www.czasopisma.pan.pl PAN POLSKA AKADEMIA NAUK www.journals.pan.pl

BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 66, No. 4, 2018
DOI: 10.24425/124264

A local truncation error estimation for a SubIval solver

M. SOWA*

Silesian University of Technology, 2a Akademicka St., 44-100 Gliwice, Poland

Abstract. The paper concerns an analysis for SubIval (the subinterval-based method for fractional derivative computations in initial value problems). A time step size adaptive solver is discussed, for which the formula of a local truncation error is derived. A general form for a system of linear equations is given for the considered class of problems (for which the analysis is performed in the paper). Two circuit examples are introduced to display the usefulness of the SubIval solver. For the examples that have been chosen it is possible to obtain referential solutions through completely different methods. The results obtained through the numerical solver are compared with evaluations of the referential solutions. The error estimation results obtained for the time steps of the SubIval solver are compared with the actual errors, being the differences between the numerical solutions and the referential solutions. The paper also contains a comparison of the accuracy of results obtained through the SubIval solver with the accuracies of other solvers.

Key words: fractional calculus, numerical method, adaptive step size, local truncation error, circuit analysis.

1. Introduction

The first major developments in the field of fractional calculus date back to the 19^{th} century [1, 2]. Since then, many mathematical considerations have been carried out. Apart from it being a challenging mathematical topic, it has also become popular of late because of its many applications, e.g. in heat conduction analyses [3–5], the design and implementation of fractional order filters [6, 7], in the design and analysis of fractional order controllers [8, 9] (including coupled analyses of control of synchronous machines [10, 11]), electric and magnetic field analyses (when materials of complex properties are considered [12–14]), inverse kinematics [15], continuum mechanics [16], viscoelasticity [17, 18].

Fractional calculus is also applied in circuit analyses (computational examples of this field have been applied in later parts of this paper). In circuit analyses, fractional capacitors are applied in modeling supercapacitors [19–22] and fractional coils have shown to accurately resemble the responses of some coils with ferromagnetic cores [23, 24].

Elements of fractional calculus include definitions of fractional derivatives and integrals (or generally differintegrals) with respect to both time (more commonly studied) and space [25]. The study presented in this paper concerns the Caputo time derivative of order $\alpha \in [0, 1]$ [26]:

$$ {}_{t_\mathrm{a}}\mathrm{D}^{\alpha}_{t_\mathrm{b}} x(t) = \frac{1}{\Gamma(1-\alpha)} \int_{t_\mathrm{a}}^{t_\mathrm{b}} \frac{x^{(1)}(\tau)}{(t-\tau)^{\alpha}} \mathrm{d}\tau. \tag{1} $$

The Caputo definition is equivalent to the oldest definition (which is that of Riemann and Liouville [27]) if zero initial conditions are imposed [28].

The increasing interest in fractional calculus can also be attributed to the fact that many studies are performed concerning the methods allowing to solve problems with fractional derivatives and integrals.

The possibility of solving a problem with fractional derivatives by means of a selected method depends on the form of the problem. For linear problems that can be expressed in the form of a system of fractional state equations, for some known source time functions one can apply analytical solutions based on the Mittag-Leffler function [29,30]. For other problems one can apply semi-analytical methods (the general form of the solution is assumed as a series of terms whose coefficients are computed), which are especially well described and tested for problems which do not require many equations (although they may contain nonlinear dependencies). These are e.g. the Adomian decomposition method [31,32] and the differential transform method [33]. For a more general approach, which can deal with the widest variety of problems, one can apply time stepping solvers, using numerical methods. Some of the most common are fractional linear multistep methods (backward difference methods, explained in general in [34]), product integration (PI) rule methods [35], collocation methods [36, 37] and methods using the Grünwald-Letnikov definition of the fractional derivative [38].

The study in this paper concerns the application of a method called SubIval (the subinterval-based numerical method for fractional derivative computations in initial value problems, first discussed in [39]). It is a numerical method that can be applied in time stepping solvers.

2. Application of SubIval

A special notation relating to intervals is used for the differintegration:

$$ \mathrm{d}^{\alpha}_{\Xi} x(t) = {}_{t_\mathrm{a}}\mathrm{D}^{\alpha}_{t_\mathrm{b}} x(t), \tag{2} $$

where $\Xi \in [t_\mathrm{a}, t_\mathrm{b}]$. SubIval relies on a partition of the differintegration interval $\Xi_{\mathrm{tot}} = [t_0, t_{\mathrm{now}}]$ into subintervals (with

*e-mail: marcin.sowa@polsl.pl

Manuscript submitted 2017-11-16, revised 2018-03-26, initially accepted for publication 2018-05-22, published in August 2018.

t_0 being the initial time instance and t_{now} being the computed time instance):

$$\mathrm{d}^{\alpha}_{\Xi_{\text{tot}}} x(t) = \sum_{s=1}^{M} \mathrm{d}^{\alpha}_{\Xi_s} x(t). \tag{3}$$

For each subinterval $\Xi_s = [t_{s\,\text{start}}, t_{s\,\text{end}}]$ if $s > 1$ then $t_{s\,\text{start}} = t_{s-1\,\text{end}}$. In the first subinterval $t_{1\,\text{start}} = t_0$, while in the last one $t_{M\,\text{end}} = t_{\text{now}}$.

In each of these subintervals the solution is approximated by means of a polynomial:

$$\mathrm{d}^{\alpha}_{\Xi_{\text{tot}}} x(t) \approx \sum_{s=1}^{M} \mathrm{d}^{\alpha}_{\Xi_s} \widetilde{x}_s(t). \tag{4}$$

Each approximation $\widetilde{x}_s(t)$ is built upon nodes denoted by $t_{s,1}, t_{s,2}, \ldots t_{s,n_s}$ (the order of each polynomial $\widetilde{x}_s(t)$ can, therefore, be given by $q_s = n_s - 1$). These are covered by subintervals denoted by $\Theta_s = [t_{s,1}, t_{s,n_s}]$, where for each subinterval pair $\Theta_s \supseteq \Xi_s$. The subintervals are formed and modified according to a subinterval dynamics algorithm described with great detail (with examples) in [40].

Each approximation can be given by:

$$\widetilde{x}_s(t) = \sum_{j=1}^{n_s} x_{s,j} L_{s,j}(t) \tag{5}$$

where $L_{s,j}$ denote Lagrange basis polynomials:

$$L_{s,j}(t_{s,k}) = \begin{cases} 0, & \text{if } k \neq j, \\ 1, & \text{if } k = j, \end{cases} \tag{6}$$

while $x_{s,j}$ are the values of the considered variable computed at their respective $t_{s,j}$ nodes.

SubIval can be used in a typical time stepping solver, in which for each subsequently computed time instance $t = t_{\text{now}}$ the only values treated as unknowns are the variables selected at this time instance. Values of previous time instances are not modified anymore and treated as known values.

Through the subinterval dynamics and local polynomial differintegrations (applying analytical formulae for monomials [42, 43]) SubIval leads to a convenient implicit formula:

$$\mathrm{d}^{\alpha}_{\Xi_{\text{tot}}} x(t) \approx a x_{M,n_M} + b, \tag{7}$$

where a and b are computed for each variable separately every time when a new time instance t_{now} is selected. a results from the differintegrations of the Lagrange basis polynomial being the multiplier of the still unknown value x_{M,n_M}:

$$a = \mathrm{d}^{\alpha}_{\Xi_M} L_{M,n_M}(t). \tag{8}$$

One can distinguish two terms in b:

$$b = b_M + b_{\text{prev}}, \tag{9}$$

where b_M depends on the differintegrals in the Ξ_M subinterval:

$$b_M = \mathrm{d}^{\alpha}_{\Xi_M} \sum_{j=1}^{n_M - 1} x_{M,j} L_{M,j}(t), \tag{10}$$

while b_{prev} depends on the differintegrals in the remaining subintervals with indices $s = 1, 2, \ldots M-1$:

$$b_{\text{prev}} = \sum_{s=1}^{M-1} \mathrm{d}^{\alpha}_{\Xi_s} \sum_{j=1}^{n_s} x_{s,j} L_{s,j}(t). \tag{11}$$

3. Approximation of the local truncation error

One of the properties of SubIval and its subinterval dynamics is that they make only the M-th subinterval (being the rightmost subinterval on the time axis) cover the node with unknown variables. Also, in the typical time stepping solver it is designed for, one only controls the time step size between t_{now} and the previous node (this step size is further on denoted by Δt_{now}).

Assuming the solution $x(t)$ is infinitely differentiable in Θ_M then it can be expressed by means of a power series:

$$x(t_{\text{loc}}) = c_0 + \sum_{k=1}^{\infty} c_k t_{\text{loc}}^k, \tag{12}$$

where $t_{\text{loc}} = t - t_{M,1}$.

The error of the fractional derivative approximation resulting from SubIval, in the subinterval Ξ_M, is given by the formula:

$$e_{\text{True } M} = \mathrm{d}^{\alpha}_{\Xi_M} x(t_{\text{loc}}) - \mathrm{d}^{\alpha}_{\Xi_M} \widetilde{x}_M(t_{\text{loc}}). \tag{13}$$

One of the purposes of estimating the local truncation error is to be able to control the error of the numerical computations. If one can only control the error by changing Δt_{now} then $e_{\text{True } M}$ must be given in terms of this variable.

Because the considerations concern only the subinterval Ξ_M and nodes covered by Θ_M – a simpler notation: $t_j = t_{M,j}$ will be used in this section along with the number of nodes $n = n_M$.

The step sizes of Θ_M are denoted by:

$$\Delta t_j = t_j - t_{j-1}, \tag{14}$$

with their indices starting from $j = 2$. Hence, the time instances can be given by:

$$t_j = \begin{cases} 0, & \text{if } j = 1, \\ \sum\limits_{k=2}^{j} \Delta t_k, & \text{if } j = 2, \ldots n. \end{cases} \tag{15}$$

Each subsequent step size will be modified by multiplying the previous one by a computed coefficient (denoted by η_j):

$$\Delta t_j = \eta_j \Delta t_{j-1}, \tag{16}$$

with the indices of η_j starting from $j = 3$. By introducing $\mu_j = \eta_j^{-1}$ the step sizes can be given by:

$$\Delta t_j = \begin{cases} \left(\prod\limits_{r=j+1}^{n} \mu_r \right) \Delta t_{\text{now}}, & \text{if } j = 2, \ldots n-1, \\ \Delta t_{\text{now}}, & \text{if } j = n. \end{cases} \tag{17}$$

Finally, the time instances are given by:

$$t_j = \begin{cases} 0, & \text{if } j = 0, \\ \Delta t_{\text{now}} \sum\limits_{k=2}^{j} \prod\limits_{r=k+1}^{n} \mu_r, & \text{if } j = 2, \ldots n-1, \\ \Delta t_{\text{now}} \left(1 + \sum\limits_{k=2}^{j} \prod\limits_{r=k+1}^{n} \mu_r\right), & \text{if } j = n. \end{cases} \tag{18}$$

The following steps are involved in obtaining a formula for $e_{\text{True } M}$:

- as it is done in a typical local truncation error estimation [41] the errors of the computed variable at the selected time instances are omitted and each x_j is substituted by the respective $x(t_j)$,
- Lagrange basis polynomials are given in terms of dependencies on Δt_{now} and the μ coefficients,
- $x(t_{\text{loc}})$ is truncated into a sum of finite terms, where the remaining polynomial is at least one order higher than that of the approximation $\widetilde{x}_M(t_{\text{loc}})$,
- the fractional derivative of $\widetilde{x}(t_{\text{loc}})$ and that of the finite sum representation of $x(t_{\text{loc}})$ are computed (for the interval Ξ_M) and the difference between them is obtained.

In the last step – the differintegration of both the exact solution and the approximating polynomial can be done by means of the already mentioned analytical formulae for monomials (which SubIval uses). For a monomial ct_{loc}^k (with $c \in \mathbb{R}$ and $k \in \mathbb{Z}_+$) the differintegration in the interval $\Xi = [t_{\text{loc a}}, t_{\text{loc b}}]$ results in [42, 43]:

$$\mathrm{d}^{\alpha}_{\Xi} c t_{\text{loc}}^k = \frac{kc}{\Gamma(1-\alpha)} t_{\text{loc}}^{k-\alpha} \left(\mathrm{B}_{\frac{t_{\text{loc b}}}{t_{\text{loc}}}}(k, 1-\alpha) - \mathrm{B}_{\frac{t_{\text{loc a}}}{t_{\text{loc}}}}(k, 1-\alpha) \right), \tag{19}$$

where

$$\mathrm{B}_{\rho}(k, 1-\alpha) = \frac{\Gamma(k)\Gamma(1-\alpha)}{\Gamma(k+1-\alpha)} \left(1 - (1-\rho)^{1-\alpha} \sum_{j=0}^{k-1} \frac{\rho^j \prod\limits_{i=0}^{j-1} (1-\alpha+i)}{j!} \right). \tag{20}$$

In order to avoid human error – the tiresome derivations of the truncated form of $e_{\text{True } M}$ can be done by means of a selected computer algebra system allowing for symbolic computation. The SymPy Python library [44] has been used for this purpose. Initially the symbolic form of the truncated $e_{\text{True } M}$ has been studied for the polynomial orders q from 1 to 5 with $\Xi_M = \Theta_M$. The obtained formulae have the general form:

$$e_{\text{True } M} = \frac{\nu_q}{\Gamma(q+2-\alpha)} c_{q+1} \Delta t_{\text{now}}^{q+1-\alpha} + O(\Delta t_{\text{now}}^{q+2-\alpha}). \tag{21}$$

The ν_q coefficients obtained for q = 1, 2 and 3 are given in Table 1 (for higher polynomial orders the expressions are very long and therefore have not been presented in the paper). The obtained form suggests that the reduction of the step sizes leads to smaller errors as is the case when using BDF (backward differentiation formulae) for the derivative order $\alpha = 1$. Moreover, the above formula represents the BDF error for $\alpha = 1$.

In order to obtain an error estimation in a single time step one must compare at least two approximations. A two stage approach is proposed in the study where the first one (called stage A) applies a polynomial of order $q-1$ to obtain $\widetilde{x}_M(t)$, while the second stage (called stage B) applies a polynomial of order q. Obviously, the theoretically more accurate results of the stage B will be selected in the final solution, however the error estimation:

$$e_{\text{stages}} = \mathrm{d}^{\alpha}_{\Xi_{M\,\mathrm{A}}} \widetilde{x}_{M\,\mathrm{B}}(t_{\text{loc}}) - \mathrm{d}^{\alpha}_{\Xi_{M\,\mathrm{A}}} \widetilde{x}_{M\,\mathrm{A}}(t_{\text{loc}}) \tag{22}$$

will take into account a term lacking in stage A (the interval $\Xi_{M\,\mathrm{A}}$ is Ξ_M from stage A). In order to obtain a general formula for this term (and the remainder appearing as a consequence of terms not included in both stage approximations), like before, the SymPy library has been applied. The obtained general form is:

$$e_{\text{stages}} = \frac{\nu_{q-1}}{\Gamma(q+1-\alpha)} c_q \Delta t_{\text{now}}^{q-\alpha} + O(\Delta t_{\text{now}}^{q+1-\alpha}). \tag{23}$$

where a resemblance to $e_{\text{True } M}$ for the polynomial order $q-1$ is evident.

Table 1
ν_q coefficients of $e_{\text{True } M}$ for q = 1, 2 and 3

q	ν_q
1	α
2	$\alpha(\mu_3+1)^{2-\alpha}(\alpha\mu_3 - \mu_3 + 2)$
3	$\alpha(\mu_3\mu_4+\mu_4+1)^{2-\alpha}(\alpha^2\mu_3^2\mu_4^2 + \alpha^2\mu_3\mu_4^2$ $-3\alpha\mu_3^2\mu_4^2 - \alpha\mu_3\mu_4^2 + 4\alpha\mu_3\mu_4 + 2\alpha\mu_4^2$ $+2\alpha\mu_4 + 2\mu_3^2\mu_4^2 - 4\mu_3\mu_4 - 2\mu_4^2 + 4\mu_4 + 6)$

4. Step size adaptivity

For the considerations in this section an error approximation is assumed by considering only the first term of e_{stages}:

$$e = \frac{\nu_{q-1}}{\Gamma(q+1-\alpha)} c_q \Delta t_{\text{now}}^{q-\alpha}. \tag{24}$$

Assuming e_{ctrl} represents an acceptable error then the step size for which e takes this value is given by:

$$\Delta t_{\text{new}} = \sqrt[q-\alpha]{\frac{e_{\text{ctrl}}}{c}}, \tag{25}$$

where for simplicity the c coefficient has been introduced:

$$c = \frac{\nu_{q-1} c_q}{\Gamma(q+1-\alpha)}. \tag{26}$$

The denominator of (25) can be obtained from the relation between the computed error estimation e and the step size Δt_{now} applied at that moment:

$$\sqrt[q-\alpha]{c} = \frac{\sqrt[q-\alpha]{e}}{\Delta t_{\text{now}}}, \tag{27}$$

which leads to the relation:

$$\eta = \frac{\Delta t_{\text{new}}}{\Delta t_{\text{now}}} = \sqrt[q-\alpha]{\frac{e_{\text{ctrl}}}{e}}, \tag{28}$$

meaning that an applied step size Δt_{now} should be multiplied by η in order for the error to tend to e_{ctrl}.

A few remarks can be made at this stage of the analysis:

- the obtained error estimation e appears in the denominator, hence if the obtained value is 0 then η can be set to 1, thus not modifying the step size,
- in some cases when a user sets e_{ctrl} too low the SubIval solver could take a great amount of time to obtain the solution, hence, for extreme cases a minimum step size value $\Delta t_{\min}$ can be set,
- for very small error values, a rational approach could be not to greatly increase the step size (even for very small errors, as that might change), hence, a maximum step size $\Delta t_{\max}$ can also be set,
- not knowing the scale of a variable it is better to compute the errors as relative values (also, e_{ctrl} would be given in percentages),
- for several variables under fractional derivatives (state variables) the smallest η can be selected, following a pessimistic approach,
- the error estimation is performed after an approximation of the derivative (and the solution) had been obtained, the time step size modification is, hence, performed for the next step size; however, if the obtained error value is too large (exceeding a previously set parameter $e_{\max}$) then the time step should be repeated with a new Δt_{now} which leads to a new time instance t_{now}.

When a new time instance t_{now} is selected then a new set of parameters a, b and b_M can be obtained for each variable in both stages. In stage A one can apply a and b_M to obtain:

$$d_{\text{A}} = \mathrm{d}^{\alpha}_{\Xi_{M\,\text{A}}} \widetilde{x}_{M\,\text{A}}(t_{\text{loc}}) \tag{29}$$

appearing in (22). However, in stage B the coefficients a and b_M are obtained from a differintegration in a new interval $\Xi_M \neq \Xi_{M\,\text{A}}$, hence an additional differintegration must be performed to obtain the term:

$$d_{\text{B}} = \mathrm{d}^{\alpha}_{\Xi_{M\,\text{A}}} \widetilde{x}_{M\,\text{B}}(t_{\text{loc}}). \tag{30}$$

In practice, the approximations of the derivative and the error estimation that follows can be performed using various strategies, which have been explained in Sec. 6.

5. Considered class of problems

The problems for which the local truncation error estimation is considered can be described by the following system of equations:

$$\begin{cases} \boldsymbol{M}_{\text{I}}\boldsymbol{y}(t) + \boldsymbol{M}_{\text{II}}\boldsymbol{x}(t) = \boldsymbol{T}\boldsymbol{v}(t), \\ \mathbf{d}^{\boldsymbol{\alpha}}_{\Xi}\boldsymbol{x}(t) + \boldsymbol{M}_{\text{III}}\boldsymbol{y}(t) + \boldsymbol{M}_{\text{IV}}\boldsymbol{x}(t) = \mathbf{0}_{n_x}, \end{cases} \tag{31}$$

where

- $\boldsymbol{x}(t)$ is the vector of state variables (its length is denoted by n_x),
- $\boldsymbol{y}(t)$ is a vector of the remaining variables introduced to the solution (the length of the vector is denoted by n_y),
- $\boldsymbol{v}(t)$ contains the source time functions (the vector length is given by n_v),
- $\mathbf{d}^{\boldsymbol{\alpha}}_{\Xi}\boldsymbol{x}(t)$ is a vector of fractional derivatives of the state variables:

$$\mathbf{d}^{\boldsymbol{\alpha}}_{\Xi}\boldsymbol{x}(t) = [\ \mathrm{d}^{\alpha_1}_{\Xi} x_1(t)\ \ \mathrm{d}^{\alpha_2}_{\Xi} x_2(t)\ \ \ldots\ \ \mathrm{d}^{\alpha_{n_x}}_{\Xi} x_{n_x}(t)\ ]^{\text{T}} \tag{32}$$

  with $\boldsymbol{\alpha}$ representing a vector of the derivative orders $\alpha_1, \alpha_2, \ldots \alpha_{n_x}$; with the interval $\Xi = [t_0, t]$,
- $\mathbf{0}_{n_x}$ is a vector of n_x zeros,
- $\boldsymbol{M}_{\text{I}}$ is an $n_y \times n_y$ matrix,
- $\boldsymbol{M}_{\text{II}}$ is an $n_y \times n_x$ matrix,
- $\boldsymbol{M}_{\text{III}}$ is an $n_x \times n_y$ matrix,
- $\boldsymbol{M}_{\text{IV}}$ is an $n_x \times n_x$ matrix,
- $\boldsymbol{T}$ is an $n_y \times n_v$ matrix,

Also, for further consideration, the vector:

$$\boldsymbol{w}(t) = [\ \boldsymbol{y}(t)\quad \boldsymbol{x}(t)\ ]^{\text{T}} \tag{33}$$

is defined, containing all the variables placed in the solution.

For each stage of the time stepping solver one can obtain the a and b coefficient pair. All of the a coefficients can be placed in a vector with indices corresponding to those of the state variables:

$$\boldsymbol{a} = [\ a_1\quad a_2\quad \ldots\quad a_{n_x}\ ]^{\text{T}}. \tag{34}$$

The same can be done for the b coefficients:

$$\boldsymbol{b} = [\ b_1\quad b_2\quad \ldots\quad b_{n_x}\ ]^{\text{T}}. \tag{35}$$

By applying the SubIval approximation (7) at the time instance $t = t_{\text{now}}$ one obtains the system of equations:

$$\begin{bmatrix} \boldsymbol{M}_{\text{I}} & \boldsymbol{M}_{\text{II}} \\ \boldsymbol{M}_{\text{III}} & \boldsymbol{M}_{\text{IV}} + \operatorname{diag}(\boldsymbol{a}) \end{bmatrix} \boldsymbol{w} = \begin{bmatrix} \boldsymbol{T}\boldsymbol{v}(t_{\text{now}}) \\ -\boldsymbol{b} \end{bmatrix}, \tag{36}$$

with $\boldsymbol{w}$ being the numerical solution for $\boldsymbol{w}(t_{\text{now}})$.

6. Error computation strategies

The most obvious approach to the error estimation is to take the following steps:

(a) compute a and b for both stages, along with b_M for stage A, while for stage B compute the additional coefficients:

$$a_{M\,\text{A}} = \mathrm{d}^{\alpha}_{\Xi_{M\,\text{A}}} L_{M,n_M}(t) \tag{37}$$

and

$$b_{M\,\text{A}} = \mathrm{d}^{\alpha}_{\Xi_{M\,\text{A}}} \sum_{j=1}^{n_M-1} x_{M,j} L_{M,j}(t), \tag{38}$$

(b) compute the solution of (36) at both stages (for a selected state variable x the solution is further on denoted by $x_{\text{now A}}$ for stage A and $x_{\text{now B}}$ for stage B),

(c) obtain d_{A} mentioned in (29) through:

$$d_{\text{A}} = a x_{\text{now A}} + b_M, \tag{39}$$

and d_{B} mentioned in (30) through:

$$d_{\text{B}} = a_{M\,\text{A}} x_{\text{now B}} + b_{M\,\text{A}}, \tag{40}$$

(d) finally, a relative difference between d_{A} and d_{B} is taken into account to evaluate the error estimation e (this is done for each variable).

The above is further on referred to as the d approach.

A slightly modified approach would be to compute step (a) but only obtain the solution for stage B. d_{A} would be then computed through (39) using $x_{\mathrm{now\ B}}$ instead of $x_{\mathrm{now\ A}}$. d_{B} is still computed using (40). This simplified approach is further on called the d_{s} approach.

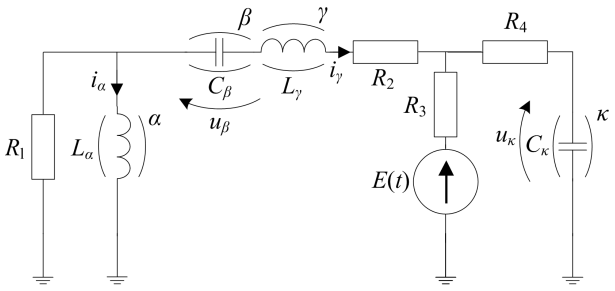
Another approach is tested in the study (being an extension to the d-based approach), where in addition to the differences between the variable approximations – selected variables from the solution vector $\boldsymbol{w}$ are compared for both stages. This is further on called the w approach.

Finally, an approach is tested, which also adds error computations of $\boldsymbol{w}$, but for the derivative error computations – the component d_{A} is obtained from the solution of stage B (like in the d_{s} approach). This is further on called the w_{s} approach.

7. Computational examples

Two circuit examples have been selected, which can be described by the system of equations given by (31). For the selected examples one can provide referential solutions that can be obtained through completely different methods (not involving the approximation of fractional derivatives).

7.1. Circuit in periodic steady-state. The first considered example is presented in Fig. 1. The problem has a relatively easily obtainable referential steady-state solution using a complex number approach. As it is a linear problem then for the source frequencies $f = 50, 150$ Hz the solutions can be obtained separately and then added using the superposition principle.



$E(t) = E_1 \sin(\omega t) + E_3 \sin(3\omega t)$
$R_1 = 20\,\Omega$
$R_2 = 200\,\Omega$
$R_3 = 50\,\Omega$
$R_4 = 200\,\Omega$
$E_1 = 325$ V
$E_3 = 60$ V
$\omega = 2\pi f$
$f = 50$ Hz

$\alpha = 0.9$
$L_\alpha = 20\,\mu\mathrm{H}\cdot\mathrm{s}^{\alpha-1}$
$\gamma = 0.8$
$L_\gamma = 5\,\mu\mathrm{H}\cdot\mathrm{s}^{\gamma-1}$
$\beta = 0.9$
$C_\beta = 0.25\,\mathrm{nF}\cdot\mathrm{s}^{\beta-1}$
$\kappa = 0.2$
$C_\kappa = 10\,\mathrm{F}\cdot\mathrm{s}^{\kappa-1}$

Fig. 1. Circuit with periodic excitation and fractional coils and capacitors (each fractional element is marked by parentheses and the order of the element)

For a single frequency a complex number approach can be applied where one solves the system:

$$\begin{bmatrix} \boldsymbol{M}_{\mathrm{I}} & \boldsymbol{M}_{\mathrm{II}} \\ \boldsymbol{M}_{\mathrm{III}} & \boldsymbol{M}_{\mathrm{IV}} + \mathrm{diag}(\underline{\boldsymbol{s}}) \end{bmatrix} \underline{\boldsymbol{w}} = \begin{bmatrix} \boldsymbol{T}\underline{\boldsymbol{v}} \\ \boldsymbol{0}_{n_x} \end{bmatrix}, \tag{41}$$

with

- $\underline{\boldsymbol{w}}$ being a vector of the complex number representations of the variables in $w(t)$ (for the considered frequency),
- $\underline{\boldsymbol{v}}$ is the complex number representation of the source vector for the considered frequency,
- $\underline{\boldsymbol{s}}$ results from the periodic steady-state differintegration and is given by:

$$\underline{\boldsymbol{s}} = [\ \underline{\mathrm{j}}_{\alpha_1}\omega^{\alpha_1} \quad \underline{\mathrm{j}}_{\alpha_2}\omega^{\alpha_2} \quad \dots \quad \underline{\mathrm{j}}_{\alpha_{n_x}}\omega^{\alpha_{n_x}}\ ]^{\mathrm{T}}, \tag{42}$$

where for the indices $i = 1, 2, \dots n_x$:

$$\underline{\mathrm{j}}_{\alpha_i} = \exp\left(\mathrm{j}\alpha_i \frac{\pi}{2}\right). \tag{43}$$

The options for the numerical solver are given in Table 2.

Table 2
Parameters for the numerical computations of the SubIval solver. p_{mov} is the maximum value of q (being the polynomial order of $\tilde{x}_M$ in stage B)

parameter name	p_{mov}	$e_{\max}$	e_{ctrl}	$\Delta t_{\min}$	$\Delta t_{\max}$
value	4	$0.1\,\%$	$10^{-2}\,\%$	$T/20$	$T/10^3$

The SubIval solver allows to obtain a transient solution. For each period $t \in [(k-1)T, kT]$ the value has been evaluated:

$$e_{\mathrm{steady}} = \max_{i=1,2,\dots n_w} \frac{100}{T} \frac{\left| \int_{t=(k-1)T}^{kT} w_i(t)\,\mathrm{d}t \right|}{w_{i\ \mathrm{abs\ max}}}\ \% \tag{44}$$

representing a measure determining whether the circuit is in a steady-state ($w_i(t)$ is the i-th variable in the $\boldsymbol{w}(t)$ vector and $w_{i\ \mathrm{abs\ max}}$ is the maximum of absolute values of $w_i(t)$ in the considered period). The transient solution has been obtained for $n_T = 20$ periods, which was enough for e_{steady} to become lower than $10^{-3}\%$.

A comparison between the results (for the first and last periods) obtained through the SubIval solver and the referential (periodic steady-state) solution is given in Fig. 2. The variables selected for the comparison are the state variables:

$$\boldsymbol{x}(t) = [\ i_\alpha \quad i_\gamma \quad u_\beta \quad u_\kappa\ ]^{\mathrm{T}}. \tag{45}$$

So far one can observe that the numerical solver is performing properly as the results are very close in the steady-state. An analysis of both the estimated and actual error of the derivative approximation (for the discussed example) is presented in Sec. 8. The results depicted in the figure have been obtained with an application of the d_{s} approach. However, the other approaches yielded results that are not visibly different.

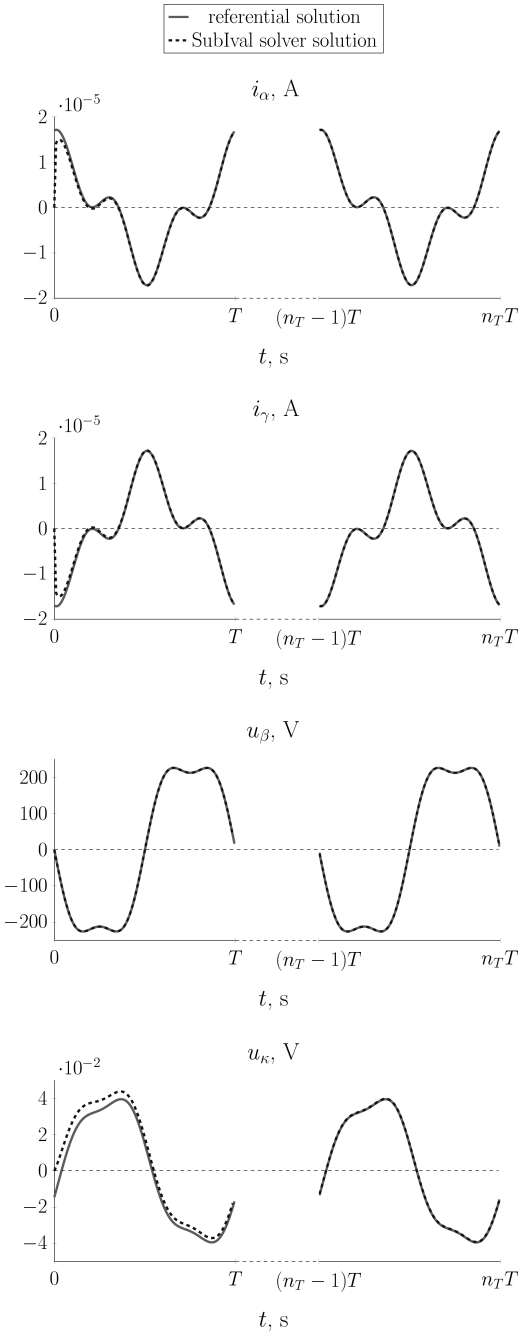


Fig. 2. Comparison of time functions obtained by the SubIval solver and those of the referential (periodic steady-state) solution

7.2. Transient example. The second example that has been studied is one concerning the transient response of the parallel RLC circuit presented in Fig. 3.

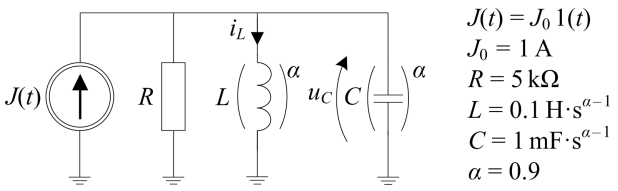


Fig. 3. Circuit with unit step excitation (each fractional element is marked by parentheses and the order of the element)

For this simple example it is possible to obtain a referential analytical solution. It can be obtained similarly as in [45], where a fractional series RLC circuit was studied. According to this solution – the capacitor voltage u_C is:

$$u_C(t) = J_0 \frac{\lambda_1 \mathrm{E}_{\alpha,\alpha+1}(\lambda_1 t^\alpha) - \lambda_2 \mathrm{E}_{\alpha,\alpha+1}(\lambda_2 t^\alpha)}{C(\lambda_1 - \lambda_2)} t_\alpha, \quad (46)$$

while the current through the coil i_L is given by:

$$i_L(t) = J_0 \frac{\mathrm{E}_{\alpha,\alpha+1}(\lambda_1 t^\alpha) - \mathrm{E}_{\alpha,\alpha+1}(\lambda_2 t^\alpha)}{2LC\sqrt{\Delta}} t_\alpha, \quad (47)$$

where

$$\Delta = \frac{1}{4R^2L^2} - \frac{1}{LC}, \quad (48)$$

while

$$\lambda_1 = -\frac{1}{2RL} + \sqrt{\Delta} \quad (49)$$

and

$$\lambda_2 = -\frac{1}{2RL} - \sqrt{\Delta}. \quad (50)$$

In (46) and (47) the notation:

$$\mathrm{E}_{\alpha,\beta}(z) = \sum_{k=0}^{\infty} \frac{z^k}{\Gamma(\alpha k + \beta)} \quad (51)$$

is used for the two parameter Mittag-Leffler function. The solution has been obtained for the interval $t \in [0, T_{\max}]$, where $T_{\max} = 0.14$ s was selected. A comparison of the results obtained through the SubIval solver and through evaluations of the analytical solution is presented in Fig. 4.

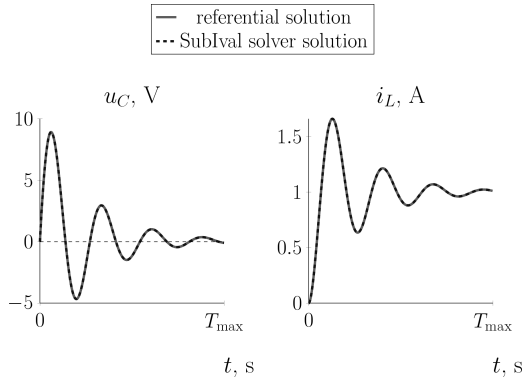


Fig. 4. Comparison of time functions obtained by the SubIval solver and those of the referential solution

For this example one can also notice that the results obtained through both methods are very close. Again the d_s approach has been applied to obtain the depicted results. However, as it was the case with the previous example, the results obtained when the other error computation approaches have been applied are not visibly different than those presented.

8. Error analysis

In the SubIval solver the error estimation is computed for each fractional derivative, which for simplicity can denoted by:

$$d_i(t) = \mathrm{d}^\alpha_\Xi x_i(t). \quad (52)$$

A measure of the actual accuracy of the SubIval solver is then computed by means of the following formula for each d_i:

$$e_i = 100 \cdot \frac{|d_i(t_j) - d_{i\ \mathrm{num}\ j}|}{\max\limits_{j=1,2,\ldots n_t} |d_i(t_j)|}\ \%, \tag{53}$$

where $t_1, t_2, \ldots\ t_{n_t}$ are the nodes on the time axis that have been selected by the SubIval solver, $d_i(t_j)$ is the value obtained for t_j by means of the referential solution and $d_{i\ \mathrm{num}\ j}$ is the value of $d_i(t_j)$ obtained by the SubIval solver.

A comparison of the maximum of errors (for each selected time instance) computed through the above formula with the maximum of estimated error values is depicted in Fig. 5 for the steady-state periodic example discussed in Subsec. 7.1. The figure presents results for each of the considered error computation approaches given in Sec. 6.

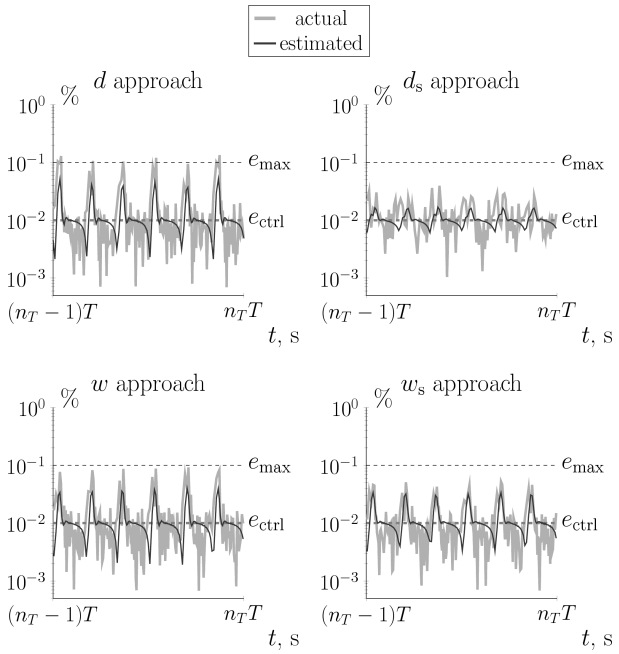


Fig. 5. Comparison of the maximum of actual and estimated error values for the periodic steady-state example

There are differences between the maximum values of the estimated and actual error. The pattern is, however, similar and both errors tend to a value near that of the selected e_{ctrl} (where, for both examples, the value 10^{-2} % was selected for this parameter). This can be observed in Table 3, where the average values of the actual and estimated errors are given for each error estimation approach.

Table 3
Comparison of the average of actual and estimated error values (i.e. maximums for all variables) for both considered examples (values are rounded to 3 significant digits, all values are in percentages)

approach	periodic example		transient example	
	actual	estimated	actual	estimated
d	$1.59 \cdot 10^{-2}$	$1.14 \cdot 10^{-2}$	$4.11 \cdot 10^{-2}$	$1.02 \cdot 10^{-2}$
d_{s}	$1.24 \cdot 10^{-2}$	$1.02 \cdot 10^{-2}$	$7.09 \cdot 10^{-3}$	$1.14 \cdot 10^{-2}$
w	$1.34 \cdot 10^{-2}$	$1.14 \cdot 10^{-2}$	$4.05 \cdot 10^{-2}$	$1.01 \cdot 10^{-2}$
w_{s}	$1.13 \cdot 10^{-2}$	$1.08 \cdot 10^{-2}$	$7.09 \cdot 10^{-3}$	$1.14 \cdot 10^{-2}$

The d_{s} approach seems to yield the best results as the error is kept closest to e_{ctrl} throughout the solving process. The w_{s} approach works similarly, with no surprise as it also takes into account the same error values (though adding its own – basing on other variables, as explained previously in Sec. 6.

Figure 6 displays the maximum values of the actual and estimated error for the transient example of Subsec. 7.2.

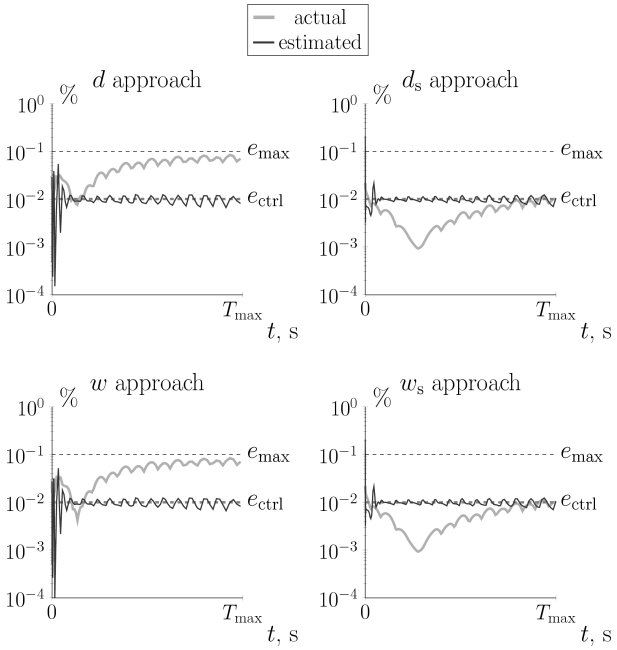


Fig. 6. Comparison of the maximum of actual and estimated error values for the transient circuit example

This time one can clearly see an advantage of the d_{s} and w_{s} approaches. It can also be observed in the average values of the error in Table 3.

9. Remarks on computation tools

The computations of SubIval and the adaptive step size solver have been performed with own C# programs applying a DLL available at [46]. The library uses a part of the code given in [47] for the computation of the gamma function.

The SubIval step size adaptive solver required a system of equations to be solved for each iteration. In these computations classes from the MathNet Numerics library [48] have been applied for matrix and vector operations.

10. SubIval in comparison to other methods

SubIval is a method that can be applied in time stepping solvers for the approximation of the fractional derivative in the Caputo definition of order $\alpha \in (0, 1]$. A comparison has been performed with methods also belonging to this category. The methods can also be those that can be applied for the Riemann-Liouville fractional derivative as for the examples used in this paper zero initial conditions are applied.

There are not many methods whose implementations are available so that one can build a FDAE (fractional differential algebraic equations) solver upon them.

Some rare cases of solvers applicable in Matlab [49] (or GNU Octave [50]) are described in [51, 52], which can be found at [53]. They are based on PI (product integration) rule methods and FLMM (fractional linear multistep methods). The FLMM solvers have been implemented only for the case of a common fractional derivative order, hence these solvers cannot be applied for the periodic example. The available solvers are based on the following methods (the abbreviations in parentheses are for further reference):

- explicit product integration of rectangular type (PI1_Ex solver),
- implicit product integration of rectangular type (PI1_Im solver),
- implicit product integration of trapezoidal type (PI2_Im solver),
- product integration with predictor-corrector (PI12_PC solver),
- FLMM applying trapezoidal rule integration (FLMM_T solver),
- FLMM applying the Newton-Gregory formula for integration (FLMM_NG solver),
- FLMM applying the second order backward differentiation formula (FLMM_BDF2 solver).

Another possibility is the implementation of a method basing on the Grünwald-Letnikov approximation [54]. The Grünwald-Letnikov definition is equivalent with the Riemann-Liouville definition [55]. For the Caputo derivative it can be described by:

$$ {}_{t_0}\mathrm{D}_t^{\alpha} x(t) \approx \frac{1}{\Delta t^{\alpha}} \sum_{j=0}^{j=N} (-1)^j \frac{\Gamma(\alpha+1)}{j!\Gamma(\alpha-j+1)} x(t-j\Delta t) - \frac{x(t_0)(t-t_0)^{-\alpha}}{\Gamma(1-\alpha)}, \tag{54} $$

with N being the index of the current time step. The solver basing on this method has been implemented as an implicit solver in GNU Octave. It is further on called the GL solver.

Both of the computational examples of Section 7 have been solved with the PI solvers, the FLMM solvers and the GL solver. As their implementations are performed in a Computer Algebra System (as opposed to the SubIval solver, which has been written in C#) it will take much longer for them to complete the computations, hence the computation times are not compared. What can be reliably compared is the accuracy of the results. The referential solutions have been implemented in GNU Octave – an auxiliary function [56, 57] has been used for the computations of the Mittag-Leffler function in (46) and (47).

In order to verify the result, error computations have been performed just like for the SubIval solver, i.e. applying (53). For all state variables, the average of errors for all time instances have been computed. The maximum of those values, for all solvers and both examples, is given in Table 4.

The mentioned solvers are only implemented for constant step sizes, hence in all cases the minimum step size applied for the SubIval solver $\Delta t_{\min}$ has been applied initially and further tests have been performed to observe if an increase of this value resulted in an increase in accuracy. Ultimately, the lowest error has been recorded. In the case of SubIval – the results are presented for the time step size adaptive solver with the d_{s} approach.

Table 4
Solver comparison: average error values (maximums for all variables) for both considered examples (*the solvers suffered from stability issues)

solver	periodic example	transient example
SubIval solver	$1.24 \cdot 10^{-2}$	$7.09 \cdot 10^{-3}$
PI1_Ex solver	failed*	1.29
PI1_Im solver	$8.02 \cdot 10^{-1}$	$1.34 \cdot 10^{-1}$
PI2_Im solver	$4.66 \cdot 10^{-2}$	$1.10 \cdot 10^{-1}$
PI12_PC solver	failed*	$1.10 \cdot 10^{-1}$
FLMM_T solver	–	$1.11 \cdot 10^{-1}$
FLMM_NG solver	–	$1.11 \cdot 10^{-1}$
FLMM_BDF2 solver	–	$1.08 \cdot 10^{-1}$
GL solver	$7.23 \cdot 10^{-1}$	$1.24 \cdot 10^{-1}$

The following observations can be made:

- the most accurate results have been obtained through SubIval solver,
- as for the other solvers – in general the best performance has been observed for the PI2_Im solver,
- the explicit solvers failed to properly solve the periodic example (stability issues have been observed).

11. Summary and concluding remarks

The local truncation error estimation for a solver based on SubIval has been studied.

The details on the core computations of SubIval have been first presented (Sec. 2) and an approximate formula for a local truncation error has been derived (Sec. 3).

Then the step size adaptivity in the SubIval solver has been discussed (Sec. 4).

A class of linear problems has been defined (Sec. 5) for which the research of this paper has been conducted.

For the error formula – various approaches have been proposed (Sec. 6) for the computation of the estimated error.

Two circuit problems with known referential solutions have been introduced (Subsec. 7.1 and 7.2 respectively) to check the correctness of the error estimation. The error estimation results (in comparison to the actual differences between the numerical and referential solutions) have been presented in Sec. 8.

A few additional remarks can be added as to the error estimation results. All strategies for the error estimation performed similarly – keeping the error within range of the e_{ctrl} value for the periodic steady-state example, while the d_{s} and w_{s} approaches performed significantly better for the transient example. This could be caused by the fact that the local truncation error formula is actually derived from the beginning with an assumption of errors omitted for the values computed at each of the time instances. The d_{s} and w_{s} approaches do take the same solutions for the computations of the fractional

derivatives in both stages (called stage A and stage B), which could be the reason for the more accurate resemblance of the actual error when using these approaches.

One source of inaccuracies in the error estimation that is worth mentioning is the fact that the step size is always adapted using results from preceding time steps (unless the time step is repeated, although such an occurrence was not necessary as the computed error values were all below $e_{\max}$).

Another source of inaccuracies is that the estimation is done by computing the differintegrals for Ξ_M and the actual error has been obtained for the entire interval.

Judging by the results presented in the paper – the d_{s} approach should be used for the considered class of problems as it yields the best results and requires only one solution to be computed at each time step Additionally, no advantages have been observed when adding errors of selected variables in $\boldsymbol{w}(t)$ (which resulted in the w_{s} approach).

The w_{s} approach could provide a safeguard in nonlinear problems if the derivative error causes much greater errors for other variables. This, however, is a topic reserved for future studies.

REFERENCES

[1] K.B. Oldham and J. Spanier, *The Fractional Calculus*, Academic Press, New York (1974).

[2] T. Kaczorek and K. Borawski, "Positive stable realization problem for linear continuous-time fractional-order systems with symmetric system Metzler matrix", *PAK* 60(10), 822–825 (2014).

[3] R. Brociek, D. Słota, and R. Wituła, "Reconstruction Robin Boundary Condition in the Heat Conduction Inverse Problem of Fractional Order", *Theory and Applications of Non-Integer Order Systems*, Springer, 147–156 (2017).

[4] D. Sierociuk, T. Skovranek, M. Macias, I. Podlubny, I. Petras, A. Dzielinski, and P. Ziubinski, "Diffusion process modeling by using fractional-order models", *Applied Mathematics and Computation* 257, 2–11 (2015).

[5] S. Kukla and U. Siedlecka, "An analytical solution to the problem of time-fractional heat conduction in a composite sphere", *Bull. Pol. Ac.: Tech.* 65(2), 179–186 (2017).

[6] W. Bauer and A. Kawala-Janik, "Implementation of Bi-fractional Filtering on the Arduino Uno Hardware Platform", *Theory and Applications of Non-Integer Order Systems*, Springer, 419–428 (2017).

[7] A. Kawala-Janik, M. Podpora, A. Gardecki, W. Czuczwara, J. Baranowski, and W. Bauer, "Game controller based on biomedical signals", *Methods and Models in Automation and Robotics (MMAR) 2015 20th International Conference*, 934–939 (2015).

[8] P.W. Ostalczyk, P. Duch, D.W. Brzeziński, and D. Sankowski, "Order Functions Selection in the Variable-, Fractional-Order PID Controller", *Advances in Modelling and Control of Non-integer-Order Systems*, Springer, 159–170 (2015).

[9] J. Baranowski, W. Bauer, M. Zagórowska, A. Kawala-Janik, T. Dziwiński, and P. Piątek, "Adaptive Non-Integer Controller for Water Tank System", *Theoretical Developments and Applications of Non-Integer Order Systems*, Springer, 271–280 (2016).

[10] D. Spałek, "Synchronous generator model with fractional order voltage regulator $\mathrm{PI}^b\mathrm{D}^a$", *Acta Energetica* 2/23, 78–84 (2015).

[11] P. Mercorelli, "A Discrete-Time Fractional Order PI Controller for a Three Phase Synchronous Motor Using an Optimal Loop Shaping Approach", *Theory and Applications of Non-Integer Order Systems*, Springer, 477–487 (2017).

[12] R. Garrappa and G. Maione, "Fractional Prabhakar derivative and applications in anomalous dielectrics: a numerical approach", *Theory and Applications of Non-Integer Order Systems*, Springer, 429–439 (2017).

[13] L. Mescia, P. Bia, and D. Caratelli, "Fractional derivative based FDTD modeling of transient wave propagation in Havriliak-Negami media", *IEEE Transactions on Microwave Theory and Techniques* 62(9), 1920–1929 (2014).

[14] G. Litak, B. Ducharne, G. Sebald, and D. Guyomar, "Dynamics of magnetic field penetration into soft ferromagnets", *Journal of Applied Physics* 117(24), 243907 (2015).

[15] A. Babiarz, A. Łęgowski, and M. Niezabitowski, "Robot path control with Al-Alaoui rule for fractional calculus discretization", *Theory and Applications of Non-Integer Order Systems*, Springer, 405–418 (2017).

[16] W. Sumelka, "Fractional calculus for continuum mechanics – anisotropic non-locality", *Bull. Pol. Ac.: Tech.* 64(2), 361–372 (2017).

[17] M. Faraji Oskouie and R. Ansari, "Linear and nonlinear vibrations of fractional viscoelastic Timoshenko nanobeams considering surface energy effects", *Applied Mathematical Modelling* 43, 337–350 (2017).

[18] M.A. Ezzat, A.S. El-Karamany, and A.A. El-Bary, "Thermo-viscoelastic materials with fractional relaxation operators", *Applied Mathematical Modelling* 39(23–24), 7499–7512 (2015).

[19] W. Mitkowski and P. Skruch, "Fractional-order models of the supercapacitors in the form of RC ladder networks", *Bull. Pol. Ac.: Tech.* 61(3), 580–587 (2013).

[20] J. Walczak and A. Jakubowska, "Analysis of parallel resonance RLC_α circuit with supercapacitor", *Computer Applications in Electrical Engineering* 12, 30–37 (2014).

[21] A. Jakubowska-Ciszek and J. Walczak, "Analysis of the transient state in a parallel circuit of the class $RL_\beta C_\alpha$", *Applied Mathematics and Computation* 319, 287–300 (2018).

[22] K.J. Latawiec, R. Stanisławski, M. Łukaniszyn, W. Czuczwara, and M. Rydel, "Fractional-order modeling of electric circuits: modern empiricism vs. classical science", *Progress in Applied Engineering (PAEE) 2017*, 1–4 (2017).

[23] I. Schäfer and K. Krüger, "Modelling of lossy coils using fractional derivatives", *Phys. D: Appl. Phys.* 41, 1–8 (2008).

[24] A. King and F.T. Agerkvist, "State-space modeling of loudspeakers using fractional derivatives", *Audio Engineering Society Convention 139*, Audio Engineering Society (2015).

[25] J.F. Gómez Aguilar and M.M. Hernández: "Space-time fractional diffusion-advection equation with Caputo derivative", *Abstract and Applied Analysis*, 8 p. (2014).

[26] M. Caputo, "Linear models of dissipation whose Q is almost frequency independent – II", *Geophysical Journal International* 13(5), 529–539 (1967).

[27] J.D. Munkhammar, "Riemann-Liouville fractional derivatives and the Taylor-Riemann series", *UUDM Project Report* 7, 1–18 (2004).

[28] T. Abdeljawad, "On Riemann and Caputo fractional differences", *Computers and Mathematics with Applications* 62, 1602–1611 (2011).

[29] T. Kaczorek, "Positivity of a class of fractional descriptor continuous-time nonlinear systems", *Bull. Pol. Ac.: Tech.* 65(2), 561–565 (2015).

[30] A. Jakubowska and J. Walczak, "Analysis of the transient state in a circuit with supercapacitor", *Poznan University of Technology Academic Journals. Electrical Engineering* 81, 27–34 (2015).
[31] N.S. Khodabakhshi, S.M. Vaezpour, and D. Baleanu, "Numerical solutions of the initial value problem for fractional differential equations by modification of the Adomian decomposition method", *Fractional Calculus and Applied Analysis* 17(2), 382–400 (2014).
[32] S. Momani and M.A. Noor, "Numerical methods for fourth order fractional integro-differential equations", *Appl. Math. Comput.* 182, 754–760 (2006).
[33] A. Arikoglu and I. Ozkol, "Solution of fractional differential equations by using differential transform method", *Chaos, Solitons and Fractals* 34, 1473–1481 (2007).
[34] C. Lubich, "Fractional linear multistep methods for Abel-Volterra integral equations of the second kind", *Math. Comput.* 45, 463–469 (1985).
[35] R. Garrappa, "On accurate product integration rules for linear fractional differential equations", *Journal of Computational and Applied Mathematics* 235, 1085–1097 (2011).
[36] W.-H. Luo, T.-Z. Huang, G.-C. Wu, and X.-M. Gu, "Quadratic spline collocation method for the time fractional subdiffusion equation", *Applied Mathematics and Computation* 276, 252–265 (2016).
[37] A. Pirkhedri and H.H.S. Javadi, "Solving the time-fractional diffusion equation via Sinc–Haar collocation method", *Applied Mathematics and Computation* 257, 317–326 (2015).
[38] D.W. Brzeziński and P. Ostalczyk, "High-accuracy numerical integration methods for fractional order derivatives and integrals computations", *Bull. Pol. Ac.: Tech.* 62(4), 723–733 (2014).
[39] M. Sowa, "A subinterval-based method for circuits with fractional order elements", *Bull. Pol. Ac.: Tech.* 62(3), 449–454 (2014).
[40] M. Sowa, "Application of SubIval in solving initial value problems with fractional derivatives", *Applied Mathematics and Computation* 319, 86–103 (2018).
[41] V. Dolejší and P. Kůs, "Adaptive backward difference formula–Discontinuous Galerkin finite element method for the solution of conservation laws" *Int. J. Numer. Meth. Engng* 73, 1739–1766 (2008).
[42] M. Sowa, "Application of SubIval, a Method for Fractional-Order Derivative Computations in IVPs", *Theory and Applications of Non-Integer Order Systems*, Springer, 405–418 (2017).
[43] http://functions.wolfram.com/GammaBetaErf/Beta3/03/01/02/0003/
[44] http://www.sympy.org
[45] M. Włodarczyk and A. Zawadzki: "Obwody RLC w aspekcie pochodnych niecałkowitych rzędów", *Prace Naukowe Politechniki Śląskiej* 1, 75–88 (2011).
[46] http://msowascience.com
[47] J.D. Cook, "C# code for gamma and log gamma", http://www.johndcook.com/Gamma.cs
[48] http://numerics.mathdotnet.com
[49] https://www.mathworks.com/products/matlab.html
[50] https://www.gnu.org/software/octave/
[51] R. Garrappa, "Numerical Solutions of Fractional Differential Equations: A Survey and a Software Tutorial", *Mathematics* 6(2), 16 p. (2018).
[52] R. Garrappa, "Trapezoidal methods for fractional differential equations: Theoretical and computational aspects", *Mathematics and Computers in Simulation* 110, 96–112 (2015).
[53] R. Garrappa, "Software for Fractional Differential Equations and related problems", https://www.dm.uniba.it/Members/garrappa/Software.
[54] R. Scherer, S.L. Kalla, Y. Tang, and J. Huang, "The Grünwald-Letnikov method for fractional differential equations", *Computers and Mathematics with Applications* 62, 902–917 (2011).
[55] I. Podlubny, *Fractional differential equations*, San Diego, Academic Press (1999).
[56] R. Garrappa, "Numerical evaluation of two and three parameter Mittag-Leffler functions", *SIAM J. Numer. Anal.* 53(3), 1350–1369 (2015).
[57] http://www.mathworks.com/matlabcentral/fileexchange/48154-the-mittag-leffler-function.