J. Webster (ed.), *Wiley Encyclopedia of Electrical and Electronics Engineering*
Copyright © 1999 John Wiley & Sons, Inc.

SOFTWARE FOR CONTROL SYSTEM ANALYSIS AND DESIGN: SYMBOL MANIPULATION

The computer revolution has radically changed every area of engineering, and control/systems engineering is not an exception. Indeed, computers have become essential tools in modeling, analysis, design, and implementation of control systems. In particular, they enable the design engineer to tackle problems of ever increasing complexity (1). The topic of this article is the effectiveness of symbolic computation software so-called *computer algebra*, in the analysis and design of control systems.

Symbolic computation software should be distinguished from numerical computation software. The former performs *exact* computation and can manipulate whole mathematical (symbolic) expressions, whereas the latter is limited to *approximate* computation based on numerical expressions. MATLAB (The Mathworks, Inc.) and SCILAB (Copyright INRIA; freely available via http://www-rocq.inria.fr/scilab/ are popular numerical software packages in the control engineering community. Such software assists in a wide variety of control engineering tasks, from modeling to real-time control implementation. The simulation facilities offered in these packages make them very popular in educational programs. Similarly, there are a number of readily available symbolic computation software packages. Mathematica (Wolfram Research, Inc.), Maple (Wiferloo Maple Inc.), REDUCE (©Anthony C. Hearn), DERIVE (Software House Inc., Texas Instruments), Macaulay, COCOA, (freely available via http://cocoa.dima.unige.it and MACSYMA (Macsyma, Inc.) are some of the more popular ones. Most of these software packages incorporate numerical as well as symbolic computation so that the user can resort to numerical analysis of symbolic expressions if required. Observe also that MATLAB, through its Symbolic Math Toolbox, adds to its numerical core symbolic manipulation capability. In general the distinction between purely numerical and symbolic computation software begins to blur, the trend being to offer both capabilities in the same software environment.

In general, a particular control problem can be approached using numerical and/or symbolic software. More precisely, all computational problems can be classified as (2 p. 275):

(1) *Purely Numerical Problems* Problems that can be handled only using numerical software and for which there is no symbolic solution, such as the computation of roots of a univariate polynomial.
(2) *Numerically Reducible Problems* Problems for which both symbolic and numerical solutions exist, such as finding the greatest common divisor of a set of univariate polynomials.
(3) *Numerically Irreducible Problems* Problems that can be handled only using symbolic software, such as the computation of Gröbner based for multivariate polynomials.

From the above classification it is clear that a familiarity with symbolic and numerical software can benefit the control/systems engineer enormously. In this article, the power and usefulness of symbolic software are brought to the fore. It is of course understood that numerical software is an equally indispensable tool in the control engineer's toolbox, since, in general, the control designer needs to resort to both symbolic and numerical software to successfully carry out control design.

At first contact, symbolic software is reminiscent of university-level engineering analysis and calculus. It offers the flexibility, excitement, and expectation of computing exactly with a completely abstract representation of the problem at hand. Unfortunately, it shares the same drawback, in that the problems that can be dealt with are limited in their complexity. Nevertheless, using symbolic computation, software computations that involve cumbersome algebraic expressions that are far beyond error-prone manual manipulations become routine. As a result, symbolic computation software has significantly pushed the boundaries of what can be regarded as an analytically tractable mathematical problem, and so it has become an invaluable tool in the control designer's toolbox for modeling, design, and analysis of control systems.

A plethora of symbolic computation algorithms targeting control system analysis and design are available. Symbolic computation is useful in almost every step of control system modeling and the analysis and controller design of linear and nonlinear control systems. In particular, symbolic manipulation of bond graphs and multidimensional Laplace transforms, which are important tools in the modeling of nonlinear control systems, are facilitated by computer algebra. The analysis of qualitative and quantitative system properties under parameter variations is supported by symbolic computation through sensitivity and bifurcation analysis. Robust control problems in either linear or nonlinear systems are facilitated by computer algebra software. The important geometric approach to nonlinear control design (e.g., computing a linearizing feedback or determining zero dynamics) is enabled by commercially available symbolic software. A more elaborate overview of other symbolic software algorithms used in modeling, analysis, and design of control systems is provided in 2.

In this article, *polynomial control system* analysis and design are targeted, and in particular three important methods are discussed: *Gröbner bases, differential algebra* (in particular, Ritt's algorithm), and *quantifier elimination*. In essence, these methods deal with polynomials. Computation of Gröbner bases is implemented in most commercial symbolic software packages such as Maple and Mathematica. On the other hand, Ritt's algorithm is not as yet commercially available, but software implementations may be obtained from the specialists working in the field. Quantifier elimination has been implemented in Mathematica.

Polynomial systems are a natural generalization of linear systems; they possess a nice algebraic structure, and their analysis is amenable to computer algebra. Moreover, as polynomials are universal approximators, polynomial control models can be used as valid models for almost any physical system. Furthermore, the restriction to polynomial systems is not severe, since any nonlinear function that satisfies a polynomial differential or algebraic equation, such as trigonometric functions, can be handled by introducing artificial states; see 3 and 4 for more details.

The state-space model of a polynomial control system is given by

$$\begin{aligned} \sigma x(t) &= f(x(t), u(t)) \\ y(t) &= h(x(t), u(t)) \end{aligned} \tag{1}$$

Here $x \in \mathbb{R}^n$, $y \in \mathbb{R}^p$, $u \in \mathbb{R}^m$ represent respectively the state, the observed output, and the manipulated control input of the system. The operator σ is either the derivative $\sigma x(t) = dx(t)/dt$ for continuous-time systems (when $t \in \mathbb{R}$) or the forward difference $\sigma x(t) = x(t + 1)$ for discrete-time systems (when $t \in \mathbb{N}$). The symbols $\mathbb{R}$, $\mathbb{Q}$, $\mathbb{N}$, and $\mathbb{C}$ denote respectively the sets of real, rational, natural, and complex numbers. The vector functions f and h are assumed to have entries that are polynomial functions in all their variables. Moreover, for computational reasons the polynomials are assumed to have rational coefficients. Continuous-time (or discrete-time) linear systems

$$\begin{aligned} \sigma x(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned} \tag{2}$$

form a special subclass of polynomial control systems (1). A wide range of systems can be adequately modeled using Eq. (1). Indeed, systems as diverse as biochemical reactors, grain dryers, satellites, robots, and airplanes have been successfully modeled using polynomial systems (1) (see, e.g., 5).

In the context of polynomial control systems, problems like determining equilibria, estimating a domain of attraction of a stable equilibrium, and locating approximately periodic solutions, as well as testing controllability and observability properties, reduce naturally to the analysis of sets of algebraic polynomial equations. Such problems can be addressed using the Gröbner basis method. In any modeling exercise, the elimination of variables from sets of differential and algebraic equations plays a crucial role. This problem also arises naturally in the context of determining if two state representations are equivalent. Differential algebra and Ritt's algorithm are very useful in this context. A range of different control problems can be formulated as questions involving quantifiers "there exists" and "for all," and this naturally leads to the so-called quantifier elimination algorithms.

The Gröbner Basis Method

Central objects in the Gröbner basis theory are polynomial ideals and affine varieties (6). Let $p_1,\ldots, p_s$ be multivariate polynomials in the variables $x_1,\ldots, x_n$ whose coefficients are in the field k. For the collection of ordered n-tuples of elements of k the notation k^n is used. The variables $x_1,\ldots, x_n$ are considered as "place markers" in the polynomial. The notation $p_1,\ldots, p_s \in k[x_1,\ldots, x_n]$ is adopted. The affine variety (or *variety*) defined by the s polynomials $p_1,\ldots, p_s$ is the collection of all solutions in k^n of the system of equations

$$p_1(x_1,\ldots,x_n)=0,\ldots, \qquad p_s(x_1,\ldots,x_n)=0 \tag{3}$$

Formally, the variety is

$$\boldsymbol{V}(p_1,\ldots,p_s) := \{(a_1,\ldots,a_n)\in k^n : p_i(x_1,\ldots,x_n)=0, i=1,\ldots s\}. \tag{4}$$

For instance, a straight line, a parabola, an ellipse, a hyperbola, and a single point are all examples of varieties in $\mathbb{R}^2$. The polynomial *ideal* I that is generated by $p_1,\ldots, p_s$ is a set of polynomials obtained by combining these polynomials through multiplication and addition with other polynomials: formally,

$$I=\langle p_1,\ldots,p_s\rangle := \left\{ f=\sum_{i=1}^{s} g_i p_i \ : \ g_i \in k[x_1,\ldots,x_n]\right\}. \tag{5}$$

The polynomials p_i, $i = 1,\ldots, s$, form a *basis* for the ideal I. A very useful interpretation of a polynomial ideal I is in terms of the equations (3). On multiplying p_i by arbitrary polynomials $g_i \in k[x_1,\ldots, x_n]$ and adding the products, the implication of Eq. (3) is that $f = g_1p_1 + \cdots + g_s p_s = 0$, and of course $f \in I$. Hence, the ideal $I = \langle p_1,\ldots, p_s\rangle$ contains all the "polynomial consequences" of the equations (3).

A notion at the core of the Gröbner basis method is that of the *monomial ordering* (a monomial is a polynomial consisting of a single term), since it introduces an appropriate extension of the notion of the leading term and the leading coefficient familiar from univariate polynomials to multivariate polynomials. One can define many different monomial orderings [lexicographic, graded lexicographic, graded reverse lexicographic, etc. (6)], but to be specific, consider the so-called lexicographic, or *lex*, ordering. Let α,β be two n-tuples of integers [$\alpha = (\alpha_1,\ldots, \alpha_n)$, $\beta = (\beta, \ldots, \beta_n) \in\mathbb{N}^n$. The n-tuple α is said to *succeed* β (in lex ordering), denoted as $\alpha \succ \beta$, if in the vector difference $\alpha-\beta = (\alpha_1 - \beta_1,\ldots, \alpha_n - \beta_n)$ the leftmost nonzero entry is positive. Obviously, it is possible to define $n!$ different lex orderings for polynomials in n variables. For the polynomial $f = 2{x_1}^3x_2{x_3}^3 + 4{x_1}^3{x_3}^5$, using the lex ordering $x_1 \succ x_2 \succ x_3$ the monomial ${x_1}^3x_2{x_3}^3$ succeeds the monomial ${x_1}^3{x_3}^5$, as the multidegrees of the monomials satisfy $(3,1,3) \succ (3,0,5)$. With this ordering, the leading coefficient and the leading term are respectively $\mathrm{LC}(f) = 2$ and $\mathrm{LT}(f) = 2{x_1}^3x_2{x_3}^3$. Using the alternative lex ordering $x_1 \succ x_2 \succ x_1$, the leading term becomes $\mathrm{LT}(f) = 4{x_1}^3{x_3}^5$.

In general, an ideal I does not have a unique basis, but given any two different bases $\langle p_1,\dots, p_s\rangle$ and $\langle g_1,\dots, g_m\rangle$ of I, the varieties $\mathbf{V}(p_1,\dots, p_s)$ and $\mathbf{V}(g_1,\dots, g_m)$ are equal. In other words, a variety only depends on the ideal generated by its defining equations. Some bases of an ideal may be simpler than other bases. Intuitively, if all the polynomials in a given basis of an ideal have a degree that is lower than the degree of any other polynomial in the ideal with respect to a particular monomial ordering, then this basis is in some sense the simplest basis. In particular, a Gröbner basis of an ideal (for a given monomial ordering) has such a property and can be thought of as the simplest or canonical basis. Given an ideal I and a monomial ordering, denote the set of leading terms of elements of I as $\mathrm{LT}(I)$. The ideal generated by elements of $\mathrm{LT}(I)$ is denoted $\langle \mathrm{LT}(I)\rangle$. In general, the ideal generated by the leading terms of a particular ideal I is not the same as the ideal generated by the leading terms of the polynomials in a basis for that particular ideal I. A Gröbner basis is a special basis for which this property holds, and it is formally defined as the set of polynomials $g_1,\dots, g_m$ for which $\langle \mathrm{LT}(I)\rangle = \langle \mathrm{LT}(g_1),\dots, \mathrm{LT}(g_m)\rangle$. When computing Gröbner bases, the user specifies a monomial ordering; different monomial orderings produce different Gröbner bases. Given a monomial ordering, the two most important properties of Gröbner bases are:

(1) Every ideal $I \subset k[x_1,\dots, x_n]$, other than the trivial ideal $\langle 0\rangle$, has a Gröbner basis. Furthermore, any Gröbner basis of an ideal I is a basis for I.
(2) Given an ideal $I \subset k[x_1,\dots, x_n]$, other than the trivial ideal $\langle 0\rangle$, a Gröbner basis of I can be computed in a finite number of algebraic operations.

The first algorithm for computation of Gröbner bases, published in the 1960s, is attributed to B. Buchberger (6). Since then a number of improvements have been reported and the algorithm has been implemented in most commercial symbolic software packages. Buchberger's algorithm generalizes two well-known algorithms: Gauss elimination for sets of multivariate linear algebraic equations, and Euclid's algorithm for computing the greatest common divisor of a set of univariate polynomials.

Solving Sets of Multivariate Polynomial Equations. Gröbner bases facilitate solving a set of multivariate polynomial equations (3) in the same way as the Gauss elimination algorithm facilitates solving a set of linear algebraic equations. Indeed, in a given lex ordering the Gröbner basis has a triangular structure reminiscent of the triangular structure in Gauss elimination. If the Gröbner basis is given by $\{1\}$, then the system of polynomial equations has no solution.

Equilibria for Polynomial Systems. Consider a polynomial system without inputs $\sigma x(t) = f(x(t))$. A state $x* \in \mathbb{R}^n$ is called an *equilibrium* of this system if $x(0) = x*$ implies that $x(t) \equiv x* \;\forall t$. Equilibria for polynomial systems are therefore obtained as solutions of a set of polynomial equations of the form $f(x) = 0$ or $f(x) = x$ for continuous-time or discrete-time systems respectively. Gröbner bases facilitate finding all equilibria.

Example: The equilibria of the polynomial system

$$\begin{aligned}\dot{x}_1 &= x_1 + x_2 - x_3^3\\ \dot{x}_2 &= x_1^2 + x_2 - x_3\\ \dot{x}_3 &= -x_1 + x_2^2 + x_3\end{aligned}$$

are obtained as solutions of the set of polynomial equations

$$p_1 := x_1 + x_2 - x_3^3 = 0, \quad p_2 := x_1^2 + x_2 - x_3 = 0, \quad p_3 := -x_1 + x_2^2 + x_3 = 0 \quad (6)$$

which is hard to solve. The Gröbner basis for the ideal $\langle p_1, p_2, p_3 \rangle$ using the lexicographic ordering $x_1 \succ x_2 \succ x_3$ is computed using Maple:

$$\begin{aligned} g_1 &= x_1 + x_2 - x_3^3, \\ g_2 &= x_2 - x_3^3 + x_2^2 + x_3, \\ g_3 &= 6x_3 - 3x_3^3 - 2x_3^4 - x_3^9 + 4x_3x_2 \\ g_4 &= 4x_3 - 8x_3^3 + 3x_3^5 + x_3^{11} \end{aligned} \quad (7)$$

By construction the equations $g_i = 0$, $i = 1, 2, 3, 4$, have the same solutions as Eq. (6). But obviously the polynomials g_i have a much simpler structure than the polynomials p_i. Indeed, g_4 depends only on x_3, and this allows $g_4 = 0$ to be solved numerically for x_3. The solutions can be substituted into $g_2 = 0$ and $g_3 = 0$ to obtain polynomials in x_2 only that can then be solved numerically. This process of back-substitution can be continued until all solutions are found.
Observe that a basis for an ideal does not have a fixed number of basis polynomials, and indeed the "simpler" Gröbner basis contains more polynomials than the original basis. As may be observed, it even contains more polynomials than there are variables.

Periodic Solutions for Polynomial Systems. Consider the system $f(d^ny/dt^n, \ldots, dy/dt, y, t) = 0$, where y can be thought of as the output of a closed-loop control system. Let f be a polynomial in all its variables. To approximately compute a periodic solution $y(t)$ of this system, the method of harmonic balancing can be used. It can be shown that under mild conditions an approximate periodic solution implies the existence of an exact periodic solution "close" to the approximate one (7). A truncated Fourier series can be considered as a candidate approximate periodic solution $y(t) = \Sigma_{k=-N}^{N} C_k e^{-j\omega kt}$, $c_k \in \mathbb{C}$, where c_k and c_{-k} are complex conjugates. Postulating that $y(t)$ is a solution leads to a set of polynomial equations, because f is polynomial. Using the Gröbner basis method with lex ordering, this set of equations can be solved for the unknowns c_0, $c_1, \ldots, c_N$ and ω to obtain $y(t)$.

In practice the actual solution may not be so important, and it often suffices to know that a periodic solution exists and to know its oscillation frequency ω. This can be achieved by finding the Gröbner basis of the polynomial equations with any lex ordering where ω is the lowest-ranking variable.

Example: Consider the van der Pol equation (8) $\ddot{y} - a\,(1 - by^2)\,\dot{y} + y = 0$, and postulate a periodic solution $y(t) = \Sigma$, where $c_k = c_{kr} + c_{ki}\,j$, $k = 0, 1, 2, 3$, and $j = \sqrt{-1}$. Substituting $y(t)$ into the van der Pol differential equation and equating the coefficients on the left-hand and right-hand sides of the equation leads to

$$\begin{aligned} p_1 &:= -c_{1r}\omega^2 - abc_{1r}^2 c_{3i}\omega + c_{1r} = 0 \\ p_2 &:= 2abc_{1r}c_{3r}^2\omega + abc_{1r}^2 c_{3r}\omega + 2abc_{1r}c_{3i}^2\omega + abc_{1r}^3\omega - ac_{1r}\omega = 0 \\ p_3 &:= -9c_{3r}\omega^2 - 3abc_{3i}c_{3r}^2\omega - 3abc_{3i}^3\omega - 6abc_{1r}^2 c_{3i}\omega + 3ac_{3i}\omega + c_{3r} = 0 \\ p_4 &:= 3abc_{3r}^3\omega - 9c_{3i}\omega^2 + 3abc_{3i}^2 c_{3r}\omega + 6abc_{1r}^2 c_{3r}\omega - 3ac_{3r}\omega + abc_{1r}^3\omega + c_{3i} = 0 \end{aligned}$$

and $c_0 = 0$, $c_2 = 0$, and $c_{1i} = 0$. The coefficients a and b are regarded as variables. To simplify the computations, p_1 is divided by $c_{1\tau}$, and p_2 by $ac_{1\tau}\omega$. A Gröbner basis of the ideal $\langle p_1, p_2, p_3, p_4\rangle$ with the lex ordering $c_{3\tau} \succ c_{3i} \succ c_{3\tau} \succ b \succ a \succ \omega$ contains seven polynomials of nontrivial complexity (42 pages of Maple output). It contains one polynomial in ω and a only:

$$\begin{aligned} g_7 : \ = \ & 3249\omega^{10} + (549a^2 - 6213)\omega^8 + (3745 - 567a^2)\omega^6 \\ & + (159a^2 - 842)\omega^4 + (53 - 13a^2)\omega^2 - 1. \end{aligned}$$

Obviously, for $a = 0$ the van der Pol oscillator is a simple harmonic oscillator with frequency $\omega = 1$. In the presence of nonlinear damping $a \neq 0$, the solution of $g_7 = 0$, which continues $\omega = 1$, can be expanded as

$$\omega^2 = 1 + \alpha_1 a^2 + \alpha_2 a^4 + \alpha_3 a^6 + \cdots$$

and using Maple, the coefficients α_i are obtained: $\alpha_1 = -\frac{1}{8}, \alpha_2 = -\frac{25}{256}, \alpha_3 = -\frac{625}{4096}$, etc.

Gröbner Bases and Elimination. Gröbner bases can be used to eliminate some variables of interest from a set of polynomial equations, as illustrated by Example 1, where x_1 and x_2 are eliminated from g_4 in Eq. (7). The variables that are ranked higher in the lex ordering are eliminated first. This feature of Gröbner bases can be used to obtain an estimate of the domain of attraction of equilibria for polynomial systems.

Estimation of the Domain of Attraction. If the closed-loop control system is stable, it is of interest to determine those initial conditions from which the solutions converge to a specified equilibrium. The collection of all the initial conditions that converge to the same equilibrium is called the *domain of attraction* of that particular equilibrium. Estimating domains of attractions is a hard problem. One way of obtaining a (conservative) estimate for the domain of attraction is to use Lyapunov functions. It is a standard result in Lyapunov theory that if $x = 0$ is an equilibrium point for the continuous-time system $d/dt\, x = f(x)$, if $D \subset \mathbb{R}^n$ is a domain containing $x = 0$, and if $W : D \to \mathbb{R}$ is a continuously differentiable function such that $W(0) = 0$ and for all $x \in D - \{0\}$ one has $W(x) > 0$ and $\partial W/\partial x\, f(x) < 0$, then $x = 0$ is asymptotically stable. Given such a Lyapunov function, consider the sets $\Omega = \{x \in \mathbb{R}^n : \partial W/\partial x f(x) < 0\}$ and $B_d = \{x \in \mathbb{R}^n : W(x) \leq d\}$. If $B_d \subset \Omega$ for some $d > 0$, then the set B_d is an estimate for the domain of attraction. For polynomial systems with a polynomial Lyapunov function W, Gröbner bases can be used to compute B_d systematically. Indeed, it is feasible to construct the largest such set by finding d such that B_d is as large as possible and still inside Ω. For polynomial systems with polynomial Lyapunov functions, $W(x) - d$ and $\partial W/\partial x\, f(x)$ are polynomials, and hence the boundaries of the sets B_d and Ω are varieties. At the points where the varieties $\mathbf{V}(W - d)$ and $\mathbf{V}\,(\partial W/\partial x f(x))$ touch each other, the gradients of W and $\dot{x}$ are parallel. Using this information, a system of $n + 2$ polynomial equations in $n + 2$ variables is obtained:

$$W - d = 0, \qquad \frac{\partial W}{\partial x} f(x) = 0, \qquad \frac{\partial}{\partial x}\left(\frac{\partial W}{\partial x} f(x)\right) - \lambda \frac{\partial W}{\partial x} = 0. \tag{8}$$

computing a Gröbner basis for the above system of equations, where the variable d has the least rank in the lex ordering, a polynomial equation in d only is obtained. The least positive solution to this equation is the "best" value of d for which $B_d \subset \Omega$, and this yields in turn the best estimate for the domain of attraction that could be obtained with the particular Lyapunov function W.

Example: Consider the system (8)

$$\begin{aligned} \dot{x_1} &= -x_1 + 2x_1^2 x_2 \\ \dot{x_2} &= -x_2 \end{aligned} \tag{9}$$

with the Lyapunov function $W(x) = 3x^2{}_1 + 4x_1x_2 + 4x^2{}_2$. The polynomials (8) are in this case

$$\begin{aligned} p_1 &:= 3x_1^2 + 4x_1x_2 + 4x_2^2 - d \\ p_2 &:= -6x_1^2 + 12x_1^3x_2 - 8x_1x_2 + 8x_1^2x_2^2 - 8x_2^2 \\ p_3 &:= 6x_1 + 4x_2 - \lambda(36x_2x_1^2 - 12x_1 - 8x_2 + 16x_2^2x_1) \\ p_4 &:= 4x_1 + 8x_2 - \lambda(12x_1^3 - 8x_1 + 16x_2x_1^2 - 16x_2) \end{aligned} \tag{10}$$

and using the lexicographic ordering $x_2 \succ \lambda \succ x_1 \succ d$, the Gröbner basis of the ideal $\langle p_1, p_2, p_3, p_4\rangle$ is computed. It contains the polynomial $g(d) = 4d^4 - 147d^3 + 786d^2 + 2048d$. The approximate solutions of $g(d) = 0$ *are* -1.9223, 0, 8.9657, 29.707. The smallest positive value of d for which there is a solution to the system of equations $p_i = 0$ *is* 8.9657. The corresponding estimate for the domain of attraction is therefore $\{x \in \mathbb{R}^2 : W(x) \leq 8.9657\}$.

Equality of Ideals. A special type of a Gröbner basis can be used to decide if two ideals are the same or not. This is the so-called *reduced* Gröbner basis. A reduced Gröbner basis for an ideal I is a Gröbner basis G for I such that $\mathrm{LC}(p) = 1$ for all $p \in G$, and for all $p \in G$, no monomial of p lies in $\langle \mathrm{LT}(G - \{\mathrm{p}\})\rangle$. The main property of reduced Gröbner bases is that given an arbitrary ideal $I \neq \langle 0\rangle$ and a particular monomial ordering, I has a *unique* reduced Gröbner basis. Hence, two ideals J_1 and J_2 are the same if and only if their reduced Gröbner bases G_1 and G_2 are the same (G_1 and G_2 must be computed with the same monomial ordering). Most commercial computer algebra systems, such as Maple and Mathematica, provide finite algorithms for computing reduced Gröbner bases.

Analysis of Discrete-Time Polynomial Systems. Fundamental control-theoretic concepts such as controllability or observability can be reduced to the problem of computing maximal control-invariant varieties. This is well known for linear systems (computation of control-invariant subspaces), but it holds equally well for discrete-time polynomial systems of the form $x(t + 1) = f(x(t),u(t))$. For such systems, a variety V is control-invariant if $f(V,u) \subset V$ for all possible control inputs u. The computation of the maximal control-invariant subset of a given variety V can be completed in finitely many operations. Consider the defining ideal of the variety V; say $J_1 = \langle g_{1,1},\ldots, g_{1,m1}\rangle$, where $g_{j,k} \in \mathbb{Q}[x]$. If the variety corresponding to J_1 were control-invariant, then $g_{1,\mathrm{k}} \circ f(x,u) \equiv 0$ for all u and all $k = 1,\ldots, m_1$. The polynomials $g_{1,k} \circ f(x,u)$ can be viewed as polynomials in u with coefficients in $\mathbb{Q}[x]$. Denote the collection of all these coefficients as $g_{2,\mathrm{k}}$, $k = 1, 2,\ldots, m_2$, and the corresponding ideal as $J_2 = \langle g_{2,1},\ldots, g_2,m_2\rangle$. Invariance would imply that $J_1 = J_2$, and if this is not the case, then obviously $J_1 \subset J_2$ and the corresponding varieties satisfy $V_1 \supset V_2$. This process can be continued to construct an ascending chain of ideals (or descending chain of varieties) $J_1 \subset J_2 \subset J_3 \subset \cdots$. This chain must terminate in finitely many steps (6). That is, there exists an integer N such that $\cdots \subset J_{N-1} \subset J_N = J_{N+1} = \cdots$. The variety $V(J_N)$ is the maximal control-invariant subset of $V(J_1)$. The check whether $J_k = J_{k+1}$ can be completed via the computation of reduced Gröbner bases for J_k and J_{k+1}.

Observability. The discrete-time polynomial system

$$\begin{aligned} x(t+1) &= f(x(t), u(t)) \\ y(t) &= h(x(t)) \end{aligned} \tag{11}$$

as said to be *observable* if for each pair of initial states $\zeta \neq \eta$, there exists an integer N and an input sequence $U_N = \{u(0), \ldots, u(N-1)\}$ such that the solutions starting at ζ and η produce different outputs after N steps, that is $h(x(N, \zeta, U_N)) \neq h(x(N, \eta, U_N))$. The polynomial $h(x(N, \zeta, U_N)) - h(x(N, \eta, U_N))$ can be regarded as a polynomial in elements of U_N with coefficients in $\mathbb{Q}[\xi, \eta]$. A chain of ideals J_k is constructed using these coefficients, and at each step the condition $J_k = J_{k+1}$ is tested. It can be shown that if J_N for some N has a reduced Gröbner basis $\{\zeta_1 - \eta_1, \ldots, \zeta_n - \eta_n\}$, then $J_{N+1} = J_N$ and the system is observable. The above-discussed algorithm for computing invariant sets streamlines these computations and allows a systematic determination of the integer N.

Example: Consider the simple Wiener system (9)

$$\begin{aligned} x_1(k+1) &= x_2(k), \\ x_2(k+1) &= -x_1(k) - 2x_2(k) + u(k), \\ y(k) &= x_1^2(k). \end{aligned} \tag{12}$$

The system consists of a linear dynamical system and a quadratic static output nonlinearity. The ideal $J_1 = \langle \eta_1{}^2 - \zeta_1{}^2 \rangle$ and the output equation (12) are used to construct the following ideals:

$$\begin{aligned} J_2 &= \langle \eta_1^2 - \xi_1^2, \eta_2^2 - \xi_2^2 \rangle, \\ J_3 &= \langle \eta_1^2 - \xi_1^2, \eta_2^2 - \xi_2^2, (\eta_1 + 2\eta_2)^2 - (\xi_1 + 2\xi_2)^2, \\ &\quad (\eta_1 + 2\eta_2) - (\xi_1 + 2\xi_2) \rangle, \\ J_4 &= \langle \eta_1^2 - \xi_1^2, \eta_2^2 - \xi_2^2, (\eta_1 + 2\eta_2)^2 - (\xi_1 + 2\xi_2)^2, \\ &\quad (\eta_1 + 2\eta_2) - (\xi_1 + 2\xi_2), (2\eta_1 + \eta_2)^2 - (2\xi_1 + \xi_2)^2, \\ &\quad 2\eta_1 + \eta_2 - 2\xi_1 - \xi_2, -2\eta_1 - 3\eta_2 + 2\xi_1 + 3\xi_2 \rangle. \end{aligned} \tag{13}$$

Using the lex ordering $\zeta_1 \succ \zeta_2 \succ \eta_1 \succ \eta_2$, the reduced Gröbner basis for J_4 is $G_4 = \{\eta_1 - \zeta_1, \eta_2 - \zeta_2\}$, *and therefore the system* (12) *is observable with* $N = 4$.

A Brief Overview of the Literature and Related Problems. Gröbner bases are useful for a range of other control problems, such as the inverse kinematic problem and motion planning in robotics (6), the computation of the switching surfaces in the solution of time optimal control problems (10,11), identifiability, input–output equivalence of different state-space realizations, normal forms and zero dynamics (12), analysis of hybrid control systems, computation of limit cycles for discrete-time polynomial systems, observability of continuous-time polynomial systems, and forward accessibility of discrete-time polynomial systems.

In control design for linear systems, the linear functional observer problem, the model-matching problem, the deterministic identification problem, and the disturbance decoupling problem play a central role. All these problems can be seen as the characterization of a maximally control-invariant subspace of a given subspace of the state space (2). In control theory this problem is known as the *cover problem*. The cover problem can be solved via a computation of elementary divisors of matrix pencils, which in turn leads to multilinear equations where the unknowns are the basis vectors of the invariant subspace. The Gröbner basis method facilitates the analytic solution of this problem. The elegance of this approach stems from the fact in one single algorithm all possible (typically infinitely many) control-invariant subspaces can be explicitly characterized. The degrees of freedom established in this process can then be further exploited to optimize other desirable system properties, such as sensitivity.

We have focused on the use of Gröbner bases for commutative rings over infinite fields, that is, finding Gröbner bases in the context of polynomials with rational coefficients. Two important related areas are Gröbner bases for commutative rings over finite fields and Gröbner bases in the context of noncommutative rings. The former is of interest in the context of discrete-event dynamic systems (13,14) and coding theory in communications. The latter is useful for control-theoretic problems involving polynomial matrices (15). In particular, in 15 a new extra term in the expansion of the classical state-feedback optimal control problem in the singular perturbation form was obtained using noncommutative Gröbner bases.

An important drawback of Buchberger's algorithm is that even with the best known versions of the algorithm, it is easy to generate examples for which the computation of a Gröbner basis requires inordinate amounts of computer resources (time and/or memory). The main bottlenecks are that the total degrees of the intermediate polynomials that the algorithm generates may be quite large and that the coefficients of the Gröbner basis may be complicated rational numbers. This may be the case even if the original ideal generators are polynomials of small degrees with small integer coefficients. In general, the intermediate polynomials observed in the computation of a Gröbner basis can have total degrees of the order of 2^{2d}, where d is the total degree of the ideal generators. Although this appears to be a rather negative result, it typifies the worst-case scenario. It appears that the running time and the storage requirements seem to be much more manageable on average. It is important to emphasize that different monomial orderings may produce very different computational times, and some experimentation with the ordering may yield significant reductions in computation time.

Differential Algebra In Control

In the 1980s differential algebra was applied to control problems (16). Differential algebra can be used to transform a polynomial system from one representation to another. In general differential algebra plays an important role in the realization theory of nonlinear control systems. In the context of control, the problems that lend themselves to differential algebra are varied: the determination of observability, identifiability, the calculation of zero dynamics, regulator computations, tracking control, etc.

Differential-Algebraic Tools. Differential algebra provides tools for dealing with systems of polynomial differential equations. In this algebra the derivative operation occupies center stage. A multivariate polynomial in variables $y_1, \ldots, y_N$ and their derivatives is called a *differential polynomial* in $y_1, \ldots, y_N$. For instance, $f(d^n y/dt^n, \ldots, dy/dt, y)$ is a differential polynomial in y if f is a polynomial in all its variables.

The concept of *ranking* is introduced for differential polynomials. It is very similar to the concept of monomial ordering for polynomials. Ranking is a total ordering of all variables and their derivatives. Examples involving two variables are

$$u \prec y \prec \dot{u} \prec \dot{y} \prec \ddot{u} \prec \ddot{y} \prec \cdots$$

and

$$u \prec \dot{u} \prec \ddot{u} \prec \cdots \prec y \prec \dot{y} \prec \ddot{y} \prec \cdots$$

where $\prec$ denotes "is ranked lower than." Any ranking is possible provided it satisfies two conditions:

$$u^{(\mu)} \prec u^{(\mu+\sigma)}$$
$$u^{(\mu)} \prec y^{(v)} \quad \Rightarrow \quad u^{(\mu+\sigma)} \prec y^{(v+\sigma)}$$

for all variables u and y, all nonnegative integers μ and v, and all positive integers σ. The highest-ranking variable or derivative of a variable in a differential polynomial is called the *leader*. The ranking of variables gives a ranking of differential polynomials. They are simply ranked as their leaders. If two have the same leader, they are considered as polynomials in their leader and the one of lower degree is ranked lower.

Let A, B be two differential polynomials, and let A have the leader v. Then B is said to be *reduced* with respect to A if there is no derivative of v in B and if B has lower degree than A when both are regarded as polynomials in v. A set

$$A_1, \ldots, A_p$$

of differential polynomials is called *autoreduced* if all the A_i are pairwise reduced with respect to each other. Normally autoreduced sets are ordered so that $A_1,\ldots, A_p$ are in increasing rank. Autoreduced sets are ranked as follows. Let $A = A_1,\ldots, A_r$ and $B = B_1,\ldots, B_s$ be two ordered autoreduced sets. A is ranked lower if either there is an integer k, $0 \leq k \leq \min(s, r)$, such that

$$\text{rank } A_j = \text{rank } B_j, \qquad j = 0, \ldots, k-1$$
$$\text{rank } A_k \prec \text{rank } B_k$$

or else if $r > s$ and

$$\text{rank } A_j = \text{rank } B_j, \qquad j = 0, \ldots, s$$

A *characteristic set* for a given set of differential polynomials is an autoreduced subset such that no other autoreduced subset is ranked lower.

The *separant* S_A of a differential polynomial A is the partial derivative of A with respect to the leader, while the *initial* $\boldsymbol{I}_{\mathbf{A}}$ is the coefficient of the highest power of the leader in A. If a differential polynomial f is not reduced with respect to another differential polynomial g, then either f contains some derivative of the leader u_g of g or else f contains u_g to a higher power. In the former case one can differentiate g a suitable number (say σ) of times and perform a pseudodivision to remove that derivative, giving a relation

$$S^{v} f = Qg^{(\sigma)} + R \tag{14}$$

where S is the separant of g and R does not contain the highest derivative of u_g that is present in f. If f contains u_g to a higher power, a pseudodivision of f by g can be performed to obtain

$$I^{v} f = Qg + R \tag{15}$$

where I is the initial of g, and R is reduced with respect to g.

Observability. In control theory a system is called observable if it is possible to compute the state from inputs and outputs. Using differential algebra, the observability question can be answered in a constructive manner for continuous-time polynomial systems (1).

Example: Consider the system

$$\dot{x}_1 = x_2^2 \tag{16}$$

$$\dot{x}_2 = u \tag{17}$$

$$y = x_1 \tag{18}$$

which is in state-space form. Suppose an input–output description (i.e., a description directly relating u and y) is called for. The original set of equations is equivalent to

$$y - x_1 = 0, \qquad \dot{y} - x_2^2 = 0, \qquad \dot{x}_2 - u = 0 \tag{19}$$

It is now possible to eliminate the derivative of x_2 by forming

$$p = \frac{d}{dt}(\dot{y} - x_2^2) + 2x_2(\dot{x}_2 - u) = \ddot{y} - 2x_2u \tag{20}$$

From this construction it follows that $p = 0$ whenever the equations of (19) are satisfied. The last equation of (19) can be replaced by $p = 0$ to get the system description

$$y - x_1 = 0, \qquad \dot{y} - x_2^2 = 0, \ddot{y} - 2x_2u = 0 \tag{21}$$

From (Eq. 20) it follows that every solution of Eq. (19) also solves Eq. (21). If, moreover, it is known that $x_2 \neq 0$, then the converse is also true, and Eqs. (19) and (21) are equivalent. It is now possible to form

$$x_2(\ddot{y} - 2x_2u) - 2u(\dot{y} - x_2^2) = x_2\ddot{y} - 2u\dot{y} \tag{22}$$

and it readily follows that

$$y - x_1 = 0, \qquad \ddot{y} - 2x_2u = 0, \qquad x_2\ddot{y} - 2u\dot{y} = 0 \tag{23}$$

is equivalent to (Eq. 21) if also $u \neq 0$. Finally form

$$\ddot{y}(\ddot{y} - 2x_2u) + 2u(x_2\ddot{y} - 2u\dot{y}) = \ddot{y}^2 - 4u^2\dot{y} \tag{24}$$

to conclude that, provided $ux_2 \neq 0$, Eq. (19) is equivalent to the following system description:

$$\ddot{y}^2 - 4u^2\dot{y} = 0, \qquad y - x_1 = 0, \qquad \ddot{y} - 2x_2u = 0 \tag{25}$$

The leftmost equation is an input–output relation, while the middle and right equations show how x_1 and x_2 can be computed from the input and output. This establishes the observability, and more.
Using the terminology of differential algebra, the state-space form is an autoreduced set under the ranking

$$u \prec x_1 \prec x_2 \prec y \prec \dot{u} \prec \dot{x}_1 \prec \dot{x}_2 \prec \dot{y} \prec \cdots$$

while Eq. (25) is an autoreduced set under the ranking

$$u^{(\cdot)} \prec y^{(\cdot)} \prec x_1^{(\cdot)} \prec x_2^{(\cdot)}$$

Autoreduced sets thus give a generalization of several of the standard forms for dynamic systems.

Global Identifiability. A polynomial system that is parametrized with a (constant) parameter θ as

$$\begin{aligned} \dot{x} &= f(x, u, \theta) \\ y &= h(x, \theta) \end{aligned} \tag{26}$$

as said to be *globally identifiable* if, given input and output measurements $u(t)$ and $y(t)$ over an interval $t \in [0,T]$, there is only one constant value of θ that satisfies Eq. (26). Global identifiability of polynomial systems (1) can be checked in a constructive manner using the tools from differential algebra. Moreover, conditions on u and y under which the system is not globally identifiable can also be obtained, as is illustrated below.

Example: Consider the system

$$\ddot{y} + 2\theta\dot{y} + \theta^2 y = 0$$

where θ is a constant parameter. Can θ be identified by observing the output y? Using the notation $p = \ddot{y} + 2\theta\,\dot{y} + \theta^2 y$, compute the expression

$$\dot{y}p - y\dot{p} = \dot{y}\ddot{y} - yy^{(3)} + 2\theta(\dot{y}^2 - y\dot{y})$$

It follows readily that θ is uniquely determined by y (and its derivatives), provided that along the output trajectory

$$(\dot{y}^2 - y\dot{y}) \neq 0$$

This is thus an excitation condition that guarantees global identifiability.

The Algorithms of Ritt and Seidenberg. The concepts of the previous section were introduced in the 1930s by the American mathematician Ritt, (17). He devised an algorithm by which it is possible to start with an arbitrary number of differential polynomials, introducing a suitable ranking of the variables and performing successive operations of the type described by Eqs. (14), (15) to arrive at an equivalent representation in the form of an autoreduced set of equations. This set has the property that an f belonging to the original differential polynomials can be reduced to zero using (Eqs. 14) and (15) for different g's belonging to the set. If certain factorization conditions are met, then Ritt showed that the generated autoreduced set is a characteristic set, not only for the explicitly generated polynomials, but also for an infinite set of polynomials constituting the so-called differential ideal. Ritt used his algorithmic procedure to produce an extensive theory for polynomial systems of differential equations. Later a more systematic algebraic treatment was given by Kolchin (18). The theory was also extended to partial differential equations (18) and difference equations (19).

From the viewpoint of practical calculations, a major drawback of Ritt's algorithm is the factorization it requires, since this is a task of high computational complexity. Seidenberg (20) has proposed an algorithm for deciding the solvability of systems of equations and inequations. The algorithm uses (Eqs. 14), (15) repeatedly to eliminate variables and reduce the problem to the single-variable case. As in the simple example above, equivalence of the original and final sets of equations requires the separant S and initial I of (Eqs. 14), (15) to be nonzero. To handle this, Seidenberg's algorithm splits the problem into different subproblems, considering the cases $S = 0$, $S \neq 0$ (or $I = 0$, $I \neq 0$) together with the original equations and inequations. By repeated splitting a tree of subproblems is generated. Each subproblem is finally reduced to a single-variable problem for which solvability can be decided. In recent years several variations of Ritt's and Seidenberg's algorithms have been implemented in symbolic manipulation languages like Maple or Mathematica.

Other Control Problems. Differential-algebra calculations can be used to compute the zero dynamics of a system. This dynamics is obtained by restricting the output to be zero. By adding the equation $y = 0$ and using Ritt's algorithm with a suitable ranking, it is possible to obtain a description of the resulting dynamics. In a similar manner a regulator description can be obtained. Suppose a differential equation describing the desired response from reference signal to output is given. Using this equation together with the system dynamics as the input to Ritt's algorithm, it is possible to arrive at an equation containing the control signal together with the reference and output signals. This equation can be interpreted as a description of the regulator, although it can be difficult to implement due to its implicit nature in the general case.

Quantifier Elimination

Quantifier elimination is a method for rewriting formulae that include quantifiers such as "for all" ($\forall$) and "there exists" ($\exists$) in an equivalent form without the quantifiers and the quantified variables. According to a theorem by Tarski (21), it is always possible to eliminate the quantified variables in formulae consisting of logical combinations of multivariate polynomial equations and inequalities. Tarski provided a constructive proof in the late 1940s, but the corresponding algorithm has such complexity that it is impractical for most problems. In the mid 1970s Collins (22) presented a new method, the so-called cylindrical algebraic decomposition, which exhibits much better complexity. Since then the algorithmic development has made significant progress (23,24, 25). Nevertheless, cylindrical algebraic decomposition and quantifier elimination are known to be inherently complex (26). It is therefore of importance to identify classes of problems for which the computational complexity is much lower, and for which specialized algorithms can be developed (27,28). Implementations of cylindrical algebraic decomposition and quantifier elimination are available in for example Mathematica (29). Depending on the specific nature of the problem posed, different algorithms are used in order to minimize the computational burden. The algorithmic developments are ongoing.

Introductions to cylindrical algebraic decomposition and quantifier elimination can be found in Ref. 30. An extensive bibliography covering early papers can be found in Ref. 31, and a survey of the algorithmic development of cylindrical algebraic decomposition is given in Ref. 32.

An early application of quantifier elimination techniques in control theory was made by Anderson et al. (33). This contribution predates the introduction of efficient computational tools, and so it was of theoretical interest only. With the availability of the experimental software QEPCAD (23), the number of papers related to control increased. Now, as these methods are implemented in widely available symbolic computation software packages like Mathematica, the control practitioner can explore their potential.

Systems of Real Algebraic Equations and Inequalities. In the context of polynomial equations with real coefficients, inequalities arise naturally, for example to express when a quadratic polynomial has real solutions. The need of using not only equations but also inequalities in describing computational problems triggered the development of methods for doing symbolic computations with so-called *semialgebraic* sets. A semialgebraic set is a generalization of an algebraic set or variety, as it is the solution set of a system of real algebraic equations and inequalities.

An *algebraic expression in variables* $\{x_1,\dots, x_n\}$ is an expression constructed with $\{x_1,\dots, x_n\}$ and rational numbers, using addition, multiplication, rational powers, and algebraic numbers and functions. A *system of real algebraic equations and inequalities in variables* $\{x_1,\dots, x_n\}$ is a logical combination of equations and inequalities with both sides being algebraic expressions in $\{x_1,\dots, x_n\}$. The notation $\wedge$ (and), $\vee$ (or), $\Rightarrow$ (implies) for Boolean operators is used.

The following is an example of a system of real algebraic equations and inequalities in the variables a, b, x, y:

$$(x^2 + ax + b \geq 0 \quad \wedge \quad x \geq 0 \quad \wedge \quad b^3 + b - 2 = 0) \quad \Longrightarrow \quad x^2 + y^2 - 1 \leq 0$$

Alternatively, a semialgebraic set can be characterized as a set obtained by finitely many unions, intersections, and complementations of sets of the form $\{x \in \mathbb{R} | f(x) \geq 0\}$. Here f is a multivariate polynomial with rational coefficients. Semialgebraic sets are thus closed under projection, union, and intersection. As Gröbner bases provide a way to replace a system of multivariate polynomials by a simpler equivalent set, there are systematic ways to simplify a system of real algebraic equations and inequalities. It can be shown that the set of solutions of any system of real algebraic equations and inequalities in variables $\{x_1,\dots, x_n\}$ can be written as a disjunction of a finite number of *cylindrical* parts of the form

$$\begin{aligned} f_1 \preceq x_1 \preceq g_1 \quad \wedge \quad f_2(x_1) \preceq x_2 \preceq g_2(x_1) \quad \wedge \cdots \\ \wedge \quad f_n(x_1, \dots, x_{n-1}) \preceq x_n \preceq g_n(x_1, \dots, x_{n-1}) \end{aligned}$$

In the above expression, $\preceq$ stands for one of $<$, $\leq$, and $=$; and f_i and g_i are either $-\infty$, ∞, or algebraic expressions in the variables $\{x_1,\dots, x_{i-i}\}$ that are real-valued for all tuples of real numbers $\{a_1,\dots, a_{i-i}\}$ satisfying

$$\begin{aligned} f_1 \preceq a_1 \preceq g_1 \quad \wedge \quad f_2(a_1) \preceq a_2 \preceq g_2(a_1) \quad \wedge \dots \\ \wedge \quad f_{i-1}(a_1, \dots, a_{n-2}) \preceq a_{i-1} \preceq g_{i-1}(a_1, \dots, a_{i-2}) \end{aligned}$$

The method of rewriting a real algebraic system as a disjunction of the above form is called *cylindrical algebraic decomposition*; see Refs. 22,30,32. Observe the triangular nature of the resulting system.

Example: An example of a script in Mathematica for computing a cylindrical algebraic decomposition of an ellipsoid in $\mathbb{R}^3$ is

$$\begin{aligned} &\mathrm{In[1]} := \mathrm{CylindricalAlgebraicDecomposition}[\mathrm{x}^2 + 4\mathrm{y}^2 + 8\mathrm{z}^2 < 1, \{\mathrm{x}, \mathrm{y}, \mathrm{z}\}] \\ &\mathrm{Out[1]} = -1 < \mathrm{x} < 1\&\& -\frac{1}{2}\sqrt{1-\mathrm{x}^2} < \mathrm{y} < \frac{\sqrt{1-\mathrm{x}^2}}{2}\&\& \\ &\qquad -\frac{1}{2\sqrt{2}}\sqrt{1-\mathrm{x}^2-4\mathrm{y}^2} < \mathrm{z} < \frac{\sqrt{1-\mathrm{x}^2-4\mathrm{y}^2}}{2\sqrt{2}} \end{aligned}$$

Here the functions f_i and g_i can be expressed in terms of simple algebraic operations and radicals. More typically the full generality offered by algebraic functions is required:

$$\begin{aligned}
\mathrm{In}[1] :=\ & \mathrm{CylindricalAlgebraicDecomposition}[\mathrm{x}^2 + 4\mathrm{y}^2 + 8z^{12} < 1, \{\mathrm{x}, \mathrm{y}, \mathrm{z}\}] \\
\mathrm{Out}[1]=\ & -1 < \mathrm{x} < 1\&\& - \frac{1}{2}\sqrt{1-\mathrm{x}^2} < \mathrm{y} < \frac{\sqrt{1-\mathrm{x}^2}}{2}\&\& \\
& \mathrm{Root}[-1 + \mathrm{x}^2 + 4\mathrm{y}^2 + 8\#1^{12}\&, 1] < \\
& \mathrm{z} < \mathrm{Root}[-1 + \mathrm{x}^2 + 4\mathrm{y}^2 + 8\#1^{12}\&, 2]
\end{aligned}$$

In Mathematica algebraic functions are represented by special Root objects. In the above case the placeholder variable #1 is a function of x and y. The last argument enumerates the branches of the algebraic function, which in this case are the first two roots according to Mathematica's enumeration scheme of the 12 possible complex roots.

A quantified system of real algebraic equations and inequalities in variables $\{x_1,\ldots, x_n\}$ is an expression of the form

$$QS = Q_1(y_1)Q_2(y_2)\ldots Q_m(y_m)S(x_1,\ldots,x_n\,; y_1,\ldots,y_m)$$

where the Q_i's are the quantifiers $\exists$ or $\forall$, and S is a system of real algebraic equations and inequalities in $\{x_1,\ldots, x_n; y_1,\ldots, y_m\}$. According to Tarski's theorem the solution set of quantified systems of real algebraic equations and inequalities is a semialgebraic set, which means that QS always can be rewritten as an equivalent expression without any quantifiers and quantified variables. The process of eliminating quantified variables from such systems is called *quantifier elimination*. The y_i's in QS are called *bounded* variables, and the x_i's *free* variables. Problems without free variables are called *decision problems*.

Example: The proof of the well-known inequality between the arithmetic and the geometric mean of two real positive numbers can be formulated as a decision problem:

$$\begin{aligned}
&\mathrm{In}[1] := \mathrm{Resolve}[\forall_{\{\mathrm{a},\mathrm{b}\}\,\mathrm{a}\geq 0\&\&\mathrm{b}\geq 0}\sqrt{\mathrm{ab}} \leq \frac{\mathrm{a}+\mathrm{b}}{2}] \\
&\mathrm{Out}[1] = \mathrm{True}
\end{aligned}$$

The notation for quantifiers in Mathematica is

$$Q_{\mathrm{var,con}}$$

where Q is either $\forall$ or $\exists$, var is a single variable or list of variables, and cond gives additional conditions on the variables such as their domain and/or semialgebraic constraints. The command to perform quantifier elimination in Mathematica is called Resolve.

Projection of semialgebraic sets is equivalent to quantifier elimination on existential quantifiers. Consider for example the projection of the ellipsoid in Example 7 onto the *xy* plane:

$$\mathrm{In}[1]:=\mathrm{Resolve}[\exists_{z,(x|y|z)\in\mathrm{Reals}}x^2+4y^2+8z^2<1\{x,y\}]$$
$$\mathrm{Out}[1]=-1<x<1\&\&-\frac{1}{2}\sqrt{1-x^2}<y<\frac{\sqrt{1-x^2}}{2}$$

Set inclusion for semialgebraic sets is another useful property that can be formulated as a decision problem: $A \subseteq B$ is the decision problem $\forall_{x\in A}B$:

$$\mathrm{In}[1]:=\mathrm{Resolve}[\forall_{\{x,y\}x^2+xy+3y^2\le 1\&\&\{x,y\}\in\mathrm{Reals}}x^4+y^2<2]$$
$$\mathrm{Out}[1]=\mathrm{True}$$

Sets of Equilibria. As illustrated before, Gröbner bases are useful to determine equilibria for polynomial systems. In polynomial control systems it is of importance to describe the steady-state solutions: the set of reachable equilibria, for admissible input values. Characterizing the set of reachable equilibria in state space is a quantifier elimination problem.

Example: Add a control input to the last equation in Example 1. Let this input be constrained to the interval $-1 \le u \le 1$. Compute the projection of the set of corresponding equilibria onto the x_1x_2 plane.

$$\begin{aligned}\mathrm{In}[1]:=\ &\mathrm{Resolve}[\\ &\exists_{\{u,x_3\},-1\le u\le 1\&\&(x_1|x_2|x_3|u)\in\mathrm{Reals}}x_1+x_2-x_3^3=0\,\&\&\\ &x_1^2+x_2-x_3=0\&\&-x_1+x_2^2+x_3+u=0]\\ \mathrm{Out}[1]=\ &x_1-x_1^6+x_2-3x_1^4x_2-3x_1^2x_2^2-x_2^3=0\,\&\&\\ &-1-x_1+x_1^2+x_2+x_2^2\le 0\,\&\&-1+x_1-x_1^2-x_2-x_2^2\le 0\end{aligned}$$

Stability. To determine the local stability of equilibria of dynamic systems the stability tests of Routh and Hurwitz (34) may be used. These stability tests give explicit conditions on the coefficients of the characteristic equation of the Jacobian of the dynamical system at equilibrium in terms of a number of polynomial inequalities. Hence the Routh–Hurwitz test can be handled using cylindrical algebraic decomposition and quantifier elimination. A less well-known, but equivalent stability test is the Lienard–Chipart criterion (34), which has the advantage over the Routh criterion that it involves polynomial inequalities of lower degree.

Given a polynomial in $\mathbb{R}[s]$ $a_0s^n + a_1s^{n-1} + \cdots + a_{n-1}s + a_n$, $a_i > 0$, the Lienard–Chipart criterion states that all its zeros have strictly negative real parts if and only if the following inequalities holds:

$$a_n>0,\quad a_{n-2}>0,\quad a_{n-4}>0,\ldots\quad\text{and}\quad D_2>0\quad D_4>0,\ldots$$

where D_i, $i = 1,\dots, n$, are the so-called Hurwitz determinants of order i, which are defined as follows:

$$D_i \triangleq \det \begin{bmatrix} a_1 & a_3 & a_5 & a_7 & \dots & (a_{2i}-1) \\ a_0 & a_2 & a_4 & a_6 & \dots & (a_{2i}-2) \\ 0 & a_1 & a_3 & a_5 & \dots & (a_{2i}-3) \\ 0 & a_0 & a_2 & a_4 & \dots & (a_{2i}-4) \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & a_i \end{bmatrix}$$

where $a_k = 0$ for $k > n$.

Example: Consider the following polynomial system with real parameters a and b. Determine for which values of a and b the origin is a stable equilibrium (this is an essential step in any bifurcation analysis of a dynamical system—the classification of possible asymptotic behaviors of the system):

$$\begin{aligned} \dot{x_1} &= ax_1 + x_2 - x_3^3 \\ \dot{x_2} &= x_1^2 + bx_2 - x_3 \\ \dot{x_3} &= -x_1 + x_2^2 + bx_3 + u \end{aligned}$$

For zero input $u = 0$ the zero state is an equilibrium. Linearizing the system around $(x,u) = (0,0)$ and computing the characteristic polynomial of the resulting state transition matrix gives

$$-1 - ab^2 + (2ab + b^2)s - (a + 2b)s^2 + s^3$$

Compute the solution set corresponding to the Lineard–Chipart stability inequalities for this polynomial:

$$\begin{aligned} &\text{In[1]} := \text{CylindricalAlgebraicDecomposition[} \\ &\qquad -\text{a} - 2\text{b} > 0 \,\&\&\, -1 - \text{ab}^2 > 0 \,\&\& \\ &\qquad -1 + 2\text{a}^2\text{b} + 3\text{ab}^2 + 2\text{b}^3 + 2\text{a}^3\text{b}^3 + 4\text{a}^2\text{b}^4 + 2\text{ab}^5 > 0] \\ &\text{Out[1]} = \text{a} < 0 \,\&\&\, \text{b} < -\frac{1}{\sqrt{-\text{a}}} \end{aligned}$$

Example: Consider the problem of stabilizing in a unit-feedback scheme an unstable system G, using a lead compensator F that is parametrized by the parameters B and b, where

$$G = \frac{4}{s^2 - 2s + 2}, \qquad F = B\frac{s+b}{s+Bb}$$

The Lienard–Chipart inequalities for the closed-loop system are

$$6bB > 0 \,\&\& -2 + bB > 0 \quad \text{and} \quad -24bB - 48B^2 + 24b^2B^3 - 12b^3B^3 > 0$$

These can be rewritten using cylindrical algebraic decomposition to either

$$0 < b < 2 \quad \wedge \quad B \leq -\frac{2}{(-2+b)b} + \sqrt{2}\sqrt{\frac{-2-2B+b^2}{(-2b+b^2)^2}}$$

or

$$B > 2+\sqrt{6} \quad \wedge \quad 1-\sqrt{\frac{-2-4B+B^2}{B^2}} < b < 1+\sqrt{\frac{-2-4B+B^2}{B^2}}$$

depending on the choice of the order for b and B. Using quantifier elimination, questions like "For which values of b can it be guaranteed that for all values of $5 < B < 10$ the closed loop system is stable?" can be answered:

$$\begin{aligned}\text{In[1]} :&= \text{Resolve}[\forall_{\text{B},5<\text{B}<10\,\&\&(\text{b}|\text{B})\in\text{Reals}}\\ &\quad 6\text{bB} > 0\,\&\& - 2 + \text{bB} > 0\,\&\&\\ &\quad -24\text{bB} - 48\text{bB}^2 + 24\text{b}^2\text{B}^3 - 12\text{b}^3\text{B}^3 > 0]\\ \text{Out[1]} &= \frac{1}{5}(5-\sqrt{3} < \text{b} < \frac{1}{5}(5-\sqrt{3})\end{aligned}$$

Numerically this corresponds to $0.65359 < b < 1.34641$.

Nonlinear Tracking. Tracking a particular output is an important control objective. There are many control design methods that deal with this problem, mainly based on optimization techniques. However, most (analytic) methods do not take the ever present constraints on control signals or states into account. As a consequence the resulting control system must be validated, often through extensive simulations. For a large class of nonlinear tracking problems, quantifier elimination can be used to decide if there exists a solution to a given tracking problem subject to the constraints of interest.

Consider the continuous-time polynomial control system

$$\dot{x} = f(x,u) \tag{27}$$

where $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$ and each component of f is a polynomial in all its variables. Furthermore, the input value $u(t)$ is required to belong to some semialgebraic set U:

$$u(t) \in U \qquad \forall t$$

called the *admissible* control set. Let Γ be a rationally parametrized curve in the state space, i.e.,

$$\Gamma = \{x \in \mathbb{R}^n | x = g(s) \quad \wedge \quad \alpha \leq s \leq \beta\}$$

where $g : \mathbb{R} \rightarrow \mathbb{R}^{\text{n}}$ is a rational function and the orientation of Γ is defined by increasing values of s.

For the state of the system (27) to follow the curve Γ, there has to be an admissible control u such that the vector field $f(x_p, u)$ is tangent to Γ at each x_p on Γ. Since the curve Γ is parametrized in terms of s, this tangency requirement can be formulated as

$$f(g(s),u) = \lambda \frac{dg(s)}{ds} \quad \wedge \quad \lambda > 0 \quad \wedge \alpha \leq s \leq \beta$$

where $\lambda > 0$ ensures that the state follows Γ in only one direction. The tracking problem is thus equivalent to deciding if the following formula is true or false:

$$\forall_{s,\quad \alpha\leq s\leq\beta} \exists\{u,\lambda\}, \quad u \in U \quad \wedge \quad \lambda > 0 \qquad \left(f(g(s),u) = \lambda \frac{dg(s)}{ds} \right)$$

The above problem becomes a quantifier elimination problem if f or g include some free (design) parameters.

Example: Consider the polynomial control system

$$\begin{aligned} \dot{x_1} &= a x_2 + \frac{1}{2} x_1 x_2 \\ \dot{x_2} &= -x_1 - x_2 + (1 + x_1^2) u \end{aligned}$$

and the curve $\Gamma = \{x \in \mathbb{R}^2 | x_1 = s \wedge x_2 = -s^3 + 1.5 s^2 \wedge 0 \leq s \leq 1\}$. Is it possible to steer the system state along Γ using control signals constrained to $-1 \leq u \leq 1$? Using Mathematica, this can be resolved as follows:

$$\begin{aligned}
&\mathrm{In}[1] := \mathrm{Resolve}[\\
&\qquad \forall_{s,0\leq s\leq 1} \exists_{\{u,\lambda\},(u|\lambda|a)\in \mathrm{R}\,\&\&-1\leq \mathrm{u}\leq 1\&\&\lambda>0} \\
&\qquad \mathrm{a}(\tfrac{3}{2}\mathrm{s}^2 - \mathrm{s}^3) + \tfrac{1}{2}\mathrm{s}(\tfrac{3}{2}\mathrm{s}^2 - \mathrm{s}^3) == \mathrm{s}\lambda\ \&\& \\
&\qquad -\mathrm{s} - \tfrac{3}{2}\mathrm{s}^2 + \mathrm{s}^3 + (1+\mathrm{s}^2)u == (\tfrac{3}{2}\mathrm{s}^2 - \mathrm{s}^3)\lambda \\
&\mathrm{Out}\,[1] = 0 \leq a \leq \\
&\qquad \mathrm{Root}[-5451780 + 890888\#1 + 11716648\#1^2 - 1339948\#1^3 \\
&\qquad -13279197\#1^4 - 2423132\#1^5 + 5128016\#1^6 + 2939328\#1^7\&, 1] \\
&\mathrm{In}[2] := \mathrm{N}[\%] \\
&\mathrm{Out}[2] = 0. \leq \mathrm{a} \leq 1.12769
\end{aligned}$$

The tracking problem has a solution as long as $0 \leq a \leq 1.12769$.

Consider now $a = 1$, *and compute the limits on u such that the above tracking problem has a solution*:

$$\begin{aligned}
\mathrm{In}[1]:= \quad & \mathrm{Resolve}[\\
& \forall_{\mathrm{a},0\le \mathrm{a}\le 1}\exists_{\{\mathrm{u},\lambda\},(\mathrm{u}|\lambda|\mathrm{b})\in\mathbf{R}\,\&\&-\mathrm{b}\le u\le \mathrm{b}\&\&\lambda>0} \\
& \frac{3}{2}\mathrm{s}^2-\mathrm{s}^3+\frac{1}{2}\mathrm{s}(\frac{3}{2}\mathrm{s}^2-\mathrm{s}^3) == \mathrm{s}\lambda\,\&\& \\
& -\mathrm{s}-\frac{3}{2}\mathrm{s}^2+\mathrm{s}^3+(1+\mathrm{s}^2)u == (\frac{3}{2}\mathrm{s}^2-\mathrm{s}^3)\lambda \\
\mathrm{Out}[1] = \quad & \mathrm{b}\ge \mathrm{Root}[964416+5975536\#1-2029956\#1^2-64880931\#1^3 \\
& +7875552\#1^4+61263296\#1^5-7610880\#1^6+262144\#1^7\&,3 \\
\mathrm{In}\,[2]: \;= \quad & \mathrm{N}[\%] \\
& \mathrm{Out}\,[2] = \mathrm{b}\ge 0.98077
\end{aligned}$$

For the tracking problem to have a solution it suffices that $-0.98077 \le u \le 0.98077$.

Multiobjective Feedback Design. Many objectives in linear system design are formulated in terms of frequency-domain inequalities. Let the plant be represented by $G(s, p)$ and the controller by $F(s, q)$. Both G and F are assumed to be scalar-valued rational functions of the Laplace variables s, real plant parameter p, and real controller parameter q. It is now possible to write many robust design problems in a form suitable for quantifier elimination.

Stability. The unit-negative-feedback closed-loop system consisting of plant and controller is asymptotically stable if and only if all zeros of the rational function $1 + G(s, p)F(s, q)$ have strictly negative real part. This can be converted into polynomial inequalities in p and q using the Lineard–Chipart criterion.

Tracking Error. The tracking error at the output of the unit-negative-feedback loop of plant and controller is governed by the so-called sensitivity transfer function

$$S(s) = \frac{1}{1 + F(s, q)G(s, p)}$$

Acceptable levels of the tracking error can be specified by the inequality $|S(iw)| < \alpha_{\mathrm{T}}$, $0 \le w \le w_1$, which can be rewritten as a real algebraic inequality in the variables w, p, q, and α_{T}. Through quantifier elimination it can be verified that for the class of systems $p \in P$ a single stabilizing controller q exists that meets the sensitivity objective.

In essence any linear control design question formulated in frequency domain leads naturally to a quantifier elimination or cylindrical algebraic decomposition problem. More importantly, symbolic manipulation software is up to the task of systematically approaching these control design questions in practical and moderately complex situations.

Related Problems and Other Control Applications. There is a vast array of problems in control that can be posed as quantifier elimination problems or systems of multivariate polynomial inequalities, whose solutions can be described by cylindrical algebraic decomposition. For example, quantifier elimination has been used to investigate stability, stabilizability, and controllability of discrete-time polynomial systems (35,36), stability and stabilizability of switched polynomial systems and unstable zero dynamics via switching (30,37), frequency-domain design (38), and multiobjective robust control design (39). Quantifier elimination has also been used for design of nonlinear control systems for nonlinear aircraft dynamics (40) and for robust nonlinear feedback design (41).

The generality of the quantifier elimination method comes at the cost of very high computational complexity. Quantifier elimination software tools are not a panacea allowing one to approach every control problem by brute force. The challenge in successfully applying quantifier elimination software tools lies in formulating the problems in such a way that their inherent structure is maximally exploited. Most importantly, instead of applying quantifier elimination to a complete problem, there are often simplifications that can be carried out so that only a small problem (with far fewer variables) has to be handled by quantifier elimination. Control designers should start focusing on providing the right heuristics and tools, built on their experience, in order to make general-purpose quantifier elimination tools better attuned to control problems. In parallel, algorithm designers now focus on developing specialized methods as components of a general quantifier elimination package in order to exploit structure and to handle complexity. For instance, in Mathematica (25) there are a modified simplex linear optimization algorithm for linear systems of equations and inequalities with inexact or rational coefficients, linear quantifier elimination for equations and inequalities with exact coefficients according to 27, preprocessing by linear equation solvers, Gröbner bases, special care and simplification for variables appearing at most linearly in the systems, a simplified cylindrical algebraic decomposition algorithm for generic solutions (24), and quantifier elimination by partial cylindrical algebraic decomposition (23) for the general case.

Conclusion. The possibilities offered through symbolic (and numeric) software in order to address control problems are as varied and exciting as the control problems themselves. This article just presents the tip of an iceberg. Dealing with complexity in control design is the main issue. For control algorithm developers there is now more than ever a compelling need to focus on reliable software algorithms that allow nonspecialists to approach a control problem with confidence. Software tools that provide information to the user about the likelihood of success and the difficulties in a particular solution are called for to make this a reality.

BIBLIOGRAPHY

1. F. Cucker L. Blum S. Smale M. Shub *Complexity and Real Computation*, Berlin: Springer-Verlag, 1998.
2. N. Munro (ed.) *Symbolic Methods in Control System Analysis and Design*, Institution of Electrical Engineers, 1999.
3. L. A. Rubel M. F. Singer A differentially algebraic elimination theorem with applications to analog computability in the calculus of variations, *Proc. Amer. Math. Soc.* **94**: 653–658, 1985.
4. P. Lindskog *Methods, algorithms and tools for system identification based on prior knowledge*, PhD thesis 436, Department of Electrical Engineering, Linköping University, Sweden, 1996.
5. D. Nešić *Dead-beat control for polynomial systems*, PhD thesis, RSISE, Australian National University, Canberra, Australia, 1996.
6. J. Little D. Cox D. O'Shea *Ideals, Varieties and Algorithms*, Berlin: Springer-Verlag, 1992.
7. A. Mees *Dynamics of Feedback Systems*, Chichester: Wiley, 1981.
8. K. Forsman Constructive commutative algebra in nonlinear control, PhD thesis, Department of Electrical Engineering, Linköping University, Linköping, Sweden, 1997.
9. D. Nešić A note on observability tests for general polynomial and simple Wiener–Hammerstein systems, *Syst. Control Lett.*, **35**: 219–227, 1998.
10. S. T. Glad An algebraic approach to bang–bang control, *European Control Conference, ECC 95*, Rome, 1995, Vol. 4, pp. 2892–2895.
11. T. Georgiou U. Walther A. Tannenbaum Computational algebraic geometry and switching surfaces in optimal control, *Proc. Conf. Decision Control*, Phoenix, AZ, 1999, pp. 4724–4729.
12. H. Fortell Algebraic approaches to normal forms and zero dynamics, PhD thesis, Department of Electrical Engineering, Linköping University, Linköping, Sweden, 1995.
13. R. Germundsson Symbolic systems, PhD thesis, Department of Electrical Engineering, Linköping University, Linköping, Sweden, 1995.
14. J. Gunnarsson Symbolic methods and tools for discrete event dynamic systems, PhD thesis, Department of Electrical Engineering, Linköping University, Linköping, Sweden, 1997.
15. F. D. Kronewitter III Non-commutative computer algebra, PhD thesis, University of California, San Diego, 2000.

16. M. Fliess S. T. Glad An algebraic approach to linear and nonlinear control, in H. L. Trentelman and J. C. Willems (eds.), *Essays on Control: Perspectives in the Theory and Its Applications*, Boston: Birkhäuser, 1993.
17. J. F. Ritt *Differential Algebra*, New York: American Mathematical Society, 1950.
18. E. R. Kolchin *Differential Algebra and Algebraic Groups*, New York: Academic Press, 1973.
19. R. M. Cohn *Difference Algebra*, Huntington, NY: R. E. Krieger, 1979.
20. A. Seidenberg An elimination theory for differential algebra, *University of California Publications in Mathematics, New Series*, 1956, pp. 31–66.
21. A. Tarski *A Decision Method for Elementary Algebra and Geometry*, 2nd ed. University of California Press, 1948.
22. G. E. Collins Quantifier elimination for real closed fields by cylindrical algebraic decomposition, *Second GI Conf. Automata Theory and Formal Languages, Kaiserslauten, Lecture Notes in Comput. Sci.*, 33 Berlin: Springer-Verlag, 1975, pp. 134–183.
23. G. E. Collins H. Hong Partial cylindrical algebraic decomposition for quantifier elimination, *J. Symbolic Comput.* **12**: 299–328, 1991.
24. S. McCallum An improved projection for cylindrical algebraic decomposition, in B. F. Caviness and J. R. Johnson (eds.), *Quantifier Elimination and Cylindrical Algebraic Decomposition*, Monographs in Symbolic Computation, Berlin: Springer-Verlag, 1998, pp. 242–268.
25. A. Strzebonski Solving algebraic inequalities with Mathematica version 4, *Mathematica J.***7**(4): 525–541, 2000.
26. S. Basu R. Pollack M.-F. Roy On the combinatorial and algebraic complexity of quantifier elimination, *Assoc. Comput. Mach.*, **43**(6): 1002–1045, 1996.
27. R. Loos V. Weispfenning Applying linear quantifier elimination, *Comput. J.* **5**(36): 450–461, 1993.
28. A. Strzebonski An algorithm for systems of strong polynomial inequalities, *Mathematica J.* **4**(4): 74–77, 1994.
29. S. Wolfram *The Mathematica Book*, 4th ed., Champaign, IL: Wolfram Media, Cambridge University Press, 1998.
30. M. Jirstrand Constructive methods for inequality constraints in control, PhD thesis, Department of Electrical Engineering, Linköping University, Linköping, Sweden, May 1998.
31. D. S. Arnon A bibliography of quantifier elimination for real closed fields, *J. Symbolic Comput.* **5**(1–2): 267–274, 1988.
32. G. E. Collins Quantifier elimination by cylindrical algebraic decomposition—twenty years of progress, in B. F. Caviness and J. R. Johnson (eds.), *Quantifier Elimination and Cylindrical Algebraic Decomposition*, Berlin: Springer-Verlag, 1998.
33. B. Anderson N. Bose E. Jury Output feedback stabilization and related problems—solution via decision methods, *IEEE Trans. Autom. Control*, **AC-20**: 53–65, 1975.
34. F. R. Gantmacher *Matrix Theory*, Vol. II, Chelsea: New York, 1960.
35. D. Nešić I. M. Y. Mareels Stabilizability and stability for implicit and explicit polynomial systems: a symbolic computation approach, *Eur. J. Control* **5**: 32–43, 1999.
36. D. Nešić I. M. Y. Mareels Dead-beat controllability of polynomial systems: symbolic computation approaches. *IEEE Trans. Autom. Control* **43**: 162–175, 1998.
37. D. Nešić M. Jirstrand Stabilization of switched polynomial systems, in *IMACS Conf. on Applications of Computer Algebra (ACA'98)*, Prague, 1998.
38. P. Dorato W. Yang C. T. Abdallah Quantifier elimination approach to frequency domain design, in S. Tarbouriech and G. Garcia (eds.), *Control of Uncertain Systems with Bounded Inputs*, Lecture Notes in Control and Information Sciences, 227, Berlin: Springer-Verlag, 1997, pp. 165–172.
39. P. Dorato W. Yang C. Abdallah Robust multi-objective feedback design by quantifier elimination, *J. Symbolic Comput.* **24**(2): 153–159, 1997.
40. M. Jirstrand Nonlinear control system design by quantifier elimination, *J. Symbolic Comput.* **24**(2): 137–152, 1997.
41. P. Dorato D. Famularo C. T. Abdallah W. Yang Robust nonlinear feedback design via quantifier elimination theory, *Int. J. Robust Nonlinear Control*, **9**: 817–822, 1999.

D. NEŠIĆ
I. M. Y. MAREELS
The University of Melbourne
S. T. GLAD
Linköping University
M. JIRSTRAND
MathCore AB