

# Fractional Derivatives, Fractional Integrals, and Fractional Differential Equations in Matlab

Ivo Petráš

*Technical University of Košice  
Slovak Republic*

## 1. Introduction

The term fractional calculus is more than 300 years old. It is a generalization of the ordinary differentiation and integration to non-integer (arbitrary) order. The subject is as old as the calculus of differentiation and goes back to times when Leibniz, Gauss, and Newton invented this kind of calculation. In a letter to L'Hospital in 1695 Leibniz raised the following question (Miller and Ross, 1993): "Can the meaning of derivatives with integer order be generalized to derivatives with non-integer orders?" The story goes that L'Hospital was somewhat curious about that question and replied by another question to Leibniz. "What if the order will be  $1/2$ ?" Leibniz in a letter dated September 30, 1695 replied: "It will lead to a paradox, from which one day useful consequences will be drawn." The question raised by Leibniz for a fractional derivative was an ongoing topic in the last 300 years. Several mathematicians contributed to this subject over the years. People like Liouville, Riemann, and Weyl made major contributions to the theory of fractional calculus. The story of the fractional calculus continued with contributions from Fourier, Abel, Leibniz, Grünwald, and Letnikov.

Nowadays, the fractional calculus attracts many scientists and engineers. There are several applications of this mathematical phenomenon in mechanics, physics, chemistry, control theory and so on (Caponetto et al., 2010; Magin, 2006; Monje et al., 2010; Oldham and Spanier, 1974; Oustaloup, 1995; Podlubny, 1999). It is natural that many authors tried to solve the fractional derivatives, fractional integrals and fractional differential equations in Matlab. A few very good and interesting Matlab functions were already submitted to the MathWorks, Inc. Matlab Central File Exchange, where they are freely downloadable for sharing among the users. In this chapter we will use some of them. It is worth mentioning some addition to Matlab toolboxes, which are appropriate for the solution of fractional calculus problems. One of them is a toolbox created by CRONE team (CRONE, 2010) and another one is the Fractional State-Space Toolkit developed by Dominik Sierociuk (Sierociuk, 2005). Last but not least we should also mention a Matlab toolbox created by Dingyü Xue (Xue, 2010), which is based on Matlab object for fractional-order transfer function and some manipulation with this class of the transfer function. Despite that the mentioned toolboxes are mainly for control systems, they can be "abused" for solutions of general problems related to fractional calculus as well.

## 2. Fractional calculus fundamentals

### 2.1 Special functions

Here we should mention the most important function used in fractional calculus - Euler's *gamma* function, which is defined as

$$\Gamma(n) = \int_0^{\infty} t^{n-1} e^{-t} dt. \quad (1)$$

This function is a generalization of the factorial in the following form:

$$\Gamma(n) = (n-1)! \quad (2)$$

This gamma function is directly implemented in the Matlab with syntax `[] = gamma()`.

Another function, which plays a very important role in the fractional calculus, was in fact introduced in 1953. It is a two-parameter function of the *Mittag-Leffler* type defined as (Podlubny, 1999):

$$E_{\alpha,\beta}(z) = \sum_{k=0}^{\infty} \frac{z^k}{\Gamma(\alpha k + \beta)}, \quad (\alpha > 0, \beta > 0). \quad (3)$$

The Mittag-Leffler function (3) can be expressed in the integral representation as

$$E_{\alpha,\beta}(z) = \frac{1}{2\pi i} \int_C \frac{t^{\alpha-\beta} e^t}{t^{\alpha} - z} dt, \quad (4)$$

the contour  $C$  starts and ends at  $-\infty$  and circles around the singularities and branch points of the integrand.

There are some relationships (given e.g. in (Djrbashian, 1993; Podlubny, 1999)):

$$\begin{aligned} E_{1,1}(z) &= e^z, & E_{1/2,1}(\sqrt{z}) &= \frac{2}{\sqrt{\pi}} e^{-z} \operatorname{erfc}(-\sqrt{z}), \\ E_{2,1}(z) &= \cosh(\sqrt{z}), & E_{2,1}(-z^2) &= \cos(z), & E_{1,2}(z) &= \frac{e^z - 1}{z}. \end{aligned}$$

For  $\beta = 1$  we obtain the Mittag-Leffler function in one parameter (Podlubny, 1999):

$$E_{\alpha,1}(z) = \sum_{k=0}^{\infty} \frac{z^k}{\Gamma(\alpha k + 1)} \equiv E_{\alpha}(z). \quad (5)$$

For the numerical evaluation of the Mittag-Leffler function (3) with the default accuracy set to  $10^{-6}$  the Matlab routine `[e]=mlf(alf,bet,c,fi)`, written by Podlubny and Kacenak (2005) can be used. Another Matlab function `f=gml_fun(a,b,c,x,eps0)` for a generalized Mittag-Leffler function was created by YangQuan Chen (Chen, 2008).

In Fig. 1(a) and Fig. 1(b) are plotted the well-known functions ( $e^z$  and  $\cos(z)$ ) computed via the Matlab routine `[] = mlf()` created for the evaluation of the Mittag-Leffler function.

Another important function  $\mathcal{E}_k(t, \lambda; \mu, \nu)$  of the Mittag-Leffler type was introduced by (Podlubny, 1999). The function is defined by

$$\mathcal{E}_k(t, \lambda; \mu, \nu) = t^{\mu k + \nu - 1} E_{\mu, \nu}^{(k)}(\lambda t^{\mu}), \quad (k = 0, 1, 2, \dots), \quad (6)$$

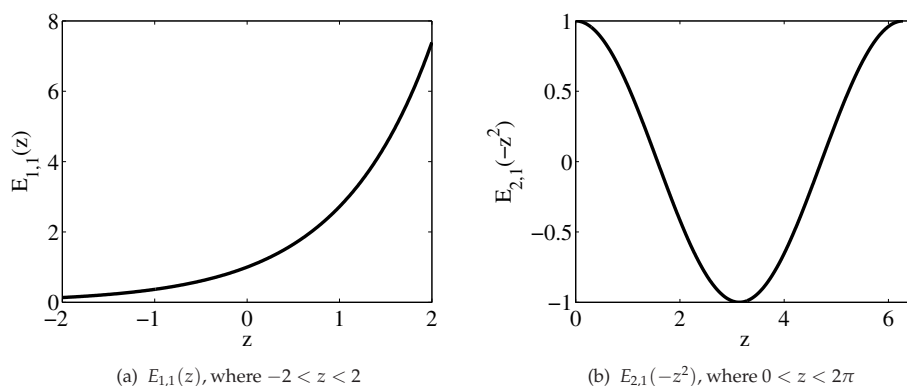


Fig. 1. Mittag-Leffler function (3) for various parameters

where  $E_{\mu,\nu}^{(k)}(z)$  is  $k$ -th derivative of the Mittag-Leffler function of two parameters given by

$$E_{\mu,\nu}^{(k)}(z) = \sum_{i=0}^{\infty} \frac{(i+k)! z^i}{i! \Gamma(\mu i + \mu k + \nu)}, \quad (k = 0, 1, 2, \dots). \quad (7)$$

## 2.2 Fractional derivatives and integrals

Fractional calculus is a generalization of integration and differentiation to non-integer-order fundamental operator  ${}_a D_t^\alpha$ , where  $a$  and  $t$  are the bounds of the operation and  $\alpha \in \mathbb{R}$ . The continuous integrodifferential operator is defined as

$${}_a D_t^\alpha = \begin{cases} \frac{d^\alpha}{dt^\alpha} & : \alpha > 0, \\ 1 & : \alpha = 0, \\ \int_a^t (d\tau)^\alpha & : \alpha < 0. \end{cases}$$

The three most frequently used definitions for the general fractional differintegral are: the Grünwald-Letnikov (GL) definition, the Riemann-Liouville (RL) and the Caputo definition (Miller and Ross, 1993; Oldham and Spanier, 1974; Podlubny, 1999). Other definitions are connected with well-known names as for instance Weyl, Fourier, Cauchy, Abel, etc.

The GL definition is given as

$${}_a D_t^\alpha f(t) = \lim_{h \rightarrow 0} h^{-\alpha} \sum_{j=0}^{\lfloor \frac{t-a}{h} \rfloor} (-1)^j \binom{\alpha}{j} f(t-jh), \quad (8)$$

where  $\lfloor \cdot \rfloor$  means the integer part. The RL definition is given as

$${}_a D_t^\alpha f(t) = \frac{1}{\Gamma(n-\alpha)} \frac{d^n}{dt^n} \int_a^t \frac{f(\tau)}{(t-\tau)^{\alpha-n+1}} d\tau, \quad \text{for } (n-1 < \alpha < n), \quad (9)$$

where  $\Gamma(\cdot)$  is the *gamma* function. The Caputo definition of fractional derivatives can be written as

$${}_a D_t^\alpha f(t) = \frac{1}{\Gamma(n-\alpha)} \int_a^t \frac{f^{(n)}(\tau)}{(t-\tau)^{\alpha-n+1}} d\tau, \quad \text{for } (n-1 < \alpha < n). \quad (10)$$

In this chapter we will consider mainly the GL, the RL, and the Caputo definitions. This consideration is based on the fact that, for a wide class of functions, the three best known definitions - GL, RL, and Caputo - are equivalent under some conditions (Podlubny, 1999).

As it was already mentioned, under the homogeneous initial conditions the Riemann-Liouville and the Caputo derivative are equivalent. Let us denote the Riemann-Liouville fractional derivative as  ${}_a^{RL} D_t^\alpha f(t)$  and the Caputo definition as  ${}_a^C D_t^\alpha f(t)$ , then the relationship between them is:

$${}_a^{RL} D_t^\alpha f(t) = {}_a^C D_t^\alpha f(t) + \sum_{k=0}^{n-1} \frac{(t-a)^{k-\alpha}}{\Gamma(k-\alpha+1)} f^{(k)}(a),$$

for  $f^{(k)}(a) = 0$  ( $k = 0, 1, \dots, n-1$ ).

The initial conditions for the fractional-order differential equations with the Caputo derivatives are in the same form as for the integer-order differential equations. It is an advantage because applied problems require definitions of fractional derivatives, where there are clear interpretations of initial conditions, which contain  $f(a)$ ,  $f'(a)$ ,  $f''(a)$ , etc.

In addition, for approximation purpose we consider the Laplace transform method of the fractional differintegral  ${}_a D_t^\alpha$ . The Laplace transform of the join operator of order  $\alpha$  for the GL, RL and Caputo definitions, under the zero initial conditions is:

$$\mathcal{L}\{ {}_0 D_t^\alpha f(t); s \} = s^\alpha F(s). \quad (11)$$

Laplace transform technique is considered as an efficient way in solving differential equations with integer order and fractional order as well. For differential equations with fractional order, the Laplace transform technique works effectively only for relatively simple equations, because of the difficulties of calculating inversion of Laplace transforms. This problem was solved by applying a numerical inverse Laplace transform algorithms in fractional calculus (Sheng et al., 2011). Three numerical inverse Laplace transform algorithms in Matlab, named `invlap()`, `gavsteh()`, and `nilt()`, were described there and tested.

The Laplace transforms for several known Mittag-Leffler type functions are summarized as follows (Magin, 2006; Podlubny, 1999):

$$\begin{aligned} \mathcal{L}\{E_\alpha(-\lambda t^\alpha)\} &= \frac{s^{\alpha-1}}{s^\alpha + \lambda}, & \mathcal{L}\{t^{\alpha-1} E_{\alpha,\alpha}(-\lambda t^\alpha)\} &= \frac{1}{s^\alpha + \lambda}, \\ \mathcal{L}\{t^{\beta-1} E_{\alpha,\beta}(-\lambda t^\alpha)\} &= \frac{s^{\alpha-\beta}}{s^\alpha + \lambda}, & \mathcal{L}\{\mathcal{E}_k(t, \pm\lambda; \alpha, \beta)\} &= \frac{k! s^{\alpha-\beta}}{(s^\alpha \mp \lambda)^{k+1}}. \end{aligned} \quad (12)$$

### 2.3 Numerical methods for fractional calculus

There are many numerical methods appropriate for the fractional calculus computation. They were described in several works and a good survey can be found in (Vinagre et al., 2000; 2003). In the above-mentioned papers are described the time domain and frequency methods in

the form of IIR and FIR approximations together with illustrative examples. Some of the mentioned frequency methods in both forms of approximation have been realized as the Matlab routines in Duarte Valerio's toolbox called *ninteger* (see detail review in (Valério, 2005)). Also created in this toolbox was a Simulink block *nid* for fractional derivative and integral, where the order of derivative/integral and method of its approximation can be selected.

### 2.3.1 Grünwald-Letnikov method

For numerical calculation of fractional-order derivatives we can use the relation (13) derived from the GL definition (8). This approach is based on the fact that for a wide class of functions the three definitions - GL (8), RL (9), and Caputo's (10) - are equivalent if  $f(a) = 0$ . The relation for the explicit numerical approximation of  $q$ -th derivative at the points  $kh$ , ( $k = 1, 2, \dots$ ) has the following form (Dorčák, 1994; Podlubny, 1999):

$${}_{(k-L_m/h)}D_{t_k}^q f(t) \approx h^{-q} \sum_{j=0}^k (-1)^j \binom{q}{j} f(t_{k-j}) = h^{-q} \sum_{j=0}^k c_j^{(q)} f(t_{k-j}), \quad (13)$$

where  $L_m$  is the "memory length",  $t_k = kh$ ,  $h$  is the time step of calculation and  $c_j^{(q)}$  ( $j = 0, 1, \dots, k$ ) are binomial coefficients. For their calculation we can use for instance the following expression (Dorčák, 1994):

$$c_0^{(q)} = 1, \quad c_j^{(q)} = \left(1 - \frac{1+q}{j}\right) c_{j-1}^{(q)}. \quad (14)$$

The binomial coefficients  $c_j^{(q)}$  ( $j = 0, 1, \dots, k$ ) can be also expressed by using a factorial. Writing the factorial as a *gamma* function (2), it allows the binomial coefficient to be generalized to non-integer arguments. We can write the relations:

$$(-1)^j \binom{q}{j} = (-1)^j \frac{\Gamma(q+1)}{\Gamma(j+1)\Gamma(q-j+1)} = \frac{\Gamma(j-q)}{\Gamma(-q)\Gamma(j+1)}. \quad (15)$$

Obviously, for this simplification we pay a penalty in the form of some inaccuracy. If  $f(t) \leq M$ , we can easily establish the following estimate for determining the memory length  $L_m$ , providing the required accuracy  $\epsilon$ :

$$L_m \geq \left( \frac{M}{\epsilon |\Gamma(1-q)|} \right)^{1/q}. \quad (16)$$

The above-described numerical method is called Power Series Expansion (PSE) of a generating function. It is important to note that PSE leads to approximation in the form of polynomials, that is, the discretized fractional operator is in the form of FIR filter, which has only zeros (Chen and Moore, 2002; Vinagre et al., 2003).

The resulting discrete transfer function, approximating fractional-order operators, can be expressed in  $z$ -domain as follows:

$${}_0D_{kT}^{\pm r} G(z) = \frac{Y(z)}{F(z)} = \left(\frac{1}{T}\right)^{\pm r} \text{PSE} \left\{ \left(1 - z^{-1}\right)^{\pm r} \right\}_n \approx T^{\mp r} R_n(z^{-1}), \quad (17)$$

where  $T$  is the sample period,  $\text{PSE}\{u\}$  denotes the function resulting from applying the power series expansion to the function  $u$ ,  $Y(z)$  is the Z transform of the output sequence  $y(kT)$ ,  $F(z)$  is the Z transform of the input sequence  $f(kT)$ ,  $n$  is the order of the approximation, and  $R$  is polynomial of degree  $n$ , respectively, in the variable  $z^{-1}$ , and  $k = 1, 2, \dots$ . A Matlab routine `dfod2()` of this method (17) can be downloaded (see (Petráš, 2003b)).

EXAMPLE 2.1. Let us consider the fractional-order derivative  $\alpha \in [0, 1]$  of the function  $y = \sin(t)$  for  $t \in [0, 2\pi]$ . The following Matlab code using a function `fderiv()`, which was created by Bayat (2007) and is based on relation (13), can be used:

```
clear all; close all;
h=0.01; t=0:h:2*pi;
y=sin(t);
order=0:0.1:1;
for i=1:length(order)
    yd(i,:)=fderiv(order(i),y,h);
end
[X, Y]=meshgrid(t,order);
mesh(X,Y,yd)
xlabel('t'); ylabel('\alpha'); zlabel('y')
```

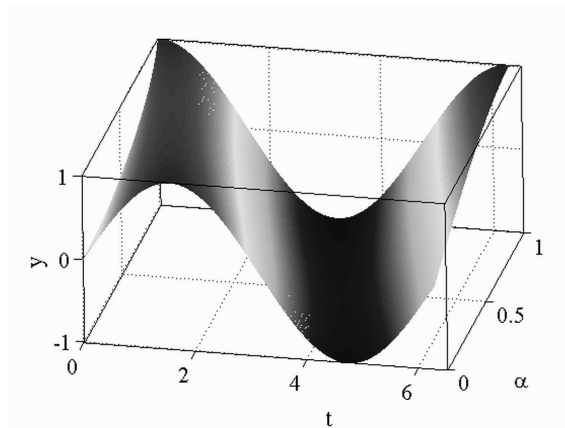


Fig. 2. Fractional derivatives of function  $y = \sin(t)$

In Fig. 2 is depicted the fractional derivative of the sine function for fractional order of derivative  $0 < \alpha < 1$  and  $0 < t < 2\pi$ .

### 2.3.2 Continuous and discrete-time approximation techniques

Other approach can be realized by Continued Fraction Expansion (CFE) of the generating function and then the approximated fractional operator is in the form of IIR filter, which has poles and zeros.

Taking into account that our aim is to obtain equivalents to the fractional integrodifferential operators in the Laplace domain,  $s^{\pm r}$ , the result of such approximation for an irrational

function,  $G(s)$ , can be expressed in the form (Vinagre et al., 2003):

$$\begin{aligned} G(s) &\simeq a_0(s) + \frac{b_1(s)}{a_1(s) + \frac{b_2(s)}{a_2(s) + \frac{b_3(s)}{a_3(s) + \dots}}} \\ &= a_0(s) + \frac{b_1(s)}{a_1(s) +} \frac{b_2(s)}{a_2(s) +} \frac{b_3(s)}{a_3(s) +} \dots \end{aligned} \quad (18)$$

where  $a_i s$  and  $b_i s$  are rational functions of the variable  $s$ , or are constants. The application of the method yields a rational function, which is an approximation of the irrational function  $G(s)$ .

In other words, for evaluation purposes, the rational approximations obtained by CFE frequently converge much more rapidly than the PSE and have a wider domain of convergence in the complex plane. On the other hand, the approximation by PSE and the short memory principle is convenient for the dynamical properties consideration.

For interpolation purposes, rational functions are sometimes superior to polynomials. This is, roughly speaking, due to their ability to model functions with poles. These techniques are based on the approximations of an irrational function,  $G(s)$ , by a rational function defined by the quotient of two polynomials in the variable  $s$  in frequency  $s$ -domain:

$$G(s) \simeq R_{i(i+1) \dots (i+m)} = \frac{P_\mu(s)}{Q_\nu(s)} = \frac{p_0 + p_1 s + \dots + p_\mu s^\mu}{q_0 + q_1 s + \dots + q_\nu s^\nu}, \quad (m+1 = \mu + \nu + 1) \quad (19)$$

passing through the points  $(s_i, G(s_i)), \dots, (s_{i+m}, G(s_{i+m}))$ .

The resulting discrete transfer function, approximating fractional-order operators, can be expressed as (Vinagre et al., 2003):

$${}_0 D_{kT}^{\pm r} G(z) = \frac{Y(z)}{F(z)} = \left(\frac{2}{T}\right)^{\pm r} \text{CFE} \left\{ \left(\frac{1-z^{-1}}{1+z^{-1}}\right)^{\pm r} \right\}_{p,n} \approx \left(\frac{2}{T}\right)^{\pm r} \frac{P_p(z^{-1})}{Q_n(z^{-1})}, \quad (20)$$

where  $T$  is the sample period,  $\text{CFE}\{u\}$  denotes the function resulting from applying the continued fraction expansion to the function  $u$ ,  $Y(z)$  is the Z transform of the output sequence  $y(kT)$ ,  $F(z)$  is the Z transform of the input sequence  $f(kT)$ ,  $p$  and  $n$  are the orders of the approximation, and  $P$  and  $Q$  are polynomials of degrees  $p$  and  $n$ , respectively, in the variable  $z^{-1}$ , and  $k = 1, 2, \dots$

In general, the discretization of fractional-order differentiator/integrator  $s^{\pm r}$  ( $r \in \mathbb{R}$ ) can be expressed by the *generating function*  $s \approx \omega(z^{-1})$ . This generating function and its expansion determine both the form of the approximation and the coefficients (Lubich, 1986).

In this section, for direct discretizing  $s^r$ , ( $0 < r < 1$ ), we will concentrate on the IIR form of discretization where as a generating function we will adopt an Al-Alaoui idea on mixed scheme of Euler and Tustin operators (Al-Alaoui, 1993; 1997), but we will use a different ratio between both operators. The mentioned new operator, raised to power  $\pm r$ , has the form (Petráš, 2003a):

$$(\omega(z^{-1}))^{\pm r} = \left( \frac{1+a}{T} \frac{1-z^{-1}}{1+az^{-1}} \right)^{\pm r}, \quad (21)$$

where  $a$  is the ratio term and  $r$  is the fractional order. The ratio term  $a$  is the amount of phase shift and this tuning knob is sufficient for most engineering problems being solved.

In expanding the above in rational functions, we will use the CFE. It should be pointed out that, for control applications, the obtained approximate discrete-time rational transfer function should be stable. Furthermore, for a better fit to the continuous frequency response, it would be of high interest to obtain discrete approximations with poles and zeros interlaced along the line  $z \in (-1, 1)$  of the  $z$  plane. The direct discretization approximations proposed in this paper enjoy the desired properties.

The result of such approximation for an irrational function,  $\hat{G}(z^{-1})$ , can be expressed by  $G(z^{-1})$  in the CFE form (Vinagre et al., 2000):

$$\begin{aligned} G(z^{-1}) &\simeq a_0(z^{-1}) + \frac{b_1(z^{-1})}{a_1(z^{-1}) + \frac{b_2(z^{-1})}{a_2(z^{-1}) + \frac{b_3(z^{-1})}{a_3(z^{-1}) + \dots}}} \\ &= a_0(z^{-1}) + \frac{b_1(z^{-1})}{a_1(z^{-1}) +} \frac{b_2(z^{-1})}{a_2(z^{-1}) +} \dots \frac{b_3(z^{-1})}{a_3(z^{-1}) +} \dots \end{aligned}$$

where  $a_i$  and  $b_i$  are either rational functions of the variable  $z^{-1}$  or constants. The application of the method yields a rational function,  $G(z^{-1})$ , which is an approximation of the irrational function  $\hat{G}(z^{-1})$ .

The resulting discrete transfer function, approximating fractional-order operators, can be expressed as:

$$\begin{aligned} (\omega(z^{-1}))^{\pm r} &\approx \left(\frac{1+a}{T}\right)^{\pm r} \text{CFE} \left\{ \left(\frac{1-z^{-1}}{1+az^{-1}}\right)^{\pm r} \right\}_{p,q} \\ &= \left(\frac{1+a}{T}\right)^{\pm r} \frac{P_p(z^{-1})}{Q_q(z^{-1})} = \left(\frac{1+a}{T}\right)^{\pm r} \frac{p_0 + p_1z^{-1} + \dots + p_mz^{-p}}{q_0 + q_1z^{-1} + \dots + q_nz^{-q}}, \quad (22) \end{aligned}$$

where  $\text{CFE}\{u\}$  denotes the continued fraction expansion of  $u$ ;  $p$  and  $q$  are the orders of the approximation and  $P$  and  $Q$  are polynomials of degrees  $p$  and  $q$ . Normally, we can set  $p = q = n$ . A Matlab routine `dfod1()` for this described method (22) can be downloaded (see (Petráš, 2003a)).

**EXAMPLE 2.2.** Here we present some results for fractional order  $r = \pm 0.5$  (half-order derivative/integral). The value of approximation order  $n$  is truncated to  $n = 3$  and weighting factor  $a$  was chosen  $a = 1/3$ . Assume sampling period  $T = 0.001$  sec.

For  $r = 0.5$  we have the following approximation of the fractional half-order derivative:

$$G(z^{-1}) = \frac{985.9 - 1315z^{-1} + 328.6z^{-2} + 36.51z^{-3}}{27 - 18z^{-1} - 3z^{-2} + z^{-3}} \quad (23)$$

A Matlab sequence for the the fractional half-order derivative approximation (23) follows:

```
sys=dfod1(3, 0.001, 1/3, 0.5)
bode(sys)
step(sys)
```



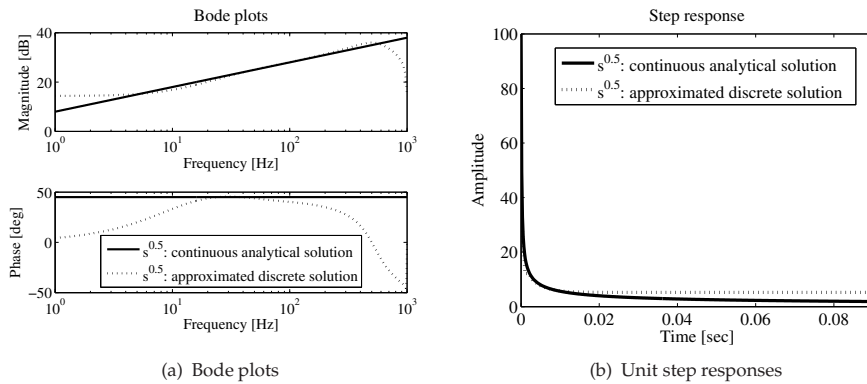


Fig. 3. Characteristics of approximated fractional-order differentiator (23) obtained from (22) for  $r = 0.5$ ,  $n = 3$ ,  $a = 1/3$ , and  $T = 0.001$  sec

The Bode plots and unit step response of the digital fractional-order differentiator (23) and the analytical continuous solution of a fractional semi-derivative are depicted in Fig. 3. Poles and zeros of the transfer function (23) lie in a unit circle.

For  $r = -0.5$  we have the following approximation of the fractional half-order integral:

$$G(z^{-1}) = \frac{0.739 - 0.493z^{-1} - 0.0822z^{-2} + 0.0274z^{-3}}{27 - 36z^{-1} + 9z^{-2} + z^{-3}} \quad (24)$$

A Matlab sequence for the the fractional half-order integral approximation (24) is the following:

```
sys=dfod1(3, 0.001, 1/3, -0.5)
bode(sys)
step(sys)
```

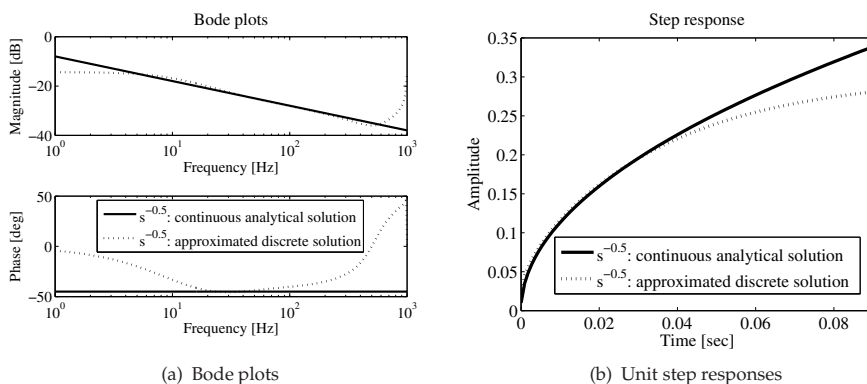


Fig. 4. Characteristics of approximated fractional-order integrator (24) obtained from (22) for  $r = -0.5$ ,  $n = 3$ ,  $a = 1/3$ , and  $T = 0.001$  sec

The Bode plots and unit step response of the digital fractional-order integrator (24) and the analytical continuous solution of a fractional semi-derivative are depicted in Fig. 4. Poles and zeros of the transfer function (24) lie in a unit circle.

For simulation purpose, here we also present the Oustaloup's Recursive Approximation algorithm (Oustaloup et al., 2000). The method is based on the approximation of a function of the form:

$$H(s) = s^r, \quad r \in \mathbb{R}, \quad r \in [-1; 1] \quad (25)$$

for the frequency range selected as  $(\omega_b, \omega_h)$  by a rational function:

$$\hat{H}(s) = C_o \prod_{k=-N}^N \frac{s + \omega'_k}{s + \omega_k} \quad (26)$$

using the following set of synthesis formulas for zeros, poles and the gain:

$$\omega'_k = \omega_b \left( \frac{\omega_h}{\omega_b} \right)^{\frac{k+N+0.5(1-r)}{2N+1}}, \quad \omega_k = \omega_b \left( \frac{\omega_h}{\omega_b} \right)^{\frac{k+N+0.5(1-r)}{2N+1}}, \quad C_o = \left( \frac{\omega_h}{\omega_b} \right)^{-\frac{r}{2}} \prod_{k=-N}^N \frac{\omega_k}{\omega'_k}, \quad (27)$$

where  $\omega_h, \omega_b$  are the high and low transitional frequencies. An implementation of this algorithm in Matlab as a function `ora_foc()` is given in (Chen, 2003).

**EXAMPLE 2.3.** Using the described Oustaloup's-Recursive-Approximation (ORA) method with:

$$\omega_b = 10^{-2}, \quad \omega_h = 10^2 \quad (28)$$

the obtained approximation for fractional function  $H(s) = s^{-\frac{1}{2}}$  nad for  $N = 3$  is as follows:

$$\hat{H}(s) = \frac{0.03162s^7 + 22.42s^6 + 1940s^5 + 2.292 \times 10^4s^4 + 3.755 \times 10^4s^3 + 8523s^2 + 264.3s + 1}{s^7 + 264.3s^6 + 8523s^5 + 3.755 \times 10^4s^4 + 2.292 \times 10^4s^3 + 1940s^2 + 22.42s + 0.03162}. \quad (29)$$

A Matlab sequence for the the fractional half order derivative approximation (29) follows:

```
sys=ora_foc(-0.5,3,0.01,100)
bode(sys)
step(sys)
```

The Bode plots and the unit step response of the approximated half-order integrator (29) obtained for the frequency range (28) and  $N = 3$  are depicted in Fig. 5.

### 2.3.3 Podlubny's matrix approach

This approach is based on the fact that operation of the fractional calculus can be expressed by matrix (Podlubny et al., 2009). It follows from Podlubny (2000), that the left-sided Riemann-Liouville or Caputo fractional derivative  $v^{(\alpha)}(t) = {}_0D_t^\alpha v(t)$  can be approximated in all nodes of the equidistant discretization net  $t = j\tau$  ( $j = 0, 1, \dots, n$ ) simultaneously with the help of the upper triangular strip matrix  $B_n^{(\alpha)}$  as (Podlubny, 2000):

$$\begin{bmatrix} v_n^{(\alpha)} & v_{n-1}^{(\alpha)} & \dots & v_1^{(\alpha)} & v_0^{(\alpha)} \end{bmatrix}^T = B_n^{(\alpha)} \begin{bmatrix} v_n & v_{n-1} & \dots & v_1 & v_0 \end{bmatrix}^T, \quad (30)$$

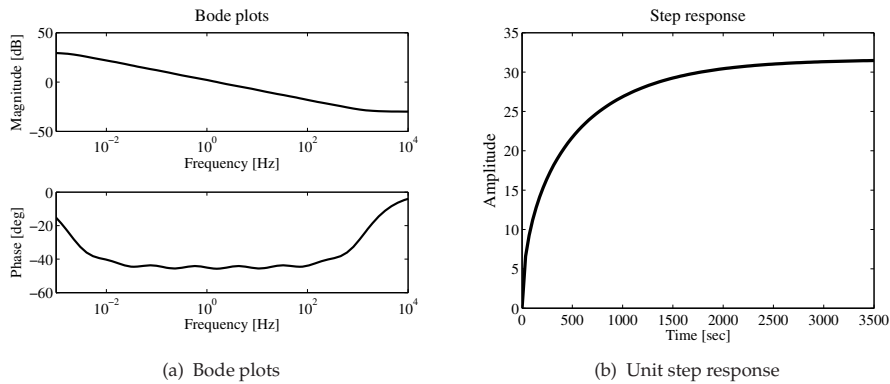


Fig. 5. Characteristics of approximated fractional-order integrator (29) obtained from (27) for  $r = -0.5$ , frequency range (28) and  $N = 3$

where

$$B_n^{(\alpha)} = \frac{1}{\tau^\alpha} \begin{bmatrix} \omega_0^{(\alpha)} & \omega_1^{(\alpha)} & \ddots & \ddots & \omega_{n-1}^{(\alpha)} & \omega_n^{(\alpha)} \\ 0 & \omega_0^{(\alpha)} & \omega_1^{(\alpha)} & \ddots & \ddots & \omega_{n-1}^{(\alpha)} \\ 0 & 0 & \omega_0^{(\alpha)} & \omega_1^{(\alpha)} & \ddots & \ddots \\ \dots & \dots & \dots & \ddots & \ddots & \ddots \\ 0 & \ddots & 0 & 0 & \omega_0^{(\alpha)} & \omega_1^{(\alpha)} \\ 0 & 0 & \dots & 0 & 0 & \omega_0^{(\alpha)} \end{bmatrix}, \quad (31)$$

$$\omega_j^{(\alpha)} = (-1)^j \binom{\alpha}{j}, \quad j = 0, 1, \dots, n. \quad (32)$$

Similarly, the right-sided Riemann-Liouville or Caputo fractional derivative  $v^{(\alpha)}(t) = {}_t D_b^\alpha v(t)$  can be approximated in all nodes of the equidistant discretization net  $t = j\tau$  ( $j = 0, 1, \dots, n$ ) simultaneously with the help of the lower triangular strip matrix  $F_n^{(\alpha)}$  (Podlubny, 2000):

$$\begin{bmatrix} v_n^{(\alpha)} & v_{n-1}^{(\alpha)} & \dots & v_1^{(\alpha)} & v_0^{(\alpha)} \end{bmatrix}^T = F_n^{(\alpha)} \begin{bmatrix} v_n & v_{n-1} & \dots & v_1 & v_0 \end{bmatrix}^T, \quad (33)$$

$$F_n^{(\alpha)} = \frac{1}{\tau^\alpha} \begin{bmatrix} \omega_0^{(\alpha)} & 0 & 0 & 0 & \dots & 0 \\ \omega_1^{(\alpha)} & \omega_0^{(\alpha)} & 0 & 0 & \dots & 0 \\ \omega_2^{(\alpha)} & \omega_1^{(\alpha)} & \omega_0^{(\alpha)} & 0 & \dots & 0 \\ \ddots & \ddots & \ddots & \ddots & \dots & \dots \\ \omega_{n-1}^{(\alpha)} & \ddots & \omega_2^{(\alpha)} & \omega_1^{(\alpha)} & \omega_0^{(\alpha)} & 0 \\ \omega_n^{(\alpha)} & \omega_{n-1}^{(\alpha)} & \ddots & \omega_2^{(\alpha)} & \omega_1^{(\alpha)} & \omega_0^{(\alpha)} \end{bmatrix}. \quad (34)$$

The symmetric Riesz derivative of order  $\beta$  can be approximated based on its definition as a combination of the approximations (30) and (33) for the left and right-sided Riemann-Liouville derivatives, or using the centered fractional differences approximation of the symmetric Riesz derivative suggested recently by Ortigueira (2006). The general formula is the same (Podlubny et al., 2009):

$$\begin{bmatrix} v_m^{(\beta)} & v_{m-1}^{(\beta)} & \dots & v_1^{(\beta)} & v_0^{(\beta)} \end{bmatrix}^T = R_m^{(\beta)} \begin{bmatrix} v_m & v_{m-1} & \dots & v_1 & v_0 \end{bmatrix}^T \quad (35)$$

In the first case, the approximation for the left-sided Caputo derivative is taken one step ahead, and the approximation for the right-sided Caputo derivative is taken one step back. This leads to the matrix

$$R_m^{(\beta)} = \frac{h^{-\alpha}}{2} \begin{bmatrix} -1U_m & +1U_m \end{bmatrix} \quad (36)$$

In the second case (Ortigueira's definition) we have the following symmetric matrix:

$$R_m^{(\beta)} = h^{-\beta} \begin{bmatrix} \omega_0^{(\beta)} & \omega_1^{(\beta)} & \omega_2^{(\beta)} & \omega_3^{(\beta)} & \dots & \omega_m^{(\beta)} \\ \omega_1^{(\beta)} & \omega_0^{(\beta)} & \omega_1^{(\beta)} & \omega_2^{(\beta)} & \dots & \omega_{m-1}^{(\beta)} \\ \omega_2^{(\beta)} & \omega_1^{(\beta)} & \omega_0^{(\beta)} & \omega_1^{(\beta)} & \dots & \omega_{m-2}^{(\beta)} \\ \vdots & \vdots & \vdots & \vdots & \dots & \dots \\ \omega_{m-1}^{(\beta)} & \vdots & \omega_2^{(\beta)} & \omega_1^{(\beta)} & \omega_0^{(\beta)} & \omega_1^{(\beta)} \\ \omega_m^{(\beta)} & \omega_{m-1}^{(\beta)} & \vdots & \omega_2^{(\beta)} & \omega_1^{(\beta)} & \omega_0^{(\beta)} \end{bmatrix} \quad (37)$$

$$\omega_k^{(\beta)} = \frac{(-1)^k \Gamma(\beta + 1) \cos(\beta\pi/2)}{\Gamma(\beta/2 - k + 1) \Gamma(\beta/2 + k + 1)}, \quad k = 0, 1, \dots, m. \quad (38)$$

Both these approximations of symmetric Riesz derivatives give practically the same numerical results and in the case of numerical solution of partial fractional differential equations lead to a well-posed matrix of the resulting algebraic system.

Similarly, if in addition to fractional-order time derivative we also consider symmetric fractional-order spatial derivatives, then we have to use all nodes at the considered time layer from the leftmost to the rightmost spatial discretization node.

Let us consider the nodes  $(ih, j\tau)$ ,  $j = 0, 1, 2, \dots, n$ , corresponding to all time layers at  $i$ -th spatial discretization node. It has been shown in Podlubny (2000) that all values of  $\alpha$ -th order time derivative of  $u(x, t)$  at these nodes are approximated using the discrete analogue of differentiation of arbitrary order (Podlubny et al., 2009):

$$\begin{bmatrix} u_{i,n}^{(\alpha)} & u_{i,n-1}^{(\alpha)} & \dots & u_{i,2}^{(\alpha)} & u_{i,1}^{(\alpha)} & u_{i,0}^{(\alpha)} \end{bmatrix} = B_n^{(\alpha)} \begin{bmatrix} u_{i,n} & u_{i,n-1} & \dots & u_{i,2} & u_{i,1} & u_{i,0} \end{bmatrix}^T. \quad (39)$$

In order to obtain a simultaneous approximation of  $\alpha$ -th order time derivative of  $u(x, t)$  in all nodes, we need to arrange all function values  $u_{ij}$  at the discretization nodes to the form of

a column vector (Podlubny et al., 2009):

$$u_{nm} = \begin{bmatrix} u_{m,n} & u_{m-1,n} & \dots & u_{1,n} & u_{0,n} & & & & \\ & u_{m,n-1} & u_{m-1,n-1} & \dots & u_{1,n-1} & u_{0,n-1} & & & \\ & & & & & & \dots & & \\ & & & & & & & u_{m,1} & u_{m-1,1} & \dots & u_{1,1} & u_{0,1} \\ & & & & & & & & & & & u_{m,0} & u_{m-1,0} & \dots & u_{1,0} & u_{0,0} \end{bmatrix}^T$$

In visual terms, we first take the nodes of  $n$ -th time layer, then the nodes of  $(n-1)$ -th time layer, and so forth, and put them in this order in a vertical column stack.

The matrix that transforms the vector  $U_{nm}$  to the vector  $U_t^{(\alpha)}$  of the partial fractional derivative of order  $\alpha$  with respect to time variable can be obtained as a Kronecker product of the matrix  $B_n^{(\alpha)}$ , which corresponds to the fractional ordinary derivative of order  $\alpha$  (recall that  $n$  is the number of time steps), and the unit matrix  $E_m$  (recall that  $m$  is the number of spatial discretization nodes):

$$T_{mn}^{(\alpha)} = B_n^{(\alpha)} \otimes E_m. \quad (40)$$

Similarly, the matrix that transforms the vector  $U$  to the vector  $U_x^{(\beta)}$  of the partial fractional derivative of order  $\beta$  with respect to spatial variable can be obtained as a Kronecker product of the unit matrix  $E_n$  (recall that  $n$  is the number of spatial discretization nodes), and the matrix  $R_m^{(\beta)}$ , which corresponds to a symmetric Riesz ordinary derivative of order  $\beta$  (Ortigueira, 2006), (recall that  $m$  is the number of time steps):

$$S_{mn}^{(\alpha)} = E_n \otimes R_m^{(\beta)}. \quad (41)$$

Having these approximations for partial fractional derivatives with respect to both variables, we can immediately discretize the general form of the fractional diffusion equation by simply replacing the derivatives with their discrete analogs. Namely, the equation (Podlubny et al., 2009):

$${}_0^C D_t^\alpha u - \chi \frac{\partial^\beta u}{\partial |x|^\beta} = f(x, t) \quad (42)$$

is discretized as

$$\left\{ B_n^{(\alpha)} \otimes E_m - \chi E_n \otimes R_m^{(\beta)} \right\} u_{nm} = f_{nm}. \quad (43)$$

The initial and boundary conditions must be equal to zero. If it is not so, then an auxiliary unknown function must be introduced, which satisfies the zero initial and boundary conditions. In this way, the non-zero initial and boundary conditions moves to the right-hand side of the equation for the new unknown function.

### 3. Ordinary fractional differential equations

A general fractional-order system can be described by a fractional differential equation of the form

$$\begin{aligned} a_n D_t^{\alpha_n} y(t) + a_{n-1} D_t^{\alpha_{n-1}} y(t) + \dots + a_0 D_t^{\alpha_0} y(t) = \\ = b_m D_t^{\beta_m} u(t) + b_{m-1} D_t^{\beta_{m-1}} u(t) + \dots + b_0 D_t^{\beta_0} u(t), \end{aligned} \quad (44)$$

where  $D_t^\gamma \equiv {}_0D_t^\gamma$  denotes the Grünwald-Letnikov, the Riemann-Liouville or Caputo fractional derivative (Podlubny, 1999). The corresponding transfer function of *incommensurate* real orders has the following form (Podlubny, 1999):

$$G(s) = \frac{b_ms^{\beta_m} + \dots + b_1s^{\beta_1} + b_0s^{\beta_0}}{a_ns^{\alpha_n} + \dots + a_1s^{\alpha_1} + a_0s^{\alpha_0}} = \frac{Q(s^{\beta_k})}{P(s^{\alpha_k})}, \quad (45)$$

where  $a_k$  ( $k = 0, \dots, n$ ),  $b_k$  ( $k = 0, \dots, m$ ) are constant, and  $\alpha_k$  ( $k = 0, \dots, n$ ),  $\beta_k$  ( $k = 0, \dots, m$ ) are arbitrary real or rational numbers and without loss of generality they can be arranged as  $\alpha_n > \alpha_{n-1} > \dots > \alpha_0$ , and  $\beta_m > \beta_{m-1} > \dots > \beta_0$ .

In the particular case of *commensurate* order systems, it holds that,  $\alpha_k = \alpha k, \beta_k = \alpha k, (0 < \alpha < 1), \forall k \in \mathbb{Z}$ , and the transfer function has the following form:

$$G(s) = K_0 \frac{\sum_{k=0}^M b_k(s^\alpha)^k}{\sum_{k=0}^N a_k(s^\alpha)^k} = K_0 \frac{Q(s^\alpha)}{P(s^\alpha)}. \quad (46)$$

With  $N > M$ , the function  $G(s)$  becomes a proper rational function in the complex variable  $s^\alpha$  which can be expanded in partial fractions of the following form:

$$G(s) = K_0 \left[ \sum_{i=1}^N \frac{A_i}{s^\alpha + \lambda_i} \right], \quad (47)$$

where  $\lambda_i$  ( $i = 1, 2, \dots, N$ ) are the roots of the pseudo-polynomial  $P(s^\alpha)$  or the system poles which are assumed to be simple without loss of generality. The analytical solution of the system (47) can be expressed as

$$y(t) = \mathcal{L}^{-1} \left\{ K_0 \left[ \sum_{i=1}^N \frac{A_i}{s^\alpha + \lambda_i} \right] \right\} = K_0 \sum_{i=1}^N A_i t^\alpha E_{\alpha, \alpha}(-\lambda_i t^\alpha), \quad (48)$$

where  $E_{\mu, \nu}(z)$  is the Mittag-Leffler function defined as (3).

A fractional-order plant to be controlled can be described by a typical  $n$ -term linear homogeneous fractional-order differential equation (FODE) in the time domain

$$a_n D_t^{\alpha_n} y(t) + \dots + a_1 D_t^{\alpha_1} y(t) + a_0 D_t^{\alpha_0} y(t) = 0 \quad (49)$$

where  $a_k$  ( $k = 0, 1, \dots, n$ ) are constant coefficients of the FODE;  $\alpha_k$  ( $k = 0, 1, 2, \dots, n$ ) are real numbers. Without loss of generality, assume that  $\alpha_n > \alpha_{n-1} > \dots > \alpha_0 \geq 0$ .

The analytical solution of the FODE (49) is given by general formula in the form (Podlubny, 1999):

$$y(t) = \frac{1}{a_n} \sum_{m=0}^{\infty} \frac{(-1)^m}{m!} \sum_{\substack{k_0+k_1+\dots+k_{n-2}=m \\ k_0 \geq 0, \dots, k_{n-2} \geq 0}} (m; k_0, k_1, \dots, k_{n-2}) \times \prod_{i=0}^{n-2} \left( \frac{a_i}{a_n} \right)^{k_i} \mathcal{E}_m \left( t, -\frac{a_{n-1}}{a_n}; \alpha_n - \alpha_{n-1}, \alpha_n + \sum_{j=0}^{n-2} (\alpha_{n-1} - \alpha_j) k_j + 1 \right), \quad (50)$$

where  $(m; k_0, k_1, \dots, k_{n-2})$  are the multinomial coefficients.

Consider a control function which acts on the FODE system (49) as follows:

$$a_n D_t^{\alpha_n} y(t) + \dots + a_1 D_t^{\alpha_1} y(t) + a_0 D_t^{\alpha_0} y(t) = u(t). \quad (51)$$

By the Laplace transform, we can get a fractional transfer function:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{1}{a_n s^{\alpha_n} + \dots + a_1 s^{\alpha_1} + a_0 s^{\alpha_0}}. \quad (52)$$

The fractional-order linear time-invariant (LTI) system can also be represented by the following state-space model:

$$\begin{aligned} {}_0 D_t^{\mathbf{q}} x(t) &= \mathbf{A}x(t) + \mathbf{B}u(t) \\ y(t) &= \mathbf{C}x(t), \end{aligned} \quad (53)$$

where  $x \in \mathbb{R}^n$ ,  $u \in \mathbb{R}^r$  and  $y \in \mathbb{R}^p$  are the state, input and output vectors of the system and  $\mathbf{A} \in \mathbb{R}^{n \times n}$ ,  $\mathbf{B} \in \mathbb{R}^{n \times r}$ ,  $\mathbf{C} \in \mathbb{R}^{p \times n}$ , and  $\mathbf{q} = [q_1, q_2, \dots, q_n]^T$  are the fractional orders. If  $q_1 = q_2 = \dots = q_n \equiv \alpha$ , system (53) is called a commensurate-order system, otherwise it is an incommensurate-order system.

EXAMPLE 3.1. Let us consider a simple two-term fractional differential equation with zero initial condition in general form:

$$a D_t^{\alpha} y(t) + b y(t) = 1 \quad (54)$$

Solution can be obtained by using the Laplace transform method. In the Laplace domain we can express the solution as

$$Y(s) = \frac{1/a}{s(s^{\alpha} + b/a)} \quad (55)$$

and by using the useful relations (6) and (12), in the time domain we can write a general solution

$$y(t) = \frac{1}{a} \mathcal{E}_0\left(t, -\frac{b}{a}; \alpha, \alpha + 1\right) \equiv \frac{1}{a} t^{\alpha} E_{\alpha, \alpha+1}\left(-\frac{b}{a} t^{\alpha}\right). \quad (56)$$

For obtaining the solution in Matlab we can use the following sequence of commands:

```
clear all; close all;
a=2; b=1; alpha=1.5;
t=0:0.001:20; % for time step 0.001 and computational time 20 sec
y=(1/a)*t.^(alpha).*mlf(alpha,alpha+1,((-b/a)*t.^(alpha)));
plot(t,y);
xlabel('Time [sec]');
ylabel('y(t)');
```

EXAMPLE 3.2. Let us consider a three-term fractional differential equation in the form

$$a_2 D_t^{\alpha_2} y(t) + a_1 D_t^{\alpha_1} y(t) + a_0 y(t) = u(t). \quad (57)$$

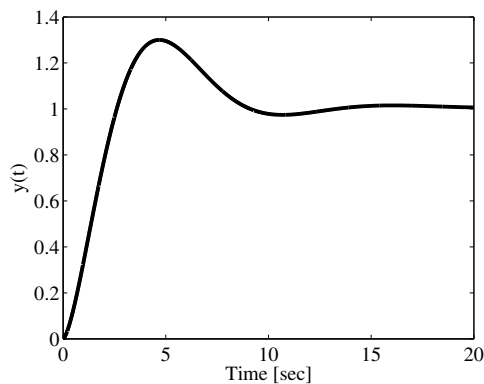


Fig. 6. Solution of the equation (55) for parameters:  $\alpha = 1.5$ ,  $a = 2$ , and  $b = 1$ , under zero initial conditions, time step 0.001 and computation time 20 sec

Substituting (13) into the equation (57), one can write

$$\frac{a_2}{h^{\alpha_2}} \sum_{j=0}^k q_j^{(\alpha_2)} y(t_k - j) + \frac{a_1}{h^{\alpha_1}} \sum_{j=0}^k q_j^{(\alpha_1)} y(t_k - j) + a_0 y(t_k) = u(t_k), \quad (58)$$

where  $t_k = kh$  ( $k = 1, 2, \dots, N$ ) and  $q_j^{(\alpha)}$  are binomial coefficients calculated according to (14). After some rearrangement of the terms in relation (58) we can obtain the numerical solution of the fractional differential equation (57) in the following form:

$$y(t_k) = \frac{u(t_k) - \frac{a_2}{h^{\alpha_2}} \sum_{j=1}^k q_j^{(\alpha_2)} y(t_k - j) - \frac{a_1}{h^{\alpha_1}} \sum_{j=1}^k q_j^{(\alpha_1)} y(t_k - j)}{\frac{a_2}{h^{\alpha_2}} + \frac{a_1}{h^{\alpha_1}} + a_0}, \quad (59)$$

where  $k = 1, 2, \dots, N$  for  $N = T_{sim}/h$  and where  $T_{sim}$  is the total time of the calculation. The above approach is general and can be used for  $n$ -term fractional differential equation (44).

```
clear all; close all;
a2=0.8; a1=0.5; a0=1.0; alpha2=2.2; alpha1=0.9;
h=0.05; TSim=35;
n=round(TSim/h);
cp1=1; cp2=1; Y0=0; u=1.0;
for j=1:n
    c1(j)=(1-(1+alpha1)/j)*cp1;
    c2(j)=(1-(1+alpha2)/j)*cp2;
    cp1=c1(j); cp2=c2(j);
end
Y(1)=Y0;
for i=2:n
    Y(i)=(u - (a2)*h^(-alpha2)*memo(Y,c2,i) - ...
    (a1)*h^(-alpha1)*memo(Y,c1,i))/(a2/(h^alpha2)+a1/(h^alpha1)+a0);
end
```



```

T=0:h:TSim;
plot(T,Y);
xlabel('Time [sec]'); ylabel('y(t)');

function [yo] = memo(r, c, k)
%
temp = 0;
for j=1:k-1
    temp = temp + c(j)*r(k-j);
end
yo = temp;

```

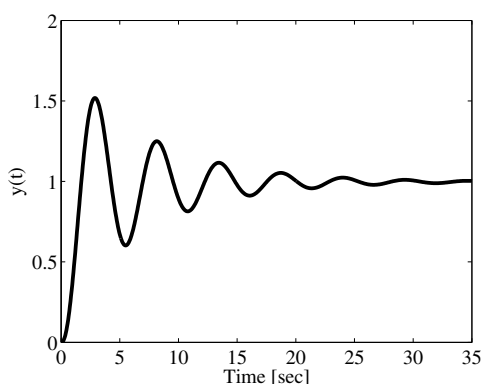


Fig. 7. Solution of the equation (57) for parameters:  $a_2 = 0.8$ ,  $a_1 = 0.5$ ,  $a_0 = 1$ ,  $\alpha_2 = 2.2$ ,  $\alpha_1 = 0.9$  for  $u(t) = 1$ , under zero initial conditions, time step  $h = 0.05$  and computation time  $T_{sim} = 35$  sec

EXAMPLE 3.3. Let us consider the following three-term fractional differential equations, also called the Bagley-Torvik equation, in the form (Podlubny, 1999):

$$Ay''(t) + BD_t^\alpha y(t) + Cy(t) = F(t), \quad (60)$$

where  $F(t) = 8$  for  $0 \leq t \leq 1$  and  $F(t) = 8$  for  $t > 1$ , and with zero initial conditions  $y(0) = y'(0) = 0$ .

In this case we will use a matrix approach and the following syntax of the Matlab functions (Podlubny et al., 2008):

```

clear all; close all;
alpha = 1.5; A = 1; B = 0.5; C = 1;
h=0.05;
TSim=40;
T=0:h:TSim;
N=TSim/h + 1;
M=zeros(N,N);

```

```

Dalphi = ban(alpha,N,h);
D2 = ban(2,N,h);
M=A*D2 + B*Dalphi + C*eye(size(Dalphi));
F=8*(T<=1)';
M = eliminator(N,[1 2])*M*eliminator(N,[1 2])';
F= eliminator(N,[1 2])*F;
Y=M\F;
Y0 = [0; 0; Y];
figure(1)
plot(T,Y0')
xlabel('Time [sec]');
ylabel('y(t)');

```

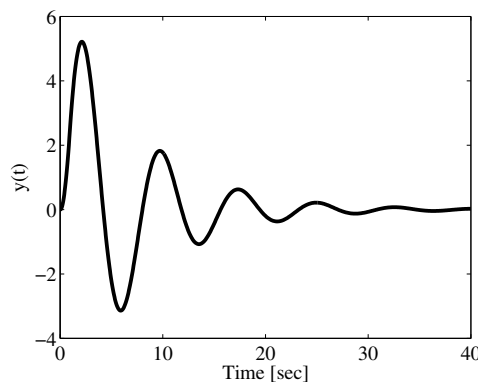


Fig. 8. Solution of the equation (60) for parameters:  $A = 1$ ,  $B = 0.5$ ,  $C = 1$ , and  $\alpha = 1.5$  under zero initial conditions, time step  $h = 0.05$  and computation time 40 sec

EXAMPLE 3.4. The transfer function of the closed feedback control loop with the DC motor (DCM) and the fractional-order controller designed for desired value of phase margin  $\Phi_m = 45^\circ$  and gain margin equal to infinity, has the following form (Petráš, 2009):

$$G_c(s) = \frac{Y(s)}{W(s)} = \frac{G_o(s)}{1 + G_o(s)} = \frac{G_{DCM}(s)C(s)}{1 + G_{DCM}(s)C(s)} = \frac{0.05s + 1}{0.05s^{2.5} + s^{1.5} + 0.05s + 1}, \quad (61)$$

where  $G_o(s)$  is the transfer function of the open control loop. Transfer function (61) corresponds to the fractional differential equation:

$$0.05D_t^{2.5}y(t) + D_t^{1.5}y(t) + 0.05y'(t) + y(t) = 0.05w'(t) + w(t) \quad (62)$$

The feedback control loop described above can be simulated in Matlab environment by using the approximation technique described in previous section, namely Oustaloup's recursive approximation function `ora_foc()` for desired frequency range  $\omega_b = 10^{-2}$ ,  $\omega_h = 10^2$ , and  $N = 6$ .

```

close all; clear all;
Gs_DCM=tf([0.08],[0.05 1 0]);
Cs=(0.625*ora_foc(0.5,6,0.01,100))+(12.5*ora_foc(-0.5,6,0.01,100));
Gs_close=(Gs_DCM*Cs)/(1+(Gs_DCM*Cs));
step(Gs_close,15);
Gs_open=(Gs_DCM*Cs);
figure; bode(Gs_open);
[Gm,Pm] = margin(Gs_open);

```

The results obtained via described Matlab scripts are depicted in Fig. 9. The continuous model is shown with solid line. Phase margin is  $\Phi_m \approx 44.9^\circ$  and gain margin is infinity.

The discrete version of the continuous fractional-order transfer function can be obtained by using the digital operator (21) and Matlab function for approximation of digital fractional-order derivative/integral `dfod1()` for sampling time  $T = 1$  sec, ratio  $a = 1/3$ , and  $n = 5$ .

```

close all; clear all;
T=0.1; a=1/3;
z=tf('z',T,'variable','z^-1');
Hz=((1+a)/T)*((1-z^-1)/(1+a*z^-1));
Gz_DCM=0.08/(Hz*(0.05*Hz+1));
Cz=0.625*dfod1(5,T,a,0.5)+12.5*dfod1(5,T,a,-0.5);
Gz_close=(Gz_DCM*Cz)/(1+(Gz_DCM*Cz));
step(Gz_close,15);
Gz_open=(Gz_DCM*Cz);
figure; bode(Gz_open);
[Gm,Pm] = margin(Gz_open);

```

The results obtained via described Matlab scripts are depicted in Fig. 9. The discrete model is shown with dashed line. Phase margin is  $\Phi_m \approx 45.1^\circ$  and gain margin is infinity.

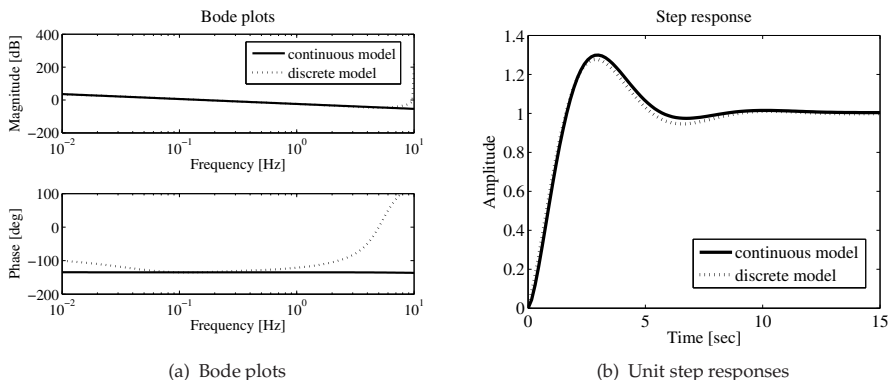


Fig. 9. Characteristics of fractional order transfer function (61)

#### 4. Partial fractional differential equations

For the solution of partial differential equations with derivatives of any order (integer and non-integer), with time and space fractional derivatives, we can use a matrix approach.

EXAMPLE 4.1. Let us consider the following one-dimensional fractional diffusion equation with fractional derivatives with respect both to time and to the spatial variable:

$${}_0^C D_t^\alpha y(x, t) - a^2 \frac{\partial^\beta y(x, t)}{\partial |x|^\beta} = f(x, t) \quad (63)$$

with the initial condition  $y(x, 0) = h(x)$  and the boundary conditions  $y(0, t) = y(L, t) = K$ . Note that  $\partial^\beta y(x, t) / \partial |x|^\beta$  is symmetric Riesz derivative. Let us consider the following parameters:  $\alpha = 1.2$ ,  $\beta = 2.1$ ,  $a^2 = 1$ ,  $f(x, t) = 5$ ,  $L = 2$ ,  $T_{sim} = 3$  sec and  $K = 0$ . The Matlab code is (for more details see also (Podlubny et al., 2008; 2009)):

```
clear all; close all;
% set the input parameters:
alpha = 1.2; beta = 2.1; a2 = 1; f_x = 5; L = 2; TSim = 3; K = 0;
% set the calculation parameters:
m = 21; n = 148; h = L / (m - 1); tau = TSim / n;
% alpha-th order derivative with respect to time:
B1 = ban(alpha, n - 1, tau)';
% time derivative matrix:
TD = kron(B1, eye(m));
% beta-th order derivative with respect to x:
B2 = ransym(beta, m, h);
% spatial derivative matrix:
SD = kron(eye(n - 1), B2);
% matrix corresponding to discretization in space and time:
SystemMatrix = TD - a2 * SD;
S = eliminator(m, [1 m]);
SK = kron(eye(n - 1), S);
SystemMatrix_without_columns_1_m = SystemMatrix * SK';

% remove rows with '1' and 'm' from SystemMatrix_without_columns_1_m
S = eliminator(m, [1 m]);
SK = kron(eye(n - 1), S);
SystemMatrix_without_rows_columns_1_m = ...
SK * SystemMatrix_without_columns_1_m;

% initial conditions:
U0 = zeros(1, m);

% right hand side:
F = f_x * ones(size(SystemMatrix_without_rows_columns_1_m, 1), 1);

% solution of the system:
```

```

Y = SystemMatrix_without_rows_columns_1_m\F;

% reshape solution array --
% values for k-th time step are in the k-th column of YS:
YS = reshape(Y,m-2,n-1);
YS = fliplr(YS);

% final solution (take into account the initial condition):
for k = 1:(n-1)
    U(:,k) = (YS(:,k)+U0(2:(m-1))');
end

% plot graph
[rows, columns] = size(U);
U = [ zeros(1, columns)+K; U; zeros(1, columns)+K];
U = [U0' U];
[XX,YY]=meshgrid(tau*(0:n-1),h*(0:m-1));
mesh(XX,YY,U)
xlabel('t'); ylabel('x'); zlabel('y(x,t)');
title(['\alpha =', num2str(alpha), ', \beta = ', num2str(beta)])

```

When we consider non-zero initial condition, e.g. in the form of auxiliary function  $y(x, 0) = h(x) = x(x - 1)$  and non-zero boundary conditions in the form of the constant  $K$ , where  $y(0, t) = y(L, t) = K$ , we have to modify the following lines in above Matlab code:

```

K=2;
...
% initial conditions:
k = 1:m;
U0 = (k-1).*(m-1) - k + 1)*h^2 + K;

```

In Fig. 10(a) and Fig. 10(b) are depicted the solutions of the fractional partial differential equation (63) for zero and non-zero initial and boundary conditions, respectively.

## 5. Nonlinear fractional differential equations

A general numerical solution of the nonlinear fractional differential equation in the form

$${}_a D_t^q y(t) = f(y(t), t),$$

can be expressed by using the relations (13) and (14) as:

$$y(t_k) = f(y(t_k), t_k) h^q - \sum_{j=v}^k c_j^{(q)} y(t_{k-j}). \quad (64)$$

For the *memory term* expressed by the sum, a "short memory" principle can be used. Then the lower index of the sums in relations (64) will be  $v = 1$  for  $k < (L_m/h)$  and  $v = k - (L_m/h)$  for  $k > (L_m/h)$ , or without using the "short memory" principle, we put  $v = 1$  for all  $k$ .

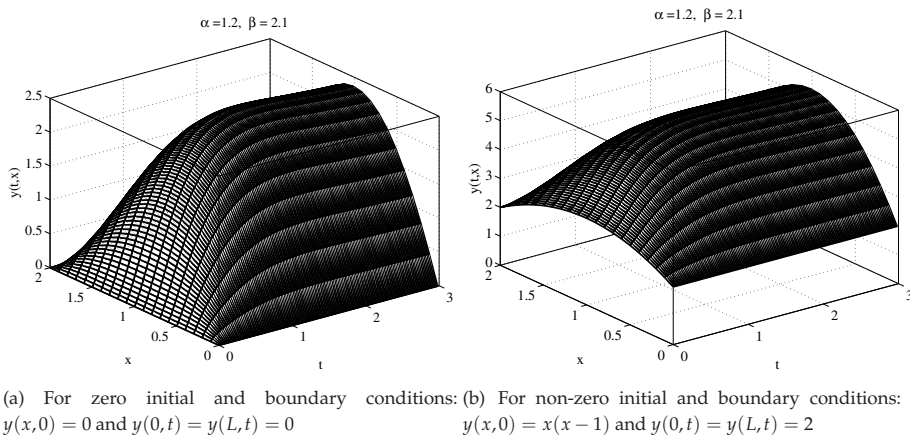


Fig. 10. Solution of the equation (63) for parameters:  $\alpha = 1.2$ ,  $\beta = 2.1$ ,  $a = 1$ , and  $f(x, t) = 5$  for  $L = 2$  and  $T_{sim} = 3$  sec

EXAMPLE 5.1. In this section, we will demonstrate the proposed numerical solution on the set of three nonlinear fractional differential equations, which are used for describing an economical system. The fractional-order financial system is described as follows (Chen, 2008):

$$\begin{aligned} {}_0D_t^{q_1} x(t) &= z(t) + (y(t) - a)x(t), \\ {}_0D_t^{q_2} y(t) &= 1 - by(t) - x(t)^2, \\ {}_0D_t^{q_3} z(t) &= -x(t) - cz(t), \end{aligned} \quad (65)$$

where the total order of the system is denoted by  $\bar{q} = (q_1, q_2, q_3)$ ,  $a$  is the saving amount,  $b$  is the cost per investment, and  $c$  is the elasticity of demand of commercial market. The state variables are:  $x(t)$  is the interest rate,  $y(t)$  is the investment demand, and  $z(t)$  is the price index.

The numerical solution of the fractional-order financial system (65) has the following form (Petráš, 2011):

$$\begin{aligned} x(t_k) &= (z(t_{k-1}) - (y(t_{k-1}) - a)x(t_{k-1})) h^{q_1} - \sum_{j=v}^k c_j^{(q_1)} x(t_{k-j}), \\ y(t_k) &= \left(1 - by(t_k) - x^2(t_k)\right) h^{q_2} - \sum_{j=v}^k c_j^{(q_2)} y(t_{k-j}), \\ z(t_k) &= (-x(t_k) - cz(t_{k-1})) h^{q_3} - \sum_{j=v}^k c_j^{(q_3)} z(t_{k-j}), \end{aligned} \quad (66)$$

where  $T_{sim}$  is the simulation time,  $k = 1, 2, 3, \dots, N$ , for  $N = [T_{sim}/h]$ , and  $(x(0), y(0), z(0))$  is the start point (initial conditions). The binomial coefficients  $c_j^{(q_i)}$ ,  $\forall i$ , are calculated according to relation (14).

Let us consider the following parameters of the system (65):  $a = 1.0$ ,  $b = 0.1$ ,  $c = 1.0$ , orders  $q_1 = 1.1$ ,  $q_2 = 0.9$ ,  $q_3 = 0.8$ , computational time  $T_{sim} = 200$  days, for time step  $h = 0.04166$ , and initial conditions  $(x(0), y(0), z(0)) = (1, -1, 1)$ . We will use  $v = 1$  in (66) for all  $k$ .

The Matlab code for the solution of system (65) follows:

```
close all; clear all;
[t, y]=FOFinanc([1 0.1 1],[1.1 0.9 0.8],200, [1 -1 1]);
plot3(y(:,1), y(:,2), y(:,3),'k'); % in 3D state space
xlabel('x(t)'); ylabel('y(t)'); zlabel('z(t)'); grid;
figure; plot(y(:,1), y(:,2),'k'); % projection onto x-y plane
xlabel('x(t)'); ylabel('y(t)'); grid;
```

where routine FOFinanc() consists of the following code (Petráš, 2010):

```
function [T, Y]=FOFinanc(parameters, orders, TSim, Y0)
h=0.04166;
% number of calculated mesh points:
n=round(TSim/h);
%orders of derivatives, respectively:
q1=orders(1); q2=orders(2); q3=orders(3);
% constants of financial system:
a=parameters(1); b=parameters(2); c=parameters(3);
% binomial coefficients calculation:
cp1=1; cp2=1; cp3=1;
for j=1:n
    c1(j)=(1-(1+q1)/j)*cp1;
    c2(j)=(1-(1+q2)/j)*cp2;
    c3(j)=(1-(1+q3)/j)*cp3;
    cp1=c1(j); cp2=c2(j); cp3=c3(j);
end
% initial conditions setting:
x(1)=Y0(1); y(1)=Y0(2); z(1)=Y0(3);
% calculation of phase portraits /numerical solution/:
for i=2:n
    x(i)=(z(i-1)+(y(i-1)-a)*x(i-1))*h^q1 - memo(x, c1, i);
    y(i)=(1-b*y(i-1)-x(i)^2)*h^q2 - memo(y, c2, i);
    z(i)=(-x(i)-c*z(i-1))*h^q3 - memo(z, c3, i);
end
for j=1:n
    Y(j,1)=x(j); Y(j,2)=y(j); Y(j,3)=z(j);
end
T=0:h:TSim;
```

Supporting function memo() was already introduced in previous section of this chapter.

In Fig. 11 are depicted the simulation results of the financial system (65) for the following parameters:  $a = 1$ ,  $b = 0.1$ , and  $c = 1.0$ , orders  $q_1 = 1.1$ ,  $q_2 = 0.9$ ,  $q_3 = 0.8$ , computational time 200 days, and for time step  $h = 0.04166$  (i.e. approximately one hour sampling).

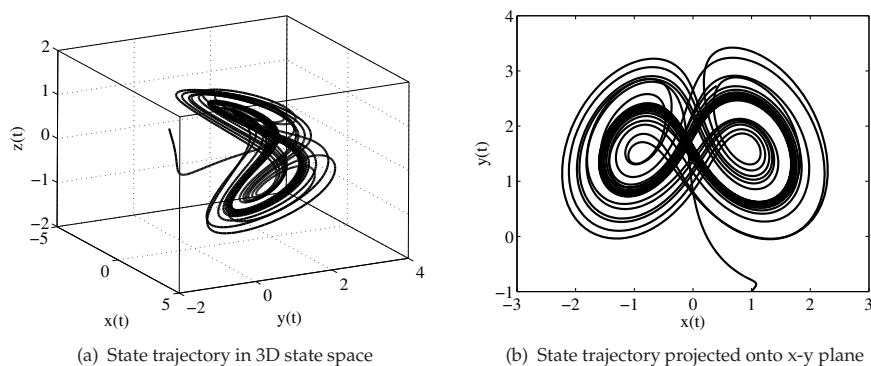


Fig. 11. Simulation result of the fractional-order financial system (65) in state space for the initial conditions  $(x(0), y(0), z(0)) = (1, -1, 1)$

## 6. Conclusion

In this chapter are presented the methods for calculation of the fractional derivative and fractional integral together with methods for solution of the fractional differential equations in Matlab. Those methods are based on the numerical approximation of the fractional derivatives and integral in the continuous time, discrete time and frequency domains. For each mentioned methods are presented illustrative examples with a Matlab code. Moreover, all described problems can be solved also in Simulink environment but such approach was omitted in this chapter. Obviously the described problems could be solved via other methods and also by different Matlab functions but we shown the way how one can create its own *Fractional Calculus Toolbox* for Matlab from functions, which are freely downloadable from the MathWorks, Inc. Matlab Central File Exchange. It depends on the problem which should be solved in Matlab. Except above-mentioned open source routines, there are very useful and effective Matlab functions for solution and analysis of the fractional calculus problems, which are described in (Monje et al., 2010). Especially helpful is the linear fractional-order differential equations solver `fode_sol()` and also the set of functions already published in paper (Chen et al., 2009).

It is worth mentioning that there are many additional useful functions at the Matlab Central File Exchange web site, which are not used in this chapter. For instance variable-order, distributed-order, and random-order fractional equations, predictor corrector numerical method, impulse and step responses invariant discretization of fractional-order differentiators/integrators as well as various fractional-order digital filters.

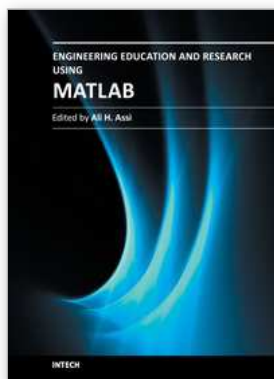
## 7. References

- Al-Alaoui, M. A. (1993). Novel digital integrator and differentiator, *Electron. Lett.*, Vol. 29, 376–378.
- Al-Alaoui, M. A. (1997). Filling the gap between the bilinear and the backward difference transforms: An interactive design approach, *Int. J. Elect. Eng. Edu.*, Vol. 34, 331–337.
- Bayat, F. M. (2007). Fractional Differentiator, MathWorks, Inc. Matlab Central File Exchange, URL: [www.mathworks.com/matlabcentral/fileexchange/13858](http://www.mathworks.com/matlabcentral/fileexchange/13858).



- Caponetto, R.; Dongola, G.; Fortuna, L. and Petráš, I. (2010). *Fractional Order Systems: Modeling and Control Applications*, World Scientific, Singapore.
- Chen, Y. Q. and Moore, K. L. (2002). Discretization Schemes for Fractional-Order Differentiators and Integrators, *IEEE Trans. On Circuits and Systems - I: Fundamental Theory and Applications*, Vol. 49, No. 3, 363–367.
- Chen, Y. Q. (2003). Oustaloup–Recursive–Approximation for Fractional Order Differentiators, MathWorks, Inc. Matlab Central File Exchange, URL: [www.mathworks.com/matlabcentral/fileexchange/3802](http://www.mathworks.com/matlabcentral/fileexchange/3802).
- Chen, Y. Q. (2008). Generalized Mittag-Leffler Function, MathWorks, Inc. Matlab Central File Exchange, URL: [www.mathworks.com/matlabcentral/fileexchange/20849](http://www.mathworks.com/matlabcentral/fileexchange/20849).
- Chen, Y. Q.; Petráš, I. and Xue, D. (2009). Fractional order control - A tutorial, *Proc. of the American Control Conference*, pp. 1397–1411, June 10–12, 2009, St. Louis, USA.
- Chen, W. Ch. (2008). Nonlinear dynamic and chaos in a fractional-order financial system, *Chaos, Solitons and Fractals*, Vol. 36, 1305–1314.
- CRONE Research Group. (2010). CRONE Toolbox, URL: [www.ims-bordeaux.fr/CRONE/toolbox/](http://www.ims-bordeaux.fr/CRONE/toolbox/).
- Dorčák, Ľ. (1994). Numerical Models for the Simulation of the Fractional-Order Control Systems, *UEF-04-94, The Academy of Sciences, Inst. of Experimental Physic*, Košice, Slovakia.
- Djrbashian, M. M. (1993). *Harmonic Analysis and Boundary Value problems in the Complex Domain*, Birkhäuser Verlag, Basel.
- Lubich, Ch. (1986). Discretized fractional calculus, *SIAM J. Math. Anal.*, Vol. 17, 704–719.
- Magin, R. L. (2006). *Fractional Calculus in Bioengineering*, Begell House Publishers.
- Miller, K. S. and Ross, B. (1993). *An Introduction to the Fractional Calculus and Fractional Differential Equations*, John Wiley and Sons. Inc., New York.
- Monje, C. A.; Chen, Y. Q.; Vinagre, B. M.; Xue, D. and Feliu, V. (2010). *Fractional-order Systems and Controls*, Series: Advances in Industrial Control, Springer.
- Oldham, K. B. and Spanier, J. (1974). *The Fractional Calculus*, Academic Press, New York.
- Ortigueira, M. D. (2006). Riesz potential operators and inverses via fractional centered derivatives, *Int. J. Math. Math. Sci.*, Article ID 48391, 1–12.
- Oustaloup, A. (1995). *La Derivation Non Entiere: Theorie, Synthese et Applications*, Hermes, Paris.
- Oustaloup, A.; Levron, F.; Mathieu, B. and Nanot, F. M. (2000). Frequency-band complex noninteger differentiator: characterization and synthesis, *IEEE Trans. on Circuits and Systems I: Fundamental Theory and Applications I*, Vol. 47, 25–39.
- Petráš, I. (2003a). Digital Fractional Order Differentiator/integrator – IIR type, MathWorks, Inc. Matlab Central File Exchange, URL: [www.mathworks.com/matlabcentral/fileexchange/3672](http://www.mathworks.com/matlabcentral/fileexchange/3672).
- Petráš, I. (2003b). Digital Fractional Order Differentiator/integrator – FIR type, MathWorks, Inc. Matlab Central File Exchange, URL: [www.mathworks.com/matlabcentral/fileexchange/3673](http://www.mathworks.com/matlabcentral/fileexchange/3673).
- Petráš, I. (2009). Fractional-Order Feedback Control of a DC Motor, *Journal of Electrical Engineering*, Vol. 60, No. 3, 117–128.
- Petráš, I. (2010). Fractional Order Chaotic Systems, MathWorks, Inc. Matlab Central File Exchange, URL: [www.mathworks.com/matlabcentral/fileexchange/27336](http://www.mathworks.com/matlabcentral/fileexchange/27336).
- Petráš, I. (2011). *Fractional-Order Nonlinear Systems: Modeling, Analysis and Simulation* Series: Nonlinear Physical Science, Springer, HEP.
- Podlubny, I. (1999). *Fractional Differential Equations*, Academic Press, San Diego.

- Podlubny, I. (2000). Matrix approach to discrete fractional calculus, *Fractional Calculus and Applied Analysis*, Vol. 3, 359–386.
- Podlubny, I. and Kacenak, M. (2005). Mittag-Leffler Function, MathWorks, Inc. Matlab Central File Exchange, URL: [www.mathworks.com/matlabcentral/fileexchange/8738](http://www.mathworks.com/matlabcentral/fileexchange/8738).
- Podlubny, I.; Skovranek, T. and Vinagre, B. M. (2008). Matrix Approach to Discretization of ODEs and PDEs of Arbitrary Real Order, MathWorks, Inc. Matlab Central File Exchange, URL: [www.mathworks.com/matlabcentral/fileexchange/22071](http://www.mathworks.com/matlabcentral/fileexchange/22071).
- Podlubny, I.; Chechkin, A.; Skovranek, T.; Chen, Y. Q. and Vinagre, B. M. J. (2009). Matrix approach to discrete fractional calculus II: Partial fractional differential equations, *Journal of Computational Physics*, Vol. 228, 3137–3153.
- Valério, D. (2005). Toolbox Ninteger for MatLab, v. 2.3, MathWorks, Inc. Matlab Central File Exchange, URL: [www.mathworks.com/matlabcentral/fileexchange/8312](http://www.mathworks.com/matlabcentral/fileexchange/8312).
- Vinagre, B. M.; Podlubny, I.; Hernández, A. and Feliu, V. (2000). Some approximations of fractional order operators used in control theory and applications, *Fractional Calculus and Applied Analysis*, Vol. 3, 231–248.
- Vinagre, B. M.; Chen, Y. Q. and Petráš, I. (2003). Two direct Tustin discretization methods for fractional-order differentiator/integrator, *J. Franklin Inst.*, Vol. 340, 349–362.
- Sierociuk, D. (2005). Fractional Order Discrete State–Space System Simulink Toolkit User Guide, [www.ee.pw.edu.pl/~sieroci/fsst/fsst.htm](http://www.ee.pw.edu.pl/~sieroci/fsst/fsst.htm).
- Sheng, H.; Li, Y. and Chen, Y. Q. (2011). Application of numerical inverse Laplace transform algorithms in fractional calculus, *J. Franklin Inst.*, Vol. 348, 315–330.
- Xue, D. (2010). Computational Aspect of Fractional-Order Control Problems, Tutorial Workshop on Fractional Order Dynamic Systems and Controls, Proceedings of the WCICA'2010, Jinan, China. (URL: [mechatronics.ece.usu.edu/foc/cdc10tw/code-matlab-simulink/xdy\\_foccode.rar](http://mechatronics.ece.usu.edu/foc/cdc10tw/code-matlab-simulink/xdy_foccode.rar)).



## **Engineering Education and Research Using MATLAB**

Edited by Dr. Ali Assi

ISBN 978-953-307-656-0

Hard cover, 480 pages

**Publisher** InTech

**Published online** 10, October, 2011

**Published in print edition** October, 2011

MATLAB is a software package used primarily in the field of engineering for signal processing, numerical data analysis, modeling, programming, simulation, and computer graphic visualization. In the last few years, it has become widely accepted as an efficient tool, and, therefore, its use has significantly increased in scientific communities and academic institutions. This book consists of 20 chapters presenting research works using MATLAB tools. Chapters include techniques for programming and developing Graphical User Interfaces (GUIs), dynamic systems, electric machines, signal and image processing, power electronics, mixed signal circuits, genetic programming, digital watermarking, control systems, time-series regression modeling, and artificial neural networks.

### **How to reference**

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Ivo Petrás (2011). Fractional Derivatives, Fractional Integrals, and Fractional Differential Equations in Matlab, Engineering Education and Research Using MATLAB, Dr. Ali Assi (Ed.), ISBN: 978-953-307-656-0, InTech, Available from: <http://www.intechopen.com/books/engineering-education-and-research-using-matlab/fractional-derivatives-fractional-integrals-and-fractional-differential-equations-in-matlab>

**INTeCH**  
open science | open minds

### **InTech Europe**

University Campus STeP Ri  
Slavka Krautzeka 83/A  
51000 Rijeka, Croatia  
Phone: +385 (51) 770 447  
Fax: +385 (51) 686 166  
[www.intechopen.com](http://www.intechopen.com)

### **InTech China**

Unit 405, Office Block, Hotel Equatorial Shanghai  
No.65, Yan An Road (West), Shanghai, 200040, China  
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元  
Phone: +86-21-62489820  
Fax: +86-21-62489821