Записки научных
семинаров ПОМИ
Том 258, 1999 г.

C. Hespel, G. Jacob

ON THE ALGEBRAIC IDENTIFICATION PROBLEM: RESULTS AND PERSPECTIVES

ABSTRACT. As shown by M. Fliess, the input/output behaviour of the nonlinear analytical dynamical systems can be symbolically encoded in a non commutative formal power series. We get so, for the non linear dynamical systems, an equivalent of the Laplace transform. Here we are concerned by the converse process: starting from some input/output behaviour, is it possible to effectively compute its symbolic description, starting from some finite panels of input/output data?

More specifically, we present here several ways for effectively computing the terms of the generating series, taking as data the jets of some input at time 0 and the jets at time 0 of the corresponding output. And we restrict ourselves to the theoretic case of numerically exact data.

First we give a commutative approach, based on the representation of the output as an analytical expansion on a basis of the shuffle algebra.

Then we present a noncommutative combinatorial approach, based of the analysis of recurrence relations via Chen series and its successive derivatives. By this way, we have obtained a very compact and efficient identification software.

These results should have mainly two types of applications (not yet developed). We can use them as an help for real identification softwares. It would also allow to test real process, in order to decide if they reflect the behaviour of some unknown causal analytic functional.

1. INTRODUCTION

The "causal analytical functionals" were defined by M. Fliess [2] as the description framework of input/output behaviour of analytical systems, affine in the inputs.

1.1. Analytical systems

Let us consider an evolution equation system (Σ) of the form:

$$\begin{aligned} \dot{q} &= f_0(q) + f_1(q)\, a_1(t) + \cdots + f_m(q)\, a_m(t) \\ y(t) &= h(q), \end{aligned}$$

where

- the q belongs to the state space $\mathcal{V}$,

- the f_i are analytical vector fields on $\mathcal{V}$
- h is an analytic function from $\mathcal{V}$ in $\mathbb{R}$.

The function is $\mathbf{a(t)} = (\mathbf{a_i(t)})_{\mathbf{j=1\cdots m}}$ is *the input function*, and it determines the solution $y(t)$ of the differential system, which is called the *output function.*

The system is called *"homogenous on the input"* if $f_0(q) = 0$. If it is not the case, we can give all the same an homogeneous treatment by setting a fictious input component $a_0(t) \equiv 0$.

1.1.1. The symbolic encoding

The computation of the "output" y (see for instance [6]) can be obtained by the mean of two families of coefficients, indexed by an encoding alphabet $Z = \{z_0, z_1 \cdots z_m\}$, and summarizing respectively the system, and the input contribution.

1. *Firstly,* we compute the *"system contributions"*, that summarized in a noncommutative formal power series, called the Fliess series of the system:
$$G_\Sigma = \sum_{w \in Z^\star} \langle G_\Sigma \mid w > w,$$
where for each word $w = z_{i_1} \circ z_{i_2} \circ \cdots \circ z_{i_m}$ we have set:
$$\langle G_\Sigma \mid w \rangle = f_{i_1} \circ f_{i_2} \circ \cdots \circ f_{i_m} \circ h_{|_{q_0}}$$
is the iterated Lie derivative of the observation function.
2. *Secondly,* regarding the input of the system as being $[\mathbf{a(t)} = = (\mathbf{a_j(t)})_{\mathbf{j=0\cdots m}}]$, we "evaluate" G by replacing each word w by its *"input contribution,"* namely the iterated integral $\int_0^t \delta_{\mathbf{a}} w$, the recursive definition of which is given as follows:
$$\int_0^t \delta_{\mathbf{a}}(\varepsilon) \quad =1$$
$$\int_0^t \delta_{\mathbf{a}}(wz_i) = \int_0^t \left[\int_0^\tau \delta_{\mathbf{a}}(w) \right] a_i(\tau) d\tau.$$
These iterated integrals depend only on the input $\mathbf{a}$. They define also a noncommutative power series which is call the Chen series

$\mathcal{C}_{\mathbf{a}}(t)$ of the input $\mathbf{a}$:

$$\forall w \in Z^\star \qquad \langle C_{\mathbf{a}}(t) \mid w\rangle = \int_0^t \delta_{\mathbf{a}}(w).$$

And the output corresponding to the input $\mathbf{a}$ is given by the infinite sum:

$$y(t) = \sum_{w \in Z^\star} \langle G_\Sigma \mid w\rangle \, \langle \mathcal{C}_{\mathbf{a}}(t) \mid w\rangle = \langle G_\Sigma \mid\mid \mathcal{C}_{\mathbf{a}}(t)\rangle.$$

1.1.2. Causal functionals

In the same way, for *any noncommutative power series* G on the encoding alphabet $Z = \{z_0, z_1, \cdots, z_m\}$ (up to some convergence condition on its coefficients), we can define a *causal functional* as follows: its value on the input function $\mathbf{a(t)} = (\mathbf{a_j(t)})_{\mathbf{j}=\cdots\mathbf{m}}$ is given by the same summation formula:

$$y(t) = \langle G \mid\mid \mathcal{C}_{\mathbf{a}}(t)\rangle.$$

1.2. Identification problem

Conversely, any unknown dynamical system can be considered as a black-box, defining a functional $\mathbf{a(t)} \mapsto \mathbf{y}(t)$. Then, in so far as we are able to measure or compute as many as required informations on this "input/output behaviour," two questions have to be set:

- *are the symbolic coefficients* $\langle G_\Sigma \mid w\rangle$ *univcally defined?*
- *is there a finite algorithm to compute (exactly) any finite set of them?*

The answer to *the first question* is: yes. Three different proves was given of this fact (that we have summarized in [4]). The first (Fliess [3]) uses generalized inputs, that are similar in the use as Dirac function. The second (Reutenauer [4]) requires the use of the main properties of analytical functions. The third (Sonntag and Wang [9]) is based on the definition of some noncommutative derivations. In brief, these three proves required computations of limits, and continous topological properties. They probably could allow a *numerical approximated computation* of the coefficients of the generating series. But no algorithm is actually developed on these tools.

The *second question* is the **Algebraic Identification Problem** ([4, 5]). It can be precised as follows:

> *For each word* w*, can we determine a finite panel set of correspondences between selected inputs and corresponding outputs, that will be sufficient for computing the coefficient* $\langle G \mid w\rangle$ *of the generating series?*

A positive answer to this second question would immediately produce a new positive constructive answer to the first question.

1.3. Content of the paper

After a short presentation of the formulas allowing to symbolically compute the iterated derivative of the Chen series (Sec. 2) (and then of the output $y(t)$), we present two computation methods. The first (Sec. 3) is based on a commutative analytical description of the output. It requires a presentation of the shuffle algebra. The second approach (Sec. 4) is based on a joint use of the combinatorial description of the differential monomials on the input, and the description of noncommutative recursive equations.

2. Output and input iterated derivatives

In this section we give the ground of the symbolic computation of the iterated time derivatives of the Chen series, with an example of computation. It allows to anounce the two ways proposed here for solving the Algebraic Identification Problem.

2.1. Time derivatives of the Chen series

For any input function $\mathbf{a}$, the Chen series $\mathcal{C}_{\mathbf{a}}$ ([8]) satisfies the differential equation:

$$\frac{d}{dt}\mathcal{C}_{\mathbf{a}}(t) = \mathcal{C}_{\mathbf{a}}(t)\ \mathcal{L}_{\mathbf{a}}(t)$$

with the initial condition:

$$C_{\mathbf{a}}(0) = 1,$$

where $\mathcal{L}_{\mathbf{a}}(t) = \sum_{j\in Z} a_j(t)\ z_j$ is a *time dependent noncommutative polynomial* on the encoding alphabet Z. Then the iterated time derivatives of $\mathcal{C}_{\mathbf{a}}(t)$ can be recursively computed in the form:

$$\frac{d^n}{dt^n}\mathcal{C}_{\mathbf{a}}(t) = \mathcal{C}_{\mathbf{a}}(t)\ \mathbf{A}_n(t),$$

where the noncommutative polynomials $\mathbf{A}_n$ are defined as:

$$\text{the recursive equations}\quad \begin{cases} \mathbf{A}_1(t) = \mathcal{L}_{\mathbf{a}}(t) = \sum_{j\in Z} a_j(t)\ z_j \\ \qquad \dots \\ \mathbf{A}_{n+1}(t) = \mathcal{L}_{\mathbf{a}}(t)\mathbf{A}_n(t) + \dfrac{d}{dt}\mathbf{A}_n(t). \end{cases}$$

The algebra of noncommutative polynomials (resp. noncommutative series) on Z will be noted $k\langle Z\rangle$ (resp. $k\langle\langle Z\rangle\rangle$). They are non commutative associative algebras for the "Cauchy product" (induced by the concatenation of words). These two vector spaces can also be considered as commutative algebras for the shuffle product (see (3.1)), and will be then rather noted $Sh\langle Z\rangle$ (resp. $Sh\langle\langle Z\rangle\rangle$).

2.2. Example. In order to underline the combinatorial concepts used fully in the paper, we specify the cases $n = 2$ and $n = 3$:

$$\begin{aligned}
\mathbf{A_2} &= \mathcal{L}_{\mathbf{a}}\mathcal{L}_{\mathbf{a}} + \frac{f}{dt}\mathcal{L}_{\mathbf{a}} \\
&= \sum_{i<j} a_i a_j \,.\, z_i z_j + \sum_j \dot{a}_j z_j \\
&= \sum_{i<j} a_i a_j \,.\, (z_i z_j + z_j z_i) + \sum_j a_j^2 \,.\, z_j^2 + \sum_j \dot{a}_j z_j\,,
\end{aligned}$$

$$\begin{aligned}
\mathbf{A_3} &= \mathcal{L}_{\mathbf{a}}\mathcal{L}_{\mathbf{a}}\mathcal{L}_{\mathbf{a}} + 2\mathcal{L}_{\mathbf{a}}\left(\frac{d}{dt}\mathcal{L}_{\mathbf{a}}\right) + \left(\frac{d}{dt}\mathcal{L}_{\mathbf{a}}\right)\mathcal{L}_{\mathbf{a}} + \frac{d}{dt}2t^2\mathcal{L}_{\mathbf{a}} \\
&= \sum_{i,j,k} a_i a_j a_k \,.\, z_i z_j z_k + 2\sum_{i,j} a_i \dot{a}_j \,.\, z_i z_j + \sum_{i,j} \dot{a}_i a_j \,.\, z_i z_j + \sum_i \ddot{a}_i \,.\, z_i \\
&= \sum_{i,j,k} a_i a_j a_k \,.\, z_i z_j z_k + \sum_{i,j} a_i \dot{a}_j \,.\, (2 z_i z_j + z_j z_i) + \sum_i \ddot{a}_i \,.\, z_i\,.
\end{aligned}$$

Then clearly appears the fact that each $\mathbf{A_n}$ is a polynomial term on the *commutative algebra* generated by the input components a_i and their derivatives, as well as on the *noncommutative words* $w \in Z^\star$.

We recover the third derivative $\dddot{y}(0)$ by applying the G-contribution on $\mathbf{A_3}$ as follows:

$$\begin{aligned}
\dddot{y}(0) &= \sum_{i,j,k} a_i a_j a_k \,.\, \langle G \mid z_i z_j z_k\rangle \\
&\quad + \sum_{i,j} a_i \dot{a}_j \,.\, \langle G \mid (2 z_i z_j + z_j z_i)\rangle + \sum_i \ddot{a}_i \langle G \mid z_i\rangle.
\end{aligned}$$

And we infer a way of computing the generating series coefficients from some panel tests on the iterated derivatives at time $t = 0$ of the input, and corresponding iterated derivatives at time $t = 0$ of the output. Indeed, we have to achieve two computation steps:

- The first consists in computing such polynomials contributions as $\langle G \mid (2z_iz_j + z_jz_i)\rangle$ by a good choice of the input panel set.
- The second consists in recovering each word coefficient $\langle G \mid w\rangle$ of the series G. This can be done by showing that the set of polynomials occuring (as $z_iz_jz_k, (2z_iz_j+z_jz_i)$, etc.) is a basis in the vector space $k\langle Z\rangle$.

We present in the sequel two ways for carrying out this computation. The first (Sec. 3) one is by by interpreting the noncommutative polynomials into commutative ones, thanks to the the shuffle algebra. The second way (Sec. 4) is purely noncommutative, and uses specific tools in combinatorial algebra.

3. First approach via commutative analysis

For sake of simplicity, from now and in all the paper, we specialize, all the following developments on the system:

$$\begin{aligned} \dot{q} &= f_0(q) + f_1(q)\ a_1(t) \\ y(t) &= h(q). \end{aligned}$$

This system is nonhomogenous with a single input $a_1(t)$, and we shall specify a fictious input $a_0(t) \equiv 0$. Then the encoded alphabet is $Z = \{z_0, z_1\}$.

The conversion of the problem into commutative computation is obtained by rewriting the output function y in the *commutative shuffle algebra* $Sh\langle\langle Z\rangle\rangle$.

3.1. The shuffle algebra

The shuffle products becomes from the law of integration by part. The shuffle product of two words u and v is the non commutative polynomial $u \sqcup\!\sqcup\, v$ that satisfies:

$$\int_0^t \delta_{\mathbf{a}}(u \sqcup\!\sqcup\, v) = \Big(\int_0^t \delta_{\mathbf{a}}(u)\Big)\Big(\int_0^t \delta_{\mathbf{a}}(v)\Big)$$

Definition 3.1. *The formal recursive definition of the shuffle product is as follows:*

$$\begin{aligned} &\forall u \in Z^\star, && u \sqcup\!\sqcup\, \varepsilon = \varepsilon \sqcup\!\sqcup\, u = u \\ &\forall x, y \in Z,\quad u, v \in Z^\star, && ux \sqcup\!\sqcup\, vy = (ux \sqcup\!\sqcup\, v)y + (u \sqcup\!\sqcup\, vy)x. \end{aligned}$$

3.2. Generating sets of the commutative shuffle algebra

We use for example the Shirshov basis (But Lyndon basis, or Hall bases as well would be suitable): the first elements of this basis are given here with their interpretation (via Evaluation tranform) in terms of iterated integrals:

$$\begin{array}{lll}
z_0 & \xi_0(t) & = \int_0^t a_0(\tau)d\tau = t \\
z_1 & \xi_1(t) & = \int_0^t a_1(\tau)d\tau \\
z_1 z_0 & \xi_{10}(t) & = \int_0^t \xi_1(\tau)a_0(\tau)d\tau \\
z_1 z_0 z_0 & \xi_{100}(t) & = \int_0^t \xi_{10}(\tau)a_0(\tau)d\tau \\
z_1 z_1 z_0 & \xi_{110}(t) & = \frac{1}{2}\int_0^t \xi_1^2(\tau)a_0(\tau)d\tau.
\end{array}$$

By a theorem of Radford (see [7]), the set of Lyndon words is a transcendence basis of $k\langle Z\rangle$. As a corollary, the output $y(t)$ of a dynamical system can be expressed as an analytical expansion (on an infinite basis)

$$y(t) = \phi(\xi_0, \xi_1, \xi_{10}, \xi_{100}, \xi_{110}, \cdots).$$

If the output $y(t)$ depends only on a finite subset of the set of Lyndon words, this expansion becomes an analytical function of the corresponding ξ_l.

3.3. Computational example

We show as example how the computation is done if the output depends only of the three Lyndon words $z_0, z_1, z_0 z_1$.

Then we have:

$$y = \phi(\xi_0, \xi_1, \xi_2), \quad \text{where we have set } \ \xi_2 = \xi_{10}.$$

We have $a_0(t) \equiv 1$, and the first derivatives are given by:

$$\dot{\xi}_0 = 0, \quad \dot{\xi}_1 = a_1, \quad \dot{\xi}_2 = \dot{\xi}_{10} = \xi_1$$

and with initial conditions: $\xi_0(0) = \xi_1(0) = \xi_2(0) = 0$. We adopt the index notation for the partial derivatives, for instance:

$$\phi_{0122} = \frac{\partial}{\xi_0}\frac{\partial}{\xi_2}\frac{\partial^{2\,2}}{\xi_2}\phi.$$

Then we get as follows the derivatives of $y(t)$ at time $t = 0$:

$$y(0) = \phi(0,0,0).$$

The *first derivative* of y is: $\dot{y} = \phi_0 + \phi_1 a_1 + \phi_2 \xi_1$ and then

$$\dot{y}(0) = \phi_0(0) + \phi_1(0) a_1(0)$$

For an input that satisfies $a_1(0) = 0$, we get $\dot{y}(0) = \phi_0(0)$.

For an input that satisfies $a_1(0) = 1$, we get $\dot{y}(0) = \phi_0(0) + \phi_1(0)$.

Mixing these results, we deduce the values of ϕ_0 and ϕ_1.

The *second derivative* is:

$$\ddot{y} = \phi_{00} + \phi_{11} a_1^2 + \phi_{22}\xi_1^2 + 2\phi_{01} a_1 + 2\phi_{02}\xi_1 + 2\phi_{12} a_1 \xi_1 + \phi_1 \dot{a}_1 + \phi_2 a_1.$$

Specialized in $t = 0$, we obtain:

$$\ddot{y}(0) = \phi_{00} + a_1(2\phi_{01} + \phi_2) + a_1^2 \phi_{11} + \dot{a}_1 \phi_1.$$

By getting the value of $\ddot{y}(0)$ for some few choices of a_1 and $\dot{a}_1$, (and by inverting some Vandermonde matrix), we get, at time $t = 0$, the values of ϕ_{00} and ϕ_{11}, and we get anew the value of ϕ_1. We deduce also the value of $2\phi_{01} + \phi_2$.

The *third derivative* is:

$$\begin{aligned}\dddot{y} &= \phi_{000} + \phi_{111} a_1^3 + \phi_{222}\xi_1^3 + 3\phi_{001} a_1 + 3\phi_{002}\xi_1 + 3\phi_{011} a_1^2 \\ &\quad + 3\phi_{022}\xi_1^2 + 3\phi_{112} a_1^2 \xi_1 + 3\phi_{122} a_1 \xi_1^2 + 2\phi_{11} a_1 \dot{a}_1 + 2\phi_{22} a_1 \xi_1 + 2\phi_{01}\dot{a}_1 \\ &\quad + 2\phi_{02} a_1 + 2\phi_{12} a_1^2 + 2\phi_{12}\dot{a}_1 \xi_1 + \phi_1 \ddot{a}_1 + \phi_2 \dot{a}_1.\end{aligned}$$

Specialized in $t = 0$, we obtain:

$$\begin{aligned}\dddot{y}(0) &= \phi_{000} + a_1(3\phi_{001} + 2\phi_{02}) + a_1^2(3\phi_{011} + 2\phi_{12} a_1^2) + a_1^3 \phi_{111} \\ &\quad + \dot{a}_1(2\phi_{01} + \phi_2) + \ddot{a}_1 \phi_1 + a_1 \dot{a}_1 2\phi_{11}.\end{aligned}$$

By getting the value of $\dddot{y}(0)$ for some few adequate assignments of a_1, $\dot{a}_1$ and $\ddot{a}_1$, we deduce, at time $t = 0$, the values of ϕ_{000} and ϕ_{111}, we find new computations of ϕ_1 and ϕ_{11} and $(2\phi_{01} + \phi_2)$, and we get finally the values of the combined expressions:

$$(3\phi_{001} + 2\phi_{02}) \qquad (3\phi_{011} + 2\phi_{12} a_1^2).$$

The computation must continue, and we hope that can be done without encounter contradictory values.

3.3.1. Commentaries

The identification by this way, besides the hypothesis on the finite set of basis element, requires

- the iterated calculus of derivatives of composed function,
- a factorisation property allowing to separate and identify the involved linear combinations of partial derivatives of ϕ,
- a linear inversion algorithm for isolating these partial derivatives themselves from their occuring linear combinations.

It seems that this process will be carried out only for *some simple cases.* Moreover, this computation is specific to the selected finite set of Lyndon words. *The computation is lost if we have to take in account additional Lyndon words.*

4. A non commutative combinatorial computation

In these part, we shall define a family of noncommutative polynomials $\mathbf{g}_{1^p\otimes\nu}$ describing the system contribution relative to each differential monomial in the inputs and its derivatives. The algorithm that we present now is working on two steps:

- the first one is devoted to identify all the contributions $\langle G \mid \mathbf{g}_{1^p\otimes\nu}\rangle$, by chosing an adequate panel of iterated derivatives of the input at time $t = 0$. The algorithm has been completely presented in [4], and an implementation is in progress.
- the second step is devoted to acquire separetely the contribution of each non commutative word $w \in Z^\star$, by expanding each word w a linear combination of the non commutative polynomials $\mathbf{g}_{1^p\otimes\nu}$. We shortly present here summarily the computations and results that we present with more details in [5].

We begin that part by giving the combinatorial interpretation of the multiderivatives of the Chen series.

4.1. Combinatorial analysis of differential monomials

4.1.1. Homogeneous case on a single input

On a single input $a_1 = a$, the *differential monomials* must be written:

$$a^\mu = \left(a^{(i_1-1)}\right)^{e_1}\left(a^{(i_2-1)}\right)^{e_2}\ldots\left(a^{(i_q-1)}\right)^{e_q}, \quad 1 \leqslant i_1 < i_2 < \ldots < i_q.$$

Such a monomial a^μ is indexed by the following increasing integer sequence, that is called partition:

$$\mu = \left(i_1^{e_1} i_2^{e_2} \ldots i_q^{e_q}\right), \quad 1 \leqslant i_1 < i_2 < \ldots < i_q.$$

4.1.2. Derivation law

The **derivation law D on partitions** reflects the effect of time derivative $\frac{d}{dt}$ on differential monomials.

- For a differential letter: $\frac{d}{dt}(a^{(i_k)}) = a^{(i_{k+1})}$, and then: $D(i_k) = i_{k+1}$.
- We extend D to any partition by the Leibnitz derivation rule.

Then we can write:

$$D\mu = D\left(i_1^{e_1} i_2^{e_2}, \ldots, i_q^{e_q}\right) = \sum_{k=1\ldots q} \mathbf{e_k}\left[i_{k+1}\frac{\partial}{\partial i_k}(\mu)\right]^+.$$

Thus, $D\mu$ is a linear combinations of partitions. For example, we have:

$$D(1^1 2^3 3^2 6^1) = 2^4 3^2 6^1 + 3 1^1 2^2 3^3 6^1 + 2 1^1 2^3 3^1 4^1 6^1 + 1^1 2^3 3^2 7^1.$$

We obtain so a linear equation, with adequate definition of the coefficients $\langle\mu|\tau\rangle$:

$$D\mu = \sum_{\tau \in \text{partitions}} \langle\mu|\tau\rangle\tau \quad \text{with} \quad \langle\mu|\tau\rangle \in \mathbb{N}.$$

Remark. If $\langle\mu|\tau\rangle \neq 0$, then $\text{size}(\mu) = \text{size}(\tau)$. In other words, the operator D preserves the size.

4.1.3. Nonhomogenous case and single input

In this case, we use a fictious input $a_0(t) \equiv 1$. Thus there are in fact two inputs a_0 and a_1, and we have to use *two-colors partitions*. But the differential symbol a_0 is not free: it is subject to the relations $a_0 \equiv 1$ and $\dot{a}_0 \equiv 0$.

Hence the 2-colored mutiplicities of weight n must be written as:

$$\mu = \mu_0 \otimes \nu \quad \text{where} \begin{cases} \mu_0 = 1^p \\ \text{wgt}\,(\mu) = p + \text{wgt}\,(\nu). \end{cases}$$

The derivation law must be adapted. Indeed, we have to set $D(\mu_0) = 0$, and then:

$$D(1^p \otimes \nu) = 1^p \otimes D(\nu).$$

The integer p is called *the depth* of the colored partition $1^p \otimes \nu$. In addition we have:

$$\text{wgt}\,(1^p \otimes \nu) = p + \text{wgt}\,(\nu) \quad \text{and} \quad \text{size}(1^p \otimes \nu) = p + \text{size}(\nu).$$

4.2. Output derivatives and the generic equation

4.2.1. Iterated output derivatives

Since the generating series G_Σ does not depend of time, we get the following formula for computing the iterated derivatives of the output $y(t)$ of the functional:

$$\frac{d^n}{dt^n}y(t) = \langle G_\Sigma \mid\mid \frac{d^n}{dt^n}\mathcal{C}_{\mathbf{a}}(t)\rangle = \langle G_\Sigma \mid\mid C_{\mathbf{a}}(t)\mathbf{A}_n(t) >$$

and if we compute in time $t = 0$, that becomes:

$$\frac{d^n}{dt^n}y(0) = \langle G_\Sigma \mid\mid \mathbf{A}_n(0)\rangle.$$

4.2.2. The generic equation

Let us denote by $\mathcal{A}$ the infinite sum

$$\mathcal{A} = \sum_{n\in\mathbb{N}} \mathbf{A}_n \quad \text{where we set} \quad \mathbf{A}_0 = 1.$$

This series satisfy the following **generic equation**:

$$\mathcal{A} = 1 + \mathcal{L}_{\mathbf{a}}\mathcal{A} + \frac{d}{dt}\mathcal{A} \quad \text{où} \quad \mathcal{L}_{\mathbf{a}} = \sum_{z_j\in Z} a_j(t)z_j$$

This equation sums up a lot of recurrence relations. We underline the expansion of $\mathcal{A}$ on the differential monomials by setting:

proclaimDefinition 4.1 We define as follows the family of polynomials $(\mathbf{g}_{1^p\otimes\nu})$ by the expansion:

$$\mathcal{A} = \sum_{\nu\in\text{partitions}} a^{1^p\otimes\nu}\, \mathbf{g}_{1^p\otimes\nu}.$$

Theorem 1 ([4]). *By adequate choice panels of the inputs, we can effectively compute each of the coefficients* $\langle G \mid \mathbf{g}_{1^p\otimes\nu}\rangle$.

This theorem is proved in [4], and an implementation of this result is in progress.

4.2.3. Analysis of the generic equation

We analyse the generic equation by factorizing, in the two members, the multipliers (in fact non commutative polynomials) involving the same multiplicity. We obtain:

$$\mathbf{g}_{1^p\otimes\nu} = z_0\mathbf{g}_{1^{p-1}\otimes\nu} + z_1\mathbf{g}_{1_{1^p\otimes\nu\triangleright 1}} + \sum_\sigma \langle\sigma|\nu\rangle \mathbf{g}_{1^p_1\otimes\sigma},$$

where the symbol $\triangleright$ is used a follows:

$$\nu \triangleright 1 = \begin{cases} \sigma & \text{if} \quad 1\sigma = \nu, \\ 0 & \text{in other cases.} \end{cases}$$

The coefficients $\langle \sigma | \nu \rangle$ rely σ to its *derivatives* and ν to its *primitives.*

Thus, in the right hand side, the first term vanishes if $p = 0$.

The second term vanishes if the partition ν has no part equal to 1.

The third sum, extended to the "primitives" σ of ν, is finite.

Let us rewrite these equations as follows:

$$\mathbf{g}_{1^p \otimes \nu} - \sum_{\sigma} \langle \sigma | \nu \rangle \mathbf{g}_1^p \otimes \sigma = z_0 \mathbf{g}_{1^{p-1} \otimes \nu} + z_1 \mathbf{g}_{1 1^p \otimes \nu \triangleright 1}.$$

Now the left member is a linear combinations of partitions of the same size $n = p + \mathrm{size}(\nu)$. The right member is a linear combinations of partitions of size $n - 1$. Thus this equation expressed the left member in terms of "smaller ones." We shall use this fact to put the partitions occurring in this left member into a triangular set.

4.3. Matricial computation

The matrix computation of the algebraic identification problem is done by a vector encoding of the recursive equation. This is make easy by a natural bijection between partitions and words.

4.3.1. Partitions and words ordering

For any integers m and k, let $\mathcal{M}_m^k$ the set of partitions that can be written in the form:

$$j_1 j_2 \cdots j_m \quad \text{with} \quad 1 \leqslant j_1 \leqslant j_2 \leqslant j_3 \leqslant \ldots \leqslant j_m \leqslant k$$

ordered by the lexicographical ordering.

Let us set $z_1 < z_0$, and let $Z^{m,k}$ the set of words on Z, of length m in z_1 and $k-1$ on z_0, ordered by the lexicographical ordering. Then the application defined by

$$j_1 j_2 \cdots j_m \mapsto z_0^{j_1 - 1} z_1 z_0^{j_2 - j_1} z_1 z_0^{j_3 - j_2} \ldots z_1 z_0^{j_m - j_{m-1}} z_1 z_0^{k - j_m}$$

is a ordered bijection of $\mathcal{M}_m^k$ on $Z^{m,(k-1)}$. Their common cardinality is equal to $\binom{m+k-1}{m}$.

4.3.2. Vector encoding

Let k a fixed integer. For any integers m, p we note $\mathbf{G}^m/p(k)$ the column vector of all the polynomials $\mathbf{g}_{1^p\otimes\nu}$ for $\nu \in \mathcal{M}_m^k$, presented in the lexicographical ordering. We obtain then a vector equation:

$$(Id - \mathcal{P}_m^k) \circ \mathbf{G}_{/p}^m(k) = z_1 \mathrm{ext}\,(\mathbf{G}_{/p}^{m-1}(k), 0) + z_0\ \mathbf{G}_{/p-1}^m(k),$$

where we denote by $\mathrm{ext}\,(\mathbf{G}_{/p}^{m-1}(k), 0)$ the column vector $\mathbf{G}_{/p}^{m-1}(k)$ (of dimension $\binom{m+k-2}{m-1}$) completed by polynomials equal to 0 up to dimension $\binom{m+k-1}{m}$. The transformation $\mathcal{J}_m^k = (Id - \mathcal{P}_m^k)$ is represented by a matrix $\mathbf{J}_m^k$, where appear only some primitive coefficients, and which is sparse and lower triangular, with 1 only on the main diagonal. We denote by $\mathbf{T}_m^k$ the inverse matrix of $\mathbf{J}_m^k$.

4.3.3. The recursive splitting

The previous matrix equation leads to a very easy recursive computation of the matrices of the polynomials $\mathbf{g}_{1^p\otimes\nu}$. It leads also to an analysis of these matrices by a recursive spitting, and to compute so the vector $\mathbf{G}_{/p}^m(k)$ on the basis $Z^{m,k}$. In the right member of the previous equation, the splitting sum with the prefixes z_1 and z_0 must then be interpreted as horizontal concatenation of sub-blocks (that we note here by $/\!\!/$). We obtain so:

$$\mathbf{G}_{/p}^m(p) = \mathbf{T}_m^p \left[\begin{pmatrix} \mathbf{G}_{/p}^{m-1}(p) \\ 0 \end{pmatrix} /\!\!/ \mathbf{G}_{/p-1}^m(p) \right]$$

$$\mathbf{G}_{/p-1}^m(p) = \mathbf{T}_m^p \left[\begin{pmatrix} \mathbf{G}_{/p-2}^{m-1}(p) \\ 0 \end{pmatrix} /\!\!/ \mathbf{G}_{/p-2}^m(p) \right]$$

$$\ldots \mathbf{G}_{/1}^m(p) = \mathbf{T}_m^p \left[\begin{pmatrix} \mathbf{G}_{/1}^{m-1}(p) \\ 0 \end{pmatrix} /\!\!/ \mathbf{G}_{/0}^m(p) \right]$$

$$\mathbf{G}_{/0}^m(p) = \mathbf{T}_m^p \begin{pmatrix} \mathbf{G}_{/0}^{m-1}(p) \\ 0 \end{pmatrix} \ldots \mathbf{G}_{/0}^1(p) = \mathbf{T}_m^p \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

A development of these techniques allows to achieve a recursive splitting of the main matrix. In [5], we were albe to identify so all the words of length less or equal to 11. (A set of 2048 words).

5. Conclusion

We have presented two way of solve the Algebraic Identification Problem. The commutative analysis allows to solve the problem (if we suppose that a solution exists) in the hypothesis that the output function depens only of a small finite numbers of Shirshov words.

The second way appears as a generic method that allows to precise a priori the required choice panels of input derivatives at time $t = 0$. And the second part is completely generic, and can be done once for all.

References

1. K. T. Chen, *Iterated path integrals. Bull. Am. Math. Soc.*, **83** (1977), 831–879.
2. M. Fliess, *Fonctionnelles causales non linéaires et indéterminées non commutatives. Bull. Soc. Math. France*, **109** (1981), 3–40.
3. M. Fliess, *On the concept of derivatives and Taylor expansions for nonlinear input/output systems. Proceedings of the 22nd IEEE Conf. Decision and Control*, San Antonio, Texas (1983), pp. 643–646.
4. C. Hespel and G. Jacob, *First steps towards Exact Algebraic Identification. Discrete Math.*, **180** (1998), 211–210.
5. C. Hespel and G. Jacob, *On Algebraic Identification of Causal Functionals.* FPSCA98 (*Formal Power Series ans Algebraic Combinatorics*), Toronto, CANADA, 15–19 June (1998).
6. G. Jacob, *Algebraic methods and computer algebra for nonlinear systems' study.* IMACS Symposium MCTS, *Modelling and Control of Technochological systems*, **2** (Lille, 1991), pp. 599–608.
7. D. E. Radford, *A natural ring basis for shuffle algebra and an application to group schemes. J. Algebra*, **58** (1979), 432–454.
8. H. J. Sussmann, *A product expansion for the Chen series. Theory and Applications of nonlinear Control Systems* C. Byrnes and A. Lindquist Eds., (Elsevier Science Publishers, North-Holland), **32** (1986), pp. 3–335.
9. Y. Wang and E. D. Sontag, *On two definitions of observation spaces. Systems and Control letters*, **13** (1989), 279–289.

LANS, INSA,
20 Avenue Buttes de COESMES,
35 043 RENNES Cedex,
France

Поступило 12 мая 1999 г.

LIFL (URA 369 CNRS)
Université Lille I,
59024 Lille Cedex,
France