Swansea University
Prifysgol Abertawe

Doctoral Thesis

ALGEBRAIC METHODS IN FEEDBACK CONTROL AND SPLINES WITH BOUNDARY CONDITIONS

---

by:

SAMUEL GUE

A thesis submitted in fulfilment of the requirements for the Doctor of Philosophy

of the

Department of Mathematics,
Faculty of Science and Engineering,
Swansea University

Supervised by Professor Nelly Villamizar

January 25, 2026

Declaration of Authorship

I, Samuel Gue, declare that this thesis titled, "Algebraic Methods in Feedback Control and Splines with Boundary Conditions" and the work presented in it are my own. I confirm that:

- This work has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.
- This thesis is the result of my own investigations, except where otherwise stated. Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.
- I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.
- The University's ethical procedures have been followed and, where appropriate, that ethical approval has been granted.

Signed: Samuel Gue

Date: January 25, 2026

"Don't get somewhere as fast as possible. Get somewhere as good as possible."

Eddie Izzard, 2017

Abstract

This thesis applies methods from algebraic geometry and topology to two distinct problems: one in optimal control and one in the theory of spline functions.

On the optimal control side, we use algebraic tools to develop a computational method for the synthesis of time-optimal feedback control laws for nilpotent systems. In particular, we study the polynomial systems derived from nilpotent linear systems, and use Newton's method and the Hermite quadratic form to solve them. We create a synthetic dataset with the solutions to these equations which we use to train a binary classifier neural network to solve nilpotent systems. To demonstrate the applicability of this tool, we solve chain of integrator systems of increasing dimension, focusing on the robustness of the method in the presence of perturbations.

On the splines side, we derive a formula for the dimensions of vector spaces of splines with boundary conditions over simplicial complexes embedded in $\mathbb{R}^2$ for high enough polynomial degree. We use tools from algebraic topology to reframe some classic results from spline theory to account for the boundary conditions. We demonstrate the use of the formula by finding the dimensions of vector spaces of splines with boundary conditions over various example simplicial complexes.

Acknowledgements

Firstly, I would like to dedicate this thesis to my partner Anushka, whose support has never once wavered, who has encouraged me at every stage to be strong, and without whom I would not have been able to complete this PhD.

I also thank my parents Robert and Elaine for their constant encouragement and sage advice throughout the entire process.

I want to thank my brother David for never letting me lose sight of what I believe in.

I want to thank my supervisor Prof. Nelly Villamizar for her invaluable advice and sharing her interest in splines with me. Also, Prof. Tomasz Brzezinski for chairing my Viva, Dr. Edwin Beggs for being my internal examiner, and Dr. Gabriel Barrenechea for being my external examiner. I would like to thank everyone for their comments and corrections.

Finally, I would like to thank everyone I have met at Swansea University with whom I have become good friends. The CoFo crew have made this whole experience fantastic from start to finish.

Contents

Abstract vi

Acknowledgements viii

1 Introduction 1

1.1 Layout of the thesis 5

2 Preliminaries 7

2.1 Linear Time-Optimal Control Problems 7

2.1.1 Nilpotent Systems 14

2.1.2 Newton's Method 18

2.1.3 Multivariate Newton's Method 19

2.1.4 Deflation 20

2.2 Root Counting 22

2.2.1 Polynomial Algebra 22

2.2.2 Gröbner Bases 24

2.3 The Hermite Quadratic Form 26

3 Time-Optimal Neural Feedback Control 31

3.1 Introduction 31

3.2 Time-Optimal Control of Nilpotent Systems 34

3.3 A deflated Newton's Method 37

3.4 The Hermite Quadratic Form 39

3.5 Construction of an open-loop solver and an algebraic black-box 43
3.6 Constructing a neural feedback law as a binary classifier 46
3.6.1 Feedforward Neural Network Classifier 46
3.6.2 Synthetic Data Generation 47
3.6.3 Training of the Model 48
3.7 Numerical Tests 49
3.7.1 The double integrator 49
3.7.2 Triple integrator 52
3.7.3 4th-order tests 53
3.7.4 5th-order integrator 56
3.8 Concluding remarks 57

4 Algebraic Splines with Boundary Conditions 59
4.1 Simplicial Complexes 60
4.2 Splines with Boundary Conditions on Simplicial Complexes 63
4.3 Chain Complexes and Simplicial Homology 75
4.3.1 Exact Sequences 76
4.4 The Dimension of the Space of Splines with Boundary Conditions . . 79
4.5 Examples 84
4.6 Subcomplexes and Splines with Partial Boundary Conditions 91
4.7 Concluding remarks 96

5 Conclusions and and Future Work 99

Appendices 100

A 101

Chapter 1

Introduction

This work represents an application of algebraic geometry to optimal control and approximation theory.

Optimal control theory is a culmination of myriad tools from across mathematics to form a theory for dynamical optimisation. It concerns the designing of optimal forcing (control) signals in order to modify the trajectory of a given dynamical system to pursue a desired goal. While the state of the art in optimal control is always expanding, some of the most influential early advancements are still in use today. An exposition to the theory of optimal control can be found in the book [65]. Contained is the maximum principle, an important result in mathematical optimal control which states necessary optimality conditions. The pioneering work of Feldbaum [39, 40] presented the result ubiquitously known as the Bang-Bang principle, which states that for linear optimal control functions in which the optimisation criterion is minimising the total time, the input function must take only extremal values in order to yield the optimal solution, and also that it can only change values a finite number of times that is bounded by the dimension of the system. The Bang-Bang principle reduces linear time-optimal control problems to polynomial systems. This transition heralded the use of polynomial elimination methods for time-optimal control, and created a link between optimal control and algebra.

Linear control systems are ubiquitous in control system designs, and arise in robotics [3], vibration control [35], and data-driven control [25], to name a few. The problem of determining the optimal actuator shape is a specific, and contemporary instance of optimal control. The design of actuators can be found in [36, 47].

As a consequence of the bang-bang theorem, many authors have been led to

observe a way to express the solutions to linear time-optimal control problems in terms of solutions to systems of polynomial equations. While not being the first to observe this, the process by which nilpotent linear time-optimal control problems could be reframed as questions about the roots of systems of polynomial equations has been made clear in [66]. Gröbner bases, a tool from computational algebraic geometry, have been used in [44, 61–63, 82] to compute optimal controls and to determine the structure of switching surfaces. A comprehensive review of the use of Gröbner bases in control theory can be found in [60]. The computation of Gröbner bases is known to be highly sensitive to the number of variables and the degrees of the input polynomials. An analysis of this degree of polynomials in a Gröbner basis in [34] shows that a basis for a set of polynomials of degree at most d in n variables is a set of polynomials of degree at most $2\left(\frac{d^2}{2}+d\right)^{2^{n-1}}$. This intrinsic complexity poses significant challenges for practical computations, particularly in systems with many variables or high-degree generators, and motivates the development of specialized algorithms that combine numerical and algebraic methods.

With a combination of numerical and algebraic techniques, one can easily construct a non-feedback control scheme, known as an open-loop control scheme, which determines the optimal trajectory given any initial condition. In real-world applications however, model uncertainties and disturbances will lead to deviations from the optimal trajectory, requiring a re-computation of the optimal action. Such a computationally intensive task limits the applicability of such a synthesis procedure for real-time control. These issues are well understood in the control literature, particularly in the context of stabilising control, and are typically mitigated by adopting a feedback control approach, which is inherently more robust. Time-optimal feedback controls are static maps that depend solely on the current state of the system and are evaluated as the system evolves. This adaptability allows them to correct deviations from the optimal trajectory, improving their robustness to uncertainties and disturbances.

Focusing on nilpotent linear time-optimal control problems, we set out to create a black-box for the computation of the optimal trajectory and implement this into a real-time feedback control. Our original motivation was to explore the scalability of the approach originally developed in [82], where Gröbner bases and real algebraic geometry techniques were proposed to solve polynomial systems arising in time-optimal control. In this respect, the best of our experience, purely algebraic techniques relying on Gröbner bases are either too difficult to scale to higher dimensions, or difficult

to implement in a real-time system due to their complexity. This makes them inapplicable for real-time feedback control in at least the cases we considered. Inspired by a need to overcome this computational complexity of purely algebraic feedback control schemes, we propose a hybrid approach to the feedback control of nilpotent linear time-optimal control problems. This is not to say, however, that pure algebraic methods for feedback control are not plausible or even applicable, as off-line precomputation may be used to offload expensive calculations so that algebraic methods can be effectively implemented in real-time. We propose an algebraic method using the algebraic tools used in the hybrid approach, an alternative to Gröbner bases in an aim to provide an algebraic method with no polynomial solving required.

Spline functions are piecewise polynomial functions defined on partitions of real domains, subject to smoothness properties on the borders between subregions. The first mathematical reference to these constructions as splines is due to Schoenberg in [69, 70]. The Bernstein-Bézier method, which has been extensively used in approximation theory to study spline functions on triangulations, is discussed in [74]; see also [72] and the references therein. Splines, especially those with boundary conditions, are fundamental for the finite element method, a method for numerically solving systems of differential equations by first subdividing the domain, and then approximating the solution curve (surface, or hypersurface depending on the dimension) by polynomials in each region, satisfying smoothness conditions on the boundaries between regions, and the boundary conditions of the differential equations. The development of the finite element method traces back to at least as early as the work of Courant [27], and since then it has been independently developed in the structural engineering community, and later reformulated mathematically [77]. In [75, 76], Strang conjectured that the dimension of the vector space of spline functions defined on a triangulation embedded in $\mathbb{R}^2$ with maximum polynomial degree d and continuous derivatives up to order $r = 1$, is determined by combinatorial data. While splines started out as a part of the numerical analysts toolbox, the work of Billera in [11] was foundational and opened brand new avenues for the study of splines using homological algebra. Billera proved Strang's conjecture true for generic triangulations in [11] using these homological techniques. Such an approach was developed further in [67, 68], and its use as a method for determining the dimensions of vector spaces of splines is widely represented in much of the modern research of splines, being the primary algebraic tool, known primarily as the Billera-Schenck-Stillman chain complex.

The algebraic treatment of splines with boundary conditions by McDonald in

[57] showed a link between the Hilbert polynomials of the module of splines with boundary conditions and the module of splines without boundary conditions. There is a chain complex that appears in the study of splines with boundary conditions that is analogous to the Billera-Schenck-Stillman chain complex, and has much the same properties, and it is these properties that allow for the study of the dimensions of spaces of splines with boundary conditions.

Recently, in [55], the authors developed an algebraic framework for studying spline functions defined on collections of patches instead of fixed embeddings of polyhedral complexes. These splines, known as geometrically continuous splines, received considerable attention for their uses in solving partial differential equations using isogeometric analysis, see [24] and the references therein.

We set out to find a formula for the dimension of spaces of splines with boundary conditions. In the process, we explored the algebraic and topological techniques levelled on splines in previous works, and explored how they could be modified, or whether they could naturally extend to provide similar or extended results that dealt with splines with boundary conditions. We found that the coning construction of Billera in [11], as well as the Billera-Schenck-Stillman chain complex [11, 67, 68] could naturally be extended to prove results about splines with boundary conditions analogously to how they prove results about splines without boundary conditions. Our original inspiration to consider splines of this nature was to answer the question of whether there was a way to determine for a given smoothness, the degrees for which there existed a spline with boundary conditions that was equal to the same polynomial everywhere on the triangulation.

Set against our aims, the contributions of this thesis are as follows. Proposition 3.2.1, a formula for the polynomials that must be satisfied by the times where an optimal solution to a nilpotent linear time-optimal control problem changes value. This makes explicit the polynomial system, and allows for methods to be more simply implemented in higher dimensions as we no longer rely on symbolic integration to determine the polynomial system. We design a neural feedback law for the closed-loop control of nilpotent linear time-optimal control functions which we describe in Section 3.6. This neural network is the black-box that we set out to find. We compare its accuracy to open-loop solving, showing that without perturbation the feedback law yields near time-optimal control, and in the presence of perturbation yields a near-time optimal solution when the open-loop solver does not.

In Chapter 4, we reformulate a number of theorems from algebraic spline geometry

to account for boundary conditions, namely Propositions 4.2.1, 4.2.2, 4.2.3 and 4.3.3. We then use these and other results to state and prove Theorem 4.4.4, a formula for the dimension of the space of splines with boundary conditions of degree at most d. Specifically, this formula gives the dimension of the space of r-smooth splines with boundary conditions with degree at most a given d for all d, and we then prove that elements of this formula vanish for d sufficiently large, leaving us with a calculable formula that matches the dimension for d sufficiently large. We compare the value for the dimension predicted by this calculable formula and the actual dimension, and state for what values of d they are equal in a number of examples. We then define the space of splines with boundary conditions on a subset of the boundary edges. We then prove Proposition 4.6.1 which provides a lower bound on the dimension of spline spaces on planar triangulations in terms of the dimension of spline subspaces.

1.1 Layout of the thesis

The rest of the thesis is laid out as follows:

Chapter 2: This chapter is a collection of preliminary results including an exposition of the basic results about nilpotent time-optimal control problems. We introduce Newton's method, a numerical solving method for finding the roots of polynomial equations. We modify it with the deflation process, which allows us to use it to find all possible roots of a polynomial system. Inspired by the need to improve the efficiency and accuracy of the deflated Newton's method, we introduce the Hermite quadratic form.

Chapter 3: The material in this chapter is based on [9], which is an article developed in collaboration with Sara Bicego, Dante Kalise, and Nelly Villamizar. We transform nilpotent time-optimal control problems into polynomial systems, in the process proving Proposition 3.2.1, and solve them using Newton's method and the Hermite quadratic form. We use the solutions to generate a synthetic data set on which we train a binary classifier via supervised learning that can solve nilpotent time-optimal control problems in real-time. We then assess the performance and robustness of the proposed approach on chains of integrators up to dimension 5, comparing the efficacy and efficiency of the method to open-loop solutions. We conclude the section with some remarks and future research directions to signal the end

of the first part of the thesis. In this thesis we expand beyond the scope of the paper [9], we suggest an approach to purely algebraic feedback control based on properties of the Hermite quadratic form of an ideal that warrants further research.

Chapter 4: This chapter features an application of algebraic geometry and topology to determine the dimension vector spaces of splines with boundary conditions defined over simplicial complexes embedded in $\mathbb{R}^2$. We modify some important results from the theory of splines to account for the imposition of boundary conditions, namely Propositions 4.2.1, 4.2.2, 4.2.3 and 4.3.3. We derive and prove Theorem 4.4.4, a formula for the dimension of vector spaces of splines with boundary conditions for high enough polynomial degree, and demonstrate it in action with a number of examples to show the applicability and highlight some interesting special cases. After defining splines on simplicial complexes satisfying partial boundary conditions, we construct an embedding of the direct sum of the spaces of splines over two complexes into the space of splines of the complex formed by gluing them along a set of common edges in Proposition 4.6.1.

Chapter 2

Preliminaries

2.1 Linear Time-Optimal Control Problems

Linear time-optimal control problems take the form of a minimisation problem with constraints given by a system of coupled differential equations. This section will serve as an introduction to linear time-optimal control problems in theory, as well as an exposition of results such as the maximum principle, the bang-bang theorem, and existence and uniqueness theorems that will be important in Chapter 3. For a more detailed introduction to optimal control theory, we refer the reader to [1]. Consider real numbers a_{ij} for $1 \leq i, j \leq n$. Linear systems of coupled differential equations, such as

$$\begin{aligned} \dot{x}_1(t) &= a_{11}x_1(t) + a_{12}x_2(t) + \cdots + a_{1n}x_n(t), \\ \dot{x}_2(t) &= a_{21}x_1(t) + a_{22}x_2(t) + \cdots + a_{2n}x_n(t), \\ &\vdots \\ \dot{x}_n(t) &= a_{n1}x_1(t) + a_{n2}x_2(t) + \cdots + a_{nn}x_n(t), \end{aligned} \tag{2.1.1}$$

for differentiable real functions x_i in t, are ubiquitous in control theory, since the dynamics of many real world systems are subject to coupled differential equations. Physical systems operating under Newtonian mechanics are simple canonical examples such as rocket acceleration or damped harmonic oscillations in one dimension.

By letting $\boldsymbol{x}(t) = (x_1(t), \ldots, x_n(t))^\top$, we can write (2.1.1) as a matrix equation

$$\dot{\boldsymbol{x}}(t) = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} \boldsymbol{x}(t). \tag{2.1.2}$$

We write A for the matrix in (2.1.2), and thus have the equation $\dot{\boldsymbol{x}}(t) = A\boldsymbol{x}(t)$. Solving a linear system of coupled differential equations is computationally fairly simple using linear algebra based methods. The system is homogeneous as the rate of change of the vector $\boldsymbol{x}$ depends only on the current value of $\boldsymbol{x}$, not the time t. A linear control problem is a modification of this system, by adding an inhomogeneous function of t. We do not specify the exact function, but will allow it to be any function within a certain set of what is referred to as admissible functions. By adding this unknown function, we turn what is a dynamical system which, given an initial condition $\boldsymbol{x}(0) = \boldsymbol{x}_0$, the trajectory $\boldsymbol{x}(t)$ is predetermined, into a dynamical system where the trajectory is a path in the phase space (here $\mathbb{R}^n$) that is determined by the choice of unknown function. In such a sense, we can say that this unknown function controls the trajectory, and is thusly referred to as a control function. Traditionally, in the case of linear optimal control this control function is in the form of a vector function, often written as vector of linear combinations of real-valued functions in terms of t, and the number of real-valued functions depends on the nature of the system being modelled, and which parts of the system are being controlled. Instead of having arbitrarily many real-valued functions constituting the control term such as in [65], we limit our case to the use of a single real-valued control function, giving a control term that is given exclusively by the output of a single function multiplied by a vector in $\mathbb{R}^n$. The control term can be thought of, in this case, as a sort of "push" to the system in the physical sense. The direction of the trajectory $\boldsymbol{x}(t)$ is given by $\dot{\boldsymbol{x}}(t)$, and adding a non-zero vector to (2.1.2) at a given time amounts to pushing the trajectory in the direction of the vector. Consequently, the choice of the vector that we add can affect the trajectory more than others.

The control is, almost ubiquitously, referred to with the symbol $u(t)$, and introducing the control to (2.1.2) by multiplying it with a vector $\boldsymbol{b}$ in $\mathbb{R}^n$ gives us the system

$$\dot{\boldsymbol{x}}(t) = A\boldsymbol{x}(t) + \boldsymbol{b}u(t).$$

Evidently, if we decide to set $u(t) = 0$, then we recover the original dynamical system, so if no control is added, the trajectories remain the same as expected. The question

that we should ask, as it seems, is that since we can use the control $u(t)$ to affect the trajectories of the dynamical system, is there a way to affect the trajectories in such a way that a specific (or any) desired output is achieved? Formally, we can say that if we have an initial condition $\boldsymbol{x}_0$, is there a control function $u(t)$ for which the trajectory $\boldsymbol{x}(t)$ with $\boldsymbol{x}(0) = \boldsymbol{x}_0$ eventually intercepts a target point say $\boldsymbol{x}_F$ (F for final)? Another question then would be, if the previous question can be answered in the affirmative, then is there a control function $u(t)$ that gives a trajectory of $\boldsymbol{x}_0$ that intercepts with $\boldsymbol{x}_F$ in the shortest possible time? We will see from Proposition 2.1.2, that both of these questions can be answered in the affirmative, assuming some conditions on A and $\boldsymbol{b}$, as well as the possible values that a control function $u(t)$ can take. Like with many other authors, such as [61] but also in many other papers, we will be choosing hereafter our target to be the origin $\boldsymbol{x}_F = \mathbf{0}$ for simplicity.

In control theory in general, the control functions are typically not allowed to be unbounded in their values. For this reason, the values that a control function can take is typically restricted to a convex, closed, and bounded region of $\mathbb{R}$, which corresponds to a closed and bounded interval. Convex refers to not having disjoint subsets of $\mathbb{R}$ being part of the same set of acceptable values. Closed ensures that the endpoints of the interval are included. Not including the endpoints can also lead to the non-existence of optimal controls. Let $\mathbb{U} \subseteq \mathbb{R}$ be such a closed bounded interval in $\mathbb{R}$. We want control functions to only take values within $\mathbb{U}$ for the aforementioned reasons, so we define the set of *admissible controls*:

$$\mathcal{U} = L^{\infty}\left([0, +\infty); \mathbb{U}\right),$$

the set of essentially bounded functions on $[0, +\infty)$ taking values in $\mathbb{U}$. We call a control u is admissible if it is in the set $\mathcal{U}$. The notation L^{∞} is a special case of the definition of L^{∞} presented in [15, Section 4.2], but this book is certainly not the first instance of this construction.

The question of finding an admissible control that produces a trajectory starting at the initial condition, and intercepting with the target point in the shortest possible time is, mathematically, the process of finding an admissible control that solves the following minimisation problem:

$$\min_{u \in \mathcal{U}} T \geq 0 \text{ subject to } \begin{cases} \dot{\boldsymbol{x}}(t) = A\boldsymbol{x}(t) + \boldsymbol{b}u(t), \\ \boldsymbol{x}(0) = \boldsymbol{x}_0, \\ \boldsymbol{x}(T) = \mathbf{0}, \end{cases} \tag{2.1.3}$$

where A is an $n \times n$ real matrix, $\boldsymbol{b} \in \mathbb{R}^n$, and $\boldsymbol{x}_0 \in \mathbb{R}^n$ a given initial condition. To determine such a control, the goal is to find some conditions that an optimal control must have, and use these to determine first whether an optimal solution exists, as well as then to restrict the search space to allow the optimal control to be found.

We now give a primer on the properties that an optimal control must have. We start with a fundamental result in the field of optimal control known as the maximum principle. In the following we recall some results from [65] that we will use in Chapter 3.

Proposition 2.1.1 ([65, Theorem 1])**.** *Consider the control problem (2.1.3). Let $u^*(t)$ be a time-optimal control function that solves it. Let $\boldsymbol{x}^*(t)$ be the corresponding trajectory, and T^* be the value for which $\boldsymbol{x}^*(T^*) = \mathbf{0}$. For any admissible control $u(t)$ with corresponding trajectory $\boldsymbol{x}(t)$ and time T, define the following function, referred to as the Hamiltonian,*

$$h_u(\boldsymbol{p}, \boldsymbol{x}(t)) = \boldsymbol{p} \cdot (A\boldsymbol{x} + \boldsymbol{b}u).$$

Then, there exists a real continuous vector-function

$$\boldsymbol{p}(t) \colon [0, T] \to \mathbb{R}^n \setminus \{\mathbf{0}\},$$

such that the following conditions hold for almost all $t \in [0, T]$:

$$\begin{aligned} \dot{\boldsymbol{p}}(t) &= -\frac{\partial h_{u^*}}{\partial \boldsymbol{x}}(\boldsymbol{p}(t), \boldsymbol{x}^*(t)), \\ h_{u^*(t)}(\boldsymbol{p}(t), \boldsymbol{x}^*(t)) &= \max_{u \in \mathcal{U}} h_u(\boldsymbol{p}(t), \boldsymbol{x}^*(t)), \\ h_{u^*(t)}(\boldsymbol{p}(t), \boldsymbol{x}^*(t)) &\geq 0. \end{aligned} \tag{2.1.4}$$

The formulation of the maximum principle that we present in Proposition 2.1.1 is a specialisation of the more general theorem to time-optimal controls. The statement "for almost all $t \in [0, T]$" in Proposition 2.1.1 is saying that the properties (2.1.4) hold true for all except a finite or countably infinite set of values of t in $[0, T]$. Most of these results from [65] proved using the maximum principle are under some slightly restricted assumptions about the matrix A and vector $\boldsymbol{b}$ known as Kalman's controllability criterion. It is a form of "genericity" condition on A, $\boldsymbol{b}$. We say that Kalman's controllability criterion is satisfied for the pair $(A, \boldsymbol{b})$ if

$$\boldsymbol{b}, A\boldsymbol{b}, \ldots, A^{n-1}\boldsymbol{b},$$

are linearly independent vectors in $\mathbb{R}^n$, that is, if there exist coefficients $c_0, \ldots, c_{n-1}$ so that

$$c_0\boldsymbol{b} + c_1 A\boldsymbol{b} + \cdots + c_{n-1}A^{n-1}\boldsymbol{b} = \mathbf{0},$$

then $c_0 = c_1 = \cdots = c_{n-1} = 0$. In [48], Kalman showed that this is necessary and sufficient for a solution to (2.1.3) to exist. Under this criterion, Pontryagin et. al. used the maximum principle to show a further implied condition on this solution.

Proposition 2.1.2 ([65, Theorem 9]). *If Kalman's controllability criterion holds for the pair $(A, \boldsymbol{b})$, then there is an optimal solution $u(t)$ to (2.1.3). In addition, $u(t)$ is piecewise constant, and it takes only extreme values in $\mathbb{U}$.*

This result is known as the bang-bang principle as stated in [52], and earlier in [18]. It is a result that implies that the only values of $\mathbb{U}$ which we need to consider are the boundary values. This is why we required $\mathbb{U}$ to be closed. An extension of this result gives a further restriction on the shape of optimal controls:

Proposition 2.1.3 ([65, Theorem 10]). *Suppose Kalman's controllability criterion holds for the pair $(A, \boldsymbol{b})$, and suppose all eigenvalues of A are real. Then, there is an optimal solution $u(t)$ to (2.1.3). In addition, $u(t)$ is piecewise constant, takes only extreme values in $\mathbb{U}$, and does not have more than $n-1$ switchings (i.e., not more that n intervals on which $u(t)$ is constant).*

In particular, Proposition 2.1.3 motivates a parametrisation of the total time $T = t_1 + \cdots + t_n$ for non-negative t_i's, which determine the switching times of the control signal. The time domain $\Omega = [0, T]$ is the union of subintervals $\Omega_i = [T_{i-1}, T_i)$, with $T_0 = 0$, and $T_i = \sum_{j=1}^{i} t_j$, for every $1 \leq i \leq n-1$, and $\Omega_n = [T_{n-1}, T_n]$. This induces a further parametrisation of the optimal control signal $u(t)$ as a piecewise-constant function in time. We may be concerned that, even if a control is optimal, that it is not unique, as Proposition 2.1.3 only posits that there is a bang-bang control that is optimal.

Proposition 2.1.4 ([65, Theorem 12]). *Let $u_1(t)$ and $u_2(t)$ be admissible solutions for (2.1.3) that are piecewise constant, taking only extremal values in $\mathbb{U}$, and do not have more than $n-1$ switchings each on the intervals $[0, T]$ and $[0, S]$ respectively for $T, S \geq 0$. Then, these controls coincide, in the sense that $T = S$, and $u_1(t) = u_2(t)$, almost everywhere on $[0, T]$.*

What this tells us is that not only is there an optimal bang-bang control if the hypotheses of Proposition 2.1.3 are satisfied, but also that it is unique up to a measure 0 set of points in $[0, T]$. As a result, the above parametrisation is uniquely determined by A, $\boldsymbol{b}$, and $\boldsymbol{x}_0$. Given the optimal solution to (2.1.3), we can determine the optimal trajectory $\boldsymbol{x}(t)$ from $\boldsymbol{x}_0$ to $\mathbf{0}$.

Proposition 2.1.5. *Consider the linear control problem (2.1.3). Let $u(t)$ be any admissible control defined on the interval $[0,T]$ such that the system of differential equations in (2.1.3) has a solution in $[0,T]$. The corresponding solution to the system of differential equations, $\boldsymbol{x}(t)$, is*

$$\boldsymbol{x}(t) = e^{tA}\boldsymbol{x}_0 + e^{tA}\int_0^t e^{-\tau A}\boldsymbol{b}u(\tau)d\tau. \tag{2.1.5}$$

Proof of Proposition 2.1.5. Using the integrating factor e^{-tA}, we modify the differential equation $\dot{\boldsymbol{x}} = A\boldsymbol{x} + \boldsymbol{b}u$ into

$$e^{-tA}\dot{\boldsymbol{x}} - e^{-tA}A\boldsymbol{x} = e^{-tA}\boldsymbol{b}u.$$

The matrix exponential

$$e^{-tA} = \sum_{r=0}^{\infty}\frac{(-t)^r A^r}{r!},$$

is a well-defined, absolutely convergent series, and A commutes with itself, so A and e^{-tA} commute, we see that

$$\frac{d}{dt}\left(e^{-tA}\boldsymbol{x}\right) = e^{-tA}\dot{\boldsymbol{x}} - Ae^{-tA}\boldsymbol{x} = e^{-tA}\boldsymbol{b}u.$$

Then, integrating from 0 to t gives

$$e^{-tA}\boldsymbol{x}(t) - \boldsymbol{x}(0) = \int_0^t e^{-\tau}\boldsymbol{b}u(\tau)d\tau,$$

and rearranging yields the result. □

The minimisation problem of (2.1.3) can be written as the system (2.1.5), the integral in this equation is dependent on u, a piecewise constant function. We want to find a non-integral formula for (2.1.5) that takes into account the terminal condition $\boldsymbol{x}(T) = \boldsymbol{0}$. Since an optimal control functions are known to be piecewise constant, we should expect their integrals to be piecewise polynomial. In the following example, we show how to use Propositions 2.1.3, 2.1.2, 2.1.4, and 2.1.5 to find an optimal control that solves a nilpotent control problem.

Example 2.1.1. Let $x(t)$ and $v(t)$ be the displacement and velocity respectively of a rocket propelled car moving in one dimension. Suppose we can control the acceleration of the rocket car with a control function $u(t)$, and suppose starting at a displacement of $-1m$ with an initial velocity of $-1ms^{-1}$, we wish to power the

rockets in such a way so as to steer the rocket car to $0m$ displacement with a velocity of $0ms^{-1}$ in time-optimal way. The mechanics of the situation are thus as follows:

$$\min_{u\in\mathcal{U}} T \text{ subject to } \begin{cases} \dot{x}(t) = v(t), \\ \dot{v}(t) = u(t), \\ (x(0), v(0)) = (-1, -1), \\ (x(T), v(T)) = (0, 0). \end{cases} \tag{2.1.6}$$

Here, we will limit acceleration to being bounded with $\mathbb{U} = [-1, 1]$, so the set of admissible controls is then $\mathcal{U} = L^\infty([0, +\infty); [-1, 1])$. In the notation of (2.1.3), we have

$$A = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \ \boldsymbol{b} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \ \boldsymbol{x}_0 = \begin{pmatrix} -1 \\ -1 \end{pmatrix}.$$

We can see that Kalman's controllability criterion holds here, since

$$\boldsymbol{b} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \ A\boldsymbol{b} = \begin{pmatrix} 1 \\ 0 \end{pmatrix},$$

are clearly linearly independent. This is a very common example for demonstrating the synthesis of an optimal control, for example it is represented in [65, Example 1], as well as many series of lecture notes [1, 37]. Here we will use the parametrisation of the total time T that is due to Proposition 2.1.3. Since $n = 2$ in this case, we can write $T = t_1 + t_2$, and $\Omega = [0, t_1 + t_2]$ as the union of subintervals $\Omega_1 = [0, t_1)$ and $\Omega_2 = [t_1, t_1 + t_2]$. An optimal control $u(t)$ that solves (2.1.6) is then a piecewise constant function that is constant on Ω_1 and Ω_2 taking values 1 or -1. Suppose $u(t) = 1$ in Ω_1, then solving the differential equations (2.1.6) using Proposition 2.1.5, and letting $t = T$ gives

$$\begin{aligned} \begin{pmatrix} 0 \\ 0 \end{pmatrix} &= \begin{pmatrix} 1 & T \\ 0 & 1 \end{pmatrix} \begin{pmatrix} -1 \\ -1 \end{pmatrix} + \begin{pmatrix} 1 & T \\ 0 & 1 \end{pmatrix} \int_0^T \begin{pmatrix} 1 & -\tau \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} u(\tau) d\tau \\ &= \begin{pmatrix} -1 - T \\ v_0 \end{pmatrix} + \begin{pmatrix} 1 & T \\ 0 & 1 \end{pmatrix} \int_0^T \begin{pmatrix} -\tau \\ 1 \end{pmatrix} u(t) d\tau. \end{aligned}$$

Since $u(t) = \begin{cases} 1 & t \in \Omega_1, \\ -1 & t \in \Omega_2, \end{cases}$ and recalling $T = t_1 + t_2$, we see that

$$v(T) = t_1 - t_2 - 1, \quad \text{and} \quad x(T) = \frac{t_1^2}{2} + t_1 t_2 - \frac{t_2^2}{2} - t_1 - t_2 - 1.$$

Then, solving these equations for t_1 and t_2 gives

$$t_1 = \pm\sqrt{\frac{3}{2}} + 1,\ t_2 = \pm\sqrt{\frac{3}{2}}.$$

Since t_1 and t_2 must be positive, and Kalman's controllability criterion holds and so a control function steering $\boldsymbol{x}_0$ to $\mathbf{0}$ exists by Proposition 2.1.2:

$$u(t) = \begin{cases} 1 & t \in \left[0, \sqrt{\frac{3}{2}} + 1\right), \\ -1 & t \in \left[\sqrt{\frac{3}{2}} + 1, 2\sqrt{\frac{3}{2}} + 1\right]. \end{cases}$$

The control is unique by Proposition 2.1.4, and the trajectory of the particle is then seen in Figure 2.1. ◇

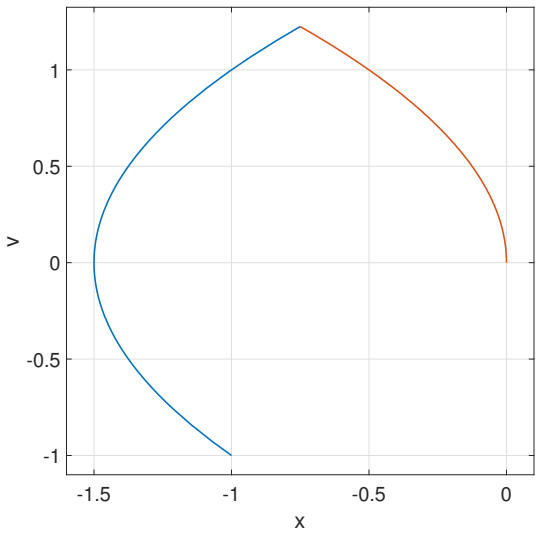

Figure 2.1: The trajectory of the a particle from Example 2.1.1 starting with initial displacement of $-1m$ and initial velocity $-1m/s$ steered to a displacement of $0m$ and velocity $0m/s$ with control function $u(t)$. The blue curve is the trajectory $x(t)$ while $u(t) = 1$, and the red curve is the trajectory of $x(t)$ while $u(t) = -1$.

Of course, we can choose any interval in general but for simplicity we herein assume $\mathbb{U} = [-1, 1]$, so that $u(0) = \pm 1$ in Ω_1, and $(-1)^{i-1}u(0)$ in the interval Ω_i.

2.1.1 Nilpotent Systems

We have thus far introduced linear time-optimal control problems in general. In this section, we focus our attention at a special class of linear time-optimal control

problems known as nilpotent control problems. Nilpotent control problems refer to the minimisation problem (2.1.3) under the assumption that the matrix A is nilpotent. We say a matrix A is nilpotent if there is some power of A, say k, for which $A^k = 0$. The smallest such positive number k is called the *index* of A, and is an integer that is at most n, the dimension of the matrix A. We will be considering the case where A has index of nilpotency n. This is the maximum possible index for a nilpotent matrix, and it is evident that it is necessary for Kalman's controllability criterion to hold for any pair $(A, \boldsymbol{b})$ in the nilpotent case, as if $A^{n-1} = 0$, then $\boldsymbol{b}, A\boldsymbol{b}, \ldots, A^{n-1}\boldsymbol{b} = 0$ would not be linearly independent. It can be seen from the following result that not only is this necessary (as we already know), but we can also determine additional required conditions on $\boldsymbol{b}$.

Lemma 2.1.6 ([80, Proposition 10.12])**.** *Let $\mathbf{0} \neq \boldsymbol{v} \in \mathbb{R}^n$, and let m be the least positive integer such that $A^m\boldsymbol{v} = 0$. Then, the vectors*

$$A^{m-1}\boldsymbol{v}, A^{m-2}\boldsymbol{v}, \ldots, A\boldsymbol{v}, \boldsymbol{v},$$

are linearly independent.

If A has index n, then the only way for $(A, \boldsymbol{b})$ to satisfy Kalman's controllability criterion is if $\boldsymbol{b}$ is not in $\ker(A^{n-1})$. Herein, we assume this is the case.

Since A is nilpotent, an important observation is that the only eigenvalue of A is 0, so any pair $(A, \boldsymbol{b})$ where A is nilpotent of index n and $\boldsymbol{b} \notin \ker(A^{n-1})$ satisfies the condition that A has only real eigenvalues, and so satisfies Kalman's controllability criterion by Proposition 2.1.6. We can, in fact, make a significant simplification when A is nilpotent as well. The following is a specialisation of a result of linear algebra, a proof of the more general result can be found in the cited text [80], but for clarity, we present a proof of the specific specialisation that we will use. This result will allow us to transition from arbitrary nilpotent matrices to those of a much simpler form known as Jordan canonical (or normal) form.

Proposition 2.1.7 ([80, Theorem 10.18])**.** *Let A be an $n \times n$ nilpotent matrix with index n. Then, there is an invertible matrix P such that*

$$A = P\mathfrak{J}P^{-1},$$

where $\mathfrak{J} = (\delta_{i,j-1})_{i,j}$ is the index n nilpotent $n \times n$ Jordan block.

Proof. Let $\boldsymbol{v} \in \mathbb{R}^n$ be a vector for which $A^{n-1}\boldsymbol{v} \neq 0$. Such a vector exists because $A^{n-1} \neq 0$. Then, consider the matrix

$$P = \left(\begin{array}{c|c|c|c|c} A^{n-1}\boldsymbol{v} & A^{n-2}\boldsymbol{v} & \cdots & A\boldsymbol{v} & \boldsymbol{v} \end{array} \right).$$

Then, since $A(A^{n-1}\boldsymbol{v}) = A^n\boldsymbol{b} = \mathbf{0}$:

$$AP = \begin{pmatrix} \mathbf{0} \mid A^{n-1}\boldsymbol{v} \mid \cdots \mid A^2\boldsymbol{v} \mid A\boldsymbol{v} \end{pmatrix} = P\mathfrak{J}.$$

We then need only to show that P is invertible. We can see this by Proposition 2.1.6 as n is the least integer such that $A^n\boldsymbol{v} = 0$, and so the columns of P are linearly independent, it is therefore invertible [80, Theorem 6.21]. □

We say that if such an invertible matrix P exists, then A is similar to $\mathfrak{J}$, and we see from this Proposition 2.1.7, that every nilpotent matrix A of index n is similar to $\mathfrak{J}$. With the following observation, we can then restrict A as well as $\boldsymbol{b}$.

Proposition 2.1.8. *Let A be an $n \times n$ nilpotent matrix with index n, and $\boldsymbol{b} \in \mathbb{R}^n$. Let P be an invertible matrix such that $A = P\mathfrak{J}P^{-1}$ where $\mathfrak{J}$ is the $n \times n$ nilpotent Jordan block with index n (P exists by Proposition 2.1.7). If Kalman's controllability criterion holds for the pair $(A, \boldsymbol{b})$, then Kalman's controllability criterion holds for the pair $(\mathfrak{J}, P^{-1}\boldsymbol{b})$.*

Proof. Kalman's controllability criterion holds for $(A, \boldsymbol{b})$. So the $A^i\boldsymbol{b}$'s for $0 \leq i \leq n-1$ are linearly independent. Since P is invertible, multiplying each of these vectors on the left by P^{-1} gives another set of linearly independent vectors, so

$$P^{-1}A^i\boldsymbol{b} = P^{-1}P\mathfrak{J}^iP^{-1}\boldsymbol{b} = \mathfrak{J}^i(P^{-1}\boldsymbol{b}),$$

for $0 \leq i \leq n-1$ are linearly independent, so Kalman's controllability criterion holds for $(\mathfrak{J}, P^{-1}\boldsymbol{b})$ as clearly $\mathfrak{J}$ has all real eigenvalues (only 0). □

This result shows that, without loss of generality, we may assume that $A = \mathfrak{J}$. Then, due to [20, Theorem 1] we must have $b_n \neq 0$ for Kalman's controllability criterion to hold, where b_n is the nth coordinate of $\boldsymbol{b}$. This is not too hard to see as well by inspection in our case. Kalman's criterion here ensures that the bang-bang theorem applies, giving us the parametrisation of T into switching times $t_1, \ldots, t_n$ as before. This parametrisation can then be used to in turn determine the switching times of the optimal control function. We will see in Chapter 3 the Proposition 3.2.1, which shows that if $\boldsymbol{x}_0 = (x_1, \ldots, x_n)^\top$ and $\boldsymbol{b} = (b_1, \ldots, b_n)^\top$ with $b_n \neq 0$, then the switching times satisfy the following polynomial equations (see equation (3.2.1) in Chapter 3)

$$0 = -u(0)\sum_{k=1}^{n-i+1} b_{k+i-1} \sum_{\substack{\alpha_1,\ldots,\alpha_n \geq 0 \\ \alpha_1+\cdots+\alpha_n = k}} (-1)^{\max\{j\colon \alpha_j \neq 0\}} \frac{t_1^{\alpha_1}\cdots t_n^{\alpha_n}}{\alpha_1!\cdots\alpha_n!} + \sum_{k=i}^{n} \frac{T^{k-i}}{(k-i)!}x_k.$$

As the switching times are non-negative, our aim is to find a non-negative solution of this system of equations, and by the existence and uniqueness results of Proposition 2.1.3 and Proposition 2.1.4, we find that any non-negative solution to this system of equations will give the switching times for the optimal control. Suppose there was some physical system modelled by the differential equations in (2.1.3), and we were aiming to actually steer $\boldsymbol{x}_0$ to the origin optimally. If we solved the system of equations and determine the optimal control, we could then implement this control in some way to the physical system and then we would obtain the desired trajectory. This would be referred to as an open-loop control scheme [59], as the output is determined only by the initial input. However, if at any time there is a perturbation, such as is common in real-world systems, if we keep using the same control function determined for this original initial condition, then the trajectory may no longer intercept the origin. In the presence of perturbations, open-loop control schemes do not have a mechanism for correction of errors. Additionally, we need to know precisely the switching times t_i in order to accurately steer the initial condition to the origin, and any imprecision in these values may lead to a failure of the trajectory to reach the origin. For this, we consider feedback control. The design of a feedback (or closed-loop) control scheme will begin not by finding the optimal control that steers the initial condition to the origin as normal, but by determining the initial value of the optimal control and letting the system evolve for a short amount of time under the assumption that the optimal control has the same value as its initial value over sufficiently short time scales. Then, when perturbations occur, or simply at regular intervals, the current position on the trajectory is taken to be a new initial condition, and then the initial value of the optimal control for that initial condition is computed, and the system then evolves according to this control function. In this way, even if perturbations occur, we will always be able to find our way to the origin.

In reality, we can only sample the position and optimal control at discrete time intervals $\Delta t > 0$ which we will choose to be orders of magnitude smaller than 1. If we discretise the system of differential equations in (2.1.3), then we can express it as a difference equation

$$\begin{cases} \frac{\boldsymbol{x}(t+\Delta t)-\boldsymbol{x}(t)}{\Delta t} = A\boldsymbol{x}(t) + \boldsymbol{b}u(t), \\ \boldsymbol{x}(0) = \boldsymbol{x}_0, \\ \boldsymbol{x}(T) = \boldsymbol{0}, \end{cases}$$

for some small time-step Δt. Rearranging the difference equation gives

$$\begin{cases} \boldsymbol{x}(t+\Delta t) = \boldsymbol{x}(t) + (A\boldsymbol{x}(t) + \boldsymbol{b}u(t))\,\Delta t, \\ \boldsymbol{x}(0) = \boldsymbol{x}_0, \\ \boldsymbol{x}(T) = \boldsymbol{0}. \end{cases}$$

By setting $t = 0$ in the difference equation, we see that

$$\boldsymbol{x}(\Delta t) = \boldsymbol{x}_0 + (A\boldsymbol{x}_0 + \boldsymbol{b}u(0))\,\Delta t.$$

From one point in time, this allows us to approximately determine where we will be after one time step, where the approximation gets better with smaller time-steps Δt. If perturbations occur, then the actual location of the point will be different from the expected location, and so we should modify the control function to compensate. By determining the initial value of the optimal control at every time-step, we eliminate the need to know what the switching times are, a problem that we noticed with open-loop controls, as we can detect switchings by a change in the initial value of the optimal control. This observation tells us that while we are searching for non-negative solutions to (3.2.1), we only care about their existence, and not the specific data of each solution.

Starting from $\boldsymbol{x}_0$, we then generate a sequence of points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ for some N, where $\boldsymbol{x}_N$ is sufficiently close to $\boldsymbol{0}$. Suppose we have a black-box that can, given an initial condition $\boldsymbol{x}_0$, determine the initial value of the optimal control function steering $\boldsymbol{x}_0$ to the origin. We refer to this unknown black-box as `BlackBox`. Then, we can express the real-time feedback control in terms of this black-box as Algorithm 1. The most notable method that has been implemented to determine the existence of a non-negative solution to the systems of equations found from nilpotent linear time-optimal control problems is Gröbner bases in [61–63, 82]. The methods we choose to perform this task of finding a non-negative solution are Newton's method and the Hermite's quadratic form, a numerical solving technique and a construction that can be used to design a root counting technique respectively.

2.1.2 Newton's Method

In order to determine the existence of a system of polynomial equations, we will use a method based on Newton's method. Newton's method is originally a method for finding sufficiently accurate approximations of the roots of univariate real valued

Algorithm 1: Feedback Control

Data: Dimension n, initial condition $\boldsymbol{x}_0 \in \mathbb{R}^n$, time-step Δt, convergence tolerance $\varepsilon \ll 1$

$i = 0$, `sequence` $\leftarrow \{\}$; `initialising`

`diff` $= \|\boldsymbol{x}_0\|$; `initial condition`

while $\boldsymbol{diff} > \varepsilon$ **do**

- $i+ = 1$;
- $u(t) \leftarrow$ `BlackBox`$(\boldsymbol{x}_{i-1})$; `determine optimal control`
- $\boldsymbol{x}_i \leftarrow \boldsymbol{x}_{i-1} + (A\boldsymbol{x}_{i-1} + \boldsymbol{b}u(0))\,\Delta t$; `compute next point`
- `sequence` $\leftarrow$ `sequence` $\cup \{\boldsymbol{x}_i\}$; `store next point`
- `diff` $= \|\boldsymbol{x}_i\|$;

Result: list `sequence` containing the points along the trajectory

functions. An exposition to Newton's method in the univariate case can be found in, for example, [79].

Definition 2.1.1 ([79, Definition 1.6])**.** Let $a_0 \in \mathbb{R}$ be a real number such that $f'(a_0) \neq 0$. The nth *Newton iterate* is the real number a_n defined recursively as

$$a_n = a_{n-1} - \frac{f(a_{n-1})}{f'(a_{n-1})},$$

for $n \geq 1$. The nth newton iterate is only defined, of course, if $f'(a_{n-1}) \neq 0$.

Depending on the choice of a_0, the Newton iterate will converge to a root of the equation $f(x) = 0$. This is a process that generalises readily to finding solutions of equations in multiple variables.

2.1.3 Multivariate Newton's Method

Our main aim is to find the roots of multivariate systems of polynomial equations. There is an analogue of Newton's method for multivariate functions which we expose here. Let $\mathcal{F}\colon \mathbb{R}^n \to \mathbb{R}^n$ be a multivariable function. Where Newton's method is used to determine the roots of univariate equations, the multivariate form can be used to find the roots of multivariate equations

$$\mathcal{F}(\boldsymbol{t}) = \boldsymbol{0}, \tag{2.1.7}$$

where $\boldsymbol{t} = (t_1, \ldots, t_n)$ is a tuple of n variables. A Newton's method for multivariate functions necessitates an analogue of both the derivative of a multivariate function,

and also the inverse of said derivative, as well as a meaningful way to multiply this inverse with the function. We begin with the analogue of the derivative.

Definition 2.1.2 ([79, Definition 4.3])**.** Let $\mathcal{F} = (f_1(\boldsymbol{t}), \ldots, f_n(\boldsymbol{t})) \colon \mathbb{R}^n \to \mathbb{R}^n$ be a multivariable function that is defined and continuous in some open set of $\mathbb{R}^n$ containing a point $\boldsymbol{a} \in \mathbb{R}^n$. Suppose further that the partial derivatives $\frac{\partial f_i}{\partial t_j}$ for $j = 1, \ldots n$ of f_i exist at $\boldsymbol{a}$ for $i = 1, \ldots, n$. The *Jacobian matrix* $\mathcal{F}'(\boldsymbol{a})$ of $\mathcal{F}$ at $\boldsymbol{a}$ is the $n \times n$ matrix with elements

$$\mathcal{F}'(\boldsymbol{a})_{i,j} = \frac{\partial f_i}{\partial t_j}(\boldsymbol{a}), \ i = 1, \ldots, n.$$

Given this choice of derivative analogue, it then becomes clear that in order to determine the inverse, we should simply take the inverse matrix $\mathcal{F}'(\boldsymbol{a})^{-1}$ assuming it exists. Then, we can easily see the multivariate analogue of Newton's method.

Definition 2.1.3. Let $\mathcal{F} \colon \mathbb{R}^n \to \mathbb{R}^n$ be a multivariate function, and let $\boldsymbol{a}_0 \in \mathbb{R}^n$ be a point with the property that $\mathcal{F}'(\boldsymbol{a})$ is invertible. Then, the kth *(multivariate) Newton iterate* $\boldsymbol{a}_k$ is defined as

$$\boldsymbol{a}_k = \boldsymbol{a}_{k-1} - \mathcal{F}'(\boldsymbol{a}_{n-1})^{-1}\mathcal{F}(\boldsymbol{a}_n),$$

assuming $\mathcal{F}'$ is invertible at each Newton iterate.

With this, [79, Theorem 4.4] gives sufficient conditions for the sequence of Newton iterates to converge (quadratically) to a solution of (2.1.7) for a multivariate function $\mathcal{F}$. When roots of higher multiplicity are present, then the convergence may be linear, or the sequence of Newton iterates may fail to converge. The sequence of Newton iterates may also not converge if the initial guess is not sufficiently close to any roots. To remedy these issues, we can randomly choose a new initial guess if the Newton iterates fail to converge within a certain maximum number of iterations, or to within a specified tolerance of a root. We choose these maximum number of iterations and tolerances on a problem-by-problem basis.

2.1.4 Deflation

Given a multivariate function $\mathcal{F} \colon \mathbb{R}^n \to \mathbb{R}^n$, we may use Newton's method to determine a real root of the equation (2.1.7). Given an initial point $\boldsymbol{t}_0$, suppose the sequence of Newton iterates converge to a root of $\mathcal{F}$. To find another root, we may choose to start from a different starting point. However, doing so does not guarantee that we will find a different root. To overcome the issue of having multiple

starting guesses converging to the same root, we will introduce here the concept of deflation. This is a process by which we can eliminate the roots of a multivariate equation (2.1.7) in such a way that we can then use Newton's method to find every real root. Deflation originally referred to a process on univariate polynomials, such as discussed in [84]. The idea, so it is described, is that if $f\colon \mathbb{R} \to \mathbb{R}$ is a univariate polynomial (therefore $f \in \mathbb{R}[t]$), and there is a root, say $r \in \mathbb{R}$, of $f(t) = 0$, then there is a polynomial $g \in \mathbb{R}[t]$ such that $f(t) = (t - r)g(t)$. This is the well known Factor Theorem. We can think of $g(t)$ as being the "quotient" $f(t)/(t-r)$, and it is referred to as a *deflation* of $f(t)$ by the root r. This can be extended to a definition for the deflation of any univariate function by a root r:

Definition 2.1.4. Let $f\colon \mathbb{R} \to \mathbb{R}$ be a univariate function. Let $r \in \mathbb{R}$ be so that $f(r) = 0$. Then, the function $g\colon \mathbb{R} \to \mathbb{R}$ defined by

$$g(t) = \lim_{t \to r} \frac{f(t)}{t - r}, \tag{2.1.8}$$

is the *deflation* of $f(t)$ by r.

In 1971, Brown and Gearhart [16] extended the concept of deflation to multivariate functions by replacing the quotient in (2.1.8) with a *deflation matrix*.

Definition 2.1.5 ([16])**.** Let $\boldsymbol{r} \in \mathbb{R}^n$, and let $M(\boldsymbol{t}, \boldsymbol{r})$ be an $n \times n$ matrix. We say that $M(\boldsymbol{t}, \boldsymbol{r})$ is a *deflation matrix* if for any differentiable $\mathcal{F}\colon \mathbb{R}^n \to \mathbb{R}^n$ such that $\mathcal{F}(\boldsymbol{r}) = 0$ and $\mathcal{F}'(\boldsymbol{r})$ is invertible, we have

$$\liminf_{i \to \infty} \|M(\boldsymbol{t}_i, \boldsymbol{r})\mathcal{F}(\boldsymbol{t}_i)\|_2 > 0,$$

for any sequence $\boldsymbol{t}_i \to \boldsymbol{r}$, where for $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_n) \in \mathbb{R}^n$, we have $\|\boldsymbol{\tau}\|_2 = \sqrt{\sum_{i=1}^n \tau_i^2}$.

Suppose we have a deflation matrix $M(\boldsymbol{t}, \boldsymbol{r})$ for every $\boldsymbol{r}$. Given a system of polynomial equations $\mathcal{F}$, we may use Newton's method to determine a root of $\mathcal{F}$, say $\boldsymbol{r}_1$. Then we modify the system of polynomial equations to be $M(\boldsymbol{t}, \boldsymbol{r})\mathcal{F}$, and perform Newton's method on this modified system to find a root $\boldsymbol{r}_2$. Repeating this, we can find every root of $\mathcal{F}$. However, a-priori there is no way of knowing when every root has been found. The stopping condition of this deflation method is determined by the number of roots it is expected to find. To this end, we are urged to find a way to count the number of roots of a polynomial system.

2.2 Root Counting

This section is dedicated to counting the roots of a system of polynomial equations. The construction that we are building towards is the Hermite quadratic form. We start with some an introduction to polynomial algebra, and then define Gröbner bases, a common tool from computation algebraic geometry. We give a full definition of the Hermite quadratic form here, as well as how to compute its signature.

2.2.1 Polynomial Algebra

We begin by introducing some results from the world of algebraic geometry. We start by introducing some basic notions of algebra, a more complete exposition can be found in [4, Chapter 1], and we will be using the definitions from this book, expanded slightly for clarity. We will be working over the fields $\mathbb{R}$ and $\mathbb{C}$, and will state clearly in which field we are operating when it is unclear from context. Consider the polynomial ring $\mathbb{R}[t_1, \ldots, t_n]$ in variables $t_1, \ldots, t_n$. A *system of polynomial equations* over $\mathbb{R}[t_1, \ldots, t_n]$ is a set of equations

$$\{f_1 = 0, f_2 = 0, \ldots, f_k = 0\},$$

where $f_1, \ldots, f_k \in \mathbb{R}[t_1, \ldots, t_n]$ are polynomials. A *solution* to this system of polynomial equations in $\mathbb{C}$ is a tuple $(z_1, \ldots, z_n) \in \mathbb{C}^n$ with the property that $f_i(z_1, \ldots, z_n) = 0$ for each $i = 1, \ldots, k$. The set of every solution to this system of polynomial equations, also known more commonly as the *variety*, is the set denoted

$$\mathbb{V}_{\mathbb{C}}(f_1, \ldots, f_k) = \{(z_1, \ldots, z_n) \in \mathbb{C}^n \mid f_i(z_1, \ldots, z_n) = 0 \text{ for } i = 1, \ldots, k\}.$$

We then define the set of real solutions of the system of equations to be the set denoted

$$\mathbb{V}_{\mathbb{R}}(f_1, \ldots, f_k) = \mathbb{V}_{\mathbb{C}}(f_1, \ldots, f_k) \cap \mathbb{R}^n.$$

For optimal control, we seek non-negative real roots corresponding to physically meaningful switching times, and so we are interested mainly in the set of non-negative real solutions of the system of equations $\mathbb{V}_{\mathbb{R}}(f_1, \ldots, f_n) \cap \mathbb{R}^n_{\geq 0}$. Recall an *ideal* of $\mathbb{R}[t_1, \ldots, t_n]$ is a subset I of $\mathbb{R}[t_1, \ldots, t_n]$ that is closed under addition and subtraction, and closed under multiplication by polynomials. We denote by $\langle f_1, \ldots, f_k \rangle$ the *ideal generated by* the f_i's, defined as

$$\langle f_1, \ldots, f_n \rangle = \left\{ \sum_{i=1}^{k} g_i f_i \mid g_i \in \mathbb{R}[t_1, \ldots, t_n] \right\}.$$

and say that the f_i's form a generating set of $\langle f_1, \ldots, f_n \rangle$.

Recall the *quotient* of a polynomial ring $\mathbb{R}[t_1, \ldots, t_n]$ by an ideal $I \subseteq \mathbb{R}[t_1, \ldots, t_n]$ is the set of classes

$$\mathbb{R}[t_1, \ldots, t_n]/I = \{f + I \mid f \in \mathbb{R}[t_1, \ldots, t_n]\},$$

where

$$f + I = \{f + g \mid g \in I\}.$$

The solutions of a polynomial system is the same as the solutions of the ideal they generate, so if I is the ideal defined by a polynomial system $\{f_i = 0 \mid i = 1, \ldots, n\}$, then we write $\mathbb{V}_{\mathbb{C}}(I) = \mathbb{V}_{\mathbb{C}}(f_1, \ldots, f_n)$ and $\mathbb{V}_{\mathbb{R}}(I) = \mathbb{V}_{\mathbb{R}}(f_1, \ldots, f_n)$. This allows us to switch from considering systems of polynomial equations over $\mathbb{R}[t_1, \ldots, t_n]$, to ideals in $\mathbb{R}[t_1, \ldots, t_n]$. Of course, a given system of polynomial equations may have finite, or infinitely many solutions. Systems of polynomial equations that have finitely many solutions are the types that we would like to consider, since we wish to use Newton's method to exhaust every real solution. An ideal corresponding to a system of polynomial equations is said to be *zero-dimensional* if the system of equations has finitely many solutions. Zero-dimensional ideals are critical because they guarantee finitely many solutions, enabling numerical methods like Newton's method. An important result is that if $I \subseteq \mathbb{R}$ is a zero-dimensional ideal, then in fact the quotient ring $\mathbb{R}[t_1, \ldots, t_n]/I$ is a finite dimensional $\mathbb{R}$-vector space, which is to say that there is a finite set of elements $p_1 + I, \ldots, p_k + I \in \mathbb{R}[t_1, \ldots, t_n]/I$ that are linearly independent and spanning.

The quotient ring $\mathbb{R}[t_1, \ldots, t_n]/I$ is the key to counting the roots of a polynomial system. The Eigenvalue theorem (so called by Cox in [28]) which can be found in [6, Theorem 4.96] is that if I is a zero-dimensional ideal, for a polynomial $q \in \mathbb{R}[t_1, \ldots, t_n]$ the eigenvalues of the linear map

$$\begin{aligned} \mathcal{L}_q \colon \mathbb{R}[t_1, \ldots, t_n]/I &\to \mathbb{R}[t_1, \ldots, t_n]/I \\ g + I &\mapsto qg + I, \end{aligned}$$

is $\{q(\boldsymbol{z}) \mid \boldsymbol{z} \in \mathbb{V}_{\mathbb{C}}(I)\}$. The Eigenvalue theorem immediately implies that the trace of $\mathcal{L}_1$ is equal to the number of complex roots of I. This result shows a clear link between the roots of a system of polynomial equations, and the quotient ring, specifically the multiplication maps.

Recall the problem of finding the switching times of an optimal control for the system (2.1.3). As we have shown, this involves finding non-negative solutions of the

system of polynomial equations (3.2.4). We can write the solution set of the system of polynomial equations as the variety of an ideal, and then the relevant solutions will be found in the intersection of the variety with the 1st orthant $\mathbb{R}^n_{\geq 0} = \{(x_1, \ldots, x_n) \in \mathbb{R}^n \mid x_i \geq 0 \forall i = 1, \ldots, n\}$. Suppose we are given an initial condition $\boldsymbol{x}_0 \in \mathbb{R}$ in (2.1.3), and suppose we choose a value for $u(0) = \pm 1$. We set $f_k(t_1, \ldots, t_n)$ to be the polynomial in $\mathbb{R}[t_1, \ldots, t_n]$ that is the kth row of the vector (3.2.4) when $u_0 = \pm 1$. We write $I = \langle f_1, \ldots, f_n \rangle$ to be the ideal defined by the system of polynomial equations (3.2.4), and then $\mathbb{V}_{\mathbb{C}}(I)$ and $\mathbb{V}_{\mathbb{R}}(I)$ to be the corresponding complex and real varieties. Since we care only for non-negative real solutions, we wish to find the set

$$\mathbb{V}_{\mathbb{R}}(I) \cap \mathbb{R}^n_{\geq 0}.$$

For this purpose we will introduce the Hermite quadratic form. To do so, we give a brief exposition of Gröbner bases.

2.2.2 Gröbner Bases

A Gröbner basis for an ideal I in a polynomial ring $\mathbb{R}[t_1, \ldots, t_n]$ is a generating set for I with some additional useful properties. Practically, finding a Gröbner basis of an ideal I mirrors the Gaussian elimination algorithm for linear polynomial equations. Gröbner bases were developed by Buchberger in his 1965 thesis [17] as a practical way to generate a basis for the quotient ring $\mathbb{R}[t_1, \ldots, t_n]/I$ when I is a zero-dimensional ideal of $\mathbb{R}[t_1, \ldots, t_n]$ (recall that $\mathbb{R}[t_1, \ldots, t_n]/I$ is a finite dimensional $\mathbb{R}$-vector space when I is zero-dimensional). A detailed introduction to theory of Gröbner bases can be found in [29]. Before we can define a Gröbner basis, we must choose before any computation a total ordering on the monomials in the polynomial ring $\mathbb{R}[t_1, \ldots, t_n]$, where by monomial we mean a polynomial in $\mathbb{R}[t_1, \ldots, t_n]$ that can be written as $t_1^{\alpha_1} \cdots t_n^{\alpha_n}$ where $\alpha_1, \ldots, \alpha_n \in \mathbb{Z}_{\geq 0}$.

Definition 2.2.1. We represent a monomial $t_1^{\alpha_1} \cdots t_n^{\alpha_n}$ using the n-tuple $(\alpha_1, \ldots, \alpha_n)$. A *monomial order* "$\succeq$" is a total order on the set of monomials satisfying the property that if $\alpha \succeq \beta$ for monomials α and β, then for any other monomial γ, we have $\alpha\gamma \succeq \beta\gamma$.

With this, we can speak about the concept of a "largest" monomial of a given selection $\alpha_1, \ldots, \alpha_m$ as the monomial α so that $\alpha = \alpha_i$ for some $i = 1, \ldots, m$ and $\alpha_i \succeq \alpha_j$ or $\alpha_i = \alpha_j$ for any $j = 1, \ldots, n$.

Definition 2.2.2. The *initial element* of a polynomial f is the monomial $\text{In}(f)$ in the monomial support of f that is largest with respect to $\succeq$. That is, it is the monomial α such that for any other monomial β in the monomial support of f (the set of monomials for which the polynomial f has a non-zero coefficient), we have $\beta \succeq \alpha$. Given an ideal I, the *initial elements* of I is the set $\text{In}(I) = \{\text{In}(f) \mid f \in I\}$, and the *initial ideal* is $\langle \text{In}(I) \rangle$.

With this, we can define Gröbner bases.

Definition 2.2.3. Let I be an ideal in $\mathbb{R}[t_1, \ldots, t_n]$ and $\succeq$ a monomial ordering. A *Gröbner basis* of I is a set of polyomials $G = \{g_1, \ldots, g_n\} \subseteq I$ with the property that G is a generating set for I, and $\text{In}(G) = \{\text{In}(g_1), \ldots, \text{In}(g_n)\}$ is a generating set for $\langle \text{In}(I) \rangle$.

A Gröbner basis computed for an ideal is not unique, and varies based on the choice of monomial ordering. Different choices of monomial orderings are useful in different contexts. A good exposition of the most common orderings can be found in [29, Page 8], but we focus on the graded reverse lexicographic ordering. We will refer to this ordering with the symbol $\succeq_{\text{GRevLex}}$. This ordering is defined as, for different monomals $\alpha = (\alpha_1, \ldots, \alpha_n)$ and $\beta = (\beta_1, \ldots, \beta_n)$ in $\mathbb{R}[t_1, \ldots, t_n]$, we say that $\alpha \succeq_{\text{GRevLex}} \beta$ if $\sum_{i=1}^n \alpha_i > \sum_{i=1}^n \beta_i$, or if $\sum_{i=1}^n \alpha_i = \sum_{i=1}^n \beta_i$ and the rightmost non-zero entry of $\alpha - \beta$ is negative.

Many computer algebra softwares that compute Gröbner bases such as `Macaulay2` [45] use this monomial order as standard. Another common monomial ordering is the lexicographic ordering, written $\succeq_{\text{lex}}$, defined so that if α and β are different monomials in $\mathbb{R}[t_1, \ldots, t_n]$, then $\alpha \succeq_{\text{lex}} \beta$ if and only if the leftmost non-zero entry in $\alpha - \beta$ is positive.

The reasons for the use of Gröbner bases as tools for solving polynomial equations, is that using lexicographic ordering, the Gröbner basis of an ideal is what is referred to in [51] as a shape basis. If $t_i \succ t_j$ whenever $i < j$, and I is zero-dimensional, then the shape basis consists of polynomials of the form $t_i - g_i(t_n)$, where g_i is a univariate polynomial in t_n for $i = 1, \ldots, n$. This then allows the solutions of the system of polynomial equations to be easily solved for. The reason for the use of Gröbner basis for root counting however, is that they can be used to provide a basis of the quotient ring $\mathbb{R}[t_1, \ldots, t_n]/I$. Recalling the Eigenvalue theorem now makes it clear the importance of Gröbner bases. Gröbner bases not only help solve systems of polynomial equations, but also construct the quotient ring basis [17] needed for

the Hermite quadratic form introduced in Section 2.3, and we present the method for finding this basis:

Proposition 2.2.1. *Let $\succeq$ be a monomial ordering on $\mathbb{R}[t_1, \ldots, t_n]$, and I a zero-dimensional ideal in $\mathbb{R}[t_1, \ldots, t_n]$. Let G be a Gröbner basis of I with respect to the ordering $\succeq$. Then the set of monic monomials that are not multiples of any $\mathrm{In}(g)$ for $g \in G$, restricted to $\mathbb{R}[t_1, \ldots, t_n]/I$ form a basis of $\mathbb{R}[t_1, \ldots, t_n]/I$.*

To demonstrate this method of using a Gröbner basis of an ideal to find a basis of the quotient, we will consider a small example in the ring $\mathbb{R}[t_1, t_2]$ using the zero-dimensional ideal

$$I = \left\langle t_1 - t_2 + 1, \frac{t_1^2}{2} + t_1 t_2 - \frac{t_2^2}{2} + 1 + t_1 + t_2 \right\rangle.$$

A Gröbner basis of I, computed using the graded reverse lexicographic order, is

$$G = \{t_1 - t_2 + 1, 2t_2^2 + 1\}.$$

The set of initial elements of G are t_1 and t_2^2, and so the monic monomials that are not multiples of either of these are 1 and t_2, which form a basis for $\mathbb{R}[t_1, t_2]/I$. Armed with a basis of $\mathbb{R}[t_1, \ldots, t_n]/I$ for a zero-dimensional ideal I, we now have everything we need to define the Hermite quadratic form of an ideal.

2.3 The Hermite Quadratic Form

The Hermite quadratic form is a quadratic form on the quotient ring $\mathbb{R}[t_1, \ldots, t_n]/I$. The applications of quadratic forms in root counting has been explored from as early as 1853 by Sylvester in [78]. To define the generalised Hermite quadratic form as presented in [6], we begin by choosing a polynomial $q \in \mathbb{R}[t_1, \ldots, t_n]$. Recall the Eigenvalue theorem stated that the roots of a polynomial system evaluated at a polynomial q could be found as the eigenvalues of the matrix $\mathcal{L}_q$. The Hermite quadratic form is a development on this concept [28] that will allow us to compute the number of real solutions as well as the number of complex solutions. Let $\mathcal{F}$ be a system of polynomial equations in $\mathbb{R}[t_1, \ldots, t_n]$ defining an ideal I that is zero-dimensional. Let $\mathcal{B} = \{\beta_1, \ldots, \beta_r\}$ be a basis of $\mathbb{R}[t_1, \ldots, t_n]/I$. Recall the linear map $\mathcal{L}_{\beta_i \beta_j q}(I)$ to be the map on $\mathbb{R}[t_1, \ldots, t_n]/I$ defined by

$$f + I \mapsto \beta_i \beta_j q f + I.$$

Definition 2.3.1. The *Hermite quadratic form* of (I, q) is the map on $\mathbb{R}[t_1, \ldots, t_n]/I$ defined by the matrix $(\mathcal{H}(I, q))_{ij} = \text{trace}(L_{ij}(I, q))$ where $L_{ij}(I, q)$ is the matrix of the linear map $\mathcal{L}_{\beta_i \beta_j q}(I)$.

To find the basis $\mathcal{B}$ of $\mathbb{R}[t_1, \ldots, t_n]/I$, we can use Gröbner bases as explained in section 2.2.2. Recall that the rank of the matrix $\mathcal{H}(I, q)$, denoted $\text{Rank}(\mathcal{H}(I, q))$, is equal to the number of non-zero eigenvalues of $\mathcal{H}(I, p)$ counted with multiplicity, and the *signature*, denoted $\text{Sign}(\mathcal{H}(I, q))$, is the difference between the number of its positive an negative eigenvalues. The following result establishes the link between the Hermite quadratic form $\mathcal{H}(I, q)$ and the number of roots of $\mathcal{F}$.

Proposition 2.3.1 ([6, Theorem 4.100])**.** *Suppose the system of polynomial equations $\mathcal{F}$ in $\mathbb{R}[t_1, \ldots, t_n]$ has finitely many solutions and $q \in \mathbb{R}[t_1, \ldots, t_n]$. Let I the zero-dimensional ideal defined by this system of equations, and $\mathcal{H}(I, q)$ be the Hermite quadratic form of (I, q). Then,*

$$\text{Rank}(\mathcal{H}(I, q)) = |\mathbb{V}_{\mathbb{C}}(I) \cap \{\boldsymbol{\tau} \in \mathbb{C} \mid q(\boldsymbol{\tau}) \neq 0\}|,$$
$$\text{Sign}(\mathcal{H}(I, q)) = |\mathbb{V}_{\mathbb{R}}(I) \cap \{\boldsymbol{\tau} \in \mathbb{R} \mid q(\boldsymbol{\tau}) > 0\}| - |\mathbb{V}_{\mathbb{R}}(I) \cap \{\boldsymbol{\tau} \in \mathbb{R} \mid q(\boldsymbol{\tau}) < 0\}|.$$

Immediately, we can see that the number of real roots of I can be determined from the signature of the Hermite quadratic form $\mathcal{H}(I, 1)$. If $q = 1$, then we abuse notation and write $\mathcal{H}(I) = \mathcal{H}(I, 1)$. Finding the rank and signature of $\mathcal{H}(I, q)$ involves counting the positive and negative eigenvalues of $\mathcal{H}(I)$. This consists of finding the roots of the characteristic polynomial

$$\det(\mathcal{H}(I, q) - \lambda \mathbb{I}) = 0, \tag{2.3.1}$$

a polynomial in $\mathbb{R}[\lambda]$, where $\mathbb{I}$ is the $r \times r$ identity matrix. In order to compute the signature of $\mathcal{H}(I, q)$, we would need to compute the number of positive roots and the number of negative roots of (2.3.1). There are many ways we could perform this, the most salient one considering the content so far would be by using a deflated Newton's method in one variable. While this will produce every real root, it is in many ways more work than is needed. There is, however, a simple method that merely involves observing the coefficients of (2.3.1). This method is implemented in the `RealRoots` [42, 43] package in `Macaulay2`, and we shall, for completeness, give an overview. The way that the signature is computed is using a result known as Descartes' rule of signs. As explained by De Gua in [31], this result was first stated but not proved by Descartes, and is related to the variations in the signs of the coefficients of a polynomial.

Definition 2.3.2. Let $f(\lambda) = a_0 + a_1\lambda + \cdots + a_{d-1}\lambda^{d-1} + a_d\lambda^d \in \mathbb{R}[\lambda]$ be a polynomial. Suppose none of $a_0, a_1, \ldots, a_{d-1}, a_d$ are zero. The *variations* of the coefficients of f is

$$\mathrm{Var}(f) = \mathrm{Var}(a_0, a_1, \ldots, a_{d-1}, a_d) = |\{i = 0, \ldots, d-1 \mid \mathrm{Sign}(a_i) \neq \mathrm{Sign}(a_{i+1})\}|.$$

If any coefficients are 0, then the variations of the coefficients of f is defined to be the variations of the same sequence with zeros removed.

Effectively, this is the number of times the sign flips when we read through the coefficients of f. We demonstrate its computation with a few simple examples.

Example 2.3.1. Consider the polynomial

$$f(\lambda) = 3 - \lambda + 4\lambda^2 + \lambda^3 - 5\lambda^4 + 9\lambda^5.$$

The variations of the sequence of coefficients of f is then

$$\mathrm{Var}(f) = \mathrm{Var}(3, -1, 4, 1, -5, 9) = |\{0, 1, 3, 4\}| = 4.$$

Consider then the polynomial

$$g(\lambda) = 1 + 6\lambda - \lambda^2 - 8\lambda^3 + 3\lambda^5.$$

The variations of the sequence of coefficients of g is then

$$\mathrm{Var}(g) = \mathrm{Var}(1, 6, -1, -8, 0, 3) = \mathrm{Var}(1, 6, -1, -8, 3) = |\{1, 3\}| = 2.$$

◇

The variations in the sequence of coefficients of a polynomial can be used to count the number of roots of a polynomial.

Proposition 2.3.2 ([31, Theorem VI])**.** *Let $f(\lambda) = a_0 + a_1\lambda + \cdots + a_{d-1}\lambda^{d-1} + a_d\lambda^d \in \mathbb{R}[\lambda]$ be polynomial with only real roots. Then*

$$\mathrm{Var}(a_0, a_1, \ldots, a_{d-1}, a_d),$$

is equal to the number of positive roots of $f(\lambda) = 0$ counted with multiplicity.

This result is far simpler to compute as opposed to finding all roots using a reduced Newton method. We demonstrate this by some examples

Example 2.3.2. Consider the polynomial

$$f(\lambda) = 2 - 7\lambda + \lambda^2 + 8\lambda^3 - 2\lambda^4.$$

The variations of the sequence of coefficients of f is then

$$\mathrm{Var}(f) = \mathrm{Var}(2, -7, 1, 8, -2) = |\{0, 1, 3\}| = 3.$$

That this polynomial only has real roots we will just take as given. Thus, by Proposition 2.3.2, this polynomial has 3 positive real roots and hence 1 negative real root. ◇

In order to be able to use the rule of signs to determine the signature of the Hermite quadratic form of (I, q), we must first be able to prove that the characteristic polynomial has exclusively real roots. In other words, we need $\mathcal{H}(I, q)$ to have only real eigenvalues.

Proposition 2.3.3. *Let A be a real symmetric matrix. Then A has exclusively real eigenvalues.*

Proof. See for example [80, Proposition 9.8]. □

By definition, the Hermite quadratic form $\mathcal{H}(I, q)$ is a real symmetric matrix, and so it only has real eigenvalues. We then compute the signature of $\mathcal{H}(I, q)$. Suppose

$$\chi(\lambda) = \det(\mathcal{H}(I, q) - \lambda\mathbb{I}),$$

is the characteristic polynomial of $\mathcal{H}(I, q)$. If 0 is an eigenvalue of $\mathcal{H}(I, q)$, then we write $\chi(\lambda) = \lambda^p \chi^*(\lambda)$ where $p > 0$ is an integer and $\chi^*(0) \neq 0$. Since $\mathcal{H}(I, q)$ has only real eigenvalues, the polynomial $\chi(\lambda)$, and also $\chi^*(\lambda)$ have only real roots. The rule of signs then states that there are as many positive roots as the number of variations in the sequence of coefficients of $\chi^*(\lambda)$. We set r_+ and r_- to denote the number of positive and negative real roots of χ^* with multiplicity. We have that $r_+ + p + r_- = \deg(\chi)$, and we know $r_+ = \mathrm{Var}(\chi^*)$ by Proposition 2.3.2. So, $r_- = \deg(\chi) - p - \mathrm{Var}(\chi^*)$, and $\deg(\chi) - p = \deg(\chi^*)$ so

$$\mathrm{Sign}(\mathcal{H}(I, q)) = 2\mathrm{Var}(\chi^*) - \deg(\chi^*).$$

As multiplying χ^* by λ^p does not change the sequence of coefficients (except for adding zeros to the start), it also does not change the variations, so $\mathrm{Var}(\chi^*) = \mathrm{Var}(\chi)$.

Since $\deg(\chi^*)$ is the total number of non-zero eigenvalues of the matrix $\mathcal{H}(I,q)$, we can see that $\deg(\chi^*) = \mathrm{Rank}(\mathcal{H}(I,q))$. This allows us to compute the signature and rank of the Hermite quadratic form $\mathcal{H}(I,q)$ in an efficient way:

$$\begin{aligned} \mathrm{Sign}(\mathcal{H}(I,q)) &= 2\mathrm{Var}(\chi) - \deg(\chi) + p, \\ \mathrm{Rank}(\mathcal{H}(I,q)) &= \deg(\chi) - p. \end{aligned} \tag{2.3.2}$$

With this, it is simple and efficient to compute the number of real roots of a system of polynomial equations.

Chapter 3

Time-Optimal Neural Feedback Control

This chapter is based on the paper 'Time-Optimal Neural Feedback Control of Nilpotent Systems as a Binary Classification Problem' which can be found at [9]. Section 3.5 has been added to further extend on the content of [9].

3.1 Introduction

A computational method for the synthesis of time-optimal feedback control laws for linear nilpotent systems is proposed. The method is based on the use of the bang-bang theorem, which leads to a characterisation of the time-optimal trajectory as a parameter-dependent polynomial system for the control switching sequence. A deflated Newton's method is then applied to exhaust all the real roots of the polynomial system. The root-finding procedure is informed by the Hermite quadratic form, which provides a sharp estimate on the number of real roots to be found. In the second part of the chapter, the polynomial systems are sampled and solved to generate a synthetic dataset for the construction of a time-optimal deep neural network, interpreted as a binary classifier, via supervised learning. Numerical tests with nilpotent systems of increasing dimension assess the accuracy, robustness, and real-time-control capabilities of the approximate control law. Time-optimal control problems have a prominent place in control theory due to their many applications in robotics, aerospace, and trajectory planning [14, 81, 83]. Recall the Bang-Bang theorem, here listed as Proposition 2.1.3. This classical result dating back to the

early days of optimal control states that, for finite-dimensional, linear time-optimal processes where the control variable is constrained to a convex, closed, and bounded polyhedron, the optimal control signal is piecewise constant, taking all its values at the vertices of the polyhedron (see the different formulations in e.g. [7, 52, 65]). Moreover, under Kalmans controllability criterion, an upper bound on the number of switches of the piecewise-constant control can be determined as being 1 less than the dimension. From a state space perspective, such a result transforms the time-optimal control synthesis into a geometric problem, that is, the identification of a switching surfaces splitting the state space into regions where different vertices of the control polyhedron are selected as the instantaneous optimal control action. Hence, while the original bang-bang theorem follows from first-order optimality conditions for a given initial state, it also provides a characterisation of the optimal feedback control law as a piecewise constant map with a finite number of output values. With no loss of generality, in this chapter we restrict our presentation to scalar control signals taking values in $[-1, 1]$, for which the optimal feedback law corresponds to a binary classifier to $\{-1, 1\}$. We study the construction of such a binary classifier via supervised learning with synthetic data obtained from sampling optimal trajectories, which are parametrised in terms of the switching structure of the optimal control signal.

In the first part of this chapter, we follow a similar approach as in [82]. Here we cast a time-optimal control problem, for which a parametrisation of the switching control sequence and time integration of the dynamics leads to a polynomial system of equations for the switching times. Since there is a unique solution to the time-optimal control problem, we only need to determine the existence of a non-negative solution of the polynomial system given a choice of the initial term of the switching control sequence, -1 or 1. This solvability problem is studied using Gröbner bases, Sturm sequences, and `Macaulay2` as an effective computer algebra system for related calculations. Our contribution is to propose a methodology that circumvents the computational complexity and scalability issues of the use of Gröbner bases:

- Instead of analyzing the existence of non-negative solutions to the polynomial system, we apply a variant of Newton's method which can effectively exhaust all possible roots. We follow a deflated Newton approach as originally proposed in [16] for polynomial systems and later extended to a wider class of problems and applications [10, 19, 38, 53].
- The stopping of the deflation method is determined by the number of roots

it is expected to find. We apply Hermite's quadratic form as a mechanism to establish the number of roots of the polynomial systems under study. The idea of using Hermite quadratic forms to solve polynomial systems with real parameters is discussed in [64] and its use for sign determination in [6, 8, 23]. A recent application of the above to semi-algebraic systems with real parameters can be found in [41].

- A real-time feedback control is built using supervised learning. Recall the black-box which determines the initial value of the optimal control steering a given initial condition to the origin. We can replace solving the optimal control problem (2.1.3) with finding the solution of the polynomial system associated to this control problem. The black-box receives as an input an initial condition, which enters as a parameter in the polynomial system, and outputs the first control value of the optimal switching sequence, which leads to the construction of a binary classifier mapping the nth dimensional state space to $\{-1, 1\}$. Inspired by recent work on supervised learning for optimal feedback control [2, 5, 22, 33, 49, 58] we train a deep neural network to approximate the state-to-control feedback map using synthetic data. This approach bypasses the need for iterative recomputation of optimal trajectories, which is replaced by real-time model evaluation.

The rest of the chapter is organised as follows. In Section 3.2, we introduce nilpotent linear time-invariant systems with a chain of integrators as a prototypical case, and we provide a characterisation of the time-optimal control problem as a polynomial system for the switching times of the control sequence. In Section 3.3, we present the deflated Newton's method we apply to find all possible roots of the polynomial system. In Section 3.4, we introduce the Hermite quadratic form and its use to determine the existence of non-negative real roots of a system of polynomial equations. We present the methodology by which we count the number of real roots of systems of polynomial equations and construct the black-box. We then in Section 3.5 present an open-loop solver for the optimal control problem that uses the deflated Newton's method informed by the Hermite quadratic form, as well as describing a black-box for computing the number of strictly non-negative roots of a polynomial system using the Hermite quadratic form. In Section 3.6, we discuss the approximation of a binary classifier via supervised learning with deep neural networks and synthetic data generated from sampling polynomial systems for different initial conditions. In Section 3.7, we assess the performance and robustness of the proposed

approach on chains of integrators up to dimension 5. Section 3.8 presents concluding remarks and future research directions.

3.2 Time-Optimal Control of Nilpotent Systems

We study the linear time-optimal control problem (see (2.1.3))

$$\min_{u\in\mathcal{U}} \; T \quad \text{subject to} \quad \begin{cases} \dot{\boldsymbol{x}}(t) = A\boldsymbol{x}(t) + \boldsymbol{b}u(t), \\ \boldsymbol{x}(0) = \boldsymbol{x}_0, \\ \boldsymbol{x}(T) = \boldsymbol{0}, \end{cases}$$

where $\mathcal{U} \equiv L^\infty\big([0,+\infty);[-1,1]\big)$, A is an $n\times n$ real matrix, $\boldsymbol{b}\in\mathbb{R}^n$, and $\boldsymbol{x}_0\in\mathbb{R}^n$ a given initial condition. We begin by recalling a well-known characterisation of the solution to time-optimal control problem (2.1.3) Proposition 2.1.3.

Recall that this result gives us a parametrisation of the total time $T = t_1+\cdots+t_n$ for non-negative t_i's, and recall the time domain Ω is the union of subintervals $\Omega_i = [T_{i-1}, T_i)$, with $T_0 = 0$, and $T_i = \sum_{j=1}^{i} t_j$, for every $1\le i\le n-1$, and $\Omega_n = [T_{n-1}, T_n]$. This induces a further parametrisation of the optimal control signal u as a piecewise-constant function in time, taking the value $u(0) = \pm 1$ in Ω_1, and $(-1)^{i-1}u(0)$ in the interval Ω_i. As we are studying the case where the matrix A is a nilpotent matrix with nilpotency index n, and $A = (A_{ij})_{i,j}$ is similar to an $n\times n$ nilpotent Jordan block of index n by Proposition 2.1.7, it is no loss of generality to assume that $A_{ij} = \delta_{i,j-1}$, for $i,j = 1,\dots,n$, where δ is the Kronecker delta due to Proposition 2.1.8. This covers a class of relevant problems, such as nth order chain of integrators and allows the following characterisation of the optimal control signal.

Proposition 3.2.1 ([9, Proposition 2.2])**.** *Let A be an $n\times n$ nilpotent Jordan block and $\boldsymbol{b}\in\mathbb{R}^n$ satisfying the assumptions in Theorem 2.1.3. Given an initial condition $\boldsymbol{x}_0$, the non-negative increments $\{t_i\}_{i=1}^n$ such that $T = t_1+\cdots+t_n$, associated to the time-optimal control signal $u(t) = u_i$, $\forall t\in\Omega_i$, with $u_i = \pm 1$, satisfy the polynomial system*

$$0 = -u(0)\sum_{k=1}^{n-i+1} b_{k+i-1} \sum_{\substack{\alpha_1,\dots,\alpha_n\ge 0\\ \alpha_1+\cdots+\alpha_n=k}} (-1)^{\max\{j\colon \alpha_j\neq 0\}} \frac{t_1^{\alpha_1}\cdots t_n^{\alpha_n}}{\alpha_1!\cdots\alpha_n!} + \sum_{k=i}^{n} \frac{T^{k-i}}{(k-i)!}x_k, \quad (3.2.1)$$

for $i = 1,\dots,n$, where b_j, x_j denote the coordinates of $\boldsymbol{b}$ and $\boldsymbol{x}_0$, respectively.

Proof. First, recall that for a given control signal, the trajectory associated to the linear system in (2.1.3) is given in Proposition 2.1.5 by

$$\boldsymbol{x}(t) = e^{tA}\boldsymbol{x}(0) + e^{tA}\int_0^t e^{-\tau A}\boldsymbol{b}\,u(\tau)\,d\tau.$$

Evaluating at $T = t_1 + \cdots + t_n$, and imposing that $\boldsymbol{y}(T) = \mathbf{0}$ implies

$$\mathbf{0} = e^{TA}\boldsymbol{x}_0 + e^{TA}\int_0^T e^{-\tau A}\boldsymbol{b}\,u(\tau)\,d\tau,$$

and integration by parts yields

$$\begin{aligned}\mathbf{0} &= e^{AT}\boldsymbol{x}_0 + e^{TA}\sum_{k=1}^{n}\big(A^{k-1}\,e^{-TA}\boldsymbol{b}\,U_k(T) - A^{k-1}\boldsymbol{b}\,U_k(0)\big)\\ &= e^{TA}\boldsymbol{x}_0 + \sum_{k=1}^{n} A^{k-1}\,\boldsymbol{b}\,U_k(T),\end{aligned} \tag{3.2.2}$$

where

$$U_k(t) = \int_0^t U_{k-1}(\tau)\,d\tau, \text{ for } i \geq 1, \text{ with } U_1(t) = \int_0^t u(\tau)d\tau.$$

Since u is piecewise constant on the partition of Ω determined by the t_i's,

$$U_1(T) = u(0)\sum_{i=1}^{n}\int_{T_{i-1}}^{T_i}(-1)^{i-1}d\tau = u(0)(t_1 - t_2 + \ldots(-1)^{n-1}t_n),$$

where, as above, $T_i = t_1 + \cdots + t_i$. For any $1 \leq k \leq n-1$, we have

$$\begin{aligned}U_k(T) &= \int_0^T U_{k-1}(\tau)d\tau = \sum_{i=1}^{n}\int_{T_{i-1}}^{T_i} U_{k-1}(\tau)d\tau = u(0)\sum_{i=1}^{n}\int_{T_{i-1}}^{T_i}(-1)^{i-1}\frac{\tau^{k-1}}{(k-1)!}d\tau\\ &= u(0)\sum_{i=1}^{n}(-1)^{i-1}\left[\frac{(t_i + T_{i-1})^k}{k!} - \frac{(T_{i-1})^k}{k!}\right] = u(0)\sum_{i=1}^{n}\sum_{j=1}^{k}\frac{(-1)^{i-1}}{(k-j)!j!}t_i^j T_{i-1}^{k-j}.\end{aligned}$$

Since

$$(t_1 + \cdots + t_{i-1})^{k-j} = \sum_{\substack{\alpha_1+\cdots+\alpha_{i-1}=k-j\\ \alpha_1,\ldots,\alpha_{i-1}\geq 0}}\frac{(k-j)!}{\alpha_1!\alpha_2!\cdots\alpha_{i-1}!}t_1^{\alpha_1}\cdots t_{i-1}^{\alpha_{i-1}},$$

then

$$\begin{aligned}U_k(T) &= u(0)\sum_{i=1}^{n}\sum_{\substack{\alpha_1+\cdots+\alpha_i=k\\ \alpha_i,\ldots,\alpha_{i-1}\geq 0,\,\alpha_i>0}}\frac{(-1)^{i-1}}{\alpha_1!\cdots\alpha_i!}t_1^{\alpha_1}\cdots t_i^{\alpha_i}\\ &= -u(0)\sum_{\substack{\alpha_1+\cdots+\alpha_n=k\\ \alpha_1,\ldots,\alpha_n\geq 0}}\frac{(-1)^{\max\{i:\,\alpha_i\neq 0\}}}{\alpha_1!\cdots\alpha_n!}t_1^{\alpha_1}\cdots t_n^{\alpha_n}.\end{aligned} \tag{3.2.3}$$

On the other hand, since A is we have $A_{ij} = \delta_{i,j-1}$, for $i, j = 1, \dots, n$. Then, for $1 \leq k \leq n$ we have $(A^{k-1})_{i,j} = \delta_{i,i+k-1}$. Thus, $\left(A^{k-1}\boldsymbol{b}\right)_i = b_{k+i-1}$ for $i = 1, \dots, n-k+1$, and $\left(A^{k-1}\boldsymbol{b}\right)_i = 0$ otherwise.

Finally, replacing (3.2.3) in (3.2.2), and noting that the ith component of $e^{TA}\boldsymbol{x}_0$ is given by

$$(e^{TA}\boldsymbol{x}_0)_i = \sum_{k=1}^{n} (e^{TA})_{i,k} x_k = \sum_{k=i}^{n} \frac{T^{k-i}}{(k-i)!} x_k,$$

we obtain the equality (3.2.1), as required. □

Example 3.2.1. In $\mathbb{R}^3$, consider the vector $\boldsymbol{b} = (0, 0, 1)^\top$. A solution of (2.1.3) leads to the system of polynomial equations:

$$\begin{aligned}
0 &= u(0)(t_1 - t_2 + t_3) + x_3, \\
0 &= u(0)\left(\frac{t_1^2}{2} + t_1 t_2 + t_1 t_3 - \frac{t_2^2}{2} - t_2 t_3 + \frac{t_3^2}{2}\right) + x_2 + (t_1 + t_2 + t_3) x_3, \\
0 &= u(0)\left(\frac{t_3^3}{6} + \frac{t_1^2 t_2}{2} + \frac{t_1^2 t_3}{2} + \frac{t_1 t_2^2}{2} + t_1 t_2 t_3 + \frac{t_1 t_3^2}{2} - \frac{t_2^3}{6} - \frac{t_2^2 t_3}{2} - \frac{t_2 t_3^2}{2} + \frac{t_3^3}{6}\right) \\
&\quad + x_1 + (t_1 + t_2 + t_3) x_2 + \frac{(t_1 + t_2 + t_3)^2}{2} x_3,
\end{aligned}$$

and $u(0) = \pm 1$. ◇

With no loss of generality since we know that for scalar control systems in Jordan form, controllability is equivalent to requiring $b_n \neq 0$ (See also [20] for another proof), we will henceforth assume $\boldsymbol{b} = (0, \dots, 0, 1)^\top$, leading to a simplified version of (3.2.1)

$$0 = -u(0) \sum_{\substack{\alpha_1, \dots, \alpha_n \geq 0 \\ \alpha_1 + \cdots + \alpha_n = n - i + 1}} (-1)^{\max\{j \colon \alpha_j \neq 0\}} \frac{t_1^{\alpha_1} \cdots t_n^{\alpha_n}}{\alpha_1! \cdots \alpha_n!} + \sum_{k=i}^{n} \frac{(t_1 + \cdots + t_n)^{k-i}}{(k-i)!} x_k, \tag{3.2.4}$$

for $i = 1, \dots, n$. This system characterises the optimal control signal in the following way. For every initial condition $\boldsymbol{x}_0$, there exists a unique optimal control signal, which is determined by the choice of the initial control $u(0) = \pm 1$, and the existence of a set of non-negative t_i's related to that choice. In other words, if a non-negative sequence of t_i's is found for a given $\boldsymbol{x}$ and $u(0)$, then the switching sequence is the time-optimal solution to the problem. Otherwise, we are guaranteed a unique non-negative solution to (3.2.4) by taking $-u(0)$ as the starting element of the sequence and computing the associated switching times. In the following section, we turn our attention onto Newton's method to find roots of the polynomial system.

3.3 A deflated Newton's Method

In this section, we present a numerical technique that combines Newton's method with a deflation routine to determine all possible solutions of the polynomial system (3.2.4).

We denote by $f_1, \dots, f_n$ the polynomials in $\boldsymbol{t} = (t_1, \dots, t_n)$ from (3.2.4), which reads

$$\mathcal{F}(\boldsymbol{t}) = \big(f_1(\boldsymbol{t}), \dots, f_n(\boldsymbol{t})\big)^\top = \mathbf{0}.$$

Given an initial guess $\boldsymbol{t}_0$, recall that the multivariate Newton's method 2.1.3 generates a sequence

$$\boldsymbol{t}_{n+1} = \boldsymbol{t}_n - \mathcal{F}'(\boldsymbol{t}_n)^{-1}\mathcal{F}(\boldsymbol{t}_n) \tag{3.3.1}$$

where $\mathcal{F}'$ denotes the Jacobian of $\mathcal{F}$. Assuming that this iteration converges to a root of (3.2.4), we have no guarantees that it will converge to a non-negative solution and, unless all solutions have been found, this cannot be interpreted as non-existence of non-negative solutions. Hence, we resort to a deflated Newton method to find all possible roots, which we describe in the following.

When using an iterative algorithm to find the roots of a function $\mathcal{F}(\boldsymbol{t})$, one can search for new, distinct roots of $\mathcal{F}(\boldsymbol{t})$ by applying the same iterative algorithm to a new deflated function $\mathcal{G}(\boldsymbol{t})$ whose zeros coincide with the roots of $\mathcal{F}(\boldsymbol{t})$, except for those already found. Specifically, if $\mathcal{F}(\boldsymbol{r}) = \mathbf{0}$, for some $\boldsymbol{r} \in \mathbb{R}^n$, we define

$$\mathcal{G}(\boldsymbol{t}) = \mathcal{M}(\boldsymbol{t}, \boldsymbol{r})\mathcal{F}(\boldsymbol{t}),$$

where $\mathcal{M}(\boldsymbol{t}, \boldsymbol{r})$ is the deflation matrix defined in 2.1.5, which is invertible for all $\boldsymbol{t} \in \mathbb{R}^n \setminus \{\boldsymbol{r}\}$, and $\mathcal{M}(\boldsymbol{t}, \boldsymbol{r})\mathcal{F}(\boldsymbol{t}) = \mathbf{0}$ if and only if $\mathcal{F}(\boldsymbol{t}) = \mathbf{0}$. This involves systematically altering the residual of the polynomial system to eliminate solutions that have already been identified. In this work, we consider

$$\mathcal{G}(\boldsymbol{t}) = \left(\mathbb{I}\frac{1}{\eta(\boldsymbol{t})} + \mathbb{I}\xi\right)\mathcal{F}(\boldsymbol{t})\,, \text{ for } \eta(\boldsymbol{t}) = \|\boldsymbol{t} - \boldsymbol{r}\|_2^p\,, \tag{3.3.2}$$

where $\mathbb{I} \in \mathbb{R}^{n\times n}$ is the identity matrix, the *deflation power* p is a positive integer, and the *deflated residual* ξ is a non-negative real number. The *deflating term* $\eta(\boldsymbol{t})^{-1}$ ensures that $\mathcal{G}(\boldsymbol{r}) \neq \mathbf{0}$, while the shift $\xi \geq 0$ forces the deflated residual to not vanish artificially as $\|\boldsymbol{t} - \boldsymbol{r}\|_2 \to \infty$.

The use of Newton's method requires the computation of the derivative of $\mathcal{G}(\boldsymbol{t})$. This can be done efficiently both in memory and computational cost by leveraging

the information about the previous deflation iterates. If $\boldsymbol{r}_0, \dots, \boldsymbol{r}_{m-1}$ are roots of $\mathcal{F}$, at the mth iteration, the deflation term reads

$$\eta_m = \eta_{m-1}(\boldsymbol{t})\tau(\boldsymbol{t})\,, \quad \text{with} \quad \eta_{m-1}(\boldsymbol{t}) = \prod_{i=0}^{m-1} \|\boldsymbol{t} - \boldsymbol{r}_i\|_2^p\,,$$

for a temporal variable $\tau(\boldsymbol{t}) = \|\boldsymbol{t} - \boldsymbol{r}_m\|_2^p$. The derivative of the polynomial system after m deflations is given by

$$\mathcal{G}' = \frac{\mathcal{F}'}{\eta_m} + \xi\mathcal{F}' - \frac{\mathcal{F}}{\eta_m^2} \otimes \left(\eta'_{m-1}\,\tau + \eta_{m-1}\,\tau'\right). \tag{3.3.3}$$

For the time-optimal polynomial system (3.2.4), the partial derivative of the ith component f_i of $\mathcal{F}$ with respect to the switching time t_q is given by the polynomial

$$\partial_{t_q} f_i = -u(0) \sum_{\substack{\alpha_1,\dots,\alpha_n \geq 0,\, \alpha_q \neq 0 \\ \alpha_1+\cdots+\alpha_n = n-i+1}} (-1)^{\max\{j\colon \alpha_j \neq 0\}} \frac{t_1^{\alpha_1} \cdots t_q^{\alpha_q - 1} \cdots t_n^{\alpha_n}}{\alpha_1! \cdots (\alpha_q - 1)! \cdots \alpha_n!} + \sum_{k=i+1}^{n} \frac{T^{k-i-1}}{(k-i-1)!} x_k.$$

A pseudocode for this routine is provided in Algorithm 2. A crucial consideration for the deflation algorithm is the selection of suitable stopping criteria, as once all roots have been found, the algorithm will naturally fail to converge. However, numerical instability can also prevent convergence as the number of roots in the deflation operator increases. If we choose the upper bound given to us by Masser and Wüstholz as a generalisation of Bézout's theorem (see [56, Theorem II]), which is that the number of real roots of a system of polynomial equations with degrees $d_1, \dots, d_k$ is at most the product $\prod_{i=1}^{k} d_i$. In (3.2.4), this corresponds to having at most $n!$ real roots. This, as it turns out, is drastically larger than the exact number in most situations. Therefore, establishing a tighter bound on the number of real solutions in our polynomial system is fundamental. The next section explores this bound by drawing from real algebraic geometry and utilising the Hermite quadratic form.

Algorithm 2: Deflated Newton

Data: Function $\mathcal{F}$ and its derivative $\mathcal{F}'$, initial `guess`, expected number of roots $n_{\texttt{roots}}$, deflation parameters p, ξ, convergence tolerance $\varepsilon \ll 1$

`max_iteration` $= 10^3$, `solutions` $\leftarrow \{\}$; `initialising`

$\eta_0(*) \leftarrow 1$, $\eta_0'(*) \leftarrow 0$, $\tau(*) \leftarrow 1$, $\tau'(*) \leftarrow 0$; `deflation operators`

$m \leftarrow 0$; `count of identified roots`

while $m < n_{roots}$ **do**

  $\boldsymbol{t} \leftarrow$ `guess` ; `initial guess`

  `update` $\gg 1$, `iteration` $\leftarrow 0$; `reset convergence flags`

  while *update* $> \varepsilon$ *and* *iteration* $<$ *max_iteration* **do**

    `iteration` $+= 1$;

    $f \leftarrow \mathcal{F}(\boldsymbol{t})$, $df \leftarrow \mathcal{F}'(\boldsymbol{t})$;

    $\mathcal{G} \leftarrow$ (3.3.2), $\mathcal{G}' \leftarrow$ (3.3.3); `compute deflated system`

    $\boldsymbol{t}_{\text{new}} \leftarrow \boldsymbol{t} - \texttt{pinv}(\mathcal{G}') \cdot \mathcal{G}$; `update root candidate`

    `update` $\leftarrow \|\boldsymbol{t}_{\text{new}} - \boldsymbol{t}\|$;

    $\boldsymbol{t} \leftarrow \boldsymbol{t}_{\text{new}}$;

  if *iteration* $=$ *max_iteration* **then**

    break

  else

    `solutions` $\leftarrow$ `solutions` $\cup \{\boldsymbol{t}\}$; `store solution`

    $m += 1$

    $\tau(*) \leftarrow \| * - \boldsymbol{t}\|^p$; `Update deflation operators`

    $\tau'(*) \leftarrow p \cdot \| * - \boldsymbol{t}\|^{p-2} \cdot (* - \boldsymbol{t})$;

    $\eta_m(*) = \eta_{m-1}(*)\tau(*)$;

Result: list `solutions`, containing the roots of $\mathcal{F}$

3.4 The Hermite Quadratic Form

We present the Hermite quadratic form method, as introduced in [6], as a tool to determine the number of roots in the polynomial system (3.2.4). We denote by $R = \mathbb{R}[t_1, \dots, t_n]$ the ring of polynomials in $\boldsymbol{t}$ with real coefficients, hence for every f_i in the system $\mathcal{F}$, $f_i \in R$. Let

$$I = \langle f_i \colon i = 1, \dots, n \rangle = \left\{ \sum_{i=1}^{n} g_i f_i \colon g_i \in R \right\} \subseteq R, \tag{3.4.1}$$

be the ideal generated by $\mathcal{F}$.

To apply the Hermite quadratic form method, it is necessary to ensure that R/I

is a finite-dimensional vector space, which is equivalent to I being zero-dimensional. This property can be verified using a monomial ordering, which can be used either to establish certain properties of the monomials which are powers of the variables t_i or to compute a Gröbner basis for the ideal I. Alternatively, if we can verify that the polynomial system $\mathcal{F}$ defining J has only a finite number of solutions in $\mathbb{C}$ this also implies that J is zero-dimensional. For further details on these methods, see, for example, [30, §3, Chapter 5]. Recall the definition of the Hermite quadratic form of the pair $(I, 1)$, denoted $\mathcal{H}(I)$. We show an example computation of the Hermite quadratic form of an ideal.

Example 3.4.1. Taking $n = 2$ and $u(0) = 1$ in (3.2.4) leads to

$$f_1 = t_1 - t_2 + x_2, \tag{3.4.2}$$

$$f_2 = \frac{t_1^2}{2} + t_1 t_2 - \frac{t_2^2}{2} + x_1 + (t_1 + t_2)x_2. \tag{3.4.3}$$

These are polynomials in $R = \mathbb{R}[t_1, t_2]$, and we take $J = \langle f_1, f_2 \rangle$. This system has a finite number of solutions because if $t_2 = t_1 + x_2$ by (3.4.2), then (3.4.3) leads to a quadratic equation in t_1, which has at most two solutions. We see that $\mathcal{B} = \{1, t_2\}$ is a basis for the vector space R/J. Taking $\beta_1 = 1$, and $\beta_2 = t_2$, then the map $\mathcal{L}_{12}(I)$ is given by $1 + J \mapsto t_2 + I$, and $t_2 + J \mapsto t_2^2 + I = -x_1 + \frac{1}{2}x_2^2 + I$. Thus,

$$L_{12}(I) = \begin{pmatrix} 0 & -x_1 + \frac{1}{2}x_2^2 \\ 1 & 0 \end{pmatrix},$$

and so $\operatorname{trace}(L_{12}(I)) = 0$. Similarly, we compute $L_{ij}(I)$ for $i, j = 1, 2$, and we get the Hermite quadratic form of J, which is given by the matrix $\mathcal{H}(I) = \begin{pmatrix} 2 & 0 \\ 0 & x_2^2 - 2x_1 \end{pmatrix}$. ◇

In practice, for a given value of $\boldsymbol{x}$, we use `Macaulay2` to compute the dimension of the ideal I and verify if it is zero. The Hermite quadratic form for parametric systems of polynomial equations has been studied in [64]. Here, it is shown that for a given parameter, it is possible to obtain the Hermite quadratic of the associated polynomial system, as long as the denominator of the parameter-augmented Hermite quadratic form does not vanish.

If the ideal I of the polynomial system $\mathcal{F}$ is zero-dimensional, the Hermite quadratic form $\mathcal{H}(I)$ can be used to count the number of real roots of $\mathcal{F}$ by Proposition 2.3.1. We present the special case of this result when $q = 1$ here for completeness.

Proposition 3.4.1 ([6, Theorem 4.100])**.** *Suppose the system of polynomial equation $\mathcal{F} \subseteq R$ has finitely many solutions. Let $I = \langle f_i \colon i = 1, \ldots, n \rangle$, and $\mathcal{H}(I)$ be the Hermite quadratic form of I. Then,*

$$\text{Rank}(\mathcal{H}(I)) = |\mathbb{V}_{\mathbb{C}}(I)|, \text{ and } \text{Sign}(\mathcal{H}(I)) = |\mathbb{V}_{\mathbb{R}}(I)|.$$

Remark 3.4.1. By Proposition 3.4.1, the number of real roots of a polynomial system can be computed by counting the positive and negative real roots of the characteristic polynomial of the matrix $\mathcal{H}(I)$.

Example 3.4.2. In Example 3.4.1, the signature of $\mathcal{H}(I)$ is determined by the sign of the eigenvalue $x_2^2 - 2x_1$. If $x_2^2 - 2x_1 > 0$, then both eigenvalues of $\mathcal{H}(I)$ are positive, meaning that the system of polynomial equations has two real solutions. If $x_2^2 - 2x_1 = 0$, then the system has only one real solution. If $x_2^2 - 2x_1 < 0$, then $\text{Sign}(\mathcal{H}(I)) = 0$, and the system has no real solutions. These three cases are illustrated in Figure 3.1 ◇

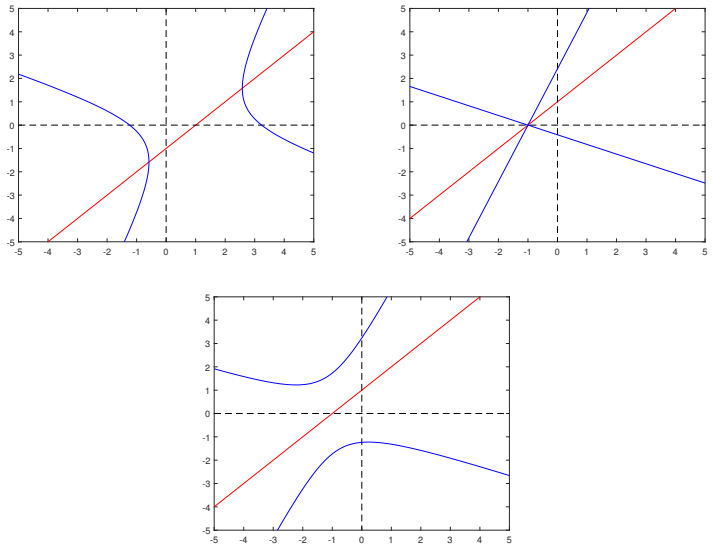

Figure 3.1: Graph of the two dimensional polynomial system from Example 3.4.1. The number of real solutions of the system depend on the sign of the diagonal elements in the matrix $\mathcal{H}(I)$. We have three cases: $x_2^2 - 2x_1 > 0$ (left), $x_2^2 - 2x_1 = 0$ (middle), and $x_2^2 - 2x_1 < 0$ (right).

The Hermite quadratic form feeds into Algorithm 2 by providing n_{roots}, the expected of number of real roots. We set $n_{\text{roots}} = \text{Sign}(\mathcal{H}(I))$. This quantity is valid

for fixed $\boldsymbol{x}$ and $u(0)$, and once n_{roots} has been reached without finding a non-negative solution, we are guaranteed the existence of a non-negative solution by taking $-u(0)$.

All of our computations are completed in `Macaulay2` using the `RealRoots` package [42]. The command "traceCount(I)" returns the signature of Hermite quadratic form of I. Within this computation, a Gröbner basis of J is computed when finding a basis of R/I. As shown in [34], the algorithms to compute the Gröbner basis of the ideal J will generate polynomials with total degree bounded above by $2(n^2/2+n)^{2^{n-1}}$. As a result, as we increase the number of variables n, the Gröbner basis computation will increase double-exponentially. In Section 6, we report results where the Hermite quadratic form has been computed for polynomial systems up to dimension 5, although computations have been successfully performed up to dimension 7.

Recall that if we set $u(0) = \pm 1$ in (3.2.4), then the existence of a non-negative real solution corresponds to the existence of an optimal control with initial value ± 1. Suppose we choose a value for $u(0)$, and let $\mathcal{F}$ be the polynomial system in (3.2.4), and J the corresponding ideal in $\mathbb{R}[t_1, \ldots, t_n]$. We can use the Hermite quadratic form of J to create a purely algebraic black-box.

Proposition 3.4.2. *The polynomial system $\mathcal{F}$ has a non-negative real root if and only if the following inequation holds*

$$\sum_{\substack{0 \leq \alpha_i \leq 1 \\ 1 \leq i \leq n}} \text{Sign}(\mathcal{H}(J, t_1^{\alpha_1} \cdots t_n^{\alpha_n})) > 0.$$

Proof. We need only to show that $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_n) \in \mathbb{R}^n$ is a root of $\mathcal{F}$ that is not in $\mathbb{R}^n_{\geq 0}$ if and only if

$$\sum_{\substack{0 \leq \alpha_i \leq 1 \\ 1 \leq i \leq n}} \text{Sign}(\tau_1^{\alpha_1} \cdots \tau_n^{\alpha_n}) = 0.$$

We will say here as convention that $0^0 = 1$, because we consider it to be the value of polynomial $\tau_i^0 = 1$ at $\tau_i = 0$ for any i. Suppose $\boldsymbol{\tau}$ is not in $\mathbb{R}^n_{\geq 0}$, and without loss of

generality that $\tau_1 < 0$. We have

$$\begin{aligned}\sum_{\substack{0\leq\alpha_i\leq 1\\ 1\leq i\leq n}} \operatorname{Sign}(\tau_1^{\alpha_1}\cdots\tau_n^{\alpha_n}) &= \sum_{\substack{0\leq\alpha_i\leq 1\\ 1\leq i\leq n}} \operatorname{Sign}(\tau_1^{\alpha_1})\operatorname{Sign}(\tau_2^{\alpha_2}\cdots\tau_n^{\alpha_n})\\ &= \sum_{\alpha_1=0}^{1} \operatorname{Sign}(\tau_1^{\alpha_1}) \sum_{\substack{0\leq\alpha_i\leq 1\\ 2\leq i\leq n}} \operatorname{Sign}(\tau_2^{\alpha_2}\cdots\tau_n^{\alpha_n})\\ &= \sum_{\alpha_1=0}^{1} (-1)^{\alpha_1} \sum_{\substack{0\leq\alpha_i\leq 1\\ 2\leq i\leq n}} \operatorname{Sign}(\tau_2^{\alpha_2}\cdots\tau_n^{\alpha_n})\\ &= 0.\end{aligned}$$

Then suppose $\boldsymbol{\tau}$ is a root of $\mathcal{F}$ as before, but is in $\mathbb{R}^n_{\geq 0}$. Then suppose without loss of generality that $\tau_i = 0$ for $i = 0, \ldots, k$ for some $k \leq n$, and $\tau_i > 0$ for $k < i \leq n$. Then,

$$\begin{aligned}\sum_{\substack{0\leq\alpha_i\leq 1\\ 1\leq i\leq n}} \operatorname{Sign}(\tau_1^{\alpha_1}\cdots\tau_n^{\alpha_n}) &= \sum_{\substack{0\leq\alpha_i\leq 1\\ 1\leq i\leq k}} \operatorname{Sign}(\tau_1^{\alpha_1}\cdots\tau_k^{\alpha_k}) \sum_{\substack{0\leq\alpha_i\leq 1\\ k+1\leq i\leq n}} \operatorname{Sign}(\tau_{k+1}^{\alpha_{k+1}}\cdots\tau_n^{\alpha_n})\\ &= \sum_{\substack{\alpha_i=0\\ 1\leq i\leq k}} \operatorname{Sign}(\tau_1^{\alpha_1}\cdots\tau_k^{\alpha_k}) \sum_{\substack{0\leq\alpha_i\leq 1\\ k+1\leq i\leq n}} \operatorname{Sign}(\tau_{k+1}^{\alpha_{k+1}}\cdots\tau_n^{\alpha_n}).\end{aligned}$$

The, since $\tau_1^0\cdots\tau_k^0 = 1$ by convention, and $\operatorname{Sign}(\tau_{k+1}^{\alpha_{k+1}}\cdots\tau_n^{\alpha_n}) = 1$ for each α_i for $i = k+1, \ldots, n$, we are done. □

We should note that the value of

$$\sum_{0\leq\alpha_1,\ldots,\alpha_n\leq 1} \operatorname{Sign}(\mathcal{H}(J, t_1^{\alpha_1}\cdots t_n^{\alpha_n})),$$

is not the total number of non-negative real roots of $\mathcal{F}$. It is, in most cases, larger, but what makes it useful is that it vanishes if and only if there are no non-negative real roots of $\mathcal{F}$, and so it can be used as an indicator for their existence, and hence it can be used as the black-box.

3.5 Construction of an open-loop solver and an algebraic black-box

In this section we show how to use Newton's method and Hermite's quadratic form to create an open-loop solver for the polynomials (3.2.4). We name it `OpenLoopSolver`.

Algorithm 3: `OpenLoopSolver`

Data: Initial condition $\boldsymbol{x}_0 = (x_1, \dots, x_n)^\top$, an initial control value $u(0)$.
Initial `guess`, deflation parameters p, ξ, convergence tolerance $\varepsilon \ll 1$

for $i = 1, \dots, n$ **do**
  $f_i \leftarrow -u(0) \sum_{\alpha_1,\dots,\alpha_n \geq 0 \atop \alpha_1+\cdots+\alpha_n = n-i+1} (-1)^{\max\{j\colon \alpha_j \neq 0\}} \frac{t_1^{\alpha_1}\cdots t_n^{\alpha_n}}{\alpha_1!\cdots\alpha_n!} + \sum_{k=i}^{n} \frac{(t_1+\cdots+t_n)^{k-i}}{(k-i)!} x_k$;
$\mathcal{F} \leftarrow (f_1, \dots, f_n)^\top$;
$I \leftarrow \langle f_1, \dots, f_n \rangle$;
$n_{\texttt{roots}} \leftarrow \mathrm{Sign}(\mathcal{H}(I))$;
`solutions` $\leftarrow$ Deflated Newton$(\mathcal{F}, \mathcal{F}', \texttt{guess}, n_{\texttt{roots}}, p, \xi, \varepsilon)$;
for $(t_1, \dots, t_n) \in$ *`solutions`* **do**
  if $t_1, \dots, t_n \geq 0$ **then**
    `SwitchingTimes` $\leftarrow (t_1, \dots, t_n)$;
    $u(0) \leftarrow 1$;
else
  `OpenLoopSolver`$(\boldsymbol{x}_0, -u(0), \texttt{guess}, p, \xi, \varepsilon)$;

Result: $u(0)$, the initial value of the optimal control. $(t_1, \dots, t_n)$, the switching times of the optimal control.

It is a combination of the deflated Newton's method with the Hermite quadratic form and is constructed in Algorithm 3. In Section 3.6, we will use this algorithm to generate a synthetic dataset for the training of a binary classifier neural network.

As discussed at the end of Section 3.4, there is a way to use the Hermite quadratic form to create a black-box. It is a purely algebraic method that was not discussed in [9]. We name it `BlackBoxHermite` and describe the procedure in Algorithm 4.

Notice that in `BlackBoxHermite`, we presuppose that $u(0) = 1$ when defining the polynomial system. As discussed, if a non-negative solution to (3.2.4) is found with $u(0) = 1$, then the optimal control starts with 1, and if there is no non-negative solution we must have $u(0) = -1$. We note that implementing `BlackBoxHermite` as a real-time way to determine the optimal control will be difficult. Mainly for the reasons that Gröbner bases have been found to be unfeasible. Since at every time-step in Algorithm 1 we are required to compute Hermite quadratic forms, we must be aware that this procedure is purely algebraic, and thus no consideration has been taken for efficiency. Luckily, for each of the 2^n quadratic forms whose signatures we must calculate, each one is defined over the same basis, and so there is only one Gröbner basis calculation per time-step.

Algorithm 4: `BlackBoxHermite`

Data: Initial condition $\boldsymbol{x}_0 = (x_1, \ldots, x_n)^\top$.

for $i = 1, \ldots, n$ **do**

$f_i \leftarrow -\sum_{\substack{\alpha_1,\ldots,\alpha_n \geq 0 \\ \alpha_1+\cdots+\alpha_n = n-i+1}} (-1)^{\max\{j\colon \alpha_j \neq 0\}} \frac{t_1^{\alpha_1}\cdots t_n^{\alpha_n}}{\alpha_1!\cdots\alpha_n!} + \sum_{k=i}^{n} \frac{(t_1+\cdots+t_n)^{k-i}}{(k-i)!} x_k$;

$I \leftarrow \langle f_1, \ldots, f_n \rangle$;

`rootCounter` $\leftarrow \sum_{0 \leq \alpha_1,\ldots,\alpha_n \leq 1} \mathrm{Sign}(\mathcal{H}(I, t_1^{\alpha_1}\cdots t_n^{\alpha_n}))$;

if *rootCounter* > 0 **then**

$u_0 \leftarrow 1$;

else

$u_0 \leftarrow -1$;

Result: $u(0) = u_0$, the initial value of the optimal control.

A potential way to overcome this is via the work in [64], which proves that the roots of *parametric* polynomial systems can be found in the same way using Hermite quadratic forms. They consider polynomials over function fields such as $\mathbb{Q}(x_1, \ldots, x_n)$ instead of $\mathbb{R}$, and show that Hermite matrices over $\mathbb{Q}(x_1, \ldots, x_n)$, when values are chosen for the parameters $x_1, \ldots, x_n$, then the corresponding matrix is the Hermite matrix of the original polynomial system when the values for the parameters are substituted in. If we parametrise (3.2.4) in terms of $x_1, \ldots, x_n$, the initial condition, then it may be possible to precompute

$$\mathcal{H}(I, t_1^{\alpha_1} \cdots t_n^{\alpha_n}),$$

off-line for each monomial $t_1^{\alpha_1} \cdots t_n^{\alpha_n}$. Compared to the computation of the Hermite quadratic form itself, the signature is a fast computation, and so the only on-line computations needed will be substitutions into polynomials, computations of signatures of matrices, addition of 2^n such signatures, and verifying inequalities. Of course, working over function fields such as $\mathbb{Q}(x_1, \ldots, x_n)$ introduces the possibility of denominators that may vanish when certain values are chosen for the parameters. To the best of our knowledge there is no explicit way to know in general which parameter values will cause the denominators that appear to vanish, and so this precomputation step is not fully justified in the absence of further investigation.

3.6 Constructing a neural feedback law as a binary classifier

The Deflation algorithm 2 generates open-loop controls, which are effective in an ideal setting where both system dynamics and initial conditions are exactly known, but not in real-world applications where perturbations lead to deviations from the optimal trajectory. This necessitates the development of a feedback control, and the construction of a black-box based on the open-loop solver. In deterministic optimal control, the construction of an optimal feedback law is obtained via dynamic programming, leading to a static, first-order Hamilton-Jacobi-Bellman PDE for the value function of the control problem. The optimal feedback law is obtained as a by-product. However, this method requires an accurate tracking of the discontinuous switching law, which can only be achieved using local, grid-based numerical methods, and hence it is limited to low-dimensional problems. In this section, we present an alternative synthesis method that circumvents the solution of the Hamilton-Jacobi-Bellman PDE by resorting to supervised learning. Here, we train a neural network using synthetic data from sampling open-loop, optimal trajectories obtained with the methodology presented in previous sections.

3.6.1 Feedforward Neural Network Classifier

Given the bang-bang (binary) nature of the control in the problem under consideration, we frame the feedback approximation as a classification task rather than relying on regression. This means that the output of our model will be of discrete nature in $\{-1, 1\}$.

We consider a class of models within the family of feedforward neural networks (NN), characterised by a sequential composition of nonlinear activation functions applied component-wise to affine transformations of the inputs. If we define the mth layer as $l_m(\boldsymbol{z}_m) = \sigma_m\left(A_m \boldsymbol{z}_m + \boldsymbol{b}_m\right)$, the parametric model reads

$$u_\theta(\boldsymbol{z}) = l_M \circ \ldots \circ l_2 \circ l_1(\boldsymbol{z})\,,$$

where u_θ is the estimated probability that $u(\boldsymbol{z}) = 1$, with trainable parameters $\{A_m, \boldsymbol{b}_m\}_{m=1}^{M-1}$, $A_m \in \mathbb{R}^{n_{m-1} \times n_m}$ and $\boldsymbol{b}_m \in \mathbb{R}^{n_m}$ being the weight matrix and bias vector of the mth layer respectively, which has n_m neurons. Here, the σ_ms, $m = 1, \cdots, M-1$, are nonlinear activation functions, applied component-wise to the

layer input variable $\boldsymbol{z}_m \in \mathbb{R}^{n_{m-1}}$, and outputting values to the next layer. We fix the final layer to be sigmoid

$$l_M(\boldsymbol{z}_M) = \frac{1}{1+e^{-\boldsymbol{z}_M}},$$

where, as before, the function is applied component-wise to the entries of the layer input $\boldsymbol{z}_M$. This function maps any real-valued input into the range $(0,1)$, making it particularly suitable for binary classification tasks, as the output can be interpreted as a probability:

$$\mathbb{P}\big[u(\boldsymbol{z})=1\big] = u_\theta(\boldsymbol{z}), \text{ and } \mathbb{P}\big[u(\boldsymbol{z})=-1\big] = 1-u_\theta(\boldsymbol{z}). \tag{3.6.1}$$

By denoting the approximation of the feedback control as $\tilde{u}(\boldsymbol{z}) \approx u(\boldsymbol{z})$, we define it as the most likely event under the probability u_θ:

$$\tilde{u}(\boldsymbol{z}) = \begin{cases} 1 & \text{if } u_\theta(\boldsymbol{z}) \geq 0.5, \\ -1 & \text{otherwise.} \end{cases}$$

This formulation not only provides a classification decision but also quantifies the confidence of the model in the forecast. In Section 3.7, we will state in each case the non-linear activation functions as these are problem specific choices.

3.6.2 Synthetic Data Generation

For the generation of the training and testing datasets, we utilise the deflated Newton's method discussed in Section 3.3 in the form of Algorithm 2, and solve the polynomial system in (3.2.4). We begin by generating N_s sample points in the state space, denoted by $\{\boldsymbol{x}_0^{(i)}\}_{i=1}^{N_s}$. For each initial condition, we evaluate the solvability of the polynomial system under the two possible initial control values, $u_0 = 1$ and $u_0 = -1$.

To ensure the accurate detection of switching surfaces, we further store the state and control information for a sequence of 100 points along the resulting optimal trajectory generated from each initial condition. This process results in a dataset consisting of $N = 100N_s$ entries:

$$\mathcal{T} = \{\boldsymbol{x}_0^{(i)}, u_0^{(i)}\}_i \cup \{\boldsymbol{x}^{(i)}(\tau_k), u^{(i)}(\tau_k)\}_{i,k}, \quad \text{for } i = 1,\dots,N_s, \text{ and } k = 1,\dots,100,$$

where $\boldsymbol{x}^{(i)}(t)$ is the optimal trajectory departing from $\boldsymbol{x}_0^{(i)}$ and τ_k's are uniformly spaced discrete times in $(0, T]$. For convenience, we will denote the input $u_\theta(\cdot)$ as $\boldsymbol{z}$, and rearrange the training set as

$$\mathcal{T} = \{\boldsymbol{z}^{(j)}, u^{(j)}\}_{j=1}^{N},$$

where $N = 100N_s$.

3.6.3 Training of the Model

The network is trained to minimise a cross-entropy loss function (3.6.2) on data, ensuring the output probabilities align with the provided labels. The training is performed using the Adam optimiser [50]. The dataset is split into two subsets: 90% for training, 10% for testing. The number of layers and neurons is selected to balance model complexity and computational efficiency, and hyperparameters such as learning rate and batch size are tuned using a grid search and aiming at maximising the approximation performance. These choices are problem-specific and will be clarified for each numerical test in Section 3.7.

In this case, the binary cross-entropy loss function measures how well the neural network's predicted probabilities align with the true binary labels ($u(\boldsymbol{z}) \in \{-1, 1\}$). The binary cross-entropy loss for a single data point $\boldsymbol{z}$ is given by:

$$L(\boldsymbol{z}, u(\boldsymbol{z})) = -\big[\lambda \log\big(u_\theta(\boldsymbol{z})\big) + (1-\lambda)\log\big(1 - u_\theta(\boldsymbol{z})\big)\big],$$

where:

- $\lambda \in \{0, 1\}$ is the true label, with $\lambda = 0$ corresponding to $u(\boldsymbol{z}) = -1$ and $\lambda = 1$ corresponding to $u(\boldsymbol{z}) = 1$,
- $u_\theta(\boldsymbol{z})$ is the network's predicted likelihood that $u(\boldsymbol{z}) = 1$, as in (3.6.1).

For a dataset of N samples, the total binary cross-entropy loss is:

$$\mathcal{L} = \frac{1}{N}\sum_{j=1}^{N}\big[-\lambda_j \log\big(u_\theta\big(\boldsymbol{z}^{(j)}\big)\big) - (1-\lambda_j)\log\big(1 - u_\theta\big(\boldsymbol{z}^{(j)}\big)\big)\big]. \tag{3.6.2}$$

This loss function encourages the network to predict probabilities $u_\theta(\boldsymbol{z}^{(j)})$ close to 1 when $\lambda_j = 1$ and close to 0 when $\lambda_j = 0$, effectively learning the mapping from states $\boldsymbol{z}^{(j)}$ to control labels $u^{(j)}$.

The model's performance is assessed by measuring the proportion of correctly predicted labels out of the total number of samples

$$\text{Accuracy} = \frac{1}{N}\sum_{j=1}^{N}\mathbf{1}_{\left(\tilde{u}(\boldsymbol{z}^{(j)}) = u^{(j)}\right)},$$

where $\mathbf{1}$ denotes the indicator function.

3.7 Numerical Tests

In this section, we evaluate the proposed methodology on an nth order nilpotent systems, where n ranges from 2 to 5. The discussion includes the process of data generation, the architecture of the model, and the choice of hyperparameters. We compare the trajectories controlled by the trained model against the optimal trajectories obtained using Algorithm 3 to solve (3.2.4). Across the numerical examples, we demonstrate that interpreting u_θ as a binary classifier with confidence information improve its approximation performance. We evaluate the robustness of the feedback approximation under noisy trajectories and highlight the computational advantages of using Hermite quadratic forms to determine a bound for the number of roots to be identified through deflation.

3.7.1 The double integrator

We consider the time-optimal control problem

$$\min_{u(t)} T \geq 0 \qquad \text{s.t.} \qquad \dot{\boldsymbol{x}} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \boldsymbol{x} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u, \quad \boldsymbol{x}(0) = \boldsymbol{x}_0, \quad \boldsymbol{x}(T) = \mathbf{0},$$

whose solution is given by a piecewise constant control, starting from an initial control $u_0 \in \{-1, 1\}$ and a switching sequence $\boldsymbol{t} = (t_1, t_2)^\top$, computed by solving the associated polynomial system (3.2.4), which for $n = 2$ reads:

$$\begin{cases} t_1 - t_2 + x_2 = 0, \\ \dfrac{t_1^2}{2} + t_1 t_2 - \dfrac{t_2^2}{2} + x_1 + x_2(t_1 + t_2) = 0, \end{cases} \quad \text{for} \quad (u_0 = 1),$$

$$\begin{cases} t_1 - t_2 - x_2 = 0, \\ \dfrac{t_1^2}{2} + t_1 t_2 - \dfrac{t_2^2}{2} - x_1 - x_2(t_1 + t_2) = 0, \end{cases} \quad \text{for} \quad (u_0 = -1). \tag{3.7.1}$$

Admissible solutions of (3.7.1) specify the optimal trajectory, starting from $\boldsymbol{x}_0 = (x_1, x_2)^\top$ with control u_0 and switching sequence[1] $\boldsymbol{t}$. We generate a dataset by sampling $N_s = 50$ uniformly distributed initial conditions $\left\{\boldsymbol{x}_0^{(i)}\right\}_{i=1}^{N_s}$ in the numerical domain $[-1, 1]^2$, generating couples of optimal state-actions $\left\{\boldsymbol{x}^{(i)}(\tau_k), u^{(i)}(\tau_k)\right\}$ for $0 = \tau_0 < \cdots \tau_k < \cdots \tau_{100} = T$ using deflation with an initial guess of $(t_1, t_2) =$

[1] For the sake of clarity, we recall that this switching sequence is to be interpreted in the following way: there is an initial control value u_0 in $[0, t_1)$, which then switches to $-u_0$ in $[t_1, t_1 + t_2]$, being the minimum time $T = t_1 + t_2$.

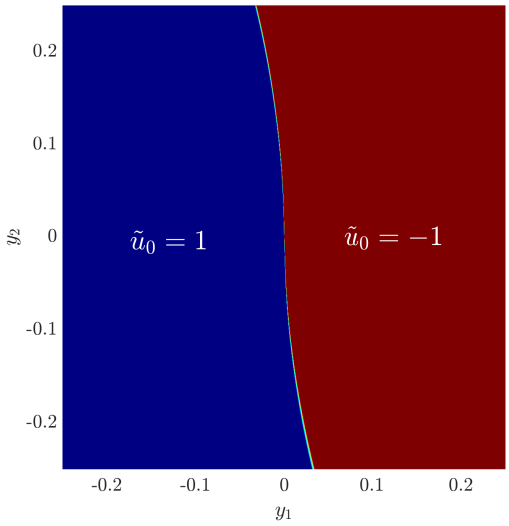

Figure 3.2: The trained classifier identifies the switching surface as a low-confidence region.

$(0.1, 0.1)$, a tolerance of 1×10^{-10}, and a maximum number of iterations of 1×10^3 per initial guess. When the deflated Newton's method converged, it required an average of 19.6721 iterations to converge. The average number of iterations required for convergence after each successive deflation can be seen in Appendix A.

We rely on the $N = 100 \cdot N_s$ generated data pairs

$$\mathcal{T} = \{\boldsymbol{z}^{(j)}, u^j\}_{j=1}^N,$$

to train a NN classifier mapping states $\boldsymbol{z}$ into the associated feedback $u_\theta(\boldsymbol{z})$. After a grid search over a number of possible feedforward architectures, we select $u_\theta(\cdot)$ to have a single hidden layer of width $n_2 = 100$ neurons and $\sigma_2 = \tanh(\cdot)$ as activation function. We train the model to minimise the loss function (3.6.2) over the training set using the Adam optimiser [50].

After training, the model achieves a test set accuracy of 99.38% with a corresponding loss $\mathcal{L} = 0.0156$. In Figure 3.2, we show how the trained classifier labels points in $[-0.25, 0.25]^2$. The model assigns controls with high probability to points on either side of the switching surface, while points lying directly on the surface are associated with lower confidence, as indicated by the lighter colors in the figure.

All optimal trajectories eventually intersect the switching hyperplane and travel along it until reaching the origin. Therefore, the model's accuracy on the switching surface is crucial for achieving a reliable feedback approximation. At this point, we

have two options:

- we rely on the classification provided by the neural network u_θ, disregarding its low confidence,
- or we define a confidence threshold, and if the confidence falls below this threshold, we invoke the open-loop solver, Algorithm 3 to determine the feedback control at the low-confidence point.

We compare these two procedures in Figure 3.3. The trajectory controlled through the neural feedback approximation is integrated via explicit Euler, with time-step 5^{-4}. The enhanced approximation, which is based on the polynomial system solution when the classifier confidence is below $\varepsilon = 0.01$, is triggered on 1% of the trajectory points and doesn't improve significantly the approximated optimal horizon T_{NN}. We conclude that, in this first example, even low-confidence points are classified correctly via u_θ.

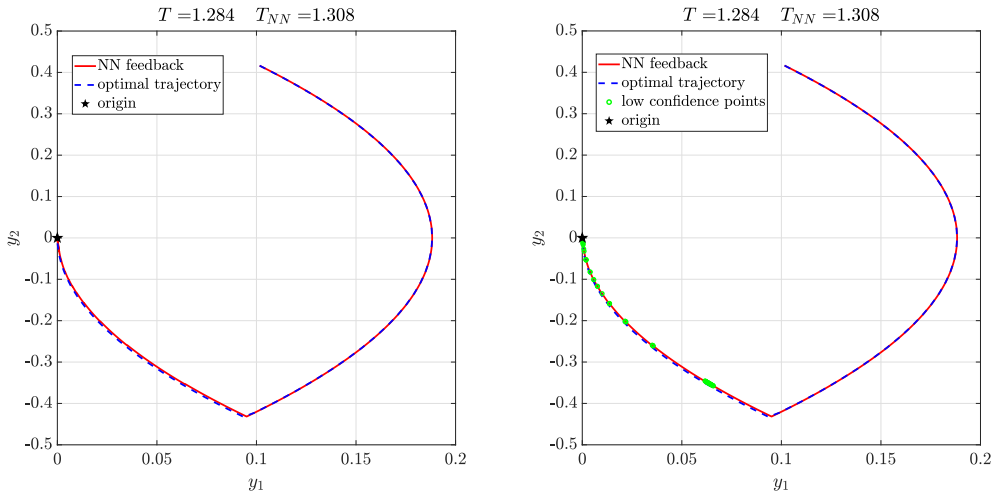

Figure 3.3: Comparison of the optimal trajectory computed via deflation with the one controlled with the classifier u_θ. On the left, the NN feedback is solely determined via u_θ, whilst on the right we rely on the identification of the solution of the polynomial system (3.7.1) to identify the feedback at low-confidence points. T denotes the exact minimum time.

3.7.2 Triple integrator

Similarly as in the first test, we consider the time-optimal control problem subject to

$$\dot{\boldsymbol{x}} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \boldsymbol{x} + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} u,$$

with boundary conditions $\boldsymbol{x}(0) = \boldsymbol{x}_0 = (x_1, x_2, x_3)^\top$, $\boldsymbol{x}(T) = \mathbf{0}$. The polynomial system (3.2.4) features 3 equations for the unknown switching sequence $\boldsymbol{t} = (t_1, t_2, t_3)$. We sample state and controls along the optimal trajectories departing from $N_s = 5000$ uniformly distributed initial conditions in $[-1, 1]^3$ using deflation with an initial guess of $(t_1, t_2, t_3) = (0.1, 0.1, 0.1)$, a tolerance of 1×10^{-15}, and a maximum number of iterations of 1×10^3 per initial guess. When the deflated Newton's method converged, it required an average of 54.2912 iterations to converge. The average number of iterations required for convergence after each successive deflation can be seen in Appendix A. The resulting dataset is used to train a model u_θ characterised by a single hidden layer with width $n_2 = 80$ and $\sigma_2 = \tanh(\cdot)$ activation function. After training, the model reaches 99.12% accuracy in the test set, associated with a loss $\mathcal{L} = 0.0334$. In this example, the enhanced feedback approximation, calling the polynomial solver for identifying the control at points with confidence below $\varepsilon = 0.005$, occurs on 4.08% of the trajectory points, and improves the approximated horizon when compared to the trajectory controlled solely via u_θ. We show the results in Figure 3.4, where the NN-controlled trajectory is computed via a forward Euler scheme with time-step size $\Delta t = 10^{-3}$, whilst the optimal trajectory is computed through exact integration.

We motivated the introduction of the feedback NN approximation as way to achieve robustness against noise. To evaluate this, we introduce white noise into the last component of the system. Specifically, at every discrete integration time t we add $\big(0, 0, \eta(t)\big)$, where $\eta(t) \sim \mathcal{N}(0, \sigma^2)$, with $\sigma^2 = 0.02$. Starting from the same initial condition, we perform a Monte Carlo simulation for both the open-loop and the approximated feedback control approaches. Figure 3.5 illustrates the mean and variance obtained from 1000 noisy controlled trajectories. As expected, while the open-loop solution diverges on average from the target final state, the approximated feedback law successfully drives the system towards the origin.

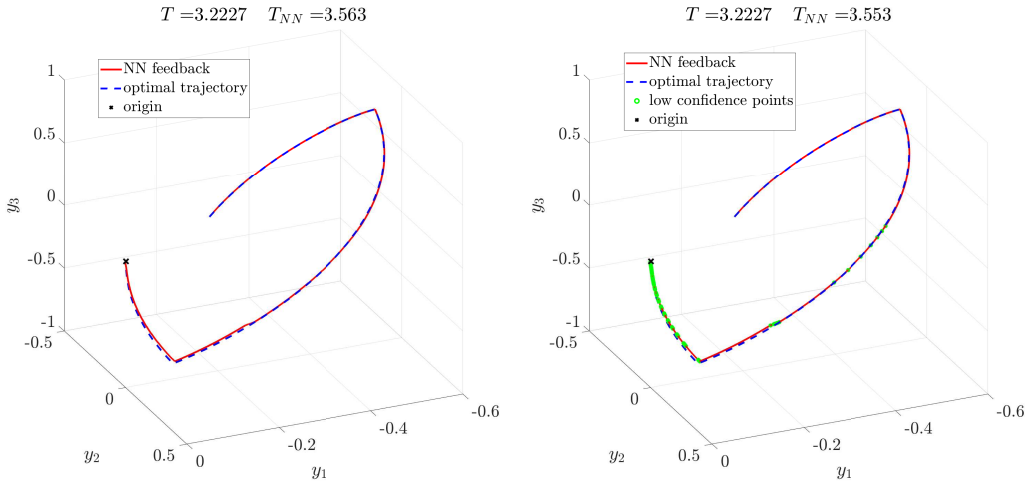

Figure 3.4: Comparison of the optimal trajectory with the u_θ-controlled (left) and with the confidence-enhanced approximation (right). Relying on the polynomial solver for low-confidence points improves the approximation, as the time T_{NN} needed to reach the origin decreases.

3.7.3 4th-order tests

We consider the time-optimal control problem subject to the fourth-order integrator for state variable $\boldsymbol{x}(t)$ and boundary conditions $\boldsymbol{x}(0) = \boldsymbol{x}_0 = (x_1, \ldots, x_4)^\top$, $\boldsymbol{x}(T) = \mathbf{0}$. In this higher-dimensional example, we examine the upper bound on the number of solutions of the associated system (3.2.4) for the switching sequence $t_1, \ldots, t_4$. The theoretical bound on the total number of real and complex solutions is $4! = 24$. However, as discussed in Section 3.4, the use of Hermite quadratic forms allows us to refine this bound by identifying the number of real solutions more precisely.

We sample initial conditions $\boldsymbol{x}^{(i)} \in \big([-1,1] \cap \mathbb{Q}\big)^4$, for $i = 1, \ldots, N_s = 10^4$, and for each $\boldsymbol{x}^{(i)}$ we compute the number of real roots of the polynomial system associated with $u_0 = 1$ and $u_0 = -1$, denoted by I^+ and I^- respectively. The number of real roots of I^+ and I^- is computed by finding the signatures of the Hermite quadratic forms, denoted by $\mathcal{H}(I^+)$ and $\mathcal{H}(I^-)$, respectively. Note that the transition from $\mathbb{R}$ to $\mathbb{Q}$, which we do as computation of the Hermite quadratic form over the reals is computationally unreliable, if not impossible. This preliminary step reduces the computational cost of data generation, as the bound for the deflation algorithm 2 is tightened relative to the theoretical one. The range of real roots given by the Hermite quadratic form varies from 0 to 8, depending on the initial condition. For $\boldsymbol{x}^{(i)}$'s with 0 real solutions for I^+ or I^-, this directly determines the sign of the initial control

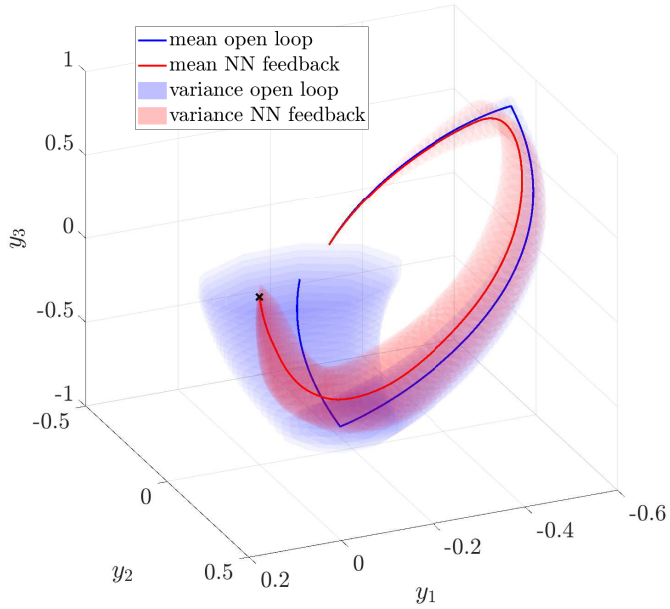

Figure 3.5: Mean and variance of a Monte Carlo simulation with 1000 trajectories perturbed by Gaussian noise. The robustness of the approximated feedback law ensures that the controlled system reaches the target destination, while the open-loop solution, obtained as the solution of the polynomial system (3.2.4), diverges from it.

u_0. We then use deflation to determine the roots of I^+ and I^- with with an initial guess of $(t_1, t_2, t_3, t_4) = (1, 1, 1, 1)$, a tolerance of 1×10^{-5}, and a maximum number of iterations of 1×10^4 per initial guess, and with a bound on the number of roots given by the traces of the Hermite quadratic forms of I^+ and I^-. These new bounds reduce the CPU time for data generation from 107 to 49 minutes when compared to the theoretical bound. When the deflated Newton's method converged, it required an average of 879.9081 iterations to converge. The average number of iterations required for convergence after each successive deflation can be seen in Appendix A.

We train a model u_θ characterised by 2 hidden layers, with width $n_2 = n_3 = 100$ and $\sigma_2 = \sigma_3 = \tanh(\cdot)$ activation functions. After training, we evaluate the model in the test set, obtaining the accuracy of 96.76% and the loss $\mathcal{L} = 0.0882$. In Figure 3.6, we analyze the controlled trajectories comparing the optimal solution against its approximation along the dimensions x_2, x_3, x_4. We consider a low-confidence threshold $\varepsilon = 0.005$, under which we rely on the deflation routine to determine the control. Low-confidence classification triggers in 1.3% of the trajectory. Even more

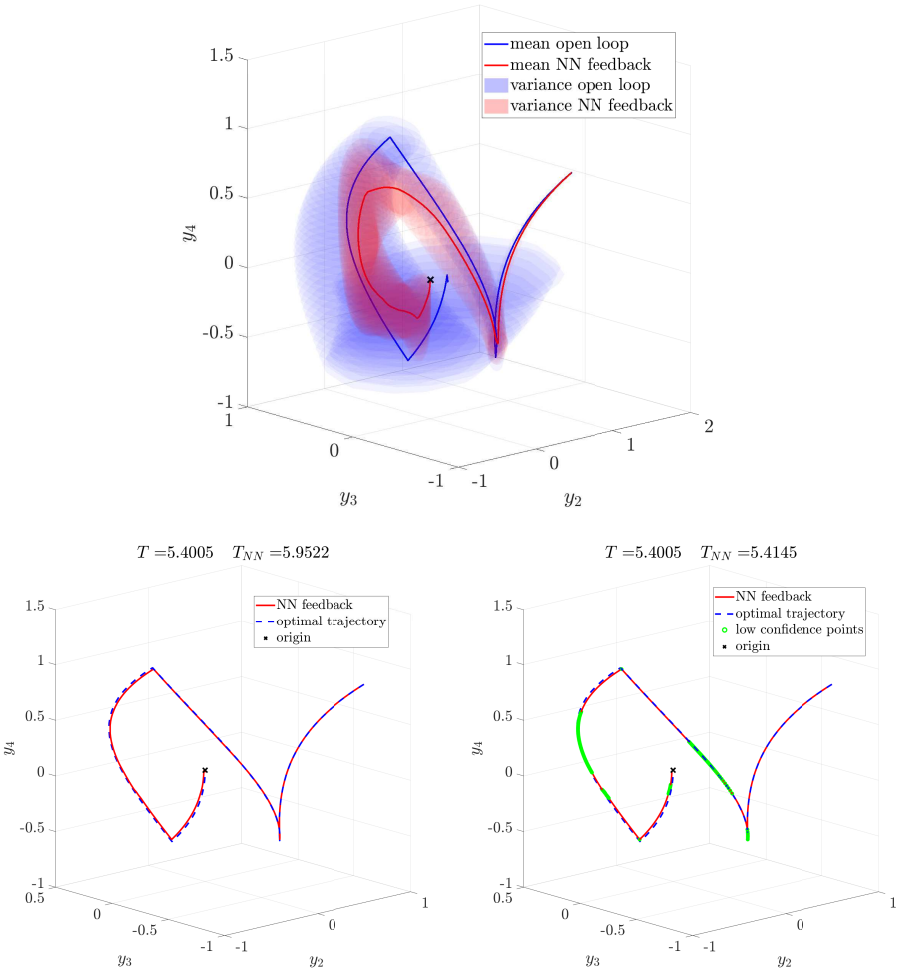

Figure 3.6: Monte Carlo simulation of 1000 controlled noisy trajectories via open loop control signal vs. approximated feedback control (top). Comparison of the optimal trajectory with the one controlled via the approximated feedback law (left). Calling the polynomial solver for the low-confidence points improves the approximation performance in terms of time horizon T_{NN} (right).

prominently than in the lower order examples, the enhanced approximate feedback improves the performance of the control law. As in the previous example, we also test the robustness of the feedback approximation by injecting additive noise of the form $\big(0, 0, 0, \eta(t)\big)$, with $\eta(t) \sim \mathcal{N}(0, \sigma^2)$, where $\sigma = 2^{-2}$. A Monte Carlo simulation of 1000 controlled noisy trajectories shows how the approximated feedback control successfully steers the system towards the origin, while the open-loop solution diverges.

3.7.4 5th-order integrator

Finally, we consider the 5th order integrator for state $\boldsymbol{x}(t)$ with boundary conditions $\boldsymbol{x}(0) = \boldsymbol{x}_0 = (x_1, \dots, x_5)^\top$ and $\boldsymbol{x}(T) = \mathbf{0}$. The theoretical upper bound for the number of solutions $(t_1, \dots, t_5)$ of the associated polynomial system is $5! = 120$. However, excessively deflating a polynomial system can lead to instability. Heuristically, as the number of identified and deflated roots increases, the root-finding procedure may diverge, failing to locate additional existing solutions. This pathological behaviour naturally worsens as the upper bound increases. While still feasible, the CPU time associated to the computation of the Hermite quadratic form can be used instead to sample different initial conditions with the un-informed deflation algorithm.

We sample $N_s = 50,000$ points $\boldsymbol{x}_0^{(i)} \in [-1, 1]^5$, for $i = 1, \dots, N_s$, and include in the dataset $\mathcal{T}$ only the optimal trajectories corresponding to initial conditions for which the deflation algorithm successfully identifies an admissible solution with an initial guess of $(1, 1, 1, 1, 1)$, a tolerance of 1×10^{-8}, and a maximum number of iterations of 1×10^4 per initial guess. When the deflated Newton's method converged, it required an average of 7465.5006 iterations to converge. The average number of iterations required for convergence after each successive deflation can be seen in Appendix A. This selection process excludes approximately 20% of the initial conditions. We consider a model u_θ with 2 hidden layers having width $n_2 = n_3 = 80$ and $\sigma_2 = \sigma_3 = \tanh(\cdot)$ activation functions. The trained classifier achieves accuracy 99.61% in the test set, associated with loss $\mathcal{L} = 0.0141$. In Figure 3.7, we test the performance of the NN feedback controller against the optimal trajectory. The performance of the approximation is significantly improved by selectively intervening with the polynomial system solver whenever the classifier confidence drops below $\varepsilon = 0.1$. This occurs only at 5 low-confidence points, all located at the intersection of switching surfaces.

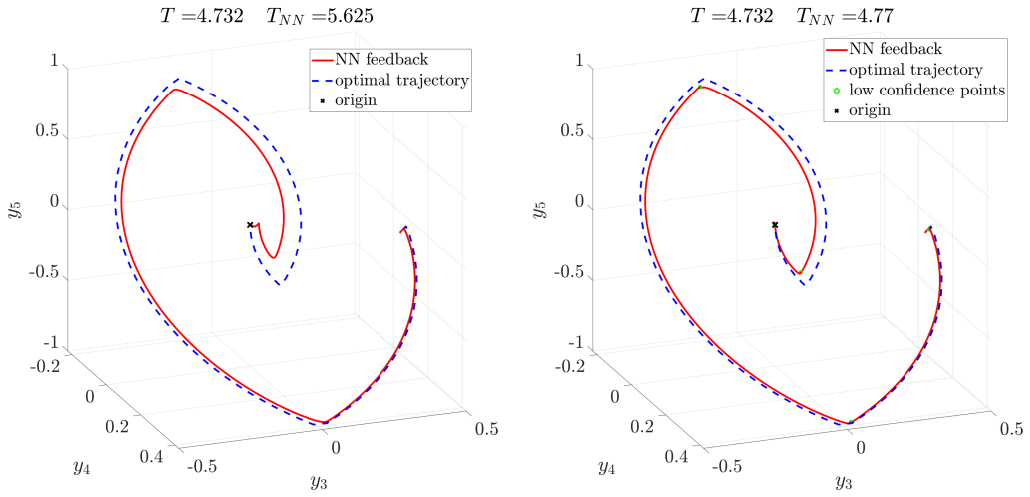

Figure 3.7: Comparison of the optimal trajectory with the trajectory controlled using the approximated feedback law (left). Leveraging the polynomial solver for low-confidence points enhances the approximation's performance in terms of the time horizon T_{NN} (right).

3.8 Concluding remarks

In this chapter, we have revisited a classical time-optimal control problem from a perspective combining polynomial systems, numerical analysis, and statistical machine learning. Our original motivation was to explore the scalability of the approach originally developed in [82], where the use of Gröbner bases and techniques from real algebraic geometry were proposed to solve polynomial systems arising in time-optimal control. In this respect, we have reached a negative result: to the best of our experience, the computational cost and complexity of such tools become prohibitively expensive as the dimension of the state space increases beyond 3, rendering them inapplicable for constructing real-time feedback controllers. Hence, we have resorted to the numerical solution of polynomial systems using Newton's method on a deflated variant, which can effectively exhaust all possible roots. Here, the Hermite quadratic form is proposed as a tool to establish a bound on the number of roots of the polynomial system, increasing the efficiency of the deflation algorithm. Overall, this constitutes a consistent computational workflow for the solution of time-optimal trajectories for nilpotent linear systems.

In the second part of the chapter, we adopt a statistical machine learning approach to interpret the time-optimal feedback controller as a binary classifier over the state

space. This allows us to construct a control law via supervised learning, yielding a bang-bang control with a confidence indicator. This is particularly relevant to develop a robust, chattering-free control in the vicinity of the switching surface. Our numerical tests show that, enhancing the feedback law with an open-loop solver whenever the confidence of the classifier is low, leads to a fair trade-off between accuracy of the approximate optimal trajectory, robustness, and computational cost.

This work suggests different avenues for future research. The use of the Hermite quadratic form to determine the number of roots of time-optimal polynomial systems is promising, and merits a deeper consideration. We suggest a purely algebraic method for feedback control of nilpotent systems using Hermite quadratic forms for sign determination. The proposed method is, while possible, currently unfeasible as a real-time method due to the complexity involved in computing the Hermite quadratic form on-line. We do, however, acknowledge that the work of Phuoc Le and Safey El Din in [64] is a way to precompute the Hermite quadratic form by parametrising the polynomial system associated to a nilpotent linear time-optimal control problem. We propose this as a further avenue of research that may lead to a purely algebraic feedback control for nilpotent linear systems, as well as for other linear time-optimal control problems.

The interpolation of the bang-bang feedback law as a binary classifier enables the application of machine learning methods for a class of control problems that remains elusive to other type of global approximation techniques due to the non-smoothness of the optimal control field. Finally, the proposed methodology can be applied to other time-optimal and sparse control problems with bang-bang or bang-zero-bang structures.

Chapter 4

Algebraic Splines with Boundary Conditions

Another application of algebraic techniques is given here, where we consider splines with boundary conditions. A spline is a piecewise function defined on a partition of $\mathbb{R}^n$ such that on each part of the partition, the function is given by a polynomial, and the function is r times continuously differentiable everywhere for some $r \geq 0$. Splines are a ubiquitous tool in numerical analysis as finite elements, as well as in computer-aided design, and data fitting, see for instance [26, 73].

Splines with boundary conditions are spline functions defined on a real domain that satisfy smoothness conditions not only across the interior faces of the partition but also along the boundary. Their use in the finite element method for solving partial differential equations can be found, for instance, in [77]. For their use in this and many other contexts, often a basis is needed for the space of splines with boundary conditions of a given degree. In this chapter, we demonstrate how algebraic methods can be used to complete a slightly more basic task that aids construction of a basis, which is computing the dimension of the space of splines with boundary conditions of a given degree defined over a fixed partition of $\mathbb{R}^2$. This is done for splines without boundary conditions in [32].

In this chapter we focus on splines with boundary conditions over simplicial complexes embedded in $\mathbb{R}^2$. The main result of this chapter is Theorem 4.4.4, which computes the dimension of the space of splines with boundary conditions with sufficiently large degrees. The proof makes use of powerful homological algebra techniques developed by Billera, Schenck, and Stillman in [11, 67, 68], generalised to account

for boundary conditions, as well as results by McDonald in [57].

This chapter is organised as follows. In Section 4.1 we set up the notation of simplicial complexes. In Section 4.2 we formally define splines with boundary conditions over simplicial complexes embedded in $\mathbb{R}^2$ and reformulate a result of Billera [11] to account for boundary conditions. We then use it to demonstrate the construction of splines with boundary conditions over some example simplicial complexes. We also recall the definition of the cone of a simplicial complex and homogeneous splines, modifying a proof of Billera and Rose [12] to connect splines and homogeneous splines with boundary conditions. In Section 4.3 we recall some basic notions from algebraic topology and use them to construct a chain complex that writes the dimension of the space of splines with boundary conditions as a homology group like the chain complex constructed in [11, 67, 68]. In Section 4.4 we prove Theorem 4.4.4. Section 4.5 is then devoted to illustrating the use of the formula that we proved in the prior section in some examples. In Section 4.6 we prove Proposition 4.6.1 that connects the dimension of the space of splines with partial boundary conditions on a simplicial complex with the splines defined on its subcomplexes, and Section 4.7 contains concluding remarks.

4.1 Simplicial Complexes

A simplicial complex Δ is a set of simplices satisfying two properties:

- Every face of a simplex in Δ is a simplex in Δ.
- If the intersection of two simplices in Δ is non-empty, then the intersection is a face of both simplices.

An exposition to simplicial complexes can be found in many textbooks, including [46]. We consider simplicial complexes embedded in $\mathbb{R}^2$ so that the maximal dimension simplex in any simplicial complex is 2. Given a simplicial complex Δ, we define the sets $\Delta_0, \Delta_1, \Delta_2$ as the set of 0, 1, and 2 dimensional simplices in Δ respectively. The set $\partial\Delta$ is the subset of all edges and vertices that are contained within the boundary of the embedding of Δ. A simplex is called maximal if it is not part of the boundary of any higher dimensional simplex. Let Δ be a simplicial complex embedded in $\mathbb{R}^2$. We say Δ is *pure* if every maximal simplex is of dimension 2. The simplicial complex is *hereditary* if for every two maximal simplices $\sigma, \sigma' \in \Delta_2$ intersecting at at a vertex

$\gamma = \sigma \cap \sigma'$, there exists a sequence $\sigma = \sigma_1, \ldots, \sigma_m = \sigma' \in \Delta_2$ of maximal faces satisfying that $\gamma \in \sigma_i$ for all $i = 1, \ldots, m-1$, and $\sigma_i \cap \sigma_{i+1} \in \Delta_1$ for $i = 1, \ldots, m-1$ [85]. See Figure 4.1 for an example of a simplicial complex that is not pure, a region of $\mathbb{R}^2$ that is not a simplicial complex, and a simplicial complex that is not hereditary. For simplicity, we define the following notation:

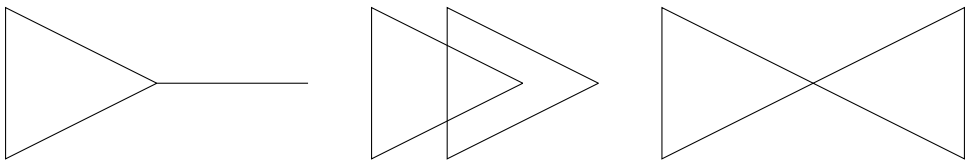

Figure 4.1: (Left) A simplicial complex that is not pure, it has a 1-dimensional maximal simplex. (Center) Not a simplicial complex. The intersection of the two maximal simplices is neither empty nor a face of either simplex. (Right) A simplicial complex that is not hereditary, the intersection of the two maximal simplices is a 0-simplex and there is no way to connect the two 2-simplices with other 2-simplices containing the 0-simplex.

Definition 4.1.1. We refer to the set of boundary i-simplices as the set $\Delta_i \cap \partial\Delta$ for $i = 0, 1$. Since a 2-simplex cannot be on the boundary, we define the set of boundary 2-simplices is the set $\Delta_2^\partial = \{\sigma \in \Delta_2 \colon \sigma \text{ contains a boundary edge}\}$. Additionally, for $0 \leq i \leq 2$, we denote the set of interior i-faces to be $\Delta_i^\circ = \Delta_i \setminus \Delta_i^\partial$.

For a demonstration of these sets, see Figure 4.2 for an example simplicial complex embedded in $\mathbb{R}^2$ that is pure and hereditary. We can define the 1-simplices and 2-simplices of a simplicial complex by writing them in terms of the vertices (0-simplices) that constitute the endpoints or corners respectively. This notation for the simplices in a simplicial complex is used in [46], but has been used elsewhere, and we introduce it here for clarity.

Definition 4.1.2. Let Δ be a simplicial complex embedded in $\mathbb{R}^2$ which we associate with its embedding. Label the 0-simplices (vertices) of Δ as $\gamma_1, \ldots, \gamma_n$ where $n = |\Delta_0|$, and suppose that these vertices are embedded at points $v_1, \ldots, v_n \in \mathbb{R}^2$ respectively. A 2-simplex σ in $\mathbb{R}^2$ with vertices $\gamma_i, \gamma_j, \gamma_k$ with $i < j < k$ is then writen $\sigma = [v_i, v_j, v_k]$. If we embed a 1-simplex τ in $\mathbb{R}^2$ with vertices γ_i, γ_j for $i < j$, then we write $\tau = [v_i, v_j]$, and of course a 0 simplex γ_i is represented by $\gamma_i = [v_i]$.

Notice that by definition this imposes some sort of ordering on the vertices. This order suggests an orientation of the edges. We know that the boundary of a 2-simplex

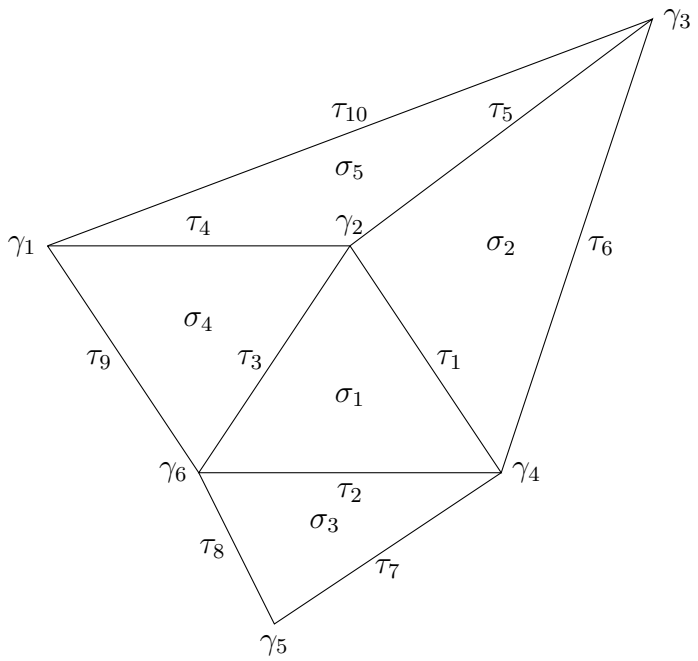

Figure 4.2: A simplicial complex Δ embedded in $\mathbb{R}^2$. We have $\Delta_0^\partial = \{\gamma_1, \gamma_3, \gamma_4, \gamma_5, \gamma_6\}$, $\Delta_0^\circ = \{\gamma_2\}$, $\Delta_1^\partial = \{\tau_6, \tau_7, \tau_8, \tau_9, \tau_{10}\}$, $\Delta_1^\circ = \{\tau_1, \tau_2, \tau_3, \tau_4, \tau_5\}$, $\Delta_2^\partial = \{\sigma_2, \sigma_3, \sigma_4, \sigma_5\}$ and $\Delta_2^\circ = \{\sigma_1\}$

σ, denoted $\partial_2(\sigma)$, is made up of a collection of 1-simplices, and so a natural way to express this boundary is as a formal sum of the edges. The orientation of the edges therefore can be used to impose signs on this formal sum. The orientation of the edges should be chosen so that if two 2-simplices σ and σ' share an edge τ on their boundary, then the sign associated to τ in the boundary of σ should be opposite to the sign associated to τ in σ'. In this way we do not see τ on the boundary of $\sigma + \sigma'$ (represented by the region $\sigma \cup \sigma'$ of $\mathbb{R}^2$), as we would expect. In terms of defining the boundary of a 1-simplex, we see that it must be a signed formal sum of 0-simplices, with the signs determined by the orientation, and by convention, we define the boundary of a 0-simplex to be 0.

We define here formally the boundary of σ in the way described by Billera in [11]: Order all vertices of Δ as $\gamma_1, \gamma_2, \ldots, \gamma_n$ where $n = |\Delta_0|$ and γ_i is embedded at $v_i \in \mathbb{R}^2$. Then, if $\sigma = [v_{i_0}, v_{i_1}, v_{i_2}]$, $i_0 < i_1 < i_2$, and $\tau = \sigma \setminus \{v_{i_j}\}$, then the coefficient of τ in $\partial_2(\sigma)$ is $(-1)^j \text{Sign}\left(\det(v_{i_1} - v_{i_0}, v_{i_2} - v_{i_0})\right)$, associating the vertices of Δ with the vectors in $\mathbb{R}^2$ that they are mapped to in the embedding of Δ. This defines an orientation on Δ that is unique up to sign. We choose one orientation, which can be represented by putting a direction on each edge, and then determine the boundary

of an edge $\tau = [v_{i_0}, v_{i_1}]$, denoted $\partial_1(\tau)$, as follows. The sign of v_{i_j} in τ is 1 if the direction on τ points towards v_{i_j}, and -1 otherwise.

Example 4.1.1. Consider the simplicial complex shown in Figure 4.3. Call this simplicial complex Δ, and order the vertices of Δ as shown in the diagram. We have that $\sigma = [v_0, v_1, v_2]$, and $\tau_0 = [v_1, v_2]$, $\tau_1 = [v_0, v_2]$, $\tau_2 = [v_0, v_1]$. The coefficients of τ_0, τ_1, τ_2 in $\partial_2(\sigma)$ are:

$$(-1)^0 \text{Sign} \det \begin{pmatrix} 1 & -1 \\ -1 & -1 \end{pmatrix} = -1,$$
$$(-1)^1 \text{Sign} \det \begin{pmatrix} 1 & -1 \\ -1 & -1 \end{pmatrix} = 1,$$
$$(-1)^2 \text{Sign} \det \begin{pmatrix} 1 & -1 \\ -1 & -1 \end{pmatrix} = -1,$$

respectively. So, $\partial_2(\sigma) = -\tau_0 + \tau_1 - \tau_2$. We choose directions on the edges of σ to point from vertices with lower indices to higher indices, so that

$$\partial_1(\tau_0) = \gamma_2 - \gamma_1,$$
$$\partial_1(\tau_1) = \gamma_2 - \gamma_0,$$
$$\partial_1(\tau_2) = \gamma_1 - \gamma_0.$$

◇

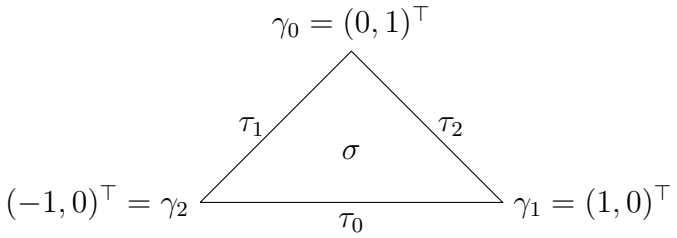

Figure 4.3: Simplicial complex Δ for Example 4.1.1.

4.2 Splines with Boundary Conditions on Simplicial Complexes

Suppose Δ is a pure and hereditary simplicial complex embedded in $\mathbb{R}^2$. Let $r \geq 0$ be an integer, we take $S^r(\Delta)$ to be the set of piecewise polynomial functions on Δ whose

first r derivatives are continuous on Δ and vanish up to order r on the boundary $\partial\Delta$ of Δ. More precisely,

$$S^r(\Delta, \partial\Delta) = \left\{ f \in C^r(\Delta) \colon \begin{array}{c} f|_\sigma \in \mathbb{R}[x,y] \text{ for every face } \sigma \in \Delta_2 \\ f \text{ vanishes to order } r \text{ on every boundary edge } \tau \in \Delta_1^\partial \end{array} \right\},$$

where $C^r(\Delta)$ is the set of functions $f \colon \Delta \to \mathbb{R}$ that are continuously r times differentiable. An element of $S^r(\Delta, \partial\Delta)$ is called a *spline with boundary conditions*. Throughout this thesis we will occasionally refer to elements of $S^r(\Delta, \partial\Delta)$ as simply splines, where it is assumed implicitly that they satisfy the boundary conditions. Where we mean splines that do not satisfy the boundary conditions, we refer to it as a spline without boundary conditions.

Given an integer $d \geq 0$, we define the set

$$S^r_d(\Delta, \partial\Delta) = \{ f \in S^r(\Delta, \partial\Delta) \colon \deg(f|_\sigma) \leq d \text{ for every face } \sigma \in \Delta_2 \},$$

to be the set of splines with boundary conditions whose restriction to each maximal face is a polynomial of degree at most d. If f is a spline in $S^r(\Delta, \partial\Delta)$ and the largest degree of any $f|_\sigma$ for a maximal face σ is d, then we say f has degree d. Notice that $S^r(\Delta, \partial\Delta) = \bigcup_{d \geq 0} S^r_d(\Delta, \partial\Delta)$ and, in fact, we have the following structural result, a proof of which for splines without boundary conditions can be found in [54].

Proposition 4.2.1. *Let $d \geq 0$ be given. The space $S^r_d(\Delta, \partial\Delta)$ is a real vector space.*

Proof. Let $f, g \in S^r_d(\Delta, \partial\Delta)$ are splines with boundary conditions, and $c \in \mathbb{R}$ is a scalar. We first show that $(cf + g)|_\sigma \in \mathbb{R}[x,y]$ for each maximal face $\sigma \in \Delta_2$. Then, we show that $\deg((cf+g)|_\sigma) \leq d$. For a maximal face $\sigma \in \Delta_2$, we have

$$(cf + g)|_\sigma = cf|_\sigma + g|_\sigma \in \mathbb{R}[x,y].$$

Then,

$$\deg((cf+g)|_\sigma) = \deg(cf|_\sigma + g|_\sigma) \leq \max\{\deg(cf|_\sigma, g|_\sigma)\} = d.$$

Let $0 \leq k \leq r$, and let $a + b = k$. For a boundary edge $\tau \in \Delta_1 \cap \partial\Delta$, we have

$$\left(\frac{\partial^k}{\partial x^a \partial y^b}(cf+g) \right) |_\tau = c \left(\frac{\partial^k}{\partial x^a \partial y^b} f \right) |_\tau + \left(\frac{\partial^k}{\partial x^a \partial y^b} g \right) |_\tau = 0,$$

by the linearity of the derivative, and that f and g are splines with boundary conditions. Finally, we must show that $cf + g \in C^r(\Delta)$. We already know that on every maximal simplex σ, $cf + g \in \mathbb{R}[x,y]$, and all polynomials are infinitely continuously

differentiable, so we simply must show that $cf + g$ is r times continuously differentiable on the boundaries between maximal simplices. Let σ and σ' be maximal simplices meeting at an edge $\tau \in \Delta_1$. We know that

$$(cf + g)|_\tau = cf|_\tau + g|_\tau,$$

and so $cf + g$ is r-times continuously differentiable on the edges of Δ by the linearity of the derivative. □

Each $\tau \in \Delta_1$ is embedded in $\mathbb{R}^2$ on a line, that is there is a linear polynomial vanishing on τ that is unique up to a scalar multiple. For $\tau \in \Delta_1$, let $\ell_\tau \in \mathbb{R}[x, y]$ be a choice of linear polynomial vanishing on τ. As of now, we have an analytical definition of splines with boundary conditions, where the smoothness conditions are expressed as conditions on the derivatives of the polynomials defining the spline. The following result is an algebraic criterion that gives an equivalent definition, expressing the smoothness conditions algebraically. A proof of this for splines without boundary conditions can be found in [12, Proposition 1.2 & Corollary 1.3], the idea for the proof of which can be found in [21], and yet another proof can be found in [11, Lemma 2.2]. For completeness, we give a full proof here for splines with boundary conditions. We will use the fact that if a polynomial f vanishes on τ, then f is divisible as a polynomial by ℓ_τ, a proof of which can be found in [11, Lemma 2.1].

Proposition 4.2.2. *A function $f\colon \Delta \to \mathbb{R}$ on Δ is in $S^r(\Delta, \partial\Delta)$ if and only if*

- *The restriction of f to any maximal face $\sigma \in \Delta_2$ is a polynomial, $f|_\sigma \in \mathbb{R}[x, y]$.*
- *If a pair of maximal faces $\sigma, \sigma' \in \Delta_2$ share an edge $\sigma \cap \sigma' = \tau \in \Delta_1$, then $f|_\sigma - f|_{\sigma'} \in \langle \ell_\tau^{r+1} \rangle$.*
- *For each boundary face $\sigma \in \Delta_2^\partial$ with boundary edges $\tau_1, \ldots, \tau_m$, we have $f|_\sigma \in \bigcap_{i=1}^m \langle \ell_{\tau_i}^{r+1} \rangle$. Here, of course, m must be 1, 2 or 3 as a boundary face σ is a 2-simplex and so has at most 3 edges on the boundary.*

Proof. Let $f \in S^r(\Delta, \partial\Delta)$ be a spline on Δ with boundary conditions. By definition, the restriction of f to any maximal face of Δ_2 is polynomial. Given a pair of maximal faces σ and σ' sharing an edge τ, we know that

$$\left.\frac{\partial^r f|_\sigma}{\partial x^a \partial y^b}\right|_\tau = \left.\frac{\partial^r f|_{\sigma'}}{\partial x^a \partial y^b}\right|_\tau,$$

for all integers a, b with $a + b = r$. We write $\ell_\tau = \alpha x + \beta y + \gamma$ with $\alpha, \beta, \gamma \in \mathbb{R}$ to be a non-zero linear polynomial vanishing on τ, so at least one of α or β is non-zero. We know that $(f|_\sigma - f|_{\sigma'})|_\tau = 0$, so $f|_\sigma - f|_{\sigma'} = g_0\ell_\tau$ for some $g_0 \in \mathbb{R}[x, y]$, wherein $f|_\sigma - f|_{\sigma'} \in \langle \ell_\tau \rangle$. Then, notice that

$$\frac{\partial f|_\sigma}{\partial x} - \frac{\partial f|_{\sigma'}}{\partial x} = \frac{\partial g_0}{\partial x}\ell_\tau + g_0\frac{\partial \ell_\tau}{\partial x} = \frac{\partial g_0}{\partial x}\ell_\tau + \alpha g_0 \in \langle \ell_\tau \rangle,$$
$$\frac{\partial f|_\sigma}{\partial y} - \frac{\partial f|_{\sigma'}}{\partial y} = \frac{\partial g_0}{\partial y}\ell_\tau + g_0\frac{\partial \ell_\tau}{\partial y} = \frac{\partial g_0}{\partial y}\ell_\tau + \beta g_0 \in \langle \ell_\tau \rangle,$$

We get that $\alpha g_0, \beta g_0 \in \langle \ell_\tau \rangle$, and since at least one of α and β is non-zero in $\mathbb{R}$, we get that $g_0 \in \langle \ell_\tau \rangle$. Notice that for integers a, b with $a + b = k$ for some $1 \leq k \leq s \leq r$, then

$$\frac{\partial^k}{\partial x^a \partial y^b}\ell_\tau^s = k!\alpha^a\beta^b\ell_\tau^{s-k}.$$

We can obtain this using the product rule and induction on k. If $f|_\sigma - f|_{\sigma'} = g_s\ell_\tau^s$ for $1 \leq s \leq r$ where $g_k \in \mathbb{R}[x, y]$, then

$$\frac{\partial^s}{\partial x^a \partial y^b}(f|_\sigma - f|_{\sigma'}) = \frac{\partial^s}{\partial x^a \partial y^b}(g_s\ell_\tau^s) = h\ell_\tau + g_s s!\alpha^a\beta^b \in \langle \ell_\tau \rangle.$$

We may choose a and b to be so that $s!\alpha^a\beta^b$ is non-zero, and thus giving $g_s \in \langle \ell_\tau \rangle$. Thus, $f|_\sigma - f|_{\sigma'} \in \langle \ell_\tau^{r+1} \rangle$. A similar proof shows that if τ is a boundary edge and σ is a 2-simplex with τ in its boundary, then $f|_{\sigma'} \in \langle \ell_\tau^{r+1} \rangle$, and the third condition follows by considering all boundary edges of σ. □

Remark 4.2.1. For each edge τ, the choice of polynomial ℓ_τ is unique up to real multiples. We know that if there is some other linear polynomial ℓ'_τ that vanishes on the edge τ, then by [11, Lemma 2.1], the polynomial ℓ'_τ divides ℓ_τ and vice versa. So there are polynomials g and g' for which $\ell_\tau = g'\ell'_\tau$ and $\ell'_\tau = g\ell_\tau$. Thus, $\ell_\tau = gg'\ell_\tau$, so $gg' = 1$. The product of two polynomials is 1 if and only if they are inverses in $\mathbb{R}[x, y]$, which corresponds only to if g and g' are real numbers, so the choice of polynomial ℓ_τ is unique up to real multiples.

We refer to Proposition 4.2.2 as the algebraic spline criterion. Since Δ is hereditary, by Proposition 4.2.2, we can view $S^r(\Delta, \partial\Delta)$ as the kernel of a matrix [12]. Let $f \in S^r(\Delta, \partial\Delta)$ be a spline on Δ. Let $\Delta_2 = \{\sigma_1, \ldots, \sigma_t\}$ and $\Delta_1 = \{\tau_1, \ldots, \tau_s\}$. If two 2-faces σ_i and σ_j meet at an edge τ_k, then by the algebraic spline criterion, we have $f|_{\sigma_i} - f|_{\sigma_j} \in \langle \ell_{\tau_k}^{r+1} \rangle$ where ℓ_{τ_k} is a choice of linear form vanishing on τ_k. This can be written as $f|_{\sigma_i} - f|_{\sigma_j} = g_{\tau_k}\ell_{\tau_k}^{r+1}$ where $g_{\tau_k} \in \mathbb{R}[x, y]$ is a polynomial. If τ_k is an exterior edge of a 2-simplex σ_i, then the relationship is $f|_{\sigma_i} = g_{\tau_k}\ell_{\tau_k}^{r+1}$ where $g_{\tau_k} \in \mathbb{R}[x, y]$ is

a polynomial. So, for each edge $\tau \in \Delta_1$, there is a polynomial relation that is linear in the variables $f|_{\sigma_1}, \ldots, f|_{\sigma_t}, g|_{\tau_1}, \ldots, g|_{\tau_s}$. Let $\boldsymbol{f} = (f|_{\sigma_1}, \ldots, f|_{\sigma_t}, g|_{\tau_1}, \ldots, g|_{\tau_s})^\top$, then these linear relations can be summarised in a matrix equation. In [12], it is shown that this matrix equation has the form

$$\left(\begin{array}{c|c} M & L^{r+1} \end{array} \right) \boldsymbol{f} = \boldsymbol{0},$$

where M is an $s \times t$ matrix defined by $(M)_{i,j}$ taking the value of the coefficient of τ_i in the boundary of σ_j, and L is the $s \times s$ diagonal matrix with $(L)_{i,i} = \ell_{\tau_i}$ for $i = 1, \ldots, s$. We want to determine $f|_{\sigma_1}, \ldots, f|_{\sigma_t}$, so the first t entries of a solution to this matrix equation corresponds to a spline on Δ. It is therefore clear that splines are the first t entries of elements in the kernel of $\left(\begin{array}{c|c} M & L^{r+1} \end{array} \right)$. We write $A(\Delta, \partial\Delta, r)$ to denote this matrix, determined by the simplicial complex Δ and the smoothness r. Finding the kernel of $A(\Delta, \partial\Delta, r)$ is equivalent to finding the syzygies of the columns, and we demonstrate how this can be used to find splines defined on some simple simplicial complexes.

Example 4.2.1. Consider the following simplicial complex Δ shown in Figure 4.4. Notice that without any boundary conditions, then the ring of splines is simply

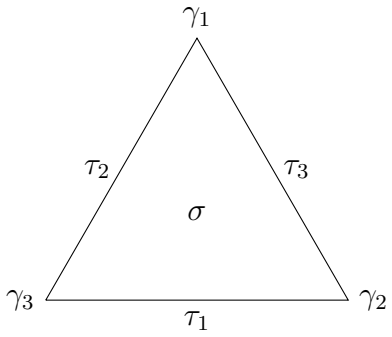

Figure 4.4: Simplicial complex Δ for Example 4.2.1

equal to the the ring of polynomials, because there is only one 2-simplex with no edge conditions. To find the splines with boundary conditions, we need to find which polynomials vanish on every edge, wherein these polynomials will account for every spline. For each edge $1 \leq i \leq 3$, we write ℓ_{τ_i} to be a linear polynomial in $\mathbb{R}[x, y]$ vanishing on τ_i. If we require smoothness at the edges τ_1, τ_2, τ_3, then we need a spline to be a polynomial whose derivative vanishes up to order r on each edge. By the algebraic spline criterion (Proposition 4.2.2, this is equivalent to requiring the polynomial to vanish with multiplicity $r+1$ on each τ_i. Combining these conditions,

we can see that a spline on Δ is a polynomial that is a multiple of $\ell_{\tau_1}^{r+1}$, $\ell_{\tau_2}^{r+1}$, and $\ell_{\tau_3}^{r+1}$ simultaneously. We can confirm this by computing the matrix

$$A(\Delta, \partial\Delta, r) = \left(\begin{array}{c|ccc} 1 & \ell_{\tau_1}^{r+1} & 0 & 0 \\ 1 & 0 & \ell_{\tau_2}^{r+1} & 0 \\ 1 & 0 & 0 & \ell_{\tau_3}^{r+1} \end{array}\right).$$

A spline, therefore, can be represented by a solution of the matrix equation

$$A(\Delta, \partial\Delta, r) \begin{pmatrix} f|_\sigma \\ g_1 \\ g_2 \\ g_3 \end{pmatrix} = \mathbf{0},$$

recalling here that $f|_\sigma$ refers to the restriction of a spline f to the simplex σ. As such, splines are represented by elements of the kernel of the matrix $A(\Delta, \partial\Delta, r)$. We can compute this kernel, for example using `Macaulay2`, and we find that the kernel is given by the span of the vector

$$\begin{pmatrix} \ell_{\tau_1}^{r+1}\ell_{\tau_2}^{r+1}\ell_{\tau_3}^{r+1} \\ -\ell_{\tau_2}^{r+1}\ell_{\tau_3}^{r+1} \\ -\ell_{\tau_1}^{r+1}\ell_{\tau_3}^{r+1} \\ -\ell_{\tau_1}^{r+1}\ell_{\tau_2}^{r+1} \end{pmatrix}. \tag{4.2.1}$$

Therefore, every spline in $S^r(\Delta, \partial\Delta)$ is a polynomial multiple of the polynomial $\ell_{\tau_1}^{r+1}\ell_{\tau_2}^{r+1}\ell_{\tau_3}^{r+1}$. Restricting our view to only splines of degree at most d, we see that if $d < 3(r+1)$, we have no splines at all, except for the zero spline. If $d \geq 3(r+1)$, then we can form a basis for $S_d^r(\Delta, \partial\Delta)$ by multiplying $\ell_{\tau_1}^{r+1}\ell_{\tau_2}^{r+1}\ell_{\tau_3}^{r+1}$ by monomials of degree $d - 3(r+1)$. For example, the splines of degree $d = 3r+5$ have a basis

$$x^2\ell_{\tau_1}^{r+1}\ell_{\tau_2}^{r+1}\ell_{\tau_3}^{r+1},\ xy\ell_{\tau_1}^{r+1}\ell_{\tau_2}^{r+1}\ell_{\tau_3}^{r+1},\ y^2\ell_{\tau_1}^{r+1}\ell_{\tau_2}^{r+1}\ell_{\tau_3}^{r+1}.$$

Assuming the embedding of Δ in $\mathbb{R}^2$ is a non-degenerate triangle, then we see that $S^r(\Delta) = \langle \ell_{\tau_1}^{r+1}\ell_{\tau_2}^{r+1}\ell_{\tau_3}^{r+1} \rangle$ as the three edges are non-colinear. It is well known that the dth graded piece of an ideal in $\mathbb{R}[x, y]$ generated by a single polynomial of degree k has dimension $\binom{d-k+2}{2}$ where for integers A and B, we have

$$\binom{A}{B} = \begin{cases} \frac{A!}{B!(A-B)!} & A \geq B \geq 0, \\ 0 & \text{otherwise.} \end{cases}$$

With this, we arrive at a formula for the dimension of $S_d^r(\Delta, \partial\Delta)$:

$$\dim S_d^r(\Delta, \partial\Delta) = \binom{d-3r-1}{2} = \begin{cases} \frac{1}{2}d^2 - \frac{6r+3}{2}d + \frac{9r^2+9r+2}{2} & d \geq 3(r+1), \\ 0 & d < 3(r+1). \end{cases}$$

◇

Since every non-degenerate embedding of a pure 2-dimensional simplicial complex in $\mathbb{R}^2$ has at least one boundary face, we see that a spline must have at least degree $r+1$ unless it is the 0 spline. If the boundary face meets 2 boundary edges then splines must have degree at least $2(r+1)$, and degree at least $3(r+1)$ like in the above example if there is a boundary face with all edges on the boundary of Δ. Herein, we will always be assuming that $d \geq r+1$. We demonstrate the method for finding splines using the matrix $A(\Delta, \partial\Delta, r)$ with another example.

Example 4.2.2. Consider the following simplicial complex Δ as shown in as shown

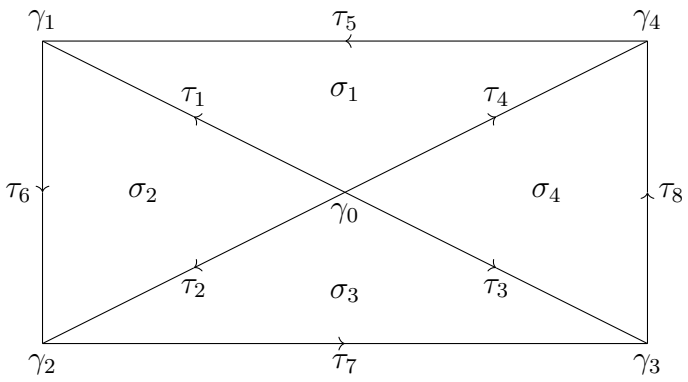

Figure 4.5: Simplicial complex Δ in Example 4.2.2.

in Figure 4.5. We write out each edge condition as described above using the algebraic spline criterion:

$$\begin{aligned} f|_{\sigma_1} - f|_{\sigma_2} &= g_1 \ell_{\tau_1}^{r+1}, \quad f|_{\sigma_1} = g_5 \ell_{\tau_5}^{r+1}, \\ f|_{\sigma_2} - f|_{\sigma_3} &= g_2 \ell_{\tau_2}^{r+1}, \quad f|_{\sigma_2} = g_6 \ell_{\tau_6}^{r+1}, \\ f|_{\sigma_3} - f|_{\sigma_4} &= g_3 \ell_{\tau_3}^{r+1}, \quad f|_{\sigma_3} = g_7 \ell_{\tau_7}^{r+1}, \\ f|_{\sigma_4} - f|_{\sigma_1} &= g_4 \ell_{\tau_4}^{r+1}, \quad f|_{\sigma_4} = g_8 \ell_{\tau_8}^{r+1}. \end{aligned} \tag{4.2.2}$$

We then write the matrix $A(\Delta, r)$ using these linear relations as per the definition. Using `Macaulay2`, we can compute the kernel of this matrix for various values of r.

To show what splines look like, we choose an embedding for Δ. We embed Δ in $\mathbb{R}^2$ so that

$$\begin{aligned} \ell_{\tau_1} &= y + 2x, \quad \ell_{\tau_5} = y - 2, \\ \ell_{\tau_2} &= y - 2x, \quad \ell_{\tau_6} = x + 1, \\ \ell_{\tau_3} &= y + 2x, \quad \ell_{\tau_7} = y + 2, \\ \ell_{\tau_4} &= y - 2x, \quad \ell_{\tau_8} = x - 1. \end{aligned} \tag{4.2.3}$$

For $r = 0$, we can easily construct splines following Courant's construction of hat functions [27]. We take the polynomial supported on σ_i to be the linear polynomial that is 1 at $\gamma_0 = (0,0)^\top$, and vanishes on τ_{i+4}, then $f = (f|_{\sigma_i})_{i=1}^4$ is a spline of smoothness 0 and degree 1.

Then, for $d = 1$, we have a single generator for the basis:

$$\left(1 - \frac{y}{2}, 1 + x, 1 + \frac{y}{2}, 1 - x\right).$$

Since we have only one generator of degree 1, the dimension of $S_1^0(\Delta, \partial\Delta)$ is 1 as an $\mathbb{R}$-vector space. In degree 2, a basis for the splines of $S_2^0(\Delta)$ is given by both x and y multiplied by the generator of $S_1^0(\Delta, \partial\Delta)$, as well as 2 additional degree 2 generators:

$$\begin{gathered} (2xy - y^2 - 4x + 2y, 4xy + 4y, 2xy + y^2 + 4x + 2y, 0) \\ (-2xy + y^2 + 4x - 2y, 4x^2 - 2xy + 4x - 2y, 0, 0). \end{gathered}$$

Thus, the dimension of $S_2^0(\Delta)$ is 5. Finally, for $d = 3$, the generators of $S_3^0(\Delta, \partial\Delta)$ are every monomial multiple of the prior generators, as well as 1 more spline of degree 3:

$$(4x^2y - y^3 - 8x^2 + 2y^2, 0, 0, 0),$$

giving $\dim S_3^0(\Delta, \partial\Delta) = 13$. Herein, for $d \geq 3$, the generators are given by monomial multiples of the generators of $S_3^0(\Delta, \partial\Delta)$, one of degree 1, two of degree 2, and one of degree 3. The number of smoothness 0 degree d splines, therefore is:

$$\dim S_d^0(\Delta, \partial\Delta) = \binom{d+1}{2} + 2\binom{d}{2} + \binom{d-1}{2} = \begin{cases} 2d^2 - 2d + 1 & d \geq 1, \\ 0 & d = 0. \end{cases}$$

◇

Repeating the above procedure in Example 4.2.2 will also produce $\dim S_d^r(\Delta, \partial\Delta)$ for any pure and hereditary simplicial complex Δ embedded in $\mathbb{R}^2$, and for every d and r, but doing so becomes prohibitively difficult as r, and the number of edges

and faces in Δ increases, and this methodology is not a practical way to compute the dimension of $S_d^r(\Delta, \partial\Delta)$ in general. One main issue is that this process cannot be easily scaled, as any change to Δ or to r will necessitate completely recomputing $A(\Delta, \partial\Delta, r)$. Another problem is that $A(\Delta, \partial\Delta, r)$ is based entirely on the choice of embedding of Δ in $\mathbb{R}^2$. If the vertices of Δ were to be slightly moved, we should accept no change to occur to the number of splines on Δ (assuming that the vertices started in generic positions). Unfortunately, this property is not at all clear from this method. We require an improved method if we wish to overcome the pitfalls of this procedure, which ought take into account not just the geometry of the particular embedding of Δ, but the topology as well.

As a primer, consider the generating function for the dimension of $S_d^r(\Delta, \partial\Delta)$:

$$\sum_{d=0}^{\infty} \dim S_d^r(\Delta, \partial\Delta) t^d,$$

in the ring of formal power series over $\mathbb{Z}$. Readers familiar with commutative algebra will recognise this as having a similar structure to that of the Hilbert polynomial of a graded module (see for example [4]). This is the observation that Billera and Rose made in [12], and we shall give a brief overview of how they created this connection with the dimension of $S_d^r(\Delta, \partial\Delta)$ and the Hilbert polynomial of a graded module. First, we recall the concept of a graded ring and module:

Definition 4.2.1 ([4])**.** We say the ring $R = \mathbb{R}[x, y, z]$ is *graded* if there is a family $(R_d)_{d\geq 0}$ of additive subgroups of R such that

$$R \cong \bigoplus_{d=0}^{\infty} R_d,$$

and $R_m R_n \subseteq R_{m+n}$ for all $m, n \geq 0$. In the case of the polynomial ring R, we let the R_d's be the groups of polynomials of degree equal to d. Recall an R-module is an abelian group M with the properties

$$f(x+y) = fx + fy,\ (f+g)x = fx + gx,\ (fg)x = f(gx),\ 1x = x,$$

for all $f, g \in R$ and $x, y \in M$ (Note that $f(x+y)$ is the product of f and $x+y$ here). An R-module is *graded* if there is a family $(M_d)_{d\geq 0}$ of subgroups of M such that

$$M \cong \bigoplus_{d=0}^{\infty} M_d,$$

and $R_m M_n \subseteq M_{m+n}$ for all $m, n \geq 0$.

With these properties, we can already see that $S^r(\Delta, \partial\Delta)$ is an $\mathbb{R}[x, y]$-module. To obtain a grading, we can perform a process known as homogenisation on the splines defined on Δ. For a polynomial, say in $p \in \mathbb{R}[x, y]$, the process involves introducing a new variable z, and constructing the polynomial $\hat{p} = z^{\deg(p)} p\left(\frac{x}{z}, \frac{y}{z}\right)$ in $\mathbb{R}[x, y, z]$. In the process, we obtain a homogeneous polynomial, and when setting $z = 1$, we restore our original polynomial p. This operation acts much the same way for splines, as we can perform this operation componentwise. However, by performing this operation, we are creating a spline in 3 variables, that naturally should be a spline defined over a simplical complex embedded in $\mathbb{R}^3$. To reconcile this, Billera and Rose [12] introduce the coning construction, a way to create a simplicial complex in $\mathbb{R}^3$ which, when restricted to the plane $\{z = 1\}$ recovers the original simplicial complex. We give an overview of the construction here:

Let Δ be a simplicial complex embedded in $\mathbb{R}^2$. First, embed Δ in the plane $\{z = 1\}$ in $\mathbb{R}^3$ using the map $\phi\colon \mathbb{R}^2 \to \mathbb{R}^3$ defined by $\phi(x, y) = (x, y, 1)$. For each simplex σ in Δ, define the cone over σ to be $\hat{\sigma}$ which is the simplex in $\mathbb{R}^3$ formed as the convex hull of $\phi(\sigma)$ and the origin in $\mathbb{R}^3$. The cone over Δ, denoted $\hat{\Delta}$, is then the simplicial complex in $\mathbb{R}^3$ consisting of the simplices $\{\hat{\alpha}\colon \alpha \in \Delta\}$ along with the origin in $\mathbb{R}^3$. An example of the coning construction can be seen in Figure 4.6. As all simplices in $\hat{\Delta}$ pass through the origin, the linear forms vanishing on them

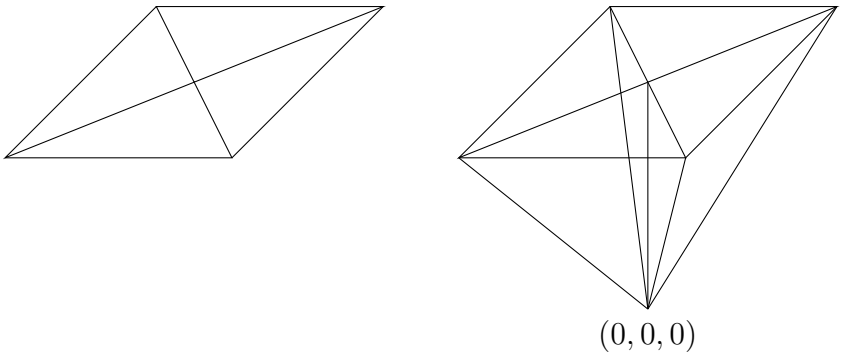

Figure 4.6: Left: A simplicial complex Δ. Right: The cone $\hat{\Delta}$ of Δ. Note that for splines without boundary conditions, since the interior edges of Δ already pass through the origin in $\mathbb{R}^2$, the linear forms are already homogeneous. However, for splines with boundary conditions, the exterior edges do not pass through the origin unless we construct the cone over Δ.

are necessarily homogeneous. This prompts the definition of a special kind of spline known as a homogeneous spline, which is only defined when each codimension 1 simplex passes through the origin.

Definition 4.2.2. A degree d *homogeneous* spline with boundary conditions on a simplicial complex Δ (embedded in $\mathbb{R}^2$ or $\mathbb{R}^3$) is a spline f with boundary conditions of degree d on Δ such that for every maximal degree simplex σ, the polynomial $f|_\sigma$ is a degree d homogeneous polynomial. The space of homogeneous splines with boundary conditions on Δ of degree d is denoted $\mathscr{H}_d^r(\Delta, \partial\Delta)$.

Under this, we see that any spline on $\hat{\Delta}$ must be homogeneous, and we now have a natural grading of the space of splines with boundary conditions defined over $\hat{\Delta}$.

$$S^r(\hat{\Delta}, \partial\hat{\Delta}) \cong \bigoplus_{i=0}^{\infty} \mathscr{H}_i^r(\hat{\Delta}, \partial\hat{\Delta}), \text{ and } S_d^r(\hat{\Delta}, \partial\hat{\Delta}) \cong \bigoplus_{i=0}^{d} \mathscr{H}_i^r(\hat{\Delta}, \partial\hat{\Delta}),$$

where these are $\mathbb{R}$-vector space isomorphisms. It is not hard to show that $S^r(\hat{\Delta}, \partial\hat{\Delta})$ is then a graded $\mathbb{R}[x, y, z]$-module with piecewise addition and multiplication, and that $\left(S^r(\hat{\Delta}, \partial\hat{\Delta})\right)_d = \mathscr{H}_d^r(\hat{\Delta}, \partial\hat{\Delta})$.

Given a spline on Δ, we can form a homogeneous spline defined over $\hat{\Delta}$ by homogenising each polynomial component. More specifically, if $f \in S_d^r(\Delta, \partial\Delta)$, and $\sigma \in \Delta$ is a maximal simplex, then define a spline $\hat{f} \in \mathscr{H}_d^r(\hat{\Delta}, \partial\hat{\Delta})$ by taking $\hat{f}|_{\hat{\sigma}} = z^d f|_\sigma \left(\frac{x}{z}, \frac{y}{z}\right)$. This gives a canonical injective map from $\psi\colon S_d^r(\Delta, \partial\Delta) \to \mathscr{H}_d^r(\hat{\Delta}, \partial\hat{\Delta})$. We will see that this is, in fact, an isomorphism as $\mathbb{R}$-vector spaces. Billera and Rose proved an analagous result for splines without boundary conditions in [12, Theorem 2.6], and for completeness we prove this result for splines with boundary conditions.

Proposition 4.2.3. *The canonical injective map $\psi\colon S_d^r(\Delta, \partial\Delta) \to \mathscr{H}_d^r(\hat{\Delta}, \partial\hat{\Delta})$ is an isomorphism of $\mathbb{R}$-vector spaces.*

Proof. We must show that ψ is well defined, and an isomorphism of $\mathbb{R}$-vector spaces. Let σ, σ' be maximal simplices in Δ, and $\tau = \sigma \cap \sigma'$. Then let $f \in S_d^r(\Delta, \partial\Delta)$, we know that $f|_\sigma - f|_{\sigma'} = g\ell_\tau^{r+1}$ for some $g \in \mathbb{R}[x, y]$. We also have that the $r+1$st power of the linear form vanishing on $\hat{\tau}$ is $\ell_{\hat{\tau}}^{r+1} = z^{r+1}\ell_\tau\left(\frac{x}{z}, \frac{y}{z}\right)^{r+1}$. Then, since $\deg(g) + r + 1 = \deg(f|_\sigma - f|_{\sigma'}) \leq d$,

$$\begin{aligned}
\psi(f)|_{\hat{\sigma}} - \psi(f)|_{\hat{\sigma'}} &= z^d f|_\sigma \left(\frac{x}{z}, \frac{y}{z}\right) - z^d f|_{\sigma'} \left(\frac{x}{z}, \frac{y}{z}\right) \\
&= z^d g\left(\frac{x}{z}, \frac{y}{z}\right) \ell_\tau^{r+1}\left(\frac{x}{z}, \frac{y}{z}\right) \\
&= z^a \hat{g} \ell_{\hat{\tau}}^{r+1} \in \langle \ell_{\hat{\tau}}^{r+1} \rangle,
\end{aligned}$$

where $a = d - (r+1) - \deg(g) \geq 0$. Then, suppose σ is a boundary face with boundary edge τ. Then $f|_\sigma = h\ell_\tau^{r+1}$ for some $h \in \mathbb{R}[x, y]$, and since $\deg(h) + r + 1 =$

$\deg(f|_\sigma) \leq d$, we have

$$\begin{aligned}\psi(f)|_{\hat{\sigma}} &= z^d f|_\sigma \left(\frac{x}{z}, \frac{y}{z}\right) \\ &= z^d h\left(\frac{x}{z}, \frac{y}{z}\right) \ell_\tau \left(\frac{x}{z}, \frac{y}{z}\right) \\ &= z^a \hat{h} \ell_{\hat{\tau}}^{r+1} \in \langle \ell_{\hat{\tau}}^{r+1} \rangle,\end{aligned}$$

where $a = d - (r+1) - \deg(h) \geq 0$. So, $\psi(f) \in \mathscr{H}_d^r(\hat{\Delta}, \partial\hat{\Delta})$ by the algebraic spline criterion, and thus ψ is well-defined.

- Injectivity: If $\psi(f) = 0$, then $z^d f|_\sigma \left(\frac{x}{z}, \frac{y}{z}\right) = 0$ for each $\sigma \in \Delta_2$. But then when $z = 1$, we have $f|_\sigma(x, y) = 0$, so ψ is injective.

- Surjectivity: If $g \in \mathscr{H}_d^r(\hat{\Delta}, \partial\hat{\Delta})$, then $g|_{\hat{\sigma}}(x, y, 1)$ is a polynomial defined on σ, and $z^d g|_{\hat{\sigma}} \left(\frac{x}{z}, \frac{y}{z}, 1\right) = g|_{\hat{\sigma}}(x, y, z)$ so we define $f \colon \Delta \to \mathbb{R}$ as $f|_\sigma = g|_{\hat{\sigma}}(x, y, 1)$, and we see that $\psi(f) = g$ as long as $f \in S_d^r(\Delta, \partial\Delta)$. We know $f|_\sigma$ is a polynomial with degree at most d, and if σ, σ' share an edge τ, then $\hat{\sigma}, \hat{\sigma'}$ share a face $\hat{\tau}$, and

  $$g|_{\hat{\sigma}} - g|_{\hat{\sigma'}} = h\ell_{\hat{\tau}}^{r+1},$$

  for some $h \in \mathbb{R}[x, y, z]$. Then, setting $z = 1$ gives

  $$\begin{aligned}f|_\sigma - f|_{\sigma'} &= g|_{\hat{\sigma}}(x, y, 1) - g|_{\hat{\sigma'}}(x, y, 1) \\ &= h(x, y, 1)\ell_{\hat{\tau}}^{r+1}(x, y, 1) \\ &= h(x, y, 1)\ell_\tau^{r+1} \in \langle \ell_\tau^{r+1} \rangle.\end{aligned}$$

  If σ is a boundary face and τ a boundary edge of Δ, an exactly analogous proof finishes the process of showing that f satisfies the algebraic spline criterion, and so $f \in S_d^r(\Delta, \partial\Delta)$, showing ψ is surjective.

□

It is clear from this result that the generating function of $S_d^r(\Delta, \partial\Delta)$ is equal to the Hilbert series of $\mathscr{H}_d^r(\hat{\Delta}, \partial\hat{\Delta})$. By making use of the grading by degree that we have on $\mathscr{H}^r(\hat{\Delta}, \partial\hat{\Delta})$, we introduce the homological methods required to find a formula for the dimension of $\mathscr{H}_d^r(\hat{\Delta}, \partial\hat{\Delta})$. During this process, as a result of the equivalence of $S_d^r(\Delta, \partial\Delta)$ and $\mathscr{H}_d^r(\hat{\Delta}, \partial\hat{\Delta})$, we will display simplicial complexes Δ as being embedded in $\mathbb{R}^2$, but implicitly we be working with their cones $\hat{\Delta}$ in $\mathbb{R}^3$.

4.3 Chain Complexes and Simplicial Homology

In this section, we will recall some basic results about algebraic structures known as chain complexes. Let $R = \mathbb{R}[x, y, z]$ be the ring of polynomials in x, y, z over $\mathbb{R}$. Recall that splines defined on the cone of a simplicial complex embedded in $\mathbb{R}^2$ are piecewise homogeneous polynomials in R. It is pertinent therefore to consider the grading of R by degree, $\mathscr{H}^r_d(\hat{\Delta}, \partial\hat{\Delta})$. The use of homological methods to study this graded module was first explored by Billera in [11], and expanded upon by Billera and Rose in [12, 13]. The homological approach to splines was then formalised into a well known chain complex by Schenck and Stillman in [67, 68]. For a more thorough introduction to chain complexes and their link to simplicial complexes, see for example [46]. Throughout this section, we will assume that all R-modules are graded. A chain complex $\mathcal{C}$ of modules over a ring R is a set of R-modules $\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2, \ldots$ and R-module homomorphisms $\partial_i \colon \mathcal{C}_i \to \mathcal{C}_{i-1}$ for $i \geq 1$, and $\partial_0 \colon \mathcal{C}_0 \to 0$ satsifying that for each $i \geq 1$, $\mathrm{Im}(\partial_i) \subseteq \ker(\partial_{i-1})$. Notice that this condition implies that for each $i \geq 1$, $\partial_{i-1} \circ \partial_i \colon \mathcal{C}_i \to \mathcal{C}_{i-2}$ is the zero homomorphism. Typically, we represent a chain complex via a diagram

$$\mathcal{C} \colon \cdots \xrightarrow{\partial_{n+1}} \mathcal{C}_n \xrightarrow{\partial_n} \cdots \xrightarrow{\partial_3} \mathcal{C}_2 \xrightarrow{\partial_2} \mathcal{C}_1 \xrightarrow{\partial_1} \mathcal{C}_0 \xrightarrow{\partial_0} 0.$$

A chain complex need not be infinite in length, in the sense that a finite length chain complex will have $\mathcal{C}_i = 0$ for all sufficiently large i. For a general chain complex $\mathcal{C}$, the ith *homology group* of $\mathcal{C}$ is the quotient

$$H_i(\mathcal{C}) = \frac{\ker(\partial_i)}{\mathrm{Im}(\partial_{i+1})},$$

for each $i \geq 0$. Since the $\mathcal{C}_i$'s are R-modules, then the $H_i(\mathcal{C})$'s are also R-modules. Notice that since the $\mathcal{C}_i$'s are graded R-modules, the homology groups are also graded R-modules, and the dimensions of the graded pieces $(\mathcal{C}_i)_d$ are related to the graded pieces $(H_i(\mathcal{C}))_d$ of the homology groups by the graded Euler-Poincaré characteristic [46], denoted $\chi(\mathcal{C}, d)$ which is defined as

$$\chi(\mathcal{C}, d) = \sum_{i=0}^{n} (-1)^{n-i} \dim(\mathcal{C}_i)_d.$$

The exact relation is:

Proposition 4.3.1. *Let $\mathcal{C}$ be a finite chain complex of graded R-modules $\mathcal{C}_0, \ldots, \mathcal{C}_n$, that is*

$$\mathcal{C} \colon 0 \to \mathcal{C}_n \to \cdots \to \mathcal{C}_1 \to \mathcal{C}_0 \to 0.$$

Let $d \geq 0$ be an integer. Then, the Euler-Poincaré characteristic of degree d of $\mathcal{C}$ satisfies

$$\chi(\mathcal{C}, d) = \sum_{i=0}^{n} (-1)^{n-i} \dim(H_i(\mathcal{C}))_d.$$

We omit the proof here, but it follows from the Rank-Nullity theorem. A concise proof of the case for non-graded modules can be found in [46, Theorem 2.44], and the case for graded modules follows by taking graded pieces.

4.3.1 Exact Sequences

Any chain complex $\mathcal{C}$ with boundary maps ∂_i for $i \geq 0$ has the property that $\mathrm{Im}(\partial_i) \subseteq \ker(\partial_{i-1})$ for $i \geq 1$, but a chain complex with the further property that $\mathrm{Im}(\partial_i) = \ker(\partial_{i-1})$ for each $i \geq 1$ are called *exact* (or an *exact sequence*). The sequence is called *short exact* if $\mathcal{C}_i = 0$ for each $i > 2$, and *long exact* otherwise. Exact sequences have the property that their homology groups are all 0, and thus every graded Euler-Poincaré characteristic must also vanish. If we have a triple of chain complexes $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$, with boundary maps $\partial_i^{\mathcal{A}}$, $\partial_i^{\mathcal{B}}$, and $\partial_i^{\mathcal{C}}$ respectively, then we say that they form a short exact sequence of chain complexes

$$0 \to \mathcal{A} \to \mathcal{B} \to \mathcal{C} \to 0, \tag{4.3.1}$$

if there are maps $\phi_i \colon \mathcal{A}_i \to \mathcal{B}_i$ and $\psi_i \colon \mathcal{B}_i \to \mathcal{C}_i$ for $i \geq 0$ so that $\psi_i \circ \phi_i = 0$ and the diagram in Figure 4.7 is commutative, where we say a diagram commutes if any two compositions of maps starting at one point in the diagram and ending at another are equal.

A short exact sequence of chain complexes induces a long exact sequence in the homology groups of each chain complex. For chain complexes $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$ as in (4.3.1), Figure 4.8 depicts this long exact sequence. This result connects a short exact sequence of chain complexes with their homology groups. The exactness allows one to determine large amounts of information about the homology groups, especially if one or more of the involved chain complexes $\mathcal{A}, \mathcal{B}$ or $\mathcal{C}$ has known homology groups. This result is also the basis for many topological constructions, such as the Mayer-Vietoris long exact sequence (if it is not taken to be an axiom) which we state now, but make full use of later.

Proposition 4.3.2 ([46, Section 2.2])**.** *Let Δ be a simplicial complex, and $\Lambda_1, \Lambda_2 \subseteq \Delta$ subcomplexes so that $\Delta = \Lambda_1 \cup \Lambda_2$. Then, the following Figure 4.9 is a long exact sequence known as the Mayer-Vietoris long exact sequence for Λ_1, Λ_2:*

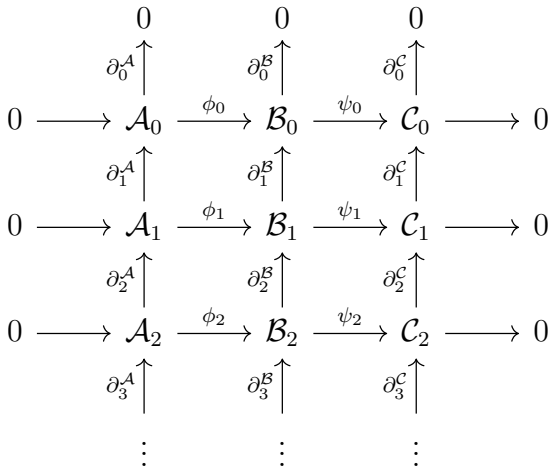

Figure 4.7: The commutative diagram corresponding to the short exact sequence of chain complexes (4.3.1)

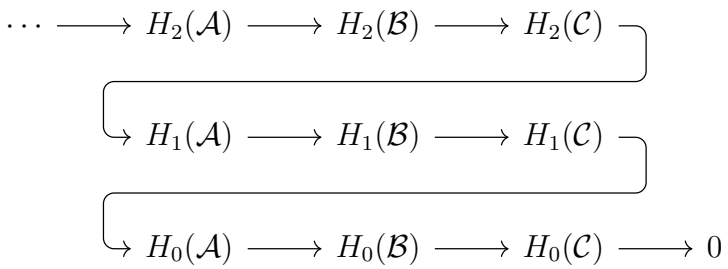

Figure 4.8: The long exact sequence of homology groups from the short exact sequence of chain complexes (4.3.1)

We can construct a chain complex associated to a spline space as follows. Recall the definition of the simplicial boundary map ∂_i for $i = 0, 1, 2$ defined by mapping formal sums of i-simplices to formal sums of $i-1$-simplices. We note that by definition, the simplicial boundary maps satisfy $\partial_{i-1} \circ \partial_i = 0$ for $i = 1, 2$. We notice that a formal sum of i-simplices can be represented as a direct sum of copies of R since the coefficients in the formal sum can be any polynomial in R. We associate to each i-simplex a copy of R representing the polynomial associated with σ, and so ∂_i is a map from $R^{|\Delta_i|}$ to $R^{|\Delta_{i-1}|}$ where $i = 1, 2$, and ∂_0 is a map from $R^{|\Delta_0|}$ to 0, since the boundary of any 0-simplex is 0 by convention. It then follows that we have a chain complex

$$\mathcal{R}\colon 0 \to R^{|\Delta_2|} \xrightarrow{\partial_2} R^{|\Delta_1|} \xrightarrow{\partial_1} R^{|\Delta_0|} \xrightarrow{\partial_0} 0. \tag{4.3.2}$$

The homology groups of this chain complex are known to provide information about the topological structure of the simplicial complex Δ, but not about the specific embedding of Δ. We then proceed to define a subchain complex of $\mathcal{R}$ in the following

$$\cdots \longrightarrow H_2(\Lambda_1 \cap \Lambda_2) \longrightarrow H_2(\Lambda_1) \oplus H_2(\Lambda_2) \longrightarrow H_2(\Delta) \longrightarrow$$
$$H_1(\Lambda_1 \cap \Lambda_2) \longrightarrow H_1(\Lambda_1) \oplus H_1(\Lambda_2) \longrightarrow H_1(\Delta) \longrightarrow$$
$$H_0(\Lambda_1 \cap \Lambda_2) \longrightarrow H_0(\Lambda_1) \oplus H_0(\Lambda_2) \longrightarrow H_0(\Delta) \longrightarrow 0$$

Figure 4.9: The Mayer-Vietoris long exact sequence for Λ_1 and Λ_2.

way. The following construction is due to Schenck and Stillman in [68]:

- For each face $\sigma \in \Delta_2$, we associate the ideal $J(\sigma) = 0$.
- For each edge $\tau \in \Delta_1$, we associate the ideal $J(\tau) = \langle \ell_\tau^{r+1} \rangle$, where ℓ_τ is a linear polynomial vanishing on τ.
- For each vertex $\gamma \in \Delta_0$, we associate the ideal $J(\gamma) = \langle \ell_\tau^{r+1} : \gamma \in \tau \text{ for } \tau \in \Delta_1 \rangle$.

We can then construct a subchain complex of $\mathcal{R}$ by simply restricting $R^{|\Delta_i|}$ to $\bigoplus_{\alpha \in \Delta_i} J(\alpha)$:

$$\mathcal{J} : 0 \to 0 \to \bigoplus_{\tau \in \Delta_1} J(\tau) \xrightarrow{\partial_1} \bigoplus_{\gamma \in \Delta_0} J(\gamma) \xrightarrow{\partial_0} 0. \tag{4.3.3}$$

By forming the quotient of each element in $\mathcal{R}$ with each element in $\mathcal{J}$, we create the quotient chain complex $\mathcal{R}/\mathcal{J}$:

$$\mathcal{R}/\mathcal{J} : 0 \to \bigoplus_{\sigma \in \Delta_2} R \to \bigoplus_{\tau \in \Delta_1} \frac{R}{J(\tau)} \to \bigoplus_{\gamma \in \Delta_0} \frac{R}{J(\gamma)} \to 0. \tag{4.3.4}$$

There is a simple inclusion map from the chain complex $\mathcal{J}$ to $\mathcal{R}$ that is injective on each element of the chain complex. The first isomorphism theorem then implies that the image of the map is isomorphic to $\mathcal{J}$. There is also the canonical map from $\mathcal{R}$ to $\mathcal{R}/\mathcal{J}$ that is the quotient map, taking $r_i \in R^{|\Delta_i|}$ to $r_i + \bigoplus_{\alpha \in \Delta_i} J(\alpha)$. The kernel of this map is exactly $\mathcal{J}$, meaning there is a short exact sequence of chain complexes

$$0 \to \mathcal{J} \to \mathcal{R} \to \mathcal{R}/\mathcal{J} \to 0, \tag{4.3.5}$$

connecting $\mathcal{J}$, $\mathcal{R}$, and $\mathcal{R}/\mathcal{J}$. Therefore, there is a long exact sequences in the homology groups of $\mathcal{J}$, $\mathcal{R}$ and $\mathcal{R}/\mathcal{J}$ by replacing $\mathcal{A}, \mathcal{B}, \mathcal{C}$ with $\mathcal{J}, \mathcal{R}, \mathcal{R}/\mathcal{J}$ respectively in Figure 4.8.

For splines with no boundary conditions, the equivalent chain complex for $\mathcal{R}/\mathcal{J}$ is commonly referred to in the literature as the Billera-Schenck-Stillman chain complex,

after the authors who first investigated it in [11, 67, 68]. The homology groups of the Billera-Schenck-Stillman chain complex gives much information about the splines on Δ, and we will show that the chain complex (4.3.4) contains the same information about the splines with boundary conditions over Δ. As shown in [13, Proposition 5.1], the ring of splines of smoothness r is isomorphic to the second homology group of the Billera-Schenck-Stillman chain complex, albeit in a slightly different way than we use here. We will show the equivalent result for the chain complex (4.3.4).

Proposition 4.3.3. *Let Δ be a pure, hereditary, simplicial complex. Then,*

$$S^r(\Delta, \partial\Delta) \cong H_2(\mathcal{R}/\mathcal{J}).$$

Proof. We first embed $S^r(\Delta, \partial\Delta)$ in $\bigoplus_{\sigma\in\Delta_2} R$ into the chain complex (4.3.4) in the natural way. Then for a spline $f \in S^r(\Delta, \partial\Delta)$, if $\tau = \sigma_i \cap \sigma_j$ for some $\sigma_i, \sigma_j \in \Delta_2$, then we have $f|_{\sigma_i} - f|_{\sigma_j} \in J(\tau)$ by the algebraic spline criterion, Proposition 4.2.2. Thus, f must map to 0 under the boundary map $\partial_2^{\mathcal{R}/\mathcal{J}}$. Since a tuple $(f_1, \ldots, f_n)$ satisfying the algebraic spline criterion is necessary and sufficient to say that it is a spline, we have $S^r(\Delta, \partial\Delta) = \ker\left(\partial_2^{\mathcal{R}/\mathcal{J}}\right) = H_2(\mathcal{R}/\mathcal{J})$. □

We will use Proposition 4.3.3 to compute the dimension of the space of splines with boundary conditions, $S^r_d(\Delta, \partial\Delta)$.

4.4 The Dimension of the Space of Splines with Boundary Conditions

This section is dedicated to computing the dimension of $S^r_d(\Delta, \partial\Delta)$ for sufficiently large d.

Theorem 4.4.1. *Let Δ be a pure, hereditary simplicial complex embedded in $\mathbb{R}^2$. Then, the following holds:*

$$\begin{aligned}\dim S^r_d(\Delta, \partial\Delta) = &\sum_{\tau\in\Delta_1} \dim J(\tau)_d - \sum_{\gamma\in\Delta_0} \dim J(\gamma)_d \\ &+ \dim H_0(\mathcal{R})_d - \dim H_1(\mathcal{R})_d + \dim H_2(\mathcal{R})_d \\ &+ \dim H_1(\mathcal{R}/\mathcal{J})_d - \dim H_0(\mathcal{R}/\mathcal{J})_d.\end{aligned} \tag{4.4.1}$$

Proof. Applying the dth graded Euler-Poincaré characteristic on the long exact sequence in homology of the short exact sequence (4.3.5) gives

$$\begin{aligned} 0 = &\dim H_2(\mathcal{J})_d - \dim H_2(\mathcal{R})_d + \dim H_2(\mathcal{R}/\mathcal{J})_d \\ &- \dim H_1(\mathcal{J})_d + \dim H_1(\mathcal{R})_d - \dim H_1(\mathcal{R}/\mathcal{J})_d \\ &+ \dim H_0(\mathcal{J})_d - \dim H_0(\mathcal{R})_d + \dim H_0(\mathcal{R}/\mathcal{J})_d \end{aligned}$$

By applying the dth graded Euler-Poincaré characteristic on the chain complex (4.3.3) gives

$$\dim H_0(\mathcal{J})_d - \dim H_1(\mathcal{J})_d = \dim \bigoplus_{\gamma \in \Delta_0} J(\gamma)_d - \dim \bigoplus_{\tau \in \Delta_1} J(\tau)_d.$$

We also know that $H_2(\mathcal{J})_d = 0$, and that $\dim H_2(\mathcal{R}/\mathcal{J})_d = \dim S^r_d(\Delta, \partial\Delta)$ by Proposition 4.3.3. Combining each result proves the theorem. □

We talk about each part of this dimension formula separately. We begin with some homological results about the chain complex $\mathcal{R}$. Knowing the homology groups of the chain complex $\mathcal{R}$ allows us to write the homology group $H_2(\mathcal{R}/\mathcal{J})$ in terms of the homology groups of the chain complex $\mathcal{J}$, $H_1(\mathcal{R}/\mathcal{J})$, and $H_0(\mathcal{R}/\mathcal{J})$. It is of great import to determine the dimension of the graded pieces of these homology groups. We cannot explicitly compute the dimensions of their graded pieces for general configurations, but we will make use of a result proved by McDonald in [57], the proof of which is based on one of a similar result of Schenck and Stillman in [68]

Lemma 4.4.2 ([57],Corollary 3.2, Proposition 3.5)**.** *Let Δ be a pure, hereditary simplicial complex. Then,* $\dim H_1(\mathcal{R}/\mathcal{J})_d = \dim H_0(\mathcal{R}/\mathcal{J})_d = 0$ *for* $d \gg 0$.

We say that a property is true for $d \gg 0$ if and only if there is some $D > 0$ such that the property is true for $d \geq D$, and say that the property is true for d *sufficiently large*. We move our attention to the dimensions $H_0(\mathcal{R})$, $H_1(\mathcal{R})$, and $H_2(\mathcal{R})$. As these appear with alternating signs in (4.4.1), we can rewrite their contribution in terms of the dth graded Euler-Poincaré characteristic of the chain complex $\mathcal{R}$ by Proposition 4.3.1 in terms of R_d, the set of polynomials of degree d in $\mathbb{R}[x, y, z]$:

$$\begin{aligned} \chi(\mathcal{R}, d) &= \dim H_0(\mathcal{R})_d - \dim H_1(\mathcal{R})_d + \dim H_2(\mathcal{R})_d \\ &= \dim R_d^{|\Delta_0|} - \dim R_d^{|\Delta_1|} + \dim R_d^{|\Delta_2|} \\ &= (|\Delta_0| - |\Delta_1| + |\Delta_2|) \dim R_d. \end{aligned} \tag{4.4.2}$$

The remaining contributions to the dimension of $S^r_d(\Delta, \partial\Delta)$ arises from the dimensions of $J(\tau)$ for edges $\tau \in \Delta_1$, and from $J(\gamma)$ for vertices $\gamma \in \Delta_0$. We begin with

the dimension of $J(\tau)$, as we will see that it is the same, regardless of the edge we choose [68]. The final contributions are from the dimensions of the ideals $J(\gamma)$ for vertices $\gamma \in \Delta_0$. The dimension of $J(\gamma)_d$ can be found very easily in the case where the vertex γ has only two incident edges, but in practice, most vertices have at least 3 incident edges. The number of incident edges, however, is not exactly what we want to be considering, more the number of distinct slopes of incident edges. To start with, if the three incident edges involve a pair of colinear edges, such as shown below in Figure 4.10 then

$$J(\gamma) = J(\tau_1) + J(\tau_2) + J(\tau_3) = \langle \ell_{\tau_1}^{r+1} \rangle + \langle \ell_{\tau_2}^{r+1} \rangle + \langle \ell_{\tau_3}^{r+1} \rangle = \langle \ell_{\tau_1}^{r+1} \rangle + \langle \ell_{\tau_3}^{r+1} \rangle,$$

and the dimension can be computed by inclusion exclusion. If we change the slope

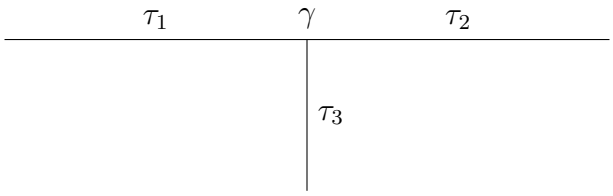

Figure 4.10: A diagram displaying a vertex γ with three incident edges τ_1, τ_2, τ_3, but only two distinct slopes of incident edges.

of either τ_1 or τ_2 in Figure 4.10 minutely, then the dimension of $J(\gamma)_d$ will change, but as it stands, the dimension of $J(\gamma)_d$ is indistinguishable from the dimension of $J(\gamma')_d$ of any vertex γ' that has only two incident edges.

In [67, Theorem 3.1], Schenck and Stillman found a free resolution of $\bigoplus_{\gamma \in \Delta_0} R/J(\gamma)_d$, which can be used to compute $\dim J(\gamma)_d$ for each vertex γ. We will recall this result here, and use it to compute the dimension of $J(\gamma)_d$ for vertices γ with any number of distinct slopes of edges incident to γ.

Lemma 4.4.3 ([67]). *Let $r \geq 0$ be an integer. Let γ be a vertex in Δ_0, and let n_γ be the number of edges incident to γ with distinct gradients, and $t_\gamma = \min\{n_\gamma, r+2\}$. We define the following values $q_\gamma = \lfloor \frac{t_\gamma(r+1)}{t_\gamma - 1} \rfloor$, $a_\gamma = t_\gamma(r+1) - (t_\gamma - 1)q_\gamma$, and $b_\gamma = t_\gamma - 1 - a_\gamma$. Then*

$$\dim J(\gamma)_d = t_\gamma \binom{d-r+1}{2} - a_\gamma \binom{d-q_\gamma+1}{2} - b_\gamma \binom{d-q_\gamma+2}{2},$$

for all d.

The dimension of $J(\gamma)_d$ depends on more than the topology of the simplicial complex. The particular embedding of a simplicial complex impacts the dimension of the spline space $S^r_d(\Delta, \partial\Delta)$ as seen in this formula. With this result, we can finally combine everything into our main result.

Theorem 4.4.4. *Let Δ be a pure, hereditary simplicial complex, and let $r, d \geq 0$ be integers. Let*

$$Q(\Delta, r, d) = |\Delta_1| \binom{d-r+1}{2} + (|\Delta_0| - |\Delta_1| + |\Delta_2|) \binom{d+2}{2} - \sum_{\gamma \in \Delta_0} \left(t_\gamma \binom{d-r+1}{2} - a_\gamma \binom{d-q_\gamma+1}{2} - b_\gamma \binom{d-q_\gamma+2}{2} \right). \quad (4.4.3)$$

Then, $\dim S^r_d(\Delta, \partial\Delta) = Q(\Delta, r, d) + \dim H_1(\mathcal{R}/\mathcal{J})_d - \dim H_0(\mathcal{R}/\mathcal{J})_d$ *for all* $d, r \geq 0$, *and* $\dim S^r_d(\Delta, \partial\Delta) = Q(\Delta, r, d)$ *for* $d \gg 0$.

Proof. Recall equation (4.4.1) from Theorem 4.4.1. We know that

$$\sum_{\tau \in \Delta_1} J(\tau) = \sum_{\tau \in \Delta_1} \binom{d-r+1}{2} = |\Delta_1| \binom{d-r+1}{2}.$$

By equation (4.4.2),

$$\dim H_0(\mathcal{R})_d - \dim H_1(\mathcal{R})_d + \dim H_2(\mathcal{R})_d = (|\Delta_0| - |\Delta_1| + |\Delta_2|) \binom{d+2}{2}.$$

By Lemma 4.4.3,

$$\sum_{\gamma \in \Delta_0} \dim J(\gamma)_d = \sum_{\gamma \in \Delta_0} \left(t_\gamma \binom{d-r+1}{2} - a_\gamma \binom{d-q_\gamma+1}{2} - b_\gamma \binom{d-q_\gamma+2}{2} \right),$$

and combining these values in (4.4.1) yields the first part of the theorem. Then Lemma 4.4.2 verifies that $\dim S^r_d(\Delta, \partial\Delta) = Q(\Delta, r, d)$ for $d \gg 0$. □

We know that $\mathcal{R}$ is the simplicial chain complex of Δ, and we can quite easily determine its homology groups. If we associate the simplicial complex Δ with its embedding in $\mathbb{R}^2$, then the homology groups of Δ can be determined from the number of holes and the number of connected components, and we can thus identify the effect of creating a hole in a simplicial complex.

Proposition 4.4.5. *Let Δ be a pure and hereditary simplicial complex embedded in $\mathbb{R}^2$ with c connected components and h holes in the embedding of $\mathbb{R}^2$. Then the homology groups are*

$$H_i(\mathcal{R}) \cong \begin{cases} R^c & i = 0, \\ R^h & i = 1, \\ 0 & i \geq 2. \end{cases}$$

Proof. Note that the homology of Δ is the direct sum of the homologies of its connected components, so we consider Δ to have a single connected component without loss of generality. We will use, in this proof, knowledge of the homology groups of a point, and of a circle, see [46, Proposition 2.8] and [46, Example 2.2] respectively. Suppose first that Δ has no holes, and so is homeomorphic to a point, then $H_0(\mathcal{R}) \cong R$, $H_1(\mathcal{R}) \cong 0$, and $H_i(\mathcal{R}) \cong 0$ for $i \geq 2$. Then, suppose if Δ has h holes, that $H_0(\mathcal{R}) \cong R$, $H_1(\mathcal{R}) \cong R^h$, and $H_i(\mathcal{R}) \cong 0$ for $i \geq 2$. Then, suppose Δ has $h+1$ holes, and so is homeomorphic to $h+1$ circles connected together. Split Δ into $\Lambda_1 \cup \Lambda_2$ where Λ_1 has h holes and Λ_2 has 1 hole, and their intersection $\Lambda_1 \cap \Lambda_2$ is homeomorphic to a point. Then, the Mayer-Vietoris long exact sequence states that the following are exact:

$$0 \to H_i(\mathcal{R}) \to 0 \text{ for } i \geq 2, \tag{4.4.4}$$

$$0 \xrightarrow{\phi_5} R^h \oplus R \xrightarrow{\phi_4} H_1(\mathcal{R}) \xrightarrow{\phi_3} R \xrightarrow{\phi_2} R \oplus R \xrightarrow{\phi_1} H_0(\mathcal{R}) \xrightarrow{\phi_0} 0. \tag{4.4.5}$$

The sequence (4.4.4) implies that $H_i(\mathcal{R}) \cong 0$ for $i \geq 2$, and [46, Proposition 2.7] shows that $H_0(\Delta) \cong R$ since Δ has a single connected component. So, we now just consider (4.4.5). Using the first isomorphism theorem and exactness, we have

$$\frac{R^{h+1}}{\ker \phi_4} \cong \mathrm{Im}\phi_4, \ \frac{H_1(\mathcal{R})}{\ker \phi_3} \cong \mathrm{Im}\phi_3, \ \frac{R}{\ker \phi_2} \cong \mathrm{Im}\phi_2, \ \frac{R^2}{\ker \phi_1} \cong \mathrm{Im}\phi_1, \ \frac{R}{\ker \phi_0} \cong \mathrm{Im}\phi_0.$$

Additionally, $\mathrm{Im}\phi_5 = 0$ and $\mathrm{Im}\phi_0 = 0$, we can then logically determine $H_1(\mathcal{R})$ following the steps, noting that all groups here are isomorphic to powers of R (and thus have no torsion).

1. $\mathrm{Im}\phi_0 \cong 0 \implies \ker \phi_0 = \mathrm{Im}\phi_1 \cong R$,
2. $\mathrm{Im}\phi_1 \cong R \implies \ker \phi_1 = \mathrm{Im}\phi_2 \cong R$,
3. $\mathrm{Im}\phi_2 \cong R \implies \ker \phi_2 = \mathrm{Im}\phi_3 \cong 0$,
4. $\mathrm{Im}\phi_3 \cong 0 \implies \ker \phi_3 = \mathrm{Im}\phi_4 \cong H_1(\mathcal{R})$,

5. $\ker \phi_4 = \mathrm{Im}\phi_5 = 0$,

6. $\ker \phi_4 \cong 0$ and $\mathrm{Im}\phi_4 \cong H_1(\mathcal{R}) \implies R^{h+1} \cong H_1(\mathcal{R})$.

This completes the proof. □

By Proposition 4.3.1 we can can see that

$$(|\Delta_0| - |\Delta_1| + |\Delta_2|)\binom{d+2}{2} = \chi(\mathcal{R}, d) = (c-h)\binom{d+2}{2},$$

where $\chi(\mathcal{R}, d)$ is the dth graded Euler-Poincaré characteristic of the chain complex $\mathcal{R}$, c is the number of connected components of Δ, and h is the number of holes of Δ.

4.5 Examples

In this section we demonstrate a number of examples of the formula in Theorem 4.4.4 for $\dim S_d^r(\Delta, \partial\Delta)$ for sufficiently large d. We will reconsider the two previously explored simplicial complexes first, and then consider some modifications and special cases that lead to interesting behavoiur.

Example 4.5.1. Consider once again the following simplicial complex Δ in Figure 4.4. We found a basis for $S_d^r(\Delta, \partial\Delta)$ before using $A(\Delta, \partial\Delta, r)$. We may obtain the value of $\frac{d^2-(6r+3)d+9r^2+9r+2}{2}$ for the dimension of $S_d^r(\Delta, \partial\Delta)$ in a different way using the formula (4.4.3) in Theorem 4.4.4. For each edge τ_i, ℓ_{τ_i} is the linear form vanishing on τ_i, and $J(\tau_i) = \langle \ell_{\tau_i}^{r+1} \rangle$. We have $|\Delta_0| = |\Delta_1| = 3$ and $|\Delta_2| = 1$, so we now only consider the sum over the vertices. Each vertex is essentially indistinguishable from the other, so consider any vertex γ. This vertex has exactly 2 incident edges, each with distinct slopes, and so $t_\gamma = 2$. Then, we can easily find

$$q_\gamma = \left\lfloor \frac{2(r+1)}{2-1} \right\rfloor = 2r+2, \; a_\gamma = 0, \; b_\gamma = 1.$$

Combining this, we have that

$$Q(\Delta, r, d) = 3\binom{d-r+1}{2} + \binom{d+2}{2} - 3\left(2\binom{d-r+1}{2} - \binom{d-2r}{2}\right).$$

Expanding each term and simplifying we find that

$$\dim S_d^r(\Delta, \partial\Delta) = \frac{1}{2}d^2 - \frac{6r+3}{2}d + \frac{9r^2+9r+2}{2},$$

for $d \gg 0$. However, we see that this is the exact value for $\dim S_d^r(\Delta, \partial\Delta)$ for all $d \geq 1$ by the computation in Example 4.2.1, which can be verified using `Macaulay2`. ◇

The following two examples will demonstrate the case of having two different embeddings of the same simplicial complex, first by reconsidering the crossed rectangle of example 4.2.2.

Example 4.5.2. We consider the case of the rectangle with a central vertex shown in Figure 4.5. In this example, the central vertex has four incident edges, but only two unique slopes incident. Each exterior vertex has three incident edges with three unique slopes, and so we will have to consider their contributions to the dimension of $S_d^r(\Delta, \partial\Delta)$ individually. Let $\gamma = \gamma_0$. Since there are only two incident slopes, we have seen the dimension of $J(\gamma_0)_d$ already as

$$\dim J(\gamma_0)_d = 2\binom{d-r+1}{2} - \binom{d-2r}{2}.$$

Now, since all external edges are indistinguishable from each other, let γ be any of $\gamma_1, \ldots, \gamma_4$. In Proposition 4.4.3, we have

$$t_\gamma = 3,\ q_\gamma = \left\lfloor \frac{3(r+1)}{2} \right\rfloor.$$

Now, the values of q_γ, a_γ and b_γ depend on the value of $r+1$ modulo 2, as this will dictate when $\frac{3(r+1)}{2}$ is an integer, and thus whether $a_\gamma = 0$ or not. We can see

$$t_\gamma = 3,\ q_\gamma = \begin{cases} \frac{3r+3}{2} & r+1 \equiv 0 \mod 2, \\ \frac{3r+2}{2} & r+1 \equiv 1 \mod 2, \end{cases}$$

$$a_\gamma = \begin{cases} 0 & r+1 \equiv 0 \mod 2, \\ 1 & r+1 \equiv 1 \mod 2, \end{cases} \quad b_\gamma = \begin{cases} 2 & r+1 \equiv 0 \mod 2, \\ 1 & r+1 \equiv 1 \mod 2. \end{cases}$$

We can then split $\dim J(\gamma)_d$ into two classes based on the value of $r+1$ mod 2:

$$\dim J(\gamma)_d = \begin{cases} 3\binom{d-r+1,}{2} - 2\binom{d-\frac{3r+3}{2}+2,}{2} & r+1 \equiv 0 \mod 2, \\ 3\binom{d-r+1}{2} - \binom{d-\frac{3r+2}{2}+1}{2} - \binom{d-\frac{3r+2}{2}+2}{2} & r+1 \equiv 1 \mod 2, \end{cases}$$

for d sufficiently large. Expanding these gives

$$\dim J(\gamma)_d = \begin{cases} \frac{1}{2}d^2 + \frac{3}{2}d - \frac{3r^2+6r-1}{4} & r+1 \equiv 0 \mod 2, \\ \frac{1}{2}d^2 + \frac{3}{2}d - \frac{3r^2+6r}{4} & r+1 \equiv 1 \mod 2, \end{cases}$$

Combining this, we get

$$\dim S_d^r(\Delta, \partial\Delta) = 8\binom{d-r+1}{2} + \binom{d+2}{2} - 2\binom{d-r+1}{2} + \binom{d-2r}{2} - 4\dim J(\gamma)_d,$$

wherein we can expand this and find

$$\dim S_d^r(\Delta, \partial\Delta) = \begin{cases} 2d^2 - (8r+2)d + 8r^2 + 4r & r+1 \equiv 0 \mod 2, \\ 2d^2 - (8r+2)d + 8r^2 + 4r + 1 & r+1 \equiv 1 \mod 2, \end{cases}$$

for sufficiently large d. In fact, recalling Example 4.2.2, setting $r = 0$ here recovers the formula $\dim S_d^r(\Delta, \partial\Delta) = 2d^2 - 2d + 1$ which we see from Example 4.2.2 gave the exact dimension for all $d \geq 1$. Computations in `Macaulay2` also confirm that $\dim S_d^r(\Delta, \partial\Delta)$ matches $Q(\Delta, r, d)$ for $d \geq r+1$ for $r \leq 2$. ◇

In Example 4.5.2 the vertex γ_0 is not in generic position, which is to say that if we were to move γ_0 slightly, then every edge incident to γ_0 would have different slopes. This dramatically changes the calculation of $\dim S_d^r(\Delta, \partial\Delta)$ if we do perturb the vertex, and Example 4.5.3 will demonstrate the difference between having vertices in generic versus non-generic positions.

Example 4.5.3. We consider the case of the rectangle with a central vertex shown in Figure 4.11, but now in generic position so that each incident edge has a unique slope. Each exterior vertex has three incident edges with three unique slopes, and we have already calculated their contributions to $\dim S_d^r(\Delta, \partial\Delta)$ in Example 4.5.2. We know already that

$$\dim J(\gamma_i)_d = \begin{cases} \frac{1}{2}d^2 + \frac{3}{2}d - \frac{3r^2+6r-1}{4} & r+1 \equiv 0 \mod 2, \\ \frac{1}{2}d^2 + \frac{3}{2}d - \frac{3r^2+6r}{4} & r+1 \equiv 1 \mod 2, \end{cases} \quad \text{for } i = 1, \ldots, 4,$$

and so we focus our attention now on γ_0. The central vertex has four unique incident slopes, and so we have

$$t_{\gamma_0} = 4, \; q_{\gamma_0} = \left\lfloor \frac{4(r+1)}{3} \right\rfloor.$$

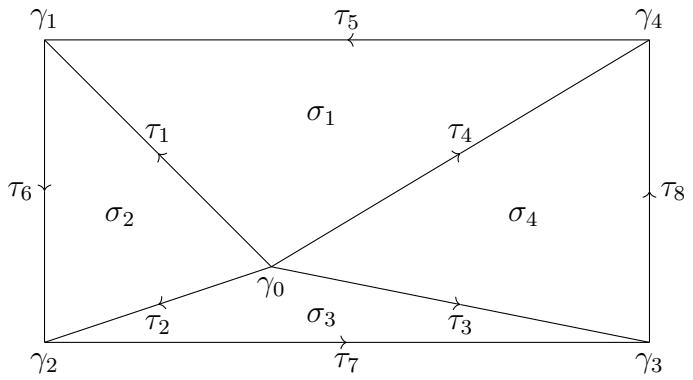

Figure 4.11: Rectangular simplicial complex with a central vertex Δ in generic position.

Now, the values of q_{γ_0}, a_{γ_0} and b_{γ_0} depend on the value of $r+1$ modulo 3, as this will dictate when $\frac{4(r+1)}{3}$ is an integer, and thus whether $a_{\gamma_0} = 0$ or not. We can see

$$t_{\gamma_0} = 4,\ q_{\gamma_0} = \begin{cases} \frac{4r+4}{3} & r+1 \equiv 0 \mod 3, \\ \frac{4r+3}{3} & r+1 \equiv 1 \mod 3, \\ \frac{4r+2}{3} & r+1 \equiv 2 \mod 3, \end{cases}$$

$$a_{\gamma_0} = \begin{cases} 0 & r+1 \equiv 0 \mod 3, \\ 1 & r+1 \equiv 1 \mod 3, \\ 2 & r+1 \equiv 2 \mod 3, \end{cases} \quad b_{\gamma_0} = \begin{cases} 3 & r+1 \equiv 0 \mod 3, \\ 2 & r+1 \equiv 1 \mod 3, \\ 1 & r+1 \equiv 2 \mod 3. \end{cases}$$

We can then split $\dim J(\gamma_0)_d$ into two classes based on the value of $r+1$ mod 3:

$$\dim J(\gamma_0)_d = \begin{cases} 4\binom{d-r+1}{2} - 3\binom{d-\frac{4r+4}{3}+2}{2} & r+1 \equiv 0 \mod 3, \\ 4\binom{d-r+1}{2} - \binom{d-\frac{4r+3}{3}+1}{2} - 2\binom{d-\frac{4r+3}{3}+2}{2} & r+1 \equiv 1 \mod 3, \\ 4\binom{d-r+1}{2} - 2\binom{d-\frac{4r+2}{3}+1}{2} - \binom{d-\frac{4r+2}{3}+2}{2} & r+1 \equiv 2 \mod 3, \end{cases}$$

for d sufficiently large. Expanding these gives

$$\dim J(\gamma_0)_d = \begin{cases} \frac{1}{2}d^2 + \frac{3}{2}d - \frac{2r^2+4r-1}{3} & r+1 \equiv 0 \mod 3, \\ \frac{1}{2}d^2 + \frac{3}{2}d - \frac{2r^2+4r}{3} & r+1 \equiv 1 \mod 3, \\ \frac{1}{2}d^2 + \frac{3}{2}d - \frac{2r^2+4r}{3} & r+1 \equiv 2 \mod 3. \end{cases}$$

While we know the values of $J(\gamma_0)_d$ in terms of the value of $r+1$ module 3, we know the values of $J(\gamma_i)_d$ for $i = 1, \ldots, 4$ in terms of the value of $r+1$ modulo 2. When combining these into an equation for $\dim S_d^r(\Delta, \partial\Delta)$, we therefore need to consider all possible values of $r+1$ modulo 6. Doing so, we see that

$$\sum_{\gamma\in\Delta_0} \dim J(\gamma)_d = \begin{cases} \frac{5}{2}d^2 + \frac{15}{2}d - \frac{11r^2+22r-4}{3} & r+1 \equiv 0 \mod 6, \\ \frac{5}{2}d^2 + \frac{15}{2}d - \frac{11r^2+22r}{3} & r+1 \equiv 1 \mod 6, \\ \frac{5}{2}d^2 + \frac{15}{2}d - \frac{11r^2+22r-3}{3} & r+1 \equiv 2 \mod 6, \\ \frac{5}{2}d^2 + \frac{15}{2}d - \frac{11r^2+22r-1}{3} & r+1 \equiv 3 \mod 6, \\ \frac{5}{2}d^2 + \frac{15}{2}d - \frac{11r^2+22r-3}{3} & r+1 \equiv 4 \mod 6, \\ \frac{5}{2}d^2 + \frac{15}{2}d - \frac{11r^2+22r}{3} & r+1 \equiv 5 \mod 6. \end{cases}$$

Combining this, we get

$$\dim S_d^r(\Delta, \partial\Delta) = 8\binom{d-r+1}{2} + \binom{d+2}{2} - \sum_{\gamma\in\Delta_0} \dim J(\gamma)_d,$$

wherein we can expand this and find

$$\dim S_d^r(\Delta, \partial\Delta) = \begin{cases} 2d^2 - (8r+2)d + \frac{23r^2+10r-1}{3} & r+1 \equiv 0 \mod 6, \\ 2d^2 - (8r+2)d + \frac{23r^2+10r+3}{3} & r+1 \equiv 1 \mod 6, \\ 2d^2 - (8r+2)d + \frac{23r^2+10r}{3} & r+1 \equiv 2 \mod 6, \\ 2d^2 - (8r+2)d + \frac{23r^2+10r+2}{3} & r+1 \equiv 3 \mod 6, \\ 2d^2 - (8r+2)d + \frac{23r^2+10r}{3} & r+1 \equiv 4 \mod 6, \\ 2d^2 - (8r+2)d + \frac{23r^2+10r+3}{3} & r+1 \equiv 5 \mod 6. \end{cases}$$

for sufficiently large d. We may note here that the only difference between the dimension of $S_d^r(\Delta, \partial\Delta)$ in this example, and the corresponding dimension in Example 4.5.2, is the coefficient of d^0. If we call Δ to be the simplicial complex in Figure 4.11, and Δ' to be the simplicial complex in Figure 4.5, then the difference between the case of having generic vertex positions and non-generic vertex positions manifests itself as

$$\dim S_d^r(\Delta', \partial\Delta') - \dim S_d^r(\Delta, \partial\Delta) = \begin{cases} \frac{r^2+2r+1}{3} & r+1 \equiv 0 \mod 3, \\ \frac{r^2+2r}{3} & r+1 \equiv 1 \mod 3, \\ \frac{r^2+2r}{3} & r+1 \equiv 2 \mod 3. \end{cases}$$

This difference highlights the effect of having generic vertex positions compared to non-generic vertex positions. ◇

In Examples 4.5.2 and 4.5.3, we computed the spline spaces of two distinct embeddings of the same simplicial complex. That the dimensions of their spline spaces for sufficiently large d are not equal was remarked by Strang in [75], and showed that the dimension of the spline space of a simplicial complex is cannot in general be solely determined by topological data, such as the number of vertices, edges, faces in the simplicial complex.

Example 4.5.4. We will be considering the following simplicial complex in Figure 4.12. All vertices of this simplicial complex are indistinguishable, as they all have four

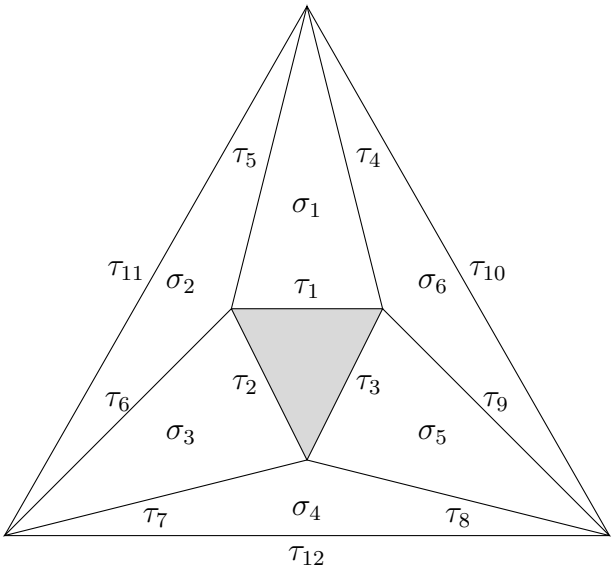

Figure 4.12: Simplicial complex Δ that has a single hole. The grey shaded triangle is not a face of Δ, it is the hole.

distinct edges incident, each with unique slopes. We have computed the dimension of the ideals of vertices with this property in Example 4.5.3, and so we simply recall that for each vertex γ in Δ_0, we have

$$\dim J(\gamma)_d = \begin{cases} \frac{1}{2}d^2 + \frac{3}{2}d - \frac{2r^2+4r-1}{3} & r+1 \equiv 0 \mod 3, \\ \frac{1}{2}d^2 + \frac{3}{2}d - \frac{2r^2+4r}{3} & r+1 \equiv 1 \mod 3, \\ \frac{1}{2}d^2 + \frac{3}{2}d - \frac{2r^2+4r}{3} & r+1 \equiv 2 \mod 3. \end{cases}$$

For this example, we have

$$|\Delta_0| = 6,\ |\Delta_1| = 12,\ |\Delta_2| = 6,$$

wherein we find the dimension is

$$\dim S_d^r(\Delta, \partial\Delta) = 12\binom{d-r+1}{2} - 6\dim J(\gamma)_d$$
$$= \begin{cases} 3d^2 - (12r+3)d + 10r^2 + 2r - 2 & r+1 \equiv 0 \mod 3, \\ 3d^2 - (12r+3)d + 10r^2 + 2r & r+1 \equiv 1 \mod 3, \\ 3d^2 - (12r+3)d + 10r^2 + 2r & r+1 \equiv 2 \mod 3. \end{cases}$$

If we fill the hole to form a new simplicial complex Δ' shown in Figure 4.13. We

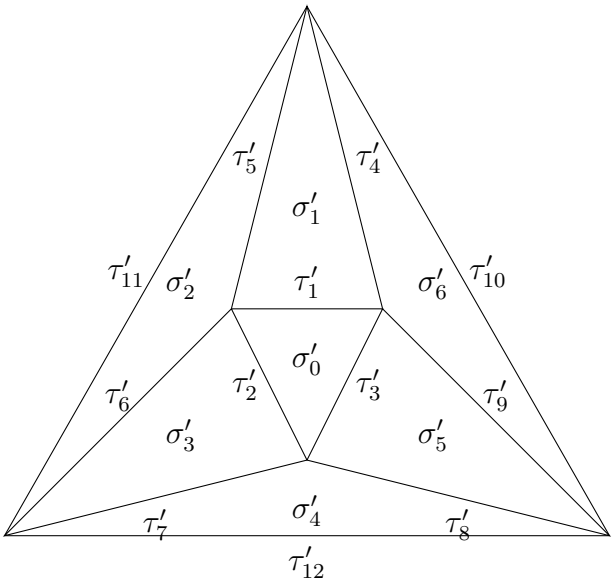

Figure 4.13: Simplicial complex Δ' that is Δ with no hole.

can compute the dimension of the spline space in a similar way, but notice that Δ' differs from Δ by adding a single face, and so we can see that

$$Q(\Delta', r, d) = Q(\Delta, r, d) + \binom{d+2}{2},$$

and we can notice this by recalling that $|\Delta_0| - |\Delta_1| + |\Delta_2| = c - h$ by the Euler-Poincaré characteristic in Proposition 4.3.1, the number of connected components minus the number of holes of Δ. Thus, reducing the number of holes by 1 (by filling it with a single simplex) will increase the number of splines of degree d by $\binom{d+2}{2}$.

$\diamond$

4.6 Subcomplexes and Splines with Partial Boundary Conditions

In this section we describe a method of gluing simplicial complexes. We begin by defining the notation for the space of splines that satisfy boundary conditions on some (but not necessarily all) of its boundary edges. Then, we create an injective map from the space of splines over two subcomplexes of a simplicial complex satisfying partial boundary conditions, and the space of splines of the simplicial complex that glues the two subcomplexes together along the edges that satisfy boundary conditions. Given a pure, hereditary simplicial complex Δ embedded in $\mathbb{R}^2$, let $B \subseteq \partial\Delta$ be (possibly proper) collection of boundary edges of Δ and their vertices. We denote the edges in B by $\tau_1, \ldots, \tau_s$ if $B \neq \emptyset$. We write

$$S^r(\Delta, B) = \left\{ f \in C^r(\Delta) : \begin{array}{c} f|_\sigma \in \mathbb{R}[x,y] \text{ for every face } \sigma \in \Delta_2 \\ f \text{ vanishes to order } r \text{ on every boundary edge } \tau \in B \end{array} \right\},$$

to be the set of splines defined on Δ satisfying boundary conditions at B. We can clearly see the following

- If $B = \partial\Delta$, then $S^r(\Delta, B)$ is consistent with the definition of $S^r(\Delta, \partial\Delta)$.
- The set $S^r(\Delta, \emptyset)$ is the set of splines defined on Δ satisfying no boundary conditions, and this is equal to the space usually written $S^r(\Delta)$, the set of splines defined on Δ.

Recall the algebraic spline criterion, Proposition 4.2.2. If we impose boundary conditions on B instead of $\partial\Delta$, then we can see that a function $f\colon \Delta \to \mathbb{R}$ is in $S^r(\Delta, B)$ if and only if the first two conditions of Proposition 4.2.2 hold, and if for each boundary face σ with boundary edges $\tau_{i_1}, \ldots, \tau_{i_m} \in B$, we have $f|_\sigma \in \bigcap_{a=1}^m \langle \ell_{\tau_{i_a}}^{r+1} \rangle$, where of course m must be 1, 2, or 3 as before. As a result, we can define a matrix $A(\Delta, B, r)$ as follows: Order all edges of Δ to be $\tau_1, \ldots, \tau_s$ for some $s \geq 3$ such that the final b edges are the boundary edges not in B. Then, define $A(\Delta, B, r)$ to be the submatrix of $A(\Delta, \partial\Delta, r)$ given by removing the final b rows and columns of $A(\Delta, \partial\Delta, r)$. The ith row of $A(\Delta, \partial\Delta, r)$ refers to the algebraic condition satisfied by a spline across the edge τ_i, and thus by removing the rows and columns corresponding to the boundary edges not in B, we see that if we label the maximal simplices $\sigma_1, \ldots, \sigma_t$, then the first t entries of elements in the kernel of $A(\Delta, B, r)$ correspond to the splines in $S^r(\Delta, B, r)$.

Example 4.6.1. Consider the simplicial complex shown in Figure 4.4 in Example 4.2.1. Suppose we impose boundary conditions only on a single edge τ_1. As such, we set $B = \{\tau_1\}$ and we see that

$$A(\Delta, B, r) = \left(\begin{array}{c|c} 1 & \ell_{\tau_1}^{r+1} \end{array} \right).$$

The kernel of $A(\Delta, B, r)$ has a basis given by

$$\begin{pmatrix} \ell_{\tau_1}^{r+1} \\ -1 \end{pmatrix},$$

and so a spline on Δ satisfying boundary conditions on τ_1 is a polynomial multiple of $\ell_{\tau_1}^{r+1}$. Thus,

$$\dim S_d^r(\Delta, B) = \binom{d-r+1}{2} = \begin{cases} \frac{1}{2}d^2 - (r - \frac{1}{2})d + \frac{r^2}{2} - \frac{r}{2} & d \geq r+1, \\ 0 & d \leq r. \end{cases}$$

◇

Splines with partial boundary conditions are a useful class of splines. We can observe this by considering a pair of simplicial complexes, say Λ and Λ' shown below in Figure 4.14, embedded so that they share a common edge (shown in orange): Let $r \geq 0$ be an integer. Then, suppose $\ell_\tau = x$. We can see that any spline f on

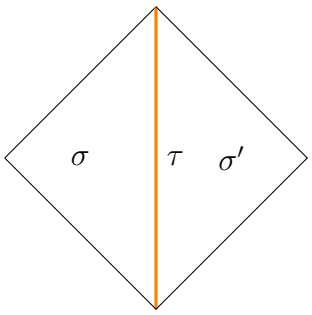

Figure 4.14: Two simplicial complexes Λ and Λ' with maximal simplices σ and σ' respectively, with a shared edge τ.

Λ that satisfies boundary conditions on τ is a multiple of x^{r+1} for smoothness r. Additionally, any spline g on Λ' that satisfies boundary conditions on τ is a multiple of x^{r+1} as well. So, we can see that if we define a spline h on the union of Λ and Λ' by $h|_\sigma = f$ and $h|_{\sigma'} = g$, then automatically $h|_\sigma - h|\sigma'$ is a multiple of x^{r+1}, and so h satisfies the algebraic spline criterion. This suggests that splines on simplicial complexes can be built up using splines on subcomplexes. Note that this is similar

to the process of gluing spaces of splines together, and the concept of geometric continuity for splines [55].

We state this observation formally in Proposition 4.6.1. If two simplicial complexes embedded in $\mathbb{R}^2$ share some part of their boundaries, then we can consider them as subcomplexes of their union. Alternatively, we can describe this by first defining a simplicial complex Δ, and then writing it as the union of two subcomplexes Λ and Λ' that intersect only along their boundaries. We can do so by creating a subset of the maximal simplices of Δ and declaring them to be the maximal simplices of Λ, and then the remaining maximal simplices of Δ are declared to be the maximal simplices of Λ'. We then make the definition of Λ and Λ' complete by taking all edges and vertices of each maximal simplex of Λ or Λ' to be a part of Λ or Λ' respectively. In order to construct splines on Λ and Λ' however, we should insist that they are pure and hereditary, which limits which maximal simplices we can choose to be part of Λ or Λ'. By their construction, $\Lambda \cap \Lambda'$ is then a subcomplex of Δ of dimension at most 1. This is dimension exactly 1 when both Λ and Λ' are non-empty. Additionally, if we want splines on Λ and Λ' to combine to create a spline on Δ, then any boundary conditions that spines must satisfy on Δ must also be satisfied by the splines on Λ and Λ'.

Proposition 4.6.1. *Let Δ be a pure, hereditary, 2-dimensional simplicial complex. Let $F \subseteq \Delta_2$ be a (possibly proper) collection of 2-simplices in Δ. Let Λ be the simplicial complex embedded in $\mathbb{R}^2$ with simplices in F as well as their edges and vertices. Let Λ' be the simplicial complex embedded in $\mathbb{R}^2$ with simplices in $\Delta_2 \setminus F$ as well as their edges and vertices. Let $B \subseteq \partial\Delta$ be a (possibly proper) collection of boundary edges of Δ and their vertices. Let $F \subseteq \Delta_2$ be such that Λ and Λ' are, pure, hereditary, and 2-dimensional. Let*

$$B_\Lambda = \partial\Lambda \cap (\partial\Lambda' \cup B) \text{ and } B_{\Lambda'} = \partial\Lambda' \cap (\partial\Lambda \cup B).$$

There is a graded $\mathbb{R}$-algebra homomorphism

$$\iota \colon S^r(\Lambda, B_\Lambda) \oplus S^r(\Lambda', B_{\Lambda'}) \to S^r(\Delta, B),$$

defined by $\iota(f,g)|_\sigma = f|_\sigma$ when $\sigma \in \Lambda$, and $\iota(f,g)|_{\sigma'} = g|_{\sigma'}$ when $\sigma' \in \Lambda'$ that is well-defined and injective.

Proof. Suppose we are given $(f,g) \in S^r(\Lambda, B_\Lambda) \oplus S^r(\Lambda', B_{\Lambda'})$. Let $\sigma \in \Lambda$ be a maximal simplex, then $\iota(f,g)|_\sigma = f|_\sigma \in \mathbb{R}[x,y]$. Then for two maximal simplices $\sigma_i, \sigma_j \in \Lambda$

intersecting along a non-empty edge $\tau = \sigma_i \cap \sigma_j \in \Lambda$, we have, since $f \in S^r(\Lambda, B_\Lambda)$, that $f|_{\sigma_i}$ and $f|_{\sigma_j}$ join along τ with smoothness r. Now, suppose $\sigma \in \Lambda$ and $\sigma' \in \Lambda'$ intersect along an edge $\tau \in \partial\Lambda \cap \partial\Lambda'$. We know that $f|_\sigma \in J(\tau), g|_{\sigma'} \in J(\tau)$, and so $f|_\sigma - g|_{\sigma'} \in J(\tau)$. Thus, for any maximal simplices σ_i, σ_j in Δ intersecting at a non-empty edge $\tau = \sigma_i \cap \sigma_j$, then $\iota(f,g)|_{\sigma_i}$ and $\iota(f,g)|_{\sigma_j}$ join along τ with smoothness r. Let $\tau \in B \cap \partial\Lambda$, and σ the maximal simplex with this edge on its boundary. Then $f|_\sigma \in J(\tau)$ and so $\iota(f,g)|_\sigma \in J(\tau)$. Replacing every instance of Λ with Λ' and vice versa in the above argument then completes the proof that $\iota(f,g) \in S^r(\Delta, B)$.

That the map ι is a graded $\mathbb{R}$-algebra homomorphism is then clear from its definition.

It remains only to show that ι is injective. To see this, suppose $\iota(f,g) = 0$. Then, $f|_\sigma = 0$ and $g|_{\sigma'} = 0$ for all maximal simplices $\sigma, \sigma' \in \Lambda, \Lambda'$. Thus, $(f,g) = (0,0)$, and ι is injective. □

Notice that B_Λ and $B_{\Lambda'}$ are the boundary edges where boundary conditions are satisfied by splines on Δ that are also in the boundary of Λ (resp. Λ') as well as every edge along the shared boundary of Λ and Λ'. Insisting on boundary conditions here ensures that when the two splines are combined, they meet along the shared boundary with smoothness r, and they also satisfy the required boundary conditions on Δ as needed.

Since ι is injective, there is a subspace of $S^r(\Delta, B)$ isomorphic to $S^r(\Lambda, B_\Lambda) \oplus S^r(\Lambda', B_{\Lambda'})$, and since ι is a graded map,

$$\dim S^r_d(\Lambda, B_\Lambda) + \dim S^r_d(\Lambda', B_{\Lambda'}) \leq \dim S^r_d(\Delta, B),$$

for each $d \geq 0$.

Remark 4.6.1. Notice for each pure, hereditary simplicial complex Δ embedded in $\mathbb{R}^2$, and integers $r, d \geq 0$, the quantity $Q(\Delta, r, d)$ is a polynomial in d of degree 2. The coefficient of d^2 is then given by

$$\frac{|\Delta_1|}{2} + \frac{|\Delta_0| - |\Delta_1| + |\Delta_2|}{2} - \sum_{\gamma \in \Delta_0} \left(\frac{t_\gamma}{2} - \frac{a_\gamma}{2} - \frac{b_\gamma}{2} \right) = \frac{|\Delta_2|}{2},$$

since $a_\gamma + b_\gamma = t_\gamma - 1$. If we let $B = \partial\Delta$, and choose some $F \subseteq \Delta_2$ such that Λ and Λ' are such as in Proposition 4.6.1, then $B_\Lambda = \partial\Lambda$, $B_{\Lambda'} = \partial\Lambda'$, and the coefficient of d^2 in $Q(\Lambda, r, d) + Q(\Lambda', r, d)$ is given by $\frac{|\Lambda_2|}{2} + \frac{|\Lambda'_2|}{2} = \frac{|\Delta_2|}{2}$. Therefore, the difference between $\dim S^r_d(\Delta, \partial\Delta)$ and $\dim \left(S^r_d(\Lambda, \partial\Lambda) \oplus S^r_d(\Lambda', \partial\Lambda')\right)$ is a polynomial of degree 1 in d for sufficiently large d.

We now demonstrate the use of Proposition 4.6.1 in some examples. Throughout, let a smoothness $r \geq 0$ be given.

Example 4.6.2. Recall the simplicial complex Δ in Figure 4.14. The edge τ is associated to the ideal $J(\tau) = \langle x^{r+1} \rangle$. In the notation of Proposition 4.6.1, we define $F = \{\sigma_1\}$, so that Λ is the simplicial complex with 2-simplex σ_1 and its edges and vertices. Then Λ' is the simplicial complex with 2-simplex σ_2 and its edges and vertices. Then, suppose $B = \emptyset$, and thus $B_\Lambda = B'_\Lambda = \{\tau\}$. We can compute the dimension of $S^r_d(\Lambda, B_\Lambda)$ and of $S^r_d(\Lambda', B_{\Lambda'})$ as shown in Example 4.6.1 to get

$$\dim S^r_d(\Lambda, B_\Lambda) = \dim S^r_d(\Lambda', B_{\Lambda'}) = \binom{d-r+1}{2},$$

for $d \geq 0$. The injectivity of the map ι shows that

$$\dim S^r_d(\Delta, \emptyset) \geq \begin{cases} d^2 - (2r-1)d + r^2 - r & d \geq r+1, \\ 0 & d \leq r. \end{cases}$$

Using `Macaulay2`, we compute the dimension $S^r_d(\Delta, \emptyset)$ as

$$\dim S^r_d(\Delta, \emptyset) = \begin{cases} d^2 - (r-2)d + \frac{r^2}{2} - \frac{r}{2} + 1 & d \geq r+1, \\ \frac{1}{2}d^2 + \frac{3}{2}d + 1 & d \leq r. \end{cases}$$

We can see that splines on Λ and Λ' satisfying boundary conditions on τ are multiples of x^{r+1}, and thus by Proposition 4.6.1, we see that for every polynomials $f, g \in \mathbb{R}[x, y]$, the function h defined on Δ satisfying $h|_{\sigma_1} = x^{r+1}f$ and $h|_{\sigma_2} = x^{r+1}g$ is a spline over Δ satisfying no boundary conditions. $\diamond$

Example 4.6.3. Consider the simplicial complex Δ' in Example 4.5.4, shown in Figure 4.13. Suppose we choose $F = \{\sigma'_1, \sigma'_2, \sigma'_3, \sigma'_4, \sigma'_5, \sigma'_6\}$ in the notation of Proposition 4.6.1. Then, Λ is the simplicial complex with 2-simplices $\sigma'_1, \sigma'_2, \sigma'_3, \sigma'_4, \sigma'_5, \sigma'_6$, as well as their edges and vertices and Λ' is the simplicial complex with single 2-simplex σ'_0, as well as its edges and vertices. We know that both Λ and Λ' are pure, hereditary and 2-dimensional, and we notice, in fact, that Λ is the simplicial complex Δ shown in Figure 4.12. Suppose we apply boundary conditions on all edges of Δ', so $B = \partial\Delta'$. Then, we have $B_\Lambda = \partial\Lambda$ and $B_{\Lambda'} = \partial\Lambda'$. We know from Theorem 4.4.4, Example 4.2.1, and the formulae in Example 4.5.4, that

$$\dim S^r_d(\Lambda, \partial\Lambda) = Q(\Lambda, r, d), \dim S^r_d(\Lambda', \partial\Lambda') = \binom{d-3r-1}{2},$$

for $d \gg 0$. We then see that

$$\dim S_d^r(\Delta', \partial\Delta') \geq Q(\Lambda, r, d) + \binom{d-3r-1}{2},$$

by Proposition 4.6.1. We also know from Example 4.5.4 that

$$\dim S_d^r(\Delta', \partial\Delta') = Q(\Lambda, r, d) + \binom{d+2}{2},$$

for $d \gg 0$, and that the difference between the dimensions of $S_d^r(\Delta', \partial\Delta')$ and $S_d^r(\Lambda, \partial\Lambda) \oplus S_d^r(\Lambda', \partial\Lambda')$ is then

$$Q(\Lambda, r, d) + \binom{d+2}{2} - Q(\Lambda, r, d) - \binom{d-3r-1}{2} = (3r+3)d - \frac{9}{2}r^2 - \frac{9}{2}r,$$

for $d \gg 0$, a degree 1 polynomial in d as expected. ◇

4.7 Concluding remarks

Splines with boundary conditions are ubiquitous. They appear in fundamental numerical analysis techniques for solving partial differential equations, and ways to construct bases for the spaces of these splines are fundamentally important. Even in the study of splines, the application of algebraic methods to splines with boundary conditions is minimal, suggesting there is still much that can still be done in future research. Within this chapter we adapted homological techniques from the wider study of splines to find a formula for the dimension of the space of splines with boundary conditions for sufficiently large d embedded in $\mathbb{R}^2$. The scope of the work means that the question of what happens when increasing the dimension of the simplicial complexes is still open. We also leave open the question of what happens when the conditions of pure and hereditary are potentially weakened. The existence of strict boundary conditions may allow for these assumptions to be relaxed somewhat, but we leave this as a potential area of future research.

We explored the question of what is the dimension of the space of splines of degree d with boundary conditions over pure and hereditary simplicial complexes embedded in $\mathbb{R}^2$. We have answered this question by giving an exact value for $\dim S_d^r(\Delta, \partial\Delta)$ for sufficiently large degrees using tools from algebraic topology, adapting the chain complex of Billera, Schenck and Stillman. We have employed combinatorial results

about the dimensions of the graded pieces of ideals in $\mathbb{R}[x, y, z]$, and have been able to demonstrate the efficacy of the formula for some example simplicial complexes.

This work suggests different areas future research. What we have presented is a formula that is exact for $d \gg 0$, but the natural question can be asked of what values of d does this formula hold for. For example, in [71], Schumaker proves the lower bound for the dimension of the space of splines of degree d without boundary conditions over a certain class of triangulations of a rectangle matches the exact dimension for all $d > r$ if $r = 0, 1, 2$, as well as proving a similar formula for a partitions known as cross-cut partitions of a subset of $\mathbb{R}^2$ that holds for $d \geq 2r + 1$. A similar question could be asked here, for what values of d do $\dim S_d^r(\Delta, \partial\Delta)$ and $Q(\Delta, r, d)$ coincide? Also, it is not known when if ever $Q(\Delta, r, d)$ is an upper or lower bound, and the question one could as is for which values of d is this the case? Both of these questions could be answered by obtaining an exact value for $\dim H_0(\mathcal{R}/\mathcal{J})_d$, for $\dim H_1(\mathcal{R}/\mathcal{J})$, or both, as if we know the first, we obtain the values of d for which $Q(\Delta, r, d)$ is an upper bound, and knowing the other gives the values of d for which $Q(\Delta, r, d)$ is a lower bound. Knowing both would give us the exact values of d where both vanish, and thus would give us the degrees where $Q(\Delta, r, d)$ is equal to $\dim S_d^r(\Delta, \partial\Delta)$.

A slight modification to the definition of $S_d^r(\Delta, \partial\Delta)$ is to impose the condition that splines are "flat" on the boundary, in the sense that the splines do not vanish on the boundary, but it's derivative does, instead of imposing the condition that splines vanish on the boundary. Many systems of partial differential equations impose "flat" boundary conditions, and splines with flat boundary conditions may be a way to approximate the solutions to these partial differential equations.

We could also attempt to analyse the case where we have splines with higher smoothness on interior edges, with the condition that splines only vanish on the boundary with smoothness $< r$. This gives a, what is known as, supersmoothness condition on the interior of the simplicial complex. The proposed methodology maps readily into such a framework, but the computation of the dimension is likely to be a more involved task.

We extended the definition of splines with boundary conditions to splines with partial boundary conditions. By partitioning a simplicial complex Δ into two subcomplexes, we proved a bound for the dimension of the space of splines of Δ from below given by the sum of the dimensions of the spaces of splines over the subcomplexes satisfying boundary conditions along the border between them.

In Remark 4.6.1, we noted that if $\Delta, \Lambda, \Lambda'$ are as in Proposition 4.6.1, then for $d \gg 0$ the difference between the dimension of $S^r_d(\Delta, \partial\Delta)$ and the dimension of $S^r(\Lambda, \partial\Lambda)_d \oplus S^r(\Lambda', \partial\Lambda')_d$ is a polynomial of degree 1 in d. We may consider the question of quantifying this difference further, for instance can we find a space that is isomorphic to $S^r(\Delta, \partial\Delta)$ that can be written as a direct sum of spline spaces and ideals in $\mathbb{R}[x, y]$. Such an isomorphism could then be used to compute the dimension and bases of spline spaces over pure, hereditary, 2-dimensional simplicial complexes iteratively, possibly providing a divide-and-conquer style algorithm for finding a basis for the space of splines over a pure, hereditary, 2-dimensional simplicial complex.

Chapter 5

Conclusions and and Future Work

In this thesis, we have applied methods from algebraic geometry and topology to the fields of control theory and splines. In Chapter 3 we focused our efforts on nilpotent control problems, where we derived the system of polynomial equations (3.2.4). We created a neural network method for binary classification, taking as input an initial condition of the control problem, and outputting the value of the optimal control at time $t = 0$. With this, we obtained a feedback control that was robust to perturbations. The method is demonstrated for chain of integrator systems in dimensions up to 5; to improve the stability of the algorithm in higher dimensions we may look to use a different root finding algorithm as we know that Newton's method can sometimes fail to find the roots of a system of equations.

We may look to expand the usage of the method of neural networks to other control problems where the switching times can be derived from the roots of a system of polynomial equations. Examples of such systems can be found in [61] and [62].

In Chapter 4 we derived the formula (4.4.3) for the dimension of $S^r_d(\Delta, \partial\Delta)$ for a pure, hereditary simplicial complex Δ embedded in $\mathbb{R}^2$, with $d, r \geq 0$ integers, the splines of degree at most d and smoothness r satisfying boundary conditions (vanishing with smoothness r on $\partial\Delta$). Future research might look to find formulae for the dimensions of spaces of splines satisfying partial boundary conditions. Additionally, one could consider splines that match the value of a fixed polynomial on the boundary of Δ, instead of merely vanishing.

We suggest a future area to look into is the construction of a basis of $S^r_d(\Delta, \partial\Delta)$. Furthermore, one could aim to compute the dimension of $S^r_d(\Delta, \partial\Delta)$ for small degrees d.

Appendices

Appendix A

In this appendix, we display the total number of iterations for Algorithm 2, Newton's method with deflation, to converge when finding the solutions of the polynomial systems in sections 3.7.1, 3.7.2, 3.7.3, and 3.7.4. For each dataset, we run Newton's method with deflation on each data point, and log the number of iterations taken for the algorithm to converge if the algorithm successfuly converged. The "All" row lists the average number of iterations required for convergence for $n = 2, \ldots, 5$. We additionally log the average number of iterations required for convergence after each successive deflation in the subsequent rows. We show only rows for which we have data. All entries are written to 4 decimal places.

Deflation Number	$n = 2$	$n = 3$	$n = 4$	$n = 5$
All	19.6721	54.2912	879.9081	7465.5006
0	6.4542	37.6695	121.0836	216.5999
1	33.4720	98.9743	352.9485	394.2635
2		498.2727	235.0756	579.9901
3		252.0000	603.6256	756.8997
4			523.2417	487.5127
5			525.2022	535.2663
6			570.8972	599.4868
7			848.3295	793.9409
8			881.2142	1184.8344
9			843.9108	1969.8031
10			842.8357	3366.0184
11			1021.3409	5590.6281
12			1101.9209	8963.4740
13			1196.9325	13221.7239
14			1276.5889	18850.0609
15			1418.9785	25957.1046
16			1609.5288	33437.6267
17			1781.5641	40411.1383
18			2121.1343	47206.3865
19			2506.8267	53473.1065
20			2910.4040	58572.4845
21			3412.8474	62772.8934
22			4009.0160	65096.2746
23			4824.6132	67796.8419
24				71359.1224
25				74815.6432
26				76897.9555
27				77955.5455
28				76364.6364
29				76667.6667
30				75001.0000
31				80001.0000

Bibliography

[1] A. Agrachev. Introduction to optimal control theory. In *Mathematical control theory, Part 1, 2 (Trieste, 2001)*, ICTP Lecture Notes, VIII, pages 453–513. Abdus Salam International Center for Theoretical Physics, Trieste, 2002.

[2] G. Albi, S. Bicego, M. Herty, Y. Huang, D. Kalise, and C. Segala. Data/moment-driven approaches for fast predictive control of collective dynamics, 2024. URL: https://arxiv.org/abs/2402.15611, arXiv:2402.15611.

[3] M. Anderle and S. Čelikovský. Chain of four integrators as a possible essence of the under-actuated planar walking. *IFAC-PapersOnLine*, 54(14):60–65, 2021. 3rd IFAC Conference on Modelling, Identification and Control of Nonlinear Systems MICNON 2021.

[4] M. Atiyah and I. MacDonald. *Introduction to Commutative Algebra*. Addison-Wesley Publishing Company, 1 edition, 1969.

[5] B. Azmi, D. Kalise, and K. Kunisch. Optimal feedback law recovery by gradient-augmented sparse polynomial regression. *Journal of Machine Learning Research*, 22(48):1–32, 2021.

[6] S. Basu, R. Pollack, and M. Roy. *Algorithms in Real Algebraic Geometry*. Algorithms and Computation in Mathematics. Springer Berlin, Heidelberg, second edition, 2006.

[7] R. Bellman, I. Glicksberg, and O. Gross. On the "bang-bang" control problem. *Quarterly of Applied Mathematics*, 14(1):11–18, 1956.

[8] M. Ben-Or, D. Kozen, and J. Reif. The complexity of elementary algebra and geometry. *Journal of computer and System Sciences*, 32:251–264, 1986.

[9] S. Bicego, S. Gue, D. Kalise, and N. Villamizar. Time-optimal neural feedback control of nilpotent systems as a binary classification problem, 2025. URL: https://arxiv.org/abs/2503.17581, arXiv:2503.17581.

[10] S. Bicego, D. Kalise, and G. A. Pavliotis. Computation and control of unstable steady states for mean field multiagent systems, 2024. arXiv:2406.11725. URL: https://arxiv.org/abs/2406.11725, arXiv:2406.11725.

[11] L. Billera. Homology of smooth splines: generic triangulations and a conjecture of Strang. *Transactions of the American Mathematical Society*, 310(1):325–340, 1988.

[12] L. Billera and L. Rose. A dimension series for multivariate splines. *Discrete & Computational Geometry*, 6:107–128, 1991.

[13] L. Billera and L. Rose. Modules of piecewise polynomials and their freeness. *Mathematische Zeitschrift*, 209:485–497, 1992.

[14] B. Bonnard, O. Cots, Y. Privat, and E. Trélat. Zermelo navigation on the sphere with revolution metrics. In *Ivan Kupka Legacy: A Tour Through Controlled Dynamics*, volume 12 of *AIMS Applied Math Books - Special issue in honor of I. Kupka*, pages 35–65. 2024.

[15] H. Brezis. *Functional Analysis, Sobolev Spaces and Partial Differential Equations*. Universitext. Springer New York, 1 edition, 2010.

[16] K. Brown and W. Gearhart. Deflation techniques for the calculation of further solutions of a nonlinear system. *Numerische Mathematik*, 16:334–342, 1971.

[17] B. Buchberger. *An algorithm for finding the basis elements of the residue class ring of a zero dimensional polynomial ideal*. Phd thesis, Leopold-Franzens University, Innsbruck, 1965. English translation by M. Abramson.

[18] D. Bushaw. *Differential Equations with a Discontinuous Forcing Term*. Phd thesis, Princeton University, 1953.

[19] E. Charalampidis, N. Boullé, P. Farrell, and P. Kevrekidis. Bifurcation analysis of stationary solutions of two-dimensional coupled Gross–Pitaevskii equations using deflated continuation. *Communications in Nonlinear Science and Numerical Simulation*, 87:105255, 2020.

[20] C. Chen and C. Desoer. A proof of controllability of Jordan form state equations. *IEEE Transactions on Automatic Control*, AC-13:195–196, 1968.

[21] C. Chui and R. Wang. On smooth multivariate spline functions. *Mathematics of Computation*, 41(163):131–142, 1983.

[22] C. Cipriani, A. Scagliotti, and T. Wöhrer. A minimax optimal control approach for robust neural odes. In *2024 European Control Conference (ECC)*, pages 58–64, 2024.

[23] C. Cohen. Formalization of a sign determination algorithm in real algebraic geometry. Preprint, June 2021. URL: https://inria.hal.science/hal-03274013.

[24] A. Collin, G. Sangalli, and T. Takacs. Analysis-suitable G^1 multi-patch parametrizations for C^1 isogeometric spaces. *Computer Aided Geometric Design*, 47:93–113, 2016.

[25] G. Costante, F. Ferrante, and M. Leomanni. Time-optimal control of a multi-dimensional integrator chain with applications. *IEEE Control Systems Letters*, 6:2371–2376, 2022.

[26] J. Cottrell, T. Hughes, and Y. Bazilevs. *Isogeometric Analysis*. John Wiley and Sons, Ltd., 1 edition, 2009.

[27] R. Courant. Variational methods for the solution of problems of equilibrium and vibrations. *Bulletin of the American Mathematical Society*, 49(1):1–23, 1943.

[28] D. Cox. Stickelberger and the eigenvalue theorem, 2020. arxiv:2007.12573. URL: https://arxiv.org/pdf/2007.12573, arXiv:2007.12573.

[29] D. Cox, J. Little, and D. O'Shea. *Using Algebraic Geometry*. Graduate Texts in Mathematics. Springer New York, second edition, 2005.

[30] D. Cox, J. Little, and D. O'Shea. *Ideals, varieties, and algorithms*. Undergraduate Texts in Mathematics. Springer, Cham, fourth edition, 2015. An introduction to computational algebraic geometry and commutative algebra.

[31] J. De Gua. Démostrations de la régle de descartes, pour connoître le nombre des racines positives & négatives dans les équations qui n'ont point de racines imaginaires. *Histoire de l'Académie royale des sciences avec les mémoires de*

mathématique & de physique tirez des registres de cette Académie, pages 72–96, 1741.

[32] M. DiPasquale and N. Villamizar. A lower bound for splines on tetrahedral vertex stars. *SIAM Journal of Applied Algebra and Geometry*, 5(2):250–277, 2021.

[33] S. Dolgov, D. Kalise, and L. Saluzzi. Data-driven tensor train gradient cross approximation for hamilton–jacobi–bellman equations. *SIAM Journal on Scientific Computing*, 45(5):A2153–A2184, 2023.

[34] T. Dubé. The structure of polynomial ideals and Gröbner bases. *SIAM Journal on Computing*, 19(4):750–773, 1990.

[35] S. Edalatzadeh, D. Kalise, K. Morris, and K. Sturm. Optimal actuator design for vibration control based on lqr performance and shape calculus, 2019. URL: https://arxiv.org/abs/1903.07572, arXiv:1903.07572.

[36] S. Edalatzadeh, D. Kalise, K. Morris, and K. Sturm. Shape optimization of actuators over banach spaces for nonlinear systems, 2020. URL: https://arxiv.org/abs/2002.07172, arXiv:2002.07172.

[37] L. Evans. An introduction to mathematical optimal control theory version 0.2, 2013. URL: https://math.berkeley.edu/~evans/control.course.pdf.

[38] P. Farrell, A. Birkisson, and S. Funke. Deflation techniques for finding distinct solutions of nonlinear partial differential equations. *SIAM Journal on Scientific Computing*, 37(4), 2015.

[39] A. Feldbaum. The simplest relay systems of automatic control. *Avtomatika i Telemekhanika*, 10:249–266, 1949.

[40] A. Feldbaum. Optimal processes in automatic control systems. *Avtomatika i Telemekhanika*, 14:712–728, 1953.

[41] L. Gaillard and M. Safey El Din. Solving parameter-dependent semi-algebraic systems. In *International Symposium on Symbolic and Algebraic Computation*, pages 447–456, 2024.

[42] J. L. Garcia, K. Maluccio, F. Sottile, and T. Yahl. RealRoots: A *Macaulay2* package. Version 1.1. A *Macaulay2* package available at https://github.com/Macaulay2/M2/tree/master/M2/Macaulay2/packages.

[43] J. L. Garcia, K. Maluccio, F. Sottile, and T. Yahl. Real solutions to systems of polynomial equations in *Macaulay2*. *Journal of Software for Algebra and Geometry*, 14, 2024.

[44] T. Georgiou and A. Tannenbaum. Switching surfaces and Groebner bases. In *Learning, control and hybrid systems (Bangalore, 1998)*, volume 241 of *Lecture Notes in Control and Information Sciences*, pages 81–89. Springer, London, 1999.

[45] D. Grayson and M. Stillman. Macaulay2, a software system for research in algebraic geometry. Available at www.math.uiuc.edu/Macaulay2.

[46] A. Hatcher. *Algebraic Topology*. Cambridge University Press, 2002.

[47] D. Kalise, K. Kunisch, and K. Sturm. Optimal actuator design based on shape calculus, 2024. URL: https://arxiv.org/abs/1711.01183, arXiv:1711.01183.

[48] R. Kalman. Contributions to the theory of optimal control. *Boletin de la Sociedad Matematica Mexicana*, 5:102–119, 1960.

[49] B. Karg and S. Lucia. Efficient representation and approximation of model predictive control laws via deep learning. *IEEE Transactions on Cybernetics*, 50(9):3866–3878, 2020.

[50] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL: http://arxiv.org/abs/1412.6980.

[51] Y. Lakshman. On the complexity of computing a Gröbner basis for the radical of a zero dimensional ideal. *Symposium on Theory of Computing*, pages 555–563, 1990.

[52] J. LaSalle. Time optimal control systems. *Proceedings of the National Academy of Sciences*, 45(4):573–577, 1959.

[53] L. Li, L. Wang, and H. Li. An efficient spectral trust-region deflation method for multiple solutions. *Journal of Scientific Computing*, 95:105–255, 2020.

[54] T. Lyche, C. Manni, and H. Speleers. B-splines and spline approximation. *Lecture notes*, 2017.

[55] A. Mantzaflaris, B. Mourrain, N. Villamizar, and B. Yuan. An algebraic framework for geometrically continuous splines. *Mathematics of Computation*, 95(358):925–968, 2026.

[56] D. Masser and G. Wüstholz. Fields of large transcendence degree generated by values of elliptic functions. *Inventiones Mathematicae*, 72:407–464, 1983.

[57] T. McDonald. Splines with boundary conditions. *Computers and Mathematics with Applications*, 50:1234–1239, 2007.

[58] T. Nakamura-Zimmerer, Q. Gong, and W. Kang. Adaptive deep learning for high-dimensional hamilton–jacobi–bellman equations. *SIAM Journal on Scientific Computing*, 43(2):A1221–A1247, 2021.

[59] S. Nawab, A. Oppenheim, and A. Willsky. *Signals and Systems*. Prentice-Hall Signal Processing. Pearson, 2 edition, 1996.

[60] H. Park and G. Regensburger, editors. *Gröbner Bases in Control Theory and Signal Processing*. De Gruyter, Berlin, Boston, 2007.

[61] D. Patil and D. Chakraborty. Computation of time optimal feedback control using Groebner basis. *IEEE Transactions on Automatic Control*, 59(8):2271–2276, 2014.

[62] D. Patil and D. Chakraborty. Time optimal feedback control using chebyshev polynomials and gröbner basis. *IFAC-PapersOnLine*, 48(14):50–55, 2015. 8th IFAC Symposium on Robust Control Design ROCOND 2015.

[63] D. Patil, A. Mulla, D. Chakraborty, and H. Pillai. Computation of feedback control for time optimal state transfer using groebner basis. *Systems & Control Letters*, 79:1–7, 2015.

[64] H. Phuoc Le and M. Safey El Din. Solving parametric systems of polynomial equations over the reals through hermite matrices. *Journal of Symbolic Computation*, 112:25–61, 2022.

[65] L. Pontryagin, V. Boltyanskii, R. Gamkrelidze, and E. Mishchenko. *The mathematical theory of optimal processes*. A Pergamon Press Book. The Macmillan Company, New York, 1964. Translated by D. E. Brown.

[66] F. Rauscher and O. Sawodny. Efficient online trajectory planning for integrator chain dynamics using polynomial elimination. *IEEE Robotics and Automation Letters*, 6(3), 2021.

[67] H. Schenck and M. Stillman. A family of ideals of minimal regularity and the hilbert series of $C^r(\Delta)$. *Advances in Applied Mathematics*, 19:169–182, 1997.

[68] H. Schenck and M. Stillman. Local cohomology of bivariate splines. *Journal of Pure and Applied Mathematics*, 117–118:535–548, 1997.

[69] I. J. Schoenberg. Contributions to the problem of approximation of equidistant data by analytic functions: Part A–on the problem of smoothing or graduation. a first class of analytic approximation formulae. *Quarterly of Applied Mathematics*, 4(1):45–99, 1946.

[70] I. J. Schoenberg. Contributions to the problem of approximation of equidistant data by analytic functions: Part B–on the problem of osculatory interpolation. a second class of analytic approximation formulae. *Quarterly of Applied Mathematics*, 4(2):112–141, 1946.

[71] L. Schumaker. Bounds on the dimension of spaces of multivariate piecewise polynomials. *The Rocky Mountain Journal of Mathematics*, 14(1):251–264, 1984.

[72] L. Schumaker. *Spline Functions: Basic Theory*. Cambridge University Press, 3 edition, 2007.

[73] L. Schumaker. *Spline Functions: Computational Methods*. Society for industrial and Applied Mathematics, Philadelphia, PA, 2015.

[74] L. Schumaker and M. Lai. *Spline Functions on Triangulations*, volume 110. Cambridge University Press, illustrated edition, 2007.

[75] G. Strang. Piecewise polynomials and the finite element method. *Bulletin of the American Mathematical Society*, 79(6):1128–1137, 1973.

[76] G. Strang. The dimension of piecewise polynomial spaces, and one-sided approximation. In *Conference on the Numerical Solution of Differential Equations*, volume 363, pages 144–152. Springer Berlin Heidelberg, 1974.

[77] G. Strang and G. Fix. *An Analysis of the Finite Element Method*. Prentice-Hall Series in Automatic Computation. Prentice-Hall, Inc., 1 edition, 1973.

[78] J. J. Sylvester. On a remarkable modification of Sturm's theorem. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 5:446–456, 1853.

[79] E. Süli and D. Mayers. *An introduction to numerical analysis*. Cambridge University Press, 2003.

[80] R. Valenza. *Linear Algebra: An Introduction to Abstract Mathematics*. Undergraduate Texts in Mathematics. Springer, 1 edition, 1993.

[81] D. C. Vu, M. D. Pham, T. T. H. Nguyen, T. V. A. Nguyen, and T. L. Nguyen. Time-optimal trajectory generation and observer-based hierarchical sliding mode control for ballbots with system constraints. *International Journal of Robust Nonlinear Control*, 34(11):7580–7610, 2024.

[82] U. Walther, T. Georgiou, and A. Tannenbaum. On the computation of switching surfaces in optimal control: a Gröbner basis approach. *IEEE Transactions on Automatic Control*, 46(4):534–540, 2001.

[83] Y. Wang, C. Hu, Z. Li, S. Lin, S. He, and Y. Zhu. Time-optimal control for high-order chain-of-integrators systems with full state constraints and arbitrary terminal states. *IEEE Transactions on Automatic Control*, 70(3):1499–1514, 2025.

[84] J. Wilkinson. *Rounding errors in algebraic processes*. Dover books on advanced mathematics. Courier Corporation, 1994.

[85] G. Ziegler. *Lectures on Polytopes*. Graduate Texts in Mathematics. Springer Verlag, 1 edition, 1995.