International Journal of Scientific Research in Science, Engineering and Technology (www.ijsrset.com)
© 2020 IJSRSET | Volume 7 | Issue 3 | Print ISSN: 2395-1990 | Online ISSN : 2394-4099
DOI : https://doi.org/10.32628/IJSRSET207383

Numerical Method for Fractional-order Problems Using MATLAB software

Aye Mya Mya Moe[1], Aye Thida Myint[2], Zin Nwe Khaing[3]

[1]Department of Engineering Mathematics, University of Computer Studies, (Taungoo) / Taungoo City, Bago Region, Myanmar

[2]Department of Engineering Mathematics, Technological University, (Hpa-An) / Hpan-An City, Kayin State, Myanmar

[3]Department of Engineering Mathematics, University of Computer Studies, (Taungoo) / Taungoo City, Bago Region, Myanmar

ABSTRACT

Solving of Fractional differential equations of fractional (i.e., non-integer) order in an accurate, reliable and efficient way is much more difficult than in the standard integer-order case; moreover, the majority of the computational tools do not provide built-in functions for this kind of problem. In this paper was included the effective families of numerical methods for fractional-order problems, and the major computational issues such as the efficient treatment of the persistent memory term and the solution of the nonlinear systems involved in implicit methods using MATLAB routines specifically devised for solving three families of fractional-order problems: fractional differential equations (FDEs) (also for the non-scalar case), multi-order systems (MOSs) of FDEs and multi-term FDEs (also for the non-scalar case); some examples are provided to illustrate the use of the routines.

Keywords: Fractional Differential equations (FDEs); numerical methods; multi-term equations; product integration (PI); fractional linear multi-step methods (FLMMs); MATLAB routines.

I. INTRODUCTION

The increasing interest in applications of fractional calculus has motivated the development and the investigation of numerical methods specifically devised to solve fractional differential equations (FDEs). Finding analytical solutions of FDEs is, indeed, even more difficult than solving standard ordinary differential equations (ODEs) and, in the majority of cases, it is only possible to provide a numerical approximation of the solution.

Although several computing environments (such as, for instance, Maple, Mathematical, MATLAB and Python) provide robust and easy-to-use codes for numerically solving ODEs, the solution of FDEs still seems not to have been addressed by almost all computational tools, and usually, researchers have to write codes by themselves for the numerical treatment of FDEs. When numerically solving FDEs, one faces some non-trivial difficulties, mainly related to the presence of a persistent memory (which makes the computation extremely slow and expensive), to the low-order accuracy of the majority of the methods, to the not always straightforward computation of the coefficients of several schemes, and so on.

Writing reliable codes for FDEs can be therefore a quite difficult task for researchers and users with no

particular expertise in computational mathematics, and it would be surely preferable to rely on efficient and already tested routines.

The aim of this paper is to illustrate the basic principles behind some methods for FDEs, thus to provide a short tutorial on the numerical solution of FDEs, and discuss some non-trivial issues related to the effective implementation of methods as, for instance, the treatment of the persistent memory term, the solution of equations involved by implicit methods using MATLAB routines for the solution of a wide range of FDEs.

II. Preliminary Material on Fractional Calculus

As the starting point for introducing fractional-order operators, the Riemann–Liouville (RL) integral; for a function $y(t) \in L^1([t_0, T])$ (as usual, L^1 is the set of Lebesgue integrable functions), the RL fractional integral of order $a > 0$ and origin at t_0 is defined as:

$$\int_{t_0}^{\alpha} y(t) = \frac{1}{r(\alpha)} \int_{t_0}^{t} (t-\tau)^{\alpha-1} y(\tau) d\tau \qquad 1$$

It provides a generalization of the standard integral, which, indeed, can be considered a particular case of the RL integral (1) when $a = 1$. The left inverse of $J_{t_0}^{\alpha}$ is the RL fractional derivative:

$$\hat{D}_{t_0}^{\alpha} y(t) := D^m \int_{t_0}^{m-\alpha} y(t)$$

$$= \frac{1}{r(m-\alpha)} \frac{d^m}{dt^m} \int_{t_0}^{t} (t-\tau)^{m-\alpha-1} y(\tau) d\tau \qquad 2$$

Where, $m = [\alpha]$ is the smallest integer greater or equal to a and D^m, $y^{(m)}$ or d^m/dt^m denotes the standard integer-order derivative.

An alternative definition of the fractional derivative, obtained after interchanging differentiation and integration in Equation (2), is the so-called Caputo derivative, which, for a sufficiently differentiable function, namely for $y \in A^m [t_0, T]$ (i.e., $y^{(m-1)}$ absolutely continuous), is given by:

$$D_{t_0}^{\alpha_1} x(t) := \int_{t_0}^{m-\alpha} D^m y(t)$$

$$= \frac{1}{r(m-\alpha)} \int_{t_0}^{t} (t-\tau)^{m-\alpha-1} y^{(m)}(\tau) d\tau \qquad 3$$

$D_{t_0}^{\alpha_1} x(t)$ is a left inverse of the RL integral,

where $T_{m-1}[y; t_0](t)$ is the Taylor polynomial of degree m-1 for the function y(t) centered at t_0, that is:

$$T_{m-1}[y : t_0](t) = \sum_{k-0}^{m-1} \frac{(t-t_0)^k}{k!} y^{(k)}(t_0) \qquad 4$$

More generally speaking, by combining (Lemma 2.3) and (Theorem 3.8), it is also possible to observe that for any $\beta > \alpha$, it holds:

$$\int_{t_0}^{\beta} D_{t_0}^{\alpha} y(t) := \int_{t_0}^{\beta} \hat{D}_{t_0}^{\alpha_1} [y(t) - T_{m-1}](t)]$$

$$= \int_{t_0}^{\beta-\alpha} \hat{D}_{t_0}^{\alpha_1} [y(t) - T_{m-1}[y; t_0](t)] \qquad 5$$

A relationship that will be useful, in a particular way, on multi-term FDEs.

The two definitions (2) and (3) are interrelated, and indeed, by deriving both sides of Equation (4) in the RL sense, it is possible to observe that:

$$D_{t_0}^{\alpha} y(t) = \hat{D}_{t_0}^{\alpha_1} [y(t) - T_{m-1}[y; t_0](t)] \qquad 6$$

and, consequently:

$$\hat{D}_{t_0}^{\alpha_1} y(t) = D_{t_0}^{\alpha} y(t) + \sum_{k-0}^{m-1} \frac{(t-t_0)^{k-a}}{r(k+1-\alpha)} y^{(k)}(t_0) \qquad 7$$

Observe that in the special case $0 < a < 1$, the above relationship becomes:

$$\hat{D}_{t_0}^{\alpha_1} y(t) = D_{t_0}^{\alpha} y(t) + \frac{(t-t_0)^{-\alpha}}{r(1-a)} y(t_0) \qquad 8$$

Clearly showing how the Caputo derivative is a sort of regularization of the RL derivative at t0. Another feature that justifies the introduction of the Caputo derivative is related to the differentiation of constant function; indeed, since:

$$\hat{D}_{t_0}^{\alpha_1} 1 = \frac{1}{r(1-a)} (t-t_0)^{-a}, D_{t_0}^{\alpha} 1 = 0 \qquad 9$$

In several applications, it is preferable to deal with operators for which the derivative of a constant is zero as in the case of Caputo's derivative.

One of the most important applications of Caputo's derivative is however in FDEs. Unlike FDEs with the RL derivative, which are initialized by derivatives of non-integer order, an initial value problem for an FDE (or a system of FDEs) with Caputo's derivative can be formulated as:

$$D_{t_0}^{\alpha_1} x(t) = f(t, y(t))$$
$$y(t_0) = y_{0,} y'^{(t_0)} = (y_0)^{(1)} \ldots, (y_0)^{(m-1)}(t_0) = (y_0)^{(m-1)} \quad 10$$

where f (t, y) is assumed to be continuous and $y_{0,}(y_0)^{(1)},\ldots,(y_0)^{m-1}$ are the assigned values of the derivatives at t_0. Clearly, initializing the FDE with assigned values of integer-order derivatives is more useful since they have a more clear physical meaning with respect to fractional-order derivatives.

The application to both sides of Equation (6) of the RL integral $\int_{t_0}^{\alpha}$, together with Equation (4), leads to the reformulation of the FDE in terms of the weakly-singular Volterra integral equation (VIE):

$$y(t) = T_{m-1}[y; t_0](t) + \frac{1}{r(\alpha)} \int_{t_0}^{t} (t-\tau)^{\alpha-1} f(\tau, y(\tau)) d\tau \quad 11$$

The integral Formulation (7) is surely useful since it allows exploiting theoretical and numerical results already available for this class of VIEs in order to study and solve FDEs.

The nonlocal nature of FDEs: the presence of a real power in the kernel makes it not possible to split the solution of Equation (7) at any point t_n as the solution at some previous point $t_n - h$ plus the increment term related to the interval $[t_n - h, t_n]$, as is common with ODEs.

Furthermore, as proved by Lubich [9], the solution of the VIE (7) presents an expansion in mixed (i.e., integer and fractional) powers:

$$y(t) = T_{m-1}(t) + \sum_{i,j \in N} (t-t_0)^{i+j\alpha} Y_{i,j} \quad 12$$

Thus showing a non-smooth behavior at t0; as is well-known, the absence of smoothness at $t = t_0$ poses some problems for the numerical computation since methods based on polynomial approximations fail to provide accurate results in the presence of some lack of smoothness.

III. Multi-Step Methods for FDEs

Most of the step-by-step methods for the numerical solution of differential equations can be roughly divided into two main families: one-step and multi-step methods.

In one-step methods, just one approximation of the solution at the previous step is used to compute the solution and, hence, they are particularly suited when it is necessary to dynamically change the step-size in order to adapt the integration process to the behavior of the solution. In multi-step methods, it is instead necessary to use more previously evaluated approximations to compute the solution.

Because of the persisting memory of fractional-order operators, multi-step methods are clearly a natural choice for FDEs; anyway, although multi-step methods for FDEs are usually derived from multi-step methods for ODEs, when applied to FDEs, the number of steps involved in the computation is not fixed, but it increases as the integration proceeds forward, and the whole history of the solution is involved in each step's computation.

Multi-step methods for the FDEs (6) are therefore convolution quadrature formulas, which can be written in the general form:

$$y(t_n) = T_{m-1}[y - t_0](t_n) + \frac{1}{r(\alpha)} \sum_{j-0}^{n-1} \int_{tj}^{t_{j+1}} (t_n - \tau)^{\alpha-1} f(\tau, y(\tau)) d\tau \quad 13$$

where Φ_n and c_n are known coefficients and $t_n = t_0 + nh$ is an assigned grid, with a constant step-size $h > 0$ just for simplicity; the way in which the coefficients are derived depends on the specific method. In particular, the following two classes of multi-step methods for FDEs are as follow:

- product-integration (PI) rules,
- fractional linear multi-step methods (FLMMs).

Both families of methods are based on the approximation of the RL integral in the VIE (7) and generalize, on different bases, standard multi-step methods for ODEs. They allow one to write general-purpose methods requiring just the knowledge of the vector field of the differential equation.

The several other approaches have been however discussed in the literature: see, for instance, the generalized Adams methods, extensions of the Runge-Kutta methods, generalized exponential integrators, spectral methods, spectral collocation methods, methods based on matrix functions, and so on. In this paper, for brevity, we focus only on PI rules and FLMMs, and we refer the reader to the existing literature for alternative approaches.

IV. Applicative Example

A classical fractional-order dynamical system consisting of the nonlinear Brusselator system:

$$D_{t_0}^{\alpha_1} x(t) = A - (B+1)x(t) + x(t)^2 z(t)$$

$$D_{t_0}^{\alpha_2} z(t) = Bx(t) - x(t)^2 z(t)$$

$$x(t_0) = x_0, z_0(t) = z_0,$$

and the computation for (a_1, a_2) = (0.8, 0.7), (A, B) = (1.0,3.0) and (x_0, z_0) = (1.2, 2.8) by means of the following MATLAB lines:

```
alpha=[0.8,0.7];
A=1;B=3;
param=[A,B];
f_fun=@(t,y,par)[...
par(1)-(par(2)+1)*y(1)+y(1)^2*y(2);...
par(2)*y(1)-y(1)^2*y(2)];
J_fun=@(t,y,par)[...
-(par(2)+1)+2*y(1)*y(2),y(1)^2;...
par(2)-2*y(1)*y(2),-y(1)^2];
t0=0;T=100;
y0=[1.2;2.8];
```

After showing in Figure 1 the behavior of the solution, the errors and the EOCs are presented in Table 1.

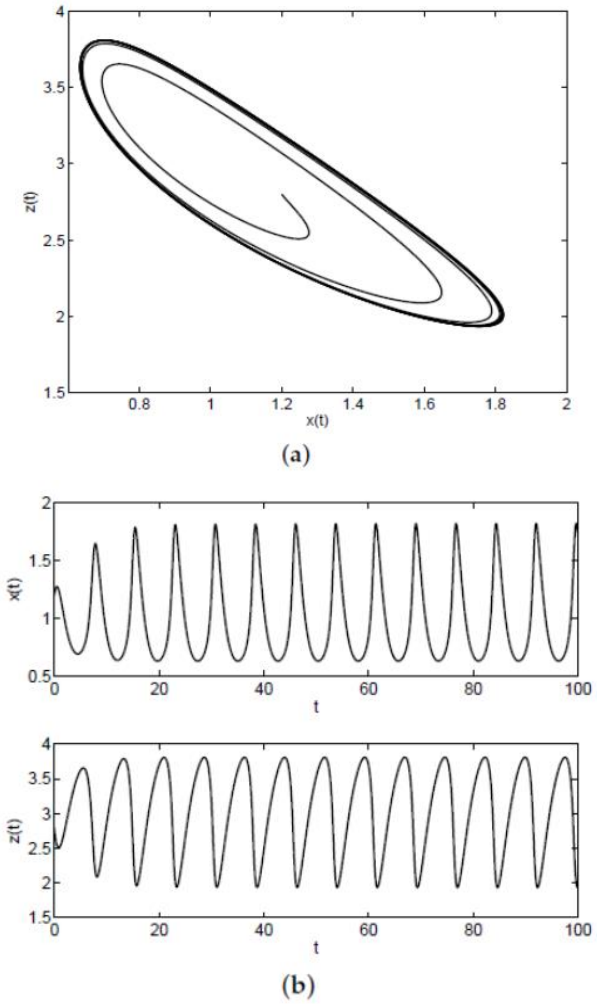

Figure 1. Behavior of the solution of the Brusselator multi-order system (MOS) in the phase plane (a) and in the (t, x) and (t, z) planes (b).

Table 1. Errors and EOC at T = 100.0 for the Brusselator system of FDEs with (a_1, a_2) = (0.8, 0.7), (A, B) = (1.0, 3.0) and (x_0, z_0) = (1.2, 2.8).

h	PI 1 Exp1		PI 1 Impl		PI 1 Exp1		PI P.C	
	Error	EOC	Error	EOC	Error	EOC	Error	EOC
2^{-2}	4.64(-1)		1.03(0)		4.90(-2)		1.16(0)	
2^{-3}	2.32(-1)	0.996	5.20(-1)	0.988	7.48(-3)	2.643	2.92(-1)	1.994
2^{-4}	1.22(-1)	0.926	2.25(-1)	1.211	2.85(-3)	1.460	5.80(-2)	2.333
2^{-5}	6.86(-2)	0.834	9.84(-1)	1.191	7.63(-4)	1.903	1.28(-2)	2.179
2^{-6}	3.69(-2)	0.896	5.20(-1)	1.124	1.92(-4)	1.991	3.41(-3)	1.910
2^{-7}	1.92(-2)	0.941	5.20(-1)	1.071	4.60(-5)	2.060	1.01(-3)	1.758

V. CONCLUSIONS

In this paper have been presented the existing methods for numerically solving systems of FDEs and have been discussed their application to multi-order systems and linear multi-term FDEs. This paper is particular focused on the efficient implementation of product integration rules and presented MATLAB routines by providing a tutorial guide to their use. Their application has been moreover illustrated in details by means of example.

VI. REFERENCES

[1]. Garrappa, R.; Popolizio, M. Generalized exponential time differencing methods for fractional order problems. Comput. Math. Appl. 2011, 62, 876–890.

[2]. Zayernouri, M.; Karniadakis, G.E. Fractional spectral collocation method. SIAM J. Sci. Comput. 2014, 36, A40–A62.

[3]. Zayernouri, M.; Karniadakis, G.E. Exponentially accurate spectral and spectral element methods for fractional ODEs. J. Comput. Phys. 2014, 257, 460–480.

[4]. Burrage, K.; Cardone, A.; D'Ambrosio, R.; Paternoster, B. Numerical solution of time fractional diffusion systems. Appl. Numer. Math. 2017, 116, 82–94.

[5]. Garrappa, R.; Moret, I.; Popolizio, M. On the time-fractional Schr¨odinger equation: Theoretical analysis and numerical solution by matrix Mittag-Leffler functions. Comput. Math. Appl. 2017, 74, 977–992.

[6]. Popolizio, M. A matrix approach for partial differential equations with Riesz space fractional derivatives. Eur. Phys. J. Spec. Top. 2013, 222, 1975–1985.

[7]. Popolizio, M. Numerical approximation of matrix functions for fractional differential equations. Bolletino dell Unione Matematica Italiana 2013, 6, 793–815.

[8]. Popolizio, M. Numerical Solution of Multiterm Fractional Differential Equations Using the Matrix Mittag-Leffler Functions. Mathematics 2018, 6, 7, doi:10.3390/ math 6010007.

[9]. Young, A. Approximate product-integration. Proc. R. Soc. Lond. Ser. A 1954, 224, 552–561.

[10]. Lambert, J.D. Numerical Methods for Ordinary Differential Systems; JohnWiley & Sons, Ltd.: Chichester, UK, 1991; p. 293.

[11]. Dixon, J. On the order of the error in discretization methods for weakly singular second kind Volterra integral equations with nonsmooth solutions. BIT 1985, 25, 624–634.

[12]. Diethelm, K. Smoothness properties of solutions of Caputo-type fractional differential equations. Fract. Calc. Appl. Anal. 2007, 10, 151–160.

Cite this article as :

Sh

Aye Mya Mya Moe, Aye Thida Myint, Zin Nwe Khaing, "Numerical Method for Fractional-order Problems Using MATLAB software", International Journal of Scientific Research in Science, Engineering and Technology (IJSRSET), Online ISSN : 2394-4099, Print ISSN : 2395-1990, Volume 7 Issue 3, pp. 437-441, May-June 2020. Available at

doi : https://doi.org/10.32628/IJSRSET207383

Journal URL : http://ijsrset.com/IJSRSET207383