POLYCON - Computer Algebra Software for Polynomial Control Systems

Krister Forsman

Dept. of Electrical Engineering, Linköping University
S-581 83 Linköping, Sweden

email: `krister@isy.liu.se`

1993-02-15

Technical reports from the automatic control group in Linköping are available by anonymous ftp at the address `130.236.24.1` (`joakim.isy.liu.se`)

This report is contained in the compressed ps-file named `/pub/reports/LiTH-ISY-R-1447.ps.Z`

POLYCON - Computer Algebra Software for Polynomial Control Systems

Krister Forsman
Dept. of Electrical Engineering
Linköping University, S-581 83 Linköping, Sweden
email: `krister@isy.liu.se`

1993-02-15

Abstract. This paper describes the features and implementation of the Maple package POLYCON which is intended to assist the control theorist in the analysis of nonlinear dynamical systems, in continuous and discrete time. POLYCON handles systems where all nonlinearities are polynomial or rational functions. It implements functions that are not available at a “usable” level in other programs, to the author’s knowledge. It is supposed to be accessible to non-experts and those that are not familiar with computer algebra, commutative algebra or differential algebra. POLYCON is included in the Maple Share Library and thus available by anonymous ftp.

Keywords: computer algebra, symbolic manipulation, CACSD, polynomial control systems, state space analysis, external behavior, Gröbner bases, commutative algebra, differential algebra

1 Introduction

When discussing how constructive or how algorithmic a theory is, the following levels provide a rough classification:

- abstract** meaning that e.g. existence of some objects is proved, but it is not clear how to construct those objects or if this is even possible.
- effective** algorithms for constructing the objects discussed are given, even though these algorithms may be of small value for practical purposes, due to high complexity
- efficient** efficient algorithms for constructing the objects involved are given; there is a computer program that somebody knows how to run that computes the desired entities
- usable** there is a computer program for computing the desired objects that is not machine dependent and that other people than its author(s) are able to run.

The difference between efficient and effective is of course depending on many circumstances, such as available hardware etc, so it is often very difficult to tell which case is at hand.

If we look at the development of commutative, differential and difference algebra in the theory of nonlinear control systems we may discern approximately the following features:

abstract	Much of the work by M. Fliess [9, 10, 8] some work by E.D. Sontag [31], T. Glad and others
effective/efficient	Most of the work by Glad [22, 23, 26] and S. Diop [5, 6], some of the work by Fliess.

The word abstract does definitely not have a negative meaning in this context. A large part (maybe the majority) of nonlinear control systems research today would probably be classified as abstract with the definitions above.

The POLYCON package, described in this paper, is intended to be one step in the direction from efficient to usable. Its theoretical basis is commutative algebra rather than differential algebra, the reason being that the constructive aspects of commutative algebra are better understood than those of differential algebra. There is a large amount of research going on in the area of constructive commutative algebra both from a mathematical and a computer science point of view, whereas only a limited number of persons are developing the algorithmic aspects of differential algebra, unfortunately.

Maple [3] is probably the most widespread symbolic algebra program today, along with Mathematica. It has been available for more than ten years and its engineering applications abound. Some examples of Maple software for dealing with nonlinear control systems are [4, 32, 34]. The package POLYCON is included in the Maple Share Library and is thus available by anonymous ftp at the addresses 129.132.101.33 (`neptune.inf.ethz.ch`) and 129.97.140.58 (`daisy.waterloo.edu`). It is also possible to get it anonymously from Linköping at the address 130.236.24.1 (`joakim.isy.liu.se`), under the directory `/pub/src/maple/polycon`. The size of the file containing all the Maple source code for POLYCON, including help texts, is 85 kbytes.

POLYCON is a collection of Maple procedures for the analysis of polynomial and rational dynamical systems written either in state space form, i.e. in the form

$$\dot{x}(t) = f(x(t), u(t)), \quad y = h(x, u) \quad (1)$$

or

$$x(t+1) = f(x(t), u(t)), \quad y = h(x, u) \quad (2)$$

where h and all components of f are rational functions in x and u , or in input-output form, i.e.

$$p(y_0, \dots, y_n, u_0, \dots, u_r) = 0 \quad (3)$$

where subindices denote either time-derivatives or time-shifts and p is a polynomial. Most functions are available in a continuous time and a discrete time version; the convention is that the names of continuous time functions end in `c` and those of discrete time functions end in `d`. Most functions handle systems with several inputs.

The package is aimed at questions related to state space realizations, such as

- retrieving external behavior from a state space description, i.e. “conversion” from a state space description ((1) or (2)) to an input-output description (3): `ss2ioc`, `ss2iod`
- realization of input-output equations: `par2ssc`, `io2ssc`, `io2ssd`
- (algebraic) observability, as defined in e.g. [19, 23]: `obsvc`, `obsvd`

- state transformations of systems: `newsysc`, `newsysd`
- finding the transformation between two i/o-equivalent systems: `sstrac`, `sstrad`

There is also a procedure `loclyap` for analysis of local Lyapunov functions.

Note that there are many topics in control theory that are not represented at all in POLYCON, such as controllability, optimal control and feedback. The reasons for this is that there are basically no algorithms general enough for solving these problems in the algebraic framework, to the author's knowledge. One can say that the emphasis is on analysis rather than design, mainly because analysis is simpler. . .

This paper is organized as follows:

- In section 2 we give a user-level description of the most important POLYCON functions.
- Section 3 very briefly discusses some of the mathematical background and the algorithms used, mainly by referring to earlier work.
- In section 4 we show how POLYCON works on a simple example.
- Section 5 contains an outline of the possible extensions of the procedures and the functionality of POLYCON.

2 The Main Functions

The functions available in POLYCON are, in alphabetical order:

<code>auxlieder</code>	<code>io2ssc</code>	<code>io2ssd</code>	<code>lieder</code>	<code>liederlist</code>	<code>liehom</code>	<code>liehomlist</code>
<code>loclyap</code>	<code>newsysc</code>	<code>newsysd</code>	<code>obsvc</code>	<code>obsvd</code>	<code>par2ssc</code>	<code>ss2ioc</code>
<code>ss2iod</code>	<code>sstrac</code>	<code>sstrad</code>	<code>uyder</code>	<code>uyhom</code>	<code>xss2ioc</code>	<code>xss2iod</code>

All functions have detailed help texts. For more information about a particular function, use the help command `?` after having loaded POLYCON, e.g. `?sstrac`

You can also type `?polycon` to obtain general information on POLYCON.

In POLYCON time derivatives and time shifts of dependent variables are represented by subindices, as done by e.g. Ritt [30]. This means that e.g. in continuous time

$$y_0 = y(t), \quad y_1 = \dot{y}(t), \quad y_2 = \ddot{y}(t), \quad \text{etc.} \quad (4)$$

while in discrete time

$$y_0 = y(t), \quad y_1 = y(t+1), \quad y_2 = y(t+2), \quad \text{etc.} \quad (5)$$

This rule holds for input and output variables. Higher derivatives or shifts of state variables do not occur, so state variables are still called $x_1, \dots, x_n$ by default. If two state descriptions are involved the second one has state variables $z_1, \dots, z_n$ by default. In functions where the first derivative or shift of the state variables may occur, e.g. `io2ss` and `newsys`, they are denoted by $dx_1, \dots, dx_n$. In Maple, subscript is represented by concatenation so that e.g. y_2 is written `y2`.

Figure 1: Diagram describing continuous time functions.

Figure 2: Diagram describing discrete time functions.

The largest part of the package is aimed at questions related to rational state realizations of control systems. The diagram in figure 1 summarizes the use of the different procedures in the continuous time case. In the diagram p and φ_i are differential polynomials in u, y , i.e. they involve time derivatives of u and y (but not of x_i).

For discrete time systems the functions in figure 2 are available. Here p and φ_i involve time shifts of u and y (but not of x_i).

The functions `ss2ioc` and `ss2iod` take as input a system in state space form, represented by the polynomial rhs vector field f (an object of type `vector` in Maple) and the output-map h (a polynomial) and return a polynomial in the input, the output and their derivatives representing the input-output behavior of the system.

These two procedures work for many rational systems as well, i.e. systems of the type (1) or (2) where h and all f_i are rational functions of their arguments. However, there are examples where the algorithm fails, see e.g. [13].

The functions `obsvc` and `obsvd` take as input a system in state space form and one of the state variables, x_i say, and return two polynomials in the input, the output and their

derivatives and x_i constituting an observer relation for x_i . Such a relation exists iff the system is algebraically observable, i.e. the input-output equation is of order equal to the state space dimension.

The user can choose if he only wants to accept as an answer a list of rational expressions for x_i in inputs and output (and their derivatives) or if he allows x_i to occur nonlinearly. In the latter case the answer is a list of polynomials φ_i in inputs and output and x_i such that $\varphi_i = 0$ under the system in question.

The functions `io2ssc` and `io2ssd` can be thought of as the inverses of `obsvc` and `obsvd`. The input arguments are a polynomial input-output equation (represented by a polynomial p only) and a list of rational expressions for state variable candidates, i.e. a list S of rational functions in u, y and their derivatives such that the i :th element of S might serve as the i :th state. The function returns a list consisting of the rhs and the output map of the state space equation for the continuous time polynomial SISO system given by the i/o-equation $p = 0$.

The procedure `par2ssc` returns the rhs vector of the state equation for a continuous time rational SISO system given in i/o-form. Its input is a list H of rational functions that parametrize the hypersurface $p = 0$, where p is the i/o-relation.

There is no function `par2ssd`, because it is not clear at the moment whether it is always possible to go from a rational parametrization of a hypersurface to a realization of the corresponding discrete time system.

The procedures `sstrac` and `sstrad` suppose that the two systems involved are i/o-equivalent and then find the state space transformation between them.

The functions `newsysc` and `newsysd` take a system in state space form and a transformation as inputs and returns the system in the new coordinates, defined by the transformation.

For a full description of all POLYCON functions we refer either to the online help texts or to the technical report [15] which is available by anonymous ftp. This report contains exactly the same information as is available online in Maple plus some source code.

3 The Implementation

Let us here very briefly mention something about the mathematics behind POLYCON.

POLYCON heavily relies on elimination theory for commutative rings and in particular Gröbner bases (GB) computations. For an introduction to the theory of GB see e.g. [2, 18, 21, 29]. All major computer algebra programs (Maple, Mathematica, Reduce, Axiom and Macsyma) have GB packages of varying depth and quality. The branch of mathematics dealing with e.g. GB is currently very active and a lot of progress is still made in this research area.

Gröbner bases can be seen as the generalization of Gaussian elimination to systems of polynomial equations. A GB is a generating set for a polynomial ideal having some appealing properties. Most important in this context is the following elimination property, which we state without giving any details, since that would lead much too far:

Let $\mathfrak{a}$ be an ideal in $k[X_1, \dots, X_n]$ and partition $X_1, \dots, X_n$ into two disjoint sets S and T . If $\mathbf{G}$ is a GB for $\mathfrak{a}$ w.r.t. a pure lexicographic term ordering ranking S lower than T then $\langle \mathbf{G} \cap k[S] \rangle = \mathfrak{a} \cap k[S]$.

This statement is proved in e.g. [21] or [2].

Many functions in POLYCON use the Boege-Gebauer-Kredel (BGK) algorithm [1] for eliminating variables since this is in general more efficient than computing a GB w.r.t. a pure lexicographic term ordering. The BGK algorithm first computes a GB which does not perform elimination, but which is comparatively cheap to compute. Then linear algebra techniques are used to find the elimination ideal; see [1, 17, 20] for more details. The BGK algorithm is implemented as `finduni` in Maple. Most of the time, the BGK algorithm cannot be applied directly to the ideals considered in POLYCON. Instead the ideals must first be localized so that they become zero-dimensional. This simply means that some of the variables are considered as parameters, i.e. the field of coefficients is extended to contain rational functions in some of the variables. Theoretical investigations of the localization procedure are presented in [16, 17].

The theoretical background, with proofs for the correctness of the algorithms in POLYCON, is given in chapters 5 and 6 of the author's PhD-thesis [12], where also some source code for primitive versions of the functions is given. Related theoretical work can be found in [5, 6, 19, 22, 23, 28, 33].

The theory behind `obsvc`, `obsvd`, `io2ssc` and `io2ssd` is also described in [16] and [19], that of `loclyap` in [11] and that of `par2ssc` in [14].

4 An Example

Below is an example showing how some computations on a simple control system can be performed using POLYCON. The example is taken from [27], p. 147.

The following equations roughly describe the vertical motion of a hot air balloon:

$$\begin{aligned} \dot{x}_1 &= Ku - \frac{1}{\tau}(x_1 - T) \\ \dot{x}_2 &= x_3 \\ \dot{x}_3 &= g \left(\frac{W_a}{W} \left(1 - \frac{T}{x_1} \right) - 1 - \frac{\mu}{W} x_3 \right) \end{aligned} \tag{6}$$

Here x_2 is the altitude above sea level, $x_3 = \frac{d}{dt}x_2$ and x_1 is the air temperature (in Kelvin) inside the balloon. The constants τ, K, W_a, μ are related to the thermal and physical characteristics of the balloon, W is the weight of the balloon including payload and air (in kg), T is the outdoor temperature, and g is the acceleration of gravity. The input u of the system is the heat transfer rate supplied to the balloon by the heater. Let us consider the altitude as the output of the system: $y = x_2$. If we put $g = 10, W = 200, W_a = 400$ (I can't guarantee that these are realistic values...) we can use the procedure `ss2ioc` in POLYCON to obtain a differential equation relating the input and the output; this equation is

$$\begin{aligned} 8000 T \tau y_3 - 400(T + K \tau u) y_2^2 - 40 \left(\mu (K \tau u + T) y_1 + 200 K \tau u + 10 \mu T \tau \right) y_2 \\ - \mu^2 (K \tau u + T) y_1^2 + 400 \mu K \tau u y_1 + 40000(T - K \tau u) = 0 \end{aligned} \tag{7}$$

where $y_1 = \frac{d}{dt}y(t)$, $y_2 = \frac{d^2}{dt^2}y(t)$ etc.

Now suppose that we have accurate measurements of the altitude x_2 (so that we can compute the velocity and acceleration with acceptable precision) and want to estimate the temperature x_1 in the balloon. We can then use the function `obsvc` to find an expression for x_1 in the output and its derivatives. It turns out that

$$x_1 = \frac{400 T}{200 - 20 y_2 - \mu y_1} \quad (8)$$

(In fact, the observer relation is quite easy to derive by hand, in this case.)

Below is a Maple session for doing the computations above:

```
> with(linalg): read('polycon.m'):
> f:=vector([- (1/tau)*(x1-T)+K*u, x3, g*((Wa/W)*(1-(T/x1))-1-(mu/W)*x3)]):
> g:=10: Wa:=400: W:=200:
> p:=ss2ioc(f,x2):
> collect(p,[y3,y2,y1]);

      2
8000 T tau y3 + (- 400 T - 400 K u0 tau) y2
+ ((- 40 mu K u0 tau - 40 mu T) y1 + 8000 K u0 tau + 400 mu T tau) y2
+ (- mu T - mu K u0 tau) y1 + 400 mu K u0 tau y1 + 40000 T
- 40000 K u0 tau

> obsvc(f,x2,x1);

      T
[- 400 -----]
      20 y2 - 200 + y1 mu
```

5 Future Extensions

The current plans for future versions of POLYCON include:

- **Full support for rational functions.** As mentioned in section 2 some functions fail on some rational systems. It is probably not difficult to amend this using the Rabinovich trick [7, 13].
- **Identifiability.** It is possible to test global identifiability of parameters in polynomial systems in state space form using GB. Some references, treating an even more general problem, are [24, 25, 35]. It seems straightforward to implement such functions in POLYCON.
- **Implicit equations.** Systems described by generalized state space equations, i.e. differential equations that are nonlinear in the derivatives can in some cases be handled within the GB framework.
- **Inequations.** As pointed out in e.g. [5] and [33] it is sometimes necessary to add inequations, i.e. expressions of the type $q(u, y) \neq 0$, to the input-output relation in order for it to have the same solutions as the state equation. This is an important aspect of the `ss2io`-problem which is solved in the cited work, but still unclear in the present framework.

- **Manifolds.** Most of the POLYCON functions are possible to extend to systems having for state space a differentiable manifold that is also an algebraic variety.
- **Time-varying systems.**

As mentioned, there are many important parts of nonlinear control theory that are not covered by POLYCON, such as controllability and feedback. As the theoretical research continues, algorithms for dealing with these problems will hopefully emerge and be implemented as far as possible.

Acknowledgement

This work was financially supported by the Swedish Council for Technical Research (TFR).

References

- [1] W. Boege, R. Gebauer, and H. Kredel. Some examples for solving systems of algebraic equations by calculating Gröbner bases. *J. Symbolic Computation*, 1:83–98, 1986.
- [2] B. Buchberger. Gröbner bases: An algorithmic method in polynomial ideal theory. In N.K. Bose, editor, *Multidimensional Systems Theory*, pages 184–232. Dordrecht Reidel, 1985.
- [3] B. Char, K.O. Geddes, G.H. Gonnet, B.L. Leong, M.B. Monagan, and S.M. Watt. *Maple V Language Reference Manual*. Springer, 1991.
- [4] B. de Jager. Symbolic calculation of zero dynamics for nonlinear control systems. In S.M. Watt, editor, *Proc. ISSAC'91*, pages 321–322, Bonn, Germany, July 1991. ACM Press.
- [5] S. Diop. Elimination in control theory. *Math. Control Signals Systems*, 4(1):17–32, 1991.
- [6] S. Diop and M. Fliess. On nonlinear observability. In *Proc. First European Control Conf.*, volume 1, pages 152–157, Grenoble, France, July 1991. Hermès.
- [7] A. Ferro and G. Gallo. Gröbner bases, Ritt's algorithm and decision procedures for algebraic theories. In L. Huguet and A. Poli, editors, *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, volume 356 of *LNCS*, pages 230–237. Springer, 1989. Proc. AAECC-5, Menorca.
- [8] M. Fliess. Some remarks on nonlinear invertibility and dynamic state-feedback. In C. Byrnes and A. Lindquist, editors, *Theory and Applications of Nonlinear Control Systems*, pages 115–121. North Holland, 1986.
- [9] M. Fliess. Automatique et corps différentiels. *Forum Mathematicum*, 1:227–238, 1989.
- [10] M. Fliess. Automatique en temps discret et algèbre aux différences. *Forum Mathematicum*, 2:213–232, 1990.

- [11] K. Forsman. Applications of Gröbner bases to nonlinear systems. In *Proc. First European Control Conf.*, volume 1, pages 164–169, Grenoble, France, July 1991. Hermès.
- [12] K. Forsman. *Constructive Commutative Algebra in Nonlinear Control Theory*. PhD thesis, Dept. of Electrical Engineering, Linköping University, S-581 83 Linköping, Sweden, 1991.
- [13] K. Forsman. Elementary aspects of constructive commutative algebra. Technical Report LiTH-ISY-I-1395, Dept. of Electrical Engineering, Linköping University, S-581 83 Linköping, Sweden, September 1992. Available by anonymous ftp 130.236.24.1.
- [14] K. Forsman. On rational state space realizations. In M. Fliess, editor, *Proc. NOLCOS'92*, pages 197–202, Bordeaux, 1992. IFAC.
- [15] K. Forsman. POLYCON - a Maple package for polynomial and rational control systems. Technical Report LiTH-ISY-I-1386, Dept. of Electrical Engineering, Linköping University, S-581 83 Linköping, Sweden, August 1992. Included in the Maple share library. Also available by anonymous ftp 130.236.24.1.
- [16] K. Forsman. Some generic results on algebraic observability and connections with realization theory. Technical Report LiTH-ISY-I-1403, Dept. of Electrical Engineering, Linköping University, S-581 83 Linköping, Sweden, September 1992. Available by anonymous ftp 130.236.24.1. Accepted for ECC '93.
- [17] K. Forsman. Localization and base change techniques in computational algebra. Technical Report LiTH-ISY-R-1445, Dept. of Electrical Engineering, Linköping University, S-581 83 Linköping, Sweden, February 1993. Available by anonymous ftp 130.236.24.1.
- [18] K.O. Geddes, S.R. Czapor, and G. Labahn. *Algorithms for Computer Algebra*. Kluwer Academic Publishers, 1992.
- [19] R. Germundsson and K. Forsman. A constructive approach to algebraic observability. In *Proc. 30:th CDC*, volume 1, pages 451–452, Brighton, UK, 1991. IEEE.
- [20] P. Gianni and T. Mora. Algebraic solution of systems of polynomial equations using Gröbner bases. In L. Huguet and A. Poli, editors, *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, volume 356 of *LNCS*, pages 247–257. Springer, 1989. Proc. AAECC-5, Menorca.
- [21] P. Gianni, B. Trager, and G. Zacharias. Gröbner bases and primary decomposition of polynomial ideals. In L. Robbiano, editor, *Computational Aspects of Commutative Algebra*, pages 15–33. Academic Press, 1989. From J. Symb. Comp. Vol. 6, nr. 2-3.
- [22] S.T. Glad. Nonlinear state space and input output descriptions using differential polynomials. In J. Descusse, M. Fliess, A. Isidori, and D. Leborgne, editors, *New Trends in Nonlinear Control Theory*, pages 182–189. Springer, 1988.
- [23] S.T. Glad. Differential algebraic modelling of nonlinear systems. In M.A. Kaashoek, J.H. van Schuppen, and A.C.M. Ran, editors, *Realization and Modelling in System Theory. Proc. Intl. Symp. MTNS-89*, volume I, pages 97–105, Amsterdam, 1990. Birkhäuser.
- [24] S.T. Glad and L. Ljung. Model structure identifiability and persistence of excitation. In *Proc. 29th CDC*, volume 6, pages 3236–3240, Honolulu, Hawaii, 1990. IEEE.

- [25] S.T. Glad and L. Ljung. Parametrization of nonlinear model structures as linear regressions. In *Proc. 11th IFAC World Congress*, volume 6, pages 67–71, Tallinn, USSR, 1990. IFAC.
- [26] T. Glad. Implementing Ritt’s algorithm of differential algebra. In M. Fliess, editor, *Proc. NOLCOS’92*, pages 610–614, Bordeaux, 1992. IFAC.
- [27] N.H. McClamroch. *State Models of Dynamic Systems - A Case Study Approach*. Springer, 1980.
- [28] F. Ollivier. *Le problème de l’identifiabilité structurelle globale: approche théorique, méthodes effectives et bornes de complexité*. PhD thesis, École Polytechnique, 1990.
- [29] F. Pauer and M. Pfeifhofer. The theory of Gröbner bases. *L’Enseignement Mathématique*, 34:215–232, 1988.
- [30] J.F. Ritt. *Differential Algebra*. Dover, 1950.
- [31] E.D. Sontag. *Polynomial Response Maps*, volume 13 of *Lecture Notes in Control and Information Sciences*. Springer, 1979.
- [32] T. Svensson. *Mathematical Tools and Software for Analysis and Design of Nonlinear Control Systems*. PhD thesis, Dept. of Electrical Engineering, Linköping University, S-581 83 Linköping, Sweden, 1992.
- [33] A.J. van der Schaft. Representing a nonlinear state space system as a set of higher-order differential equations in the inputs and outputs. *Systems & Control Letters*, 12(2):151–160, February 1989.
- [34] H. van Essen. Symbols speak louder than numbers: Analysis and design of nonlinear control systems with the symbolic computation system Maple. Master’s thesis, Dept. of Mechanical Engineering, Eindhoven University of Technology, Eindhoven, The Netherlands, June 1992. Number WFW 92.061.
- [35] E. Walter, Y. Lecourtier, and A. Raksanyi. Test of structural properties of state space models through algebraic computation. In *Proc. 9th IFAC World Congress*, volume 8, pages 250–255, Budapest, 1984. IFAC.