A NUMERICAL METHOD TO SOLVE HIGHER ORDER ORDINARY AND FRACTIONAL DIFFERENTIAL EQUATIONS

Dr. Mamta Oberoi, Head of Department, Department of Statistics,
MLN College Yamunanagar, Haryana Email id: mamtamlnc@gmail.com

Abstract: This paper investigates numerical approaches for solving ordinary and fractional differential equations by combining traditional techniques with modern neural network algorithms. Mittag-Leffler capabilities and the Riemann-Liouville and Caputo subsidiary are hypothetical foundations for fractional differential condition analysis. The study demonstrates how to evaluate solutions for both direct and non-straight fractional equations using brain networks built using Genetic Algorithms (GA) and Molecule Multitude Improvement (PSO). The results from real-world examples show that these neural network approaches offer excellent accuracy and versatility when compared to more traditional methods like the GrünwaldLetnikov method. The solutions demonstrate how machine learning may increase the precision of numerical solutions for complex fractional systems by iteratively optimizing network weights. This study demonstrates the effectiveness of combining advanced computational approaches with conventional numerical methods, offering a strong foundation for solving differential equations in a range of scientific and engineering applications.

Keywords: Fractional Differential Equations, Numerical Methods, ordinary differential equations, Riemann-Liouville

1. INTRODUCTION

Differential equations are fundamental mathematical tools used to explain a wide range of phenomena in science, engineering, and technology. Ordinary differential equations (ODEs) and fractional differential equations (FDEs) are crucial tools for representing systems that change over time or space and capturing the relationships between variables and their rates of change. Numerous academic fields, including engineering, physics, biology, and economics, have studied and employed ODEs. From population dynamics to electrical circuits, they shed light on a range of processes by developing models that describe how these systems respond in various situations. Only the traditional analytical methods for resolving ODEs, such as variable separation, integrating factors, and exact solutions, can resolve some circumstances. Because of this, numerical methods are now crucial for solving ODEs involving complex systems that exhibit non-linear behavior or lack obvious analytical solutions. As the complexity of real-world problems increases, interest in studying fractional differential equations (FDEs), which generalize ODEs by allowing derivatives of non-integer (fractional) order, has grown. FDEs have grown in popularity because conventional integer-order models cannot accurately represent systems with memory effects, hereditary features, and anomalous diffusion. Applications of FDEs can be found in domains including finance, control theory, fluid dynamics, and viscoelasticity where processes rely on historical conditions nonlocals. However, adding fractional calculus poses serious challenges for numerical computation since fractional derivatives are non-local and require the computation of integrals over enlarged domains. This study aims to provide a comprehensive overview of numerical methods for solving ordinary and fractional differential equations. The Euler technique, Runge-Kutta methods, and finite difference methods are popular approaches for ODEs that strike a compromise between accuracy, stability, and computational economy. Specialized methods for FDEs, such as the Caputo derivative, fractional Adams-Bashforth-Moulton method, and Grünwald-Letnikov approach, are examined in light of their application to fractional operators. Convergence, stability, and error estimation issues are specifically examined, along with the benefits and drawbacks of using these numerical methods to solve beginning and boundary value problems. The study also emphasizes how important it is to select the appropriate numerical techniques based on the specifics of the differential equation that has to be resolved. Variables including processing cost, system boundary behavior, and solution smoothness all have a significant impact on how efficient the chosen strategy is. Through numerical simulations and examples, the study illustrates the value of different strategies, helping practitioners and scholars select the most effective methods for their unique problems. By bridging the gap between theoretical techniques and real-world implementations, this study advances the ongoing development and optimization of numerical methods for solving differential equations across a variety of scientific disciplines. A thorough review of numerical solutions for FDEs is given by Garrappa (2018), who also offers a detailed examination of the various approaches and how they are implemented using software. Numerous numerical systems, such as those based on Grünwald-Letnikov, Caputo, and other fractional operators, are the subject of the study. Because Garrappa's work bridges the gap between theory and practice in solving FDEs by providing practical help through software tutorials in addition to discussing theoretical elements, it is especially beneficial for scholars and practitioners. Building on this, Atangana and Owolabi (2018) address some of the inherent difficulties with conventional methods by introducing a novel numerical method for solving fractional differential equations. Their method makes use of cutting-edge algorithms that improve precision and effectiveness, especially when working with intricate, non-linear fractional models. In addition to offering comparative evaluations that highlight the benefits of their approach over current methodologies, the authors investigate the applicability of these methods in real-world events. Their work represents a major advancement in the creation of reliable numerical techniques for fractional models, which will facilitate the application of these equations to real-world scientific and engineering issues. Sun, Chang, Zhang, and Chen (2019) make a significant contribution to this topic by providing a comprehensive overview of variable-order fractional differential equations, which generalist constant-order FDEs by permitting the derivative's order to vary with time or space. The mathematical underpinnings of variable-order models are reviewed, along with their physical interpretations and numerical techniques for solving them. In addition to examining the difficulties presented by variable-order derivatives, such as stability and convergence problems, the authors also offer insights into the real-world uses of these models in a variety of academic fields. The increasing significance of variable-order FDEs in effectively simulating dynamic systems with intricate memory and heredity features is highlighted by their work. Sandev and Tomovski (2019) provide a thorough analysis of fractional equations and their numerous models in their work Fractional Equations and Models: Hypothesis and Applications. The researchers provide a thorough analysis of the numerical ideas behind fractional math's and its uses in a variety of fields, including science, design, and physical science. Their work demonstrates the versatility of fractional models in capturing complex dynamics, especially in systems with memory and genetic traits. The usefulness of several fractional operators, including as Riemann-Liouville, Caputo, and others, for simulating real-world processes is systematically reviewed by Sandev and Tomovski. Their book is a priceless tool for scholars and professionals seeking a thorough grasp of fractional differential models and their practical applications.

2. MAIN DEFINITIONS

Definition 1: The Riemann-Liouville fractional fundamental of request $\alpha \in$ R, $\alpha > 0$ of a capability f(x) ∈ Cμ, $\mu \geq -1$ is characterized as

$$(I^{\alpha}_{0+}f(t))(x) := \frac{1}{\Gamma(\alpha)}\int_0^x \frac{f(t)dt}{(x-t)^{1-\alpha}},\ (x > 0) \quad (1)$$

Definition 2: The Riemann-Liouville and Caputo fractional subsidiaries of request $\alpha \in R, \alpha > 0$

$$^{\mathrm{RL}}D^{\alpha}_{0+}f(x) := \left(\frac{d}{dx}\right)^n I^{n-\alpha}_{0+}f(x) = \frac{1}{\Gamma(n-\alpha)}\left(\frac{d}{dx}\right)^n \int_0^x \frac{f(t)dt}{(x-t)^{\alpha-n+1}},$$
$$D^{\alpha}_{0+}f(x) := I^{n-\alpha}_{0+}\left(\frac{d}{dx}\right)^n f(x) = \frac{1}{\Gamma(n-\alpha)}\int_0^x \frac{(d/dt)^n f(t)dt}{(x-t)^{\alpha-n+1}}, \quad (2)$$
where $(n = [\alpha] + 1, x > 0)$.

Definition 3: The traditional Mittag-Leffler capability is characterized by

$$\mathrm{E}_{\alpha}(x) := \sum_{k=0}^{\infty} \frac{x^k}{\Gamma(\alpha k+1)},\ (x \in C, \alpha > 0) \quad (3)$$

The definition of the generalized Mittag-Leffler function is

$$\mathrm{E}_{\alpha,\beta}(x) := \sum_{k=0}^{\infty} \frac{x^k}{\Gamma(\alpha k+\beta)},\ (x, \beta \in C, \alpha > 0) \quad (4)$$

Definition 4: The capabilities Sin $\alpha, \beta(x)$, Cos $\alpha, \beta(x)(x, \beta \in C, \alpha > 0)$ are characterized b

$$\mathrm{Sin}_{\alpha,\beta}(x) = \sum_{k=1}^{\infty} (-1)^{k+1} \frac{x^{2k-1}}{\Gamma(\alpha(2k-1)+\beta)}$$
$$\mathrm{Cos}_{\alpha,\beta}(x) = \sum_{k=0}^{\infty} (-1)^{k} \frac{x^{2k}}{\Gamma(\alpha(2k)+\beta)} \quad (5)$$

Euler's equations clearly take the accompanying structures:

$$\mathrm{E}_{\alpha,\beta}(ix) = \mathrm{Cos}_{\alpha,\beta}(x) + i\mathrm{Sin}_{\alpha,\beta}(x)$$
$$\mathrm{E}_{\alpha,\beta}(-ix) = \mathrm{Cos}_{\alpha,\beta}(x) - i\mathrm{Sin}_{\alpha,\beta}(x) \quad (6)$$

Definition 5: If Sin $\alpha, \beta(x)$ and Cos $\alpha, \beta(x)$ are defined as in Definition 4, then

$$D^{\alpha}_{a+}(x-a)^{\beta-1}\mathrm{Sin}_{\mu,\beta}[\lambda(x-a)^{\mu}] = (x-a)^{\beta-\alpha-1}\mathrm{Sin}_{\mu,\beta-\alpha}[\lambda(x-a)^{\mu}]$$
$$D^{\alpha}_{a+}(x-a)^{\beta-1}\mathrm{Cos}_{\mu,\beta}[\lambda(x-a)^{\mu}] = (x-a)^{\beta-\alpha-1}\mathrm{Cos}_{\mu,\beta-\alpha}[\lambda(x-a)^{\mu}] \quad (7)$$

A mathematical equation was used to define the beta function, and it looks like this:

$$\tilde{\beta}(p,q) = \frac{\Gamma(p)\Gamma(q)}{\Gamma(p+q)} \quad (8)$$

Then, using the Caputo fractional derivatives formulation, we obtain

$$D^{\alpha}_{a+}(x-a)^{\beta-1}\mathrm{Sin}_{\mu,\beta}[\lambda(x-a)^{\mu}] = \frac{1}{\Gamma(n-\alpha)}\int_{a+}^{x}\frac{(d/dt)^n(t-a)^{\beta-1}\mathrm{Sin}_{\mu,\beta}[\lambda(t-a)^{\mu}]}{(x-t)^{\alpha-n+1}}dt$$
$$= \frac{1}{\Gamma(n-\alpha)}\int_{0+}^{x-a}\frac{(d/d\xi)^n\xi^{\beta-1}\mathrm{Sin}_{\mu,\beta}[\lambda\xi^{\mu}]}{(x-\xi-a)^{\alpha-n+1}}d\xi \ (\xi = t-a)$$
$$= \frac{1}{\Gamma(n-\alpha)}\int_{0_*}^{1}\frac{(d/dt)^n t^{\beta-1}(x-a)^{\beta-1}\mathrm{sin}_{\mu,\beta}[\lambda t^{\mu}(x-a)^{\mu}]}{(x-a)^n(x-a)^{\alpha-n+1}(1-t)^{\alpha-n+1}}(x-a)dt \ (\xi = t(x-a))$$
$$= \frac{(x-a)^{\beta-\alpha-1}}{\Gamma(n-\alpha)}\int_{0+}^{1}\frac{(d/dt)^n t^{\beta-1}\sum_{k=1}^{\infty}([\lambda t^{\mu}(x-a)^{\mu}]^{2k-1}/\Gamma(\mu(2k-1)+\beta))}{(1-t)^{\alpha-n+1}}dt$$
$$= \frac{(x-a)^{\beta-\alpha-1}\sum_{k=1}^{\infty}(-1)^{k+1}[\lambda(x-a)^{\mu}]^{2k-1}}{\Gamma(n-\alpha)\ \Gamma(\mu(2k-1)+\beta)}\int_{0+}^{1}\frac{(d/dt)^n t^{\beta-1+\mu(2k-1)}}{(1-t)^{\alpha-n+1}}dt$$
$$= \frac{(x-a)^{\beta-\alpha-1}}{\Gamma(n-\alpha)}\prod_{i=1}^{n}[\beta-i+\mu(2k-1)]\int_{0+}^{1}\frac{t^{\beta-1+\mu(2k-1)-n}}{(1-t)^{\alpha-n+1}}dt$$
$$= \frac{(x-a)^{\beta-\alpha-1}\sum_{k=1}^{\infty}(-1)^{k+1}[\lambda(x-a)^{\mu}]^{2k-1}}{\Gamma(n-\alpha)\ \Gamma(\mu(2k-1)+\beta-n)}\tilde{\beta}(\beta+\mu(2k-1)-n, n-\alpha)$$
$$= \frac{(x-a)^{\beta-\alpha-1}\sum_{k=1}^{\infty}(-1)^{k+1}[\lambda(x-a)^{\mu}]^{2k-1}\ \Gamma(\beta+\mu(2k-1)-n)\Gamma(n-\alpha)}{\Gamma(n-\alpha)\ \Gamma(\mu(2k-1)+\beta-n)\ \Gamma(\beta+\mu(2k-1)-\alpha)}$$
$$= (x-a)^{\beta-\alpha-1}\sum_{k=1}^{\infty}\frac{(-1)^{k+1}[\lambda(x-a)^{\mu}]^{2k-1}}{\Gamma(\mu(2k-1)+\beta-\alpha)} = (x-a)^{\beta-\alpha-1}\mathrm{Sin}_{\mu,\beta-\alpha}[\lambda(x-a)^{\mu}]. \quad (9)$$

After that, (8) is true. In the same way, we get (9). Specifically, for $\beta = 1, \mu = 1$, we get

3. METHODOLOGY

The Initial Neural Network

We examine (1) with the initial condition $y(0) = C$ in order to explain the procedure. The j th trial solution that satisfies the first requirement is expressed as

$$y_j(x) = \sum_{i=1}^{M} w_{i,j}\cos(ix) + (C - \sum_{i=1}^{\mathbb{M}} w_{i,j})\cos((\cdot Å + 1)x) \quad (10)$$

where the number of neurons is represented by a mathematical expression and wi,j are the network's unknown weights that were chosen during training to lower the error function:

$$J = \frac{1}{2}\| E_j \|_2^2 = \frac{1}{2}\sum_{k=1}^{\mathcal{N}} (e_j(k))^2, \quad E_j = (e_j(1), e_j(2), \dots, e_j(\mathcal{N})^T, \qquad (11)$$

where ‖·‖ 2 is the Euclidean norm, the number of sample points is represented by a mathematical equation, and

$$e_j(k) = f(x_k, y_j(x_k)) - D_{0+1}^{\alpha} y_j(x_k)$$
$$= f(x_k, y_j(x_k)) - x_k^{-\alpha}(\sum_{i=1}^{\mathcal{M}} w_{i,j}\mathrm{Cos}_{1,1-\alpha}(ix_k) \qquad (12)$$
$$+(C - \sum_{i=1}^{\mathcal{M}} w_{i,j})\mathrm{Cos}_{1,1-\alpha}((\mathcal{M}+1)x_k))$$

where mathematical equation is present, we can use the following equation to modify the weights wi,j:

$$w_{i,j+1} = w_{i,j} + \Delta w_{i,j}, \qquad (13)$$

$$\Delta w_{i,j} = -\mu \frac{\partial J}{\partial w_{i,j}} = -\mu \sum_{k=1}^{x} \frac{\partial J}{\partial e_j(k)} \frac{\partial e_j(k)}{\partial w_{i,j}}$$
$$= -\mu \sum_{k=1}^{W} e_j(k) f_y(x_k, y_j(x_k)) \cdot (\cos(x_k) - \cos((\mathcal{M}+1)x_k)) - (x_k)^{-\alpha} e_j(k)$$
$$\cdot (\mathrm{Cos}_{1,1-\alpha}\, x_k - \mathrm{Cos}_{1,1-\alpha}((\mathcal{H}+1)x_k)) \quad (14)$$

Example 1: We start by examining the following linear fractional differential equation:

$$D_{0+}^{\alpha} y(x) = x^2 + \frac{2}{\Gamma(3-\alpha)} x^{2-\alpha} - y(x) \qquad (15)$$

under the criterion y(0) = 0. The precise answer is y(x) = x2. The following techniques can also be used to solve this equation: Particle Swarm Optimisation (PSO) algorithm Genetic Algorithm (GA) and Grünwald-Letnikov classical numerical methodology (GL). The neural network is trained 4500 times with the parameters $\mu = 0.001$, mathematical equation, and mathematical equation. Table 1 lists the network weights for Example 1.

Table 1: Weights ($\times$ 10 − 4) acquired in conjunction with Cases 1, 2, and 3's resolution.

α	Example 1	Example 2	Example 3

	1	0.7	0.5	1	0.7	0.5	1	0.7	0.5
w_1	4857	6310	6665	8226	9941	8880	5415	4070	5101
w_2	- 0506	- 3466	- 4424	4080	0219	2914	- 1589	0680	- 1582
w_3	- 4434	- 2721	- 1362	- 3468	- 0621	- 3593	- 5771	- 5800	- 3972
w_4	- 3170	- 3110	- 4990	1680	1372	3715	- 1309	- 2908	- 4142
w_5	1896	4204	5290	- 1455	- 1468	- 3147	3295	3947	6096
w_6	5534	- 0926	0222	1895	0482	1372	2815	2901	- 0121
w_7	- 6316	0182	- 2151	- 1350	0424	0273	- 4609	- 4203	- 2015

Example 2: Second, we have a look at the subsequent linear fractional differential equation:

$$D_{0+}^{\alpha} y(x) = \cos(x) + x^{-\alpha}\mathrm{Cos}_{1,1-\alpha}(x) - y(x), \quad (16)$$

assuming that y(0) = 1. The precise answer is y(x) = cos(x). The neural network is trained 1000 times with the parameters $\mu = 0.001$, mathematical equation, and mathematical equation. Table 2 provides the network weights for Example 2.

Table 2: Accuracy, approximation, and exact solution for Example 2

a		Numerical solution			Accuracy		
x	cos(x)	1	0.7	0.5	1	0.7	0.5

0.1	0.9950	0.9945	0.9967	0.9972	10^{-4}	10^{-3}	10^{-3}
0.2	0.9800	0.9788	0.9852	0.9867	10^{-3}	10^{-3}	10^{-3}
0.3	0.9553	0.9538	0.9620	0.9638	10^{-3}	10^{-3}	10^{-3}
0.4	0.9210	0.9203	0.9249	0.9256	10^{-4}	10^{-3}	10^{-3}
0.5	0.8775	0.8777	0.8758	0.8747	10^{-4}	10^{-3}	10^{-3}
0.6	0.8253	0.8254	0.8196	0.8176	10^{-4}	10^{-3}	10^{-3}
0.7	0.7648	0.7639	0.7607	0.7601	10^{-4}	10^{-3}	10^{-3}
0.8	0.6967	0.6951	0.6990	0.7016	10^{-3}	10^{-3}	10^{-3}
0.9	0.6216	0.6213	0.6286	0.6328	10^{-4}	10^{-3}	10^{-2}
1	0.5403	0.5414	0.5414	0.5405	10^{-3}	10^{-3}	10^{-4}

Example 3: The following nonlinear fractional differential equation is what we look at in our third step.

$$D^{\alpha}_{0+} y(x) = x^6 + \frac{\Gamma(3.5)}{\Gamma(3.5-\alpha)} x^{2.5-\alpha} - xy^2(x), \ldots.(17)$$

under the criterion $y(0) = 0$. The precise answer is $y(x) = x5/2$. The neural network is trained 1000 times with the parameters $\mu = 0.001$, mathematical equation, and mathematical equation. Table 3 lists the weights of the network for Example 2.

Table 3: For Example, 3, the exact, approximate, and accurate solutions

a		Numerical solution			Accuracy		
x	$x^{5/2}$	1	0.7	0.5	1	0.7	0.5
0.1	0.0031	0.0022	0.0055	0.0066	10^{-4}	10^{-3}	10^{-3}

0.2	0.0178	0.0133	0.0234	0.0266	10^{-3}	10^{-3}	10^{-3}
0.3	0.0492	0.0426	0.0566	0.0603	10^{-3}	10^{-3}	10^{-3}
0.4	0.1011	0.0972	0.1075	0.1093	10^{-3}	10^{-3}	10^{-3}
0.5	0.1767	0.1773	0.1783	0.1772	10^{-4}	10^{-3}	10^{-4}
0.6	0.2788	0.2797	0.2733	0.2711	10^{-4}	10^{-3}	10^{-3}
0.7	0.4099	0.4055	0.4010	0.4009	10^{-3}	10^{-3}	10^{-3}
0.8	0.5724	0.5643	0.5712	0.5738	10^{-3}	10^{-3}	10^{-3}
0.9	0.7684	0.7670	0.7832	0.7847	10^{-3}	10^{-2}	10^{-2}
1	1	1.0064	1.0105	1.0056	10^{-3}	10^{-2}	10^{-3}

4. CONCLUSION

This study combines neural network techniques with traditional approaches to investigate numerical solutions for fractional and ordinary differential equations. The basic concepts, including the Mittag-Leffler functions and the Riemann-Liouville and Caputo derivatives, are used to solve fractional differential equations. Through practical examples, the study demonstrates how neural networks trained with Genetic Algorithms (GA) and Particle Swarm Optimisation (PSO) can be used to solve linear and non-linear fractional equations. The examples show how optimal training can iteratively increase the correctness of solutions, confirming the accuracy and versatility of neural networks when compared to more traditional methods such as the Grünwald-Letnikov approach. According to the analysis, machine learning integration offers scalable and flexible solutions for complex fractional systems, making it a valuable tool for solving differential equations in a range of scientific and engineering applications, even though older methodologies are still helpful.

5. REFRENCES

1. Almeida, R., Malinowska, A. B., and Monteiro, M. T. T. (2018). Fractional differential equations with a Caputo derivative with respect to a kernel function and their applications. *Mathematical Methods in the Applied Sciences,* 336-352.

2. Li, C., and Cai, M. (2019). Theory and numerical approximations of fractional integrals and derivatives. *Society for Industrial and Applied Mathematics,* 1-14.
3. Dzherbashian, M. M., and Nersesian, A. B. (2020). Fractional derivatives and Cauchy problem for differential equations of fractional order. *Fractional Calculus and Applied Analysis*, 1810-1836.
4. Garrappa, R. (2018). Numerical solution of fractional differential equations: A survey and a software tutorial. *Mathematics,* 14- 16.
5. Atangana, A., and Owolabi, K. M. (2018). New numerical approach for fractional differential equations. *Mathematical Modelling of Natural Phenomena,* 1-13.
6. D'Elia, M., Du, Q., Glusa, C., Gunzburger, M., Tian, X., and Zhou, Z. (2020). Numerical methods for nonlocal and fractional models. *Acta Numerica,* 1-124.
7. Dormand, J. R. (2018). Numerical methods for differential equations: a computational approach. *CRC press,* 1-11.
8. Abbas, S., Benchohra, M., Graef, J. R., & Henderson, J. (2018). Implicit fractional differential and integral equations: existence and stability. *Walter de Gruyter GmbH& Co KG*, 1-15.
9. Sun, H., Chang, A., Zhang, Y., and Chen, W. (2019). A review on variable-order fractional differential equations: mathematical foundations, physical models, numerical methods and applications. *Fractional Calculus and Applied Analysis,* 27-59.
10. Sandev, T., and Tomovski, Z. (2019). Fractional equations and models. Theory and applications. *Cham, Switzerland: Springer Nature Switzerland AG,* 14-22.