UNIVERSITY OF CALIFORNIA, SAN DIEGO

Noncommutative Computer Algebra in Linear Algebra and Control Theory

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Mathematics

by

Frank Dell Kronewitter III

Committee in charge:

Professor J. William Helton, Chair
Professor Jim Agler
Professor Kenneth Kreutz-Delgado
Professor Bob Skelton
Professor Nolan Wallach

2000

The dissertation of Frank Dell Kronewitter III is approved, and it is acceptable in quality and form for publication on microfilm:

Chair

University of California, San Diego

2000

To my mother, Barbara Zahn Kronewitter;

my father, Frank Dell Kronewitter, Jr.;

and all my friends from

Santa Barbara,

Newport Beach,

Mammoth Lakes,

and

San Diego

So Far,

So Good,

So What ?

— Megadeath

TABLE OF CONTENTS

V New Directions 204

LIST OF FIGURES

ACKNOWLEDGEMENTS

First of all, I would like to thank my thesis advisor, Bill Helton, who has taken the time over the last few years to listen to my incomprehensible ramblings. He has shown me how to make sense of them and fashion ideas into concrete mathematics. I probably would not have been able to prove 31,824 theorems without him, but definitely would not have been able to tell anyone about them.

I will never forget the puzzled look Bill gave me when I introduced myself and, in the same breath, asked him to be my thesis advisor. This was not the usual case where the student takes classes from or at least reads with potential advisors. Indeed, I was not accepted immediately, but was made to prove myself via many lectures on my research, usually attended by just Eric and Bill. Professor Helton has an insatiable appetite for mathematics. I was often surprised at what interested him and what bored him. His excitement level was often inversely proportional to my own.

I would also like to thank Dr. Charles Akemann of UC Santa Barbara, my alma mater, for telling me to leave the field of mathematics and study economics. While I'm not sure he was wrong, this was a challenge I had to accept.

Furthermore, I am grateful to Dr. Chia-Lin Wu and Dr. Weian Zheng for making my years at UC Irvine palatable and educational. I wish them both the best of luck.

Special thanks go to Jennifer Morse, Mike Mastropietro, Eric Rowell, Greg Leibon, Jeb Willenbring, and Peter Dower for providing emotional support and stimulating conversation during my years at UC San Diego and hopefully long into the future. Zelinda Collins has also provided a ray of sunshine at UCSD with her seventh floor sanctuary and typographic assistance.

VITA

July 27, 1967	Born
1991	B. S., University of California Santa Barbara
1991–1993	Teaching assistant, Department of Mathematics, University of California Irvine
1993	M. S., University of California Irvine
1993–1994	Wide Area Network embedded software engineer, Newport Systems/Cisco Systems
1994–1998	Teaching assistant, Department of Mathematics, University of California San Diego
1997–1999	Research assistant, Department of Mathematics, University of California San Diego
2000	Ph. D., University of California San Diego

PUBLICATIONS

Computer algebra in the control of singularly perturbed dynamical systems. (With J. William Helton and Mark Stankus) Proceedings of the 38th IEEE Conference in Decision and Control, Phoenix, Arizona, USA, Dec 7-10, 1999.

Using noncommutative Gröbner bases in solving partially prescribed matrix inverse completion problems. Submitted to Linear Algebra and It's Applications, 1999.

Singularly perturbed control systems using noncommutative computer algebra . (With J. William Helton, William M. McEneaney, and Mark Stankus) Submitted to International Journal of Robust and Nonlinear Control, 1999.

Noncommutative elimination ideal structure in structured algebraic problem solving. (With J. William Helton and Mark Stankus) preprint.

ABSTRACT OF THE DISSERTATION

Noncommutative Computer Algebra in Linear Algebra and Control Theory

by

Frank Dell Kronewitter III

Doctor of Philosophy in Mathematics

University of California San Diego, 2000

Professor J. William Helton, Chair

We will show how noncommutative computer algebra can be quite useful in solving problems in linear algebra and linear control theory. Such problems are highly noncommutative, since they typically involve block matrices. Conventional (commutative) computer algebra systems cannot handle such problems and the noncommutative computer algebra algorithms are not yet well understood. Indeed, the Gröbner basis algorithm, which plays a central role in many computer algebra computations, is only about thirteen years old in the noncommutative case.

We will demonstrate the effectiveness of our algorithms by investigating the partially prescribed matrix inverse completion problem and computations involving singularly perturbed dynamic systems. On both of these sorts of problems our methods proved to be quite effective. Our investigations into the partially prescribed matrix completion problem resulted in formulas which solve all 3x3 problems with eleven known and seven unknown blocks. One might even say that these formulas represent 31,824 new theorems. Our singular perturbation efforts focus on both the standard singular perturbed dynamic system and the information state equation. Our methods easily perform the sort of calculations needed to find solutions for the standard singular perturbation problem and in fact we are able to carry the standard expansion out one term further than has been done

previously. We are also able to generate (new) formulas for the solution to the singularly perturbed information state equation.

After demonstrating how useful our methods are, we will pursue the formal analysis of our techniques which are generally refered to as Strategies. The formal definition of a strategy allows some human intervention and therefore is not as rigid as an algorithm. Still, a surprising amount of rigorous analysis can be done especially when one adds some simple hypotheses as we will. In particular, we will introduce the notion of a *good* polynomial and the *gap* of a polynomial ideal which will prove useful in our formal analysis. We will show how successful strategies correspond to low gap ideals. Also introduced is the strategy+, which allows the user a bit more freedom than a strategy.

The lion's share of our noncommutative computer algebra investigations have been in the field of linear system theory. We will describe our accomplishments and demonstrate the strategy technique with some highly algebraic theorems on positive real transfer functions.

Finally, we will turn to controllability and observability operators which play an important role in linear system theory and are expressed symbolically as infinite sequences. We will offer finite algebraic characterizations of these operators, and use these characterizations to derive the state space isomorphism theorem.

Chapter 1

Introduction

This dissertation investigates the use of noncommutative computer algebra algorithms in solving problems in linear algebra and the analysis and control of linear dynamic systems. Many problems in linear algebra can be formulated as matrix equations and therefore handled with noncommutative symbolic algebra. Indeed, most algebraic calculations which one sees in linear systems theory, for example in the *IEEE Transactions in Automatic Control*, involve block matrices and so are highly noncommutative. Thus conventional commutative computer algebra packages, such as Mathematica and Maple, do not address them. This dissertation shows how these noncommutative algebraic calculations may be done with a computer.

Before delving into the proving of new theorems and the analysis of old theorems we will review some of the theory behind our computer algebra techniques. In particular we shall describe the (noncommutative) Gröbner basis algorithm, polynomial reduction, and some results from elimination theory.

We shall investigate the partially prescribed matrix inverse completion problem. In particular we will investigate a certain class of inverse block matrix completion problems where eleven of the blocks are known and seven are unknown, thereby proving 31,824 new theorems. We also demonstrate the use of noncommutative computer algebra in computing an optimal controller for the standard singularly perturbed dynamic system. The expansion of a near-optimal controller is carried out one term further than had previously been done. We shall illustrate the same technique by analyzing another singularly perturbed problem, the information state equation.

The research in this thesis did not only result in the generation of new formulas, but also resulted in some new results in terms of elimination ideals on a methodology called a strategy, for solving highly algebraic problems. This thesis gives a more rigorous analysis of the strategy than has previously appeared.

We will report on how basic properties of positive real matrix functions may be developed easily with noncommutative computer algebra. We will demonstrate our techniques by presenting some computer algebra proofs of some basic theorems in linear dynamic systems which exhibit a large amount of algebra. We will in fact show how these computer algebra proofs fit into the formalized classifications of pre-strategies and strategies as introduced in [11].

We shall conclude with an analysis of some basic properties of discrete linear systems. This analysis requires an algebraic interpretation of controllability and observability operators which are typically expressed symbolically as infinite sequences. This infiniteness brings these problems out of the realm of the standard computer algebra proof. We will present some purely algebraic relations which capture some of the properties of controllability and observability operators. We will demonstrate the power of these relations by giving computer algebra proofs of the Youla-Tissi state space isomorphism theorem and the discrete Grammian equation.

1.1 One use of Gröbner bases

Gröbner bases are a useful tool in the manipulation and analysis of polynomials. The Gröbner basis is dependent on an order placed on the indeterminates which make up the polynomials. An important property is that the Gröbner basis methods may be used to transform a set of equations $\{p_j = 0 : 1 \leq j \leq k_1\}$ into an equivalent set with a "triangular" form shown below.

$$q_1(x_1) = 0 \tag{1.1}$$

$$q_2(x_1, x_2) = 0 \tag{1.2}$$

$$q_3(x_1, x_2) = 0 \tag{1.3}$$

$$q_4(x_1, x_2, x_3) = 0 \tag{1.4}$$

$$\vdots$$

$$q_{k_2}(x_1, \ldots, x_n) = 0\,. \tag{1.5}$$

Here, the set of solutions to the collection of polynomial equations $\{q_j = 0 : 1 \le j \le k_2\}$ equals the set of solutions to the collection of polynomial equations $\{p_j = 0 : 1 \le j \le k_1\}$. This canonical form greatly simplifies the task of solving the collection of polynomial equations by facilitating back-solving for x_j in terms of $x_1, \ldots, x_{j-1}$.

1.1.1 Equations in one unknown

If an unknown (perhaps matrix) variable x satisfies an equation $p(x, a_1, \ldots, a_n) = 0$ which contains only known variables $a_1, \ldots, a_n$ and x a person could typically regard the variable x as known. For example, if p is a Riccati equation one could use the Matlab Riccati solver to determine a set of x's, matrices with numerical entries, which satisfy p if given specific matrices (with numerical entries) $a_1, \ldots, a_n$. A recurrent goal throughout this thesis will be transforming complicated sets of equations into equations in one unknown.

1.2 Some applications of noncommutative computer algebra and new results

To demonstrate the effectiveness of noncommutative computer algebra we have used it to solve some problems which exhibit a large amount of tedious noncommutative algebra. We will outline the methods we used to solve certain problems and present some new results. Types of problems we researched included the fields of matrix completion, singular perturbation, and standard linear system theory.

1.2.1 Matrix completion

In Chapter 4 we investigate the use of noncommutative Gröbner bases in solving partially prescribed matrix inverse completion problems. The type of problems considered here are similar to those in [2]. There the authors, among other things, gave necessary and sufficient conditions for the solution of a two by two block matrix completion problem. Our approach is exceedingly different from theirs and relies on symbolic computer algebra.

Here we describe a general method by which all block matrix completion problems of this type may be analyzed with sufficient computational power. We also demonstrate our method with an analysis of all three by three block matrix inverse completion problems with eleven known blocks and seven unknown. We discover that the solutions to all such problems are of a relatively simple form.

We then do a more detailed analysis of what can be considered the "central problem" of the 31,824 three by three block matrix completion problems with eleven known blocks and seven unknown. A solution to this problem of the form derived in [2] is presented.

Not only do we give a proof of our detailed result, but we describe the method used in discovering our theorem and proof, since it is somewhat unusual for these types of problems.

Chapter 4 closely follows the paper [19] which has been submitted to a linear algebra journal.

1.2.2 Singular perturbation of linear dynamic systems

In Chapter 5 we investigate the usefulness of noncommutative computer algebra in a particular area of control theory–singularly perturbed dynamic systems–where working with the noncommutative polynomials involved is especially tedious. Our conclusion is that they have considerable potential for helping practitioners with such computations.

Indeed we shall see that they are useful in manipulating the messy sets of noncommutative polynomial equations which arise in singular perturbation calculations. We will demonstrate our approach on two different singularly perturbed problems. We illustrate the method on the classical state feedback optimal control problem, see [18], where we obtain one more (very long) term than was done previously. Then we use it to derive singular perturbation expansions for the relatively new (linear) information state equation.

Chapter 5 closely follows the conference paper [14] which appears in the 1999 IEEE Conference on Decision and Control and the longer journal article [13] which has been submitted to the International Journal of Robust and Nonlinear Control.

1.3 Some research on strategy theory

In [28] a formal methodology was developed for "discovering" highly algebraic theorems. Since this methodology allows for some human intervention it cannot be refered to as an algorithm. The method is called a strategy.

In the original formulation of a strategy the user was allowed to select any polynomial. Still, a surprising amount of formal analysis was developed. In the pursuit of rigor we will require selected polynomials to satisfy certain criteria. Such polynomials will be simply called *good*.

An important part of this more rigorous strategy is elimination ideals which will be developed below in noncommutative algebras. In fact, the strategy formalism allows one to classify certain algebraic theorems as n-Strategies or n-prestrategies. We will use this strategy concept to classify purely algebraic problems. This is done in Chapter 7.

1.3.1 Standard linear system theory with computer algebra

This research began with the study of computer algebra proofs of known theorems of linear systems. Many of the theorems in [28] exhibit a significant amount of algebra which was done with our methods. We have seen instances where three pages of published algebra were performed with a three minute computer run. We will describe our linear system theory accomplishments and present a couple of examples in Part IV.

1.4 Infinite sequences and computer algebra

Certain objects called controllability and observability operators which are important in linear system theory are infinite sequences of operators. Thus they are not directly amenable to computer algebra methods. A purely algebraic characterization of them requires more care than the standard computer proof. In Part V (Chapter 12) we offer inherently finite definitions of controllability and observability operators which require the introduction of new symbols, such as shifts $\mathcal{S}$ and projections $\mathcal{P}$. We illustrate the use and strength of our formalism by deriving the state space isomorphism theorem, a central fact in control theory.

Part I

Background

Chapter 2

Some Necessary Concepts

Here we present some concepts which are essential to a thorough understanding of the research done in this thesis. An impatient reader may wish to skip this chapter and return when directed to do so by the index.

2.1 Enough NCAlgebra to get by

Although the analysis done in this document could be done using any noncommutative Gröbner basis software (e.g. [17]) here we use the software NCGB, [10]. This software runs under Mathematica [27] and may be used concurrently with the package NCAlgebra [9] which assists in the manipulation of noncommutative expressions.

Here we describe enough of the NCGB commands to read this document and understand the computer runs which provide the proofs to the problems addressed here. Since NCGB runs within Mathematica, the syntax should be somewhat familiar to those versed in Mathematica.

`**` – Noncommutative multiply.

`SetNonCommutative[ ]` – This command takes as an argument a list of indeterminants and sets them to be noncommutative.

`MatMult[ ]` – This command takes as an argument a set of compatibly dimensioned matrices of noncommuting elements and returns the matrix consisting of the product of the inputs.

`SetMonomialOrder[ ]` – This command sets the monomial order to be used by Gröbner Basis computations and noncommutative polynomial reduction computations.

`NCAutomaticOrder[ ]` – This command takes as input a set of relations and a list of indeterminates. The list of indeterminates implies a monomial order. Indeed, if all of the indeterminates in the input set of relations can be found in the list of indeterminates, then this command is equivalent to `SetMonomialOrder[]`. The remaining indeterminates in the relations which are not in the given monomial order are placed in some "appropriate" place in the order based upon suggestions given by the supplied variables. For example, if X^T is found in the relations and X has been placed in the order, then `NCAutomaticOrder[]` will place X^T in the order directly above X. Then the relation $X == X^T$ will correspond to the rule $X^T \to X$, rather than $X \to X^T$.

`NCMakeGB[ ]` – This command creates a noncommutative Gröbner Basis using the order specified in `SetMonomialOrder[]`. It takes as arguments a set of polynomials and a maximal number of iterations.

`NCProcess[ ]` – The general goal of NCProcess is to take a set of noncommutative polynomials and return an equivalent set of noncommutative polynomials which is as "useful" as possible.

This command makes heavy use of `NCMakeGB[]`. It creates a Gröbner basis, and then sorts the output based on the unknown variables present in the polynomials. It is somewhat unfortunate that at this point the criteria for "unknownness" is directly determined by the monomial order. The unknowns are declared to be the indeterminates which lie above the first $\ll$ in the order. That is if one is searching for an expression for x_j which does not contain the variables a_1 and a_2 the order must be specified as $\ldots \ll x_j < a_1 < a_2 < \ldots$ and this categorizing process will be done as if the a_1 and a_2 are unknowns.

`NCMakeRelations[ ]` – This command creates relations such as invertibility relations. Note that, since our method is using ideal relations in a free algebra, to imply an indeterminant X is invertible we create a new indeterminant X^{-1} and add the relations $XX^{-1} - 1$ and $X^{-1}X - 1$ to our generating set.

`SmallBasis[ ]` – This command takes an ordered list of polynomials and attempts to return a small basis for the ideal generated by them. We iteratively include successive elements of our basis, create a partial Gröbner Basis from them, and try to reduce the other polynomials with each partial Gröbner Basis. As the generating set grows if the excluded relations are elements of the ideal generated from the included relations at some iteration of the algorithm our goal has been achieved. Obviously, the sequence in which relations are presented to the small basis algorithm is important. The small basis algorithm acting on $(x^3, x^2, x, 1)$ returns the unenlightening $(x^3, x^2, x, 1)$, but when presented with $(1, x, x^2, x^3)$ the algorithm returns (1). (Computational difficulties prevent some idealized implementation which would consider all permutations of our relations.)

`NCReduction[ ]` – This command takes as input two lists of polynomials and attempts to reduce each polynomial in the first list with the set consisting of the second polynomials.

`NCAddTranspose[ ]` – This command takes as input a set of relations and adds the transposed equivalent of them. This is useful in an algebra with involution where the user would rather not enter both $a^T b == c$ and $b^T a == c^T$.

2.2 System theory basics

The central object in our studies is the finite dimensional linear dynamic system. This system may be written as

$$\dot{x}(t) = Ax(t) + Bu(t)$$

$$y(t) = Cx(t) + Du(t)$$

where A, B, C, and D are matrices and u, x, and y are time varying functions of t. u is said to be an input vector, x, a state vector, and y, an output vector. A is refered to as the state transition matrix; B, the input matrix; C, the output matrix; and D, the throughput matrix.

We may take the Laplace transform of this system to study the characteristics of a system in the frequency domain. The resulting system is

$$\tilde{y} = (C(sI - A)^{-1}B + D)\tilde{u}. \tag{2.1}$$

We write this transfer function as

$$\left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$$

Pictorially we have Figure 2.1.

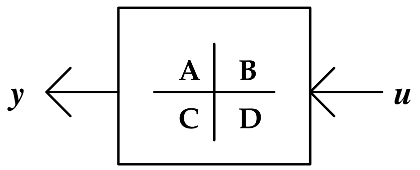


Figure 2.1: A general system

A matrix $M(s)$ whose entries are rational functions of $s \in \mathbb{C}$ is said to be *stable* if all of the poles of $M(s)$ have a negative real value. We will denote the portion of $\mathbb{C}$ with real part strictly less than 0, the open left half-plane, as $\mathbb{C}_{-}$.

It can easily be shown that if the spectrum (eigenvalues) of A are in the open left half plane the transfer function of a linear dynamical system with state transition matrix A will be stable. Letting σ return the spectrum of a matrix we have

$$\sigma(A) \subset \mathbb{C}_{-} \rightarrow \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \text{ is analytic in the right half plane.}$$

We will refer to a constant matrix A whose spectrum lies in $\mathbb{C}_{-}$ as *stable*.

It is a common problem in linear control theory to take an unstable linear dynamic system and "move" the poles of it's associated transfer function into the left half plane by designing some controller. Typically the controller has access to a subset of the components of y and is able to control a subset of the input components u. In Chapter 5, we present a method for designing controllers for a certain type of linear dynamic system. Such a controller is refered to as a *feedback control law*.

For those who are totally disoriented, it might help to remember that entries like A, B, C, and D are typically matrices. Objects denoted P, K, and G are most likely transfer functions, that is matrix valued functions, with entries, say, $\frac{1+s}{1+4s-s^2}$. In this purely symbolic framework it is easy to get confused if the reader doesn't have the background.

2.3 Some useful algebraic identities

The symbolic methods we use are purely algebraic and therefore a task which is central to our computer algebra investigations is mapping general hypotheses into pure algebra equations.

An example of this is the invertibility of some indeterminate. To imply an indeterminant X is invertible we create a new indeterminant X^{-1} and add the relations $XX^{-1} - 1 = 0$ and $X^{-1}X - 1 = 0$ to our generating set.

An equation, which often appears in system theory computations, is the Sylvester Equation in X,

$$AX + XB = C. \tag{2.2}$$

This equation can equivalently be written as

$$\left(B^T \oplus A\right) Vec(X) = Vec(C) \tag{2.3}$$

where $X \oplus Y$, the Kronecker sum, is defined to be $(X \otimes I) + (I \otimes Y)$ and $\otimes$ is the usual Kronecker (tensor) product. $Vec(X)$ is simply the vectorization of the matrix X.
The spectrum of $(B^T \oplus A)$ is $\lambda_i + \lambda_j$ where $\lambda_i \in \sigma(B^T)$ and $\lambda_j \in \sigma(A)$.

Definition 2.3.1 *If $\sigma(B) \cap \sigma(-A) = \emptyset$ we call the equation $AX + XB = C$ a* **full rank Sylvester equation** *and if we have $C = 0$ we call this a* **full rank null Sylvester equation***.*

It follows that if we have a full rank null Sylvester equation in X,

$$AX + XB = 0 \text{ where } \sigma(B) \cap \sigma(-A) = 0,$$

$X = 0$. This fact will be used many times in our automatic theorem proving.

2.4 Tricks of the trade

There are certain nuances in the use of the noncommutative software, NCGB. Some are briefly described here. In particular there are a few rules of thumb which should be followed, especially when one encounters computational difficulties. The reader who is only interested in seeing the results of these computations may wish to skip this section. On the other hand it may help to clarify the Mathematica input.

2.4.1 Commutative indeterminates

Presently NCGB cannot handle commutative indeterminates, that is indeterminates in the center of the algebra, without adding all of the n relations which would imply this in the free algebra. ($cx_j - x_jc$ for all indeterminates, x_j.) This adds considerable computational complexity to the problem.

This problem is encountered quite often in noncommutative computations, for example in the case where one has scalars in a problem with matrix relations.

This problem may be avoided by substituting a prime number for the commutative indeterminate. Making the substitution $z \to 7$ the output of the Gröbner Basis Algorithm may then contain a 49 which the user would recognize as z^2 or a 14 which the user would recognize as $2z$. Making the substitution $z \to 3$ would be more likely to present a problem since a $3x$ could be interpreted as a $3x$ or a zx.

Chapter 3

Ideals and Gröbner Bases

The research in this thesis involves finding algebraic formulas. We have found it useful to distinguish between known variables which we will label a_i and unknown variables which we will label x_i. Throughout this thesis we will denote an arbitrary field of characteristic 0 by $\mathbb{K}$. The corresponding algebra $\mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$ consists of polynomials which are sums of terms, for example $4a_1x_2a_1 + \frac{1}{6}x_1a_2$. A *term* is made up of a coefficient in $\mathbb{K}$ and a product of variables–a *monomial*, for example $5a_1x_2a_2$. A subset of $\mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$, $\mathcal{I}$, is called an *ideal* if $f, g \in \mathcal{I}$ implies that $l_f f r_f + l_g g r_g \in \mathcal{I}$ for l_f, r_f, l_g and r_g arbitrary elements of $\mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$. The smallest ideal which contains $\{p_1, \ldots, p_s\}$ is called the *ideal generated by* $\{p_1, \ldots, p_s\}$ and is denoted $\langle p_1, \ldots, p_s \rangle$. For terms m and n, we say that m divides n and write $m \mid n$ if there exist monomials L and R and a nonzero coefficient $c \in \mathbb{K}$ such that $n = cLmR$.

Given an ideal $\mathcal{I}$ any set of polynomials $\mathcal{B}$ is said to be a *basis* of $\mathcal{I}$ if $\mathcal{B}$ generates $\mathcal{I}$. We will be especially interested in a certain basis (which is unique under certain conditions) called the Gröbner basis.

3.1 Polynomials and equations

When studying a system of linear equations an equivalent system of linear equations in reduced echelon form can be constructed via Gaussian elimination. Many properties of the set of solutions of the linear system can be derived from this reduced echelon form including a full parameterization in terms of the free variables. Here we study solutions to a set of polynomial equations in more than one variable and propose the use of elimination ideals and a notion analogous to free variables. We will consider the case when the polynomials involve variables which commute and when they involve variables which do not commute. (That is, some of the results apply for members of the free *abelian* algebra generated by the variables $x_1, \ldots, x_n$, $(\mathbb{K}[x_1, \ldots, x_n])$, and some apply to the free (nonabelian) algebra generated by the variables $\{x_1, \ldots, x_n\}$ over some field $\mathbb{K}$.)

3.2 Gröbner bases

When applying Gaussian elimination to a system of linear equations one could define a total order on the variables. For example, when considering the system

$$\begin{aligned} 2x + 3y + 4z &= 9 \\ 2y + z &= 8 \end{aligned}$$

we could write $x > y > z$, since x appears to the left of y and y appears to the left of z. It is common to insist that any linear equation is written with the variables occuring in decreasing order. For example, one would not want to express the second equation above as $z + 2y = 8$ while performing Gaussian elimination.

In the case of polynomials, rather than linear expressions, we need to order all of the monic monomials and consider a "leading term" of a nonzero polynomial. After defining leading term, we consider replacement rules. Theory related to Gröbner bases is often developed without reference to any notion similar to replacement rules and we could have developed this theory without refering to replacement rules. We feel, however, that the theory is more accessible to those nonspecialists who have problems which Gröbner theory applies to (e.g. engineers).

3.2.1 Monomial orders

Essential to the creation of Gröbner bases is the consideration of a monomial order which is a total order on the monic monomials of the polynomial algebra under consideration which satisfies certain properties, [5], [22]. We assume the reader is familiar with the concept of monomial orders. We now discuss the concepts of leading coefficient, leading monomial and leading term, which depend on the monomial order under consideration, for a nonzero polynomial p.

For a polynomial p, let *support*(p) be the smallest set of monic monomials such that p lies in the k-span of support(p). For example, support($x^2y + 3yx + z) = \{x^2y, yx, z\}$. For a nonzero polynomial p and any particular monomial, some element of support(p) will be greater than the others. This monomial is called the *leading monomial* and is denoted LM(p). Exactly one term of p, called the *leading term* of p and denoted LT(p) will have the form $\mathrm{LT}(p) = \mathrm{LC}(p)\ \mathrm{LM}(p)$ for some scalar LC(p). LC(p) is called the leading coefficient of p. For example, under any monomial order with $x_3 < x_2 < x_1$ and $p = x_1 - x_2x_3 + 9x_1^2$, we have $\mathrm{LT}(p) = 9x_1^2$, $\mathrm{LM}(p) = x_1^2$ and $\mathrm{LC}(p) = 9$.

For any monomial order, the leading term of a product is the product of the leading terms. That is, for nonzero polynomials p and q,

$$\mathrm{LT}(pq) = \mathrm{LT}(p)\mathrm{LT}(q).$$

We will often use monomial orders which exhibit the following property.

Definition 3.2.1 *Let j and n be natural numbers such that $1 \leq j \leq n$. A mononial order is* **of j-th elimination type** *provided that any monic monomial involving $x_j, \ldots, x_n$ is greater than any monic monomial of $\mathbb{K}[x_1, \ldots, x_{j-1}]$.*

With the above definitions we can introduce replacement rules.

3.2.2 Gröbner rules

A *replacement rule* is $m \to q$ (that is, m followed by an arrow pointing to the right followed by q) for monic monomials m and polynomials q. The left hand side of a rule $m \to q$ is m and the right hand side is q. Every polynomial p corresponds to a *replacement rule* $\Gamma(p)$ where the left hand side of the rule is the leading monomial of the polynomial and the right hand side is the negative of the sum of the remaining terms in the polynomial divided by the coefficient of the leading term.

Example 3.2.2 If we have $x_1 > x_2 > x_3$ and $p = 4x_1 - 8x_2x_3 + 2x_1^2$, then $\Gamma(p)$ is $x_1^2 \to 4x_2x_3 - 2x_1$.

We can apply a replacement rule to a polynomial in the following fashion.

Example 3.2.3 To the polynomial

$$x_1x_1x_2x_1 + x_2x_3x_1 + x_2$$

we may apply

$$\{x_1^2 \to x_2x_3 - x_1\}$$

to get

$$(x_2x_3 - x_1)x_2x_1 + x_2x_3x_1 + x_2 = x_2x_3x_2x_1 - x_1x_2x_1 + x_2x_3x_1 + x_2.$$

3.2.3 Reduction

Definition 3.2.4 *Given a monomial order, let $P = \{f_1, \ldots, f_k\}$ be a set of polynomials. Now let p be any polynomial. We say that* p **is reducible to** g **with respect to** P *if there exists $f_i \in P$ such that $g = p - c\,u\,f_i\,v$ where c is a constant and u, v are monomials chosen so that the leading term of $c\,u\,f_i\,v$ coincides with one of the terms of p.*

The effect of the reduction is to replace a term of the polynomial p with terms of lower order. A polynomial p may be *reduced* to a simpler form with a set of polynomials P by using the replacement rules $\Gamma(P)$. Reducing a polynomial by a set of polynomials is similar to the Euclidean division algorithm one might use to find the remainder of, say, $\frac{x^3+3x^2+x+1}{x^2+2}$. For our problems, the Euclidean division algorithm doesn't apply, since there is more than one indeterminate. In this scenario it is common to view reduction as replacing a polynomial with a "simpler" polynomial.

Example 3.2.5 The polynomial $x_2x_1x_2x_1 + x_2$ may be reduced by the polynomial $x_1x_2 + 3$.

The Euclidean division algorithm yields

$$(x_2x_1x_2x_1 + x_2) - x_2(x_1x_2 + 3)x_1 = -3x_2x_1 + x_2$$

and if we use replacement rules we have that $x_2x_1x_2x_1 + x_2$ reduces to $-3x_2x_1 + x_2$ by applying $\Gamma(x_1x_2 + 3)$ which is $x_1x_2 \to -3$.

If repeated application of these types of rules to a polynomial transforms the equation to 0, then we have shown that the polynomial under consideration is an element of the (two-sided) ideal generated by the relations used to create the rules.

Given a polynomial p, one may apply a set of rules F repeatedly until no further reduction can occur. At this stage $m \nmid u$ for all terms u in our reduced p and all left hand sides m in the set F. We call this a *normal form* of p with respect to F. This normal form is denoted

$$\mathrm{NForm}(p, F)$$

and is not unique in general. In the next section we will give conditions on F which make $\mathrm{NForm}(p, F)$ unique.

3.2.4 Gröbner bases

Definition 3.2.6 *Given a polynomial ideal $\mathcal{I}$ we call a set of polynomials $\mathcal{G}$, a* **Gröbner Basis** *for $\mathcal{I}$ if f is in $\mathcal{I}$ if and only if f may be reduced to 0 with repeated application of all rules $\Gamma(\mathcal{G})$.*

If the repeated application of the rules $\Gamma(\mathcal{G})$ does not reduce a polynomial p to 0, then there is still some profound information provided about p's relationship to the ideal $\langle\mathcal{G}\rangle$ as the following theorem attests.

Theorem 1 *([7], pg.186) If $\mathcal{G}$ is a Gröbner basis for an ideal $\mathcal{I}$, then any polynomial p has a unique normal form with respect to $\Gamma(\mathcal{G})$. That is the normal form of p is invariant under the order in which the rules, $\Gamma(\mathcal{G})$, are applied. The resulting irreducible polynomial is denoted NForm$(p, \mathcal{G})$.*

If the indeterminates commute, then the Gröbner basis is always a finite set of polynomials. Buchberger developed an algorithm which finds this set when given a generating set of relations for the ideal. In the non-commuting case the Gröbner basis for an ideal may be infinite, for example a Gröbner basis associated with the ideal $\langle xyxy + yxyx\rangle$.

Nevertheless there exists a similar algorithm due to F. Mora [22] which iteratively finds a Gröbner basis and terminates if the Gröbner basis is finite. In practice even this finite Gröbner basis may be incomputable when computer resources are taken into account and one stops the algorithm after a specified number of iterations. The result is some finite approximation to a Gröbner basis, a *partial Gröbner basis.*

This finite approximation, though not exhibiting the powers of a Gröbner basis, is often useful in reduction as shown below in Section 4.4.6 and transparently throughout the document. For example, successful applications of the small basis algorithm introduced in Section 3.3.1 are dependent on the reduction properties of partial Gröbner bases.

These finite approximations generated by Mora's algorithm are made up of elements of the ideal generated by the original relations. There are few available software implementations of this non-commutative algorithm (NCGB, [10] and Opal, [17]). For our computations, we used the package NCGB.

3.2.5 Buchberger's algorithm

Buchberger's algorithm generates a (commutative) Gröbner basis. It takes as input a set of generators (polynomials in commuting indeterminates) for an ideal and returns the Gröbner basis for the ideal. To describe the noncommutative analogue (Mora's algorithm) we need to introduce the correct notion of a common multiple.

Let S be the free semigroup generated by a finite alphabet A (i.e., the collection of words in the letters of A). Let (m_1, m_2) be an ordered pair of elements of S. By a match of (m_1, m_2) we mean a 4-tuple (l_1, r_1, l_2, r_2) of elements of S which satisfy one of the following conditions:

(1) $l_1 = r_1 = 1$, $m_1 = l_2 m_2 r_2$.

(2) $l_2 = r_2 = 1$, $m_2 = l_1 m_1 r_1$.

(3) $l_1 = r_2 = 1$, $l_2 \neq 1$, $r_1 \neq 1$, there is a $w \neq 1$ with $m_1 = l_2 w$, $m_2 = w r_1$.

(4) $l_2 = r_1 = 1$, $l_1 \neq 1$, $r_2 \neq 1$, there is a $w \neq 1$ with $m_1 = w r_2$, $m_2 = l_1 w$.

These conditions make $l_1\ m_1\ r_1 = l_2\ m_2\ r_2$. This is a common multiple of m_1 and m_2 which is minimal in some sense.

In the commutative case, the Basis Algorithm makes use of a kind of resolvent of two polynomials called the *S-Polynomial*. $\text{S-Pol}(f, g) = c_2 u_1 f_1 - c_1 u_2 f_2$ where $c_i = \text{LC}(f_i)$ and where u_i is chosen so that $u_i \text{LM}(f_i)$ is the least common multiple of $\text{LM}(f_1)$ and $\text{LM}(f_2)$ for $i = 1, 2$. In the noncommutative case, there are several such resolvents—one for each match. If M is a 6-tuple $(f_1, f_2, l_1, r_1, l_2, r_2)$ where (l_1, r_1, l_2, r_2) is a match for $(\mathrm{LM}(f_1), \mathrm{LM}(f_2))$ and $c_i = \mathrm{LC}(f_i)$, then we set

$$\text{S-Pol}(M) = c_2 l_1 f_1 r_1 - c_1 l_2 f_2 r_2 \ .$$

Example:

Consider the polynomials

$$f_1 = aaba + ab \qquad f_2 = abaa + ba \ .$$

There are four matches for (f_1, f_2):

1. $(aba, 1, 1, aba)$. In this case the S-Polynomial is

$$(aba)(f_1) - (f_2)(aba) = -baaba + abaab \ .$$

2. $(ab, 1, 1, ba)$. In this case the S-polynomial is

$$(ab)(f_1) - (f_2)(ba) = -baba + abab \ .$$

3. $(1, baa, aab, 1)$. In this case the S-Polynomial is

$$(f_1)(baa) - (aab)(f_2) = abbaa - aabba \ .$$

4. $(1, a, a, 1)$. In this case the S-Polynomial is

$$(f_1)(a) - (a)(f_2) = 0 \ .$$

The algorithm is iterative and starts with a finite set of polynomials G_1. The k-th step has available to it a set G_k such that the ideal generated by G_1 equals the ideal generated by G_k. The k-th step of the algorithm creates a set G_{k+1} by setting it equal to the union of G_k and the set of all nonzero reductions of S-Polynomials for all pairs in G_k. The process repeats as long as there are S-Polynomials with nonzero reductions. In other words, the process repeats until $G_k = G_{k+1}$.

As noted above this algorithm may not terminate in the noncommutative case and may even be intractible in the commutative setting. We call the map which generates a true Gröbner basis (possibly infinite) $idealGBA$. We call the map which returns the l-th iteration, $\mathcal{G}_l$, in the algorithm described above as the $partialGBA_l$.

3.3 Removing redundant relations

There is another part of our algorithms which is so important it deserves some discussion. A Gröbner basis or even a partial Gröbner basis contains many polynomials, a few of which are important, since they contain few unknowns. However, they also contain many, many polynomials which are long and uninteresting. Two of the techniques we use to remove these redundant polynomials are the small basis algorithm and remove redundant.

3.3.1 The small basis algorithm

Often what a human wishes to find is not a set of relations with the reduction properties described in Section 3.2.4, but a small set of relations which describes the solution set, a small basis. Such a goal may be accomplished by iteratively including the first k elements of our basis B in a set B_k, creating a partial Gröbner basis $\mathcal{G}_{B_k}$ from B_k, and trying to reduce the other polynomials, $B \backslash B_k$, with this partial Gröbner basis $\mathcal{G}_{B_k}$. When B_k has the property that the excluded relations, $B \backslash B_k$, are elements of the ideal generated by the included relations B_k our goal has been achieved. *All* of our relations lie in the polynomial ideal generated by B_k. We call such an algorithm the *small basis algorithm*.

Obviously the sequence in which relations are presented to the small basis algorithm is important. The small basis algorithm acting on $(x^3, x^2, x, 1)$ returns the unenlightening $(x^3, x^2, x, 1)$, but when presented with $(1, x, x^2, x^3)$ the algorithm returns (1). (The computer time required descourages some idealized implementation which would consider all permutations of a set of polynomials)

3.3.2 Remove redundant

A particularly fast method of eliminating unnecessary polynomials is the Remove Redundant command. Briefly, it records the history of the production of the Gröbner basis as a tree and then throws away all polynomials which correspond to nodes which are not seminal nodes.

3.4 Elimination theory

In this section we will introduce and analyze elimination ideals which are central to our analysis of the solvability properties of systems of polynomial equations.

3.4.1 Multigraded lexicographic order

Recall the definitions of pure lexicographic and length (graded) lexicographic monomial orders on commutative monomials as discussed in [5]. The non-commutative versions are essentially similar, but to ensure a well defined total order a monomial may[1] be parsed from left to right in the tie breaking length lexicographic order criteria.

We will find useful a combination of these two types of orders. It lets one define sequential subsets of indeterminates such that each subset is ordered with graded lexicographic ordering within the subset, but indeterminates of a higher set are lexicographically higher than indeterminates of a lower set. That is a monomial consisting of one element in a higher set will sit higher in the monomial ordering than a monomial consisting of the product of any number of elements in lower sets. We use the $\ll$ symbol to discern the subset breakpoints discussed above. For example, when we write $x_1 < x_2 < x_3 \ll x_4$ we get that $x_1x_2 < x_2x_1$, $x_3x_2x_1 < x_4$, and $x_3 < x_1x_2$. We call such an ordering *multigraded lexicographic.*

[1]In fact this is the scheme used in the NCGB computations.

3.4.2 Elimination ideals

As promised above we now motivate the multigraded lexicographic order with a central concept in elimination theory.

Definition 3.4.1 *Let j and n be natural numbers such that $1 \leq j \leq n$. A monomial order is of* j**-th elimination type** *provided that any monic monomial containing one of $\{x_j, \ldots, x_n\}$ is greater than a monomial in*

$$\mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}].$$

If we have a multigraded lexicographic monomial order on the monomials in $\mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$ which is of j-th elimination type for $1 \leq j \leq n$, then we have, in the notation introduced in Section 3.4.1, $x_{i-1} \ll x_i$ for $1 \leq i \leq n$ and $a_m \ll x_1$.

A Gröbner basis $\mathcal{G}$ for an ideal $\mathcal{I}$ created under an order of j-th elimination type exhibits the following property.

$$\begin{gathered} \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}] \\ \text{is a Gröbner basis for } \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}] \end{gathered} \tag{3.1}$$

and therefore

$$\langle \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}] \rangle = \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}] \rangle.$$

This property, well known in the commutative case, was extended recently to non-commutative algebras in [11], Theorem 11.3.

3.4.3 Complexity concerns

For commutative Gröbner bases, it has been observed that the Gröbner basis algorithm often finishes more quickly when one uses a graded lexicographic order rather than a lexicographic order. We have also observed this phenomenon in the noncommutative case, but have not put much effort into its analysis. For this reason we typically try to use orders which contain the least amount of $\ll$'s.

For a detailed analysis of the computational difficulties involved in solving a particular set of noncommutative problems see the Ph.D. thesis of Ben Keller, [16].

Part II

New Formulas Derived With Computer Algebra

Chapter 4

Partially Prescribed Matrix Inverse Completion Problems

4.1 The problem

In this chapter we consider block matrix completion problems similar to that in [2]. Here we take two partially prescribed, square matrices, A and B, and describe conditions which make it possible to complete the matrices so that they are inverses of each other. That is, we wish the completed matrices to satisfy

$$AB = I \text{ and } BA = I. \tag{4.1}$$

4.1.1 A sample problem

An example of such a problem is, given matrices k_1, k_2, k_3, and k_4, is it possible to find matrices, u_1, u_2, u_3, and u_4, such that equation (4.1) is satisfied where

$$A = \begin{pmatrix} k_1 & u_2 \\ u_1 & k_2 \end{pmatrix}, B = \begin{pmatrix} u_3 & k_4 \\ k_3 & u_4 \end{pmatrix} \tag{4.2}$$

Thus, A and B are 2x2 block matrices. The answer to this question, due to [2], is given in Section 4.2.4. We now describe our problem in detail.

4.1.2 The general block matrix inverse completion problem

Begin by partitioning two matrices, A and B, whose entries are elements in the field $\mathbb{F}$, conformally for matrix multiplication into n by n block matrices. Next, choose l of these blocks to be known and $2n^2 - l$ to be unknown. Give some conditions on the known matrices, which may be expressed algebraically, such as invertibility or self-adjointness. Our problem has then been defined.

We ask if it is possible to fill in the $2n^2 - l$ unknown blocks so that equation (4.1) is satisfied and seek to derive formulas for these matrices in terms of the prescribed blocks. To be more specific, we might even call this problem the *purely algebraic partially prescribed matrix inverse completion problem.*

The solution to such a problem will be a set of matrix equations in the known and unknown submatrices.

4.2 What a solution means

In general it is not known how to solve a system of matrix equations where several of the matrices are unknown. Unknown matrices can appear in matrix equations in several ways of which some are more computationally acceptable than others. We will analyze these forms and classify certain solution sets.

4.2.1 A good, triangular solution

It is generally a tractable problem to solve a triangular system of matrix equations of the form

$$\begin{aligned} q_1(k_1, \dots, k_l) &= 0 && (4.3) \\ q_2(k_1, \dots, k_l) &= 0 && (4.4) \\ &\vdots \\ q_m(k_1, \dots, k_l) &= 0 && (4.5) \end{aligned}$$

$$\begin{aligned} q_{m+1}(k_1, \dots, k_l, \mathbf{u}_{\sigma(\mathbf{1})}) &= 0 && (4.6) \\ q_{m+2}(k_1, \dots, k_l, u_{\sigma(1)}, \mathbf{u}_{\sigma(\mathbf{2})}) &= 0 && (4.7) \\ q_{m+3}(k_1, \dots, k_l, u_{\sigma(1)}, \mathbf{u}_{\sigma(\mathbf{2})}) &= 0 && (4.8) \\ q_{m+4}(k_1, \dots, k_l, u_{\sigma(1)}, u_{\sigma(2)}, \mathbf{u}_{\sigma(\mathbf{3})}) &= 0 && (4.9) \\ &\vdots \\ q_{s_2}(k_1, \dots, k_l, u_{\sigma(1)}, \dots, \mathbf{u}_{\sigma(\mathbf{2n^2-1})}) &= 0 && (4.10) \end{aligned}$$

One can first verify that a completion is possible by verifying equations (4.3-4.5) with the given (known) matrices. We refer to these equations as *compatibility conditions*. Then, solve for the (possibly non-unique) $u_{\sigma(1)}$ with equation (4.6), use this to next solve for $u_{\sigma(2)}$ with equations (4.7-4.8), etcetera. The defining characteristic of equations (4.3-4.10) is that for every polynomial $q(k_1, \dots, k_l, u_{\sigma(1)}, \dots, u_{\sigma(k)})$ there exists $\tilde{q}(k_1, \dots, k_l, u_{\sigma(1)}, \dots, u_{\sigma(k-1)})$.

Definition 4.2.1 *We will call a system of polynomial equations,* $\{p_j = 0 : 1 \leq j \leq s_1\}$*,* **formally backsolvable** *if there exists a set of equations,* $\{q_j = 0 : 1 \leq j \leq s_2\}$*, in the form of (4.3-4.10) whose solution set is equal to the solution set of the* $\{p_j\}$*.*

4.2.2 A better, essentially decoupled solution

An even more computationally useful set of matrix equations will have the essentially decoupled form

$$
\begin{aligned}
q_1(k_1, \ldots, k_l) &= 0 && (4.11)\\
q_2(k_1, \ldots, k_l) &= 0 && (4.12)\\
&\vdots\\
q_m(k_1, \ldots, k_l) &= 0 && (4.13)
\end{aligned}
$$

$$
\begin{aligned}
q_{m+1}(k_1, \ldots, k_l, \mathbf{u}_{\sigma(\mathbf{1})}) &= 0 && (4.14)\\
q_{m+2}(k_1, \ldots, k_l, \mathbf{u}_{\sigma(\mathbf{1})}) &= 0 && (4.15)\\
&\vdots\\
q_{m+s_p}(k_1, \ldots, k_l, \mathbf{u}_{\sigma(\mathbf{p})}) &= 0 && (4.16)
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{u}_{\sigma(\mathbf{p+1})} &= q_{m+s_p+1}(k_1, \ldots, k_l, u_{\sigma(1)}, \ldots, u_{\sigma(p)}) && (4.17)\\
&\vdots\\
\mathbf{u}_{\sigma(\mathbf{n^2-l})} &= q_{s_2}(k_1, \ldots, k_l, u_{\sigma(1)}, \ldots, u_{\sigma(p)}) && (4.18)
\end{aligned}
$$

$$
\begin{aligned}
q_{s_2+1}(k_1, \ldots, k_l, u_1, \ldots, u_n) &= 0 && (4.19)\\
&\vdots\\
q_{s_3}(k_1, \ldots, k_l, u_1, \ldots, u_n) &= 0 && (4.20)
\end{aligned}
$$

A solution of this type is made up of *compatibility conditions*, equations (4.11-4.13) and (4.19-4.20); *equations in one unknown*, equations (4.14-4.16); and equations (4.17-4.20) we call *singletons* which are described in more detail below.

Given such an essentially decoupled set of equations one can use equations (4.14-4.16), $q_{m+1}, \ldots, q_{m+s_p}$, to solve for potential $u_{\sigma(1)}, \ldots, u_{\sigma(p)}$ simultaneously. It is then a simple matter to find matrices $u_{\sigma(p+1)}, \ldots, u_{\sigma(n^2-l)}$ by evaluating polynomials $q_{m+s_p+1}, \ldots, q_{s_2}$. Finally, one must check that the solutions derived, $u_{\sigma(p+1)}, \ldots, u_{\sigma(n^2-l)}$, are acceptable by validating equations (4.19-4.20).

Equations of the form (4.17-4.18) will be refered to as *singletons*. This type of equation is characterized by the fact that there is a single instance of an unknown variable which does not occur in equations (4.14-4.16). This unknown variable appears in the singleton equation as a monomial consisting of only itself. The singleton variable is the left hand side of equations (4.17-4.20).

The singleton equation has a very attractive form for a human who wishes to find polynomials in few unknowns. Given an equation in knowns and unknowns, $\mathcal{E}$, it allows one to eliminate the unknown singleton variable, say $u_{\sigma(p+1)}$, from $\mathcal{E}$ by replacing instances of the unknown indeterminant with its equivalent polynomial representation, say q_{m+s_p+1}. After this substitution has been performed the equation $\mathcal{E}$ will not contain the singleton unknown.

Since all unknown variables in equations (4.11-4.20) which are not singleton unknowns appear in equations without any other unknown variables, we think of these as essentially decoupled, but not totally decoupled due to the coupling equations (4.19-4.20).

Definition 4.2.2 *We will call a system of polynomial equations,* $\{p_j = 0 : 1 \leq j \leq s_1\}$*,* **essentially decoupled** *if there exists a set of equations,* $\{q_j = 0 : 1 \leq j \leq s_2\}$*, in the form of (4.11-4.20) whose solution set is equal to the solution set of the* $\{p_j\}$*.*

4.2.3 The best, formally decoupled solution

An even more computationally useful set of matrix equations will have the following formally decoupled form which is similar to an essentially decoupled solution (4.11-4.20), but doesn't contain coupling conditions on the solutions (4.19-4.20).

$$\begin{aligned} q_1(k_1, \dots, k_l) &= 0 && (4.21)\\ q_2(k_1, \dots, k_l) &= 0 && (4.22)\\ &\vdots \\ q_m(k_1, \dots, k_l) &= 0 && (4.23)\end{aligned}$$

$$\begin{aligned} q_{m+1}(k_1, \dots, k_l, \mathbf{u}_{\sigma(\mathbf{1})}) &= 0 && (4.24)\\ q_{m+2}(k_1, \dots, k_l, \mathbf{u}_{\sigma(\mathbf{1})}) &= 0 && (4.25)\\ &\vdots \\ q_{m+s_p}(k_1, \dots, k_l, \mathbf{u}_{\sigma(\mathbf{p})}) &= 0 && (4.26)\end{aligned}$$

$$\begin{aligned} \mathbf{u}_{\sigma(\mathbf{p+1})} &= q_{m+s_p+1}(k_1, \dots, k_l, u_{\sigma(1)}, \dots, u_{\sigma(p)}) && (4.27)\\ &\vdots \\ \mathbf{u}_{\sigma(\mathbf{n^2-l})} &= q_{s_2}(k_1, \dots, k_l, u_{\sigma(1)}, \dots, u_{\sigma(p)}) && (4.28)\end{aligned}$$

A solution of this type is made up of *compatibility conditions*, equations (4.21-4.23); *equations in one unknown*, equations (4.24-4.26); and singletons (4.27-4.28).

Given such a decoupled set of equations one can use equations (4.24-4.26), $q_{m+1}, \dots, q_{m+s_p}$, to solve for $u_{\sigma(1)}, \dots, u_{\sigma(p)}$ simultaneously. It is then a simple matter to find matrices $u_{\sigma(p+1)}, \dots, u_{\sigma(n^2-l)}$ by evaluating polynomials

$$q_{m+s_p+1}, \dots, q_{s_2}.$$

Since all unknown variables in equations (4.21-4.28) which are not singleton unknowns appear in equations without any other unknown variables, we think of these as decoupled. This is obviously a better form of solution than equations (4.11-4.20), since any solutions for $u_{\sigma(1)}, \ldots, u_{\sigma(p)}$ will do. The coupling compatibility equations are absent.

Definition 4.2.3 *We will call a system of polynomial equations,* $\{p_j = 0 : 1 \leq j \leq s_1\}$, **formally decoupled** *if there exists a set of equations,* $\{q_j = 0 : 1 \leq j \leq s_2\}$, *in the form of (4.21-4.28) whose solution set is equal to the solution set of the* $\{p_j\}$.

4.2.4 A sample answer

This formally decoupled form is the type of solution to the problem in Section 4.1.1 derived in [2]. There it was shown that, for invertible k_i, the matrices A and B defined in (4.2) satisfy (4.1) if and only if the unknown submatrix u_4 satisfies the following relation

$$u_4 k_2 u_4 = u_4 + k_3 k_1 k_4. \tag{4.29}$$

Then, the other unknown submatrices are given in terms of u_4:

$$u_1 = k_4^{-1} - k_2 u_4 k_4^{-1} \tag{4.30}$$

$$u_2 = k_3^{-1} - k_3^{-1} u_4 k_2 \tag{4.31}$$

$$u_3 = k_4 k_3 u_4 k_4^{-1} k_1^{-1} \tag{4.32}$$

This answer contains no compatibility conditions. Equation (4.29) is an equation in one unknown u_4. The remaining equations (4.30-4.32) are singletons.

In addition to the original results presented in this chapter, this theorem was also proven using our noncommutative Gröbner methods, see [11] or

`http://math.ucsd.edu/~ncalg`.

4.3 Main results on 3x3 matrix inverse completion problems

We have performed extensive analysis of the 3x3 block matrix inverse completion problem. In particular, the problem described in Section 4.1.2 where n is three and l is eleven. We have assumed in our detailed analysis that all eleven known blocks are invertible. First, we will define a property certain matrix completion problems have. We have done no analysis of matrix completion problems with this property.

Assume that the pair of block matrices, A and B, are partitioned into known and unknown blocks compatible for matrix multiplication.

Definition 4.3.1 *A and B are said to be* **strongly undetermined** *if there exists an entry of the block matrices AB or BA which is a polynomial consisting entirely of unknown blocks.*

Notice that A and B being strongly undetermined is equivalent to the existence of *both* an entire row (column) of unknown blocks in A *and* an entire column (row) of unknown blocks in B. For example, the following configuration of known and unknown blocks is strongly undetermined.

$$A = \begin{pmatrix} \mathbf{u_1} & \mathbf{u_2} & \mathbf{u_3} \\ k_1 & k_2 & k_3 \\ k_4 & k_5 & k_6 \end{pmatrix} \text{ and } B = \begin{pmatrix} k_7 & k_8 & \mathbf{u_4} \\ k_9 & u_5 & \mathbf{u_6} \\ k_{10} & k_{11} & \mathbf{u_7} \end{pmatrix} \tag{4.33}$$

The product of these two matrices, AB, has the following form.

$$\begin{pmatrix} u_1k_7 + u_2k_9 + u_3k_{10} & u_1k_8 + u_2u_5 + u_3k_{11} & \mathbf{u_1u_4 + u_2u_6 + u_3u_7} \\ k_1k_7 + k_2k_9 + k_3k_{10} & k_1k_8 + k_2u_5 + k_3k_{11} & k_1u_4 + k_2u_6 + k_3u_7 \\ k_4k_7 + k_5k_9 + k_6k_{10} & k_4k_8 + k_5u_5 + k_6k_{11} & k_4u_4 + k_5u_6 + k_6u_7 \end{pmatrix}$$

Since the upper right entry (in boldface) is a polynomial made up entirely of unknown blocks, configuration (4.33) is strongly undetermined.

4.3.1 A class of 31,834 3x3 matrix inverse completion problems

In our investigations we have analyzed (via computer) a certain collection of 3x3 matrix completion problems. Two 3x3 block matrices have a total of 18 entries. We have analyzed those which have seven unknown and 11 known blocks and do not have the strongly undetermined property. We have chosen to put efforts into this ratio of known to unknown blocks because we believe Theorem 2, the initial subject of our research, to be surprising and yet lack the computational resources to study all 3x3 matrix completion problems or even one 4x4 matrix completion problem. Section 4.4.3 describes how the motivated researcher with unlimited computational power can go about analyzing any block matrix problem of any size of the type addressed in this chapter.

The following theorem shows that all of our seven unknown, 11 known block matrix completion problems (which are not strongly undetermined) have a particularly nice solution.

Theorem 1 *Let A and B be three by three block matrices such that 11 of the 18 blocks are known and seven are unknown. Let the known blocks be invertible. The corresponding partially prescribed inverse completion problems may be classified as follows.*

1. *If the configuration of unknown blocks is not strongly undetermined, then the partially prescribed inverse completion problem is* formally backsolvable*, in the sense of Definition 4.2.1.*

2. *If the configuration of unknown blocks is not strongly undetermined and is not of the form given in (4.34) or a permutation of such configuration then the partially prescribed inverse completion problem is* essentially decoupled*, in the sense of Definition 4.2.2.*

These answers satisfy a technical non-redundancy condition, 3-non-degeneracy*, which will be defined in Section 4.2.2 once we have built up our Gröbner machinery.*

$$A = \begin{pmatrix} k_1 & k_2 & k_3 \\ k_4 & k_5 & k_6 \\ u_1 & u_2 & u_3 \end{pmatrix} \text{ and } B = \begin{pmatrix} k_7 & k_8 & u_4 \\ k_9 & u_5 & k_{10} \\ k_{11} & u_6 & u_7 \end{pmatrix} \tag{4.34}$$

The proof of this theorem, which requires noncommutative symbolic software, will be given in Section 4.4.4. Answers to the individual problems, which consist of sets of polynomials similar to that found in equations (4.29-4.32), can be found on the internet at `http://math.ucsd.edu/~ncalg`.

4.3.2 Detailed analysis of a particular 3x3 matrix inverse completion problem

Now we give a closer analysis, than that given in the last section, of a particular matrix inverse completion problem which satisfies the assumptions of Theorem 1. We show how someone interested in a particular matrix completion problem might arrive at a finer analysis of the problem, instead of the rather terse conclusion given in Theorem 1. Our goal in this section is to present a short, computationally simple set of formulas which give the solution to a particular partially prescribed inverse matrix completion problem. Our conclusions will have the same flavor as those presented in [2].

We will analyze the "central problem" of those addressed in Theorem 1, the known/unknown configuration:

$$A = \begin{pmatrix} a & \underline{t} & b \\ \underline{u} & c & \underline{v} \\ d & \underline{w} & e \end{pmatrix} \text{ and } B = \begin{pmatrix} \underline{x} & f & g \\ h & \underline{y} & i \\ j & k & \underline{z} \end{pmatrix} \tag{4.35}$$

or the equivalent permuted form,

$$A = \begin{pmatrix} a & b & \underline{t} \\ d & e & \underline{w} \\ \underline{u} & \underline{v} & c \end{pmatrix} \text{ and } B = \begin{pmatrix} \underline{x} & g & f \\ j & \underline{z} & k \\ h & i & \underline{y} \end{pmatrix} \tag{4.36}$$

where a through k are known and invertible block matrices and the underlined t through z are unknown block matrices.

Theorem 2 *Given A and B as in (4.35) with invertible knowns a,b,c,d,e,f,g,h,i,j, and k, as well as the invertibility of the matrix made up of the outer known blocks of A in (4.35),*

$$\begin{pmatrix} a & b \\ d & e \end{pmatrix}, \tag{4.37}$$

then $AB = I$ and $BA = I$ if and only if the knowns satisfy the following compatibility conditions :

$$\begin{aligned}
\tilde{p}(da^{-1} - eb^{-1}) &= (a^{-1}b - d^{-1}e)\tilde{q} && (4.38)\\
(da^{-1} - eb^{-1})^{-1}\tilde{q} &= (da^{-1} - eb^{-1})^{-1}\tilde{q}e(da^{-1} - eb^{-1})^{-1}\tilde{q} && (4.39)\\
&\quad + (da^{-1} - eb^{-1})^{-1}\tilde{q}dg + jb(da^{-1} - eb^{-1})^{-1}\tilde{q} - kci + jag \\
\tilde{p}(a^{-1}b - d^{-1}e)^{-1} &= \tilde{p}(a^{-1}b - d^{-1}e)^{-1}e\tilde{p}(a^{-1}b - d^{-1}e)^{-1} && (4.40)\\
&\quad + \tilde{p}(a^{-1}b - d^{-1}e)^{-1}dg + jb\tilde{p}(a^{-1}b - d^{-1}e)^{-1} - kci + jag
\end{aligned}$$

where

$$\begin{aligned}
\tilde{p} &= (-a^{-1}h^{-1}i + a^{-1}bjh^{-1}i - d^{-1} - d^{-1}ejh^{-1}) && (4.41)\\
\tilde{q} &= (-kf^{-1}a^{-1} + kf^{-1}gda^{-1} - b^{-1} - kf^{-1}geb^{-1}). && (4.42)
\end{aligned}$$

The unknown matrices can then be given as:

$$\begin{aligned}
z &= (-kf^{-1}a^{-1} + kf^{-1}gda^{-1} - b^{-1} - kf^{-1}geb^{-1})(da^{-1} - eb^{-1})^{-1} \\
&= \tilde{q}(da^{-1} - eb^{-1})^{-1} && (4.43)
\end{aligned}$$

or equivalently

$$\begin{aligned}
z &= (a^{-1}b - d^{-1}e)^{-1}(-a^{-1}h^{-1}i + a^{-1}bjh^{-1}i - d^{-1} - d^{-1}ejh^{-1}) \\
&= (a^{-1}b - d^{-1}e)^{-1}\tilde{p} && (4.44)
\end{aligned}$$

and then

$$t = -agi^{-1} - bzi^{-1} \tag{4.45}$$

$$u = -k^{-1}ja - k^{-1}zd \tag{4.46}$$

$$v = k^{-1} - k^{-1}jb - k^{-1}ze \tag{4.47}$$

$$w = i^{-1} - dgi^{-1} - ezi^{-1} \tag{4.48}$$

$$x = a^{-1} + fk^{-1}j - gda^{-1} + fk^{-1}zda^{-1} \tag{4.49}$$

$$y = c^{-1}k^{-1}jaf + c^{-1}k^{-1}jbk + c^{-1}k^{-1}zdf + c^{-1}k^{-1}zek \tag{4.50}$$

The proof of this theorem will be given in Section 4.4.5.

This answer consists of three compatibility conditions (4.38-4.40), an equation in one unknown (4.43) or (4.44), and singletons (4.45-4.50) and is therefore *formally decoupled.*

More effort has gone into this problem than the mechanical process of verifying a solution form as done in Theorem 1. In Theorem 2 degenerate equations (in a sense to be made precise in Section 4.4.2) have been eliminated in a more careful manner. Furthermore, the mild assumption of the invertibility of (4.37) was added when we observed that this gave the solution a simpler form. This invertibility assumption is discussed further in Section 4.4.6.

Solutions to all of the problems (configurations) addressed in Theorem 1 can be found via the internet at `http://math.ucsd.edu/~ncalg/`. These solutions are a formatted list of equations in both LaTeX and *Mathematica* form.

4.4 Solving the purely algebraic inverse matrix completion problem

We will next describe a method for solving general matrix completion problems of the type described above. The main tool we will use for our solution of the problem is the Gröbner basis.

4.4.1 A pure algebra interpretation of the purely algebraic partially prescribed inverse matrix completion

This section describes our matrix completion problem in the language of an algebraist. The reader may skip this section, if so desired, with no loss of continuity to the chapter.

Labeling the known blocks as k_i, we may consider the free algebra, $\mathbb{F}[k_1, \ldots, k_l]$, over the field under consideration, $\mathbb{F}$, modulo some presupposed conditions (e.g. the invertibility of a known submatrix, k_i, which is expressed in ideal theoretic notation as $\langle k_i k_i^{-1} - 1, k_i^{-1} k_i - 1 \rangle$)

$$S = \frac{\mathbb{F}[k_1, \ldots, k_l]}{\langle \text{ conditions on the knowns } \rangle}. \tag{4.51}$$

Let the size of the submatrices be m. Picking the known block matrices consists of defining a map,

$$\phi : S \to M_m(\mathbb{F}).$$

Defining

$$T = S[u_1, \ldots, u_{2n^2-l}], \tag{4.52}$$

completing the matrices A and B may be viewed as a map

$$\Phi : T \to M_m(\mathbb{F})$$

such that

$$\Phi|_S = \phi.$$

Here we are interested in a special completion, Φ, so that our matrices satisfy (4.1). If we let *Flatten* be the operation which takes a set of matrices and makes their components into an unordered list and J be the ideal generated by relations (4.1),

$$J = \langle Flatten(AB - I, BA - I) \rangle, \tag{4.53}$$

our goal is achieved when J lies in the kernel of Φ.

The ideal $J \cap S$ can be regarded as a set of compatibility conditions on the known submatrices, conditions which must hold if we are to form an inverse completion. If the known submatrices satisfy a set of relations which form a generating set for $J \cap S$ the unprescribed blocks may be defined so that the completed matrices satisfy

$$AB = I \text{ and } BA = I.$$

Given G a generating set for J so that

$$\langle G \rangle = J,$$

our compatibility conditions may be described as $G \cap S$, equations (4.3-4.5) in the backsolvable case or (4.11-4.13) in the decoupled case. The completion relations are $G \cap (J \setminus S)$, equations (4.6-4.10) in the backsolvable case or (4.14-4.20) in the decoupled case. Notice that all of the relations making up $G \cap (J \setminus S)$ will contain at least one u_i.

4.4.2 Nondegenerate solutions

On closer analysis of the backsolvable form described in equations (4.3-4.10), the existence of q_1 implies the existence of $q_{m+1}, \ldots, q_{s_2}$. Simply multiply q_1 by the appropriate $\mathbf{u_i}$. A more interesting set of relations has the following property.

Definition 4.4.1 *A set of relations,* $\{p_j = 0 : 1 \leq j \leq s\}$*, will be called* **non-degenerate** *if*

$$p_i \notin \langle \{p_j\}_{j \neq i} \rangle. \tag{4.54}$$

Due to the infiniteness of the noncommutative Gröbner basis, condition (4.54) cannot in general be verified. A condition which can be verified computationally is the following.

Definition 4.4.2 *A set of relations,* $\{p_j = 0 : 1 \leq j \leq s\}$*, will be called* **k-non-degenerate** *if*

$$p_i \notin \langle \{p_j\}_{j \neq i} \rangle_k. \tag{4.55}$$

where $\langle \{p_j\}_{j\neq i}\rangle_k$ *is the* k *iteration partial Gröbner basis created from* $\{p_j\}_{j\neq i}$

Recall that all of the solutions to the problems addressed in Theorem 1 were shown to be formally backsolvable and 3-non-degenerate.

The special case of nondegenerate decoupled solutions

In the formally decoupled case, the notion of nondegeneracy is somewhat simpler. For a set of equations of the form (4.11-4.20) to be nondegenerate we merely require that

$$q_{m+j} \notin \langle \{q_1, \ldots, q_m\}\rangle \text{ for } j = 1, \ldots, p. \tag{4.56}$$

That is, the relations which define the $u_{\sigma(i)}$ for $i = 1, \ldots, p$ are not trivial, and merely consequences of the compatibility conditions (4.11-4.13). The relations associated with equations (4.17-4.20), those which define $u_{\sigma(i)}$ for $i = p+1, \ldots, n^2-l$, are obviously not trivial.

We also have the computational analogue.

Definition 4.4.3 *We will call a system of equations in the form of (4.11-4.20)* k**-non-degenerate** *if* $q_{m+j}, j = 1, \ldots, p$, *is not reduced by the Gröbner rules associated with the* k *iteration partial Gröbner basis of* $\{q_1, \ldots, q_m\}$, $\langle q_1, \ldots, q_m\rangle_k$.

Beware that this definition is algorithm dependent. This form of non-redundancy was used to verify 3-non-degeneracy for all the problems in Theorem 1 which were formally decoupled. All but 36 were of this form.

4.4.3 A recipe for solving the general block matrix inverse completion problem

We are given matrices A and B partitioned conformally for matrix multiplication into n^2 blocks each and a configuration of l prescribed (known) and $2n^2 - l$ unknown blocks. We may also be given conditions on these matrices which are expressed algebraically (e.g. invertibility, $aa^{-1} - 1 = 0$ and $a^{-1}a - 1 = 0$). We look to discover compatibility conditions on the known matrices and formulas for the unknown matrices to solve our problem, that is ensure (4.1) is satisfied. This goal may often be achieved by following the steps below.

I Fill in the known blocks of A and B with symbolic, noncommuting indeterminates, $k_1, \ldots, k_l$.

II Fill in the unknown blocks of A and B with symbolic, noncommuting indeterminates, $u_1, \ldots, u_{2n^2-l}$.

III Create the noncommutative polynomials resulting from the operations $AB-I$ and $BA-I$.

IV Create a (noncommutative, partial) Gröbner basis for the polynomials derived in step III and any assumed algebraic conditions on the matrices under the order:

$$k_1 < k_2 < \cdots < k_l \ll u_1 \ll u_2 \ll \cdots \ll u_{2n^2-l} \tag{4.57}$$

V Check the result is decoupled, of the form described in Section 4.2.2, (4.11-4.20), or at least in the form given in Section 4.2.1, (4.3-4.10).

VI Verify that the relations defining unknown matrices are not merely consequences of the other relations by using the Small Basis Algorithm, or some variant of it.

4.4.4 Proof of seven unknown, 11 known theorem

We created a *Mathematica* procedure which iteratively searches through all permutations of seven unknown blocks and 11 known blocks and performs the sort of analysis described in Section 4.4.3. For a given configuration (i.e. permutation) of knowns and unknowns one may apply permutation matrices, Π and Ψ, to A and B, to get $\Pi^{-1}A\Psi$ and $\Psi^{-1}B\Pi$, and obtain at most 36 other equivalent configurations. This property was exploited to reduce the computations needed from 31,824 cases to about 1,500 cases. Only one matrix inverse completion problem was analyzed from each equivalence class.

First, we will describe the procedures followed for a particular configuration. Then, we will list the pseudo-code which performed the necessary analysis for the entire problem. Our proof will be completed with a discussion of the results of our *Mathematica* procedure.

A particular configuration

We created a two iteration partial Gröbner basis from the polynomial matrix equations resulting from AB and BA along with the invertibility relations of the knowns.

The order we used to create the Gröbner basis was the following

$$\begin{aligned} & k_1 < k_2 < k_3 < k_4 < k_5 < k_6 < k_7 < k_8 < k_9 < k_{10} < k_{11} \\ & \ll u_1 \ll u_2 \ll u_3 \ll u_4 \ll u_5 \ll u_6 \ll u_7 \end{aligned} \tag{4.58}$$

where the k_j represents the j^{th} known block and u_i represents the i^{th} unknown block. Inverses have been suppressed in our lists of knowns for clarity. Any listing of known variables should be accompanied by their inverses. These inverses are placed directly above and in the same group as the original variable. So our order truly begins $k_1 < k_1^{-1} < k_2 < k_2^{-1} < \ldots$.

The output of the Gröbner basis algorithm in virtually all cases was of the form described in Section 4.2.2, equations (4.11-4.20).

To establish non-redundancy we used the output of the Gröbner basis algorithm which consisted solely of known indeterminates, equations (4.11-4.13) or $G \cap S$ in the language of Section 4.4.1. We ran the Gröbner basis algorithm for one more iteration on these known relations, creating a three iteration Gröbner basis. We used this Gröbner basis to attempt to reduce the relations which contain the unknown indeterminates.

After applying the Gröbner rules associated with this Gröbner basis to the set of relations containing unknown indeterminates our set of relations still had the form given in equations (4.11-4.20). That is we verified 3-non-redundancy, condition (4.54). This verification was done by computer.

This shows that the problem associated with this particular configuration is formally decoupled.

Pseudo-code

Here we give some pseudo-code with a Mathematica slant which performs the sort of analysis described in the above section for all seven unknown and 11 known configurations. An essential part of the algorithm is the function `NCMake-GroebnerBasis[ polys, k]` which creates a k iteration partial Gröbner basis from *polys*.

```
inverses = NCMakeRelations[{Inv, k1,k2,k3,k4,k5,k6,k7,k8,
                     k9,k10,k11}]

SetMonomialOrder[ k1<k2<k3<k4<k5<k6<k7<k8<k9<k10<k11
                     <<u1<<u2<<u3<<u4<<u5<<u6<<u7 ]

permList = Permutations[ {0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1} ]

For[ i = 1, i++, i <= Length[ permList ],

    If[ MemberQ[ alreadyDoneList, permList[[i]] ]
        Continue[]
        ]

    AppendTo[ alreadyDoneList,
                      MakeTransformations[ permList[[i]] ] ]

    {A,B} = MakeSymbolicMatrices[ permList[[i]] ]

    relations = Union[ inverses, Flatten[
        MatrixMultiply[ A,B ] - IdentityMatrix[3],
        MatrixMultiply[ B,A ] - IdentityMatrix[3]  ]  ]
```

```
output = NCMakeGroebnerBasis[ relations, 2 ]

polysInKnowns = FindPolysInOnlyTheVariables[ output,
                     {k1,k2,k3,k4,k5,k6,k7,k8,k9,k10,k11}]

reductionSet = NCMakeGroebnerBasis[ knownPolys, 1 ]

output = NCReduction[ output, PolyToRule[ reductionSet ]  ]

determinedIndeterminates =
            PolysInOneUnknown[ output,{u1,u2,u3,u4,u5,u6,u7} ]

singleIndeterminates =
            PolysExplicit[output,{u1,u2,u3,u4,u5,u6,u7}]

If[Union[determinedIndeterminates,singleIndeterminates]
            =={u1,u2,u3,u4,u5,u6,u7},
        Print["SUCCESSFUL"]
    ]
Else[
        Print["UNSUCCESSFUL"]
    ]

]  (* End of For[] loop *)
```

End game

The problems which were strongly undetermined did not have the formally backsolvable form. Of the problems which were not strongly undetermined, there were six cases in which the output of the two iteration partial Gröbner basis did not have the form of equations (4.11-4.20). For these six cases we performed the same analysis, but created a three iteration partial Gröbner basis instead of halting the algorithm after two iterations, as done originally. In all six of these cases the three iteration partial bases had the form of equations (4.11-4.20) and were shown to be 3-non-redundant. Therefore, the result stated in the theorem follows.

The *Mathematica* code associated with the pseudo-code given above ran for approximately 3 days on a Sun Ultra II with two 166Mhz processors and 1Gb of RAM. The computer was a departmental machine and therefore the processes associated with these computations were only given a portion of the total computational resources available. The computations on a similar machine dedicated to this problem might take half the time. $\square$

4.4.5 Proof of particular theorem in Section 4.3.2

We shall need the following lemma for our proof:

Lemma 1 (Schur) *If $x_{1,1}, x_{1,2}, x_{2,1}, x_{2,2}$ are invertible block matrices of the same size*

$$\begin{pmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{pmatrix}$$

is invertible if and only if $-x_{2,1}x_{1,1}^{-1}x_{1,2} + x_{2,2}$ is invertible.

Proof:

$$\begin{pmatrix} I & 0 \\ x_{2,1}x_{1,1}^{-1} & I \end{pmatrix} \begin{pmatrix} x_{1,1} & 0 \\ 0 & -x_{2,1}x_{1,1}^{-1}x_{1,2} + x_{2,2} \end{pmatrix} \begin{pmatrix} I & x_{1,1}^{-1}x_{1,2} \\ 0 & I \end{pmatrix} = \begin{pmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{pmatrix}$$

$\square$

Proof: [Of Theorem]

$\Rightarrow$

Creating a three iteration partial Gröbner basis with the relations

$$AB = I, BA = I, \text{ and the invertibility of the knowns} \tag{4.59}$$

using the NCGB command NCProcess[1] yields a set of polynomials which includes relations

$$
\begin{aligned}
z &= zez + zdg + jbz - kci + jag \qquad (4.60)\\
a^{-1}h^{-1} - a^{-1}bjh^{-1} + a^{-1}bzi^{-1} &= -d^{-1}i^{-1} - d^{-1}ejh^{-1} + d^{-1}ezi^{-1} \qquad (4.61)\\
f^{-1}a^{-1} - f^{-1}gda^{-1} + k^{-1}zda^{-1} &= -k^{-1}b^{-1} - f^{-1}geb^{-1} + k^{-1}zeb^{-1} \qquad (4.62)
\end{aligned}
$$

and relations (4.45-4.50). See Appendix 2 pages 7-9. The order used is given on page 49, order (4.63). Since polynomials created through the Gröbner basis algorithm are in the polynomial ideal generated by the original relations, the validity of relations (4.60-4.62) and (4.45-4.50) is a consequence of relations (4.59).

For us to write the relations (4.60-4.62) in the form (4.38-4.44) we require the invertibility of $(da^{-1} - eb^{-1})$ and $(a^{-1}b - d^{-1}e)$. These invertibility relations are provided by the Schur lemma given above, since the outer matrix (4.37) consisting of a,b,d, and e, all knowns, is assumed to be invertible. With this we can solve for z explicitly in equations (4.61-4.62) and write the relations (4.43-4.44) defining z. Furthermore we may use these definitions of z to write the relations (4.60-4.62) as (4.38-4.40).

$\Leftarrow$

The converse is again approached with a Gröbner basis method. As above the Schur complement formulas give the invertibility of $(da^{-1} - eb^{-1})$ and $(a^{-1}b - d^{-1}e)$ which shows that relations (4.60-4.62) follow from (4.38-4.44). The question then becomes whether or not relations (4.59) are in the ideal generated by polynomials (4.45-4.50) and (4.60-4.62). We create a seven iteration partial Gröbner basis, $\mathcal{G}_7$, from the polynomials (4.45-4.50) and (4.60-4.62) with the NCGB command NCMakeGB under the graded (length) lexicographic monomial order. One can verify that the original equations (4.59) reduce to 0 with respect to $\mathcal{G}_7$. This shows that the relations $AB = I$ and $BA = I$ are elements of the non-commutative polynomial ideal generated by the relations (4.45-4.50) and (4.60-4.62) and the

[1] Appendix 2 page 6 contains the input to the NCProcess command, the "unraveled" equations (4.59).

invertibility of the knowns. The result follows.

□

4.4.6 Discovering Theorem 2 and its proof

The process of discovering our particular theorem, Theorem 2, follows the formal notion of a *strategy* rigorously developed in [11] and reviewed in Chapter 7. In particular this discovery uses a bit different application of Gröbner bases, elimination theory, which was described in Section 3.4.

Addressing our problem

In light of our goal, creating polynomials in few unknowns, we used this monomial order[2]

$$a < b < c < d < e < f < g < h < i < j < k \ll z \ll u \ll v \ll w \ll x \ll y \quad (4.63)$$

and ran the Gröbner basis algorithm with an iteration limit of three.

The output of this Gröbner basis computation included relations (4.45-4.50) as well as (4.60-4.62). (See Appendix 2 pages 7-9 for the entire output of the GBA.) Thus these relations are a consequence of the original relations. The necessity part of the proof is complete modulo a bit of Schur complement beautification done in Section 4.4.6.

Converse : a smaller basis

It is true that the original relations (4.59) are members of the ideal generated by the long and ugly relations taking up pages 7-9 of Appendix 2. (The *partiality* of a Gröbner basis at some iteration is only in respect to its reduction properties

[2]Inverses have been suppressed in our lists of knowns for clarity. Any listing of known variables should be accompanied by their inverses. These inverses are placed directly above and in the same group as the original variable. So our order truly begins $a < a^{-1} < b < b^{-1} < \ldots$.

and not the ideal generated by these relations.) We could have written these down instead of equations (4.38-4.40), our final conclusion, and stopped, but we would prefer to have a more concise set of relations which imply the original relations. In other words, we would like to have a smaller basis for this ideal.

Although the computer commands in NCGB have the ability to simplify the basis in the manner above in the same step as generating it, by setting certain options, we did not have the computing power, or perhaps the patience, to isolate the few relations on z given above (4.60-4.62) using this method under the original order. To this end, the monomial order was changed to graded lexicographic. In NCGB notation we replaced all of the $\ll$'s with $<$'s. The purely graded lexicographic order computations are often of much less computational complexity, since monomials usually must be merely checked for number of elements. When our original order was imposed on the small basis algorithm the two iteration application did not complete after several days running on a Sun SPARCstation-4 computer, while under the purely graded order the algorithm finished in a few minutes.

We tried several different sequences of which most gave unsatisfactory results. The bases found were not small enough in these cases. An acceptable small basis obtained through this procedure consisted of the invertibility relations on the knowns, the relations which give the unknowns other than z in terms of z (4.45-4.50), and relations concerning z and the knowns (4.60-4.62). The computer work associated with this is given in Appendix 3.

Confirmation

To confirm that these relations (4.60-4.62, 4.45-4.50, and invertibility of the knowns) imply the original relations we created a non-commutative partial Gröbner basis from these relations and reduced the original relations with this partial Gröbner basis. The original relations all reduced to 0. Thus it was shown that the original relations were elements of the ideal generated by the relations given above. Interestingly enough a five iteration partial Gröbner basis did not reduce the original relations (4.59) although a seven iteration Gröbner basis did. (See Appendix 4.) The order used for this computation was again the graded lexicographic.

Beautification with Schur Complements

Equations (4.61-4.62) are especially appreciated because they are linear in one unknown variable z. A more satisfying situation, though, would be to have an expression for z entirely in terms of the knowns. This may be accomplished by assuming the invertibility of

$$(a^{-1}b - d^{-1}e) \quad and \quad (da^{-1} - eb^{-1}), \tag{4.64}$$

in equations (4.61-4.62). By the Schur Lemma 1, this is equivalent to the invertibility of the outer matrix $\begin{pmatrix} a & b \\ d & e \end{pmatrix}$, since all entries of this matrix are themselves invertible. At the outset of our investigations we had no reason to assume this more restrictive condition. It was only after realizing the utility of this assumption that it was added to our conditions.

With this, z is given explicitly by the equations (4.43-4.44) and each of these must satisfy the quadratic (4.60). Hence the equations (4.38-4.40) on the knowns are a necessary and sufficient set of conditions for $AB = I$ and $BA = I$.

4.5 Matrix completion conclusion

In this chapter we have investigated the use of noncommutative symbolic algebra software in the analysis of partially prescribed inverse matrix completion problems. We described a method for solving such problems with a computer. We have shown that the solutions to all 3x3 block inverse matrix completion problems with seven unknown and 11 known blocks are of a relatively simple form. We presented one particular theorem and showed how it can be massaged into a more palatable form by making some mild assumptions on the prescribed (known) blocks.

4.6 Matrix completion appendices

Appendices for the matrix completion results found in Chapter 4 may be found at `http://math.ucsd.edu/~ncalg`.

Chapter 5

Singularly Perturbed Control Systems

5.1 Singular perturbation vs computer algebra

Singular perturbation is a commonly used technique in the analysis of systems whose dynamics consist of two pieces. One piece might be slow, the other fast, or one might be known where the other is somewhat uncertain. Extensive analysis has been done of this type of plant for the LQR and H^∞ control problems, for example [18], [23], [24].

Typically one has an equation where some coefficients depend on a parameter $\frac{1}{\varepsilon}$. To solve this equation, one postulates an expansion in ε for the solutions x_ε to the equation, then

(a) substitutes x_ε into the equation,

(b) sorts the equation according to powers of ε, and

(c) finds simple equations for successive terms in the expansion of x_ε.

The sorting in (b) can be tedious and the business of solving (c) can be very involved.

This chapter concerns methods we are developing for doing these steps automatically. As we shall illustrate, NCAlgebra constructions and commands easily handle steps (a) and (b), thereby producing the (long) list of equations which must be solved. This is straightforward and saves considerable calculation time for those engaged in singular perturbation calculations. Step (c) involves solving complicated systems of equations and this is always tricky business. Thus there is no way of knowing in advance if noncommutative Gröbner basis methods will be effective for (reducing to a simple form) the equations found in large classes of singular perturbation problems. This is the focus of experiments (using the package NCGB which runs under NCAlgebra) we have been conducting and on which we report here.

Most of this chapter shows how one can treat the most classic of all singular perturbation problems using computer algebra. Ultimately, we see that Mora's Gröbner basis algorithms are very effective on the equations which result. Indeed our method carries out the expansion one step further than has previously been done, see Section 5.3.3. Then we sketch another newer H^{∞} estimation problem called the "cheap sensor" problem (see [12]). On this our computer techniques proved effective.

5.1.1 Hardware

Computer computations for Section 5.3 were performed with NCGB on a Sun Ultra I with one 166 MHz processor and 192MB of RAM. The computations done in Section 5.4 were performed with NCGB on a Sun Ultra II with two 166MHz processors and 1Gb of RAM. The Sun Ultra II was a departmental machine and therefore equivalent computations on a dedicated computer might take less than half of the times reported here.

5.2 The standard state feedback singular perturbation problem

The standard singularly perturbed linear time-invariant model consists of a differential state equation; which depends on some perturbation parameter ϵ; and an output equation. The general control problem is to design some feedback law which specifies the input as a function of the state so that the controlled system will satisfy some given performance objective.

5.2.1 The system

Here we study the two time scale dynamic system previously analyzed in [18]:

$$\begin{bmatrix} \frac{dx}{dt} \\ \epsilon\frac{dz}{dt} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} u \tag{5.1}$$

$$y = \begin{bmatrix} M_1 & M_2 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} \tag{5.2}$$

where $x \in \mathbb{R}^n$, $z \in \mathbb{R}^m$, $u \in \mathbb{R}^p$, and $y \in \mathbb{R}^q$. Here, m, n, p and q are integers, A_{11} is an $n \times n$ matrix, A_{12} is a $n \times m$ matrix, A_{21} is a $m \times n$ matrix, A_{22} is a $m \times m$ matrix, B_1 is a $n \times p$ matrix, B_2 is a $m \times p$ matrix, M_{11} is a $q \times n$ matrix, and M_{22} is a $q \times m$ matrix.

5.2.2 The near-optimal LQR problem

The infinite-time optimal linear regulator problem is to find a control, $u(t), t \in [0, \infty]$ which minimizes the quadratic cost

$$J = \int_0^\infty (y^T y + u^T R u)\, dt \tag{5.3}$$

where R is a positive definite weighting matrix. It is well known that the solution to this problem is of the form

$$u^* = -R^{-1}B^T K(\epsilon) \begin{bmatrix} x \\ z \end{bmatrix} = G(\epsilon) \begin{bmatrix} x \\ z \end{bmatrix} \tag{5.4}$$

where $K(\epsilon)$ is a solution to the Algebraic Riccati Equation (ARE)

$$KA + A^T K - KBR^{-1}B^T K + M^T M = 0 \tag{5.5}$$

with

$$A = \begin{bmatrix} A_{11} & A_{12} \\ \frac{A_{21}}{\epsilon} & \frac{A_{22}}{\epsilon} \end{bmatrix}, B = \begin{bmatrix} B_1 \\ \frac{B_2}{\epsilon} \end{bmatrix}, \text{ and } M = \begin{bmatrix} M_1 & M_2 \end{bmatrix} \tag{5.6}$$

$$K(\epsilon) = K_0 + \epsilon K_1 + \epsilon^2 K_2 + \dots \tag{5.7}$$

If K is the solution to this optimal state feedback control problem, then it also may be used to express the optimal cost as a function of the initial state of the system as

$$J^* = \begin{bmatrix} x^T(0) & z^T(0) \end{bmatrix} K \begin{bmatrix} x(0) \\ z(0) \end{bmatrix}. \tag{5.8}$$

Notice that the solution J^* as presented involves solving equation (5.5) which is a Riccati equation of size $n+m$ by $n+m$. Since the structures present in the system (5.1) are partitioned, it seems likely that we might decompose the problem and significantly reduce the sizes of the matrices while deriving an only slightly sub-optimal controller. Indeed, it is standard to divide the problem of solving the $n+m$ dimensional Riccati, (5.5), into solving two smaller decoupled Riccatis, one which is n dimensional, the other which is m dimensional, and then use these solutions to obtain a control law which gives a performance J nearly equal to the optimal performance J^*. This regulator is known as the *near-optimal regulator.*

In the next two subsections we review this decomposition of the state space into slow and fast parts. This is mostly a matter of setting our notation, which in fact is the same notation as [18]. The noncommutative Gröbner computer algebra which is the subject of our investigations is well suited for manipulating polynomials into a triangular or even decoupled form.

5.2.3 Decomposing the problem

Here, we decompose our two time scale system into its fast parts and slow parts. We will assume throughout that A_{22} is invertible.

The slow system

The dynamics of the slow system can be found by setting ϵ to zero in equation (5.1) obtaining what is often called the *quasi-steady state* of the system. This transforms the equation involving ϵ in system (5.1) into an algebraic equation rather than a differential equation,

$$z_s(t) = -A_{22}^{-1}(A_{21}x_s(t) + B_2u_s(t)), \tag{5.9}$$

and then substitution of this z_s into the top equation in (5.1) yields

$$\frac{dx_s}{dt} = A_0x_s(t) + B_0u_s(t), \ x_s(t_0) = x^0 \tag{5.10}$$

where

$$A_0 \triangleq A_{11} - A_{12}A_{22}^{-1}A_{21}, \qquad B_0 \triangleq B_1 - A_{12}A_{22}^{-1}B_2.$$

Here the subscript s indicates that the vectors in equations (5.10)-(5.9) are the slow parts of the vectors in (5.1).

The fast system

The fast system has dynamics

$$\frac{dz_f}{dt} = A_{22}z_f(t) + B_2u_f(t), \qquad z_f(t_0) = z^0 - z_s(t_0) \tag{5.11}$$

where

$$z_f = z - z_s \text{ and } u_f = u - u_s. \tag{5.12}$$

Here the subscript f indicates that the vectors in equations (5.11)-(5.12) are the fast parts of the vectors in (5.1).

5.2.4 Decomposing the measurement

We may then also decompose (5.2) into its slow and fast parts

$$\begin{aligned} y &= M_1 x + M_2 z \\ &= M_1[x_s + O(\epsilon)] + M_2[-A_{22}^{-1}(A_{21}x_s + B_2 u_s) + z_f + O(\epsilon)] \\ &= y_s(t) + y_f(t) + O(\epsilon) \end{aligned}$$

where

$$y_s = M_0 x_s + N_0 u_s \text{ and } y_f = M_2 z_f$$

and

$$M_0 \triangleq M_1 - M_2 A_{22}^{-1} A_{21} \text{ and } N_0 \triangleq -M_2 A_{22}^{-1} B_2.$$

5.3 Computer algebra vs. the standard singular perturbation problem

The matrix $K(\epsilon)$ in (5.7) must be partitioned compatibly with the states, x and z, and is the limit of the power series which is conventionally written

$$K_N(\epsilon) = \sum_{i=0}^{N} \epsilon^i \begin{bmatrix} k_{(1,i)} & \epsilon k_{(2,i)} \\ \epsilon k_{(2,i)}^T & \epsilon k_{(3,i)} \end{bmatrix} \tag{5.13}$$

where $k_{(j,i)}$ are appropriately sized matrices (see (5.1)). We shall use k_{ji} synonymously with $k_{(j,i)}$, since this saves space and actually corresponds to the TEX output of NCAlgebra.

The remainder of this section will be devoted to finding formulas for the $k_{(j,i)}$ for $j \in \{1,2,3\}$ and $i \geq 0$.

5.3.1 The zero-th order term of the Riccati equation (constant (i.e., ϵ^0) coefficients)

We begin our investigations of the perturbed control problem by searching for the first term of the series (5.13) consisting of matrices, $k_{(1,0)}$, $k_{(2,0)}$, and $k_{(3,0)}$. We substitute $K_0(\epsilon)$ into (5.5) and take only the zero-th order terms in ϵ of the resulting equations. This is problem (b) mentioned in the introduction. In the next section, Section 5.3.1, we will show how this may be done with the assistance of a computer.

This finally brings us to the subject of our investigations, the manipulation of matrix polynomials with computer algebra methods.

Computer algebra finds the basic equations

We begin by using computer algebra to assist in finding the zero-th order terms in ϵ of the equations given in (5.5).

First, using NCAlgebra, we define the block matrices in (5.6). The matrix, $\begin{bmatrix} a & b \\ c & d \end{bmatrix}$, is represented in Mathematica by `{{a,b},{c,d}}`. The suffix, `[[`j,i`]]`, extracts the element in the j-th row and the i-th column from a given matrix. In NCAlgebra, `MatMult` performs the matrix multiplication operation, `tpMat` performs the symbolic transpose operation, and `**` indicates noncommutative multiplication.

```
A = {{A11,A12},{1/ep A21,1/ep A22}};
B = {{B1},{1/ep B2}}; M = {{M1,M2}};                    (5.14)
```

We also define a K_0.

```
K0 = {{k10,ep k20}, {ep tp[k20],ep k30}};
```

The following Mathematica *function* takes as an argument a matrix K and generates the Riccati (5.5).

```
Riccati[K_] := MatMult[K,A] + MatMult[tpMat[A],K] -
MatMult[K,B,Inv[R],tpMat[B],K] + MatMult[tpMat[M],M]             (5.15)
```

We next use the NCAlgebra command `NCTermsOfDegree`[1]. The following Mathematica commands will extract the 0-th order terms in ϵ, creating the polynomials in (5.18), (5.19), and (5.20).

```
Ep10 = NCTermsOfDegree[ Riccati[K0][[1,1]] , {ep},{0}]
Ep20 = NCTermsOfDegree[ Riccati[K0][[1,2]] , {ep},{0}]
Ep30 = NCTermsOfDegree[ Riccati[K0][[2,2]] , {ep},{0}]           (5.16)
```

The output

Input (5.16) creates three polynomials, the third of which is

```
k30**A22+tp[A22]**k30+tp[M2]**M2
-k30**B2**Inv[R]**tp[B2]**k30.                                  (5.17)
```

When all three are output in TEX, which is done easily by NCAlgebra, we get that `Riccati[K0] = 0` corresponds to the equations

$$\begin{aligned} 0 &= k_{10}\, A_{11} + A_{11}^T\, k_{10} + k_{20}\, A_{21} + A_{21}^T\, k_{20}^T + M_1^T\, M_1 - k_{10}\, B_1 R^{-1}\, B_1^T\, k_{10} \\ &- k_{20}\, B_2\, R^{-1}\, B_1^T\, k_{10} - k_{10}\, B_1 R^{-1}\, B_2^T\, k_{20}^T + k_{20}\, B_2\, R^{-1}\, B_2^T\, k_{20}^T \qquad (5.18) \\ 0 &= k_{20}\, A_{22} - k_{20}\, B_2\, R^{-1}\, B_2^T\, k_{30} + k_{10}\, A_{12} \\ &+ A_{21}^T\, k_{30} + M_1^T\, M_2 - k_{10}\, B_1 R^{-1}\, B_2^T\, k_{30} \qquad (5.19) \\ 0 &= k_{30}\, A_{22} + A_{22}^T\, k_{30} + M_2^T\, M_2 - k_{30}\, B_2\, R^{-1}\, B_2^T\, k_{30} \qquad (5.20) \end{aligned}$$

Simple analysis of the basic equations

Notice that (5.20), the TEX form of (5.17), contains only one unknown k_{30} and has the form of a Riccati equation. Thus k_{30} is uniquely determined by this equation if we assume it is the "stabilizing solution". That is $A_{22} - B_2 R^{-1} B_2^T k_{30}$ has all

[1]`NCTermsOfDegree` takes 3 arguments: a (noncommutative) polynomial, a list of variables, and a set of indices. The command returns an expression such that each term is homogeneous with degree given by the indices. For example, the call `NCTermsOfDegree[ A**B + B**C**C + B**A**C + C**D, {C}, {1} ]` returns `B**A**C + C**D`.

eigenvalues in the strict left half plane and is therefore invertible. We have found in our computer experiments that it is best to make heavy invertibility assumptions, especially at the outset. For computer algebra the key property here is invertibility of $A_{22} - B_2 R^{-1} B_2^T k_{30}$.

We may also motivate the invertibility of $A_{22} - B_2 R^{-1} B_2^T k_{30}$ by purely algebraic arguments as follows. Equation (5.18) contains two unknowns, k_{10} and k_{20}, and equation (5.19) contains all three unknowns, k_{10}, k_{20}, and k_{30}. To solve for k_{20} we use equation (5.19) and call `NCCollect[ Ep20, k20 ]` to get the following relation

```
k20**(A22-B2**Inv[R]**tp[B2]**k30)+tp[A21]**k30 + tp[M1]**M2 -
k10**B1**Inv[R]**tp[B2]**k30 + k10**A12                       (5.21)
```

Upon examination of (5.21) it is immediate that we may give k_{20} explicitly in terms of k_{10} and k_{30} by assuming the invertibility of the parenthesized expression in the above relation, $A_{22} - B_2 R^{-1} B_2^T k_{30}$. We have

$$\begin{aligned} k_{20} &= [-k_{10} A_{12} + k_{10} B_1 R^{-1} B_2^T k_{30} - A_{21}^T k_{30} - M_1^T M_2] \\ &\cdot \quad (A_{22} - B_2 R^{-1} B_2^T k_{30})^{-1}. \end{aligned} \tag{5.22}$$

We use this expression for k_{20} to change (5.18) into an equation involving k_{10} and k_{30}. The unknown matrices k_{i0} could then be found by first using (5.20) to solve for k_{30}, then using our transformed (5.18) to solve for k_{10}, and finally using (5.22) to obtain k_{20}. A better situation would be to have some decoupling of the unknown matrices so that certain unknown matrices could be computed concurrently rather than sequentially. Our next objective is to find decoupled equations which determine the unknown matrices.

Heavy analysis of the basic equations

We will show how (5.18) which involves two unknowns may be replaced by an equation involving only one unknown, k_{10}, so that k_{30} and k_{10} may be computed using two independent Riccati equations. We will use the bulk of our Gröbner basis machinery here.

All algebraic identities which hold

As described in Section 3.2.5, the GBA takes as input a set of polynomials and an order on the variables involved. It outputs a (generally) more desirable set of equations. It is not necessary to know *which* polynomials are needed to derive a certain relation. It is only necessary that all needed polynomials are present in the input. For this reason one generally gives as input to the GBA *all polynomial relations known to hold.* There is no harm in having superfluous (but true) relations in the input. Now we will list the input to our computer algebra program which will generate *all polynomial relations known to hold.*

First the basic relations we wish to study were produced in the previous section, `Ep10`, `Ep20` and `Ep30`; equations (5.18), (5.19), and (5.20).

In light of the slow system terminology introduced above in Sections 5.2.3 and 5.2.4 we make the following abbreviations.

```
Abbreviations = { N0 == - M2**Inv[A22]**B2,
M0 == M1 - M2**Inv[A22]**A21, A0 == A11 - A12 ** Inv[A22]**A21,
B0 == B1 - A12**Inv[A22]**B2, R0 == R + tp[N0]**N0 }           (5.23)
```

We add the abbreviation R_0 for convenience as done in [18], although it is not essential. `Inv[R]` is the NCAlgebra representation of R^{-1}. Since `=` denotes assignment, Mathematica uses `==` to denote equality (for equations).

Several of the matrices or matrix polynomials in our problem are assumed to be invertible. It is common to take the matrices A_{ii} to be of full rank, since otherwise a transformation could be applied to the original system to reduce the size of the state. The matrix $A_{22} - B_2 R^{-1} B_2^T k_{30}$ has already been assumed to be invertible to facilitate the definition of k_{20} in (5.22). The matrices R and R_0 are positive definite and so must be invertible.

We generate the relations which result from these observations with the following command,

```
Inverses = NCMakeRelations[{Inv, R, R0, A0, A11, A22,
(A22 - B2**Inv[R]**tp[B2]**k30) }]                              (5.24)
```

Several of the matrices are known to be self adjoint, and therefore the following relations must hold:

```
SelfAdjoints = { k10 == tp[k10], k30 == tp[k30],
R == tp[R], R0 == tp[R0], Inv[R] == tp[Inv[R]],
Inv[R0] == tp[Inv[R0]] }                                        (5.25)
```

We combine all of our relations with

```
Relations = Union[Ep10, Ep20, Ep30, Abbreviations,
SelfAdjoints, Inverses]                                         (5.26)
```

If $p == 0$ is a true equation, then $tp[p] == 0$ is also. We add these "transposed" equations:

```
AllRelations = NCAddTranspose[Relations]                        (5.27)
```

Orders

In order to find a polynomial in k_{10}, A_0, B_0, M_0, N_0, R_0, and other variables with a minimal number of occurrences of k_{10}, we should create a Gröbner basis for *all polynomial relations known to hold* under the following order.

$$N_0 < M_0 < R_0 < A_0 < B_0 \ll k_{10} \ll \text{ other variables} \ll k_{20} \tag{5.28}$$

The order mentioned in (5.28) is specified using the NCAlgebra command

```
NCAutomaticOrder[ {{N0,M0,R0,A0,B0},{k10}, {B1,B2,M1,M2,R,A11,A12,
A21,A22, Inv[A22- B2**Inv[R]**tp[B2]**k30],
tp[Inv[A22-B2**Inv[R]**tp[B2]**k30]]} {k30},{k20}},
AllRelations ]                                                  (5.29)
```

This command scans `AllRelations` for unassigned letters and places them in the order compatibly with the order given in the first argument.

Finally, the call to make the Gröbner basis is made. This call will create a four iteration partial Gröbner basis from the polynomials included in `AllRelations` and the output will be stored in the file, "FindK10".

```
NCProcess[AllRelations,4,"FindK10",SBByCat->False ]              (5.30)
```

The "option" `SBByCat`, which removes large numbers of "redundant" equations, can be ignored by the reader, since in fact we have turned it off to save time.

The output

The output of the command (5.30) is a set of polynomials which make up the partial Gröbner basis to which *RemoveRedundant* has been applied created from the polynomials in `AllRelations` under the order specified in (5.29). The software we use actually does more than just create a partial Gröbner basis. `NCProcess` removes redundant relations and categorizes the output depending on how many unknowns lie in each relation. Then it automatically sets them in TEX, TEX's the file, and opens a window displaying them using 'xdvi'. In this case a category was found which consisted of a single relation in the one unknown k_{10} which we will refer to as `k10rel`. NCProcess automatically performs `NCCollect[k10rel,k10]` and displays

---

The expressions with unknown variables $\{k_{10}\}$
and knowns $\{A_0,\, B_0,\, M_0,\, N_0,\, A_0^T,\, B_0^T,\, M_0^T,\, N_0^T,\, R_0^{-1}\}$

$$k_{10}\left(A_0 - B_0\, R_0^{-1}\, N_0^T\, M_0\right) + \left(A_0^T - M_0^T\, N_0\, R_0^{-1}\, B_0^T\right) k_{10} + M_0^T\, M_0 - k_{10}\, B_0\, R_0^{-1}\, B_0^T\, k_{10} - M_0^T\, N_0\, R_0^{-1}\, N_0^T\, M_0 \tag{5.31}$$

This gives us a desirable decoupled form for the unknown variables k_{10} and k_{30}. There exist independent Riccati equations for k_{10} (see (5.31)), and k_{30} (see (5.20)) and it is easy to see from the full output that no equations coupling k_{10} and k_{30} exist. After solving these two equations for k_{10} and k_{20}, we may find k_{20}

by substituting k_{10} and k_{30} into (5.22).

The calculation which computes (5.31) is no easy feat by hand, as the substitutions and non-standard notation on pages 116-117 of [18] will attest. The answer we found with Gröbner computer algebra is the same as derived there by hand. After the commands were typed into a file, this computation took less than 3 minutes.

The zero-th order term of the controller

The optimal controller (5.4) has first term in ϵ equal to

$$G = -R^{-1}\begin{bmatrix} B_1^T & \frac{B_2^T}{\epsilon} \end{bmatrix}\begin{bmatrix} k_{(1,0)} & \epsilon k_{(2,0)} \\ \epsilon k_{(2,0)}^T & \epsilon k_{(3,0)} \end{bmatrix}. \tag{5.32}$$

and the previous section tells how to compute the k_{j0}.

Note that in many cases it may be advantageous to have an ϵ independent controller. Such a goal may be achieved by setting the upper right entry of K to zero. This gives us

$$G\begin{bmatrix} x \\ z \end{bmatrix} = \begin{bmatrix} G_{10} & G_{20} \end{bmatrix}\begin{bmatrix} x \\ z \end{bmatrix} \tag{5.33}$$
$$= -R^{-1}(B_1^T k_{(1,0)}x + B_2^T k_{(2,0))}^T x + B_2^T k_{(3,0)}z)$$

where $k_{(i,0)}$ is defined by equations (5.31), (5.22), and (5.20) for i equal to 1,2, and 3 respectively.

5.3.2 The order ϵ term of the Riccati equation

In Section 5.3.1, a controller was presented which does not depend on the parameter ϵ. This is especially appropriate if ϵ represents some very small unknown parameter. In fact, there are many circumstances when the parameter ϵ, while small, is known. In such a case, even though the optimal controller is an infinite power series in ϵ one can make an n^{th} order approximation to $G(\epsilon)$ in (5.4) and arrive at a controller with enhanced performance.

A major obstruction to such an improved approach is the tedious computation required to generate formulas for the coefficients of higher powers of ϵ. We did not find references where anyone generated formulas for coefficients of ϵ higher than 1. The methods in this thesis do, see Section 5.3.3.

As done in [18] we will now obtain decoupled formulas for the matrices $k_{(1,1)}$, $k_{(2,1)}$, and $k_{(3,1)}$ described in (5.13). Our approach will require considerably less work than doing it by hand.

Instead of truncating the series (5.13) to only one term as done in Section 5.3.1 (input (5.14)) here we define symbolic entries for the second term of K as well.

```
K1 = {{ k10, ep k20 }, {ep tp[k20], ep k30 }}
+ ep {{ k11, ep k21 }, {ep tp[k21], ep k31 }}                (5.34)
```

We also append the following abbreviations for the controller discussed above in Section 5.3.1 and defined in equation (5.33). These formulas are standard, [18].

```
Abbreviations = Union[ Abbreviations, {
G10 == -Inv[R]**(tp[B1]**k10 + tp[B2]**tp[k20]),
G20 == -Inv[R]**tp[B2]**k30 } ].                             (5.35)
```

Since $A_{22} + B_2 G_{20} = A_{22} - B_2 R^{-1} B_2^T k_{30}$ was previously assumed to be invertible, we also add the invertibility relation,

```
Inverses = Union[ Inverses,
NCMakeRelations[{Inv, (A22 + B2**G20)}] ].                   (5.36)
```

Extracting the coefficients of ϵ^1

The Riccati expression in K_1, `Riccati[K1]`, is an equation with linear and quadratic terms in ϵ. As done in the last section, the approach here is to equate coefficients of ϵ. Of course, equating coefficients of ϵ^2 is bogus, since the actual power series (5.13) would have $k_{(i,2)}$ which have not been introduced in computer input (5.34). We can extract the coefficients of ϵ in equation (5.5) with the following commands.

```
Ep11 = NCTermsOfDegree[ Riccati[K1][[1,1]] , {ep},{1}]        (5.37)
```

creates the following polynomial

```
ep k11**A11+ep k21**A21+ep tp[A11]**k11+ep tp[A21]**tp[k21]-
ep k10**B1**Inv[R]**tp[B1]**k11-ep k10**B1**Inv[R]**tp[B2]**tp[k21]-
ep k20**B2**Inv[R]**tp[B1]**k11-ep k20**B2**Inv[R]**tp[B2]**tp[k21]-
ep k11**B1**Inv[R]**tp[B1]**k10-ep k11**B1**Inv[R]**tp[B2]**tp[k20]-
ep k21**B2**Inv[R]**tp[B1]**k10 -
ep k21**B2**Inv[R]**tp[B2]**tp[k20]                           (5.38)
```

and

```
Ep21 = NCTermsOfDegree[ Riccati[K1][[1,2]] , {ep},{1}]        (5.39)
Ep31 = NCTermsOfDegree[ Riccati[K1][[2,2]] , {ep},{1}]        (5.40)
```

give similar looking formulas.

Solving for the unknowns

These valid relations can now be added to *all relations known to hold*, (5.14), (5.15), (5.23), (5.25), and (5.24), with the following command. Since the output of the `NCTermsOfDegree[ ]` command includes the variable `ep` and we want just the coefficients of `ep`, we set the unwanted variable, `ep`, to 1. We do this by appending the Mathematica suffix `/.ep->1` to expressions involving `ep`.

```
AllRelations = Union[Ep11/.ep->1, Ep21/.ep->1,Ep31/.ep->1,
Ep10,Ep20,Ep30,
Abbreviations,SelfAdjoints,Inverses]                          (5.41)
```

Considering the analysis done in Section 5.3.1, $k_{(1,0)}$, $k_{(2,0)}$, and $k_{(3,0)}$ (`k10, k20, k30`) can be regarded as known and we are now looking at the second term of the series (5.13), which is made up of these and other knowns and unknowns `k11, k21`, and `k31` introduced above in (5.34). We wish to find simple formulas which determine the unknowns.

With this distinction between known variables and unknown variables, the following order on the variables is appropriate

$$N_0 < M_0 < R < A_0 < B_0 < B_1 < B_2 < M_1 < M_2 < R_0 < A_{11} < A_{12} < A_{21} < A_{22} < G_{10} < G_{20} < k_{30} < k_{20} < k_{10} \ll k_{11} \ll \text{ other variables} \tag{5.42}$$

This order is imposed with the command

```
NCAutomaticOrder[ {{N0,M0,R,A0,B0,
A11,A12,A21,A22,B1,B2,M1,M2,R0,G10,G20,
k30,k20,k10},{k11},{k31},{
Inv[A22- B2**Inv[R]**tp[B2]**k30],
tp[Inv[A22- B2**Inv[R]**tp[B2]**k30]]
},{k21}}, AllRelations ];                        (5.43)
```

We next call `NCProcess` with an iteration count of 3. The computer input is similar to (5.30). With this input, `NCProcess` took less than 7 minutes. The output of this command contains a single relation with 24 terms involving the single unknown matrix $k_{(1,1)}$, `k11rel`. Collecting around the `k11` with the command `NCCollect[ k11rel, k11 ]` gives us the following relation

$$-1\, k_{11}\,(A_0 - B_0\, R_0^{-1}\, B_0^T\, k_{01} - B_0\, R_0^{-1}\, N_0^T\, M_0) + (k_{01}\, B_0\, R_0^{-1}\, B_0^T + M_0^T\, N_0\, R_0^{-1}\, B_0^T - A_0^T)\, k_{11} + A_0^T\, k_{02}\, A_{22}^{-1}\, A_{21} + A_{21}^T\, A_{22}^{T-1}\, k_{02}^T\, A_0 - k_{01}\, B_0\, R_0^{-1}\, B_0^T\, k_{02}\, A_{22}^{-1}\, A_{21} - k_{01}\, B_0\, R_0^{-1}\, B_2^T\, A_{22}^{T-1}\, k_{02}^T\, A_0 - A_0^T\, k_{02}\, A_{22}^{-1}\, B_2\, R_0^{-1}\, B_0^T\, k_{01} - A_0^T\, k_{02}\, A_{22}^{-1}\, B_2\, R_0^{-1}\, N_0^T\, M_0 - A_{21}^T\, A_{22}^{T-1}\, k_{02}^T\, B_0\, R_0^{-1}\, B_0^T\, k_{01} - A_{21}^T\, A_{22}^{T-1}\, k_{02}^T\, B_0\, R_0^{-1}\, N_0^T\, M_0 - M_0^T\, N_0\, R_0^{-1}\, B_0^T\, k_{02}\, A_{22}^{-1}\, A_{21} - M_0^T\, N_0\, R_0^{-1}\, B_2^T\, A_{22}^{T-1}\, k_{02}^T\, A_0 + k_{01}\, B_0\, R_0^{-1}\, B_0^T\, k_{02}\, A_{22}^{-1}\, B_2\, R_0^{-1}\, B_0^T\, k_{01} + k_{01}\, B_0\, R_0^{-1}\, B_0^T\, k_{02}\, A_{22}^{-1}\, B_2\, R_0^{-1}\, N_0^T\, M_0 + k_{01}\, B_0\, R_0^{-1}\, B_2^T\, A_{22}^{T-1}\, k_{02}^T\, B_0\, R_0^{-1}\, B_0^T\, k_{01} + k_{01}\, B_0\, R_0^{-1}\, B_2^T\, A_{22}^{T-1}\, k_{02}^T\, B_0\, R_0^{-1}\, N_0^T\, M_0 + M_0^T\, N_0\, R_0^{-1}\, B_0^T\, k_{02}\, A_{22}^{-1}\, B_2\, R_0^{-1}\, B_0^T\, k_{01} + M_0^T\, N_0\, R_0^{-1}\, B_0^T\, k_{02}\, A_{22}^{-1}\, B_2\, R_0^{-1}\, N_0^T\, M_0 + M_0^T\, N_0\, R_0^{-1}\, B_2^T\, A_{22}^{T-1}\, k_{02}^T\, B_0\, R_0^{-1}\, B_0^T\, k_{01} + M_0^T\, N_0\, R_0^{-1}\, B_2^T\, A_{22}^{T-1}\, k_{02}^T\, B_0\, R_0^{-1}\, N_0^T\, M_0 \tag{5.44}$$

The coefficients of $k_{(1,1)}$ in equation (5.44), the polynomials in parentheses, suggest that we make the following abbreviation[2]

```
F0 == A0-B0**Inv[R0]**(tp[N0]**M0 + tp[B0]**k01)                    (5.45)
```

With computer input (5.45) appended to *all relations known to hold*, `AllRelations`, we can put F_0 low in the order and run `NCProcess` again, creating a new (partial) Gröbner basis. The output of this command contains the aesthetically pleasing relation defining $k_{(1,1)}$

$$-k_{11}\,F_0 - 1\,F_0^T\,k_{11} + A_{21}^T\,(A_{22} + B_2\,G_{20})^{-T}\,k_{20}^T\,F_0 + F_0^T\,k_{20}\,(A_{22} + B_2\,G_{20})^{-1}\,A_{21} + F_0^T\,k_{20}\,(A_{22} + B_2\,G_{20})^{-1}\,B_2\,G_{10} + G_{10}^T\,B_2^T\,(A_{22} + B_2\,G_{20})^{-T}\,k_{20}^T\,F_0. \tag{5.46}$$

This is a simple Lyapunov equation whose solution, $k_{(1,1)}$, is unique as long as `F0`, is Hurwitz. We will therefore regard $k_{(1,1)}$ as *known* from this point forward.

Similar to the zero-th order case the equation defining $k_{(3,1)}$ is an immediate consequence of (5.47) and takes the collected form

$$k_{31}\,\left(A_{22} - B_2\,R^{-1}\,B_2^T\,k_{30}\right) + \left(A_{22}^T - k_{30}\,B_2\,R^{-1}\,B_2^T\right)k_{31} + A_{12}^T\,k_{20} + k_{02}^T\,A_{12} - k_{30}\,B_2\,R^{-1}\,B_1^T\,k_{20} - k_{20}^T\,B_1\,R^{-1}\,B_2^T\,k_{30} \tag{5.47}$$

Also similar to the zero-th order case we have an explicit formula for $k_{(2,1)}$ in terms of $k_{(1,1)}$ and $k_{(3,1)}$. The following relatively simple formula was also in the output of the `NCProcess` command which generated (5.46).

$$k_{21} \rightarrow -1\,k_{11}\,A_{12}\,A_{22}^{-1} - A_{11}^T\,k_{20}\,(A_{22} + B_2\,G_{20})^{-1} - A_{21}^T\,k_{31}\,(A_{22} + B_2\,G_{20})^{-1} - G_{10}^T\,B_1^T\,k_{20}\,(A_{22} + B_2\,G_{20})^{-1} - G_{10}^T\,B_2^T\,k_{31}\,(A_{22} + B_2\,G_{20})^{-1} + k_{11}\,B_0\,R_0^{-1}\,\left(N_0^T\,M_2\,A_{22}^{-1} - B_2^T\,A_{22}^{T-1}\,k_{30}\right) \tag{5.48}$$

[2] More analytic methods can be used, [20], to show that this expression (5.45) is of the form $A_0 + B_0G_0$ where G_0 is the optimal control for the slow part of the LQR optimal problem, (5.10). The focus here is on the computer algebra and the expression F_0 is discovered purely algebraically.

Here, and in the rest of this thesis, an arrow can be interpreted to mean equal sign. Expressions equivalent to (5.46), (5.48), and (5.47) can be found in [18]. Notice that a similar procedure could be done as described in Section 5.3.1 to derive an order ϵ controller.

5.3.3 The order ϵ^2 term of the Riccati equation

At this point the tale is growing long and the weary reader can most likely guess what will be done in this section from the title. For the sake of presenting a formula which has not appeared before we create a three term approximation to $K(\epsilon)$,

```
K2 = {{ k10, ep k20 }, {ep tp[k20], ep k30 }}
+ ep {{ k11, ep k21 }, {ep tp[k21], ep k31 }}
+ ep^2 {{ k21, ep k22 }, {ep tp[k22], ep k32 }},                (5.49)
```

For this problem a three iteration partial Gröbner basis was created and we arrived at formulas defining $k_{(1,2)}$, $k_{(2,2)}$, and $k_{(3,2)}$. Even without running `NCProcess` one sees that $k_{(3,2)}$ satisfies a Riccati equation. This is analogous to the lower order cases.

We found one equation which expresses $k_{(2,2)}$ in terms of $k_{(1,2)}$ and $k_{(3,2)}$. We found a Lyapunov equation in the unknown $k_{(1,2)}$ consisting of 150 lines in Mathematica notation. There were also several equations in the unknowns $k_{(1,2)}$ and $k_{(3,2)}$. In analogy with the lower order cases we expect that these "coupled" equations are redundant and provided no additional information or contraints. Our algebraic software in principle if given enough time can determine this but these algorithms are more computer intensive and did not finish when run on this problem.

We used a version of `NCProcess` which was specialized to display only equations involving the unknowns; $k_{(1,2)}$ and $k_{(2,2)}$. This substantially speeds up run times. Still, our rather formidable conclusion took 21.5 minutes. The formulas can be found at `http://math.ucsd.edu/~ncalg/SingularPerturbation.`

It is gratifying that our Gröbner equation processing techniques prevailed on such a large problem. It leads us to think that many singular perturbation problems are well within the scope of our computer algebra techniques.

5.4 Perturbing singular solutions of the Information State Equation

We would also like to mention that the techniques illustrated on the previous problem apply to other problems. In particular we mention a singular perturbation analysis of an important entity in the output feedback H^∞ control problem, the information state. It corresponds not to fast and slow time scales but to sensors becoming increasingly accurate, for details see [12].

5.4.1 The general problem

Consider the system

$$\begin{aligned} \frac{dx}{dt} &= A(x) + B(x)v && (5.50)\\ out &= C(x) + D(x). && (5.51) \end{aligned}$$

An equation useful in estimation associated to this (details in [12]) is the *information state equation*, **ISE**:

$$\begin{aligned} -\frac{dp}{dt} &= (A(x) + B(x)\cdot v(t))^T \nabla_x p_t(x) - (\nabla_x p_t(x))^T Q(x) \nabla_x p_t(x) && (5.52)\\ &- [C(x) - Dv(t)]^T J [C(x) - Dv(t)] \\ &+ \frac{1}{\epsilon^2}(Lv(t) - x)^T R(Lv(t) - x) \end{aligned}$$

where J is self adjoint.

5.4.2 The linear case

Assuming that the associated vector field is linear and ϵ is fixed, it is known that a solution exists of the form

$$p_t(x) = \frac{1}{2}(x - x_\epsilon)^T P_\epsilon (x - x_\epsilon) + \phi^\epsilon(t), \tag{5.53}$$

where P_ϵ *is a symmetric matrix which does not depend on* t*, but* x *and* x_ϵ *do depend on* t. Then

$$\nabla_x p = P_\epsilon(x - x_\epsilon).$$

Noting that $(Ax + Bv(t))^T \nabla p$ is scalar, we can symmetrize equation (5.52), the information state equation, ISE, and arrive at

$$\begin{aligned} -\frac{dp}{dt} &= [Ax + Bv(t)]^T P_\epsilon (x - x_\epsilon) + (x - x_\epsilon)^T P_\epsilon [Ax + Bv(t)] \qquad (5.54) \\ &- (x - x_\epsilon)^T P_\epsilon Q P_\epsilon (x - x_\epsilon) - [Cx - Dv(t)]^T J [Cx - Dv(t)] \\ &+ \frac{1}{\epsilon^2}(Lv(t) - x)^T R (Lv(t) - x). \end{aligned}$$

where R is positive semi-definite.

In seeking an asymptotic expansion one can try various possible forms for a solution. We have found our computer algebra methods very effective in going from a postulated form of an expansion to explicit formulas for their coefficients. Let us illustrate this with the following postulates for P_ϵ and x_ϵ. P_ϵ is a function of ϵ which has a series form

$$P_\epsilon = \frac{1}{\epsilon} P_{-1} + P_0 + \epsilon P_1 + \epsilon^2 P_2 + \dots \tag{5.55}$$

and we expand x_ϵ as

$$x_\epsilon(t) = x_{\epsilon,0}(t) + \epsilon x_{\epsilon,1}(t) + \epsilon^2 x_{\epsilon,2}(t) + \dots \tag{5.56}$$

The treatment of ϕ^ϵ is less critical as we shall see.

5.4.3 Finding the expansions of unknowns using computer algebra

We now apply NCAlgebra methods very similar to the ones demonstrated in Section 5.3 to the ISE singular perturbation problem. We do not give as much detail, since now the idea should be clear. However, we state our expansions precisely, since this must be done to set notation.

Setting the expansions of unknowns

We will begin by creating a symbolic entry for P_ϵ. This is done with the following computer input

```
Pe = (1/ep) Pm1 + P0 + ep P1 + ep^2 P2 + ep^3 P3 ;                (5.57)
```

where `ep`, the symbolic entry for ϵ, is declared to be commutative and `Pm1, P0, P1, P2`, and `P3`, symbolic entries for P_{-1}, P_0, P_1, P_2, and P_3 respectively, are declared to be noncommutative.

We do not need a formula for p_t itself, since what we use is `GP` $= \nabla_x p_t(x)$. It is

```
GP = Pe ** (x-xe)                                                  (5.58)
```

The x_ϵ described in (5.56) is defined next

```
xe = xe0 + ep xe1 + ep^2 xe2 + ep^3 xe3                            (5.59)
```

We also need the derivative $\frac{dx_\epsilon}{dt}$ and it has the expansion

```
dxe = dxe0 + ep dxe1 + ep^2 dxe2 + ep^3 dxe3.                      (5.60)
```

Since $\phi^\epsilon(t)$ introduced in equation (5.53) does not depend on x, its derivative in t appears and that is on the left side of (5.54). This means that whatever we find as an approximation to P_ϵ and x_ϵ can be put in the right side of the equation

$$\begin{aligned} -\frac{d\phi^\epsilon(t)}{dt} &= -(Bv)^T P_\epsilon x_\epsilon - x_\epsilon^T P_\epsilon Bv + x_\epsilon^T P_\epsilon Q P_\epsilon x_\epsilon \\ &+ (Dv)^T J Dv + \frac{1}{\epsilon^2}(Lv)^T RLv \end{aligned} \tag{5.61}$$

which can be integrated to produce $\phi^\epsilon(t)$. Also we can expand $\frac{d\phi^\epsilon(t)}{dt}$ out a few terms in ϵ and obtain formulas for these terms in the series.

The equations

The ISE (5.52) is implemented next, and the command `NCTermsOfDegree` is used to sort this by powers of ϵ and x. This produced an equation from each coefficient of $\frac{1}{\epsilon^2}$, $\frac{1}{\epsilon}$, ϵ^0, and x, x^T and x and x^T, to produce a set SE of equations. We did not include terms without x and so no ϕ's will appear in SE. Details are strictly analogous to what we just did in Section 5.3 so they will be omitted.

We will assume the matrix A is invertible, by defining its inverse:

```
inverses = {Inv[A]**A - 1 == 0, A**Inv[A] - 1 == 0 };            (5.62)
```

Many of our matrices are known to be self adjoint, so the equations establishing this are also included with SE and `inverses` to obtain the full set of equations we put into our Gröbner basis algorithm.

Applying the Gröbner basis algorithm

We need to select a monomial order. A, B, C, D, J, Q, R, v, and L are known and we certainly do not wish to solve for them, so we set them at the bottom of the order. Since the P_ϵ and x_ϵ, are unknown and must be solved for, we set the symbolic unknowns in their expansion above the knowns in the monomial order. The order we choose on the unknowns puts P_ϵ at the bottom of the unknowns, but still above the knowns which means that once they are solved for they can be used in in formulas for x_ϵ. To implement this and automatically impose a corresponding order on transposes and inverses of variables we use the command

`NCAutomaticOrder` and produce the following order

$$A < A^{-1} < A^T < B < B^T < Q < J < C < C^T <$$
$$R < D < D^T < v < v^T < L < L^T \ll$$
$$Pm_1 < P_0 < P_1 < P_2 < P_3 \ll$$
$$xe_0 < {xe_0}^T \ll xe_1 < {xe_1}^T \ll xe_2 < {xe_2}^T \ll xe_3 < {xe_3}^T \ll$$
$$dxe_0 < {dxe_0}^T \ll dxe_1 < {dxe_1}^T \ll dxe_2 < {dxe_2}^T \ll dxe_3 < {dxe_3}^T$$

Finally a partial Gröbner basis is created and "redundant equations" are removed using the `NCProcess[ ]` command

```
NCProcess[AllRelations,3,"Answer"] ;                                  (5.63)
```

5.4.4 The answer

The output of this command includes the following relations which will give us formulas and differential equations which have been derived purely algebraically. The computations took 3 minutes and 7 seconds.

Relations defining P_ϵ

The relations involving P_ϵ in the output of NCProcess are exactly

---

The expressions with unknown variables $\{Pm_1\}$
and knowns $\{Q, R\}$

$$Pm_1\, Q\, Pm_1 \rightarrow R \tag{5.64}$$

The expressions with unknown variables $\{P_0, Pm_1\}$
and knowns $\{A, Q, A^T\}$

$$P_0\, Q\, Pm_1 \rightarrow \tfrac{1}{2}\, Pm_1\, A + \tfrac{1}{2}\, A^T\, Pm_1 - Pm_1\, Q\, P_0 \tag{5.65}$$

The expressions with unknown variables $\{P_1,\, P_0,\, Pm_1\}$
and knowns $\{A,\, C,\, J,\, Q,\, A^T,\, C^T\}$

$$P_1\, Q\, Pm_1 \to \tfrac{1}{2}\, P_0\, A + \tfrac{1}{2}\, A^T\, P_0 - P_0\, Q\, P_0 - Pm_1\, Q\, P_1 - C^T\, J\, C \tag{5.66}$$

The expressions with unknown variables $\{P_2,\, P_1,\, P_0,\, Pm_1\}$
and knowns $\{A,\, Q,\, A^T\}$

$$P_2\, Q\, Pm_1 \to \tfrac{1}{2}\, P_1\, A + \tfrac{1}{2}\, A^T\, P_1 - P_0\, Q\, P_1 - P_1\, Q\, P_0 - Pm_1\, Q\, P_2 \tag{5.67}$$

Indeed this TeX display is exactly what one sees on the screen. In view of the above equations we have the following recursive formula for the matrices P_i. We begin with the defining relation for P_{-1}.

$$P_{-1}QP_{-1} = R$$

Then we must also have that P_0 satisfies the algebraic Lyapunov equation

$$P_{-1}A + A^T P_{-1} = 2\left(\begin{bmatrix} P_0Q & P_{-1}Q \end{bmatrix}\begin{bmatrix} P_{-1} \\ P_0 \end{bmatrix}\right) \tag{5.68}$$

and P_1 satisfies

$$P_0A + A^T P_0 = 2\left(\begin{bmatrix} P_1Q & P_0Q & P_{-1}Q \end{bmatrix}\begin{bmatrix} P_{-1} \\ P_0 \\ P_1 \end{bmatrix} + C^TJC\right) \tag{5.69}$$

Inspection strongly suggests that the coefficient matrices of higher powers of ϵ are given by the recursive formula

$$P_{i-1}A + A^T P_{i-1} = 2\left(\begin{bmatrix} P_{-1}Q & P_0Q & \cdots & P_iQ \end{bmatrix}\begin{bmatrix} P_i \\ \vdots \\ P_0 \\ P_{-1} \end{bmatrix}\right) \tag{5.70}$$

which alternatively may be written as

$$\begin{aligned} P_i Q P_{-1} + P_{-1} Q P_i &= \frac{1}{2}(P_{i-1} A + A^T P_{i-1}) \\ &- P_{i-1} Q P_0 - P_{i-2} Q P_1 - \ldots - P_1 Q P_{i-2} - P_0 Q P_{i-1} \end{aligned} \tag{5.71}$$

In many cases we will have $Q = BB^T$ where B is an input matrix corresponding to a dynamical system. Since we expect B to be a tall, skinny matrix we also expect $BB^T P_{-1}$ to have a significant number of eigenvalues equal to zero and hence have a high degree of non uniqueness in the P_i solving equation (5.71).

Relations defining $x_\epsilon(t)$

The entire output of NCProcess which involves x_ϵ is exactly the following plus their transposes.

The expressions with unknown variables $\{xe_0\}$
and knowns $\{L, R, v\}$

$$R\,(xe_0 - L\,v) == 0 \tag{5.72}$$

The expressions with unknown variables $\{dxe_0, xe_1, xe_0, Pm_1\}$
and knowns $\{A, B, R, v\}$

$$Pm_1\,dxe_0 \to 2\,R\,xe_1 + Pm_1\,A\,xe_0 + Pm_1\,B\,v \tag{5.73}$$

The expressions with unknown variables $\{dxe_1, dxe_0, xe_2, xe_1, xe_0, P_0, Pm_1\}$
and knowns $\{A, B, C, D, J, R, v, C^T\}$

$$Pm_1\,dxe_1 \to -1\,P_0\,dxe_0 + 2\,R\,xe_2 + P_0\,A\,xe_0 + P_0\,B\,v + Pm_1\,A\,xe_1 - 2\,C^T\,J\,C\,xe_0 + 2\,C^T\,J\,D\,v \tag{5.74}$$

The expressions with unknown variables $\{dxe_2, dxe_1, dxe_0, xe_3, xe_2, xe_1, xe_0, P_1, P_0, Pm_1\}$
and knowns $\{A, B, C, J, R, v, C^T\}$

$Pm_1\, dxe_2 \;\rightarrow\; -1\, P_0\, dxe_1 - P_1\, dxe_0 + 2\, R\, xe_3 + P_0\, A\, xe_1 + P_1\, A\, xe_0 + P_1\, B\, v + Pm_1\, A\, xe_2 - 2\, C^T\, J\, C\, xe_1 \qquad (5.75)$

These relations may be written quite succinctly in the following way.

We have the following relations governing the behavior of the terms of the expansion of x_ϵ introduced in equation (5.56).

$$x_{\epsilon,0} = Lv \text{ on the subspace orthogonal to the kernel of } R \tag{5.76}$$

or in other words

$$R(x_{\epsilon,0} - Lv) = 0. \tag{5.77}$$

The terms $x_{\epsilon,0}$, $x_{\epsilon,1}$, and $x_{\epsilon,2}$ are governed by the pair of differential equations

$$P_{-1}\frac{dx_{\epsilon,0}}{dt} = P_{-1}Ax_{\epsilon,0} + 2Rx_{\epsilon,1} \tag{5.78}$$

$$\begin{aligned} P_{-1}\frac{dx_{\epsilon,1}}{dt} + P_0\frac{dx_{\epsilon,0}}{dt} &= \begin{bmatrix} P_{-1}A & P_0A \end{bmatrix} \begin{bmatrix} x_{\epsilon,1} \\ x_{\epsilon,0} \end{bmatrix} \\ &+ 2Rx_{\epsilon,2} - 2C^TJCx_{\epsilon,0} + 2\left(C^TJD + P_0B\right)v \end{aligned} \tag{5.79}$$

and then the higher terms, $x_{\epsilon,k}$ for $k \geq 2$, satisfy

$$\begin{aligned} \begin{bmatrix} P_{-1} & P_0 & \cdots & P_{k-1} \end{bmatrix} \begin{bmatrix} \frac{dx_{\epsilon,k}}{dt} \\ \vdots \\ \frac{dx_{\epsilon,1}}{dt} \\ \frac{dx_{\epsilon,0}}{dt} \end{bmatrix} &= \begin{bmatrix} P_{-1}A & P_0A & \cdots & P_{k-1}A \end{bmatrix} \begin{bmatrix} x_{\epsilon,k} \\ \vdots \\ x_{\epsilon,1} \\ x_{\epsilon,0} \end{bmatrix} \\ -2C^TJCx_{\epsilon,k-1} &+ 2Rx_{\epsilon,k+1} + P_{k-1}Bv \end{aligned} \tag{5.80}$$

Relations defining $\phi^\epsilon(t)$

There are two ways to proceed for computing $\phi^\epsilon(t)$. An approximation to $\phi^\epsilon(t)$ may be computed using the formula (5.61) along with the expansions just obtained.

The longer method, which we do not reproduce here, is to give full expansions for $\phi^\epsilon(t)$ in terms of $\phi_l^\epsilon(t)$ where $l \geq -2$. We did this with our computer and found rather long formulas for the first few terms. This was easy, but there is no point in listing them here.

Part III

Strategy Theory (and Elimination Ideals)

Chapter 6

Automated Theorem Proving

Our general goal in automated theorem proving is to use computers to assist a human as much as possible in the discovering and proving of mathematical theorems. Donald MacKenzie in his survey article on the history of automated theorem proving [21] classifies the subject into three categories

a. automatic theorem proving where the aim is to simulate human processes of deduction;

b. automatic theorem proving where any resemblance to how humans deduce is considered to be irrelevant; and

c. interactive theorem proving, where the proof is directly guided by a human being.

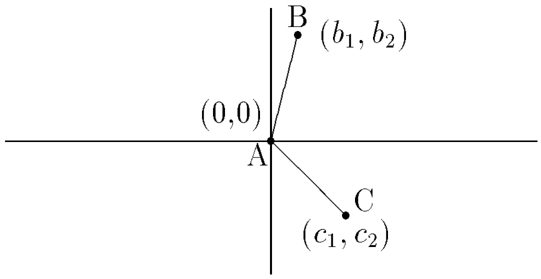
Research in category (a) has ties to artificial intelligence. The investigation done in this thesis and the geometric work described in the following sections falls into category (b). Category (c) is typically done by those who are interested in the validation of hardware designs.

The processes we seek to automate are those which contain a large amount of noncommutative algebra, for example many of the theorems in linear system theory and control. We will show how a computer may be used to perform some noncommutative computations typically done (at least for now) by hand. Since we are not interested in Daniel MacKenzie's category (a) and the functioning of the human mind the particular computations our computer may perform would probably not be done by any sane human being.

In the next section we will briefly mention one of the more successful applications of automatic theorem proving, Euclidean geometry. The computer methods used for this research are closely related to the methods we use.

6.1 Automated geometric theorem proving

One of the more successful applications of automated theorem proving has been to proving theorems about plane geometric objects. The basic idea is as follows. Once cartesian coordinates are introduced into the Euclidean plane hypotheses and conclusions about geometric objects may be expressed as polynomial equations in symbolic indeterminates which represent coordinates. For example the hypothesis that points B and C are the same distance from the point A where the (normalized) coordinates of the points A, B, and C are labeled as follows



may be expressed as $f_1 = b_1^2 + b_2^2 - c_1^2 - c_2^2 = 0$.

A suprising portion of geometric theorems may be proven automatically by various methods of polynomial manipulation. Particularly useful is a reduction method tailored to these geometric problems known as Wu's method.

This work began with Wu Wen-Tsün in 1977 with [25], [26]. His hardware in 1984 consisted of a computer with 512K of RAM. His successor, Shing-Chang

Chou, was able to prove 256 theorems from high school geometry texts and various other sources with a computer [4]. Gröbner basis methods are sometimes used for the polynomial manipulation in these geometric automated proofs.

Chapter 7

A Theory of Strategies

A strategy is a methodology for using a combination of computer assistance and human intervention to to discover highly algebraic theorems. Considerable testing has been done in matrix and linear systems engineering theory. Since the reader is already familiar with the functionality of NCProcess we may describe the pre-strategy (which is slightly less general than the strategy) without further ado. The idea of a pre-strategy is:

S0. The input to the pre-strategy is a set of polynomials.

S1. Run NCProcess which creates a display of the output. Suppose that there is an equation q which involves only one unknown x^*.

S2. The user must now make a decision about equations in x^* (e.g. q is a Riccati equation so I shall not try to simplify it, but solve it using Matlab). Now the user declares the unknown x^* to be known. The user selects q.

S3. The process repeats.

S4. The prestrategy stops when "acceptable" equations have been discovered.

We say a theorem *can be discovered by a* 1*-prestrategy*. In fact the strategy provides such structure that it allows one to classify certain theorems. A question one might ask is "What theorems can be proven using a 1-prestrategy?" In [11] the strategy formalism was introduced and several theorems were proven using pre-strategies or strategies. In Part IV we will offer some new classifications of theorems from linear system theory.

An integral part of our more rigorous Strategy analysis is elimination ideals. This chapter will begin with some necessary definitions and basic lemmas of elimination ideals and grading. We will then be able to classify polynomials as *good*.

The concept of a "good" polynomial to some extent formalizes the statement *the user selects* in *S2*. The importance of such formal studies is that they guide us to what can be automatically be implemented on the computer. Formal definitions lead to rules which can be used to make pre-programed decisions.

We shall define a useful measure of sets of polynomials called *gap*. Proposition 1 shows how there are several different ways to measure this *gap*. This proposition is quite essential to the theory developed in this chapter.

After defining gap we will show its relationship to computationally tractable sets of polynomials. We will discuss how to compute the gap for a given order on the indeterminates using Gröbner basis methods. A more general notion of the *gap of an ideal* is given which doesn't depend on a specific order and is, in fact, the infimum of the ordered gap over all orders.

With the notions of good and gap under our belt we will be able to give a more rigorous analysis of the pre-strategy outlined above. We will show how a successful 1-prestrategy corresponds to a gapless order.

We introduce the strategy which requires the definition of *motivated unknown*. We show how a slight modification to the 1-prestrategy analysis allows one to incorporate these motivated unknowns.

We also introduce the Strategy+ formalism which is new and is in fact even more general than the Strategy, since it allows the user a bit more freedom in adding algebraic hypotheses. This is the first time that the Strategy+ has appeared in print.

We give a more efficient algorithm for finding these successful 1-prestrategies, if such an order exists. To prove the validity of the more efficient algorithm we introduce maps which are *elimination finding* and *singleton finding* in a sense which will be made precise. These maps, in turn, provide greater insight to the strategy process.

Finally the proof of the proposition mentioned above, Proposition 1, is presented.

7.1 Elimination ideals

Recall the definition and properties of elimination ideals introduced in Section 3.4.2. We will make heavy use of elimination ideals in this chapter.

Here we must regrettably give a somewhat non-standard definition. Typically in the literature the j-th elimination ideal for ideal $\mathcal{I} \subset \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$ is defined as

$$\mathcal{I}_j = \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}].$$

We, however, are interested in the relation between an elimination ideal and some sub-elimination ideal contained in it. Notice that if

$$\mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_j] \neq \{0\},$$

then

$$\mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}] \neq \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_j]. \tag{7.1}$$

To see this take, $p \in \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_j]$. The polynomial $x_j p$ will be in the ideal on the right side of (7.1), but not in the smaller ideal on the left side of (7.1). Note that, for $1 \leq j \leq n$, $\mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}]$ is an ideal of $\mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}]$, but is not an ideal of $\mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$.

We will define a *bloated j-th elimination ideal* to be the smallest ideal in $\mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$ which contains the traditional elimination ideal

$$\mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}]$$

of $\mathbb{K}[a_1,\ldots,a_m,x_1,\ldots,x_{j-1}]$. We will always write it in the unambiguous form

$$\langle \mathcal{I} \cap \mathbb{K}[a_1,\ldots,a_m,x_1,\ldots,x_j]\rangle.$$

7.1.1 Grading

A monomial m in $\mathbb{K}[a_1,\ldots,a_m,x_1,\ldots,x_n]$ is said to be *of total degree k* if there exist k indeterminants in m counting multiplicities. For example, $x_1^2x_2x_3a_2x_1$ is of total degree 6 and we write $\mathrm{tdeg}(x_1^2x_2x_3a_2x_1) = 6$. A monomial m in $\mathbb{K}[a_1,\ldots,a_m,x_1,\ldots,x_n]$ is said to be *of x_j-degree k* if x_j appears k times in m. A polynomial in $\mathbb{K}[a_1,\ldots,a_m,x_1,\ldots,x_n]$, p, is said to be *x_j homogenous of x_j-degree k* if every monomial in p is of x_j-degree k. To denote this we will write

$$\deg_{x_j}(p) = k.$$

For example, every term of $x_1x_3^2 + 5x_3x_2a_1x_3 + x_3x_1^5x_3x_2$ is of x_3-degree 2 and so this polynomial is x_3 homogenous of x_3-degree 2. Notice that the following identity will hold for $m \in \mathbb{K}[a_1,\ldots,a_m,x_1,\ldots,x_n]$,

$$\sum_{j=1}^{m} \deg_{a_j}(m) + \sum_{j=1}^{n} \deg_{x_j}(m) = \mathrm{tdeg}(m).$$

Given an algebra $R = \mathbb{K}[a_1,\ldots,a_m,x_1,\ldots,x_n]$ we may consider its *grading by total degree* and write

$$R = R_0 \oplus R_1 \oplus \cdots,$$

where R_d is the vector space of homogenous polynomials of degree d. Given $f \in R$, f may be written uniquely as

$$f = f_0 + f_1 + \cdots \text{ with } f_i \in R_i.$$

Similarly, for an indeterminant x_j, the *grading by x_j degree* of the algebra $R = \mathbb{K}[a_1,\ldots,a_m,x_1,\ldots,x_n]$ is

$$R = R_0^{x_j} \oplus R_1^{x_j} \oplus \cdots,$$

where $R_d^{x_j}$ is the vector space of x_j homogenous polynomials of x_j-degree d.

For example, we may write the polynomial $x_1x_2x_1^2x_3 + 3x_1x_2x_1 + 5x_3 + 4x_1 + a_1a_2a_1 + a_2$ as $(5x_3 + a_1a_2a_1 + a_2) + (4x_1) + (3x_1x_2x_1) + (x_1x_2x_1^2x_3)$ where $5x_3 + a_1a_2a_1 + a_2 \in R_0^{x_1}$, $4x_1 \in R_1^{x_1}$, $3x_1x_2x_1 \in R_2^{x_1}$ and $x_1x_2x_1^2x_3 \in R_3^{x_1}$.

Lemma 2 *Let $p \in \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$ be $p = h_0 + h_1 + h_2 + \ldots + h_r$ where $h_d \in R_d^{x_j}$ for some $j \in \{1, \ldots, n\}$. If $p = 0$, then $h_d = 0$ for all $d \geq 0$.*

Proof: Suppose that $h_k \neq 0$ for some k. Then

$$h_k = -h_0 - h_1 - h_2 - \ldots - h_{k-1} - h_{k+1} - \cdots - h_r. \tag{7.2}$$

Since the left side of (7.2) is x_j homogenous of x_j-degree k we must have

$$\deg_{x_j}(-h_0 - h_1 - h_2 - \ldots - h_{k-1} - h_{k+1} - \cdots - h_r) = k.$$

But this is impossible, since

$$-h_0 - h_1 - h_2 - \ldots - h_{k-1} - h_{k+1} - \cdots - h_r \in \left(\bigoplus_{i=0}^{\infty} R_i^{x_j}\right) \setminus R_k^{x_j}.$$

$\square$

Lemma 3 *Let $p \in \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$ be $p = h_0+h_1+h_2+\ldots+h_r$ where $h_d \in R_d^{x_j}$ for some $j \in \{1, \ldots, n\}$. Let $A \subset \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_l]$. Let NForm$(p, A)$ be with respect to a monomial order $\mathcal{O}$ which is of i-th elimination type for $1 \leq i \leq n$. Then NForm$(h_k, A) \in R_k^{x_j} \cup \{0\}$ for all $l < j$.*

Proof: Let m_i^k be such that

$$h_k = m_1^k + m_2^k + \ldots + m_{r_k}^k.$$

This implies that $\deg_{x_j}(m_i^k) = k$ for all i. Suppose that

$$\mathrm{NForm}(h_k, A) \notin R_k^{x_j} \cup \{0\}.$$

This implies that there exists a rule $\Gamma(q)$, $q \in A$, and a monomial m_t^k such that $m_t^k \xrightarrow{\Gamma(q)} w$ and $w \notin R_k \cup \{0\}$. As described above in Section 3.2.2, the rule $\Gamma(q)$ is of the form

$$L \to S_1 + \ldots + S_s$$

where L and S_i are monomials. Recall that $\deg_{x_j}(m_t^k) = k$. If $w \notin R_k \cup \{0\}$ we must have that

$$\deg_{x_j}(L) \neq \deg_{x_j}(S_a) \text{ for some } a \in \{1, \ldots, s\}.$$

But

$$\deg_{x_j}(L) = 0 \text{ and } \deg_{x_j}(S_i) = 0 \text{ for all } i,$$

since $q \in \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_l]$ and $l < j$. We have found our contradiction and therefore $\mathrm{NForm}(h_k, A) \in R_k^{x_j} \cup \{0\}$. □

7.2 Elimination ideal structure

With our triangularization goal described in Section 1.1 we might ask how a set of elimination ideals is structured for a certain problem. What we wish to investigate is the existence of non-trivial polynomials, *good* polynomials, which determine x_j provided $x_1, \ldots, x_{j-1}$ have already been determined.

We will begin by defining the following two properties, which will prove useful in understanding the ideal structure, good and gap.

7.2.1 Good

First, we describe various criteria for distinguishing a “good” polynomial. Take I and J to be ideals in the polynomial algebra $\mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$.

Definition 7.2.1 *We will write $I \subsetneq J$ to mean that I is a* **proper** *subset of J and we say that an polynomial in their set theoretic difference, $p \in J \setminus I$, (which necessarily exists) is a* **good** *polynomial for J with respect to I.*

7.2.2 Non-redundant

We are often interested in sets of *good* polynomials, that is polynomials which are not merely "consequences" of others in the set. These are polynomials which contain new information.

Definition 7.2.2 *Let X be a set of polynomials. Let us agree to say that X is* **non-redundant** *if $p \notin \langle X \setminus \{p\} \rangle$ for every $p \in X$.*

If X is a set of nonzero polynomials such that $\mathrm{LM}(p) \neq \mathrm{LM}(q)$ for $p, q \in X, p \neq q$ and no order on X is explicitly specified, then we shall use the order $\prec$ induced by the monomial order which is defined by requiring $q \prec p$ if and only if $\mathrm{LM}(p) < \mathrm{LM}(q)$ where '$<$' denotes the monomial order being considered.

Definition 7.2.3 *Let X be a set of polynomials ordered with respect to a ordering $\prec$. X is* **order non-redundant** *if $p \notin \langle \{q : q \prec p\} \rangle$ for every $p \in X$.*

Notice that if the input to the idealized small basis algorithm is ordered, then the output of the idealized small basis algorithm, introduced in Section 3.3.1, is order non-redundant.

Definition 7.2.4 *Given an ordered set of polynomials $P, \{p_1, p_2, \ldots\}$ we call the output of the small basis algorithm applied to P the* **sequentially reduced** *P, $\{p_{n_1}, p_{n_2}, \ldots\}$. Since the idealized small basis algorithm is a deterministic algorithm the output of the idealized small basis algorithm is unique.*

Lemma 4 *Let $\mathcal{G} \subset \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$ be a reduced Gröbner basis with respect to a monomial order $\mathcal{O}$. Let A and B be subsets of $\mathcal{G}$ such that $\langle A \rangle = \langle B \rangle = \langle \mathcal{G} \rangle$. If A and B are both order non-redundant, then $A = B$.*

Proof: Let X be the symmetric difference of A and B, $X = (A \setminus B) \cup (B \setminus A)$. We wish to show that X is the empty set. Suppose for the sake of proof by contradiction that X is not the empty set. Since a monomial order is a total order, we may choose the polynomial p in X with the smallest leading monomial.

Assume that there exists a $q \in A$ such that $q \prec p$. If $q \notin B$, then $q \in A \setminus B \subseteq X$ which contradicts the minimality of p. Therefore we have $q \in B$.

An analogous argument shows that $q \in B$, $q \prec p$ implies that $q \in A$.

Thus for all $q \prec p$ we have $q \in A$ if and only if $q \in B$. That is, for the p we have taken in X with smallest leading monomial we have shown the equivalence of the following two sets:

$$\{q \in B : \mathrm{LM}(q) < \mathrm{LM}(p)\} = \{q \in A : \mathrm{LM}(q) < \mathrm{LM}(p)\}. \tag{7.3}$$

We will next contradict the existence of p. Since $X = (A \setminus B) \cup (B \setminus A)$ there are two cases, $p \in A \setminus B$ or $p \in B \setminus A$.

Case 1 $p \in A \setminus B$

Since $p \in A$ and A is order non-redundant

$$p \notin \langle \{q \in A : \mathrm{LM}(q) < \mathrm{LM}(p)\} \rangle.$$

But since $p \in \mathcal{G}$, we have, by (7.3),

$$p \notin \langle \{q \in B : \mathrm{LM}(q) < \mathrm{LM}(p)\} \rangle.$$

It must be the case that

$$p \in B$$

which is a contradiction.

Case 2 $p \in B \setminus A$

Since $p \in B$ and B is order non-redundant

$$p \notin \langle \{q \in B : \mathrm{LM}(q) < \mathrm{LM}(p)\} \rangle.$$

But since $p \in \mathcal{G}$ and we have, by (7.3),

$$p \notin \langle \{q \in A : \mathrm{LM}(q) < \mathrm{LM}(p)\} \rangle.$$

It must be the case that

$$p \in A$$

which is a contradiction.

We have therefore contradicted the existence of p. We must have X empty and

$$A = B.$$

□

Lemma 5 *Let $\mathcal{G} \subseteq \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$ be a reduced Gröbner basis with respect to a monomial order $\mathcal{O}$ and let $\tilde{\mathcal{G}}$ be the unique (see lemma 4) order non-redundant subset of $\mathcal{G}$ such that $\langle \mathcal{G} \rangle = \langle \tilde{\mathcal{G}} \rangle$. If $\mathcal{O}$ is of j-th elimination type*[1]*, then*

$$\langle \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}] \rangle = \langle \tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}] \rangle. \tag{7.4}$$

Proof: Since $\tilde{\mathcal{G}} \subseteq \mathcal{G}$,

$$\langle \tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}] \rangle \subseteq \langle \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}] \rangle \tag{7.5}$$

which gives us one half of our set inclusion argument.

We next turn to the reverse inclusion

$$\langle \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}] \rangle \subseteq \langle \tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}] \rangle. \tag{7.6}$$

We begin by taking an arbitrary element p of the ideal on the left side of (7.6),

$$p \in \langle \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}] \rangle. \tag{7.7}$$

Since $\mathcal{G}$ is a Gröbner basis under a monomial order $\mathcal{O}$ of j-th elimination type, Theorem 11.3 in [11] implies that $\mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}]$ is a Gröbner basis for

$$\langle \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}] \rangle.$$

There must then exist $c_i \in \mathbb{K}$, L_i, R_i monomials in $\mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$ and polynomials $q_i \in \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}]$ such that

$$p = \sum_{i=1}^{N} c_i L_i q_i R_i. \tag{7.8}$$

---

[1]See Definition 3.2.1

We wish to show that

$$q_i \in \langle \tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_{j-1}] \rangle \tag{7.9}$$

for all i. Consider an arbitrary

$$q_k \in \{q_i : 1, \dots, N\}.$$

We have two cases for q_k.

Case 1:

$$q_k \in \tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_{j-1}]$$

This, of course, implies that

$$q_k \in \langle \tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_{j-1}] \rangle.$$

Case 2:

$$q_k \notin \tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_{j-1}]$$

Recall that

$$q_k \in \mathcal{G} \cap \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_{j-1}]$$

and if q_k has been removed from $\mathcal{G}$ in the creation of the non-redundant subset $\tilde{\mathcal{G}}$, then it was because

$$q_k \in \langle \{g \in \tilde{\mathcal{G}} : g \prec q_k\} \rangle.$$

Since $q_k \in \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_{j-1}]$ and $\mathcal{O}$ is of j-th elimination type, taking any $g \in \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_n]$ such that $g \prec q_k$ will imply

$$g \in \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_{j-1}].$$

Therefore, the following inclusion holds,

$$\langle \{g \in \tilde{\mathcal{G}} : g \prec q_k\} \rangle \subset \langle \tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_{j-1}] \rangle.$$

We must then have

$$q_k \in \langle \tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_{j-1}] \rangle.$$

We have now shown that (7.9) holds, as desired. Equation (7.8) and the definition of ideal imply that

$$p \in \langle \tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{j-1}] \rangle.$$

Equation (7.6) follows.

Equations (7.5) and (7.6) give us the sought for set equivalence, (7.4). □

7.2.3 Gap

In describing the degree to which a set of equations may be triangularized it will be useful to define the notion of gap. Notice that the triangularization is dependent on the existence of a non-trivial or "good" polynomial concept. A set of polynomials can be triangularized with non-redundant polynomials to the extent pictured in (1.1-1.5) if and only if there exists a good q_2 with respect to $\mathcal{I} \cap \mathbb{K}[x_1]$ (in this case there are two), a good q_4 with respect to $\mathcal{I} \cap \mathbb{K}[x_1, x_2]$, etcetera.

Definition 7.2.5 *Given a polynomial ideal $\mathcal{I}$ and order $\mathcal{O}$ we define the* **Gap** *as follows*

$$\begin{aligned} Gap(\mathcal{I}, \mathcal{O}) = \quad & \sup_{i,\gamma}\{\gamma : \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle = \\ & \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma}] \rangle\} \end{aligned} \tag{7.10}$$

We will show below that there are three other equivalent supremums which could be used in the definition of Gap.

Definition 7.2.6 *We call a pair $(\{f_1, \ldots, f_s\}, \mathcal{O})$ such that*

$$Gap(\langle \{f_1, \ldots, f_s\} \rangle, \mathcal{O}) = 0$$

gapless *or* **of full elimination dimension***.*

Given a set of polynomials $\{f_1, \ldots, f_s\}$ with $f_i \in \mathbb{K}[a_1, \ldots, a_k, x_1, \ldots, x_n]$, and an order $\mathcal{O}$ on the unknown indeterminates $x_i \ll x_{i+1}$, and letting $\langle f_1, \ldots, f_s \rangle = \mathcal{I}$ we will show that four identities are equivalent.

Four equivalent definitions

Proposition 1 *Let $\mathcal{I}$ be an ideal of $\mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$. Let $\mathcal{G}$ be a reduced Gröbner basis for $\mathcal{I}$ with respect to a monomial order $\mathcal{O}$ which is of j-th elimination type for $1 \leq j \leq n$. Let $\tilde{\mathcal{G}}$ be the order non-redundant subset of the ordered set $\mathcal{G}$. Let $1 \leq i \leq n$ and $1 \leq \gamma \leq n - i$. The following are equivalent.*

1. *There are no good polynomials in $\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma}] \rangle$ with respect to $\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle$* (7.11)

2. $$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle = \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma}] \rangle \tag{7.12}$$

3. $$\tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] = \tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma}] \tag{7.13}$$

4. *For all $p \in \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i, \ldots, x_{i+\gamma}]$, for all $j \in \{i+1, \ldots, i+\gamma\}$ writing* $$p = h_0^{x_j} + h_1^{x_j} + h_2^{x_j} + \ldots \tag{7.14}$$ *where $h_d^{x_j}$ has degree d in the variable x_j implies $h_d^{x_j} \in \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i]$, for all $d \geq 1$.*

The proof of Proposition 1 is given in Section 7.8.

With this proposition, we may restate the definition of Gap given above in a more flexible way.

Definition 7.2.7 *Given a polynomial ideal $\mathcal{I}$ and order $\mathcal{O}$ we define the* **Gap** *as* *follows*

$$\begin{aligned} Gap(\mathcal{I},\mathcal{O}) &= \sup_{i,\gamma}\{\gamma : \textit{Equation (7.11) is satisfied}\ \} \\ &= \sup_{i,\gamma}\{\gamma : \textit{Equation (7.12) is satisfied}\ \} \\ &= \sup_{i,\gamma}\{\gamma : \textit{Equation (7.13) is satisfied}\ \} \\ &= \sup_{i,\gamma}\{\gamma : \textit{Equation (7.14) is satisfied}\ \} \end{aligned}$$

Note that if $\mathrm{Gap}(\mathcal{I},\mathcal{O}) = \gamma$

$$\langle \mathcal{I} \cap \mathbb{K}[a_1,\dots,a_m,x_{\sigma(1)},\dots,x_{\sigma(i+\gamma^*)}]\rangle = \langle \mathcal{I} \cap \mathbb{K}[a_1,\dots,a_m,x_{\sigma(1)},\dots,x_{\sigma(i)}]\rangle$$

implies that $\gamma^* \leq \gamma$.

Remark 1 *We find that Definitions 7.13 and 7.14 are more useful for doing computations, while Definitions 7.12 and 7.11 illustrate the power of these computations.*

7.3 Triangular collection of equations

The subject of this chapter is the conversion of systems of polynomial equations to triangular form. This type of triangularization has already been mentioned in Sections 1.1 and 4.2.1. In this section we will see how the notion of the Gap introduced in the previous section, Section 7.2.3, helps refine what is meant by triangularization.

If a pair

$$(\langle\{f_1,\dots,f_s\}\rangle, x_1 < \cdots < x_n)$$

is gapless, then there will exist a set of polynomials $\{q_i\}$ whose solution set is equivalent to the $\{f_i\}$ which has the following triangular form:

$$\begin{aligned}
q_1(a_1, \ldots, a_m) &= 0 && (7.15)\\
q_2(a_1, \ldots, a_m) &= 0 && (7.16)\\
&\vdots \\
q_r(a_1, \ldots, a_m) &= 0 && (7.17)
\end{aligned}$$

$$\begin{aligned}
q_{r+1}(a_1, \ldots, a_m, \mathbf{x_1}) &= 0 && (7.18)\\
q_{r+2}(a_1, \ldots, a_m, x_1, \mathbf{x_2}) &= 0 && (7.19)\\
q_{r+3}(a_1, \ldots, a_m, x_1, \mathbf{x_2}) &= 0 && (7.20)\\
q_{r+4}(a_1, \ldots, a_m, x_1, x_2, \mathbf{x_3}) &= 0 && (7.21)\\
&\vdots \\
q_{s_2}(a_1, \ldots, a_m, x_1, \ldots, \mathbf{x_n}) &= 0 && (7.22)
\end{aligned}$$

One can first verify that the solution set exists by verifying equations (7.15-7.17) with the given (known) matrices. We refer to these equations as *compatibility conditions*. Then, solve for the (possibly non-unique) x_1 with equation (7.18), use this to next solve for x_2 with equations (7.19-7.20), etcetera. The defining characteristic of equations (7.15-7.22) is that for every polynomial $q(a_1, \ldots, a_m, x_1, \ldots, x_k)$ there exists $\tilde{q}(a_1, \ldots, a_m, x_1, \ldots, x_{k-1})$.

In Section 7.9 we will introduce, for the sake of completeness, even more computationally tractable solution forms which we refer to as *decoupled*.

7.4 The Gap of an ideal

The discussion in the previous section motivates an interest in orders which correspond to a low Gap. With this in mind we may drop the specification of an order in our definition of Gap and define the Gap of an ideal in the following way.

Definition 7.4.1 *We define the* **Gap of an ideal** $\mathcal{I}$ *to be*

$$Gap(\mathcal{I}) = \inf_{\mathcal{O}} Gap(\mathcal{I}, \mathcal{O}).$$

We now outline ideas which will let us compute $\mathrm{Gap}(\mathcal{I}, \mathcal{O})$. One can use the characterization of bloated elimination ideals (7.13) in the following way.

Given a set P of polynomials which generates the ideal $\mathcal{I}$

1. Create a Gröbner basis $\mathcal{G}$ for $\mathcal{I}$ using Buchberger's (Mora's) algorithm under the monomial order $\mathcal{O}$.

2. Use the small basis algorithm to find the sequentially reduced subset $\tilde{\mathcal{G}}$ of $\mathcal{G}$.

3. Check the elements of $\tilde{\mathcal{G}}$ to determine $\mathrm{Gap}(\mathcal{I}, \mathcal{O})$.

There exists an obvious method for computing $\mathrm{Gap}(\mathcal{I})$ from its definition. Begin by taking all permutations of the indeterminants in $\mathcal{O}$, $\{\mathcal{O}_1, \ldots, \mathcal{O}_{n!}\}$. Compute $\mathrm{Gap}(\mathcal{I}, \mathcal{O}_i)$ for all i using the steps given above. Then

$$\mathrm{Gap}(\mathcal{I}) = \min_i \mathrm{Gap}(\mathcal{I}, \mathcal{O}_i).$$

This method requires $n!$ Gröbner basis computations (under a pure lexicographic-$\ll$-order on the unknowns) and $n!$ applications of the small basis algorithm, a daunting computational task indeed.

Section 7.7 will be devoted to reducing the computational complexity required in determining $Gap(\mathcal{I})$. Indeed, this pursuit will give us a deeper understanding of the strategy process.

7.5 Pre-Strategies

In this discussion, we present the major motivation for the research done in this chapter. We briefly present a structured method for problem solving. For an in-depth introduction, development, and examples of classic theorems from operator, matrix, and linear system engineering theory the reader is refered to [11].

A strategy is a method for solving problems. Many problems contain some algebra or, even if not obviously so, can be formulated to contain some algebra. The strategy process utilizes the algebraic methods described above to add the following variant to Occam's Razor principle.

If one can automate the algebraic portion of a proof, the "simpler" proof will be the proof which requires less work by hand. In fact, it becomes unnecessary to describe the specific algebra techniques used. In our case, we may simply specify the order and cannonical algebraic manipulations (the Gröbner basis algorithm) derive the result. We have seen several instances where three pages of published algebra can be performed in a two minute computer run.

With this tool it becomes reasonable to expend effort on converting a given problem into a more algebraic one, in fact as algebraic as possible. It is difficult to guess a priori how successful one's pursuit of "algebraicness" will be when presented with a problem. Should you climb the stairs or walk to the elevator?

7.5.1 Singletons

Singletons are perhaps the most appreciated type of relation for an algebraic problem solver. The singleton gives an explicit description of an unknown indeterminant, although perhaps in terms of other unknowns. The formal definition follows.

Definition 7.5.1 *For some set of polynomials F we call the members of the set $\{q \in F : tdeg(LM_{\mathcal{O}}(q)) = 1\}$ for some multigraded lexicogaphic order $\mathcal{O}$* **singletons**. *An x singleton is a singleton whose leading monomial consists of x.*

Example 7.5.2 We have a singleton

$$x + 5yzy^2 + zy$$

for the order $z \ll y \ll x$, since

$$\text{LeadTerm}(x + 5yzy^2 + zy) = x.$$

Notice that if an indeterminant x associated with a singleton is placed lexicographically in the order above the other indeterminants, then these singleton indeterminants will only appear once in the output of the Gröbner basis algorithm. This appearance will be in the x-singleton. Any other instance of x will be immediately removed via application of the x singleton replacement rule. A similar situation occurs when a set of singleton indeterminants is placed lexicographically highest in the order and we see that the order within the set is irrelevant to the corresponding Gröbner basis as long as there is a $\ll$ below them.

7.5.2 An idealized pre-strategy

Here we describe an idealized 1-prestrategy. This process is idealized in the sense that we make the assumption that each bloated elimination ideal is generated by one extra polynomial. That is, we assume that

$$\begin{aligned}\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(i)}], p_{\sigma(i)} \rangle = \\ \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(i+1)}] \rangle.\end{aligned} \tag{7.23}$$

We have found this to often be the case in practice, but have also generated counter-examples. We will begin by describing the pre-strategy under this assumption and then decribe the general pre-strategy. Notice that in both definitions Step 1 could be incorporated into Step 3 with a little more notational effort. We believe that leaving Step 1 out of the loop makes these definitions clearer.

Definition 7.5.3 *Let $F_0 \subset \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$, $\mathcal{I} = \langle F_0 \rangle$, and the initial order be $x_1 < \cdots < x_n$. A* **1-prestrategy for** F_0 *consists of the following algorithm.*

1. *There exists "good"*

$$q_1 \in \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}] \rangle$$

with respect to

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m] \rangle$$

for some $\sigma(1) \in \{1, \ldots, n\}$. *We then change the order to* $a_1 < \cdots < a_m \ll x_{\sigma(1)} \ll x_1 \ll \cdots \ll x_{\sigma(1)-1} \ll x_{\sigma(1)+1} \ll \cdots \ll x_n$.[2],

2. *Set* i *to 2.*

3. *There exists "good"*

$$q_i \in \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(i)}]$$

with respect to

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(i-1)}] \rangle$$

for some $\sigma(i) \in \{1, \ldots, n\} \setminus \{\sigma(1), \sigma(2), \ldots, \sigma(i-1)\}$. *We then change the order to* $a_1 < \cdots < a_m \ll x_{\sigma(1)} \ll x_{\sigma(2)} \ll \cdots \ll x_{\sigma(i)} \ll x_1 \ll \cdots \ll x_n$

4. *Set* i *to* $i+1$. *Repeat step 3.*

5. **The stopping criteria:** *On iteration* ℓ

$$\langle I \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(\ell)}] \cup \{q \in F_\ell : tdeg(LM(f)) = 1\} \rangle = \langle F_0 \rangle$$

and we may place the indeterminates $\{x_\tau\}$ *which are not in the image of* σ *(i.e. those associated with singletons) arbitrarily in the order above* $x_{\sigma(\ell)}$ *so that* σ *will be a bijection (permutation)*

$$\sigma : \{1, \ldots, n\} \to \{1, \ldots, n\}.$$

The result is an order

$$\mathcal{O} = a_1 < \cdots < a_m \ll x_{\sigma(1)} \ll \cdots \ll x_{\sigma(n)}.$$

[2] Yes, there's a small abuse of notation here. It's quite possible that $1 \in \sigma$'s.

7.5.3 A nonidealized pre-strategy

Here we describe the pre-strategy where equation (7.23) need not hold.

Definition 7.5.4 *A* **1-prestrategy for a set of polynomials**

$$F_0 \subset \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_n],$$

consists of the following algorithm. Set $\mathcal{I} = \langle F_0 \rangle$*,*

1. *"Get" new set* F_1 *such that* $\langle F_1 \rangle = \mathcal{I}$ *and there exists good*

$$q_1 \in F_1 \cap \mathbb{K}[a_1, \dots, a_m, x_{\sigma(1)}]$$

   with respect to

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \dots, a_m] \rangle$$

   for some $\sigma(1) \in \{1, \dots, n\}$ *we then change the order to* $a_1 < \cdots < a_m \ll x_{\sigma(1)} \ll x_1 \ll \cdots \ll x_{\sigma(1)-1} \ll x_{\sigma(1)+1} \ll \cdots \ll x_n$.

2. *Set* i *to* 2.

3. *"Get" new set* F_i *such that* $\langle F_i \rangle = \langle F_{i-1} \rangle$ *and there exists good*

$$q_i \in F_i \cap \mathbb{K}[a_1, \dots, a_m, x_{\sigma(1)}, \dots, x_{\sigma(i)}]$$

   with respect to

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \dots, a_m], x_{\sigma(1)}, \dots, x_{\sigma(i-1)}] \rangle$$

   for some $\sigma(i) \in \{1, \dots, n\} \setminus \{\sigma(1), \sigma(2), \dots, \sigma(i-1)\}$ *we then change the order to* $a_1 < \cdots < a_m \ll x_{\sigma(1)} \ll x_{\sigma(2)} \ll \cdots \ll x_{\sigma(i)} \ll x_1 \ll \cdots \ll x_n$.

4. *Set* i *to* $i+1$*. Repeat step 3.*

5. **The stopping criteria:** *On iteration* ℓ*:*

$$\langle F_\ell \cap \mathbb{K}[a_1, \dots, a_m, x_{\sigma(1)}, \dots, x_{\sigma(\ell)}] \cup \{q \in F_\ell : tdeg(LM(f)) = 1\} \rangle = \langle F_0 \rangle.$$

We may place the indeterminate $\{x_\tau\}$ which are not in the image of σ (i.e. those associated with singletons) arbitrarily in the order above $x_{\sigma(\ell)}$ so that σ will be a bijection (permutation)

$$\sigma : \{1, \ldots, n\} \to \{1, \ldots, n\}.$$

The result is an order

$$\mathcal{O} = a_1 < \cdots < a_m \ll x_{\sigma(1)} \ll \cdots \ll x_{\sigma(n)}.$$

Notice that the main difference between the two definitions, idealized prestrategy and non-idealized prestrategy, is in steps 1 and 3. In the idealized prestrategy we assume that

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(i)}] \rangle = \langle \{q_i\} \cup (\mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(i-1)}] \rangle,$$

while in the non-idealized pre-strategies it may take several polynomials $q_{i,1}, \ldots, q_{i,r}$ to generate the larger ideal, rather than just q_i.

7.5.4 Gapless pre-strategies

In this section we will examine 1-prestrategies and show their relationship to gapless orders.

Strategies without singletons

For the purpose of analysis it is useful to look at strategies with and without singletons seperately. First we consider the strategy without singletons.

Lemma 2 *A 1-prestrategy for $\{f_1, \ldots, f_s\}$ resulting in order $\mathcal{O}$ with no singletons is of full elimination dimension.*

Proof:
By definition $\text{Gap}(\langle f_1, \ldots, f_s \rangle) \geq 0$. Suppose, for the sake of contradiction, there

exists a σ and j such that

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j)}] \rangle =$$
$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j)}, \ldots, x_{\sigma(j+\gamma)}] \rangle \text{ for } \gamma \geq 1.$$

This implies

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j)}] \rangle =$$
$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j)}, \ldots, x_{\sigma(j+1)}] \rangle.$$

This, however, contradicts our notion of pre-strategy, since by Step 3 we must have

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j)}] \rangle \neq \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j+1)}] \rangle.$$

This is because when we relocated $x_{\sigma(j+1)}$ in the order it was accomplished with a q_{j+1} such that

$$q_{j+1} \in \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j+1)}],$$

but

$$q_{j+1} \notin \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j)}] \rangle,$$

since q_{j+1} was assumed to be *good*. □

Strategies with only singletons

Lemma 3 *A 1-prestrategy for* $\{f_1, \ldots, f_s\}$ *where*

$$\langle f_1, \ldots, f_s \rangle \neq \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$$

resulting in order $\mathcal{O}$ *with only singletons is of full elimination dimension.*

Proof:

Suppose that the 1-prestrategy has resulted in order

$$\{a_1, \ldots, a_m, x_{\tau(1)}, \ldots, x_{\tau(n)}\}$$

where each x_τ has a singleton associated to it. Let $\mathcal{I} = \langle f_1, \ldots, f_s \rangle$.

Now for $i \leq n$, $x_{\tau(i)}$ did not need to be selected because a singleton was found of the form

$$x_{\tau(i)} + p(a_1, \ldots, a_m, x_{\tau(1)}, \ldots, x_{\tau(i-1)}).$$

Assume, for the sake of contradiction, that

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\tau(1)}, \ldots, x_{\tau(i)}] \rangle = \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\tau(1)}, \ldots, x_{\tau(i-1)}] \rangle.$$

Considering the singleton $x_{\tau(i)} + p$ in its graded by $x_{\tau(i)}$ degree form (as developed in Section 7.1.1), we have

$$p(a_1, \ldots, a_m, x_{\tau(1)}, \ldots, x_{\tau(i-1)}) = h_0^{x_{\tau(i)}} \text{ and } x_{\tau(i)} = h_1^{x_{\tau(i)}}.$$

Then, by Proposition 1, we must have

$$x_{\tau(i)} \in \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\tau(1)}, \ldots, x_{\tau(i-1)}] \rangle,$$

but if $x_{\tau(i)}$ is in an ideal which is generated by polynomials which do not contain $x_{\tau(i)}$, then

$$1 \in \langle I \cap \mathbb{K}[a_1, \ldots, a_m, x_{\tau(1)}, \ldots, x_{\tau(i-1)}] \rangle.$$

This implies

$$\mathcal{I} = \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$$

which is contradicted by hypothesis. The lemma follows. □

The general pre-strategy

Theorem 3 *A 1-prestrategy for* $\{f_1, \ldots, f_s\}$ *where*

$$\langle \{f_1, \ldots, f_s\} \rangle \neq \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$$

resulting in order $\mathcal{O}$ *is of full elimination dimension.*

Proof:

Suppose that the 1-pre-strategy has resulted in order

$$\{a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j)}, x_{\tau(1)}, \ldots, x_{\tau(l)}\}$$

where the x_σ were chosen by the "good" criteria and each x_τ has a singleton associated to it. Parenthetically, this implies that $l + j = n$.

By lemma (2), $(\mathcal{I}, \{a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j)}\})$ is of full elimination dimension.

By lemma (3), $(\mathcal{I}, \{a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j)}, x_{\tau(1)}, \ldots, x_{\tau(l)}\})$ is of full elimination dimension. $\square$

Lemma 4 *Conversely, any pair $(\{f_1, \ldots, f_s\}, \mathcal{O})$ may be discovered with a 1-prestrategy with no singletons, provided it is gap free.*

Proof:

Since $\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1] \rangle \neq \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m] \rangle$ there exists some $q_1 \in (\mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1]) \setminus \mathbb{K}[a_1, \ldots, a_m]$. We add q_1 to the initially empty set, F_1, in the manner described in the pre-strategy process. The order generated by the pre-strategy process is equal to $\mathcal{O}$ and so is very transparent.

Similarly there exists

$$q_i \in \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \setminus \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i-1}] \rangle$$

and x_i is kept in it's place. After the n^{th} step of this process we have isolated all of the x's and the process terminates. $\square$

7.6 Strategies

Strategies are a slight generalization of pre-strategies and in fact every pre-strategy is a strategy. The difference between the two methods is that a strategy can incorporate a change of variables through a concept which is described in the next section.

7.6.1 Motivated unknowns

In a prestrategy "unknowns" refers to the original unknowns presented in the problem. However, it is often the case that there will be some combination of variables which is a "natural" unknown for the problem. In this section we will define a particular class of these new unknowns, called *motivated unknowns*.

Suppose that for a polynomial $p(a_1, \ldots, a_m, x_1, \ldots, x_n)$ there exist

$$L(a_1, \ldots, a_m, x_1, \ldots, x_n) \text{ and } R(a_1, \ldots, a_m, x_1, \ldots, x_n)$$

such that

$$LpR = k(a_1, \ldots, a_m, q(a_1, \ldots, a_m, x_{s_1}, \ldots, x_{s_{n_s}}))$$

so that k is a polynomial in one unknown. We will refer to the $\{x_s\}$ as being *subsumed*. By the definition of an ideal, LpR is in the ideal represented by the output of the spreadsheet. The polynomial LpR will not appear on the spreadsheet, since p appears on the spreadsheet. Therefore, in practice, a person must recognize the L and R.

Of course, from the perspective of finding zeros of collections of commutative polynomials, if p has a zero, then LpR has a zero and so k has a zero. Since k is a polynomial in only one unknown variable, finding the zeros of k is bound to be easier than finding the zeros of p.

7.6.2 Finding motivated unknowns in practice: Collect on knowns

The concept of motivated unknowns is good in theory, but they are difficult to find in practice. However, the algorithm described in this section is often useful in finding motivated unknowns. The collected polynomial often has a form which can help the user see that a polynomial in many unknowns is truly a polynomial in one motivated unknown.

NCCollectOnVariables is a command which collects products of knowns out of expressions. For example, if A and B are known and X, Y, and Z are unknown NCCollectOnVariables applied to

$$XABZ + YABZ + AX + AY$$

returns

$$(X+Y)ABZ + A(X+Y),$$

an expression in one motivated unknown $X+Y$ and one unknown Z.

NCCollect defined formally

Given a polynomial p and a set of known variables V, NCCollect first writes the polynomial p as a sum of polynomials which are V homogenous. For each summand, NCCollect writes (to the extent possible) the polynomial into a "parenthesized" form using the rules

$$\begin{aligned}
c_1p_1v + c_2p_1vp_2 &= p_1v(c_1 + c_2p_2) & c_1p_1vp_3 + c_2p_1v &= p_1v(c_1p_3 + c_2) \\
c_1p_3vp_2 &= (c_1p_3 + c_2)vp_2 & c_1vp_2 + c_2p_1vp_2 &= (c_1 + c_2p_1)vp_2 \\
c_1p_1vp_3 + c_2p_1vp_2 &= p_1v(c_1p_3 + c_2p_2) & c_1p_3vp_2 + c_2p_1vp_2 &= (c_1p_3 + c_2p_1)vp_2 \\
c_1v + c_2vp_2 &= v(c_1 + c_2p_2) & c_1v + c_2p_1v &= (c_1 + c_2p_1)v
\end{aligned}$$

where v is a variable in V, c_1 and c_2 are scalars, and p_1, p_2, and p_3 are polynomials.

If none of the above rules apply to a V-homogeneous polynomial, then we say that its only collected form is trivial. If a polynomial is a sum of polynomials whose only collected form is trivial, then we say that this sums only collected form is trivial.

When given a polynomial p and a set of *products of variables* $\{q_1, \ldots, q_n\}$ (where each q_i is a product of variables), NCCollect begins by creating new variables $\{v_1, \ldots, v_n\}$, transforms p by replacing instances of the polynomial q_j in p with v_j, performs NCCollectOnVariables as described above and then replaces v_j with q_j.

NCCollectOnVariables

A key concept for NCCollectOnVariables is *maximal products of knowns* within a monic monomial.

Definition 7.6.1 *$v_a v_{a+1} \ldots v_b$ is a* **maximal product of knowns** *in $v_1 \ldots v_n$ if*

1. *$1 \leq a \leq b \leq n$*
2. *v_j is a known for $a \leq j \leq b$*
3. *if $a > 1$, then v_{a-1} is unknown*
4. *if $b < n$, then v_{b+1} is unknown*

NCCollectOnVariables takes as input a polynomial p. On each term of p, NCCollectOnVariables computes a list of maximal products of knowns. NCCollectOnVariables then repeatedly applies NCCollect to transform p into parenthesized expressions with respect to the maximal products of knowns. NCCollectOnVariables uses the maximal products of knowns with the largest degree before using the maximal products of knowns with lower degrees. For example, if A, B, C, D, E, F and G are knowns and a, b, d, e and h are unknowns, then when NCCollectOnVariables is applied to $abABCd + De + FGhAC$, it returns first tries to collect with respect to $\{ABC\}$ and then collect with respect to $\{FG, AC\}$ and then collect with respect to $\{D\}$.

7.6.3 The strategy defined informally

The idea of a strategy is the similar to that of a prestrategy, except that it incorporates motivated unknowns. The idea of a strategy is :
The input to a strategy is a set of equations C.

1. Set $C' = \{\}$

2. Run NCProcess which creates a display of the output containing polynomial equations to which NCCollectOnVariables has been applied. Look at the list of polynomials involving only one unknown or one motivated unknown (say a particular equation only depends upon the motivated unknown).

3. The user must now make a decision about equations in the output (e.g., q_{42} is a Ricatti equation so I shall not try to simplify it, but leave it for Matlab). Now the user declares a new unknown y and adds the relation as a user creation. The user would also select the equation as important. User selecting an equation corresponds to adding it to the set C'.

4. Either the theorem has been discovered or Go to Step 2.

The above listing is, in fact, a statement of a 1-strategy. Sometimes one needs a 2-strategy in that the key is equations in 1 or 2 unknowns (motivated or not). Typically, if a problem is solved with a 1-strategy, then the computer has done a lot for you, while if the problem requires a 2-strategy, then it has done less, and with a 3-strategy much less.

As with a prestrategy, the point is to isolate and to minimize what the user must do. This is the crux of a strategy.

7.6.4 The formal strategy

Notice that the strategy begins with a set of polynomials

$$\{p_1, p_2, \ldots, p_m\} \tag{7.24}$$

and results in motivated unknowns

$$y_i = r_i(a_1, \ldots, a_m, x_{\tau_1^i}, \ldots, x_{\tau_{n_i}^i}) \tag{7.25}$$

and the polynomial output

$$\{q_1, q_2, \ldots, q_{k_4}\}. \tag{7.26}$$

The sets defined in (7.24), (7.25), and (7.26) are in the polynomial algebra

$$\mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_n, y_1, \dots, y_l].$$

They will satisfy the following ideal equality

$$\langle p_1, p_2, \dots, p_m, y_1 - r_1, \dots, y_l - r_l \rangle = \langle q_1, q_2, \dots, q_{k_4}, y_1 - r_1, \dots, y_l - r_l \rangle. \quad (7.27)$$

Following the formal strategy definition we will give an example of this highly abstract concept. In the following definition we use k_i to denote "known" variables. These are not related to the k_i introduced in Chapter 4. The k_i will assume values x_j or y_j.

Definition 7.6.2 *Let $F_0 \subset \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_n]$, $\mathcal{I} = \langle F_0 \rangle$, and the initial order be $x_1 < \cdots < x_n$. A* **1-strategy for** *F_0 consists of the following algorithm.*

1. *"Get" new set F_1 such that either*

   a. *there exists good*

   $$q_1 \in F_1 \cap \mathbb{K}[a_1, \dots, a_m, x_{\delta(1)}]$$

   with respect to

   $$\langle \mathcal{I} \cap \mathbb{K}[a_1, \dots, a_m] \rangle$$

   for some $\delta(1) \in \{1, \dots, n\}$. We then change the order to

   $$a_1 < \cdots < a_m \ll x_{\delta(1)} \ll x_1 \ll \cdots \ll x_{\delta(1)-1} \ll x_{\delta(1)+1} \ll \cdots \ll x_n.$$

   Set $k_1 = x_{\delta(1)}$.

   b. **or** *create a new "motivated unknown"*

   $$y_1 \in \mathbb{K}[a_1, \dots, a_m, x_{s(1)}, \dots, x_{s(\gamma)}],$$

   $y_1 = r_1(a_1, \dots, a_m, x_{s(1)}, \dots, x_{s(\gamma)})$, such that there exists good

   $$q_1 \in \mathbb{K}[a_1, \dots, a_m, y_1]$$

with respect to

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m] \rangle,$$

$q_1 \in F_1$, $F_1 \subset \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n, y_1]$, *where* $\langle F_1 \rangle = \langle F_0 \cup \{y_1 - r_1\} \rangle$. *Declare* $x_{s(1)}, \ldots, x_{s(\gamma)}$ *as being* subsumed. *Declare* y_1 *to be known. Change the order to*

$$a_1 < \cdots < a_m \ll y_1 \ll x_{\upsilon(1)} \ll \cdots \ll x_{\upsilon(n-\gamma)} \ll x_{s(1)} \ll \cdots \ll x_{s(\gamma)}$$

where $x_\upsilon \notin \{x_{s(1)}, \ldots, x_{s(\gamma)}\}$. *Set* $k_1 = y_1$.

2. Set i *to* 2.

3. "Get" new set F_i *such that* $\langle F_i \rangle = \langle F_{i-1} \rangle$ *and either*

a. there exists good

$$q_i \in F_i \cap \mathbb{K}[a_1, \ldots, a_m, k_1, \ldots, k_{i-1}, x_{\delta(i)}]$$

with respect to

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, k_1, \ldots, k_{i-1}] \rangle$$

for some

$$x_{\delta(i)} \notin \{k_1, \ldots, k_{i-1}\} \cup \{x_s\} \cup \{x_\sigma\}.$$

We then set $k_i = x_{\delta(i)}$ *and change the order to*

$$a_1 < \cdots < a_m \ll k_1 \ll \cdots \ll k_i \ll x_{\upsilon(1)} \ll \cdots \ll x_{\upsilon(n_i)}$$
$$\ll x_{s_1(1)} \ll \cdots \ll x_{s_1(\gamma_1)} \ll \cdots \ll x_{s_i(1)} \ll \cdots \ll x_{s_i(\gamma_i)}$$

where $x_\upsilon \notin \{x_s\}$ *(i.e. It is currently unknown).*

b. **or** *create a new "motivated unknown"*

$$y_i \in \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n, y_1, \ldots, y_{n_y(i)}]$$

$y_i = r_i(a_1, \ldots, a_m, x_{s_i(1)}, \ldots, x_{s_i(\gamma_i)})$, *such that there exists good*

$$q_i \in \mathbb{K}[a_1, \ldots, a_m, k_1, \ldots, k_{i-1}, y_1]$$

with respect to

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, k_1, \ldots, k_{i-1}] \rangle,$$

$q_i \in F_i$,

$$F_i \subset \mathbb{K}[a_1, \ldots, a_m, k_1, \ldots, k_{i-1}, y_i]$$

where $\langle F_i \rangle = \langle F_{i-1} \cup \{y_i - r_i\} \rangle$. *Declare* $x_{s_i(1)}, \ldots, x_{s_i(\gamma_i)}$ *as being subsumed. Declare* y_i *to be known and set* $k_i = y_i$. *Change the order to*

$$a_1 < \cdots < a_m \ll k_1 \ll \cdots \ll k_i \ll x_{\upsilon(1)} \ll \cdots \ll x_{\upsilon(n_i)}$$
$$\ll x_{s_1(1)} \ll \cdots \ll x_{s_1(\gamma_1)} \ll \cdots \ll x_{s_i(1)} \ll \cdots \ll x_{s_i(\gamma_i)}$$
$$\ll x_{\sigma(1)} \ll \cdots \ll x_{\sigma(n_\sigma)}$$

4. *Consider the set* $\{q \in F_i : tdeg(LM(q)) = 1\}$ *(the singletons) add* $LM(q)$ *to the set* $\{x_\sigma\}$ *for all such* q. *Change the order to*

$$a_1 < \cdots < a_m \ll k_1 \ll \cdots \ll k_i \ll x_{\upsilon(1)} \ll \cdots \ll x_{\upsilon(n_i)}$$
$$\ll x_{s_1(1)} \ll \cdots \ll x_{s_1(\gamma_1)} \ll \cdots \ll x_{s_i(1)} \ll \cdots \ll x_{s_i(\gamma_i)} \ll x_{\sigma_1} \ll \cdots \ll x_{\sigma_i}$$

   where

$$x_\upsilon \notin \{x_s\} \cup \{x_\sigma\} \cup \{k_1, \ldots, k_i\}$$

   (i.e. It is currently unknown).

5. *Set* i *to* $i + 1$. *Repeat step 3.*

6. **The stopping criteria:** *On iteration* ℓ

$$\langle F_\ell \rangle = \langle F_0 \cup \{y_1 - r_1, \ldots, y_{j_\ell} - r_{j_\ell}\} \rangle$$

   and the set $\{x_\upsilon\}$ *is empty.*

What follows is a very simple example of a 1-strategy.

Example 7.6.3 Suppose that we have the following set which consists of one relation

$$\{p_1 = A^T(X + Y) + (X + Y)A + (X + Y)R(X + Y) + Q\}$$

where X and Y are unknown and A, R, and Q are known. This set has a gap of size 1 under order

$$A < R < Q \ll X \ll Y \text{ or } A < R < Q \ll Y \ll X,$$

since we believe[3] that the order non-redundant Gröbner basis associated with either order does not contain a polynomial in just X or Y. We can eliminate this gap by creating a motivated unknown

$$y_1 = X + Y.$$

Under the order

$$A < R < Q \ll y_1 \ll X \ll Y$$

the order non-redundant result is an output set

$$\{q_1 = A^T y_1 + y_1 A + y_1 R y_1 + Q, y_1 - (X + Y)\}$$

which has a gap of size 0 if we consider X and Y to subsumed.

Notice that (7.27) holds, since

$$\langle p_1, y_1 - (X + Y)\rangle = \langle q_1, y_1 - (X + Y)\rangle. \tag{7.28}$$

Theorem 4 *Given polynomials $\{p_1, p_2, \ldots, p_s\}$ in knowns $\{a_1, \ldots, a_m\}$ and unknowns $\{x_1, \ldots, x_n\}$. If polynomials $\{q_1, \ldots, q_\ell\}$ in knowns and original and motivated unknowns, $\{y_1, \ldots, y_{n_y}\}$, can be discovered with a 1-Strategy, then there exists an order*

$$a_1 < \cdots < a_m \ll k_1 \ll \cdots \ll k_{n_k} \ll x_{s_1} \ll \cdots \ll x_{s_\gamma} \ll x_{\sigma_1} \ll \cdots \ll x_{\sigma_{n_\sigma}}$$

[3]Since the actual Gröbner basis associated with either order is infinite we don't known every polynomial in the order non-redundant Gröbner basis.

where

$$k_i \in \{x_1, \ldots, x_n, y_1, \ldots, y_{n_y}\} \setminus \{x_{s_1}, \ldots, x_{s_\gamma}, x_{\sigma_1}, \ldots, x_{\sigma_{n_\sigma}}\},$$
$$x_{s_i} \textit{ appears in a } y_j \textit{ for some } j, \textit{ and}$$
$$x_{\sigma_i} \textit{ appears as a singleton,}$$

such that the gaps between

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, k_1, \ldots, k_i] \rangle \textit{ and } \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, k_1, \ldots, k_{i-1}] \rangle$$

are of size 0 *for* $i \in \{1, \ldots, n_k\}$. *Here* $\mathcal{I} = \langle q_1, \ldots, q_\ell \rangle$.

Notice that since $\{x_\sigma\}$ *are singletons we also have gaps between*

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, k_1, \ldots, k_i, x_{s_1}, \ldots, x_{s_\gamma}, x_{\sigma_1}, \ldots, x_{\sigma_i - 1}] \rangle$$
$$\textit{and}$$
$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, k_1, \ldots, k_i, x_{s_1}, \ldots, x_{s_\gamma}, x_{\sigma_1}, \ldots, x_{\sigma_i}] \rangle$$

of size 0 *for* $i \in \{1, \ldots, n_\sigma\}$.

Proof: Lemma 2 shows that the gaps between

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, k_1, \ldots, k_i] \rangle \text{ and } \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, k_1, \ldots, k_{i-1}] \rangle$$

are of size 0 for $i \in \{1, \ldots, n_k\}$. Unknown variables not included in $\{k_i\}$ must be either singletons or subsumed variables, since this is the criteria for stopping a strategy. □

7.6.5 The strategy+

A Strategy+ allows the user a little more freedom than the Strategy and therefore every Strategy is a Strategy+. Specifically, the Strategy+ allows the user to add two different sorts of polynomial hypotheses.

1. Suppose that motivated unknown q appears as the only unknown in a polynomial $p(a, q)$. We allow the user to set the motivated unknown q to equal a polynomial $q = Q(a)$ provided $q = Q(a)$ is a solution to

$$p(a, Q(a)) = 0$$

for all a. Usually this amounts to taking $q = 0$ or $q = 1$. Formalize the notion by replacing the "user sets" with there exists.

2. If there are two polynomials, each in one unknown, say

$$p_1(a, v_1), p_2(a, v_2),$$

then any polynomial relating v_1 and v_2

$$r(v_1, v_2)$$

which holds for all a can be added to the list of relations known to hold (e.g. if $p_1(a, v_1) = 0$ and $p_1(a, v_2) = 0$, then we can add $r = v_1 - v_2$).

In Section 9.2 we give an example of a 2-Strategy+ and discuss ramifications of our looser Strategy definitions to the specific example. The sort of polynomial additions described in the Strategy+ formalism are somewhat mechanical and the computer could easily offer some assistance with these tasks, thereby further automating the discovery process.

7.7 A more efficient algorithm for finding low gap orders

This section is devoted to finding better algorithms for computing Gap($\mathcal{I}$). Obviously, one could compute the Gap for all orders as mentioned in Section 7.4. Here, however, we will show how acceptable orders may be generated (and unacceptable orders eliminated) by computing Gröbner bases under orders which are only elimination orders for certain variables. That is the orders may not contain $\ll$ between *every* unknown variable.

The crux of the problem is finding non-empty elimination ideals. We refer to this in a general sense as *elimination finding*. We primarily use the fact that an idealized run of the Gröbner basis algorithm under an elimination order implies the existence or absence of good polynomials.

7.7.1 Elimination finding

In this section we formally define the sort of maps we seek. We will show how the commonplace Gröbner Basis Algorithm fails to satisfy our requirements.

Let $X \subset \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$, a set (not necessarily finite) of polynomials.

Definition 7.7.1 *Let $\mathcal{P}$ be a function from the power set of $\mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$ into itself. We will say that $\mathcal{P}$ is* **ideal preserving** *if $\langle \mathcal{P}(X) \rangle = \langle X \rangle$.*

Example 7.7.2 Examples of ideal preserving maps are $idealGBA()$ and $partialGBA_\ell()$.

Definition 7.7.3 *Let $\mathcal{P}$ be an ideal preserving map. We say that $\mathcal{P}$ is* **j-elimination finding**, *if for every set $X \subset \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$, then either*

(1) there exists a function $\sigma : \{1, \ldots, j\} \to \{1, \ldots, n\}$ which is injective such that there is a polynomial which involves $x_{\sigma(1)}$ through $x_{\sigma(j)}$ which is in $\mathcal{P}(X)$ *That is, we found an equation in j (or fewer) unknowns.*

(2) or for every function $\sigma : \{1, \ldots, j\} \to \{1, \ldots, n\}$ which is injective,

$$\langle \{g \in X : g \in \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j)}]\} \rangle \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j)}]$$
$$=$$
$$\langle X \rangle \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j)}]$$

There do not exist any equations in j unknowns.

Remark 2 *Here is an example of a situation where the idealized GBA under the order* $a_1 < a_2 < \cdots < a_n \ll x_1 \ll \cdots \ll x_n$ *is* **not** j*-elimination finding.*

Consider the polynomial set

$$\{x_2\, x_2 + a_1\, x_2\, a_2, -x_1 + a_1\, x_2\}$$

and let

$$\mathcal{I} = \langle x_2\, x_2 + a_1\, x_2\, a_2, -x_1 + a_1\, x_2\rangle.$$

Notice that

$$x_2\, x_2 + a_1\, x_2 \ \in \mathcal{I} \cap \mathbb{K}[a_1, a_2, x_2].$$

The Gröbner basis algorithm applied to these equations under the order $a_1 < a_2 \ll x_1 \ll x_2$ results in

$$\{a_1\, x_2 - x_1, x_2\, x_2 + \ x_1\, a_2\}. \tag{7.29}$$

So even though there exists a σ such that there exists a $p \in I \cap \mathbb{K}[a_1, a_2, x_{\sigma(1)}]$ where $\sigma(1) = 2$ the GBA does not find it. Since the actual GBA stops this calculation is equivalent to an idealized GBA calculation and the remark holds for the idealized GBA as well.

7.7.2 An elimination finding map

There does however exist j-th elimination finding maps. (Well, modulo the existence of the idealized GBA.) We present one now.

IdealizedElimFind is designed to be such an "elimination finding function". Basically we systematically go through all necessary orders with the Gröbner basis algorithm to find the sought for ideal.

Definition 7.7.4 IdealizedElimFind$_j$*:*
Given a set of polynomials, $\{f_1, \ldots, f_s\} \subset \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$

1. Let $\pi_i : \{1, \ldots, n\} \to \{1, \ldots, n\}$ for $i \in \{1, \ldots, \frac{n!}{(n-j)!j!}\}$ be a set of permutations such that $\{\pi_r(1), \ldots, \pi_r(j)\} \neq \{\pi_s(1), \ldots, \pi_s(j)\}$ for $r \neq s$. That is for any j-subset U of $\{x_1, \ldots, x_n\}$ there exists a k such that
$$\{x_{\pi_k(1)}, \ldots, x_{\pi_k(j)}\} = U.$$

2. Set i to 1.

3. Run the GBA on $\{f_1, \ldots, f_s\}$ under monomial order $a_1 < \cdots < a_m \ll x_{\pi_i(1)} < \cdots < x_{\pi_i(j)} \ll x_{\pi_i(j+1)} < \cdots < x_{\pi_i(n)}$

4. If the output of the GBA contains a polynomial $p \in \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_j] \setminus \mathbb{K}[a_1, \ldots, a_m]$ **STOP. Output** p.

5. If $i < \frac{n!}{(n-j)!j!}$ set i to $i+1$ and repeat step 3. Otherwise go to step 6.

6. There does not exist a function $\sigma : \{1, \ldots, j\} \to \{1, \ldots, n\}$ which is injective such that there is a polynomial, $p \in \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j)}]$. *That is, there does not exist a new equation in j unknowns.*

Lemma 5 *$IdealizedElimFind_j$ is a j-th elimination finding map.*

Proof:

To show that IdealizedElimFind$_j$ is a j-th elimination finding map we must show that it satisfies conditions (1) and (2) of Definition 7.7.3.

Condition 1:

What we need to prove is that the existence of a "good" p implies the existence of a "good" Gröbner basis element.

Suppose there exists a function $\sigma : \{1, \ldots, j\} \to \{1, \ldots, n\}$ which is injective such that there is a good polynomial,
$$p \in \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j)}] \rangle \setminus \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m] \rangle.$$

By our definition of π there exists a π_ℓ such that

$$\pi_\ell(\sigma(i)) < j \text{ for } i \in \{1, \ldots, j\}.$$

Then the Gröbner basis $\mathcal{G}$ created under the order

$$a_1 < \cdots < a_m \ll x_{\pi_\ell(1)} \ll \cdots \ll x_{\pi_\ell(n)}$$

will satisfy the following set equality

$$\langle \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\pi_\ell(1)}, \ldots, x_{\pi_\ell(j)}] \rangle = \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j)}] \rangle$$

by Lemma 11.2 of [11]. Letting $\mathcal{G} = \{g_i : i \in \mathbb{N}\}$ we may then write

$$p = \sum_{i=1}^{N} L_i g_i R_i$$

where $L_i, R_i \in \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$.

We will next show that one of these g_i must be good with respect to $\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m] \rangle$. Assume, for the sake of contradiction, that all of the g_i do not contain any of the $\{x_{\sigma(1)}, \ldots, x_{\sigma(j)}\}$. That is $g_i \in \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m]$ for all i. This implies that $p \in \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m] \rangle$ which is a contradiction, since p was assumed to be good.

We have shown that the existence of a "good" p implies the existence of a "good" Gröbner basis element.

Condition 2:

Suppose there does not exist such a polynomial in j unknowns. That is there does not exist a function $\sigma : \{1, \ldots, j\} \to \{1, \ldots, n\}$ which is injective such that there is a polynomial, $p \in \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j)}] \setminus \mathbb{K}[a_1, \ldots, a_m]$.

Since the set $\{\pi_i\}$ are simply permutations, the idealized Gröbner basis algorithm under orders $\{\pi_i\}$

$$a_1 < \cdots < a_m \ll x_{\pi_i(1)} \ll \cdots \ll x_{\pi_i(n)}$$

will not generate any

$$p \in \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\pi_\ell(1)}, \ldots, x_{\pi_\ell(j)}] \setminus \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m] \rangle.$$

$\square$

7.7.3 A better elimination finding map

The definition of elimination finding function is a bit too weak for our needs. For example in the case of n linearly independent linear equations in n unknowns n different 1-elimination finding maps could give n different $\sigma(1)$'s. This leads to ambiguity in the strategy process which will be discussed below. What would be more useful is a map which finds *all* j good orders and not just the existence of one.

Definition 7.7.5 *An unordered set, $F \subset \{x_1, \ldots, x_m\}$, will be called a j* subset *if it contains exactly j distinct elements. Since the polynomial algebra*

$$\mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(n)}] = \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$$

for any permutation σ we might write $\mathbb{K}[a_1, \ldots, a_m, U]$ where U is a j subset of unknowns and our meaning should be clear.

Definition 7.7.6 *Let $\mathcal{P}$ be an ideal preserving map. We say that $\mathcal{P}$ is* **all j-elimination finding** *if for every set $X \subset \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$, $\mathcal{P}$ finds all j subsets of $\{x_1, \ldots, x_n\}$, U_α, such that*

$$\langle X \cap \mathbb{K}[a_1, \ldots, a_m, U_\alpha] \rangle \neq \langle X \cap \mathbb{K}[a_1, \ldots, a_m] \rangle.$$

That is, we find all unordered sets $\{\sigma(1), \ldots, \sigma(j)\}$ such that there exists a new equation in these j unknowns.

Fortunately, all elimination finding functions exist as well and we can obtain one with a slight modification to the function $ElimFind$ given in Definition 7.7.4.

Definition 7.7.7 IdealizedAllElimFind$_j$*:*
Given a set of polynomials, $\{f_1, \ldots, f_s\} \subset \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$*, follow the same algorithm as described in* $IdealizedElimFind_j$*, Section 7.7.4, but do not stop when a polynomial* $p \in \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_j] \setminus \mathbb{K}[a_1, \ldots, a_m]$*. Instead add that* j*-subset to a list of* j*-subsets and keep going. When the algorithm terminates (by going through all* j*-subsets of our* n *unknowns) this list will be of the form* $\{U : |U| = j, I \cap \mathbb{K}[a_1, \ldots, a_m, U] \neq I \cap \mathbb{K}[a_1, \ldots, a_m]\}$*, a list of* j*-subsets.*

Example 7.7.8

$$AllElimFind_j(\{x_1 + x_2 + x_3 + x_4 + a_1a_2a_3\})$$

where the x_i are unknown and the a_i are known returns the following lists of j-subsets for the corresponding j's:

$$\{\} \text{ for } j = 1, 2, 3$$

and

$$\{\{x_1, \ldots, x_4\}\} \text{ for } j = 4.$$

Example 7.7.9

$$AllElimFind_j(\{x_1 + a_1, x_2 + a_2, x_3 + a_1, x_4 + a_1a_2\})$$

where the x_i are unknown and the a_i are known returns:

$$\{\{x_1\}, \{x_2\}, \{x_3\}, \{x_4\}\} \text{ for } j = 1,$$
$$\{\{x_1, x_2\}, \{x_1, x_3\}, \{x_1, x_4\}, \{x_2, x_3\}, \{x_2, x_4\}, \{x_3, x_4\}\} \text{ for } j = 2,$$
$$\{\{x_1, x_2, x_3\}, \{x_1, x_2, x_4\}, \{x_1, x_3, x_4\}, \{x_2, x_3, x_4\}\} \text{ for } j = 3,$$
and
$$\{\{x_1, x_2, x_3, x_4\}\} \text{ for } j = 4.$$

Here is a somewhat trivial upper bound on the gap which is based on the cardinality of the output from $AllElimFind_1$.

Proposition 2 *$Gap(\{f_1, \dots, f_s\}, \mathcal{O}) \leq n - |\,AllElimFind_1(\{f_1, \dots, f_s\})\,|$ for any order on the unknowns, $\mathcal{O}$ and so of course we have*

$$Gap(\{f_1, \dots, f_s\}) \leq n - |\,AllElimFind_1(\{f_1, \dots, f_s\})\,|\,.$$

Proof: Without loss of generality let $\{\{x_1\}, \dots, \{x_r\}\}$ be the output of $\mathrm{AllElimFind}_1$. Then

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \dots, a_m, x_j] \rangle \neq \langle \mathcal{I} \cap \mathbb{K}[a_1, \dots, a_m] \rangle \text{ for } j = 1, \dots, r. \tag{7.30}$$

We will show that for any order $\mathcal{O}$

$$\mathrm{Gap}(\{f_1, \dots, f_s\}, \mathcal{O}) \leq n - |\,\mathrm{AllElimFind}_1(\{f_1, \dots, f_s\})\,|\,.$$

For a given order $\mathcal{O}$, $x_{\sigma(1)} \ll \cdots \ll x_{\sigma(n)}$, the gap will be defined using the following ideals,

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \dots, a_m, x_{\sigma(1)}, \dots, x_{\sigma(j)}] \rangle \text{ for } j = 1, \dots, n.$$

Equation (7.30) implies that

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \dots, a_m, x_{\sigma(j)}] \rangle \neq \langle \mathcal{I} \cap \mathbb{K}[a_1, \dots, a_m] \rangle \text{ for } j = 1, \dots, r.$$

That is, there exists a good p_j such that

$$p_j \in \langle \mathcal{I} \cap \mathbb{K}[a_1, \dots, a_m, x_{\sigma(j)}] \rangle \setminus \langle \mathcal{I} \cap \mathbb{K}[a_1, \dots, a_m] \rangle$$

so, in fact,

$$p_j \in \langle \mathcal{I} \cap \mathbb{K}[a_1, \dots, a_m, x_{\sigma(j)}] \rangle \setminus \langle \mathcal{I} \cap \mathbb{K}[a_1, \dots, a_m, x_{\sigma(j-1)}] \rangle.$$

So we have

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(i_0)}] \rangle \neq \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(i_0-1)}] \rangle$$
$$\text{if } \sigma(i_0) \leq r. \tag{7.31}$$

Take the gap to be γ. Then there exists an $l \in \{1, \ldots, n-\gamma\}$ such that

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(l)}] \rangle = \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(l+1)}] \rangle = \ldots$$
$$= \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(l+\gamma)}] \rangle.$$

Assume, for the sake of contradiction, that

$$\mathrm{Gap}(\{f_1, \ldots, f_s\}, \mathcal{O}) > n - |\,\mathrm{AllElimFind}_1(\{f_1, \ldots, f_s\})\,|$$

that is $\gamma > n - r$. This implies that there exist $n - r + 1$ unknowns, $\{x_{\sigma(j)}\}$, such that

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j)}] \rangle = \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j-1)}] \rangle,$$

but by the above equation (7.31) there exist r unknowns which do not have this property. This brings the total number of unknowns to $(n - r + 1) + (r) > n$.

We have found our contradiction, and therefore $\gamma \leq n - r$. $\square$

Generating j gap orders

If we find a good polynomial using, say, $x_{\sigma(1)}$; then we can in some sense consider $x_{\sigma(1)}$ to be known at least in the sense of Section 1.1.1. We could then regard $x_{\sigma(1)}$ as known and use $AllElimFind_1$ taking only $\{x_1, \ldots x_n,\} \setminus x_{\sigma(1)}$ as unknown. Continuing in this fashion we will eventually discover an order of full elimination dimension *if such an order exists.* One might say that each run of $AllElimFind_1$ returns a set of acceptable steps to take. Similarly $AllElimFind_2$ could be used to find 1 Gap orders.

A quicker algorithm will be given below.

7.7.4 A singleton finding map

Given a set of polynomials we would like to have an algorithm at hand to find any singletons in the ideal generated by these polynomials if they exist. Here we describe formally what we expect from such an algorithm and present an algorithm with these qualities.

Definition 7.7.10 *Let $\mathcal{P}$ be an ideal preserving map. We say that $\mathcal{P}$ is* **singleton finding** *if for every set $X \subset \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$ there exists a singleton in $\mathcal{P}(X)$ if there exists a singleton in $\langle X \rangle$.*

Definition 7.7.11 *SingletonFind:*

Given a set of polynomials, $\{f_1, \ldots, f_s\} \subset \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$

1. Let $\pi_i : \{1, \ldots, n\} \to \{1, \ldots, n\}$ for $i \in \{1, \ldots, n\}$ be a set of permutations such that $\pi_r(n) \neq \pi_s(n)$ for $r \neq s$. (π could be, for example, the complete set of cyclic permutations.)
2. Set i to 1.
3. Run the GBA on $\{f_1, \ldots, f_s\}$ under monomial order $a_1 < \cdots < a_m < x_{\pi_i(1)} < \cdots < x_{\pi_i(n-1)} \ll x_{\pi_i(n)}$
4. If the output of the GBA contains an $x_{\pi(n)}$ singleton **STOP. Output this singleton.**
5. If $i < n$ set i to $i+1$ and repeat step 3. Otherwise go to step 6.
6. There does not exist a singleton for any order on the unknowns, $\mathcal{O}$.

Proposition 3 *SingletonFind is a singleton finding function.*

Proof: What we need to show is that the existence of a x_n singleton in $\langle f_1, \dots, f_s\rangle$ implies the existence of a x_n singleton Gröbner basis element. Let $\mathcal{G}$ be a Gröbner basis for $\langle f_1, \dots, f_s\rangle$ created under the order

$$x_{\sigma(1)} < \cdots < x_{\sigma(n-1)} \ll x_n \tag{7.32}$$

where $\sigma : \{1, \dots, n-1\} \to \{1, \dots, n-1\}$ is a permutation map.

Let p be a x_n singleton in the ideal $\langle f_1, \dots, f_s\rangle$,

$$p = x_n + q(x_1, \dots, x_{n-1}).$$

Since $\mathcal{G}$ is a Gröbner basis

$$\mathrm{NForm}(p, \mathcal{G}) = 0.$$

Assume, for the sake of contradiction, that there are no x_n singletons in $\mathcal{G}$. That is g is not a x_n singleton for all $g \in \mathcal{G}$. This implies that either g does not contain x_n or terms involving x_n are of the form cux_nv where u and v are not both 1. For any $g \in \mathcal{G}$ the rule $\Gamma(g)$ will not reduce the x_n term of p. Considering $p - x_n = q(x_1, \dots, x_{n-1})$ no x_n will be introduced by applying a rule $\Gamma(g)$ because any x_n would appear on the left hand side of a rule under the monomial order described above (7.32) and q has no x_n.

Since the x_n term will never be eliminated by application of the rules $\Gamma(\mathcal{G})$ this implies

$$\mathrm{NForm}(p, \mathcal{G}) \neq 0,$$

a contradiction, which implies that a x_n singleton must appear in $\mathcal{G}$.

Alternatively, if there are no x_n singletons in $\langle f_1, \dots, f_s\rangle$ there will be no x_n singletons in $\mathcal{G}$ since $\mathcal{G} \subset \langle f_1, \dots, f_s\rangle$. $\square$

7.7.5 The prestrategy in practice

The algorithm described above in Section 7.7.3 is nice conceptually, but we are disregarding a lot of useful information. What happens in practice is that one would only use such exhaustive search techniques as ElimFind_1 and SingletonFind if necessary. A reasonable practitioner would run the Gröbner basis algorithm for some specified number of iterations, under an order which "looks good", and parse the output looking for polynomials in one unknown and singletons. Hopefully, polynomials in each of these two categories would be found.

As mentioned above on page 99 singletons are, for the problem solver, the best that can be hoped for. Unknowns associated with singletons are "taken out of the mix" by placing them highest in the order. Polynomials involving these singleton indeterminants can be instantly transformed into polynomials which do not involve these indeterminants by applying the singleton rule. In the current section, we will index the singletons with σ's.

The unknowns associated with polynomials in one unknown are appreciated as well. These indeterminates we can regard as knowns and call them *determined*. Further analysis of our polynomial ideal would say that a polynomial involving one unknown, determined unknowns, and knowns is really a polynomial in one unknown. This is really a more verbose description of the backsolving/triangularization structure described on page 30 and pictured in equations (4.3-4.10). We would place these determined unknowns low in the order with the knowns. In the current section we will label these indeterminants with δ.

The order will then be of the form

$$\begin{aligned} a_1 < \cdots < a_m < x_{\delta(1)} < \cdots < x_{\delta(m)} \ll x_{\upsilon(1)} < \cdots < x_{\upsilon(l)} \\ \ll x_{\sigma(1)} < \cdots < x_{\sigma(k)} \end{aligned} \tag{7.33}$$

where the x_υ's are currently unknown.

This process may now be continued by running the Gröbner basis algorithm again under the order specified in (7.33). With luck, the practitioner will find polynomials in one unknown and singletons for the x_υ. This will then increase the size of the sets

$$\{x_\delta\} \text{ and } \{x_\sigma\}$$

and decrease the size of the set

$$\{x_v\}.$$

At some point the 1-prestrategy stops since we do not have a polynomial in one current unknown. At this point we will have current unknowns, $\{x_{v(1)}, \ldots, x_{v(b)}\}$. If we use this set as the unknowns for $ElimFind_1$ and get nothing, then there does not exist a 1-prestrategy without singletons for this polynomial set.

Also take all of these current unknowns and put them above all the rest of the indeterminates to look for singletons.

A concrete example of how this might be done is to cyclicly permute the unknown set $\{x_v\}$ running the GBA under the order

$$\begin{aligned} a_1 < \cdots < a_m < x_{\delta(1)} < \cdots < x_{\delta(m)} \ll x_{v(j)} \ll x_{v(j+1)} < \cdots < x_{v(l)} \\ < x_{v(1)} < \cdots < x_{v(j-2)} \ll x_{v(j-1)} \ll x_{\sigma(1)} < \cdots < x_{\sigma(k)}. \end{aligned} \tag{7.34}$$

If both of these procedures return nothing THEN there does not exist a 1-prestrategy.

We have shown the following.

Theorem 5 *Given that there exists a gap free order*

$$a_1 < \cdots < a_m \ll x_{\sigma(1)} \ll \cdots \ll x_{\sigma(n)}$$

for the polynomial set $\{f_1, \ldots, f_s\}$ *any order on the knowns will discover it as well if the procedure presented in this section is used.*

Notice that the procedure in this section may be implemented on a computer. Indeed, we have created such a function, called `NCXWholeProcess[ ]`. `NCXWholeProcess[ ]` takes four arguments: a set of polynomials, an ordered list of indeterminates, a file name, and an iteration count. This function makes repeated calls to `NCProcess[]`, changing the order as described in this section, attempting to find a gapless order. It creates a set of LaTeX files, each of which corresponds to a Gröbner basis created.

7.8 Proof of Proposition 1

Next we turn to the proof of Proposition 1.

Proof:

This proof will be done in six parts, equation (7.12) implies equation (7.13), equation (7.13) implies equation (7.12), equation (7.12) implies equation (7.14), equation (7.14) implies equation (7.12), equation (7.12) implies equation (7.11), and equation (7.11) implies equation (7.12).

Case (7.12) → (7.13)

By the condition given in (7.12) we have

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle = \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma}] \rangle. \tag{7.35}$$

Assume, for the sake of contradiction, that there exists some $g^* \in \tilde{\mathcal{G}}$ such that

$$g^* \in \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i, \ldots, x_{i+\gamma}] \setminus \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i]. \tag{7.36}$$

By the elimination property of Gröbner bases we have

$$\langle \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle = \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle. \tag{7.37}$$

Since $\tilde{\mathcal{G}}$ is an order non-redundant subset of $\mathcal{G}$,

$$\langle \tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle = \langle \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle$$

by Lemma 5. Since

$$g^* \in \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma}] \subset \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle \tag{7.38}$$

by (7.35), by the previous two equations

$$g^* \in \langle \tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle. \tag{7.39}$$

It suffices to show that $g^* \succ g$ for all $g \in \tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i]$ which contradicts the order non-redundancy of the set $\tilde{\mathcal{G}}$.

We now show

$$g^* \succ g \text{ for all } g \in \tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i].$$

Notice that g^* must contain some x_r for $r \in \{i+1, \ldots, i+\gamma\}$ by (7.36). But g cannot contain any x_r for $r > i$, since $g \in \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i]$. Therefore we have $\mathrm{LeadMon}(g^*) > \mathrm{LeadMon}(g)$ which by our definition of order on polynomials implies that $g^* \succ g$.

Case (7.13) → (7.12)

By the condition given in (7.13)

$$\tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] = \tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma}]. \tag{7.40}$$

We wish to show that

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle = \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i, \ldots, x_{i+\gamma}] \rangle. \tag{7.41}$$

By the elimination property of Gröbner bases we have

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i, \ldots, x_{i+\gamma}] \rangle = \langle \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma}] \rangle.$$

But then by Lemma 5

$$\langle \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma}] \rangle = \langle \tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma}] \rangle.$$

By equation (7.40) we have

$$\langle \tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma}] \rangle = \langle \tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle$$

and by Lemma 5 we get

$$\langle \tilde{\mathcal{G}} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle = \langle \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle.$$

Finally we have, by the Gröbner elimination ideal property, that

$$\langle \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle = \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle. \tag{7.42}$$

The result is shown by following the =’s from (7.41) to (7.42).

Case (7.12) → (7.14)

By condition (7.12) we have

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_i] \rangle = \langle \mathcal{I} \cap \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_{i+\gamma}] \rangle.$$

For $i + 1 \leq j \leq \gamma$ we may consider the x_j grading of our algebra,

$$\mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_n] = \bigoplus_{d=0}^{\infty} R_d^{x_j}$$

where

$$q \in R_d^{x_j} \text{ if and only if } deg_{x_j}(q) = d.$$

Then we may write an arbitrary

$$p \in \mathcal{I} \cap \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_{i+\gamma}]$$

as

$$p = h_0^{x_j} + h_1^{x_j} + h_2^{x_j} + \dots$$

where $h_d^{x_j} \in R_d^{x_j}$. Since

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_i] \rangle = \langle \mathcal{I} \cap \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_{i+\gamma}] \rangle,$$

we have

$$p \in \langle \mathcal{I} \cap \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_i] \rangle. \tag{7.43}$$

Let $\mathcal{G}_i = \mathcal{G} \cap \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_i]$, then by (7.43) and the property of elimination for Gröbner bases, we have

$$\mathrm{NForm}(p, \mathcal{G}_i) = 0.$$

Since $\mathcal{G}_i \subset \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_i]$ and $x_j \notin \mathbb{K}[a_1, \dots, a_m, x_1, \dots, x_i]$, by Lemma 3 and Lemma 2 we have

$$\mathrm{NForm}(h_d, \mathcal{G}_i) = 0 \text{ for } d \geq 1$$

which then implies that

$$h_d \in \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \text{ for } d \geq 1.$$

Case (7.14) → (7.12)

Condition (7.14) implies that for all

$$p \in \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i, \ldots, x_{i+\gamma}],$$

its x_j degree grading for $i+1 \leq j \leq \gamma$,

$$p = h_0^{x_j} + h_1^{x_j} + h_2^{x_j} + \ldots,$$

where $h_d^{x_j}$ is of degree d in x_j, will have the property that

$$h_d \in \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \text{ for } d \leq 1.$$

Assume, for the sake of proof by contradiction,

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle \neq \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma}] \rangle.$$

This then implies the existence of a polynomial p,

$$p \in \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma}] \rangle \setminus \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle. \quad (7.44)$$

Condition (7.44) implies the next two relations,

$$\mathrm{NForm}(p, \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma}]) = 0, \quad (7.45)$$

but

$$\mathrm{NForm}(p, \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i]) \neq 0. \quad (7.46)$$

We will write p in its graded form

$$p = h_0^{i+\gamma} + h_1^{i+\gamma} + h_2^{i+\gamma} + \ldots \text{ where } h_d^{i+\gamma} \text{ is of degree } d \text{ in } x_{i+\gamma}.$$

By the condition given in (7.14) we must have

$$\mathrm{NForm}(h_k^{i+\gamma}, \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i]) = 0$$

for all $k \geq 1$. So

$$\begin{gathered}\mathrm{NForm}(p, \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i]) = \\ \mathrm{NForm}(h_0^{i+\gamma}, \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i])\end{gathered}$$

where $h_0^{i+\gamma} \in \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma-1}]$.

We may then consider the $x_{i+\gamma-1}$ degree grading of $h_0^{i+\gamma}$,

$$h_0^{i+\gamma} = h_0^{i+\gamma-1} + h_1^{i+\gamma-1} + h_2^{i+\gamma-1} + \ldots$$

where $h_d^{i+\gamma}$ is of degree d in $x_{i+\gamma-1}$. Then (7.14) implies that

$$\begin{gathered}\mathrm{NForm}(p, \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i]) = \\ \mathrm{NForm}(h_0^{i+\gamma-1}, \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i])\end{gathered}$$

where $h_0^{i+\gamma-1} \in \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma-2}]$.

Continuing in this fashion we have

$$\mathrm{NForm}(p, \mathcal{G}\cap\mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i]) = \mathrm{NForm}(h_0^{i+1}, \mathcal{G}\cap\mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i])$$

where $h_0^{i+1} \in \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i]$.

We now have that

$$\begin{gathered}\mathrm{NForm}(p, \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i]) = \\ \mathrm{NForm}(h_0^{i+1}, \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i]) \neq 0,\end{gathered} \tag{7.47}$$

but

$$\begin{gathered}\mathrm{NForm}(p, \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma}]) = \\ \mathrm{NForm}(h_0^{i+1}, \mathcal{G} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma}]) = 0.\end{gathered} \tag{7.48}$$

Since $h_0^{i+1} \in \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i]$, this is a contradiction and (7.12) follows.

Case (7.12) → (7.11)

Assume, for the sake of contradiction, that there exists a good polynomial

$$q \in \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma}] \rangle$$

with respect to $\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle$. Then

$$q \notin \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle$$

and therefore

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma}] \rangle \neq \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle$$

which is a contradiction.

Case (7.11) → (7.12)

Assume, for the sake of contradiction,

$$\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma}] \rangle \neq \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle,$$

then there exists a

$$q \in \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_{i+\gamma}] \rangle$$

such that

$$q \notin \langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle.$$

The polynomial q is good with respect to $\langle \mathcal{I} \cap \mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_i] \rangle$ and we have found our contradiction.

$\square$

7.9 Appendix: Decoupled systems of equations

In this section, for the sake of completeness, we will describe solution forms which are better than the "triangular form" given in Section 4.2.1. We call these solution forms decoupled, since a given unknown will appear in an equation with only knowns. They have been mentioned in Sections 4.2.2 and 4.2.3, but we will give a more rigorous definition here.

7.9.1 Essentially decoupled

An even more computationally useful set of matrix equations will have the "essentially decoupled form", the formal definition will follow.

$$\begin{aligned}
q_1(a_1, \ldots, a_m) &= 0 & (7.49)\\
q_2(a_1, \ldots, a_m) &= 0 & (7.50)\\
&\vdots\\
q_m(a_1, \ldots, a_m) &= 0 & (7.51)
\end{aligned}$$

$$\begin{aligned}
q_{m+1}(a_1, \ldots, a_m, \mathbf{x_1}) &= 0 & (7.52)\\
q_{m+2}(a_1, \ldots, a_m, \mathbf{x_1}) &= 0 & (7.53)\\
&\vdots\\
q_{m+s_p}(a_1, \ldots, a_m, \mathbf{x_p}) &= 0 & (7.54)
\end{aligned}$$

$$\begin{aligned}
\mathbf{x_{p+1}} &= q_{m+s_p+1}(a_1, \ldots, a_m, x_1, \ldots, x_p) & (7.55)\\
&\vdots\\
\mathbf{x_n} &= q_{s_2}(a_1, \ldots, a_m, x_1, \ldots, x_p) & (7.56)
\end{aligned}$$

$$\begin{aligned}
q_{s_2+1}(a_1, \ldots, a_m, x_1, \ldots, x_n) &= 0 & (7.57)\\
&\vdots\\
q_{s_3}(a_1, \ldots, a_m, x_1, \ldots, x_n) &= 0 & (7.58)
\end{aligned}$$

A solution of this type is made up of *compatibility conditions*, equations (7.49-7.51) and equations (7.57-7.58); *equations in one unknown*, equations (7.52-7.54); and equations (7.55-7.56) which are *singletons* defined in Definition 7.5.1.

Given such a decoupled set of equations one can use equations (7.52-7.54), $q_{m+1}, \ldots, q_{m+s_p}$, to solve for $x_1, \ldots, x_p$ simultaneously. One must then validate these solutions with the solution compatibility conditions (7.57-7.58). It is then a simple matter to find matrices $x_{p+1}, \ldots, x_n$ by evaluating polynomials

$$q_{m+s_p+1}, \ldots, q_{s_2}.$$

Definition 7.9.1 *Let $\mathcal{I}$ be an ideal of $\mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$. We say that $\mathcal{I}$* **can be essentially decoupled** *if there exist j, $1 \leq j \leq n$; k, $j \leq k \leq n$ and injective maps $\sigma : \{1, \ldots, j\} \to \{1, \ldots, n\}$ and $\tau : \{j+1, \ldots, n\} \to \{1, \ldots, n\} \setminus$ image(σ); a set G^*; and a set $G \subset \mathcal{I}$ such that*

$$G = G_0 \cup G_1 \cup \ldots \cup G_j \cup \{x_{\tau(i)} - g_i(a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j)}) : j+1 \leq i \leq n\} \cup G^*$$

where $G_0 = G \cap \mathbb{K}[a_1, \ldots, a_m]$, no subset of G generates $\mathcal{I}$, and

$$G_i \subset \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(i)}] \setminus \mathbb{K}[a_1, \ldots, a_m], \qquad G_i \neq \emptyset,$$

for $1 \leq i \leq j$.

7.9.2 Formally decoupled

An even more computationally useful set of matrix equations will have the "formally decoupled form" which does not have compatibility conditions on the unknowns. The formal definition will follow.

$$q_1(a_1, \ldots, a_m) = 0 \qquad (7.59)$$

$$q_2(a_1, \ldots, a_m) = 0 \qquad (7.60)$$

$$\vdots$$

$$q_m(a_1, \ldots, a_m) = 0 \qquad (7.61)$$

$$q_{m+1}(a_1, \ldots, a_m, \mathbf{x_1}) = 0 \tag{7.62}$$
$$q_{m+2}(a_1, \ldots, a_m, \mathbf{x_1}) = 0 \tag{7.63}$$
$$\vdots$$
$$q_{m+s_p}(a_1, \ldots, a_m, \mathbf{x_p}) = 0 \tag{7.64}$$

$$\mathbf{x_{p+1}} = q_{m+s_p+1}(a_1, \ldots, a_m, x_1, \ldots, x_p) \tag{7.65}$$
$$\vdots$$
$$\mathbf{x_n} = q_{s_2}(a_1, \ldots, a_m, x_1, \ldots, x_p) \tag{7.66}$$

A solution of this type is made up of *compatibility conditions only on the knowns*, equations (7.59-7.61); *equations in one unknown*, equations (7.62-7.64); and equations (7.65-7.66) which are *singletons* as defined in Definition 7.5.1.

Given such a decoupled set of equations one can use equations (7.62-7.64), $q_{m+1}, \ldots, q_{m+s_p}$, to solve for $x_1, \ldots, x_p$ simultaneously. It is then a simple matter to find matrices $x_{p+1}, \ldots, x_n$ by evaluating polynomials $q_{m+s_p+1}, \ldots, q_{s_2}$.

Definition 7.9.2 *Let $\mathcal{I}$ be an ideal of $\mathbb{K}[a_1, \ldots, a_m, x_1, \ldots, x_n]$. We say that $\mathcal{I}$* **can be formally decoupled** *if there exist j, $1 \leq j \leq n$; k, $j \leq k \leq n$ and injective maps $\sigma : \{1, \ldots, j\} \to \{1, \ldots, n\}$ and $\tau : \{j+1, \ldots, n\} \to \{1, \ldots, n\} \setminus image(\sigma)$ and a set $G \subset \mathcal{I}$ such that*

$$G = G_0 \cup G_1 \cup \ldots \cup G_j \cup \{x_{\tau(i)} - g_i(a_1, \ldots, a_m, x_{\sigma(1)}, \ldots, x_{\sigma(j)}) : j+1 \leq i \leq n\}$$

where $G_0 = G \cap \mathbb{K}[a_1, \ldots, a_m]$, no subset of G generates $\mathcal{I}$, and

$$G_i \subset \mathbb{K}[a_1, \ldots, a_m, x_{\sigma(i)}] \setminus \mathbb{K}[a_1, \ldots, a_m]$$

for $1 \leq i \leq j$.

Notice that both of these solution forms, essentially decoupled and formally decoupled, satisfy the formally backsolvable criteria and we have the following set inclusion relationship.

Formally decoupled $\subset$ Essentially decoupled $\subset$ Formally backsolvable

Part IV

Discovering Formulas in System Theory and Control

In this part there are many theorems from the rather comprehensive book in linear system theory and control, "Robust and Optimal Control" [28], which are solved using the noncommutative Gröbner basis methods.

We will give proofs and "discover" which fit into the strategy formalism or in some cases almost fit into the strategy formalism.

The first chapter, Chapter 8, treats basic properties of Riccati equations. The Riccati equation

$$A^T X + XA + XRX + Q = 0 \tag{7.67}$$

is ubiquitous in linear control theory. See [28] for the particulars. We have seen that many basic facts about Riccati equations follow quickly from noncommutative Gröbner basis methods. To be more precise we have discovered the key formulas in Theorems 13.1, 13.2, 13.3, 13.4, 13.5, 13.6, and 13.9 from Chapter 13 of [28] using a 1-prestrategy.

The second chapter, Chapter 9, treats Theorems 13.25, 13.26, and 13.29 from [28] which concern the positive real and the inner properties of transfer functions.

In Chapter 10, Theorem 8.5 from [28] is "discovered" which gives formulas for dilating a system to all-pass. This theorem is useful in the construction of optimal Hankel norm approximations.

Chapter 11 treats a model uncertainty problem where the perturbation has the coprime factored structure. This theorem appears as Theorem 9.6 in [28].

Chapter 8

A Development of Algebraic Riccati Equations Using Noncommutative Gröbner Bases

Here we study properties and solutions of the algebraic Ricatti equation (ARE). If A, Q, and R are real, square matrices of similar size where Q and R are symmetric, then an ARE in X is the following equation:

$$A^T X + XA + XRX + Q = 0 \tag{8.1}$$

Often useful in studying the Algebraic Ricatti Equation is the so called Hamiltonian matrix:

$$H \triangleq \begin{bmatrix} A & R \\ -Q & -A^T \end{bmatrix} \tag{8.2}$$

Here we wish to show that H has a symmetric spectrum about the $j\omega$-axis. This can be accomplished by showing that there exists an invertible matrix, say J, such that $J^{-1}HJ = -H^*$. We could discover J by the methods given here, but let us for the moment just verify that $J \triangleq \begin{bmatrix} 0 & -I \\ I & 0 \end{bmatrix}$ accomplishes the task at hand.

The Mathematica input:[1]

```
J = {{0,-1},{1, 0}};
Hamiltonian = {{A,R},{-Q,-tp[A] }};
tp[Q] = Q; tp[R] = R;
```

```
MatMult[J,Hamiltonian,Inverse[J]] - (-tpMat[Hamiltonian])        (8.3)
```

returns

```
{{0,0},{0,0}}                                                    (8.4)
```

which verifies the above assertion.

8.1 Solving the ARE: an invariant subspace of H

We begin by showing the relationship of the two newly defined entities, (7.67) and (8.2). We show how one might discover the solution of the above ARE through an analysis of the Hamiltonian matrix. We begin by proving the following theorem, Theorem 13.1 in [28].

Theorem 6 *Let $V \subset \mathbb{C}^{2n}$ be an n-dimensional invariant subspace of H, and let $X_1, X_2 \in \mathbb{C}^{n\times n}$ be two complex matrices such that*

$$V = Im \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}.$$

If X_1 is invertible, then $X = X_2 X_1^{-1}$ is a solution to the Riccati equation 7.67 and $\sigma(A + RX) = \sigma(H\mid_V)$. Furthermore, the solution X is independent of a specific choice of V.

Proof:

Begin by defining an n-dimensional invariant subspace of H, $\text{Im}\begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$. Since our

[1] This instance of Mathematica input is a little different than the Mathematica input presented in the rest of this document, since it does not involve Gröbner Bases.

subspace is invariant there must exist a matrix L such that

$$\begin{bmatrix} A & R \\ -Q & -A^T \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} L \tag{8.5}$$

Notice that to get this result we need not set Q and R symmetric.

```
Hamiltonian = {{A,R},{-Q,-tp[A]}};
AllRelations = MatMult[Hamiltonian,
{{X1},{X2}}] - MatMult[{{X1},{X2}},{{L}} ];

AllRelations = Join[ AllRelations, NCMakeRelations[{Inv,X1}]];
SetMonomialOrder[ {A,tp[A],R,X1,X2,L,Inv[X1]},{Q} ];
NCProcess[AllRelations,3,"Discover"];                          (8.6)
```

This results in the following output.

THE ORDER IS NOW THE FOLLOWING:

$A < A^T < R < X_1 < X_2 < L < X_1^{-1} \ll Q < L$

THE FOLLOWING VARIABLES HAVE BEEN SOLVED FOR:

$\{L, Q\}$

The corresponding rules are the following:

$$L \to X_1^{-1}\, A\, X_1 + X_1^{-1}\, R\, X_2 \tag{8.7}$$

$$Q \to -1\, X_2\, X_1^{-1}\, A - A^T\, X_2\, X_1^{-1} - X_2\, X_1^{-1}\, R\, X_2\, X_1^{-1} \tag{8.8}$$

From the category $\{Q, L\}$ we have in equation (8.8) the solution to the ARE (7.67). Obviously $X = X_2 X_1^{-1}$.

We also see that $\sigma(A + RX) = \sigma(L)$: Multiplying the $X_1^{-1}\, R\, X_2$ term of the equation in L given in the spreadsheet, (8.7) by $X_1^{-1} X_1$ and making the substitution for X given above we have the appropriate similarity transformation. That is $X_1^{-1}(A + RX)X_1 = L$.

In particular if we have taken the invariant subspace corresponding to the stable subspace of H (recall that n of H's eigenvalues are in $\mathbb{C}_-$), then $\sigma(L) \subset \mathbb{C}_-$ and $A + RX$ is stable. $\square$

8.2 Converse

Next we discover the converse, the solution to the ARE (7.67) corresponds to an invariant subspace of the Hamiltonian (8.2). This is Theorem 13.2 in [28].

Theorem 7 *If $X \in \mathbb{C}^{n\times n}$ is a solution to the Riccati equation (7.67), then there exist matrices $X_1, X_2 \in \mathbb{C}^{n\times n}$, with X_1 invertible, such that $X = X_2 X_1^{-1}$ and the columns of $\begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$ form a basis of an n-dimensional invariant subspace of H.*

Proof:

Discover solution

Take X to be a solution to the ARE (7.67) and let our invariant subspace have parameterization $\begin{bmatrix} I \\ X_2 \end{bmatrix}$. Taking the parameterization $\begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$ where X_1 is invertible led to no useful results. So we set $X_1 = I$ to reduce freedom in the system of equations. This might seem like a drastic reduction, but experience has shown that if this is incompatible it will be obvious quickly. From our experience with Theorem 13.1 we expect X_1 to be invertible and so do not pick $X_1 = 0$.

```
Hamiltonian = {{A,R},{-Q,-tp[A]}};
AllRelations =
MatMult[Hamiltonian, {{1},{X2}}] - MatMult[{{1},{X2}}, {{L}} ];

AllRelations = Join[ AllRelations, { tp[A]**X + X**A + X**R**X + Q }
];
```

```
SetMonomialOrder[ {X,Q,A,tp[A],R,Inv[X1],X1,X2}, {L} ];
NCProcess[AllRelations,3,"Solution1"];                        (8.9)
```

The output:

THE ORDER IS NOW THE FOLLOWING:

$X < Q < A < A^T < R < X_1^{-1} < X_1 < X_2 \ll L$

THE FOLLOWING VARIABLES HAVE BEEN SOLVED FOR:

$\{L\}$

The corresponding rules are the following:

$L \to A + R\, X_2$

The expressions with unknown variables $\{\}$

and knowns $\{A,\, Q,\, R,\, X,\, X_2,\, A^T\}$

$X\, R\, X \to -1\, Q - X\, A - A^T\, X$

$X_2\, R\, X_2 \to -1\, Q - X_2\, A - A^T\, X_2$

This suggests that $X_2 = X$.

Verify solution

We will have found an invariant subspace of H, $Im \begin{bmatrix} I \\ X \end{bmatrix}$, if

$$H \begin{bmatrix} I \\ X \end{bmatrix} - \begin{bmatrix} I \\ X \end{bmatrix} L - \begin{bmatrix} chk_1 \\ chk_2 \end{bmatrix} = 0$$

implies that both chk_1 and chk_2 are 0. This can be verified easily with our computer algebra methods. We supply the following computer input.

```
L = A + R**X;
```

```
AllRelations = Flatten [{{chk1},{chk2}}+
MatMult[Hamiltonian, {{1},{X}}] -
MatMult[{{1},{X}}, {{L}} ] ];

AllRelations = Join[ AllRelations, {
tp[A]**X + X**A + X**R**X + Q } ];

SetMonomialOrder[{X,Q,A,tp[A],R,Inv[X1]},{chk1,chk2}];
NCProcess[AllRelations,3,"Solution2"];                    (8.10)
```

This computer input resulted in the following LaTeX output:

THE FOLLOWING VARIABLES HAVE BEEN SOLVED FOR:

$\{chk_1, chk_2\}$

The corresponding rules are the following:

$chk_1 \rightarrow 0$

$chk_2 \rightarrow 0$

So $Im \begin{bmatrix} I \\ X \end{bmatrix}$ is an invariant subspace of H where $X = X_2 X_1^{-1} = X I^{-1}$. $\square$

8.3 Special Riccati solutions

Next we find hermitian solutions to the ARE (7.67) by proving Theorem 13.3 from [28]:

Theorem 8 *Let V be an n-dimensional H-invariant subspace and let $X_1, X_2 \in$*

$\mathbb{C}^{n\times n}$ be such that

$$V = Im \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}.$$

Then $\lambda_i + \bar{\lambda}_j \neq 0$ for all $i, j = 1, \ldots, n, \lambda_i, \lambda_j \in \sigma(H|_V)$ implies that $X_1^* X_2$ is hermitian. Furthermore, if X_1 is nonsingular, then $X = X_2 X_1^{-1}$ is hermitian.

Proof:
Again take $\begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$ as a parameterization of an invariant subspace. Create an indeterminant $wanted = X_2^* X_1 - X_1^* X_2$ to analyze the hermitian properties of $X_2^* X_1$. If it can be shown in some way that $wanted = 0$ we will have shown $X_1^* X_2$ is hermitian.

```
AllRelations = MatMult[Hamiltonian, {{X1}, {X2}}]
- MatMult[{{X1},{X2}}, {{L}} ];

AllRelations = Union[ AllRelations, NCMakeRelations[
{Inv, X1}] , {wanted == tp[X2]**X1 - tp[X1]**X2,
Q == tp[Q], R == tp[R] } ];

AllRelations = NCAddTranspose[AllRelations];

NCAutomaticOrder[{{L,wanted},{X1,X2,Q,A,tp[A],R,Inv[X1]}},
AllRelations];
NCProcess[AllRelations,2,"Solution1"];                    (8.11)
```

The output includes the following:

THE ORDER IS NOW THE FOLLOWING:
$L < L^T < wanted < wanted^T \ll X_1 < X_1^{T-1} < X_1^T < X_1^{-1} < X_2 < X_2^T < Q < Q^T < A < A^T < R < R^T$

The expressions with unknown variables {}
and knowns $\{L,\, wanted,\, L^T\}$

$$wanted\, L \to -1\, L^T\, wanted \tag{8.12}$$

From rule (8.12) above we see that *wanted* satisfies a full rank null Sylvester equation (2.2) if L enjoys the property that $\lambda_i + \lambda_j^* \neq 0$ for $\lambda \in \sigma(L)$. Then for X the solution derived in Theorem 13.1 $X = X_2 X_1^{-1} = X_1^{-*}(X_1^* X_2)X_1^{-1}$ so X is hermition as well. □

The following theorem, Theorem 13.4 in [28], gives necessary and sufficient conditions for a *solution to be real.*

Theorem 9 *Let V be an n-dimensional H-invariant subspace, and let $X_1, X_2 \in \mathbb{C}^{n\times n}$ be such that X_1 is nonsingular and the columns of $\begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$ form a basis of V. Then $X = X_2 X_1^{-1}$ is real if and only if V is conjugate symmetric, i.e. $v \in V$ implies that $\overline{v} \in V$.*

Proof:
$(\Rightarrow)$
Defining $X \triangleq X_2 X_1^{-1}$, X must be real; and, since $Im \begin{bmatrix} I \\ X \end{bmatrix}$ is our H-invariant subspace $\mathcal{V}$, we have $\mathcal{V}$ conjugate symmetric.

$(\Leftarrow)$
Assuming our space is conjugate symmetric we have the following relation satisfied.

$$\begin{bmatrix} \overline{X_1} \\ \overline{X_2} \end{bmatrix} = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} P \tag{8.13}$$

where the $^-$ denotes complex conjugate.

In the Mathematica code below we use the symbols $CnjX1$ and $CnjX2$ to denote $\bar{X}_1$ and $\bar{X}_2$ respectively. We also introduce $\bar{X} = \bar{X}_2 \bar{X}_1{}^{-1}$.

The input:

```
AllRelations = {{CnjX1},{CnjX2}} - MatMult[{{X1},{X2}}, {{P}} ];
AllRelations = Union[ AllRelations, NCMakeRelations[Inv,X1,CnjX1,P],X
== X2**Inv[X1], { CnjX == CnjX2**Inv[CnjX1] }];

AllRelations = NCAddTranspose[AllRelations];

NCAutomaticOrder[ {{CnjX},{X},{P,X1,Q,A,tp[A],R,
Inv[CnjX1],CnjX1,CnjX2,X2,Inv[X1]}}, AllRelations ];

NCProcess[AllRelations,3,"Solution"];                          (8.14)
```

This corresponds to the following output:

THE ORDER IS NOW THE FOLLOWING:

$CnjX < CnjX^T \ll X < X^T \ll P < P^{T-1} < P^T < P^{-1} < X_1 < X_1^{T-1} < X_1^T < Q < A < A < R < CnjX_1^{-1} < CnjX_1 < CnjX_1^{T-1} < CnjX_1^T < CnjX_2 < CnjX_2^T < X_2 < X_2^T < X_1^{-1}$

THE FOLLOWING VARIABLES HAVE BEEN SOLVED FOR:

$\{X, X^T\}$

The corresponding rules are the following:

$X \rightarrow CnjX$

$X^T \rightarrow CnjX^T$

From the spreadsheet we see that $X = \overline{X}$ so our solution is real.

□

8.4 Summary of above results

Definition 8.4.1 *$H \in dom(Ric)$ if the following conditions are satisfied:*

1. *H has no purely imaginary eigenvalues.*
2. *$\mathcal{X}_-(H)$ (stable invariant subspace) and $Im \begin{bmatrix} 0 \\ I \end{bmatrix}$ are complementary.*

If $\mathcal{X}_-(H)$ is spanned by $\begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$, then $X = X_2X_1^{-1}$ is unique. We write $X = Ric(H)$.
(Under change of basis P, $\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} P = \begin{bmatrix} X_1P \\ X_2P \end{bmatrix}$ and $X = (X_2P)(X_1P)^{-1} = X_2PP^{-1}X_1^{-1} = X_2X_1^{-1}$.)

Some of the above results on Riccatis are summarized in Theorem 13.5 from [28] which we prove next.

Theorem 10 *Suppose $H \in dom(Ric)$ and $X = Ric(H)$. Then*

1. *X is real symmetric;*
2. *X satisfies the algebraic Riccati equation*

$$A^*X + XA + XRX + Q = 0;$$

3. *$A + RX$ is stable.*

Proof: Here we create an invariant stable subspace of the Hamiltonian. It is parameterized by $\begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$. We define H_{stable} with the equation

$$H \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} H_{stable}$$

where $\sigma(H_{stable}) \subset \mathbb{C}_-$.

(i)

H_{stable}'s eigenvalues are such that $\lambda_i + \lambda_j^* \neq 0$ for $\lambda \in \sigma(H_{stable})$. So by the argument with Theorem 13.3 we have $X_1^T X_2$ is hermition and therefore $X = X_1^{-T}(X_1^T X_2)X_1^{-1}$ is hermition.

From Theorem 13.4 we have that it is real after the following observation.

H_{stable} is conjugate symmetric:

Take $v \in H_{stable}$ (w.l.o.g. a generalized eigenvector and so again with no l.o.g. an eigenvector), then $H_{stable} v = \lambda v$ so $\overline{H}_{stable}\overline{v} = \overline{\lambda}\overline{v}$ or further $H_{stable}\overline{v} = \overline{\lambda}\overline{v}$, since H_{stable} has real entries. This conjugation, however, does not change λ's stability. $\overline{\lambda} \in \mathbb{C}_{-}$. So $\overline{v} \in H_{stable}$. ($H_{stable}$ is conjugate symmetric.)

(ii,iii)

Here we set $X = X_1^{-T}(X_1^T X_2)X_1^{-1}$ and show that X is a solution to the Riccati associated with H (8.2). We also show that $A + RX$ is stable.

Here is the Mathematica input.

```
AllRelations = MatMult[Hamiltonian, {{X1},{X2}}]
- MatMult[{{X1},{X2}},{{Hstable}}];

AllRelations = Union[ AllRelations, NCMakeRelations[ {Inv, X1}],
{ X == tp[Inv[X1]]**tp[X1]**X2**Inv[X1],
tp[A]**X + X**A + X**R**X + Q == wanted,
L == A + R ** X , tp[X2]**X1 - tp[X1]**X2, Q == tp[Q], R == tp[R] } ];

AllRelations = NCAddTranspose[AllRelations];

NCAutomaticOrder[ {{L,X,X1,X2,Q,A,tp[A],R,Inv[X1]}, {wanted,Hstable}},
AllRelations ];
NCProcess[AllRelations,2,"Solution"];                              (8.15)
```

Mathematica input (8.15) gives the following TEX output.

THE ORDER IS NOW THE FOLLOWING:

$L < L^T < X < X^T < X_1 < X_1^{T-1} < X_1^T < X_2 < X_2^T < Q < Q^T < A < A^T < R < R^T < X_1^{-1} \ll wanted < wanted^T < Hstable < Hstable^T$

THE FOLLOWING VARIABLES HAVE BEEN SOLVED FOR:

$\{Hstable, wanted, Hstable^T, Q^T, R^T, wanted^T, X^T\}$

The corresponding rules are the following:

$$Hstable \to X_1^{-1}\, L\, X_1 \tag{8.16}$$

$$wanted \to 0 \tag{8.17}$$

$$Hstable^T \to X_1^T\, L^T\, X_1^{T-1}$$

(ii)

Since $wanted = 0$ in the spreadsheet, (8.17), we have shown that this $X = X_2 X_1^{-1}$ is a solution to the Riccati.

(iii)

Since we have defined $L = A + RX$ from the spreadsheet we have $X_1^{-1} L X_1 = X_1^{-1}(A + RX)X_1 = Hstable$. This implies that $A + RX$ is stable.

□

8.5 Stabilizing solutions

Definition 8.5.1 *(A, R) is* **stabilizable** *if $\begin{bmatrix} A - \lambda I & R \end{bmatrix}$ has full row rank for all λ, $Re(\lambda) \geq 0$.*

The next theorem, Theorem 13.6 in [28], will give conditions for the existence of a stabilizable solution to the Algebraic Riccati Equation.

Theorem 11 *Suppose H has no imaginary eigenvalues and R is either positive semidefinite or negative semidefinite. Then $H \in dom(Ric)$ if and only if (A, R) is stabilizable.*

Proof:

$(\Leftarrow)$

Again we create an invariant stable subspace of the Hamiltonian, Hstable. This is $\begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$. This time we cannot assume that X_1^{-1} exists because this is the complementary condition. Since we have condition (i) of $dom(Ric)$ all we must show is condition (ii) which is equivalent to the existence of X_1^{-1}.

We will show that if X_1 has a non-trivial kernel it leads to a contradiction. Here we let $x \in Ker(X_1)$ so $X_1 x = 0$.

Here a slight deficiency of NCGB shows it's head. The software cannot easily handle commutative elements. A simple work around for this problem when scalars must be introduced is to use a prime number in place of this commutative indeterminate. In our NCGB run below we use 7 in place of λ, a stable eigenvalue.

```
Hamiltonian = {{A,R},{-Q,-tp[A]}};

AllRelations = MatMult[Hamiltonian,{{X1},{X2}}]
- MatMult[{{X1},{X2}},{{Hstable}} ];

AllRelations = Join[ AllRelations, { X1**x == 0,
Hstable ** x == 7 x, (* Re(7)< 0 *)
Q == tp[Q],R == tp[R],
tp[X2]**X1 == tp[X1]**X2
} ];

NCAutomaticOrder[ {{x,X2,Q,A,tp[A],R,Hstable},{X1}},AllRelations];
```

```
NCProcess[AllRelations,3,"Solution"];
(8.18)
```

The above Mathematica input resulted in the following spreadsheet.

THE ORDER IS NOW THE FOLLOWING:
$x < x^T < X_2 < X_2^T < Q < Q^T < A < A^T < R < R^T < Hstable < Hstable^T \ll X_1 < X_1^T$

The expressions with unknown variables $\{\}$
and knowns $\{A,\ Hstable,\ R,\ x,\ X_2,\ A^T,\ Hstable^T,\ x^T,\ X_2^T\}$

$R\, X_2\, x \rightarrow 0$

$(7 + A^T)\, X_2\, x == 0$

Looking at the spreadsheet we have

$$x^T\, X_2^T\, (7 + A) = 0$$

$$x^T\, X_2^T\, R = 0,$$

then

$$x^T X_2^T \begin{bmatrix} A - (-7) & R \end{bmatrix} = 0$$

But -7 is a positive eigenvalue ($7 \in \mathbb{C}_{-}$) and (A,R) is stabilizable. This implies $X_2 x = 0$ so $X_1 x = 0$ and $X_2 x = 0$ where $x \neq 0$. $\begin{pmatrix} X_1 \\ X_2 \end{pmatrix}$ has full column rank, since it is a parameterization of a subspace. We have found our contradiction. There cannot exist such an x. We must have X_1 nonsingular.

So $Im \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$ and the $Im \begin{bmatrix} 0 \\ I \end{bmatrix}$ are complementary.

$$H \in dom(Ric)$$

(⇒)

Follows from the definition.

□

The next theorem, Theorem 13.9 from [28], relates the structure of a certain matrix pencil with the existence of unobservable modes.

Theorem 12 *Suppose D has full column rank and let $R = D^*D > 0$; then the following are equivalent*

1. $\begin{bmatrix} A - \jmath\omega I & B \\ C & D \end{bmatrix}$ *has full column rank for all $\omega \in \mathbb{R}$.*
2. $((I - DR^{-1}D^*)C, A - BR^{-1}D^*C)$ *has no unobservable modes on the $\jmath\omega$-axis.*

Proof:

(i) (⇒) **(ii)**

Here again we have to use the lucky number 7 to represent $j\omega$. This causes a slight difficulty. We have been using the `tp[ ]` construction to denote complex conjugate and so far this has not caused any difficulties. Here, however, our commutative "indeterminate" should have the property that

```
tp[7] = - 7,
```

since

$$(j\omega)^* = -j\omega \text{ for } \omega \in \mathbb{R} \,.$$

The major ramification for our computer algebra techniques is that one cannot use the command `NCAddTranspose[ ]` blindly. Below we will handle our 7's carefully.

The main idea of the proof is as follows. We assume that we have a x which is an eigenvector corresponding to $j\omega$ (7) of $A - BR^{-1}D^TC$ and in the null space of $(I - DR^{-1}D^T)C$ (i.e. an unobservable mode) and show that this leads to a contradiction. We begin by assuming the invertibility of X_1 and X_2 to facilitate the algebra.

The NCGB input :

```
jWmatrix = {{A-7, B},{C,D}};

AllNonTpedRelations = Union[ NCMakeRelations[{Inv, R,A,X1,X2 }],
{ MatMult[ jWmatrix, {{X1},{X2 }} ],
R == tp[D] ** D,
(A - B**Inv[R]**tp[D]**C)**x == 7 x,
(1 - D**Inv[R]**tp[D])**C**x == 0 };                          (8.19)
```

We separate these relations, `AllNonTpedRelations`, into those which contain 7's, `RelsWith7`, and those which do not, `Non7Relations`.

```
RelsWith7 = {
(A-B**Inv[R]**tp[D]**C)**x==7 x,(-7+A)**X1+B**X2 };

(* Notice the sign changes *)
RelsWith7 = Union[ RelsWith7,
{ tp[(A-B**Inv[R]**tp[D]**C)**x]== -tp[ 7 x ],
tp[ 7 X1]+A**X1+B**X2 } ];
AllRelations = Union[ NCAddTranspose[ Non7Relations ],
RelsWith7 ];

NCAutomaticOrder[{{Q,D,x,R,A,B,C},{X1,X2}},AllRelations];
NCProcess[AllRelations,3,"Solution",RR->False];               (8.20)
```

The Mathematica input (8.20) results in the following output.

The expressions with unknown variables $\{X_2,\, X_1^{-1}\}$

and knowns $\{C,\, D^T,\, R^{T-1}\}$

$$X_2\, X_1^{-1} \rightarrow -1\, R^{T-1}\, D^T\, C \tag{8.21}$$

This suggests we pick $X_1 = 1$ and $X_2 = -R^{-1}\, D^T\, C$. Recall that neither X_1 nor X_2 is necessarily invertible. Since the expression on the right hand side of (8.21) is not (necessarily) invertible, but the inverted X_i is on the right this leads us to make the choice $X_1 = I$.

We make the substitution and see if $chk1$ and $chk2$ go to 0:

```
X1 = 1;
X2 = - tp[Inv[R]]**tp[D]**C;

AllRelations =
{ MatMult[ mtx1 ,{{X1},{X2}}]
- {{chk1},{chk2}},
NCMakeRelations[{Inv,R,A}],
R == tp[D] ** D,
(A - B**Inv[R]**tp[D]**C)**x == 7 x,
(1 - D**Inv[R]**tp[D])**C**x };

NCAutomaticOrder[{{chk1,chk2,D,R,A,B,C},{x}},AllRelations];
NCProcess[AllRelations,3,"Solution"];                        (8.22)
```

Mathematica input (8.22) gives the following output.

The expressions with unknown variables $\{x\}$

and knowns $\{chk_1,\, chk_2\}$

$chk_1\, x \rightarrow 0$

$chk_2\, x \rightarrow 0$

So it appears that $\begin{bmatrix} chk_1 \\ chk_2 \end{bmatrix} x = 0$ or equivalently

$$\begin{bmatrix} A - 7 & B \\ C & D \end{bmatrix} \begin{bmatrix} I \\ -R^{-1}D^TC \end{bmatrix} * x = 0$$

So the unobservable mode of $((I - DR^{-1}D^T)C, A - BR^{-1}D^TC)$ has given us a nonzero vector in the null space of the matrix in **(i)**. This is a contradiction. This matrix has full column rank for all ω.

(ii) $(\Rightarrow)$ **(i)**

Not complete.

$\square$

Chapter 9

Positive Real Matrix Functions

Here we prove some theorems about positive real matrix functions using NCGB. First let's recall the definition of positive real.

Definition 9.0.2 *A rational matrix function $G(s)$ is said to be* **positive real** *if it has the following property*

$$G(j\omega) + G^{\sim}(j\omega) > 0$$

for all $\omega \in \mathbb{R}$ and $G(s)$ is analytic in the right half plane.

9.1 Example of a pre-strategy

We prove the "algebraic part" of the following theorem with a pre-strategy, Theorem 13.25 [28]:

Theorem 13 *Let $\left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$ be a state space realization of $G(s)$ with A stable (not necessarily a minimal realization). Suppose there exist an $X \geq 0, Q,$ and W*

such that

$$\begin{aligned} XA + A^*X &= -Q^*Q && (9.1) \\ B^*X + W^*Q &= C && (9.2) \\ D + D^* &= W^*W. && (9.3) \end{aligned}$$

Then $G(s)$ is positive real and

$$G(s) + G^{\sim}(s) = M^{\sim}(s)M(s)$$

with $M(s) = \left[\begin{array}{c|c} A & B \\ \hline Q & W \end{array}\right]$. *Theorem 13.25 [28] also states that if $M(j\omega)$ has full column rank for all $\omega \in \mathbb{R}$, then $G(s)$ is strictly positive real, a fact we do not address here.*

Proof:

Here we create an indeterminant, $Wanted = G^{\sim} + G - M^{\sim}M$ and show that the relations which make up this theorem imply $Wanted = 0$. This will then show that $G^{\sim} + G = M^{\sim}M$. Recall the work around for a single commutative variable, in this case s, introduced in Section 2.4.1 which uses 7.

Here is the input:

```
newVariables = {X,Q,W};
origVariables = {A,B,C,D};

Invert = { NCMakeRelations[{Inv, A,X, 7-A, 7+A }],
tp[X] == X };

eqnObs = { X**A + tp[A]**X + tp[Q]**Q == 0 };
eqn2 = { tp[B]**X + tp[W]**Q - C == 0 };
eqn3 = { D + tp[D] == tp[W]**W };
NewEquation = {
Wanted == C**Inv[7-A]**B + D -
```

```
tp[B]**tp[Inv[7+A]]**tp[C] + tp[D]
- (( -tp[B]**tp[Inv[7+A]]**tp[Q] + tp[W]) **
( Q**Inv[7-A]**B + W ) ) };

AllRelations = Flatten[{Invert,eqnObs,eqn2,eqn3,NewEquation}];
AllRelations = NCAddTranspose[AllRelations];
NCAutomaticOrder[ {Flatten[ {origVariables,
newVariables}],{Wanted}}, AllRelations ];

NCProcess[AllRelations,3,"Other",RR->False,SBByCat->False];    (9.4)
```

This gives the following result:

THE ORDER IS NOW THE FOLLOWING:

$A < (7-A)^{-1} < A^{-1} < (7+A)^{-1} < A^T < (7-A)^{T-1} < A^{T-1} < (7+A)^{T-1} < B < B^T < C < C^T < D < D^T < X < X^{-1} < X^T < X^{T-1} < Q < Q^T < W < W^T \ll Wanted$

THE FOLLOWING VARIABLES HAVE BEEN SOLVED FOR:

$\{Wanted, X^T, X^{T-1}\}$

The corresponding rules are the following:

$Wanted \rightarrow 0$

Now since $Wanted = 0$ we have

$$G(s) + G^{\sim}(s) - (M^{\sim}(s)M(s)) = 0$$

or

$$G(s) + G^{\sim}(s) = M^{\sim}(s)M(s).$$

This is the desired algebraic result and we have shown the equivalence of the transfer functions under consideration. Note that we have not proved A is stable. Correspondingly we did not use the assumption $X \geq 0$. □

9.2 Example of a 2-Strategy (Converse of Section 9.1)

We discover the algebraic portion of the following theorem with a 2-Strategy+, Theorem 13.26 in [28], under an added minimality assumption on one of the systems. It is clear from the outset that a 1-Strategy will not work, since one of the equations (9.5) contains two variables which are not in the statement of the problem. It is gratifying that a 1-Strategy+ is used through all of the derivations until the last step. This bodes well for computer "automation" of the process.

The 2-Strategy+ discovery of (9.5) and (9.6) does not imply that alternatives to these equations do not exist, since the Strategy+ rules allow us to be rather free in taking various motivated unknowns to be 0. However, the computer runs produce output equations which a human observer would readily see have only 0 as a solution. All of this will be seen in the derivation below.

Theorem 14 *Suppose(A, B, C, D) is a minimal realization of $G(s)$ with A stable and $G(s)$ is positive real. Then there exist an $X \geq 0, Q$, and W such that*

$$XA + A^*X = -Q^*Q \tag{9.5}$$

$$B^*X + W^*Q = C \tag{9.6}$$

$$D + D^* = W^*W. \tag{9.7}$$

and

$$G(s) + G^{\sim}(s) = M^{\sim}(s)M(s)$$

with $M(s) = \left[\begin{array}{c|c} A & B \\ \hline Q & W \end{array}\right]$. *Furthermore, if* $G(s)$ *is strictly positive real, then* $M(j\omega)$ *given above has full column rank for all* $\omega \in \mathbb{R}$.

Discovery:

We start with the minimal system $G(s) = \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$ and a system $M = \left[\begin{array}{c|c} A_1 & B_1 \\ \hline C_1 & D_1 \end{array}\right]$ and our goal is to find simple equations which are equivalent to $G(s) + G^{\sim}(s) = M^{\sim}(s)M(s)$. Hopefully, we will obtain (9.5)-(9.7) or equations equally as elegant.

9.2.1 The setup

In state space coordinates we have:

$$G + G^{\sim} = \left[\begin{array}{cc|c} A & 0 & B \\ 0 & -A^T & -C^T \\ \hline C & B^T & D + D^T \end{array}\right] \tag{9.8}$$

$$M^{\sim}M = \left[\begin{array}{cc|c} A_1 & 0 & B_1 \\ -C_1^T C_1 & -A_1^T & -C_1^T D_1 \\ \hline D_1^T C_1 & B_1^T & D_1^T D_1 \end{array}\right] \tag{9.9}$$

We shall make the assumption (not actually required by the theorem) that both systems are minimal. Thus the statespace isomorphism theorem, see Chapter 12, implies there exists an invertible map T from the statespace of (9.8) to the statespace of (9.9).

Let's denote the state space matrices of system (9.8) (resp. (9.9)) by $\mathcal{A}_g$, $\mathcal{B}_g$, $\mathcal{C}_g$, and $\mathcal{D}_g$ (resp. $\mathcal{A}_m$, $\mathcal{B}_m$, $\mathcal{C}_m$, $\mathcal{D}_m$). In the spirit of minimizing the number of indeterminates we avoid explicitly introducing the inverse of T by writing the similarity transformation between the two above composed systems as

$$T\mathcal{A}_g = \mathcal{A}_m T,\ \mathcal{C}_g T = \mathcal{C}_m,\ \mathcal{B}_g = T\mathcal{B}_m,\ \mathcal{D}_g = \mathcal{D}_m$$

A common property, although obviously not always true, of block transformation matrices is that the diagonal blocks are invertible. We will make such an assumption about the diagonal blocks of the block 2x2 matrix T.

By the throughput matrix equality we have

$$D + D^T = D_1^T D_1.$$

We re-label D_1 as W so our derivations will have the same notation as the statement of the theorem.

9.2.2 Discovering via a strategy

Essential to the discovery process is the monomial order inherited from the classification of symbolic indeterminates into knowns and unknowns. Since in the statement of the theorem we are given system G and seek to find system M which satisfies certain criteria, it seems natural to regard A, B, and C as known and A_1, B_1, and C_1 as unknown. Likewise the observability Grammian of the unknown system M, introduced below as X_3, will be considered unknown.

The blocks of the transformation

$$T = \begin{pmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{pmatrix}$$

to the unknown system will likewise be considered unknown.

The computer input

Here is the Mathematica input we begin with:

```
origVariables = {A,B,C,D};
grams = {X1,X2,X3};
Ts = {T11,T12,T21,T22};
newVariables = {A1,B1,C1,W};

SetNonCommutative[newVariables];
SetNonCommutative[grams];
SetNonCommutative[Ts];
SetNonCommutative[origVariables];

Invert = NCMakeRelations[{Inv,T11,T22,A,A1,X1,X2,X3}];

eqnCtrl = { X1 ** tp[A] + A ** X1 + B ** tp[B] == 0 };
eqnObs = { tp[A] ** X2 + X2 ** A + tp[C] ** C == 0 ,
tp[A1] ** X3 + X3 ** A1 + tp[C1] ** C1 == 0 };

SelfAdjoints = { X1 == tp[X1], X2 == tp[X2], X3 == tp[X3] };

(* Tmtx is the similarity transformation between G  + G and M  M. *)
Tmtx = { { T11, T12 } , { T21, T22 } };

Similarities = {
(* The A transformation *)
MatMult[Tmtx,{{A,0},{0,-tp[A]}}]
- MatMult[{{A1,0},{-tp[C1]**C1,-tp[A1]}},Tmtx],

(* The B transformation *)
MatMult[Tmtx,{{B},{-tp[C]}}]
- {{B1},{-tp[C]**W}},
```

```
(* The C transformation *)
{{C,tp[B]}}
- MatMult[{{tp[W]**C1,tp[B1]}},Tmtx],

(* The D transformation *)
D + tp[D] == tp[W]**W
};

Rels = Flatten [{ Invert,eqnCtrl,eqnObs,Similarities,
SelfAdjoints } ];

Rels = NCAddTranspose[ Rels ];

NCAutomaticOrder[ {Flatten[{origVariables, X1,X2}],
{X3}, Ts , newVariables}, Rels ] ;

NCProcess[Rels,2,"Run1"];                                  (9.10)
```

The computer output

What follows is the output of NCProcess. Here is the order as reported by NCGB:

THE ORDER IS NOW THE FOLLOWING:

$A < A^{-1} < A^T < A^{T-1} < B < B^T < C < C^T < D < D^T < X_1 < {X_1}^{-1} < {X_1}^T < {X_1}^{T-1} < X_2 < {X_2}^{-1} < {X_2}^T < {X_2}^{T-1} \ll T_{11} < {T_{11}}^{-1} < {T_{11}}^T < {T_{11}}^{T-1} \ll X_3 < {X_3}^{-1} < {X_3}^T < {X_3}^{T-1} \ll A_1 < {A_1}^T \ll T_{12} < {T_{12}}^T \ll T_{21} < {T_{21}}^T \ll T_{22} < {T_{22}}^{-1} < {T_{22}}^T < {T_{22}}^{T-1} \ll B_1 < {B_1}^T < C_1 < {C_1}^T < W < W^T \ll {A_1}^{-1} < {A_1}^{T-1}$

THE FOLLOWING VARIABLES HAVE BEEN SOLVED FOR:

$\{A_1,\ B_1,\ {A_1}^T,\ {B_1}^T,\ {X_1}^T,\ {X_2}^T,\ {X_3}^T,\ {X_1}^{T-1},\ {X_2}^{T-1},\ {X_3}^{T-1}\}$

The corresponding rules are the following:

$$A_1 \to T_{11}\, A\, {T_{11}}^{-1} \tag{9.11}$$

$$B_1 \to T_{11}\, B - T_{12}\, C^T \tag{9.12}$$

Important categories follow:

The expressions with unknown variables $\{{T_{12}}^T,\ {T_{11}}^{T-1},\ {T_{11}}^T\}$
and knowns $\{A,\ A^T\}$

$${T_{12}}^T\, {T_{11}}^{T-1}\, A^T\, {T_{11}}^T \to -1\, A\, {T_{12}}^T \tag{9.13}$$

The expressions with unknown variables $\{{T_{21}}^T,\ X_3,\ {T_{11}}^T,\ T_{11}\}$
and knowns $\{A,\ A^T\}$

$$({T_{21}}^T\, T_{11} - {T_{11}}^T\, X_3\, T_{11})\, A + A^T\, ({T_{21}}^T\, T_{11} - {T_{11}}^T\, X_3\, T_{11}) == 0 \tag{9.14}$$

The expressions with unknown variables $\{T_{22},\ T_{21},\ {T_{12}}^T,\ T_{12},\ {T_{11}}^T,\ {T_{11}}^{-1}\}$
and knowns $\{C,\ B^T\}$

$$-1\, C\, ({T_{11}}^{-1}\, T_{12} - {T_{12}}^T\, T_{22} + {T_{12}}^T\, T_{21}\, {T_{11}}^{-1}\, T_{12}) + B^T\, (1 - {T_{11}}^T\, T_{22} + {T_{11}}^T\, T_{21}\, {T_{11}}^{-1}\, T_{12}) == 0 \tag{9.15}$$

Analysis

Equations (9.11) and (9.12) are singletons and define A_1 and B_1 respectively. The equation for B_1 will become simpler by the time our computations are done, but the important thing is that this unknown has been solved for.

Multiplying (9.13) on the right by T_{11}^{-T} and setting

$$q_4 = T_{12}^T T_{11}^{-T}$$

we have

$$q_4 A^T + A q_4 = 0, \tag{9.16}$$

an equation in one (motivated) unknown. Equation (9.16) implies that $q_4 = 0$, since A is assumed to be stable.

Using that A is stable does not operate purely at the level of computer algebra. This step requires human intervention. On the other hand, equation (9.16) has $q_4 = 0$ as a solution for any A, thus the definition of a Strategy+ allows us to take $q_4 = 0$. This operates purely at the level of computer algebra algorithms and could be implemented in generic software.

Then, since T_{11}^{-T} is non-singular, we have that

$$T_{12} = 0.$$

Setting

$$q_5 = T_{21}^T T_{11} - T_{11}^T X_3 T_{11} \tag{9.17}$$

equation (9.14) becomes

$$q_5 A^T + A q_5 = 0, \tag{9.18}$$

which in turn implies that $q_5 = 0$. Factoring the T_{11} from equation (9.17) we have

$$(T_{21}^T - T_{11}^T X_3) T_{11} = 0 \tag{9.19}$$

and so

$$T_{21}^T = T_{11}^T X_3.$$

Examining (9.15) after setting $T_{12} = 0$ we are left with

$$B^T (1 - T_{11}^T T_{22}) = 0.$$

This becomes an equation in one variable by setting

$$q_1 = 1 - T_{11}^T T_{22}.$$

Clearly, $B^T q_1 = 0$ has $q_1 = 0$ as a solution for all choices of B. Thus the rules of a 1-strategy+ allow us to set $q_1 = 0$. This is amusing from an engineering viewpoint, since B is usually a very low rank matrix, so while q_1 may be 0 we have not proved that it is. Another computer run which does not assume $q_1 = 0$ produces the result $q_1 = 0$ in a way which is consistent with engineering practice.

The computer input

Next we put our newly derived relations into the Mathematica input and call `NCProcess` again.

```
T12 = 0;
DiscoveredRels = {
T21 - X3 ** T11,
q1 == 1 - T11 **tp[T22] }

Rels = Flatten [{ Invert,eqnCtrl,eqnObs,
Similarities,SelfAdjoints, DiscoveredRels } ];

NCProcess[Rels,2,"Run2", SBByCat->False, RR->False];           (9.20)
```

The computer output

The output contained the following category:

The expressions with unknown variables $\{T_{11}, q_1\}$
and knowns $\{A, B, X_1, A^{-1}, A^{T-1}\}$

$$q_1\, T_{11}\, B \rightarrow 0 \tag{9.21}$$

$$q_1\, T_{11}\, A\, B \rightarrow 0$$

$$q_1\, T_{11}\, A\, A\, B \rightarrow 0$$

$$q_1\, T_{11}\, A\, A\, A\, B \rightarrow 0 \tag{9.22}$$

This strongly suggests that

$$q_1 T_{11} A^n B = 0 \text{ for all integers } n \geq 0. \tag{9.23}$$

The hypothesis of Theorem 13 of minimality and therefore controllability of the original system implies the controllability operator is right invertible. (See Section 12.1 for details.) Equation (9.23) implies that

$$q_1 = 0.$$

Then we have

$$1 - T_{11}^T T_{22}$$

or if you prefer

$$T_{11}^T = T_{22}^{-1}.$$

The computer input

We make the corresponding changes and run `NCProcess` again.

```
DiscoveredRels = {
T21 - X3 ** T11,
q1 == 1 - T11 **tp[T22] ,
tp[T11] == Inv[T22] };

NCAutomaticOrder[ {Flatten[{origVariables, X2, X1, W }], {q1}, {T11},
```

```
{C1},{X3},{T22},{T21},{A1,tp[A1],B1}, {Inv[A1], tp[Inv[A1]] }},
Rels ] ;

NCProcess[Rels,2,"Run3", SBByCat->False, RR->False ];          (9.24)
```

The computer output

The output contained the following category:

The expressions with unknown variables $\{X_3,\, {C_1}^T,\, C_1,\, {T_{11}}^T,\, T_{11}\}$
and knowns $\{A,\, A^{-1},\, A^T\}$

$$ {T_{11}}^T\, X_3\, T_{11}\, A \rightarrow -1\, A^T\, {T_{11}}^T\, X_3\, T_{11} - {T_{11}}^T\, {C_1}^T\, C_1\, T_{11} \tag{9.25} $$

At this stage we may make our procedure a 2 strategy by defining two motivated unknowns,

$$ Q = C_1 T_{11} \text{ and } X = T_{11}^T X_3 T_{11}. $$

This will make equation (9.25) an equation in two unknowns:

$$ Q^T\, Q \rightarrow -1\, X\, A - A^T\, X. $$

The computer input

We make the following Mathematica input.

```
DiscoveredRels = {
T21 - X3 ** T11,
q1 == 1 - T11 **tp[T22],
tp[T11] == Inv[T22],
Q == C1 ** T11,
```

```
X == tp[T11]**X3 ** T11 };

NCProcess[Rels,2,"Run4", SBByCat->False, RR->False ];        (9.26)
```

The computer output

The output contains the following categories:

The expressions with unknown variables $\{X, Q\}$
and knowns $\{C, B^T, W^T\}$

$W^T\, Q \rightarrow C - B^T\, X$

The expressions with unknown variables $\{X, Q^T, Q\}$
and knowns $\{A, A^T\}$

$Q^T\, Q \rightarrow -1\, X\, A - A^T\, X$

These are the desired relations and therefore we have discovered the algebraic part of Theorem 13. Our algebraic computations may be said to fit in the 2-Strategy+ formalism.

9.3 Inner transfer functions

Lemma 13.29 from [28] gives some algebraic results on systems which are used to analyze inner transfer functions. This result is derived next.

Theorem 15 *Suppose* $N = \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \in \mathbb{RH}_\infty$ *and* $X = X^* > 0$ *satisfies*

$$A^*X + XA + C^*C = 0 \tag{9.27}$$

Then

1. $D^*C + B^*X = 0$ *implies* $N^*N = D^*D$.
2. (A, B) *controllable, and* $N^*N = D^*D$ *implies* $D^*C + B^*X = 0$.

Proof:

(a)

The NCGB input:

```
newVariables = {X1,X2,Wanted};
SetNonCommutative[newVariables];
origVariables = { Inv[A], A , B, C, D };
SetNonCommutative[ origVariables ];

tp[X1] = X1;
tp[X2] = X2;

Invert = {
NCMakeRelations[{Inv,A, (7-A),(7+A) }] };

eqnObs = { tp[A] ** X1 + X1 ** A + tp[C] ** C == 0 };
eqnCtrl = { X2 ** tp[A] + A ** X2 + B ** tp[B] == 0 };
```

```
eqn1 = { tp[D] ** C + tp[B] ** X1 };

(* Wanted == N~ N *)
NewEquation = { Wanted == - tp[D] ** D +
(- tp[B] ** tp[Inv[7+A]] ** tp [ C] + tp[D])**
( C** Inv[7-A] ** B + D ) };

AllRelations = Flatten [{ Invert, eqnObs, eqnCtrl, eqn1 , NewEquation
} ] ;

AllRelations = NCAddTranspose[AllRelations];
NCAutomaticOrder[{{X1,A,B,C,D},{Wanted}}, AllRelations ];
NCProcess[AllRelations,3,"PosReal",RR->False,SBByCat->False]; (9.28)
```

This computer input resulted in the following LaTeX output:

THE ORDER IS NOW THE FOLLOWING:
$X_1 < A < (7+A)^{T-1} < A^{T-1} < (7-A)^{T-1} < A^T < (7+A)^{-1} < A^{-1} < (7-A)^{-1} < B < B^T < C < C^T < D < D^T \ll Wanted < Wanted^T \ll X_2$

THE FOLLOWING VARIABLES HAVE BEEN SOLVED FOR:
$\{Wanted, Wanted^T\}$

The corresponding rules are the following:

$Wanted \rightarrow 0$

$Wanted^T \rightarrow 0$

(b)

Not complete. We hope that the controllability and observability constructions developed in Chapter 12 may be used to prove this result but have been unsuccessful so far. □

Chapter 10

Hankel Norm Approximations

The theorem presented in this section can be used to solve the optimal Hankel norm approximation problem. We will use noncommutative computer algebra to generate formulas for matrix blocks which dilate an arbitrary system to an all pass system.

10.1 Dilating a transfer function to all pass

Definition 10.1.1 *A matrix function $G(s)$ is said to be* **all pass** *if it has the following property*

$$G(s)G^{\sim}(s) = I$$

Given system $G(s) = \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$, the condition stated above is equivalent via Theorem 8.4[1] in [28] to the existence of a P and a Q symmetric (where the throughput matrix D must be unitary) which satisfy the following:

$$AP + PA^T + B^T B = 0$$

$$A^T Q + QA + CC^T = 0$$

[1]which incidentally was proven using the NCGB techniques.

$$PQ = I$$

Here we take an arbitrary system and dilate it to an all pass system. More specifically we prove Theorem 8.5 in [28]. We go to great lengths to proceed through the proof in a motivated fashion. It turns out that the constructions are highly non-unique which forces us to set many variables to 0. This is what keeps us from discovering the theorem with a true 1-strategy.

Theorem 6 *Let* $G(s) = \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$ *with* $A \in \mathbb{C}^{n\times m}, C \in \mathbb{C}^{m\times n}\mathbb{C}^{m\times m}$ *satisfy*

$$\begin{aligned} AP + PA^* + BB^* &= 0 \\ A^*Q + QA + C^*C &= 0 \end{aligned}$$

for

$$\begin{aligned} P = P^* &= \mathrm{diag}(\Sigma_1, \sigma I_r) \\ Q = Q^* &= \mathrm{diag}(\Sigma_2, \sigma I_r) \end{aligned}$$

with Σ_1 *and* Σ_2 *diagonal,* $\sigma \neq 0$ *and* $\delta(\Sigma_1\Sigma_2 - \sigma^2 I) = 0$.

Patition (A, B, C) *conformably with* P*, as*

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, B = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}, C = [C_1 \quad C_2]$$

and define $W(s) := \left[\begin{array}{c|c} \tilde{A} & \tilde{B} \\ \hline \tilde{C} & \tilde{D} \end{array}\right]$ *with*

$$\begin{aligned} \tilde{A} &= \Gamma^{-1}(\sigma^2 A_{11}^* + \Sigma_2 A_{11}\Sigma_1 - \sigma C_1^* U B_1^*) \\ \tilde{B} &= \Gamma^{-1}(\Sigma_2 B_1 + \sigma C_1^* U) \\ \tilde{C} &= C_1\Sigma_1 + \sigma U B_1^* \\ \tilde{D} &= D - \sigma U \end{aligned}$$

where U is a unitary matrix satisfying

$$B_2 = -C_2^* U$$

and

$$\Gamma = \Sigma_1 \Sigma_2 - \sigma^2 I.$$

Also define the error system

$$E(s) = G(s) - W(s) = \left[\begin{array}{c|c} A_e & B_e \\ \hline C_e & D_e \end{array}\right]$$

with

$$A_e = \begin{bmatrix} A & 0 \\ 0 & \tilde{A} \end{bmatrix}, B_e = \begin{bmatrix} B \\ \tilde{B} \end{bmatrix}, C_e = [C \quad -\tilde{C}], D_e = D - \tilde{D}.$$

Then

(1) (A_e, B_e, C_e) *satisfy*

$$A_e P_e + P_e A_e^* + B_e B_e^* = 0 \qquad (10.1)$$

$$A_e^* Q_e + Q_e A_e + C_e^* C_e = 0 \qquad (10.2)$$

with

$$P_e = \begin{bmatrix} \Sigma_1 & 0 & I \\ 0 & \sigma I_r & 0 \\ I & 0 & \Sigma_2 \Gamma^{-1} \end{bmatrix} \qquad (10.3)$$

$$Q_e = \begin{bmatrix} \Sigma_2 & 0 & -\Gamma \\ 0 & \sigma I_r & 0 \\ -\Gamma & 0 & \Sigma_1 \Gamma \end{bmatrix} \qquad (10.4)$$

$$P_e Q_e = \sigma^2 I \qquad (10.5)$$

(2) $E(s) E^{\sim}(s) = \sigma^2 I$.

Here we attempt to "discover" this theorem. That is we show how a naive investigator might discover the dilation (if it exists) which satisfies the criteria of Theorem 6. The setup of this problem is straightforward. We take the realization of our system to have

$$\text{diagonal Grammians}, \Sigma_1 \text{ and } \Sigma_2.$$

We also take

$$\Gamma = I - \Sigma_1 \Sigma_2$$

as in the theorem above. Then Σ_1, Σ_2, and Γ are all diagonal. Therefore they and their inverses commute with each other. This structure is exploited with the NCGB command `SetPairWiseCommutative[]`. The dilation of the system has three entries, $(\tilde{A}, \tilde{B}, \tilde{C})$, but there are many unknowns, since the Grammians of the dilated system are somewhat undetermined. We call these 3x3 grammians P_e and Q_e and require

$$P_e Q_e = I$$

to satisfy the all pass theorem, Theorem 6 above. We go about trying to see if there exists such a dilation. The diagonal entries of P_e and Q_e are set invertible, since this is a property of most block dilations.

We attempt to label our symbolic indeterminates in an informative fashion:

$$\texttt{Gamma} = \Gamma, \texttt{Sig1} = \Sigma_1, \texttt{Sig2} = \Sigma_2$$

$$\texttt{Atil} = \tilde{A}, \texttt{Btil} = \tilde{B}, \text{ and} \texttt{Ctil} = \tilde{C}$$

Here is the original input to NCProcess.

```
SysVars = {A11,A12,A21,A22,B1,B2,C1,C2};

OtherVars = {Sig1,Sig2,U,Gamma};
Pvars = {P11, P12, P13, P21, P22, P23,P31, P32, P33};
Qvars = {Q11, Q12, Q13,Q21, Q22, Q23,Q31, Q32, Q33};
```

```
SetNonCommutative[SysVars];
SetNonCommutative[OtherVars];
SetNonCommutative[Qvars];
SetNonCommutative[Pvars];

unknowns = { Atil, Btil, Ctil , Dtil };

SetNonCommutative[unknowns];

(* Notational convenience *)
P = tp[P] = {{Sig1,0}, {0, 1}};
Q = tp[Q] = {{Sig2,0}, {0, 1}};

A = {{A11, A12}, {A21, A22}};
B = {{B1}, {B2}};
Cn = {{ C1, C2 }};

Ae = {{A11, A12, 0 }, {A21, A22, 0 }, {0, 0, Atil} };
Be = {{B1},{B2},{Btil}};
Ce = {{C1, C2, -Ctil}};
De = D - Dtil;

Pe = {{P11, P12, P13}, {P21, P22, P23}, {P31, P32, P33}};
Qe = {{Q11, Q12, Q13}, {Q21, Q22, Q23}, {Q31, Q32, Q33}};

Inverts = NCMakeRelations[{Inv,Sig1,Sig2,Gamma,P11,P22,
P33,Q11,Q22,Q33}];

GivenRels = {
```

```
U ** tp[U] == 1, tp[U] ** U == 1,
B2 == - tp[C2] ** U ,
- tp[U]**C2 == tp[B2],
Gamma == Sig1** Sig2 - 1 ,
tp[Sig1] == Sig1, tp[Sig2] == Sig2,
MatMult[A,P] + MatMult[P,tpMat[A]] + MatMult[B,tpMat[B]],
MatMult[tpMat[A],Q] + MatMult[Q,A]+ MatMult[tpMat[Cn],Cn], tp[Gamma]
== Gamma,
SetPairWiseCommutative[Sig1,Sig2,Gamma,Inv[Gamma]] } ;

ConditionsToSatisfy = {
MatMult[Ae,Pe] + MatMult[Pe,tpMat[Ae]] + MatMult[Be,tpMat[Be]],
MatMult[tpMat[Ae],Qe] + MatMult[Qe,Ae] + MatMult[tpMat[Ce],Ce],
MatMult[Pe,Qe] - IdentityMatrix[3] };

AllRelations = Flatten[{ Inverts, GivenRels, ConditionsToSatisfy }];

AllRelations = NCAddTranspose[AllRelations];
NCAutomaticOrder[{{SysVars,OtherVars},{unknowns}},AllRelations ];

NCProcess[AllRelations,1,"DiscoverReally",DegreeCap->10,
DegreeSumCap->17, SBByCat->False];                                (10.6)
```

Notice at the beginning and the end of our discovery process we are working with approximately the same number of equations. Modulo some small realizations (i.e. $VariableX == VariableY + VariableZ$) it is only the number and type of unknown variables which changes. It is not hard to believe at this point that our polynomial equations are terribly underdetermined. We have a huge amount of freedom here. In light of this fact we take whatever suggestions NC Collect on

Unknown Variables gives us.

Here are some of these suggestions:

The expressions with unknown variables $\{Q_{21}, Q_{12}, Q_{11}, P_{33}^{-1}, P_{32}, P_{31}, Ctil^T, Btil, Atil^T, Atil\}$

and knowns $\{A_{11}, A_{12}, A_{21}, A_{22}, C_1, C_2, Sig_2, A_{11}^T, A_{12}^T, A_{21}^T, A_{22}^T, B_1^T, B_2^T, C_1^T\}$

$(P_{33}^{-1} P_{32} + P_{33}^{-1} P_{31} Q_{12}) A_{21} + Ctil^T C_1 + P_{33}^{-1} Atil P_{31} Q_{11} - P_{33}^{-1} Atil P_{31} Sig_2 + P_{33}^{-1} Btil B_1^T (Q_{11} - Sig_2) + P_{33}^{-1} P_{31} Q_{11} A_{11} + P_{33}^{-1} P_{31} A_{11}^T (Q_{11} - Sig_2) - P_{33}^{-1} P_{32} A_{22}^T Q_{21} + Atil^T P_{33}^{-1} P_{31} Q_{11} + Atil^T P_{33}^{-1} P_{32} Q_{21} == 0$

$(P_{33}^{-1} P_{32} + P_{33}^{-1} P_{31} Q_{12}) A_{22} + Ctil^T C_2 + P_{33}^{-1} Atil P_{31} Q_{12} + P_{33}^{-1} Btil B_1^T Q_{12} + P_{33}^{-1} P_{31} Q_{11} A_{12} + P_{33}^{-1} P_{31} A_{11}^T Q_{12} + P_{33}^{-1} P_{32} A_{22}^T (1 - Q_{22}) + Atil^T P_{33}^{-1} P_{31} Q_{12} + Atil^T P_{33}^{-1} P_{32} Q_{22} == 0$

$-1\, A_{21} (1 + Gamma - Sig_1 Q_{11}) + A_{22} Q_{21} + B_2 (Btil^T P_{33}^{-1} P_{31} Q_{11} + Btil^T P_{33}^{-1} P_{32} Q_{21}) - P_{22} A_{21} + (P_{21} C_1^T - P_{23} Ctil^T) C_1 + A_{12}^T (Q_{11} - Sig_2) + A_{22}^T Q_{21} + A_{21} P_{11} Sig_2 + A_{22} P_{21} Sig_2 + P_{21} A_{11}^T Sig_2 == 0$

$A_{21} (P_{12} + Sig_1 Q_{12}) - A_{22} (1 - P_{22} - Q_{22}) + B_2 (Btil^T P_{33}^{-1} P_{31} Q_{12} + Btil^T P_{33}^{-1} P_{32} Q_{22}) + P_{21} A_{21}^T - P_{22} A_{22} + (P_{21} C_1^T - P_{23} Ctil^T) C_2 + A_{12}^T Q_{12} - A_{22}^T (1 - Q_{22}) == 0$

$-1 (A_{11}^T Q_{11}^T + A_{21}^T Q_{12}^T + Q_{11}^T A_{11}) Sig_1 - (Q_{11}^T B_1 - Q_{11}^T P_{31}^T P_{33}^{T-1} Btil - Q_{21}^T P_{32}^T P_{33}^{T-1} Btil) B_1^T - C_1^T (C_1 (Sig_1 - P_{11}^T) - C_2 P_{12}^T + Ctil P_{13}^T) + Q_{21}^T A_{12}^T == 0$

$-1\, A_{21} P_{21}^T + A_{22} (1 - P_{22}^T) - (Q_{12}^T B_1 - Q_{12}^T P_{31}^T P_{33}^{T-1} Btil - Q_{22}^T P_{32}^T P_{33}^{T-1} Btil) B_2^T - A_{12}^T (Sig_2 P_{21}^T + Q_{21}^T) + A_{22}^T (1 - P_{22}^T - Q_{22}^T) - Q_{12}^T A_{12} + Q_{22}^T A_{22}^T - C_2^T Ctil P_{23}^T == 0$

$A_{12} Q_{23} - B_1 (Btil^T P_{33}^{-1} - Btil^T P_{33}^{-1} P_{31} Q_{13} - Btil^T P_{33}^{-1} P_{32} Q_{23}) + A_{11} Sig_1 Q_{13} - P_{12} C_2^T Ctil + P_{13} Ctil^T Ctil + P_{13} Ctil^T Ctil - (P_{11} - Sig_1) C_1^T Ctil - Sig_1 Q_{13} Atil == 0$

Since these are such long equations in many unknowns it seems likely that those collected on subexpressions are, in fact, 0. Making unabashed use of suggestions made by NCCV we have "solved" for the following variables:

```
P11 = Sig1;
P22 = 1;
Q22 = 1;
Q11 = Sig2;                                                      (10.7)
```

Computer input (10.6) and (10.7) results in the following output.

The expressions with unknown variables $\{P_{12}\}$
and knowns $\{A_{11},\ A_{22}^T\}$
$P_{12}\, A_{22}^T \to -1\, A_{11}\, P_{12}$

The expressions with unknown variables $\{P_{21}\}$
and knowns $\{A_{22},\ A_{11}^T\}$
$P_{21}\, A_{11}^T \to -1\, A_{22}\, P_{21}$

The expressions with unknown variables $\{Q_{12}\}$
and knowns $\{A_{22},\ A_{11}^T\}$
$Q_{12}\, A_{22} \to -1\, A_{11}^T\, Q_{12}$

The expressions with unknown variables $\{Q_{21}\}$
and knowns $\{A_{11},\ A_{22}^T\}$
$Q_{21}\, A_{11} \to -1\, A_{22}^T\, Q_{21}$

Since A is a completely arbitrary matrix there is no reason for the spectrum of square A_{11} to have anything to do with the spectrum of square A_{22}. By general intuiton we assume the four categories above are full rank Sylvesters and take:

```
P12 = 0;
P21 = 0;
Q12 = 0;
Q21 = 0;                                                    (10.8)
```

Mathematica inputs (10.6), (10.7), and (10.8) result in the following output.

The expressions with unknown variables $\{Q_{31}, P_{13}\}$
and knowns $\{Gamma\}$
$P_{13}\,Q_{31} \to -1\,Gamma$

The expressions with unknown variables $\{Q_{31}, P_{23}\}$
and knowns $\{\}$
$P_{23}\,Q_{31} \to 0$

The expressions with unknown variables $\{Q_{32}, P_{13}\}$
and knowns $\{\}$
$P_{13}\,Q_{32} \to 0$

From the first category above we have that P_{13} and Q_{31} are of full rank, since Γ was assumed to be of full rank. Therefore the other categories imply that $P_{23} = 0$ and $Q_{32} = 0$. Also there were enough clues to imply that the other variables were not 0. They were set invertible. We add these relations:

```
P23 = 0;
P32 = 0;
Q23 = 0;
```

```
Q32 = 0;

Inverts = NCMakeRelations[{Inv,Sig1,Sig2,Gamma,P33,
Atil,P13,P31,Q13,Q31,Q33}];

AllRelations = Flatten[{ Inverts, GivenRels, ConditionsToSatisfy }];

AllRelations = NCAddTranspose[AllRelations];
NCAutomaticOrder[{{SysVars,OtherVars},{unknowns}},AllRelations ];
NCProcess[AllRelations,1,"DiscoverReally2",DegreeCap->10,
DegreeSumCap->17, SBByCat->False];                              (10.9)
```

Here was the somewhat ambiguous result:

The expressions with unknown variables $\{Q_{13},\, P_{13}^T\}$
and knowns $\{Gamma\}$

$Q_{13}\, P_{13}^T \rightarrow -1\, Gamma$

The expressions with unknown variables $\{Q_{13},\, P_{13}^{T-1}\}$
and knowns $\{Gamma,\, Sig_1,\, Sig_2,\, Gamma^{T-1},\, Sig_1^{T-1},\, Sig_2^{T-1}\}$

$Gamma\, P_{13}^{T-1} \rightarrow -1\, Q_{13}$

$Sig_2\, P_{13}^{T-1} + Sig_1^{T-1}\,(Q_{13} - P_{13}^{T-1}) == 0$

$Gamma^{T-1}\, Q_{13} \rightarrow -1\, P_{13}^{T-1}$

$Sig_1\, P_{13}^{T-1} + Sig_2^{T-1}\,(Q_{13} - P_{13}^{T-1}) == 0$

$Sig_2^{T-1}\, Sig_1^{T-1}\,(Q_{13} - P_{13}^{T-1}) + P_{13}^{T-1} == 0$

The expressions with unknown variables $\{Q_{13}^T,\, P_{13}\}$
and knowns $\{Gamma\}$

$P_{13}\, Q_{13}^T \rightarrow -1\, Gamma$

The expressions with unknown variables $\{Q_{13}^T,\, P_{13}^{-1}\}$
and knowns $\{Gamma,\, Sig_1,\, Sig_2,\, Gamma^{T-1},\, Sig_1^{T-1},\, Sig_2^{T-1}\}$

$P_{13}^{-1}\, Gamma \rightarrow -1\, Q_{13}^T$

$Q_{13}^T\, Gamma^{T-1} \rightarrow -1\, P_{13}^{-1}$

$P_{13}^{-1}\, Sig_2 - (P_{13}^{-1} - Q_{13}^T)\, Sig_1^{T-1} == 0$

$P_{13}^{-1}\, Sig_1 - (P_{13}^{-1} - Q_{13}^T)\, Sig_2^{T-1} == 0$

$P_{13}^{-1}\, Sig_1\, Gamma \rightarrow -1\, Q_{13}^T\, Sig_1$

$P_{13}^{-1}\, Sig_2\, Gamma \rightarrow -1\, Q_{13}^T\, Sig_2$

$Q_{13}^T\, Sig_1\, Gamma^{T-1} \rightarrow -1\, P_{13}^{-1}\, Sig_1$

$P_{13}^{-1}\, Sig_1\, Sig_1 - (P_{13}^{-1} - Q_{13}^T)\, Sig_1\, Sig_2^{T-1} == 0$

$P_{13}^{-1}\, Sig_1\, Sig_1\, Gamma \rightarrow -1\, Q_{13}^T\, Sig_1\, Sig_1$

$P_{13}^{-1}\, Sig_2\, Sig_2\, Gamma \rightarrow -1\, Q_{13}^T\, Sig_2\, Sig_2$

$Q_{13}^T\, Sig_1\, Sig_1\, Gamma^{T-1} \rightarrow -1\, P_{13}^{-1}\, Sig_1\, Sig_1$

After staring at these relations for a while the following two solutions for P_{13} and Q_{13} are apparent.

1. $Q_{13} = \pm\Gamma$ and $P_{13} = \mp I$ **or**

2. $P_{13} = \pm\Gamma$ and $Q_{13} = \mp I$.

The straight route

Choosing $Q_{13} = -\Gamma$ and $P_{13} = I$ we are following the solution in [28].

We make the following changes:

```
P13 = 1;
P31 = 1;
Q13 = -Gamma;
Q31 = -Gamma;                                                        (10.10)
```

run NCProcess again, and generate the following output:

$P_{33} \rightarrow Sig_1^{T-1}\, Gamma^{T-1} + Sig_1^{T-1}$

$Q_{33} \rightarrow Sig_1\, Gamma$

This is a nice answer for Q_{33}, but not too satisfying for P_{33}. Let's change the order and run NCGB again.

```
NCAutomaticOrder[{ Flatten[{ SysVars, U,Sig2,Gamma ,unknowns}],
{Sig1,P33, Q33 } } , AllRelations ]

NCProcess[AllRelations,1,"DiscoverReally2",DegreeCap->10,
DegreeSumCap->17,SBByCat->False];                                    (10.11)
```

This computer input resulted in the following LaTeX output:

The corresponding rules are the following:

$P_{33} \rightarrow Sig_2\, Gamma^{T-1}$

$Q_{33} \rightarrow Sig_2^{T-1}\, Gamma + Sig_2^{T-1}\, Gamma\, Gamma$

$Sig_1 \rightarrow Sig_2^{T-1}\, Gamma + Sig_2^{T-1}$

We now have $P_{33} = \Sigma_2 \Gamma^{-1}$ and $Q_{33} = \Sigma_1 \Gamma$ and so this coincides with the solution in the text.

This is the same result which is derived by choosing $Q_{13} = \Gamma$ and $P_{13} = -I$. In fact using these identifications both the P_{33} and Q_{33} identities were found in one run of NCProcess.

Now the 3x3 matrices P_e and Q_e have been discovered. We make the P_{33} and Q_{33} identification and begin with a search for $\tilde{B}$. The order may look strange to the uninitiated. We would like to avoid the submatrices partitioned around the last Grammian value, the entries with a 2 in the subscript, as well as other new dilated elements. Since in previous runs we see that $Btil$ and $B2$ appear together we set them together in the order.

```
P33 = Sig2**Inv[Gamma];
Q33 = Sig1**Gamma;

NCAutomaticOrder[{Flatten[{{ OtherVars, A11, A22 , C1}, {B1 }} ],
{ Btil, B2} , { Atil, Ctil, A21, A12, C2 } } , AllRelations]

NCProcess[AllRelations,1,"DiscoverReally3",DegreeCap->10,
DegreeSumCap->17,SBByCat->False];                           (10.12)
```

The expressions with unknown variables $\{B_2^T, Btil\}$
and knowns $\{B_1, Sig_2, U, C_1^T, Gamma^{T-1}\}$
$Btil\, B_2^T \rightarrow Sig_2\, Gamma^{T-1}\, B_1\, B_2^T + Gamma^{T-1}\, C_1^T\, U\, B_2^T$

This suggests that

$$\tilde{B} = \Gamma^{-1}(\Sigma_2 B_1 + C_1^T U)$$

which coincides with the text.
Next we search for $\tilde{C}$:

```
NCAutomaticOrder[{Flatten[{{OtherVars,A11,A22,C1}, {B1}}],
{Ctil,C2},{Atil,Btil,A21,A12,B2}}, AllRelations ]

NCProcess[AllRelations,1,"DiscoverReally3",DegreeCap->10,
DegreeSumCap->17,SBByCat->False];                      (10.13)
```

The expressions with unknown variables $\{C_2^T,\, Ctil\}$
and knowns $\{C_1,\, Sig_1,\, U,\, B_1^T\}$
$C_2^T\, Ctil \rightarrow C_2^T\, C_1\, Sig_1 + C_2^T\, U\, B_1^T$

So we now believe that

$$\tilde{C} = C_1 \Sigma_1 + U B_1^T$$

We turn to $\tilde{A}$ with

```
NCAutomaticOrder[{{Sig1, Sig2}, Flatten[{ U, Gamma, C1, A11,
B1}],{Atil},{A21,A12,A22,B2,C2}}, AllRelations ];

NCProcess[AllRelations,1,"DiscoverReally3",DegreeCap->10,
DegreeSumCap->17,SBByCat->False];                      (10.14)
```

And a truly satisfying answer is finally derived:

THE FOLLOWING VARIABLES HAVE BEEN SOLVED FOR:
$\{Atil,\, Gamma^{-1},\, Sig_1^{-1},\, Sig_2^{-1},\, Atil^T,\, Gamma^T,\, Sig_1^T,\, Sig_2^T\}$

The corresponding rules are the following:
$Atil \rightarrow Gamma^{T-1}\, A_{11}^T \;+\; Sig_1^{T-1}\, A_{11}\, Sig_1 \;-\; Gamma^{T-1}\, C_1^T\, U\, B_1^T \;+\; Sig_1^{T-1}\, Gamma^{T-1}\, A_{11}\, Sig_1$

So we see that with a little manipulation :

$$\tilde{A} = \Gamma^{-1}(A_{11}^T + \Sigma_2 A_{11}\Sigma_1 - C_1^T U B_1^T)$$

Let's confirm that our dilation and associated Grammians do what they're supposed to do. Lot's of our choices have been very ad hoc. We have no reason to believe that B_2 has full row rank, for example, in our discovery of Btil. We will verify our result by creating "check indeterminates" and showing that they are, in fact, 0. In the following we will assume that `Pe, Qe, Ae, Be,` and `Ce` are defined as in computer input (10.6).

```
Atil=Inv[Gamma]**(tp[A11]+Sig2**A11**Sig1-tp[C1]**U**tp[B1]);
Btil = Inv[Gamma] ** ( Sig2 ** B1 + tp[C1] ** U );
Ctil = C1 ** Sig1 + U ** tp[B1];

CheckVars = { Cp11, Cp12, Cp13, Cp21, Cp22, Cp23, Cp31, Cp32, Cp33, Cq11,
Cq12, Cq13, Cq21, Cq22, Cq23, Cq31, Cq32, Cq33 };
SetNonCommutative[ CheckVars ];

(* Check matrices *)
Ckp = {{Cp11, Cp12, Cp13}, {Cp21, Cp22, Cp23}, {Cp31, Cp32, Cp33}};
Ckq = {{Cq11, Cq12, Cq13}, {Cq21, Cq22, Cq23}, {Cq31, Cq32, Cq33}};
Cke = {{Ce11, Ce12, Ce13}, {Ce21, Ce22, Ce23}, {Ce31, Ce32, Ce33}};

Search = {
MatMult[Ae,Pe] + MatMult[ Pe,tpMat[Ae]] + MatMult[Be,tpMat[Be]] + Ckp,
MatMult[tpMat[Ae],Qe] + MatMult[Qe,Ae] + MatMult[tpMat[Ce],Ce] + Ckq,
```

```
MatMult[Pe,Qe] - IdentityMatrix[3] + Cke };

NCAutomaticOrder[{Flatten[{SysVars,OtherVars}],
{CheckVars}}, AllRelations];

NCProcess[AllRelations,1,"DiscoverReally3",DegreeCap->10,
DegreeSumCap->17,SBByCat->False];                              (10.15)
```

The result of the above computer input is the following.

THE FOLLOWING VARIABLES HAVE BEEN SOLVED FOR:

$\{Ce_{11}, Ce_{12}, Ce_{13}, Ce_{21}, Ce_{22}, Ce_{23}, Ce_{31}, Ce_{32}, Ce_{33}, Cp_{11}, Cp_{12}, Cp_{13}, Cp_{21}, Cp_{22}, Cp_{23}, Cp_{31}, Cp_{32}, Cp_{33}, Cq_{11}, Cq_{12}, Cq_{13}, Cq_{21}, Cq_{22}, Cq_{23}, Cq_{31}, Cq_{32}, Cq_{33}, Gamma^{-1}, Sig_1^{-1}, Sig_2^{-1}, Cp_{11}^T, Cp_{12}^T, Cp_{13}^T, Cp_{21}^T, Cp_{22}^T, Cp_{23}^T, Cp_{31}^T, Cp_{32}^T, Cp_{33}^T, Cq_{11}^T, Cq_{12}^T, Cq_{13}^T, Cq_{21}^T, Cq_{22}^T, Cq_{23}^T, Cq_{31}^T, Cq_{32}^T, Cq_{33}^T, Gamma^T, Sig_1^T, Sig_2^T\}$

The corresponding rules are the following:

$Ce_{11} \to 0$

$Ce_{12} \to 0$

$Ce_{13} \to 0$

$Ce_{21} \to 0$

$Ce_{22} \to 0$

$Ce_{23} \to 0$

$Ce_{31} \to 0$

$Ce_{32} \to 0$

$Ce_{33} \to 0$

$Cp_{11} \to 0$

$Cp_{12} \to 0$

$Cp_{13} \to 0$

$Cp_{21} \to 0$

$Cp_{22} \to 0$

$Cp_{23} \to 0$

$Cp_{31} \to 0$

$Cp_{32} \to 0$

$Cp_{33} \to 0$

$Cq_{11} \to 0$

$Cq_{12} \to 0$

$Cq_{13} \to 0$

$Cq_{21} \to 0$

$Cq_{22} \to 0$

$Cq_{23} \to 0$

$Cq_{31} \to 0$

$Cq_{32} \to 0$

$Cq_{33} \to 0$

Since all of our desired relations hold we have verified the identities found and now the theorem has been proven as well as discovered.

A new direction

The *choice* $P_{13} = I$, $Q_{13} = -\Gamma$ was made in the previous section and in the book, but nothing has ruled out the alternative: $P_{13} = -\Gamma$, $Q_{13} = I$. We make this choice here. (The identifications $P_{13} = \Gamma$ and $Q_{13} = -I$ lead to the same results.)

These four additions were made to computer input (10.9) (instead of computer input (10.10) as done in the previous section) :

```
P13 = -Gamma;
P31 = -Gamma;
Q13 = 1;
Q31 = 1;                                                    (10.16)
```

At this point the 3x3 block matrices P_e and Q_e are completely determined except for P_{33} and Q_{33}. We can fix this with another run of NCGB.

THE FOLLOWING VARIABLES HAVE BEEN SOLVED FOR:
$\{P_{33},\, Q_{33},\, Gamma^{-1},\, P_{33}^{-1},\, Q_{33}^{-1},\, Sig_1^{-1},\, Sig_2^{-1},\, Gamma^T,\, P_{33}^T,\, Q_{33}^T,\, Sig_1^T,\, Sig_2^T,\, P_{33}^{T-1},\, Q_{33}^{T-1}\}$
The corresponding rules are the following:

$P_{33} \rightarrow Sig_2\, Gamma$

$Q_{33} \rightarrow Sig_1\, Gamma^{T-1}$

So we have found a new P_{33} and Q_{33} which satisfy the sought for conditions. These are somewhat different than those found in the text as well.

Now P_e and Q_e have been completed and we need to find the actual dilation which corresponds to these grammians.
We begin our search for Btil by adding the following relations and creating the following order:

```
P33 = Sig2** Gamma;
Q33 = Sig1**tp[Inv[Gamma]];

NCAutomaticOrder[{Flatten[{ A11, A22, B1, OtherVars }],
```

```
{ Btil,B2},{Ctil,Atil,C1,C2,A12,A21}}, Rels ]

NCProcess[AllRelations,1,"DiscoverReally3",DegreeCap->10,
DegreeSumCap->17,SBByCat->False];                         (10.17)
```

The indeterminates above the {`Btil,B2`} group are things we do not want to see when solving for `Btil`. The fact that {`Btil,B2`} are in fact a group was observed by categories in the above runs and discovering the inevitable coupling of these terms.

THE ORDER IS NOW THE FOLLOWING:
$A_{11} < A_{11}^T < A_{22} < A_{22}^T < B_1 < B_1^T < Sig_1 < Sig_1^{T-1} < Sig_1^T < Sig_1^{-1} < Sig_2 < Sig_2^{T-1} < Sig_2^T < Sig_2^{-1} < U < U^T < Gamma < Gamma^{T-1} < Gamma^T < Gamma^{-1} \ll Btil < Btil^T < B_2 < B_2^T \ll Ctil < Ctil^T < Atil < Atil^{T-1} < Atil^T < Atil^{-1} < C_1 < C_1^T < C_2 < C_2^T < A_{12} < A_{12}^T < A_{21} < A_{21}^T$

The expressions with unknown variables $\{C_1^T,\, B_2^T,\, Btil\}$
and knowns $\{B_1,\, Sig_2,\, U\}$
$C_1^T\, U\, B_2^T \rightarrow -1\, Btil\, B_2^T - Sig_2\, B_1\, B_2^T$

This suggests we have

$$\tilde{B} = -C_1^T U - \Sigma_2 B_1.$$

We next seek to discover $\tilde{C}$ with the following order.

```
NCAutomaticOrder[{Flatten[{ A11, B1, C1, OtherVars }],
{Ctil,C2},{Btil, Atil, A22, B2, A12, A21}}, AllRelations ];

NCProcess[AllRelations,1,"DiscoverReally3",DegreeCap->10,
```

```
DegreeSumCap->17,SBByCat->False];                               (10.18)
```

This computer input resulted in the following LaTeX output:

THE ORDER IS NOW THE FOLLOWING:

$A_{11} < A_{11}^T < B_1 < B_1^T < C_1 < C_1^T < Sig_1 < Sig_1^{T-1} < Sig_1^T < Sig_1^{-1} < Sig_2 < Sig_2^{T-1} < Sig_2^T < Sig_2^{-1} < U < U^T < Gamma < Gamma^{T-1} < Gamma^T < Gamma^{-1} \ll Ctil < Ctil^T < C_2 < C_2^T \ll Btil < Btil^T < Atil < Atil^{T-1} < Atil^T < Atil^{-1} < A_{22} < A_{22}^T < B_2 < B_2^T < A_{12} < A_{12}^T < A_{21} < A_{21}^T$

The expressions with unknown variables $\{C_2^T, Ctil\}$
and knowns $\{C_1, Sig_1, U, B_1^T, Gamma^{T-1}\}$
$C_2^T\, Ctil \to -1\, C_2^T\, C_1\, Sig_1\, Gamma^{T-1} - C_2^T\, U\, B_1^T\, Gamma^{T-1}$

This suggests

$$\tilde{C} = (C_1\Sigma_1 - UB_1^T)\Gamma^{-1}.$$

Our next task is to find $\tilde{A}$. As was the case previously we simply change the monomial order.

```
NCAutomaticOrder[{Flatten[{ A11,B1,C1,OtherVars }],
{Atil},{A22,C2,B2,A12,A21}}, AllRelations ]
NCProcess[AllRelations,1,"DiscoverReally3",DegreeCap->10,
DegreeSumCap->17,SBByCat->False];                               (10.19)
```

THE ORDER IS NOW THE FOLLOWING:

$A_{11} < A_{11}^T < Sig_1 < Sig_1^{T-1} < Sig_1^T < Sig_1^{-1} < Sig_2 < Sig_2^{T-1} < Sig_2^T < Sig_2^{-1} < U < U^T < Gamma < Gamma^{T-1} < Gamma^T < Gamma^{-1} < C_1 < C_1^T < B_1 <$

$B_1^T \ll Atil < Atil^{T-1} < Atil^T < Atil^{-1} \ll A_{22} < A_{22}^T < C_2 < C_2^T < B_2 < B_2^T < A_{12} < A_{12}^T < A_{21} < A_{21}^T$

THE FOLLOWING VARIABLES HAVE BEEN SOLVED FOR:

$\{Atil, Gamma^{-1}, Sig_1^{-1}, Sig_2^{-1}, Atil^T, Gamma^T, Sig_1^T, Sig_2^T\}$

The corresponding rules are the following:

$Atil \to A_{11}^T\, Gamma^{T-1} + Sig_2\, A_{11}\, Sig_1\, Gamma^{T-1} - C_1^T\, U\, B_1^T\, Gamma^{T-1}$

So we have found a suitable expression for $\tilde{A}$. Compare with the expression in the book (here on page 178). What we have found is a similarity transformation away from the relations given there. Our newly discovered dilation is completed with

$$\tilde{A} = (\Sigma_2 A_{11} \Sigma_1 - C_1^T U B_1^T + A_{11}^T)\Gamma^{-1}$$

Let's confirm that our dilation and associated grammians do what they're supposed to do. Lot's of our choices have been very ad hoc. We have no reason to believe that B_2 has full row rank, for example, in our discovery of $\tilde{B}$.

```
Btil = - Sig2**B1 - tp[C1]**U;
Ctil = - (C1**Sig1 + U**tp[B1])**tp[Inv[Gamma]];
Atil = -tp[C1]**C1**Sig1**tp[Inv[Gamma]]
- tp[C1]**U**tp[B1]**tp[Inv[Gamma]] - tp[A11];

(* Check matrices *)
Ckp = {{Cp11, Cp12, Cp13}, {Cp21, Cp22, Cp23 }, {Cp31, Cp32, Cp33}};
Ckq = {{Cq11, Cq12, Cq13}, {Cq21, Cq22, Cq23}, {Cq31, Cq32, Cq33}};
Cke = {{Ce11, Ce12, Ce13}, {Ce21, Ce22, Ce23}, {Ce31, Ce32, Ce33}};

VerifyRels = {
```

```
MatMult[Ae,Pe] + MatMult[ Pe,tpMat[Ae]] + MatMult[Be,tpMat[Be]] + Ckp,
MatMult[tpMat[Ae],Qe] + MatMult[Qe,Ae] + MatMult[tpMat[Ce],Ce] + Ckq,
MatMult[Pe,Qe] - IdentityMatrix[3] + Cke };

NCAutomaticOrder[{Flatten[{SysVars,OtherVars}],{CheckVars}}, Rels ];

NCProcess[AllRelations,1,"CheckAnswer", SBByCat->False];     (10.20)
```

This gives us the same output as listed on pages 191-192 confirming the validity of our result.

So we have found a new dilation which does not appear in the text and is equally satisfying. NCGB didn't know it was the wrong one.

Chapter 11

Model Uncertainty and Robustness

11.1 Coprime factor uncertainty

The next result we will prove is Theorem 9.6 from [28].

9.3.3 Coprime Factor Uncertainty

As another example, consider a left coprime factor perturbed plant described in Figure 9.6.

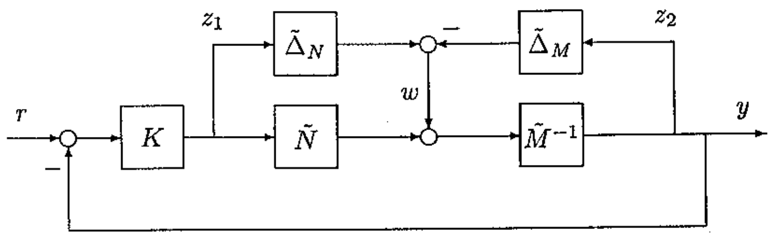


Figure 9.6: Left Coprime Factor Perturbed Systems

Theorem 9.6 *Let*

$$\Pi = (\tilde{M} + \tilde{\Delta}_M)^{-1}(\tilde{N} + \tilde{\Delta}_N)$$

with $\tilde{M}, \tilde{N}, \tilde{\Delta}_M$, $\tilde{\Delta}_N \in \mathcal{RH}_\infty$. The transfer matrices $(\tilde{M}, \tilde{N})$ are assumed to be a stable left coprime factorization of P (i.e., $P = \tilde{M}^{-1}\tilde{N}$), and K internally stabilizes the nominal system P. Define $\Delta := \begin{bmatrix} \tilde{\Delta}_N & \tilde{\Delta}_M \end{bmatrix}$. Then the closed-loop system is well-posed and internally stable for all $\|\Delta\|_\infty < 1$ if and only if

$$\left\| \begin{bmatrix} K \\ I \end{bmatrix} (I + PK)^{-1}\tilde{M}^{-1} \right\|_\infty \leq 1.$$

Proof:

$(\Rightarrow)$

The inequalities present in this theorem are a fancy way of stating algebraic facts. The results of this theorem will hold if

$$(I + \Delta \begin{bmatrix} K \\ I \end{bmatrix} (I + PK)^{-1}M^{-1})^{-1} \in \mathbb{RH}_\infty \tag{11.1}$$

By the small gain theorem this would imply that

$$\left\| \begin{bmatrix} K \\ I \end{bmatrix} (I + PK)^{-1}M^{-1})^{-1} \right\|_\infty \leq 1$$

for all $\|\Delta\|_\infty < 1$.

Along with the left coprime factorization of the plant, $P = M^{-1}N$, we take a right coprime factorization of the controller, $K = UV^{-1}$, to facilitate the algebraic approach. We assume that the closed loop system is internally stable and hence $((\tilde{N}+\tilde{\Delta}_N)U+(\tilde{M}+\tilde{\Delta}_M)V)^{-1} \in \mathbb{RH}_\infty$ by Lemma 5.10. Since we have assumed that K stabilizes the nominal system P we may take the sensitivity matrix $(I+PK)$ to be invertible.

The approach here is to assume the invertibility of $((\tilde{N}+\tilde{\Delta}_N)U+(\tilde{M}+\tilde{\Delta}_M)V)$ and show that (11.1) holds.

The Mathematica input:

```
MyVars = {
M,N,Dn,Dm,U,V,UnkInv,Xk,Yk,Xp,Yp };

SetNonCommutative[ MyVars ];

K = U**Inv[V];
P = Inv[M]**N;

DummyFactors = {Xk,Yk,Xp,Yp};
RealFactors = {M,N,U,V, Inv[M], Inv[V] };

Inverts = {
Inv[((N+Dn)**U)+ ((M+Dm)**V) ] ** ( (N+Dn)**U+(M+Dm)**V) - 1,
((N+Dn)**U+(M+Dm)**V)** Inv[((N+Dn)**U) +((M+Dm)**V) ] - 1,

Inv[V] ** V - 1,
V ** Inv[V] - 1,
Inv[M] ** M - 1,
M ** Inv[M] - 1 };

SysProps = {
```

```
(* K stabilizes P so the sensitivity matrix is invertible *)
Inv[1+P**K] ** (1+P**K) - 1,
(1+P**K) ** Inv[1 + P**K] - 1,
(* P has lcf *)
M**Xp + N**Yp - 1,
(* K has rcf *)
Xk**U + Yk**V - 1 };

Unknown = {
(1+MatMult[{{Dn,Dm}},{{K},{1}}, {{Inv[1+P**K]**Inv[M]}}][[1,1]])
**UnkInv - 1 };

AllRelations = Flatten[{Inverts,SysProps,Unknown}];

SetMonomialOrder[ RealFactors, { Inv[N**U+Dn**U+M**V+Dm**V]},
DummyFactors,{UnkInv},{Dn,Dm}, {Inv[1+P**K]} ];

NCProcess[AllRelations,7,"CoprimeUncert", SBByCat->False];    (11.2)
```

The output :

Input =

$-1 + ((Dm + M)\,V + (Dn + N)\,U)^{-1}\,((Dm + M)\,V + (Dn + N)\,U)$

$-1 + ((Dm + M)\,V + (Dn + N)\,U)\,((Dm + M)\,V + (Dn + N)\,U)^{-1}$

$-1 + V^{-1}\,V$

$-1 + V\,V^{-1}$

$-1 + M^{-1}\,M$

$-1 + M\,M^{-1}$

$-1 + (1 + M^{-1}\,N\,U\,V^{-1})^{-1}\,(1 + M^{-1}\,N\,U\,V^{-1})$

$-1 + (1 + M^{-1}\, N\, U\, V^{-1})\, (1 + M^{-1}\, N\, U\, V^{-1})^{-1}$

$-1 + M\, Xp + N\, Yp$

$-1 + Xk\, U + Yk\, V$

$-1 + (1 + (Dm + Dn\, U\, V^{-1})\, (1 + M^{-1}\, N\, U\, V^{-1})^{-1}\; M^{-1})\, UnkInv$

THE FOLLOWING VARIABLES HAVE BEEN SOLVED FOR:

$\{UnkInv\}$

The corresponding rules are the following:

$UnkInv \rightarrow M\, V\, (Dm\, V + Dn\, U + M\, V + N\, U)^{-1} + N\, U\, (Dm\, V + Dn\, U + M\, V + N\, U)^{-1}$

This says that the unknown inverse is a matrix polynomial of elements of $\mathbb{RH}_\infty$. Recall that M, V, N, and U are all in $\mathbb{RH}_\infty$. The result follows.

($\Leftarrow$) This time we assume that equation (11.1) holds and try to show the stability of the closed-loop system. By lemma 5.10 this is equivalent to

$$((\tilde{N} + \tilde{\Delta}_N)U + (\tilde{M} + \tilde{\Delta}_M)V)^{-1} \in \mathbb{RH}_\infty.$$

The only change we make is the order and the name of the (now) known inverse: The input:

```
SetMonomialOrder[ RealFactors, KnownInv, DummyFactors,
{Dn,Dm}, Inv[1 + P**K] , Inv[N**U+Dn**U + M**V+Dm**V ] ];

Known = {
(1+MatMult[{{Dn,Dm}},{{K},{1}}, {{Inv[1+P**K]**Inv[M]}}][[1,1]] )
** KnownInv - 1,
KnownInv **
( 1+ MatMult[ {{Dn,Dm}},{{K},{1}},
{{Inv[1+P**K]**Inv[M]}}][[1,1]] ) - 1
```

```
};                                                        (11.3)
```

The output:

THE FOLLOWING VARIABLES HAVE BEEN SOLVED FOR:

$\{(Dm\,V + Dn\,U + M\,V + N\,U)^{-1}\}$

The corresponding rules are the following:

$(Dm\,V + Dn\,U + M\,V + N\,U)^{-1} \to V^{-1}\,(1 + M^{-1}\,N\,U\,V^{-1})^{-1}\;M^{-1}\,KnownInv$

Then pulling the M^{-1} and V^{-1} inside the polynomial inverse we have the quantity in question equal to $(MV+NU)^{-1}*KnownInv$. Since the nominal system is stable we have $(Dm\,V + Dn\,U + M\,V + N\,U)^{-1} \in \mathbb{RH}_\infty$ which implies the stability of the perturbed closed loop system.

$\square$

Part V

New Directions

Chapter 12

Infinite Sequences and the State Space Isomorphism Theorem

In this chapter we prove some facts about time discrete linear systems. Objects essential to this analysis include the observability and controllability operators. These are expressed symbolically with infinite sequences and therefore require more care than standard noncommutative indeterminates.

We will define controllability and observability, introduce controllability and observability operators, and present some purely algebraic relations which capture some of their properties. The practicioner may add these relations in a computer algebra session working with time discrete time domain linear systems. In fact, we will demonstrate the utility of these relations by using these purely algebraic relations to prove some basic linear system theorems, the Youla-Tissi State Space Isomorphism Theorem and the discrete Grammian equation.

12.1 Controllability and observability

In this section we will define controllability and observability and recall some basic facts about systems which have these properties.

Informally, if a system is controllable the state of the system may be controlled with the input vector. The formal definition follows.

Definition 12.1.1 *A linear discrete-time dynamic system is said to be* controllable *if for any initial state $x_0 = x(0)$ and final state x_1 there exists an integer t_0 such that for any integer $t_1 > t_0$ there exists input $u(k)$ for $k \in \{0, 1, \ldots, t_1\}$ where the solution to $x(k+1) = Ax(k) + Bu(k)$ satisfies $x(t_1) = x_1$.*

The following lemma gives some equivalent characterizations of controllability and defines the *controllability operator*, $\mathcal{C}$.

Lemma 7 *Let $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$. Let A be asymptotically stable. It is well known that the the following are equivalent*

(1) The pair (A, B) is controllable.

(2) The controllability operator

$$\mathcal{C} = \begin{bmatrix} B & AB & A^2B & \cdots \end{bmatrix} \tag{12.1}$$

has full row rank.

*(3) There exists a positive definite matrix L_c (*the controllability Grammian*) which satisfies*

$$AL_cA^* + L_c + BB^* = 0 \tag{12.2}$$

Informally, if a system is observable the state of the system may be determined from knowledge of the output and input vector. The formal definition follows.

Definition 12.1.2 *A linear discrete-time dynamic system is said to be* observable *if there exists an integer t_0 such that for any $t_1 > t_0$ the initial state $x(0)$ can be determined from knowledge of $u(k)$ and $y(k)$ for all $k \in \{0, 1, \ldots, t_1\}$.*

The following lemma gives some equivalent characterizations of observability and defines the *observability operator*, $\mathcal{O}$.

Lemma 8 *Let $A \in \mathbb{R}^{n\times n}$ and $C \in \mathbb{R}^{m\times n}$. Let A be asymptotically stable. It is also well known (in fact a dual result of Lemma 7) that the the following are equivalent*

(1) The pair (C, A) is observable.

(2) The observability operator

$$\mathcal{O} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \end{bmatrix} \tag{12.3}$$

has full column rank.

*(3) There exists a positive definite matrix L_o (*the observability Grammian*) which satisfies*

$$A^* L_o A + L_o + C^* C = 0 \tag{12.4}$$

Definition 12.1.3 *A linear dynamic system which is both controllable and observable is said to be* minimal.

From the point of view of computer algebra a great problem with (12.1) and (12.3) is that they are infinite quantaties and so can not be handled directly. This brings us to the main issue of this chapter.

> Given a system (A, B, C, D) what information should we routinely feed into a computer algebra session involving it? More specifically how can we effectively deal with controllability and observability operators?

This chapter gives one possible answer to this specific question. We give a finite set (in fact 7) of relations which characterize $\mathcal{C}$ as the controllability and $\mathcal{O}$ as the observability operator of a system. Then we illustrate the use and the strength of our formalism by deriving the state space isomorphism theorem, a core fact in control theory.

12.1.1 The state space isomorphism theorem

The following theorem is known as the Youla-Tissi State Space Isomorphism Theorem.

Theorem 16 *Let* (A_1, B_1, C_1, D_1) *and* (A_2, B_2, C_2, D_2) *be two minimal linear discrete-time dynamic systems. If* $y_1(t_1) = y_2(t_1)$ *for* $u_1(t) = u_2(t)$ *for all* $u_i(t)$, $t \in \{0, \ldots, t_1\}$, *then there exists an invertible matrix* T *such that*

$$A_1 = T^{-1} A_2 T,$$

$$B_1 = T^{-1} B_2, \text{ and}$$

$$C_1 = C_2 T.$$

This theorem is a chestnut in linear system theory. See, for instance, any introductory text in the subject such as [3] or [1].

Notice that we have used the above theorem elsewhere in this thesis, for example Theorem 14. It is therefore a more "atomic" theorem than those which are proved through the use of it. Our investigations suggest that this property makes the theorem more difficult to prove. We see a similar phenomena in automatic geometric theorem proving, see [4] and Section 6.1. There Shing-Chang Chou had no hope of proving the Pythagorean theorem via computer, but was able to use this "atomic" theorem (among others) to prove 256 theorems in Euclidean geometry.

12.2 Algebraic controllability and observability

In this section we will develop purely algebraic characterizations of controllability and observability.

12.2.1 Motivation

Since our derivations will be purely algebraic, it seems that our results will be valid over any reasonable input and output space. However, since we wish to use the standard adjoint notation, we will restrict our attention to the infinite dimensional Hilbert space $\ell_2^+(\mathbb{C}^{n\times m})$, defined formally at the end of this section.

We will let $\mathcal{S}$ be the forward shift operator on $\ell_2^+(\mathbb{C}^{n\times m})$

$$\mathcal{S}([a_0, a_1, a_2, \ldots]) = [0, a_0, a_1, a_2, \ldots].$$

It is easy to see that the adjoint of $\mathcal{S}$ is the backward shift operator $\mathcal{S}^*$,

$$\mathcal{S}^*([a_0, a_1, a_2, \ldots]) = [a_1, a_2, a_3, \ldots].$$

We will then have the following *algebraic identities*

$$\mathcal{S}^*\mathcal{S} = I, \mathcal{S}\mathcal{S}^* = I - P, \text{ and } P^T = P \tag{12.5}$$

where I is the identity operator on $\ell_2^+(\mathbb{C}^{n\times m})$ and P is the projection operator

$$P([a_0, a_1, a_2, \ldots]) = [a_0, 0, 0, \ldots].$$

The well known relation $P^2 = P$ holds as well although it turns out to be a consequence of those given above. In practice a person might add this relation to computer input to speed up computation time.

Given a stable linear dynamic system $\left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$ and letting $\mathcal{C}$ and $\mathcal{O}$ be the members of $\ell_2^+(\mathbb{C}^{n\times m})$ defined by (12.1) and (12.3), respectively, we will have the following algebraic identities

$$A\mathcal{C} = \mathcal{C}\mathcal{S} \text{ and } \mathcal{O}A = \mathcal{S}^*\mathcal{O}. \tag{12.6}$$

If the system $\left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$ is controllable then $\mathcal{C}$ will be right invertible. This implies the existence of a $\mathcal{C}^{-R}$ such that

$$\mathcal{C}\mathcal{C}^{-R} = I. \tag{12.7}$$

One specific $\mathcal{C}^{-R}$ is the pseudo-inverse,

$$\mathcal{C}^T\left(\mathcal{C}\mathcal{C}^T\right)^{-1}.$$

If the system $\left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$ is observable then $\mathcal{O}$ will be left invertible. This implies the existence of a $\mathcal{O}^{-L}$ such that

$$\mathcal{O}^{-L}\mathcal{O} = I. \tag{12.8}$$

One specific $\mathcal{O}^{-L}$ is the pseudo-inverse,

$$\left(\mathcal{O}^T\mathcal{O}\right)^{-1}\mathcal{O}^T.$$

Notice that equations (12.6), (12.7), and (12.8) are purely algebraic identities which can be used as input to the Gröbner basis algorithm.

Next we summarize the algebraic relations developed above relating to controllability.

Formal algebraic definitions

Definition 12.2.1 *A discrete linear dynamic system* (A, B, C, D) *will have* **algebraic controllability operator** $\mathcal{C}$ *if the following relations hold*

$$\mathcal{S}^*\mathcal{S} = I, \tag{12.9}$$

$$\mathcal{S}\,\mathcal{S}^* = I - P, \tag{12.10}$$

$$P^T = P, \tag{12.11}$$

$$A\,\mathcal{C} = \mathcal{C}\,\mathcal{S}, \tag{12.12}$$

$$\mathcal{C}P = BP \tag{12.13}$$

Furthermore, we will say that the system is algebraically controllable *if the following relation holds*

$$\mathcal{C}\,\mathcal{C}^{-R} = I \tag{12.14}$$

Next we summarize the algebraic relations developed above relating to observability.

Definition 12.2.2 *A discrete linear dynamic system* (A,B,C,D) *will have* **algebraic observability operator** $\mathcal{O}$ *if the following relations hold*

$$\mathcal{S}^*\mathcal{S} = I, \tag{12.15}$$

$$\mathcal{S}\,\mathcal{S}^* = I - P, \tag{12.16}$$

$$P^T = P, \tag{12.17}$$

$$\mathcal{O}\,A = \mathcal{S}\,\mathcal{O}, \tag{12.18}$$

$$P\mathcal{O} = PC. \tag{12.19}$$

Furthermore, we will say that the system is algebraically observable *if the following relation holds*

$$\mathcal{O}^{-L}\,\mathcal{O} = I. \tag{12.20}$$

In practice a person would probably add the relation $P^2 = P$ to save computation time although this relation is a consequence of those given above.

In this subsection we have used the standard notion of $\ell_2^+(\mathbb{C}^{n\times m})$. We let $\ell_2^+(\mathbb{C}^{n\times m})$ be the set of one sided square summable sequences. An element of which is

$$a = [a_0, a_1, a_2, \ldots] \text{ and } a_i \in \mathbb{C}^n \text{ for all } i$$

such that

$$\sum_{i=0}^{\infty} \|a_i\|^2 < \infty$$

where $\|\cdot\|$ is the norm induced by the scalar product on $\ell_2^+(\mathbb{C}^{n\times m})$

$$\langle a, b\rangle = \sum_{i=0}^{\infty} \mathrm{Trace}(a_i^* b_i) \tag{12.21}$$

for $a, b \in \ell_2^+(\mathbb{C}^{n\times m})$. It is well known that $\ell_2^+(\mathbb{C}^{n\times m})$ is a Hilbert space.

12.2.2 A formal algebraic description of similar systems

Let system $\left[\begin{array}{c|c} A_1 & B_1 \\ \hline C_1 & D_1 \end{array}\right]$ have controllability operator $\mathcal{C}_1$ and observability operator $\mathcal{O}_1$. Let $\left[\begin{array}{c|c} A_2 & B_2 \\ \hline C_2 & D_2 \end{array}\right]$ have controllability operator $\mathcal{C}_2$ and observability operator $\mathcal{O}_2$.

If these two systems have the same input-output characteristics we will have

$$G(s) = C_1(sI - A_1)^{-1}B_1 + D_1 = C_2(sI - A_2)^{-1}B_2 + D_2$$

which implies

$$C_1 A_1^j B_1 = C_2 A_2^j B_2 \text{ for all } j \geq 0.$$

Then the following equations will be satisfied

$$\begin{aligned} \mathcal{O}_1\, \mathcal{C}_1 &= \mathcal{O}_2\, \mathcal{C}_2, && (12.22) \\ C_1\, \mathcal{C}_1 &= C_2\, \mathcal{C}_2, \\ \mathcal{O}_1\, B_1 &= \mathcal{O}_2\, B_2, \text{ and} \\ D_1 &= D_2. && (12.23) \end{aligned}$$

12.3 Derivation of the Youla-Tissi theorem

We have now introduced enough algebraic identities to prove the Youla-Tissi state space isomorphism theorem via computer algebra.

12.3.1 The computer input

In the following section we will assume that all variables have been set non-commutative. We will begin with the definition of controllability and observability operators as in (12.6) and (12.5).

```
SysProps = {
```

```
A1 ** Ctrl1 == Ctrl1 ** S,
A2 ** Ctrl2 == Ctrl2 ** S,
Obsv1 ** A1 == tp[S] ** Obsv1 ,
Obsv2 ** A2 == tp[S] ** Obsv2,
tp[S] ** S == 1, S ** tp[S] == 1 - P,
tp[P] == P };                                            (12.24)
```

Since the two given systems are minimal, equations (12.7-12.8) may be entered with the following computer input.

```
System1Controllable = {
Ctrl1 ** Rctl1 == 1 };                                   (12.25)

System2Controllable = {
Ctrl2 ** Rctl2 == 1 };                                   (12.26)

System1Observable = {
Lobsv1** Obsv1 == 1 };                                   (12.27)

System2Observable = {
Lobsv2** Obsv2 == 1 };                                   (12.28)
```

The equivalence of the two systems and equations (12.22-12.23) will give us

```
SystemsEquivalent = {
Obsv1 ** Ctrl1 - Obsv2 ** Ctrl2 == 0,
C1** Ctrl1 == C2 ** Ctrl2,
Obsv1 ** B1 == Obsv2 ** B2,
D1 == D2 } ;                                             (12.29)
```

We next combine all the relevant equations

```
AllRelations = Union[ eqnMinimal, eqnTimeDomain,
inverses, SysProps, assumption ] ;                       (12.30)
```

We next set the order.

```
NCAutomaticOrder[ {{C2,D2,B2,A2},
{Ctrl2,Rctl1,Rctl2,Ctrl1},
{Obsv1,Obsv2,Lobsv1,Lobsv2},{S,P},
{A1,B1,C1,D1}
}, AllRelations] ;                                    (12.31)
```

Finally the call to `NCProcess`.

```
NCProcess[AllRelations,3,"YoulaTissi",RR->False, SBByCat->False ]; (12.32)
```

12.3.2 The computer output

The result is a spreadsheet with the following category

THE FOLLOWING VARIABLES HAVE BEEN SOLVED FOR:

$\{A_1, B_1, C_1, D_1, A_1^{-1}\}$

The corresponding rules are the following:

$A_1 \rightarrow Ctrl_1\, Rctl_2\, A_2\, Ctrl_2\, Rctl_1$

$B_1 \rightarrow Ctrl_1\, Rctl_2\, B_2$

$C_1 \rightarrow C_2\, Ctrl_2\, Rctl_1$

$D_1 \rightarrow D_2$

$A_1^{-1} \rightarrow Ctrl_1\, Rctl_2\, A_2^{-1}\, Ctrl_2\, Rctl_1$

The expressions with unknown variables $\{Ctrl_1, Rctl_1, Ctrl_2\}$

and knowns $\{\}$

$$Ctrl_2\, Rctl_1\, Ctrl_1 \rightarrow Ctrl_2 \tag{12.33}$$

The expressions with unknown variables $\{Ctrl_1, Rctl_2, Ctrl_2\}$

and knowns $\{\}$

$$Ctrl_1\, Rctl_2\, Ctrl_2 \rightarrow Ctrl_1 \tag{12.34}$$

The Youla-Tissi state space isomorphism theorem has been proved. We have shown the existence of T

$$T = \mathcal{C}_1\, \mathcal{C}_2^{-R} \text{ and } T^{-1} = \mathcal{C}_2\, \mathcal{C}_1^{-R}$$

where T is the similarity transformation described in Theorem 12.1.1. Relations (12.34) and (12.33) show that $\mathcal{C}_1\mathcal{C}_2^{-R}$ is indeed the inverse of $\mathcal{C}_2\mathcal{C}_1^{-R}$. It is interesting to note that (12.34) and (12.33) do not appear in the computer output if the `NCProcess` option `RemoveRedundent` is set to True.

One can arrive at different characterizations of T by using different orders. For example, running the Gröbner basis algorithm on `Rels` defined in (12.30) under the order

$$\begin{gathered} C_2 < D_2 < B_2 < A_2 < \mathcal{O}_1 < \mathcal{O}_2 < \mathcal{O}_1^{-L} < \mathcal{O}_2^{-L} \ll \\ \mathcal{C}_2 < \mathcal{C}_2 < \mathcal{C}_1^{-R} < \mathcal{C}_2^{-R} < S < P < A_1 < B_1 < C_1 < D1 \end{gathered} \tag{12.35}$$

produces the following output

$$A_1 \rightarrow Lobsv_1\, Obsv_2\, A_2\, Lobsv_2\, Obsv_1,$$

$$B_1 \rightarrow Lobsv_1\, Obsv_2\, B_2,$$

and

$$C_1 \rightarrow C_2\, Lobsv_2\, Obsv_1.$$

We arrive at a different characterization of the same T,

$$T = \mathcal{O}_1^{-L}\, \mathcal{O}_2 \text{ and } T^{-1} = \mathcal{O}_2^{-L}\, \mathcal{O}_1.$$

12.4 Derivation of the discrete Grammian equations

We will next show how the methods developed in the previous section imply the controllability Grammian equation (12.2) for the controllable system (A_1, B_1, C_1, D_1).

12.4.1 The computer input and output

Assume that computer inputs (12.24) and (12.25) are in place. These equations are a consequence of the controllability of the system. Join these equations with

```
AllRelations = Union[ SysProps, System1Observable, { Ctrl1**P == B1**P
} ];                                                            (12.36)
```

and symmetrize the relations with

```
AllRelations = NCAddTranspose[ AllRelations ];                  (12.37)
```

We next set the order

```
NCAutomaticOrder[{{A1,B1},{Ctrl1},{P,S}}, AllRelations ] ;      (12.38)
```

and call `NCProcess`

```
NCProcess[AllRelations ,3,"FindGram", SBByCat->False ];         (12.39)
```

The result includes the following relation

$$B_1\, P\, B_1^T \rightarrow Ctrl_1\, Ctrl_1^T - A_1\, Ctrl_1\, Ctrl_1^T\, A_1^T \qquad (12.40)$$

---

which is the discrete Grammian equation (12.2) with $L_c = \mathcal{C}_1 \mathcal{C}_1^T$. Also included in the output is the well known property of projection operators

---

$$P\, P \rightarrow P.$$

An analogous procedure can be used to discover the observability Grammian equation (12.4) with a change of order.

12.5 Chapter conclusions

We have given purely algebraic relations for defining controllability operators $\mathcal{C}$ and observability operators $\mathcal{O}$. Of course, it is possible that we have not presented a complete set of defining relations. That is other algebraic consequences which may be shown analytically to follow from controllability and observability may not follow algebraically from our relations. However, we have been able to use these relations to generate the following formulas automatically

1 The Youla-Tissi state space isomorphism theorem.

2 The Discrete Grammian equation.

This work represents the first stages of an attempt to give practical definitions of linear systems of use to computer algebra. Open questions include

1. What properties should a "complete set of defining relations" have?

2. Once a "complete set" has been defined what is one?

This final chapter has described what many would regard as the least developed research presented in this thesis. The problems handled in this section often represent the axioms which are used to prove the results found elsewhere in this thesis.

Proving the more atomic theorems seems to be a thankless pursuit, requiring a significant amount of effort to formulate algebraically, until the final tidbits of algebra one is left with could easily be done by hand. Still, a development of the subject with such an unyielding use of algebra is new and, as computers become more powerful, perhaps useful to those who seek to discover new theorems. It is a long walk to the elevator, but it's nice to know the way.[1]

The actual computer calculations done in this section were quite computationally *inexpensive*. All of the computer runs in this section completed in less than half a minute.

[1]See page 99.

Index

Bibliography

[1] P. J. Antsaklis and A. N. Michel. *Linear Systems*. McGraw-Hill, 1997.

[2] W. Barrett, C. Johnson, M. Lundquist, and H. Woerdeman. Completing a block diagonal matrix with a partially prescribed inverse. *Linear Algebra and It's Applications*, 223–224:73–87, 1995.

[3] C. T. Chen. *Linear System Theory and Design*. Holt, Rinehart, and Winston, New York, 1984.

[4] S. C. Chou. *Mechanical Geometry Theorem Proving*. D. Reidel Publishing Company, Dordrecht, Netherlands, 1988.

[5] D. Cox, J. Little, and D. O. Shea. *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Springer-Verlag, Undergraduate Texts in Mathematics, 1992.

[6] M. Fiedler. Matrix inequalities. *Numer. Math*, 9:109–119, 1966.

[7] E. Green. An introduction to noncommutative Gröbner bases. *Computational algebra*, 151:167–190, 1993.

[8] E. Green, L. Heath, and B. Keller. Opal: A system for computing noncommutative Gröbner bases. In H. Comon, editor, *Eighth International Conference on Rewriting Techniques and Applications (RTA-97)*, LNCS# 1232, pages 331–334. Springer-Verlag, 1997.

[9] J. Helton, R. Miller, and M. Stankus. *NCAlgebra: A Mathematica Package for doing noncommuting Algebra*. available from http://math.ucsd.edu/~ncalg, 1996. Beware that to perform the computations in this paper you need NCGB. (See [HS97]).

[10] J. Helton and M. Stankus. *NCGB: Noncommutative Groebner bases*. available from http://math.ucsd.edu/~ncalg, 1997.

[11] J. Helton and M. Stankus. Computer assistance for discovering formulas in system engineering and operator theory. *Journal of Functional Analysis*, 161:289–368, 1999.

[12] J. W. Helton, M. R. James, and W. M. McEneaney. Nonlinear control: The joys of having an extra sensor. *Proceedings of the 37th IEEE CDC,Tampa, Florida, USA, Dec 16-18*, 4:3609–13, 1998.

[13] J. W. Helton, F. D. Kronewitter, W. M. McEneaney, and M. Stankus. Singularly perturbed control systems using noncommutative computer algebra. *Submitted to International Journal of Robust and Nonlinear Control*, 1999.

[14] J. W. Helton, F. D. Kronewitter, and M. Stankus. Computer algebra in the control of singularly perturbed dynamical systems. *Proceedings of the 38th IEEE Conference in Decision and Control, Phoenix, Arizona, USA, Dec 7-10*, 1999.

[15] J. W. Helton, F. D. Kronewitter, and M. Stankus. Noncommutative elimination ideal structure in structured algebraic problem solving. *preprint*, 1999.

[16] B. Keller. *Algorithms and Orders for Finding Noncommutative Groebner Bases.* PhD thesis, Virginia Polytechnic Institute and State University, 1997.

[17] B. Keller and E. Green. *The OPAL System.* available from http://hal.cs.vt.edu/opal, 1998.

[18] P. V. Kokotovic, H. K. Khalil, and J. O'Reilly. *Singular Perturbation Methods in Control: Analysis and Design.* Academic Press, 1986.

[19] F. D. Kronewitter. Using noncommutative Gröbner bases in solving partially prescribed matrix inverse completion problems. *Submitted to Linear Algebra and It's Applications*, 1999.

[20] H. Kwakernaak and R. Sivan. *Linear optimal control systems.* Wiley Interscience., 1972.

[21] D. Mackenzie. The automation of proof: a historical and sociological exploration. *IEEE Annals of the History of Computing*, 17:7–29, 1995.

[22] F. Mora. Groebner bases for non-commutative polynomial rings. *Lecture Notes in Computer Science*, 229:353–362, 1986.

[23] Z. Pan and T. Basar. H_∞ optimal control for singularly perturbed systems 1. perfect state measurements. *Automatica*, 29:401–423, 1993.

[24] Z. Pan and T. Basar. H_∞ optimal control for singularly perturbed systems 2. imperfect state measurements. *IEEE Transactions on Automatic Control*, 39:280–299, 1994.

[25] W. Wen-tsun. On the decision problem and the mechanization of theorems in elementary geometries. *Scientia Sinica*, 21:159–172, 1978.

[26] W. Wen-tsun. Toward mechanization of geometry-some comments on Hilbert's "Grandlagen der Geometrie". *Acta Math. Scientia*, 2:125–138, 1982.

[27] S. Wolfram. *Mathematica: A System for Doing Mathematics by Computer.* Addison Wesley Publishing, 1988.

[28] K. Zhou, J. C. Doyle, and K. Glover. *Robust and Optimal Control.* Prentice-Hall, Inc., 1995.