A review and evaluation of numerical tools for fractional calculus and fractional order control

Article *in* International Journal of Control · November 2015

DOI: 10.1080/00207179.2015.1124290 · Source: arXiv

5 authors, including:

Lu Liu
Northwestern Polytechnical University
18 PUBLICATIONS **121** CITATIONS
SEE PROFILE

Sina Dehghan
University of California, Merced
11 PUBLICATIONS **43** CITATIONS
SEE PROFILE

YangQuan Chen
University of California, Merced
908 PUBLICATIONS **23,795** CITATIONS
SEE PROFILE

Dingyu Xue
Northeastern University (Shenyang, China)
148 PUBLICATIONS **4,243** CITATIONS
SEE PROFILE

International Journal of Control

ISSN: 0020-7179 (Print) 1366-5820 (Online) Journal homepage: http://www.tandfonline.com/loi/tcon20

A review and evaluation of numerical tools for fractional calculus and fractional order controls

Zhuo Li, Lu Liu, Sina Dehghan, YangQuan Chen & Dingyü Xue

To cite this article: Zhuo Li, Lu Liu, Sina Dehghan, YangQuan Chen & Dingyü Xue (2017) A review and evaluation of numerical tools for fractional calculus and fractional order controls, International Journal of Control, 90:6, 1165-1181, DOI: 10.1080/00207179.2015.1124290

To link to this article: https://doi.org/10.1080/00207179.2015.1124290

Accepted author version posted online: 30 Nov 2015.
Published online: 06 Jan 2016.

INTERNATIONAL JOURNAL OF CONTROL, 2017
VOL. 90, NO. 6, 1165–1181
http://dx.doi.org/10.1080/00207179.2015.1124290

Taylor & Francis
Taylor & Francis Group

A review and evaluation of numerical tools for fractional calculus and fractional order controls

Zhuo Li[a], Lu Liu[a,b], Sina Dehghan[a], YangQuan Chen [a,c] and Dingyü Xue[b]

[a]Department of Mechanical Engineering, School of Engineering, University of California, Merced, CA, USA; [b]College of Information Science and Technology, Northeastern University, Shenyang, Liaoning Province, China; [c]Mechatronics, Embedded Systems and Automation Laboratory (MESA Lab), University of California, Merced, CA, USA

ABSTRACT

In recent years, as fractional calculus becomes more and more broadly used in research across different academic disciplines, there are increasing demands for the numerical tools for the computation of fractional integration/differentiation, and the simulation of fractional order systems. Time to time, being asked about which tool is suitable for a specific application, the authors decide to carry out this survey to present recapitulative information of the available tools in the literature, in hope of benefiting researchers with different academic backgrounds. With this motivation, the present article collects the scattered tools into a dashboard view, briefly introduces their usage and algorithms, evaluates the accuracy, compares the performance, and provides informative comments for selection.

ARTICLE HISTORY
Received 12 July 2015
Accepted 21 November 2015

KEYWORDS
Fractional calculus; fractional order controls; numerical tools

1. Introduction

The fractional calculus (FC) got birth 300 years ago, and the research on FC experienced its boom in the past decades (Machado, Kiryakova, & Mainardi, 2011; Miller & Ross, 1993; Sabatier, Agrawal, & Machado, 2007). Besides the fundamental mathematical study, more and more researchers from different academic disciplines begin to utilise it in a variety of subject-associated research, such as in biology and biomedical (Magin, 2006; West, 2006), sociology (Lewis, 2014; West, Turalska, & Grigolini, 2014), economics (Ding, Granger, & Engle, 1993; Malkiel, 1999), and control engineering (Li & Chen, 2014; Monje, Chen, Vinagre, Xue, & Feliu, 2006; Yin, Chen, & Zhong, 2014). Along with the rapid development of theoretical study, the numerical methods and practical implementation also made considerable progress (Barbosa & Machado, 2006; Bohannan, 2008; Jiang, Hartley, Carletta, & Veillette, 2013).

Sharp tools are prerequisite to a successful job. In this paper, an extensive collection of Matlab-based tools are exhibited for the numerical computation of fractional order (FO) integration/differentiation, as well as some toolboxes for engineering applications, with an emphasis on FO controls. A comprehensive table (Table 1) is created to list the recapitulative information of these tools in a dashboard view. Brief description and basic evaluation of these numerical algorithms are presented, in terms of usage, accuracy, unique features, advantages, and drawbacks. Through such efforts, it is hoped that an informative guidance is provided to the readers when they face the problem of selecting a numerical tool for a specific application. Thanks to the authors of these tools. It is these pioneers who bring great convenience for the practical use of FC and FO control. While a text descriptive survey on some of the tools under discussion can be found in the book (Das & Pan, 2012), and 28 alternatives for the time-domain implementation of FO derivatives are documented in Valério and da Costa (2005), the present paper addresses more quantitative comparison and practical usage.

The rest of the paper are organised as follows: Section 2 reviews 20 selected numerical tools through brief description; Section 3 evaluates and compares the quantitative performance of the tools in three categories; Section 4 gives comments for tool selection based on empirical use.

2. Review and description

This article mainly covers the tools for fundamental FC, such as the numerical computation of fractional integration/differentiation of a function or a signal, and the Laplace transform of fractional differential equations (Podlubny, 1999a). Since automatic control is one of the engineering disciplines that got the earliest exposure to

CONTACT YangQuan Chen ychen53@ucmerced.edu; http://mechatronics.ucmerced.edu

Table 1. Matlab-based numerical tools for computation of fractional operations and fractional order controls.

#	Name	Typical usage	Sample syntax	Author(s)	Source	Delay	MIMO
1	fotf	FO control toolbox	s = fotf('s')	Dingyü Xue	Xue and Chen (2014a)	✓	Could
2	ninteger	FC and FOC toolbox	nid(k, a, [w1w2], 5,'crone')	D. Valério	Valério and da Costa (2004)	Could	Could
3	Crone	FO control toolbox	frac_tf(1, frac_poly_exp(1, 0.5)	CRONE team	The CRONE Team (2014)	×	✓
4	FOMCON	FO modeling and control	sys_foss = tf2ss(g)	A. Tepljakov	Tepljakov et al. (2011)	✓	✓
5a	mlf	2-param M–L func	y = mlf(a, b, −t)	I. Podlubny	Podlubny (2005)		
5b	ml_func	1 ∼ 4 param M–L func	y = ml_func([a, b, r], −t)	Dingyü Xue	Monje et al. (2006)		
5c	ml_fun	2-param M–L func	y = ml_fun(a, b, x, n, e)	S. Mukhopadhyay	Mukhopadhyay (2008)	N/A	N/A
5d	gml_fun	Generalized M–L func	gml_fun(a, b, r, x, eps0)	Y.Q. Chen	Chen (2008a)		
5e	ml	1,2,3-param M–L func	e = ML(x, a, b, r)	R. Garrappa	Garrappa (2014)		
6a	NILT	Num Inverse of Laplace	Script based	L. Brančík	Brančík (1999)	✓	×
6b	INVLAP	Num Inverse of Laplace	[t, y] = INVLAP('1/s', 1, 10, 100)	Code by Juraj	Juraj (2011)	✓	×
7	dfod1,2,3	Digital FO diff/int	sysdfod = dfod3(n, T, r)	I. Petráš	Petráš (2003a)	N/A	N/A
8	irid_fod ...	Impulse Resp Invariant	df = irid_fod(−.5, .1, 5)	Y.Q. Chen	Chen (2008b)	N/A	N/A
9	ora_foc	Oustaloup-Rec-Approx	ora_foc(0.5, 2, 0.1, 100)	Y.Q. Chen	Chen (2003)	N/A	N/A
10	fderiv	FO diff of r(t)	y = fderiv(0.5, r, Ts)	F. M. Bayat	bayat (2007)	N/A	N/A
11	glfdiff	Finite Diff of G–L	y1 = glfdiff(y, t, r)	Dingyü Xue	Xue and Chen (2014a)	N/A	N/A
12	fourier_diffint	FO diff of f(x)	fourier_diffint(f, x, ...)	G. Papazafeiropoulos	Papazafeiropoulos (2014)	N/A	N/A
13	FIT	FO integration toolbox	fracIntegrationSIM(...)	Marinov et al.	Marinov et al. (2013b)	N/A	N/A
14	DFOC	Discrete FO PID	DFOC(K, Ti, Td, m, d, Ts, n)	I. Petráš	Petráš (2011b)	N/A	×
15	FOPID	FO PID	—	Lachhab et al.	Lachhab et al. (2013)	—	×
16	FOCP	Fractional optimal control	Calling RIOTS	C. Tricaud et al.	Tricaud (2009)	×	✓
17	FSST	FO S-S toolkit	Simulink blocks	D. Sierociuk	Sierociuk (2003)	✓	✓
18	FVO	Fractional variable order	ban(alpha, N, h)	Podlubny et al.	Podlubny (2000)	N/A	N/A
19	forlocus	RL plot of FO TFs	forlocus(num, den, l)	Zhuo Li et al.	Li (2015)	N/A	N/A

The 'Delay' column denotes whether the script/toolbox is able to handle time delay in the FO model.
The 'MIMO' column denotes whether the script/toolbox is able to handle MIMO FO models.

FC (Axtell & Bise, 1990; Bagley & Calico, 1989; Bode, 1945; Podlubny, 1999b), the tools for the application of FO controls are given more focus, associated with the authors' expertise.

2.1 @fotf

@fotf (fractional order transfer function) is a control toolbox for FO systems developed by Xue et al. Most of the functions inside are extended from the Matlab built-in functions. In Chen, Petráš, and Xue (2009), the code and usage of the @fotf toolbox are described in very detail. It uses the overload programming technique to enable the related methods of the Matlab built-in functions to deal with FO models. The transfer function objects generated from it can be interactive with those generated from the Matlab transfer function class. Yet, the overloading of associated functions, such as impulse() and step(), lost the plotting functionality. As a work around, users can simply define a time vector as the second input to these functions. fotf toolbox supports time delay in the TF, e.g. fotf(a, na, b, nb, delay). It does not directly support transfer function matrix, hence, MIMO systems cannot be simulated directly. However, since it provides Simulink block encapsulation of the involved function fotf(), multiple input/output relationship can be established by manually adding loop interactions in Simulink block diagrams. Therefore, the remark 'could' is put in the 'MIMO' column in Table 1 (where the 'Delay' column denotes whether the script/toolbox is able to handle time delay in the FO model; and the 'MIMO' column denotes whether the script/toolbox is able to handle MIMO FO models.).

A small drawback with @fotf is that the sampling time has relatively big impact on the accuracy, which has been remarked in the validation comments in Chen et al. (2009). Encouragingly, an update is upcoming according to the author.

2.2 Ninteger

Ninteger, non-integer control toolbox for Matlab, is a toolbox intended to help with developing FO controllers and assessing their performance (Valério & da Costa, 2004). It uses integer order transfer functions to approximate the FO integrator/differentiator, $C(s) = ks^{\nu}$, $\nu \in \mathbb{R}$. It offers three frequency domain approximation methods,

(1) The CRONE method, which uses a recursive distribution,

$$C(s) = k' \prod_{n=1}^{N} \frac{1 + s/\omega_{zn}}{1 + s/\omega_{pn}};$$

(2) The Carlson's method, which solves $C^{\alpha}(s)$ using Newton's iterative method,

$$C_n(s) = C_{n-1}(s) \frac{(\alpha - 1)C_{n-1}^{\alpha}(s) + (\alpha + 1)g(s)}{(\alpha + 1)C_{n-1}^{\alpha}(s) + (\alpha - 1)g(s)};$$

(3) The Matsuda's method, which approximates C with a gain known at several frequencies.

$$C(s) = [d_0(\omega_0);\quad (s-\omega_{k-1})/d_k(\omega_k)]_{k-1}^{+\infty},$$
$$d_0(\omega) = |C(j\omega)|,\quad d_{k+1}(\omega) = \frac{\omega-\omega_k}{d_k(\omega)-d_k(\omega_k)}.$$

It also provides Simulink block encapsulation of the involved functions, such as 'nid' and 'nipid' blocks. Moreover, it offers a user-friendly GUI for fractional order PID (FOPID) controller design.

There is a problem with ninteger toolbox in Matlab version 2013*a* or later. Without additional editing, it has conflicts with some built-in functions due to the overload editing of the Matlab built-in function isinteger(). For example, calling the mean() function will prompt an error.

2.3 ooCroneToolbox

The CRONE Toolbox, developed since the nineties by the CRONE team, is a Matlab and Simulink toolbox dedicated to applications of non-integer derivatives in engineering and science (Oustaloup, Melchior, Lanusse, Cois, & Dancla, 2000). It evolved from the original script version to the current object-oriented version. A good feature of the Crone toolbox is that some of the methods are implemented for MIMO fractional transfer functions. For example, executing sysMIMO = [sys, sys; sys2, sys2] generates a two-input-two-output TF matrix. Many simulation results in the literature are obtained using the CRONE toolbox such as the design of centralised CRONE controller with the combination of the MIMO-QFT approach in Yousfi, Melchior, Rekik, Derbel, and Oustaloup (2012). Several other toolboxes are inspired by CRONE, e.g. ninteger and FOMCON. A drawback of the CRONE toolbox is that time delay cannot be incorporated into the generated FO TF. Manually multiplying the delay to the frac_tf object does not work either because the exp() operation is not overloaded by frac_tf class. CRONE is a toolbox much more powerful than merely simulating FO systems. In spite of this basic functionality, it is also capable of FO system identification and robust control analysis and design.

2.4 FOMCON

The FOMCON (fractional-order modelling and control) toolbox is developed by Tepljakov, Petlenkov, and Belikov (2011). Its kernel utilises the algorithms in FOTF, Ninteger, and CRONE. It encapsulates some of the major functionalities of those three toolboxes, and builds a GUI shell on top, aiming at extending classical control schemes for FO controller designs. The relation of FOMCON with the three toolboxes is shown in Figure 1. Some notable changes/patches to the original FOTF are:

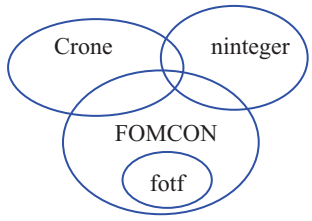


Figure 1. FOMCON's relation to other numerical tools (Tepljakov, 2011).

- newfotf() uses the string parser to enable users to input TF as a string;
- tf2ss() is overloaded and foss() is added, which makes the conversion between an FO TF object and an FO state-space (SS) object possible. The CRONE toolbox is also able to do the task, yet the script is encrypted in Matlab P code format.

2.5 M–L functions

M–L functions, as the name implies, are Matlab functions developed for numerically computing the Mittag–Leffler function (definitions can be found in Magin (2006), etc). There are several versions of code by different authors available in the literature. Five of them are listed in Table 1, where

(1) mlf(α, β, x, p) is for the calculation of the two-parameter M–L function in the form of $E_{\alpha,\beta}(x)$ with the precision of p for each element in x;
(2) ml_func([α, β, γ, q], z, n, ε_0) is capable of computing the M–L function with either 1, 2, 3, or 4 parameters, and the script is available in the books by Xue and Chen (2014b) and Monje et al. (2006). It uses the fast truncation algorithm to improve the efficiency, and embeds the mlf() in the file such that when the fast truncation algorithm is not convergent, solution is guaranteed by trading off some efficiency;
(3) ml_fun(α, β, x, n, ε_0) ($\alpha > 0$, $\beta > 0$) is also for two-parameter M–L function with error tolerance of ε_0, which is implemented using C-MEX .dll (dynamic-link library) technique and can be used in Simulink through s-functions;
(4) gml_fun(α, β, γ, x, ε_0) calculates the generalised M–L function with three parameters in the form of $E^{\gamma}_{\alpha,\beta}(x)$ (Prabhakar, 1971);
(5) ml(x, α, β, γ) can calculate the M–L function with either 1, 2, or 3 parameters.

Alternatively, the generalised hypergeometric function [pfq] = genHyper(a, b, z, lnpfq, ix, nsigfig) in

Barrowes (2005), or [y, tt, nterms] = pfq(a, b, z, d) in Huntley (2012) can also achieve the numerical computation of the generalised M–L functions under certain conditions. For more details, refer to Chaurasia and Pandey (2010).

2.6 NILT

The inversion of Laplace transform is fundamentally important in the applications of the Laplace transform method. It can be carried out with one of the following three approaches: (1) analytical solution using definition and basic properties; (2) Laplace transform tables; and (3) numerical computation. While analytical solutions are usually too hard to be obtained, and tables do not cover arbitrary cases, the numerical computation becomes an inevitable way. Among the numerous algorithms for numerical inversion of Laplace transform (NILT), NILT in de Hoog, Knight, and Stokes (1982), Liang (2005) and the 'improved NILT' in Valsa and Brancˇík (1998), Brancˇík (1999), and Brancˇík (2001) have relatively bigger literature exposure. Lubomir's NILT method applies the fast Fourier transformation (FFT) and the ε-algorithm to speed up the convergence of infinite complex Fourier series. A very detailed description and performance evaluation of these methods is available in Sheng, Li, and Chen (2011). Hence, repetitive comparison among different NILTs are not presented here. Focus is mainly put on the comparison between NILT and other numerical methods.

A good feature of the two NILT code is that both support the direct input of time delay in the form of exp(−Ls). Yet, INVLAP() gives some glitch at the end of the delay, for example, [x, y] = INVLAP('1/(s* (s^0.5+1)) * exp(-s)', 0.01,10,1000). There is a tricky part need to be noted in evaluating the computational error of NILT. If the same initial, terminating, and sampling time (t_0, t_f and T_s) for other tools are used in the script, the NILT actually computes one point less than the other tools which use regularly spaced time vector. That is because: let $M = \frac{t_f - t_0}{T_s}$ represent the amount of points computed by NILT, then, the time interval is actually $T_s' = \frac{t_f - t_0}{M-1}$ due to the script t = linspace(0, tm, M), whereas the conventional assignment of time vector (t = t0 : Ts : tf) generates M + 1 points. In order to compute the same amount of points aligned to the time stamps used for baseline analytical solution, the time vector for analytical computation needs to be adjusted so as to adapt to that used by NILT. This means to let analytical computation use the time vector generated by NILT, which can be achieved by either (1) t = 0 : M ∗ Ts/(M − 1) : M ∗ Ts, or (2) t = linspace(0, tm, M). This cannot be done the other way around, i.e. replaced by t = 0 : Ts : M ∗ Ts − Ts nor t = linspace(0, tm − Ts, M). Otherwise, cumulated computation error will cause inaccuracy of the final simulation result. Alternatively, if t_f is not a concern, user can assign one point less to M in the NILT script while keeping T_s unchanged. Thus, NILT generates the same time stamps except a t_f shortened by one sampling period. The difference in dealing with time vectors can be easily visualised if longer sampling time is assigned. An example of the resulting computation error is demonstrated in Figure 2. A similar time stamp assignment issue exists in INVLAP(). In addition, the initial time stamp is not allowed to be 0 due to the constraint in the INVLAP() script.

2.7 dfod

dfod (digital fractional order differentiator/integrator) is a set of Maltab functions written by Petráš, for the approximation of FO differentiators and integrators. There are three versions of dfod:

(1) dfod1() is the infinite impulse response (IIR) type based on continued fraction expansion (CEF), shown in Equation (1), of weighted operator with the mixed scheme of the trapezoidal (Tustin) rule and the backward difference (Euler) rule (Petráš, 2003a);

$$Z\{D^{\alpha}x(t)\} = CFE\left\{\left(\frac{1-z^{-1}}{T}\right)^{\alpha}\right\}X(z) \approx \left(\frac{1}{T}\right)^{\alpha}\frac{P_p\left(z^{-1}\right)}{Q_q(z^{-1})}X(z). \quad (1)$$

(2) dfod2() is the finite impulse response (FIR) type based on power series expansion (PSE), shown in Equation (2), of the backward difference (Euler) rule (Petráš, 2003b);

$$D^{\mp\alpha}(z) = \frac{1}{(1-z^{-1})^{\pm\alpha}} = \frac{T^{\mp\alpha}}{\sum_{j=0}^{\infty}(-1)^j\binom{\pm\alpha}{j}z^{-j}} \approx \frac{T^{\mp\alpha}}{Q_q(z^{-1})} \quad (2)$$

(3) dfod3() is a new IIR type based on power series expansion of the trapezoidal (Tustin) rule Petráš (2011a).

$$\text{Euler}: s^{\alpha} \approx \left[\frac{1-z^{-1}}{T}\right]^{\alpha}, \quad \text{Tustin}: s^{\alpha} \approx \left[\frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}\right]^{\alpha}. \quad (3)$$

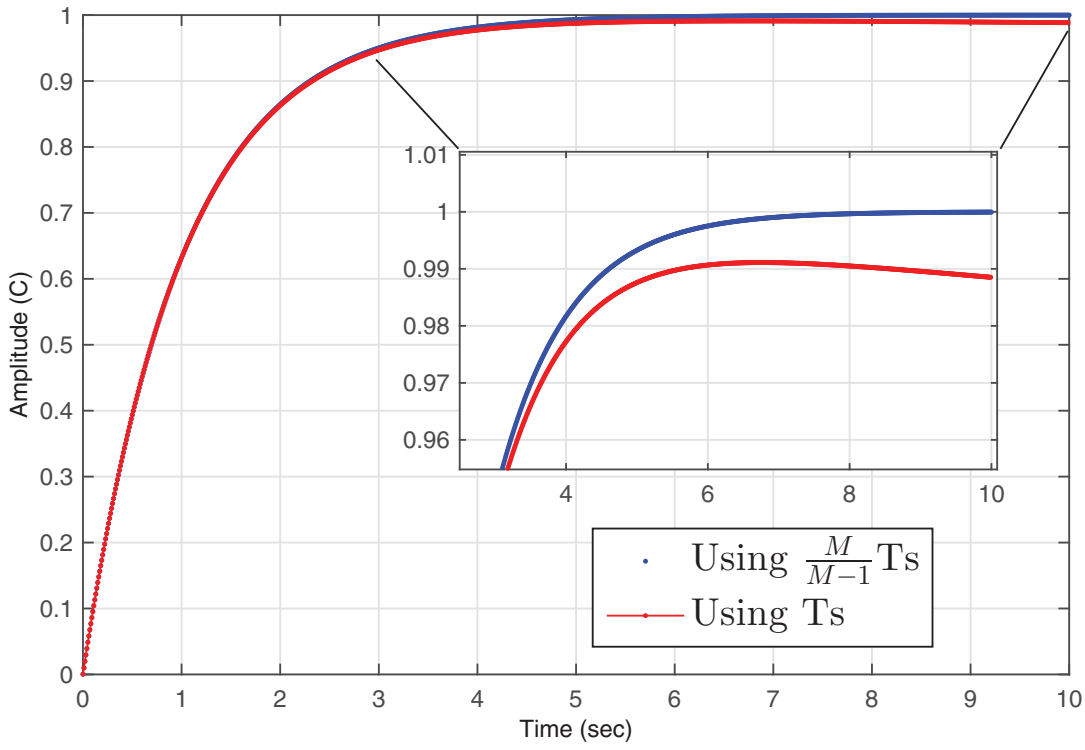


Figure 2. Computation error of NILT caused by mis-assignment of T_s.

There are other FO algorithms based on IIR, such as newfod() by Chen and Vinagre (2003).

Regarding discretisation, besides the aforementioned methods used in the various tools, other methods exist such as the Prony's technique, direct discretisation, and the binomial expansion of the backward difference (Caponetto, Dongola, Fortuna, & Petráš, 2010).

2.8 IRID

The impulse response invariant discretisation (IRID) is a family of functions designed by Li, Sheng, and Chen (2010) and Chen (2008b), for different approximation purposes based on the algorithm as its name implies. It includes the following members:

(1) irid_fod() is designed to compute a discrete-time finite dimensional (z) transfer function to approximate a continuous irrational transfer function s^α where 's' is the Laplace transform variable and $-1 < \alpha < 1$. It has been tested that the algorithm still works for $\alpha > 1$ and $\alpha < -1$, by removing the input checking statement.

(2) irid_doi() is for the approximation of distributed order integrator $\int_a^b \frac{1}{s^\alpha} d\alpha$, where '$a$' and '$b$' are arbitrary real numbers in the range of (0.5, 1), and $a < b$.

(3) irid_dolp() is for the approximation of a continuous-time fractional order low-pass filter in the form of $1/(\tau s + 1)^\alpha$

(4) irid_fsof() is for the approximation of fractional second-order filter in the form of $1/(s^2 + as + b)^\alpha$ where $0 < \alpha < 1$.

(5) BICO_irid() is for the approximation of BICO (Bode's ideal cut-off) transfer function in the form of $1/(s/w_0 + \sqrt{(s/w_0)^2 + 1})^\alpha$, where $\alpha > 0$.

2.9 ora_foc

ora_foc() is for the approximation of FO differentiators, $\frac{1}{s^\alpha}$ (Xue, Chen, & Atherton, 2009), using the Oustaloup-recursive-approximation method described in Oustaloup, Levron, Mathieu, and Nanot (2000).

2.10 fderiv

fderiv() calculates the fractional derivative of order α for the given function $r(t)$ using the Grünwald–Letnikov (G–L) definition (Bayat, 2007). The input of the given function is represented by a vector of signal values.

There is an improved implementation of this function, fgl_deriv(), by Jonathan, which uses vectorisation for faster computation with Matlab (Jonathan, 2014).

2.11 glfdiff

glfdiff(y,t,α) (G–L finite diff) is a Matlab function written by Xue and Chen (2014a) for calculating the αth derivative of a given function, whose inputs y and t are the signal and time vectors. It is based on the forward finite difference approximation of the G–L definition,

$$ {}_aD_t^{\alpha} f(t) \approx \frac{1}{h^{\alpha}} \sum_{j=0}^{(t-1)/h} \omega_j^{(\alpha)} f(t - jh), \tag{4} $$

where the binomial coefficients are recursively calculated (Xue and Chen, 2014a):

$$ \omega_0^{(\alpha)} = 1,\ \omega_j^{(\alpha)} = \left(1 - \frac{\alpha + 1}{j}\right) \omega_{(j-1)}^{(\alpha)},\ j = 1, 2, \ldots \tag{5} $$

2.12 Fractional differentiation and integration

Many of the above functions approximate the FO integral or derivative operator. This Matlab function calculates the αth-order derivative or integral of a function, defined in a given range through Fourier series expansion. The necessary integrations are performed with the Gauss–Legendre quadrature rule (Papazafeiropoulos, 2014). Three examples are offered in this package, namely FO differ/integral of identity, cubic polynomial, and tabular functions, respectively. The main call function is fourier_diffint().

2.13 FIT

FIT is the fractional integration toolbox developed by Santamaria Laboratory at the University of Texas at San Antonio (Marinov, Ramirez, & Santamaria, 2013b). It is for the numerical computation of fractional integration and differentiation of the Riemann–Liouville (R–L) type, and is designed for large data size, which allows parallel computing of multiple fractional integration/differentiation on GPUs (graphical processing units). The extrapolation and interpolation algorithms used by this toolbox are implemented in C++ and are integrated with Matlab via the MEX mechanism. A detailed explanation can be found in Marinov, Ramirez, and Santamaria (2013a).

2.14 DFOC

DFOC, written by Petráš, is a digital version of the FOPID controller of the form:

$$ C(s) = K + T_i \frac{1}{s^m} + T_d s^d. \tag{6} $$

It provides a transfer function of the FO PID controller for given parameters (Petráš, 2011b).

2.15 FOPID

The FOPID controller toolbox, presented by Lachhab, Svaricek, Wobbe, and Rabba (2013), is for the design of robust fractional order $PI^{\alpha}D^{\beta}$ controllers. The tuning rules for the parameters follow those promoted in Li, Luo, and Chen (2010) and Luo and Chen (2013). Thus, the FOPID tuning is converted to a five-parameter optimisation problem. This toolbox utilise the 'non-smooth' H_{∞} synthesis in Apkarian and Noll (2006) to perform the minimisation. For now, there is not a publicly available source for download.

2.16 Sysquake FO PID

Pisoni, Visioli, and Dormido (2009) presented an interactive tool for FOPID controllers developed on the Sysquake software environment, which is a similar effort with that for integer order PIDs done by Åström et al. in Guzman et al. (2008). Sysquake is a numerical computing environment based on a programming language mostly compatible with Matlab. However, the interactive tool for FOPID runs in the Sysquake environment instead of Matlab. Hence, it is not reviewed in detail here.

2.17 FOCP

Tricaud and Chen (2009) formulated the fractional optimal control problems (FOCPs) into the integer order format by using a rational approximation of the fractional derivative obtained from the singular value decomposition (SVD) of the Hankel matrix of the impulse response. Then, RIOTS_95 (Tricaud & Chen, 2008; Zhao, Chen, & Li, 2014) is used to perform the optimisation. The scheme is potentially able to solve any type of FOCPs and is implemented in Matlab for public accessibility (Tricaud, 2009). It supports MIMO FO optimal control, but does not handle time delay due to the limitation of RIOTS.

2.18 FSST

FSST is a simulation toolkit in Matlab/Simulink for the FO discrete SS system education. The toolkit consists of a set of C-MEX s-functions which are encapsulated

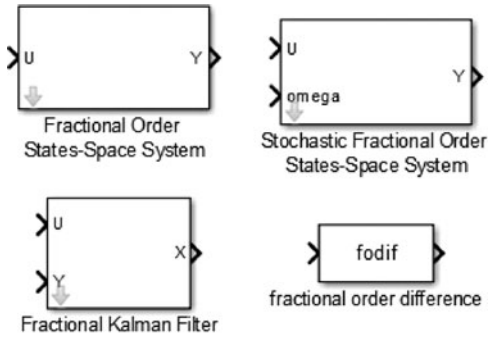


Figure 3. The Simulink block set provided in FSST.

in Simulink blocks. Several typical FO system simulation examples are provided as shown in Figure 3, such as the FO SS model and the fractional Kalman filter (FKF) (Dzieliński & Sierociuk, 2008). The version 1.7 is available for free download at Sierociuk (2003). Two of the superior strengths of FSST are (1) it can directly simulate MIMO systems since it is a Simulink block kit handling SS representations; (2) it is able to incorporate the initial conditions into the dynamic equations to be simulated, which is a unique feature among all the aforementioned tools. The drawback of FSST is that the step size has large impact on the simulation results, even larger than the impact by 'circular' buffer size. A sample illustration is plotted in Figure 4.

2.19 Fractional variable orders

All the above tools/toolboxes (except irid_doi()) deal with constant FOs. Yet, there exists a type of differentiations that have fractional variable orders (FVOs). The definitions in the G–L format are given as follows (Lorenzo & Hartley, 2002):

Theorem 1 (The first type FVO):

$$ {}_0D_t^{\alpha(t)} f(t) = \lim_{h\to 0} \frac{1}{h^{\alpha(t)}} \sum_{r=0}^{n} (-1)^r \binom{\alpha(t)}{r} f(t-rh). \tag{7} $$

The second and third types can be found in the same reference.

Regarding the FVO differentiation, there are dedicated tools. Podlubny (2000) offers a matrix approach that unifies the numerical differentiation of integer order and the n-fold integration, using the so-called triangular strip matrices. It is available for download at Podlubny (2008) and can be applied on the solution to FODEs and FPDEs.

Sierociuk (2012) provides a C-MEX s-function-based Simulink toolkit, 'fvoderiv', for this purpose. It supports real-time workshop.

The toolbox 'vod' created by Valério et al. calculates variable fractional or complex order derivatives. R–L, Caputo, and G–L definitions are provided; the three types of definitions in Lorenzo and Hartley (2002) are all considered. Fuzzy-supervised implementations in Simulink are also provided (Valério, 2009).

2.20 FO root locus

Three Matlab-based scripts for plotting root locus of FO TFs are available. Two early works are frlocus() in bayat (2008), and the code attached in the paper by Machado

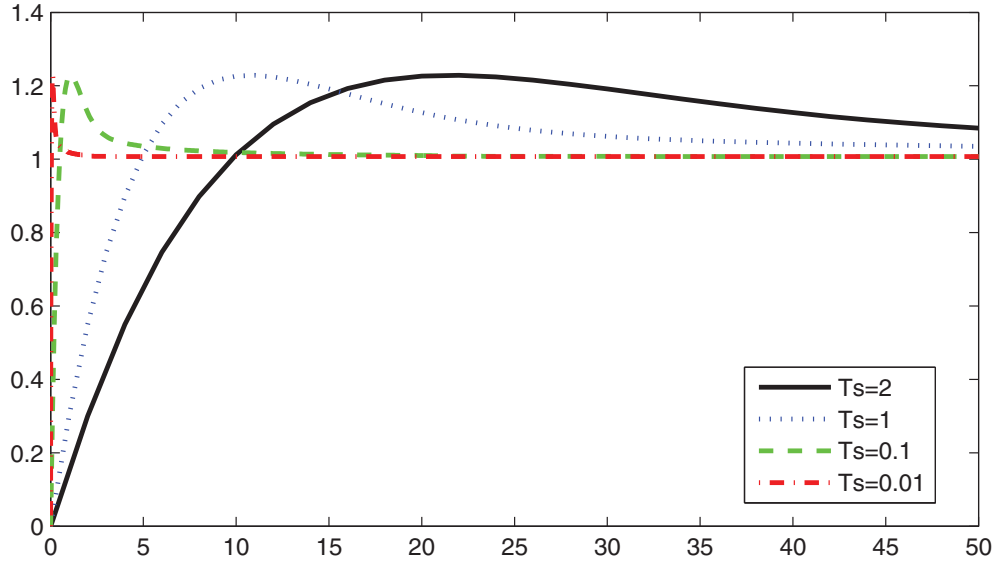
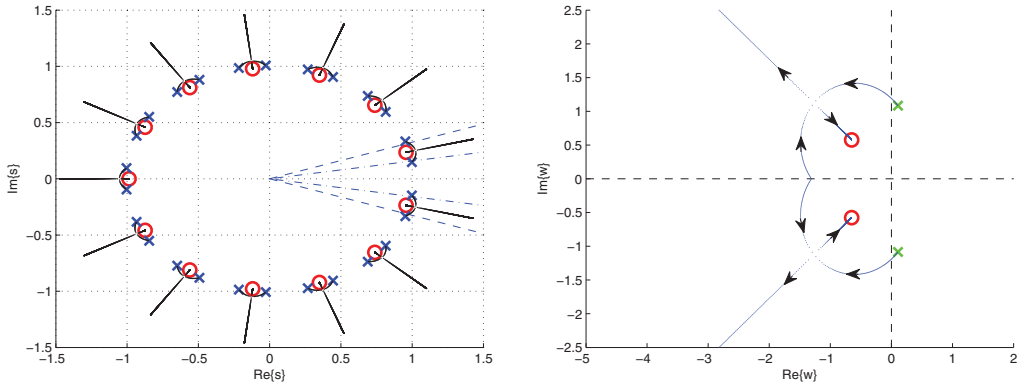


Figure 4. The impact of simulation step size on the FSST toolbox.



(a) On s-plane. (b) On w-plane.

Figure 5. The RL plot of equation (8) on different planes. (a) On s-plane. (b) On w-plane.

(2011). The other is forlocus() developed by the author which is listed in the last row in Table 1 and can be downloaded from Li (2015). Besides, the newest version of @fotf toolbox also features the root locus plot of FO systems. Figure 5 shows a demonstrating plot of the root locus of the following FO transfer function:

$$G(s) = \frac{1.2s^{1.3} + 1}{0.8s^{2.6} + 0.6s^{1.3} + 1}, \tag{8}$$

where Figure 5(a) shows the plot on the Laplace s-plane and Figure 5(b) shows the plot on the $w = s^{1.3}$ plane. A closer view of the second quadrant in Figure 5(b) tells that the root locus in this example has two branches on the first layer of the Riemann sheet (Corless & Jeffrey, 1998; Farkas & Kra, 1980). One starts from the pole marked in green, and the other is from the next Riemann sheet. As the system gain varies, they aggregate at $(-1.25 + 1.1i)$ and then bifurcate. One approaches the open loop zero marked in red and the other goes to infinity.

2.21 Other tools

Text description of a few tools listed above can also be found in Petráš (2011c). There are other FC-related tools or Matlab scripts available for specific applications, such as the fractional Fourier transform (FrFT) (Narayanana & Prabhu, 2003; Ozaktas, Zalevsky, & Kutay, 2001), closed-form solutions to linear FO differential equations, fode_sol() (Monje et al., 2006), the M–L random number generator mlrnd() (Germano, 2008), digital fractional order Savitzky–Golay differentiator (Chen, Chen, & Xue, 2011), and the functions for simulating FO chaotic systems (Petráš, 2011d). Considering the scope of research, they are not enumerated here and only fundamental FC and FO control-related tools are reviewed.

3. Evaluation and comparison

3.1 Comparison I

To evaluate the collected tools, several groups of benchmark problems and inputs are designed. For the FO control toolboxes, the following problems are used:

(1) Baseline model: first-order transfer function,

$$g_b(s) = \frac{1}{s+1},$$

whose time domain analytical solution of its step response is $y(t) = 1 - e^{-t}$;

(2) Impulse response of half-order integrator:

$$g_{hint}(s) = \frac{1}{\sqrt{s}},$$

whose time domain analytical solution is $\frac{1}{\sqrt{\pi t}}$;

(3) TF with a half-order pole:

$$g_{hp}(s) = \frac{1}{\sqrt{s}+1},$$

whose time domain analytical solution is

$$\frac{1}{\sqrt{t}} E_{\frac{1}{2},\frac{1}{2}}(-\sqrt{t}), \text{ or equivalently,}$$
$$\frac{1}{\sqrt{\pi t}} - e^t erfc[\sqrt{t}];$$

(4) The commensurate order TF:

$$g_{com}(s) = \frac{6s^{1.2} + s^{0.8} + 2s^{0.4} + 3}{5s^{1.6} + s^{0.8} + 2};$$

Table 2. Quantitative evaluation results for the test problems 1–5.

Method\ Error	1	2	3	4	5
M	0	–	–	–	–
1	1.4955	8.4176	6.6813	'0'	'0'
2	3.18×10^{-13}	2.5287	0.3831	9.8434	2.5519
3	0.4956	2.9627	1.4254	10.454	3.1857
5a	0	–	4.69×10^{-4}	–	–
5c	8.62×10^{-12}	–	1.08×10^{-10}	–	–
6a	0.0016	0.0236	0.0206	6.1528	2.4477
6b	0.0059	0.0012	2.49×10^{-5}	3.4722	1.5042
8	0.5327	0.0071	0.2189	–	–

(5) Step response of the irrational order TF:

$$g_{ir}(s) = \frac{2s^{\sqrt{3}} + 1}{s^{\sqrt{5}} + 3s^{\sqrt{2}} + 1}.$$

The accuracy is quantified by the conventional integral absolute error (IAE) criteria, $S = \int_0^T |e(t)|dt$. All comparisons have been kept as fair as possible. The numerical values of the time domain analytical solution using Matlab built-in functions are assumed to be accurate and are adopted as the baseline. The computational errors when $T_s = 0.05$ are summarised in Table 2, where the row indices represent the methods numbered in Table 1, and the column indices represent the test problems respectively. Besides, 'M' denotes the Matlab built-in TF and '–' means the underlying method is not applicable for the test problem. Two sample plots of the step responses of problems 1 and 5 are shown in Figures 6 and 7. For problems 4 and 5, since analytical solution is hard to obtain, all methods are compared to the values computed by fotf.

For the impulse response of the half-order integrator, the first point is ignored for error calculation because it is infinity. Two graphic views of the comparison are shown in Figure 8 a and 8 b, with Ts = 0.01 and Ts = 0.1, respectively.

As stated in Chen (2008b), irid_fod() uses finite dimensional (z) TF as the approximation method. Hence, the order of the (z) TF has impact on the approximation accuracy. The error listed in Table 2 is based on the 10^{th} order approximation. An illustrative plot is shown in Figure 11. The sampling time also has impact on its accuracy. The anti-intuitive fact is that relatively greater Ts gives higher accuracy. A heat map of the error on the field of Ts = 0.01 : 0.001 : 0.1 and order = 3 : 30 is plotted in Figure 12. At some particular high orders, 'rank deficiency' would occur during the call of prony() inside irid_fod(). Users can choose appropriate orders according to their specific accuracy requirement.

The analytical expression of M–L function is a summation of infinite terms. Hence, it is not surprising to see the numerical computation-induced error in the results.

3.2 Comparison II

fderiv(), glfdiff(), fourier_diffint() and FIT are integration /differentiation tools for functions. For this group of

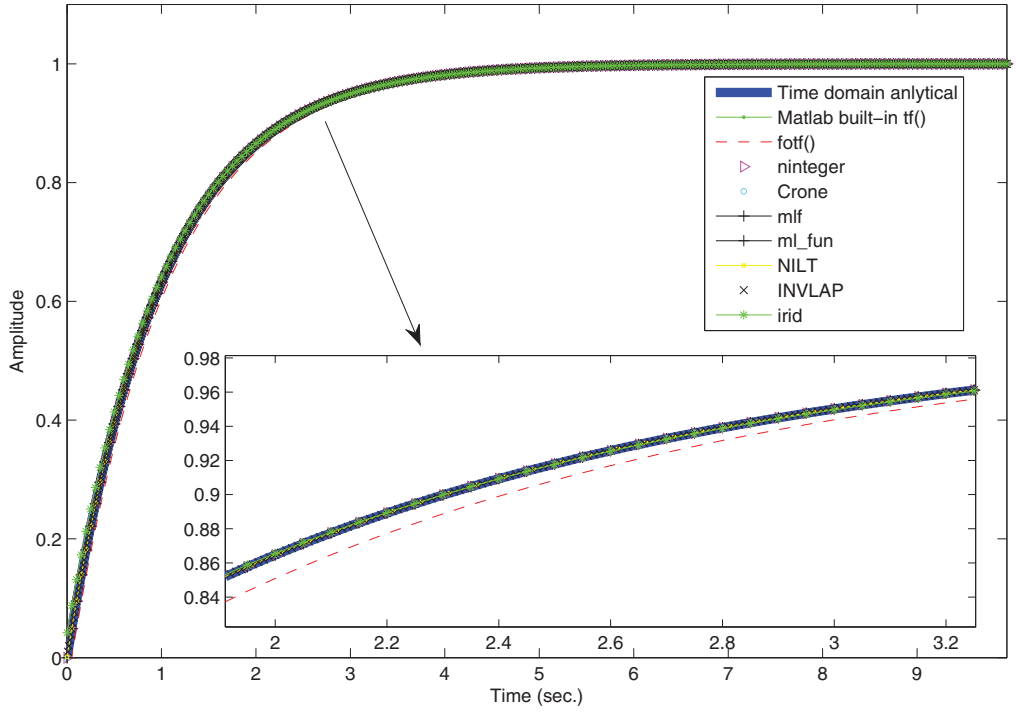


Figure 6. Comparison of the step responses of problem 1.

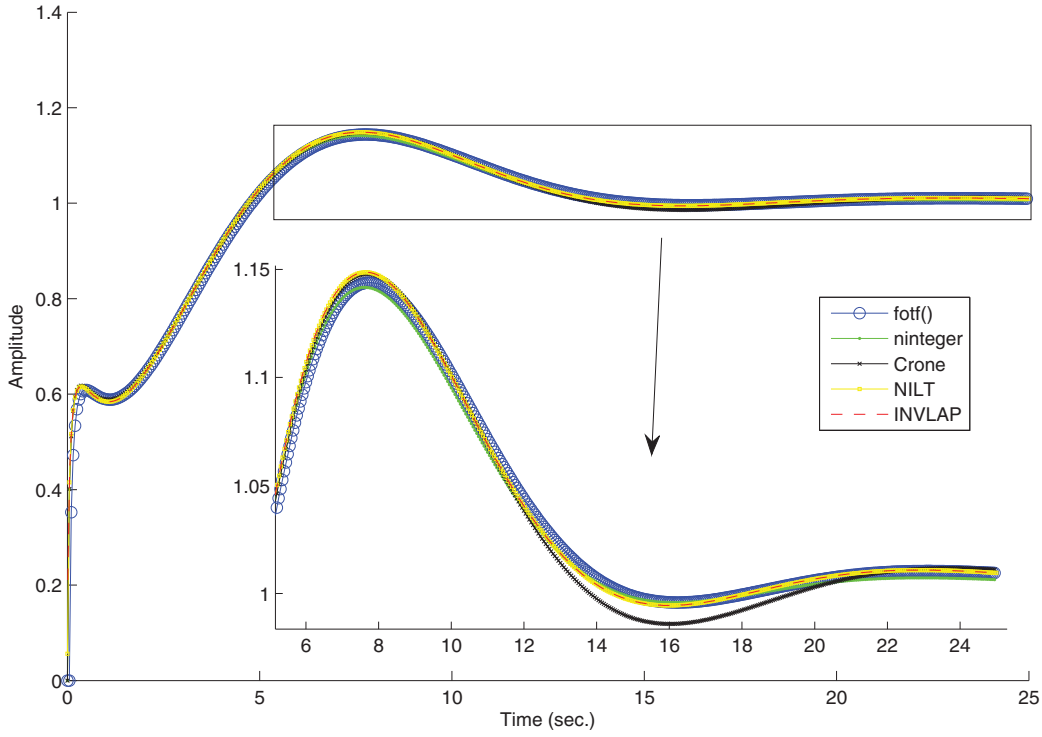


Figure 7. Comparison of the step responses of problem 5.

tools, the following two problems are designed to compare the performance:

(1) Half-order derivative of the function $y(t) = 3t$ on the interval of $[0, 5]$, whose analytical solution is

$$ {}_0D_t^{0.5} y = \frac{3\Gamma(2)}{\Gamma(1.5)}\sqrt{t}. \tag{9} $$

(2) 0.75 -order integration of the function $y(t) = \sqrt{t}$, whose analytical solution is

$$ {}_0D_t^{-0.75} y = \frac{\Gamma(1.5)}{\Gamma(2.25)} t^{1.25}. \tag{10} $$

The time steps are all set to 0.01 sec. It can be seen that fourier_diffint() performs not as well as other methods although a big number of Fourier and Gaussian coefficients have been assigned (default values are 260 and 520 for identity polynomial). Its performance on a third-order polynomial is better. The results are plotted in Figures 9 and 10. Quantitative comparison including computational error and averaged elapsed time (for 20 runs each) are listed in Table 3, for the above two problems respectively.

Table 3. Quantitative comparison of function int/diff tools.

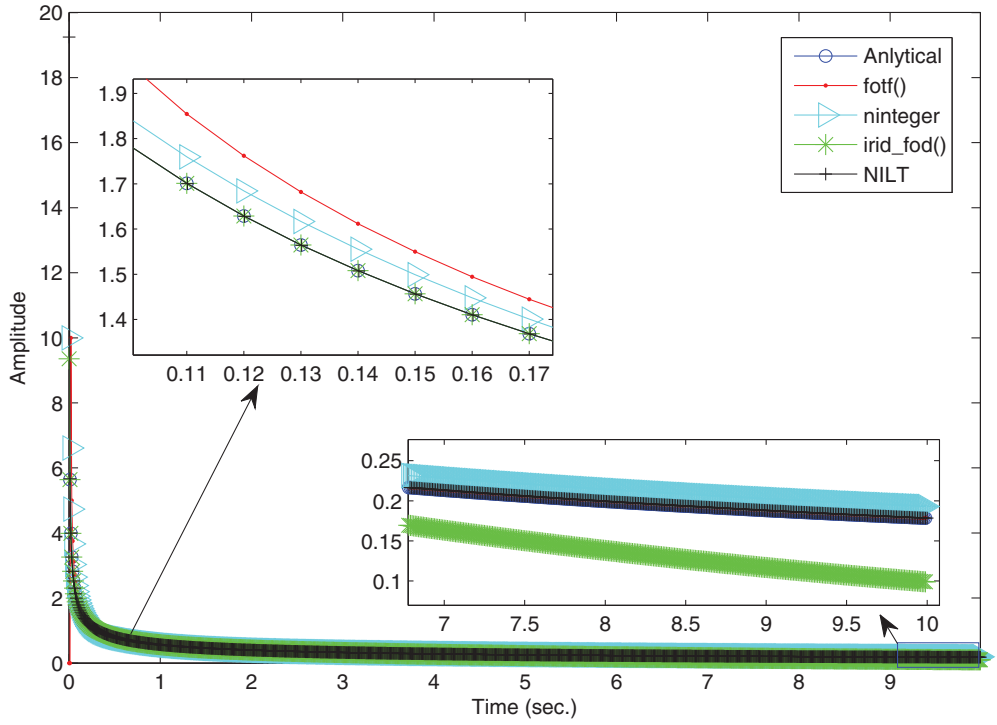
Criteria\ Methods	Analytical	fderiv()	glfdiff()	fourier_diffint()	FIT
Error 1	–	140.5000	1.8232	792.4660	0.0000
Elapsed T1	0.0001	1.4028	0.0029	0.0874	0.0209
Error 2	–	339.7973	2.1208	250.5433	0.0743
Elapsed T2	0.0001	1.4105	0.0029	0.0893	0.0201

3.3 Comparison III

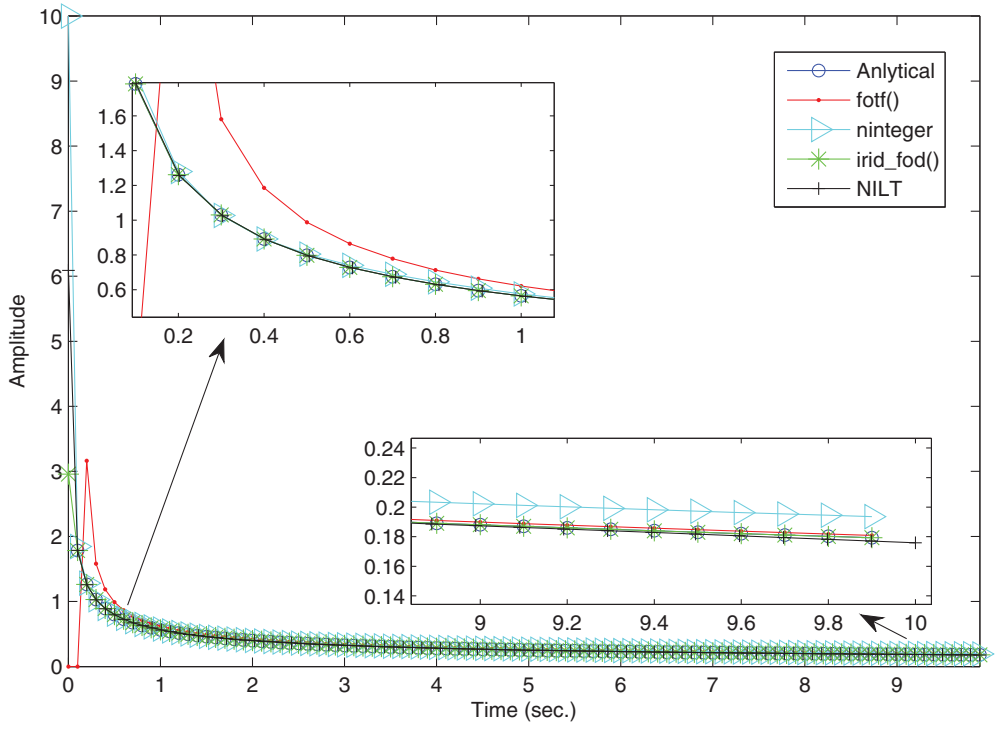
Although the simulation of FO pseudo-SS models can be achieved indirectly, some toolboxes do provide the direct simulation capability, such as the CRONE toolbox and FSST. Since the function frac_ss in CRONE toolbox only adopts the input of commensurate order systems, for comparison purposes, the following commensurate order pseudo-SS model is selected:

$$ \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^{(0.7)} = \begin{bmatrix} 0 & 1 \\ -0.1 & -0.2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u $$
$$ y = \begin{bmatrix} 0.1 & 0.3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{11} $$

To involve more tools into comparison, the FO integrator blocks in the FOTF and Ninteger toolboxes are used to represent the above fractional differential equations in Simulink, as shown in Figure 13. The comparison of the unit step responses computed by the four toolboxes is plotted in Figure 14, from which it can be seen that the result obtained using FSST (1 sec for step size) has bigger difference from the others. However, since analytical



(a) Ts=0.01



(b) Ts=0.1

Figure 8. Comparison of the impulse responses of the half-order integrator.

solution is not easy to obtain, it is insufficient to claim which method gives the highest accuracy. Hence, quantitative comparison is not provided. As an alternative, users can transform the above FO SS model to an FO transfer function model, assuming zero initial conditions,

$$G(s) = C(s^{\alpha}I - A)^{-1}B = \frac{3s^{0.7} + 1}{10s^{1.4} + 2s^{0.7} + 1}. \quad (12)$$

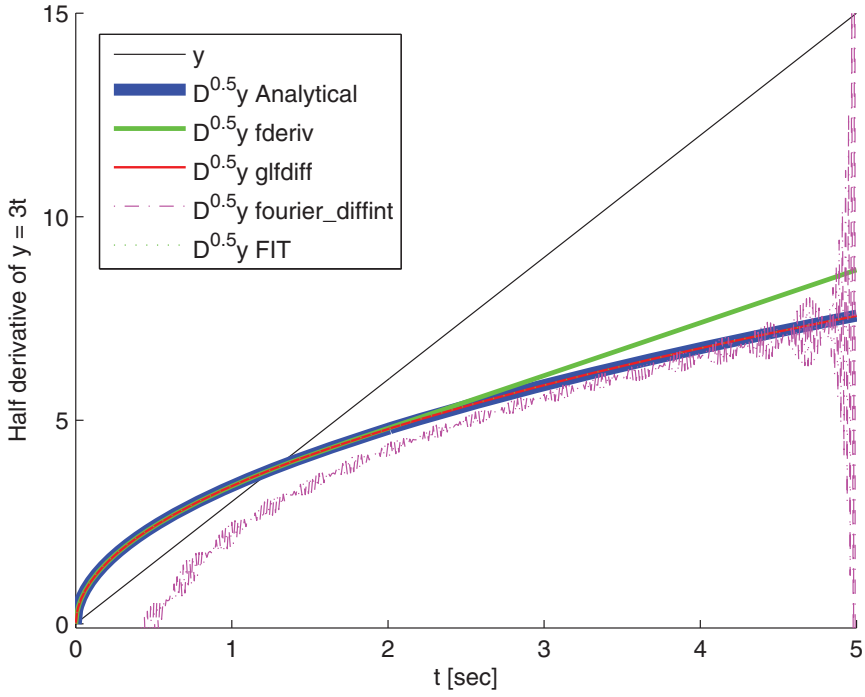


Figure 9. Comparison of half derivative of function $y(t) = 3t$, using fderiv(), glfdiff(), fourier_diffint() and FIT, respectively.

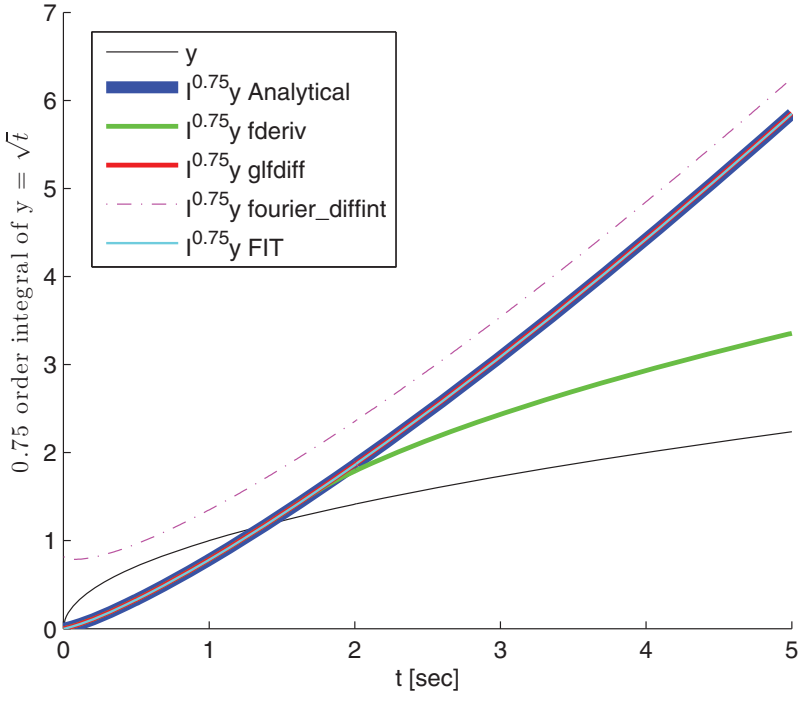


Figure 10. Comparison of 0.75th order integration of function $y(t) = \sqrt{t}$, using analytical solution, fderiv(), glfdiff(), fourier_diffint() and FIT, respectively.

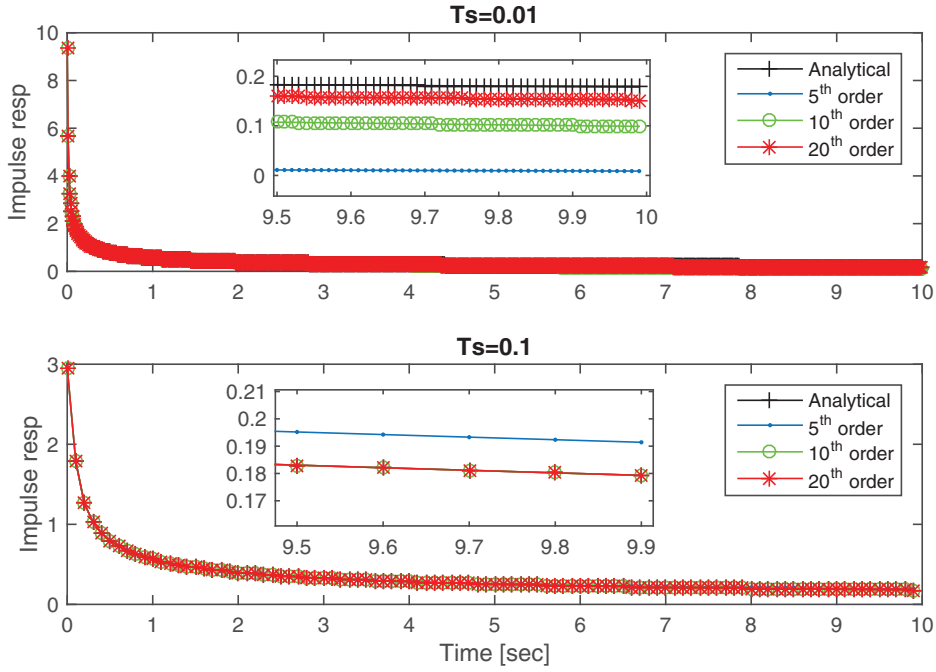


Figure 11. The T_s and order impact on irid_fod().

Thus, the NILT scripts can be used to compute the numerical solution, which has relatively higher reliability according to the authors' observation.

4. Comments for selection

A tricky part for the simulation of FO systems is that even if the system is broken down to the bottom layer, i.e. the analytical solution, it usually still involves the computation of M–L functions, which still needs to rely on the numerical tools or scripts. From the comparison, it can be seen that in the category of integrating/differentiating a function, glfdiff and FIT outperform other tools in terms of accuracy; in the category of control system simulation, NILT always provides higher accuracy. However, other toolboxes have some advantages, for example, ninteger and CRONE toolbox provide integrator blocks in Simulink, which makes the simulation of nonlinear systems possible.

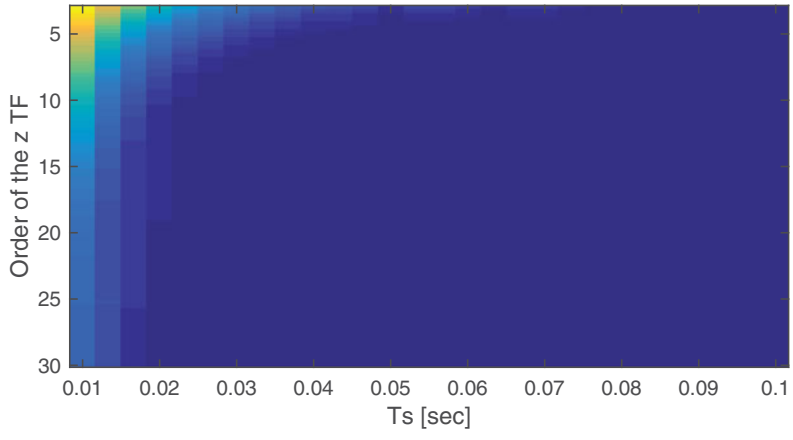


Figure 12. The heat map of approximation error of irid_fod() versus the order and sampling time.

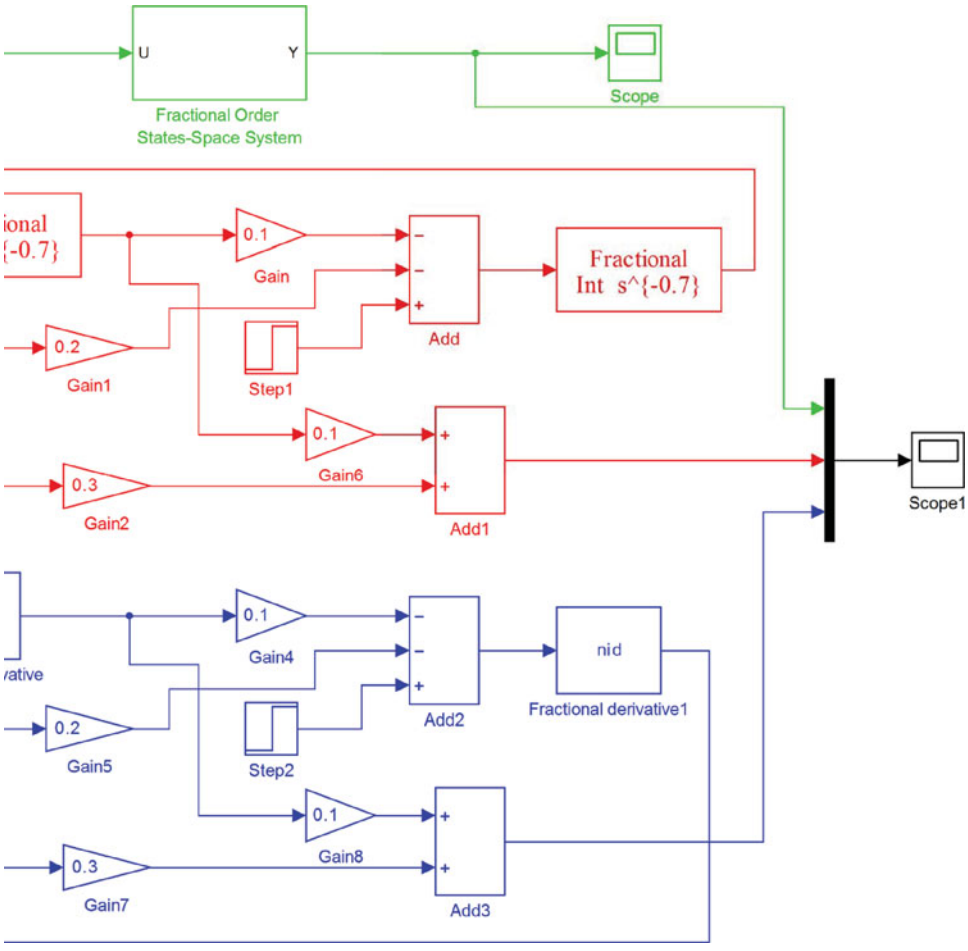


Figure 13. The Simulink block diagrams for simulating the FO pseudo-state space model in Equation (11).

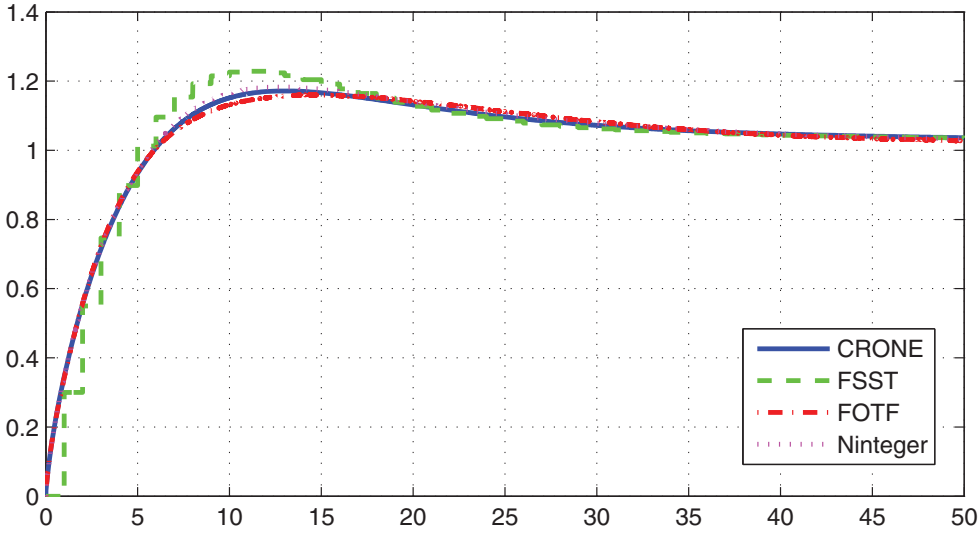


Figure 14. Comparison of the simulation results of the FO SS model obtained with different toolboxes.

5. Conclusion

In this paper, a comprehensive review of the Matlab-based numerical tools for FC and FO controls is presented. Quantitative evaluation of the selected tools is conducted. The summarised description and numerical comparison are designated to serve as a reference and guidance for readers when selecting tools for specific applications.

Disclosure statement

No potential conflict of interest was reported by the authors.

ORCID

YangQuan Chen http://orcid.org/0000-0002-7422-5988

References

Apkarian, P., & Noll, D. (2006). Nonsmooth H_∞ synthesis. *IEEE Transactions on Automatic Control, 51*(1), 71–86.

Axtell, M., & Bise, M.E. (1990). *Fractional calculus applications in control systems*. National aerospace and electronic conference (pp. 563–566), New York, NY.

Bagley, R.L., & Calico, R.A. (1989). Fractional order state equations for the control of viscoelastically damped structure. *Journal of Guidance, 14*(2), 304–310.

Barbosa, R., & Machado, J.A.T. (2006). Implementation of discrete-time fractional-order controllers based on LS approximations. *Acta Polytechnica Hung, 3*(4), 5–22.

Barrowes, B. (2005). Generalized hypergeometric function [Online] (Matlab Central). Retrieved from http://www.mathworks.com/matlabcentral/fileexchange/5616

Bayat, F.M. (2007). Fractional differentiator [Online] (Matlab Central). Retrieved from http://www.mathworks.com/matlabcentral/fileexchange/13858

Bayat, F.M. (2008). Root-locus plot of fractional order systems [Online] (Matlab Central). Retrieved from http://www.mathworks.com/matlabcentral/fileexchange/20577

Bode, H. (1945). *Network analysis and feedback amplifier design*. New York, NY: D. Van Nostrand Company.

Bohannan, G.W. (2008). Analog fractional order controller in temperature and motor control applications. *Journal of Vibration and Control, 14*(9), 1487–1498.

Branc˘ík, L. (1999). *An improved numerical inversion of two-dimensional Laplace transforms with application to transient analysis of transmission lines*. Proceedings EDS'99 Brno, Czech Republic.

Branc˘ík, L. (2001). *Utilization of quotient-difference algorithm in FFT-based numerical ILT method*. Proceedings of the 11th international Czech-Slovak scientific conference Radioelektronika, Czech Republic.

Caponetto, R., Dongola, G., Fortuna, L., & Petráš, I. (2010). *Fractional order systems: Modelling and control applications*. Hackensack, NJ: World Scientific.

Chaurasia, V., & Pandey, S. (2010). On the fractional calculus of generalized Mittag–Leffler function. *SCIENTIA Series A: Mathematical Sciences, 20*, 113–122.

Chen, D., Chen, Y.Q., & Xue, D. (2011). Digital fractional order Savitzky–Golay differentiator. *IEEE Transactions on Circuits and Systems – II: Express Briefs, 58*(11), 758–762.

Chen, Y.Q. (2003). Oustaloup-recursive-approximation for fractional order differentiators [Online] (Matlab Central). Retrieved from http://www.mathworks.com/ matlabcentral/fileexchange/3802

Chen, Y.Q. (2008a). Generalized Mittag–Leffler function [Online] (Matlab Central). Retrieved from http://www.mathworks.com/matlabcentral/fileexchange/20849

Chen, Y.Q. (2008b). Impulse response invariant discretization of fractional order integrators/differentiators [Online] (Matlab Central). Retrieved from http://www.mathworks.com/matlabcentral/fileexchange/21342

Chen, Y.Q., Petráš, I., & Xue, D. (2009, June). *Fractional order control – a tutorial*. American control conference, 2009. ACC'09 (pp. 1397–1411), St. Louis, MO.

Chen, Y.Q., & Vinagre, B.M. (2003). A new IIR-type digital fractional order differentiator. *Signal Processing, 83*, 2359–2365.

Corless, R.M., & Jeffrey, D.J. (1998). Graphing elementary Riemann surface. *SIGSAM Bulletin, 32*(1), 11–17.

Das, S., & Pan, I. (2012). *Fractional order signal processing: Introductory concepts and applications*. Berlin: Springer.

de Hoog, F., Knight, J., & Stokes, A.N. (1982). An improved method for numerical inversion of Laplace transforms. *SIAM Journal on Scientific Computing, 3*(3), 357–366.

Ding, Z., Granger, C.W., & Engle, R.F. (1993). A long memory property of stock market returns and a new model. *Journal of Empirical Finance, 1*, 83–106.

Dzieliński, A., & Sierociuk, D. (2008). *Simulation and experimental tools for fractional order control education*. Proceedings IFAC World Congress (pp. 11 654–11 659), Seoul, Korea.

Farkas, H.M., & Kra, I. (1980). *Riemann surfaces* (2nd ed.). New York, NY: Springer-Verlag.

Garrappa, R. (2014). The Mittag–Leffler function [Online] (Matlab Central). Retrieved from http://www.mathworks.com/matlabcentral/fileexchange/48154

Germano, G. (2008). Mittag–Leffler random number generator [Online] (Matlab Central). Retrieved from http://www.mathworks.com/matlabcentral/fileexchange/19392

Guzman, J.L., Astrom, K.J., Dormido, S., Hagglund, T., Berenguel, M., & Piguet, Y. (2008). Interactive modules for PID control. *EEE Control Systems Magazine I, 28*(5), 118–134.

Huntley, J. (2012). Generation of random variates [Online] (Matlab Central). Retrieved from http://www.mathworks.com/matlabcentral/fileexchange/35008

Jiang, C., Hartley, T.T., Carletta, J., & Veillette, R.J. (2013). A systematic approach for implementing fractional-order operators and systems. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems, 3*(3), 301–312.

Jonathan. (2014). Fractional derivative [Online] (Matlab Central). Retrieved from http://www.mathworks.com/matlabcentral/fileexchange/45982

Juraj. (2011). Numerical inversion of Laplace transforms in Matlab [Online] (Matlab Central). Retrieved from http://www.mathworks.com/matlabcentral/fileexchange/32824

Lachhab, N., Svaricek, F., Wobbe, F., & Rabba, H. (2013). *Fractional order PID controller (FOPID) – toolbox*. 2013 European control conference (ECC) (pp. 3694–3699), Zuich, Switzerland.

Lewis, T.G. (2014). *Book of extremes: Why the 21st century isn't like the 20th century*. Switzerland: Springer International Publishing.

Li, H., Luo, Y., & Chen, Y.Q. (2010). A fractional order proportional and derivative (FOPD) motion controller: Tuning rule and experiments. *IEEE Transactions on Control Systems Technology, 18*(2), 516–520.

Li, Y., Sheng, H., & Chen, Y.Q. (2010, July). *Impulse response invariant discretization of a generalized commensurate fractional order filter*. Proceedings of the 8th World Congress on Intelligent Control and Automation, Jinan, China.

Li, Z. (2015). Fractional order root locus [online] (Matlab Central). Retrieved from http://www.mathworks.com/matlabcentral/fileexchange/50458

Li, Z., & Chen, Y.Q. (2014). *Identification of linear fractional order systems using the relay feedback approach*. 2014 American control conference (ACC), Portland, OR.

Liang, J. (2005). *Control of linear time-invariant distributed parameter systems: From integer order to fractional order* (Doctoral dissertation). Logan, UT: Utah State University.

Lorenzo, C.F., & Hartley, T.T. (2002). Variable order and distributed order fractional operators. *Nonlinear Dynamics, 29*(1), 57–98.

Luo, Y., & Chen, Y.Q. (2013). *Fractional order motion controls*. West Sussex: John Wiley & Sons.

Machado, J.A.T. (2011). Root locus of fractional linear systems. *Communications in Nonlinear Science and Numerical Simulation, 16*, 3855–3862.

Machado, J.T., Kiryakova, V., & Mainardi, F. (2011). Recent history of fractional calculus. *Communications in Nonlinear Science and Numerical Simulation, 16*, 1140–1153.

Magin, R.L. (2006). *Fractional calculus in bioengineering*. Redding, CA: Begell House.

Malkiel, B.G. (1999). *A random walk down wall street* (7th ed.). New York, NY: W.W. Norton & Company.

Marinov, T.M., Ramirez, N., & Santamaria, F. (2013a). Fractional integration toolbox. *Fractional Calculus & Applied Analysis, 16*(3), 670–681.

Marinov, T.M., Ramirez, N., & Santamaria, F. (2013b). Fractional integration toolbox (FIT) [Online] (Santamaria Lab). Retrieved from http://www.cbi.utsa.edu/FIT

Miller, K.S., & Ross, B. (1993). *An introduction to the fractional calculus and fractional differential equations* (1st ed.). New York, NY: Wiley-Interscience.

Monje, C.A., Chen, Y.Q., Vinagre, B.M., Xue, D., & Feliu, V. (2006). *Fractional order systems and controls: Fundamentals and applications*. London: Springer-Verlag.

Mukhopadhyay, S. (2008). Mittag-leffler function, M-file, cmex DLL, and S-function [Online] (Matlab Central). Retrieved from http://www.mathworks.com/matlabcentral/ fileexchange/20731

Narayanana, V.A., & Prabhu, K. (2003). The fractional fourier transform: Theory, implementation and error analysis. *Microprocessors and Microsystems, 27*, 511–521.

Oustaloup, A., Levron, F., Mathieu, B., & Nanot, F.M. (2000). Frequency-band complex noninteger differentiator: Characterization and synthesis. *IEEE Transactions on Circuits and Systems – I: Fundamental Theory and Applications, 47*(1), 25–39.

Oustaloup, A., Melchior, P., Lanusse, P., Cois, O., & Dancla, F. (2000). The CRONE toolbox for matlab. In *Computer-aided control system design* (pp. 190–195), Anchorage, AK. doi:10.1109/CACSD.2000.900210

Ozaktas, H.M., Zalevsky, Z., & Kutay, M.A. (2001). *The fractional Fourier transform*. Hoboken, NJ: John Wiley & Sons.

Papazafeiropoulos, G. (2014). Fractional differentiation and integration [Online] (Matlab Central). http://www.mathworks.com/matlabcentral/fileexchange/45877

Petráš, I. (2003a). Digital fractional order differentiator/ integrator – IIR type [Online] (Matlab Central). Retrieved from http://www.mathworks.com/matlabcentral/fileexchange/3672

Petráš, I. (2003b). Digital fractional order differentiator/integrator – FIR type [Online] (Matlab Central). Retrieved from http://www.mathworks.com/ matlabcentral/fileexchange/3673,

Petráš, I. (2011a). Digital fractional order differentiator/integrator – new IIR type [Online] (Matlab Central). http://www.mathworks.com/matlabcentral/fileexchange/31358

Petráš, I. (2011b). Discrete fractional-order PID controller [Online] (Matlab Central). Retrieved from http://www.mathworks.com/matlabcentral/fileexchange/33761

Petráš, I. (2011c). Fractional derivatives, fractional integrals, and fractional differential equations in Matlab. In *Engineering education and research using MATLAB* (pp. 239–264). Winchester: InTech.

Petráš, I. (2011d). *Fractional-order nonlinear systems: Modeling, analysis and simulation*. Berlin: Springer Science & Business Media.

Pisoni, E., Visioli, A., & Dormido, S. (2009). *An interactive tool for fractional order PID controllers*. IECON '09. 35th annual conference of IEEE. Porto, Portugal.

Podlubny, I. (1999a). *Fractional differential equations*. Waltham, MA: Academic Press.

Podlubny, I. (1999b). Fractional-order systems and $PI^{\lambda}D^{\mu}$ controllers. *IEEE Transactions on Automatic Control, 44*(1), 208–214.

Podlubny, I. (2000). Matrix approach to discrete fractional calculus. *Factional Calculus and Applied Analysis, 29*(4), 281–296.

Podlubny, I. (2005). Mittag–Leffler function [Online] (Matlab Central). Retrieved from http://www.mathworks.com/matlabcentral/fileexchange/8738

Podlubny, I. (2008). Matrix approach to discretization of ODEs and PDEs of arbitrary real order [Online] (Matlab Central). Retrieved from http://www.mathworks.com/matlabcentral/fileexchange/22071

Prabhakar, T. (1971). A singular integral equation with a generalised Mittag–Leffler function in the kernel. *Yokohama Mathematical Journal, 19*, 7–15.

Sabatier, J., Agrawal, O., & Machado, J.T. (Eds.). (2007). *Advances in fractional calculus: Theoretical developments and applications in physics and engineering*. Berlin: Springer.

Sheng, H., Li, Y., & Chen, Y.Q. (2011). Application of numerical inverse laplace transform algorithms in fractional calculus. *Journal The Franklin Institute, 348*, 315–330.

Sierociuk, D. (2003). Fractional states-space toolkit (FSST). [Online] http://www.ee.pw.edu.pl/ dsieroci/fsst/fsst.htm

Sierociuk, D. (2012). Fractional variable order derivative Simulink toolkit [Online] (Matlab Central). Retrieved from http://www.mathworks.com/matlabcentral/fileexchange/38801

Tepljakov, A. (2011). *Fractional-order calculus based identification and control of linear dynamic systems* (Master's thesis). Tallinn: Tallinn University of Technology.

Tepljakov, A., Petlenkov, E., & Belikov, J. (2011). FOMCON: Fractional-order modeling and control toolbox for MATLAB. 18th international conference on "Mixed Design of Integrated Circuits and Systems", Gliwice, Poland.

The CRONE Team (2014, February). The CRONE toolbox homepage [Internet]. Retrieved from http://www.ims-bordeaux.fr/CRONE/toolbox

Tricaud, C. (2009). Solution of fractional optimal control problems [Online] (Matlab Central). Retrieved from http://www.mathworks.com/matlabcentral/fileexchange/22196

Tricaud, C., & Chen, Y.Q. (2008). *Solving fractional order optimal control problems in RIOTS_95 – a general-purpose optimal control problem solver*. Proceedings of the 3rd IFAC workshop on fractional differentiation and its applications, Ankara, Turkey.

Tricaud, C., & Chen, Y.Q. (2009). *Solution of fractional order optimal control problems using SVD-based rational approximations*. The 2009 American control conference (ACC) (pp. 1430–1435), St. Louis, MO.

Valério, D. (2009). Variable order derivatives [Online] (Matlab Central). Retrieved from http://www.mathworks.com/matlabcentral/fileexchange/24444

Valério, D., & da Costa, J.S. (2004). Ninteger: A non-integer control toolbox for Matlab. 1st IFAC workshop on fractional differentiation and applications (pp. 208–213), Bordeaux, France.

Valério, D., & da Costa, J.S. (2005, September). Time-domain implementation of fractional order controllers. *IEE Proceedings – Control Theory and Applications, 152*(5), 539–552.

Valsa, J., & Brancˇík, L. (1998). Fractional order state equations for the control of viscoelastically damped structure. *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields, 11*(3), 153–166.

West, B.J. (2006). *Where medicine went wrong*. Singapore: World Scientific.

West, B.J., Turalska, M., & Grigolini, P. (2014). *Complex networks: From social crises to neuronal avalanches* (pp. 509–524). Hoboken, NJ: Wiley-VCH Verlag GmbH.

Xue, D., & Chen, Y.Q. (2014a). *Modeling, analysis and design of control systems in Matlab and Simulink*. Singapore: World Scientific.

Xue, D., & Chen, Y.Q. (2014b). *System simulation techniques with Matlab and Simulink*. Singapore: John Wiley & Sons.

Xue, D., Chen, Y.Q., & Atherton, D. (2009). *Linear feedback control – analysis and design with Matlab 6.5*. Hoboken, NJ: Society for Industrial and Applied Mathematics.

Yin, C., Chen, Y.Q., & Zhong, S.M. (2014). Fractional-order sliding mode based extremum seeking control of a class of nonlinear systems. *Automatica, 50*, 3173–3181.

Yousfi, N., Melchior, P., Rekik, C., Derbel, N., & Oustaloup, A. (2012). Design of centralized CRONE controller combined with MIMO-QFT approach applied to non-square multivariable systems. *International Journal of Computer Applications, 45*, 6–14.

Zhao, T., Chen, Y.Q., & Li, Z. (2014). *Fractional order nonlinear model predictive control using RIOTS_95*. The international conference on fractional differentiation and its applications (ICFDA), Catania, Italy.