An Algorithm for Converting Nonlinear Differential Equations to Integral Equations with an Application to Parameter Estimation from Noisy Data

François Boulier, Anja Korporal, François Lemaire, Wilfrid Perruquetti, Adrien Poteaux, Rosane Ushirobira

An Algorithm for Converting Nonlinear Differential Equations to Integral Equations with an Application to Parameter Estimation from Noisy Data

François Boulier[1], Anja Korporal[2], François Lemaire[1], Wilfrid Perruquetti[2,3], Adrien Poteaux[1], and Rosane Ushirobira[2]

[1] Université Lille 1, LIFL, UMR CNRS 8022, Computer Algebra Group,
FirstName.LastName@univ-lille1.fr
[2] Inria, Non-A team, FirstName.LastName@inria.fr
[3] École Centrale de Lille, LAGIS, UMR CNRS 8219

Abstract. This paper provides a contribution to the parameter estimation methods for nonlinear dynamical systems. In such problems, a major issue is the presence of noise in measurements. In particular, most methods based on numerical estimates of derivations are very noise sensitive. An improvement consists in using integral equations, acting as noise filtering, rather than differential equations. Our contribution is a pair of algorithms for converting fractions of differential polynomials to integral equations. These algorithms rely on an improved version of a recent differential algebra algorithm. Their usefulness is illustrated by an application to the problem of estimating the parameters of a nonlinear dynamical system, from noisy data.

In Engineering, a wide variety of information is not directly obtained through measurement. Various parameters or internal variables are unknown or not measured. In addition, sensor signals are very often distorted and tainted by measurement noises. To simulate, control or supervise such processes, and to extract information conveyed by the signals, a system has to be identified and parameters and variables must be estimated. Most of traditional estimation methods are related to asymptotic statistics. However, there exist some difficulties that have been long known as inherent to these existing methods. Among them, two important limitations can be pointed out: these methods apply essentially to linear systems and they are noise sensitive due to the use of numerical derivation. The parameter estimation problem has been tackled by many different approaches in control theory. Algebraic techniques to this end were notably introduced in the works by M. Fliess et al. [8, 15, 7, 9, 6] and inspired for instance, algebraic methods for the parameter estimation of a multi-sinusoidal waveform signal from noisy data [22].

This paper[4] provides a contribution to these issues. Two algorithms are provided which convert differential equations to integral equations. They rely on a differential algebra [18, 14] algorithm for integrating differential fractions, which was presented in [3, Algorithm 4]. They are applied on the so-called *differential input-output equation* of a given nonlinear dynamical system in order to obtain *integral input-output equations*. For some systems, such as the one considered in this paper, the integral equation does not involve any derivative of the time varying variables. This property implies that its numerical evaluation does not require any numerical derivation. Indeed, numerical experiments confirm that, on white noisy data, integral forms of an input-output equation yield much better estimates of parameters than differential forms. It is well-known that numerical integration process has a filtering property on noisy data.

The paper is organized as follows. First, parameter estimation methods are presented in Section 1, notably by applying the notion of modulating functions. This approach is classical in automatic control theory and perhaps it is not so well-known in other fields and might be of interest for experts in integro-differential algebras, for instance. Section 1 features our new algorithms as well. Basic notions of differential algebra, required to understand the new algorithms, are presented in Section 2. An improved version of [3, Algorithm 4] is presented as Algorithm 3 in Section 3, together with additional properties (Propositions 1 and 2). Finally, two algorithms for computing integral equations are presented in Section 4 as Algorithms 4 and 5.

1 A parameter estimation method

1.1 Problem formulation

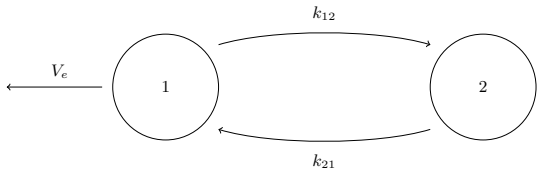

Fig. 1. A two-compartment model featuring three parameters.

We consider the academic two-compartment model depicted in Figure 1. Compartment 1 represents the blood system and compartment 2 represents some organ. A medical drug is injected in compartment 1 at $t = 0$. It is diffused between the two compartments, following linear laws: the proportionality

[4] This work was partially supported by the French ANR-10-BLAN-0109 LEDA project.

constants are named k_{12} and k_{21}. The drug exits compartment 1, following a law of Michaelis-Menten type. Such a law indicates an implicit enzymatic reaction and in general, it depends on two constants V_e and k_e. For the sake of simplicity, it is assumed that $k_e = 1$.

The state variables in this system are $x_1(t)$ and $x_2(t)$. They represent the concentrations of drugs in each compartment. The system has no input. Its output, denoted $y(t)$, is equal to $x_1(t)$, meaning that some numerical data are available for $x_1(t)$. No data is available for $x_2(t)$. To simplify the problem formulation, assume that both compartments have unit volumes. We obtain the following nonlinear dynamical system, which features three parameters to be estimated : k_{12}, k_{21} and V_e.

$$\begin{aligned} \dot{x}_1(t) &= -k_{12}\, x_1(t) + k_{21}\, x_2(t) - \frac{V_e\, x_1(t)}{1 + x_1(t)}\,, \\ \dot{x}_2(t) &= k_{12}\, x_1(t) - k_{21}\, x_2(t)\,, \\ y(t) &= x_1(t)\,. \end{aligned} \tag{1}$$

Estimation methods are applied on data obtained as follows: some made-up numerical values are assigned to the three parameters and the two initial values $x_1(0)$ and $x_2(0)$. A numerical curve is obtained by numerically integrating (1). Some white Gaussian noise, depending on a given standard deviation σ, is added to the curve, for $\sigma \in [0, 0.2]$. These curves are displayed in Figure 2.

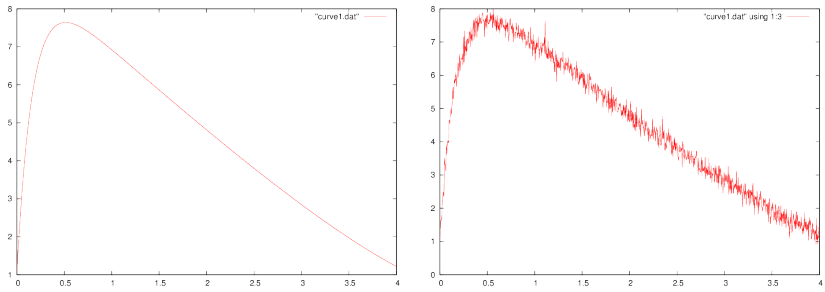

Fig. 2. The leftmost curve is obtained by numerically integrating (1) for $t = [0, 4]$, with $(x_1(0), x_2(0), k_{12}, k_{21}, V_e) = (1, 10, 1, 5, 3)$. The rightmost one is obtained by adding to it a white Gaussian noise with standard deviation $\sigma = 0.2$.

1.2 The input-output equations

In general, an input-output equation of a dynamical system is a differential equation, which belongs to the differential ideal defined by the model equations. It depends only on the input, the output, their derivatives and the model parameters (observe that our example has no input). Such input-output relations are

used for parameter identification since the only measurable variables are the input and the output. See [5, 8, 9, 6, 7, 17] and references therein. Using a differential elimination method [2] and a ranking that eliminates state variables:

$$\cdots > \ddot{x}_2 > \ddot{x}_1 > \dot{x}_2 > \dot{x}_1 > x_2 > x_1 > \cdots > \ddot{y} > \dot{y} > y > (k_{12},\, k_{21},\, V_e)\,,$$

it is possible to automatically compute the differential input-output equation of (1). Since our new algorithms are all about rewriting a single equation into a more convenient form, it is important to give the result almost as in the same form as it is returned by the elimination procedure:

$$\begin{aligned}&\ddot{y}(t)\,y(t)^2 + 2\,\ddot{y}(t)\,y(t) + \ddot{y}(t)\\ &\qquad + \dot{y}(t)\,y(t)^2\,\theta_2 + 2\,\dot{y}(t)\,y(t)\,\theta_2 + \dot{y}(t)\,\theta_3 + y(t)^2\,\theta_1 + y(t)\,\theta_1 = 0\end{aligned} \tag{2}$$

where the θ_i stand for the blocks of parameters:

$$\theta_1 = k_{21}\,V_e\,, \quad \theta_2 = k_{12} + k_{21}\,, \quad \theta_3 = k_{12} + k_{21} + V_e\,.$$

Dividing (2) by the coefficient of $\ddot{y}(t)$ (its *initial*, in the terminology of differential algebra) and observing it can be factored, one obtains a normalized differential input-output equation:

$$\theta_1\,\frac{y(t)}{y(t)+1} + \theta_2\,\frac{y(t)\,\dot{y}(t)\;(y(t)+2)}{(y(t)+1)^2} + \theta_3\,\frac{\dot{y}(t)}{(y(t)+1)^2} = -\ddot{y}(t) \tag{3}$$

Equation (3) depends on the first and the second derivative of $y(t)$. Before applying our algorithms, let us take a few lines to see what can be done easily or not from this equation, *i.e.* where the issue lies. It is actually easy to decrease by one the order of (3) since $\ddot{y}(t)$ occurs with degree one. Indeed

$$\int_a^t \ddot{y}(t)\;\mathrm{d}t = \int_a^t \frac{\mathrm{d}}{\mathrm{d}\tau}\,\dot{y}(\tau)\;\mathrm{d}\tau = \left[\dot{y}(\tau)\right]_a^t = \dot{y}(t) - \dot{y}(a)\,. \tag{4}$$

It is easy to obtain the following equivalent equation by integrating (3):

$$\begin{aligned}\theta_1 \int_a^t \frac{y(\tau)}{y(\tau)+1}\;\mathrm{d}\tau + \theta_2 \int_a^t \frac{y(\tau)\,\dot{y}(\tau)\;(y(\tau)+2)}{(y(\tau)+1)^2}\;\mathrm{d}\tau\\ + \theta_3 \int_a^t \frac{\dot{y}(\tau)}{(y(\tau)+1)^2}\;\mathrm{d}\tau = -\dot{y}(t) + \dot{y}(a)\,.\end{aligned} \tag{5}$$

Let us stress that in (4), the integral operator and the derivation operator are simplified thanks to the fact that the derivation operator is factored out under the integral sign, *i.e.* the expression has the following form. However, the derivation operators still occuring under the integral signs of (5) do not satisfy this pattern and the simplification cannot be performed easily. There lies the issue.

$$\int_a^t \frac{\mathrm{d}}{\mathrm{d}\tau}\,\text{something (possibly nonlinear)}\,\mathrm{d}\tau\,. \tag{6}$$

A recent algorithm [3, Algorithm 4] applied to equation (3) solves it and returns the following expression. This new equation (7) is a sum of expressions that are prepared to be in the form of (6). Moreover, all differential fractions in the above equation have order zero.

$$\theta_1 \frac{y(t)}{y(t)+1} + \theta_2 \frac{\mathrm{d}}{\mathrm{d}t} \frac{y(t)^2}{y(t)+1} - \theta_3 \frac{\mathrm{d}}{\mathrm{d}t} \frac{1}{y(t)+1} = -\frac{\mathrm{d}^2}{\mathrm{d}t^2} y(t) \tag{7}$$

We now describe the two possibilities for computing an integral equation from (7).

First approach. Apply twice the integration operator on (7). It results an integral input-output equation, that can be used for estimating parameters. This formula still involves a derivative: $\dot{y}(a)$. We consider it as a new parameter to be estimated.

$$\begin{aligned} &\theta_1 \int_a^t \int_a^{\tau_1} \frac{y(\tau_2)}{y(\tau_2)+1} \,\mathrm{d}\tau_2 \,\mathrm{d}\tau_1 \\ &+ \theta_2 \left(\int_a^t \frac{y(\tau)^2}{y(\tau)+1} \,\mathrm{d}\tau - \frac{y(a)^2}{y(a)+1} (t-a) \right) \\ &- \theta_3 \left(\int_a^t \frac{1}{y(\tau)+1} \,\mathrm{d}\tau - \frac{1}{y(a)+1} (t-a) \right) - \dot{y}(a)\,(t-a) = -y(t) + y(a) \end{aligned} \tag{8}$$

Second approach. A second possibility is to use some particular filter functions called *modulating functions*. They were introduced by M. Shinbrot in the 50's for system identification problems [20]. Shinbrot suggested the use of integral transformations on these problems to facilitate the identification for higher-order nonlinear dynamical systems. In addition, the effets of the initial conditions are annihilated by the modulating functions making this method more propitious to applications on noisy signals. Other authors have used modulating functions for estimating parameters for different two-compartment models, for instance A. Pearson applies Fourier modulating functions in [17] and K. Godfrey applies a successive derivatives method for the particular model considered here [11].

A modulating function of order n is a real-valued function $\phi_n(\tau)$ defined on a time interval $[a, t]$ that satisfies the $2n$ end-point conditions:

$$\frac{\mathrm{d}^\ell \phi_n}{\mathrm{d}\tau^\ell}(a) = \frac{\mathrm{d}^\ell \phi_n}{\mathrm{d}\tau^\ell}(t) = 0\,, \qquad 0 \le \ell < n\,.$$

Thus integration by parts yields for any function $f(\tau)$:

$$\begin{aligned} \int_a^t \phi_n(\tau) \frac{\mathrm{d}}{\mathrm{d}\tau} f(\tau) \,\mathrm{d}\tau &= \left[\phi_n(\tau) f(\tau)\right]_a^t - \int_a^t \left(\frac{\mathrm{d}}{\mathrm{d}\tau} \phi_n(\tau) \right) f(\tau) \,\mathrm{d}\tau \\ &= -\int_a^t \left(\frac{\mathrm{d}}{\mathrm{d}\tau} \phi_n(\tau) \right) f(\tau) \,\mathrm{d}\tau\,. \end{aligned}$$

Hence, multiplying (7) by a modulating function $\phi_2(\tau)$, integrating once and applying integration by parts as many times as needed gives a second integral input-output equation, that can be used for estimating parameters:

$$\begin{aligned}\theta_1 \int_a^t \phi_2(\tau)\,\frac{y(\tau)}{y(\tau)+1}\,\mathrm{d}\tau - \theta_2 \int_a^t \dot{\phi}_2(\tau)\,\frac{y(\tau)^2}{y(\tau)+1}\,\mathrm{d}\tau \\ + \theta_3 \int_a^t \dot{\phi}_2(\tau)\,\frac{1}{y(\tau)+1}\,\mathrm{d}\tau = -\int_a^t \ddot{\phi}_2(\tau)\,y(\tau)\,\mathrm{d}\tau\,.\end{aligned} \tag{9}$$

For our experiments, we tested three types of modulating functions:

- *Hartley modulating functions* $\phi_n(\tau)$, where $\mu = 0$, $n = 2$ and $\mathrm{cas}(\tau) = \cos(\tau) + \sin(\tau)$, for all τ.

$$\phi_n(\tau) = \sum_{\ell=0}^{n} (-1)^\ell \binom{n}{\ell} \mathrm{cas}\left((n+\mu-\ell)\,\frac{2\,\pi}{t}\,\tau\right)\,, \tag{10}$$

- *Modulating functions based on Hermite polynomials*: they are based on the weight functions for Hermite polynomials by a change of variables $t \mapsto \frac{t-2\,\tau}{t}$:

$$\phi(t) = \mathrm{e}^{-\alpha\left(\frac{t-2\,\tau}{t}\right)^2}\,. \tag{11}$$

  Remark that the graph of $\phi(t)$ is symmetric with respect to $\tau = \frac{t}{2}$. Strictly speaking, $\phi(t)$ is not a modulating function since it does not vanish at $\tau = 0$ and $\tau = t$. However, its values are arbitrarily close to zero at $\tau = 0$ and $\tau = t$, depending on the factor α. Experiments were performed using $\alpha = \frac{25}{2}$.
- *Modulating polynomial functions defined by interpolation*: we set

$$\phi(t) = a_4\,\tau^4 + \cdots + a_1\,\tau + a_0\,, \tag{12}$$

  such that $\phi(0) = \phi(t) = \frac{d\phi}{\mathrm{d}\tau}(0) = \frac{d\phi}{\mathrm{d}\tau}(t) = 0$ and $\phi\left(\frac{t}{2}\right) = 1$. A polynomial of degree four is sufficient to obtain a modulating function of order $n = 2$.

1.3 Method, Implementation and Results

The method (we borrowed it from [16, 5]) consists in evaluating a chosen input-output equation for many different values of t, over the available data. Thereby, one gets an overdetermined linear system that can be solved by linear least squares. If one chooses a differential equation (3), (7) or the integral equation (9) which relies on modulating functions, the unknowns are the blocks of parameters θ_1, θ_2 and θ_3. If one chooses the integral equation (8), the unknowns are the blocks of parameters plus the extra parameter $\dot{y}(a)$. In our experiments, we picked many different values (about 20) for a. For each new value of a, we had to introduce a new indeterminate.

The different methods were implemented in FORTRAN 77. The code is available at [21]. Linear algebra (least squares) was performed using the LAPACK

library. The numerical integration of the dynamical system (needed to produce the experimental data) was performed using the RADAU integrator, borrowed from [13]. The code for evaluating the modulating functions and their derivatives was produced in FORTRAN, from MAPLE, using the `CodeGeneration` package of MAPLE. The input-output equation (2) was produced by the BLAD libraries [1], through the `DifferentialAlgebra` package of MAPLE. The numerical derivations needed when using (3), (7) were computed over degree 3 polynomials best fitting about 20 consecutive data points. The numerical integrations were performed using the composite Simpson's formula.

For each input-output equation (3), (7), (8), (9), the accuracy of the estimation of the blocks of parameters θ_1, θ_2 and θ_3 was tested over noisy data, varying the noise from $\sigma = 0$ to $\sigma = 0.8$. For each value of σ, about 50 different simulations were performed.

Experiments show that the methods based on the two integral equations (8), and (9) give the best results. Modulating functions give simpler formulas and algorithms but need to be tuned carefully. See Figure 3 for details. Tuning was possible here, because we knew in advance the exact values of the parameter blocks. In a real life situation, it would be more complicated.

The methods based on the two different formulations of the differential input-output equation give similar results. We were expecting the one based on (7) to be more accurate, but we observed the opposite phenomenon.

2 Basic Notions of Differential Algebra

The reference books for this Section are [18, 14]. In this paper, we restrict ourselves to ordinary differential rings, which are rings endowed with a single derivation, that we assimilate do $\delta = \mathrm{d}/\mathrm{d}t$. To this derivation, one associates an *independent variable* t, defined by $\delta\, t = 1$. One denotes $\mathscr{X} = \{t\}$. In order to form differential polynomials, one introduces a set $\mathscr{U} = \{u_1, \ldots, u_n\}$ of n *differential indeterminates*. In Engineering, differential indeterminates would be called *dependent variables*. The derivatives of various orders of the differential indeterminates are simply called *derivatives*. We sometimes write $\dot{u}$ instead of $\delta\, u$ and $\ddot{u}$ instead of $\delta^2\, u$. The order of a derivative is the number of times it is differentiated: u, $\dot{u}$ and $\ddot{u}$ have respectively orders 0, 1 and 2.

For our concerns, it is crucial that one can also handle *parametric* differential equations. Parameters are nothing but symbolic *constants*, i.e., symbols whose derivatives are zero. Let $\mathscr{C}$ denote the set of constants.

The differential fractions considered in this paper are ratios of differential polynomials taken from the differential ring $\mathscr{R} = \mathbb{Z}[\mathscr{X} \cup \mathscr{C}]\{\mathscr{U}\}$, using the notations of [14]. A differential fraction is said to be *reduced* if its numerator and denominator do not have any common factor. A differential fraction is said to be a *coefficient* if it belongs to the field $\mathscr{K} = \mathbb{Q}(\mathscr{X} \cup \mathscr{C})$. The field $\mathscr{K}$ contains a field of constants, $\mathscr{K}_c = \mathbb{Q}(\mathscr{C})$. A fraction which is not in $\mathscr{K}$ thus depends on, at least, a derivative.

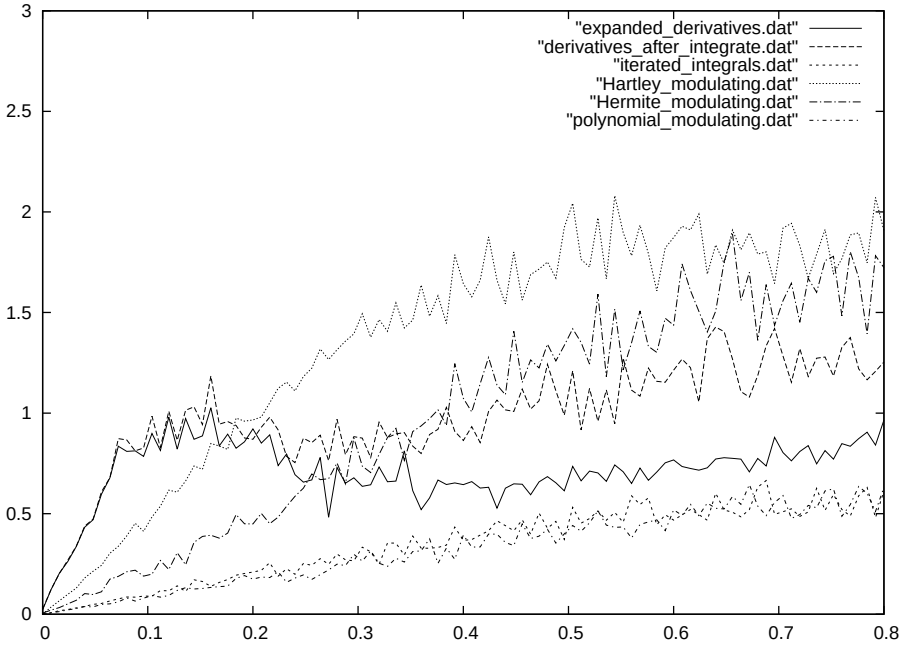

Fig. 3. Estimation of the blocks of parameters θ_i using different forms of the input-output equation. In abscissa, the standard deviation σ used to produce the white gaussian noise. In ordinate, the average value, for 50 different simulations, of the relative error, i.e. the 2-norm of the difference between the vector of estimated values and the vector of known values, divided by the 2-norm of the vector of known values. Therefore, all relative errors above 1 are equivalent, since they all correspond to computed values void of informations. For all formulas, two phases seem to occur: first, the relative error grows linearly with the standard deviation σ ; second, the error reaches some plateau. The best results are obtained using the integral equation (8) and the integral equation (9), with the polynomial modulating function (12). The two intermediate curves (for $\sigma \in [0, 0.15]$) are obtained using the integral equation (9), with modulating functions of Hermite type (11), below, and Hartley type (10), above. The worst results (for $\sigma \in [0, 0.15]$) are obtained by estimating derivatives on the differential equations (3) and (7). The curve corresponding to Equation (3) is strange since the error seems to decrease, temporarily, while σ increases. We suspect an artefact due to the example.

A *ranking* is a total ordering over the set of all derivatives that satisfies the two following axioms:

1. $u \leq \delta^n u$ for any $u \in \mathscr{U}$ and nonnegative integer n,
2. $u < v \Rightarrow \delta^n u < \delta^n v$ for all derivatives u, v and nonnegative integers n.

Rankings are well-orderings, i.e., every strictly decreasing sequence of derivatives is finite [14, §I.8]. Rankings such that $n < m \Rightarrow \delta^n u < \delta^m v$ for all nonnegative integers n, m and differential indeterminates $u, v \in U$ are called *orderly*.

Fix a ranking and consider some differential polynomial $P \notin \mathscr{K}$. The highest derivative v w.r.t. the ranking such that $\deg(P, v) > 0$ is called the *leading derivative* of P. The monomial $v^{\deg(P,v)}$ is called the *rank* of P. The leading coefficient of P w.r.t. v is called the *initial* of P.

Extensions of these definitions to differential fractions were introduced in [3]. The *order* of a reduced fraction $F \notin \mathscr{K}$ is the maximum of the orders of the derivatives it depends on. It is denoted $\operatorname{ord}(F)$. The order of fractions which are not in $\mathscr{K}$ is defined to be zero. The leading derivative of a fraction $F \notin \mathscr{K}$ is denoted $\operatorname{ld}(F)$. It is the highest derivative v, with respect to the ranking, such that $\partial F/\partial v \neq 0$. We do not need to define leading derivatives for elements of $\mathscr{K}$. Let $F = P/Q$ be a fraction which is not in $\mathscr{K}$. Its *degree* is defined as $\deg(F) = \deg(P, \operatorname{ld}(F)) - \deg(Q, \operatorname{ld}(F))$. Its rank is the pair $(\operatorname{ld}(F), \deg(F))$. Ranks of fractions are ordered lexicographically. Though we have not defined the rank of elements of $\mathscr{K}$, it is sufficient to consider that they have lower rank than any fraction which is not in $\mathscr{K}$.

Lemma 1. *If $F \notin \mathscr{K}$ is a reduced fraction then* $\operatorname{ord}(\delta F) = \operatorname{ord}(F) + 1$.

Proof. By [3, Proposition 3], we have $\operatorname{ld}(\delta F) = \delta \operatorname{ld}(F)$, whatever the ranking. In particular, this Proposition holds for orderly rankings. For such rankings, $\operatorname{ord}(F) = \operatorname{ord}(\operatorname{ld}(F))$. Thus the lemma is true for orderly rankings. Since the order of a fraction is ranking independent, the lemma is true for any ranking.

The expressions computed by Algorithms 4 and 5 are finite sums of differential fractions, integrated finitely many times from some lower bound a to the independent variable t, multiplied afterwards by elements of $\mathscr{K}$. Though, after a single call to one of our algorithms, a single lower bound a occurs, we would like to permit an expression to feature many different such bounds. Such expressions look like elements of integro-differential algebras. The introduction of [10] provides a nice survey on integro-differential algebras and their relationship with (differential) Rota-Baxter algebras. See also [19, 12]. A precise definition of an algebraic structure containing the output of our algorithms, and its algorithmic properties, could be much helpful for designing sound computer algebra packages. We do not provide it here.

In the sequel, we will use the classical notation for integral operators, with an explicit upper bound t and a trailing symbol $\mathrm{d}t$ which plays the role of a closing parenthesis. To lighten notations, we will however avoid to introduce new variables τ and write :

$$\int_a^t F(t)\,\mathrm{d}t \quad \text{instead of} \quad \int_a^t F(\tau)\,\mathrm{d}\tau\,.$$

3 Improvements of the integrate Algorithm

In this section, we recall the specifications of [3, Algorithms 3 and 4] and adapt them to the context of this paper. Moreover, we fix a bug, due to a possibly non-terminating auxiliary function. Two versions of the integrate algorithm were given: [3, Algorithm 3] is the core algorithm while [3, Algorithm 4] is its "iterated" version.

[3, Algorithm 3] gathers as input a differential fraction F_0 and an independent variable, which is t, here. It returns two differential fractions R and W such that

1. $F_0 = \delta\, R + W$,
2. W is zero iff there exists R such that $F_0 = \delta\, R$
3. The fractions $\delta\, R$ and W have ranks lower than or equal to that of F_0.

[3, Algorithm 4] gathers as input a differential fraction F_0 and an independent variable, which is necessarily t, here. It returns a possibly empty list (if is empty if, and only if $F_0 = 0$) of differential fractions $[W_0, W_1, \ldots, W_s]$ such that

1. W_s is nonzero
2. $F_0 = W_0 + \delta\, W_1 + \cdots + \delta^s\, W_s$
3. $W_0, W_1, \ldots, W_i$ are zero if, and only if there exists a differential fraction R and an index $i < s$ such that $F_0 = \delta^{i+1}\, R$
4. The differential fractions $W_0, \delta\, W_1, \ldots, \delta^s\, W_s$ have ranks lower than or equal to that of F_0.

Unfortunately, both these algorithms, as they are stated in [3], are flawed, for they rely over an auxiliary algorithm, [3, Algorithm 2, integrateWithRemainder], which may not terminate over some inputs. We take the opportunity of this paper to fix this mistake, by replacing [3, Algorithm 2, integrateWithRemainder] by [4, Mack's linear version of Hermite reduction, page 44]. The new version is provided in Algorithm 1.

Since $\alpha/(\alpha+u)+u/(\alpha+u) = 1$ one sees that a fraction whose denominator is free of α may very well be decomposed as a sum of fractions whose denominators depend on α. Therefore, though the next Proposition is not surprising, it cannot be considered as obvious.

Proposition 1. *Let α denote either a constant of $\mathscr{C}$ or a derivative, and $\mathscr{R}_\alpha$ the set of all fractions whose denominators are free of α. If $F_0 \in \mathscr{R}_\alpha$ then the two fractions R, W returned by [3, Algorithm 3] belong to $\mathscr{R}_\alpha$.*

Proof. The set $\mathscr{R}_\alpha$ is stable under addition, multiplication and derivations (as well δ as $\partial/\partial v$, for any v): it is a differential subring of the field of fractions of $\mathscr{R}$. Moreover, consider two differential polynomials A and B s.t. $\deg(B, \alpha) = 0$. Then, given any variable v (v may either be a constant or a derivative), the quotient and remainder of A by B w.r.t. v are elements of $\mathscr{R}_\alpha$.

We claim that, if $F_0 \in \mathscr{R}_\alpha$ then the two fractions returned by Algorithm 1 belong to $\mathscr{R}_\alpha$. Consider Algorithm 1 and assume $F_0 \in \mathscr{R}_\alpha$. Then, the polynomials $D, D^-, D^*, D^{-2}, D^{-*}$ are always free of α. Thus, Algorithm 2 is always

Algorithm 1 The integrateWithRemainder (fixed version) Algorithm is nothing but a slight variant of [4, Mack's linear version of Hermite reduction, page 44]

Require: F_0 a reduced fraction and v a variable
Ensure: Two fractions R and W such that (1) $F_0 = \partial R/\partial v + W$; (2) the denominator of W is squarefree; (3) $\deg(W) < 0$.

$cont :=$ the content of $\mathrm{denom}(F_0)$ w.r.t. v
$A := \mathrm{numer}(F_0)$
$D := \mathrm{denom}(F_0)/cont$
$G := 0$
$D^- :=$ the multivariate gcd of D and $\partial D/\partial v$
$D^* := D/D^-$
while $\deg(D^-, v) > 0$ **do**
  $D^{-2} :=$ the multivariate gcd of D^- and $\partial D^-/\partial v$
  $D^{-*} := D^-/D^{-2}$
  $(B, C) :=$ extendedEuclidean$(-D^* (\partial D^-/\partial v)/D^-, D^{-*}, A, v)$
  $A := C - (\partial B/\partial v)\, D^*/D^{-*}$
  $G := G + B/D^-$
  $D^- := D^{-2}$
end while
$R := G/cont$
$W := A/(D^- cont)$
return R, W

Algorithm 2 The extendedEuclidean Algorithm is a restatement of the diophantine version of Euclide's extended algorithm, given in [4, Sect. 1.3, page 13]

Require: P_1, P_2 and A are multivariate polynomials and v is a variable. Viewed as univariate polynomials in v, the polynomials P_i are coprime.
Ensure: Two fractions B and C such that $B\,P_1 + C\,P_2 = A$

$G :=$ the gcd (a fraction) of P_1 and P_2 viewed as univariate polynomials in v
$S, T :=$ fractions such that $S\,P_1 + T\,P_2 = G$
$Q :=$ the quotient of A by G w.r.t. v
$S, T := Q\,S, Q\,T$
$Q, R :=$ the quotient and remainder of S by P_2 w.r.t. v
$B := R$
$C := T + Q\,P_1$
return B, C

called with its two first parameters, P_1 and P_2, free of α. Now, if $A \in \mathscr{R}_\alpha$ then, according to the remarks stated in the previous paragraph, the returned fractions $B, C \in \mathscr{R}_\alpha$. Initially, A is a polynomial, thus an element of $\mathscr{R}_\alpha$. Therefore, at line 11, we have $B, C \in \mathscr{R}_\alpha$ whence $A \in R_\alpha$ again. Turning this argument into an inductive proof, we see A is always an element of $\mathscr{R}_\alpha$. A similar argument proves that $G \in \mathscr{R}_\alpha$. The proof of the claim is then easily completed.

To complete the proof of the Proposition, it is necessary to study the code of [3, Algorithm 3]. We will not give details. As for Algorithm 1, the argument is a straightforward application of the remarks stated at the beginning of this proof.

When dealing with elements of $\mathscr{K}$, [3, Algorithm 4] may have a counter-intuitive behaviour. For instance, applied on a constant a, this algorithm returns the list $[a]$. However, applied on a differential polynomial $a + u(t)$, it returns $[u(t), a\,t]$. The reason is informally this one: the algorithm tries to integrate fractions as much as it can but stops whenever the current fraction is an element of $\mathscr{K}$, since such an element could be integrated indefinitely. Thus, elements of $\mathscr{K}$ are not handled the same way when they occur alone or mixed with fractions which are not in $\mathscr{K}$. In our context, this behaviour is a problem. To overcome it, we introduce a slight modification of [3, Algorithm 4], in Algorithm 3.

Algorithm 3 The integrate (coefficient free version) Algorithm is a slight variant of [3, Algorithm 4], which forbids the W_i $(i \geq 1)$ to be nonzero elements of $\mathscr{K}$

Require: F_0 a reduced differential fraction and an independent variable t

Ensure: A list $[W_0, W_1, \ldots, W_s]$ satisfying the same properties as [3, Algorithm 4] plus the following condition:

$$W_i \notin \mathscr{K}, \text{ unless it is 0, for each } 1 \leq i \leq s \tag{13}$$

1: $[W_0, W_1, \ldots, W_s]$:= the list of fractions returned by [3, Algorithm 4] over F_0 and t
2: **while** $s \geq 1$ and $W_s \in \mathscr{K}$ **do**
3: $\quad W_{s-1} := W_{s-1} + \delta\, W_s$
4: $\quad s := s - 1$
5: **end while**
6: **for** i from $s-1$ to 1 by -1 **do**
7: $\quad$ **if** $W_i \in \mathscr{K}$ **then**
8: $\quad\quad W_{i-1} := W_{i-1} + \delta\, W_i$
9: $\quad\quad W_i := 0$
10: $\quad$ **end if**
11: **end for**
12: **return** $[W_0, W_1, \ldots, W_s]$

The following Proposition is new.

Proposition 2. *Assume the ranking is orderly and consider some differential fraction F_0. Let $[W_0, W_1, \ldots, W_s]$ be the list returned by the application of Al-*

gorithm 3 to F_0 and t. Then $i + \mathrm{ord}(W_i) \leq \mathrm{ord}(F_0)$ for each $0 \leq i \leq s$. In particular, $s \leq \mathrm{ord}(F_0)$.

Proof. If $F_0 \in \mathscr{K}$ then $s = 0$ and the proposition is satisfied. Assume from now on that $F_0 \notin \mathscr{K}$. If $W_0 \in \mathscr{K}$ then its order is 0 and the condition $0 + \mathrm{ord}(W_0) \leq \mathrm{ord}(F_0)$ holds. If it is not, then none of the nonzero W_i belong to $\mathscr{K}$, by condition (13), stated in the specifications of Algorithm 3. Then, by Lemma 1, we have $\mathrm{ord}(\delta^i W_i) = i + \mathrm{ord}(W_i)$. Since $\delta^i W_i$ has rank lower than, or equal to F_0 (this specification of [3, Algorithm 4] holds for Algorithm 3 also) and the ranking is orderly, one concludes that $i + \mathrm{ord}(W_i) \leq \mathrm{ord}(F_0)$ for each $0 \leq i \leq s$. The last sentence is a consequence of this result, and the fact that Algorithm 3 forbids W_s to be an element of $\mathscr{K}$.

4 Two Algorithms for Computing Integral Equations

Observe that, in the specifications of Algorithm 4, the differential ring which contains (14) may be a proper differential ring extension of the differential ring $\mathscr{R}$ which contains the input fraction F_0. Indeed, in order to represent the values of the derivatives at $t = a$, one may need to add arbitrary constants [18, chapter III, page 57] to $\mathscr{R}$ and the integration process may generate polynomials in t. Note also that one may want to put some of these constants in the set of parameters Θ, as mentioned in the remark following (8).

Proposition 3. *Algorithm 4 terminates and is correct.*

Proof. Termination is clear. Let us address the correctness. According to the specifications of Algorithm 3, recalled in Section 3, we have the following formula. Algorithm 4 applies s times the integration from a to t operator over it:

$$F_0 = W_0 + \frac{\mathrm{d}}{\mathrm{d}t} W_1 + \cdots + \left(\frac{\mathrm{d}}{\mathrm{d}t}\right)^s W_s \,.$$

A difficulty needs to be adressed at line 4 since, without any information on the shape of W_i, the decomposition needs not be possible. First notice that, since Q_1 is a constant, the list of the W_i can be obtained at line 1 by applying first Algorithm 3 over P/Q_2, then dividing all the obtained fractions by Q_1. Then, notice that, with the notations of Proposition 1, the fraction $P/Q_2 \in \mathscr{R}_\alpha$ for any $\alpha \in \Theta$. According to this Proposition, all the W_i have the same shape as F_0 (i.e, if one excepts a possible factor $Q_1 \in K[\Theta]$ at the denominator, all parameters occur at the numerator of the fraction). In such a case, the decomposition performed at line 4 is possible.

Thanks to this special shape of the W_i, the blocks of parameters are moved outside the integral operators.

The exponent $\max(\ell_i)$ occuring in (15) is actually equal to the index s determined at line 1. The statement about orderly rankings follows Proposition 2.

Algorithm 4 The iteratedIntegral algorithm

Require: F_0, Θ, a where Θ is a set of parameters ; F_0 is a reduced ordinary differential fraction (dependent variable t) which can be written as the product of two differential fractions $F_0 = P/(Q_1\,Q_2)$ such that $Q_1 \in K[\Theta]$ and Q_2 does not depend on Θ ; and a is an evaluation point (numeric or symbolic)

Ensure: an expression of the form

$$F = \sum_{i=1}^{N} C_i \underbrace{\int_a^t \cdots \int_a^t}_{\ell_i \text{ times}} G_i \underbrace{\mathrm{d}t \cdots \mathrm{d}t}_{\ell_i \text{ times}} \tag{14}$$

such that the $C_i \in K(\Theta)$, the differential fractions G_i do not depend on Θ and the nonnegative integers ℓ_i satisfy the following property (in the case of an orderly ranking, we have $\ell_i + \mathrm{ord}(G_i) \leq \mathrm{ord}(F_0)$ for each i):

$$F_0 = \left(\frac{\mathrm{d}}{\mathrm{d}t}\right)^{\max(\ell_i)} F\,. \tag{15}$$

1: $[W_0, W_1, \ldots, W_s]$:= the list of fractions returned by Algorithm 3 over F_0 and t
2: $F := 0$
3: **for** i from 0 to s **do**
4: decompose $W_i = \sum_{k=1}^{n} C_k\,T_k$ where the $C_k \in K(\Theta)$ and T_k do not depend on Θ
5: **for** k from 1 to n **do**
6: $Q := 0$
7: **for** j from 0 to s **do**
8: **if** $j \leq i$ **then**
9: $B := \left(\frac{\mathrm{d}}{\mathrm{d}t}\right)^{i-j} T_k$
10: $B_a := B(a)$
11: **else**
12: $B := \underbrace{\int_a^t \cdots \int_a^t}_{j-i \text{ times}} T_k \ \underbrace{\mathrm{d}t \cdots \mathrm{d}t}_{j-i \text{ times}}$
13: $B_a := 0$
14: **end if**
15: $P := Q - B_a$
16: $Q := \int_a^t P\,\mathrm{d}t$
17: **end for**
18: $F := F + C_k\,(B + P)$
19: **end for**
20: **end for**
21: **return** F

Algorithm 5 The modulatedIntegral algorithm

Require: F_0, Θ, a, ϕ where Θ is a set of parameters ; F_0 is a reduced ordinary differential fraction (dependent variable t) which can be written as the product of two differential fractions $F_0 = P/(Q_1\,Q_2)$ such that $Q_1 \in K[\Theta]$ and Q_2 does not depend on Θ ; a is an evaluation point ; and ϕ is a function of t

Ensure: an expression of the form

$$F = \sum_{i=1}^{N} C_i \int_a^t \frac{\mathrm{d}^{\ell_i}\phi}{\mathrm{d}t^{\ell_i}}\, G_i \,\mathrm{d}t \tag{16}$$

such that the $C_i \in K(\Theta)$, the differential fractions G_i do not depend on Θ and, provided that ϕ is a modulating function of sufficient order, we have

$$F = \int_a^t \phi\, F_0 \,\mathrm{d}t\,. \tag{17}$$

In the case of an orderly ranking, we have $\mathrm{ord}(G_i) \le \mathrm{ord}(F_0)$ for each i, and the order of the modulating function can be chosen less than or equal to $\mathrm{ord}(F_0)$.

1: $[W_0, W_1, \ldots, W_s]$:= the list of fractions returned by Algorithm 3 over F_0 and t
2: $F := 0$
3: **for** i from 0 to s **do**
4: decompose $W_i = \sum_{k=1}^{n} C_k\, T_k$ where the $C_k \in K(\Theta)$ and T_k do not depend on Θ
5: **for** k from 1 to n **do**
6: $F := F + C_k\,(-1)^i \int_a^t \frac{\mathrm{d}^i\,\phi}{\mathrm{d}t^i}\, T_k \,\mathrm{d}t$
7: **end for**
8: **end for**
9: **return** F

Proposition 4. *Algorithm 5 terminates and is correct.*

Proof. Termination is clear. Let us address the correctness. The proof starts as that of Proposition 3. According to the specifications of Algorithm 3 we have

$$F_0 = W_0 + \frac{\mathrm{d}}{\mathrm{d}t} W_1 + \cdots + \left(\frac{\mathrm{d}}{\mathrm{d}t}\right)^s W_s \,.$$

The issue at line 4 is solved as in Proposition 3. Algorithm 5 then computes

$$\begin{aligned} F &= \int_a^t \phi\, F_0 \,\mathrm{d}t \\ &= \int_a^t \phi\, W_0 \,\mathrm{d}t + \int_a^t \phi \frac{\mathrm{d}}{\mathrm{d}t} W_1 \,\mathrm{d}t + \cdots + \int_a^t \phi \left(\frac{\mathrm{d}}{\mathrm{d}t}\right)^s W_s \,\mathrm{d}t \\ &= \int_a^t \phi\, W_0 \,\mathrm{d}t + (-1) \int_a^t \frac{\mathrm{d}\phi}{\mathrm{d}t} W_1 \,\mathrm{d}t + \cdots + (-1)^s \int_a^t \frac{\mathrm{d}^s\phi}{\mathrm{d}t^s} W_s \,\mathrm{d}t \,. \end{aligned}$$

The third formula is obtained by applying the integration by part axiom plus the hypothesis that ϕ is a modulating function of order at least s. Thanks to the shape of the W_i, blocks of parameters are moved outside the integral operators.

The statement concerning orderly rankings follows from Proposition 2 and the fact that the order of the modulating function should be s.

5 Conclusion

We have presented two algorithms for converting nonlinear differential equations to integral equations, taking this opportunity to fix a flaw[5] in a recent algorithm. Such conversions permit to decrease the orders of differential equations, a feature of dramatic importance whenever equations need to be numerically evaluated, since numerical derivation methods can be avoided, at least partially.

An interesting theoretical question, which was asked[6] to us, is left open in this paper: could we characterize the class of dynamical systems for which our algorithms provide order zero integral equations, i.e. equations whose numerical evaluations do not require any numerical derivation?

[5] The authors would like to thank Joseph Lallemand, who contributed to fix the flaw.
[6] The authors would like to thank Cédric Join and Mamadou Mboup.

Bibliography

[1] F. Boulier. `http://www.lifl.fr/~boulier/BLAD`, 2004.

[2] F. Boulier, Daniel Lazard, F. Ollivier, and M. Petitot. Computing representations for radicals of finitely generated differential ideals. *AAECC*, 20(1):73–121, 2009.

[3] F. Boulier, F. Lemaire, G. Regensburger, and M. Rosenkranz. On the Integration of Differential Fractions. In *ISSAC'13*, pages 101–108, New York, 2013. ACM.

[4] M. Bronstein. *Symbolic Integration I.* Springer Verlag, 1997.

[5] L. Denis-Vidal, G. Joly-Blanchard, and C. Noiret. System identifiability (symbolic computation) and parameter estimation (numerical computation). In *Numerical Algorithms*, volume 34, pages 282–292, 2003.

[6] M. Fliess, C. Join, and H. Sira-Ramírez. Non-linear estimation is easy. *Int. J. Modelling Identification and Control*, 4(1):12–27, 2008.

[7] M. Fliess, M. Mboup, H. Mounier, and H. Sira-Ramírez. Questioning some paradigms of signal processing via concrete examples. In G. Silva-Navarro H. Sira-Ramírez, editor, *in Algebraic Methods in Flatness, Signal Processing and State Estimation*, pages 1–21. Editorial Lagares, 2003.

[8] M. Fliess and H. Sira-Ramírez. An algebraic framework for linear identification. *ESAIM Control Optim. Calc. Variat.*, 9:151–168, 2003.

[9] M. Fliess and H. Sira-Ramírez. Identification of continuous-time models from sampled data. *chapter Closed-loop parametric identification for continuous-time linear systems via new algebraic techniques. Advances in Industrial Control*, pages 362–391, 2008.

[10] Xing Gao and Li Guo. Constructions of Free Commutative Integro-Differential Algebras. In M. Barkatou, T. Cluzeau, G. Regensburger, and M. Rosenkranz, editors, *Algebraic and Algorithmic Aspects of Differential and Integral Operators*, volume 8372 of *LNCS*, pages 1–22, 2014.

[11] K. R. Godfrey. The identifiability of parameters of models used in biomedicine. *Mathematical Modelling*, 7(912):1195 – 1214, 1986.

[12] Li Guo, G. Regensburger, and M. Rosenkranz. On integro-differential algebras. *JPAA*, 218(3):456–473, 2014.

[13] E. Hairer. Homepage. `http://www.unige.ch/~hairer`, 2000.

[14] E. R. Kolchin. *Differential Algebra and Algebraic Groups.* Academic Press, New York, 1973.

[15] M. Mboup. Parameter estimation for signals described by differential equations. *Applicable Analysis*, 88:29–52, 2009.

[16] C. Noiret. *Utilisation du calcul formel pour l'identifiabilité de modèles paramétriques et nouveaux algorithmes en estimation de paramètres.* PhD thesis, Université de Technologie de Compiègne, 2000.

[17] A. E. Pearson. Explicit parameter identification for a class of nonlinear input/output differential operator models. In *Decision and Control, 1992., Proceedings of the 31st IEEE Conference on*, pages 3656–3660 vol.4, 1992.

[18] J. F. Ritt. *Differential Algebra*, volume 33 of *American Mathematical Society Colloquium Publications*. AMS, New York, 1950.

[19] M. Rosenkranz and G. Regensburger. Integro-differential polynomials and operators. In *ISSAC'08*, pages 261–268, New York, 2008. ACM.

[20] M. Shinbrot. *On the analysis of linear and nonlinear dynamical systems from transient-response data*. NACA, Washington, D.C., 1954.

[21] The Ametista Group. `http://www.lifl.fr/Ametista`, 2013.

[22] R. Ushirobira, W. Perruquetti, M. Mboup, and M. Fliess. Algebraic parameter estimation of a multi-sinusoidal waveform signal from noisy data. In *European Control Conference*, Zurich, April 2013.