Relationship between two polynomial realization methods

Ü. Kotta M. Tõnso

Institute of Cybernetics at TUT, Akadeemia tee 21, 12618 Tallinn, Estonia. fax : +372 620 4151
e-mails : {kotta, maris}@cc.ioc.ee

Abstract: The aim of the paper is to show that two different polynomial realization methods, one of them based on adjoint polynomials and the other on the polynomial quotients, are equivalent. It is proved that both methods provide exactly the same set of basis vectors of the subspace determining the differentials for the state coordinates.

Keywords: nonlinear control system, state space realization, non-commutative polynomials, adjoint polynomials

1. INTRODUCTION

Identification of a nonlinear system provides a mathematical model of the system in the form of input-output (i/o) differential equation. At the same time, most of the control system theory is developed for the systems represented by the state equations. Thus, it is necessary to bring the obtained i/o equation into the state-space form. This task is called realization of the system. For nonlinear systems realization is a sophisticated problem; note that every input-output equation does not necessarily admit a state-space representation. There are several (equivalent) necessary and sufficient realizability conditions available in the literature. Among them the most known are algebraic conditions formulated in terms of integrability of the subspaces of differential one-forms, see Conte et al. (2007) and those in terms of Lie brackets by Delaleau and Respondek (1995). Recently, the theory of the noncommutative polynomial rings was applied to the realization problem. The polynomial approach is built upon the approach of differential one-forms and is most efficient from the computational point of view. The aim of the present paper is to find relations between the basis one-forms of the subspaces used to find differentials of the state coordinates in Halás and Kotta (2009) and Tõnso and Kotta (2009).

2. PROBLEM STATEMENT AND ALGEBRAIC FRAMEWORK

Consider a single-output nonlinear system, described by a higher order i/o differential equation, relating the input $u = [u_1, \dots, u_m]^T$, the output y and a finite number of their time derivatives

$$y^{(n)} = \phi(y, \dot{y}, \dots, y^{(n-1)}, u, \dot{u}, \dots, u^{(n-1)}). \tag{1}$$

In (1) $u \in U \subset \mathbb{R}^m$, $y \in Y \subset \mathbb{R}$, $t \geqslant 0$ and ϕ is a real analytic function.

The realization problem is defined as follows. Given a nonlinear system, described by the i/o equation of the form (1), find, if possible, the state coordinates $x \in \mathbb{R}^n$, $x = \psi(y, \dots, y^{(n-1)}, u, \dots, u^{(n-1)})$ such that in these coordinates the system takes the classical state space form

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

and the sequences $\{u, y, t \geqslant 0\}$, generated by (2) (for different initial states), coincide with the sequences $\{u, y, t \geqslant 0\}$, satisfying equations (1). Then (2) is called *a realization of (1).* A system (1) is said to be *realizable* if for it exists a realization of the form (2).

Below we briefly recall the algebraic formalism, described in Conte et al. (2007). Let $\mathcal{K}$ denote the field of meromorphic functions in a finite number of the independent system variables $\{y, \dots, y^{(n-1)}, u^{(k)},\ k \geq 0\}$ and $s : \mathcal{K} \to \mathcal{K}$ denote the time derivative operator $\mathrm{d}/\mathrm{d}t$. Then the pair $(\mathcal{K}, s)$ is a differential field Kolchin (1973). Over the field $\mathcal{K}$ one can define a differential vector space, $\mathcal{E} := \mathrm{span}_{\mathcal{K}}\{\mathrm{d}\varphi \mid \varphi \in \mathcal{K}\}$ spanned by the differentials of the elements of $\mathcal{K}$. Consider a one-form $\omega \in \mathcal{E} : \omega = \sum_i \alpha_i \mathrm{d}\varphi_i$, $\alpha_i, \varphi_i \in \mathcal{K}$. Its derivative $\dot{\omega}$ is defined by $\dot{\omega} = \sum_i \dot{\alpha}_i \mathrm{d}\varphi_i + \alpha_i \mathrm{d}\dot{\varphi}_i$.

3. POLYNOMIAL FRAMEWORK

Polynomial framework is built upon the linear algebraic framework. The differential field $(\mathcal{K}, s)$ induces a ring of left polynomials $\mathcal{K}[Z, s]$. The elements of $\mathcal{K}[Z, s]$ can be uniquely written in the form

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

where Z is a polynomial indeterminate and $a(Z) \neq 0$ if and only if at least one of the functions a_i, $i = 0, \dots, n$ is nonzero. If $a_0 \not\equiv 0$, then the positive integer n is called the degree of the polynomial $a(Z)$ and denoted by $\deg a(Z)$. In addition, we set $\deg 0 = -\infty$. For $a \in \mathcal{K}$ let us define the multiplication

$$Z \cdot a = a \cdot Z + s(a). \tag{3}$$

If the multiplication is defined by (3), the ring $\mathcal{K}[Z, s]$ is proved to satisfy left Ore condition McConnel and Robson (1987), and $Z^n \cdot a \in \mathcal{K}[Z, s]$, for $n \geq 1$, and

$$Z^n \cdot a = \sum_{i=0}^{n} C_n^i s^{n-i}(a) Z^i.$$

A ring D is called an integral domain, if it does not contain any zero divisors. This means that if a and b are two elements of D such that $ab = 0$, then $a = 0$ or $b = 0$.

Lemma 1. McConnel and Robson (1987)

(i) The ring $\mathcal{K}[Z, s]$ is an integral domain.
(ii) If a and b are nonzero left polynomials, then $\deg(a\,b) = \deg a + \deg b$.

Definition 2. Abramov et al. (2005) The adjoint of a Ore polynomial ring $\mathcal{K}[Z, s]$ is defined as the Ore polynomial ring $\mathcal{K}[Z^*, s^*]$, where $s^* = -s$.

From the definition it follows that in the adjoint polynomial ring multiplication is defined by the commutation rule $Z^* \cdot a = a \cdot Z^* - \dot{a}$, where $a \in \mathcal{K}$. If

$$p(Z) = p_n Z^n + \ldots + p_1 Z + p_0 \tag{4}$$

is a polynomial in $\mathcal{K}[Z, s]$ then the adjoint polynomial $p^*(Z^*)$ is defined by the formula

$$p^*(Z^*) = Z^{*n} p_n + \ldots + Z^* p_1 + p_0 \in \mathcal{K}[Z^*, s^*], \tag{5}$$

where the products $Z^{*i} p_i$ must be computed in $\mathcal{K}[Z^*, s^*]$.

For $\Phi \in \mathcal{K}$ we define $\mathrm{d} : \mathcal{K} \to \mathcal{E}$ as follows:

$$\mathrm{d}\Phi := \sum_{i=0}^{n-1} \frac{\partial \Phi}{\partial y^{(i)}} \mathrm{d}y^{(i)} + \sum_{j=1}^{m} \sum_{l=0}^{k} \frac{\partial \Phi}{\partial u_j^{(l)}} \mathrm{d}u_j^{(l)}.$$

$\mathrm{d}\Phi$ is said to be the total differential (or simply the differential) of the function Φ and it is a differential one-form. It is proved in Conte et al. (2007) that $s(\mathrm{d}\Phi) = \mathrm{d}(s\Phi)$. Let us define $Z^k \mathrm{d}y := \mathrm{d}(s^k y)$ and $Z^l \mathrm{d}u := \mathrm{d}(s^l u)$, for $k, l \geq 0$ in the vector space $\mathcal{E}$. Since every one-form $\omega \in \mathcal{E}$ has the form

$$\omega = \sum_{i=0}^{n-1} a_i \mathrm{d}y^{(i)} + \sum_{j=1}^{m} \sum_{l=0}^{k} b_{j,l} \mathrm{d}u_j^{(l)},$$

where $a_i, b_j \in \mathcal{K}$, so ω can be expressed in terms of the left polynomials

$$\omega = \left(\sum_{i=0}^{n-1} a_i Z^i \right) \mathrm{d}y + \sum_{j=1}^{m} \left(\sum_{l=0}^{k} b_{j,l} Z^l \right) \mathrm{d}u_j.$$

A polynomial can be considered as an operator acting on the elements of $\mathcal{E}$:

$$\left(\sum_{i=0}^{k} a_i Z^i \right) (\alpha \mathrm{d}\nu) := \sum_{i=0}^{k} a_i \left(Z^i \cdot \alpha \right) \mathrm{d}\nu,$$

with $a_i, \alpha \in \mathcal{K}$ and $\mathrm{d}\nu \in \{\mathrm{d}y, \mathrm{d}u_1, \ldots, \mathrm{d}u_m\}$. It is easy to notice that $Z(\omega) = s(\omega)$, for $\omega \in \mathcal{E}$. Additionally, using the induction principle, one can show that $Z^n(\mathrm{d}\Phi) = \mathrm{d}(s^n \Phi)$.

Instead of working with equation (1), describing the control system, we can work with its differential

$$\mathrm{d}y^{(n)} - \sum_{i=0}^{n-1} \frac{\partial \phi}{\partial y^{(i)}} \mathrm{d}y^{(i)} = \sum_{j=1}^{m} \sum_{i=0}^{n-1} \frac{\partial \phi}{\partial u_j^{(i)}} \mathrm{d}u_j^{(i)} \tag{6}$$

that can be rewritten as

$$a(Z)\mathrm{d}y = \sum_{j=1}^{m} b_j(Z) \mathrm{d}u_j, \tag{7}$$

with

$$a(Z) = Z^n - \sum_{i=0}^{n-1} \frac{\partial \phi}{\partial y^{(i)}} Z^i, \quad b_j(Z) = \sum_{i=0}^{n-1} \frac{\partial \phi}{\partial u_j^{(i)}} Z^i \tag{8}$$

and $a(Z), b_j(Z) \in \mathcal{K}[Z, s]$ for $j = 1, \ldots, m$.

4. REALIZABILITY CONDITIONS

Realizability conditions in terms of adjoint polynomials can be found in Halás and Kotta (2009) for single-input single-output (SISO) systems and in Halás and Kotta (2011) for multi-input single-output systems:

Theorem 3. Given a nonlinear control system defined by i/o equation (1), or equivalently by (6), let

$$\omega_i := \sum_{j=1}^{m} b^*_{j,i-1} \mathrm{d}u_j - a^*_{i-1} \mathrm{d}y, \quad i = 1, \ldots, n. \tag{9}$$

Then there exists a state space realization of the form (2) if and only if

$$\begin{aligned} \mathrm{span}_{\mathcal{K}}\{&\mathrm{d}y, \mathrm{d}\dot{y} - \omega_n, \mathrm{d}\ddot{y} - \dot{\omega}_n - \omega_{n-1}, \ldots, \\ &\mathrm{d}y^{(n-1)} - \omega_n^{(n-2)} - \omega_{n-1}^{(n-3)} - \ldots - \omega_2, \} \end{aligned} \tag{10}$$

is integrable.

Realizability conditions based on division of non-commutative polynomials are given in Tõnso and Kotta (2009) for SISO systems and in Belikov et al. (2011) for multi-input multi-output systems:

Theorem 4. Given a nonlinear control system defined by i/o equation (1), or equivalently by (6), let

$$\bar{\omega}_l := \left[\bar{a}_l(Z), -\bar{b}_{\cdot,l}(Z) \right] \begin{bmatrix} \mathrm{d}y \\ \mathrm{d}u \end{bmatrix}, \quad l = 1, \ldots, n \tag{11}$$

where $\bar{a}_l(Z)$ and $\bar{b}_{\cdot,l}(Z)$ can be computed recursively from

$$\begin{aligned} \bar{a}_{l-1}(Z) &= Z\,\bar{a}_l(Z) + r_l \\ \bar{b}_{\cdot,l-1}(Z) &= Z\,\bar{b}_{\cdot,l}(Z) + \rho_{\cdot,l}, \end{aligned} \tag{12}$$

with the initial polynomials

$$\bar{a}_0(Z) = a(Z), \quad \bar{b}_{\cdot,0}(Z) = [b_1(Z), \ldots, b_m(Z)].$$

Then there exists a state space realization of the form (2) if and only if

$$\mathrm{span}_{\mathcal{K}}\{\bar{\omega}_l,\, l = 1, \ldots, n\} \tag{13}$$

is integrable.

State coordinates necessary for realization can be found by integrating the basis vectors of the subspaces (10) or (13). Of course, one cannot find the integrable one-forms $\mathrm{d}x_1(t)$, …, $\mathrm{d}x_n(t)$ for an arbitrary i/o equation. No matter which way the the subspace is calculated, either by Theorem 3 or by Theorem 4, its integrability can be checked by the Frobenius theorem.

Theorem 5. Choquet-Bruhat et al. (1989)(Frobenius) Let $\mathcal{V} = \mathrm{span}_{\mathcal{K}}\{\omega_1, \ldots, \omega_r\}$ be a subspace of $\mathcal{E}$. $\mathcal{V}$ is closed iff $\mathrm{d}\omega_k \wedge \omega_1 \wedge \ldots \wedge \omega_r = 0$, for all $k = 1, \ldots, r$.

5. MAIN RESULT

Since Theorems 3 and 4 both present necessary and sufficient realizability conditions, these conditions are obviously equivalent. The goal of this section is to show the precise relationship between the codistribution (10) and (13).

Proposition 6. The basis one-forms (10) in Theorem 3 coincide with the one-forms $\bar{\omega}_l$, $l = 1, \ldots, n$ in Theorem 4.

Proof. First note that basis one-forms of the codistribution (10) can be written in a form

$$\mathrm{d}y^{(i-1)} - \omega_n^{(i-2)} - \ldots - \dot{\omega}_{n-i+3} - \omega_{n-i+2},$$

where $i = 1, \ldots, n$. Definition (9) allows us to rewrite the basis vectors in the form:

$$(Z^{i-1} + Z^{i-2}a^*_{n-1} + \ldots + Za^*_{n-i+2} + a^*_{n-i+1})\mathrm{d}y$$
$$-\sum_{j=1}^{m}(Z^{i-2}b^*_{j,n-1} + \ldots + Zb^*_{j,n-i+2} + b^*_{j,n-i+1})\mathrm{d}u_j \quad (14)$$

Next, observe that the (polynomial) coefficient of $\mathrm{d}y$, denoted by $\tilde{a}_{n-i+1}(Z)$, is similar to the polynomial

$$a(Z) = Z^n + Z^{n-1}a^*_{n-1} + \ldots + Za^*_1 + a^*_0.$$

We only need to multiply $\tilde{a}_{n-i+1}(Z)$ from left by Z^{n-i+1} and add a missing part $z^{n-i}a^*_{n-i} + \ldots + Za^*_1 + a^*_0$. The latter means

$$Z^{n-i+1}\tilde{a}_{n-i+1}(Z) + Z^{n-i}a^*_{n-i} + \ldots + Za^*_1 + a^*_0 = a(Z).$$

For the sake of simplicity we can replace in previous equality the index i by $l = n - i + 1$, $l = 1, \ldots n$, keeping in mind that it just means reversing the order of the coefficients:

$$Z^l\tilde{a}_l(Z) + Z^{l-i}a^*_{l-1} + \ldots + Za^*_1 + a^*_0 = a(Z).$$

Thus, we have represented a coefficient $\tilde{a}_l(Z)$ of $\mathrm{d}y$ as a left quotient of $a(Z)$ and Z^l, $l = 1, \ldots, n$. Note that such quotients can be computed using the recursive formula

$$Z\tilde{a}_l(Z) + a^*_{l-1} = \tilde{a}_{l-1}(Z).$$

Since the quotient of two polynomials is unique, the polynomial $\tilde{a}_l(Z)$ has to be equal to $\bar{a}_l(Z)$ in (12), while $r_l = a^*_{l-1}$. Analogously, it is possible to prove that the coefficients of $\mathrm{d}u_j$ for $j = 1, \ldots, m$ in (14) equal to $\bar{b}_{\cdot,l}(Z)$ in (12). Thus, we have shown that the basis one-forms (10) and (11) coincide, except that the order is reversed. ■

Remark about the discrete-time case

Realizability conditions for discrete-time systems are largely analogous to their continuous-time counterparts. The analogue of Theorem 3 for discrete-time SISO systems can be found in Halás and Kotta (2010); the only difference is that in basis (10) instead of the derivative operator there is the forward shift operator δ. The forward shift operator is defined by shifting the arguments of the function according to the rules $\delta y(t) = y(t+1)$, $\delta u_j(t) = u_j(t+1)$, $j = 1, \ldots, m$. The inverse operator of δ is denoted by δ^{-1} and called backward-shift operator. We should also keep in mind that in discrete-time case multiplication of polynomials and adjoint polynomials is defined by different commutation rules, $Z \cdot a = \delta(a) \cdot Z$ and $Z^* \cdot a = \delta^{-1}(a) \cdot Z^*$, respectively. Due to the latter commutation rule, for an adjoint polynomial $p^*(Z^*) = p^*_n Z^{*n} + \ldots + p^*_1 Z^* + p^*_0$, the equalities $p^*_i = \delta^{-i}(p_i)$, $i = 0, \ldots, n$ hold. Therefore, for the discrete-time systems, the equality (14) takes the form

$$(Z^{i-1} + \delta^{i-n-1}(a_{n-1})Z^{i-2} + \ldots + \delta^{i-n-1}(a_{n-i+1}))\mathrm{d}y$$
$$-\sum_{j=1}^{m}(\delta^{i-n-1}(b_{j,n-1})Z^{i-2} + \ldots + \delta^{i-n-1}(b^*_{j,n-i+1}))\mathrm{d}u_j \quad (15)$$

for $i = 1, \ldots, n$. The coefficients of $\mathrm{d}y$ and $\mathrm{d}u_j$, $j = 1, \ldots, m$ can be computed by polynomial division as in Theorem 4; however, the shorter way is to use the cut-and-shift operator defined by

$$\delta_c^{-1}(p(Z)) = \delta^{-1}(p(Z) - p_0), \quad (16)$$

see also Kotta and Tõnso (2008). In terms of the cut-and-shift operator the equality (15) may expressed as

$$\delta_c^{-l}a(Z)\mathrm{d}y - \sum_{j=1}^{m}\delta_c^{-l}(b_{j,l})\mathrm{d}u_j = \delta_c^{-l}[a(Z), -b(Z)]\begin{bmatrix}\mathrm{d}y\\ \mathrm{d}u\end{bmatrix} \quad (17)$$

for $l = 1, \ldots, n$. The latter formula agrees with the result in Kotta and Tõnso (2008).

6. EXAMPLES

Example 1. Consider the control system $\ddot{y} = u + y\dot{u}^2$. Let us compute the differentials of state coordinates necessary for realization by two different methods, by Theorem 3 and Theorem 4. As a common step of both methods, one has to find the polynomial representation of the system, i.e polynomials $a(Z)$ and $b_1(Z) = b(Z)$:

$$\begin{aligned} a(Z) &= Z^2 - \dot{u}^2 \\ b(Z) &= 2y\dot{u}Z + 1. \end{aligned}$$

Following Theorem 3, one has to compute adjoint polynomials $a^*(Z^*)$ and $b^*(Z^*)$. By (5),

$$\begin{aligned} a^*(Z^*) &= Z^{*2} + \dot{u}^2 \\ b^*(Z^*) &= 2y\dot{u}Z^* + (1 - 2\dot{y}\dot{u} - 2y\ddot{u}). \end{aligned}$$

After computing ω_1 and ω_2 defined by (9)

$$\begin{aligned} \omega_1 &= (1 - 2\dot{y}\dot{u} - 2y\ddot{u})\mathrm{d}u - \dot{u}^2\mathrm{d}y \\ \omega_2 &= 2y\dot{u}\mathrm{d}u \end{aligned}$$

it is easy to write down the basis one-forms (14)

$$\mathrm{span}_{\mathcal{K}}\{\mathrm{d}y,\ \mathrm{d}\dot{y} - \omega_2\} = \mathrm{span}_{\mathcal{K}}\{\mathrm{d}y,\ \mathrm{d}\dot{y} - 2y\dot{u}\,\mathrm{d}u\}. \quad (18)$$

According to Frobenius condition the latter subspace is not integrable and thus the system does not admit the classic state space representation.

Alternatively, one may follow Theorem 4 and find $\bar{a}_l(Z)$, $\bar{b}_l(Z)$, $l = 1, \ldots, n$ by dividing $a(Z)$ and $b(Z)$ from the left by the polynomial Z repeatedly:

$$\begin{aligned} &\bar{a}_0(Z) = Z^2 - \dot{u}^2 && \bar{b}_0(Z) = 2y\dot{u}Z + 1 \\ &\bar{a}_1(Z) = Z && \bar{b}_1(Z) = 2y\dot{u} \\ &\bar{a}_2(Z) = 1 && \bar{b}_2(Z) = 0 \end{aligned}$$

By (11),

$$\begin{aligned} &\bar{\omega}_1 = Z\mathrm{d}y - 2y\dot{u}\mathrm{d}u = \mathrm{d}\dot{y} - 2y\dot{u}\mathrm{d}u \\ &\bar{\omega}_2 = \mathrm{d}y, \end{aligned}$$

which coincide with the basis one-forms of (18).

Example 2. Consider the control system

$$y^{(3)} = u_1\dot{y} + y\dot{u}_1 + \dot{u}_2^2 + u_2\ddot{u}_2 \quad (19)$$

that can be described as in (8) as follows:

$$\begin{aligned} a(Z) &= Z^3 - u_1 Z \\ b_1(Z) &= Z + \dot{y} \\ b_2(Z) &= u_2 Z^2 + 2\dot{u}_2 Z + \ddot{u}_2. \end{aligned} \quad (20)$$

Obeying Theorem 3, the adjoint polynomials $a^*(Z^*)$, $b_1^*(Z^*)$ and $b_2^*(Z^*)$ are as follows:

$$\begin{aligned} a^*(Z^*) &= Z^{*3} - u_1 Z^* + \dot{u}_1 \\ b_1^*(Z^*) &= Z^* + \dot{y} \\ b_2^*(Z^*) &= u_2 Z^{*2}. \end{aligned}$$

Computation of ω_i's, defined by (9), yields

$$\begin{aligned} \omega_1 &= \dot{y}\mathrm{d}u_1 - \dot{u}_1\mathrm{d}y \\ \omega_2 &= \mathrm{d}u_1 + u_1\mathrm{d}y \\ \omega_3 &= u_2\mathrm{d}u_2, \end{aligned} \tag{21}$$

which allow to find the basis one-forms defined by (10):

$$\mathrm{span}_{\mathcal{K}}\{\mathrm{d}y,\ \mathrm{d}\dot{y}-\omega_3, \mathrm{d}\ddot{y}-\dot{\omega}_3-\omega_2\} = \mathrm{span}_{\mathcal{K}}\{\mathrm{d}y,\ \mathrm{d}\dot{y}-u_2\mathrm{d}u_2, \mathrm{d}\ddot{y}-\dot{u}_2\mathrm{d}u_2-u_2\mathrm{d}\dot{u}_2-\mathrm{d}u_1-u_1\mathrm{d}y\}. \tag{22}$$

Alternative solution starts by computing quotients by (12)

$$\begin{aligned} &\bar{a}_0(Z) = Z^3 - u_1 Z & &\bar{b}_{\cdot,0} = [\,Z+\dot{y},\ u_2Z^2+2\dot{u}_2Z+\ddot{u}\,] \\ &\bar{a}_1(Z) = Z^2 - u_1 & &\bar{b}_{\cdot,1} = [\,1,\ u_2Z+\dot{u}_2\,] \\ &\bar{a}_2(Z) = Z & &\bar{b}_{\cdot,2} = [\,0,\ u_2\,] \\ &\bar{a}_3(Z) = 1 & &\bar{b}_{\cdot,3} = [\,0,\,0\,], \end{aligned} \tag{23}$$

that allow to find the basis one-forms $\bar{\omega}_1$, $\bar{\omega}_2$ and $\bar{\omega}_3$ according to (11):

$$\begin{aligned} \bar{\omega}_1 &= (Z^2-u_1)\mathrm{d}y - \mathrm{d}u_1 - (u_2Z-\dot{u}_2)\mathrm{d}u_2 \\ &= \mathrm{d}\ddot{y} - u_1\mathrm{d}y - \mathrm{d}u_1 - u_2\mathrm{d}\dot{u}_2 - \dot{u}_2\mathrm{d}u_2 \\ \bar{\omega}_2 &= Z\mathrm{d}y - 0\cdot\mathrm{d}u_1 - u_2\mathrm{d}u_2 \\ &= \mathrm{d}\dot{y} - u_2\mathrm{d}u_2 \\ \bar{\omega}_3 &= \mathrm{d}y. \end{aligned}$$

As expected, the above one-forms coincide with (22). According to Frobenius condition, the subspace $\mathrm{span}_{\mathcal{K}}\{\bar{\omega}_1, \bar{\omega}_2, \bar{\omega}_3\}$ is integrable and the differentials of the state coordinates

$$\begin{aligned} \mathrm{d}x_1 &= \bar{\omega}_3 = \mathrm{d}y \\ \mathrm{d}x_2 &= \bar{\omega}_2 = \mathrm{d}(\dot{y} - \frac{1}{2}u_2^2) \\ \mathrm{d}x_3 &= \bar{\omega}_1 + u_1\bar{\omega}_3 = \mathrm{d}(\ddot{y} - u_1 - u_2\dot{u}_2) \end{aligned}$$

yield the classical state equations

$$\begin{aligned} \dot{x}_1 &= \frac{1}{2}u_2^2 + x_2 \\ \dot{x}_2 &= u_1 + x_3 \\ \dot{x}_3 &= u_1(\frac{1}{2}u_2^2 + x_2). \end{aligned}$$

7. MATHEMATICA IMPLEMENTATION

We have implemented both realization methods in computer algebra system Mathematica. On that purpose we use the functions from the package NLControl, allowing to solve various modelling, analysis and synthesis problems for nonlinear control systems, see Tõnso et al. (2009); Tõnso (2010). NLControl package also includes the basic tools for polynomials from Ore rings. If compared with the Maple OreTools package, the polynomial functions in NLControl have one essential benefit. Namely, NLControl allows take into account that the derivative (or shift) operator is defined by the control systems equations, see Halás et al. (2009). The OreTools package lacks such possibility, therefore it may sometimes provide a wrong result when applied to nonlinear control problems.

The given Mathematica code below has two advantages worth to mention. First, it is constructed in a way it can handle both continuous- and discrete-time systems; in case of discrete-time systems one has just to replace the word `TimeDerivative` by the word `Shift`. Second, though the scope of this paper is limited to single-output systems, the above code allows to find state coordinates for multi-output systems, too. Therefore an additional pair of curly braces may appear around Mathematica expressions below. Consider the system (19). After loading the package by the command

```
<<NLControl`Master`
```

let us create the object `IO[]`, representing the i/o equations for this system.

```
eqs = {y'''[t]->u1[t]y'[t]+u1'[t]+u2'[t]^2+
u2[t]u2''[t]};
Ut = {u1[t],u2[t]};
Yt = {y[t]};
ioeq = IO[eqs,Ut,Yt,t,TimeDerivative]
```

First, let us compute the state coordinates by Theorem 3. For that we need to construct the Ore ring and adjoint Ore ring associated with the system `ioeq`.

```
R = DefineOreRing[Z, ioeq];
adR = DefineAdjointOreRing[Z, ioeq];
```

The function `FromIOToOreP` finds the polynomials $a(Z)$ and $b_j(Z)$, $j = 1, \ldots, m$ for the system `ioeq`. Note that the polynomials $b_j(Z)$ are chosen with the opposite sign to (7) and (20).

```
AB = MapThread[Join, FromIOToOreP[ioeq]]
```

```
{{OreP[1,0,-u1[t],0],
  OreP[-1,-y'[t]],
  OreP[-u2[t],-2 u2'[t],-u2''[t]]}}
```

The function `OreP[`p_n`, ...,`p_1`, `p_0`]` represents the Ore polynomial in the form (4). The function `Adjoint` allows to compute adjoints of the polynomials being elements of the matrix `AB`.

```
adAB = Map[Adjoint[#, adR]&, AB, 2]
```

```
{{OreP[1,0,-u1[t],u1'[t]], OreP[-1,-y'[t]],
  OreP[-u2[t],0,0]}}
```

By the next code row the coefficients of the polynomials are represented as the individual 0-degree polynomials. The constant terms (the last argument of `OreP[]`) are removed, since they correspond to ω_1, which is not necessary in further computations.

```
adAB = Map[ If[ Head[#]===OreP,
Drop[ List @@ OreP /@ #, -1], ]&, adAB, 2]
```

```
{{{1, 0, OreP[-u1[t]]}, {OreP[-y[t]]},
  {OreP[-u2[t]], 0}}}
```

To obtain the the polynomials, respective to the one-forms ω_i, defined by (9), we need to equalize the length of the rows by adding the missing zeros to the beginning of each row and the transpose the obtained matrix.

```
omega = Transpose/@ PadLeft/@ adAB
```

```
{{{1,0,0}, {0,0,OreP[-u2[t]]},
  {OreP[-u1[t]],-1,0}}}
```

In above output the vector `{1,0,0}` corresponds to dy, which can be considered as $\omega_{n+1} = \omega_4$, the vector `{0,0,OreP[-u2[t]]}` corresponds to $-\omega_3$ and `{OreP[-u1[t]], OreP[-y[t]],0}` to $-\omega_2$, given by (21). Our next task is to compute the basis one-forms (10). Denoting the basis elements by $\theta_1, \ldots, \theta_n$ (and recalling that our ω_i, $i = 2, \ldots, n$ are of opposite signs, to (9)), allows to compute the basis by the time-saving recursive formula $\theta_1 = \mathrm{d}y$, $\theta_{i+1} = \dot{\theta}_i + \omega_{n-i+1}$ for $i = 1, \ldots, n-1$. Instead of computing the derivatives of θ_i, $i = 1, \ldots, n-1$ we have chosen to multiply the respective polynomials by s from right. The reason for such choice is that the routines available in NLControl are more suitable for this method and it allows to avoid several clumsy data transformation.

```
statedifpoly = Rest @ FoldList[
OreMultiply[ OreP[1,0],#1,K]+#2&, 0, #]&
/@ omega
```

```
{{{1,0,0}, {OreP[1,0],0,OreP[-u2[t]]},
  {OreP[1,0,-u1[t]], -1,
  OreP[-u2[t],-u2′[t]]}}}
```

The function `FromOrePToSpanK` converts the list of Ore polynomials into the set of one-forms.

```
statedif = FromOrePToSpanK[
Join @@ statedifpoly, ioeq]
```

```
SpanK[{{1,0,0,0,0,0}, {0,1,0,0,-u2[t],0},
  {-u1[t],0,1,-1,-u2′[t],-u2[t]}},
  {y[t],y′[t],y′′[t],u1[t],u2[t],u2′[t]},
  -1, t]
```

We have computed the basis of the subspace (10). The function `BookForm` allows to print the result in a user-friendly form:

```
BookForm[statedif]
```

```
SpanK{dy, dy′ - u2 du2,
 -u1 dy + dy′′- du1 - u2′du2 - u2 du2′}
```

Integrating the one-forms

```
states = IntegrateOneForms[statedif]
```

```
{y[t], 1/2 (u2[t]^2 - 2y′[t]),
  u1[t] + u2[t]u2′[t] - y′′[t]}
```

yields the state coordinates, which allow to find the state equations:

```
BookForm[Realization[ioeq,
{x1[t],x2[t],x3[t]}, states]]
```

```
x1′ = (u2^2 - 2*x2)/2
x2′ = -u1 + x3
x3′ = -((u2^2 - 2*x2)*u1)/2
  y = x1

 x1 = y
 x2 = (u2^2 - 2*y′)/2
 x3 = u1 + u2 u2′ - y′′)
```

The basis (10) can be also found by the single function `StateDifferentialsAdjoint[ioeq]` (the name of the function may be confusing, because in fact, it gives the *linear combination* of the differentials of the state coordinates.) The realization procedure can be performed by the function `Realization[ioeq, {x1[t],x2[t],...}, Method->Adjoint]`.

Second, let us find the state coordinates by Theorem 4. We can employ the Ore ring `R` and the matrix `AB` already computed above. The order of the system can be determined by the function `MaxPLMOrder`, which allows to find the maximal order of time-derivative, shift or any other pseudo-linear map.

```
ni = MaxPLMOrder[eqs, #, TimeDerivative
]& /@ Yt
```

```
{3}
```

According to (12) we have to divide the elements of the matrix AB repeatedly by the polynomial s from left. By abuse of function name, we have defined the function `CutAndShift[`$p(Z)$`,R]` as a left quotient of $p(Z)$ and s for the Ore rings, associated with the continuous-time systems. The reason is that in case of discrete-time systems the quotients can be found by applying cut-and-shift operator as in (17). This extension allows to compute the polynomials $\bar{a}_l$ and $\bar{b}_{\cdot,l}$, $l = 1, \ldots, n$ in (12) for both, continuous- and discrete time systems, by the following compact row:

```
ABquot = MapThread[ NestList[
CutAndShift[#, K]&, #1, #2]&, AB, ni]
```

```
{{{OreP[1,0,-u1[t],0],
   OreP[-1,-y′[t]],
   OreP[-u2[t],-2 u2′[t],-u2′′[t]]},
 {OreP[1,0,-u1[t]], -1,
   OreP[-u2[t],-u2′[t]]},
 {OreP[1,0],0,OreP[-u2[t]]},
 {1,0,0}}}
```

The obtained list corresponds to $\{\{\{\bar{a}_0(Z), \bar{b}_{\cdot,0}(Z)\}, \ldots, \{\bar{a}_3(Z), \bar{b}_{\cdot,3}(Z)\}\}\}$, where the polynomials are given by (23). Removing the first row $\{\bar{a}_0(Z), \bar{b}_{\cdot,0}(Z)\}$ and reversing the order of the remaining rows yields

```
ABquot = Reverse /@ Rest /@ ABquot
```

```
{{{1, 0, 0}, {OreP[1, 0], 0, OreP[-u2[t]]},
 {OreP[1, 0, -u1[t]], -1,
    OreP[-u2[t], -u2′[t]]}}}
```

The funtion `FromOrePToSpanK` converts Ore polynomials into the set of one-forms.

```
FromOrePToSpanK[ Join @@ ABquot, ioeq]
```

```
SpanK[{{1,0,0,0,0,0}, {0,1,0,0,-u2[t],0},
  {-u1[t],0,1,-1,-u2′[t],-u2[t]}},
  {y[t],y′[t],y′′[t],u1[t],u2[t],u2′[t]},
   -1, t]
```

By that the basis of the subspace (13) has been computed. The one-forms (13) can be also found by the function `StateDifferentialsLeftQuoteint[ioeq]` and the realization can be performed by `Realization[ioeq, {x1[t],x2[t]...}, Method->LeftQuotient]`.

Preliminary comparison of two realization methods, based on relatively small examples, suggests that the program employing adjoint polynomials works faster than the one based on the polynomial quotients. The reason is obvious

– adjoint polynomial method utilizes only polynomial multiplication, which is a primary function (it is immaterial whether we work with polynomials or with adjoint polynomials) and is therefore performed faster than polynomial division.

However, in discrete-time case the situation is different. Polynomial quotient operator can be replaced by cut-and-shift operator, which means that the program can compute the basis one-forms without calling neither multiplication nor division functions and therefore, it can produce the results faster than the program involving adjoint polynomials.

8. CONCLUSION

In this paper the realizability problem of nonlinear control system has been addressed. The paper focuses on establishing the explicit relationship between the two necessary and sufficient realizability conditions. Both conditions are formulated in terms of integrability of a certain subspace. While in the first condition the basis one-forms of the subspace are achieved from the adjoints of the polynomials describing the system; in the second condition the basis one-forms are found using polynomial quotients. As both conditions are necessary and sufficient it is obvious that they are equivalent; however, in this paper it is proved that the basis one-forms used in both conditions are equal.

ACKNOWLEDGMENTS

This work was partially supported by the ESF Grant no 8787 and target funding project of Estonian Ministry of Education and Research SF0140018s08.

REFERENCES

S.A. Abramov, H.Q. Le, and Z. Li. Univariate ore polynomial rings in computer algebra. *Journal of Mathematical Sciences*, 131(5):5885–5903, 2005.

J. Belikov, M. Tõnso, and Ü. Kotta. Realization of MIMO nonlinear equations: polynomial approach. 2011. Submitted for publication.

Y. Choquet-Bruhat, C. DeWitt-Morette, and M. Dillard-Bleick. *Analysis, Manifolds and Physics, Part I: Basics.* North Holland, Amsterdam, 1989.

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

E. Delaleau and W. Respondek. Lowering the orders of derivatives of controls in generalised state space systems. *Journal of Mathematical Systems, Estimation and Control*, 5:1–27, 1995.

M. Halás and Ü. Kotta. Realization problem of SISO nonlinear systems: a transfer function approach. In *IEEE International Conference on Control and Automation*, pages 546–551. Christchurch, New Zealand, 2009.

M. Halás and Ü. Kotta. Extension of the transfer function approach to the realization problem of nonlinear systems to discrete-time case. In *8th IFAC Symposium on Nonlinear Control Systems, University of Bologna*, pages 179–184, Bologna, Italy, 2010.

M. Halás and Ü. Kotta. A polynomial approach to the realization problem of nonlinear systems. 2011. Submitted for publication.

M. Halás, Ü. Kotta, Z. Li, H. Wang, and C. Yuan. Submersive rational difference systems and their accessibility. In *Proceedings of the 2009 international symposium on Symbolic and algebraic computation*, ISSAC '09, pages 175–182, 2009.

E.R. Kolchin. *Differential algebra and algebraic groups.* Academic Press, 1973.

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

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

M. Tõnso. *Computer Algebra Tools for Modelling, Analysis and Synthesis for Nonlinear Control Systems.* PhD thesis, Tallinn University of Technology, Institute Of Cybernetics, 2010.

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

M. Tõnso, H. Rennik, and Ü. Kotta. Webmathematica based tools for discrete-time nonlinear control systems. *Proceedings of the Estonian Academy of Sciences*, 58(4): 224–240, 2009.