University of Bath

PHD

Curtains in Cylindrical Algebraic Decomposition

Nair, Akshar Sajive

Award date:
2021

Awarding institution:
University of Bath

Link to publication

Alternative formats
If you require this document in an alternative format, please contact:
openaccess@bath.ac.uk

Copyright of this thesis rests with the author. Access is subject to the above licence, if given. If no licence is specified above, original content in this thesis is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC-ND 4.0) Licence (https://creativecommons.org/licenses/by-nc-nd/4.0/). Any third-party copyright material present remains the property of its respective owner(s) and is licensed under its existing terms.

Take down policy
If you consider content within Bath's Research Portal to be in breach of UK law, please contact: openaccess@bath.ac.uk with the details.
Your claim will be investigated and, where appropriate, the item will be removed from public view as soon as possible.

Download date: 13. Jul. 2023

Curtains in Cylindrical Algebraic Decomposition

submitted by

Akshar S. Nair

for the degree of Doctor of Philosophy

of the

University of Bath

Department of Computer Science

May 2021

COPYRIGHT

Attention is drawn to the fact that copyright of this thesis rests with the author. A copy of this thesis has been supplied on condition that anyone who consults it is understood to recognise that its copyright rests with the author and that they must not copy it or use material from it except as permitted by law or with the consent of the author.

Signature of Author..

Akshar S. Nair

Dedicated to my mother
Dr. Poornima Nair
and
to my late father
Mr. Sajive Nair

Summary

Cylindrical Algebraic Decomposition was first introduced by Collins in 1975, to help understand and analyse the real algebraic geometry of a system of polynomials. There are various applications of CAD, specifically quantifier elimination problems in fields ranging from motion planning to analysing financial markets.

CAD algorithms are extremely useful for solving a system of polynomials, but the complexity of CAD algorithms is doubly exponential in the number of variables. This makes it difficult to use for large examples. McCallum attempted to tackle this problem first by switching from sign invariant CADs to order invariant CADs and then by exploiting equational constraints in the input formulae.

However, with the switch to order invariance, if the input formulae contain polynomial constraints that nullify over a region, the CAD algorithms produce an error. Lazard formulated a theory using the lex-least valuation, which removed the nullification problem for general CADs.

In this thesis we carry forward Lazard's theory to exploit equational constraints and tackle the doubly exponential complexity of CAD algorithms. To do this we study the geometry of loci where nullification occurs, which we call curtains. Further to this we adapt our modifications to the most recent algorithm proposed by Brown-McCallum in 2020. We end with a comparison of the theoretical complexity of all the modifications presented in this thesis.

Acknowledgements

First and foremost, I would like to thank my late father, Mr. Sajive Nair, because none of this would be possible without him. He sacrificed a great deal by constantly working hard, that kept him away from his family most days, so that I could have my education done in the finest schools of Mumbai. It was his lifelong dream for me to study abroad, and due to his hard work, this dream was possible. I owe all of this to you.

I would like to thank my supervisors Prof. James Davenport and Prof. Gregory Sankaran, for their constant guidance and support during my time at Bath. During my PhD, they provided the best pastoral care when I was dealing with a few personal matters. I have been extremely fortunate to have them as my supervisors. They have given me immense knowledge and support to become a good researcher.

I would also like to thank Dr. Scott McCallum, Dr. Matthew England, Dr. Russell Bradford and Zak Tonks for their many discussions and collaborations and for making my research with the CAD community truly enjoyable. I would also like to thank Dr. Benjamin Pring for his constant guidance at Oxford and Bath, especially helping me to settle into early stages of research at Bath. During my time at Bath, I would also like to thank Prof. Guy McCusker and Dr. James Laird for their constant support to help to set up various events in the department and liaise student problems to upper management. Thank you to Susan Paddock, Claudia Emery, Lisa Newberry and Alison Humphries for helping through all of the admin processes at the university.

A special thank you to Dr. Russell Bradford and Dr. Christopher W. Brown for the invaluable discussion and feedback during my PhD examination.

I would like to thank Prof. Oleg Chalykh from the University of Leeds, who is one of the main reasons why I chose to pursue research in a field related to pure mathematics. Throughout my academic career he has always been there for guidance and support and I am grateful to him for it. I would also like to thank Prof. Marc Lackenby for supervising my dissertation at Oxford and for backing my PhD application to Bath. I would like to thank Mr. George Washington and Dr. S N Venkatarangan for introducing me to the world of programming and the fields of number theory at the year age of 9. Thank you to Mr. Naveen Mandala and Mr. Vijay Singh for their constant help and support in my education. I would like to thank Mr. Kersi Gazdar for his wonderful piano lessons. I would also like to thank Mrs. Annie Thomas for all her help. Finally, I would like to thank Dr. Ghoshal (fondly called Goshal Sir) for all his support and encouragement for me to study in the UK.

I would like to thank my aunt, Dr. Parimala Shenoy, my late grandmother and my late grandfather for all their support and guidance throughout my life. I would like to thank uncle Amin Thakkar for his constant help throughout the time I have been in the UK. I would like to thank uncle Niko Issac and my late aunt Vinay Mai, for their constant support and encouragement for my move to the UK to pursue my undergraduate at Leeds.

I am very lucky to be surrounded by a great group of supportive and special friends. First of all I would like to thank Crescent Jicol for his constant banter, support, and for the random "Bro-nights" filled with doing overnight research, gaming and binge watching classic movies. I am incredibly grateful to Alexandra Gkolia and Anisia Jicol, who have played an integral role in my personal and professional developments. I would like to thank Fahid Mohammed and YuQing Che for their support and guidance in setting up side projects during my PhD research. I would like to thank Catherine Taylor and Holly Wilson for making my time at the office fun and enjoyable. I would like to thank Calla Tschanz and Tristram Broady for the frequent double dinner dates on their lawn, which provided as a great stress buster whilst writing up my thesis. I would like to thank Joshua Languade and Thomas Sykes for making my time at Leeds the best I could have had and helping me excel at my studies. Thank you Eswara Velan for having faith in me and encouraging me to fill out my application to Oxford; even though it resulted in a very fancy dinner for our flatmates at my expense. I would like to thank all the above mentioned friends again, I have been lucky to call you all my friends and thanks to the all of you the last few years have been extremely enjoyable.

I would also like to thank Karolina Pakenaite for her constant support and caring throughout the final stages of my PhD. Thank you for making these last few stressful months exciting and enjoyable.

I would also like to thank the cutest guide dog in the world, Bosley; and a super special thanks to Gappu, Noddy, Tira and Nibble for making my childhood incredibly eventful, couldn't have asked for any better doggos.

Finally, I would like to end the acknowledgements by thanking the person that I owe all my success till now and yet to come, none of which would have been possible without her, my mother, Dr. Poornima Nair. Throughout my life she has always been the rock behind my back that I could rely on. It was through her nurturing and guidance I have been able to achieve all of my academic accolades. Even after the passing away of my father, she has been there for me, through thick and thin, making sure that I received the best possible education I could. Thank you for everything and thank you for providing me the best life possible for the past 25 years, I owe it all to you.

Contents

1 Introduction **10**
- 1.1 CAD Algorithms 10
- 1.2 Outline of results 11
  - 1.2.1 CAD, Procedural Steps 12
- 1.3 Author's Contribution and Collaborators 13
  - 1.3.1 Funding 14

2 Preliminaries **15**
- 2.1 Polynomial Notations and Properties 15
- 2.2 Semi-Algebraic sets and Polynomial Ideals 18
  - 2.2.1 Squarefree and Irreducible Bases 20
- 2.3 Resultants and Discriminants 21
- 2.4 Quantified Formulae and Quantifier Elimination 23
- 2.5 Cylindrical Algebraic Decomposition 24
- 2.6 The first CAD algorithm 27
- 2.7 McCallum's developments of CAD 28
  - 2.7.1 Single Hypersurface Decomposition 31
  - 2.7.2 Multiple Hypersurface Decomposition 32
  - 2.7.3 Limitations 33
- 2.8 Application of CAD 33

3 A different valuation: Lex-Least **37**
- 3.1 Lex-Least Valuation 37
  - 3.1.1 Lazard Delineability 39
  - 3.1.2 New Terminology 40
- 3.2 Properties of Lex-least Valuation 41
- 3.3 Lifting Algorithm 43

4 Lex-Least Invariance on a Single Hypersurface 46
4.1 Main Theorem 47
4.1.1 Modified Projection Operator 47
4.1.2 Lex-least to Sign Lifting Theorem 48
4.1.3 Modified Lifting Algorithm 51
4.2 Working Example 52

5 Curtains 55
5.1 Definitions and Examples 55
5.2 CAD and Curtains 57
5.3 Curtains and Single Equational Constraints 58
5.4 Summary 60

6 Further Developments of Equational Constraints and Curtains 63
6.1 Curtains on Single Hypersurfaces 64
6.1.1 Point Curtains 64
6.1.2 Decomposing Curtain Base Set 65
6.2 Multiple Hypersurfaces 71
6.3 Summary 75

7 Brown-McCallum Projection 76
7.1 Brown-McCallum Modification 76
7.2 Brown-McCallum Projection with Equational Constraints 79
7.2.1 Single Equational Constraint 79
7.2.2 Multiple Equational Constraints 80
7.3 Summary 82

8 Complexity Analysis 84
8.1 Analysis of McCallum's Projection Operators 86
8.1.1 Single Equational Constraint McCallum 89
8.1.2 Multiple Equational Constraint McCallum 90
8.2 Analysis of Lazard's Projection Operators 92
8.2.1 Single Equational Constraint Lazard 93
8.2.2 Multiple Equational Constraint Lazard 95
8.3 Analysis of Brown-McCallum Projection Operators 96
8.3.1 Single Equational Constraint Brown-McCallum 97
8.3.2 Multiple Equational Constraint Brown-McCallum 99
8.4 Summary 101

9 Further Work 102

9.1 Valuations 102

9.2 Curtain Detection 103

9.3 Implementations 103

List of Figures

2-1 Batman is semi-algebraic! 20
2-2 Algebraic decomposition of $\mathbb{R}^2$ 25
2-3 A Cylindrical Algebraic Decomposition of $\mathbb{R}^2$ 26
2-4 Piano mover's problem 34

4-1 Equational constraint implementation 48
4-2 Curtain component 54

5-1 Different types of curtains 56
5-2 Euclidean neighbourhood 59
5-3 Different types of curtains 60
5-4 Failure at non-point curtains (Example 12) 60
5-5 Success at point curtains (Example 12) 61

6-1 Flow chart describing our approach to decompose the variety described by a single equational constraint 66
6-2 Set of sample points before Algorithm 7 67
6-3 Set of sample points after Algorithm 7 67

Chapter 1

Introduction

Herein we introduce the core subject matter of this thesis, Cylindrical Algebraic Decomposition, mostly referred to as ‘CAD’ within the Computer Algebra research community. First, we describe various pre-existing methods/algorithms used to compute CADs. We then illustrate the various drawbacks associated with these methods. After that, we provide the stages of improvements of the CAD algorithm we have established, leading to our final result of using the Brown-McCallum projection operator for multiple equational constraints. After discussing the various algorithmic modifications in this thesis, we provide an in-depth complexity analysis, further justifying our findings by a thorough comparison of the pre-existing methods.

1.1 CAD Algorithms

In 1975, Collins [Col75] introduced the mathematical structure Cylindrical Algebraic Decomposition (CAD) to gain a better understanding of the real algebraic geometry for a system of polynomials using an algorithmic approach. A CAD is a decomposition of a semi-algebraic set $X \subseteq \mathbb{R}^n$ (for any n) into semi-algebraic sets (also known as cells) homeomorphic to $\mathbb{R}^m$, where $1 \leq m \leq n$, such that the projection of any two cells onto the first k coordinates is either the same or disjoint.

Various real-world problems can be reduced to a system of polynomial equalities and inequalities. Often these problems are in the form of a Quantifier Elimination problem. An example of this is the Piano Mover’s problem [WDEB13], which is discussed later in this thesis. There are several specialised algorithms in this area, but CAD is considered one of the most effective algorithms for Quantifier Elimination.

There are two main problems with CAD algorithms: the first is the inherent complexity; the second is their inability to handle polynomials that nullify over some region. A polynomial is said to nullify if the hypersurface described by the polynomial contains the fibres. The fastest CAD algorithm has doubly exponential complexity in the number of variables of the input polynomials. McCallum has several improvements, such as [McC99] and [McC01], which improve the doubly exponential complexity of CAD algorithms. However, they fail in the case of nullification. McCallum in [McC19] verified the first algorithm (originally given by Lazard [Laz94]) that dealt with the nullification problem but did not consider the modifications made in [McC99] and [McC01].

We aim to improve the efficiency of Lazard's algorithm by adapting [McC99], and [McC01]. We also take a different view on nullification and provide algorithms to deal with it.

1.2 Outline of results

In this thesis we present our development of Lazard's theory to improve the efficiency of CAD algorithms. We also address the nullification problem by introducing geometric objects called 'curtains' and establishing the theory around computational algebraic geometry.

Concepts in topology and algebraic geometry play a pivotal role in the algorithmic modifications provided in this thesis. For this reason, a thorough survey of these concepts is provided in Chapter 2. Also in Chapter 2, we provide a comprehensive literature review around the concepts of CAD followed by a literature review of McCallum's modifications of Collins' CAD algorithm. These modifications exploit polynomial constraints in the input, known as equational constraints.

We summarise the topics and results in the thesis:

- In Chapter 3, we discuss the procedure defined by Lazard in [Laz94] and verified by McCallum and colleagues in [MPP19]. It consists of a new valuation, a new projection operator and a new algorithm that deals with the nullification problem.
- In Chapter 4, we present our first modifications to Lazard's projection operator and algorithm based on the method in [McC99].
- In Chapter 5 we introduce a more geometric point of view of nullification, concentrating on the algebraic varieties of which this happens, which we call

curtains.

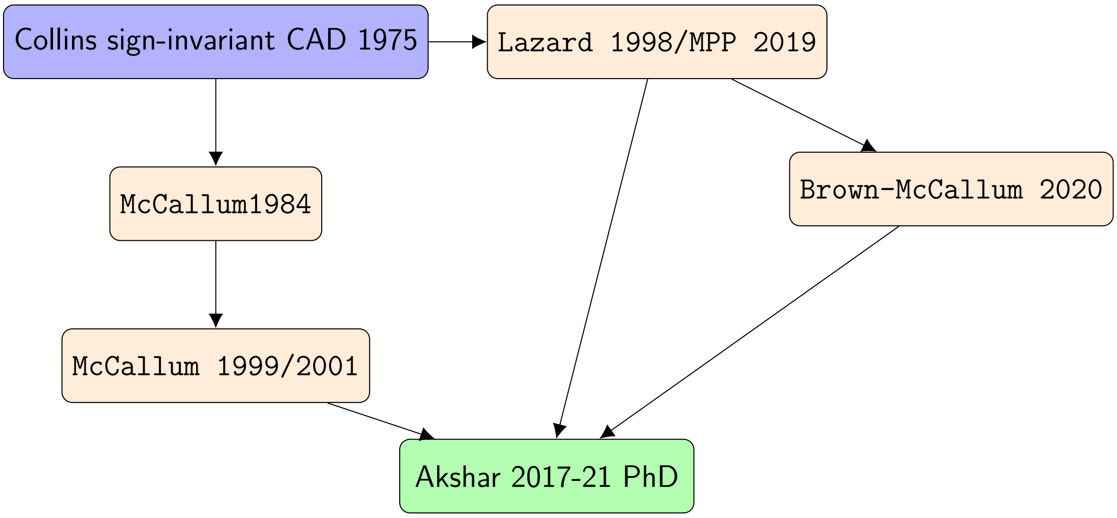
- In Chapter 6, we discuss how to deal with the curtain problem arising in Chapter 4 and provide a modified algorithm that solves it. We discuss various modifications of Lazard's projection operator and algorithm to obtain a result analogous to McCallum's in [McC01].
- In Chapter 7, we describe the recent results of Brown and McCallum in [BM20] and adapt our results from Chapters 4 and 6 to this context.
- Chapter 8 carries out the complexity analysis of the various modified CAD algorithms. It gives a comprehensive understanding of the improvements in the complexity of CAD algorithms. In order to handle the case of equational constraints appropriately, we introduce an enhancement to the existing method used to compute the complexity. This gives us an improvement to the existing results in [BDE$^+$16, EBD19] as well as results for our own algorithms.
- Chapter 9 concludes by discussing the potential extensions and future work.

1.2.1 CAD, Procedural Steps

We first describe the steps involved in producing a CAD of $\mathbb{R}^n$. The CAD procedure input is a set of polynomial constraints in $\mathbb{R}[x_1, \ldots, x_n]$.

- **Variable Ordering:** When computing a CAD we need a fixed variable ordering. This determines the order in which variables are eliminated. In quantifier elimination problems we are restricted to the given ordering, but in general for we assume the variable ordering to be $x_1 < \ldots < x_n$.
- **Projection Phase:** This phase uses a function, known as the projection operator, that reduces the number of variables of the polynomial constraints by one. The projection operator is used recursively until it obtains polynomials in $\mathbb{R}[x_1]$.
- **Base Phase:** $\mathbb{R}^1$ is decomposed to the specifications of the required CAD and assigning sample points to the cells.
- **Lifting Phase:** This phase consists of lifting the decomposition of $\mathbb{R}^1$ to $\mathbb{R}^n$ and the projected polynomials obtained in the projection phase.

Most of the work presented in this thesis is based on proposing different modifications to Lazard's projection operator. We also provide the appropriate lifting algorithms



to complement the modified projection operators. We end this work by a thorough complexity analysis of these algorithms to justify our modifications.

1.3 Author's Contribution and Collaborators

The work presented in this thesis is theoretical and developed by the author (supervised by Prof. James H. Davenport and Prof. Gregory Sankaran). This research was part of a larger project on improving the complexity of Cylindrical Algebraic Decomposition formed by Prof. James Davenport, Prof. Gregory Sankaran, Dr. Russell Bradford, Zak Tonks and the author. There were several discussions held with external collaborator Prof. Scott McCallum (Macquaire University, Australia). The results obtained from these discussions are published as part of conference proceedings for SYNASC (Symposium on Symbolic and Numeric Algorithms for Scientific Computing, Romania) and a poster presentation at ISSAC (International Symposium on Symbolic and Algebraic Computation, China).

The algorithms developed by the author and mentioned in this work have been implemented in Maple by Zak Tonks (supervised by Prof. James Davenport and Dr. Russell Bradford) as part of his PhD [Ton21]. There were several discussions hosted by Dr. Matthew England and Prof. James Davenport (as part of the project Pushing Back the Doubly-Exponential Wall of Cylindrical Algebraic Decomposition), where potential extensions of the research presented in this thesis were discussed.

1.3.1 Funding

The research conducted for this PhD was fully funded thanks to the University of Bath and EPSRC grant: EP/N509589/1.

Chapter 2

Preliminaries

Cylindrical Algebraic Decomposition (CAD) was first introduced by Collins [Col75]. Over the past 45 years, there have been several enhancements to Collins' original algorithm. Our research extends the variants proposed by McCallum [McC84] [McC99] [McC01] and Lazard [Laz94].

This chapter introduces the various mathematical preliminaries required for the validity proofs later. We also establish the standard terminologies in CAD to keep this piece of work self-contained.

2.1 Polynomial Notations and Properties

Here we standardise our notation and review the material from real algebraic geometry that we shall need. We work throughout over the field of real numbers $\mathbb{R}$.

We use the notation $\mathbb{R}[x_1, \ldots, x_n]$ for the ring of polynomials in variables $x_1, \ldots, x_n$. We also follow the convention that 0 is in $\mathbb{N}$.

Definition 1. *For $f \in \mathbb{R}[x_1, \ldots, x_n]$, we denote by $\deg_{x_i}(f)$ the highest power of x_i present in f.*

We often think of a polynomial $f \in \mathbb{R}[x_1, \ldots, x_n]$ as a univariate polynomial in x_n with coefficients in $\mathbb{R}[x_1, \ldots, x_{n-1}]$.

Definition 2. *For a polynomial f in $\mathbb{R}[x_1, \ldots, x_n]$ we define*

- $\deg(f)$*, the* degree *of the polynomial f, where* $\deg(f) = \deg_{x_n}(f)$.

- ldcf(f), *the* leading coefficient *of* f*, i.e. the coefficient of* x_n^d *where* $d = \deg_{x_i}(f)$.
- trcf(f), *the* trailing coefficient *of* f*, i.e. the coefficient of the least exponent with non-zero coefficient of* x_n.
- ldt(f)*, the non-zero term with the greatest exponent of* x_n.
- trt(f)*, the non-zero term with the least exponent of* x_n.
- ldm(f), *the* leading monomial *of* f*, where* $\operatorname{ldm}(f) = \operatorname{ldt}(f)/\operatorname{ldcf}(f)$.

Definition 3. *Let* $f \in \mathbb{R}[x_1, \dots, x_n]$ *with main variable* x_n*. Then* f *is a* monic polynomial *if the* $\operatorname{ldcf}(f) = 1$ *with respect to the main variable* x_n*. We say that* f *is a* primitive polynomial *if the gcd of all the coefficients (with respect to main variable* x_n*) is 1.*

Definition 4. *Let* f *be a polynomial in* $\mathbb{R}[x_1, \dots, x_n]$*. We define* red$(f)$*, the* reductum *of* f *as follows*

$$\operatorname{red}(f) = f - \operatorname{ldt}(f).$$

By convention $\operatorname{red}(0) = 0$*. We define the* k^{th} reductum *recursively:*

$$\operatorname{red}^0(f) = f.$$
$$\operatorname{red}^k(f) = \operatorname{red}^{k-1}(f) - \operatorname{ldt}(\operatorname{red}^{k-1}(f)).$$

We define RED(f) *as the* reducta *set of* f *as follows*

$$\operatorname{RED}(f) = \{\operatorname{red}^k(f) \mid 0 \leq k \leq \deg(f), \operatorname{red}^k(f) \neq 0\} \tag{2.1}$$

We are mainly concerned with ldcf(f) and trcf(f), but have included this terminology to keep the work self-contained.

Let $\alpha = (\alpha_1, \dots, \alpha_n) \in \mathbb{R}^n$. A *formal power series* about α over $\mathbb{R}$ is an expression of the form

$$\sum_{i_1,\dots,i_n=0}^{\infty} a_{i_1,\dots,i_n}(x_1 - \alpha_1)^{i_1} \dots (x_n - \alpha_n)^{i_n}. \tag{2.2}$$

Let U be an open subset of $\mathbb{R}^n$ and let $f : U \to \mathbb{R}$ be a function. Then f is said to be *analytic* in U if each point α in U has a neighbourhood $W \subseteq U$ such that f has a power series about α over $\mathbb{R}$

$$f(x_1, \dots, x_n) = \sum_{i_1,\dots,i_n=0}^{\infty} a_{i_1,\dots,i_n}(x_1 - \alpha_1)^{i_1} \dots (x_n - \alpha_n)^{i_n}, \tag{2.3}$$

which is absolutely convergent for every point x in W.

Theorem 1. *[McC85, Theorem 2.1.3] Let U be a connected open subset in $\mathbb{R}^n$ and let $f, g \in \mathbb{R}[x_1, \ldots, x_n]$ be analytic in U. Then*

- *The functions $f + g$ and fg (defined pointwise) are analytic in U.*
- *If $f(x) \neq 0$ for all $x \in U$, then $1/f$ is analytic in U.*
- *If $f(x) = 0$ for all points x in a nonempty open subset D of U, then $f = 0$ everywhere in U.*

The following result states that even if a polynomial cannot be factored over the ring of polynomials, it is possible to factor it locally over $\mathbb{R}[[x_1, \ldots x_n]]$, the ring of formal power series. We use this result in our validity proof that Lazard's theory can be modified to exploit the single equational constraint case.

Theorem 2. *(Hensel's Lemma)[Abh90, Page 90, Lecture 12] Let $F(x_1, \ldots, x_m, y) \in \mathbb{R}[[x_1, \ldots, x_m]][y]$ be a monic polynomial of degree $n > 0$ in y with coefficients $a_1, \ldots, a_n \in \mathbb{R}[[x_1, \ldots, x_m]]$ i.e.*

$$F(x_1, \ldots, x_m, y) = y^n + a_1(x_1, \ldots, x_m)y^{n-1} + \ldots + a_n(x_1, \ldots, x_m) \in \mathbb{R}[[x_1, \ldots, x_m]][y]$$

Assume that $F(0^m, y) = \bar{G}(y)\bar{H}(y)$ where

$$\begin{aligned} \bar{G}(y) &= y^r + \bar{b}_1 y^{r-1} + \ldots + \bar{b}_r \in \mathbb{R}[y] \\ \textit{and } \bar{H}(y) &= y^s + \bar{c}_1 y^{s-1} + \ldots + \bar{c}_s \in \mathbb{R}[y] \end{aligned}$$

are monic polynomials in y with real coefficients of degrees $r > 0$ and $s > 0$ respectively such that $\gcd(\bar{G}(y), \bar{H}(y)) = 1$. Then there exist unique monic polynomials over $\mathbb{R}[[x_1, \ldots, x_m]]$

$$\begin{aligned} G(x_1, \ldots, x_m, y) &= y^r + b_1(x_1, \ldots, x_m)y^{r-1} + \ldots + b_r(x_1, \ldots, x_m) \\ H(x_1, \ldots, x_m, y) &= y^s + c_1(x_1, \ldots, x_m)y^{s-1} + \ldots + c_s(x_1, \ldots, x_m) \end{aligned}$$

such that

- *$G(0^m, y) = \bar{G}(y), H(0^m, y) = \bar{H}(y)$*
- *$F(x_1, \ldots, x_m, y) = G(x_1, \ldots, x_m, y)H(x_1, \ldots, x_m, y)$.*

Definition 5. *A* valuation *is any map $\nu : \mathbb{R}[x_1, \ldots, x_n] \to X$ (where X is a set with a suitable total ordering) that satisfies the following properties:*

- *$\nu(a) = \infty$ if and only if $a = 0$*
- *$\nu(ab) = \nu(a) + \nu(b)$*

- $\nu(a+b) \geq \min(\nu(a), \nu(b))$, *with equality if* $\nu(a) \neq \nu(b)$

We say that a set of polynomials A is *valuation invariant* in a set S, if the valuation of every polynomial in A is constant throughout S.

2.2 Semi-Algebraic sets and Polynomial Ideals

Definition 6. *Let* $f : \mathbb{R}^n \to \mathbb{R}^{n-1}$ *be the standard projection map. Then the* fibre *of* $b \in \mathbb{R}^{n-1}$ *is the set* $f^{-1}(b) = \{a \in \mathbb{R}^n \mid f(a) = b\}$.

Definition 7. *Let* f *be a polynomial in* $\mathbb{R}[x_1, \ldots, x_n]$. *We define* V_f, *the* zero-set *of* f, *by*

$$V_f = \{\alpha \mid \alpha \in \mathbb{R}, f(\alpha) = 0\}.$$

We also call V_f *the* hypersurface *described by* f.

Definition 8. *Let* F *be a finite set of polynomials in* $\mathbb{R}[x_1, \ldots, x_n]$. *We define*

$$V_F = \{\alpha \mid \alpha \in \mathbb{R}, \forall f \in F \ f(\alpha) = 0\}$$

Definition 9. *Let* Z *be a subset of* $\mathbb{R}^n$. *We say that* Z *is an* algebraic set *if there exists a set of polynomials* F *in* $\mathbb{R}[x_1, \ldots, x_n]$ *such that* $Z = V(F)$.

In view of Hilbert's basis theorem, we can take the set F in Definition 9 to be finite.

Definition 10. *Let* Z *be a subset of* $\mathbb{R}^n$. *We define* $I(Z)$ *as the* ideal of polynomials vanishing on Z *as*

$$I(Z) := \{f \in \mathbb{R}[x_1, \ldots, x_n] \mid \forall x \in Z \ f(x) = 0\}$$

It is easy to see that $I(Z)$ is an ideal of the polynomial ring $\mathbb{R}[x_1, \ldots, x_n]$. So far we have not used any special properties of $\mathbb{R}$, but the next proposition is not valid over most other fields.

Proposition 1. *Let* Z *be an algebraic set in* $\mathbb{R}^n$. *Then there exists a polynomial* f *in* $\mathbb{R}[x_1, \ldots, x_n]$ *such that* $Z = V_f$.

Proof. Let $\{f_1, \ldots, f_k\}$ be generators of $I(Z)$ (by Hilbert's Basis theorem we may assume k is finite). Set $f = f_1^2 + \ldots + f_k^2$ and let $\alpha \in Z$. Hence $f_1(\alpha) = 0, \ldots, f_k(\alpha) = 0$. This implies that $f(\alpha) = 0$, thus by Definition 7 $\alpha \in V(f)$. Hence $Z \subseteq V(f)$. Since Z is an algebraic set, by Definition 9 there exist polynomials $g_1, \ldots, g_m$ in

$\mathbb{R}[x_1, \ldots, x_n]$ such that $Z = V(g_1, \ldots, g_m)$. This implies that $\{g_1, \ldots, g_m\} \subseteq I(Z)$. Since $f_1, \ldots, f_k$ are the generators of $I(Z)$, if α is a root of $f_1, \ldots, f_k$ then α is also a root of $g_1, \ldots, g_m$, implying that $\alpha \in Z$. Therefore $V(f) \subseteq Z$. $\square$

Definition 11. *[BCR98] A subset S of $\mathbb{R}^n$ is called a* semi-algebraic *subset if it can be written as a union of the sets of the form*

$$\{x \in \mathbb{R}^n \mid f_1(x) = \ldots = f_l(x),\ g_1(x) > 0, \ldots, g_m(x) > 0\}$$

where $f_1, \ldots, f_l, g_1, \ldots, g_m$ are in $\mathbb{R}[x_1, \ldots, x_n]$. It can also be written in the following form

$$\bigcup_i \left(\bigcap_j f_{i,j} \sim_{i,j} 0 \right)$$

where $\sim_{i,j}$ is either $>$ or $=$.

Example 1. *Figure 2-1 is a semi algebraic set and can be defined as follows.*

$$\begin{aligned}
&\Bigg[(y < 3) \bigcap \left(x^2 + \left(y - \frac{9}{2}\right)^2 > \frac{25}{4} \right) \bigcap \left(\left(\frac{x+6}{3}\right)^2 + \left(\frac{y-1}{2}\right)^2 > 1 \right) \\
&\quad \bigcap \left(\left(\frac{x+3}{3}\right)^2 + y^2 > 1 \right) \bigcap \left(\left(\frac{x-6}{3}\right)^2 + \left(\frac{y-1}{2}\right)^2 > 1 \right) \\
&\quad \bigcap \left(\left(\frac{x-3}{3}\right)^2 + y^2 > 1 \right) \Bigg] \\
&\bigcup \left[\left(x^2 + \left(y - \frac{9}{2}\right)^2 \leq \frac{25}{4} \right) \bigcap (y \leq 3) \bigcap \left(x \leq \frac{1}{2} \right) \bigcap \left(y - 2 < \frac{7}{5}x \right) \right] \\
&\bigcup \left[\left(x^2 + \left(y - \frac{9}{2}\right)^2 \leq \frac{25}{4} \right) \bigcap \left(x \geq -\frac{1}{2} \right) \bigcap (y \leq 3) \bigcap \left(y - 2 < \frac{7}{5}x \right) \right] \\
&\bigcup \left[\left(y \leq \frac{12}{5} \right) \bigcap \left(y - 2 \geq -\frac{7}{5}x \right) \bigcap \left(y - 2 \geq \frac{7}{5}x \right) \right]
\end{aligned}$$

Theorem 3. *[BCR98] Let S be a semi-algebraic subset of $\mathbb{R}^n$ and $\pi : \mathbb{R}^n \to \mathbb{R}^{n-1}$ the projection on the first $n-1$ coordinates. Then $\pi(S)$ is a semi-algebraic set.*

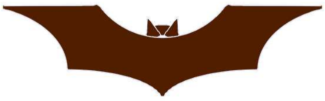


Figure 2-1: Batman is semi-algebraic!

2.2.1 Squarefree and Irreducible Bases

Both squarefree and irreducible bases are used in the implementations of CAD. To understand the difference between then, we have a detailed look at them here.

Definition 12. *Let* $f \in \mathbb{R}[x_1, \ldots, x_n]$ *with positive degree. If there exist polynomials* g *and* h *in* $\mathbb{R}[x_1, \ldots, x_n]$ *with positive degrees such that* $f = g \cdot h$*, then* f *is a* reducible polynomial*. If* f *cannot be expressed as such a product, then* f *is an* irreducible polynomial.

Definition 13. *[Wil14, Definition 2.3] Let* f *be a polynomial in* $\mathbb{R}[x_1, \ldots, x_n]$*. The* squarefree decomposition *of* f *is the decomposition*

$$f = \prod_{i=1}^{k} f_i^i$$

such that $f_i \in \mathbb{R}[x_1, \ldots, x_n]$ *and* $f_1, \ldots, f_k$ *are relatively prime and have no repeated factors.*

We say that $f \in \mathbb{R}[x_1, \ldots, x_n]$ is squarefree if $k = 1$ in its squarefree decomposition.

Definition 14. *[Col75, Page 145] Let* $A \subseteq \mathbb{R}[x_1, \ldots, x_n]$ *be a set of primitive polynomials of positive degrees.* A basis *for* A *is a set* B *of primitive polynomials of positive degree in* $\mathbb{R}[x_1, \ldots, x_n]$ *such that*

- *If* $b_1, b_2 \in B$*, then* $\gcd(b_1, b_2) = 1$*.*
- *If* $b \in B$*, then* $b \mid a$ *for some* $a \in A$*.*
- *If* $a \in A$*, there exists* $b_1, \ldots, b_n \in B$ *and positive integers* $e_1, \ldots, e_n$ *such that*

$$a = \prod_{i=1}^{n} b_i^{e_i}$$

with $n = 0$ *if* $a = 1$*.*

Definition 15. *Let A be a set of primitive polynomials in $\mathbb{R}[x_1, \ldots, x_n]$ of positive degrees and B be a basis of A. If all the elements of B are squarefree then B is a* squarefree basis *of A. If all the elements of B are irreducible then B is an* irreducible basis *of A.*

2.3 Resultants and Discriminants

This section summarises the properties of resultants and discriminants of polynomials. These results will be useful throughout this thesis but mainly in Chapter 8, when computing the complexity of various CAD algorithms. We have chosen to omit the proofs for these results as they are standard results and can be found in various sources such as [Col71], [Dav21], [GKZ94],[McC84], [Abh90] etc.

Definition 16. *[Col71] Let $f = \sum_{i=0}^{d} a_i x_n^i$ and $g = \sum_{j=0}^{e} b_j x_n^j$ with $d, e \geq 0$ and coefficients $a_i, b_i \in \mathbb{R}[x_1, \ldots, x_{n-1}]$. Then the* Sylvester matrix $\mathrm{Syl}_{x_n}(f, g)$ *is defined as follows:*

$$\mathrm{Syl}_{x_n}(f,g) = \underbrace{\begin{bmatrix} a_d & \cdots & \cdots & a_0 & & \\ & \ddots & & & \ddots & \\ & & a_d & \cdots & \cdots & a_0 \\ b_e & \cdots & \cdots & b_0 & & \\ & \ddots & & & \ddots & \\ & & b_e & \cdots & \cdots & b_0 \end{bmatrix}}_{d+e} \begin{matrix} \left.\vphantom{\begin{matrix}a\\a\\a\end{matrix}}\right\} e \\ \left.\vphantom{\begin{matrix}a\\a\\a\end{matrix}}\right\} d \end{matrix}$$

Definition 17. *[Col71] Let $f = \sum_{i=0}^{d} a_i x_n^i$ and $g = \sum_{j=0}^{e} b_j x_n^j$ with $d, e \geq 0$ and coefficients $a_i, b_i \in \mathbb{R}[x_1, \ldots, x_{n-1}]$. Then $\mathrm{res}_{x_n}(f, g)$, the resultant of f and g with respect to the variable x_n, is defined as*

$$\mathrm{res}_{x_n}(f, g) = \det(\mathrm{Syl}_{x_n}(f, g))$$

We have used Syl_{x_n} instead of just Syl, since our polynomials are multivariate and we want to emphasise the main variable.

Let $F, G \subseteq \mathbb{R}[x_1, \ldots, x_n]$. We define $\mathrm{res}_{x_n}(F, G) = \{\mathrm{res}_{x_n}(f, g) \mid f \in F, g \in G\}$.

Theorem 4. *[Col71, Theorem 2] Let $f = \sum_{i=0}^{d} a_i x_n^i$ and $g = \sum_{j=0}^{e} b_j x_n^j$ with $d, e \geq 0$ and coefficients $a_i, b_i \in \mathbb{R}[x_1, \ldots, x_{n-1}]$. Then $\mathrm{res}_{x_n}(f, g) = 0$ if and only if f and g have a common divisor of positive degree.*

Theorem 5. *[Col71, Theorem 3] Let* $f = \sum_{i=0}^{d} a_i x_n^i$ *and* $g = \sum_{j=0}^{e} b_j x_n^j$ *with* $d, e \geq 0$ *and coefficients* $a_i, b_i \in \mathbb{R}[x_1, \ldots, x_{n-1}]$*. Then there exist polynomials* $S, T \in \mathbb{R}[x_1, \ldots, x_n]$ *such that* $Sf + Tg = \operatorname{res}_{x_n}(f, g)$*,* $\deg_{x_n}(T) < d$ *and* $\deg_{x_n}(S) < e$*.*

Theorem 6. *[Col71, Theorem 5] Let* $f = \sum_{i=0}^{d} a_i x_n^i$ *and* $g = \sum_{j=0}^{e} b_j x_n^j$ *with* $d, e \geq 0$ *and coefficients* $a_i, b_i \in \mathbb{R}[x_1, \ldots, x_{n-1}]$*. Let* $\operatorname{res}_{x_n}(f, g) = h(x_1, \ldots, x_{n-1})$*. If* $(\alpha_1, \ldots, \alpha_n) \in \mathbb{R}^n$ *is a common root for* f *and* g *then* $h(\alpha_1, \ldots, \alpha_{n-1}) = 0$*. Conversely, if* $h(\alpha_1, \ldots, \alpha_{n-1}) = 0$ *then at least one of the following holds:*

- $a_d(\alpha_1, \ldots, \alpha_{n-1}) = \ldots = a_0(\alpha_1, \ldots, \alpha_{n-1}) = 0,$
- $b_e(\alpha_1, \ldots, \alpha_{n-1}) = \ldots = b_0(\alpha_1, \ldots, \alpha_{n-1}) = 0,$
- $a_d(\alpha_1, \ldots, \alpha_{n-1}) = b_e(\alpha_1, \ldots, \alpha_{n-1}) = 0,$
- *For some* $\alpha_n \in \mathbb{R}$*,* $(\alpha_1, \ldots, \alpha_n)$ *is a common root for* f *and* g*.*

Corollary 1. *Let* f, g *be two polynomials in* $\mathbb{R}[x_1, \ldots, x_n]$ *with positive degree and leading coefficients that do not vanish simultaneously over a set* $S \subseteq \mathbb{R}^{n-1}$*. Then* f *and* g *intersect over the set given by* $\{\operatorname{res}_{x_n}(f, g) = 0\} \cap S$*.*

Theorem 4 and Corollary 1 help establish the geometric relationship between the hypersurfaces described by two polynomials. Notably all projection operators for CAD algorithms use the resultants to obtain information about the intersection of two constraints.

Definition 18. *[Dav21, Section A.1.2] Let* $f \in \mathbb{R}[x_1, \ldots, x_n]$ *with positive degree* d *in* x_n *and let* f' *be the partial derivative of* f *with respect to* x_n*. Then* $\operatorname{disc}_{x_n}(f)$*, the* discriminant *of* f *with respect to the variable* x_n*, is defined as*

$$\operatorname{disc}_{x_n}(f) = (-1)^{\frac{d(d-1)}{2}} \operatorname{res}_{x_n}(f, f'). \tag{2.4}$$

Theorem 7. *[Dav21, Corollary 23] Let* $f = \sum_{i=0}^{d} a_i x_n^i$ *and* $g = \sum_{j=0}^{e} b_j x_n^j$ *with degree* $d, e \geq 1$*. Then*

$$\operatorname{disc}_{x_n}(fg) = \operatorname{disc}_{x_n}(f) \operatorname{disc}_{x_n}(g) (\operatorname{res}_{x_n}(f, g))^2. \tag{2.5}$$

Theorem 8. *[Dav21, Corollary 22] Let* $f, g, h \in \mathbb{R}[x_1, \ldots, x_n]$ *with positive degrees in* x_n*. Then*

$$\operatorname{res}_{x_n}(fg, h) = \operatorname{res}_{x_n}(f, h) \operatorname{res}_{x_n}(g, h). \tag{2.6}$$

Theorem 9. *[Dav21, Lemma 19] Let $f, g \in \mathbb{R}[x_1, \ldots, x_n]$ with positive degrees d, e in the variable x_n respectively. Then $\operatorname{res}_{x_n}(f, g)$ has a max degree of de in the variable x_{n-1}.*

Let $\operatorname{Syl}_{i,j}(f, g)$ be defined like the Sylvester matrix of f and g but removing the last j rows of the coefficients of f and g, and the last $2j - 1$ columns except the $m + n - 2j$ column, where m and n are the degrees of f and g respectively.

Definition 19. *We define the j^{th}* principal subresultant coefficient *of f and g as $\operatorname{psc}_j(f, g) = \det(\operatorname{Syl}_{j,j}(f, g))$.*

Definition 18, Theorem 7, Theorem 8 and Theorem 9 are used extensively in Chapter 8 to find an upper bound for the number of cells produced by CAD algorithms. For an in-depth understanding on resultants and discriminants look at [Dav21].

2.4 Quantified Formulae and Quantifier Elimination

In this section we present a few standard definitions in logic.

A *standard atomic formula* is of the form

$$f(x_1, \ldots, x_n) \sim 0$$

where $\sim$ is one of $\{=, <, >, \neq, \geq, \leq\}$ and f is some polynomial in $\mathbb{R}[x_1, \ldots, x_n]$.

Definition 20. *A* standard formula *is any formula which can be constructed using standard atomic formulae using propositional connectives $(\neg, \wedge, \vee, \rightarrow, \leftrightarrow)$ and quantifiers on variables $(\exists x_i, \forall x_i)$.*

A *quantifier free formula* (QFF) is a standard formula that does not contain any quantifiers on variables.

A *Tarski formula* or *standard prenex formula* is a standard formula of the following form

$$\Phi = (Q_k x_k)(Q_{k+1} x_{k+1}) \ldots (Q_n x_n) \phi(x_1, \ldots, x_n)$$

where $0 \leq k \leq n$, the Q_i are quantifiers and ϕ is a quantifier free formula.

The problem of quantifier elimination is the following.

Given a Tarski formula

$$\Phi = (Q_k x_k)(Q_{k+1} x_{k+1}) \ldots (Q_n x_n) \phi(x_1, \ldots, x_n)$$

where $0 \leq k \leq n$ and ϕ is a quantifier free formula, does there exist a quantifier free formula in the free variables, $\Psi(x_1, \ldots, x_{k-1})$, which is equivalent to Φ?

$$(\forall x_1)(\forall x_2)\ldots(\forall x_{k-1})[\Phi = \Psi]$$

2.5 Cylindrical Algebraic Decomposition

We define a i-cell (for $0 \leq i \leq n$) as a subset of $\mathbb{R}^n$ that is homeomorphic to $\mathbb{R}^i$. For any subset $X \subseteq \mathbb{R}^n$, a partition of X is a set of non empty sets $\{X_1, \ldots, X_k\}$ such that $\cup_i X_i = X$ and $X_i \cap X_j = \emptyset$ for all $i \neq j$.

Let f be a real valued continuous function on $S \in \mathbb{R}^{n-1}$. The f-section of $S \times \mathbb{R}$ is the set of points

$$\{(\alpha, f(\alpha)) \mid \alpha \in S\}$$

and any set of this form is called a *section*.

Let f and g be two real valued continuous functions on $S \subseteq \mathbb{R}^{n-1}$ (allowing the constant function $f = -\infty$ and $g = \infty$) such that $f(\alpha) < g(\alpha)$ for all $\alpha \in S$. We define the (f, g)-*sector* of $S \times \mathbb{R}$ as the set of points

$$\{(\alpha, b) \mid f(\alpha) < b < g(\alpha) \text{ and } b \in \mathbb{R}\}$$

and any set of this form is called a *sector*.

Definition 21. *Let f be a polynomial in $\mathbb{R}[x_1, \ldots, x_n]$ and $S \subseteq \mathbb{R}^{n-1}$. Then f is delineable on S if $V_f \cap (S \times \mathbb{R})$ consists of k disjoint sections of $S \times \mathbb{R}$ with $k \geq 0$. These sections are often refered to as the f-*sections *of $S \times \mathbb{R}$.*

Let $\theta_1 < \theta_2 < \ldots < \theta_k$ be a collection of continuous real valued functions defined over $S \subseteq \mathbb{R}^{n-1}$. The set $S \times \mathbb{R}$ has a natural decomposition:

- the θ_i-sections of $S \times \mathbb{R}$ for $1 \leq i \leq k$, together with
- the (θ_i, θ_{i+1})-sectors of $S \times \mathbb{R}$ for $0 \leq i \leq k$, where $\theta_0 = -\infty$ and $\theta_{k+1} = +\infty$.

We now define the basic terminologies of CADs. These are versions that have been adapted over the years from the orginal definitions found in [Col75].

Definition 22. *A* decomposition *of $X \subseteq \mathbb{R}^n$ is called* algebraic *if every set of the partition describing the decomposition is semi-algebraic.*

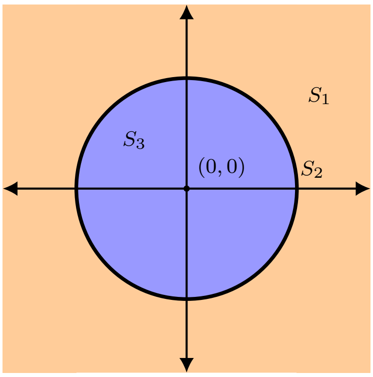


Figure 2-2: Algebraic decomposition of $\mathbb{R}^2$

Let us consider the following decomposition of $\mathbb{R}^2$ with respect to the polynomial $x^2 + y^2 - 1$. In Figure 2-2, $\mathbb{R}^2$ is partitioned into the following three sets:

$$\begin{aligned} &S_1 : x^2 + y^2 - 1 > 0, \\ &S_2 : x^2 + y^2 - 1 = 0, \\ &S_3 : x^2 + y^2 - 1 < 0. \end{aligned}$$

Note, all of these three regions are semi-algebraic sets. Hence this decomposition of $\mathbb{R}^2$ in Figure 2-2 with respect to the polynomial $x^2 + y^2 - 1$ is algebraic.

Definition 23. *Let D be a decomposition of $\mathbb{R}^n$. D is said to be* cylindrical *if the projection of any two cells onto the first k variables (with $1 \leq k < n$) is either the same or disjoint.*

Definition 24. *A decomposition of $\mathbb{R}^n$ is a* Cylindrical Algebraic Decomposition *(CAD) if it is both cylindrical and algebraic.*

The decomposition of $\mathbb{R}^2$ in Figure 2-2 is not cylindrical. The projection of S_1 onto the x_1 variable is the set $(-\infty, \infty)$ and the projection of S_2 is $[-1, 1]$. The intersection of these two projections is a non-empty intersection. However this is not the case in the decomposition of $\mathbb{R}^2$ in Figure 2-3. In Figure 2-3, $\mathbb{R}^2$ is decomposed into the

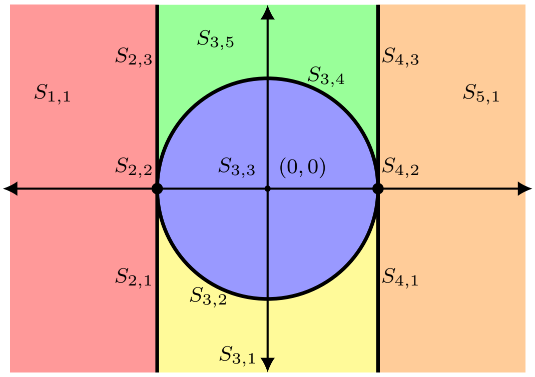


Figure 2-3: A Cylindrical Algebraic Decomposition of $\mathbb{R}^2$

following sets:

$$\begin{aligned}
&S_{1,1} = \{x < -1\}, \quad S_{5,1} = \{x > 1\},\\
&S_{2,1} = \{x = -1, y < 0\}, \quad S_{2,2} = \{x = -1, y = 0\}, \quad S_{2,3} = \{x = -1, y > 0\},\\
&S_{3,1} = \{x^2 + y^2 - 1 > 0 \wedge 1 > x > -1 \wedge y < 0\},\\
&S_{3,2} = \{x^2 + y^2 - 1 = 0 \wedge 1 > x > -1 \wedge y < 0\},\\
&S_{3,3} = \{x^2 + y^2 - 1 < 0 \wedge 1 > x > -1\},\\
&S_{3,4} = \{x^2 + y^2 - 1 = 0 \wedge 1 > x > -1 \wedge y > 0\},\\
&S_{3,5} = \{x^2 + y^2 - 1 > 0 \wedge 1 > x > -1 \wedge y > 0\},\\
&S_{4,1} = \{x = 1, y < 0\}, \quad S_{4,2} = \{x = 1, y = 0\}, \quad S_{4,3} = \{x = 1, y > 0\}.
\end{aligned}$$

Definition 25. *Let f be a polynomial in $\mathbb{R}[x_1, \ldots, x_n]$. We say f is* sign invariant *in a set $S \subset \mathbb{R}^n$ if one of the following is true*

- *$f(x) > 0$ for all $x \in S$.*
- *$f(x) = 0$ for all $x \in S$.*
- *$f(x) < 0$ for all $x \in S$.*

Definition 26. *Let D be a CAD of $\mathbb{R}^n$ and let $F \subset \mathbb{R}[x_1, \ldots, x_n]$. We say D is a* sign invariant CAD (or F-invariant) *if every polynomial in F is sign invariant in every cell of D.*

2.6 The first CAD algorithm

Collins was the first to propose an algorithm for computation of a CAD. Most algorithms for CAD computation are directly or indirectly based on Collins' work. Any such method for CAD computation consists of the following three items:

Projection Operator: This is a function $P : \mathbb{R}[x_1, \ldots, x_k] \to \mathbb{R}[x_1, \ldots, x_{k-1}]$ where $k \geq 2$.

Lifting Theorem: This theorem validates the use of the projection operator P recursively: i.e. for a given $A \subseteq \mathbb{R}[x_1, \ldots, x_k]$, if $P(A)$ is valuation invariant over $S \subset \mathbb{R}^{k-1}$, then A is valuation invariant in every section and sector of A over S.

CAD algorithm: CAD algorithms mainly consist of three phases :

- **Projection phase**: This phase consists of reducing the number of variables of the polynomial constraints (using a function known as the projection operator) until it has reached polynomials in one variable ($\mathbb{R}[x_1]$). The variable ordering is important (generally $x_1 > x_2 > \ldots > x_n$) i.e. the first variable eliminated is x_n.
- **Base phase**: Decompose $\mathbb{R}^1$ according to specifications of the required CAD. Each cell is given a sample point, which is used for tracking cells in the lifting phase.
- **Lifting phase**: This phase consists of constructing a decomposition of $\mathbb{R}[x_1, \ldots, x_k]$ from $\mathbb{R}[x_1, \ldots, x_{k-1}]$, until we obtain a decomposition of S.

  Collins' method produces a sign invariant CAD of $\mathbb{R}^n$. The following describes his projection operator.

Definition 27. *[Col75, Page 142] Let A be a set of polynomials in $\mathbb{R}[x_1, \ldots, x_n]$ and let $B = \{\mathrm{red}_k(f) \mid f \in A$ and $\deg(\mathrm{red}_k(f)) > 0\}$. Then* Collins' projection operator*, $\mathrm{CP}(A)$ consists of the following polynomials:*

- *All leading coefficients of B.*
- *The set $\{\mathrm{psc}_k(f, f') \mid f \in B$ and $0 < k < \deg(f)\}$.*
- *The set $\{\mathrm{psc}_k(f, g) \mid f, g \in B$ and $0 < k < \min(\deg(f), \deg(g))\}$.*

Theorem 10 (Collins lifting theorem)**.** *Let A be a set of non-zero polynomials in $\mathbb{R}[x_1, \ldots, x_n]$ with $n \geq 2$ and let S be a connected subset of $\mathbb{R}^{n-1}$. If every element* *of* CP(A) *is sign invariant in* S*, then the following hold:*

- *Every element of* A *is either delineable or identically zero on* S*.*
- *The product of all the elements of* A *that are not identically zero on* S *is delineable on* S*.*

Algorithm 1 Collins' Algorithm

```
$(F, S) \leftarrow \mathrm{CAD}(r, A, k)$
Input: $A$ is a set of integral polynomials in $n$ variables. $k$ satisfies $0 \leq k \leq n$.
Output: $S$ is a list of sample points for an $A$-invariant CAD of $\mathbb{R}^n$. If $k \geq 1$, $F$ is a list of defining formulas for the induced CAD of $\mathbb{R}^k$, and if $k = 0$, $F$ is the empty list.

1: Set $B \leftarrow$ the squarefree basis for $\mathrm{prim}(A)$. Set $S \leftarrow ()$ and $F \leftarrow ()$.
2: If $n > 1$ then go to step 3. Isolate the real roots of $B$. Construct the sample points for the cells of CAD and add them to $S$. If $k = 1$, then construct the defining formulas for the cells of CAD and add them to $F$. Exit.
3: If $k < r$ then set $P \leftarrow \mathrm{CP}(A)$ and $k' \leftarrow k$; otherwise set $P \leftarrow \mathrm{cont}(A) \cup \mathrm{CP}(A)$ and $k' \leftarrow k - 1$. Call CAD recursively with inputs $n - 1, P$, and $k'$ to obtain $S'$ and $F'$ which specify a $P$-invariant CAD of $\mathbb{R}^{n-1}$.
4: for each cell $c$ do
5:   Let $\alpha$ denote the sample point for $c$.
6:   $B \leftarrow$ the set of all $B_j(\alpha, x_n)$ such that $B_j \in B$ and $B_j(\alpha, x_n) \neq 0$.
7:   isolate the real roots of $B$; use $\alpha$ and isolate the intervals for the roots of $B$ to construct sample points for the $B$-sections and $B$-sectors over $c$, adding them to $S$; if $k = n$, then, from the defining formula for $c$, construct defining formulas for the $B$-sections and $B$-sectors over $c$, adding them to $F$, and if $k < r$, set $F \leftarrow F'$. Exit.
8: end for
```

2.7 McCallum's developments of CAD

McCallum's algorithm was the first adaptation of Collins' theory. The key difference in McCallum's work was that he used the order of polynomials instead of the sign of polynomials.

Definition 28. *Let* f *be a polynomial in* $\mathbb{R}[x_1, \ldots, x_n]$ *and let* $\alpha \in \mathbb{R}^n$*. The* order *of* f *at* α *denoted* $\mathrm{ord}_\alpha(f)$*, is the least* $k \in \mathbb{N}$ *such that some partial derivative of* f *of order* k *does not vanish at* α*.*

Definition 29. *[McC85, Page 44] Let f be a polynomial in $\mathbb{R}[x_1, \ldots, x_n]$ and $W \subset \mathbb{R}^n$. We say that f is* order invariant *on W if either:*

- *$f(\alpha) \neq 0$ for all $\alpha \in W$.*
- *$f(\alpha) = 0$ and $\mathrm{ord}_\alpha(f)$ is the same for all $\alpha \in W$.*

*Let $A \subset \mathbb{R}[x_1, \ldots, x_n]$. Then W is A-*order invariant *if every polynomial in A is order invariant on W. A decomposition of $\mathbb{R}^n$ is A-*order invariant *if every cell of the decomposition is A-order invariant.*

As seen in Definition 30, McCallum's projection operator is a subset of Collins' projection operator. McCallum's approach does however have some limitations, which are discussed further in this section.

Definition 30. *Let A be a finite squarefree basis in $\mathbb{R}[x_1, \ldots, x_n]$ with $r \geq 2$. Then* McCallum's projection operator, *$\mathrm{P}(A)$ consists the following polynomials:*

- *The set of coefficients of all elements of A.*
- *The set of discriminants of all elements of A.*
- *The set of all cross resultants of the elements of A (avoiding the trivial resultants $\mathrm{res}(f, f)$).*

Definition 31. *[McC85] Let A be a set of polynomials in $\mathbb{R}[x_1, \ldots, x_n]$. We define A to be* well-oriented *if no element of a basis for A vanishes identically on any connected set of positive dimension, and the same condition holds recursively for $\mathrm{P}(A)$.*

The idea of well-orientedness is that none of the polynomials nullify during the recursive parts of a CAD algorithm. This is required because if at any stage of the recursion a polynomial constraint nullifies, the algorithm produces an error.

Theorem 11. *[McC85, Theorem 3.2.3] Let A be a finite square free basis in $\mathbb{R}[x_1, \ldots, x_n]$ with $r \geq 2$ and let S be a connected subset of $\mathbb{R}^{n-1}$. Suppose that each element of A is not identically zero on S, and each element of $\mathrm{P}(A)$ is order invariant on S. Then each element of A is degree invariant and delineable on S, and the sections of A over S are pairwise disjoint and every element of A is order invariant in every section of A over S.*

In informal conversations, McCallum suggested that he would prefer computing the irreducible basis of the input polynomials over a squarefree basis. An irreducible basis is finer than a squarefree basis and hence takes a longer time to compute.

Algorithm 2 McCallum's Algorithm

```
$(S) \leftarrow \mathrm{MCAD}(A, r)$
**Input**: $A$ set of well-oriented polynomials in $n$ variables with $r \geq 1$.
**Output**: $S$ is a list of sample points for an $A$-order invariant CAD of $\mathbb{R}^n$.

 1: Set $B \leftarrow$ the finest squarefree basis for $\mathrm{prim}(A)$. Set $S \leftarrow ()$.
 2: If $n > 1$ then go to step 3. Isolate the roots of $B$. Construct sample points for
    the cells of the decomposition of $\mathbb{R}$ and add them to $S$. Return $S$.
 3: Set $P \leftarrow \mathrm{cont}(A) \cup \mathrm{P}(B)$.
 4: $S' \leftarrow \mathrm{MCAD}(P, n-1)$ (where $S'$ is a list of sample points for a smooth, $P$-order
    invariant CAD $D'$ of $\mathbb{R}^{n-1}$).
 5: **for** each cell $c$ in $D'$ **do**
 6:    $\alpha \leftarrow$ the sample point of $c$.
 7:    $\hat{B} \leftarrow$ the delineating set for $B$ over $c$.
 8:    $B^* \leftarrow \{\hat{B}_j(\alpha, x_n) \mid \hat{B}_j \in \hat{B}\}$
 9:    Isolate the real roots of $B^*$. Use $\alpha$ and the isolating intervals for the real roots
       of $B^*$ to construct sample points for the sections and sectors of $\hat{B}$ over $c$, add
       them to $S$.
10: **end for**
11: Return $S$.
```

However, McCallum argued that once an irreducible basis is computed it is much easier to work with when computing CADs.

2.7.1 Single Hypersurface Decomposition

In 1999, McCallum proposed a restricted projection operator that took advantage of equational constraints in the input QFF.

Definition 32. *[EBD19] An* Equational Constraint (EC) *is a polynomial equation logically implied by a QFF. If it is an atom of the formula, it is said to be* explicit*; if not, then it is* implicit. *If the constraint is visibly an equality one from the formula, i.e. the formula Φ is $f = 0 \wedge \Phi'$, we say the constraint is* syntactically explicit.

Although implicit and explicit ECs have the same logical status, in practice only the syntactically explicit ECs will be known to us and therefore be available to be exploited.

Example 2. *[EBD19] Let f and g be two real polynomials.*

1. *The formula $f = 0 \wedge g > 0$ has an explicit EC, $f = 0$.*
2. *The formula $f = 0 \vee g = 0$ has no explicit EC, but the equation $fg = 0$ is an implicit EC.*
3. *The formula $f^2 + g^2 \leq 0$ also has no explicit EC, but it has two implicit ECs: $f = 0$ and $g = 0$.*
4. *The formula $f = 0 \vee f^2+g^2 \leq 0$ logically implies $f = 0$, and the equation is an atom of the formula which makes it an explicit EC according to the definition. However, since this deduction is semantic rather than syntactic, it is more like an implicit EC rather than an explicit EC.*

The main idea behind exploiting equational constraints is that if the input formula is of the form $f = 0 \wedge \phi$, then to find solutions of the formula it is sufficient to decompose V_f rather than the whole of $\mathbb{R}^n$.

Definition 33. *[McC99] Let A be a set of pairwise relatively prime polynomials in $\mathbb{R}[x_1, \ldots, x_n]$ with $n \geq 2$ and let $E \subset A$.* McCallum's restricted projection operator $\mathrm{P}_E(A)$ *is defined as follows:*

$$\mathrm{P}_E(A) = \mathrm{P}(E) \cup \{\mathrm{res}_{x_n}(f, g) \mid f \in E,\ g \in A \setminus E\}. \tag{2.7}$$

Theorem 12. *[McC99, Theorem 2.2] Let $n \geq 2$, and let f, g be two polynomials in $\mathbb{R}[x_1, \ldots, x_n]$ of positive degrees in the main variable x_n and suppose that $\operatorname{res}_{x_n}(f, g) \neq 0$. Let S be a connected subset of $\mathbb{R}^{n-1}$ on which f is delineable and in which $\operatorname{res}_{x_n}(f, g)$ is order invariant. Then g is sign invariant in each section of f over S.*

Theorem 13. *[McC99, Theorem 2.3] Let A be a set of pairwise relatively prime polynomials in $\mathbb{R}[x_1, \ldots, x_n]$, with $n \geq 2$. Let E be a subset of A and let S be a connected subset of $\mathbb{R}^{n-1}$. Suppose that each element of $\mathrm{P}_E(A)$ is order invariant in S. Then the following hold:*

- *Each element of E either vanishes identically on S or is delineable.*
- *The sections over S of the elements of E that do not vanish identically on S are pairwise disjoint. Each element of E is order invariant in every such section. Each element of $A \setminus E$ is sign invariant in every such section.*

2.7.2 Multiple Hypersurface Decomposition

In 2001, McCallum modified his projection operator further so that it can be recursed over multiple equational constraints. Let us assume that the input QFF is of the form

$$(f_1 = 0) \wedge (f_2 = 0) \wedge \ldots \wedge (f_k = 0) \wedge \phi. \tag{2.8}$$

Here the equational constraints are $f_1, \ldots, f_k$. When using this method, we first choose an equational constraint, say f_1. For the next level, the equational constraints are the resultants of f_1 and f_i for all $i \neq 1$.

Definition 34. *[McC01] Let A be a set of pairwise relatively prime polynomials in $\mathbb{R}[x_1, \ldots, x_n]$ with $n \geq 2$ and let $E \subset A$.* McCallum's semi-restricted projection operator $\mathrm{P}^*_E(A)$ *is defined as follows:*

$$\mathrm{P}^*_E(A) = \mathrm{P}(A) \setminus \operatorname{res}(A \setminus E, A \setminus E). \tag{2.9}$$

Theorem 14. *[McC01, Theorem 2.1] Let $n \geq 2$, and let f, g be two polynomials in $\mathbb{R}[x_1, \ldots, x_n]$ of positive degrees in main variable x_n and suppose that $\operatorname{res}_{x_n}(f, g) \neq 0$ and $\operatorname{disc}_{x_n}(g) \neq 0$. Let S be a connected subset of $\mathbb{R}^{n-1}$ on which f is delineable and g does not vanish identically, and in which both $\operatorname{res}_{x_n}(f, g)$ and $\operatorname{disc}_{x_n}(g)$ are order invariant. Then g is order invariant in each section of f over S.*

Theorem 15. *[McC01, Theorem 2.2] Let A be a set of pairwise relatively prime polynomials in $\mathbb{R}[x_1, \ldots, x_n]$, with $n \geq 2$. Let E be a subset of A and let S be a* *connected subset of* $\mathbb{R}^{n-1}$*. Suppose that each element of* $\mathrm{P}^*_E(A)$ *is order invariant in* S*. Then the following hold:*

- *Each element of* E *either vanishes identically on* S *or is delineable.*
- *The sections over* S *of the elements of* A *that do not vanish identically on* S *are pairwise disjoint, and each element of* A *is order invariant in every such section.*

2.7.3 Limitations

Since McCallum's work is based on the order of a polynomial it has some inherent limitations. Because the order of a polynomial is not defined when a polynomial nullifies over a set, McCallum's approach is unable to decompose such regions of a polynomial (we call these regions curtains: see Definition 43). McCallum's further work in [McC99] and [McC01] face the same limitations. In Chapter 5 and Chapter 6 we explore the problems caused by curtains when exploiting equational constraints.

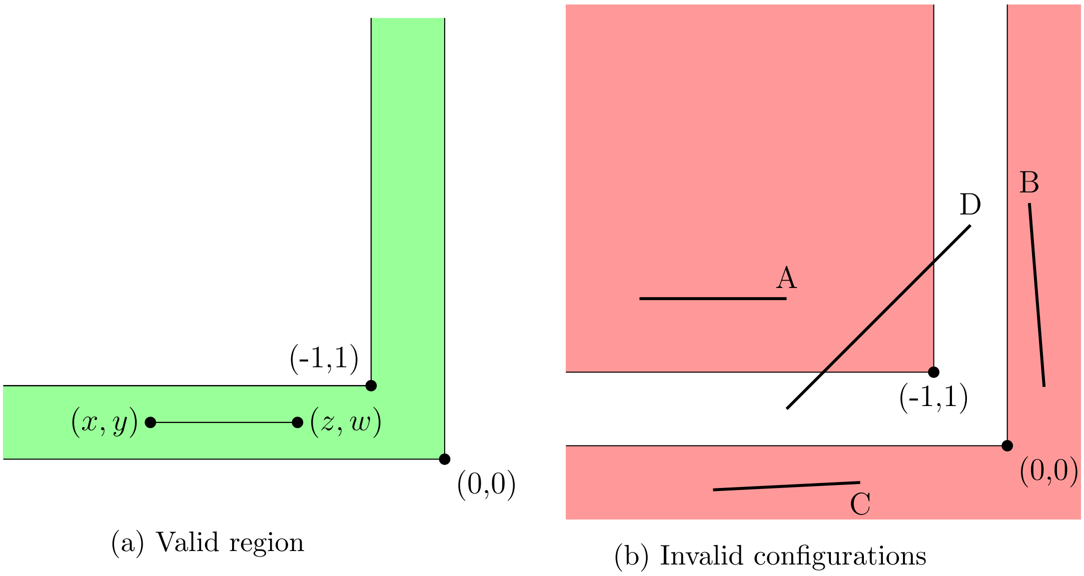
2.8 Application of CAD

In this section we look at the applications of CAD to a motion planning problem, showing how equational constraints can arise. All of the work described in the section is taken from [WDEB13].

Piano mover's problem: *Given a body* B *and a region bounded by a collection of walls, either find a continuous motion connecting two given positions and orientations of* B *during which* B *avoids collisions with the walls, or else establish that no such motion exists.*

For simplicity, let us look at the case of moving a ladder along a right-angled corridor of width 1. Figure 2-4a describes the valid regions where the ladder can be placed and Figure 2-4b describes the configurations the ladder cannot be placed in. In order to find a Tarski formula for the valid region, we first describe the invalid regions using a Tarski formula and then take the negation of it. The invalid region can be described as follows:

- $x < -1 \wedge y > 1$ or $w < -1 \wedge z > 1$: This describes the ladder being on the outside and any collisions with the inside walls.
- $x > 0$ or $w > 0$: This describes the ladder being outside the rightmost walls and any collisions it may have with that wall.



(a) Valid region (b) Invalid configurations

Figure 2-4: Piano mover's problem

- $y < 0$ or $z < 0$: This describes the ladder being below the corridor and any collisions with the bottom-most wall.

- $(\exists t)[0 < t \wedge t < 1 \wedge x + t(w - x) < -1 \wedge y + t(z - y) > 1]$: This describes an inner point of a ladder being outside the corridor when the ends are not.

The invalid regions can be described by the following formula.

$$\begin{aligned}&[x < -1 \wedge y > 1] \vee [w < -1 \wedge z > 1] \vee [x > 0] \vee [w > 0]\\&\vee [y < 0] \vee [z < 0] \vee [(\exists t)[0 < t \wedge t < 1 \wedge x + t(w - x) < -1\\&\wedge y + t(z - y) > 1]]\end{aligned} \tag{2.10}$$

Since there is a quantifier in the formula, before taking the negation we eliminate the quantifier by using *QEPCAD, B* [Bro03]. *QEPCAD, B* is the software created by Brown to eliminate quantifiers to obtain QFF using CAD implementations. Once this is done and we take the negation, we get the following formula describing the valid regions.

$$\begin{aligned}
&[w \leq 0] \wedge [x \leq 0] \wedge [y \geq 0] \wedge [z \geq 0] \wedge [x \geq -1 \vee y \leq 1] \\
&\wedge [w \geq -1 \vee z \leq 1] \\
&\wedge [wy - w + x + y < 0 \vee w + 1 \geq 0 \vee xz + z - yw + w - y - x \leq 0] \\
&\wedge [yw - w + y + x \geq 0 \vee [[z - 1 \leq 0 \vee xz + z - yw + w - y - x \geq 0] \\
&\wedge y - 1 \leq 0]]
\end{aligned} \tag{2.11}$$

Suppose now we are asked the following problem.

Is it possible to move a ladder of length 3 along a right angled corner of width 1?

The problem can be re-formulated for CAD as follows.

$$[(x - w)^2 + (y - z^2) = 9] \wedge (2.11) \tag{2.12}$$

We note that the length of the ladder becomes an equational constraint in this formulation. A more complicated configuration would lead to more equational constraints.

The authors of [WDEB13] used *QEPCAD, B* on 2.12 to obtain a solution space of the configuration space ($\mathbb{R}^4$). *QEPCAD, B* produced a CAD of $\mathbb{R}^4$ with 285,419 cells and gave the following formula as output. Note this is equivalent to 2.12.

$$\begin{aligned}
&x \leq 0 \wedge y \geq 0 \wedge z \geq 0 \wedge (y - z)^2 + (x - w)^2 = 9 \\
&\wedge [[x + 1 \geq 0 \wedge w + 1 \geq 0] \vee [y - 1 \leq 0 \wedge w + 1 \geq 0 \\
&\wedge y^2w^2 - 2yw^2 + xw^2 + 2w^2 - 2xy^2w \\
&+ 4xyw - 2x^3w - 4x^2w - 4xw + x^2y^2 - 2x^2y \\
&x^4 + 2x^3 - 7x^2 - 18x - 9 \geq 0] \\
&\vee [x + 1 \geq 0 \wedge yw - w + y + x \geq 0 \\
&\wedge w^2 - 2xw + y^2 - 2y + x^2 - 8 > 0 \wedge x - 1 \leq 0] \\
&[x + 1 \geq 0 \wedge yw - w + y + x \geq 0 \wedge y^2w^2 - 2yw^2 \\
&+ x^2w^2 + 2xw^2 + 2w^2 - 2xy^2w + 4xyw - 2x^3w \\
&- 4x^2w - 4xw + x^2y^2 - 2x^2y + x^4 + 2x^3 - 7x^2 - 18x - 9 \leq 0 \wedge z - 1 \leq 0] \\
&\vee [y - 1 \leq 0 \wedge z - 1 \leq 0]]
\end{aligned} \tag{2.13}$$

The first line in the formula gives the conditions of the problem which are in conjunction with any valid position for the ladder. The remaining lines are a disjunction of clauses that describe the various valid configurations.

The CAD produced by 2.12 is a decomposition of the configuration space, which provides us information on the existence of a solution, and then the ability to construct a path. To discuss the adjacency of cells in 4-dimensions is not an easy task. However, there have been several attempts to understand the adjacency of cells in higher dimensions such as [BGV13]. Wilson et al. further proceed to compute heuristics for different formulations of 2.12 and various values for lengths to understand the effects of formulation on CAD problems of the same type. Given that all of these methods are based on the computation of a CAD, an improvement in the algorithm for producing a CAD reduces the time it takes to solve the path finding problem.

Chapter 3

A different valuation: Lex-Least

In [Laz94], Lazard established a new method for computing CADs based on a valuation determined by the lexicographic order, which we call lex-least valuation. In Section 3.1.2 we discuss our reasons for preferring this to the term “Lazard valuation” used in [MPP19].

Lazard’s algorithm deals with curtains present in the hypersurfaces described by the constraints. This is because it lifts the projected polynomials by removing factors that form the curtain, discussed further in Section 3.3. Lazard’s algorithm also provides some complexity gain compared with its predecessors.

Unfortunately, in [Col98], Collins observed that Lazard’s proof had some gaps. This finding was disappointing as Lazard’s method provided significant improvements over the existing CAD construction algorithms at the time. However, in 2019 McCallum et al. [MPP19] provided a validity proof for Lazard’s approach to CAD construction. We have modified this method to use it for equational constraints in Chapter 4 and Chapter 6. In this chapter, we focus on Lazard’s original method, as it underlies the contents of Chapters 4 – 8.

3.1 Lex-Least Valuation

We start this chapter by looking at the lexicographic ordering $\geq_{lex}$ as it sits at the core of the lex-least valuation.

Definition 35. *Let $v, w \in \mathbb{Z}^n$. We say that $v = (v_1, \ldots, v_n) \geq_{lex} (w_1, \ldots, w_n) = w$ if and only if either $v = w$ or there exists an $i \leq n$ such that $v_i > w_i$ and $v_k = w_k$*

for all k in the range $1 \leq k < i$.

Definition 36. *[MPP19, Definition 2.4] Let $n \geq 1$ and suppose that $f \in \mathbb{R}[x_1, \ldots, x_n]$ is non-zero and $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{R}^n$. The* lex-least valuation *$\nu_\alpha(f)$ at α is the least (with respect to $\geq_{lex}$) element $v = (v_1, \ldots, v_n) \in \mathbb{N}^n$ such that f expanded about α has the term*

$$c(x_1 - \alpha_1)^{v_1} \cdots (x_n - \alpha_n)^{v_n},$$

where $c \neq 0$.

Note that $\nu_\alpha(f) = (0, \ldots, 0)$ if and only if $f(\alpha) \neq 0$. Lex-least valuation is referred to as the Lazard valuation in [MPP19]. We discuss the change in terminology below, in Section 3.1.2.

Example 3. *If $n = 1$ and $f(x_1) = x_1^3 - 2x_1^2 + x_1$, then $\nu_0(f) = 1$ and $\nu_1(f) = 2$. If $n = 2$ and $f(x_1, x_2) = x_1(x_2 - 1)^2$, then $\nu_{(0,0)}(f) = (1, 0)$, $\nu_{(2,1)}(f) = (0, 2)$ and $\nu_{(0,1)}(f) = (1, 2)$.*

The lex-least valuation is of course strongly dependent on the order of the variables, as the following example illustrates.

Example 4. *Let $f(x, y, z, w) = x^2 + y^2 z - 2yz^2 + zw$, $\alpha_1 = (0, 1, 0, 1)$ and $\alpha_2 = (0, 0, 1, 0)$. With respect to the ordering $x > y > z > w$ we get $\nu_{\alpha_1}(f) = (0, 0, 1, 0)$ and $\nu_{\alpha_2}(f) = (0, 0, 0, 1)$. With respect to the ordering $x > z > y > w$ we get $\nu_{\alpha_1}(f) = (0, 1, 0, 0)$ and $\nu_{\alpha_2}(f) = (0, 0, 0, 1)$. Note that in the case of α_2, the ordering of variables does not change the valuation unlike the case for α_1. Ordering of variables is essential and must be fixed when comparing valuations of points.*

Proposition 2. *ν_α is a valuation: that is, if f and g are non-zero elements of $\mathbb{R}[x_1, \ldots, x_n]$ and $\alpha \in \mathbb{R}^n$, then*

$$\nu_\alpha(fg) = \nu_\alpha(f) + \nu_\alpha(g) \text{ and } \nu_\alpha(f + g) \geq_{lex} \min\{\nu_\alpha(f),\ \nu_\alpha(g)\}.$$

Also $\nu_\alpha(f) = \infty$ if and only if $f = 0$.

The proof of this is quite straightforward and is left for the reader to fill in.

Definition 37. *[MPP19] Let $n \geq 2$, and suppose that $f \in \mathbb{R}[x_1, \ldots, x_n]$ is non-zero and that $\beta \in \mathbb{R}^{n-1}$. The* Lazard residue *$f_\beta \in \mathbb{R}[x_n]$ of f at β, and the* lex-least semi-valuation *$\nu'_\beta(f) = (\nu_1, \ldots, \nu_{n-1})$ of f above β, are defined to be the result of Algorithm 3 (See below).*

The lex-least semi-valuation of f at $\beta \in \mathbb{R}^{n-1}$ must not be confused with the lex-least valuation at $\alpha \in \mathbb{R}^n$, defined in Definition 36. Notice that if $b = (\beta, b_n) \in \mathbb{R}^n$ then

Algorithm 3 Lazard residue

Input: $f \in \mathbb{R}[x_1, \ldots, x_n]$ and $\beta \in \mathbb{R}^{n-1}$.

Output: Lazard residue f_β and lex-least semi-valuation of f above β.

```
$f_\beta \leftarrow f$
for $i \leftarrow 1$ to $n-1$ do
    $\nu_i \leftarrow$ greatest integer $\nu$ such that $(x_i - \beta_i)^\nu | f_\beta$.
    $f_\beta \leftarrow f_\beta / (x_i - \beta_i)^{\nu_i}$.
    $f_\beta \leftarrow f_\beta(\beta_i, x_{i+1}, \ldots, x_n)$
end for
return $f_\beta$, $(\nu_1, \ldots, \nu_{n-1})$
```

$\nu_b(f) = (\nu'_\beta(f), \nu_n)$ for some integer ν_n: in other words, $\nu'_\beta(f)$ consists of the first $n-1$ coordinates of the valuation of f at any point above β.

Note that our terminology differs from the used in [MPP19]. The reason for making these changes is explained further in Section 3.1.2.

Remark 1. *We can use Algorithm 3 to compute the lex-least valuation of f at $\alpha \in \mathbb{R}^n$. After the final loop is finished, we proceed to the first step of the loop and perform it for $i = n$ and the n-tuple $(\nu_1, \ldots, \nu_n)$ is the required valuation.*

Lex-least semi-valuation and Lazard residue are also dependent on variable ordering as demonstrated by the following example.

Example 5. *Let $f(x, y, z, w) = x^2 + y^2 z - 2yz^2 + zw$ and $\beta = (0, 1, 0)$ with ordering $x > y > z > w$ then $\nu'_\beta(f) = (0, 0, 1)$ and $f_\beta = 1 + w$. If we change the ordering to $x > y > w > z$ then $\nu'_\beta(f) = (0, 0, 0)$ and $f_\beta = z - 2z^2$.*

3.1.1 Lazard Delineability

Lazard delineability is a property that helps define Lazard sections of polynomials. The reason why Lazard's algorithm works is because it is decomposing Lazard sections rather than sections of the polynomial constraint (as defined by McCallum in [McC84]).

Definition 38. *[MPP19, Definition 2.10] Let S be a connected subset of $\mathbb{R}^{n-1}$ and $f \in \mathbb{R}[x_1, \ldots, x_n]$. We say that f is* Lazard delineable *on S if:*

i) The lex-least semi-valuation of f at β is the same for each point $\beta \in S$.

ii) There exist finitely many continuous functions $\theta_i \colon S \to \mathbb{R}$, such that $\theta_1(\beta) <$

$\ldots < \theta_k(\beta)$ with $k \geq 0$, and for all $\beta \in S$, the set of real roots of f_β is $\{\theta_1(\beta), \ldots, \theta_k(\beta)\}$.

iii) If $k \geq 1$, then there exist positive integers $m_1, \ldots, m_k$ such that, for all $\beta \in S$ and for all $1 \leq i \leq k$, m_i is the multiplicity of $\theta_i(\beta)$ as a root of f_β.

With the definition of delineability, we are able to define Lazard sections and sectors for polynomials.

Definition 39. *[MPP19, Definition 2.10] Let f be Lazard delineable on $S \subseteq \mathbb{R}^{n-1}$.*

- *i) The graphs θ_i are called* Lazard sections *and m_i is the associated multiplicity of these sections.*
- *ii) The regions between consecutive Lazard sections*[1] *are called* Lazard sectors.

Lazard delineability differs from delineability as in [Col75] and [McC99], both because we require lex-least invariance rather than sign or order invariance, and because we require it on the sections of f_β rather than f. However, if a polynomial f is Lazard delineable on S and the semi-valuation of f for every point in S is the zero vector, in which case Lazard invariance is the same as sign invariance. This means that the Lazard sections of f are the same as the sections of f defined as in [Col75] and [McC99].

For later use, we propose some geometric terminology to describe the conditions under which a polynomial is nullified in the terminology of [McC99].

Let f be a polynomial in $\mathbb{R}[x_1, \ldots, x_n]$ and let $S \in \mathbb{R}^{n-1}$. If f nullifies over S then f has a curtain over S. The formal definition for *curtains* can be found in Chapter 5.

Remark 2. *Note that Lazard delineability is defined for polynomials with curtains, as compared to the delineability in [Col75] and [McC99], which is not.*

3.1.2 New Terminology

In this subsection we explain our preferred notations for lex-least semi-valuation and Lazard residue. We have not changed the definitions themselves, just the terminology. The original terminology is as follows

Definition. *[MPP19] Let $f \in \mathbb{R}[x_1, \ldots, x_n]$, $\alpha \in \mathbb{R}^n$ and $\beta \in \mathbb{R}^{n-1}$. Then*

- *The* Lazard valuation *of f at α is $\nu_\alpha(f) \in \mathbb{N}^n$.*

---

[1] Including $\theta_0 = -\infty$ and $\theta_{k+1} = +\infty$.

- *The* Lazard valuation *of f on/above at β is $\nu_\beta(f) \in \mathbb{N}^{n-1}$.*

- *The* Lazard evaluation *of f at β is $f_\beta \in \mathbb{R}[x_n]$.*

Note that here ν is being used for two functions. When ν is used as a valuation at, it is treated as a function from $\mathbb{R}^n \times \mathbb{R}[x_1, \ldots, x_n] \rightarrow \mathbb{N}^n$. When ν is used as a valuation on/above, it is treated as a function from $\mathbb{R}^{n-1} \times \mathbb{R}[x_1, \ldots, x_n] \rightarrow \mathbb{N}^{n-1}$. We felt that it is necessary to keep these two functions separate. The older notation also has the potential to cause confusion when reading $\nu_\alpha(f)$. The reader needs to check whether $\alpha \in \mathbb{R}^n$ or $\alpha \in \mathbb{R}^{n-1}$, or whether $\nu_\alpha(f) \in \mathbb{N}^n$ or $\nu_\alpha(f) \in \mathbb{N}^{n-1}$.

This is demonstrated with the following two examples, with Example 6 using the original notation and Example 7 using our proposed terminology.

Example 6. *Let $f = x^2 + y^2z - 2yz^2$, $\alpha = (0, 1, 0) \in \mathbb{R}^3$ and $\beta = (0, 1) \in \mathbb{R}^2$. The Lazard valuation of f at α is $\nu_\alpha(f) = (0, 0, 1)$ and the Lazard valuation of f above β is $\nu_\beta(f) = (0, 0)$.*

Example 7. *Let $f = x^2 + y^2z - 2yz^2$, $\alpha = (0, 1, 0) \in \mathbb{R}^3$ and $\beta = (0, 1) \in \mathbb{R}^2$. The valuation of f at α is $\nu_\alpha(f) = (0, 0, 1)$. The lex-least semi-valuation of f at β is $\nu'_\beta(f) = (0, 0)$.*

We also prefer the term 'lex-least valuation' over 'Lazard valuation' because it carries its own definition in the name, and because the originality of Lazard's work lies not in the definition of the valuation but in the use made of it.

3.2 Properties of Lex-least Valuation

The following are the important properties of the lex-least valuation, taken from [MPP19, Section 3]. These properties help establish the relationship between the valuation of polynomials and the geometric aspects of the hypersurfaces.

Lemma 1. *[MPP19] Let $f(x, y) \in \mathbb{R}[x, y]$ be primitive with respect to y and square-free. Then for all but a finite number of points $(\alpha, \beta) \in \mathbb{R}^2$ on the curve $f(x, y) = 0$ we have $\nu_{(\alpha,\beta)}(f) = (0, 1)$.*

Proof. Let $R(x) = \mathrm{res}_x(f, \frac{\partial f}{\partial y})$. Since f is assumed to be square-free, $R(x)$ is not identically zero. Suppose that $(\alpha, \beta) \in \mathbb{R}^2$ where $f(\alpha, \beta) = 0$ and $\nu_{(\alpha,\beta)}(f) \neq (0, 1)$ then $R(\alpha) = 0$ since $f(\alpha, \beta) = 0$. But $R(x)$ has finitely many roots, so there are only finitely many possible values for α. Since f is assumed to be primitive the set of roots of $f(\alpha, y) = 0$ is also a finite set. $\square$

An important property of the lex-least valuation is upper semicontinuity. We believe that this might be an important property that a general valuation must have so that it can be used to obtain CADs. We discuss this further in Chapter 9.

Proposition 3. (Upper semicontinuity) *Let $f \in \mathbb{R}[x_1, \ldots, x_n]$ be non-zero and let $a \in \mathbb{R}^n$. Then there exists an open neighbourhood $U \subset \mathbb{R}^n$ of a, such that $\nu_b(f) \leq_{lex} \nu_a(f)$ for all $b \in U$.*

Proof. The expansion of f about $y = (y_1, \ldots, y_n) \in \mathbb{R}^n$ can be written as

$$f = \sum_{\omega \in \mathbb{N}^n} f_\omega(y)(x_1 - y_1)^{\omega_1} \ldots (x_n - y_n)^{\omega_n},$$

where each $f_\omega(y)$ for fixed ω is a polynomial in $y_1, \ldots, y_n$. Namely,

$$f_\omega(y) = \left. \frac{\partial^{\omega_1 + \ldots + \omega^n} f}{\partial x_1^{\omega_1} \ldots \partial x_n^{\omega^n}} \right|_{x=y}.$$

For any $\nu_0 \in \mathbb{N}$ we define

$$Z(\nu_0) = \{b \in \mathbb{R}^n \mid \nu_b(f) >_{lex} \nu_0\}.$$

We have

$$Z(\nu_0) = \bigcap_{\omega \leq_{lex} \nu_0} \{y \mid f_\omega(y) = 0\},$$

which is closed because $\{\omega \in \mathbb{N}^n \mid \omega \leq_{lex} \nu_0\}$ is finite. $\square$

Proposition 4. *Let f and g be non-zero elements of $\mathbb{R}[x_1, \ldots, x_n]$ and let $X \subset \mathbb{R}^n$ be connected. Then fg is lex-least invariant in X if and only if f and g are lex-least invariant in X.*

Proof. Suppose that fg is lex-least invariant, say $\nu_b(fg) = v$ for all $b \in X$, so $\nu_b(g) = \nu - \nu_b(f)$. If $a \in X$, then by Proposition 3 the set $X_+ = \{b \in X \mid \nu_b(f) \leq_{lex} \nu_a(f)\}$ is open. But

$$\begin{aligned} X_- = \{b \in X \mid \nu_b(f) \geq_{lex} \nu_a(f)\} &= \{b \in X \mid \nu - \nu_b(g) \geq \nu - \nu_a(g)\} \\ &= \{b \in X \mid \nu_b(g) \leq_{lex} \nu_a(g)\} \end{aligned}$$

is also open. Hence $X_+ \cap X_- = \{b \in X \mid \nu_b(f) = \nu_a(f)\}$ is open, for any $a \in X$, so the function $b \mapsto \nu_b(f)$ is a continuous function from X to $\mathbb{Z}^n$. As X is connected, this function is constant, i.e. f is lex-least invariant. The other implication is trivial. □

Proposition 5. *[MPP19, Proposition 2.11] Let $f \in \mathbb{R}[x_1, \ldots, x_n]$ be of positive degree in x_n and let S be a connected subset of $\mathbb{R}^{n-1}$. Suppose that f is Lazard delineable on S. Then f is lex-least invariant in each Lazard section and sector of f over S.*

Proof. We know that $\nu'_\beta(f)$ is the same for all $\beta \in S$. This means that for $\alpha \in \mathbb{R}^n$ in the Lazard sections of f over S, the first $n-1$ coordinates of $\nu_\alpha(f)$ agree with $\nu'_\beta(f)$. Consider a Lazard section given by θ with associated multiplicity m, and $\alpha = (\beta, \theta(\beta))$ the point of this Lazard section above β. Then the last coordinate of $\nu_\alpha(f)$ is m, and hence f is lex-least invariant in every Lazard section of f over S. The same is true for every Lazard sector of f over S. □

3.3 Lifting Algorithm

Lazard introduced a valuation and a reduced projection operator (see Definition 40). This improves the complexity significantly, as the middle coefficients of the polynomial constraints are not needed. His lifting algorithm also uses the lex-least valuation to remove the factors that describe curtains. Thus Lazard's method does not fail even if the polynomial constraints contain curtains. The following defines Lazard's projection operator.

Definition 40. *[MPP19, definition 2.1] Let A be a finite set of irreducible polynomials in $\mathbb{R}[x_1, \ldots, x_n]$ with $n \geq 2$. The* Lazard projection operator PL(A) *is the subset of $\mathbb{R}[x_1, \ldots, x_{n-1}]$ composed of the following polynomials.*

- *All leading coefficients of the elements of A.*
- *All trailing coefficients of the elements of A.*
- *All discriminants of the elements of A.*
- *All resultants of pairs of distinct elements of A.*

The use of this projection operator is shown in Algorithm 4. Steps 8 – 12 describe Lazard's approach in using the lex-least valuation. Lazard calculates the Lazard residue of the polynomials (step 9) over the sample points in contrast to McCallum's approach of substituting the sample points before calculating the roots of the polynomials. In doing so, his method removes the factors of the polynomials that potentially could nullify them. Hence we are able to compute roots on curtains of the polynomial constraints.

Theorem 16. *[MPP19, Theorem 5.1] Let* $f(x_1, \dots, x_n) \in \mathbb{R}[x_1, \dots, x_n]$ *have positive degree* d *in* x_n *and suppose* $\mathrm{disc}_{x_n}(f)$*,* $\mathrm{ldcf}_{x_n}(f)$ *and* $\mathrm{trcf}_{x_n}(f)$ *are non-zero (as elements of* $\mathbb{R}[x_1, \dots, x_{n-1}]$*). Let* S *be a connected subset of* $\mathbb{R}^{n-1}$ *in which* $\mathrm{disc}_{x_n}(f)$*,* $\mathrm{ldcf}_{x_n}(f)$ *and* $\mathrm{trcf}_{x_n}(f)$ *are all lex-least invariant. Then* f *is Lazard delineable on* S *and hence* f *is lex-least invariant in every Lazard section and sector over* S*. Moreover, the same conclusion holds for the polynomial* $f^*(x_1, \dots, x_n) = x_n f(x_1, \dots, x_n)$*.*

Algorithm 4 Lazard's algorithm for lex-least invariant CAD

$(I, S) \leftarrow \mathrm{LCAD}(A)$

Input = Set of polynomials A in n variables.

Output = I and S are lists of indices and sample points, respectively, of the cells of a lex-least invariant CAD of A.

```
1: If $n \geq 2$ then go to step 3.
2: Isolate the real roots of the irreducible factors of the non-zero elements of $A$.
   Construct cell indices $I$ and sample points $S$ from the real roots. Exit.
3: $B \leftarrow$ the square free basis of the primitive parts of all elements of $A$.
4: $P \leftarrow \mathrm{cont}(A) \cup \mathrm{PL}(B)$.
5: $(I', S') \leftarrow \mathrm{LCAD}(P)$.
6: $(I, S) \leftarrow$ (empty list, empty list).
7: for each $\alpha \in S'$ do
8:     Let $i$ be the index of the cell containing $\alpha$.
9:     $f^* \leftarrow \{f_\alpha \mid f \in B\}$.
10:    Isolate the real roots of all the polynomials in $f^*$.
11:    Construct cell indices and sample points for Lazard sections and sectors of
       elements of $B$ from $i$, $\alpha$ and the real roots of $f^*$.
12:    Add the sample points to $I$ and $S$.
13: end for
14: return $(I, S)$
```

Corollary 2. *Let* $f(x, x_n) \in \mathbb{R}[x, x_n]$ *satisfying the assumptions of Theorem 16. Let* S *be a connected subset of* $\mathbb{R}^{n-1}$ *in which* $\mathrm{disc}_{x_n}(f)$*,* $\mathrm{ldcf}_{x_n}(f)$ *and* $\mathrm{trcf}_{x_n}(f)$ *are all lex-least invariant. Then* f *is Lazard delineable on* S *and is lex-least invariant in every section and sector of* f *over* S*.*

Theorem 17. *Let* $A = \{f_1, \ldots, f_m\}$ *be a set of pairwise relatively prime irreducible polynomials in* n *variables* $x_1, ..., x_n$ *of positive degrees in* x_n*, where* $n \geq 2$*. Let* S *be a subset of* $\mathbb{R}^{n-1}$ *obtained via Algorithm 4 such that each element of* $\mathrm{PL}(A)$ *is lex-least invariant in* S*. Then every element of* A *is lex-least invariant in the Lazard sections and sectors of every other element.*

Proof. If either $x_n \in A$ or $-x_n \in A$ then set $f = f_1 \ldots f_m / x_n$; otherwise set $f = f_1 \cdots f_m$.

Let us first assume $x_n \notin A$. Hence f is the product of all elements of A whose trailing coefficients are non-zero. Then the trailing and leading coefficient of f are non-zero and hence by Proposition 4 lex-least invariant in S. We know $\mathrm{disc}_{x_n}(f)$ can be expanded as follows:

$$\mathrm{disc}_{x_n}(f_1 \ldots f_m) = (\prod_{i=1}^{m} \mathrm{disc}_{x_n}(f_i))(\prod_{1 \leq i < j \leq m} \mathrm{res}_{x_n}(f_i, f_j)). \tag{3.1}$$

If everything in the RHS of 3.1 is lex-least invariant in S, then $\mathrm{disc}_{x_n}(f)$ is lex-least invariant by Proposition 4. Hence by the first part of Theorem 16, f is Lazard delineable over S. Since $\mathrm{res}_{x_n}(f_i, f_j)$ for all $1 \leq i < j \leq m$ are lex-least invariant in S, the Lazard sections of any two f_i and f_j $(i \neq j)$ are either disjoint or the same. Hence every element of A is lex-least invariant in the Lazard sections of every other element.

If $x_n \in A$, then we use the second conclusion of Theorem 16 on $f^* = x_n f$ and the result follows. □

Chapter 4

Lex-Least Invariance on a Single Hypersurface

In this chapter, we look at exploiting a hypersurface described by a single equational constraint; that is we decompose the hypersurface rather than the whole space of $\mathbb{R}^n$. When considering the projection set, all our information about the non-equational constraints comes from the resultants. However, this does not give us any information on the lex-least valuation of the non-equational constraints.

This chapter closely follows McCallum's work from [McC99]. The main idea stems from obtaining a solution space for a QFF of the shape

$$(f = 0) \wedge (\ldots) \tag{4.1}$$

using CAD algorithms. Since the QFF contains an equational constraint $f = 0$, any cell (in any CAD of $\mathbb{R}^n$) that is a solution to the QFF would also be contained in the hypersurface V_f described by $f = 0$. Hence it is sufficient to decompose the hypersurface V_f rather than the whole of $\mathbb{R}^n$.

McCallum's work on equational constraints focused on lifting order invariance to sign invariance [McC99]. In doing so, he assumed that there were no curtains present in the equational constraint. However, the only reason for making this assumption was that he used his work [McC84], which could not handle polynomials with curtains. In this chapter we use Lazard's projection operator to exploit equational constraints. Unfortunately, this approach still fails when it encounters curtains in the equational constraint, but by building on Lazard's work, it can handle curtains present in other constraints of the QFF. We demonstrate this through an example at the end of the chapter. The problem caused by curtains in our case is not the same as McCallum's, which is discussed further in Chapter 5 and 6.

4.1 Main Theorem

This section contains three significant modifications of Lazard's method and its verification by McCallum et al. in [MPP19]. The three developments are as follows:

- Modified projection operator: Like McCallum in [McC99] we modify the existing Lazard projection operator to exploit the single hypersurface described by an equational constraint.
- Lifting theorem: This theorem validates the use of the modified projection operator to lift a lex-least decomposition to a sign invariant decomposition.
- Modified lifting algorithm: This algorithm sets out the procedural steps required to obtain a sign invariant CAD of the hypersurface described by an equational constraint.

4.1.1 Modified Projection Operator

We start by defining the modified projection operator. As in [McC99], we omit the cross resultants, discriminants and the coefficients of the non-equational constraints.

Definition 41. *Let A be a finite set of irreducible polynomials in $\mathbb{R}[x_1, \ldots, x_n]$ with $n \geq 2$ and let E be a subset of A. The* modified Lazard projection operator $\mathrm{PL}_E(A)$ *is the subset of $\mathbb{R}[x_1, \ldots, x_{n-1}]$ consisting of the following polynomials:*

- *All leading coefficients of the elements of E.*
- *All trailing coefficients of the elements of E.*
- *All discriminants of the elements of E.*
- *All resultants of pairs of distinct elements of E.*
- *All resultants $\mathrm{res}_{x_n}(f, g)$ where $f \in E$ and $g \in A \setminus E$.*

We can also define it as follows:

$$\mathrm{PL}_E(A) = \mathrm{PL}(E) \cup \{\mathrm{res}_{x_n}(f, g) \mid f \in E,\, g \in A \setminus E\}.$$

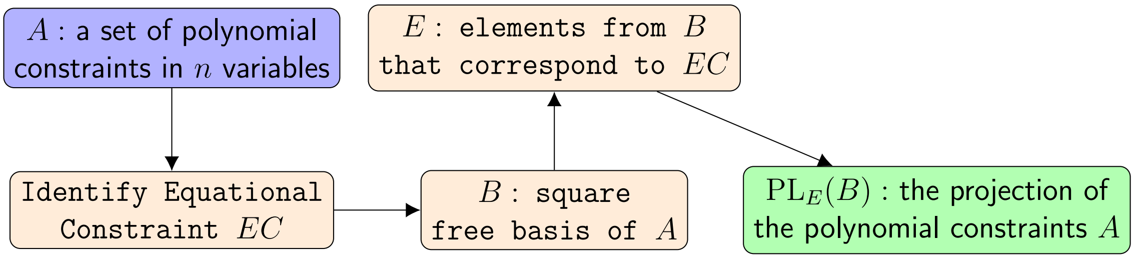

A : a set of polynomial constraints in n variables
E : elements from B that correspond to EC
Identify Equational Constraint EC
B : square free basis of A
$\mathrm{PL}_E(B)$: the projection of the polynomial constraints A

Figure 4-1: Equational constraint implementation

In practice we choose E to consist of the irreducible factors of the chosen equational constraint i.e. the equational constraint is defined by $\prod_{f\in E} f = 0$. Figure 4-1 describes how we project a set of polynomial constraints.

Note that, unlike $\mathrm{P}_E(A)$, the modified Lazard projection operator $\mathrm{PL}_E(A)$ does not contain the middle coefficients of the elements of E. This is because $\mathrm{PL}_E(A)$ is based on Lazard's work, which omits the middle coefficients of the polynomials constraints.

4.1.2 Lex-least to Sign Lifting Theorem

In this section we state and prove the lifting theorem that validates the use of the modified projection operator $\mathrm{PL}_E(A)$. In summary, this lifting theorem states that a CAD of $S \subset \mathbb{R}^{n-1}$ lifts to a sign invariant CAD of A on the hypersurface described by the equational constraint.

The following theorem describes the relationship between the resultant of two polynomials and their sign. This theorem is the first step of two to show that $\mathrm{PL}_E(A)$ can lift lex-least invariance to sign invariance.

Theorem 18. *Let $n \geq 2$ and let $f, g \in \mathbb{R}[x_1, \ldots, x_n]$ be of positive degrees in the main variable x_n, that are relatively prime (with respect to variable x_n). Let S be a connected subset of $\mathbb{R}^{n-1}$. Suppose that f is Lazard delineable on S and that $\mathrm{res}_{x_n}(f, g)$ is lex-least invariant on S, and that V_f does not have a curtain on S. Then g is sign invariant in each Lazard section of f over S.*

Proof. Since g is a continuous function, it is sufficient to show that g is sign invariant in an arbitrary neighbourhood of a point on some Lazard section of f over S. Define σ to be a Lazard section of f over S, given by the map $\theta : S \to \mathbb{R}$. Let α be an arbitrary point on σ. Without loss of generality, we can assume α to be the origin.

We can also assume $g(\alpha) = 0$ since otherwise g is sign invariant in a neighbourhood of α in $\mathbb{R}^n$, in particular in a neighbourhood in σ.

By the definition of θ, we have $\theta(0^{n-1}) = 0$ and also $f(0^n) = 0$. Suppose that $x_n = 0$ has multiplicity $m \neq 0$ as a root of $f(0^{n-1}, x_n) = 0$. This implies that $f(0^{n-1}, x_n) = \bar{q}(x_n) \cdot x_n^m$ where $\bar{q}(0) \neq 0$. Therefore by Theorem 2 (page 16) there exists a Euclidean neighbourhood N_1 of the origin in $\mathbb{R}^n$ and formal power series $q(x_1, \ldots, x_n)$ and $h(x_1, \ldots, x_n)$ such that $f(x_1, \ldots, x_n) = q(x_1, \ldots, x_n) \cdot h(x_1, \ldots, x_n)$ for $(x_1, \ldots, x_n) \in N_1$, where $q(0^{n-1}, x_n) = \bar{q}(x_n)$ and $h(0^{n-1}, x_n) = x_n^m$.

Since $q(0^n) \neq 0$, there exist $\epsilon > 0$ and an open neighbourhood N_2 of the origin in $\mathbb{R}^{n-1}$ such that $q(\gamma_1, \ldots, \gamma_n) \neq 0$ for all $(\gamma_1, \ldots, \gamma_n) \in N_2 \times (-\epsilon, \epsilon)$. Note, N_1 is a neighbourhood where q and h converge, and $N_2 \times (-\epsilon, \epsilon)$ is a subset of N_1. Since θ is continuous we can shrink N_2 to get $\theta(\gamma_1, \ldots, \gamma_{n-1}) \in (-\epsilon, \epsilon)$ for all $(\gamma_1, \ldots, \gamma_{n-1}) \in N_2$. In essence the subsection of σ (of f over N_2) lies in $N_2 \times (-\epsilon, \epsilon)$. If $\alpha' \in S \cap N_2$ and f is Lazard delineable then $\nu_{(\alpha', \theta(\alpha'))}(f) = \nu_\alpha(f)$.

On the other hand, $\nu_\alpha(h) = (0^{n-1}, m)$. Since $f = q \cdot h$ and f is Lazard delineable in $S \cap N_2$, this implies that $\nu_{(\alpha', \theta(\alpha'))}(h) = (0^{n-1}, m)$.

Let P be the resultant of h and g with respect to x_n. Clearly we have $\deg(f) \geq m$. If $\deg(f) > m$, we can take Q to be the resultant of q and g. In the case $\deg(f) = m$ take $Q = a_0(x_1, \ldots, x_m)^{\deg(g)}$.

We also have $P = 0$ at α, and h and g both vanish at α, so $\nu_\alpha(P)$ is non-zero. By assumption $\mathrm{res}_{x_n}(f, g)$ is lex-least invariant in the region N_2, so P is lex-least invariant in N_2 (by Proposition 4).

Since $h = 0$ in $\sigma \cap (N_2 \times (-\epsilon, \epsilon))$ and P is lex-least invariant in N_2, we see that $g = 0$ in $\sigma \cap (N_2 \times (-\epsilon, \epsilon))$ which is a Euclidean neighbourhood of α in σ. Thus g is sign invariant in σ, which implies that it is sign invariant in S. ☐

Theorem 18 establishes that $\mathrm{PL}_E(A)$ can lift from Lazard invariance to sign invariance. The natural question to ask next is whther it is possible to lift to Lazard invariance for small values of n. The following example demonstrates that it is not possible to do so.

Example 8. *Let $f = z - x$ be the equational constraint and let $g = z^2 - y^2 - x^2$. Let $R = \mathrm{res}_z(f, g) = -y^2$ and set S to be the x-axis. f is clearly lex-least delineable over S and R is lex-least invariant in S. But $\nu_{(0,0,0)}(g) = (0, 0, 2)$ and $\nu_{a,0,a}(g) = (0, 0, 1)$. This shows that g is not lex-least invariant in the section of f over S.*

Theorem 19. *Let A be a set of pairwise relatively prime irreducible polynomials in n variables $x_1, ..., x_n$ of positive degrees in x_n, where $n \geq 2$. Let E be a subset of A. Let S be a connected subset of $\mathbb{R}^{n-1}$. Suppose that each element of $\mathrm{PL}_E(A)$ is lex-least invariant in S. Then each element of E is Lazard delineable on S and exactly one of the following must be true.*

1. *The hypersurface defined by the product of the elements of E has a curtain over S.*
2. *The Lazard sections over S of the elements of E are pairwise disjoint. Each element of E is lex-least invariant in every such Lazard section. Each element of $A \setminus E$ is sign invariant in every such Lazard section.*

Proof. Because $\mathrm{PL}(E) \subseteq \mathrm{PL}_E(A)$, by Theorem 16 (page 43) either each element of E is Lazard delineable on S or V_E has a curtain on S and the Lazard sections of the elements of E that do not contain curtains are pairwise disjoint. Now let $f \in E$ (such that f does not have a curtain over S), let σ be a section of f over S and let $g \in A$. If $g \in E$, then by Theorem 16, g is lex-least invariant in every section and sector of f. If $g \notin E$ then, $R = \mathrm{res}(f, g) \in \mathrm{PL}_E(A)$ and by the assumption R is lex-least invariant in S. Then by Theorem 18, g is sign invariant in σ. $\square$

The reason for sign invariance and not lex-least invariance is that, we only get information of non-equational constraints is through the resultants. We later see in Chapter 6 that in order to get lex-least invariance we would need the coefficients and discriminants of the non-equational constraints.

Remark 3. *Theorems 18 and 19 first appeared in [NDS19], where they are expressed in terms of nullification (i.e. algebraically) rather than in terms of curtains (i.e. geometrically) as the curtain point of view had not yet been developed.*

This modification provides two significant benefits.

- Since the modified projection operator is based on $\mathrm{PL}(A)$, encountering curtains on levels $n-1$ to 1 will not cause the algorithm to error out.
- CAD algorithms are recursive, so using the modified projection operator $\mathrm{PL}_E(A)$ just once, for the projection from $\mathbb{R}^n$ to $\mathbb{R}^{n-1}$ yields an improvement in the overall complexity. We expand more on this in Chapter 8.

4.1.3 Modified Lifting Algorithm

Traditional CAD algorithms (that do not exploit equational constraints) use a single projection operator recursively. Our method uses the modified projection operator, $\mathrm{PL}_E(A)$ for the n-level polynomials, and then reverts to Lazard's projection operator $\mathrm{PL}(A)$ for the remaining recursions. This is described in steps 6 and 7 of Algorithm 5. Steps 10 to 15 lift a CAD of $\mathbb{R}^{n-1}$ to a CAD of the hypersurface described by the equational constraint, which is validated by Theorem 19.

Algorithm 5 Modified Lazard algorithm for sign invariant CAD

$(I, S) \leftarrow \mathrm{MLCAD}(A, f, n)$

Input = Set of polynomials A in n variables, f is the polynomial describing the equational constraint $f = 0$.

Output = I and S are lists of indices and sample points, respectively, of the cells of a sign invariant CAD of the hypersurface V_f.

```
1: If $n \geq 2$, then go to step 3:
2: Isolate the real roots of the irreducible factors of the non-zero elements of $A$.
   Construct cell indices $I$ and sample points $S$ from the real roots. Exit.
3: $B \leftarrow$ the squarefree basis of the primitive parts of all elements of $A$.
4: $E \leftarrow$ all irreducible factors of $f$.
5: $P \leftarrow \mathrm{cont}(A) \cup \mathrm{PL}_E(B)$.
6: $(I', S') \leftarrow \mathrm{LCAD}(P, n-1)$.
7: $(I, S) \leftarrow$ (empty list,empty list).
8: for each $\alpha \in S'$ do
9:    If $f$ has a non-point curtain over $\alpha$, return Fail
10:   Let $i$ be the index of the cell containing $\alpha$.
11:   $f^* \leftarrow \{f_\alpha \mid f \in B\}$.
12:   Isolate the real roots of all the polynomials in $f^*$.
13:   Construct cell indices and sample points for Lazard sections and sectors of
      elements of $B$ from $i$, $\alpha$ and the real roots of $f^*$.
14:   Add the sample points to $I$ and $S$.
15: end for
16: return $(I, S)$
```

Remark 4. *In Step 6, the algorithm* LCAD *is called and not* MLCAD*, as the projection operator* PL_E *cannot be used recursively, since it lifts Lazard invariance to sign invariance.*

This improvement does have a limitation of its own. It fails if the equational constraint has a curtain on S and S is not a singleton. This is further discussed in Chapter 6.

4.2 Working Example

Although we made significant improvements with this approach to CAD algorithms to improve the complexity (described in Chapter 8), we encountered problems with curtains. However, these problems are not the same as McCallum's problems stated at the beginning of this chapter. In this section, we address the issues caused by curtains in detail through a working example.

Let us consider a QFF having the following polynomial constraints

$$\begin{aligned} f_1 &:= (x-1)^2 + (y-1)^2 - 1 \\ f_2 &:= x^2 + y^2 + z^2 - 1 \\ f_3 &:= z - x \\ f_4 &:= z - y \end{aligned} \tag{4.2}$$

and let f_1 and f_2 be equational constraints.

The hypersurface described by $f_1 = 0$ is the cylinder shown in Figure 4-2. Our algorithm decomposes a single equational constraint, so let us first look at the case where we decompose $f_1 = 0$. In this case, we will calculate the following resultants:

$$\begin{aligned} R_1 &:= \mathrm{res}_z(f_1, f_2) = (x^2 + y^2 - 2x - 2y + 1)^2 \\ R_2 &:= \mathrm{res}_z(f_1, f_3) = x^2 + y^2 - 2x - 2y + 1 \\ R_3 &:= \mathrm{res}_z(f_1, f_4) = x^2 + y^2 - 2x - 2y + 1. \end{aligned} \tag{4.3}$$

Because $R_1 = R_2^2 = R_3^2$, the resultants do not give us any information on how f_2, f_3 and f_4 interact on the hypersurface V_{f_1}. This is the sole reason why curtains cause a problem in our method. Curtains occurring at any level from $n-1$ to 1 pose no problem since our work builds on Lazard's method which which works on curtains. This is discussed later (in Chapter 5).

Let us now look at the case where the chosen equational constraint is $f_2 = 0$. In this case, the resultants computed are

$$\begin{aligned}R'_1 &:= \mathrm{res}_z(f_2, f_1) = (x^2 + y^2 - 2x - 2y + 1)^2\\ R'_2 &:= \mathrm{res}_z(f_2, f_3) = 2x^2 + y^2 - 1\\ R'_3 &:= \mathrm{res}_z(f_3, f_4) = x^2 + 2y^2 - 1.\end{aligned} \tag{4.4}$$

If we proceed to calculate the resultants of these resultants, we obtain the following:

$$\begin{aligned}R'_{1,2} &:= \mathrm{res}_y(R_1, R_2) = (x^4 + 4x^3 + 8x^2 - 8x)^2\\ R'_{2,3} &:= \mathrm{res}_y(R_2, R_3) = (3x^2 - 1)^2\\ R'_{3,1} &:= \mathrm{res}_y(R_3, R_1) = (x^4 - 8x^3 + 30x^2 - 24x + 1)^2.\end{aligned} \tag{4.5}$$

This gives us the required information on how f_1, f_3 and f_4 interact on V_{f_2}. This provides further evidence that the curtain/nullification problem is a problem linked to CAD computation rather than the valuation property used.

To fully understand the problems caused by curtains, we choose to define them formally and reformulate the nullification problem in terms of curtains. In the next chapter, we examine the various types of curtains, their properties and their effects on the computation of CADs.

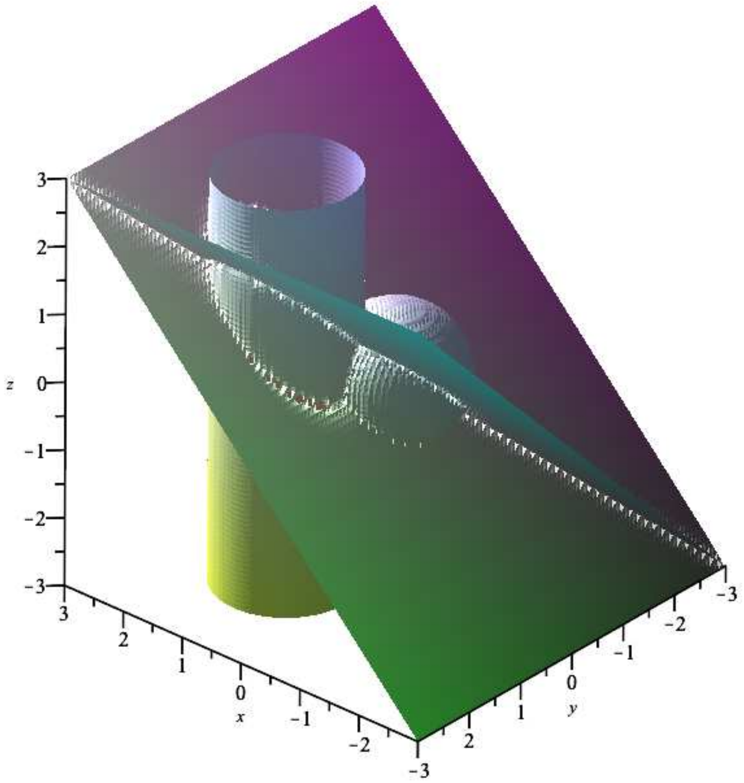

z
x
y
3
2
1
0
−1
−2
−3

Figure 4-2: Curtain component

Chapter 5

Curtains

This chapter looks into details about curtains, which we mentioned in Chapter 1 and 4. Throughout the developments made in the theory and algorithms of CAD, there have always been difficulties when input polynomial constraints nullify.

Definition 42. *Let f be a polynomial in $\mathbb{R}[x_1, \ldots, x_n]$ and S be a subset of $\mathbb{R}^m$ with $m < n$. We say that f is nullified over S, or vanishes over S, if $f(\alpha, \beta) = 0$ for all $\alpha \in S$ and for all $\beta \in \mathbb{R}^{n-m}$.*

Because $\mathrm{ord}_{(\alpha,\beta)}(f)$ is independent of β if f vanishes over α. In principle it is possible to use the information coming from order alone to construct a cylindrical algebraic decomposition, but in practice it is not. Order invariance sits at the core of McCallum's work in [McC84], [McC99] and [McC01]; hence his approach has an inherent problem with nullification.

To fully understand the problems associated with curtains, this chapter looks into the classification of curtains based on different properties, how to detect them and their relation to input polynomial constraints for CAD problems.

5.1 Definitions and Examples

Definition 43. *A variety $C \subseteq \mathbb{R}^n$ is called a* curtain *if, whenever $(x, x_n) \in C$, then $(x, y) \in C$ for all $y \in \mathbb{R}$.*

In other words, C is a curtain if it is a union of fibres of $\mathbb{R}^n \to \mathbb{R}^{n-1}$.

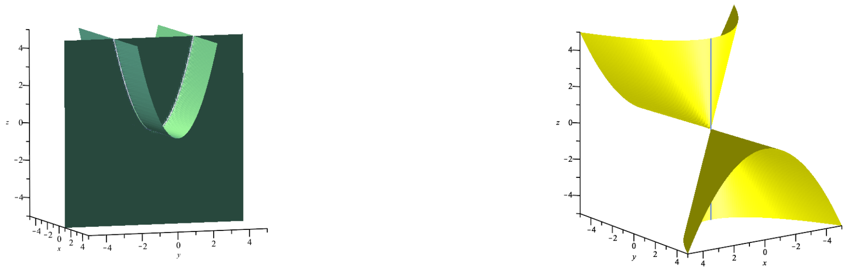
Definition 44. *Suppose* $f \in \mathbb{R}[x_1, \ldots, x_n]$ *and* $S \subseteq \mathbb{R}^{n-1}$. *We say that* V_f *(or* f*) has a curtain at* S *if for all* $(\alpha_1, \ldots, \alpha_{n-1}) \in S$ *and for all* $y \in \mathbb{R}$ *we have* $f(\alpha_1, \ldots, \alpha_{n-1}, y) = 0$. *We call* S *the* base *of the curtain.*

Definition 45. *Let* $f \in \mathbb{R}[x_1, \ldots, x_n]$. *Suppose that* f *factorises as* $f = gh$, *where* $g \in \mathbb{R}[x_1, \ldots, x_{n-1}]$ *and* $g(\alpha_1, \ldots, \alpha_{n-1}) = 0$, *then* f *is said to contain an* explicit curtain *whose base is the zero set of* g.

Example 9. *For a hypersurface there are two types of curtains.*

- *Explicit Curtain:* $f(x, y, z) = xy^2 - y^2 - xz + z = (x-1)(y^2 - z)$, *curtain at* $(1, 0)$. *The curtain can be seen in Figure 5-1a as the sheet given by* $x = 1$.

- *Implicit Curtain:* $f(x, y, z, k) = x^2 + yz$, *curtain at* $(0, y, 0)$. *This can be seen in Figure 5-1b, where the blue line represents the curtain.*



(a) Surface with an Explicit Curtain (b) Surface with an Implicit Curtain

Figure 5-1: Different types of curtains

The terminology of explicit curtains becomes slightly complicated when we look at higher codimensions. The notion of explicit curtain could be extended to higher codimension but there is more than one way to do that and at present it is not clear which of them, if any, may prove useful. For example one could distinguish irreducible components that are curtains, but they may not be explicitly specified by the original equations.

5.2 CAD and Curtains

In this section, we discuss the interactions between curtains and the work done by McCallum and Collins. Most of McCallum's work on CAD is linked to order invariance, so let us recall the definitions related to order of polynomials.

The problem with curtains occurred in the lifting algorithm, as it would try to compute the roots of nullified polynomials, resulting in no roots being found. This causes McCallum's algorithm to produce an error whenever it encounters a non-point curtain in the projection, so that the curtains of the polynomial constraints are not decomposed. There is no way of classifying curtains into delineable sections and sectors on the basis of the order of polynomials.

Lazard deals with these problems by using the lex-least valuation and Lazard delineability. Before we look at Lazard delineability, let us look at the close relation between lex-least valuations and curtains.

Lemma 2. *Let $f \in \mathbb{R}[x_1, \ldots, x_n]$ and $\alpha \in \mathbb{R}^{n-1}$. Then $\nu'_\alpha(f) \neq 0$ if and only if there is a curtain at α.*

Proof. Let $\alpha = (\alpha_1, \ldots, \alpha_{n-1})$ and $\nu'_\alpha(f) = (\nu_1, \ldots, \nu_{n-1})$. Let ν_i be the first non-zero coordinate in the valuation. Then $f(\alpha_1, \ldots, \alpha_{i-1}, x_i, \ldots, x_n)$ has a factor of $(x_i - \alpha_i)$, hence $f(\alpha_1, \ldots, \alpha_i, x_{i+1}, \ldots, x_n) = 0$ independent of x_n. This implies there is a curtain at α.

Let us now assume that f has a curtain at α. Run Algorithm 2. Assume all ν_i are zero in the output. This implies that f_α has to be trivially zero (i.e. independent of x_n), but this is only possible if f is trivially zero by Definition 37. This is a contradiction to the fact that f has a curtain at α. Hence there exists an i with $\nu_i \neq 0$. □

The major difference between the definition of delineablity stated by McCallum and Lazard is that McCallum considers the polynomial f whereas Lazard considers the residue f_α. In doing so, Lazard, in a way, is removing the curtain and only looking at the roots of the residue. Off curtains, sections and sectors defined by both McCallum and Lazard are the same. Since Lazard defines delineability of the residue of the polynomial, this results in the formation of the sections and sectors of the curtains of the polynomial.

Unfortunately, the nice relation between lex-least valuation and curtains does not hold when exploiting equational constraints using Lazard's theory. This isn't a limitations of Lazard's theory, but rather a limitation of equational constraints.

5.3 Curtains and Single Equational Constraints

McCallum, in [McC99] and [McC01], lifts order invariance to sign invariance and order invariance respectively, but only if the input polynomials did not have any curtains.

Firstly, not all types of curtains cause a problem when taking advantage of equational constraints at a single level. Let us classify curtains further.

Definition 46. *Let $f \in \mathbb{R}[x_1, \ldots, x_n]$ and $\alpha \in \mathbb{R}^{n-1}$. We say that f has a* point curtain *at α*

- *if $f(\alpha, y) = 0$ for all $y \in \mathbb{R}$, and*
- *there exists a Euclidean open neighbourhood $U \subset \mathbb{R}$ of α such there exists no $\beta \in U \backslash \{\alpha\}$ such that $f(\beta, y) = 0$ for all $y \in \mathbb{R}$.*

Note that one can view this as a single fibre of a point on the hypersurface. It also implies the fibres of a small neighbourhood around the point in question are not part of the hypersurface. In Definition 46, we need to specify Euclidean neighbourhood and not Zariski neighbourhood. The following example illustrates this.

Example 10. *Let $f = y^2 - x^2(x-1)$ in $\mathbb{R}^3$. There is a point curtain at $(0,0)$. However, no open Zariski neighbourhood of $\{(0,0)\}$ satisfies Definition 46. In Figure 5-2, any polynomial that vanishes on the left hand component also vanishes on the point curtain.*

Example 11. *We illustrate this distinction as follows.*

- *Point Curtain: $f(x,y,z) = x^2 + zy^2 - z$ has point curtains at $(0,1)$ and $(0,-1)$. These are represented by the blue lines in Figure 5-3a.*
- *Non-Point Curtain: Consider $f(x,y,z) = -x^3y^3z - xy^4z + xy^3z^2 + x^4 + 2x^3z + x^2y - x^2z + 2xyz - 2xz^2$. It has a non-point curtain on $(0,y)$ for all $y \in \mathbb{R}$.*

The example for non-point curtains in Example 11 is of the form $x \cdot$(quintic) so it is an explicit curtain. In $\mathbb{R}^3$, all non-point curtains are explicit curtains.

Now let us explain how curtains cause a problem when we try to exploit the hypersurfaces defined by an equational constraint.

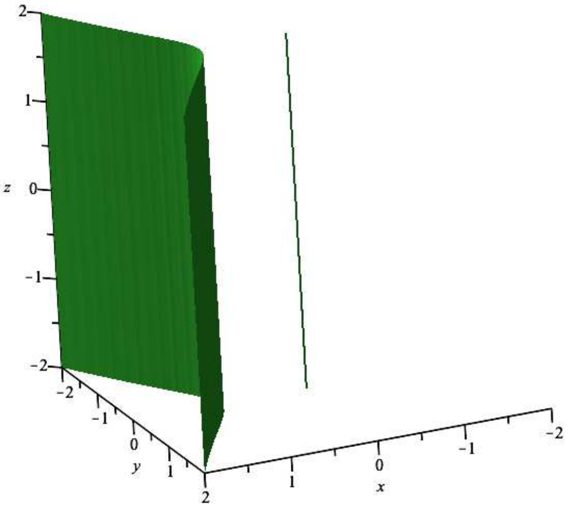
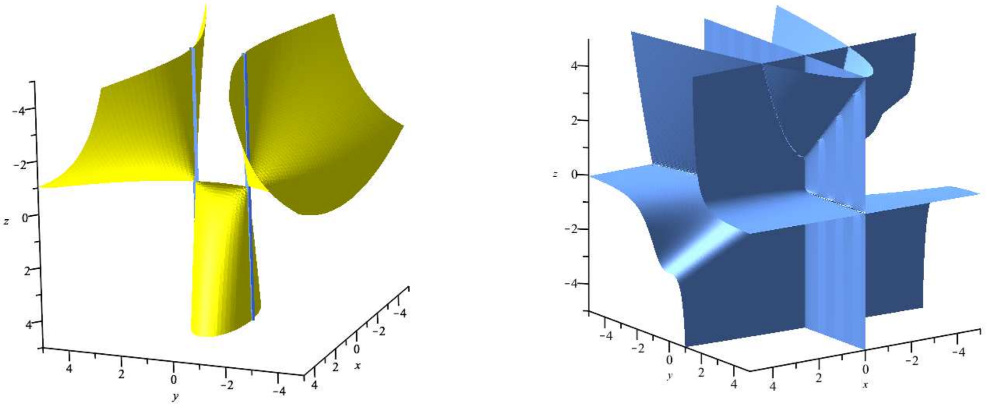


Figure 5-2: Euclidean neighbourhood

Example 12. *Let* $f = x^2+y^2-1$ *(which we assume to be an EC),* $g_1 = z-x-1$ *and* $g_2 = z-y-1$ *(which we assume are not ECs: See Figure 5-4). Then* $\mathrm{res}_z(f, g_1) = x^2+y^2-1 = \mathrm{res}_z(f, g_2)$*, and this gives us no information about* $\mathrm{res}_z(g_1, g_2)$*. In such cases, when the EC has a non-point curtain, it becomes impossible to use* PL_E *to detect the intersections of the other constraints on that curtain.*

Example 13. *Let* $f = x - yz$ *(which we assume to be an EC),* $g_1 = z - x$ *and* $g_2 = z-y$ *(which we assume are not ECs). Then* $\mathrm{res}_z(f, g_1) = yx-x$ *and* $\mathrm{res}_z(f, g_2) = y^2 - x$ *and* $\mathrm{res}_y(\mathrm{res}_z(f, g_1), \mathrm{res}_z(f, g_2)) = x^2(1-x)$*. This gives us information about the interaction of* g_1 *and* g_2*. In such cases, when the EC has a point curtain, the algorithm decomposes everything above correctly.*

In Chapter 6 we further examine this phenomenon and show how it can be used to circumvent the curtain problem for point curtains. In order to make use of this, we need to be able to distinguish point and non-point curtains as done in Algorithm 6.



(a) Point Curtain (b) Non-point Curtain

Figure 5-3: Different types of curtains

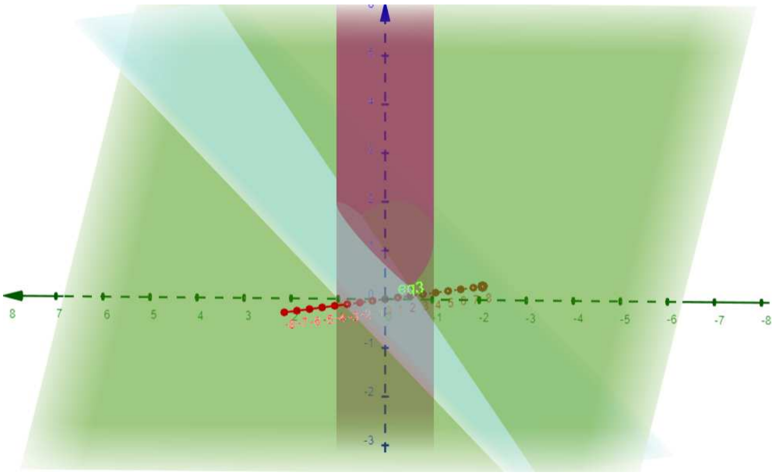


Figure 5-4: Failure at non-point curtains (Example 12)

5.4 Summary

Defining, classifying and understanding curtains has been one of our main focuses. Curtains have not been fully studied and we apply our understanding of curtains

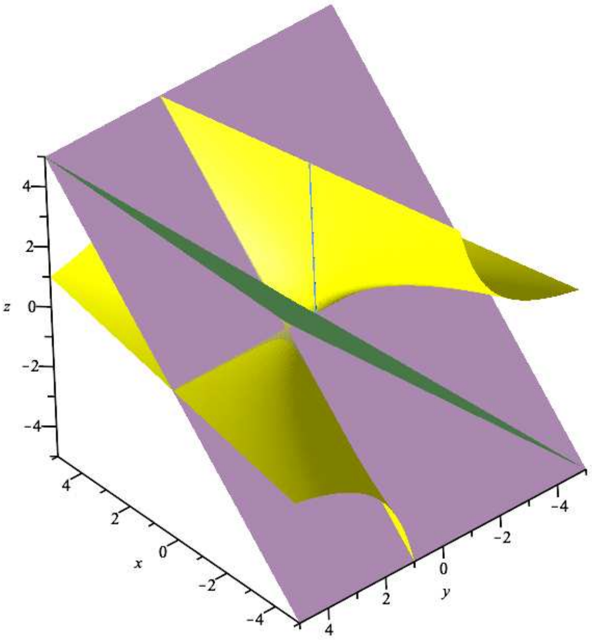


Figure 5-5: Success at point curtains (Example 12)

to equational constraints to reduce the complexity of CAD algorithms. The next chapter explores how one can use Algorithm 6 to circumvent the problem when a single equational constraint has curtains.

Algorithm 6 Detecting and Classifying Curtains

$(B, B') \leftarrow PC(f, I, S, n)$

Input = Set of indices I, set of sample points S with respect to the indices I which correspond to the CAD cells, equational constraint $f \in \mathbb{R}[x_1, \ldots, x_{n+1}]$.

Output = B, B', where B is the set of sample points that are point curtains and B' is the set of sample points that are curtains (but not point curtains).

```
B ← Empty list.
B′ ← Empty list.
for $\alpha \in S$ do
    if $\nu'_{\alpha}(f) \neq 0$ then
        Check if the nearest 1-cell neighbours have zero valuation.
        If all neighbours are zero valuation add $\alpha$ to $B$; otherwise add it to $B'$.
    end if
end for
return $(B, B')$.
```

Chapter 6

Further Developments of Equational Constraints and Curtains

This chapter addresses the problem of curtains encountered in Chapter 4. Our work on single equational constraints in Chapter 4 and [NDS19] reduces the Lazard projection by decomposing the hypersurface described by the equational constraint. In doing so, we only obtain information about the non-equational constraints from their resultants with the equational constraint. As mentioned in Chapter 4, these resultants give us no information on how the non-equational constraints interact with each other on curtains of the equational constraint. In this chapter, we discuss why point curtains do not cause Algorithm 5, which exploits a single hypersurface, to fail. We extend this by providing a subroutine that decomposes curtains of equational constraints.

The second half of this chapter looks at exploiting multiple hypersurfaces described by equational constraints. The main idea is based on obtaining a solution space for a QFF of the shape

$$(f_1 = 0) \wedge \ldots \wedge (f_k = 0) \wedge (\ldots) \tag{6.1}$$

using CAD algorithms. Since the QFF contains equational constraints $f_1, \ldots, f_k$, any cell (in any CAD of $\mathbb{R}^n$) that is a solution to the QFF would also be contained in the the intersections of of $V_{f_1}, \ldots, V_{f_k}$. Hence we restrict our algorithm to decomposition of the intersections of the equational constraints rather than the whole of $\mathbb{R}^n$. The key difference between this result and our result from Chapter 4 is that here we lift lex-least invariance to lex-least invariance rather than sign invariance. This enables us to use our projection operator recursively and takes advantage of multiple equational constraints. The specification of this is described in Section 6.2. Although this research mimics [McC01], it builds on [MPP19], and hence we do not encounter the same difficulties with curtains that arise in [McC01].

However, this algorithm, like the one from Chapter 4, fails when our equational constraints contain non-point curtains. To solve this problem, we could apply the method from Section 6.1, but this would significantly increase the computational complexity. This is further described in Chapter 9 along with the potential avenues to deal with this problem.

6.1 Curtains on Single Hypersurfaces

This section introduces a subroutine that is executed when an equational constraint contains a curtain. Our method is based on the fact that we can detect curtains during the lifting process of Algorithm 5 in Chapter 4. The subroutine is called if and only if there are curtains detected.

Theorem 19 lifts lex-least invariance to sign invariance when the equational constraint does not contain a curtain. So far little work has been done on how the structure of CAD algorithms affect the decomposition of curtains. In the following section we describe how our work from Chapter 4 does decompose curtains of a certain type.

6.1.1 Point Curtains

The following theorems describe how point curtains do not pose a problem to Algorithm 5, because of the way cells are lifted.

Proposition 6. *Let* $f, g \in \mathbb{R}[x_1, \ldots, x_n]$ *and* $\alpha \in \mathbb{R}^{n-1}$ *and suppose that the variety* V_f *defined by* f *contains a curtain at* α. *Then* V_g *contains a curtain at* α *or intersects* V_f *at finitely many points over* α.

This is the triviality that the intersection of V_f and V_g over α is either finite or infinite, rewritten in terms of curtains.

The modified projection operator $\mathrm{PL}_E(A)$ only projects information about the equational constraint and the resultants between the equational and non-equational constraints.

Theorem 20. *Let $f, g \in \mathbb{R}[x_1, \ldots, x_n]$ and suppose that $\alpha \in \mathbb{R}^{n-1}$ is a point cell obtained from Algorithm 5. Suppose that f is an equational constraint and that V_f has a point curtain at α. Then g is sign invariant in the sections and sectors of $f_\alpha g_\alpha$.*

Proof. By assumption V_f has a point curtain at α. This implies that there exists a neighbourhood B of α in $\mathbb{R}^{n-1}$ such that $\nu'_\beta(f) = 0^{n-1}$ for $\beta \in B \setminus \{\alpha\}$ and $\nu'_\alpha(f) \neq 0^{n-1}$. Because we are using the projection operator PL_E, where E is the set of irreducible factors of f, the polynomial f is lex-least invariant with respect to the CAD produced by the projection operator PL_E. Since the polynomial f is lex-least invariant with respect to the CAD and V_f has a curtain at α, the CAD consists of cells of the form $\alpha \times (a, b)$ or point cells $\alpha \times \{c\}$, where $a, b, c \in \mathbb{R}$.

We know that $\mathrm{res}(f, g)(\alpha) = 0$. Then by Proposition 6, V_g has a point curtain at α or intersects V_f at finitely many points. If V_g has a curtain at α then g is sign invariant in all cells over α. Suppose V_f only intersects V_g at finitely many points and let $\mathcal{A} = \{\beta_1, \ldots, \beta_k\}$ (arranged in increasing order) be the roots of $f_\alpha g_\alpha$. Then the cells above α will be of the form $\alpha \times (a, b)$ or point cells $\alpha \times \{c\}$, where $a, b, c \in \mathcal{A}$. Note that β_i has to be either a root of f_α or g_α for each i. Since $g = 0$ intersects $f = 0$ at only finitely many points and all roots of g_α are in $\mathcal{A}$, g is non-zero in $\alpha \times (\beta_i, \beta_{i+1})$. Otherwise g is zero on the point cells (α, β_i) where β_i is a root of g, and non-zero otherwise. □

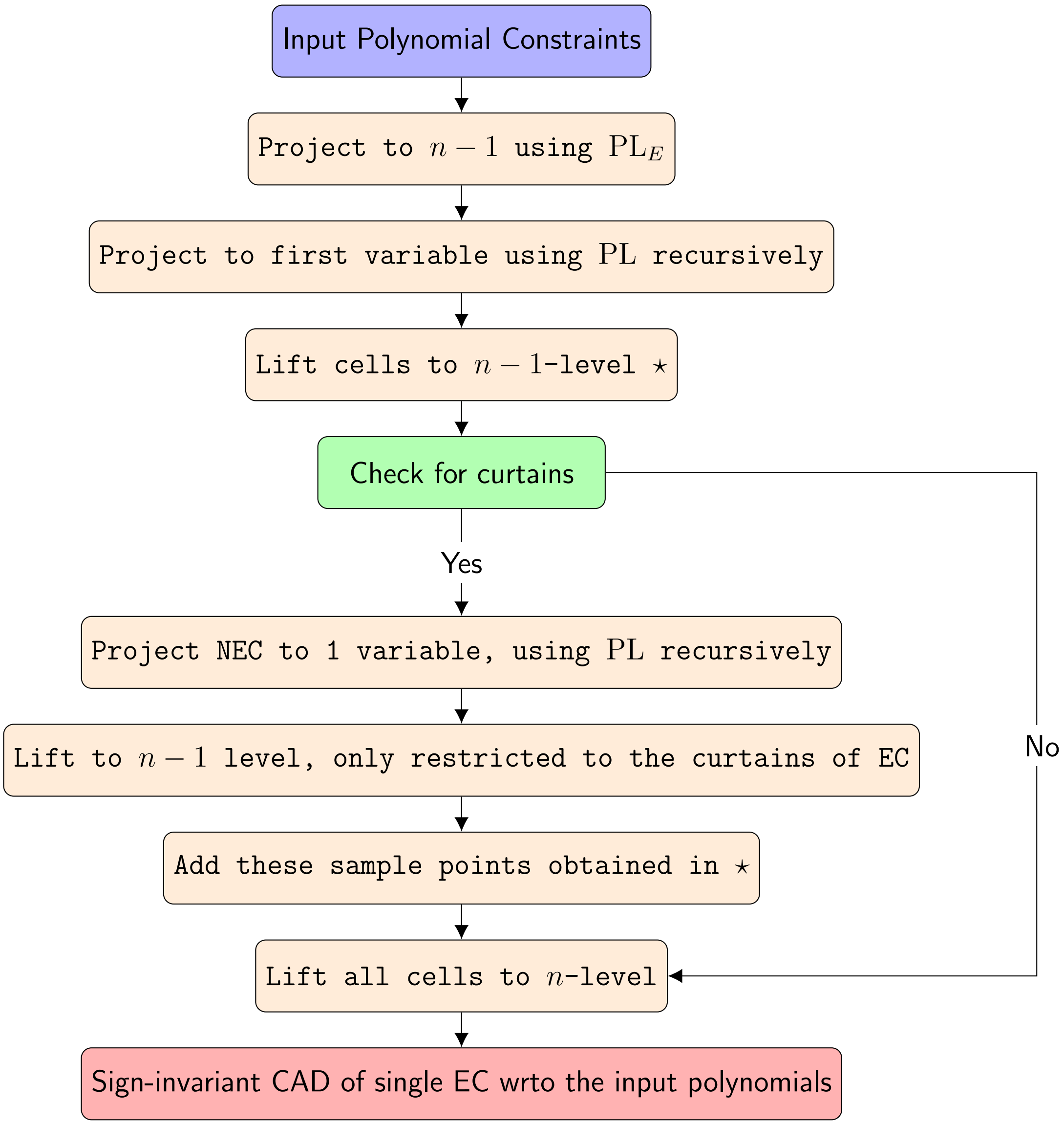
Curtains do not cause the algorithm in [Laz94] to halt. Hence in [Laz94] there was no need for a result analogous to Theorem 20. However, as described in Chapter 4 and Chapter 5, curtains pose a different problem when exploiting equational constraints, hence the need for Theorem 20.

6.1.2 Decomposing Curtain Base Set

Since we will be constantly referring to an arbitrary coordinate of a given point in the following sections, we define the following function.

Definition 47. *Let $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{R}^n$. We define $\mathrm{CPROJ}_k(\alpha)$ to be the k^{th} coordinate of α, which is α_k.*

To understand our approach, we will first demonstrate it through a rough diagram of the main algorithm and an example. We further discuss this subroutine by providing a theorem that validates its use, followed by a standard algorithm.



Input Polynomial Constraints
Project to $n-1$ using PL_E
Project to first variable using PL recursively
Lift cells to $n-1$-level $\star$
Check for curtains
Yes
No
Project NEC to 1 variable, using PL recursively
Lift to $n-1$ level, only restricted to the curtains of EC
Add these sample points obtained in $\star$
Lift all cells to n-level
Sign-invariant CAD of single EC wrto the input polynomials

Figure 6-1: Flow chart describing our approach to decompose the variety described by a single equational constraint

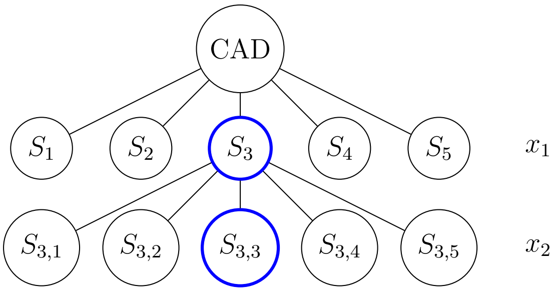


Figure 6-2: Set of sample points before Algorithm 7

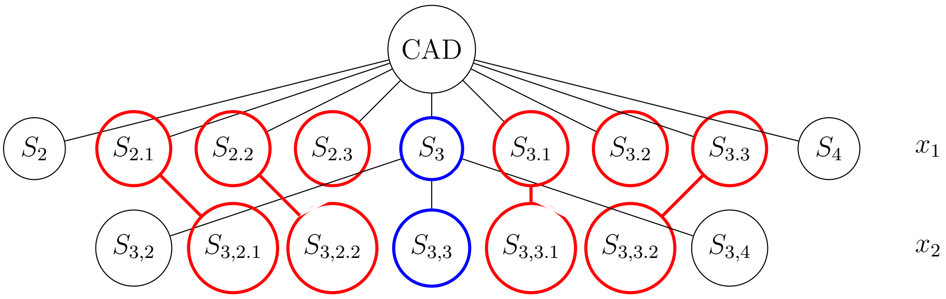


Figure 6-3: Set of sample points after Algorithm 7

Consider a QFF having a set of polynomial constraints A in $\mathbb{R}[x_1, x_2, x_3]$ with a single equational constraint f. We start by projecting to x_2 using the modified operator PL_E and then to x_1 using the standard Lazard operator PL. We then lift to cells in $\mathbb{R}^2$. Figure 6-2 describes the tree structure of the cells obtained.

We have omitted some of the branches from the diagram to make it clearer. In Figure 6-2 we have $S_1 < \ldots < S_5$ and $\mathrm{CPROJ}_2(S_{3,1}) < \ldots < \mathrm{CPROJ}_2(S_{3,5})$. Similarly we have assumed the same ordering for Figure 6-3.

We proceed by checking if the sample points at the x_2-level describe cells that are part of the base of a non-point curtain in the equational constraint. This is first done by computing the roots in the range of neighbours of the curtain, and then by computing the sample points in the range of neighbours of the curtain. In our case, we assume that there is a non-point curtain at sample point $S_{3,3}$. This sample is highlighted in blue. We now project all the non-equational constraints to x_1 using only the original Lazard operator PL. When lifting, we limit the sample points to the curtain cells, as seen in Figure 6-3.

The final step combines the sample points from Figure 6-2 and the new sample points showing in Figure 6-3 and lifts the cells to $\mathbb{R}^3$.

To decompose a curtain of an equational constraint is the same as decomposing a set $S \times \mathbb{R}$ where $S \subset \mathbb{R}^{n-1}$. When working in a curtain, the equational constraint does not give us any information on how the rest of the polynomial constraints interact. The following theorem validates our method of decomposing curtains separately.

Theorem 21. *Let* $A = \{f, g_1, \ldots g_m\}$ *be a set of polynomial constraints where* f *is the equational constraint. Let* D *be the CAD obtained using Algorithm 8. Then the constraints* $g_1, \ldots, g_m$ *are sign invariant in the cells of* D*.*

Proof. Let S be a connected subset of $\mathbb{R}^{n-1}$ such that f is Lazard delineable over S and the set $\{\mathrm{res}(f, g_j) | 1 \leq j \leq m\}$ is lex-least invariant over S. If V_f does not contain a non-point curtain on S, then by Theorem 19 and Theorem 20 we are done. Let us assume the contrary. Since V_f contains a non-point curtain on S, Algorithm 7 splits up S into connected, non-intersecting subsets, where f is Lazard delineable and $\mathrm{PL}(g_1, \ldots, g_m)$ is lex-least invariant. Hence by Theorem 17, $g_1, \ldots, g_m$ are lex-least invariant in the cells of the curtains of V_f, hence also sign invariant. $\square$

In Algorithm 7, the main parts are steps 3–7 and 14–18. For each recursion we look at the adjacent neighbours of the sample point describing the curtain. We compute the sample points within the neighbours of the curtains. In summary, we are restricting the computation of roots by using the neighbours of sample points describing curtains as boundaries.

Algorithm 7 Decomposing Curtain Base Set

$(I_C, S_C) \leftarrow \mathrm{DCBS}(A, I, S, C, m)$

Input = A a set of polynomials in m variables. Set of sample points S and their respective indices I. The set of sample points describing the curtain base $C \subset S$.

Output = S_C is a set of sample points that decompose the base of curtains described by C and their respective indices I_C.

```
 1: If $m \geq 2$ then go to step 9.
 2: for each $c \in C$ do
 3:     Let $i = (i_1, \ldots, i_n) \in I$ be the index of $c$.
 4:     Set $(c_1, c_2) \leftarrow$ Sample of indices $(i_1 - 1, 1 \ldots, 1)$ and $(i_1 + 1, 1 \ldots, 1)$.
 5:     Isolate the real roots of all the irreducible factors of the elements of $A$ between
        the neighbours of $\mathrm{CPROJ}_1(c_1)$ and $\mathrm{CPROJ}_1(c_2)$.
 6:     Construct cell indices and sample points for Lazard sections and sectors of
        elements of $A$ from $i$, $\alpha$ and the isolated real roots obtained from the previous
        step.
 7:     Add the sample points to $I_C$ and $S_C$.
 8: end for  Exit.
 9: $B \leftarrow$ the square free basis of the primitive parts of all elements of $A$.
10: $P \leftarrow \mathrm{cont}(A) \cup \mathrm{PL}(B)$.
11: $(I'', S'') \leftarrow \mathrm{DCBS}(P, I, S, C, m - 1)$.
12: If $m = n$, set $(I_C, S_C) \leftarrow (I'', S'')$; return $(I_C, S_C)$.
13: $(I', S') \leftarrow$ (empty list,empty list).
14: for each $\alpha \in S''$ do
15:     $f^* \leftarrow \{f_\alpha \mid f \in B\}$.
16:     for each $c \in C$ do
17:         Let $i = (i_1, \ldots, i_n) \in I$ be the index of $c$.
18:         Set $(c_1, c_2) \leftarrow$ Sample points given by indices $(i_1, \ldots, i_m - 1, 1 \ldots, 1)$ and
            $(i_1, \ldots, i_m + 1, 1 \ldots, 1)$
19:         Isolate the real roots of all the polynomials in $f^*$ between the neighbours of
            $\mathrm{CPROJ}_m(c_1)$ and $\mathrm{CPROJ}_m(c_2)$.
20:         Construct cell indices and sample points for Lazard sections and sectors
            of elements of $B$ from $i$, $\alpha$ and the isolated real roots obtained from the
            previous step.
21:         Add the sample points to $I_C$ and $S_C$.
22:     end for
23: end for
24: return $(I_C, S_C)$
```

Algorithm 8 Modified Lazard CAD with Curtains

$(I, S) \leftarrow \mathrm{MLCADC}(A, f, n)$

Input = Set of polynomials A in n variables, f is the polynomial describing the equational constraint $f = 0$.

Output = I and S are lists of indices and sample points, respectively, of the cells of a sign invariant CAD of the hypersurface V_f.

```
1: If $n \geq 2$ then go to step 3.
2: Isolate the real roots of the irreducible factors of the elements of $A$. Construct cell indices $I$ and sample points $S$ from the real roots. Exit.
3: $B \leftarrow$ the square free basis of the primitive parts of all elements of $A$.
4: $E \leftarrow$ all irreducible factors of $f$
5: $P \leftarrow \mathrm{cont}(A) \cup \mathrm{PL}_E(B)$.
6: $(I', S') \leftarrow \mathrm{LCAD}(P, n-1)$.
7: $(C_P, C_{NP}) \leftarrow PC(f, I', S', n)$.
8: If $C_{NP}$ is empty go to step 11.
9: $(I'', S'') \leftarrow \mathrm{DCBS}(A \backslash f, I', S')$.
10: $(I', S') \leftarrow (\mathrm{CPROJ}_{n-1}(I'') \cup I', \mathrm{CPROJ}_{n-1}(S'') \cup S')$.
11: $(I, S) \leftarrow$ (empty list,empty list).
12: for each $\alpha \in S'$ do
13:     Let $i$ be the index of the cell containing $\alpha$.
14:     $f^* \leftarrow \{f_\alpha \mid f \in B\}$.
15:     Isolate the real roots of all the polynomials in $f^*$.
16:     Construct cell indices and sample points for Lazard sections and sectors of elements of $B$ from $i$, $\alpha$ and the real roots of $f^*$.
17:     Add the sample points to $I$ and $S$.
18: end for
19: return $(I, S)$.
```

Algorithm 8 describes the procedure for obtaining a sign invariant CAD of the hypersurface described by the equational constraint, even in the presence of curtains. The main difference between Algorithm 8 and Algorithm 5 is in steps 7–11. They describe the process of checking whether the calculated sample points in $\mathbb{R}^{n-1}$ describe part of the base of a non-point curtain.

One of the main questions in choosing a CAD algorithm that exploits equational constraints is the trade-off of time vs the quality of the decomposition. Lazard [Laz94] produces a lex-least invariant CAD of $\mathbb{R}^n$: by contrast, our result produces a sign invariant CAD of the hypersurface described by the equational constraint. If we only wanted a sign invariant decomposition of the solution space, our approach would be more efficient. On the other hand, if a full decomposition of $\mathbb{R}^n$ were needed, our method would fall short. The algorithm described in this section decomposes curtains of the equational constraint, hence improving the quality of the CAD as compared to the approach of Chapter 4. However, it is computationally more expensive. This comparison is further discussed in Chapter 8.

6.2 Multiple Hypersurfaces

The content of this section is analogous to what we did in Chapter 4. We modify Lazard's projection operator and method to take advantage of multiple equational constraints in the input QFF. Let us recall what we mean by multiple equational constraints in a QFF:

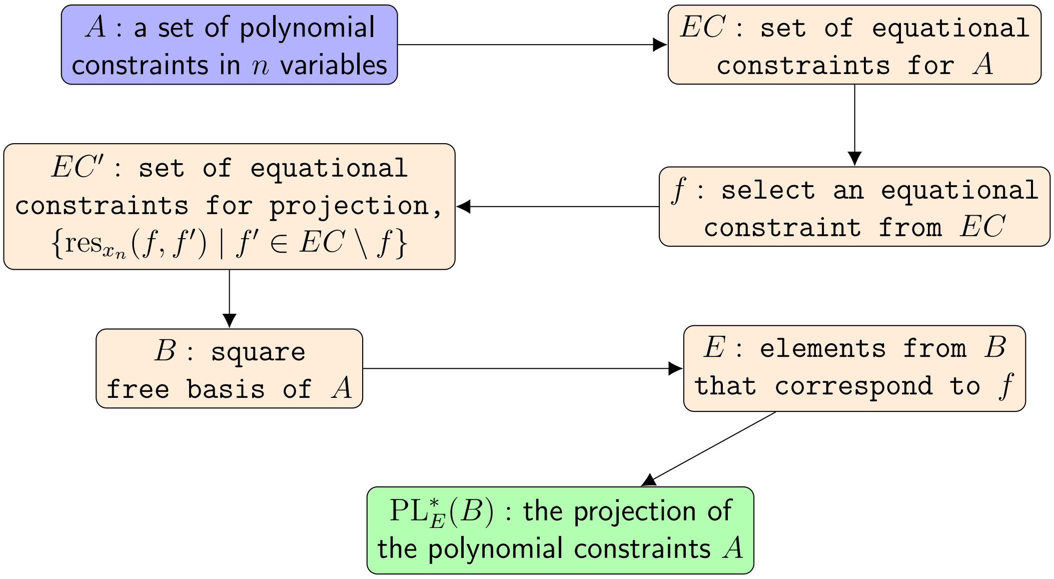
$$(f_1 = 0) \wedge \ldots \wedge (f_k = 0) \wedge (\ldots).$$

In this QFF, the equational constraints are given by the polynomials $f_1, \ldots, f_k$. Since a solution cell to the QFF must satisfy the equational constraints, we focus our attention on the intersection of equational constraints. First, let us define the projection operator.

Definition 48. *Let A be a finite set of irreducible polynomials in $\mathbb{R}[x_1, \ldots, x_n]$ with $n \geq 2$ and let E be a subset of A. The* modified Lazard projection operator for multiple equational constraints $\mathrm{PL}^*_E(A)$ *is the subset of $\mathbb{R}[x_1, \ldots, x_{n-1}]$ consisting of the following polynomials.*

- *All leading coefficients of the elements of A.*
- *All trailing coefficients of the elements of A.*
- *All discriminants of the elements of A.*
- *All resultants $\mathrm{res}_{x_n}(f, g)$, where $f \in E$ and $g \in A$ and $f \neq g$.*

The way we take advantage of multiple equational constraints is by using this operator recursively. The resultants between the chosen equational constraint and the


A : a set of polynomial constraints in n variables
EC : set of equational constraints for A
f : select an equational constraint from EC
EC' : set of equational constraints for projection, $\{\mathrm{res}_{x_n}(f, f') \mid f' \in EC \setminus f\}$
B : square free basis of A
E : elements from B that correspond to f
$\mathrm{PL}^*_E(B)$: the projection of the polynomial constraints A

remaining equational constraints are equational constraints of the projected polynomials. Hence we take the equational constraints one at a time, so if we have k equational constraints we will use $\mathrm{PL}^*_E(A)$ for the first k projections. The following theorems validate the use of $\mathrm{PL}^*_E(A)$ recursively.

Theorem 22. *Suppose f, $g \in \mathbb{R}[x_1, \ldots, x_n]$ both have positive degree in x_n, and that f, g are not identically zero. Let $S \subset \mathbb{R}^{n-1}$ be a connected subset such that f does not have a curtain over S. If $\mathrm{disc}_{x_n}(g)$, $\mathrm{ldcf}_{x_n}(g)$, $\mathrm{trcf}_{x_n}(g)$ and $\mathrm{res}_{x_n}(f, g)$ are all lex-least invariant over S, then g is lex-least invariant on every section and sector of f over S.*

Proof. From Theorem 18 we know that g is sign invariant in the sections of f. Since $\mathrm{disc}_{x_n}(g)$, $\mathrm{ldcf}_{x_n}(g)$ and $\mathrm{trcf}_{x_n}(g)$ are lex-least invariant over S, g is Lazard delineable over S from Theorem 16. Hence g is lex-least invariant in the sections and sectors of f. □

Theorem 23. *Let A be a set of pairwise relatively prime irreducible polynomials in n variables $x_1, \ldots, x_n$ of positive degrees in x_n, where $n \geq 2$, and let E be a subset of A. Let S be a connected subset of $\mathbb{R}^{n-1}$. Suppose that each element of $\mathrm{PL}^*_E(A)$ is lex-least invariant in S. Then each element of E is Lazard delineable over S and exactly one of the following is true.*

1. *The hypersurface defined by the product of the elements of E has a curtain*

over S. Each element of $A \setminus E$ is lex-least invariant in the intersection with the curtain.

2. *The Lazard sections of the elements of E are pairwise disjoint over S. Each element of A is lex-least invariant in such Lazard sections.*

Proof. The resultants are only needed to split cells that are contained in the sectors of $f \in E$ or the curtains of $f \in E$. From Theorem 22 we know that each element of $A \setminus E$ is lex-least invariant on every section of $f \in E$ independently. Since each element of $\text{res}_{x_n}(E, A \setminus E)$ is invariant on S, it follows that each element of $A \setminus E$ is lex-least invariant on all sections of $f \in E$ simultaneously. □

The difference between the projection operators $\text{PL}_E(A)$ and $\text{PL}^*_E(A)$ is that $\text{PL}_E(A)$ does not consider the leading coefficients, trailing coefficients and the discriminants of the non-equational constraints. The method using a single equational constraint is significantly more efficient than Lazard's original method with respect to time and space complexity (see Section 8.2). However, that need not be the case when comparing the multiple equational constraint approach to the single equational constraint approach. That is because it is dependent on the number of constraints in QFF, the number of irreducible factors, the number of variables etc. This is further discussed in Chapter 8.

The usage of the projection operator $\text{PL}^*_E(A)$ has been described in Algorithm 9. Steps 4–7 describe choosing an equational constraint and using the operator $\text{PL}^*_E(A)$. Step 9 is only called if there are equational constraints left to exploit.

Theorem 20 can be extended to show that the non-equational constraints are lex-least invariant on a point curtain of an equational constraint.

Corollary 3. *Let $f, g \in \mathbb{R}[x_1, \ldots, x_n]$ and $\alpha \in \mathbb{R}^{n-1}$. Suppose that f is an equational constraint and V_f has a point curtain at α. Then g is lex-least invariant in the sections and sectors of fg over α.*

Proof. The proof follows directly from above, with minor changes when g has a curtain at α. Since α is a point in $\mathbb{R}^{n-1}$, g is Lazard delineable over α. Thus every Lazard section of fg over α is contained within a Lazard section of g. Furthermore, every Lazard sector of fg over α is either a Lazard sector of f or a Lazard sector of g. Hence g is lex-least invariant in every Lazard section and sector of fg over α. □

Algorithm 9 Modified Lazard CAD Multiple Equational Constraints

$(I, S) \leftarrow \text{MLCADME}(A, E, n)$

Input = Set of polynomials A in n variables and a set of polynomials describing the equational constraints E.

Output = I and S are lists of indices and sample points, respectively, of the cells comprising a lex-least invariant CAD of the intersections of the hypersurfaces described by the elements of E.

```
1: If $n \geq 2$ then go to step 3.
2: Isolate the real roots of the irreducible factors of the non-zero elements of $A$.
   Construct cell indices $I$ and sample points $S$ from the real roots. Exit.
3: $B \leftarrow$ the square free basis of the primitive parts of all elements of $A$.
4: $f \leftarrow$ Select an equational constraint from $E$.
5: $E \leftarrow$ all irreducible factors of $f$.
6: $E' \leftarrow \{\text{res}_{x_n}(f, f') \mid f' \in E \backslash \{f\}\}$.
7: $P \leftarrow \text{cont}(A) \cup \text{PL}^*_E(B)$.
8: If $E'$ is empty go to step 11 else go to step 9.
9: $(I', S') \leftarrow \text{MLCADME}(P, E', n-1)$.
10: Go to step 12.
11: $(I', S') \leftarrow \text{LCAD}(P, n-1)$.
12: $(I, S) \leftarrow$ (empty list,empty list).
13: for each $\alpha \in S'$ do
14:     Let $i$ be the index of the cell containing $\alpha$.
15:     $f^* \leftarrow \{f_\alpha \mid f \in B\}$.
16:     Isolate the real roots of all the polynomials in $f^*$.
17:     Construct cell indices and sample points for Lazard sections and sectors of
        elements of $B$ from $i$, $\alpha$ and the real roots of $f^*$.
18:     Add the sample points to $I$ and $S$.
19: end for
20: return $(I, S)$.
```

Corollary 4. *Let* $f, g_1, \ldots, g_m \in \mathbb{R}[x_1, \ldots, x_n]$ *and* $\alpha \in \mathbb{R}^{n-1}$. *Suppose that* f *is an equational constraint and* V_f *has a point curtain at* α. *Then* $g_1, \ldots, g_m$ *are lex-least invariant in the sections and sectors of* $fg_1 \ldots g_m$ *over* α.

Proof. Let us consider the case $m = 2$. From Corollary 3 we know that f and g_1 are lex-least invariant on sections of fg_1. Applying the same Corollary to fg_1 and g_2, we get that fg_1 and g_2 are lex-least invariant on every section and sector of fg_1g_2. Hence, f, g_1 and g_2 are lex-least invariant in every Lazard section and sector of fg_1g_2 over α. The result follows by trivial induction. □

Corollary 5. *Let* $A \subset \mathbb{R}[x_1, \ldots, x_n]$ *be a set of irreducible polynomials and let* $E \subset A$. *Let* S *be a connected subset of* $\mathbb{R}^{n-1}$ *such that every element of* $\mathrm{PL}^*_E(A)$ *is lex-least invariant in* S. *Then each element of* E *either vanishes identically on* S *or is Lazard delineable on* S. *If* S *is a point, then all elements of* $A \setminus E$ *are lex-least invariant in the Lazard sections and Lazard sectors of* f. *If* S *is not a point, then the sections of any* $f \in E$ *that does not vanish identically over* S *are pairwise disjoint, and each element of* $A \setminus E$ *is lex-least invariant in every such section.*

Proof. This is a direct consequence of Corollary 4 and Theorem 23. □

6.3 Summary

Rewriting the problem of nullification using curtains has given us a better understanding of why our modifications to Lazard's work fails in the presence of curtains. We have used this knowledge to provide a subroutine that circumvents the problems caused by curtains. In doing so, we established that point curtains do not cause problems in our modifications of Lazard's algorithm. We have also described an algorithm that can take advantage of multiple equational constraints in a QFF.

In the next chapter, we adapt our algorithms from Chapters 4–6 to the recent developments made by Brown and McCallum in [BM20].

Chapter 7

Brown-McCallum Projection

Towards the end of 2020, Brown and McCallum defined a modified version of Lazard's projection operator that allows one to omit the trailing coefficients of the polynomial constraints under certain circumstances. In summary, they propose that if a polynomial constraint has only finitely many point curtains, one can omit the trailing coefficient in the projection set.

They do still use the notion of nullification, which we have translated to the terminology of curtains. The notations of [BM20] match those of [MPP19] but here we have used the same notation and terminology that is used elsewhere in this thesis. Most of the results presented in this chapter are a direct consequence of the theorems stated in Chapter 4 and Chapter 6.

7.1 Brown-McCallum Modification

In Chapter 6 we handle curtains in equational constraints and we detect the curtains when lifting. However, Brown and McCallum's improvement [BM20] is to omit the trailing coefficients of a polynomial constraint if we know that it has no curtains or finitely many point curtains. This obliges them to detect curtains in the projection phase instead. Hence our work on detecting curtains is not appropriate for their use.

We define a procedure T that takes as input a polynomial constraint $f \in \mathbb{R}[x_1, \ldots, x_n]$ and outputs either a set $S \subset \mathbb{R}^{n-1}$ or FAIL. The requirement for T is that if it returns a set S, then S must be finite and must contain all the points at which f has a curtain.

There are various ways one could approach this. A simple way would be to show that the system of equations defined by the coefficients of f (in variables $x_1, \dots, x_n$) is unsatisfiable over the reals. This can be achieved using several methods: an easy way would be to use linear substitution. If such a test fails, then we resort to any suitable finite zero-test over $\mathbb{C}$. If the tests mentioned above fail, T returns FAIL. If one of the tests discussed above succeeds, then T computes a suitable finite set $S \subset \mathbb{R}^{n-1}$.

The following defines the Brown-McCallum projection operator, which uses the procedure T to determine which polynomials' trailing coefficients can be omitted. The Brown-McCallum operator also depends on a set of points Γ, which is a list of points that track the bases of point curtains through the projection phase. When Algorithm 10 is first called, Γ is initialised to the empty set.

Definition 49. *[BM20, Definition 4] Let $A \subset \mathbb{R}[x_1, \dots, x_n]$ be a finite irreducible basis, with $n \geq 2$. Let Γ be a finite set of points in $\mathbb{R}^n$. The* Brown-McCallum operator $\mathrm{BM}(A, \Gamma)$ *is a finite subset of $\mathbb{R}[x_1, \dots, x_{n-1}]$ consisting of the following polynomials:*

- *All leading coefficients of the elements of A*
- *All discriminants of the elements of A*
- *All resultants of distinct pairs of elements of A*
- *All trailing coefficients (i.e. coefficients independent of x_n) of the elements of A for which T returns* FAIL.

together with the set of points

$$\{(\gamma_1, \dots, \gamma_{n-1}) \mid (\gamma_1, \dots, \gamma_n) \in \Gamma)\} \cup \bigcup_{f \in \hat{A}} T(f),$$

where $\hat{A} = \{f \mid f \in A \wedge T(f) \neq \mathrm{FAIL}\}$

The following theorem justifies the use of the projection operator $\mathrm{BM}(A, \Gamma)$.

Theorem 24. *[BM20, Theorem 3] Let f be a polynomial in $\mathbb{R}[x_1, \dots, x_n]$ with positive degree in x_n and let S be a connected subset of $\mathbb{R}^{n-1}$. Suppose that the discriminant and leading coefficient of f are lex-least invariant in S and f has no curtain over S. Then f is Lazard delineable over S, hence lex-least invariant in every Lazard section and sector.*

Theorem 24 states that we are allowed to use the projection operator BM if f does not contain a curtain. However, the method used by Brown and McCallum in [BM20] is similar to ours in Theorem 20. The idea stems from the structure of lifting and the fact that it is easy to identify point curtains. In summary, the base of point curtains of an input polynomial at n-level will be a point cell in the $(n-1)$-level.

Algorithm 10 Brown-McCallum algorithm for lex-least invariant CAD

$(I, S) \leftarrow \mathrm{BMCADL}(A)$

Input = Set of polynomials A in n variables and Γ a set of points in $\mathbb{R}^n$.

Output = I and S are lists of indices and sample points, respectively, of the of a lex-least invariant CAD of A.

```
 1: If $n \geq 2$ then go to step 3.
 2: Isolate the real roots of the irreducible factors of the non-zero elements of $A$.
    Construct cell indices $I$ and sample points $S$ from the real roots and the points
    in $\Gamma$. Exit.
 3: $B \leftarrow$ the squarefree basis of the primitive parts of all elements of $A$.
 4: $(P, \Gamma') \leftarrow \mathrm{BM}(B, \emptyset)$.
 5: $(I', S') \leftarrow \mathrm{BMCADL}(\mathrm{cont}(A) \cup P, \Gamma')$.
 6: $(I, S) \leftarrow$ (empty list,empty list).
 7: for each $\alpha \in S'$ do
 8:     Let $i$ be the index of the cell containing $\alpha$.
 9:     $f^* \leftarrow \{f_\alpha \mid f \in B\}$.
10:     Let $D$ be the isolated real roots of $f^*$.
11:     Set $E \leftarrow \{\gamma_n \mid (\alpha_1, \ldots, \alpha_{n-1}, \gamma_n) \in \Gamma\}$.
12:     Construct cell indices and sample points for Lazard sections and sectors of
        elements of $B$ from $i$, $\alpha$ and $D \cup E$.
13:     Add the sample points to $I$ and $S$.
14: end for
15: return $(I, S)$
```

Theorem 25. *Let $A = \{f_1, \ldots, f_m\}$ be a set of pairwise relatively prime irreducible polynomials in n variables $x_1, ..., x_n$ of positive degrees in x_n, where $n \geq 2$. Let Γ be a finite set of points in $\mathbb{R}^n$. Let S be a subset of $\mathbb{R}^{n-1}$ obtained via Algorithm 4 and suppose that each element of* $\mathrm{BM}(A, \Gamma)$ *is lex-least invariant in S. Then every element of A is lex-least invariant in the Lazard sections and sectors of every other element.*

Proof. The proof is similar to the proof of Theorem 17 and is a direct consequence of Thorem 24. □

7.2 Brown-McCallum Projection with Equational Constraints

In [BM20], the authors state that their projection operator should be compatible with the single and multiple equational constraint versions of the Lazard projection operator. In this section we provide the modified versions of the Brown-McCallum projection operator, such that it can be used for single and multiple equational constraint cases.

7.2.1 Single Equational Constraint

The modified version for Brown-McCallum projection operator is not readily translatable from Lazard's version. This is because the Brown-McCallum projection operator needs as input a set Γ of points that describe point curtains in a polynomial constraint.

Definition 50. *Let A be a finite irreducible basis in $\mathbb{R}[x_1, \ldots, x_n]$, with $n \geq 2$. Let Γ be a finite set of points in $\mathbb{R}^n$ and $E \subset A$. The* modified Brown-McCallum projection operator $\mathrm{BM}_E(A, \Gamma)$ *is a finite subset of* $\mathbb{R}[x_1, \ldots, x_{n-1}]$ *containing the following polynomials:*

- *All leading coefficients of the elements of E;*
- *All discriminants of the elements of E;*
- *All resultants $\mathrm{res}_{x_n}(f, g)$ for all $f \in E$, $g \in A$ and $f \neq g$;*
- *All trailing coefficients of the elements of E for which T returns* FAIL*;*

together with the points

$$\{(\gamma_1, \ldots, \gamma_{n-1}) \mid (\gamma_1, \ldots, \gamma_n) \in \Gamma\} \cup \bigcup_{f \in \hat{E}} T(f)$$

where $\hat{E} = \{f \mid f \in E \wedge T(f) \neq \mathrm{FAIL}\}$.

Theorem 26. *Let A be a set of pairwise relatively prime irreducible polynomials in n variables $x_1, \ldots, x_n$ of positive degrees in x_n, where $n \geq 2$. Let E be a subset of A and let Γ be a finite set of points in $\mathbb{R}^n$. Let S be a connected subset of $\mathbb{R}^{n-1}$. Suppose that each element of $\mathrm{BM}_E(A, \Gamma)$ is lex-least invariant in S. Then each element of E is Lazard delineable on S and exactly one of the following must be true.*

1. *The hypersurface defined by the product of the elements of E has a curtain over S.*

2. *The Lazard sections over S of the elements of E are pairwise disjoint. Each element of E is lex-least invariant in every such Lazard section. Each element of $A \setminus E$ is sign invariant in every such Lazard section.*

Proof. In Theorem 18, we require f to be Lazard delineable over a connected subset S without specifying how S is obtained. We can use the projection operator BM to do so. Similarly the only other requirement is that $\mathrm{res}_{x_n}(f, g)$ is lex-least invariant over S. Therefore we can use BM_E. Hence from Theorem 19 and Theorem 25 the result follows.

□

In Theorem 18, we are not concerned with the projection operator used, rather we are concerned with which polynomials are lex-least invariant. Hence the Brown-McCallum projection operator can be used to obtain the lex-least invariance stated in the assumptions of the theorem. This shows that modifying a projection operator to use it for single equational constraints is possible irrespective of the projection operator being used. This is further discussed in Chapter 9.

7.2.2 Multiple Equational Constraints

In this section we present the modified version of the Brown-McCallum projection operator for the multiple equational constraint case. This is a direct consequence of the results in Section 6.2.

Definition 51. *Let A be a finite irreducible basis in $\mathbb{R}[x_1, \ldots, x_n]$, with $n \geq 2$. Let Γ be a finite set of points in $\mathbb{R}^n$ and let $E \subset A$. The* modified Brown-McCallum projection operator *$\mathrm{BM}^*_E(A, \Gamma)$ is a finite subset of $\mathbb{R}[x_1, \ldots, x_{n-1}]$ containing the following polynomials:*

- *All leading coefficients of the elements of A;*
- *All discriminants of the elements of A;*
- *All resultants $\mathrm{res}_{x_n}(f, g)$ for all $f \in E$, $g \in A$ and $f \neq g$;*
- *All trailing coefficients of the elements of A for which T returns* FAIL*;*

Algorithm 11 Brown-McCallum modified algorithm for sign invariant CAD

$(I, S) \leftarrow \mathrm{BMCADS}(A, f, \Gamma, n)$

Input = Set of polynomials A in n variables, Γ a set of points in $\mathbb{R}^n$ and f the polynomial describing the equational constraint $f = 0$.

Output = I and S are lists of indices and sample points, respectively, of the cells of a sign invariant CAD of the hypersurface V_f.

```
 1: If $n \geq 2$, then go to step 3:
 2: Isolate the real roots of the irreducible factors of the non-zero elements of $A$.
    Construct cell indices $I$ and sample points $S$ from the real roots and the points
    in $\Gamma$. Exit.
 3: $B \leftarrow$ the squarefree basis of the primitive parts of all elements of $A$.
 4: $E \leftarrow$ all irreducible factors of $f$ from $B$
 5: $(P, \Gamma') \leftarrow \mathrm{BM}_E(B, \Gamma)$.
 6: $(I', S') \leftarrow \mathrm{BMCADL}(P, n-1)$.
 7: $(I, S) \leftarrow$ (empty list,empty list).
 8: for each $\alpha \in S'$ do
 9:     Let $i$ be the index of the cell containing $\alpha$.
10:     $f^* \leftarrow \{f_\alpha \mid f \in B\}$.
11:     Let $D$ be the isolated real roots of $f^*$.
12:     Set $E \leftarrow \{\gamma_n \mid (\alpha_1, \ldots, \alpha_{n-1}, \gamma_n) \in \Gamma\}$.
13:     Construct cell indices and sample points for Lazard sections and sectors of
        elements of $B$ from $i$, $\alpha$ and $D \cup E$.
14:     Add the sample points to $I$ and $S$.
15: end for
16: return $(I, S)$
```

together with the points

$$\{(\gamma_1, \ldots, \gamma_{n-1}) \mid (\gamma_1, \ldots, \gamma_n) \in \Gamma\} \bigcup_{f \in \hat{A}} T(f)$$

where $\hat{A} = \{f \mid f \in A \wedge T(f) \neq \mathrm{FAIL}\}$.

Theorem 27. *Let A be a set of pairwise relatively prime irreducible polynomials in n variables $x_1, \ldots, x_n$ of positive degrees in x_n, where $n \geq 2$. Let E be a subset of A and let Γ be a finite set of points in $\mathbb{R}^n$. Let S be a connected subset of $\mathbb{R}^{n-1}$. Suppose that each element of $\mathrm{BM}_E^*(A, \Gamma)$ is lex-least invariant in S. Then each element of E is Lazard delineable over S and exactly one of the following is true.*

1. *The hypersurface defined by the product of the elements of E has a curtain over S. Each element of $A \setminus E$ is lex-least invariant in the intersection with the curtain.*
2. *The Lazard sections of the elements of E are pairwise disjoint over S. Each element of A is lex-least invariant in such Lazard sections.*

Proof. This proof follows a similar argument to the proof of Theorem 26. The result follows from Theorem 25 and Theorem 23. □

7.3 Summary

We present our modified verisons of the Brown-McCallum operator for the single and multiple equational constraint case. Throughout Chapter 4 to Chapter 7, we focus on the algorithms used to produce a CAD rather than the valuation used, in our case the lex-least valuation. We discuss this further in the conclusion of this thesis, as a potential avenue to continue the research presented here.

Algorithm 12 Modified Brown-McCallum CAD for Multiple Equational Constraints

$(I, S) \leftarrow \mathrm{BMCME}(A, \Gamma, E, n)$

Input = Set of polynomials A in n variables, Γ a set of points in $\mathbb{R}^n$,
E a set of polynomials describing the equational constraints.

Output = I and S are lists of indices and sample points, respectively, of the cells of a lex-least invariant CAD of the intersections of the hypersurface described by the elements of E.

```
1:  If $n \geq 2$ then go to step 3.
2:  Isolate the real roots of the irreducible factors of the non-zero elements of $A$.
    Construct cell indices $I$ and sample points $S$ from the real roots and the points
    in $\Gamma$. Exit.
3:  $B \leftarrow$ the squarefree basis of the primitive parts of all elements of $A$.
4:  $f \leftarrow$ Select an equational constraint from $E$.
5:  $F \leftarrow$ all irreducible factors of $f$.
6:  $E' \leftarrow \{\mathrm{res}_{x_n}(f, f') \mid f' \in E \backslash \{f\}\}$.
7:  $(P, \Gamma') \leftarrow \mathrm{BM}^*_F(B, \Gamma)$.
8:  If $E'$ is empty go to step 11 else go to step 9.
9:  $(I', S') \leftarrow \mathrm{BMCME}(P, \Gamma', E', n-1)$.
10: Go to step 12.
11: $(I', S') \leftarrow \mathrm{BMCADL}(P, \Gamma', n-1)$.
12: $(I, S) \leftarrow$ (empty list,empty list).
13: for each $\alpha \in S'$ do
14:     Let $i$ be the index of the cell containing $\alpha$.
15:     $f^* \leftarrow \{f_\alpha \mid f \in B\}$.
16:     Let $D$ be the isolated real roots of $f^*$.
17:     Set $E \leftarrow \{\gamma_n \mid (\alpha_1, \ldots, \alpha_{n-1}, \gamma_n) \in \Gamma\}$.
18:     Construct cell indices and sample points for Lazard sections and sectors of
        elements of $B$ from $i$, $\alpha$ and $D \cup E$.
19:     Add the sample points to $I$ and $S$.
20: end for
21: return $(I, S)$.
```

Chapter 8

Complexity Analysis

In this thesis we have provided various improvements to the original algorithms proposed by Lazard [Laz94] and Brown-McCallum [BM20], influenced by [McC99] and [McC01]. This chapter provides a comprehensive complexity analysis of the methods presented in this thesis. Our measure of calculating the complexity of algorithms is the number of cells that a CAD algorithm produces, rather than its time complexity. A variety of previously conducted experiments show a strong correlation between the number of cells computed by a CAD algorithm and computation time [BDE⁺16]. For each algorithm we compute the dominant term in the cell count bound.

We focus on three parameters when conducting the complexity analysis: the number of variables n, the number of polynomials m and the maximum degree d (in any one variable). We first recall terminology used in this chapter.

Definition 52. *Let $\{f_i\}$ be a finite set of polynomials. The* combined maximum degree *of $\{f_i\}$ is the maximum element of the set*

$$\bigcup_j \{\deg_{x_j}(\prod_i f_i)\}.$$

For example, the set $\{x^3 + y, x^4 + y^2\}$ has combined degree 7 (in x), whereas the set $\{x^3 + y^6, x^4 + y^2\}$ has combined degree 8 (in y). Note that the chosen main variable is irrelevant and we are only interested in the maximum possible degree.

Definition 53. *[McC85, Section 6.1] A set of polynomials has the (m, d)-*property *if it can be partitioned into m sets, such that the combined maximum degree of each* *set is less than or equal to d.*

Example 14. *The set of polynomials $\{x^3+y, x^5+y^2, x^5y+x^3, x^3+y\}$ has combined maximum degree 16 and thus the (1,16) - property. If we partition this set into four sets of one polynomial each, it also has the (4,5)-property. We can also split this set into two sets, which would give us the (2,8), (2,10), (2,11) and (2,13)-properties.*

In fact, it is not necessary to require a partition in Definition 53. Equivalently, a set A has the (m, d)-property if it can be decomposed as the union $A = S_1 \cup \ldots \cup S_m$ of possibly overlapping sets, such that the combined maximum degree of each set is less than or equal to d.

Proposition 7. *[BDE⁺16, Proposition 8] If A is a set of polynomials with the (m, d)-property, then a squarefree basis of A will also have the (m, d)-property.*

This can be easily seen, as a squarefree basis of A would only increase the number of polynomials but will not increase the degree of their product.

Proposition 8. *[BDE⁺16, Proposition 9] If A is a set of polynomials with the (m, d)-property, then A has the $(\lceil \frac{m}{l} \rceil, ld)$-property.*

This can also be seen easily, by grouping the subsets together, l at a time.

Referring back to Example 14, we see that the set of polynomials has the (2,8)-property and therefore also has the (1,16)-property, in which case $l = 2$. Let us focus on the (4,5)-property instead. If we take $l = 3$, from Proposition 8 we get that it has the $(2, 15)$-property, which we can see that it does. Indeed, it has the (2,13)-property which according to Definition 54 is stronger.

The main idea behind the analysis is to find an upper bound on the number of roots and then recursively compute the total number of cells being produced in the worst case scenario. We start by analysing the complexity of the projection set.

When projecting with equational constraints we need to use an enhanced version of the (m, d)-property, so that any statement made about projection operators using equational constraints can be used recursively.

Definition 54. *Let A be a set of polynomial factors of a family of polynomial constraints. We say that A has the* $(m, d)_k$-property *if A can be written as the union of m sets each of max combined degree $\leq d$ and each of the first k sets consist of the factors of a single equational constraint.*

This chapter is split into three sections, dedicated to the advancements made on McCallum's algorithm, Lazard's algorithm and the Brown-McCallum algorithm. Within each section we go through the complexity analysis of the original algorithm, followed by the single equational constraint variant and then the multiple equational constraint variant.

If it were the case that the combined maximum degree of all our polynomial constraints were 1, then the system of equations describe a set of polytopes, which can be solved more efficiently using algorithms other than CAD, see for instance [Kar84]. Hence in all our results we make the assumption that the combined max degree d is greater than 1.

8.1 Analysis of McCallum's Projection Operators

Most of the results in this section have been taken from [BDE$^+$16]. We use the analysis presented here as a base for the analysis provided in the remainder of the chapter.

Proposition 9. *[BDE$^+$16, Lemma 11] If A is a set of polynomials with the (m, d)-property, then $\mathrm{P}(A)$ has the $(M, 2d^2)$-property with*

$$M = \left\lfloor \frac{(m+1)^2}{2} \right\rfloor . \tag{8.1}$$

Proof. Let $A = S_1 \cup \ldots \cup S_m$ according to the (m, d)-property. Let B be a squarefree basis of $\mathrm{prim}(A)$. Let $T_1 \subseteq B$ be the factors of S_1 and let $T_i \subseteq B$ be the factors of S_i that are not in T_j with $j < i$. Split $\mathrm{P}(A)$ as follows

$$\mathrm{P}(A) = \mathbf{CA} \bigcup \left\{ \bigcup_i \mathbf{CA}_i \right\} \bigcup \left\{ \bigcup_{i<j} \mathbf{CA}_{i,j} \right\}$$

1. $\mathbf{CA}$: All non-leading coefficients of B.
2. $\mathbf{CA}_i$: $\mathrm{cont}(S_i) \cup \mathrm{ldcf}(T_i) \cup \mathrm{disc}(T_i) \cup \mathrm{res}(T_i)$.
3. $\mathbf{CA}_{i,j}$: $\mathrm{res}(T_i, T_j)$ with $i \neq j$.

By Lemma 3, the set of non-leading coefficients of T_i has the $(1, d^2)$-property. Since $\mathbf{CA}$ is the union of the set of non-leading coefficients for all possible T_i, $\mathbf{CA}$ has the (m, d^2)-property. Using Proposition 8 we get that $\mathbf{CA}$ has the $(\lceil m/2 \rceil, 2d^2)$-property.

By Lemma 4, each $\mathbf{CA}_i$ has the $(1, 2d^2)$-property. There are m possible values for i, so $\cup_i \mathbf{CA}_i$ has the $(m, 2d^2)$-property.

Fix i, j and consider $\operatorname{res}(\prod_{f \in T_i} f, \prod_{g \in T_j} g)$, which contains the product of all resultants in $\operatorname{res}(T_i, T_j)$ by Equation 2.6. This is a resultant of two polynomials with at most degree d, hence the resultant will have at most degree $2d^2$. Applying this to all possible values for i, j, we get that $\cup_{i<j} \mathbf{CA}_{i,j}$ has the $(\frac{1}{2}m(m-1), 2d^2)$-property.

Combining all three we get $\mathrm{P}(A)$ has the $(M, 2d^2)$-property where

$$m + \frac{m(m-1)}{2} + \left\lceil \frac{m}{2} \right\rceil \leq \frac{m(m+1)}{2} + \left\lfloor \frac{m+1}{2} \right\rfloor \leq \left\lfloor \frac{(m+1)^2}{2} \right\rfloor = M \qquad (8.2)$$

□

Lemma 3. *Let T_i be as defined in Proposition 9. The set of all non-leading coefficients of T_i has the $(1, d)$-property.*

Proof. The product of all elements in T_i has at most degree d. This implies that the polynomials in T_i have at most d non-leading coefficients, each of whose degree is at most d. This implies that the set of non-leading coefficients of T_i has the $(1, d^2)$-property. □

Lemma 4. *The set $\mathbf{CA}_i$ as defined in Proposition 9 has the $(1, 2d^2)$-property.*

Proof. Define c as the product of all elements in $\operatorname{cont}(S_i)$. Suppose $T_i = \{F_1, \ldots, F_t\}$ for some t and put $F = cF_1 \ldots F_t$. Thus, F divides the product of all elements in S_i and has degree at most d (since A has the (m, d) property). Let F' be the derivative of F with respect to x_n. Since F has at most degree d, $\operatorname{res}(F, F')$ has degree at most $2d^2$, since it is the determinant of a $(2d-1 \times 2d-1)$ matrix. From Equation 2.4 we get that $\operatorname{res}(F, F') = c' \operatorname{disc}(F)$ where c' is a power of c. Applying Equation 2.5 once gives us

$$\operatorname{disc}(F) = \operatorname{disc}(F_t) \operatorname{disc}(F_1 \ldots F_{t-1}) \operatorname{res}(F_1 \ldots F_{t-1}, F_t)^2. \qquad (8.3)$$

Applying Equation 2.6 to Equation 8.3 repeatedly gives

$$\operatorname{disc}(F) = \operatorname{disc}(F_1 \ldots F_{t-1}) \operatorname{disc}(F_t) \prod_{j=1}^{t-1} (\operatorname{res}(F_j, F_t)^2). \qquad (8.4)$$

Applying both Equation 2.6 and Equation 2.5 recursively we get that $\operatorname{res}(F, F')$ is the product of a power of c and

$$(\prod_{j=1}^{t} \operatorname{ldcf}(F_j) \operatorname{disc}(F_j))(\prod_{1 \leq i < j \leq t} (\operatorname{res}(F_i, F_j))^2). \qquad (8.5)$$

Since this includes all elements of $\mathbf{CA}_i$ we are done. $\square$

In order to get the complexity of the whole process of CAD we must apply this recursively. To make calculations easier we weaken the bound as follows.

Corollary 6. *[McC85] If A is a set of polynomials with the (m, d)-property where $m > 1$ then $P(A)$ has the $(m^2, 2d^2)$-property.*

When computing the upper bound on the number of cells we use the result from Proposition 9 for the first recursion and use Corollary 6 for the subsequent recursions. Table 8.1 describes the growth of the projected polynomials using P(A). Note that columns in Table 8.1 do not refer to the number of polynomials and their degree, but to the number of sets and the max combined degree given by Definition 53. The number of real roots of the projected polynomials determines the size of the CAD. We can bound the number of real roots of the univariate polynomials by the product of the elements in columns 2 and 3 of Table 8.1. The number of cells produced in $\mathbb{R}^1$ is twice this product plus 1. Hence the total number of cells in the CAD can be given by the product of $2K + 1$ where K varies through all the products between column 2 and 3 of Table 8.1, which is given by

Table 8.1: Growth of polynomials in CAD

Variables	Number m_i	Degree d_i	Product $m_i d_i$
n	m	d	md
$n-1$	M	$2d^2$	$2Md^2$
$n-2$	M^2	$8d^4$	$2^3M^2d^4$
$n-3$	M^4	$128d^8$	$2^7M^4d^8$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$n-r$	$M^{2^{r-1}}$	$2^{2^r-1}d^{2^r}$	$2^{2^r-1}M^{2^{r-1}}d^{2^r}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
1	$M^{2^{n-2}}$	$2^{2^{n-1}-1}d^{2^{n-1}}$	$2^{2^{n-1}-1}M^{2^{n-2}}d^{2^{n-1}}$
Product	$M^{2^{n-1}-1}m$	$2^{2^n-1-n}d^{2^n-1}$	$2^{2^n-1-n}M^{2^{n-1}-1}md^{2^n-1}$

$$(2md+1)\prod_{i=1}^{n-1}(2(2^{2^i-1}M^{2^{i-1}}d^{2^i})+1). \tag{8.6}$$

Since we are interested in the complexity, we can omit the '+1' term in the product to give us the dominant term of the expression:

$$2md\prod_{i=1}^{n-1}(2(2^{2^i-1}M^{2^{i-1}}d^{2^i})) = 2^{2^n-1}M^{2^{n-1}-1}md^{2^n-1}. \tag{8.7}$$

Substituting the value for M from Equation 8.2 we get the upper bound

$$2^{2^n-1}\left(\frac{(m+1)^2}{2}\right)^{2^{n-1}-1}md^{2^n-1} = 2^{2^{n-1}}(m+1)^{2^n-2}md^{2^n-1}. \tag{8.8}$$

This bound is doubly exponential with respect to n. Hence even a small jump from $n = 4$ to $n = 5$ has a drastic effect on the time and space complexity of the algorithm.

8.1.1 Single Equational Constraint McCallum

Let us now consider McCallum's modified projection operator $\mathrm{P}_E(A)$ for the single equational constraint case. Assuming there is an equational constraint, it is clear that $\mathrm{P}_E(A) \subseteq \mathrm{P}(A)$ from their definitions. Since $\mathrm{P}_E(A)$ is used only at the first level when computing a CAD (i.e. when projecting from n to $n-1$), we can use Table 8.1 for our complexity calculations with a different value for M.

Theorem 28. *Let A be a set of polynomial factors of a family of polynomial constraints. Suppose that A has the $(m, d)_k$-property with $d \geq 2$ and $k \geq 1$. Let E be the first set of the $(m, d)_k$ decomposition and let F be the squarefree basis of E. Then $\mathrm{P}_F(A)$ has the $(M, 2d^2)_{k'}$-property (with respect to the family of projected polynomial constraints) where*

$$M = \left\lfloor \frac{5m+4}{4} \right\rfloor \tag{8.9}$$

and $k' < k$.

Proof. Let $A = S_1 \cup ... \cup S_m$ according to the $(m, d)_k$-property, so that $E = S_1$ and $S_2, ..., S_k$ are associated with equational constraints. Since E has the $(1, d)$-property, its squarefree basis F has the $(1, d)$-property.

- From the proof of Proposition 9 we see that the set of leading coefficients, discriminants, resultants and contents of F form a set R_1 which has the $(1, 2d^2)$-property.

- By Lemma 3 the remaining coefficients of F form a set R_2 which has the $(1, d^2)$-property. Set $R_3 = \operatorname{cont}(A) \setminus \operatorname{cont}(E)$. Clearly $R_3 \subseteq \operatorname{cont}(A \setminus E)$ and $A \setminus E$ decomposes into $(m-1)$ sets each of max combined degree d. Hence the set R_3 has the $(m-1, d)$-property, thus the $(\lceil \frac{m-1}{2} \rceil, d^2)$-property. Then $R_2 \cup R_3$ has the $(\lceil \frac{m-1}{2} \rceil + 1, d^2)$-property; hence by Proposition 8 the $\left(\left\lceil \frac{\lceil \frac{m-1}{2} \rceil + 1,}{2} \right\rceil, 2d^2\right)$-property. Hence we get that $R_2 \cup R_3$ has the $(\lfloor \frac{m+3d-2}{2d} \rfloor, 2d^2)$-property.
- From the proof of Proposition 9, $\operatorname{res}(S_i, F)$ has the $(1, 2d^2)$-property for $i > 1$. Hence $R_4 = \bigcup_{i>1} \operatorname{res}(S_i, F)$ has the $(m-1, 2d^2)$-property.

Hence the set $\mathrm{P}_F(A) = R_1 \cup R_2 \cup R_3 \cup R_4$ has the $(M, 2d^2)$-property where

$$1 + \left\lfloor \frac{m+3d-2}{2d} \right\rfloor + (m-1) \leq \left\lfloor \frac{5m+4}{4} \right\rfloor = M. \tag{8.10}$$

Moreover, the decomposition of R_4 according to the $(m-1, 2d^2)$-property is as given above, and the sets in the decomposition of $\mathrm{P}_F(A)$ associated with a projected equational constraint are precisely the nonzero $\operatorname{res}(S_i, F)$ for $1 < i \leq k$, so $\mathrm{P}_F(A)$ has the $(M, 2d^2)'_k$-property for some $k' < k$.

□

To obtain the dominant term on the cell bound we substitute the value from Theorem 28 in Equation 8.7 and get

$$2^{2^n-1} \left(\frac{5m+4}{4}\right)^{2^{n-1}-1} md^{2^n-1} = 2^{2^{n-1}} \left(\frac{5m+4}{2}\right)^{2^{n-1}-1} md^{2^n-1}. \tag{8.11}$$

Comparing the dominant terms given by Equation 8.8 and Equation 8.11, we see that

$$(m+1)^2 \geq \frac{5m+4}{2} \text{ (with } m > 1), \tag{8.12}$$

which implies that the complexity is reduced when using $\mathrm{P}_E(A)$ instead of $\mathrm{P}(A)$.

8.1.2 Multiple Equational Constraint McCallum

In this section we analyse McCallum's final modification for order invariant CADs, $\mathrm{P}^*_E(A)$ for multiple equational constraints. In the process we also fill a gap in a proof in [EBD19].

Theorem 29. *Let A be a set of polynomial factors of a family of polynomial constraints. Suppose that A has the $(m, d)_k$-property with $d \geq 2$ and $k \geq 1$. Let E be the first set of the $(m, d)_k$ decomposition and let F be the squarefree basis of E. Then $\mathrm{P}_F(A)$ has the $(3m, 2d^2)_{k'}$-property (with respect to the family of projected polynomial constraints) where $k' < k$.*

Proof. From Theorem 28 we know that $\mathrm{P}_F(A)$ decomposes into $\lfloor \frac{5m+4}{4} \rfloor$ sets, each of maximum combined degree $2d^2$. Each set in the decomposition of $A \setminus E$ has a max combined degree of d, hence will generate a discriminant of degree $2d(d-1)$ and $(d+1)$ coefficients of degree d. Taking the discriminant and leading coefficients together and all the non-leading coefficients together, we get two sets that have the $(1, 2d^2)$-property and the $(1, d^2)$-property respectively.

Since there are $(m-1)$ sets in the decomposition of $A \setminus E$, the set of discriminants and leading coefficients has the $(m-1, 2d^2)$-property and the set of non-leading coefficients has the $(m-1, d^2)$-property, and by Proposition 8 the $(\lceil \frac{m-1}{2} \rceil, 2d^2)$-property. Hence the projection $\mathrm{P}^*_F(A)$ has the $(M, 2d^2)$-property where

$$\left\lfloor \frac{5m+4}{4} \right\rfloor + (m-1) + \left\lceil \frac{m-1}{2} \right\rceil \leq \left\lfloor \frac{11m}{4} \right\rfloor \leq 3m = M. \tag{8.13}$$

There are k equational constraints, so Theorem 28 implies that $\mathrm{P}^*_F(A)$ has the $(M, 2d^2)_{k'}$-property (with respect to the family of projected polynomial constraints) where $k' < k$. ☐

If there are l equational constraints, we can apply this result in the first l rows in Table 8.2. For the remaining rows we use Corollary 6.

The total number of cells in the CAD is given by the product of $2K + 1$ where K varies through all the products between column 2 and 3 of the Table 8.2, and we get

$$\prod_{i=1}^{n}(2m_i d_i + 1) = \prod_{i=1}^{n-(l+1)} (2m_i d_i + 1) \times \prod_{s=n-l}^{n} (2m_s d_s + 1). \tag{8.14}$$

Since we are only concerned with the dominant term, we omit the '+1' and get

$$\prod_{i=1}^{n-(l+1)} (2m_i d_i) = \prod_{i=1}^{n-(l+1)} (3^{2^r l} m^{2^r} 2^{2^{l+r}-1} d^{2^{l+r}}) = 3^{ls^{n-1}-2l} m^{2^{n-l}-2} (2d)^{2^n - 2^{l+1}} \tag{8.15}$$

$$\prod_{s=n-l}^{n} (2m_s d_s) = \prod_{s=n-1}^{n} 3^s m 2^{2^s-1} d^{2^s} = 3^{l(l+1)/2} m^{l+1} (2d)^{2^{l+1}-1}. \tag{8.16}$$

Table 8.2: Growth of polynomials in CAD

Variables	Number m_i	Degree d_i	Product $m_i d_i$
n	m	d	md
$n-1$	$3m$	$2d^2$	$2 \cdot 3md^2$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$n-l$	$3^l m$	$2^{2^l-1} d^{2^l}$	$3^l m 2^{2^l-1} d^{2^l}$
$n-(l+1)$	$3^{2l} m^2$	$2^{2^{l+1}-1} d^{2^{l+1}}$	$3^{2l} m^2 2^{2^{l+1}-1} d^{2^{l+1}}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$n-(l+r)$	$3^{2^r l} m^{2^r}$	$2^{2^{l+r}-1} d^{2^{l+r}}$	$3^{2^r l} m^{2^r} 2^{2^{l+r}-1} d^{2^{l+r}}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
1	$3^{2^{n-l-1} l} m^{2^{n-l-1}}$	$2^{2^{n-1}-1} d^{2^{n-1}}$	$3^{2^{n-l-1} l} m^{2^{n-l-1}} 2^{2^{n-1}-1} d^{2^{n-1}}$

Combining these two we get

$$3^{l2^{n-l}+l(l-3)/2} m^{2^{n-l}+l-1} (2d)^{2^n-1}. \tag{8.17}$$

8.2 Analysis of Lazard's Projection Operators

All the results in this section are similar to the results in Section 8.1. We start with Lazard's original projection operator PL(A). The key difference between PL(A) and P(A) is that PL(A) does not use the middle coefficients of the elements of A.

Theorem 30. *Let A be a set of polynomials with the (m, d)-property with $d \geq 2$. Then* PL(A) *has the $(M, 2d^2)$-property with*

$$M = \left\lfloor \frac{(m+1)^2}{2} \right\rfloor$$

Proof. Let $A = S_1 \cup \ldots \cup S_m$ according to the (m, d)-property. Let B be a squarefree basis of prim(A). Let $T_1 \subseteq B$ be the factors of S_1 and let $T_i \subseteq B$ be the factors of S_i that are not in T_j with $j < i$. Split PL(A) as follows

$$\mathrm{PL}(A) = \mathbf{CA} \bigcup \left\{ \bigcup_i \mathbf{CA}_i \right\} \bigcup \left\{ \bigcup_{i<j} \mathbf{CA}_{i,j} \right\}$$

1. **CA**: All trailing coefficients of B.

2. $\mathbf{CA}_i$: $\operatorname{cont}(S_i) \cup \operatorname{ldcf}(T_i) \cup \operatorname{disc}(T_i) \cup \operatorname{res}(T_i)$.
3. $\mathbf{CA}_{i,j}$: $\operatorname{res}(T_i, T_j)$ with $i \neq j$.

The product of all elements of T_i has at most degree d, hence the set of of trailing coefficients of T_i has the $(1, d)$-property. Therefore **CA** has the (m, d)-property and by Proposition 8 it has the $(\left\lceil \frac{m}{2d} \right\rceil, 2d^2)$-property.

By Lemma 4, each $\mathbf{CA}_i$ has the $(1, 2d^2)$-property. There are m possible values for i, so $\cup_i \mathbf{CA}_i$ has the $(m, 2d^2)$-property.

Fix i, j and consider $\operatorname{res}(\prod_{f \in T_i} f, \prod_{g \in T_j} g)$, which contains the product of all resultants in $\operatorname{res}(T_i, T_j)$ by Equation 2.6. This is a resultant of two polynomials with at most degree d, hence the resultant will have at most degree $2d^2$. Applying this to all possible values for i, j, we get that $\cup_{i<j} \mathbf{CA}_{i,j}$ has the $(\frac{1}{2}m(m-1), 2d^2)$-property.

Combining all three we get $\operatorname{PL}(A)$ has the $(M, 2d^2)$-property where

$$m + \frac{m(m-1)}{2} + \left\lceil \frac{m}{2d} \right\rceil \leq \left\lfloor \frac{m^2 d + md + m + 2d - 1}{2} \right\rfloor \leq \left\lfloor \frac{(m+1)^2}{2} \right\rfloor = M.$$

$\square$

Corollary 7. *If A is a set of polynomials with the (m, d)-property where $m > 1$ then* $\operatorname{PL}(A)$ *has the $(m^2, 2d^2)$-property.*

Substituting the value for M from Theorem 30 into Equation 8.7 we get the following upper bound:

$$2^{2^n-1} m \left(\frac{(m+1)^2}{2} \right)^{2^{n-1}-1} d^{2^n-1} = 2^{2^{n-1}} m((m+1)^2)^{2^{n-1}-1} d^{2^n-1}. \tag{8.18}$$

8.2.1 Single Equational Constraint Lazard

Let us now consider our modification of Lazard's projection operator $\operatorname{PL}_E(A)$ for the single equational constraint case. Since $\operatorname{PL}_E(A)$ is used only at the first level when computing a CAD (i.e. when projecting from n to $n-1$), we can use Table 8.1 for our complexity calculations with a different value for M.

Theorem 31. *Let A be a set of polynomial factors of a family of polynomial constraints. Suppose that A has the $(m, d)_k$-property with $d \geq 2$ and $k \geq 1$. Let E be the first set of the $(m, d)_k$ decomposition and let F be the squarefree basis of E.*

Then $\mathrm{PL}_F(A)$ *has the* $(M, 2d^2)_{k'}$*-property (with respect to the family of projected polynomial constraints) where*

$$M = \left\lfloor \frac{5m+3}{4} \right\rfloor .$$

Proof. Let $A = S_1 \cup ... \cup S_m$ according to the $(m, d)_k$-property, so that $E = S_1$ and $S_2, ..., S_k$ are associated with equational constraints. Since E consists of a single polynomial (by assumption), its squarefree basis F has the $(1, d)$-property.

- From the proof of Proposition 9 we see that the set of leading coefficients, discriminants, resultants and contents of F form a set R_1 which has the $(1, 2d^2)$-property.
- The trailing coefficients of F form a set R_2 which has the $(1, d)$-property. Set $R_3 = \mathrm{cont}(A) \setminus \mathrm{cont}(E)$. Clearly $R_3 \subseteq \mathrm{cont}(A \setminus E)$ and $A \setminus E$ decomposes into $(m-1)$ sets each of max combined degree d. Hence set R_3 has the $(m-1, d)$-property. Then $R_2 \cup R_3$ has the (m, d)-property; hence by Proposition 8 the $(\lceil \frac{m}{2d} \rceil, 2d^2)$-property.
- From the proof of Proposition 9, $\mathrm{res}(S_i, F)$ has the $(1, 2d^2)$-property for $i > 1$. Hence $R_4 = \bigcup_{i>1} \mathrm{res}(S_i, F)$ has the $(m-1, 2d^2)$-property.

Hence the set $\mathrm{PL}_F(A) = R_1 \cup R_2 \cup R_3 \cup R_4$ has the $(M, 2d^2)$-property where

$$1 + \left\lceil \frac{m}{2d} \right\rceil + (m-1) \leq \left\lfloor \frac{2md + m + 2d - 1}{2d} \right\rfloor \leq \left\lfloor \frac{5m+3}{4} \right\rfloor = M. \tag{8.19}$$

Moreover, the decomposition of R_4 according to the $(m-1, 2d^2)$-property is as given above, and the sets in the decomposition of $\mathrm{PL}_F(A)$ associated with a projected equational constraint are precisely the nonzero $\mathrm{res}(S_i, F)$ for $1 < i \leq k$, so $\mathrm{PL}_F(A)$ has the $(M, 2d^2)'_k$-property for some $k' < k$. □

Substituting the value for M from Theorem 31 into Equation 8.7 we get the following upper bound:

$$2^{2^n - 1} m \left(\frac{5m+3}{4} \right)^{2^{n-1}-1} d^{2^n - 1} = 2^{2^{n-1}} m \left(\frac{5m+3}{2} \right)^{2^{n-1}-1} d^{2^n - 1}. \tag{8.20}$$

8.2.2 Multiple Equational Constraint Lazard

In this section we analyse our modification to Lazard's projection operator for the multiple equational constraint case. To show the difference in the complexity between $\mathrm{P}_E^*(A)$ and $\mathrm{PL}_E^*(A)$ we use Table 8.3.

Theorem 32. *Let A be a set of polynomial factors of a family of polynomial constraints. Suppose that A has the $(m,d)_k$-property and $k \geq 1$. Let E be the first set of the $(m,d)_k$ decomposition and let F be the squarefree basis of E. Then $\mathrm{PL}_F^*(A)$ has the $(9m/4, 2d^2)_{k'}$-property (with respect to the family of projected polynomial constraints) where $k' < k$.*

Proof. From Theorem 31 we know that $\mathrm{PL}_F(A)$ decomposes into $\lfloor \frac{5m+3}{4} \rfloor$ sets, each of maximum combined degree $2d^2$. Each set in the decomposition of $A \setminus E$ will generate a discriminant of degree $2d(d-1)$ and 2 coefficients of degree d (leading and trailing coefficients). Hence each polynomial gives a set with the $(1, 2d^2)$-property.

Since there are $(m-1)$ sets in the decomposition of $A \setminus E$, the set of discriminants, leading coefficients and trailing coefficients has the $(m-1, 2d^2)$-property and the set of non-leading coefficients has the $(m-1, d^2)$-property. Hence the projection $\mathrm{P}_F^*(A)$ has the $(M, 2d^2)$-property where

$$\left\lfloor \frac{5m+3}{4} \right\rfloor + (m-1) = \left\lfloor \frac{9m-1}{4} \right\rfloor \leq \left(\frac{9m}{4}\right). \tag{8.21}$$

There are k equational constraints, so Theorem 31 implies that $\mathrm{PL}_F^*(A)$ has the $(M, 2d^2)_{k'}$-property (with respect to the family of projected polynomial constraints) where $k' < k$. □

The number of real roots of the univariate polynomials is given by the product of the elements in columns 2 and 3 of Table 8.3. The number of cells produced in $\mathbb{R}^1$ is twice this product plus 1. Hence the total number of cells in the CAD is the product of $2K+1$ where K varies through all the products between column 2 and 3 of Table 8.3, which is given by

$$(2md+1)\prod_{i=1}^{n}(2m_i d_i + 1) = (2md+1)\prod_{i=n}^{n-l}(2m_i d_i + 1) \times \prod_{s=n-l-1}^{1}(2m_s d_s + 1). \tag{8.22}$$

Since we are only concerned with the dominant term, we remove all the '+1' in equation 8.22.

Table 8.3: Growth of polynomials in CAD

Variables	Number m_i	Degree d_i	Product $m_i d_i$
n	m	d	md
$n-1$	$\frac{9}{4}m$	$2d^2$	$2 \cdot \frac{9}{4}md^2$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$n-l$	$(\frac{9}{4})^l m$	$2^{2^l-1}d^{2^l}$	$(\frac{9}{4})^l m 2^{2^l-1}d^{2^l}$
$n-(l+1)$	$(\frac{9}{4})^{2l} m^2$	$2^{2^{l+1}-1}d^{2^{l+1}}$	$(\frac{9}{4})^{2l} m^2 2^{2^{l+1}-1}d^{2^{l+1}}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$n-(l+r)$	$(\frac{9}{4})^{2^r l} m^{2^r}$	$2^{2^{l+r}-1}d^{2^{l+r}}$	$(\frac{9}{4})^{2^r l} m^{2^r} 2^{2^{l+r}-1}d^{2^{l+r}}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
1	$(\frac{9}{4})^{2^{n-l-1}l} m^{2^{n-l-1}}$	$2^{2^{n-1}-1}d^{2^{n-1}}$	$(\frac{9}{4})^{2^{n-l-1}l} m^{2^{n-l-1}} 2^{2^{n-1}-1}d^{2^{n-1}}$

$$
\begin{aligned}
\prod_{i=1}^{n-(l+1)} (2m_i d_i) &= \prod_{i=1}^{n-(l+1)} (\left(\frac{9}{4}\right)^{2^r l} m^{2^r} 2^{2^{l+r}-1} d^{2^{l+r}}) \\
&= \left(\frac{9}{4}\right)^{ls^{n-1}-2l} m^{2^{n-l}-2} (2d)^{2^n-2^{l+1}}
\end{aligned}
\tag{8.23}
$$

$$
\prod_{s=n-l}^{n} (2m_s d_s) = \prod_{s=n-1}^{n} \left(\frac{9}{4}\right)^{s} m 2^{2^s-1} d^{2^s} = \left(\frac{9}{4}\right)^{l(l+1)/2} m^{l+1} (2d)^{2^{l+1}-1}. \tag{8.24}
$$

Combining these two we get

$$
\left(\frac{9}{4}\right)^{l2^{n-l}+l(l-3)/2} m^{2^{n-l}+l-1} (2d)^{2^n-1}. \tag{8.25}
$$

8.3 Analysis of Brown-McCallum Projection Operators

We start this section by presenting the complexity analysis of the Brown-Mccallum projection operator found in [BM20]. Further to this, we present the complexity analysis for our modifications of the Brown-McCallum projection operator. In Chapter 7 these projection operators are called $\mathrm{BM}(A, \Gamma)$, $\mathrm{BM}_E(A, \Gamma)$ and $\mathrm{BM}_E^*(A, \Gamma)$. Since Γ does not play a role in computing the complexity, we have suppressed it from the notation for simplicity.

Theorem 33. *Let A be a set of polynomials with the (m, d)-property. Then* $\mathrm{BM}(A)$ *has the $(M, 2d^2)$-property where*

$$M = \frac{m(m+1)}{2}.$$

Proof. Let $A = S_1 \cup \ldots \cup S_m$ according to the (m, d)-property. Let B be a squarefree basis of $\mathrm{prim}(A)$. Let $T_1 \subseteq B$ be the factors of S_1 and let $T_i \subseteq B$ be the factors of S_i that are not in T_j with $j < i$. Split $\mathrm{BM}(A)$ as follows

$$\mathrm{BM}(A) = \left\{ \bigcup_i \mathbf{CA}_i \right\} \bigcup \left\{ \bigcup_{i<j} \mathbf{CA}_{i,j} \right\}$$

1. $\mathbf{CA}_i$: $\mathrm{cont}(S_i) \cup \mathrm{ldcf}(T_i) \cup \mathrm{disc}(T_i) \cup \mathrm{res}(T_i)$.
2. $\mathbf{CA}_{i,j}$: $\mathrm{res}(T_i, T_j)$ with $i \neq j$.

By Lemma 4, each $\mathbf{CA}_i$ has the $(1, 2d^2)$-property. There are m possible values for i, so $\cup_i \mathbf{CA}_i$ has the $(m, 2d^2)$-property.

Fix i, j and consider $\mathrm{res}(\prod_{f \in T_i} f, \prod_{g \in T_j} g)$, which contains the product of all resultants in $\mathrm{res}(T_i, T_j)$ by Equation 2.6. This is a resultant of two polynomials with at most degree d, hence the resultant will have at most degree $2d^2$. Applying this to all possible values for i, j, we get that $\cup_{i<j} \mathbf{CA}_{i,j}$ has the $(\frac{1}{2}m(m-1), 2d^2)$-property.

Combining both sets we get $\mathrm{BM}(A)$ has the $(M, 2d^2)$ property where $M = \frac{m(m+1)}{2}$. □

Corollary 8. *Let A be a set of polynomials with the (m, d)-property where $m > 1$, then* $\mathrm{BM}(A)$ *has the $(m^2, 2d^2)$-property.*

Substituting the value for M from Theorem 33 into Equation 8.7 we get the following upper bound:

$$2^{2^n-1} m \left(\frac{m(m+1)}{2} \right)^{2^{n-1}-1} d^{2^n-1} = 2^{2^{n-1}} m(m(m+1))^{2^{n-1}-1} d^{2^n-1}. \tag{8.26}$$

8.3.1 Single Equational Constraint Brown-McCallum

Let us now consider our modification of the Brown-McCallum projection operator $\mathrm{BM}_E(A)$ for the single equational constraint case. Since $\mathrm{BM}_E(A)$ is used only at the first level when computing a CAD (i.e. when projecting from n to $n-1$), we can use Table 8.1 for our complexity calculations with a different value for M.

Theorem 34. *Let A be a set of polynomial factors of a family of polynomial constraints. Suppose that A has the $(m,d)_k$-property with $d \geq 2$ and $k \geq 1$. Let E be the first set of the $(m,d)_k$ decomposition and let F be the squarefree basis of E. Then $\mathrm{BM}_F(A)$ has the $(M, 2d^2)_{k'}$-property (with respect to the family of projected polynomial constraints) where*

$$M = \left\lfloor \frac{5m+2}{4} \right\rfloor .$$

Proof. Let $A = S_1 \cup ... \cup S_m$ according to the $(m,d)_k$-property, so that $E = S_1$ and $S_2, ..., S_k$ are associated with equational constraints. Since E consists of a single polynomial, its squarefree basis F has the $(1,d)$-property.

- From the proof of Proposition 9 we see that the set of leading coefficients, discriminants and contents of F form a set R_1 which has the $(1, 2d^2)$-property.
- Set $R_2 = \mathrm{cont}(A) \setminus \mathrm{cont}(E)$. Clearly $R_2 \subseteq \mathrm{cont}(A \setminus E)$ and $A \setminus E$ decomposes into $(m-1)$ sets each of max combined degree d. Hence R_2 has the $(m-1,d)$-property and by Proposition 8 the $(\lceil \frac{m-1}{2d} \rceil, 2d^2)$-property.
- From the proof of Proposition 9, $\mathrm{res}(S_i, F)$ has the $(1, 2d^2)$-property for $i > 1$. Hence $R_3 = \bigcup_{i>1} \mathrm{res}(S_i, F)$ has the $(m-1, 2d^2)$-property.

Hence the set $\mathrm{BM}_E(A) = R_1 \cup R_2 \cup R_3$ has the $(M, 2d^2)$-property where

$$1 + \left\lceil \frac{m-1}{2d} \right\rceil + (m-1) \leq \left\lfloor \frac{2md + m + 2d - 2}{2d} \right\rfloor \leq \left\lfloor \frac{5m+2}{4} \right\rfloor = M \tag{8.27}$$

Moreover, the decomposition of R_3 according to the $(m-1, 2d^2)$-property is as given above, and the sets in the decomposition of $\mathrm{BM}_F(A)$ associated with a projected equational constraint are precisely the nonzero $\mathrm{res}(S_i, F)$ for $1 < i \leq k$, so $\mathrm{BM}_F(A)$ has the $(M, 2d^2)'_k$-property for some $k' < k$. ☐

Substituting the value for M from Theorem 34 into equation 8.7 we get the following upper bound:

$$2^{2^n-1} m \left(\frac{5m+2}{4} \right)^{2^{n-1}-1} d^{2^n-1} = 2^{2^{n-1}} m \left(\frac{5m+2}{2} \right)^{2^{n-1}-1} d^{2^n-1}. \tag{8.28}$$

8.3.2 Multiple Equational Constraint Brown-McCallum

To compute the complexity of our modified projection operator $\mathrm{BM}_E^*(A)$ we use Table 8.4 as we find a different weaker bound for $\mathrm{P}_E^*(A)$ and $\mathrm{PL}_E^*(A)$.

Theorem 35. *Let A be a set of polynomial factors of a family of polynomial constraints. Suppose that A has the $(m,d)_k$-property with $d \geq 2$ and $k \geq 1$. Let E be the first set of the $(m,d)_k$ decomposition and let F be the squarefree basis of E. Then $\mathrm{BM}_F^*(A)$ has the $(9m/4, 2d^2)_{k'}$-property (with respect to the family of projected polynomial constraints) where $k' < k$.*

Proof. From Theorem 34 we know that $\mathrm{BM}_F(A)$ decomposes into $\left\lfloor \frac{5m+2}{4} \right\rfloor$ sets, each of maximum combined degree $2d^2$. Each polynomial in $A \setminus E$ will generate a discriminant of degree $2d(d-1)$ and 1 coefficient of degree d (leading coefficient). Hence each polynomial gives a set with the $(1, 2d^2-d)$-property, so trivially the $(1, 2d^2)$-property.

Since there are $(m-1)$ partitions in $A \setminus E$, the set of discriminants and leading coefficients has the $(m-1, 2d^2)$-property. Hence the projection $\mathrm{BM}_F^*(A)$ has the $(M, 2d^2)$-property where

$$\left\lfloor \frac{5m+2}{4} \right\rfloor + (m-1) = \left\lfloor \frac{9m-2}{4} \right\rfloor \leq \frac{9m}{4} = M. \tag{8.29}$$

There are k equational constraints, so Theorem 34 implies that $\mathrm{BM}_F^*(A)$ has the $(M, 2d^2)_{k'}$-property (with respect to the family of projected polynomial constraints) where $k' < k$. □

The number of real roots of the projected polynomials determines the size of the CAD. We can bound the number of real roots of the univariate polynomials by the product of the elements in columns 2 and 3 of Table 8.4. The number of cells produced in $\mathbb{R}^1$ is twice this product plus 1. Hence the total number of cells in the CAD is given by the product of $2K+1$ where K varies through all the products between column 2 and 3 of Table 8.4, which is given by

$$(2md+1)\prod_{i=1}^{n}(2m_i d_i + 1) = (2md+1)\prod_{i=n}^{n-l}(2m_i d_i + 1) \times \prod_{s=n-l-1}^{1}(2m_s d_s + 1). \tag{8.30}$$

Table 8.4: Growth of polynomials in CAD

Variables	Number m_i	Degree d_i	Product m_id_i
n	m	d	md
$n-1$	$\frac{9}{4}m$	$2d^2$	$2\cdot\frac{9}{4}md^2$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$n-l$	$(\frac{9}{4})^l m$	$2^{2^l-1}d^{2^l}$	$(\frac{9}{4})^l m2^{2^l-1}d^{2^l}$
$n-(l+1)$	$(\frac{9}{4})^{2l}m^2$	$2^{2^{l+1}-1}d^{2^{l+1}}$	$(\frac{9}{4})^{2l}m^2 2^{2^{l+1}-1}d^{2^{l+1}}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$n-(l+r)$	$(\frac{9}{4})^{2^r l}m^{2^r}$	$2^{2^{l+r}-1}d^{2^{l+r}}$	$(\frac{9}{4})^{2^r l}m^{2^r}2^{2^{l+r}-1}d^{2^{l+r}}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
1	$(\frac{9}{4})^{2^{n-l-1}l}m^{2^{n-l-1}}$	$2^{2^{n-1}-1}d^{2^{n-1}}$	$(\frac{9}{4})^{2^{n-l-1}l}m^{2^{n-l-1}}2^{2^{n-1}-1}d^{2^{n-1}}$

Since we are only concerned with the dominant term, we omit all the '+1' in Equation 8.22.

$$\begin{aligned}\prod_{i=1}^{n-(l+1)}(2m_id_i) &= \prod_{i=1}^{n-(l+1)}(\left(\frac{9}{4}\right)^{2^r l} m^{2^r}2^{2^{l+r}-1}d^{2^{l+r}}) \\ &= \left(\frac{9}{4}\right)^{ls^{n-1}-2l} m^{2^{n-l}-2}(2d)^{2^n-2^{l+1}}\end{aligned} \tag{8.31}$$

$$\prod_{s=n-l}^{n}(2m_sd_s) = \prod_{s=n-1}^{n}\left(\frac{9}{4}\right)^s m2^{2^s-1}d^{2^s} = \left(\frac{9}{4}\right)^{l(l+1)/2} m^{l+1}(2d)^{2^{l+1}-1}. \tag{8.32}$$

Combining these two we get

$$\left(\frac{9}{4}\right)^{l2^{n-l}+l(l-3)/2} m^{2^{n-l}+l-1}(2d)^{2^n-1}. \tag{8.33}$$

8.4 Summary

The outcome of the complexity analysis carried out in this chapter is summarised in Table 8.5. If the set of inputs has the (m, d)-property then the set of projected polynomials after projection has the $(M, 2d^2)$-property, with M as in Table 8.5.

Table 8.5: Growth of polynomials in CAD

Theory by	Original	Single EC	Multiple EC
McCallum	$\left\lfloor \frac{(m+1)^2}{2} \right\rfloor$	$\left\lfloor \frac{5m+4}{4} \right\rfloor$	$\left\lfloor \frac{11m}{4} \right\rfloor$
Lazard	$\left\lfloor \frac{(m+1)^2}{2} \right\rfloor$	$\left\lfloor \frac{5m+3}{4} \right\rfloor$	$\left\lfloor \frac{9m-1}{4} \right\rfloor$
Brown-McCallum	$\frac{m(m+1)}{2}$	$\left\lfloor \frac{5m+2}{4} \right\rfloor$	$\left\lfloor \frac{9m-2}{4} \right\rfloor$

This shows that, if we have only one equational constraint, the “Single EC” methods are better. However, if we have multiple equational constraints, the “Multiple EC” methods are better: for projections with two (or more) equational constraints, the number of polynomials is bounded by $O(m^4)$ for the original methods, $O(m^2)$ for the “Single EC” methods and $O(m)$ for the “multiple EC” methods.

Chapter 9

Further Work

There have been various improvements done to the implementations of CAD algorithms to improve the time and space complexity. In this thesis we provided enhancements to Lazard's projection operator and the Brown-McCallum projection operator, both of which are based on lex-least invariance. These enhancements are based on the exploitations of equational constraints present in the input formula. These enhancements resonate with the work done in [McC99] and [McC01], which are based on order invariance.

In Chapter 5 we discuss how to detect curtains during the lifting process of producing a CAD. This information is vital when taking advantage of equational constraints. If a curtain is detected, rather than producing an error we can run a subroutine that decomposes the curtain of the equational constraint. Below we describe some potential extensions of the work presented in this thesis.

9.1 Valuations

One usually constructs CADs that are required to be invariant for some property, meaning that there is some "valuation" function ν on $\mathbb{R}[x_1, \ldots, x_n] \times X$ (where X is a set with suitable total ordering) and if f is a constraint of the initial problem and σ is a CAD cell then $\nu(f, \alpha)$ should be constant for $\alpha \in \sigma$. For Lazard, ν is the lex-least valuation: other choices used include sign [Col75] and order [McC84]. Observe that the lex-least valuation takes account of the variable ordering but, sign and order do not.

One could ask what properties such a ν needs to have, and whether there are other possible choices of ν that might be useful. For example, one could consider replacing lexicographic by some other monomial order, or taking $\nu(f, \alpha)$ to be the Newton polygon of f near α. In any case ν would have to be able to detect when f vanishes, and it would need to satisfy some kind of semi-continuity. The property of upper-semicontinuity guarantees that if two polynomials are valuation invariant in a set, then their product is valuation invariant in the same set and vice-versa, which helps us take advantage of equational constraints.

9.2 Curtain Detection

In Definition 43, we have defined curtains as subvarieties of $\mathbb{R}^n$ whose base is a subset of $\mathbb{R}^{n-1}$. In practice, curtain behaviour could be detected earlier in the lifting process, if we have some $\alpha \in \mathbb{R}^i$ with $i < n-1$ and $f(\alpha, x_{i+1}, \ldots, x_n) = 0$. Then f has a curtain on the set $\{(\alpha, \beta) \mid \beta \in \mathbb{R}^{n-i-1}\}$. Trivially this would be a non-point curtain of f. But it remains to combine this with CAD algorithms and implement something similar to Algorithm 8.

9.3 Implementations

As mentioned in section 1.3, some implementation of the Lazard-based algorithms has been done in [Ton21]. However, an implementation of the modified Brown-McCallum algorithms for equational constraints is yet to be done. Such an implementation would work with curtains.

Benchmarking these to the implementations of McCallum's work to see the real life improvements and verifying the decrease in complexity of the algorithms, as computed in Chapter 8, has also to be done.

One could seek a better approach to detecting curtains than the probabilistic approach proposed in [BM20]. This would allow one to select equational constraints with atmost point curtains so as to avoid any problems in the liftng phase.

We have yet to investigate the problems caused by curtains in the case of multiple equational constraints, or to devise a subroutine that can be called when a curtain is detected without affecting the time complexity significantly.

Bibliography

[Abh90] S.S. Abhyankar. Algebraic Geometry for Scientists and Engineers. *Mathematical Surveys and Monographs*, 1990.

[BCR98] J. Bochnak, M. Coste, and M.-F. Roy. *Real algebraic geometry.* Springer, 1998. URL: `https://www.springer.com/us/book/9783540646631`.

[BDE+16] R.J. Bradford, J.H. Davenport, M. England, S. McCallum, and D.J. Wilson. Truth table invariant cylindrical algebraic decomposition. *J. Symbolic Computation*, 76:1–35, 2016.

[BGV13] S. Basu, A. Gabrielov, and N. Vorobjov. Semi-monotone sets. *J. Eur. Math. Soc.*, 15:635–657, 2013.

[BM20] Christopher W. Brown and Scott McCallum. Enhancements to Lazard's Method for Cylindrical Algebraic Decomposition. In François Boulier, Matthew England, Timur M. Sadykov, and Evgenii V. Vorozhtsov, editors, *Computer Algebra in Scientific Computing*, pages 129–149, Cham, 2020. Springer International Publishing.

[Bro03] C.W. Brown. QEPCAD B: a program for computing with semi-algebraic sets using CADs. *ACM SIGSAM Bulletin 4*, 37:97–108, 2003.

[Col71] G.E. Collins. The Calculation of Multivariate Polynomial Resultants. *J. ACM*, 18:515–532, 1971.

[Col75] G.E. Collins. Quantifier Elimination for Real Closed Fields by Cylindrical Algebraic Decomposition. In *Proceedings 2nd. GI Conference Automata Theory & Formal Languages*, pages 134–183, 1975.

[Col98] G.E. Collins. Quantifier elimination by cylindrical algebraic decomposition — twenty years of progess. In B.F. Caviness and J.R. Johnson, editors, *Quantifier Elimination and Cylindrical Algebraic Decomposition*, pages 8–23. Springer Verlag, Wien, 1998.

[Dav21] J.H. Davenport. *Computer Algebra*. In preparation: `http://staff.bath.ac.uk/masjhd/JHD-CA.pdf`, 2021.

[EBD19] Matthew England, Russell Bradford, and James Davenport. Cylindrical algebraic decomposition with equational constraints. *Journal of Symbolic Computation*, 07 2019. `doi:10.1016/j.jsc.2019.07.019`.

[GKZ94] I.M. Gel'fand, M.M. Kapranov, and A.V. Zelevinsky. Discriminants, resultants and multidimensional determinants. *Birkhaüser*, 1994.

[Kar84] N.K. Karmarkar. A New Polynomial-Time Algorithm for Linear Programming. *Combinatorica*, 4:373–395, 1984.

[Laz94] D. Lazard. An Improved Projection Operator for Cylindrical Algebraic Decomposition. In C.L. Bajaj, editor, *Proceedings Algebraic Geometry and its Applications: Collections of Papers from Shreeram S. Abhyankar's 60th Birthday Conference*, pages 467–476, 1994.

[McC84] S. McCallum. *An Improved Projection Operation for Cylindrical Algebraic Decomposition*. PhD thesis, University of Wisconsin-Madison Computer Science, 1984.

[McC85] S. McCallum. An Improved Projection Operation for Cylindrical Algebraic Decomposition. Technical Report 548 Computer Science University Wisconsin at Madison, 1985.

[McC99] S. McCallum. On Projection in CAD-Based Quantifier Elimination with Equational Constraints. In S. Dooley, editor, *Proceedings ISSAC '99*, pages 145–149, 1999.

[McC01] S. McCallum. On Propagation of Equational Constraints in CAD-Based Quantifier Elimination. In B. Mourrain, editor, *Proceedings ISSAC 2001*, pages 223–230, 2001.

[McC19] S. McCallum. Error in [McC01]. *E-mail 2019 January 5th*, 2019.

[MPP19] S. McCallum, A. Parusiński, and L. Paunescu. Validity proof of Lazard's method for CAD construction. *J. Symbolic Comp.*, 92:52–69, 2019.