $D_i = 0$.

" $\Leftarrow$ " If there exists k, $1 \leq k \leq n$, such that $\forall i \leq k$, $D_i > 0$ and for other i, we have $D_i = 0$. Then the revised sign list will look like this: $[1, \ldots, 1, 0, \ldots, 0]$ Therefore, the number of sign changes is 0. So $f(x)$ have no imaginary roots. Moreover, since the coefficients sequence of $f(x)$ has 0 sign variations, we know immediately that $f(x)$ has n negative real roots by Descartes Criterion. □

11.3 Stability of hyperbolic equilibria in view of bifurcation

In Section 11.2, we discussed the stability of a hyperbolic equilibria for a fixed parameter value. In this section, we study the stability of a hyperbolic equilibria under variation of parameters.

Definition 11.6 ([83])**.** *Let us consider a dynamical system that depends on parameters. The appearance of a topologically nonequivalent phase portrait under variation of parameters is called a bifurcation.*

Lemma 11.5 ([83])**.** *Given two hyperbolic equilibria of dynamical system (11.1), the phase portraits of system (11.1) near them are locally topologically equivalent if and only if at the two equilibria the Jacobian matrix* J *has the same number of eigenvalues with negative (positive) real parts.*

Theorem 11.13 (Boundary crossing theorem)**.** *Given a parameter value* α_0 *of the dynamical system (11.1) and let* β_0 *be a hyperbolic equilibrium of system (11.1) at* *the parameter α_0. Then there exists a continuous function $y(u)$ defined in a small neighborhood $O(\alpha_0)$ of α_0 satisfying $F(u, y(u)) = 0, y(\alpha_0) = \beta_0$. Moreover, the defining domain $O(\alpha_0)$ of $y(u)$ can be extended as long as $\Delta_m(u, y(u)) \neq 0$. In addition, inside the extended domain, there will be no bifurcation. In particular, the stability of $y(u)$ remains the same in the extended domain.*

Proof. Since β_0 is a hyperbolic equilibrium of system (11.1), we have $\Delta_m(\alpha_0, \beta_0) = (-1)^m \Delta_{m-1}(\alpha_0, \beta_0)\mathrm{Det}(J)(\alpha_0, \beta_0) \neq 0$. Since $\mathrm{Det}(J)(\alpha_0, \beta_0) \neq 0$, by the implicit function Theorem, we know that in a neighborhood of α_0, there is one and only one continuous function $y(u)$ defined by $F(u, y(u)) = 0$ such that $y(\alpha_0) = \beta_0$. Moreover, we can extend the domain of the function $y(u)$ if only $\mathrm{Det}(J)(u, y(u)) \neq 0$. On the other hand, the real parts of the eigenvalues of $J(u, y(u))$ will not become zero, which implies that the number of the eigenvalues of $J(u, y(u))$ with negative real parts and positive real parts will remain the same, respectively. By Lemma 11.5, the phase portraits will remain locally topologically equivalent. Therefore, the stability will not change if only $\Delta_n(u, y(u)) \neq 0$. □

Remark 11.4. *In 1929, Frazer and Duncan published a paper entitled "On the Criteria for the Stability of Small Motions" [58]. In that paper, the authors presented a theorem with the same name as above one, where they pointed out that when the system passes from a region of stability to the border of stability, Δ_n changes from positive to zero. Here by the language of bifurcation, we see that a dynamical system will keep structurally stable if only the parameter does not cross the boundary described by $\Delta_n = 0$.*

11.4 Conclusion

Based on the notion of a comprehensive triangular decomposition (CTD) presented in the last section, we have obtained a framework for analyzing the stability of the equilibria and compute the bifurcations of polynomial dynamical systems. Indeed, we can completely solve the problems introduced in Section 11.1.

Let us first have a look at Problem 1. Let $F(\mathbf{u}, \mathbf{x})$ be the right hand side polynomial equations of the dynamical system (11.1). It is usually required that $\mathbf{u}$ and $\mathbf{x}$ are both positive. Let $P(\mathbf{u}, \mathbf{x})$ be the corresponding set of positive inequality constraints. Let $(\mathcal{C}, (\mathcal{A}_C, C \in \mathcal{C}))$ be a CTD of $\mathcal{S} = [F, P_>]$. In the practice of dynamical systems, only the cells above which $\mathcal{S}$ has finitely many complex solutions are interesting. This fact has motivated our definition of the CTD of a semi-algebraic system. Let $C \in \mathcal{C}$

be a cell above which $\mathcal{S}$ has finitely many complex solutions, one of them at least being real, that is, a cell of type (*iii*) in Definition 10.1. The set C is a connected semi-algebraic subsets of $\mathbb{R}^d$, above which $\mathcal{A}_C$ is a finite set of SFSASes whose solutions are disjoint graphs of continuous functions above C; moreover the union of the graphs of these functions is exactly $C \cap Z_{\mathbb{R}}(\mathcal{S})$. Therefore, Problem 1 is solved.

Next, we look at Problem 2. A first and direct approach consists of computing a CTD of the system $\mathcal{S}$ augmented with the inequalities $\Delta_i > 0$, $1 \leq i \leq m$, where the Δ_i are the Hurwitz determinants, see Definition 11.1. A second approach consists of computing a CTD of the system $\mathcal{S}$ augmented with the inequality $\Delta_m > 0$ only and then apply the Boundary Crossing Theorem, that is Theorem 11.13.

Similarly, for each of the three other problems on bifurcation, we will first produce a semi-algebraic system by means of results in Section 11.2 and then apply CTD to solve it.

Chapter 12

Conclusion

Computing the solutions of a polynomials system is a central problem in computer algebra and has many applications in other fields. Triangular decomposition is one of the main symbolic techniques for solving polynomial systems. In this thesis, we have improved both the efficiency and effectiveness of triangular decompositions.

On the efficiency front, we revisited one of the core routines for computing triangular decompositions, namely the computation of regular GCD modulo a regular chain. We proposed a weakened usage of the concept of regular GCD, based on which a simpler and more efficient triangular decomposition algorithm was obtained. This new algorithm is structured to recycle expensive operations, such as the computation of subresultant chains, as much as possible. The experimentation shows that this new triangular decomposition algorithm outperforms solvers with similar specifications by several orders of magnitude.

On the effectiveness front, we have greatly extended the scope of usage of triangular decompositions. Before our work, triangular decompositions were mainly used for computing the complex solutions of polynomial systems. In this thesis, we introduce the concept of comprehensive triangular decomposition, which is dedicated to computing the solutions of polynomial systems depending on parameters. Moreover, we adapt the concept of regular chain and triangular decomposition to semi-algebraic systems and provide very useful tools for describing the real solutions of polynomial systems. We have also connected triangular decomposition with cylindrical algebraic decomposition (CAD), which is one of the fundamental tools in real algebraic geometry. Our new approach for computing CAD brings new insight into this field.

We have successfully applied our tools for several applications. Among them, the study of equilibria of dynamical systems actually motivated the work in this thesis. The work presented in this thesis brings new challenges and opportunities for triangular decompositions. We conclude this dissertation with three open problems.

- Better control of expression swell when computing triangular decompositions.
- Define and compute a notion of *canonical and minimal comprehensive triangular decomposition*.
- Define and compute a notion of *canonical and minimal cylindrical algebraic decomposition*.

Appendix A

Commutative Ring and Ideal theory

In this chapter, we introduce some useful mathematical concepts and results related to this thesis. The first three sections describe basic concepts and classical results for general rings. The main reference we rely on is the book "Introduction to commutative algebra" by M.F. Atiyah and I.G. Macdonald. The next two sections states fundamental results on polynomial ideals and varieties. The main reference is the book "Ideals, varieties, and algorithms" by D. Cox, J. Little and D. O'Shea.

A.1 Commutative ring

Let $\mathbb{A}$ and $\mathbb{B}$ be two sets. We denote by $\mathbb{A}\times\mathbb{B}$ the set of all pairs $\{(a,b) \mid a \in \mathbb{A}, b \in \mathbb{B}\}$, which is called the *direct product* of $\mathbb{A}$ and $\mathbb{B}$. Given a set $\mathbb{A}$, we define a function $+ : \mathbb{A} \times \mathbb{A} \to \mathbb{A}$, called addition operation, such that

(1) for all $a, b, c \in \mathbb{A}$, $+$ is associative, that is $(a + b) + c = a + (b + c)$,

(2) there exists an element of $\mathbb{A}$, denoted by 0, such that for any $a \in \mathbb{A}$, we have $a + 0 = 0 + a = a$,

(3) for any element $a \in \mathbb{A}$, there exists another element $b \in \mathbb{A}$, such that $a + b = b + a = 0$.

We call $(\mathbb{A}, +)$ a *group*. If in addition, $+$ is commutative, that is for any $a, b \in \mathbb{A}$, we have $a + b = b + a$, then $(\mathbb{A}, +)$ is called an *abelian group*.

Let $(\mathbb{A}, +)$ be an abelian group, we define another function $\cdot : \mathbb{A} \times \mathbb{A} \to \mathbb{A}$, called multiplication operation, such that

(1) for all $a, b, c \in \mathbb{A}$, $\cdot$ is associative, that is $(a \cdot b) \cdot c = a \cdot (b \cdot c)$,

(2) $\cdot$ is distributive w.r.t. $+$, that is for any $a, b, c \in \mathbb{A}$, we have $(a+b) \cdot c = a \cdot c + b \cdot c$ and $c \cdot (a + b) = c \cdot a + c \cdot b$ hold.

We call $(\mathbb{A}, +, \cdot)$ a *ring*. If in addition, we have

(3) $\cdot$ is commutative, that is for any $a, b \in \mathbb{A}$, we have $a \cdot b = b \cdot a$,

(4) there exists an element of $\mathbb{A}$, denoted by 1, such that for any $a \in \mathbb{A}$, $a \cdot 1 = 1 \cdot a = a$,

then we call $(\mathbb{A}, +, \cdot)$ a *commutative ring with identity.* **In this thesis, ring shall always mean a commutative ring with identity.** When context is clear, we also write ring $(\mathbb{A}, +, \cdot)$ as $\mathbb{A}$ for short. For any two elements $a, b \in \mathbb{A}$, $a \cdot b$ is also written as ab.

Example A.1. *All the integers* $\{\ldots, -2, -1, 0, 1, 2, \ldots\}$ *forms a ring w.r.t. integer additions and multiplications, usually denoted by* $\mathbb{Z}$. *The set of natural numbers* $\mathbb{N} = \{0, 1, 2, \ldots\}$ *is not a ring since it is not a group w.r.t. number additions.*

Let $\mathbb{A}$ and $\mathbb{B}$ be two rings. A ring *homomorphism* is a function $f : \mathbb{A} \to \mathbb{B}$ such that

(*i*) $f(a + b) = f(a) + f(b)$,

(*ii*) $f(ab) = f(a)f(b)$,

(*iii*) $f(1) = 1$.

A subset S of $\mathbb{A}$ is called a *subring* of $\mathbb{A}$ if S is closed under addition and multiplication and contains the identity element of $\mathbb{A}$.

An element $x \neq 0 \in \mathbb{A}$ is called a *zero-divisor* in $\mathbb{A}$ if there exists $y \neq 0 \in \mathbb{A}$ such that $xy = 0$. A ring with no zero-divisors is called an *integral domain.* An element $x \in \mathbb{A}$ is called a *nilpotent* if there exists $n > 0$ such that $x^n = 0$. An element x is called *regular* in $\mathbb{A}$ if x is neither zero nor zero-divisor in $\mathbb{A}$. An element x of $\mathbb{A}$ is called a *unit* if there exists y such that $xy = 1$. A *field* is a ring $\mathbb{A}$ in which $1 \neq 0$ and every non-zero element of $\mathbb{A}$ is a unit.

Example A.2. *Let* $\mathbb{Z}$ *be the set of integers. Let* m *be a positive integer. Let* $\mathbb{Z}/m\mathbb{Z} := \{0, 1, \ldots, m - 1\}$. *We define additions and multiplications on* $\mathbb{Z}/m$ *as follows: for any* $x, y \in \mathbb{Z}/m$, $x + y = (x +_{\mathbb{Z}} y) \bmod m$ *and* $x \cdot y = (x *_{\mathbb{Z}} y) \bmod m$. *Here* $x +_{\mathbb{Z}} y$

and $x *_{\mathbb{Z}} y$ *denote respectively adding and multiplying* x *and* y *as usual integers. It is easy to verify that* $\mathbb{Z}/m\mathbb{Z}$ *is a ring.*

Then in $\mathbb{Z}/4$*, the element* 2 *is a zero-divisor and also a nilpotent; the element* 3 *is regular and also a unit. If* m *is a prime number, say* 3*, then* $\mathbb{Z}/m\mathbb{Z}$ *is a field.*

A.2 Ideals

An *ideal* $\mathcal{I}$ of $\mathbb{A}$ is a subset of $\mathbb{A}$ which is an abelian group w.r.t. $+$ and such that: for any $x \in \mathbb{A}$ and $y \in \mathcal{I}$, we have $xy \in \mathcal{I}$.

Given a ring $\mathbb{A}$ and an ideal $\mathcal{I}$ of $\mathbb{A}$, we can define an equivalence relation (meaning reflexivity, symmetry and transitivity) $\sim$ on $\mathbb{A}$ as follows: two elements a, b of $\mathbb{A}$ are equivalent, denoted by $a \sim b$ if and only if $a - b \in \mathcal{I}$. We say that a and b are congruent modulo $\mathcal{I}$. The equivalent class of a in $\mathbb{A}$, denoted by $[a]$, is set of all elements equivalent to a. Clearly $[a] = a + \mathcal{I}$.

The set of all equivalent classes is denoted by $\mathbb{A}/\mathcal{I}$. One can defined two operations $+$ and $\cdot$ on $\mathbb{A}/\mathcal{I}$ as follows: $[a] + [b] = [a + b]$ and $[a] \cdot [b] = [a \cdot b]$. One can prove the two operations are well defined and $\mathbb{A}/\mathcal{I}$ forms a ring, called a *quotient ring*, under the two operations.

An ideal $\mathfrak{p}$ in $\mathbb{A}$ is called *prime* if $\mathfrak{p} \neq \mathbb{A}$ and for any $x, y \in \mathbb{A}$, if $xy \in \mathfrak{p}$, then either $x \in \mathfrak{p}$ or $y \in \mathfrak{p}$. An ideal is called *maximal* if $\mathfrak{m} \neq \mathbb{A}$ and if there is no ideal $\mathcal{I}$ such that $\mathfrak{m} \subsetneq \mathcal{I} \subsetneq \mathbb{A}$.

Proposition A.1. $\mathcal{I}$ *is a prime ideal if and only if* $\mathbb{A}/\mathcal{I}$ *is an integral domain.* $\mathcal{I}$ *is a maximal ideal if and only if* $\mathbb{A}/\mathcal{I}$ *is a field.*

Let $\mathcal{I}$ and $\mathcal{J}$ be two ideals of $\mathbb{A}$. Define the *sum* of $\mathcal{I}$ and $\mathcal{J}$ as $\mathcal{I} + \mathcal{J} := \{x + y \mid x \in \mathcal{I}, y \in \mathcal{J}\}$, which is an ideal of $\mathbb{A}$. Define the *intersection* of $\mathcal{I}$ and $\mathcal{J}$ as $\mathcal{I} \cap \mathcal{J} := \{x \mid x \in \mathcal{I} \text{ and } x \in \mathcal{J}\}$, which is an ideal of $\mathbb{A}$. Define the *product* of $\mathcal{I}$ and $\mathcal{J}$ as

$$\mathcal{I}\mathcal{J} := \{\sum_{i=1}^{r} x_i y_i \mid x_i \in \mathcal{I} \text{ and } y_i \in \mathcal{J}, r > 0\},$$

which is an ideal of $\mathbb{A}$. The union of ideals is generally not an ideal. Define the *ideal quotient* of $\mathcal{I}$ and $\mathcal{J}$ as $\mathcal{I} : \mathcal{J} = \{x \mid xy \in \mathcal{I}, \text{ for all } y \in \mathcal{J}\}$. Two ideals are said to be *coprime* if $\mathcal{I} + \mathcal{J} = \mathbb{A}$. For coprime deals, we have $\mathcal{I} + \mathcal{J} = \mathcal{I}\mathcal{J}$. Define the *radical* of $\mathcal{I}$ as $\sqrt{\mathcal{I}} := \{x \mid x^n \in \mathcal{I} \text{ for some } n > 0\}$. An ideal $\mathcal{I}$ is called a *radical ideal* if $\mathcal{I} = \sqrt{\mathcal{I}}$. Let $h \in \mathbb{A}$ The *saturated ideal* of $\mathcal{I}$ w.r.t. h, denoted by $\mathcal{I} : h^{\infty}$, is the ideal $\{q \in \mathbb{A} \mid \exists m \in \mathbb{N} \text{ s.t. } h^m q \in \mathcal{I}\}$.

Proposition A.2. *The following are some useful properties of operations on ideals.*

- $\sqrt{\cap_{i=1}^r \mathcal{I}_i} = \cap_{i=1}^r \sqrt{\mathcal{I}_i}$
- $(\cap_{i=1}^r \mathcal{I}_i) : \mathcal{J} = \cap_{i=1}^r (\mathcal{I}_i : \mathcal{J})$

Proposition A.3. (i) *Let* $\mathfrak{p}_1, \ldots, p_r$ *be prime ideals and let* $\mathcal{I}$ *be an ideal contained in* $\cup_{i=1}^r \mathfrak{p}_i$. *Then* $\mathcal{I} \subseteq \mathfrak{p}_i$ *for some* i. (ii) *Let* $\mathcal{I}_1, \ldots, \mathcal{I}_s$ *be ideals and let* $\mathfrak{p}$ *be a prime ideal containing* $\cap_{i=1}^r \mathcal{I}_i$. *Then* $\mathfrak{p} \supseteq \mathcal{I}_i$ *for some* i.

Let $\mathbb{A}$ be any ring. A *multiplicatively closed subset* of $\mathbb{A}$ is a subset S of $\mathbb{A}$ such that $1 \in S$ and S is closed under multiplication. Define a relation on $\mathbb{A} \times S$ as follows: $(a, s) \sim (b, t)$ if and only if $(at - bs)u = 0$ for some $u \in S$. One can verification that this relation is an equivalence relation. Let a/s denote the equivalent class of (a, s), and let $S^{-1}\mathbb{A}$ denote the set of equivalence classes. We define addition and multiplication on $S^{-1}\mathbb{A}$ respectively as $(a/s) + (b/t) = (at + bs)/st$ and $(a/s)(b/t) = ab/st$. One can verify that the two operations are well defined and $S^{-1}\mathbb{A}$ forms a commutative ring under the two operations. We also have a natural ring homomorphism $f : \mathbb{A} \to S^{-1}\mathbb{A}$ defined by $f(x) = x/1$. The ring $S^{-1}\mathbb{A}$ is called the ring of fractions of $\mathbb{A}$ w.r.t. S. If $\mathbb{A}$ is an integral domain and $S = A - 0$, then $S^{-1}\mathbb{A}$ is a field and is called the *field of fractions* of $\mathbb{A}$.

A.3 Noetherian rings and primary decompositions

Let $\mathcal{I}$ be an ideal in $\mathbb{A}$. We say $\mathcal{I}$ is finitely generated if there exists finitely many elements in $\mathcal{I}$, say $x_1, \ldots, x_r$ such that $\mathcal{I} = \sum_{i=1}^r \langle x_i \rangle$. A ring $\mathbb{A}$ is said to be Noetherian if every ideal in $\mathbb{A}$ is finitely generated. Let $\mathcal{I}_1 \subseteq \mathcal{I}_2 \subseteq \cdots$ be an ascending chain of ideals. It is said stationary if there exists n such that $\mathcal{I}_n = \mathcal{I}_{n+1} = \cdots$.

Proposition A.4. *The following statements are equivalent.*

- $\mathbb{A}$ *is Noetherian,*
- *every ascending chain of ideals in* $\mathbb{A}$ *is stationary,*
- *every nonempty set of ideals in* $\mathbb{A}$ *has a maximal element.*

Let $\mathbb{A}$ be a ring. An ideal $\mathfrak{q}$ in $\mathbb{A}$ is called primary if $\mathfrak{q} \neq \mathbb{A}$ and for any $x, y \in \mathfrak{q}$, if $xy \in \mathfrak{q}$, then either $x \in \mathfrak{q}$ or $y^n \in \mathfrak{q}$.

Proposition A.5. *Let $\mathfrak{q}$ be a primary ideal in $\mathbb{A}$, then $\sqrt{q}$ is the smallest prime ideal containing $\mathfrak{q}$.*

Let $\mathfrak{p} = \sqrt{q}$. We call $\mathfrak{p}$ the associated prime ideal of $\mathfrak{q}$ and we say $\mathfrak{q}$ is $\mathfrak{p}$-primary.

Proposition A.6. *If $\mathfrak{q}_i$, $i = 1, \ldots, r$ are $\mathfrak{p}$-primary. Then $\cap_{i=1}^r \mathfrak{q}_i$ is $\mathfrak{p}$-primary.*

Let $x \in \mathbb{A}$. Denote $\langle x \rangle = \{ax \mid a \in \mathbb{A}\}$. Then $\langle x \rangle$ is an ideal in $\mathbb{A}$. Let $\mathcal{I}$ be an ideal in $\mathbb{A}$. Then the ideal quotient $\mathcal{I} : \langle x \rangle$ is simply written as $\mathcal{I} : x$.

Proposition A.7. *Let $\mathfrak{q}$ be a $\mathfrak{p}$-primary ideal and x and element of $\mathbb{A}$. Then*

(i) *if $x \in \mathfrak{q}$, then $\mathfrak{q} : x = \mathbb{A}$*

(ii) *if $x \notin \mathfrak{q}$, then $\mathfrak{q} : x$ is $\mathfrak{p}$-primary*

(iii) *if $x \notin \mathfrak{p}$, then $\mathfrak{q} : x = \mathfrak{q}$*

A *primary decomposition* of an ideal $\mathcal{I}$ in $\mathbb{A}$ is an expression of $\mathcal{I}$ as a finite intersection of primary ideals, say $\mathcal{I} = \cap_{i=1}^r \mathfrak{q}_i$, where each $\mathfrak{q}_i$ is a primary ideal in $\mathbb{A}$.

In general, for a given ideal, a primary decomposition of it may not exist. However, for Noetherian ring, a primary decomposition always exists.

Proposition A.8. *In a Noetherian $\mathbb{A}$, every proper ideal $\mathcal{I} \neq \mathbb{A}$ has a primary decomposition.*

Let $\mathbb{A}$ be a Noetherian ring and let $\mathcal{I}$ be an ideal in $\mathbb{A}$. Let $\cap_{i=1}^r \mathfrak{q}_i$ be a primary decomposition of $\mathcal{I}$. If in addition, it satisfies: (1) all $\sqrt{\mathfrak{q}_i}$ are different; (2) for any $1 \leq i \leq r$, $\cup_{j \neq i} \mathfrak{q}_j \not\subseteq \mathfrak{q}_i$. Then we say the primary decomposition $\cap_{i=1}^r \mathfrak{q}_i$ is *minimal*. By Proposition A.6, any primary decomposition of $\mathcal{I}$ can be reduced to a minimal one.

Theorem A.1. *Let $\mathbb{A}$ be a Noetherian ring Let $\mathcal{I}$ be an ideal in $\mathbb{A}$ and let $\cap_{i=1}^r \mathfrak{q}_i$ be a minimal primary decomposition of $\mathcal{I}$. Let $\mathfrak{p}_i = \sqrt{q_i}$, $i = 1, \ldots, r$. Then $\mathfrak{p}_i$ are precisely the prime ideals which appear in the set of ideals $\mathcal{I} : x$, $x \in \mathbb{A}$, and therefore are independent of a particular decomposition of $\mathcal{I}$.*

The prime ideals $\mathfrak{p}_i$ in the above theorem are called the *associated prime ideals* of $\mathcal{I}$. The ideal $\mathcal{I}$ is primary if and only if it has one associated prime ideal. The minimal elements of $\{\mathfrak{p}_1, \ldots, \mathfrak{p}_r\}$ are called the *minimal* or *isolated* prime ideals associated with $\mathcal{I}$. The others are called *embedded* prime ideals.

Proposition A.9. *Let $\mathbb{A}$ be a Noetherian ring and let $\mathcal{I}$ be an ideal in $\mathbb{A}$. Let $\mathfrak{p}_1, \dots, \mathfrak{p}_s$ be the minimal associated prime ideals with $\mathcal{I}$. Then they are the associated prime ideals of $\sqrt{\mathcal{I}}$. Moreover $\sqrt{\mathcal{I}} = \cap_{i=1}^s \mathfrak{p}_i$.*

Proof. Let $\mathcal{I} = \cap_{i=1}^r \mathfrak{q}_i$ be a minimal primary decomposition of $\mathcal{I}$. Then we have $\sqrt{\mathcal{I}} = \cap_{i=1}^r \sqrt{\mathfrak{q}_i}$ by Proposition A.2. Note that $\sqrt{\mathfrak{q}_i}$, $i = 1, \dots, r$ are the associated prime ideals with $\mathcal{I}$. We pick the minimal ones and rename them as $\mathfrak{p}_1, \dots, \mathfrak{p}_s$. Then we have $\sqrt{\mathcal{I}} = \cap_{i=1}^s \mathfrak{p}_i$. Since a prime ideal is primary, $\cap_{i=1}^s \mathfrak{p}_i$ is a minimal primary decomposition of $\sqrt{\mathcal{I}}$ and therefore $\mathfrak{p}_1, \dots, \mathfrak{p}_s$ are the associated prime ideals of $\sqrt{\mathcal{I}}$. □

Proposition A.10. *Let $\mathcal{I}$ be an ideal in a Noetherian ring $\mathbb{A}$ and assume that $\mathcal{I} \neq \mathbb{A}$. Let $p \in \mathbb{A}$. Then p is regular in $A/\mathcal{I}$ if and only if p does not belong to any associated prime ideals of $\mathcal{I}$.*

Proof. By Proposition A.1, the associated prime ideals of $\mathcal{I}$ are exactly the prime ideals which occur in the set of ideals $\mathcal{I} : x$, $x \in \mathbb{A}$.

"$\Rightarrow$" Let p be regular in $A/\mathcal{I}$. We prove by contradiction. Assume that p belongs to some associated prime ideal of $\mathcal{I}$. Then there exists $x \in \mathbb{A}$ such that $\mathcal{I} : x$ is prime and $p \in \mathcal{I} : x$, which implies that $x \notin \mathcal{I}$ and $px \in \mathcal{I}$. It is a contradiction to p is regular in $A/\mathcal{I}$.

"$\Leftarrow$" Let p be an element of $\mathbb{A}$ which does not belong to any associated prime ideals of $\mathcal{I}$. We prove by contradiction. Assume p is not regular in $A/\mathcal{I}$. Then there exists $x \notin \mathcal{I}$ such that $px \in \mathcal{I}$, which implies that $p \in \mathcal{I} : x$. Let $\mathcal{I} = \cap_{i=1}^r \mathfrak{q}_i$ be a minimal primary decomposition of $\mathcal{I}$. We have $\mathcal{I} : x = \cap_{i=1}^r (\mathfrak{q}_i : x)$ by Proposition A.2. Since $x \notin \mathcal{I}$, there exists $\mathfrak{q}_i$ such that $x \notin \mathfrak{q}_i$. Let p_i be the associated prime ideal of $\mathfrak{q}_i$, which is also an associated prime ideal of $\mathcal{I}$. By Proposition A.7, $\mathfrak{q}_i : x$ is $\mathfrak{p}_i$-primary. Hence we have $p \in \mathfrak{p}_i$, which is a contradiction to the assumption. □

Proposition A.11. *Let $\mathbb{A}$ be a Noetherian ring. Let $\mathcal{I}$ be an ideal and h be an element in $\mathbb{A}$. Then there exists an integer N such that $\mathcal{I} : h^\infty = \mathcal{I} : h^N$.*

Proof. First we have $\mathcal{I} : h^\infty = \cup_{i=0}^\infty \mathcal{I} : h^i$. Note that there exists an ascending chain in $\mathbb{A}$ such that $\mathcal{I} : h^0 \subseteq \mathcal{I} : h^1 \subseteq \cdots$. Since $\mathbb{A}$ is a Noetherian ring, there exists N such that

$$\mathcal{I} : h^0 \subseteq \mathcal{I} : h^1 \subseteq \cdots \subseteq \mathcal{I} : h^N = \mathcal{I} : h^{N+1} = \cdots ,$$

which implies that $\mathcal{I} : h^\infty \subseteq \mathcal{I} : h^N$. $\mathcal{I} : h^N \subseteq \mathcal{I} : h^\infty$ is obvious. □

Corollary A.1. *Let $\mathbb{A}$ be a Noetherian ring and let $\mathcal{I}$ be an ideal and h be an element in $\mathbb{A}$. and let $\cap_{i=1}^r \mathfrak{q}_i$ be a minimal primary decomposition of $\mathcal{I}$. Let $\mathfrak{p}_i = \sqrt{q_i}$, $i = 1, \ldots, r$. Assume for the $1 \leq i \leq s$, $h \notin \mathfrak{p}_i$ and for $s < i \leq r$, $h \in \mathfrak{p}_i$. Then we have $\mathcal{I} : h^\infty = \cap_{i=1}^s \mathfrak{q}_i$.*

Proof. By Proposition A.11, there exists integers N_0, $N_1, \ldots, N_r$, such that $\mathcal{I} : h^{N_0} = \mathcal{I} : h^\infty$ and $\mathfrak{q}_i : h^{N_i} = \mathfrak{q}_i : h^\infty$. Let $N = \max(N_0, N_1, \ldots, N_r)$. Then we have $\mathcal{I} : h^N = \cap_{i=1}^r \mathfrak{q}_i : h^N$, which implies that $\mathcal{I} : h^\infty = \cap_{i=1}^r \mathfrak{q}_i : h^\infty = \cap_{i=1}^r \mathfrak{q}_i : h^N$. Moreover, we can let N large enough such that if $h \in \mathfrak{p}_i$, then $h^N \in \mathfrak{q}_i$. Then the conclusion follows directly from Proposition A.7. □

A.4 Polynomial ideals and algebraic varieties

In this section, we state related concepts on polynomial ideals and algebraic varieties.

Let $\mathbf{k}$ be a field. We say that a field $\mathbf{k}$ is *algebraically closed* if every nonconstant polynomial in $\mathbf{k}[x]$ has a root in $\mathbf{k}$. An *algebraic closure* of $\mathbf{k}$, denoted by $\mathbf{K}$, is an algebraic extension field of $\mathbf{k}$ which is algebraically closed. Up to an isomorphism that fixes every member of $\mathbf{k}$, an algebraic closure of $\mathbf{k}$ is unique. For example, the field $\mathbb{C}$ of complex numbers is the algebraic closure of the field $\mathbb{R}$ of the real numbers.

Let $f_1, \ldots, f_s$ be polynomials in $\mathbf{k}[x_1, \ldots, x_n]$. Denote $V(f_1, \ldots, f_s) = \{(a_1, \ldots, a_n) \in \mathbf{K}^n \mid f_i(a_1, \ldots, a_n) = 0 \text{ for all } 1 \leq i \leq s\}$ and call it the *algebraic variety* defined by $f_1, \ldots, f_s$ in $\mathbf{K}^n$. Sometimes we call V a $\mathbf{k}$-algebraic variety to emphasize that this variety is defined as zero sets of polynomials with coefficients in $\mathbf{k}$. Denote by $\langle f_1, \ldots, f_s \rangle$ the ideal generated by $f_1, \ldots, f_s$ in $\mathbf{k}[x_1, \ldots, x_n]$. That is $\langle f_1, \ldots, f_s \rangle = \{\sum_{i=1}^s h_i f_i \mid h_1, \ldots, h_s \in \mathbf{k}[x_1, \ldots, x_n]\}$. Let $V \subseteq \mathbf{K}^n$ be a $\mathbf{k}$-algebraic variety. Define $\mathbf{I}(V) = \{f \in \mathbf{k}[x_1, \ldots, x_n] : f(a_1, \ldots, a_n) = 0 \text{ for all} (a_1, \ldots, a_n) \in V\}$. Note that $\mathbf{I}(V)$ is an ideal in $\mathbf{k}[x_1, \ldots, x_n]$ and we call it the ideal of V.

Theorem A.2 (Hilbert basis theorem)**.** *Every ideal $\mathcal{I} \subset \mathbf{k}[x_1, \ldots, x_n]$ has a finite generating set. That is $\mathcal{I} = \langle f_1, \ldots, f_s \rangle$ for some $f_1, \ldots, f_s \in \mathcal{I}$.*

Hilbert basis theorem shows that it makes sense to speak of the algebraic variety defined by an ideal $\mathcal{I}$. Let $\mathcal{I}$ be an ideal in $\mathbf{k}[x_1, \ldots, x_n]$. Denote by $V(\mathcal{I})$ the set $V(\mathcal{I}) = \{(a_1, \ldots, a_n) \in \mathbf{K}^n : f(a_1, \ldots, a_n) = 0 \text{ for all } f \in \mathcal{I}\}$. Let $f_1, \ldots, f_s$ be the generators of $\mathcal{I}$. Then $V(\mathcal{I}) = V(f_1, \ldots, f_s)$ and therefore is an algebraic variety.

Theorem A.3 (Hilbert's Nullstellensatz)**.** *If $\mathcal{I}$ is an ideal in $\mathbf{k}[x_1, \ldots, x_n]$, then $\mathcal{I}(V(\mathcal{I})) = \sqrt{\mathcal{I}}$.*

Corollary A.2. *Let $\mathcal{I}$ and $\mathcal{J}$ be ideals in $\mathbf{k}[x_1, \ldots, x_n]$. Then $V(\mathcal{I}) \subseteq V(\mathcal{J})$ if and only if $\sqrt{\mathcal{J}} \subseteq \sqrt{\mathcal{I}}$.*

Let S be a subset of $\mathbf{K}^n$. The set $\mathbf{I}(S) = \{f \in \mathbf{k}[x_1, \ldots, x_n] \mid f(a) = 0 \text{ for all } a \in S\}$ is an ideal in $\mathbf{k}[x_1, \ldots, x_n]$. The $\mathbf{k}$-Zariski closure of S, denote by $\overline{S}$, is defined as the smallest $\mathbf{k}$-algebraic variety containing the set, which is actually $V(\mathbf{I}(S))$.

Theorem A.4. *Let $\mathcal{I}$ and f be respectively an ideal and a polynomial in $\mathbf{k}[x_1, \ldots, x_n]$. Then we have $\overline{V(\mathcal{I}) \setminus V(f)} = V(\mathcal{I} : f^\infty)$.*

An algebraic variety $V \subset \mathbf{K}^n$ is irreducible if whenever V is written in the form $V_1 \cup V_2$, where V_1 and V_2 are algebraic varieties, then either $V_1 = V$ or $V_2 = V$.

Proposition A.12. *Let $V \subseteq \mathbf{K}^n$ be an algebraic variety. Then V is irreducible if and only if $\mathbf{I}(V)$ is a prime ideal.*

Theorem A.5. *Let $V \subseteq \mathbf{K}^n$ be an algebraic variety. Then V can be written as a finite union of irreducible varieties.*

Let $V \subseteq \mathbf{K}^n$ be an algebraic variety. A decomposition $V = V_1 \cup \cdots \cup V_m$, where each V_i is an irreducible variety, is called a *minimal decomposition* if $V_i \not\subseteq V_j$ for $i \neq j$.

Theorem A.6. *Every algebraic variety $V \subseteq \mathbf{K}^n$ has a minimal decomposition. Furthermore, this minimal decomposition is unique up to the order in which $V_1, \ldots, V_m$ are written.*

A.5 Dimension of polynomial ideals and algebraic varieties

Let $\mathbb{A} = \mathbf{k}[x_1, \ldots, x_n]$ be a polynomial ring. Let $\mathcal{I}$ be an ideal in $\mathbb{A}$. A subset of variables $\{y_1, \ldots, y_s\}$ of $\{x_1, \ldots, x_n\}$ is called algebraically dependent modulo $\mathcal{I}$ if there exists a nonzero polynomial $p(y_1, \ldots, y_s) \in \mathcal{I}$. They are called algebraically independent modulo $\mathcal{I}$ if $p(y_1, \ldots, y_s) \in \mathcal{I}$ implies that p is the zero polynomial, which is equivalent to say that $\mathcal{I} \cap \mathbf{k}[y_1, \ldots, y_s] = \{0\}$. Let $\mathcal{I}$ be a polynomial ideal in $\mathbf{k}[x_1, \ldots, x_n]$. The *dimension* of $\mathcal{I}$, denoted as $\dim \mathcal{I}$, is defined to be the cardinality of a largest subset of X which is independent modulo $\mathcal{I}$. If there are no independent subsets at all (which only happens when $\mathcal{I} = \mathbf{k}[X]$, then the affine dimension of $\mathcal{I}$ is defined to be -1. The *co-dimension* or *height* of $\mathcal{I}$ is defined as $n - \dim \mathcal{I}$.

Let V be an algebraic variety of $\mathbf{K}^n$. We define $\dim V = \dim \mathbf{I}(V)$. An ideal $\mathcal{I}$ in $\mathbf{k}[X]$ is called *unmixed* if the dimensions of all its associated prime ideals are the same. $\mathcal{I}$ is said to be *equidimensional* if the dimensions of all its associated minimal prime ideals are the same. Clearly, if an ideal is unmixed, it has no embedded prime ideals.

Let $\mathbb{A}$ be a ring. The *Krull dimension* of $\mathbb{A}$, named after Wolfgang Krull (1899-1971), is defined as the supremum of the number of strict inclusions in a chain of prime ideals. The following proposition suggests another equivalent definition on the dimension of an ideal.

Proposition A.13. *The* dimension *of* $\mathcal{I}$ *is the Krull dimension of* $\mathbb{A}/\mathcal{I}$.

Proposition A.14. *Let* $\mathbb{A} = \mathbf{k}[x_1, \ldots, x_n]$. *Let* $\mathfrak{p}_1 \subsetneq \mathfrak{p}_2$ *be two prime ideals in* $\mathbb{A}$. *Then* $\dim(\mathfrak{p}_2) < \dim(\mathfrak{p}_1)$.

Proof. Clearly a longest strict chain of inclusions of prime ideals containing $\mathfrak{p}_2$ is shorter than the one containing $\mathfrak{p}_1$. Then the conclusion follows directly from the definition of dimension of an ideal. □

Proposition A.15. *Let* $\mathbb{A} = \mathbf{k}[x_1, \ldots, x_n]$. *Let* $p \in \mathbb{A}$ *and let* $\mathcal{I}$ *be an ideal in* $\mathbb{A}$. *If* $\mathcal{I}$ *is unmixed, then* p *is regular in* $\mathbb{A}/\mathcal{I}$ *if and only if* p *is regular in* $\mathbb{A}/\sqrt{\mathcal{I}}$.

Proof. Since $\mathcal{I}$ is unmixed, the associated prime ideals of $\mathcal{I}$ can not be strictly contained in each other by Proposition A.14. Therefore they are all minimal. By Proposition A.9, they are exactly the associated prime ideals of $\sqrt{\mathcal{I}}$. The the conclusion follows immediately from Proposition A.10. □

Lemma A.1. *Let* $\mathcal{I}$ *be a proper ideal in* $\mathbf{k}[x_1, \ldots, x_n]$ *and* $f \in \mathbf{k}[x_1, \ldots, x_n]$ *be a polynomial regular modulo* $\mathcal{I}$. *Then, we have:* $\dim(V(\mathcal{I}) \cap V(f)) < \dim(V(\mathcal{I})) - 1$.

Proof. Let $\mathfrak{p}_1, \ldots, \mathfrak{p}_e$ be the associated prime ideal of $\sqrt{\mathcal{I}}$. We have $V(\mathcal{I}) \cap V(f) = \cup_{i=1}^{e} V(\mathfrak{p}_i) \cap V(f)$. Thus, it is enough to show for any associated prime ideal $\mathfrak{p}$ of $\sqrt{\mathcal{I}}$, we have $\dim(\langle \mathfrak{p} + f \rangle) < \dim(\mathfrak{p})$. Since f is regular modulo $\mathcal{I}$, we have $\mathfrak{p} \subsetneq \langle \mathfrak{p} + f \rangle$. Thus $\mathfrak{p}$ is strictly contained in any prime ideal of $\langle \mathfrak{p} + f \rangle$. By Proposition A.14, we deduce the conclusion. □

Appendix B

A Property of Saturated Ideals of Regular Chains

Proposition B.7 and Theorem B.1 are the main statements of this second appendix. They are often used to prove properties on regular chains. In fact, up to presentation details, these results are established in the proof of Theorem 6.1 in the landmark paper [1]. However, the treatment there is specialized to multivariate polynomial rings over a field, whereas we work here in a univariate polynomial ring over an arbitrary commutative ring.

This more abstract treatment was proposed by Aubry in [5]. It has been simplified by Moreno Maza (unpublished notes) such that the only prerequisite for following the proof is the fact that univariate pseudo-division is uniquely defined whenever the leading coefficient of the pseudo-divisor is a regular element of the coefficient ring.

Throughout this section, we consider a commutative ring $\mathbb{A}$ and the ring $\mathbb{A}[x]$ of the univariate polynomials in x with coefficients in $\mathbb{A}$. Let $\mathcal{I}$ be an ideal of $\mathbb{A}$. We denote by $\mathcal{I}[x]$ the ideal generated by $\mathcal{I}$ in $\mathbb{A}[x]$.

Proposition B.1. *Let $f = \sum_{i=0}^{n} a_i x^i \in \mathbb{A}[x]$ be a polynomial. Then, we have*

$$f \in \mathcal{I}[x] \iff (\forall i \in \{0, \ldots, n\})\ a_i \in \mathcal{I}.$$

Proof. Assume that $f \in \mathcal{I}[x]$ holds. Then, there exists $b_1, \ldots, b_m \in \mathcal{I}$ and $g_1, \ldots, g_m \in \mathbb{A}[x]$ satisfying $f = b_1 g_1 + \cdots + b_m g_m$. From there, it is routine to show that each coefficient of f is the ideal generated by $b_1, \ldots, b_m$ and thus in $\mathcal{I}$. The converse implication is clear, which concludes the proof. □

Proposition B.2. *Let $p \in \mathbb{A}$. Then we have: p is zero in $\mathbb{A}/\mathcal{I}$ if and only if p is zero in $\mathbb{A}[x]/\mathcal{I}[x]$; p is regular in $\mathbb{A}/\mathcal{I}$ if and only if p is regular in $\mathbb{A}[x]/\mathcal{I}[x]$.*

Proof. Since $\mathcal{I} \subseteq \mathcal{I}[x]$, we deduce that p is zero in $\mathbb{A}/\mathcal{I}$ implies that p is zero in $\mathbb{A}[x]/\mathcal{I}[x]$. Conversely, if p is zero in $\mathbb{A}[x]/\mathcal{I}[x]$, by Proposition B.1, we have $p \in \mathcal{I}$ and thus p is zero in $\mathbb{A}/\mathcal{I}$.

If p is regular in $\mathbb{A}/\mathcal{I}$. Let $q = \sum_{i=0}^{n} a_i x^i \in \mathbb{A}[x]$ such that $pq \in \mathcal{I}[x]$. By Proposition B.1, we have $pa_i \in \mathcal{I}$, which implies that $a_i \in \mathcal{I}$ and therefore $q \in \mathcal{I}[x]$. Thus p is regular in $\mathbb{A}[x]/\mathcal{I}[x]$. Conversely, if p is regular in $\mathbb{A}[x]/\mathcal{I}[x]$. Let $q \in \mathbb{A}$ such that $pq \in \mathcal{I}$. Then $pq \in \mathcal{I}[x]$, which implies that $q \in \mathcal{I}[x]$ and thus $q \in \mathcal{I}$. So p is regular in $\mathbb{A}/\mathcal{I}$. □

Proposition B.3. *For any $h \in \mathbb{A}$ we have $(\mathcal{I} : h^\infty)[x] = (\mathcal{I}[x]) : h^\infty$.*

Proof. Let $a \in \mathcal{I} : h^\infty$. Clearly, we have $a \in (\mathcal{I}[x]) : h^\infty$. Consequently, Proposition B.1 shows that the ideal generated by $\mathcal{I} : h^\infty$ in $\mathbb{A}[x]$ is contained in the ideal $(\mathcal{I}[x]) : h^\infty$. Conversely, let $f \in (\mathcal{I}[x]) : h^\infty$. Then, there exists $m \in \mathbb{N}$ such that $h^m f \in \mathcal{I}[x]$, which implies, that every coefficient of f lies in $\mathcal{I} : h^\infty$. Hence, Proposition B.1, implies that $f \in (\mathcal{I} : h^\infty)[x]$ holds. □

In the sequel of this appendix, we denote by $f \in \mathbb{A}[x]$ a non-constant polynomial such that its leading coefficient, denoted by h, is not a zero-divisor in $\mathbb{A}/\mathcal{I}$. We define

$$\mathcal{J} = \langle \mathcal{I}, f \rangle.$$

Proposition B.4. *We have $\mathcal{I} = \mathcal{J} \cap \mathbb{A}$.*

Proof. Clearly, we have $\mathcal{I} \subseteq \mathcal{J} \cap \mathbb{A}$. Conversely, let $p \in \mathcal{J} \cap \mathbb{A}$. Thus p is a constant polynomial. Let us prove that p belongs to $\mathcal{I}$. Since $p \in \mathcal{J}$, there exists $q \in \mathbb{A}[x]$ satisfying

$$p - qf \in \mathcal{I}[x].$$

Assume that $q \notin \mathcal{I}[x]$. Then qf, and thus p, has a positive degree in $\mathcal{I}[x]$. Indeed, since h is regular modulo $\mathcal{I}$, we have $\deg(p) = \deg(q) + \deg(f)$. This contradicts the hypothesis that p is a constant in $\mathbb{A}[x]$. Therefore $q \in \mathcal{I}[x]$, and thus $p \in \mathcal{I}[x]$ both hold. Since p is a constant, the conclusion follows. □

Proposition B.5. *We have $\mathcal{I} = (\mathcal{J} : h^\infty) \cap \mathbb{A}$.*

Proof. We clearly have $\mathcal{J} \subseteq \mathcal{J} : h^\infty$. We deduce $\mathcal{J} \cap \mathbb{A} \subseteq (\mathcal{J} : h^\infty) \cap \mathbb{A}$. Thus, with Proposition B.4, we have $\mathcal{I} \subseteq (\mathcal{J} : h^\infty) \cap \mathbb{A}$. Conversely, let $a \in (\mathcal{J} : h^\infty) \cap \mathbb{A}$. There exists $n \in \mathbb{N}$ such that $h^n a \in \mathcal{J} \cap \mathbb{A}$. Thus, with Proposition B.4 again, we have $h^n a \in \mathcal{I}$. Since h is not a zero-divisor modulo $\mathcal{I}$, we deduce $a \in \mathcal{I}$, concluding the proof. □

Proposition B.6. *Let* $r \in \mathbb{A}[x]$ *with* $r \neq 0$ *and* $\deg(r) < \deg(f)$. *Then, the following holds:*

$$r \in \mathcal{J} : h^\infty \quad \Rightarrow \quad r \in \mathcal{I}[x].$$

Proof. We assume $r \in \mathcal{J} : h^\infty$ and prove that $r \in \mathcal{I}[x]$ holds. Let $m \in \mathbb{N}$ be such that $h^m r \in \mathcal{J}$. Then, let $q \in \mathbb{A}[x]$ satisfying

$$h^m r - qf \in \mathcal{I}[x]. \tag{B.1}$$

Assume $q \notin \mathcal{I}[x]$ holds. Since $h = \mathrm{lc}(f)$ is regular modulo $\mathcal{I}$, the degree of qf in $\mathcal{I}[x]$ is at least that of f in $\mathbb{A}[x]$. Equation (B.1) shows this contradicts $\deg(r) < \deg(f)$. Therefore, we have $q \in \mathcal{I}[x]$ which implies $r \in \mathcal{I}[x]$ as claimed. □

Proposition B.7. *For all* $p \in \mathbb{A}[x]$*, the following conditions are equivalent:*

(i) $p \in \mathcal{J} : h^\infty$,

(ii) $\mathrm{prem}(p, f) \in \mathcal{I}[x]$.

Proof. Define $r = \mathrm{prem}(p, f)$ and $q = \mathrm{pquo}(p, f)$. Let $n \in \mathbb{N}$ be such that $h^n p = qf + r$. We assume (i) and prove (ii). Both p and f belong to $\mathcal{J} : h^\infty$. Thus r belongs to $\mathcal{J} : h^\infty$ too. Applying Proposition B.6, we deduce $r \in \mathcal{I}[x]$ as expected. Now, we assume (ii) and prove (i). Since $\mathcal{I}[x] \subset \mathcal{J}$ holds, we deduce that both r and f belong to $\mathcal{J}$. This implies $h^n p \in \mathcal{J}$, that is, $p \in \mathcal{J} : h^\infty$ as claimed. □

Theorem B.1. *Let* $\mathcal{K}$ *be an ideal of* $\mathbb{A}$ *and* $a \in \mathbb{A}$ *be such that we have* $\mathcal{I} = \mathcal{K} : a^\infty$. *Assume that* $h = \mathrm{lc}(f)$ *is regular modulo* $\mathcal{I}$. *Assume that it is also regular in* $\mathbb{A}$. *Then, we have the following identity:*

$$\langle \mathcal{K} : a^\infty, f \rangle : h^\infty = \langle \mathcal{K}, f \rangle : (ah)^\infty. \tag{B.2}$$

Proof. First, we prove that $\langle \mathcal{I}, f \rangle : h^\infty$ is contained in $\langle \mathcal{K}, f \rangle : (ah)^\infty$. Let $p \in \langle \mathcal{I}, f \rangle : h^\infty$. Proposition B.7 implies $\mathrm{prem}(p, f) \in \mathcal{I}[x]$. With Proposition B.3, we deduce $\mathrm{prem}(p, f) \in (\mathcal{K}[x]) : a^\infty$. Hence, there exists $m \in \mathbb{N}$ such that we have $a^m \mathrm{prem}(p, f) \in \mathcal{K}[x]$. Using the fact that the pseudo-division by f in $\mathbb{A}[x]$ is uniquely defined (since h is regular in $\mathbb{A}$) we deduce $\mathrm{prem}(a^m p, f) \in \mathcal{K}[x]$. Thus, there exists $n \in \mathbb{N}$ such that $h^n a^m p \in \langle \mathcal{K}, f \rangle$, leading to $p \in \langle \mathcal{K}, f \rangle : (ah)^\infty$.

Conversely, let $p \in \langle \mathcal{K}, f \rangle : (ah)^\infty$. Thus, there exists $m \in \mathbb{N}$ such that $a^m p \in \langle \mathcal{K}, f \rangle : h^\infty$. Observe that we have: $\langle \mathcal{K}, f \rangle \subseteq \langle \mathcal{I}, f \rangle = \mathcal{J}$ and thus: $\langle \mathcal{K}, f \rangle : h^\infty \subseteq \langle \mathcal{I}, f \rangle : h^\infty \subseteq \mathcal{J} : h^\infty$. Hence $a^m p \in \mathcal{J} : h^\infty$ holds. Applying Proposition B.7

we deduce $\text{prem}(a^m p, f) \in \mathcal{I}[x]$. Using again the fact that the pseudo-division by f in $\mathbb{A}[x]$ is uniquely defined, we obtain $a^m \text{prem}(p, f) \in \mathcal{I}[x]$, that is $\text{prem}(p, f) \in (\mathcal{I}[x]) : a^\infty$. Since a is regular modulo $\mathcal{I}$, by Proposition B.2, a is regular modulo $\mathcal{I}[x]$. Hence, we deduce $\text{prem}(p, f) \in \mathcal{I}[x]$. Applying Proposition B.7 again, we conclude $p \in \langle \mathcal{I}, f \rangle : h^\infty$. $\square$

Bibliography

[1] I. M. Anderson and Fels M. E. Transverse group actions on bundles. *preprint*, 1999.

[2] V. I. Arnold. *Ordinary Differential Equations*. Springer-Verlag, 1992.

[3] D. S. Arnon, G. E. Collins, and S. McCallum. Cylindrical algebraic decomposition I: the basic algorithm. *SIAM J. Comput.*, 13(4):865–877, 1984.

[4] D. S. Arnon, G. E. Collins, and S. McCallum. Cylindrical algebraic decomposition II: an adjacency algorithm for the plane. *SIAM J. Comput.*, 13(4):878–889, 1984.

[5] P. Aubry. *Ensembles triangulaires de polynômes et résolution des systèmes d'équations algébriques*. PhD thesis, Université Paris 6, 1999.

[6] P. Aubry, D. Lazard, and M. Moreno Maza. On the theories of triangular sets. *J. Symb. Comp.*, 28(1-2):105–124, 1999.

[7] P. Aubry and M. Moreno Maza. Triangular sets for solving polynomial systems: A comparative implementation of four methods. *J. Symb. Comp.*, 28(1-2):125–154, 1999.

[8] J. Backelin and R. Fröberg. How we proved that there are exactly 924 cyclic 7-roots. In S. M. Watt, editor, *Proc. ISSAC'91*, pages 103–111. ACM, 1991.

[9] S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in real algebraic geometry*, volume 10 of *Algorithms and Computations in Mathematics*. Springer-Verlag, 2006.

[10] James C. Beaumont, Russell J. Bradford, James H. Davenport, and Nalina Phisanbut. Testing elementary function identities using CAD. *Appl. Algebra Eng., Commun. Comput.*, 18:513–543, November 2007.

[11] D. J. Bernstein. Factoring into coprimes in essentially linear time. *J. Algorithms*, 54(1):1–30, 2005.

[12] F. Boulier, C. Chen, F. Lemaire, and M. Moreno Maza. Real root isolation of regular chains. In *Proc. of ASCM'09*, 2009.

[13] F. Boulier, D. Lazard, F. Ollivier, and M. Petitot. Representation for the radical of a finitely generated differential ideal. In *proceedings of ISSAC'95*, pages 158–166, Montréal, Canada, 1995.

[14] F. Boulier, F. Lemaire, and M. Moreno Maza. Well known theorems on triangular systems and the D5 principle. In *Proc. of* Transgressive Computing 2006, Granada, Spain, 2006.

[15] Russell Bradford, Robert M. Corless, James H. Davenport, David J. Jeffrey, and Stephen M. Watt. Reasoning about the elementary functions of complex analysis. *Annals of Mathematics and Artificial Intelligence*, 36:303–318, 2002.

[16] C. W. Brown. Guaranteed solution formula construction. In *Proc. ISSAC'99*, pages 137–144, 1999.

[17] C. W. Brown. Improved projection for cylindrical algebraic decomposition. *J. Symb. Comput.*, 32(5):447–465, 2001.

[18] C. W. Brown. Simple cad construction and its applications. *J. Symb. Comput.*, 31(5):521–547, 2001.

[19] C. W. Brown. QEPCAD B: a program for computing with semi-algebraic sets using cads. *SIGSAM Bull.*, 37(4):97–108, 2003.

[20] C. W. Brown and J. H. Davenport. The complexity of quantifier elimination and cylindrical algebraic decomposition. In *Proc. ISSAC'07*, pages 54–60.

[21] C. W. Brown and S. McCallum. On using bi-equational constraints in cad construction. In *ISSAC'05*, pages 76–83, 2005.

[22] P. Bürgisser, M. Clausen, and M. A. Shokrollahi. *Algebraic Complexity Theory.* Springer, 1997.

[23] J. Carr. *Applications of Centre Manifold Theory.* Springer-Verlag, 1981.

[24] B. Caviness and J. Johnson, editors. *Quantifier Elimination and Cylindical Algebraic Decomposition, Texts and Mongraphs in Symbolic Computation*. Springer, 1998.

[25] C. Chen. Algebraic analysis of stability for biological systems and the implemetation of a software pakage (in chinese). Master's thesis, Peking University, 2006.

[26] C. Chen, J. H. Davenport, J. May, M. Moreno Maza, B. Xia, and R. Xiao. Triangular decomposition of semi-algebraic systems. In S.M. Watt, editor, *Proceedings ISSAC 2010*, pages 187–194, 2010.

[27] C. Chen, J. H. Davenport, J. May, M. Moreno Maza, B. Xia, and R. Xiao. Triangular decomposition of semi-algebraic systems. *J. Symb. Comp*, 2011. To appear.

[28] C. Chen, J. H. Davenport, J. May, M. Moreno Maza, B. Xia, R. Xiao, and Y. Xie. User interface design for geometrical decomposition algorithms in maple. In *Proc.* Mathematical User-Interface 2009, pages 1–12, 2009.

[29] C. Chen, J. H. Davenport, M. Moreno Maza, B. Xia, and R. Xiao. Computing with semi-algebraic sets represented by triangular decomposition. In *Proc. of ISSAC'11*, pages 75–82, 2011.

[30] C. Chen, O. Golubitsky, F. Lemaire, M. Moreno Maza, and W. Pan. *Comprehensive Triangular Decomposition*, volume 4770 of *Lecture Notes in Computer Science*, pages 73–101. Springer Verlag, 2007.

[31] C. Chen, F. Lemaire, M. Moreno Maza, W. Pan, and Y. Xie. Efficient computations of irredundant triangular decompositions with the regularchains library. Proc. of CASA 2007, 2007.

[32] C. Chen, M. Moreno Maza, W. Pan, and Y. Xie. On the verification of polynomial system solvers. *Frontiers of Computer Science in China*, 2(1):55–66, 2008.

[33] C. Chen and M. Moreno Maza. Algorithms for computing triangular decompositions of polynomial systems. In *Proc. of ISSAC'11*, pages 83–90, 2011.

[34] C. Chen and M. Moreno Maza. Semi-algebraic description of the equilibria of dynamical systems. In *Proc. CASC' 2011*, 2011.

[35] C. Chen, M. Moreno Maza, W. Pan, and Y. Xie. On the verification of polynomial system solvers. In *Proceedings of AWFS 2007*, pages 116–144, 2007.

[36] C. Chen, M. Moreno Maza, B. Xia, and L. Yang. Computing cylindrical algebraic decomposition via triangular decomposition. In *ISSAC'09*, pages 95–102, 2009.

[37] F. Chen and D. Wang, editors. *Geometric Computation*. Number 11 in Lecture Notes Series on Computing. World Scientific Publishing Co., Singapore, New Jersey, 2004.

[38] G. Chen and Jean Della Dora. Rational normal form for dynamical systems by Carleman linearization. In S. Dooley, editor, *Proc. 1999 International Symposium on Symbolic and Algebraic Computation (ISSAC)*, pages 165–172, New York, 1999. ACM Press.

[39] G. Chen and Jean Della Dora. An algorithm for computing a new normal form for dynamical systems. *Journal of Symbolic Computation*, 29(3):393–418, 2000.

[40] G. Chen, Jean Della Dora, and Laurent Stolovitch. Nilpotent normal form via Carleman linearization (for systems of ordinary differential equations). In S. Watt, editor, *Proc. 1991 International Symposium on Symbolic and Algebraic Computation (ISSAC)*, pages 281–288, New York, 1991. ACM Press.

[41] J. S. Cheng, X. S. Gao, and C. K. Yap. Complete numerical isolation of real zeros in zero-dimensional triangular systems. In *ISSAC*, pages 92–99, 2007.

[42] S. C. Chou and X. S. Gao. Computations with parametric equations. In *Proc. ISSAC'91*, pages 122–127, Bonn, Germany, 1991.

[43] S. C. Chou and X. S. Gao. Solving parametric algebraic systems. In *Proc. ISSAC'92*, pages 335–341, Berkeley, California, 1992.

[44] G. E. Collins. Quantifier elimination for real closed fields by cylindrical algebraic decomposition. *Springer Lecture Notes in Computer Science*, 33:515–532, 1975.

[45] G. E. Collins and H. Hong. Partial cylindrical algebraic decomposition. *Journal of Symbolic Computation*, 12(3):299–328, 1991.

[46] G. E. Collins, J. R. Johnson, and W. Krandick. Interval arithmetic in cylindrical algebraic decomposition. *J. Symb. Comput.*, 34(2):145–157, 2002.

[47] X. Dahan, A. Kadri, and É. Schost. Bit-size estimates for triangular sets in positive dimension. Technical report, University of Western Ontario, 2009.

[48] X. Dahan, M. Moreno Maza, É. Schost, W. Wu, and Y. Xie. Lifting techniques for triangular decompositions. In *ISSAC'05*, pages 108–115. ACM Press, 2005.

[49] X. Dahan, M. Moreno Maza, É. Schost, and Y. Xie. On the complexity of the D5 principle. In *Proc. of* Transgressive Computing 2006, Granada, Spain, 2006.

[50] J. H. Davenport, Y. Siret, and E. Tournier. *Computer Algebra*. Academic Press, 1988.

[51] J. Della Dora, C. Dicrescenzo, and D. Duval. About a new method for computing in algebraic number fields. In *Proc. EUROCAL 85 Vol. 2*, volume 204 of *Lect. Notes in Comp. Sci.*, pages 289–290. Springer-Verlag, 1985.

[52] A. Dolzman and T. Sturm. Redlog: Computer algebra meets computer logic. *ACM SIGSAM Bulletin*, 31(2):2–9, 1996.

[53] A. Dolzmann, A. Seidl, and T. Sturm. Efficient projection orders for cad. In *Proc. ISSAC '04*, pages 111–118. ACM, 2004.

[54] A. Dolzmann, T. Sturm, and V. Weispfenning. Real quantifier elimination in practice. In *Algorithmic Algebra and Number Theory*, pages 221–247, 1998.

[55] L. Donati and C. Traverso. Experimenting the Gröbner basis algorithm with the ALPI system. In *Proc. ISSAC'89*, pages 192–198. ACN Press, 1989.

[56] L. Ducos. Optimizations of the subresultant algorithm. *Journal of Pure and Applied Algebra*, 145:149–163, 2000.

[57] D. Duval. Algebraic numbers: an example of dynamic evaluation. *J. Symb. Comp.*, 18(5):429–446, November 1994.

[58] Frazer, R. A. and Duncan, W. J. On the criteria for the stability of small motions. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, 124(795):642–654, 1929.

[59] E. Freire, E. Gamero, E. Ponce, and L. García Franquelo. An algorithm for symbolic computation of center manifolds. In *Proc. of ISAAC '88*, pages 218–230, London, UK, 1989. Springer-Verlag.

[60] A.T. Fuller. Conditions for a matrix to have only characteristic roots with negative real parts. *Journal of Mathematical Analysis and Applications*, 23:71–98, 1968.

[61] G. J. Pappas G. Lafferriere and S. Yovine. Symbolic reachability computation for families of linear vector fields. *J. Symb. Comput.*, 32(3):231–253, 2001.

[62] F.R. Gantmacher. *The Theory of Matrices*. Chelsea Publishing Company, New York, N.Y., 1959.

[63] X. S. Gao, J. Van der Hoeven, Y. Luo, and C. Yuan. Characteristic set method for differential-difference polynomial systems. *J. Symb. Comput.*, 44:1137–1163, 2009.

[64] K. Gatermann, M. Eiswirtha, and A. Sensse. Toric ideals and graph theory to analyze Hopf bifurcations in mass action systems. *Journal of Symbolic Computation*, 40(6):1361–1382, 2005.

[65] K. Gatermann and S. Hosten. Computational algebra for bifurcation theory. *Journal of Symbolic Computation*, 40(4-5):1180–1207, 2005.

[66] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, 1999.

[67] K.O. Geddes, S.R. Czapor, and G. Labahn. *Algorithms for Computer Algebra*. Kluwer Academic Publishers, 1992.

[68] T. Gómez Díaz. *Quelques applications de l'évaluation dynamique*. PhD thesis, Université de Limoges, 1994.

[69] L. Gonzalez, H. Lombardi, T. Recio, and M.-F. Roy. Sturm-Habicht sequence. In *ISSAC '89: Proceedings of the ACM-SIGSAM 1989 international symposium on Symbolic and algebraic computation*, pages 136–146, New York, NY, USA, 1989. ACM.

[70] The Computational Mathematics Group. The basicmath library. NAG Ltd, Oxford, UK, 1998. http://www.nag.co.uk/projects/FRISCO.html.

[71] J. Guckenheimer, M. Myers, and B. Sturmfels. Computing hopf bifurcations i. *SIAM J. Numer. Anal.*, 34(1):1–21, 1997.

[72] J. Hale and H. Koçak. *Dynamics and Bifurcations.* Springer-Verlag, New York, Berlin, Heidelberg, London, 1991.

[73] H. Hong. An improvement of the projection operator in cylindrical algebraic decomposition. In *ISSAC '90*, pages 261–264. ACM, 1990.

[74] H. Hong. Simple solution formula construction in cylindrical algebraic decomposition based quantifier elimination. In *ISSAC '92*, pages 177–188. ACM, 1992.

[75] H. Hong, R. Liska, and S. Steinberg. Testing stability by quantifier elimination. *Journal of Symbolic Computation*, 24(2):161–187, 1997.

[76] H. Hong and J. R. Sendra. Computation of variant results. In B. Caviness and J. Johnson, editors, *Quantifier Elimination and Cylindrical Algebraic Decomposition, Texts and Monographs in Symbolic Computation.* Springer Verlag, 1998.

[77] H. Hong *et al.* QEPCAD B, `www.usna.edu/Users/cs/qepcad/`.

[78] É. Hubert. Factorization free decomposition algorithms in differential algebra. *J. Symb. Comp.*, 29(4-5):641–662, 2000.

[79] M'Hammed El Kahoui and Andreas Weber. Deciding hopf bifurcations by quantifier elimination in a software-component architecture. *J. Symb. Comput.*, 30(2):161–179, 2000.

[80] M'Hammed El Kahoui and Andreas Weber. Symbolic equilibrium point analysis in parameterized polynomial vector fields. In V. Ganzha, E. Mayr, and E. Vorozhtsov, editors, *Computer Algebra in Scientific Computing (CASC 2002)*, pages 71–83, 2002.

[81] M. Kalkbrener. A generalized euclidean algorithm for computing triangular representations of algebraic varieties. *J. Symb. Comp.*, 15:143–167, 1993.

[82] T. Krick, L. M. Pardo, and M. Sombra. Sharp estimates for the arithmetic Nullstellensatz. *Duke Math. J.*, 109(3):521–598, 2001.

[83] Y. A. Kuznetsov. *Elements of Applied Bifurcation Theory.* Springer Verlag, 1998.

[84] M. Laurent. Prion diseases and the "protein only" hypothesis: a theoretical dynamic study. *Biochem. J.*, 318:35–39, 1996.

[85] D. Lazard. A new method for solving algebraic systems of positive dimension. *Discr. App. Math*, 33:147–160, 1991.

[86] Daniel Lazard and Fabrice Rouillier. Solving parametric polynomial systems. *J. Symb. Comput.*, 42(6):636–667, 2007.

[87] G. Lecerf. Computing the equidimensional decomposition of an algebraic closed set by means of lifting fibers. *J. Complexity*, 19(4):564–596, 2003.

[88] F. Lemaire, M. Moreno Maza, and Y. Xie. The `RegularChains` library. In Ilias S. Kotsireas, editor, Maple Conference 2005, pages 355–368, 2005.

[89] X. Li, M. Moreno Maza, and W. Pan. Computations modulo regular chains. In *Proc. ISSAC'09*, pages 239–246, New York, NY, USA, 2009. ACM Press.

[90] X. Li, M. Moreno Maza, and W. Pan. Computations modulo regular chains, 2009. Submitted to ISSAC'09.

[91] X. Li, M. Moreno Maza, R. Rasheed, and É. Schost. The Modpn library: Bringing fast polynomial arithmetic into Maple. In *MICA'08*, 2008.

[92] X. Li, M. Moreno Maza, and É Schost. Fast arithmetic for triangular sets: From theory to practice. In *Proc. ISSAC'07*, pages 269–276, New York, NY, USA, 2007. ACM Press.

[93] X. Liu, R. M. Corless, and K. O. Geddes. Computation of center manifolds. Technical Report TR-00-15, at http://www.orcca.on.ca/TechReports, Ontario Research Centre for Computer Algebra, 2000. 12 pages.

[94] Zhengyi Lu, Bi He, Yong Luo, and Lu Pan. An algorithm of real root isolation for polynomial systems. In D. M. Wang and L. Zhi, editors, *Proceedings of Symbolic Numeric Computation 2005*, pages 94–107, 2005.

[95] M. Colón, S. Sankaranarayanan and H. B. Sipma. Linear invariant generation using non-linear constraint solving. In *CAV'03, LNCS 2725*, pages 420–432, 2003.

[96] M. Manubens and A. Montes. Improving dispgb algorithm using the discriminant ideal. *Journal of Symbolic Computation*, 41:1245, 2006.

[97] M. Marden. *Geometry of polynomials.* Mathematical surveys. American Mathematical Society, 1966.

[98] S. McCallum. An improved projection operation for cylindrical algebraic decomposition of 3-dimensional space. *J. Symb. Comput.*, 5(1-2):141–161, 1988.

[99] S. McCallum. Solving polynomial strict inequalities using cylindrical algebraic decomposition. *The Computer Journal*, 36(5):432–438, 1993.

[100] R. K. Miller and A. N. Michel. *Ordinary Differential Equations.* Academic Press, 1982.

[101] B. Mishra. *Algorithmic Algebra.* Springer-Verlag, New York, 1993.

[102] A. Montes. A new algorithm for discussing gröbner bases with parameters. *J. Symb. Comput.*, 33(2):183–208, 2002.

[103] M. Moreno Maza. On triangular decompositions of algebraic varieties. Technical Report TR 4/99, NAG Ltd, Oxford, UK, 1999. Presented at the MEGA-2000 Conference, Bath, England.

[104] M. Moreno Maza. On triangular decompositions of algebraic varieties. Technical Report TR 4/99, NAG Ltd, Oxford, UK, 1999. http://www.csd.uwo.ca/∼moreno.

[105] A. H. Nayfeh. *Method of Normal Forms.* Wiley Series in Nonlinear Sciences. John Wiley & Sons, New York, Chichester, Brisbane, 1993.

[106] W. Niu and D. M. Wang. Algebraic approaches to stability analysis of biological systems. *Mathematics in Computer Science*, 1:507–539, 2008.

[107] L. Perko. *Differential Equations and Dynamical Systems.* Springer-Verlag New York, Inc., New York, NY, USA, 1991.

[108] P. Philippon. Sur des hauteurs alternatives III. *J. Math. Pures Appl.*, 74(4):345–365, 1995.

[109] J. Renegar. On the computational complexity and geometry of the first-order theory of the reals. parts I–III. *J. Symb. Comput.*, 13(3):255–299, 1992.

[110] H. B. Sipma S. Sankaranarayanan and Z. Manna. Non-linear loop invariant generation using gröbner bases. In *ACM POPL'04*, pages 318–329, 2004.

[111] D. G. Schaeffer and M. Golubitsky. *Singularities and Groups in Bifurcation Theory*, volume 1. Springer-Verlag, New York, Berlin, Heidelberg, Tokyo, 1984.

[112] T. Shimoyama and K. Yokoyama. Localization and primary decomposition of polynomial ideals. *J. Symb. Comput.*, 22(3):247–277, 1996.

[113] W. Sit. Computations on quasi-algebraic sets. In R. Liska, editor, *Electronic Proceedings of IMACS ACA'98*, 1998.

[114] A. J. Sommese, J. Verschelde, and C. W. Wampler. Solving polynomial systems equation by equation. In *Algorithms in Algebraic Geometry*, pages 133–152. Springer-Verlag, 2008.

[115] A.J. Sommese, J. Verschelde, and C.W. Wampler. Numerical decomposition of the solution sets of polynomial systems into irreducible components. *SIAM J. Numer. Anal.*, 38(6):2022–2046, 2001.

[116] A. Strzeboński. Solving systems of strict polynomial inequalities. *J. Symb. Comput.*, 29(3):471–480, 2000.

[117] A. Suzuki and Y. Sato. A simple algorithm to compute comprehensive Gröbner bases. In *Proc. ISSAC'06*, pages 326–331. ACM Press, 2006.

[118] *The SymbolicData Project.* http://www.SymbolicData.org, 2000–2006.

[119] Á. Szántó. *Computation with polynomial systems.* PhD thesis, Cornell University, 1999.

[120] J. M. Thomas. *Differential System.* American Mathematical Society, New York, 1937.

[121] A. Tiwari. Termination of linear programs. In *CAV'04, LNCS 3114*, pages 387–390, 2004.

[122] L. Vallier. An algorithm for the computation of normal forms and invariant manifolds. In *Proc. of ISSAC '93*, pages 225–233, New York, NY, USA, 1993. ACM Press.

[123] D. K. Wang. The `Wsolve` package. http://www.mmrc.iss.ac.cn/~dwang/wsolve.txt.

[124] D. M. Wang. Epsilon 0.618. http://www-calfor.lip6.fr/wang/epsilon.

[125] D. M. Wang. Decomposing polynomial systems into simple systems. *J. Symb. Comp.*, 25(3):295–314, 1998.

[126] D. M. Wang. Computing triangular systems and regular systems. *J. Sym. Comp.*, 30(2):221–236, 2000.

[127] D. M. Wang. *Elimination methods.* Springer, New York, 2001.

[128] D. M. Wang and B. Xia. Stability analysis of biological systems with real solution classfication. In M. Kauers, editor, *Proc. 2005 International Symposium on Symbolic and Algebraic Computation (ISSAC)*, pages 354–361, New York, 2005. ACM Press.

[129] D. M. Wang and Z. M. Zheng. *Differential Equations with Symbolic Computation.* Birkhäuser Verlag, Basel, Boston, Berlin, 2005.

[130] V. Weispfenning. Comprehensive grobner bases. *J. Symb. Comp.*, 14:1–29, 1992.

[131] V. Weispfenning. Canonical comprehensive grobner bases. In *ISSAC 2002*, pages 270–276. ACM Press, 2002.

[132] W. T. Wu. A zero structure theorem for polynomial equations solving. *MM Research Preprints*, 1:2–12, 1987.

[133] W. T. Wu. On a projection theorem of quasi-varieties in elimination theory. *MM Research Preprints*, 4:40–53, 1989.

[134] B. Xia and L. Yang. An algorithm for isolating the real solutions of semi-algebraic systems. *J. Symb. Comput.*, 34(5):461–477, 2002.

[135] B. Xia and T. Zhang. Real solution isolation using interval arithmetic. *Comput. Math. Appl.*, 52(6-7):853–860, 2006.

[136] R. Xiao. *Parametric Polynomial System Solving.* PhD thesis, Peking University, Beijing, 2009.

[137] L. Yang. Recent advances on determining the number of real roots of parametric polynomials. *J. Symb. Comput.*, 28(1-2):225–242, 1999.

[138] L. Yang, X. R. Hou, and B. Xia. A complete algorithm for automated discovering of a class of inequality-type theorems. *Science in China, Series* **F**, 44(6):33–49, 2001.

[139] L. Yang and B. Xia. Real solution classifications of a class of parametric semi-algebraic systems. In *Proc. of the A3L'05*, pages 281–289, 2005.

[140] L. Yang and B. Xia. *Automated proving and discovering inequalities.* Science Press, Beijing, 2008.

[141] L. Yang and J. Zhang. Searching dependency between algebraic equations: an algorithm applied to automated reasoning. Technical Report IC/89/263, International Atomic Energy Agency, Miramare, Trieste, Italy, 1991.

[142] P. Yu and Y. Yuan. An efficient method for computing the simplest normal forms of vector fields. *Int. J. Bifurcations & Chaos*, 13(1):19–46, 2003.

Curriculum Vitae

Name:	**Changbo Chen**
Post-Secondary Education and Degrees:	The University of Western Ontario London, Ontario, Canada Ph.D. in Computer Science, August 2011
	Peking University Beijing, China M.Sc. in Mathematics, July 2006
	Shandong University of Science and Technology Tai'an, Shangdong, China B.Sc. in Mathematics and Applied Mathematics, July 2003
Honours and Awards:	Graduate Thesis Research Fund The University of Western Ontario, January 2010
	Excellent Student Awards Shandong University of Science and Technology, 1999-2003
Related Work Experience:	Internship Maplesoft, Waterloo, Canada May 2010 - August 2010
	Research Assistant and Teaching Assistant The University of Western Ontario, London, Canada September 2006 - August 2011
	Research Assistant and Teaching Assistant Peking University, Beijing, China September 2004 - July 2006

Publications:

- Semi-algebraic description of the equilibria of dynamical systems, joint work with Marc Moreno Maza. CASC 2011.
- Algorithms for computing triangular decompositions of polynomial systems. with Marc Moreno Maza. Proceedings of ISSAC 2011, ACM Press, 2011.
- Computing with semi-algebraic sets represented by triangular decomposition. with James H. Davenport, M. Moreno Maza, Bican Xia and Rong Xiao. Proceedings of ISSAC 2011, ACM Press, 2011.
- Triangular decomposition of semi-algebraic systems, with James H. Davenport, John P. May, M. Moreno Maza, Bican Xia and Rong Xiao. Proceedings of ISSAC 2010, ACM Press, pp. 187–194, 2010.
- Computing Cylindrical Algebraic Decomposition via Triangular Decomposition, with M. Moreno Maza, B. Xia and L. Yang. Proceedings of ISSAC 2009, pages 95-102, ACM Press, New York, 2009.
- Real Root Isolation of Regular Chains, joint work with F. Boulier, C. Chen, F. Lemaire and M. Moreno Maza. ASCM 2009.
- On the Verification of Polynomial System Solvers, with M. Moreno Maza, W. Pan and Y. Xie. Frontiers of Computer Science in China, Vol 2, Number 1, pages 55-66, 2008.
- On the Representation of Constructible Sets, with L. Li, M. Moreno Maza, W. Pan and Y. Xie. Proceedings of Milestones in Computer Algebra 2008, pages 103-108, Trinidad and Tobago, 2008.
- The ConstructibleSetTools and ParametricSystemsTools Modules of the RegularChains Library in Maple, with F. Lemaire, L. Liyun, M. Moreno Maza, W. Pan and Y. Xie. Proceedings of the International Conference on Computational Science and Applications, IEEE Computer Society, pages 342-352, 2008.
- Comprehensive Triangular Decomposition, with F. Lemaire, O. Golubitsky, M. Moreno Maza and W. Pan. Proceedings of CASC 2007: Computer Algebra in Scientific Computing, pages 73-101, Lecture Notes in Computer Science, vol. 4770, Springer-Verlag, 2007.
- On the Verification of Polynomial System Solvers, with M. Moreno Maza, W. Pan and Y. Xie. Proceedings of the Fifth Asian Workshop on Foundations of Software, pages 116-144, University of Xiamen, China, 2007.
- Efficient Computations of Irredundant Triangular Decompositions with the RegularChains Library, with F. Lemaire, M. Moreno Maza, W. Pan and Y. Xie. Proceedings of Computer Algebra Systems and Their Applications'07, Y. Shi et al. (Eds.): ICCS 2007, Part II, LNCS 4488, pp. 268-271, Springer-Verlag Berlin Heidelberg 2007.

Softwares:

- The LazyRealTriangularize, RealTriangularize, SamplePoints commands for computing semi-algebraic systems, with J.H. Davenport, J.P. May, M. Moreno Maza, B. Xia and R. Xiao. In Maple 15, Maplesoft, Canada, 2011.

- The CylindricalAlgebraicDecompose command for computing cylindrical algebraic decompositions, with B. Xia, M. Moreno Maza and L. Yang. The Intersect command for computing efficiently the intersection of a hypersurface and a quasi-component, with M. Moreno Maza. In Maple 14, Maplesoft, Canada, 2010.

- The SemiAlgebraicSetTools Module of the RegularChains Library, with F. Lemaire, M. Moreno Maza, B. Xia, R. Xiao and Y. Xie. In Maple 13, Maplesoft, Canada, 2009.

- The ConstructibleSetTools and ParametricSystemTools Modules of the RegularChains Library, with F. Lemaire, L. Li, M. Moreno Maza, W. Pan and Y. Xie. In Maple 12, Maplesoft, Canada, 2008.