CRONE Toolbox for system identification using fractional differentiation models

Rachid Malti, Stephane Victor

Available online at www.sciencedirect.com

ScienceDirect

IFAC-PapersOnLine 48-28 (2015) 769–774

CRONE Toolbox for system identification using fractional differentiation models

Rachid Malti * **Stéphane Victor** *

* *Université de Bordeaux, IMS-UMR 5218 CNRS – 351 cours de la Libération, 33405 Talence cedex, France*
firstname.lastname@ims-bordeaux.fr

Abstract: This paper presents the latest developments for the continuous-time system identification toolbox with fractional models (or fractional order systems): the CRONE toolbox. This toolbox is to be run with Matlab which includes time-domain identification algorithms for estimating continuous-time models directly from sampled data. The originality of the implemented algorithms is that they allow either fixing fractional differentiation orders or estimating them along with transfer function coefficients. One of the main issues when dealing with fractional models is their time-domain simulation. Three different time-domain simulation methods can be used independently from system identification methods. Output Error (OE), state variable filters (SVF) and (optimal) instrumental variables (IV) methods for ARX and OE models are provided to the end-user. The object oriented programming of the toolbox allows overloading standard script names. As a consequence, an end-user familiar with standard Matlab operators and scripts can use straightforwardly the CRONE toolbox.

Keywords: Similarity transformation; Fractional calculus; Subspace method; pseudo-state-space representation; system identification.

1. INTRODUCTION

Fractional models have witnessed a growing interest during the last years. Many diffusive phenomena can be modeled by fractional transfer functions. In electrochemistry for instance, diffusion of charges in acid batteries is governed by Randles models (Sabatier *et al.*, 2006) that involve Warburg impedance with an integrator of order 0.5. Electrochemical diffusion showed to have a tight relation with derivatives of order 0.5 (Oldham and Spanier, 1973). In thermal diffusion of a semi-infinite homogeneous medium, Battaglia *et al.*(Battaglia *et al.*, 2001) have shown that the solution of the heat equation links thermal flux to a half order derivative of the surface temperature on which the flux is applied.

Time-domain system identification using fractional models was initiated in the late nineties. Oustaloup *et al.* (1996) developed a method based on the discretization of the fractional differential equation using Grünwald definition and on the estimation of its coefficients using least squares. Trigeassou *et al.* (Trigeassou *et al.*, 1999) implemented their identification algorithm on the approximation of a fractional integrator by a rational model. Cois *et al.* (2001) proposed several extensions of prediction error methods, such as the state variable filters and the instrumental variable (IV), to fractional system identification. Aoun *et al.* (Aoun *et al.*, 2007) synthesized fractional orthogonal bases generalizing various rational bases (Laguerre, Kautz,...) to fractional differentiation orders for identification issues. Recently, Malti *et al.* (Malti *et al.*, 2008*a*; Victor *et al.*, 2013) have extended the concept of optimal IV methods to fractional systems. Most of system identification methods developed by the CRONE team have been integrated in the CRONE toolbox.

The CRONE Toolbox, developed gradually since the nineties (Oustaloup *et al.*, 2000) is a Matlab and Simulink toolbox dedicated to applications of fractional (or non integer) derivatives in engineering and science. The motivation for developing the CRONE toolbox was first to fill a gap, since no software support was dedicated to fractional calculus and its applications in signal processing and automatic control. During the first decade of the 21st century, the CRONE toolbox was only available for some privileged researchers and industrials who granted its development. Since 2012, the CRONE toolbox has evolved towards an object oriented programming (Malti *et al.*, 2011; Malti *et al.*, 2012) which allowed many enhancements such as overloading operators and scripts. Moreover, it has become freely available for the international scientific community for research and pedagogical purposes and can be downloaded at: `http://cronetoolbox.ims-bordeaux.fr`.

This paper intends to presenting system identification methods developed in the CRONE Toolbox. To that end, it is necessary to expose, in section 2, the main classes of the CRONE toolbox with the Unified Modeling Language (UML) diagram. One of the main difficulties, when dealing with fractional systems is the time-domain simulation methods which are presented in section 3. The main topic of the paper is then addressed in section 4. Next, a user's guide is provide as a tutorial in section 5 with simulation and experimental data. The latter data are issued from a thermal rod heated at one end by a resistor. Finally, conclusions are drawn and some further prospects given.

Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2015.12.223

2. AN OVERVIEW OF THE OBJECT ORIENTED CRONE TOOLBOX

Three main representations of fractional systems may be defined: fractional transfer functions, fractional pole-zero-gain, and fractional state-space.

A SISO fractional system is governed by a fractional differential equation:

$$y(t) + a_1\mathcal{D}^{\alpha_1}y(t) + \cdots + a_{m_A}\mathcal{D}^{\alpha_{m_A}}y(t) = b_0\mathcal{D}^{\beta_0}u(t) + b_1\mathcal{D}^{\beta_1}u(t) + \cdots + b_{m_B}\mathcal{D}^{\beta_{m_B}}u(t) \quad (1)$$

where $\mathcal{D} = \frac{d}{dt}$ is the differential operator, $(a_j, b_i) \in \mathbb{R}^2$, and the differentiation orders $\alpha_1 < \alpha_2 < \ldots < \alpha_{m_A}, \beta_0 < \beta_1 < \ldots < \beta_{m_B}$ are allowed to be non-integer positive numbers.

2.1 Fractional transfer function: frac_tf-class

Laplace transform property of a fractional derivative (Podlubny, 1999) allows to write the fractional differential equation (1) in a transfer function form:

$$H(s) = \frac{\sum_{i=0}^{M} b_i s^{\beta_i}}{1 + \sum_{j=1}^{N} a_j s^{\alpha_j}}. \quad (2)$$

It is constituted of a ratio of two explicit polynomials:

$$p(s) = \sum_{i=0}^{L} c_i s^{\gamma_i}, \quad (3)$$

the word explicit referring here to the positive real powers of s, $\gamma_i \in \mathbb{R}_{>0}$. Consequently, two linked sequences $[c_0, c_1, \ldots, c_L]$ and $[\gamma_0, \gamma_1, \ldots, \gamma_L]$ are necessary to define a fractional explicit polynomial.

Moreover, a transfer function $H(s)$ is commensurate of order ν if and only if it can be written as $H(s) = S(s^\nu)$, where $S = \frac{T}{R}$ is a rational function with T and R, two co-prime polynomials. Hence, all differentiation orders are multiples of the commensurate order ν, allowing to obtain a rational transfer function in s^ν. Taking as an example $H(s)$ of (2), assuming that $H(s)$ is commensurate of order ν, one can write:

$$H(s) = \frac{\sum_{i=0}^{m} \tilde{b}_i s^{i\nu}}{1 + \sum_{j=1}^{n} \tilde{a}_j s^{j\nu}}, \quad (4)$$

where $m = \frac{\beta_M}{\nu}$ and $n = \frac{\alpha_N}{\nu}$ are integers and: $\forall i' \in \{0, 1, \ldots, m\}, \forall j' \in \{1, \ldots, n\}$:

$$\begin{cases} \tilde{b_{i'}} = b_i \text{ if } \exists i \in \{0, 1, \ldots, M\} \text{such that } i'\nu = \beta_i \\ \tilde{b_{i'}} = 0 \text{ otherwise} \\ \tilde{a_{j'}} = a_j \text{ if } \exists j \in \{1, \ldots, N\} \text{such that } j'\nu = \alpha_j \\ \tilde{a_{j'}} = 0 \text{ otherwise} \end{cases} \quad (5)$$

All powers of s^ν in (4) are integers. In rational transfer functions ν equals 1.

Both the incommensurate (2) and the commensurate transfer function (4) are implemented in the toolbox in a class named frac_tf (see also the UML diagram presented in section section 2.4 and Fig.1).

2.2 Zero-pole-gain factorized transfer function: frac_zpk-class

The commensurate transfer function (4) can always be written in a zero-pole-gain (zpk) factorized form

$$H(s) = K\frac{\prod_{i=0}^{m}(s^\nu + z_i)}{\prod_{j=0}^{n}(s^\nu + p_j)}, \quad (6)$$

where $K \in \mathbb{R}$ is a multiplicative factor, $z_i \in \mathbb{C}, \forall i$, are the s^ν-zeros and $p_j \in \mathbb{C}, \forall j$, are the s^ν-poles. A fractional transfer function in a developed form is converted to a zpk form by finding first the commensurate order and then all s^ν-poles and s^ν-zeros of (4).

The fractional zpk representation (6) is implemented as frac_zpk-class in the CRONE toolbox (see also the UML diagram presented in Fig.1).

2.3 Fractional state-space representation: frac_ss-class

The fractional (also called pseudo- or even partial) state space representation is given by:

$$\mathbf{D}^\nu \mathbf{x}(t) = A\mathbf{x}(t) + B\mathbf{u}(t), \quad (7)$$

$$\mathbf{y}(t) = C\mathbf{x}(t) + D\mathbf{u}(t), \quad (8)$$

where $\nu \in \mathbb{R}_+$ is the commensurate order, $\mathbf{x} \in \mathbb{R}^n$ is the pseudo-state vector, $\mathbf{u} \in \mathbb{R}^m$ the input vector, $\mathbf{y} \in \mathbb{R}^p$ the output vector, $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, $D \in \mathbb{R}^{p\times m}$ are constant matrices. Zero initial conditions are considered: $\mathbf{x}(t) = 0$ for $t \leq 0$.

Matignon (Matignon, 1998) proved that the fractional system (7)-(8) (or similarly (4)) is stable if and only if:

$$0 < \nu < 2 \quad \text{and} \quad |\arg(\lambda_k)| > \nu\frac{\pi}{2} \quad \forall k = 1, \ldots, n$$

where λ_k is the k^{th}-eigenvalue of A (or similarly s^ν-pole of (4)) and $-\pi < \arg(\lambda_k) \leq \pi$.

Conversion from fractional transfer function to fractional state space representation is easily achieved using either the commensurate fractional transfer function (4) or the zpk representation (6). Control toolbox script tf2ss is then applied by passing as parameters numerator and denominator coefficients $\tilde{b_i}$ and $\tilde{a_j}$ in descending powers of s^ν. The returned matrices A, B, C, and D, are aggregated to the differentiation order ν to form the fractional state space representation. As in rational systems, it is easy to generate similarity transformations which convert the obtained results to other forms. The fractional transfer function is linked to the fractional state space representation by:

$$H(s) = C(s^\nu I - A)^{-1}B + D. \quad (9)$$

The fractional state space representation (7) and (8) are implemented as frac_ss-class in the CRONE toolbox (see also the UML diagram presented in Fig.1).

2.4 Class diagram of the CRONE Toolbox and attributes associated to each class

Class diagram of the CRONE Toolbox is illustrated in Fig.1. There is a frac_poly_exp-class for explicit fractional

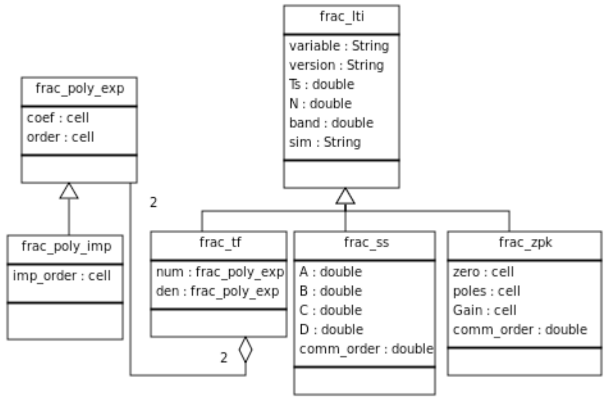

Fig. 1. UML diagram of the object oriented CRONE Toolbox

polynomials which has a frac_poly_imp child class for implicit fractional polynomials. On the other hand, for dealing with fractional systems, frac_lti-class is created with three children classes frac_tf for transfer function representation, frac_zpk for zpk representation, and finally frac_ss for state space representation.

3. TIME-DOMAIN SIMULATION OF FRACTIONAL SYSTEMS

One of the main difficulties with fractional models is related to time-domain simulation. This issue has been extensively studied and an overview of the major methods can be found in (Aoun *et al.*, 2004). Three time-domain simulation methods are implemented in the CRONE toolbox: the first one is based on the Grünwald definition of fractional derivatives and the other ones are based on Oustaloup's approximation of fractional operators in frequency domain.

3.1 Grünwald derivative

The concept of differentiation to an arbitrary order (non-integer) γ, with $\gamma \in \mathbb{R}$ was defined by Grünwald-Letnikov (see e.g. (Podlubny, 1999, chapter 2)), which results from the generalization of integer order derivatives:

$$\mathcal{D}^{\gamma} x(t) = \lim_{h \to 0} \frac{1}{h^{\gamma}} \sum_{k=0}^{\lfloor \frac{t}{h} \rfloor} (-1)^k \binom{\gamma}{k} x(t-kh), \qquad (10)$$

where $\lfloor . \rfloor$ stands for the floor operator, and $\binom{\gamma}{k}$ stands for the Newton binomial coefficient applied to real numbers:

$$\binom{\gamma}{k} = \frac{\Gamma(\gamma+1)}{\Gamma(k+1)\Gamma(\gamma-k+1)} = \frac{\gamma(\gamma-1)...(\gamma-k+1)}{k!}.$$

For numerical evaluation of the fractional derivative, the parameter h in (10) is replaced by the sampling period, thus relieving the limit:

$$\mathcal{D}^{\gamma} x(t) = \frac{1}{h^{\gamma}} \sum_{k=0}^{\lfloor \frac{t}{h} \rfloor} (-1)^k \binom{\gamma}{k} x(t-kh) + O(h). \qquad (11)$$

In doing so, the error terms are proportional to the sampling period (Podlubny, 1999, section 7.4). Consequently, the sampling period should be small enough so that the approximation error may be negligible.

3.2 Oustaloup's approximation

(Oustaloup, 1995) has proposed to synthesize fractional operators in frequency domain using a recursive distribution of zeros and poles. Hence, the frequency behavior of $s^{-\alpha}$ in the frequency range $[\omega_A, \omega_B]$ is approximated by:

$$s^{-\alpha}_{[\omega_A,\omega_B]} = G_\alpha \prod_{k=1}^{N_c} \frac{1+s/\omega'_k}{1+s/\omega_k} \qquad (12)$$

where:

- N_c is the number of cells (directly related to the quality of the approximation),
- G_α is fixed so that $s^{-\alpha}$ has the same gain as $s^{-\alpha}_{[\omega_A,\omega_B]}$ in the middle of the interval $[\omega_A, \omega_B]$,
- ω'_k and ω_k are respectively zeros and poles recursively distributed in the frequency range $[\omega_b, \omega_h] = [\sigma^{-1}\omega_A, \sigma\omega_B]$ where σ is generally set to 10 to minimize border effects (Aoun *et al.*, 2004). They are defined by the following relations:

$$\omega'_k = \gamma\omega_k, \quad \omega_{k+1} = \eta\omega'_k, \quad \alpha = 1 - \frac{\log\gamma}{\log\gamma\eta}.$$

3.3 Trigeassou's variant of Oustaloup's approximation

(Trigeassou *et al.*, 1999) proposed another variant of Oustaloup's approximation which consists of using an integrator outside the frequency band of interest (instead of a simple gain in Oustaloup's approximation) and Oustaloup's approximation inside that frequency band.

$$^{\mathrm{T}}s^{-\alpha}_{[\omega_A,\omega_B]} = s^{-1} s^{1-\alpha}_{[\omega_A,\omega_B]} \qquad (13)$$

where $s^{1-\alpha}_{[\omega_A,\omega_B]}$ is Oustaloup's recursive approximation of the differentiator $s^{1-\alpha}$.

4. SYSTEM IDENTIFICATION METHODS WITH FRACTIONAL MODELS IMPLEMENTED IN THE TOOLBOX

The CRONE toolbox includes system identification routines for fractional order models written in a transfer function form (frac_tf-class).

The toolbox supports transfer function models of the following forms:

$$y(t_k) = \frac{B(\mathcal{D})}{A(\mathcal{D})} u(t_k) + \frac{1}{F(\mathcal{D})} e(t_k), \qquad (14)$$

where $\frac{B(\mathcal{D})}{A(\mathcal{D})}$ is the fractional operator (2) or (4) obtained by replacing the Laplace variable s by the differential operator $\mathcal{D}$, $u(t_k)$ and $y(t_k)$ are respectively the deterministic input and noisy output at time instant t_k, $e(t_k)$ is a zero-mean white Gaussian sequence and $F(\mathcal{D})$ either equals $A(\mathcal{D})$ for an ARX model or $F(\mathcal{D}) = 1$ for an OE model. No initial conditions can be taken into account for the time being.

When dealing with system identification with incommensurate fractional models, the number of parameters to estimate might double up as compared to rational models of the same dimension since each coefficient b_i or a_j is associated to a differentiation order β_i or α_j which can be estimated as well. Sometimes prior knowledge allows setting differentiation orders and only coefficients are estimated which limits *de facto* the number of parameters.

Sys. Id.	OE	ARX
oe	✓	
lssvf		✓
ivsvf		✓
srivcf	✓	
oosrivcf	✓	

Table 1. Available methods for system identification using fractional models

Sys. Id. methods	Coefficients estimation	Commensurate order estimation	All order estimation
oe	✓	✓	✓
lssvf	✓		
ivsvf	✓		
srivcf	✓		
oosrivcf	✓	✓	✓

Table 2. Coefficient and/or order estimation.

Another way for downsizing the number of parameters is to estimate all the coefficients and a single differentiation order: the commensurate order ν in (4). In case where all the differentiation orders are estimated together with the coefficients, it was recommended in (Victor *et al.*, 2013) to first estimate the commensurate order and then to release the commensurability constraint in order to affine the differentiation order estimation since the commensurate model generally constitutes a good initial hit.

Various estimation methods for identifying fractional OE and ARX models are implemented in the CRONE Toolbox. Some standard methods such as the least squares combined with state variable filtering (LSSVF) and suboptimal and optimal instrumental variables (IV) have been extended to deal with fractional models (Malti *et al.*, 2008*b*; Victor *et al.*, 2013).

Models addressed by different algorithms are listed in Table 1. Some of the methods allow estimating only coefficients after fixing differentiation orders. Other allow joint estimation of coefficients and differentiation orders (see table 2).

5. USAGE IN THE TOOLBOX

The main objective of this section is to present how to handle the different objects (or classes) and principle features of the object oriented CRONE toolbox and then how to handle system identification scripts. Once the CRONE toolbox is installed, the tutorial is launched with the following command: CRONE_demos and a new window appears as illustrated in Fig. 2. Three main tutorials are proposed as detailed below.

5.1 Fractional system representation

For new users, an overview of the object oriented CRONE toolbox is proposed for fractional system representation. Also, all classes described in Section 2 and illustrated in Fig.1 are presented in detail for a better handling of these objects. For dealing with fractional systems, the frac_lti-class is created with three children classes frac_tf for transfer function representation, frac_zpk for zpk representation, and finally frac_ss for state space representation. Each class is illustrated with Matlab code examples.

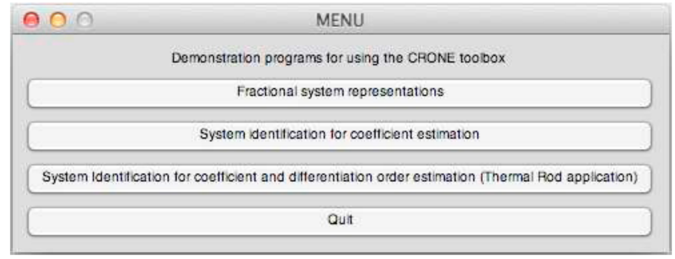

Fig. 2. Tutorials proposed when executing the command CRONE_demos .

Moreover, the time-domain simulation of fractional systems is also presented with the possibility to choose one of the three simulation methods detailed in section 3.

5.2 System identification functions : coefficient estimation

In this tutorial, a case study is proposed for parameter estimation of a fractional model with fixed differentiation orders. Henceforth, a fractional transfer function is first created with a frac_tf object:

$$sys_{id}(s) = \frac{0.5}{0.5s^3 + 1.5s^{1.5} + 1}.$$

After generating an input signal, the output is obtained using the lsim functions of the CRONE toolbox, which allows choosing one of the simulation algorithms described above. An additive white noise with SNR$= 7dB$ is applied to the output. The usage of other overloaded functions such as step, bode, nichols is illustrated.

Then, different system identification algorithms are executed for estimating coefficients. For that purpose, an initial model is first defined:

$$sys_{init}(s) = \frac{b_0}{a_2 s^{2\nu} + a_1 s^{\nu} + 1},$$

and its commensurate order is set to the true one ($\nu = 1.5$). The initial model allows also fixing the number of parameters (or coefficients) to be identified.

Then prediction error methods (LSSVF, IVSVF, SRIVCF and OE) are used in details so that the input arguments may be well defined for coefficient estimation. The estimated parameters are presented in Table 3. As it can be seen, the SRIVCF and OE methods present good results for coefficient estimation in presence of a significant noise. These methods are known to be unbiased in the presence of an output noise.

One very important aspect of fractional differential equation modeling is the determination of the differentiation orders. In system identification with rational models, where only the coefficients are estimated, the model order remains unchanged. When differentiation orders are unknown, as it is often the case in practice, it is helpful to consider order estimation along with transfer function coefficient estimation which is the topic of the following subsection.

Sys. Id.	b_0	a_1	a_2
true	**0.5**	**1.5**	**0.5**
lssvf	0.492	1.519	0.540
ivsvf	0.494	1.514	0.531
srivcf	0.504	1.484	0.499
oe	0.504	1.483	0.505

Table 3. Available methods for system identification using fractional models

```
1  load('ThermalRodData.mat')
2  data = iddata(y,u,Ts);
3  sys_init = frac_tf(1, frac_poly_exp([1 1 1], ...
4      [1.2 0.6 0]), 24, [1e-5 5e1]);
5  sys_oe_coef = oe(sys_init, data,[],'coef');
6  sys_oe_comm = oe(sys_oe_coef, data,[],'comm');
7  sys_oe_all = oe(sys_oe_comm, data,[],'all');
8  figure, subplot(211),plot(t,y,'b',...
9      t,lsim(sys_oe_coef,u,t),'r--',...
10     t,lsim(sys_oe_comm,u,t),'g-.',...
11     t,lsim(sys_oe_all,u,t),'m')
```

Fig. 3. Matlab script showing the use of oe routine of the CRONE toolbox

5.3 Fractional order model identification guidelines with "Thermal Rod" model data

In this last subsection, a thermal application is considered in order to illustrate the OE routine and the OOSRIVCF routine where all system parameters (coefficients and differentiation orders) are unknown (Malti *et al.*, 2008*b*; Victor *et al.*, 2013). The input signal is a thermal flux generated by a resistor glued at one end and the output signal is the temperature of the rod measured at a distance x = 0.5 cm from the heated end. To ensure a unidirectional heat transfer, the entire surface of the rod is insulated. The system is driven to a steady-state temperature by injecting a constant flux density of $\phi = 5$ kW m^2 for a sufficiently long period. Then a prbs is applied with a flux variation of ± 5 kW m^2 around the constant flux of 5 kW m^2. The sampling period is set to 0.5 s. The system is driven to a steady-state temperature by injecting a constant flux density for a sufficiently long period. A delay of 4 samples (2 s) is observed between the output and the input. The input density flux and the output temperature are pretreated to eliminate the constant parts and the delay. The pretreated signals are saved in the file "ThermalRodData.mat". In (Malti *et al.*, 2009), a theoretical model of the aluminum rod, obtained from heat equation under some simplifying assumptions, was compared to experimental data. Among the simplifying assumptions, the rod was assumed to be perfectly isolated. The theoretical model was found to be commensurate of order 0.5. Since the aluminum rod reaches a steady state temperature, it is not perfectly insulated. Consequently, there is no reason to have an exact commensurate order of 0.5 nor to have an integrator as in the physical model.

The oe *routine:* Fig.3 shows a usage example of the overloaded oe routine of the CRONE toolbox. For that purpose a sys_init object of the frac_tf-class is first created in lines 3 and 4. It allows initializing the oe routine by the following arbitrarily chosen transfer function

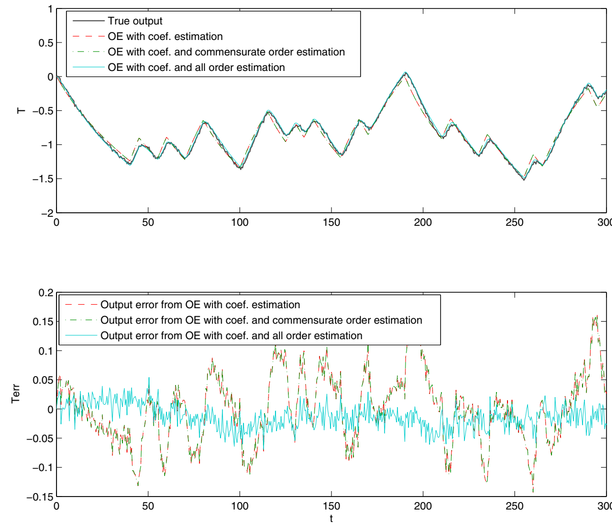

Fig. 4. Comparison among the three outputs of the oe routine. Similar results are obtained with oosrivcf routine

$$H(s) = \frac{1}{s^{1.2} + 2s^{0.6} + 1}. \tag{15}$$

The frac_poly_exp constructor creates the frac_poly_exp object and passes it as an argument to the constructor frac_tf. The additional three parameters allow choosing Oustaloup's simulation method by approximating each fractional differentiator in the frequency band $[\omega_A, \omega_B] = [10^{-5}, 50]$ by the transfer function (12) with $N_c = 24$ cells.

When calling the oe routine, the first argument must be a frac_lti object, otherwise the oe routine of *System Identification Toolbox* is executed. The second argument is an *iddata object* of the System Identification Toolbox. The oe routine calls lsqnonlin routine of *Optimisation toolbox* which uses Levenberg-Marquardt optimization algorithm. The third argument allows to set optimization options used by the latter function. Finally, the last argument allows choosing one of the four options offered. In line 5, only the coefficients are estimated, the differentiation orders remain unchanged. In line 6, the coefficients and the commensurate order are estimated. In line 7, all the parameters are estimated; here the commensurability constraint is removed and all the coefficients are further adjusted. A comparison between all three outputs is done in the following lines. It is further plotted in Fig.4.

The oosrivcf *routine:* Fig.5 shows a usage example of the oosrivcf routine of the CRONE toolbox where a sys_init object of the frac_tf-class is first created just as in the previous subsection allowing the initialization with the same transfer function (15).

When calling the oosrivcf routine of the CRONE toolbox, the first argument must be a frac_lti object. The second argument is an *iddata object* of the System Identification Toolbox. The third argument allows choosing one of the three options offered, just as in the oe routine. In line 5, only the coefficients are estimated, the differentiation orders remain unchanged; also estimating only the coeffi-

```
1  load('ThermalRodData.mat')
2  data = iddata(y,u,Ts);
3  sys_init = frac_tf(1, frac_poly_exp([1 1 1], ...
4      [1.2 0.6 0]), 24, [1e-5 5e1]);
5  sys_srivcf_coef = oosrivcf(sys_init,... `
6      data,'coef', [20, 1e-4]);
7  sys_oosrivcf_com = oosrivcf(sys_srivcf_coef,...
8      data,'comm', [200, 1e-4]);
9  sys_oosrivcf_all = oosrivcf(sys_oosrivcf_com,...
10     data,'all', [200, 1e-4]);
```

Fig. 5. Matlab script showing the use of oosrivcf routine of the CRONE toolbox

cients comes down to using the srivcf function. In line 7, the coefficients and the commensurate order are estimated. In line 9, all the parameters are estimated; here the commensurability constraint is removed and all the coefficients are further adjusted. The last (optional) input argument $[iter, \epsilon]$ gives indications for the oosrivcf routine for the number of iterations to be used and the precision desired for the parameters (default is $iter = 20$ and $\epsilon = 1e-3$). A comparison between all three outputs gives results similar to the oe routine in Fig.4.

6. CONCLUSION

The new developments outlined in this paper extend the practical application field of the CRONE toolbox. We plan in the future to concentrate on the following developments:

- implementation of several instrumental variable based methods for identifying Box-Jenkins models, and closed-loop system identification;
- implementation of a Graphical User Interface. We are working on the development of an easy-to-use interface for system identification using fractional models.

REFERENCES

Aoun, M., R. Malti, F. Levron and A. Oustaloup (2004). Numerical simulations of fractional systems: an overview of existing methods and improvements. *Nonlinear Dynamics* **38**(1-4), 117–131.

Aoun, M., R. Malti, F. Levron and A. Oustaloup (2007). Synthesis of fractional Laguerre basis for system approximation. *Automatica* **43**(9), 1640–1648.

Battaglia, J.-L., O. Cois, L. Puigsegur and A. Oustaloup (2001). Solving an inverse heat conduction problem using a non-integer identified model. *Int. J. of Heat and Mass Transfer* **44**(14), 2671–2680.

Cois, O., A. Oustaloup, T. Poinot and J.-L. Battaglia (2001). Fractional state variable filter for system identification by fractional model. In: *6th IEEE ECC*. Porto, Portugal.

Malti, R., J. Sabatier and H. Akçay (2009). Thermal modeling and identification of an aluminium rod using fractional calculus. In: *15th IFAC SYmposium on System IDentification (SYSID)*. Saint-Malo France. pp. 958–963.

Malti, R., P. Melchior, P. Lanusse and A. Oustaloup (2011). Towards an object oriented CRONE toolbox for fractional differential systems. In: *18th IFAC World Congress*. Milan, Italy. pp. 10830–10835.

Malti, R., P. Melchior, P. Lanusse and A. Oustaloup (2012). Object oriented CRONE toolbox for fractional differential signal processing. *Signal Image and Video Processing* **6**(3), 393–400.

Malti, R., S. Victor, A. Oustaloup and H. Garnier (2008*a*). An optimal instrumental variable method for continuous-time fractional model identification. In: *17th IFAC World Congress*. Seoul South Korea. pp. 14379–14384.

Malti, R., S. Victor and A. Oustaloup (2008*b*). Advances in system identification using fractional models. *Journal of Computational and Nonlinear Dynamics* **3**(2), 021401,1–7.

Matignon, D. (1998). Stability properties for generalized fractional differential systems. *ESAIM proceedings - Systèmes Différentiels Fractionnaires - Modèles, Méthodes et Applications*.

Oldham, K.B. and J. Spanier (1973). Diffusive transport to planar, cylindrical and spherical electrodes. *Electroanaly. Chem. Interfacial Electrochem.* **41**, 351–358.

Oustaloup, A. (1995). *La dérivation non-entière: théorie, synthèse et applications*. Hermès - Paris.

Oustaloup, A., L. Le Lay and B. Mathieu (1996). Identification of non integer order system in the time-domain. In: *IEEE-CESA'96, SMC IMACS Multiconference*. SMC IMACS Multiconference - Computational Engineering in Systems Application - Symposium on Control, Optimisation and Supervision.

Oustaloup, A., P. Melchior, P. Lanusse, O. Cois and F. Dancla (2000). The CRONE toolbox for Matlab. In: *11th IEEE Inrternational Symposium on Computer-Aided Control System Design, CACSD*. Anchorage, Alaska, USA.

Podlubny, I. (1999). *Fractional Differential Equations*. Academic Press. San Diego.

Sabatier, J., M. Aoun, A. Oustaloup, G. Grégoire, F. Ragot and P. Roy (2006). Fractional system identification for lead acid battery sate charge estimation. *Signal Processing* **86**(10), 2645–2657.

Trigeassou, J.-C., T. Poinot, J. Lin, A. Oustaloup and F. Levron (1999). Modeling and identification of a non integer order system. In: *ECC*. Karlsruhe, Germany.

Victor, S., R. Malti, H. Garnier and A. Oustaloup (2013). Parameter and differentiation order estimation in fractional models. *Automatica*.