UNIVERSITÉ DE BORDEAUX

ims UMR 5218 CNRS

CRONE®

Toolbox

Fractional Systems Toolbox

Brief Presentation of the Object Oriented CRONE toolbox

CRONE research group

Version Beta 1
October 15th, 2010

Contact: cronetoolbox@ims-bordeaux.fr
URL: http://cronetoolbox.ims-bordeaux.fr

UNIVERSITÉ BORDEAUX 1 Sciences Technologies

Université de Bordeaux
Université de Bordeaux 1 - IPB
IMS/LAPS - UMR 5218 CNRS
351 cours de la Libération - F33405 TALENCE Cedex - France
URL: http://www.ims-bordeaux.fr

IPB ENSEIRB MATMECA BORDEAUX

CRONE®

Toolbox

Fractional Systems Toolbox

CREDITS

The Object Oriented CRONE Toolbox is managed by Rachid MALTI and Pierre MELCHIOR.

The classical CRONE Toolbox is managed by Pierre MELCHIOR and Patrick LANUSSE.

The CRONE toolbox is the result of a continuous research work of permanent members, PhD students, and research engineers of the CRONE team.

The CRONE team.

Contact: cronetoolbox@ims-bordeaux.fr
URL: http://cronetoolbox.ims-bordeaux.fr

A brief presentation of the Object Oriented CRONE toolbox

Rachid Malti, Pierre Melchior, Patrick Lanusse, and Alain Oustaloup

The CRONE group
Contact: cronetoolbox@ims-bordeaux.fr
URL: http://cronetoolbox.ims-bordeaux.fr

Abstract

The CRONE Toolbox, developed since the nineties by the CRONE team, is a Matlab Toolbox dedicated to fractional calculus. It is currently evolving towards an object oriented version, which allows many enhancements. Three main user classes, dedicated to fractional systems representations namely fractional transfer functions (frac_tf), fractional zeros poles and gain (frac_zpk), and fractional state-space (frac_ss), are developed. All three user classes are children of a parent class (frac_lti) which contains some common attributes of fractional systems. Among enhancements of the object programming of the CRONE toolbox, is the overloading of basic operators ($+, -, \times, /, .\times, ...$) and standard Matlab scripts (lsim, bode, nichols, ...) for the new classes. As a consequence, an end user familiar with standard Matlab operators and scripts can use straightforwardly the CRONE toolbox. The main objective of this paper is to present class diagrams and principle features of the object oriented CRONE toolbox, which can be downloaded at http://cronetoolbox.ims-bordeaux.fr.

Key words: Fractional calculus, Object oriented programming, Simulation, MIMO system, Matlab toolbox, CRONE toolbox.

1 Introduction and mathematical background

The CRONE Toolbox, developed gradually since the nineties ((Oustaloup *et al.*, 2000)) by the CRONE research group, and granted by CNRS and PSA car company, is a Matlab and Simulink toolbox dedicated to applications of fractional (or non integer) derivatives in engineering and science. The toolbox is not only intended to researchers, including Ph.D students, but also to industrials whose growing interest in fractional systems and whose readiness to invest in the development of its applications are manifest. Original theoretical mathematical concepts developed in the group were integrated in the first version of the CRONE Toolbox ((Oustaloup *et al.*, 2000)) as four main modules each of which deals with a specific application domain of fractional derivatives: “Mathematical module” contains several algorithms of fractional calculus; “System identification module” contains frequency and time-domain system identification algorithms; in “CRONE control module” fractional derivatives are used as high-level design parameters in order to simplify the design of robust control-systems; “Path planning module” contains algorithms to optimize paths in dynamic environments.

During the last ten years, the CRONE toolbox was only available for some privileged researchers and industrials who granted its development. On the other hand, the last two decades have witnessed a growing interest in fractional derivatives and their applications (Ortigueira and Machado, 2006; Podlubny, 1999; Monje *et al.*, 2008; Xue and Chen, 2005). From now on, the CRONE toolbox is freely available for the international scientific community for research and pedagogical purposes. The aim of the CRONE group is to share its developments and its knowledge with scientists, researchers, and engineers world wide.

* The CRONE toolbox can be downloaded at http://cronetoolbox.
ims-bordeaux.fr. The website is developed and maintained by Michel Alexeline, IMS webmaster, gratefully acknowledged.

Fractional mathematical models are based on fractional differential equations:

$$\begin{aligned} y(t) + a_1\mathbf{D}^{\alpha_1} y(t) + \cdots + a_N \mathbf{D}^{\alpha_N} y(t) = \\ b_0 \mathbf{D}^{\beta_0} u(t) + b_1 \mathbf{D}^{\beta_1} u(t) + \cdots + b_M \mathbf{D}^{\beta_M} u(t) \end{aligned} \tag{1}$$

where differentiation orders $0 < \alpha_1 < \alpha_2 < \ldots < \alpha_N$ and $0 \leq \beta_0 < \beta_1 \ldots < \beta_M$, are allowed to be non-integer positive numbers and where coefficients $a_1, \ldots, a_N, b_0, \ldots, b_M$ are real numbers. Differentiation to an arbitrary order, $\mathbf{D}^{\gamma}$, is defined by Riemann and Liouville as being integer derivative of order $m = \lfloor \nu \rfloor + 1$ ($\lfloor \nu \rfloor$ stands for the integer part of ν) of a non-integer integral of order $1 - (m - \nu)$ ((Samko *et al.*, 1993)):

$$\mathbf{D}^{\nu} x(t) = \frac{1}{\Gamma(m-\nu)} \left(\frac{d}{dt}\right)^m \left(\int_0^t \frac{x(\tau)\, d\tau}{(t-\tau)^{1-(m-\nu)}} \right) \tag{2}$$

where $t > 0$, $\nu > 0$.

A discrete-time definition of fractional derivative was proposed by (Grünwald, 1867), $\forall \nu \in \mathbb{R}_+^*$:

$$\mathbf{D}^{\nu} x(t) = \lim_{h \to 0} \frac{1}{h^{\nu}} \sum_{k=0}^{\infty} (-1)^k \binom{\nu}{k} x(t - kh) \tag{3}$$

A more concise algebraic tool can be used to represent fractional derivatives: the Laplace transform. The Laplace transform of a ν order derivative ($\nu \in \mathbb{R}_+$) of $x(t)$ relaxed at $t = 0$ (i.e. all derivatives of $x(t)$ equal zero for all $t < 0$) is given by (Oldham and Spanier, 1974):

$$\mathscr{L}\left\{\mathbf{D}^{\nu} x(t)\right\} = s^{\nu} X(s) \tag{4}$$

The paper is organized as follows. In the next section, fractional explicit and implicit polynomials are defined. Additionally, fractional system representation is treated in three different ways, all of them implemented in the CRONE toolbox. In Section 3, one of the key points in the CRONE toolbox: time-domain simulation of fractional systems, is presented. All three implemented methods are detailed in this section. Section 4 is in the heart of the object oriented CRONE toolbox, as the class diagrams and attributes of each class are detailed. Functions and overloaded methods and operators are then presented in section 5. Finally, an example, using objects methods and operators is given in section 6.

2 Fractional system representations

Three main representations of fractional systems may be defined: fractional transfer functions, fractional pole-zero-gain, and fractional state-space. Fractional transfer function representation requires defining fractional explicit polynomials. Further, in the CRONE toolbox, fractional implicit polynomials are defined.

2.1 Fractional explicit polynomials

By analogy to classical polynomials, fractional explicit polynomials are defined as:

$$p(s) = \sum_{i=0}^{L} c_i s^{\gamma_i}, \tag{5}$$

the word explicit referring here to the positive or negative real powers of s, $\gamma_i \in \mathbb{R}$. 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.

Algorithm 1. Commensurate order computation with a tolerance of tol_ord (default value 10^{-5})

```
1: CommOrder ← round($\beta_M$/tol_ord)
2: for all D in $\{\beta_{M-1}, \dots \beta_0, \alpha_N, \dots \alpha_1\}$ do
3:     CommOrder ← gcd(CommOrder, D/tol_ord)
4: end for
5: CommOrder ← CommOrder*tol_ord
```

2.2 Fractional implicit polynomials

Further, fractional polynomials (5) can be raised to a non integer power, yielding implicit fractional polynomials:

$$\tilde{p}(s) = p(s)^{\beta} = \left(\sum_{i=0}^{L} c_i s^{\gamma_i} \right)^{\beta}, \tag{6}$$

allowing to represent Havriliak-Negami transfer functions ((Sommacal *et al.*, 2008)). The word implicit referring to the power $\beta \in \mathbb{R}$ of the fractional explicit polynomial.

2.3 Fractional transfer function in a developed form

Property (4) allows to write the fractional differential equation (1) in a transfer function form, as a ratio of two explicit polynomials:

$$H(s) = \frac{\sum\limits_{i=0}^{M} b_i s^{\beta_i}}{1 + \sum\limits_{j=1}^{N} a_j s^{\alpha_j}} \tag{7}$$

where $\alpha_1 < \alpha_2 < \dots < \alpha_N$, $\beta_0 < \beta_1 \dots < \beta_M$ and $(a_i, b_j) \in \mathbb{R}^2, \forall i = 0, 1, \dots, M, \forall j = 1, 2, \dots, N$. 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 (7), assuming that $H(s)$ is commensurate of order ν, one can write:

$$H(s) = \frac{\sum\limits_{i=0}^{m} \tilde{b}_i s^{i\nu}}{1 + \sum\limits_{j=1}^{n} \tilde{a}_j s^{j\nu}} \tag{8}$$

where $m = \frac{\beta_M}{\nu}$ and $n = \frac{\alpha_N}{\nu}$ are integers and:

$$\begin{cases} \tilde{b}_i = b_i \quad \text{if} \quad i\nu = \beta_i \quad \text{and} \quad \tilde{b}_i = 0 \quad \text{if} \quad i\nu \neq \beta_i \\ \tilde{a}_j = a_j \quad \text{if} \quad j\nu = \alpha_j \quad \text{and} \quad \tilde{a}_j = 0 \quad \text{if} \quad j\nu \neq \alpha_j \end{cases} \tag{9}$$

All powers of s^{ν} in (8) are integers. In rational transfer functions ν equals 1. A sufficient condition for $H(s)$ to be commensurate is that all differentiation orders belong to the set of rational numbers $\mathbb{Q}$. Consequently, computer numbering formats allow only coding commensurate fractional transfer functions. The commensurate order is computed in the CRONE toolbox by using a precision factor (tol_ord), set to a default value of 10^{-5} i.e. five digits precision, in algorithm 1. Once the commensurate order found, equation (9) is applied to find all the coefficients $\tilde{b}_i, \tilde{a}_j$ of the commensurate transfer function (8).

2.4 Zero-pole-gain (zpk) factorized transfer function

The commensurable transfer function (8) can always be written in a factorized form

$$H(s) = K\frac{\prod_{i=0}^{m}(s^\nu + z_i)}{\prod_{j=0}^{n}(s^\nu + p_j)}, \tag{10}$$

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 (8).

2.5 Fractional state-space representation

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

$$\mathbf{D}^\nu \mathbf{x}(t) = A\mathbf{x}(t) + B\mathbf{u}(t), \tag{11}$$
$$\mathbf{y}(t) = C\mathbf{x}(t) + D\mathbf{u}(t) \tag{12}$$

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 $t \leq 0$.

Conversion from fractional transfer function to fractional state space representation is easily achieved using either the commensurate fractional transfer function (8) or the zpk representation (10). 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:

$$H(s) = C(s^\nu I - A)^{-1}B + D \tag{13}$$

3 Time-domain simulation of fractional systems

There exists mainly two classes of methods for simulating fractional systems: the first class is based on fractional system approximation by a discrete-time rational model; the second one is based on fractional system approximation by a continuous-time rational model. The latter uses continuous-time simulation algorithms, once the continuous-time model obtained. Only simulation algorithms implemented in the CRONE Toolbox are detailed in this section.

3.1 Methods based on discrete-time models

These methods approximate a fractional model by a rational discrete-time one. The fractional differentiator is substituted by its discrete-time equivalent:

$$s^\nu \to \psi(z^{-1}). \tag{14}$$

As a result a discrete-time transfer function is obtained:

$$\mathcal{H}(z^{-1}) = H(\psi(z^{-1})) = \frac{\sum_{i=0}^{M} b_i \psi(z^{-1})^{\beta_i}}{1 + \sum_{j=1}^{N} a_j \psi(z^{-1})^{\alpha_j}} \tag{15}$$

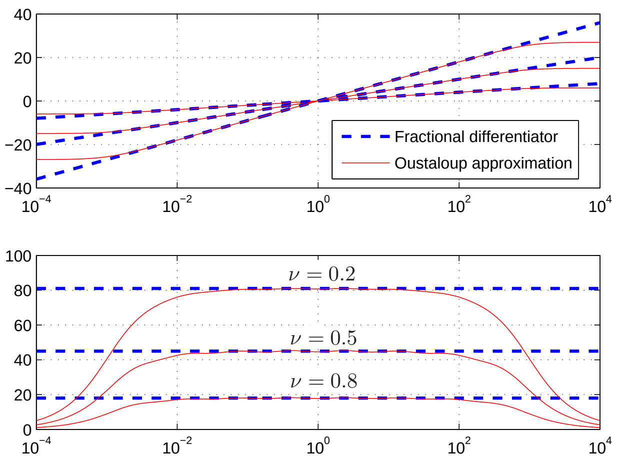

Fig. 1. Approximation using Oustaloup's method in the frequency band $[10^{-2}10^{2}]$

The discretization operator $\psi(z^{-1})$ of analogue circuits can be any of the usual operators such as Euler, Simson, Tustin, El-Alaoui ((Aoun *et al.*, 2004)).Using Euler's operator:

$$\begin{aligned}\psi(z^{-1}) &= \left(\tfrac{1-z^{-1}}{T_s}\right)^{\nu} \\ &= \left(\tfrac{1}{T_s}\right)^{\nu} \sum_{k=0}^{\infty} (-1)^k \tbinom{\nu}{k} z^{-k}\end{aligned} \tag{16}$$

is equivalent to using Grünwald definition (3) of the fractional derivative.

3.2 Methods based on continuous-time models

These methods approximate a fractional model by a rational continuous-time one in a given frequency band. Hence, the fractional operator s^{ν} is approximated by a rational continuous-time transfer function

$$s^{\nu} \rightarrow s^{\lfloor \nu \rfloor} \mathscr{A}^{(\nu - \lfloor \nu \rfloor)} \tag{17}$$

where the approximation of the differentiator $s^{\nu - \lfloor \nu \rfloor}$ with a differentiation order $0 < \nu - \lfloor \nu \rfloor < 1$, namely $\mathscr{A}^{(\nu - \lfloor \nu \rfloor)}$, is discussed below.

Let $s^{\gamma}_{[\omega_A \omega_B]}$ be a γ-order fractional operator in the frequency band $[\omega_A \omega_B]$, with $0 < \gamma < 1$:

$$s^{\gamma}_{[\omega_A \omega_B]} = s^{\gamma} \quad \forall \omega \in [\omega_A \omega_B]. \tag{18}$$

A first approximation of the fractional operator $s^{\gamma}_{[\omega_A \omega_B]}$, with $0 < \gamma < 1$, was proposed by (Oustaloup, 1983):

$$s^{\gamma}_{[\omega_A \omega_B]} \approx \mathscr{A}^{(\gamma)}_{\text{Oust}} = C_{(\gamma)} \left(\frac{1 + \frac{s}{\omega_h}}{1 + \frac{s}{\omega_b}} \right)^{\gamma}, \tag{19}$$

where $\omega_b < \omega_h$ are chosen such that $\omega_b = \sigma^{-1}\omega_A$ and $\omega_h = \sigma \omega_B$ (σ is usually set to 10) and $C_{(\gamma)}$ is chosen to get a unit gain at $\omega = 1$ rad s^{-1} :

$$C_{(\gamma)} = \left| \frac{1 + \mathrm{j}\frac{1}{\omega_h}}{1 + \mathrm{j}\frac{1}{\omega_b}} \right|^{-\gamma} = \left(\frac{\omega_h}{\omega_b} \right)^{\gamma} \left(\frac{1 + \omega_b^2}{1 + \omega_h^2} \right)^{\frac{\gamma}{2}}. \tag{20}$$

Oustaloup's approximation has an asymptotic behavior of a gain in low and high frequencies.

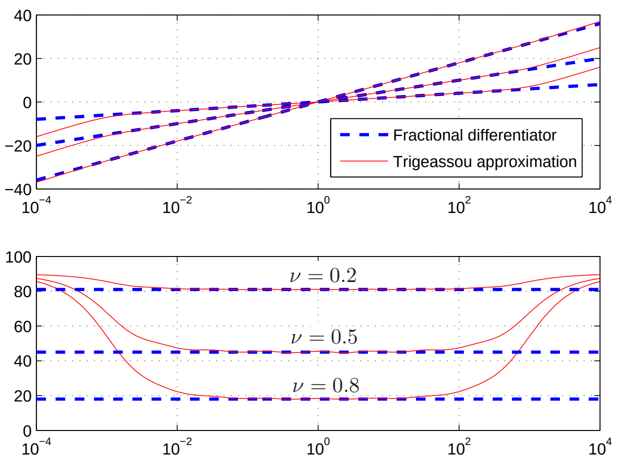

Fig. 2. Approximation using Trigeassou's method in the frequency band $[10^{-2}10^{2}]$

(Trigeassou *et al.*, 1999) propose another approximation of the non integer differentiator $s^{\gamma}_{[\omega_A\omega_B]}$, with $0 < \gamma < 1$:

$$s^{\gamma}_{[\omega_A\omega_B]} \approx \mathscr{A}^{(\gamma)}_{\text{Trig}} = C_{(\gamma-1)}\frac{1}{s}\left(\frac{1+\frac{s}{\omega_h}}{1+\frac{s}{\omega_b}}\right)^{\gamma-1}, \tag{21}$$

where $C_{(\gamma-1)}$ is deduced from (20). This approximation, $\mathscr{A}^{(\gamma)}_{\text{Trig}}$ named after Trigeassou, has an asymptotic behavior of a differentiator of order 1 in low and high frequencies.

(Oustaloup, 1995) approximates the irrational part in (19) (and so do (Trigeassou *et al.*, 1999) in (21)) by a recursive distribution of poles and zeros:

$$\left(\frac{1+\frac{s}{\omega_h}}{1+\frac{s}{\omega_b}}\right)^{\gamma} \approx \prod_{k=1}^{N}\left(\frac{1+\frac{s}{\omega_k}}{1+\frac{s}{\omega'_k}}\right). \tag{22}$$

Recursive distribution of poles and zeros yields a recursive distribution of corner frequencies ω_k and ω'_k according to the following relations:

$$\frac{\omega_{k+1}}{\omega_k} = \frac{\omega'_{k+1}}{\omega'_k} = \alpha\eta > 1, \tag{23}$$

$$\frac{\omega_k}{\omega'_k} = \alpha, \qquad \frac{\omega'_{k+1}}{\omega_k} = \eta, \tag{24}$$

$$\gamma = \frac{\log(\alpha)}{\log(\alpha)+\log(\eta)}. \tag{25}$$

The ratios α and η defined by (24) are known as recursive factors. For a given γ and corner frequencies ω_b and ω_h, recursive factors are set according to N:

$$\alpha = \left(\frac{\omega_h}{\omega_b}\right)^{\frac{\gamma}{N}} \qquad \text{and} \qquad \eta = \left(\frac{\omega_h}{\omega_b}\right)^{\frac{1-\gamma}{N}}. \tag{26}$$

The bigger N the more precise the approximation of the fractional differentiator in the frequency band $[\omega_A, \omega_B]$. As a rule of thumb, two to three corner frequencies (in the numerator and the denominator) are used per decade.

Consequently, Oustaloup's and Trigeassou's approximation methods of differentiation operators require the knowledge of the number of corner frequencies N and the frequency band $[\omega_b, \omega_h]$.

Finally, a rational transfer function, equivalent to the fractional transfer function (7), is obtained by substituting each differentiation operator by it's approximation:

$$\mathscr{H}(s) = \frac{\sum\limits_{i=0}^{M} b_i s^{\lfloor \beta_i \rfloor} \mathscr{A}^{(\beta_i - \lfloor \beta_i \rfloor)}}{1 + \sum\limits_{j=1}^{N} a_j s^{\lfloor \alpha_j \rfloor} \mathscr{A}^{(\alpha_j - \lfloor \alpha_j \rfloor)}} \tag{27}$$

where $\mathscr{A}^{(\gamma)}$ is either of Oustaloup's or Trigeassou's approximation of fractional operators s^γ, with $0 < \gamma < 1$.

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

Class diagram of the CRONE Toolbox is illustrated on Fig.3. There is a frac_poly_exp class for explicit fractional 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.

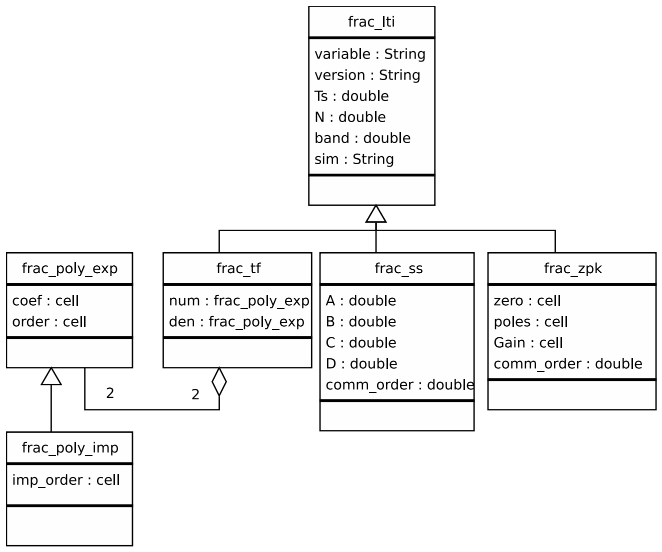

Fig. 3. Class diagram of the OO-CRONE Toolbox

4.1 The frac_poly_exp class

In the CRONE toolbox, explicit fractional polynomials (5) are defined in the frac_poly_exp class as two linked sequences $\forall i = 1, \ldots, L$:

- coef contains the sequence of coefficients c_i,
- order contains the sequence of fractional powers γ_i.

Each coefficient is associated to its corresponding differentiation order. For example, the two linked sequences $(2, 1, 3)$ and $(1.2, 0.5, 0)$ are used in the CRONE Toolbox, as frac_poly_exp($[2, 1, 3], [1.2, 0.5, 0]$), to represent the following fractional polynomial:

$$2s^{1.2} + s^{0.5} + 3 \tag{28}$$

When the sequence of differentiation orders is not ordered, while using the frac_poly_exp constructor, it is automatically reordered. Moreover, duplicated differentiation orders are merged by adding their respective coefficients, and the method cancel_zero_coef is provided to eliminate nill coefficients. The latter method is not automatically called in the constructor frac_poly_exp, to avoid eliminating a given coefficient which might get a zero value during an optimization procedure. Consequently, the constructor frac_poly_exp([3, 1, 0, 2, 3], [1.2, 0.5, 1.5, 1.2, 0]) produces the following fractional polynomial:

$$5s^{1.2} + 0s^{1.5} + s^{0.5} + 3. \tag{29}$$

Both attributes coef and order are of cell type in order to be able to create high dimensional fractional polynomials used later in frac_tf for defining Multi-Input-Multi-Output (MIMO) transfer functions. Consequently, the constructor frac_poly_exp($\{[1, 2], [1, 2, 3]; 2, 5\}$, $\{[1.2, 0.5], [1.5, 1.2, 0]; 0.5, 0\}$) produces a matrix of two by two fractional explicit polynomials:

$$\begin{bmatrix} s^{1.2} + 2s^{0.5} & 1.2s^{1.5} + 2s^{1.2} + 3 \\ 2s^{0.5} & 5 \end{bmatrix} \tag{30}$$

Coefficients and fractional powers are internally organized as cells, respectively:

$$\begin{Bmatrix} [1 \;\; 2] & [1.2 \;\; 2 \;\; 3] \\ [2] & [5] \end{Bmatrix}, \quad \begin{Bmatrix} [1.2 \;\; 0.5] & [1.5 \;\; 1.2 \;\; 3] \\ [0.5] & [0] \end{Bmatrix} \tag{31}$$

4.2 The frac_poly_imp class

In the CRONE toolbox, explicit-implicit functions (6) are defined in the frac_poly_imp class as a child of the frac_poly_exp class. It has an additional attribute: the implicit differentiation order $\beta \in \mathbb{R}$ of cell type in order to be able to create high dimensional implicit fractional polynomials. As an example, frac_poly_imp($\{[1, 2], [1.2, 2, 3]; 2, 5\}$, $\{[1.2, 0.5], [1.5, 1.2, 0]; 0.5, 0\}$, $\{0.2, 1.5, 0.5, 2\}$) creates the following matrix of fractional implicit polynomials:

$$\begin{bmatrix} (s^{1.2} + 2s^{0.5})^{0.2} & (1.2s^{1.5} + 2s^{1.2} + 3)^{1.5} \\ (2s^{0.5})^{0.5} & 5^2 \end{bmatrix} \tag{32}$$

4.3 The frac_lti class

The fractional-lti class has attributes common to and hence inherited in the three children classes, namely:

- var: contains variable name to be used in the representation of fractional transfer function and might be set to s, p, z, q. The default is s.
- ver: contains version number of different objects. It is required for further enhancements. This parameter is automatically filled and is not necessary to the end user.
- Ts: (optional) is the sampling period.
- N: (optional - necessary for Oustaloup and Trigeassou simulation methods) contains the number corner frequencies in (22) used in Oustaloup and Trigessou approximations.
- $[\omega_b, \omega_h]$: (optional - necessary for Oustaloup and Trigeassou simulation methods) contains the interval on which the fractional system is approximated.
- sim: (optional - if not set by the user, 'Grünwald' is the default simulation method when N and $[\omega_b, \omega_h]$ are not entered - otherwise the default simulation method is 'Oustaloup'.

Consequently the constructor may optionally be called with zero to five arguments ordered as specified here:frac_lti (var:char, Ts:double, N:double, Ω: $[1 \times 2]$ double, sim:string).

4.4 The frac_tf class

The fractional transfer function (7) is defined in the frac_tf class which allows representing uncommensurate fractional systems. The frac_tf class inherits attributes of the frac_lti class and has additionally two attributes, aggregated from frac_poly_exp class:

- num,

- den.

The constructor is called with two mandatory arguments: frac_tf (num:frac_poly_exp, den:frac_poly_exp) and may be called with five optional arguments, in the order specified by the frac_lti constructor. Moreover, matrix of transfer functions might be defined for Multi-Input-Multi-Output systems.

4.5 The frac_ss class

The fractional (or pseudo-) state space representation is based on (11) and (12). Consequently, the frac_ss class is constituted of the following attributes:

- A, B, C, D,
- the commensurate order ν.

This fractional state-space representation allows to represent MIMO systems and, A, B, C, D matrices are of appropriate dimensions.

4.6 The frac_zpk class

According to (10), a commensurate transfer function can entirely be defined by

- $z_i, p_j, K, \forall i = 1 \ldots m, \forall j = 1 \ldots n,$
- the commensurate order ν.

Consequently, the attributes constituting the frac_zpk class are z_i, p_j, K of cell type. The ν parameter is the commensurate order and in the case of MIMO systems, there is a single commensurate order for all subsystems.

5 Methods and functions of the CRONE-Toolbox

Inheritance property of the object oriented programming is used. Many methods being developed mainly for the frac_tf class, they are written in the frac_lti class to be used by all three children classes.

Only public methods are detailed in this section.

5.1 Methods associated to general operations

All object attributes in the CRONE Toolbox are encapsulated. The only way to access and change attributes is by using respectively get and set methods. This protects the object attributes from operations that are not intended for the object's class. Some additional methods are provided for general type operations: isnan, isempty, size, length, iscomplex.

Moreoever, methods associated to MIMO fractional transfer functions manipulations in frac_tf and frac_zpk classes are developed: horzcat, vertcat, subsref, subsasgn. The frac_ss class does not require to develop particular functions for MIMO fractional systems, since only matrix dimensions are changed.

5.2 Methods associated to operators overloading

The main operators $(+, -, \times, .\times, /, \backslash, ', =, \quad =, ...)$ are overloaded in different classes of the CRONE toolbox by implementing the associated methods plus, minus, uminus, mtimes, times, ldivide, rdivide, transpose, eq, ne, display.

5.3 Methods associated to simulation, control, and system identification of fractional systems

Several methods are developed for simulation, control and system identification: lsim, bode, nichols, nyquist, oe, srivcf, svf. Many of these methods are only developed for fractional transfer functions. They are however gathered in the frac_lti class. Prior to executing them, a calling object is first converted to a frac_tf object. Some of the methods are implemented for MIMO fractional transfer functions.

```
 1 sys=frac_tf(1,frac_poly_exp([5 1],[1.5  0]),...
 2   4, [1e-2, 1e2]);% Def of fractional sys
 3 u=[ones(1,800),zeros(1,600),ones(1,1000)];%input
 4 Te = 0.05; t = (0:length(u)-1)*Te; %time
 5
 6 % lsim with Grunwald method
 7 set(sys,'sim','grun'); yGrun = lsim(sys, u, t);
 8 % lsim with Oustaloup's rational approximation
 9 set(sys,'sim','Oust');yOust = lsim(sys, u, t);
10     ApproxOust=frac2int(sys);
11 % lsim with Trigeassou's rational approximation
12 set(sys,'sim','Trig'); yTrig = lsim(sys, u, t);
13     ApproxTrig=frac2int(sys);
14 figure(1), plot(t,yOust,'--',t,yTrig,'-.', ...
15     t,yGrun,'-', t,u,'k'), grid
16
17 [G, ph, w] = bode(sys, [1e-2  1e1]);
18 [GOust, phOust] = bode(ApproxOust, w);
19 [GTrig, phTrig] = bode(ApproxTrig, w);
20 figure(2), subplot(211)
21 semilogx(w, 10*log10(squeeze(GOust)), '--',...
22     w, 10*log10(squeeze(GTrig)), '-.', ...
23     w,10*log10(squeeze(G))),grid,
24 subplot(212), semilogx(w,squeeze(phOust),w,...
25     squeeze(phTrig), w, squeeze(ph)), grid,
26
27 sys2=(-sys^3+5*sys)
28 sysMIMO = [sys, sys ; sys2, sys2]
29 figure(3), lsim(sysMIMO, [u ; u], t);
```

Fig. 4. Matlab script using overloaded operators and methods of the CRONE toolbox

5.4 Functions associated to fractional derivatives

Some additional functions are provided in the CRONE Toolbox. dn, dnh functions allow to compute an explicit and an implicit real or complex derivative of a signal based on Grünwald's definition. tolord, used by the commensurate method, allows to change the desired precision on the differentiation order. binome, gammac, gammaic, used by other functions and methods, compute Newton's binomial, and the complete and the incomplete gamma functions in the whole complex plain.

6 Example

Consider the fractional transfer function:

$$\text{sys} = \frac{1}{5s^{1.5}+1} \tag{33}$$

defined and simulated in Matlab using the following code.

In line 1, the fractional system is defined. A constant value of 1 is entered in the numerator and a fractional explicit polynomial in the denominator. The additional parameters specify the number of cells to be used and the frequency band, in case Oustaloup's and Trigeassou's simulation methods are used. All three simulation methods, implemented in the CRONE toolbox, are used to simulate the transfer function in lines 7, 9, and 12. The comparison among these simulation methods is then plotted in Fig. 5. The overloaded method lsim of the CRONE toolbox is used for time-domain simulation. It uses the simulation method specified in the sim attribute of the frac_lti class.

The exact frequency response is computed by substituting s by $j\omega$ in the fractional transfer functions. It is computed by the overloaded bode method in line 17 of the algorithm. The exact frequency response is compared to frequency responses of their rational approximations computed in lines 18 and 19 using bode function of the Matlab Control toolbox. Note that there is no difference while using the Control Bode function and the CRONE Bode function.

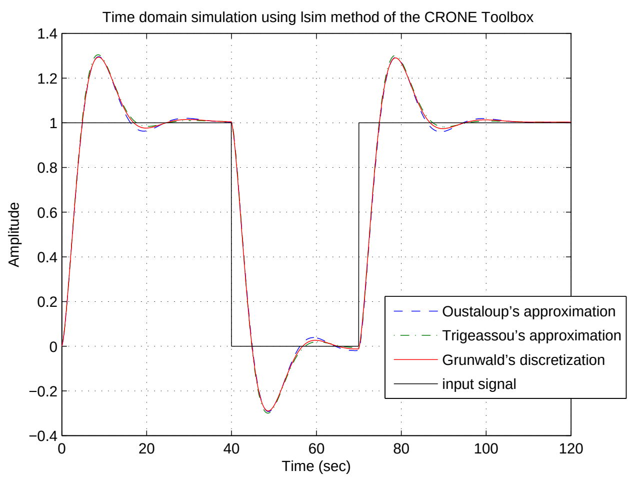

Fig. 5. Time domain simulations using three different approximation with lsim-method of the CRONE toolbox

```
1   Frac_tf transfer function :
2       ( 625 s^4.5 + 375 s^3 + 70 s^1.5 + 4 )
3  -------------------------------------------------
4  ( 625 s^6 + 500 s^4.5 + 150 s^3 + 20 s^1.5 + 1 )
5
6
7  Frac tf from input 1 to output:
8  #1 :  Frac_tf transfer function :
9       ( 1 )
10 ----------------
11 ( 5 s^1.5 + 1 )
12 #2 :  Frac_tf transfer function :
13      ( 1 )
14 ----------------
15 ( 5 s^1.5 + 1 )
16 Frac tf from input 2 to output:
17 #1 :  Frac_tf transfer function :
18      ( 625 s^4.5 + 375 s^3 + 70 s^1.5 + 4 )
19 -------------------------------------------------
20 ( 625 s^6 + 500 s^4.5 + 150 s^3 + 20 s^1.5 + 1 )
21 #2 :  Frac_tf transfer function :
22      ( 625 s^4.5 + 375 s^3 + 70 s^1.5 + 4 )
23 -------------------------------------------------
24 ( 625 s^6 + 500 s^4.5 + 150 s^3 + 20 s^1.5 + 1 )
```

Fig. 6. Displayed output while executing the script of Fig.4

In line 27 some overloaded operators are used. Since there is no semicolon at the end of line 27, the overloaded display method is automatically called and displays lines 1-4 of Fig.6.

A MIMO fractional transfer function is defined in line 28, by simple composition of SISO fractional transfer functions. The methods horzcat, vertcat, subsref, subsasgn are implemented for that purpose. Lines 7-24 of Fig.6 show the result of this composition as a fractional matrix of transfer functions. In line 29 of the algorithm the lsim method is called on a fractional MIMO transfer function. The result is plotted in Fig. 8.

7 Conclusion and Perspectives

The object oriented version of the CRONE Toolbox, a Matlab Toolbox dedicated to fractional calculus, is presented in this paper. It allows many enhacements such as the overloading of basic operators $(+, -, \times, /, .\times, ...)$ and standard

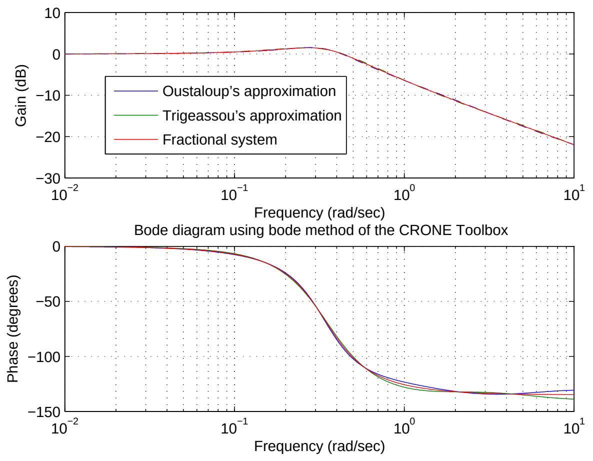

Fig. 7. Fractional system Bode diagram and its rational approximations

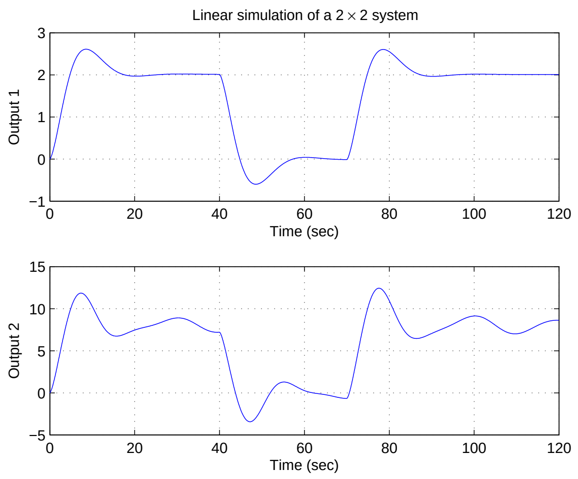

Fig. 8. Fractional system Bode diagram and its rational approximations

Matlab scripts (lsim, bode, nichols, ...) for the new classes. As a consequence, an end user familiar with standard Matlab operators and scripts can use straightforwardly the CRONE toolbox.

The object oriented CRONE toolbox is presently at a Beta version. It is provided for testing purposes. Once all the methods validated, a GUI will be developed and organized in several modules (mathematics, system identification, CRONE control, path planning).

References

Aoun, M., R. Malti, F. Levron and A. Oustaloup (2004). Numerical simulations of fractional systems: an overview of existing methods and improvements. *International Journal of Nonlinear Dynamics and Chaos in Engineering Systems. Special Issue: Fractional Derivatives and Their Applications* **38**(1-4), 117–131.

Grünwald, A.K. (1867). Ueber begrenzte derivationen und deren anwendung. *Zeitschrift für Mathematik und Physik* **12**, 441–480.

Monje, C.A., B.M. Vinagre, V. Feliu and Y.Q. Chen (2008). Tuning and auto-tuning of fractional order controllers for industry applications. *Control Engineering Practice* **16**(7), 798 – 812.

Oldham, K.B. and J. Spanier (1974). *The fractionnal calculus - Theory and Applications of Differentiation and Integration to Arbitrary Order.* Academic Press, New-York and London.

Ortigueira, M.D. and Machado, T.J.A., Eds.) (2006). *Signal Processing – special issue: Fractional calculus applications in signals and systems.* Vol. 86. Elsevier.

Oustaloup, A. (1983). *Systèmes asservis linéaires d'ordre fractionnaire.* Masson - Paris.

Oustaloup, A. (1995). *La dérivation non-entière.* Hermès - Paris.

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.

Samko, S.G., A.A. Kilbas and O.I. Marichev (1993). *Fractional integrals and derivatives: theory and applications.* Gordon and Breach Science.

Sommacal, L., P. Melchior, R. Malti and A. Oustaloup (2008). Synthesis of Havriliak-Negami functions for time-domain system identification. In: *17th World IFAC Congress.* Seoul South Korea. pp. 14283–14288.

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.

Xue, D. and Y. Chen (2005). Sub-optimum H_2 rational approximations to fractional order linear systems. In: *ASME IDETC/CIE Conferences.* Vol. DETC2005/CIE 84743.