THESIS ON INFORMATICS AND SYSTEM ENGINEERING C53

Computer Algebra Tools for Modelling, Analysis and Synthesis for Nonlinear Control Systems

MARIS TÕNSO

TALLINN UNIVERSITY OF TECHNOLOGY
Faculty of Information Technology
Institute of Cybernetics

Dissertation was accepted for the defence of the degree of Doctor of Philosophy in informatics and system engineering on December 21, 2009.

Supervisor:	D.Sc., Leading Research Scientist Ülle Kotta, Institute of Cybernetics, Tallinn University of Technology
Opponents:	D.Sc., Professor, Academician Felix L. Chernousko, Institute for Problems in Mechanics of the Russian Academy of Sciences, Moscow, Russia
	D.Sc., Professor Alain Glumineau, Institut de Recherche en Communications et en Cybernétique de Nantes, Ecole Centrale de Nantes, France

Defence of the thesis: January 27, 2010

Declaration: Hereby I declare that this doctoral thesis, my original investigation and achievement, submitted for the doctoral degree at Tallinn University of Technology has not been submitted for any academic degree.

Maris Tõnso

ISSN 1406-4731
ISBN 978-9985-59-964-8

VÄITEKIRI INFORMAATIKAST JA SÜSTEEMITHENIKAST C53

Arvutialgebra vahendid mittelineaarsete juhtimissüsteemide modelleerimiseks, analüüsiks ja sünteesiks

MARIS TÕNSO

Contents

Introduction

The most popular approach in nonlinear control theory is based on differential geometry [10, 22]. The introduction of differential algebraic methods [7] offered alternative tools. Today, the algebraic point of view has gained popularity and the related approach, based on the vector spaces of differential one-forms over suitable fields of nonlinear functions has been described in [4] for the continuous-time systems and in [9, 1] for the discrete-time systems. The tools based on differential one-forms and the related methods based on the theory of the skew polynomial rings are complementary to the differential geometric methods, but what is more important, these tools are characterized by their inherent simplicity and strong similarity to their linear counterparts. The latter makes these tools a better choice in teaching engineering courses in nonlinear control and in practical applications.

In this thesis the algebraic approach of differential one-forms is used to provide a solution for three modeling problems. First, the necessary and sufficient irreducibility condition for the multi-input multi-output (MIMO) discrete-time nonlinear control systems has been found together with the method for system reduction. Second, the realizability conditions for quadratic input-output differential (i/o) equation have been formulated directly in terms of system parameters. Third, an alternative method for computing a certain subspace of differential one-forms, which allows to simplify the solution of the realization problem, is presented for a single-input single-output (SISO) system. Additionally, we have developed computer algebra package NLControl for solving modeling, synthesis and analysis problems, which is largely based on algebraic and polynomial approaches and is implemented within Mathematica. More important functions of the NLControl package are made available over the internet using webMathematica tools.

The choice of these tasks was motivated by several observations. Most results on nonlinear identification are achieved for systems, described by i/o differential equations. At the same time the majority of techniques for nonlinear system analysis and control design are based on state-space description. To apply the numerous available methods requires realization, i.e. recovering the state-space model, if possible, starting from the i/o equations. Algebraic approach of differential one-forms has been applied for studying realization problem in [4] and in [14], respectively for the continuous- and discrete-time systems; however, there still exist some gaps related to this topic. In order to obtain accessible realization, the i/o equations have to be in the irreducible form. So far the irreducibility conditions were formulated only for SISO systems, using differential one-forms in [4] and skew polynomials in [31, 13], but not for MIMO systems.

In course of practical identification process there is a certain freedom when selecting the mathematical model describing the system. Since an arbitrary i/o model does not necessarily have a state space representation, it is desirable to choose a realizable model. Though the realizability conditions given in [4] are transparent and inherently simple, they are not helpful if we want to check realizability directly from the knowledge of the model parameters. Therefore, our aim was to provide suggestions to identifiers, by formulating realizability conditions for the continuous-time quadratic i/o model in terms of the system parameters. These conditions generalize the earlier realizability conditions for bilinear systems [16].

We have found an alternative to the method described in [1] for realization of discrete-time SISO equation, which is based on skew polynomials. The new method is faster, more direct and therefore better suited for implementation in computer algebra packages like Mathematica or Maple, while the algorithm given in [1] is more general, and applicable also for MIMO case.

And finally, since the solutions of nonlinear control problems require a huge amount of symbolic computations, additional assistance is provided by the nonlinear control system software package NLControl developed by us. The package is based on the algebraic methods of differential one-forms and skew polynomials, and is developed within (symbolic) software system Mathematica.

The thesis is organized as follows. Section 1 gives a short overview of the linear algebraic and polynomial approach and subsection 2.1 recalls some basic definitions. In subsection 2.2 necessary and sufficient irreducibility condition as well as reduction algorithm for discrete-time MIMO systems are introduced. In subsection 2.3 realizability conditions for continuous-time quadratic systems in terms of the system parameters are presented. The complete list of second- and third-order realizable i/o quadratic models is given and two subclasses of the n-th order realizable i/o quadratic systems are suggested. In subsection 2.4 solution of the realization problem in terms of skew polynomials is formulated for SISO discrete-time system. Section 3 describes NLControl package together with its web application and also provides some examples. The last section draws the conclusion. The main results of the thesis are presented in subsections 2.2, 2.3 and 2.4.

Acknowledgements

Financial support from Estonian Science Foundation (grants N^0 5405 and N^0 6922) and Estonian Doctoral School in ICT is greatly appreciated.

I am deeply grateful to my supervisor Dr. Ülle Kotta for her guidance and advices during my doctoral and master studies. Without her encouragement and motivation this work would not have been possible. I would like to thank all the coauthors of our joint papers for their contribution. My gratitude also belongs to my colleagues for providing the pleasant working atmosphere and helping me in many different ways. Many thanks to Martin Ondera from Košice, who helped to keep alive my belief in the possibility to finish the dissertation. I would also like to thank my family for their patience and understanding. And finally, I will always owe my mother for all the help and support I got from her during the writing process, but who was unfortunately not given to see the completion of this work.

1 Preliminaries

1.1 Control systems

Throughout the work we consider two representations of the nonlinear control system, state space and input-output (i/o) equations. A nonlinear multi-input multi-output (MIMO) control system can be described by the state equations

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

in the discrete-time case, or by

$$\begin{aligned} \dot{x} &= f(x, u) \\ y &= h(x) \end{aligned} \tag{2}$$

in the continuous-time case. In both cases $x \in X \subset I\!R^n$ is a state variable, $u \in U \subset I\!R^m$ is an input variable, $y \in Y \subset I\!R^p$ is an output variable, $f : U \times X \to X$ and $h : X \to Y$ are real analytic functions. The algebraic method applied in the present work is based on the difference/differential field, associated with the control system. In order to construct the difference field, associated with the system (1), the following assumption has to be satisfied:

A1 System (1) is generically submersive, i.e. almost everywhere except on the set of measure zero, the state transition map in (1) satisfies

$$\operatorname{rank}\frac{\partial f}{\partial(x(t), u(t))} = n.$$

Note that in the continuous-time case this assumption is not necessary [3].

A MIMO system in the i/o form is described by the set of higher order difference equations

$$\begin{aligned} y_i(t+n_i) &= \phi_i(y_\nu(t), \dots, y_\nu(t+n_{i\nu}-1), u_k(t), \dots, u_k(t+\alpha_{ik}), \\ &\quad\ \nu = 1, \dots, p, \quad k = 1, \dots, m), \\ i &= 1, \dots, p \end{aligned} \tag{3}$$

in the discrete-time case, and by the set of higher order differential equations

$$\begin{aligned} y_i^{(n_i)} &= \phi_i(y_\nu, \dots, y_\nu^{(n_{i\nu}-1)}, u_k, \dots, u_k^{(\alpha_{ik})}, \\ &\quad\ \nu = 1, \dots, p, \quad k = 1, \dots, m), \\ i &= 1, \dots, p \end{aligned} \tag{4}$$

in the continuous-time case. In both equations $u = (u_1, \dots, u_m) \in U \subset I\!R^m$ is an input variable, $y = (y_1, \dots, y_p) \in Y \subset I\!R^p$ is an output variable and ϕ_i are real analytic functions. Notations $n := n_1 + \ldots + n_p$ and $\alpha :=$

$\max\{\alpha_{ik},\ k = 1, \ldots, m,\ i = 1, \ldots, p\}$ are used for systems (3) and (4). In order to construct the difference field, associated with the system (3), the submersivity assumption has to be satisfied again:

A2 System (3) satisfies generically the condition

$$\operatorname{rank} \frac{\partial \phi_i(\cdot)}{\partial(y(t), u(t))} = p.$$

Additionally we require that the following assumptions hold for systems (3) and (4).

A3 System is strictly proper, i. e. $\alpha_{ik} < n_i$, for $i = 1 \ldots, p$, $k = 1, \ldots, m$.

A4 Since the mathematical tools we employ require that instead of working with the equations themselves, we work with their differentials, the systems $\phi(\cdot) = 0$ and $\phi(\cdot) + \text{const} = 0$ are not distinguished for arbitrary constant value. In order to avoid such situations we fix the constant to be zero.

The assumption A4 is restrictive, but the results of the present work can be extended to the case when *const* $\neq 0$.

1.2 Difference/differential field; differential forms

Below we give an overview of the algebraic approach based on the differential one-forms [9, 1, 14]. We provide a detailed exposition for the discrete-time case, since it is more complicated and then briefly mention the differences of the continuous-time case.

Consider the system described by the equations (1). Let $\mathcal{K}$ denote the field of meromorphic functions in a finite number of variables from the infinite set $\{x(0),\ u_j(t),\ j = 1, \ldots, m,\ t \geq 0\}$. Introduce the forward-shift operator $\delta : \mathcal{K} \to \mathcal{K}$, which is defined by shifting the arguments of the function according to the rules $\delta x_i(t) = x_i(t+1) = f_i(\cdot),\ i = 1, \ldots, n,\ \delta u_j(t) = u_j(t+1),\ j = 1, \ldots, m$. An important fact to emphasize is that in $\mathcal{K}$, $x_i(t+1), i = 1, \ldots, n$ are not independent variables and should be always replaced by $f_i(\cdot)$ from equations (1). Under assumption A1, δ is injective and the pair $(\mathcal{K}, \delta)$ is a *difference field.*

The inverse operator of δ is denoted by δ^{-1} and called backward-shift operator. The difference field $(\mathcal{K}, \delta)$ is not inversive in general, which means that $\delta^{-1}\zeta$ may not have a pre-image in $\mathcal{K}$ for all $\zeta \in \mathcal{K}$. Up to an isomorphism, there exists a unique inversive difference field $(\mathcal{K}^*, \delta^*)$, called the *inversive closure* of $(\mathcal{K}, \delta)$, such that $\mathcal{K} \subset \mathcal{K}^*$, $\delta^* : \mathcal{K}^* \to \mathcal{K}^*$ is an automorphism and the restriction of δ^* to $\mathcal{K}$ equals δ. By abuse of notation, hereinafter we assume that $(\mathcal{K}^*, \delta^*)$ is given and use the same symbol to denote $(\mathcal{K}, \delta)$ and its inversive closure. A construction of $\mathcal{K}^*$ for practical computations is given [1].

In order to define $\mathcal{K}^*$ for i/o equations we may associate with the set of i/o equations (3) its extended state-space system with the input $v(t) = u(t+\alpha+1)$, the state $z(t) = [y_1(t), \ldots, y_1(t+n_1-1), \ldots, y_p(t), \ldots, y_p(t+n_p-1), u_1(t), \ldots, u_1(t+\alpha), \ldots, u_m(t), \ldots, u_m(t+\alpha)]^T$ and a state transition map $f_e(z(t), v(t))$ defined as

$$\begin{array}{rcl}
z_1(t+1) & = & z_2(t) \\
 & \cdots & \\
z_{n_1}(t+1) & = & \phi_1(z(t)) \\
 & \cdots & \\
z_{n_1+\ldots+n_{p-1}+1}(t+1) & = & z_{n_1+\ldots+n_{p-1}+2}(t) \\
 & \cdots & \\
z_{n_1+\ldots+n_p}(t+1) & = & \phi_p(z(t)) \\
z_{n+(j-1)(\alpha+1)+k}(t+1) & = & z_{n+(j-1)(\alpha+1)+k+1}(t) \\
z_{n+j(\alpha+1)}(t+1) & = & v_j(t)
\end{array} \qquad (5)$$

for $j = 1, \ldots, m,\ k = 1, \ldots, \alpha$. After that, $\mathcal{K}^*$ for equations (3) can be constructed in a similar manner as for state equations (1).

For the continuous-time case the best reference on algebraic approach of differential one-forms is [4]. Consider the system described by the equations (2). Now $\mathcal{K}$ denotes the field of meromorphic functions in a finite number of variables from $\{x, u_j^{(k)},\ j = 1, \ldots, m,\ k \geq 0\}$. Let $s : \mathcal{K} \to \mathcal{K}$ denote the time derivative operator $\mathrm{d}/\mathrm{d}t$. The pair $(\mathcal{K}, s)$ is a *differential field.*

Over the field $\mathcal{K}$ one can define a vector space $\mathcal{E} := \mathrm{span}_{\mathcal{K}}\{\mathrm{d}\varphi \mid \varphi \in \mathcal{K}\}$. The elements of $\mathcal{E}$ are called *one-forms.* In the discrete-time case the operator $\delta : \mathcal{K} \to \mathcal{K}$ induces a forward-shift operator $\delta : \mathcal{E} \to \mathcal{E}$ by

$$\sum_i a_i \mathrm{d}\varphi_i \mapsto \sum_i (\delta a_i)\mathrm{d}(\delta\varphi_i), \quad a_i, \varphi_i \in \mathcal{K} \qquad (6)$$

and $\delta^{-1} : \mathcal{K} \to \mathcal{K}$ induces a backward-shift operator $\delta^{-1} : \mathcal{E} \to \mathcal{E}$ by

$$\sum_i a_i \mathrm{d}\varphi_i \mapsto \sum_i (\delta^{-1} a_i)\mathrm{d}(\delta^{-1}\varphi_i), \quad a_i, \varphi_i \in \mathcal{K}. \qquad (7)$$

In the continuous-time case the operator $s : \mathcal{K} \to \mathcal{K}$ induces a derivative operator $s : \mathcal{E} \to \mathcal{E}$ by

$$\sum_i a_i \mathrm{d}\varphi_i \to \sum_i s a_i \mathrm{d}(\varphi_i) + \sum_i a_i \mathrm{d}(s\varphi_i), \quad a_i, \varphi_i \in \mathcal{K}.$$

Definition 1 The relative degree r of a one-form $\omega \in \mathcal{E}$ is defined to be the least integer such that $\delta^r\omega \notin \mathrm{span}_{\mathcal{K}}\{\mathrm{d}x(0)\}$ or $s^r\omega \notin \mathrm{span}_{\mathcal{K}}\{\mathrm{d}x\}$, in the discrete- and continuous-time cases, respectively. If such an integer does not exist, we set $r = \infty$.

A sequence of subspaces $\{\mathcal{H}_k\}$ of $\mathcal{E}$ associated with the system (1) is defined by

$$\begin{aligned} \mathcal{H}_1 &= \operatorname{span}_{\mathcal{K}}\{\mathrm{d}x(0)\} \\ \mathcal{H}_{k+1} &= \{\omega \in \mathcal{H}_k \mid \delta\omega \in \mathcal{H}_k\},\ k \geq 1. \end{aligned} \tag{8}$$

Analogously, $\{\mathcal{H}_k\}$ associated with the system (2) is defined by

$$\begin{aligned} \mathcal{H}_1 &= \operatorname{span}_{\mathcal{K}}\{\mathrm{d}x\} \\ \mathcal{H}_{k+1} &= \{\omega \in \mathcal{H}_k \mid \dot{\omega} \in \mathcal{H}_k\},\ k \geq 1. \end{aligned} \tag{9}$$

Each $\mathcal{H}_k$ contains the one-forms with relative degree equal to k or greater than k. The sequences (8) and (9) are decreasing. Denote by k^* the least integer such that $\mathcal{H}_1 \supset \ldots \supset \mathcal{H}_{k^*} \supset \mathcal{H}_{k^*+1} = \mathcal{H}_{k^*+2} = \ldots = \mathcal{H}_\infty$. The subspaces $\mathcal{H}_k$ are invariant with respect to the regular static state feedback and state coordinate transformation.

Theorem 1 *(Frobenius) Let $\mathcal{W} = \operatorname{span}_{\mathcal{K}}\{\omega_1, \ldots, \omega_r\}$ be a subspace of $\mathcal{E}$. $\mathcal{W}$ is closed iff $\mathrm{d}\omega_k \wedge \omega_1 \wedge \ldots \wedge \omega_r = 0$ for all $k = 1, \ldots, r$.*

Under Frobenius conditions there exists locally a system of coordinates $\{\zeta_1, \ldots, \zeta_r\}$ such that $\mathcal{W}$ is generated by $\{\mathrm{d}\zeta_1, \ldots, \mathrm{d}\zeta_r\}$. In this case $\mathcal{W}$ is said to be completely integrable.

1.3 Polynomial framework

1.3.1 Non-commutative polynomial ring

Polynomial framework is built upon the framework of differential one-forms. Below we focus on the discrete-time case, the differences from the continuous-time case are briefly indicated. A *left* polynomial is an element which can be uniquely written in the form

$$a(\partial) = \sum_{i=0}^{n} a_i \partial^{n-i}, \quad a_i \in \mathcal{K}, \tag{10}$$

where ∂ is polynomial indeterminate and $a(\partial) \neq 0$ iff at least one of the functions a_i, $i = 0, \ldots, n$ is nonzero. If $a_0 \not\equiv 0$, then the positive integer n is called the *degree* of the left polynomial $a(\partial)$ and denoted by $\mathrm{d}^0(a)$. In addition, we set $\mathrm{d}^0(0) = -\infty$.

The difference field $(\mathcal{K}, \delta)$ induces a (left) noncommutative skew polynomial ring.

Definition 2 The left skew polynomial ring induced by $(\mathcal{K}, \delta)$ is the ring $\mathcal{K}[\partial, \delta]$ of polynomials in ∂ over $\mathcal{K}$ with usual addition, and the noncommutative multiplication defined by the commutation rule

$$\partial \cdot a = \delta a \cdot \partial \tag{11}$$

for any $a \in \mathcal{K} \subset \mathcal{K}[\partial, \delta]$. A ring is called an *integral domain*, if it does not contain zero divisors. This means that if a and b are two elements of the ring such that $ab = 0$, then $a = 0$ or $b = 0$.

Lemma 1 *[17]*

(i) The ring $\mathcal{K}[\partial, \delta]$ is an integral domain.

(ii) If a and b are nonzero polynomials, then $\mathrm{d}^0(a\,b) = \mathrm{d}^0(a) + \mathrm{d}^0(b)$.

Moreover, the ring $\mathcal{K}[\partial, \delta]$ satisfies the (left) Ore condition, which guarantees that for all nonzero $a, b \in \mathcal{K}[\partial, \delta]$ there exist nonzero $a_1, b_1 \in \mathcal{K}[\partial, \delta]$ such that $a_1 b = b_1 a$.

Elements of such ring are called skew polynomials or non-commutative polynomials or Ore polynomials [25].

In the continuous-time case the ring of Ore polynomials is induced by the differential field $(\mathcal{K}, s)$ and the commutation rule is given, instead of (11), by

$$\partial \cdot a = a \cdot \partial + s(a) \tag{12}$$

for any $a \in \mathcal{K} \subset \mathcal{K}[\partial, s]$.

Since $\mathcal{K}[\partial, \delta]$ is an Ore ring, one can construct the division ring of fractions. If $p(\partial) = p_1(\partial)p_2(\partial)$, then $p_1(\partial)$ is called a *left divisor* of $p(\partial)$ and $p(\partial)$ is called left divisible by $p_1(\partial)$. If for $p_1(\partial), p_2(\partial) \in \mathcal{K}[\partial, \delta]$, $p_c(\partial)$ is a left divisor of $p_1(\partial) - p_2(\partial)$, then $p_c(\partial)$ is called a *common left divisor* of $p_1(\partial)$ and $p_2(\partial)$. If the degree of $p_c(\partial)$ is the greatest of all common left divisors of $p_1(\partial) - p_2(\partial)$, then $p_c(\partial)$ is called the *greatest common left divisor* (gcld). To find the gcld one can use the *left Euclidean division algorithm.* The gcld is only unique up to multiplication by function from $\mathcal{K}$, but it can be made unique by requiring that it has to be monic.

1.3.2 Polynomial matrices

We now consider a class of matrices $R(\partial)$ whose elements are polynomials $r(\partial) \in \mathcal{K}[\partial, \delta]$ of finite, but unbounded degree and write $\mathcal{K}^{p \times q}[\partial, \delta]$ for the set of $p \times q$ matrices with entries in $\mathcal{K}[\partial, \delta]$. Like in the linear case where the polynomials have real coefficients, the polynomial matrix with entries in $\mathcal{K}[\partial, \delta]$ can be transformed by a sequence of elementary column operations to the lower left triangular form. This result allows us to obtain the irreducibility criterion for nonlinear control systems.

Definition 3 The following three *elementary column operations* $E(\partial)$ on the polynomial matrix $R(\partial)$ are defined

(i) Interchange of columns i and j.

(ii) Multiplication of column i by nonzero scalar in $\mathcal{K}$.

(iii) Replacement of column i by itself plus any polynomial multiplied by any other column j.

Definition 4 A matrix $U(\partial) \in \mathcal{K}[\partial, \delta]$ is called *unimodular* if it is invertible in $\mathcal{K}[\partial, \delta]$, that is, there exists a matrix $U^{-1}(\partial)$ in $\mathcal{K}[\partial, \delta]$.

A unimodular matrix $U(\partial)$ can be obtained from the identity matrix $\mathcal{I}$ by a finite number of elementary column operations on $\mathcal{I}$: $U(\partial) = \mathcal{I}E_1(\partial) \ldots E_k(\partial)$. Any sequence of elementary column operations on $R(\partial)$ is equivalent to right multiplication of $R(\partial)$ by an appropriate unimodular matrix $U(\partial)$.

Definition 5 Two polynomial matrices $R(\partial)$ and $\hat{R}(\partial)$ are called *column equivalent* iff $R(\partial) = \hat{R}(\partial)U(\partial)$, where $U(\partial)$ is a unimodular matrix.

Definition 6 If three polynomial matrices satisfy the relation $P(\partial) = C_L(\partial)Q(\partial)$, then $C_L(\partial)$ is called a *left divisor* of $P(\partial)$ and $P(\partial)$ is called a *right multiple* of $C_L(\partial)$. A gcld of two polynomial matrices $P(\partial)$ and $Q(\partial)$ is a common left divisor which is a right multiple of every common left divisor of $Q(\partial)$ and $P(\partial)$.

Definition 7 A pair $\{P(\partial), Q(\partial)\}$ of polynomial matrices which have the same number of rows is said to be *relatively left prime* if and only if its gclds are unimodular matrices.

1.3.3 Polynomial system description

We now represent the nonlinear system (3) in terms of two polynomial matrices, with the polynomials from the Ore ring $\mathcal{K}[\partial, \delta]$. For that we apply the differentiation operation to (3) and use the relations $\mathrm{d}y_s(t+j) = \delta^j \mathrm{d}y_s(t)$, $\mathrm{d}u_k(t+r) = \delta^r \mathrm{d}u_k(t)$ (that follow from (6)) to obtain

$$P(\partial)\mathrm{d}y(t) = Q(\partial)\mathrm{d}u(t), \tag{13}$$

where $P(\partial)$ and $Q(\partial)$ are $p \times p$ and $p \times m$-dimensional matrices respectively, whose elements $p_{ij}, q_{ij} \in \mathcal{K}[\delta]$:

$$p_{is}(\partial) = \delta^{n_i} - \sum_{j=0}^{n_{is}-1} \frac{\partial f_i}{\partial y_s(t+j)} \partial^j, \qquad q_{ik}(\partial) = \sum_{r=0}^{\alpha_{ik}} \frac{\partial f_i}{\partial u_k(t+r)} \partial^r$$

and $\mathrm{d}y(t) = [\mathrm{d}y_1(t), \ldots, \mathrm{d}y_p(t)]^T$, $\mathrm{d}u(t) = [\mathrm{d}u_1(t), \ldots, \mathrm{d}u_m(t)]^T$.

2 System reduction and realization

2.1 Introduction

The realization problem is defined as follows. Given a nonlinear system, described by the i/o equation of the form (3), find, if possible, the state coordinates $x(t) \in X \subset I\!R^n$, $x(t) = \psi(y(t), \ldots, y(t+n-1), u(t), \ldots, u(t+\alpha))$

such that in these coordinates the system takes the classical state space form (1) and the sequences $\{u(t), y(t), t \geq 0\}$ generated by (1) and (3) coincide. Then (1) is called *realization* of (3). Before formulating the realizability conditions, the two important properties of the control systems, accessibility and observability, are recalled.

Accessibility is the structural property of the nonlinear system that in the linear case reduces to the controllability property. The necessary and sufficient condition for accessibility is $\mathcal{H}_\infty = \{0\}$ [1]. A nth-order realization of equation (3) is accessible iff system (3) is irreducible. The necessary and sufficient irreducibility conditions as well as the reduction algorithm for system (3) are given in section 2.2.

Definition 8 System (1) is said to be (single-experiment) **observable** if the observability matrix has generically full rank, i.e. if

$$\operatorname{rank}_{\mathcal{K}} \frac{\partial(h(x(t)), \delta h(x(t)), \ldots, \delta^{n-1} h(x(t)))}{\partial x(t)} = n. \tag{14}$$

This condition reduces to the standard Kalman observability criterion in the special case of linear systems.

Theorem 2 *[14] The nonlinear system described by the i/o difference equation (3) has an accessible and observable state space realization of order n iff for $1 \leq k \leq \alpha + 2$ the subspaces $\mathcal{H}_k$ defined by (8) are completely integrable. Moreover, the state coordinates can be obtained by integrating the basis vectors of $\mathcal{H}_{\alpha+2}$.*

In the continuous-time case the solution of the realization problem of the set of i/o differential equations (4) is analogous. In Definition 8 one has to replace the forward-shift operator δ by the time-derivative operator $s = d/dt$.

Theorem 3 *The nonlinear system described by the set of i/o differential equations (4) has an accessible and observable state space realization of order n iff for $1 \leq k \leq \alpha + 2$ the subspaces $\mathcal{H}_k$ defined by (9) are completely integrable. Moreover, the state coordinates can be obtained by integrating the basis vectors of $\mathcal{H}_{\alpha+2}$.*

2.2 Irreducibility conditions for discrete-time systems; system reduction

The results of the present subsection are based on **paper I**. Definition of the irreducible system is based on the concept of autonomous variable.

Definition 9 A (possibly vector-valued) function φ_r (with components) in $\mathcal{K}$ is said to be an *autonomous variable* for a system (3) if there exist an integer $\mu \geq 1$ and a non-zero meromorphic function F (again possibly vector-valued) such that

$$F(\varphi_r, \delta\varphi_r, \ldots, \delta^{\mu}\varphi_r) = 0. \tag{15}$$

Definition 10 The system (3) is said to be generically irreducible if there does not exist any non-zero autonomous variable for the system (3) in a difference field $\mathcal{K}$, associated with system (3).

The next two theorems describe the important properties of the polynomial matrices, necessary to represent the irreducibility conditions for system (3).

Theorem 4 *Any $p \times q$ $(p \leq q)$ polynomial matrix $R(\partial)$ is column equivalent to the lower left triangular matrix shown below, i. e. one can always find a sequence of elementary column operations which reduces $R(\partial)$ to the form $\hat{R}(\partial) = [G_L(\partial) \vdots 0]$, where*

$$G_L(\partial) = \begin{bmatrix} g_{11}(\partial) & & & \\ g_{21}(\partial) g_{22}(\partial) & & & \\ \vdots & & & \\ g_{p1}(\partial) g_{p2}(\partial) \dots g_{pp}(\partial) & & & \end{bmatrix}. \tag{16}$$

Furthermore, in the above form, the polynomials $g_{k1}(\partial), \dots, g_{k,k-1}(\partial)$ are of lower degree than $g_{kk}(\partial)$ for all $k = 1, \dots, p$ if $\deg g_{kk}(\partial) > 0$, and are all zero, if $g_{kk}(\partial)$ is a nonzero scalar in $\mathcal{K}$.

Theorem 5 *Consider the pair $\{P(\partial), Q(\partial)\}$ of polynomial matrices which have the same number of rows. If the composite matrix $[P(\partial) \vdots Q(\partial)]$ is reduced to lower left triangular form $[G_L(\partial) \vdots 0]$ as in Theorem 4, then $G_L(\partial)$ is a gcld of $P(\partial)$ and $Q(\partial)$.*

The gcld is, in general, not unique, but it can be made unique by requiering that the polynomials in the diagonal of $G_L(\partial)$ are monic. Irreducibility conditions are stated by the next theorem, which is the main result of paper I.

Theorem 6 *The nonlinear system (3) is irreducible iff the polynomial matrices $P(\partial)$ and $Q(\partial)$ in system description (13) are relatively left prime.*

If $P(\partial)$ and $Q(\partial)$ have non-unimodular gcld $G_L(\partial)$, then (13) can be rewritten as

$$G_L(\partial)\omega = 0,$$

where $\omega = [\omega_1, \dots, \omega_p]^T$ is a column-vector of exact differential one-forms (or can be made exact by multiplying them by the integrating factor). So one can define $\omega_i = \mathrm{d}\varphi_{ri}$ for $i = 1, \dots, p$, being elements of $\tilde{P}(\partial)\mathrm{d}y(t) - \tilde{Q}(\partial)\mathrm{d}u(t)$. The reduced system equations are $\varphi_{ri} = 0, i = 1, \dots, p$.

2.3 Realizability conditions of continuous-time quadratic systems

The results of the present subsection are based on **paper II**. Consider a single-input single-output (SISO) case of the system (4) where $p = m = 1$; and define $\phi(\cdot) := \phi_1(\cdot)$, $u := u_1$, $y := y_1$, $n := n_1$ and $\alpha := \alpha_{11}$,

$$y^{(n)} = \phi(y, \ldots, y^{(n-1)}, u, \ldots, u^{(\alpha)}). \tag{17}$$

In many practical situations continuous-time i/o models of the form (17) are deduced from i/o data when no information regarding the structure of the observed dynamical system is available *a priori*. Such representations form the basis of much modern identification theory. Identification therefore involves model structure selection prior to parameter estimation. In practice, this involves selecting a form of the nonlinear function $\phi(\cdot)$, α and n, i.e. specifying the maximal derivatives for the input and output, respectively, that appear in equation (17). Typically, ϕ is assumed to be a low order polynomial, most often a bilinear or quadratic function, and not all possible terms are included since in most cases a more complex model does not necessarily equate to a better model.

Our purpose was to find the subsets of quadratic i/o equations

$$\begin{aligned} y^{(n)} &= \sum_{i=1}^{n} a_i y^{(n-i)} + \sum_{i=1}^{n} b_i u^{(n-i)} + \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} y^{(n-i)} u^{(n-j)} \\ &+ \sum_{i=1}^{n}\sum_{j=i}^{n} d_{ij} y^{(n-i)} y^{(n-j)} + \sum_{i=1}^{n}\sum_{j=i}^{n} e_{ij} u^{(n-i)} u^{(n-j)} \end{aligned} \tag{18}$$

that are guaranteed to have a state space representation (2) of order n, and as such are good candidate structures to be used in system identification. Since the quadratic model (18) is linear in the parameters, it lends itself easily to the well-established parameter estimation algorithms.

Despite the structural simplicity of the quadratic i/o model, the general realizability conditions, given in Theorem 3, yield little insight and do not tell us in terms of the parameters a_i, b_i, c_{ij}, d_{ij}, e_{ij} which quadratic model is realizable in the classical state space form and which is not. To give a more general view of the nature of parameter restrictions necessary for realizability, it is instructive to consider the special cases where $n = 1, 2, 3$.

Note that the first order quadratic i/o model $\dot{y} = a_1 y + b_1 u + c_{11} y u + d_{11} y^2 + e_{11} u^2$ is obviously realizable in the classical state space form, and the choice $x(t) = y(t)$ will yield the state space model. Propositions 1 and 2 below consider the cases $n = 2$ and $n = 3$, respectively. The computer algebra system Mathematica-based software was used to obtain the results of Proposition 2, as well as the results of subsection 2.3.2.

2.3.1 The realization of the 2nd and 3rd order quadratic i/o equations

Proposition 1 *The second order quadratic system described by i/o equation (18) with $n = 2$ is realizable in the classical state space form iff $e_{11} = 0$, and for the case $c_{11} \neq 0$ the state equations are*

$$\begin{aligned}\dot{x}_1 &= -A \\ \dot{x}_2 &= e^{-c_{11}u}\Bigg(b_2 u + e_{22}u^2 + (a_2 + c_{22}u + d_{22}x_1)x_1 \\ &- (a_1 + c_{12}u + \frac{c_{21}}{c_{11}} + d_{12}x_1)A + d_{11}A^2\Bigg)\end{aligned} \tag{19}$$

where $A = (1/c_{11}^2)[e_{12} + c_{11}(b_1 - e^{c_{11}u}c_{11}x_2 + c_{21}x_1 + e_{12}u)]$, and $y = x_1$.

Example 2 in section 3.8 demonstrates how this result can be obtained using Mathematica.

Note that the condition, specified for the second-order quadratic system as $e_{11} = 0$ in Proposition 1, has been known as necessary realizability condition for a long time. This represents the fact, that the highest-order input derivative has to appear linearly in the i/o equation, see, for example, [5]. Moreover, the above result follows also from Theorem 1 in [8] if we rewrite the second order i/o equation as a generalized state equation, containing also the input derivative $\dot{u}$. The remaining part of Proposition 1, that is sufficiency and the explicit form of the state equations, is a new contribution.

Condition $e_{11} = 0$ agrees also with the earlier result for the 2-nd order bilinear i/o equation which is always realizable in the classical state space form [16].

Our results also extend those of [29], which claim that the nonlinear i/o differential equation of the form

$$\begin{aligned}y^{(n)} &+ b_1 y^{(n-1)} + \ldots + b_n y \\ &= a_n u + a_{n-1}u^{(1)} + \ldots + a_{n-m}u^{(m)} \\ &+ N(u, y, y^{(1)}, \ldots, y^{(n-m)}), \qquad m \leq n\end{aligned} \tag{20}$$

can be realized in the classical state-space form. Note that the realizable subclass of the 2nd order quadratic systems, specified in Proposition 1, does not accommodate into the realizable subclass, given by (20) in proposition, because of the term $c_{11}\dot{y}\dot{u} + c_{21}y\dot{u} + (e_{12} + e_{21})\dot{u}u$.

Proposition 2 *The third order quadratic system described by the i/o equation (18) with $n = 3$, is realizable in the classical state space form iff either one of the following set of conditions is satisfied*

(i) $c_{11} = c_{21} = c_{31} = e_{11} = e_{12} = e_{13} = 0$, $e_{22} = -b_1 c_{12} - b_1^2 d_{11}$.

(ii) $c_{11} = d_{11} = e_{11} = e_{12} = 0$, $e_{13} = e_{22}$, $c_{12} = c_{21}$.

Remark 1 Condition (ii) extends the earlier condition for the 3rd order bilinear i/o equation [16]

$$c_{11} = 0, \qquad c_{12} = c_{21}. \tag{21}$$

Note, that condition (i) for bilinear i/o equations yields either $b_1 = c_{11} = c_{21} = c_{31} = 0$ or $c_{11} = c_{21} = c_{31} = c_{12} = 0$, but the second branch is a special case of (21).

Remark 2 For identification purposes, the conditions $c_{21} = c_{12}$, $e_{22} = e_{13}$ in (ii), unless all parameters are equal to zero, are unnatural since there is no reason to assume that the terms $\dot{y}\ddot{u}$ and $\ddot{y}\dot{u}$ or $\dot{u}^2$ and $\ddot{u}u$ should have equal coefficients. The same holds for the requirement $e_{22} = -b_1 c_{12} - b_1^2 \mathrm{d}_{11}$ in (i). For that reason we suggest the following 3rd order realizable i/o equations to be used for modelling purposes

(a) $b_1 = c_{11} = c_{21} = c_{31} = e_{11} = e_{12} = e_{13} = e_{22} = 0$,

(b) $c_{11} = c_{21} = c_{12} = d_{11} = e_{11} = e_{12} = e_{13} = e_{22} = 0$.

The state equations, corresponding to (a), are given as a special case of equations (22) for $n = 3$ (see below). The state equations, corresponding to case (b) are too complicated to present here.

2.3.2 Two realizable subclasses

The results of Propositions 1 and 2 illustrate the complicated nature of realizability conditions for i/o quadratic models. For arbitrary n, in order to get the conditions, one has to go through $n-1$ steps. At each step we obtain several restrictions on system parameters with many branches. All these conditions can be combined together in very many different ways. They yield peculiar structures and most of them are probably not important for practical applications. We suggest below two realizable subclasses of i/o quadratic models. These two subclasses provide two possible patterns of nonzero coefficients.

Subclass 1 $b_1 = \ldots = b_{n-2} = 0$, $c_{11} = \ldots = c_{1,n-2} = 0$, $c_{21} = \ldots = c_{2,n-2} = 0, \ldots, c_{n,1} = \ldots = c_{n,n-2} = 0$. The only nonzero elements of e_{ij} are $e_{n-1,n}$ and $e_{n,n}$. There are no restrictions on d_{ij}. Coefficient matrices $\mathcal{A}^T = (a_i)$, $\mathcal{B}^T = (b_i)$, $\mathcal{C} = (c_{ij})$, $\mathcal{D} = (d_{ij})$ and $\mathcal{E} = (e_{ij})$ for $i, j = 1, \ldots, n$ are shown below on scheme, where $\circ$ represents a zero and $\bullet$ a nonzero coefficient

$$\mathcal{A}^T = \begin{pmatrix} \bullet & \dots & \bullet & \bullet & \bullet \end{pmatrix} \qquad \mathcal{D} = \begin{pmatrix} \bullet & \dots & \bullet & \bullet & \bullet \\ \bullet & \dots & \bullet & \bullet & \bullet \\ \vdots & & \vdots & \vdots & \vdots \\ \bullet & \dots & \bullet & \bullet & \bullet \end{pmatrix}$$

$$\mathcal{B}^T = \begin{pmatrix} \circ & \dots & \circ & \bullet & \bullet \end{pmatrix}$$

$$\mathcal{C} = \begin{pmatrix} \circ & \dots & \circ & \bullet & \bullet \\ \circ & \dots & \circ & \bullet & \bullet \\ \vdots & & \vdots & \vdots & \vdots \\ \circ & \dots & \circ & \bullet & \bullet \end{pmatrix} \qquad \mathcal{E} = \begin{pmatrix} \circ & \dots & \circ & \circ & \circ \\ \vdots & & \vdots & \vdots & \vdots \\ \circ & \dots & \circ & \circ & \circ \\ \circ & \dots & \circ & \circ & \bullet \\ \circ & \dots & \circ & \circ & \bullet \end{pmatrix}$$

Subclass 2 $b_1 = \dots = b_{n-3} = 0$, $c_{11} = \dots = c_{1,n-1} = 0$, $c_{21} = \dots = c_{2,n-2} = 0$, $c_{31} = \dots = c_{3,n-3} = 0, \dots, c_{n,1} = \dots = c_{n,n-3} = 0$, $d_{11} = 0$. The only nonzero elements element of e_{ij} are $e_{n-1,n}$ and $e_{n,n}$.

$$\mathcal{A}^T = \begin{pmatrix} \bullet & \dots & \bullet & \bullet & \bullet & \bullet \end{pmatrix} \qquad \mathcal{D} = \begin{pmatrix} \circ & \bullet & \bullet & \dots & \bullet & \bullet \\ \circ & \bullet & \bullet & \dots & \bullet & \bullet \\ \circ & \circ & \bullet & \dots & \bullet & \bullet \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ \circ & \circ & \circ & \dots & \circ & \bullet \end{pmatrix}$$

$$\mathcal{B}^T = \begin{pmatrix} \circ & \dots & \circ & \bullet & \bullet & \bullet \end{pmatrix}$$

$$\mathcal{C} = \begin{pmatrix} \circ & \dots & \circ & \circ & \circ & \bullet \\ \circ & \dots & \circ & \circ & \bullet & \bullet \\ \circ & \dots & \circ & \bullet & \bullet & \bullet \\ \vdots & & \vdots & \vdots & \vdots & \vdots \\ \circ & \dots & \circ & \bullet & \bullet & \bullet \\ \circ & \dots & \circ & \bullet & \bullet & \bullet \end{pmatrix} \qquad \mathcal{E} = \begin{pmatrix} \circ & \dots & \circ & \circ & \circ & \circ \\ \vdots & & \vdots & \vdots & \vdots & \vdots \\ \circ & \dots & \circ & \circ & \circ & \circ \\ \circ & \dots & \circ & \circ & \circ & \bullet \\ \circ & \dots & \circ & \circ & \circ & \bullet \end{pmatrix}$$

Note that realizability conditions for the first case impose restrictions on b_i, c_{ij} and e_{ij} parameters. Further, note that if $b_j = 0$ is required, then so is c_{ij} for all i. Conversely, note that in the other case, one additional c_{ij} parameter, and one additional d_{ij} parameter, namely $c_{1,n-1}$ and d_{11} must also be zero for realizability. But now when compared with the first case, $b_{n-2}, c_{3,n-2}, \dots, c_{n,n-2}$ may be nonzero. Overall, the two subclasses do not differ much. The notable restrictions have to be imposed on e_{ij} parameters – all of them, except $e_{n-1,n}$ and e_{nn} have to be zero for both cases.

For the 1st subclass we also give the state equations, shown below:

$$\dot{x}_1 = x_2, \dots, \dot{x}_{n-2} = x_{n-1}, \quad \dot{x}_{n-1} = -A$$

$$\dot{x}_n = e^{-c_{1,n-1}u} \left(b_n u + e_{nn}u^2 + \sum_{i=1}^{n-1} B_i x_i + (d_{11}x_n - B_n)A + d_{11}A^2 \right) \tag{22}$$

where

$$
\begin{aligned}
A &= \frac{1}{c_{1,n-1}^2}\Bigg(b_{n-1}c_{1,n-1} + (1 + c_{1,n-1}u)e_{n-1,n} \\
&+ c_{1,n-1}\sum_{j=1}^{n}(-e^{c_{1,n-1}u})^{\delta_{jn}} c_{n-j+1,n-1}x_j \Bigg), \\
B_i &= \begin{cases} C_i, & i = 1 \\ C_i + \dfrac{c_{n-i+2,n-1}}{c_{1,n-1}}, & i \geq 2 \end{cases} \\
C_i &= a_{n-i+1} + c_{n-i+1,n}u + \sum_{j=1}^{i} d_{n-i+1,n-j+1}x_j.
\end{aligned}
$$

and δ_{jn} is Kronecker delta.

Though the state equations given above were suggested by applying the realization theory in [4], they can be checked directly by eliminating x from the state equations.

If we analyze the coefficients related to the quadratic terms, we notice a remarkable nonsymmetricity in non-zero e (coefficients of cross-products of different u derivatives) and d (cross-products of different y derivatives) coefficients. Note that in both realizable subclasses only two e coefficients are allowed to be non-zero whereas in the first subclass no restrictions are imposed on the d coefficients at all. In the second subclass only one d element has to be zero.

2.4 Solution of the realization problem using polynomial approach in the discrete-time case

The present subsection is based on **paper III**, which provides an alternative method for computing the sequence $\{\mathcal{H}_k\}$ in Theorem 2. If compared to the earlier method in [1] it is more direct and therefore better suited for implementation in the computer algebra packages like Mathematica or Maple. Moreover, the method, described in [1], requires solving a system of equations, which may fail if the i/o equation is complex, in particular, if it contains roots. For such systems polynomial method is frequently able to produce the result. For the discrete-time systems, polynomial method is also noticeably less time-consuming. However, the method described in [1] is more general, while polynomial method is applicable only to the SISO systems, given by i/o equations.

Consider a SISO case of the system (3) where $p = m = 1$; and define $\phi(\cdot) := \phi_1(\cdot)$, $u := u_1$, $y := y_1$, $n := n_1$ and $\alpha := \alpha_{11}$

$$y(t+n) = \phi(y(t), \ldots, y(t+n-1), u(t), \ldots, u(t+\alpha)), \tag{23}$$

and the corresponding polynomial system description (13) with $P(\partial) = p_{11}(\partial) \triangleq p(\partial)$, $Q(\partial) = q_{11}(\partial) \triangleq q(\partial)$. Equation (13) takes now a form

$$p(\partial)\mathrm{d}y(t) - q(\partial)\mathrm{d}u(t) = 0, \tag{24}$$

where

$$p(\partial) = \partial^n - \sum_{i=0}^{n-1} p_i \partial^i, \quad q(\partial) = -\sum_{j=0}^{s} q_j \partial^j.$$

The aim is to compute sequence $\mathcal{H}_k$ for equation (23). The solution is formulated in terms the (generalized) shift-and-cut operator $\delta_c^{-1} : \mathcal{K}[\partial, \delta] \to \mathcal{K}[\partial, \delta]$, defined as

$$\delta_c^{-1}(p(\partial)) = \delta^{-1}(p(\partial) - p_0).$$

The definition of δ_c^{-1} is extended to vectors and matrices of polynomial elements in a componentwise manner. Iterated application of δ_c^{-1} will be considered in the following and denoted as

$$\delta_c^{-k} = \overbrace{\delta_c^{-1} \circ \delta_c^{-1} \circ \ldots \circ \delta_c^{-1}}^{k \text{ times}}.$$

Theorem 7 [1] *For the i/o model (23), the subspaces $\mathcal{H}_k$ for $k = 2, \ldots, \alpha + 2$ can be calculated as*

$$\begin{aligned} \mathcal{H}_k \;=\; & \mathrm{span}_{\mathcal{K}}\{\mathrm{d}y(t), \ldots, \mathrm{d}y(t+n-k+1), \\ & \mathrm{d}u(t), \ldots, \mathrm{d}u(t+\alpha-k+1),\ \omega_l,\ l = 1, \ldots, k-2\}, \end{aligned} \tag{25}$$

where

$$\omega_l = \delta_c^{-l}[p(\partial), q(\partial)] \begin{bmatrix} \mathrm{d}y(t) \\ \mathrm{d}u(t) \end{bmatrix}. \tag{26}$$

Another task was to compute differentials of the state coordinates.

Corollary 1 *For the realizable i/o equation (23) the differentials of the state coordinates can be calculated as the integrable linear combinations of the one-forms*

$$\omega_i = \delta_c^{-i}[p(\partial), q(\partial)] \begin{bmatrix} \mathrm{d}y(t) \\ \mathrm{d}u(t) \end{bmatrix}, \quad i = 1, \ldots, n.$$

3 NLControl package

The present section is based on **papers IV** and **V**. The package NLControl provides basic tools for modeling, analysis and synthesis both for discrete- and continuous-time nonlinear systems and it is built within computer algebra system Mathematica. The reason for developing the package is that nonlinear

[1] In paper III the theorem was given in slightly different form.

control systems, unlike their linear counterparts, practically miss a support of professional software products. For example, both Matlab Control System Toolbox and Mathematica Control System Professional Suite are applicable only for linear systems. Some custom-made packages based on symbolic computations for nonlinear control systems have been developed, see for example [11, 6, 28, 27, 24], but their distribution is extremely limited and only the last of them implements the methods based on algebraic tools and skew polynomial rings.

Only these functions of the package, which are included into papers IV and V are described in present thesis. A few exceptions are made to functions, related with inversive closure and Ore polynomials, which were omitted from the articles due to space limitations. Note that Mathematica does not have built-in functions dealing with noncommutative polynomials. There exists a custom-made package [21] for computing Groebner basis of Ore polynomial matrices, but the general functionality of the package is very limited. For Maple there is available a general-purpose Ore polynomial package, called OreTools. However, neither of these packages is suitable for application in the field of nonlinear control, because this requires a possibility to define shift/derivative operator of the Ore ring using control system equations. Therefore it was necessary to develop all functions dealing with Ore polynomials from the beginning.

3.1 Control systems

Most NLControl functions operate on special data types, or *control objects*, that contain the information of the control system. These are `StateSpace` and `IO`.

To perform computations with the systems described by the state equations in the form (1) or (2), it is necessary to enter the system in the following form:

`StateSpace[` *f*`,` *Xt*`,` *Ut*`,` *t*`,` *h*`,` *Yt*`,` *type*`]`.

In above, *f* is a list of the components of the state function; *Xt*, *Ut* and *Yt* define a lists of the state, input and output variables, respectively; t is a time argument and h defines the output function. The argument *type* may have one of the following values: `TimeDerivative` stands for continuous-time case and `Shift` for discrete-time case. The function h and the list *Yt* can be omitted, in this case `StateSpace` fills their places with empty lists {}.

To enter the i/o system (3) or (4), one has to use the syntax

`IO[`*eqs*`,` *Ut*`,` *Yt*`,` *t*`,` *type*`]`,

where the meanings of the arguments *Ut* , *Yt* and *type* are the same as in the case of `StateSpace` and the argument *eqs* defines the i/o equation (3) or

(4). As conventional for Mathematica, instead of the lists of input and output variables, a single variable may be entered in case of SISO system. However, `StateSpace` and `IO` always embrace the single variable with curly brackets. This guarantees that SISO and MIMO systems have the same data structure and consequently, the same programs can handle them. Most of the assistant functions or low-level functions require separate programs for discrete- and continuous-time systems. The higher level functions (`Linearization`, `Accessibility`, `Observability` and transformations between different system descriptions) can share the same source code for discrete and continuous-time systems.

The tools of NLControl are not designed for approximate calculations. Therefore, all real (floating-point) numbers are transformed into rational numbers by `StateSpace` and `IO`.

Sometimes (and it is obvious from the examples of `StateSpace` and `IO` functions) the form of some objects, determined by Mathematica and NLControl package differs from the traditional and familiar form. By this reason within NLControl package the function `BookForm` is introduced which displays such objects in a traditional form.

In NLControl syntax descriptions *sseq* denotes the `StateSpace` object, *ioeq* denotes the `IO` object and *ctrlsyst* can be either of them.

3.2 Inversive closure

In the discrete-time case application of the algebraic formalism requires the construction of the inversive closure $\mathcal{K}^*$, see Section 1.2. That is, for system (1) we have to find the variable $z(t) \subset I\!R^m$ (that is not unique) and the functions $\psi(\cdot)$ such that $(x(t), u(t)) = \psi(x(t+1), z(t))$. The function

`NegativeTimeShifts[`*ctrlsyst*`]`

finds all the possibilities $z^i(t)$ to define variable $z(t)$ for system (1) (or (3)). The function

`BackwardShiftOperator[`*ctrlsyst*`]`

computes for each $z^i(t)$ the operator $\psi^i(\cdot)$ and chooses one with the simplest expression. Simplicity is measured by Mathematica function `ByteCount`. An alternative syntax

`BackwardShiftOperator[`*ctrlsyst*`,` $z^i(t)$`]`

allows to find the backward shift operator for the particular set $z^i(t)$. Finally, the function returns the backward shift operator in the form $\{x_1(t-1) \to \psi_1(x(t), z(t-1)), \ldots, x_n(t-1) \to \psi_n(x(t), z(t-1)), u_1(t-1) \to \psi_{n+1}(x(t), z(t-1)), \ldots, u_m(t-1) \to \psi_{n+m}(x(t), z(t-1))\}$.

3.3 Sequences of subspaces

NLControl uses a special object

$$\texttt{SpanK[}\{\{a_{11}, a_{12}, \ldots\}, \{a_{21}, a_{22}, \ldots\}, \ldots\}\texttt{, } \{x_1, x_2, \ldots\}\texttt{, -1, } t\texttt{]}$$

to represent the subspace of one-forms spanned over $\mathcal{K}$ by the vectors $\sum_{j\geq 1} a_{ij}\mathrm{d}x_j$ for $i \geq 1$. Replacing the third argument by 1 gives a subspace spanned over $\mathcal{K}$ by the vector fields – a mathematical object used for instance in [18, 20]. The last argument t tells Mathematica that all symbols depending on t are to be considered as variables, even if they do not appear explicitly in the coordinate list $\{x_1, x_2, \ldots\}$.

To compute the sequence of subspaces $\{\mathcal{H}_k\}$ for the discrete- or continuous-time control system a function

$$\texttt{SequenceH[}\mathit{ctrlsyst}\texttt{,}\, n\,\texttt{]}$$

is implemented. The integer n determines how many elements in the sequence will be computed. Using keyword `All` instead of n allows to find all subspaces $\{\mathcal{H}_1, \ldots, \mathcal{H}_\infty\}$.

By default, the function applies the universal algorithm given in [4], but for special cases alternative methods also exist. The first alternative, which can be used only for discrete-time systems, is the recursive formula $\mathcal{H}_{k+1} = \delta^{-1}(\mathcal{H}_k \cap \delta\mathcal{H}_k)$ for $k \geq 1$. For that one has to add an option `Method->2` as the last argument of the function. The second alternative is based on the fact that the elements of the $\{\mathcal{H}_k\}$ can be computed as the maximal annihilators of the elements in certain sequence of distributions of the vector fields, [19, 18]. The latter method has a built-in restriction: if the subspace $\mathcal{H}_k$ is integrable and $\mathcal{H}_{k+1}$ is not, then the method can be used to compute all the subspaces up $\mathcal{H}_{k+1}$, but not any more for $\mathcal{H}_{k+2}$ and further. The method can be called by option `Method->3`. And finally, there is one more alternative for the discrete-time systems given by SISO i/o equation: this is the polynomial method described in Section 2.4, which can be called by option `Method->4`.

The integrability property of the subspace of differential one-forms can be checked by function

$$\texttt{Integrability[}\mathit{oneforms}\,\texttt{]},$$

where *oneforms* has to be object `SpanK`. Actual integration may be performed by function

$$\texttt{IntegrateOneForms[}\mathit{oneforms}\,\texttt{]}.$$

3.4 Ore polynomials

NLControl uses a special object `OreRing` to store the information about the Ore ring were polynomials belong to and about difference/differential field where polynomial coefficients belong. The easiest way to create the object `OreRing` associated with the control system, is to use the function

`DefineOreRing[∂, ` *ctrlsyst* ` ].`

Note that we use ∂ as a polynomial variable, since ∂ is reserved symbol in Mathematica. It is also possible to work with Ore rings not associated with any control system. In this case, the object `OreRing` can be created by

`DefineOreRing[∂, ` *t*`, Shift],`

if the polynomial coefficients are from difference ring, and by

`DefineOreRing[∂, ` *t*`, TimeDerivative],`

if the polynomial coefficients are from differential ring.

Additionally, we define the object representing the Ore polynomial. The special object is necessary, since Mathematica automatically reorders the factors connected with standard multiplication operator "*", for instance the expression $y[t] * ∂$ is immediately rewritten as $∂y[t]$, which is wrong for the case of Ore polynomials. Therefore, the Ore polynomial in the form (10) is represented as

`OreP[`a_n`, ...,`a_1`, `a_0`],`

where $a_n, \ldots, a_0$ are polynomial coefficients. The function

`OreSimplify[`p`, `R` ]`

simplifies the polynomial p, assuming it belongs to the Ore ring R, as described below. The argument R has to be given as the object `OreRing`. If there are relations defined between polynomial coefficients, this yields that certain expressions in $\mathcal{K}$ are equal to zero. The function `OreSimplify` applies these relations to polynomial coefficients and then simplifies the result. Note that these relations are not applied automatically, since the polynomial object `OreP` have no information about them. Addition of polynomials may performed by Mathematica standard "+" operator, but multiplication requires a special function

`OreMultiply[`p_1`,..., `p_n`, `R` ],`

which computes a product of polynomials p_1,..., p_n from the Ore ring R and is based on commutation rules (11) or (12). Let p and q be polynomials from the Ore ring R. Then the following functions can be applied to them:

- `LeftQuotient[`p`, `q`, `R` ]` finds the left quotient of p and q.
- `LeftRemainder[`p`, `q`, `R` ]` finds the left remainder of p and q.
- `LeftQuotientRemandier[`p`, `q`, `R` ]` returns a list $\{\gamma,\ r\}$, where γ is the left quotient and r is the left remainder of p and q.
- `LeftGCD[`p`, `q`, `R` ]` finds the gcld of p and q.
- `LeftLCM[`p`, `q`, `R` ]` finds the least common left multiple of p and q.

Corresponding right-side functions are also available.

And finally, there are two functions for matrices with the polynomial entries belonging to the Ore ring R. Analogously with the standard matrix multiplication function `Dot`, the function

`OreDot[`A_1`,...,` A_n`, `R` ]`

finds a product of polynomial matrices A_1,..., A_n. The function

`LowerLeftTriangularMatrix[`A`, `R`]`

transforms the rectangular polynomial matrix A into lower left triangular form.

3.5 Transformations between different system descriptions

In this subsection functions are described that allow to transform one system description into another. The function

`Reduction[`*ioeq*`]`

determines whether the system described by the i/o equations is irreducible or not, and if not, finds the reduced set of i/o equations. In the discrete-time case the function is based on Theorem 6 and in the continuous-time case an analogical result proved in [15]. The function

`Realization[`*ioeq*`, `$x_{\#}[t]\&$`]`

determines whether the i/o equation (3) (or (4)) can be transformed into the state-space form and in case of the positive answer finds the state equations. In the discrete-time case the function is based on Theorem 2 and in the continuous-time case on Theorem 3. The second argument $x_{\#}[t]\&$ is a so called pure function, which determines the state variables to be denoted as $x_1[t], x_2[t], \ldots$. Alternative way is to replace the pure function by the list of state variables, for example by $\{x1[t], x2[t]\}$, but in that case one has to be careful to choose a list with a correct length. The function

`IOToPolynomials[`*ioeq*`]`

computes the matrices $P(\partial)$ and $Q(\partial)$ in (13), given the i/o equations (3). Analogously, the function finds polynomial representation for the continuous-time system (4). The function

`ClassicStateToIO[`$sseq$`]`

finds the i/o equations (3) (or (4)) from the state equations (1) or (2), respectively. The function is based on the state elimination method given in [4]. The function

`NormalForm[`$sseq$`, `$\xi_{\#}[t]\&$`]`

transforms the system given by the state equations (1) or (2), if possible, into the normal form which is a good starting point for applying the inversion-based control algorithms and for computing zero dynamics, [12]. The argument $\xi_{\#}[t]\&$ determines the new state variables to be denoted as $\{\xi_1[t], \xi_2[t], \ldots\}$.

3.6 Checking the system properties

The function

`Accessibility[`$sseq$`]`

returns `True` if the system (1) (or (2)) is accessible and `False` otherwise. In case the system is not accessible, it can be decomposed into accessible and non-accessible subsystems by the function

`AccessiblityDecomposition[`$sseq$`, `$\xi_{\#}[t]\&$` ]`.

The function

`Observability[`$sseq$`]`

returns `True` if the system (1) (or (2)) is observable and `False` otherwise. The function

`ObservabilityDecomposition[`$sseq$`, `$\xi_{\#}[t]\&$` ]`

decomposes the state equations into the observable and un-observable part.

3.7 Feedback linearization

System (1) is said to be static state feedback linearizable if there exist a state coordinate transformation $\tilde{x}(t) = \Phi(x(t))$ and a regular static state feedback of the form $u(t) = \beta(x(t), v(t))$, with $\mathrm{rank}_{\mathcal{K}}[\partial\beta(\cdot)/\partial v] = m$, the number of system inputs, such that in the new coordinates the compensated system equations are in the form

$$\begin{array}{rcl} \tilde{x}_{i1}(t+1) & = & \tilde{x}_{i2}(t) \\ & \cdots & \\ \tilde{x}_{ik_i-1}(t+1) & = & \tilde{x}_{ik_i}(t) \\ \tilde{x}_{ik_i}(t+1) & = & v_i(t), \quad i = 1, ..., m. \end{array}$$

The necessary and sufficient conditions for feedback linearizability are:

(i) $\mathcal{H}_\infty = \{0\}$,

(ii) $\mathcal{H}_k$ is completely integrable for $1 \leq k \leq k^*$,

where k^* is an integer $k^* \leq n$ such that, for $0 \leq k \leq k^*$, $\mathcal{H}_{k+1} \subset \mathcal{H}_k$ but $\mathcal{H}_{k+1} \neq \mathcal{H}_k$ and $\mathcal{H}_{k^*+1} = \mathcal{H}_{k^*+2} = \ldots = \mathcal{H}_\infty$.

The function

`Linearization[`$sseq$`, `$\tilde{x}_\#[t]\&$`, `$v_\#[t]\&$` ]`

checks whether the system is feedback linearizable and in the case of affirmative answer finds the state coordinate change, the feedback and the linear closed-loop equations.

3.8 Examples

Example 1 Consider the system described by the i/o difference equations

$$\begin{array}{rcl} y_1(t+2) & = & y_1(t+1) - u_1(t)y_1(t) + u_1(t+1)y_1(t+1) \\ y_2(t+3) & = & y_2(t+2)u_2(t+2) + y_2(t+1) - u_1(t+1)y_1(t+1) \\ & & -3y_1(t+1) + 3u_1(t)y_1(t) + y_1(t) - u_2(t)y_2(t). \end{array} \tag{27}$$

In order to use any function from the package NLControl, one first has to load the package by the following command

```
In[1]:= <<NLControl`Master`
```

Let us create the object `IO` for this system.

```
In[2]:= eqs = {y1[t+2] → y1[t+1] - u1[t]y1[t] + u1[t+1]y1[t+1],
          y2[t+3] → y2[t+2]u2[t+2] + y2[t+1] - u1[t+1]y1[t+1] -
            3y1[t+1] + 3u1[t]y1[t] + y1[t] - u2[t]y2[t]};
        Ut = {u1[t], u2[t]};
        Yt = {y1[t], y2[t]};
        ioeq=IO[eqs, Ut, Yt, t, Shift];
        BookForm[ioeq]
        y1[t+2]  =  y1[t+1] - u1[t]y1[t] + u1[t+1]y1[t+1],
Out[6]= y2[t+3]  =  (y2[t+2]u2[t+2] + y2[t+1] - u1[t+1]y1[t+1] -
                    3y1[t+1] + 3u1[t]y1[t] + y1[t] - u2[t]y2[t]
```

Next, we construct an Ore ring associated with the system (27).

```
In[7]:= K = DefineOreRing[ð, ioeq]

Out[7]= OreRing[ð,t,{Shift, t->t+1, t->t-1, 0&},
         {{{y1[i+t], -∞ ≤i≤1}, {y2[j+t], -∞ ≤j≤2},
         {u1[k+t], 0≤k≤ ∞}, {u2[l+t], 0≤l≤ ∞}}, {«4»}}]
```

The next function finds the matrices $P(ð)$ and $Q(ð)$ for the system.

```
In[8]:= {P, Q} = FromIOToOreP[ioeq];
        MatrixForm[BookForom[P, K]]
```

$$\text{Out[9]=}\begin{pmatrix} ð^2+(-1-u_1[t+1])ð+u_1[t] & 0 \\ (3+u_1[t+1])ð+(-1-3u_1[t]) & ð^3-u_2[t+2]ð^2-ð+u_2[t] \end{pmatrix}$$

```
In[10]:= MatrixForm[BookForom[Q, K]]
```

$$\text{Out[10]=}\begin{pmatrix} y_1[t+1]ð-y_1[t] & 0 \\ -y_1[t+1]ð+3y_1[t] & y_2[t+2]ð^2-y_2[t] \end{pmatrix}$$

According to Theorem 6, we can find a gcld $G_L(ð)$ of $P(ð)$ and $Q(ð)$ by reducing the composite matrix $[P(ð)\vdots Q(ð)]$ into lower left triangular form $[G_L(ð)\vdots 0]$.

```
In[11]:= PQ = MapThread[ Join, {P, Q}];
         MatrixForm[BookForom[
           G = LowerLeftTriangularMatrix[PQ, K], K]]
```

$$\text{Out[12]=}\begin{pmatrix} y_1[t+1]ð-y_1[t] & 0 \\ -y_1[t+1]ð+3y_1[t] & ð^2-1 \end{pmatrix}$$

Since $G_L(ð)$ is not a unimodular matrix, the system (27) can be reduced; that is we can find the polynomial matrices $\tilde{P}(ð)$ and $\tilde{Q}(ð)$ defining the reduced system $\tilde{P}(ð)\mathrm{d}y(t) = \tilde{Q}(ð)\mathrm{d}u(t)$ from the equations $P(ð) = G_L(ð)\tilde{P}(ð)$ and $Q(ð) = G_L(ð)\tilde{Q}(ð)$:

```
In[13]:= MatrixForm[BookForm[P̃ = LeftQuotient[P, G, K], K]]
```

$$\text{Out[13]=}\begin{pmatrix} \frac{1}{y_1[t]}ð-\frac{u_1[t]}{y_1[t]} & 0 \\ 1 & ð-u_2[t] \end{pmatrix}$$

```
In[14]:= MatrixForm[BookForm[Q̃ = LeftQuotient[Q, G, K], K]]
```

$$\text{Out[14]=}\begin{pmatrix} 1 & 0 \\ 0 & y_2[t] \end{pmatrix}$$

For this example the equations $\tilde{P}(ð)\mathrm{d}y(t) = \tilde{Q}(ð)\mathrm{d}u(t)$ can be easily integrated to provide the reduced i/o difference equations:

```
In[15]:= BookForm[sp = SimplifyBasis[FromOrePToSpanK[P̃, Q̃, ioeq]]]
```

Out[15]= SpanK[$-u_1[t]dy_1[t] + dy_1[t+1] - y_1[t]du_1[t]$,
$dy_1[t] - u_2[t]dy_2[t] + dy_2[t+1] - y_2[t]du_2[t]$]

```
In[16]:= psinew = IntegrateOneForms[sp]
```

Out[16]= {$y_1[t] - u_2[t]y_2[t] + y_2[1+t]$, $-u_1[t]y_1[t] + y_1[1+t]$}

So, the reduced i/o equations are

```
In[17]:= BookForm[IO[Thread[psinew == 0], Ut, Yt, t, Shift]]
```

Out[17]=
$$\begin{aligned} y_1[t] - u_2[t]y_2[t] + y_2[t+1] &= 0 \\ -u_1[t]y_1[t] + y_1[t+1] &= 0 \end{aligned}$$

The above reduction procedure can be also performed by the single command **Reduction[ioeq]**.

Example 2 This example demonstrates the technique used to obtain the realizability conditions and state equations presented in **Paper II**. Consider the second order quadratic i/o equation (18) with $n = 2$.

At first, compose the matrices of coefficients $\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D}, \mathcal{E}$ and matrices of variables $\mathcal{U}, \mathcal{Y}$ for equation (18) in such way that they could be easily generalized to the systems $n > 2$.

```
In[18]:= n = 2; s = 1;
         𝒜 = Table[a_i, {i, n}];
         ℬ = Table[b_i, {i, s+1}];
         𝒞 = Table[c_10*i+j, {i, n}, {j, s+1}];
         𝒟 = Table[If[i <= j, d_10*i+j, 0], {i, n}, {j, n}];
         ℰ = Table[If[i <= j, e_10*i+j, 0], {i, s+1}, {j, s+1}];
         𝒰 = Table[Derivative[s+1-i][u][t], {i, s+1}];
         𝒴 = Table[Derivative[n-i][y][t], {i, n}];
         MatrixForm /@ {𝒜, ℬ, 𝒞, 𝒟, ℰ}
```

Out[26]= $\left\{ \begin{pmatrix} a_1 \\ a_2 \end{pmatrix}, \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}, \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix}, \begin{pmatrix} d_{11} & d_{12} \\ 0 & d_{22} \end{pmatrix}, \begin{pmatrix} e_{11} & e_{12} \\ 0 & e_{22} \end{pmatrix} \right\}$

```
In[27]:= MatrixForm /@ {𝒰, 𝒴}
```

Out[27]= $\left\{ \begin{pmatrix} u'[t] \\ u[t] \end{pmatrix}, \begin{pmatrix} y'[t] \\ y[t] \end{pmatrix} \right\}$

Next, let us create the i/o equation.

```
In[28]:= SetOptions[BookForm, TimeArgument -> False];
         eq = y^(n)[t] -> Expand[𝒜.𝒴 + ℬ.𝒰 + 𝒞.𝒰.𝒴 + 𝒟.𝒴.𝒴 + ℰ.𝒰.𝒰];
         ioeq IO[eq, u[t], y[t], t, TimeDerivative]; BookForm[ioeq]
```

Out[30]= $y'' = b_2u + e_{22}u^2 + a_2y + c_{22}uy + d_{22}y^2 + b_1u' + e_{12}uu' +$
$c_{21}yu' + e_{11}u'^2 + a_1y' + c_{12}uy' + d_{12}yy' + c_{11}u'y' + d_{11}y'^2$

According to Theorem 2, we need to compute the first three elements of the sequence $\mathcal{H}_k$ to check realizability of the system `ioeq`.

```
In[31]:= BookForm[Hk = SequenceH[ioeq, 3]]
Out[30]= {SpanK[dy, dy′, du, du′], SpanK[du, dy′, dy],
          SpanK[dy, dy′ + (-b1 - e12u - c21y - 2e11u′ - c11y′)du]}
In[32]:= Integrability /@ Hk
Out[32]= {True, True, False}
```

The subspace $\mathcal{H}_3$ is not integrable, thus the system *ioeq* has no classical state space form. In order to find the restrictions on the coefficients we have to compute the wedge products in Frobenius' theorem. For that, convert $\mathcal{H}_3$ from object `SpanK` to the set of one-forms and denote the one-forms by ω_1 and ω_2.

```
In[33]:= BookForm[{ω1, ω2} = ToDifferential[Hk[[3]]]]
Out[33]= {dy, dy′ + (-b1 - e12u - c21y - 2e11u′ - c11y′)du}
```

It is clear $\mathrm{d}\omega_1 \wedge \omega_1 \wedge \omega_2 = 0$, since $\mathrm{d}\omega_1 = \mathrm{d}(\mathrm{d}y) = 0$. Next, compute $\mathrm{d}\omega_2 \wedge \omega_1 \wedge \omega_2$:

```
In[34]:= coords = Hk[[3, 2]]
Out[34]= {y[t], y′[t], u[t], u′[t], u″[t]}
In[35]:= BookForm[
           Wedge[De[ω2, Coordinates -> coords], Wedge @@ {ω1, ω2}]]
Out[35]= -2e11du ∧ dy ∧ du′ ∧ dy′
```

Above wedge product is zero iff $e_{11} = 0$. So, taking $e_{11} = 0$ in equation `ioeq` gives us a new realizable system `ioeq1`.

```
In[36]:= ioeq1 = ioeq/.e11 -> 0;
         BookForm[Hk = SequenceH[ioeq1, 3]]
Out[37]= {SpanK[dy, dy′, du, du′], SpanK[du, dy′, dy],
          SpanK[dy, dy′ + (-b1 - e12u - c21y - c11y′)du]}
```

Now the subspace $\mathcal{H}_3$ is integrable and according to Theorem 2, by integrating the basis vectors of $\mathcal{H}_3$ gives us the state coordinates for realization.

```
In[38]:= BookForm[states = IntegrateOneForms[Hk[[3]]], t]
```

$$\text{Out[38]= } \left\{y,\ \frac{e^{-c_{11}u}\,(b_1c_{11} + e_{12} + c_{11}\,(e_{12}u + c_{21}y) + c_{11}^2y')}{c_{11}^2}\right\}$$

Finally, we can find the state equations for system *ioeq1*. To make the result visually shorter, denote the expression $b_1c_{11} + e_{12}\,(1 + c_{11}u[t]) + c_{11}c_{21}x_1[t] - e^{c_{11}u[t]}c_{11}^2x_2[t]$ in state equations by A.

In[39]:= **{cls, repl} = Realization[ioeq, $x_{\#}$[t]&, states];**
BookForm[cls/. ($b_1 c_{11} + e_{12}$ (1 + c_{11}u[t]) +
c_{11} ($c_{21} x_1$[t] - $e^{c_{11} u[t]} c_{11} x_2$[t])) -> A]

Out[40]=

$$x'_1 = -A/c_{11}^2$$

$$\begin{aligned} x'_2 = {} & e^{-c_{11}u}(-A\ a_1 c_{11}^2 - A\ c_{11}c_{21} + A^2 d_{11} - A c_{11}^2(c_{12}u + d_{12}x_1) + \\ & c_{11}^3(-A + e_{12})u' - e^{c_{11}u}c_{11}^5 x_2 u' + \\ & c_{11}^4(b_2 u + e_{22}u^2 + a_2 x_1 + \\ & c_{22}u\ x_1 + d_{22}x_1^2 + b_1 u' + e_{12}u\ u' + c_{21}x_1 u'))/\ c_{11}^4 \end{aligned}$$

3.9 WebMathematica application

The functions from the package NLControl cannot be used outside the Mathematica environment. In order to overcome this limitation we have developed a webMathematica-based application, that allows certain functions from NLControl to be used on the world-wide-web, in such a way that no other software except for an internet browser needs to be installed in a computer. This allows these tools to be applied in graduate courses as well as to make them available to a wider control community and to engineers.

However, the NLControl website has several restrictions when compared with NLControl running within standard Mathematica. First, only the most important NLControl functions are available on the website. Currently there are 15 functions for discrete-time systems, 13 functions for continuous-time systems and 3 functions for the systems defined on homogeneous time-scales. Second, the website automatically interrupts all the computations lasting longer than 30 seconds. The reason is there is only one Mathematica kernel running in the server, which is shared between different users. Third, equations entered in to the website should be in plain text form, which means that instead of subscripted variables $x_1, x_2, \ldots$ one has to use $x1, x2, \ldots$ and entering Greek letters is uncomfortable, since for example instead of α one should type `\[Alpha]`.

The NLControl website is built of jsp (Java Server Pages) files, which contain elements of several programming languages: Javascript is used for opening and closing windows and communicating between different windows, Java is used for choosing random examples from examples library and finally Mathematica together with webMathematica functions is used to generate the result. The file structure of the website is simple. For every function there is one jsp file, individual for discrete- and continuous-time systems, which allows to enter the input data using html forms. Input data is sent to the other jsp file, so called result file, which sends the data to Mathematica kernel and after getting the answer prints it. Most of the functions use the same result file for discrete- and continuous-time systems, since it allows to decrease the number of files.

The developed webpage is available at
http://webmathematica.cc.ioc.ee/webmathematica/NLControl/.

Conclusions

In present thesis three modeling (realization-related) problems of nonlinear control systems are solved using algebraic approach of differential forms.

Necessary and sufficient condition for the irreducibility of MIMO difference equations, formulated in terms of the greatest common left divisor of the polynomial matrices associated with the system equations, is presented. The condition remarkably coincides with the corresponding result for the linear systems, yet in the nonlinear case polynomials are defined over the difference field, and unlike the linear case, belong to a noncommutative polynomial ring. Analogous irreducibility conditions are also valid for differential equations [15].

Noncommutative polynomials also turned out to be useful for simplifying the solution of the realization problem. A new formula is presented in terms of Ore polynomials, which allows to compute the basis of the subspace $\mathcal{H}_{\alpha+2}$ of differential one-forms that determines the differentials of the state coordinates for SISO difference equation. Note that in the linear case, our results yield the results given in [26]. The main advantage of the polynomial method is shorter computation time and compact program code, if compared to the algebraic method, based on the solution of the set of nonlinear equations. The polynomial method of computing $\mathcal{H}_{\alpha+2}$ has been also extended for differential equation [30].

The third problem studied in this thesis is the realizability of a class of higher order quadratic i/o differential equations. For the second- and third-order models sufficient and necessary conditions, in terms of restrictions on the quadratic model parameters, have been provided to establish whether it is possible to find a state space representation of the i/o system or not. The computer algebra package NLControl was used to obtain realizability conditions for the second- and third-order quadratic equations and realizable subclasses for the n-th order equations. For higher order quadratic equations two realizable subclasses were suggested. Note also that earlier results do not suggest explicit state equations for i/o models. In this thesis we provide explicit state equations for realizable second order quadratic i/o equations and for one realizable subclass of quadratic i/o equations for the general case of arbitrary order n. Future research will be directed towards the development of general model classes, other than bilinear and quadratic, which can be put into the state space form, and capture the basic non-linearities of the plants whilst remaining within limited complexity.

Additionally to above theoretical problems the Mathematica based software package NLControl for nonlinear control systems and its webMathematica application is described. Not all the tasks that can be solved by NLControl are included into this thesis. For instance, reduction and realization problem can be also solved for systems defined on homogeneous time scales [2] and realization of discrete-time composite systems is possible [23]. NLControl also provides an alternative approach to differential one-forms – vector fields, which allow to solve several control problems [20, 19, 18] and are only briefly mentioned here. Our future goal is to implement more NLControl functions into webMathematica website, improve the documentation and example library. Especially, we want to include functions that allow output feedback linearization and/or decoupling, construct the transfer function from the i/o equations or from the state equations, find the discrete-time model from the continuous-time system equations and solve the model-matching problem.

References

[1] E. Aranda-Bricaire, Ü. Kotta, and C.H. Moog. Linearization of discrete-time systems. In *SIAM Journal on Control and Optimization*, volume 34, pages 1999–2023. 1996.

[2] D. Casagrande, Ü. Kotta, **M. Tõnso**, and M. Wyrwas. Transfer equivalence and realization of nonlinear input-output delta-differential equations on homogeneous time scales. *IEEE Transactions on Automatic Control, Submitted for publication.*

[3] R.M. Cohn. *Difference algebra.* Wiley-Interscience, New York, 1965.

[4] G. Conte, C. Moog, and A. Perdon. *Algebraic Methods for Nonlinear Control Systems.* Springer-Verlag, London, 2007.

[5] P. Crouch, F. Lamnabhi-Lagarrigue, and D. Pinchon. A realisation algorithm for input-output systems. *International Journal of Control*, 62: 941–960, 1995.

[6] B. de Jager. The use of symbolic computations in nonlinear control. is it viable? In *IEEE Transactions on Automatic Control*, volume 40, pages 84–89, 1995.

[7] M. Fliess. Automatique en temps discret et algèbre aux différences. In *Forum Mathematicum*, volume 2, pages 213–232, 1990.

[8] M. Freedman and J. Willems. Smooth representation of systems with differentiated inputs. *IEEE Transactions on Automatic Control*, 23:16–21, 1978.

[9] J.W. Grizzle. A linear algebraic framework for the analysis of discrete-time nonlinear systems. *SIAM Journal on Control and Optimization*, 31: 1026–1044, 1993.

[10] A. Isidori. Nonlinear control systems (3rd ed.). Berlins, 1995. Springer.

[11] A. Kaddouri, S. Blais, M. Ghribi, and O. Akhrif. Nlsoft: An interactive graphical software for desiging nonlinear controllers. In *Mathematics and Computers in Simulation*, volume 71, pages 377–384, 2006.

[12] Ü. Kotta. Comments on “on the discrete-time normal form”. In *IEEE Transactions on Automatic Control*, volume 45(11), page 2197, 2000.

[13] Ü. Kotta. Irreducibility conditions for nonlinear input-output difference equations. In *Proc. of the 39th IEEE Conf. on Decision and Control*, pages 3404–3408. Sydney, 2000.

[14] Ü. Kotta, A.S.I. Zinober, and P. Liu. Transfer equivalence and realization of nonlinear higher order input-output difference equations. *Automatica*, 37:1771–1778, 2001.

[15] Ü. Kotta, P. Kotta, S. Nõmm, and **M. Tõnso**. Irreducibility conditions for continuous-time multi-input multi-output nonlinear systems. In *2006 9th International Conference on Control, Automation, Robotics and Vision*, pages 807–811. Singapore, 2006.

[16] Ü. Kotta, T. Mullari, A.S.I Zinober, and P. Kotta. State space realisation of bilinear continuous-time input-output equtions. *International Journal of Control*, 80:1607–1615, 2007.

[17] J.C. McConnel and J.C. Robson. *Noncommutative Noetherian Rings.* Birkhäuser, 1987.

[18] T. Mullari, Ü. Kotta, S. Nõmm, and **M. Tõnso**. Realization of nonlinear composite systems. *Control and Cybernetics*, 35(4):905–922, 2006.

[19] T. Mullari, Ü. Kotta, and **M. Tõnso**. The connection between different static state feedback linearizability conditions of discrete time nonlinear control systems. In *European Control Conference*, pages 4268–4275. Kos, Greece, 2007.

[20] T. Mullari, Ü. Kotta, P. Kotta, **M. Tõnso**, and A.S.I. Zinober. Removing the input derivatives in the generalized bilinear state equations. *Proceedings of the Estonian Academy of Sciences*, 58(2):98–107, 2009.

[21] D. Müller. *Gröbnerbasen in OreAlgebren. Eine Implementation zum Arbeiten mit Ore-Algebren und die Untersuchung des Gröbner-Walks als Anwendung.* PhD thesis, Universität Kassel, 2006.

[22] H. Nijmeijer and A. van der Schaft. Nonlinear dynamical control systems. New York, 1990. Springer.

[23] S. Nõmm, Ü. Kotta, and **M. Tõnso**. Realization of nonlinear discrete-time composite systems: computational aspects. In *Proceedings of the 16th IFAC World Congress*, pages 139–144, Prague, Czech Republic, 2006.

[24] M. Ondera. *Computer-aided design of nonlinear systems and their generalized transfer functions.* PhD thesis, Slovak University of Technology in Bratislava, FEI, 2008.

[25] O. Ore. Theory of non-commutative polynomials. *Annals of Mathematics*, 34:480–508, 1933.

[26] P. Rapisarda and J.C. Willems. State maps for linear systems. *SIAM J. Control. Optim.*, 35(3):1053–1091, 1997.

[27] J. Rodrigues-Millan. Integrated symbolic-graphic-numeric analysis and design in nonlinear control through notebooks in mathematica. In *Lecure Notes in Computer Science*, pages 405–420. Springer-Verlag, Berlin, 2001.

[28] R. Rothfuß and M. Zeitz. A toolbox for symbolic nonlinear feedback design. In *Proc. of the 13th IFAC World Congress*, pages 283–288, San Francisco, 1996.

[29] G. Sidhu and G. Franklin. Technique for analog computer realisation and choice of state variables for a class of nonlinear systems. *IEEE Transactions on Automatic Control*, 17:718–720, 1972.

[30] **M. Tõnso** and Ü. Kotta. Realization of continuoustime nonlinear inputoutput equations: polynomial approach. In *Computer Aided Systems Theory - Eurocast 2009, Lecure Notes in Computer Science*, pages 633–640. Springer Berlin/Heidelberg, 2009.

[31] Y. Zheng, J.C. Willems, and C. Zhang. A polynomial approach to nonlinear system controllability. In *IEEE Transaction on Automatic Control*, volume 46, pages 1782–1788. 2001.

List of publications

The present thesis is based on five academic papers, which are referred to in the text by their Roman numerals I-V:

I Ü. Kotta, **M. Tõnso**. Irreducibility conditions for discrete-time nonlinear multi-input multi-output systems. - In: *Proc. 6th IFAC Symposium on Nonlinear Control Systems 2004 (NOLCOS 2004)*, Stuttgart, Germany, 2005, 255–260.

II Ü. Kotta, P. Kotta, **M. Tõnso**, A.S.I. Zinober. On classical state space realizability of quadratic input-output differential equations. *International Journal of Control*, 2009, **82**, 7, 1212-1218.

III Ü. Kotta, **M. Tõnso**. Realization of discrete-time nonlinear input-output equations: polynomial approach. - In: *The 7th World Congress on Intelligent Control and Automation : WCICA'08 Proceedings and Digest Book* : June 25 - 27, 2008, Chongqing, China / Eds. P. B. Luh [et al.] : IEEE, 2008, 529-534.

IV Ü. Kotta, **M. Tõnso**. Linear algebraic tools for discrete-time nonlinear control systems with Mathematica . - In: *Nonlinear and Adaptive Control*, NCN4 2001 / Eds. A.Zinober, D.Owens. Berlin [etc.] : Springer, 2003, 195-205. (Lecture Notes in Control and Information Sciences; 281).

V **M. Tõnso**, H. Rennik, Ü. Kotta. WebMathematica based tools for discrete-time nonlinear control systems. *Proceedings of the Estonian Academy of Sciences*, 2009, 4.

In the appendix of the thesis, the copies of the papers I-V are included. The author has co-written numerous publications closely related to the subject of the thesis. Papers I-V were selected for defence by the following two principles. First, the contribution of the author in these papers was the largest and second, the discrete-time case as more complicated was preferred.

Author's contribution

Paper I Reduction problem was solved by Ü. Kotta for SISO systems. The solution was generalized to MIMO systems in collaboration with Ü. Kotta.

Paper II The contribution of Ü. Kotta, P. Kotta and M. Tõnso is approximately equal. To obtain the results described in this paper, Ü. Kotta used theoretical methods while P. Kotta used Mathematica programs developed by the author. The state equations were derived by M. Tõnso.

Paper III The results presented in this paper are mostly obtained by the author.

Paper IV At the time the paper was published, the author was the only developer of the NLControl package.

Paper V NLControl website has been developed by the author in collaboration with Heli Rennik, who was a Master student under the author's supervision. Observability-related functions mentioned in this paper were written by Vadim Kaparin, who was also a Master student under the author's supervision.

Resümee

Käesoleva dissertatsiooni teoreetilises osas lahendatakes kolm mittelineaarsete juhtimissüsteemide modelleerimisülesannet, kasutades selleks diferentsiaalvormidel ja mittekommutatiivsetel polünoomidel baseeruvat algebralist formalismi. Esiteks, on leitud mitme mitme sisendi ja mitme väljundiga diskreetse mittelineaarse juhtimissüsteemi taandumatuse tarvilikud ja piisavad tingimused. Need tingimused, mis on formuleeritud süsteemiga seotud mittekommutatiivsete polünoomide maatriksite suurima ühisteguri abil, annavad ka meetodi süsteemi taandamiseks. Teiseks vaadeldakse sisend-väljund võrrandiga kirjeldatud ruutsüsteemide realiseeritavust. Teist ja kolmandat järku mudelite joaks on leitud realiseeritavuse tarvilikud ja piisavad tingimused vahetult ruutsüsteemi parameetrites. Kõrgemat järku mudelite jaoks on välja toodud kaks realiseeritavat alamklassi. Kolmandaks on leitud meetod realisatsiooniülesande lahendamiseks mittekommutatiivsete polünoomide teooria abil ühe sisendi ja ühe väljundiga diskreetse mittelineaarse juhtimissüsteemi jaoks.

Töö teine osa kirjeldab sümbolarvutuse pgorammi NLControl, mis võimaldab lahendada mittelineaarsete juhtimissüsteemide modelleerimise, analüüsi ja sünteesiga seotud ülesnadeid. See tarkvara baseerub samuti diferentsiaalvormide ja mittekommutatiivsete poünoomide algebralisel formalismil ja on arendatud tarkvarapaketi Mathematica keskkonnas. Lühidalt on kirjeldatud NLControli veebisaiti, mis võimaldab kasutada NLControli tähtsamaid funktsioone interneti vahendusel, ilma Mathematicat lokaalsesse arvutisse installeerimata.

Abstract

In the theoretical part of this thesis algebraic approach of differential forms, supplemented by skew polynomial formalism, is applied to three modelling problems of nonlinear control systems. First, necessary and sufficient irreducibility condition for multi-input multi-output discrete-time control system is presented. The condition is formulated in terms of the greatest common left divisor of the skew polynomial matrices associated with the system equations and it also provides a natural method for system reduction. Second, realizability of the class of higher order quadratic input-output differential equation is examined. For the second- and third-order models necessary and sufficient realizability conditions in terms of the quadratic model parameters have been provided. For higher order quadratic equations two realizable subclasses were suggested. Third, the realization problem for single-input single-output difference equation is solved using the theory of skew polynomials.

The second part of the thesis is devoted to symbolic computer algebra package NLControl, which allows to solve numerous problems related to modelling, analysis and synthesis of nonlinear control systems, both continuous and discrete-time case. The package is based on the algebraic methods of differential one-forms and skew polynomials and it is created within computer algebra system Mathematica. A brief exposition of the NLControl website, which allows to perform computations with more important NLControl functions over the internet, without having Mathematica installed into local computer, is also given.

Elulookirjeldus

1. Isikuandmed

Ees- ja perekonnanimi	Maris Tõnso
Sünniaeg ja -koht	13.12.1971, Tallinn
Kodakondsus	Eesti
Perekonnaseis	Abielus
Lapsed	2 poega ja 1 tütar

2. Kontaktandmed

Aadress	Oksa 10-7, Tallinn, 11316
Telefon	+372 5645 3123
E-posti aadress	maris@cc.ioc.ee

3. Hariduskäik

Õppeasutus (nimetus lõpetamise ajal)	Lõpetamise aeg	Haridus (eriala/kraad)
Tallinna Pedagoogikaülikool	1997	matemaatika ja informaatika õpetaja, B.Sc.
Tallinna Pedagoogikaülikool	2000	matemaatika, M.Sc.

4. Keelteoskus

   Eesti keel – kõrgtase

   Inglise keel – kesktase

   Vene keel – algtase

5. Täiendõpe

   —

6. Teenistuskäik

Töötamise aeg	Tööandja nimetus	Ametikoht
1997 – 2001	TTÜ Küberneetika Instituut	insener
2001 – . . .	TTÜ Küberneetika Instituut	teadur

7. Teadustegevus

   Ajakirja- ja konverentisartiklite loetelu on toodud ingliskeelse CV juures.

8. Kaitstud lõputööd

   Kuidas kasutada Mathematicat, B.Sc., Tallinna Pedagoogikaülikool, 1997.

Sümbolarvutuse võimalused mittelineaarsete juhtimissüsteemide modelleerimisel "Mathematica" baasil, M.Sc., Tallinna Pedagoogikaülikool, 2000.

9. Teadustöö põhisuunad

   Mittelineaarsete juhtimissüsteemide teooria, Mathematica programmeerimine.

10. Teised uurimisprojektid

    Eesti-Poola ühisprojekt Poola Teaduste Akadeemia ja Eesti Teaduste Akadeemia vahelise teadusliku koostöö lepingu raames "Ajaskaaladel defineeritud mittelineaarsete juhtimissteemide ekvivalents ja taandamine" (2007 - 2009).

Curriculum vitae

1. Personal data

Name	Maris Tõnso
Date and place of birth	13.12.1971, Tallinn
Citizenship	Estonia
Marital status	Married
Children	2 sons and 1 daughter

2. Kontaktandmed

Aaddress	Oksa 10-7, Tallinn, 11316
Phone	+372 5645 3123
E-posti aadress	maris@cc.ioc.ee

3. Hariduskäik

Educational institution	Graduation year	Education (field of study/degree)
Tallinn Pedagogical University	1997	mathematics and informatics teacher, B.Sc.
Tallinn Pedagogical University	2000	mathematics, M.Sc.

4. Language competence

   Estonian – fluent

   English – average

   Russian – basic skills

5. Special courses

   —

6. Professional Employment

Period	Organization	Position
1997 – 2001	TUT Institute of Cybernetics	Engineer
2001 – ...	TUT Institute of Cybernetics	Researcher

7. Scientific work

   Journal articles:

   D. Casagrande, Ü. Kotta, M. Tõnso, and M. Wyrwas. Transfer equivalence and realization of nonlinear input-output delta-differential equations on homogeneous time scales. *IEEE Transactions on Automatic Control, Submitted for publication.*

Ü. Kotta and M. Tõnso. Removing or lowering the orders of input shifts in discrete-time generalized state-space systems with mathematica. *Proceedings of the Estonian Academy of Sciences*, 51(4):238–254, 2002.

Ü. Kotta and M. Tõnso. Linear algebraic tools for discrete-time nonlinear control systems with mathematica. Nonlinear and Adaptive Control, NCN4 2001 : Lecture Notes in Control and Information Sciences, 281:195–205, 2003.

Ü. Kotta, P. Kotta, M. Tõnso, and A.S.I. Zinober. On classical state space realisability of quadratic input-output differential equations. *International Journal of Control*, 82:1212–1218, 2009.

T. Mullari, Ü. Kotta, S. Nõmm, and M. Tõnso. Realization of nonlinear composite systems. *Control and Cybernetics*, 35(4):905–922, 2006.

T. Mullari, Ü. Kotta, P. Kotta, M. Tõnso, and A.S.I. Zinober. Removing the input derivatives in the generalized bilinear state equations. *Proceedings of the Estonian Academy of Sciences*, 58(2):98–107, 2009.

M. Tõnso and Ü. Kotta. Realization of continuous-time nonlinear input-output equations: polynomial approach. In *Computer Aided Systems Theory - Eurocast 2009, Lecure Notes in Computer Science*, pages 633–640. Springer Berlin/Heidelberg, 2009.

Conference papers:

J. Belikov, Ü. Kotta, T. Mullari, S. Nõmm, and M. Tõnso. Discretization of continuous-time nonlinear control systems with computer algebra system mathematica. In *Medzinárodná konferencia Kybernetika a Informatika SSKI, conference preprints*, page 10. Ždiar, Slovakia, 2008.

D. Casagrande, Ü. Kotta, M. Wyrwas, and M. Tõnso. Transfer equivalence and reduction of nonlinear delta differential equations on homogeneous time scale. In *American Control Conference*, pages 5136–5141, Seattle, Washington, USA, 2008.

V. Kaparin, Ü. Kotta, T. Mullari, and M. Tõnso. Transformation of nonlinear control system into observer form with computer algebra system mathematica. In M*edzinárodná konferencia Kybernetika a Informatika SSKI, conference preprints*, page 10. Ždiar, Slovakia, 2008.

Ü. Kotta and M. Tõnso. Irreducibility conditions for discrete-time nonlinear multi-input multi-output systems. In *Nonlinear Control Systems 2004 (NOLCOS 2004): A Proceedings volume from the 6th IFAC Symposium*, pages 255–260. Suttgart, Germany, 2005.

Ü. Kotta and M. Tõnso. On classical state space realizability of quadratic input-output differential equations. In *SSSC07: Preprints of the 3rd*

IFAC Symposium on System, Structure and Control, pages 1–6. Foz do Iguassu, Brazil, 2007.

Ü. Kotta and M. Tõnso. Realization of discrete-time nonlinear input-output equations: polynomial approach. In *The 7th World Congress on Intelligent Control and Automation : WCICA'08 Proceedings and Digest Book*, pages 529–534. Chongqing, China, 2008.

Ü. Kotta, P. Kotta, S. Nõmm, and M. Tõnso. Irreducibility conditions for continuous-time multi-input multi-output nonlinear systems. In *9th International Conference on Control, Automation, Robotics and Vision*, pages 807–811. Singapore, 2006.

C.H. Moog, Ü. Kotta, S. Nõmm, and M. Tõnso. Extensions of linear algebraic methods to nonlinear systems: an educational perspective. In *Advances in Control Education 2003 : A proceedings volume from the 6th IFAC Symposium*, pages 153–158. Oulu, Finland, 2004.

T. Mullari, Ü. Kotta, P. Kotta, M. Tõnso, and A.S.I. Zinober. Generalized state equations for continuous-time bilinear input-output equations: removing the input derivatives. In *NOLCOS 2007 : 7th IFAC Symposium on Nonlinear Control Systems, preprints*, pages 672–677. Pretoria, South Africa, 2007.

T. Mullari, Ü. Kotta, and M. Tõnso. The connection between different static state feedback linearizability conditions of discrete time nonlinear control systems. In *European Control Conference*, pages 4268–4275. Kos, Greece, 2007.

S. Nõmm, C.H. Moog, Ü. Kotta, and M. Tõnso. Realization of nonlinear discrete-time composite systems. In *System, structure and control 2004 : a proceedings volume from the 2nd IFAC Symposium*, pages 663–668. Oaxaca, Mexico, 2005.

S. Nõmm, Ü. Kotta, and M. Tõnso. Realization of nonlinear discrete-time composite systems: computational aspects. In *Proceedings of the 16th IFAC World Congress*, pages 139–144, Prague, Czech Republic, 2006.

S. Nõmm, Ü. Kotta, V. Kaparin, M. Tõnso, and J. Popovitš. Observability and identifiability of nonlinear control systems with computer algebra systems mathematica and maxima. In *Proceedings 16th International Conference Process Control 2007*, pages 139–1–139–7. Štrbské Pleso, Slovakia, 2007.

M. Tõnso, H. Rennik, J. Belikov, Ü. Kotta. WebMathematica Based Tools for Continuous-Time Nonlinear Control Systems. In *Proceedings*

of the 17th International Conference on Process Control 09, 625-633, Štrbské Pleso, Slovakia, 2009

8. Defended thesis

   How to use Mathematica, B.Sc., Tallinn Pedagogical University, 1997.

   Possibilities of Symbolic Computation in Modelling of Nonlinear Control Systems on the basis of Mathematica, M.Sc., Tallinn Pedagogical University, 2000.

9. Main areas of scientific work/Current research topics

   Nonlinear control systems, Mathematica programming.

10. Other research projects

    Estonian-Polish Joint Research Project under the Agreement on Scientific Cooperation between the Polish Academy of Sciences and the Estonian Academy of Sciences "Equivalence and reducibility of nonlinear control systems on time scales" (2007 - 2009).

Appendix