Transactions on Engineering Sciences vol 22 © 1999 WIT Press, www.witpress.com, ISSN 1743-3533

Symbolic computation for the analysis and synthesis of nonlinear control systems

A. Kugi, K. Schlacher, R. Novak
Department of Automatic Control, Johannes Kepler University, A-4040 Linz Auhof, Austria
Email: andi@regpro.mechatronik.uni-linz.ac.at

Abstract

Symbolic computation has proved to be an effective tool for handling large mathematical expressions which are too complex for calculation by hand. Especially in the field of nonlinear control the use of computer algebra programs is unavoidable. Since the memory requirements are overpolynomial with respect to the problem size, it is important to develop algorithms that optimally fit the symbolic treatment of these problems. Hence, the developers of algorithms in computer algebra programs must free themselves from the well known numerical solutions and they have to take care of the special features of computer algebra. In this sense the present contribution presents some selected algorithms for the analysis and design of a special class of nonlinear systems, the so called affine input systems.

1 Introduction

During the last years there have been some significant advances in the area of nonlinear control system design. The reason for the research effort on nonlinear control is the more demanding performance required in practical applications and, however, most of the physical systems are nonlinear in nature. Moreover, only according to the increasing availability of low cost digital signal processors and the increasing power of computer algebra programs for the symbolic computation the practical use of these nonlinear control strategies is made possible.

Throughout the subsequent considerations we study nonlinear control systems with a state space representation of the form

$$\frac{\mathrm{d}}{\mathrm{d}t}x = f(x) + \sum_{i=1}^{m} g_i(x)\, u_i \quad , \quad y = h(x) \tag{1}$$

where $x \in R^n$ denotes the state, $u \in R^m$ the plant input and $y \in R^l$ the plant output. We assume that f, g_1, ..., g_m and h are smooth in their arguments, i.e. all entries of eqn (1) are real–valued functions of x with continuous partial derivatives of any order. Since the control input u only appears affine on the right hand side of eqn (1), this type of nonlinear systems is known as the so called **AI**–(**A**ffine **I**nput) systems, see e.g., Isidori[5], Nijmeijer & Van der Schaft[9]. AI–systems have the special properties that the structure of an AI–system is preserved under the action of certain nonlinear changes of coordinates and that the serial connection, the parallel connection and the feedback of AI–systems again lead to a nonlinear system with AI–structure. Thus, taking a diffeomorphism (smooth and smooth invertible mapping) of the form $z = t(x)$, we obtain the system eqn (1) in the new coordinates z

$$\frac{\mathrm{d}}{\mathrm{d}t}z = \left(\frac{\partial}{\partial x}tf\right)\left(t^{-1}(z)\right) + \sum_{i=1}^{m}\left(\frac{\partial}{\partial x}tg_i\right)\left(t^{-1}(z)\right)u_i \quad , \quad y = h\left(t^{-1}(z)\right) \tag{2}$$

and this is in fact an AI–system.

The paper is organized as follows. In Section 2 the necessary differential geometric fundamentals for AI–systems are briefly summarized. Section 3 is devoted to the special features that must be taken into consideration if algorithms are implemented in a computer algebra program like MapleV. Here, the major problem with computer algebra is the overpolynomial increase of calculation time and memory (see e.g., Davenport et al.[3]) and it is a matter of fact that the investigations of nonlinear systems require a great amount of symbolic operations. Therefore, it is near at hand that de Jager[4] asked the question, whether symbolic computation is a viable approach for nonlinear systems or not. We definitely say yes to this question but it must be taken into account that the algorithms presented in literature are not suitable for computer algebra implementation in a straight forward manner. As it is shown by Kugi et al.[8] for the symbolic calculation of large electric circuits, computer algebra implementation requires new algorithms that are not based on the corresponding numerical ones. In Section 3 some of the ideas for the efficient implementation of the algorithms for nonlinear systems will be discussed. The interested reader is also referred to other contributions to the use of symbolic computation in nonlinear control, like e.g., the research works of Rothfuß et al.[10] or de Jager et al.[4] Section 4 presents some selected algorithms and a nontrivial application example is investigated in Section 5. The last section, Section 6, contains some conclusions.

2 Differential Geometric Fundamentals

The smooth mappings f, g_1, ..., g_m of eqn (1) assign each point x element of a manifold $\mathcal{M}$ a tangent vector in the tangent space $\mathcal{T}_x\mathcal{M}$ which is why they are mostly referred to as smooth vector fields of $\mathcal{T}_x\mathcal{M}$ or with regard to the MapleV implementation they are also called contravariant tensors of first order. Subsequently, we also need the dual objects of the vector fields, the so called covector fields, 1–forms or in MapleV covariant tensors of first order. These elements are defined in the cotangent space $\mathcal{T}_x^*\mathcal{M}$ (dual space of $\mathcal{T}_x\mathcal{M}$) and they form the space of all linear functions from $\mathcal{T}_x\mathcal{M}$ to R (see e.g., Boothby[1]). A covector or a 1–form of special interest is the exact differential $\mathrm{d}h(x) = \left(\frac{\partial}{\partial x}h\right)(x)$ of a real–valued scalar function $h(x)$.

2.1 Lie Derivatives

For the algorithms three different Lie derivatives are of importance.
(A) The Lie derivative $(L_f h)(x)$ of a real–valued scalar function $h(x)$ along the integral curve of a vector field $f(x)$ is defined as

$$(L_f h)(x) = \left(\frac{\partial}{\partial x} h f\right)(x) \ . \tag{3}$$

A repeated use of this operation leads to the recursion

$$\left(L_f^k h\right)(x) = \left(\frac{\partial}{\partial x} L_f^{k-1} h f\right)(x) \quad \text{with} \quad \left(L_f^0 h\right)(x) = h(x) \ . \tag{4}$$

(B) The second type of operation, the so called Lie–bracket, involves two vector fields f and v and it constructs a new smooth vector field of the form

$$(L_f v)(x) = [f, v](x) = \left(\frac{\partial}{\partial x} v f - \frac{\partial}{\partial x} f v\right)(x) \ . \tag{5}$$

The Lie–bracket is skew–commutative, bilinear and it satisfies the Jacobi identity (see e.g., Boothby[1], Isidori[5]). Here, also a recursive version can be obtained, where for the simplification of notation a new symbol is introduced

$$\left(\mathrm{ad}_f^k v\right)(x) = \left[f, \mathrm{ad}_f^{k-1} v\right](x) \quad \text{with} \quad \left(\mathrm{ad}_f^0 v\right)(x) = v(x) \ . \tag{6}$$

(C) The third operation is the Lie derivative $(L_f v^*)(x)$ of a covector field v^* along the integral curve of the vector field f. For the sake of convenience covector fields are represented as row vectors and hence $(L_f v^*)(x)$ is defined as

$$(L_f v^*)(x) = f^T \left(\frac{\partial}{\partial x} v^*\right)^T + v^* \frac{\partial}{\partial x} f \ . \tag{7}$$

2.2 Distribution and Codistribution

The smooth vector fields v_1, v_2, ..., v_k spanning at a fixed point x of the manifold $\mathcal{M}$ a vector space

$$\Delta(x) = \operatorname{span}\{v_1(x), v_2(x), \ldots, v_k(x)\}, \tag{8}$$

which is clearly a subspace of $\mathcal{T}_x\mathcal{M}$, is called smooth distribution with the dimension $\dim(\Delta(x))$. In the same manner we may also define a smooth codistribution which is given as a subspace of the cotangent space $\mathcal{T}_x^*\mathcal{M}$ spanned by the smooth covectors v_1^*, v_2^*, ..., v_k^*

$$\Delta^*(x) = \operatorname{span}\{v_1^*(x), v_2^*(x), \ldots, v_k^*(x)\} \ . \tag{9}$$

A special codistribution is the annihilator $\Delta^{\perp}(x)$ of a certain distribution $\Delta(x)$ which consists of covectors annihilating all vectors in $\Delta(x)$

$$\Delta^{\perp}(x) = \operatorname{span}\{v^* \in \mathcal{T}_x^*\mathcal{M} \,|\, v^*(v)(x) = 0 \text{ for all } v(x) \in \Delta(x)\}. \tag{10}$$

2.3 Involutive and f-invariant Distribution

A distribution $\Delta(x) = \operatorname{span}\{v_1(x), v_2(x), \ldots, v_k(x)\}$ is said to be involutive, iff the Lie bracket of any pair $[v_i, v_j](x)$ belongs to $\Delta(x)$. Now, the involutivity of a distribution $\Delta(x)$ on the manifold $\mathcal{M}$ is directly connected with the existence of an integral manifold of $\Delta(x)$ by the famous Frobenius Theorem (see e.g., Boothby[1], Isidori[5], Nijmeijer & Van der Schaft[9])

Theorem 1. *A regular distribution $\Delta(x)$ on $\mathcal{M}$ is completely locally integrable, i.e. the annihilator $\Delta^{\perp}(x)$ is spanned by exact differentials, if and only if $\Delta(x)$ is involutive.*

A distribution, which is the smallest involutive distribution containing $\Delta(x)$, is called the involutive closure of $\Delta(x)$ and is denoted by $\operatorname{inv}(\Delta(x))$.
A distribution $\Delta(x) = \operatorname{span}\{v_1(x), v_2(x), \ldots, v_k(x)\}$ is invariant under a vector field $f(x)$, or short $\Delta(x)$ is f–invariant, iff the Lie brackets $[v_i, f](x)$, $i = 1, \ldots, k$ belong to $\Delta(x)$. A similar definition can be given for a codistribution, that is, a codistribution $\Delta^*(x) = \operatorname{span}\{v_1^*(x), v_2^*(x), \ldots, v_k^*(x)\}$ is invariant under a vector field $f(x)$, if $(L_f v_i^*)(x) \in \Delta^*(x)$ for all $i = 1, \ldots, k$. The f–invariance of a smooth distribution $\Delta(x)$ implies the f–invariance of the annihilator $\Delta^{\perp}(x)$ and vice versa.

3 Features of the Computer Algebra Implementation

Most of the algorithms for the analysis and design of nonlinear control systems are based on the recursive setting up of distributions and codistributions using the various Lie derivatives and the symbolic calculation of their dimensions at a generic point x.

3.1 Check of the Linear Dependence of Vector Fields

Here, one of the most time consuming parts is the check of the linear dependence of some vector or covector fields. If this task is performed by means of the MapleV function `linalg[rank]` then with larger mathematical models one immediately gets problems with the calculation time and memory. This is why we implemented a different strategy which optimally fits the recursive nature of the algorithms. From theory we know that for vectors $v_1, \ldots, v_k \in \mathcal{V}$ and 1–forms $v_1^*, \ldots, v_k^* \in \mathcal{V}^*$ an alternating product, the so called wedge product or exterior product, is defined in the form $v_1 \wedge v_2 \wedge \ldots \wedge v_k \in \wedge^k(\mathcal{V})$ or $v_1^* \wedge v_2^* \wedge \ldots \wedge v_k^* \in \wedge^k(\mathcal{V}^*)$ and the products are called k–vectors or k–forms, respectively. The wedge product is skew–commutative, bilinear and distributive (see e.g., Burke[2], von Westenholz[11]). Now, the vectors $v_1, \ldots, v_k \in \mathcal{V}$ are linearly dependent if and only if $v_1 \wedge v_2 \wedge \ldots \wedge v_k = 0$. If $\mathcal{V}$ is an n–dimensional vector space and the nonzero element $vol \in \wedge^n(\mathcal{V}^*)$ is a volume form then this condition is equivalent to $v_1 \rfloor (v_2 \rfloor (\ldots \rfloor (v_k \rfloor vol))) = 0$, where $v \rfloor \omega$ is the interior product of a form ω by a vector field v. This operation $\rfloor$ is also referred to as contraction of ω by v and it maps the k–form ω into a $(k-1)$–form (see e.g., von Westenholz[11]).

3.2 Simplification of Large Symbolic Expressions

In order to make the decision whether a symbolic expression is zero or not, it is absolutely necessary to simplify this expression. However, the simplification of large symbolic expressions may arise great problems with the memory needed and hence often defines the limits of the usefulness of the algorithms. The result of many investigations is that a "cure–all simplifier" does not exist, on the contrary, every problem needs its own specific simplification. Therefore, the algorithms are implemented in such a way that they all use a predefined simplifier which can be easily adjusted to each problem. This predefined simplifier consists of four parts, namely (A) simplification based on side–equations (e.g., $\{\sin(x)^2 = 1 - \cos(x)^2\}$), (B) simplification based on assumptions (e.g., $\{x_1 = \pi,\ x_2 = 0\}$), (C) simplification based on the MapleV command `simplify` (e.g. `simplify(expr, trig)`) and (D) simplification based on other MapleV procedures (e.g., `expand(expr)`).

3.3 Avoidance of Redundancy

The computation time can be significantly decreased if redundant operations are avoided as it is guaranteed by the above mentioned check of the linear dependence of vector fields. For example, if we know that the vectors $v_1, \ldots, v_k$ are linearly independent, hence $w = v_1 \rfloor (v_2 \rfloor (\ldots \rfloor (v_k \rfloor vol))) \neq 0$, and we want to determine if v_{k+1} is linearly independent of $v_1, \ldots, v_k$, we just have to check the condition $v_{k+1} \rfloor w \neq 0$. In contrast to the MapleV `linalg[rank]` command, where the rank $(\text{span}\{v_1, \ldots, v_{k+1}\})$ is checked,

in our case the knowledge of the linear independency of $v_1, \ldots, v_k$ is explicitly used, since this information is contained in the expression w. In this manner many redundant operations are avoided in the recursive algorithms. Additionally, all algorithms take advantage of the skew–commutativity of the Lie–bracket operation and the wedge product.

4 Some Selected Algorithms

4.1 Strong Accessibility (Reachability)

Subsequently, $\varphi(x_0, u, T)$ denotes the solution of eqn (1) for the initial condition $x(0) = x_0$ and plant inputs $u(t)$ with $0 \leq t \leq T$. Let $\mathcal{R}^{\mathcal{U}}(x_0, T)$ be the set of all reachable points from x_0 at time $T > 0$, where the trajectories $\varphi(x_0, u, T)$ remain in the neighborhood $\mathcal{U}$ of x_0.

Definition 1. *An AI–system eqn (1) is said to be (locally) strongly accessible from x_0, if for any neighborhood $\mathcal{U}$ of x_0, the set $\mathcal{R}^{\mathcal{U}}(x_0, T)$ contains a non–empty set for any $T > 0$ sufficiently small (see Nijmeijer & Van der Schaft[9]).*

The following algorithm constructs the so called reachability distribution $\Delta_R(x)$, where $\dim(\Delta_R(x))$ corresponds to the dimension of the strongly accessible (reachable) part of the AI–system eqn (1).

Algorithm: Calculation of the reachability distribution

$vol := \mathrm{d}x_1 \wedge \mathrm{d}x_2 \wedge \ldots \wedge \mathrm{d}x_n$;
$vol := g_m \rfloor(\ldots \rfloor(g_2 \rfloor(g_1 \rfloor vol)))$;
$\Delta_0 = \mathrm{span}\{g_1, g_2, \ldots, g_m\}$;
Precondition: all g_i's are linearly independent
$new := \{u \mid u = [g_i, g_j] \cup u = [f, g_j]$, if u is linearly independent of all elements of Δ_0 and of all constructed u's so far, $i, j = 1, \ldots, \dim(\Delta_0)$, $i < j\}$;
$vol := u_i \rfloor vol$, $i = 1, \ldots, \dim(new)$;
$m := \dim(\Delta_0)$; $n := \dim(new)$;
$\Delta_R = \mathrm{span}\{g_1, g_2, \ldots, g_m\} \cup new$;
while $n > 0$ do
    $\Delta_R = \mathrm{span}\{v_1, v_2, \ldots, v_p\}$;
    $new := \{u \mid u = [v_i, v_j] \cup u = [f, v_j]$, if u is linearly independent of all elements of Δ_R and of all constructed u's so far, $i = 1, \ldots, \dim(\Delta_R) - 1$, $j = m + 1, \ldots, \dim(\Delta_R)$, $i < j\}$;
    $vol := u_i \rfloor vol$, $i = 1, \ldots, \dim(new)$;
    $m := \dim(\Delta_R)$; $n := \dim(new)$;
    $\Delta_R := \Delta_R \cup new$;
end;

A similar algorithm can be found for the determination of the dimension of the observable part of the AI–system eqn (1).

4.2 Input/Output and Input/State Exact Linearization

The idea of exact linearization is to transform the nonlinear AI–system eqn (1) by means of a nonlinear state feedback and a change of coordinates into a linear system.

Definition 2. *The AI–system eqn (1) with $m = l$ is said to have a (vector) relative degree $r = [r_1, \ldots, r_m]$ at a point x_0, if the following conditions*

(1) $\left(L_{g_j} L_f^k h_i\right)(x) = 0 \quad for \quad 1 \le j \le m, \quad 1 \le i \le m \quad and \quad k \le r_i - 2.$

(2) $\operatorname{rank}\left(\hat{g}\left(x_0\right)\right) = m \quad with \quad \hat{g}_{ij} = L_{g_j} L_f^{r_i - 1} h_i$

are satisfied for all x in a neighborhood $\mathcal{U}$ of x_0. The $(m \times m)$–matrix $\hat{g}$ is often called the decoupling matrix and for the relative degree r the relation $r = \sum r_i \le n$ holds (see e.g., Isidori[5], Nijmeijer & Van der Schaft[9]).

Consider the AI–system eqn (1) with the output functions $h_1(x), \ldots, h_m(x)$ and the corresponding relative degree $r = [r_1, \ldots, r_m]$. Using the diffeomorphism $z^T = \left[h_1, \ldots, L_f^{r_1 - 1} h_1, \ldots, h_m, \ldots, L_f^{r_m - 1} h_m, t_{d+1}, \ldots, t_n\right]$, we obtain the system eqn (1) in the new coordinates z

$$\begin{array}{rcl}
\dot{z}_{i,1} & = & z_{i,2} \\
\vdots & \vdots & \vdots \\
\dot{z}_{i,r_i - 1} & = & z_{i,r_i} \\
\dot{z}_{i,r_i} & = & \left(L_f^{r_i} h_i\right)\left(t^{-1}(z)\right) + \sum_{j=1}^{m} \left(L_{g_j} L_f^{r_i - 1} h_i\right)\left(t^{-1}(z)\right) u_j \\
\dot{z}_2 & = & f_2(z) + \sum_{j=1}^{m} g_{2,j}(z)\, u_j
\end{array} \tag{11}$$

for $i = 1, \ldots, m$ and $\dim(z_2) = n - d$. By means of the nonlinear state feedback

$$u = \hat{g}^{-1}\left(t^{-1}(z)\right)\left(\tilde{u} - \left[L_f^{r_1} h_1\left(t^{-1}(z)\right), \ldots, L_f^{r_m} h_m\left(t^{-1}(z)\right)\right]^T\right) \tag{12}$$

the system eqn (11) results in

$$\begin{array}{rcl}
\dot{z}_1 & = & A z_1 + B \tilde{u} \\
\dot{z}_2 & = & f_2(z_1, z_2) + g_2(z_1, z_2)\, \tilde{u} \\
y & = & C z_1 ,
\end{array} \tag{13}$$

where the linear subsystem z_1 is in Brunovsky canonical form with the new input $\tilde{u}$. Moreover, if the distribution span $\{g_1(x), \ldots, g_m(x)\}$ is involutive, it is always possible to choose $t_{d+1}, \ldots, t_n$ in such a way, that $\hat{g}_2(z_1, z_2) = 0$. At this point it is worth to mention that this procedure only leads to a stable closed loop system, if the unobservable subsystem z_2, also called zero dynamics, is stable. Therefore, it is quite natural to look for output functions $h_1(x), \ldots, h_m(x)$ such that the dimension of the zero dynamics becomes minimal or in the ideal case even zero.

Theorem 2. *The AI–system eqn (1) can be transformed to Brunovsky canonical form eqn (13) with* $\dim(z_2) = 0$ *in a neighborhood* $\mathcal{U}$ *of* x_0*. iff for the distribution* $\Delta_i(x) = \text{span}\left\{\text{ad}_f^k g_j,\ 0 \le k \le i, 1 \le j \le m\right\}$ *the following conditions*

(1) $\dim(\Delta_0(x_0)) = m$.

(2) $\dim(\Delta_i(x_0))$ *is constant for* $i = 0, \ldots, n-1$.

(3) $\dim(\Delta_{n-1}(x_0)) = n$ *and*

(4) $\Delta_i(x)$ *is involutive for* $i = 0, \ldots, n-2$

hold for all $x \in \mathcal{U}$*. Then the AI–system eqn (1) is said to be input/state linearizable. Equivalently, iff the conditions* (1) – (4) *are true, then* m *real–valued scalar output functions* $h_i(x)$ *can be found with relative degree* $r = [r_1, \ldots, r_m]$ *and* $\sum r_i = n$ *(see e.g., Isidori*[5]*, Nijmeijer & Van der Schaft*[9]*).*

Algorithm: Check if the system is input/state linearizable

```
Check dim(Δ0) ≟ m -> condition (1)
vol := dx1 ∧ dx2 ∧ ... ∧ dxn;
vol := gm ⌋( ... ⌋(g2 ⌋(g1 ⌋vol)));
Δ0 = span{g1, g2, ..., gm};
Check of involutivity of Δ0 -> condition (4)
if [vi, vj] ⌋vol ≠ 0, i,j = 1, ..., m, i < j then exit;
new := {u | u = [f, gi], if u is linearly independent of all elements of Δ0
     and of all constructed u's so far, i = 1, ..., m};
vol := ui ⌋vol, i = 1, ..., dim(new);
n := dim(new);
i := 1;
while Δi is involutive do
     Δi := Δi-1 ∪ new;
     Δi = span{v1, v2, ..., vp};
     Check of involutivity of Δi
     for j from 1 to m do
          for k from m + 1 to dim(Δi) do
               if [vj, vk] ∉ Δi then exit;
          end
     end
     for j from m + 1 to dim(Δi) − 1 do
          for k from j + 1 to dim(Δi) do
               if [vj, vk] ∉ Δi then exit;
          end
     end
```

$new := \{u \mid u = [f, v_j]$, if u is linearly independent of all elements of Δ_i and of all constructed u's so far, $j = m+1, \ldots, \dim(\Delta_i)\}$;
$vol := u_j \;]vol, \; j = 1, \ldots, \dim(new)$;
$m := \dim(\Delta_i)$;
$n := \dim(new)$;
$i := i + 1$;
if $n > 0$ then exit;
end
if $\dim(\Delta_{n-1}) \neq n$ then exit; -> condition (3)
system is input/state linearizable if not exit has occurred

Mostly, a general AI–system eqn (1) is not input/state linearizable. For this reason, in order to minimize the dimension of the zero dynamics, an algorithm for the determination of the largest input/output linearizable subsystem is also implemented (see Kugi[7]).

5 Chemical Application Example: Stirrer Vessel

The stirrer vessel presented by Klatt et al.[6] is a benchmark example for nonlinear control and it is described by the mathematical model

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}c_A &= \frac{\dot{V}}{V_R}(c_{A_0} - c_A) - k_1(\vartheta)\, c_A - k_3(\vartheta)\, c_A^2 \\
\frac{\mathrm{d}}{\mathrm{d}t}c_B &= -\frac{\dot{V}}{V_R} c_B + k_1(\vartheta)\, c_A - k_2(\vartheta)\, c_B \\
\frac{\mathrm{d}}{\mathrm{d}t}\vartheta &= \frac{\dot{V}}{V_R}(\vartheta_0 - \vartheta) - \frac{1}{\rho C_P}\left[k_1(\vartheta)\, c_A \Delta H_{R_{AB}} + k_2(\vartheta)\, c_B \Delta H_{R_{BC}}\right. \\
&\quad \left. + k_3(\vartheta)\, c_A^2 \Delta H_{R_{AD}}\right] + \frac{k_W A_R}{\rho C_P V_R}(\vartheta_K - \vartheta) \\
\frac{\mathrm{d}}{\mathrm{d}t}\vartheta_K &= \frac{1}{m_K C_{P_K}}\left[\dot{Q}_K + k_W A_R(\vartheta - \vartheta_K)\right]
\end{aligned} \tag{14}$$

with c_A and c_B as the concentrations of two chemical products A and B, respectively, the temperature in the stirrer vessel ϑ and the temperature of the coolant ϑ_K. The plant inputs are the input flow $\dot{V}/V_R$ of the product A with concentration c_{A_0} and temperature ϑ_0 and the power of the heat exchanger $\dot{Q}_K$. Here, $k_1(\vartheta)$, $k_2(\vartheta)$ and $k_3(\vartheta)$ are nonlinear functions of the temperature ϑ and they describe the reaction rates, $\Delta H_{R_{AB}}$, $\Delta H_{R_{BC}}$ and $\Delta H_{R_{AD}}$ denote the reaction enthalpy and ρ, C_P, V_R, k_W, A_R, m_K and C_{P_K} are system parameters. For a more detailed description of this model the reader is referred to Klatt et al.[6] By means of the algorithms of section 4 implemented in MapleV it can be shown in a few seconds that the mathematical model eqn (14), which is an AI–system with two inputs and two outputs, is strongly accessible, observable and is even input/state linearizable.

6 Conclusion

In this contribution we presented some efficient algorithms for the symbolic analysis and design of nonlinear AI–systems. The reader can get the latest version of the corresponding MapleV library by sending an email request to andi@regpro.mechatronik.uni-linz.ac.at.

References

1. Boothby, W.M., *An Introduction to Differentiable Manifolds and Riemannian Geometry*, Academic Press, London, 1986.

2. Burke, W.L., *Applied Differential Geometry*, Cambridge University Press, 1997.

3. Davenport, H., Siret, Y. & Tournier, E., *Computer Algebra*, Academic Press, San Diego, 1988.

4. de Jager, B., The Use of Symbolic Computation in Nonlinear Control: Is it Viable?, *IEEE Transactions on Automatic Control*, Vol. 40, NO. 1, pp. 84–89, 1995.

5. Isidori, A., *Nonlinear Control Systems*, Springer-Verlag, Heidelberg, 1989.

6. Klatt, K.U., Engell, S., Kremling, A. & Allgoewer, F., Testbeispiel: Rührkesselreaktor mit Parallel– und Folgereaktion, In: *Entwurf nichtlinearer Regelungen*, Engell, S. (ed.), Oldenbourg Verlag, München, pp. 425–432, 1995.

7. Kugi, A., *Nonlinear Control of Electrical Systems*, Ph.D. Thesis, Johannes Kepler University Linz/Austria, 1995.

8. Kugi, A., Schlacher, K. & Kaltenbacher, M., Object Oriented Approach for Large Circuits with Substructures in the Computer Algebra Program Maple V, In: *Software for Electrical Engineering Analysis and Design*, Silvester, P.P. (ed.), Computational Mechanics Publications, Southampton and Boston, pp. 491–500, 1996.

9. Nijmeijer, H. & Van der Schaft, A.J., *Nonlinear Dynamical Control Systems*, Springer-Verlag, New York, 1990.

10. Rothfuß, R., Schaffner, J. & Zeitz, M., Rechnergestützte Analyse und Synthese nichtlinearer Systeme, *Nichtlineare Regelung, Methoden, Werkzeuge, Anwendungen*, VDI Berichte 1026, pp. 267–291, VDI-Verlag, Düsseldorf, 1993.

11. von Westenholz, C., *Differential Forms in Mathematical Physics*, Elsevier, Amsterdam, 1986.