Research Article

Pade Method for Construction of Numerical Algorithms for Fractional Initial Value Problem

Feng Gao and Chunmei Chi

School of Science, Qingdao University of Technology, Qingdao 266033, China

Correspondence should be addressed to Chunmei Chi; chunmei_chi@163.com

Received 11 April 2020; Accepted 5 June 2020; Published 8 July 2020

Academic Editor: Guotao Wang

Copyright © 2020 Feng Gao and Chunmei Chi. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In this paper, we propose an efficient method for constructing numerical algorithms for solving the fractional initial value problem by using the Pade approximation of fractional derivative operators. We regard the Grunwald–Letnikov fractional derivative as a kind of Taylor series and get the approximation equation of the Taylor series by Pade approximation. Based on the approximation equation, we construct the corresponding numerical algorithms for the fractional initial value problem. Finally, we use some examples to illustrate the applicability and efficiency of the proposed technique.

1. Introduction

In the past decades, fractional differential equations were successfully applied to many problems in engineering, physics, chemistry, biology, economics, control theory, biophysics, and so on [1–5]. It is significant to obtain the exact or numerical solutions of fractional nonlinear equations. Since most fractional differential equations do not have exact analytic solutions, numerical techniques and seminumerical methods are used extensively, such as the homotopy analysis and homotopy perturbation method [6–8], variational iteration method [9–11], orthogonal polynomial method [12, 13], fractional Adams method [14], and some other methods [15].

Among all the numerical methods, the direct numerical method [16–18] is the basic one. Due to the special property of the fractional derivative, most of these methods have their inbuilt deficiencies mostly due to the calculation of Adomian polynomials, the Lagrange multiplier, divergent results, and some other huge computational works.

In this paper, we propose an efficient method for constructing numerical algorithms for solving the fractional initial value problem by using the Pade approximation of fractional derivative operators. The advantage of the proposed technique is that efficient numerical methods can be constructed without calculation of long historical terms of the fractional derivative.

Consider the following homogeneous fractional initial problem:

\[
\begin{cases}
\sigma D^{(a)} y = f(y, t), & t \in [0, T], \\
y^{(k)}(0) = 0, & k = 0, 1, \ldots, m - 1,
\end{cases}
\]  

(1)

where \(\sigma D^{(a)}\) is usually the Caputo derivative and \(m = [a]\). Given the condition \(y^{(k)}(0) = 0, k = 0, 1, \ldots, m - 1\), the Caputo derivative is equivalent to the Grunwald–Letnikov derivative and the Riemann–Liouville derivative, e.g.,

\[
\sigma D^{(a)} = RL D^{(a)} = GL D^{(a)}. 
\]  

(2)

For the following linear nonhomogeneous fractional initial problem,

\[
\begin{cases}
\sigma D^{(a)} y = f(y, t), & t \in [0, T], \\
y^{(k)}(0) = y_k, & k = 0, 1, \ldots, m - 1,
\end{cases}
\]  

(3)

the following transform can be employed to make it be a homogeneous one with regard to \(z(t)\):

\[
y(t) = \sum_{k=0}^{m-1} y^{(k)}(0)t^k + z(t).
\]  

(4)
Based on the $G$ algorithm [19], we have the following numerical scheme for problem (1):
\[ h^{-a} \sum_{j=0}^{N} c_{nj}^{(a)} y_j = f(t_n, y_n), \quad n = 0, 1, \ldots, \left\lceil \frac{T}{h} \right\rceil, \] (5)
where $c_{nj}^{(a)}$ and $N$ are calculated according to different algorithms. Algorithm 5 is a technique which is easy to manipulate. However, this method involves huge computational work when $T \gg 0$ because in general, these methods need to compute many terms to get the approximation to the fractional derivative. To some extent, the short memory principle [20] can be used to tackle the problem, but low accuracy will be the cost. So, it is significant to find an efficient approximant to the fractional derivative, which is of low computation cost on the one hand and highly accurate on the other hand. In this paper, we put forward a reliable method for the construction of numerical algorithms for solving the fractional differential initial value problem by using Pade approximation to fractional derivative operators. The rest of the paper is organized as follows. In Section 2, we briefly list some basicsof Pade approximation. In Section 3, we get the approximant of the fractional derivative operators. In Section 4, we propose a method for constructing algorithms for solving the fractional initial value problem by using Pade approximation. In Section 5, we use some examples to illustrate the applicability and efficiency of the proposed technique. Section 6 is the conclusion.

2. Some Basics for Pade Approximation

In numerical mathematics, Pade approximation [21] is believed to be the best approximation of a function by rational functions of a given order. Under this technique, the approximant’s power series agrees with the power series of the function it is approximating. The Pade approximant often gives better approximation of the function than truncating its Taylor series, and it may still work where the Taylor series does not converge. For these reasons, Pade approximants are often used in many fields of computations.

Given a function $f(x)$ and two integers $m > 0$ and $n > 1$, the Pade approximant of order $[m, n]$ is the rational function
\[ R(x) = \frac{\sum_{j=0}^{m} a_j x^j}{1 + \sum_{k=0}^{n} b_k x^k}, \] (6)
which agrees with $f(x)$ to the highest possible order, which amounts to
\[ f(0) = R(0), \quad f'(0) = R'(0), \quad f''(0) = R''(0), \ldots, \quad f^{m+n}(0) = R^{m+n}(0). \] (7)

Equivalently, if $R(x)$ is expanded in a Maclaurin series, its first $m + n$ terms would cancel the first $m + n$ terms of $f(x)$, and as such $f(x) - R(x) = O(x^{m+n+1})$. The Pade approximant is unique for given $m$ and $n$, that is, the coefficients $a_0, a_1, \ldots, a_m, b_1, \ldots, b_n$ can be uniquely determined. It is for reasons of uniqueness that the zeroth-order term at the denominator of $R(x)$ was chosen to be 1; otherwise, the numerator and denominator of $R(x)$ would have been unique only up to multiplication by a constant. The Pade approximant defined above is also denoted as $[m, n]_{f(x)}$.

3. Pade Approximant for the Fractional Derivative Operator

There are many definitions for the fractional derivative. The following equation is called the reverse Grunwald–Letnikov derivative:
\[ D^a f(x) = \lim_{h \to 0} \frac{1}{h^a} \sum_{m=0}^{\infty} (-1)^m \binom{\alpha}{m} f(x - mh), \] (8)
where
\[ \binom{\alpha}{m} = \frac{\Gamma(a + 1)}{(m + 1)! \Gamma(a - m + 1)} \] (9)
For any given $a > 0$, if we denote
\[ E^a f(x) = f(x + h), \quad E^{-b} f(x) = f(x - h), \] (10)
we have
\[ D^a f(x) = h^{-a} \sum_{j=0}^{\infty} (-1)^j \binom{\alpha}{j} f(x - jh). \] (11)
Therefore, we can denote
\[ D^a f(x) = h^{-a}(1 - E^{-b})^a f(x). \] (12)
So, we have
\[ h^a D^a f(x) = (1 - E^{-h})^a f(x). \] (13)
Let
\[ \frac{a_0 + a_1 x + a_2 x^2 + \cdots + a_m x^m}{1 + b_1 x + b_2 x^2 + \cdots + b_n x^n} = (1 - x)^a + O(x^{2n+1}). \] (14)
be the $[n, n]$ Pade approximant to $(1 - x)^a$; we have
\[ \frac{a_0 + a_1 x + a_2 x^2 + \cdots + a_m x^m}{1 + b_1 x + b_2 x^2 + \cdots + b_n x^n} = (1 - x)^a + O(x^{2n+1}). \] (15)
Therefore,
\[ a_0 + a_1 x + a_2 x^2 + \cdots + a_m x^m = (1 + b_1 x + b_2 x^2 + \cdots + b_n x^n)(1 - x)^a + O(x^{2n+1}). \] (16)
Replacing \( x \) in (16) with the operator \( E^{-h} \), we have
\[
a_0 + a_1E^{-h} + a_2E^{-2h} + \cdots + a_nE^{-nh} = (1 + b_1E^{-h} + b_2E^{-2h} + \cdots + b_nE^{-nh})(1 - E^{-h})^a + O\big(E^{-h}\big)^{2n+1},
\]
where 1 can be understood as the identical operator. So, we get the following approximation:
\[
h^{-a}(a_0 + a_1E^{-h} + a_2E^{-2h} + \cdots + a_nE^{-nh}) \\
\approx (1 + b_1E^{-h} + b_2E^{-2h} + \cdots + b_nE^{-nh})(1 - E^{-h})^a h^{-a}.
\]
Noticing (13), we get
\[
(1 - E^{-h})^a = h^a D^a.
\]
So, we get
\[
h^{-a}
\left(a_0 + a_1E^{-h} + a_2E^{-2h} + \cdots + a_nE^{-nh}\right)
\approx (1 + b_1E^{-h} + b_2E^{-2h} + \cdots + b_nE^{-nh})D^a.
\]
Then, we get the following approximation to fractional derivative operators:
\[
\left(a_0 + a_1E^{-h} + a_2E^{-2h} + \cdots + a_nE^{-nh}\right)
\approx h^a(1 + b_1E^{-h} + b_2E^{-2h} + \cdots + b_nE^{-nh})D^a.
\]
Consider fractional initial value problem (1):
\[
\begin{align*}
D^a y &= f(t, y), \quad t \in [0, T], \\
y^{(k)}(0) &= 0, \quad k = 0, 1, \ldots, m - 1.
\end{align*}
\]
Denote
\[
\begin{align*}
y(0) &= 0, \\
y(h) &= y_{1}, \ldots, y(nh) = y_n,
\end{align*}
\]
\[
f(x_n, y_n) = f_n,
\]
where \( h \) is the discretization step size.
From (21), we have
\[
\left(a_0 + a_1E^{-h} + a_2E^{-2h} + \cdots + a_nE^{-nh}\right)y_n \\
= h^a(1 + b_1E^{-h} + b_2E^{-2h} + \cdots + b_nE^{-nh})D^a y_n.
\]
Due to the linearity and commutativity of the operators \( E^{-h} \) and \( D^a \), we have
\[
\begin{align*}
(a_0y_n + a_1y_{n-1} + a_2y_{n-2} + \cdots + a_n y_0) \\
= h^a D^a (y_n + b_1y_{n-1} + b_2y_{n-2} + \cdots + b_n y_0).
\end{align*}
\]
Noticing
\[
D^a y_n = f_n,
\]
we get
\[
\begin{align*}
(a_0y_n + a_1y_{n-1} + a_2y_{n-2} + \cdots + a_n y_0) \\
= h^a (f_n + b_1f_{n-1} + b_2f_{n-2} + \cdots + b_n f_0).
\end{align*}
\]
Then, we get the following numerical algorithm:
\[
a_0y_n = -a_1y_{n-1} - a_2y_{n-2} - \cdots - a_n y_0 \\
+ h^a (f_n + b_1f_{n-1} + b_2f_{n-2} + \cdots + b_n f_0).
\]
This is a multiple step algorithm for problem (1), where \( a_j \) and \( b_j \) can be determined by the Pade approximation of \((1 - x)^{a}\) for any given a. From the construction process, we can see the local truncation error of this algorithm is \(2n + 1\).

4. Some Implicit Multistep Algorithms for the Fractional Initial Value Problem

Let \( \alpha = 0.5 \); the \([2, 2]\) Pade approximant to \((1 - x)^{0.5}\) is
\[
\frac{16 - 20x + 5x^2}{16 - 12x + x^2},
\]
and the \([3, 3]\) Pade approximant to \((1 - x)^{0.5}\) is
\[
\frac{64 - 112x + 56x^2 - 7x^3}{64 - 80x + 24x^2 - x^3}.
\]
Then, from (28), we get the following two multistep algorithms:
\[
16y_n = 20y_{n-1} - 5y_{n-2} + h^{0.5} (16f_n - 12f_{n-1} + f_{n-2}),
\]
in which
\[
y_n = \frac{5}{4}y_{n-1} - \frac{5}{16}y_{n-2} + h^{0.5} \left( f_n - \frac{3}{4}f_{n-1} + \frac{1}{16}f_{n-2} \right)
\]
(algorithm 1),
\[
64y_n = 112y_{n-1} - 56y_{n-2} + 7y_{n-3} + h^{0.5} (64f_n - 80f_{n-1} + 24f_{n-2} - f_{n-3}),
\]
in which
\[
y_n = \frac{7}{4}y_{n-1} - \frac{7}{8}y_{n-2} + \frac{7}{64}y_{n-3} + h^{0.5} \left( f_n - \frac{5}{4}f_{n-1} + \frac{3}{8}f_{n-2} - \frac{1}{64}f_{n-3} \right)
\]
(algorithm 2).

In the same way, from the following \([4, 4]\) Pade approximant to \((1 - x)^{0.5}\),
\[
\frac{1 - (9/4) + (27/16)x^2 - (15/32)x^3 + (9/256)x^4}{1 - (7/4) + (15/16)x^2 - (5/32)x^3 + (1/256)x^4},
\]
we get the following (algorithm 3):
\[
y_n = \frac{9}{4}y_{n-1} - \frac{27}{16}y_{n-2} + \frac{15}{32}y_{n-3} - \frac{9}{256}y_{n-4} + h^{0.5} \left( f_n - \frac{7}{4}f_{n-1} + \frac{15}{16}f_{n-2} - \frac{5}{32}f_{n-3} + \frac{1}{256}f_{n-4} \right)
\]
(algorithm 3).
In the same way, the [2, 2] Pade approximant to 
\((1 - x)^{0.8}\) is
\[
\frac{1 - (7/5)x + (21/50)x^2}{1 - (3/5)x + (1/50)x^2}.
\]  
(36)
So, we get the following numerical algorithm:
\[
y_n = \frac{7}{5}y_{n-1} - \frac{21}{50}y_{n-2} + h^{0.8}\left(\frac{3}{5}f_{n-1} + \frac{1}{50}f_{n-2}\right)
\]
(37)
(Algorithm 4).
We can also get the following algorithm by using the [3, 3] Pade approximant to 
\((1 - x)^{0.8}\):
\[
y_n = \frac{3211}{1680}y_{n-1} - \frac{6479}{6000}y_{n-2} + \frac{3287}{2000}y_{n-3}
+ h^{0.8}\left(\frac{1867}{1680}f_{n-1} + \frac{3791}{14000}f_{n-2} - \frac{1943}{420000}f_{n-3}\right)
\]
(Algorithm 5),
and many other more complicated numerical schemes can be constructed by using the Pade approximant for any given \(\alpha\).

5. Numerical Tests

5.1. Numerical Test 1.

\[
\begin{cases}
D^{0.8}y = (\sqrt{2}/2)(y + \sqrt{1 - y^2}) \\
y(0) = 0
\end{cases}
\]  
(39)
The exact solution to this problem is \(y = \sin(x)\). We use algorithms (1) and (2) to solve this problem. Because both algorithms (1) and (2) are implicit, we use the iterative method
\[
y_n = g(y_n),
\]
(40)
to get the value of \(y_n\), where
\[
g(y_n) = \frac{5}{4}y_{n-1} - \frac{5}{16}y_{n-2} + h^{0.5}\left(\frac{3}{4}f_{n-1} + \frac{1}{16}f_{n-2}\right),
\]
(41)
for algorithm 1 and
\[
g(y_n) = \frac{7}{4}y_{n-1} - \frac{7}{8}y_{n-2} + \frac{7}{64}y_{n-3}
+ h^{0.5}\left(\frac{5}{4}f_{n-1} + \frac{3}{8}f_{n-2} - \frac{1}{64}f_{n-3}\right),
\]
(42)
for algorithm 2, respectively. The computational errors are listed in Table 1 (\(h = (\pi/2)/10\)). We also compare algorithms (1) and (2) with the exact solution in Figure 1.

We also make the comparison with the corresponding G algorithm. The results are listed in Table 2. The comparison shows that algorithms (1) and (2) are more efficient than the corresponding Grunwald–Letnikov-based G algorithm.

5.2. Numerical Test 2.

\[
\begin{cases}
D^{0.8}y = y \cos\left(\frac{2\pi}{5}\right) + \sqrt{1 - y^2} \sin\left(\frac{2\pi}{5}\right), \\
y(0) = 0
\end{cases}
\]  
(43)
The exact solution to this problem is \(y = \sin(x)\). We use the following algorithm
\[
y_n = \frac{7}{5}y_{n-1} - \frac{21}{50}y_{n-2} + h^{0.8}\left(\frac{3}{5}f_{n-1} + \frac{1}{50}f_{n-2}\right),
\]
(44)
and algorithm (5)
\[
y_n = \frac{3211}{1680}y_{n-1} - \frac{6479}{6000}y_{n-2} + \frac{3287}{2000}y_{n-3}
+ h^{0.8}\left(\frac{1867}{1680}f_{n-1} + \frac{3791}{14000}f_{n-2} - \frac{1943}{420000}f_{n-3}\right),
\]
(45)
to get the numerical solution. Because both algorithms (4) and (5) are implicit schemes, we use the iterative method
\[
y_n = g(y_n),
\]
(46)
in algorithm (4) and
\[
g(y_n) = \frac{3211}{1680}y_{n-1} - \frac{6479}{6000}y_{n-2} + \frac{3287}{200000}y_{n-3}
+ h^{0.8}\left(\frac{1867}{1680}f_{n-1} + \frac{3791}{14000}f_{n-2} - \frac{1943}{420000}f_{n-3}\right),
\]
(47)
in algorithm (5). The errors of the numerical results are listed in Table 3 (\(h = (\pi/2)/10\)).

The comparison with the corresponding G algorithm is listed in Table 4. The comparison shows that algorithms (4) and (5) are more efficient than the corresponding G algorithm.

5.3. Numerical Test 3.

\[
\begin{cases}
D^{0.8}y = x^{-0.8} \Gamma(5) \Gamma(4.2) y, \\
y(0) = 0
\end{cases}
\]  
(48)
The exact solution is \(y = x^4\); we use algorithm 5 to solve this problem. The comparison between the exact and the numerical solution can be seen in Figure 2. One can see that the numerical solution is very accurate when \(T > 1\), and the result is obtained in less than 50 steps. This is much more efficient than the G algorithm.
Table 1: Computational errors for algorithms (1) and (2).

| x    | 1h | 2h | 3h | 4h | 5h | 6h | 7h | 8h | 9h | 10h |
|------|----|----|----|----|----|----|----|----|----|-----|
| Algorithm (1) | 0.000 | 0.000 | 0.0155 | 0.0318 | 0.0455 | 0.0575 | 0.0716 | 0.0992 | 0.0783 | 0.0069 |
| Algorithm (2) | 0.000 | 0.000 | 0.000 | 0.0011 | 0.0028 | 0.0043 | 0.0049 | 0.0042 | 0.0211 | 0.0263 |
| G-algm | 0.1232 | 0.1782 | 0.1642 | 0.2589 | 0.2529 | 0.2762 | 0.3856 | 0.3842 | 0.5485 | 0.5478 |

Table 2: Computational error comparison with the \(G\) algorithm.

| x    | 1h | 2h | 3h | 4h | 5h | 6h | 7h | 8h | 9h | 10h |
|------|----|----|----|----|----|----|----|----|----|-----|
| Algorithm (1) | 0.000 | 0.000 | 0.0155 | 0.0318 | 0.0455 | 0.0575 | 0.0716 | 0.0992 | 0.0783 | 0.0069 |
| Algorithm (2) | 0.000 | 0.000 | 0.000 | 0.0011 | 0.0028 | 0.0043 | 0.0049 | 0.0042 | 0.0211 | 0.0263 |
| G-algm | 0.1232 | 0.1782 | 0.1642 | 0.2589 | 0.2529 | 0.2762 | 0.3856 | 0.3842 | 0.5485 | 0.5478 |

Table 3: Computational errors for algorithms (3) and (4).

| x    | 1h | 2h | 3h | 4h | 5h | 6h | 7h | 8h | 9h | 10h |
|------|----|----|----|----|----|----|----|----|----|-----|
| Algorithm (3) | 0.000 | 0.000 | 0.0111 | 0.0123 | 0.0245 | 0.0267 | 0.0343 | 0.0355 | 0.0455 | 0.0058 |
| Algorithm (4) | 0.000 | 0.000 | 0.0000 | 0.0009 | 0.0019 | 0.0038 | 0.0037 | 0.0042 | 0.0111 | 0.0165 |

Table 4: Comparison with the \(G\) algorithm.

| x    | 1h | 2h | 3h | 4h | 5h | 6h | 7h | 8h | 9h | 10h |
|------|----|----|----|----|----|----|----|----|----|-----|
| Algorithm (1) | 0.000 | 0.000 | 0.0155 | 0.0318 | 0.0455 | 0.0575 | 0.0716 | 0.0992 | 0.0783 | 0.0069 |
| Algorithm (2) | 0.000 | 0.000 | 0.000 | 0.0011 | 0.0028 | 0.0043 | 0.0049 | 0.0042 | 0.0211 | 0.0263 |
| G-algm | 0.1255 | 0.1351 | 0.1385 | 0.1395 | 0.2452 | 0.2522 | 0.3836 | 0.3911 | 0.4263 | 0.5485 |

Figure 1: Comparison between algorithms (1) and (2) with the exact solution.
6. Conclusion

We can construct numerical algorithms for solving the fractional initial value problem based on the Padé approximation of fractional derivative operators. Numerical tests show that this method is more efficient than the corresponding $G$ algorithms. Generally speaking, we can calculate more terms to achieve more accuracy when Grunwald–Letnikov-based method is employed, but that would lead to more computational work, while the algorithms derived from Padé approximation are proved to be more efficient.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] W. F. Ames, “Mathematics in science and engineering,” in Mathematics in Science and Engineering. Elsevier, Amsterdam, Netherlands, 1999.
[2] R. Hilfer, Applications of Fractional Calculus in Physics, World Scientific Publishing Company, Singapore, 2000.
[3] R. Metzler and J. Klafter, “The random walk’s guide to anomalous diffusion: a fractional dynamics approach,” Physics Reports, vol. 339, no. 1, pp. 1-77, 2000.
[4] S. Wang and M. Xu, “Axial Couette flow of two kinds of fractional viscoelastic fluids in an annulus,” Nonlinear Analysis: Real World Applications, vol. 10, no. 2, pp. 1087–1096, 2009.
[5] X. Jiang, X. M. Yu, X. Jiang, and X. M. Yu, “Fractional anomalous diffusion caused by an instantaneous point source in disordered fractal media,” in Proceedings of the Fourth International Conference on Fluid Mechanics, Dalian, China, 2004.
[6] E. A. Abdel-Rehim, “Fundamental solutions of the fractional diffusion and the fractional fokkerplanck equations,” Journal of the Egyptian Mathematical Society, vol. 24, no. 3, pp. 337–347, 2016.
[7] M. A. Noor, “Some iterative methods for solving nonlinear equations using homotopy perturbation method,” International Journal of Computer Mathematics, vol. 87, no. 1, pp. 141–149, 2010.
[8] Y. Khan and Q. Wu, “Homotopy perturbation transform method for nonlinear equations using He’s polynomials,” Computers & Mathematics with Applications, vol. 61, no. 8, pp. 1963–1967, 2011.
[9] J.-H. He, “Variational iteration method—a kind of non-linear analytical technique: some examples,” International Journal of Non-linear Mechanics, vol. 34, no. 4, pp. 699–708, 1999.
[10] A. M. Wazwaz, “The variational iteration method for analytic treatment for linear and nonlinear odes,” Applied Mathematics and Computation, vol. 212, no. 1, pp. 120–134, 2009.
[11] A. Yldrm, “Variational iteration method for modified camassa-holm and degasperis-procesi equations,” International Journal for Numerical Methods in Biomedical Engineering, vol. 26, no. 2, pp. 266–272, 2010.
[12] I. Podlubny, T. Skovranek, and B. Datsko, “Recent advances in numerical methods for partial fractional differential equations,” in Proceedings of the 2014 15th International Carpathian Control Conference (ICCC), Velke Karlovice, Czech Republic, May 2014.
[13] A. Saadatmandi and M. Dehghan, “A new operational matrix for solving fractional-order differential equations,” Computers & Mathematics with Applications, vol. 59, no. 3, pp. 1326–1336, 2010.
[14] D. Kai, N. J. Ford, and A. D. Freed, “Detailed error analysis for a fractional adams method,” Numerical Algorithms, vol. 36, no. 1, pp. 31–52, 2004.
[15] A. A. Kilbas, H. M. Srivastava, and J. J. Trujillo, “Theory and applications of fractional differential equations,” in North-Holland Mathematics Studies, Elsevier, Amsterdam, Netherlands, 2006.
[16] R. Scherer, S. L. Kalla, Y. Tang, and J. Huang, “The Grunwald–Letnikov method for fractional differential equations,” Computers & Mathematics with Applications, vol. 62, no. 3, pp. 902–917, 2011.
[17] I. Petras, “Fractional derivatives, fractional integrals, and fractional differential equations in MATLAB,” in Engineering Education and Research Using MATLAB, IntechOpen, London, UK, 2011.

[18] G. Shchedrin, N. Smith, A. Gladkina, L. D. Carr, Fractional Derivative of Composite Functions: Exact Results and Physical Applications, 2018.

[19] I. Podlubny, Fractional Differential Equations, Elsevier, Amsterdam, Netherlands, 1999.

[20] Y. Xu and Z. He, “The short memory principle for solving abel differential equation of fractional order,” Computers & Mathematics with Applications, vol. 62, no. 12, pp. 4796–4805, 2011.

[21] G. A. Baker and P. Graves-morris, Padé Approximants, Cambridge University Press, Cambridge, UK, Second edition, 1996.