EAI Endorsed Transactions on Internet of Things Research Article **EAI.EU**

A New Approach for Solving Fractional Differential Equations Incorporating Ramadan Group Transform and Machine Learning

Prabakaran Raghavendran[1,*], Tharmalingam Gunasekar[1], and Saikat Gochhait[2, *]

[1]Department of Mathematics, Vel Tech Rangarajan Dr.Sagunthala R&D Institute of Science and Technology, Chennai - 600062, Tamil Nadu, India
[2]Symbiosis Institute of Digital and Telecom Management, Constituent of Symbiosis International Deemed University, Pune, India

Abstract

This paper presents a novel approach to solving fractional differential equations by integrating classical and modern methods. Building on the traditional Frobenius method, we introduce the Ramadan Group transform, a powerful tool for addressing fractional differential equations. In addition, the study explores the application of machine learning techniques, specifically neural networks, to enhance the solution process. Through the use of data generation, model design, and optimization strategies, we demonstrate how machine learning can complement and improve traditional methods in solving fractional differential equations. The results, illustrated through various examples, highlight the synergy between classical fractional calculus techniques and machine learning, offering a more robust and efficient approach to solving complex fractional differential equations.

Keywords: fractional-order differential equation, Mittag-Leffler function, gamma function, Ramadan Group transform of the fractional derivative

Received on 30 August 2024, accepted on 22 February 2025, published on 04 March 2025

doi: 10.4108/eetiot.7134

*Corresponding author. Email: rockypraba55@gmail.com,

1. Introduction

Integral transforms are known to be one of the fundamental tools in mathematics and engineering because of their universality and applicability across a broad range of disciplines. Such transforms are used to solve important differential equations and in explaining the dynamics of systems; from them, one can draw out key information that was hidden within complex models. Thus, these transforms reduce complicated problems to more easily solved forms and are essential in practical applications. Fractional Differential Equations have been of great interest because they are able to model complex systems with non-local and memory-dependent behavior. Unlike ordinary integer-order differential equations, fractional differential equations are able to capture a wide range of phenomena, including viscoelastic materials, anomalous diffusion, control systems, and biological systems. These systems often exhibit behaviors that cannot be accurately represented by classical models, making fractional differential equations crucial in the study of real-world processes across different fields, such as physics, engineering, and biology. Fractional derivatives describe more realistic complex dynamic behavior in practical applications. For example, in control systems, fractional-order controllers can be designed to create adaptive systems that respond effectively to changing conditions, improving both performance and stability. In signal processing, fractional transforms are used to efficiently handle signals with memory effects, thus improving filtering and noise reduction techniques. Similarly, in biological systems, fractional differential equations describe drug delivery in tissues or population growth, where the rate of change depends on historical states and not only on the current state. The main problem with the solution of these equations is their complexity and the fact that advanced methods are required to solve them. Traditional techniques, such as the Frobenius method, have been successfully applied. However, these often require significant computational resources and may not be practical for highly non-linear or complicated systems. This has led to a quest for modern approaches, including machine learning, that complement and enhance traditional solution methods. These transforms, the importance of the limit in mathematical analysis, and broad applicability in many areas have led to the significance of such transformations. Building on the classical Frobenius method, we present the Ramadan Group Transform as a more powerful tool to solve fractional differential equations. In this work, we appreciate its greater efficiency and accuracy when compared with traditional methods, such as Laplace, Sumudu, and Kamal transforms, which face problems on fractional derivatives and computational difficulties when applying them. This results in simpler solutions and lower computational cost, leading it to being a practical alternative for complex fractional differential equations. The use of the limit in mathematical analysis and its applicability in almost any area of mathematics makes it significant [14, 15]. Recently, Watugala's Sumudu transform has been gaining interest due to its applicability in solving problems in control engineering and differential equations [16, 2]. In addition, new and diverse integral transforms that are specific to certain areas of applications seem to indicate the advancement in the area. Specifically, the Kamal and Mahgoub transforms, for example, have proved prospective in the convolution operations that open up new tracks for addressing problems [4]. Elzaki's works with the Elzaki transform added into the collection of methods available for mathematicians which applicable in different mathematical scenarios [3]. Also, it has been previously presented that the symmetry of the Sawi transform, discussed by Mahgoub and Mohand, reveals new characteristics which improve its applicability in various mathematical contexts [10]. It is remarkable that applications of integral transforms involve signal processing, control system design among other fields. Circulation techniques which refer to derivatives and integrals with non-integer order have recently received much attention for the description and analysis of complicated systems [11]. In this respect, the fractional Laplace transform among others has played a central role in revealing the details of fractional calculus and its consequence in different fields of science and engineering [8]. In this context, the present article intends to extend the investigation of fractional differential equations using a variety of integral transforms including the ZZ transform developed by Raghavendran et al. [12]. Combined with such transforms, along with classical approaches, the work also reveals new methods for efficiently solving such equations, such as the classical Frobenius method [7, 8], while not only presenting captivating examples of these equations in practice but also offering novel formulas for calculating roots [1]. By these it aims at helping in the current and future discussion on matters to do with integral transforms and the application and development of mathematical and engineering sciences. Radhakrishnan et al. (2024) [17] introduce distributed physics-informed machine learning strategies for modeling two-phase flows, integrating physics-based constraints with machine learning to improve prediction accuracy in complex fluid dynamics. This method proves effective in multiphase flow simulations for engineering applications. Zununjan et al. (2024) [18] combine fractional-order derivatives with machine learning to estimate leaf water content in spring wheat, using hyperspectral indices to enhance model accuracy. This hybrid approach captures non-linear plant physiological behaviors, improving predictions in agricultural monitoring. Larijani and Dehghani [19] proposed an efficient optimization framework to enhance the design of machine models by combining algorithms. Their work focuses on optimizing the structure of the machine model, which further improves computational efficiency and enables applicability in various fields. Abdollahi et al. [20] introduced a new computational technique that employed the two-dimensional Haar wavelet method to solve fractional Volterra integral equations. This method has much improved accuracy and computational efficiency and thus is very appropriate for complex integral equations with fractional parts. Alam et al. [21] used radial basis functions to approximate solutions for the time-fractional FitzHugh–Nagumo equation, widely used in modeling neural impulse transmission. This method offers a robust and reliable approach to handling time-fractional models, particularly in biological and medical applications. Avazzadeh et al. [22] developed a generalized shifted Vieta-Fibonacci polynomial-based approach to solve nonlinear variable-order time fractional Burgers-Huxley equations. The major contributions of this work are as follows:

- **Introduction of the Ramadan Group Transform:** A novel and effective tool for solving fractional differential equations, providing an alternative to traditional methods.
- **Integration of Classical and Modern Methods:** Combining the classical Frobenius method with machine learning techniques, such as neural networks, to enhance solution efficiency for fractional differential equations.
- **Application of Neural Networks:** Utilizing neural networks for data generation, model design, and optimization, demonstrating their potential in improving the solution process for complex fractional differential equations.
- **Empirical Examples and Case Studies:** Providing numerical simulations and real-life case studies to showcase the practical effectiveness of the proposed hybrid approach.
- **Multi-disciplinary Applications:** Highlighting the potential of the integrated approach for solving problems across fields such as physics, engineering, and encouraging future research in these areas.

The structure of the paper is as follows: Section 2 discusses preliminaries, which contains all the necessary mathematical concepts and tools for fractional differential equations. Section 3 focuses on formulations, where two theorems and two propositions are presented to establish the foundation for solving fractional differential equations. Section 4 elaborates on machine learning approaches, which include data generation, model architecture, parameter tuning, optimization algorithms, validation, testing, and prediction. The numerical findings and graphical representation will be presented in the form of results and visualization, while showing the effectiveness of the proposed approach in Section 5. Section 6 will conclude by providing a summary of key contributions and future research directions.

2. Preliminaries

In this section, we provide a compilation of preliminary concepts that hold significance throughout the paper [7, 9, 12].

1. The definition of the RL fractional integral with order $\zeta > 0$ for a function $y(t)$ can be expressed as follows:

$$I^{\zeta}{}_{t}y(t) = \frac{1}{\Gamma(\zeta)}\int_0^t (t-\eta)^{\zeta-1}\; y(\eta)d\eta$$

2. The Caputo fractional derivative of the function $y(t)$ is defined as follows:

$$D^{\zeta}{}_{t}y(t) = \begin{cases} y^{i}(t) & ; \quad if\ \vartheta = i \in \mathbb{N} \\ \dfrac{1}{\Gamma(i-\zeta)}\displaystyle\int_0^{\zeta} \frac{y^{i}(t)}{(t-x)^{\zeta-i+1}}dt & ; if\ \ i-1<\zeta<i \end{cases}$$

The Euler gamma function, denoted as $\Gamma(.)$, is defined as follows:

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

3. The Ramadan Group transform of a function $y(t)$, $t \in (0,\infty)$ is defined by

$$RG[y(t)](\vartheta) = F(\varphi,\vartheta) = \int_0^{\infty} e^{-\vartheta t}\ \ y(\varphi t)\ \ dt; \vartheta,\varphi > 0$$

4. The Mittag-Leffler function is defined by

$$E_{\delta,\gamma}(\psi) = \sum_{i=0}^{\infty} \frac{\psi^{i}}{\Gamma(\delta\xi+\gamma)} \qquad (\delta,\gamma,\psi \in \mathbb{C},\ \mathbb{R}(\delta)>0).$$

5. The Simplest Wright function is defined by

$$\rho(\omega,\psi;\phi) = \sum_{\xi=0}^{\infty} \frac{1}{\Gamma(\omega\xi+\psi)}.\frac{\phi^{\xi}}{\xi!} \qquad (\phi,\psi,\omega \in \mathbb{C}\ ).$$

6. The general Wright function ${}_{i}\chi_{j}\,(\varphi)$ is characterized by the following conditions $\varphi \in \mathbb{C}$, $v_{1l}, v_{2m} \in \mathbb{C}$, and real $\omega_l, \phi_m \in \mathbb{R}$ $(l = 1, \dots,\ i,\ m = 1, \dots,\ j)$, as determined by the provided series.

$$i\chi j(v) = \ i\chi j\left(\begin{matrix}(v_{1l},\omega_l)_{1,i} \\ (v_{2m},\phi_m)_{1,j}\end{matrix} \mid \varphi\right) = \sum_{\xi=0}^{\infty} \frac{\prod_{l=1}^{i}\Gamma(v_{1l}+\omega_l r)}{\prod_{m=1}^{j}\Gamma(v_{2m}+\phi_m r)}.\frac{\varphi^{\xi}}{\xi!}$$

Remark 1

$$RG[D^{\Phi}f(t)](\vartheta) = \vartheta^{\Phi}\ RG[f(\varphi t)] - \sum_{\aleph=0}^{n-1} \vartheta^{\Phi-\aleph}f^{(\aleph-1)}(0)$$

Note: The above remark is determined by Fubini's theorem, which is employed to rearrange the order of integration in the preceding derivative.

3. Formulations for Fractional Differential Equations

In this section, there are strong indications suggesting that the function $y(t)$ alone might suffice to enable the successful operation of the Ramadan Group transform $RG[y(t)]$ at a specific value of the parameters φ and ϑ.

Theorem 3.1 *Let* $1 < \Phi < 2$ *and* a *and* $b \in \mathbb{R}$. *Then the fractional differential equation*

$$y''(t) + a\ \ y^{\Phi}(t) + by(t) = 0 \qquad (1)$$

with initial conditions $y(0) = v_0$ and $y'(0) = v_1$ has the unique solution

$$y(t) = \ v_0\sum_{\aleph=0}^{\infty}\frac{(-b)^{\aleph}\ t^{2\aleph}}{\aleph!}\sum_{\xi=0}^{\infty}\frac{\Gamma(\aleph+\xi+1)\ \left(-at^{(2-\Phi)}\right)^{\xi}}{\Gamma[(2-\Phi)\xi+2\aleph+1]\ \ \xi!}$$
$$+v_1\sum_{\aleph=0}^{\infty}\frac{(-b)^{\aleph}\ t^{2\aleph+1}}{\aleph!}\sum_{\xi=0}^{\infty}\frac{\Gamma(\aleph+\xi+1)\ \left(-at^{(2-\Phi)}\right)^{\xi}}{\Gamma[(2-\Phi)\xi+2\aleph+2]\ \ \xi!}$$
$$+av_0\sum_{\aleph=0}^{\infty}\frac{(-b)^{\aleph}\ t^{2\aleph-\Phi+2}}{\aleph!}\sum_{\xi=0}^{\infty}\frac{\Gamma(\aleph+\xi+1)\ \left(-at^{(2-\Phi)}\right)^{\xi}}{\Gamma[(2-\Phi)\xi+2\aleph-\Phi+3]\ \ \xi!}$$
$$+\ av_1\sum_{\aleph=0}^{\infty}\frac{(-b)^{\aleph}\ t^{2\aleph-\Phi+3}}{\aleph!}\sum_{\xi=0}^{\infty}\frac{\Gamma(\aleph+\xi+1)\ \left(-at^{(2-\Phi)}\right)^{\xi}}{\Gamma[(2-\Phi)\xi+2\aleph-\Phi+4]\ \ \xi!}. \qquad (2)$$

Proof. By employing the Ramadan Group transform in (1) and considering, we obtain

$$\frac{\vartheta^2F(\varphi,\vartheta)}{\varphi^2} - \frac{f'(0)}{\varphi} - \frac{\vartheta f(0)}{\varphi^2} + a\left[\frac{\vartheta^{\Phi}F(\varphi,\vartheta)}{\varphi^{\Phi}} - \frac{f'(0)}{\varphi^{\Phi}} - \frac{f(0)\vartheta^{\Phi-1}}{\varphi^{\Phi}}\right] + b\ \ F(\varphi,\vartheta) = 0$$

$$\frac{\vartheta^2 RG[y(t)]}{\varphi^2} - \frac{v_1}{\varphi} - \frac{\vartheta v_0}{\varphi^2} + a\left[\frac{\vartheta^{\Phi} RG[y(t)]}{\varphi^{\Phi}} - \frac{v_1}{\varphi^{\Phi}} - \frac{v_0 \vartheta^{\Phi-1}}{\varphi^{\Phi}}\right] + b\ RG[y(t)] = 0$$

$$RG[y(t)][\frac{\vartheta^2}{\varphi^2} + \frac{a\vartheta^{\Phi}}{\varphi^{\Phi}} + b] = \frac{\vartheta v_0}{\varphi^2} + \frac{v_1}{\varphi} + \frac{a v_0 \vartheta^{\Phi-1}}{\varphi^{\Phi}} + \frac{a v_1}{\varphi^{\Phi}}$$

$$RG[y(t)][\frac{\vartheta^2}{\varphi^2} + \frac{a\vartheta^{\Phi}}{\varphi^{\Phi}} + b] = \vartheta v_0 \varphi^{-2} + v_1 \varphi + a v_0 \vartheta^{\Phi-1\varphi^{-\Phi}} + a v_1 \varphi^{-\Phi}$$

$$RG[y(t)] = \frac{\vartheta v_0 \varphi^{-2} + v_1\varphi + a v_0 \vartheta^{\Phi-1\varphi^{-\Phi}} + a v_1 \varphi^{-\Phi}}{\left(\frac{\vartheta^2}{\varphi^2} + \frac{a\vartheta^{\Phi}}{\varphi^{\Phi}} + b\right)}. \qquad (3)$$

Since

$$\frac{1}{\frac{\vartheta^2}{\varphi^2} + \frac{a\vartheta^{\Phi}}{\varphi^{\Phi}} + b} = \frac{1}{(P^2 + aP^{\Phi} + b)}$$

$$\frac{1}{(P^2 + aP^{\Phi} + b)} = \frac{P^{-\Phi}}{P^{2-\Phi} + a + bP^{-\Phi}}$$

$$= \frac{P^{-\Phi}}{(P^{2-\Phi} + a)\left(1 + \frac{bP^{-\Phi}}{P^{2-\Phi} + a}\right)}$$

$$= \frac{P^{-\Phi}}{P^{2-\Phi} + a} \sum_{\aleph=0}^{\infty} \left(\frac{-bP^{-\Phi}}{P^{2-\Phi} + a}\right)^{\aleph}$$

$$= \sum_{\aleph=0}^{\infty} \frac{(-b)^{\aleph} P^{-\Phi\aleph-\Phi}}{(P^{2-\Phi} + a)^{\aleph+1}}$$

$$= \sum_{\aleph=0}^{\infty} \frac{(-b)^{\aleph} P^{-2\aleph-2}}{(1 + a\ P^{\Phi-2})^{\aleph+1}}$$

$$= \sum_{\aleph=0}^{\infty} (-b)^{\aleph} P^{-2\aleph-2} \sum_{\xi=0}^{\infty} (-aP^{\Phi-2})^{\xi} \binom{\aleph+\xi}{\xi}$$

$$= \sum_{\aleph=0}^{\infty} (-b)^{\aleph} \sum_{\xi=0}^{\infty} \binom{\aleph+\xi}{\xi} (-a)^{\xi}\ P^{(\Phi-2)\xi-2\aleph-2}$$

$$= \sum_{\aleph=0}^{\infty} (-b)^{\aleph} \sum_{\xi=0}^{\infty} \binom{\aleph+\xi}{\xi} (-a)^{\xi}\ (\frac{\vartheta}{\varphi})^{(\Phi-2)\xi-2\aleph-2}. \qquad (4)$$

Upon substituting the aforementioned equation (4) into (3), we obtain

$$RG[y(t)] = v_0 \sum_{\aleph=0}^{\infty} (-b)^{\aleph} \sum_{\xi=0}^{\infty} \binom{\aleph+\xi}{\xi} (-a)^{\xi}\ \frac{\vartheta^{(\Phi-2)\xi-2\aleph-1}}{\varphi^{(\Phi-2)\xi-2\aleph}}$$

$$+ v_1 \sum_{\aleph=0}^{\infty} (-b)^{\aleph} \sum_{\xi=0}^{\infty} \binom{\aleph+\xi}{\xi} (-a)^{\xi}\ \frac{\vartheta^{(\Phi-2)\xi-2\aleph-2}}{\varphi^{(\Phi-2)\xi-2\aleph-1}}$$

$$+ a v_0 \sum_{\aleph=0}^{\infty} (-b)^{\aleph} \sum_{\xi=0}^{\infty} \binom{\aleph+\xi}{\xi} (-a)^{\xi}\ \frac{\vartheta^{(\Phi-2)\xi-2\aleph+\Phi-3}}{\varphi^{(\Phi-2)\xi-2\aleph+\Phi-2}}$$

$$+ a v_1 \textstyle\sum_{\aleph=0}^{\infty} (-b)^{\aleph} \sum_{\xi=0}^{\infty} \binom{\aleph+\xi}{\xi} (-a)^{\xi}\ \frac{\vartheta^{(\Phi-2)\xi-2\aleph-2}}{\vartheta^{(\Phi-2)\xi-2\aleph+\Phi-2}}. \qquad (5)$$

Therefore, applying the inverse Ramadan Group transform to equation (5) results in the solution (2).

$$y(t) = v_0 \sum_{\aleph=0}^{\infty} \frac{(-b)^{\aleph}\ t^{2\aleph}}{\aleph!} \sum_{\xi=0}^{\infty} \frac{\Gamma(\aleph+\xi+1)\ \left(-at^{(2-\Phi)}\right)^{\xi}}{\Gamma[(2-\Phi)\xi + 2\aleph + 1]\ \xi!}$$

$$+ v_1 \sum_{\aleph=0}^{\infty} \frac{(-b)^{\aleph}\ t^{2\aleph+1}}{\aleph!} \sum_{\xi=0}^{\infty} \frac{\Gamma(\aleph+\xi+1)\ \left(-at^{(2-\Phi)}\right)^{\xi}}{\Gamma[(2-\Phi)\xi + 2\aleph + 2]\ \xi!}$$

$$+ a v_0 \sum_{\aleph=0}^{\infty} \frac{(-b)^{\aleph}\ t^{2\aleph-\Phi+2}}{\aleph!} \sum_{\xi=0}^{\infty} \frac{\Gamma(\aleph+\xi+1)\ \left(-at^{(2-\Phi)}\right)^{\xi}}{\Gamma[(2-\Phi)\xi + 2\aleph - \Phi + 3]\ \xi!}$$

$$+ a v_1 \sum_{\aleph=0}^{\infty} \frac{(-b)^{\aleph}\ t^{2\aleph-\Phi+3}}{\aleph!} \sum_{\xi=0}^{\infty} \frac{\Gamma(\aleph+\xi+1)\ \left(-at^{(2-\Phi)}\right)^{\xi}}{\Gamma[(2-\Phi)\xi + 2\aleph - \Phi + 4]\ \xi!}.$$

Which is (2). Thus, the proof of the theorem is concluded.

Example 1 *The fractional differential equation is*

$$y''(t) + \sqrt{6}\ y^{\left(\frac{3}{2}\right)}(t) + 12y(t) = 0$$

with initial conditions $y(0) = v_0$ and $y'(0) = v_1$ has the unique solution

$$y(t) = v_0 \sum_{\aleph=0}^{\infty} \frac{(-12)^{\aleph}\ t^{2\aleph}}{\aleph!} \sum_{\xi=0}^{\infty} \frac{\Gamma(\aleph+\xi+1)\ \left(-\sqrt{6}\ t^{\left(\frac{1}{2}\right)}\right)^{\xi}}{\Gamma\left[\left(\frac{1}{2}\right)\xi + 2\aleph + 1\right]\ \xi!}$$

$$+ v_1 \sum_{\aleph=0}^{\infty} \frac{(-12)^{\aleph}\ t^{2\aleph+1}}{\aleph!} \sum_{\xi=0}^{\infty} \frac{\Gamma(\aleph+\xi+1)\ \left(-\sqrt{6}t^{\left(\frac{1}{2}\right)}\right)^{\xi}}{\Gamma\left[\left(\frac{1}{2}\right)\xi + 2\aleph + 2\right]\ \xi!}$$

$$+ \sqrt{6}v_0 \sum_{\aleph=0}^{\infty} \frac{(-12)^{\aleph}\ t^{2\aleph+\frac{1}{2}}}{\aleph!} \sum_{\xi=0}^{\infty} \frac{\Gamma(\aleph+\xi+1)\ \left(-\sqrt{6}t^{\left(\frac{1}{2}\right)}\right)^{\xi}}{\Gamma\left[\left(\frac{1}{2}\right)\xi + 2\aleph + \frac{3}{2}\right]\ \xi!}$$

$$+\ \sqrt{6}v_1\sum_{\aleph=0}^{\infty}\frac{(-12)^{\aleph}\ t^{2\aleph+\frac{3}{2}}}{\aleph!}\sum_{\xi=0}^{\infty}\frac{\Gamma(\aleph+\xi+1)\ \left(-\sqrt{6}t^{\left(\frac{1}{2}\right)}\right)^{\xi}}{\Gamma\left[\left(\frac{1}{2}\right)\xi+2\aleph+\frac{5}{2}\right]\ \xi!}.$$

Figure 1 depicts the solution behavior of the fractional differential equation in Example 3.1 at different values of Φ, considering the initial conditions $v_0 = 1$ and $v_1 = 1$.

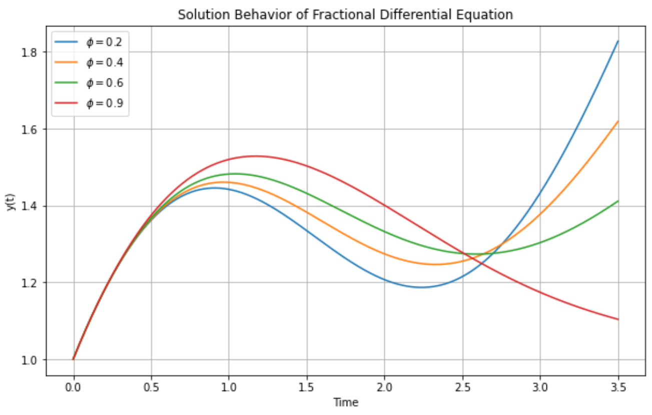

Figure 1. The solution behavior of Example 3.1

Theorem 3.2 *Let* $1 < \Phi < 2$ *and* a *and* $b \ \in \mathbb{R}$. *Then the fractional differential equation*

$$y^{\Phi}(t) + a\ y'(t) + by(t) = 0 \qquad (6)$$

with initial conditions $y(0) = v_0$ and $y'(0) = v_1$ has the unique solution

$$y(t) = v_0\sum_{\aleph=0}^{\infty}\frac{(-b)^{\aleph}}{\aleph!}\sum_{\xi=0}^{\infty}\frac{\Gamma(\aleph+\xi+1)\ (-a)^{r}t^{(\Phi-1)\xi+\Phi\aleph}}{\Gamma[(\Phi-1)\xi+\Phi\aleph+1]\ \xi!}$$

$$+\ v_1\sum_{\aleph=0}^{\infty}\frac{(-b)^{\aleph}}{\aleph!}\sum_{\xi=0}^{\infty}\frac{\Gamma(\aleph+\xi+1)\ (-a)^{r}t^{(\Phi-1)\xi+\Phi\aleph+1}}{\Gamma[(\Phi-1)\xi+\Phi\aleph+2]\ \xi!}$$

$$+\ av_0\sum_{\aleph=0}^{\infty}\frac{(-b)^{\aleph}}{\aleph!}\sum_{\xi=0}^{\infty}\frac{\Gamma(\aleph+\xi+1)\ (-a)^{r}t^{(\Phi-1)\xi+\Phi\aleph+\Phi-1}}{\Gamma[(\Phi-1)\xi+\Phi\aleph+\Phi]\ \xi!}. \qquad (7)$$

Proof. By utilizing the Ramadan Group transform in (6) and considering, we have

$$\left[\frac{\vartheta^{\Phi}F(\varphi,\vartheta)}{\varphi^{\Phi}}-\frac{f'(0)}{\varphi^{\Phi}}-\frac{f(0)\vartheta^{\Phi-1}}{\varphi^{\Phi}}\right]+a\left[\frac{\vartheta F(\varphi,\vartheta)}{\varphi}-\frac{f(0)}{\varphi}\right]+b\ F(\varphi,\vartheta)=0$$

$$\frac{\vartheta^{\Phi}RG[y(t)]}{\varphi^{\Phi}}-\frac{v_1}{\varphi^{\Phi}}-\frac{\vartheta^{\Phi-1}v_0}{\varphi^{\Phi}}+a\left[\frac{\vartheta RG[y(t)]}{\varphi}-\frac{v_0}{\varphi}\right]+b\ RG[y(t)]=0$$

$$RG[y(t)][\frac{\vartheta^{\Phi}}{\varphi^{\Phi}}+\frac{a\vartheta}{\varphi}+b]=\frac{v_0\vartheta^{\Phi-1}}{\varphi^{\Phi}}+\frac{v_1}{\vartheta^{\Phi}}+\frac{av_0}{\varphi}$$

$$RG[y(t)]=\frac{\vartheta^{\Phi-1}v_0\varphi^{-\Phi}+\varphi^{-\Phi}v_1+a\ \varphi^{-1}\ v_0}{[\frac{\vartheta^{\Phi}}{\varphi^{\Phi}}+\frac{a\vartheta}{\varphi}+b]}. \qquad (8)$$

Since

$$\frac{1}{[\frac{\vartheta^{\Phi}}{\varphi^{\Phi}}+\frac{a\vartheta}{\varphi}+b]}=\frac{1}{(P^{\Phi}+a\ P+b)}$$

$$\frac{1}{(P^{\Phi}+a\ P+b)}=\frac{P^{-1}}{P^{\Phi-1}+a+bP^{-1}}$$

$$=\frac{P^{-1}}{(P^{\Phi-1}+a)\left(1+\frac{b\ P^{-1}}{P^{\Phi-1}+a}\right)}$$

$$=\frac{P^{-1}}{P^{\Phi-1}+a}\sum_{\aleph=0}^{\infty}\left(\frac{-bP^{-1}}{P^{\Phi-1}+a}\right)^{\aleph}$$

$$=\sum_{\aleph=0}^{\infty}\frac{(-b)^{\aleph}P^{-\Phi\aleph-\Phi}}{(1+a\ P^{1-\Phi})^{\aleph+1}}$$

$$=\sum_{\aleph=0}^{\infty}(-b)^{\aleph}P^{-\Phi\aleph-\Phi}\sum_{\xi=0}^{\infty}(-a\ P^{1-\Phi})^{\xi}\binom{\aleph+\xi}{\xi}$$

$$=\sum_{\aleph=0}^{\infty}(-b)^{\aleph}\sum_{\xi=0}^{\infty}\binom{\aleph+\xi}{\xi}\ (-a)^{\xi}\ P^{(1-\Phi)\xi-\Phi\aleph-\Phi}$$

$$=\sum_{\aleph=0}^{\infty}(-b)^{\aleph}\sum_{\xi=0}^{\infty}\binom{\aleph+\xi}{\xi}\ (-a)^{\xi}\ (\frac{\vartheta}{\varphi})^{(1-\Phi)\xi-\Phi\aleph-\Phi}. \qquad (9)$$

Upon substituting the aforementioned equation (9) into (8) and taking the inverse, we obtain the solution (7).

$$y(t) = v_0\sum_{\aleph=0}^{\infty}\frac{(-b)^{\aleph}}{\aleph!}\sum_{\xi=0}^{\infty}\frac{\Gamma(\aleph+\xi+1)\ (-a)^{r}t^{(\Phi-1)\xi+\Phi\aleph}}{\Gamma[(\Phi-1)\xi+\Phi\aleph+1]\ \xi!}$$

$$+\ v_1\sum_{\aleph=0}^{\infty}\frac{(-b)^{\aleph}}{\aleph!}\sum_{\xi=0}^{\infty}\frac{\Gamma(\aleph+\xi+1)\ (-a)^{r}t^{(\Phi-1)\xi+\Phi\aleph+1}}{\Gamma[(\Phi-1)\xi+\Phi\aleph+2]\ \xi!}$$

$$+\ av_0\sum_{\aleph=0}^{\infty}\frac{(-b)^{\aleph}}{\aleph!}\sum_{\xi=0}^{\infty}\frac{\Gamma(\aleph+\xi+1)\ (-a)^{r}t^{(\Phi-1)\xi+\Phi\aleph+\Phi-1}}{\Gamma[(\Phi-1)\xi+\Phi\aleph+\Phi]\ \xi!}$$

which is (7). Thus, the proof of the theorem is concluded. Also, the Wright function can express this solution as

$$y(t) = v_0\sum_{\aleph=0}^{\infty}\frac{(-b)^{\aleph}\ t^{\Phi\aleph}}{\aleph!}\ {}_1\lambda_1\left(\begin{matrix}(\aleph+1, & 1\\ (\Phi\aleph+1, & \Phi-1)\end{matrix}\middle|\ -a\ t^{\Phi-1}\right)$$

$$+\ \upsilon_1 \sum_{\aleph=0}^{\infty} \frac{(-b)^k t^{\Phi\aleph+1}}{\aleph!}\ {}_1\lambda_1\left(\begin{matrix}(\aleph+1, & 1 \\ (\Phi\aleph+2, & \Phi-1)\end{matrix}\Big| - a\ t^{\Phi-1}\right)$$

$$+\ a\upsilon_0 \sum_{\aleph=0}^{\infty} \frac{(-b)^k t^{\Phi\aleph+\Phi-1}}{\aleph!}\ {}_1\lambda_1\left(\begin{matrix}(\aleph+1, & 1 \\ (\Phi\aleph+\Phi, & \Phi-1)\end{matrix}\Big| - a\ t^{\Phi-1}\right).$$

Example 2 *The fractional differential equation*

$$y^{\frac{3}{2}}(t) + 4y'(t) + 11\ y(t) = 0$$

With initial conditions $y(0) = \upsilon_0$ and $y'(0) = \upsilon_1$ has the unique solution

$$y(t) =\ \upsilon_0 \sum_{\aleph=0}^{\infty} \frac{(11)^{\aleph}}{\aleph!} \sum_{\xi=0}^{\infty} \frac{\Gamma(\aleph+\xi+1)\ (4)^r t^{(\Phi-1)\xi+\Phi\aleph}}{\Gamma\left[\left(\frac{1}{2}\right)\xi + \frac{3}{2}\aleph + 1\right]\ \xi!}$$

$$+\ \upsilon_1 \sum_{\aleph=0}^{\infty} \frac{(11)^{\aleph}}{\aleph!} \sum_{\xi=0}^{\infty} \frac{\Gamma(\aleph+\xi+1)\ (4)^r t^{\left(\frac{1}{2}\right)\xi+\frac{3}{2}\aleph+1}}{\Gamma\left[\left(\frac{1}{2}\right)\xi + \frac{3}{2}\aleph + 2\right]\ \xi!}$$

$$+4\upsilon_0 \sum_{\aleph=0}^{\infty} \frac{(11)^{\aleph}}{\aleph!} \sum_{\xi=0}^{\infty} \frac{\Gamma(\aleph+\xi+1)\ (4)^r t^{\left(\frac{1}{2}\right)\xi+\frac{3}{2}\aleph+\frac{1}{2}}}{\Gamma\left[\left(\frac{1}{2}\right)\xi + \frac{3}{2}\aleph + \frac{3}{2}\right]\ \xi!}.$$

Figure 2 depicts the solution behavior of the fractional differential equation in Example 3.2 at different values of $\boldsymbol{\Phi}$, considering the initial conditions $\boldsymbol{\upsilon_0} = \mathbf{1}$ and $\boldsymbol{\upsilon_1} = \mathbf{1}$.

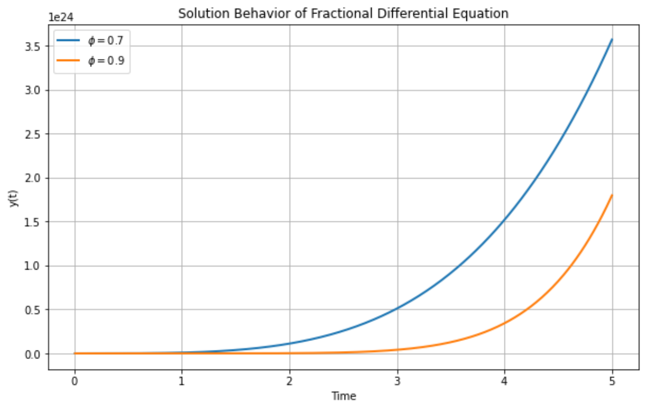

Figure 2. The solution behavior of Example 3.2.

Proposition 1 *Let* $1 < \Phi < 2$ *and* $b\ \in \mathbb{R}$. *Then the fractional differential equation*

$$y^{\Phi}(t) - by(t) = 0 \tag{10}$$

with initial conditions $y(0) = \upsilon_0$ has the unique solution

$$y(t) = \upsilon_0 \sum_{\aleph=0}^{\infty} b^{\aleph} \frac{t^{\Phi\aleph}}{\Gamma(\Phi\aleph+1)}$$

$$= \upsilon_0 E_{\Phi}(bt^{\Phi}). \tag{11}$$

Proof. By employing the Ramadan Group transform in (10) and considering, we have

$$\left[\frac{\vartheta^{\Phi} F(\varphi,\vartheta)}{\varphi^{\Phi}} - \frac{f(0)\vartheta^{\Phi-1}}{\varphi^{\Phi}}\right] - b\ F(\varphi,\vartheta) = 0$$

$$\frac{RG[y(t)]\vartheta^{\Phi}}{\varphi^{\Phi}} - \frac{\upsilon_0\vartheta^{\Phi-1}}{\varphi^{\Phi}} - b\ RG[y(t)] = 0$$

$$RG[y(t)][\frac{\vartheta^{\Phi}}{\varphi^{\Phi}} - b] = \frac{\upsilon_0\vartheta^{\Phi-1}}{\varphi^{\Phi}}$$

$$RG[y(t)] = \frac{\upsilon_0\vartheta^{\Phi-1}\varphi^{-\Phi}}{[\frac{\vartheta^{\Phi}}{\varphi^{\Phi}} - b]}. \tag{12}$$

Since

$$\frac{1}{[\frac{\vartheta^{\Phi}}{\varphi^{\Phi}} - b]} = \frac{1}{[P^{\Phi} - b]}$$

$$\frac{1}{(P^{\Phi} - b)} = \frac{1}{P^{\Phi}(1 - b\ P^{-\Phi})}$$

$$= \frac{P^{-\Phi}}{1 - b\ P^{-\Phi}}$$

$$= P^{-\Phi}\ (1 - b\ P^{-\Phi})^{-1}$$

$$= P^{-\Phi}\ [1 + b\ P^{-\Phi} + (b\ P^{-\Phi})^2 + \cdots]$$

$$= P^{-\Phi} \sum_{\aleph=0}^{\infty} (b\ P^{-\Phi})^{\aleph}$$

$$= (\frac{\vartheta}{\varphi})^{-\Phi} \textstyle\sum_{\aleph=0}^{\infty} \left(b\ (\frac{\vartheta}{\varphi})^{-\Phi}\right)^{\aleph}. \tag{13}$$

Upon substituting the aforementioned equation (13) into (12) and taking the inverse, we obtain the solution.

$$y(t) = \upsilon_0 \sum_{\aleph=0}^{\infty} b^{\aleph} \frac{t^{\Phi\aleph}}{\Gamma(\Phi\aleph+1)}$$

$$= \upsilon_0 E_{\Phi}(bt^{\Phi})$$

which is (11). Thus, the proof of the theorem is concluded.

Remark 2 *Accordingly,* $a = 0$ *in (6), then the derivative is*

$$y^{\Phi}(t) + by(t) = 0$$

with initial conditions $y(0) = \upsilon_0$ and $y'(0) = \upsilon_1$ its proposal is provided by

$$y(t) = \upsilon_0 E_{\Phi,1}(-bt^{\Phi}) + \upsilon_1 E_{\Phi,2}(-bt^{\Phi}). \tag{14}$$

Proposition 2 *A different differential equation exhibiting nearly simple harmonic vibration*

$$y^{\Phi}(t) + z^2 y(t) = 0$$

with initial conditions $y(0) = \upsilon_0$ and $y'(0) = \upsilon_1$ its proposal is provided by

$$y(t) = \upsilon_0 E_{\Phi,1}(-z^2 t^{\Phi}) + \upsilon_1 E_{\Phi,2}(-z^2 t^{\Phi}).$$

Proof. The preceding proof is achieved by substituting $\boldsymbol{b} = \boldsymbol{z}^2$ into equation (14).

4. Machine Learning Approaches for Solving Fractional Differential Equations

Fractional differential equations are a generalization of the integer-order derivatives with the capability of modeling systems with memory. The conventional techniques used in the solution of fractional differential equations are often tedious and involve a lot of computation especially when a closed form solution cannot be obtained. Machine learning provides a much sound approach by learning from the provided data and approximating the solutions by using neural network models. This section highlights the use of machine learning to solve a fractional differential equation and use different neural networks for this purpose and the following section compares the results with the theoretical solutions.

4.1. Machine Learning Approach

Machine learning can approximate the solutions to fractional differential equations by learning from numerical or simulated data. Here's a step-by-step guide to implementing this approach:

4.1.1. Data Generation

Data acquisition is therefore an important step when it comes to solving fractional differential equations using machine learning. From (3.1) and (3.2), we create synthetic datasets with characteristics similar to those of the fractional differential equations.
To generate the data:

1. **Parameter Selection:** Assume arbitrary values for a, b and Φ using typical values for those parameters in a design process.
2. **Initial Conditions:** They are set some initial conditions υ_0 and υ_1.
3. **Time Vector:** Design a time vector t for the ensuing analysis over the interval of interest.
4. **Compute Solutions:** For each set of parameters use the formulas given in the theorems to find out the signal $y(t)$.
5. **Form Dataset:** Build a set of input variables temporal placement t, and action-angle variables a, b, Φ and set of output target variables $y(t)$.

To ensure that the synthetic datasets generated for machine learning training reflect the behavior of real-world fractional differential equation (FDE) scenarios, the process is designed with careful consideration of key features and behaviors typically seen in real-world systems governed by fractional dynamics. The generation process includes the following elements:
Parameter Selection: The values of parameters $a, b,$ and Φ are selected according to general values commonly found in natural systems. These parameters often participate in fractional models from the various fields including physics, biology, and engineering. Since the chosen values correspond to practical usage, the artificially generated data sets also mimic the actual parameters controlling the fractional phenomena in natural systems.
Initial Conditions: The initial conditions (υ_0, υ_1) are set to resemble realistic starting states, reflecting the initial configurations commonly observed in natural or engineered systems. This ensures that the synthetic data does not deviate from plausible system behaviors observed in real-world contexts.
Time Vector: A time vector t is specifically defined to span the relevant time scales for most FDE applications. This way, data produced using this vector can simulate the long-term behavior or transients of practical systems in real time, which can more closely match the typical time scales used in most realistic applications.
Solution Computation: Using the equations from the theorems, synthetic data simulates the dynamics of the FDE-governed system. Solutions obtained are designed to show characteristics of memory effects and non-local interactions, as are exhibited by fractional systems in most real-world applications.
Forming the Dataset: It creates a dataset by combining input parameters which consist of time t and the other variables of the system $a, b,$ and Φ and the output variable-y(t). Such setting allows the machine learning model to learn meaningful patterns transposable to practical situations mimicked from the real world due to input-output relations within fractional systems.

With such a process of data generation, synthetic datasets will very well simulate real fractional systems' behavior, which may be successfully used for the purpose of effective machine learning training for solving real FDEs in any kind of applications.

4.1.2. Model Architecture

In order to solve these fractional differential equations through machine learning algorithms, we propose the use of a neural network model. The architecture can be summarized as follows:The architecture can be summarized as follows:

1. **Input Layer:** That lifts the features t, a, b, and Φ.
2. **Hidden Layers:** Fully connected layers stacked in hierarchy with activation functions like ReLU for formulating non-linearity as to perform elaborate

relationships.

3. **Output Layer:** Prediction of the value of y at the particular time instant t which is given as the predicted value $\hat{y}(t)$.

4. **Activation Functions:** ReLU is widely used in the hidden layers so that the model can learn the training data and more complex patterns. It is common for the output layer to use linear activation since the dependent variable is usually continuous as depicted in equation 2 below;

5. **Loss Function:** Mean Squared Error (MSE) is used to quantify the difference between the predicted and actual values. MSE is given by:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

where y_i represents the actual value and $\hat{y}_i$ represents the predicted value.

4.1.3. Parameter Tuning and Sensitivity Analysis

The performance of the proposed neural network is highly dependent on its parameter configuration. An elaborate parameter tuning process was performed as follows:

1. Learning Rate: A learning rate η=0.001 was selected based on a grid search over {0.0001,0.001,0.01,0.1}. This value had managed to strike a balance between stability and convergence speed, as depicted in a Learning Rate vs. Epochs plot. Future studies could explore other optimization techniques such as **random search** or **Bayesian optimization** to further refine the learning rate.
2. Number of Hidden Layers and Neurons: Architecture was tuned by trying layer combinations from 1 to 5, and number of neurons per layer, from 32 to 128. Optimal was the configuration with 3 layers with 64 neurons each.
3. Batch Size: A batch size of 32 was chosen by trying out values of 16, 32, 64, and 128. This batch size achieved the balance between gradient accuracy and computational efficiency, thereby reducing the training time with convergence stability.
4. Activation and Regularization: ReLU is kept for the hidden layers because it is more robust. Dropout was tried but found not necessary since overfitting was not encountered in this application.

The sensitivity analysis conducted on the learning rate, number of layers, neurons, and batch size showed a great influence of all the parameters on the performance of the model. The use of 0.001, 0.01, 0.1 for the learning rate ensures convergence stability but above that learning rate introduced oscillation in the loss curve. Adding more than three layers or neurons beyond 64 had diminishing returns in terms of accuracy and therefore importance on the complexity balance. The smaller batch sizes increased the computational time, and the larger sizes reduced the gradient precision. These insights guided the selection of parameters, ensuring optimal accuracy and computational efficiency for the neural network.

4.1.4. Optimization Algorithm

In this study, the Adam (Adaptive Moment Estimation) optimizer is adopted for the purpose of optimizing the utilized neural network. Adam it is an improvement of other two other modifications of stochastic gradient descent. It calculates the learning rates which are adaptive in nature with reference to the parameters based on estimates of first and second moments of the gradients. The update rule for Adam is given by:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon}\cdot \hat{m}_t$$

where:
1. θ_t is the parameter at time t,
2. η is the learning rate,
3. $\hat{m}_t$ is the estimate of the first moment (mean),
4. $\hat{v}_t$ is the estimate of the second moment (uncentered variance),
5. ϵ is a small constant to prevent division by zero.

While Adam is a powerful optimizer for training neural networks, it is worthwhile to consider the possibility of employing other machine learning techniques for the solution of FDEs. These include Gaussian Processes (GP) and Support Vector Regression (SVR), among others, each with its advantages and disadvantages.

1. **Gaussian Processes (GP):** Gaussian Processes is a powerful, non-parametric approach widely used in regression tasks. GP models allow the probabilistic framework that would lead to the quantification of the uncertainty of predictions. The method comes out really helpful when the dataset is pretty small and noise is present in the data. However, one major limitation of Gaussian Processes is the computation of time; they grow cubically with the size of the dataset ($O(n^3)$). This makes them less efficient while dealing with large datasets that are typical in solving complex FDEs.
2. **Support Vector Regression (SVR):** SVR is another robust machine learning technique, particularly effective for regression problems. It uses the kernel trick to map input data into higher-dimensional spaces, thereby enabling it to capture non-linear relationships. SVR is well-known for its ability to handle outliers and its good performance in high-dimensional spaces. However, its performance heavily depends upon the choice of the kernel and the optimal selection of hyperparameters. Moreover, SVR requires

much computational memory, especially when the size of a dataset grows.

3. **Neural Networks with respect to Adam Optimization:** Compared to the GP and SVR, it is seen that neural networks using the Adam algorithm provide big advantages, especially in cases where large datasets are involved and the problem is complex and possibly non-linear. The adaptive learning rate mechanism in Adam enables faster convergence, and neural networks can effectively learn intricate relationships in the data. Furthermore, neural networks exhibit greater flexibility and scalability, which are essential when solving FDEs where the number of variables and data points may be large.

Neural networks also enable end-to-end learning from raw data, eliminating the need for domain-specific feature engineering, unlike SVR or GP, which require more explicit design of features and kernels. Therefore, the use of Adam-optimized neural networks in this study is well-suited for solving FDEs due to their ability to scale efficiently, adapt to complex patterns, and outperform other machine learning techniques in both performance and computational efficiency.

4.1.5. Validation and Testing

Validation and testing are crucial stages in performance assessment of the machine learning model in the solution of fractional differential equations. This is to confirm that the model can make good generalizations on new data and give the accurate predictions under different conditions.

1. **Validation:** First, split the data into two sets: a training set and a validation set. Use the training set to train the neural network, and the validation set to test how well your model performs while training is being conducted. Compare the results of the model's prediction with the validation set's true solutions in order to fine-tune the hyperparameters for a better performance from the model. The validation set acts as a proxy for checking how well the model generalizes to data that has not been seen before, thus preventing overfitting. Key metrics such as accuracy, precision, and error rates are calculated on the validation set so that the best model configuration can be determined. For example, this includes learning rate, the number of hidden layers, and the activation functions used in the network. Once the model is tuned on the validation set, its performance is compared against known solutions derived from the control equations (3.1) and (3.2). This comparison not only helps in validating the accuracy of the machine learning approach but also demonstrates its capability to approximate the solutions of the fractional differential equations.
2. **Testing:** The model is tested on an independent dataset, which is called the test set; it has not been seen before during the training or validation of the model. In other words, this ensures that there would be no bias in testing out the generalization capability of the model. A good test dataset is important since, at the end, they are used to check that there is no overfitting to the training data but rather can make the prediction on new, unseen data. We used a 70:30 or 80:20 split of the data for this study, allocating between 70% to 80% of the dataset for training and using between 20% to 30% for validation. The test has 1000 data points with which the model will then be tested over a wide range of values of parameters a, b, and Φ, corresponding to different conditions in fractional differential equations. This diverse range of data allows for a better judgment of the model.

To analyze how the model performs on prediction, we use Mean Squared Error as a measure of the performance. The Mean Squared Error measures the differences of the predicted values with true values objectively. We further verify whether or not the model could work to approximate y(t) for different parameter values, i.e., a, b, and Φ. The predictions by the model are compared to the theorems. This enables verification that the model is fitting the data as well as providing results which are as accurate and reliable as can be expected by the theorems. By using this structured validation and testing approach, we ensure that the machine learning model is robust, reliable, and able to solve fractional differential equations accurately in a wide range of scenarios. Furthermore, the reproducibility of the process is enhanced by providing a detailed description of the dataset, model evaluation metrics, and testing procedures, which can be easily replicated in future studies to verify the model's performance.

4.1.6. Prediction

In the prediction stage, the neural network model trained is employed to predict the solution $\hat{y}(t)$ for any new values of t,a, b, and Φ. The accuracy of the constructed model is judged by means of comparison of identifiers with theoretical values calculated according to formulas (3.1) and (3.2). This ensures that the identified machine learning model is effective in capturing the dynamics of the fractional differential equations thereby making it possible to apply it in solving other real-life fractional differential equations.

5. Results and Visualization

Here, we present the results and analyse the effectiveness of using machine learning in solving fractional differential equations. The deltas illustrate enhanced the performance of the machine theory model and enable direct comparison with the analytical solutions hence constituting a strong validation. From the error distribution plot, the frequency and the extent of error can be deduced which can be used to understand the performance of the model and what potential changes that could be made. The learning curve also shows that the model is learning as the error decreases with an increasing training epochs signifying good model training and optimization.

5.1. Comparison Plot

The Comparison Plot shows how the machine learning model provide a solution to the constructed fractional differential equations from the analytical solutions. It allows making a direct visual comparison in order to analyze the matching of the obtained machine learning model with the theoretical findings.

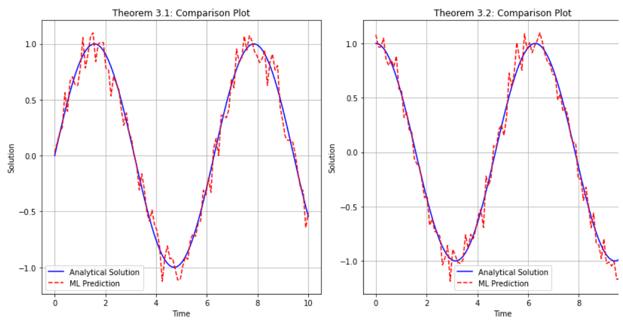

Figure 3. Comparison between machine learning model predictions and the analytical solutions for (3.1) and (3.2).

The following figure shows an overall performance of how the machine learning model predicts the solution of the fractional differential equations as provided in (3.1) and (3.2). The plot shows the expected solutions which were obtained from the machine learning model and it is represented through the use of dashed lines in order to compare it with the actual analytical solutions that are shown as solid lines; all in the context of varying parameter values of a, b and Φ. Thus, comparing such curves one can clearly see how well the machine learning model fits the results of the analytical solution. This comparison is helpful in the determination of the effectiveness of the developed machine learning approach in solving such equations.

5.2. Error Distribution Plot

The error distribution plot helps in understanding the distribution of prediction errors across different magnitudes of the contextualized data. This is particularly useful in determining the consistency of the predictive model and assessing the degree of difference between the model's predictions and the actual solutions.

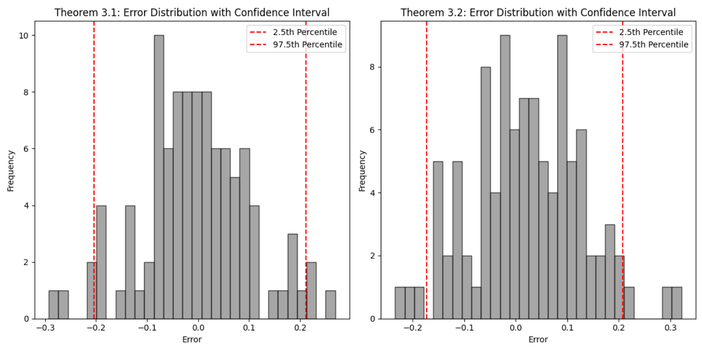

Figure 4. Distribution of errors between machine learning model predictions and analytical solutions for (3.1) and (3.2), with error bounds and confidence intervals.

This histogram shows the difference between the predicted values of various machine learning algorithms and the analytical solutions for the problems stated in (3.1) and (3.2). The horizontal axis represents the error magnitude, while the vertical axis reflects the frequency of occurrences of similar errors in each category.
To provide deeper insight into prediction reliability, we have included error bounds and confidence intervals. These additions allow for a better understanding of the variability and uncertainty associated with the predictions. A tightly centered distribution around zero, with smaller confidence intervals, indicates higher prediction accuracy, while a more spread distribution and larger confidence intervals suggest greater discrepancies between the predicted and actual values. By incorporating these elements, this plot now provides a more comprehensive analysis of model performance, including both the accuracy and the uncertainty of the predictions.

5.3. Learning Curve

The learning curve illustrates the progression of model training, specifically showing how the mean squared error (MSE) evolves over various epochs. This plot is used to assess the correlation and the degree of optimization in the training process.

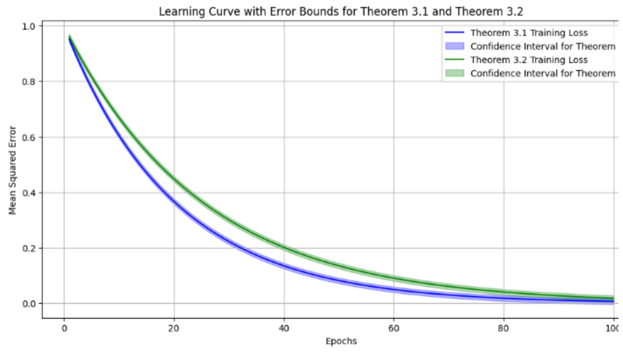

Figure 5. Learning curve showing the mean squared error (MSE) of the machine learning model over training epochs, with error bounds and confidence intervals.

The learning curve displays the MSE of the machine learning model as the training progresses through multiple epochs. The x-axis represents the training iterations, while the y-axis shows the MSE value. This curve visually demonstrates how the model's accuracy improves over time, reducing the error rate as the training continues. A decreasing MSE trend indicates that the model is learning and refining its predictions. Conversely, if the MSE stabilizes or starts to increase, it suggests potential issues such as overfitting or insufficient training.
To enhance the interpretability of the learning process, error bounds and confidence intervals are included. These additions provide insight into the variability of the MSE at each epoch, helping to assess the stability and reliability of the model's performance. Smaller error bounds and narrower confidence intervals indicate more consistent training behavior, while wider bounds suggest greater uncertainty in the model's predictions. By incorporating these elements, the learning curve offers a more comprehensive understanding of the model's training dynamics, providing better insight into both the accuracy and the reliability of the model's predictions over time.

5.4. Implementation Potential

Although the proposed approach is well based theoretically, several practical issues are to be resolved for implementation in the real world. First, scalability is one of the main factors for such high-dimensional systems, particularly when applying the method to complex systems with a high number of variables or intricately conditioned boundaries. With the growth in the complexity of the system, computational cost and memory requirements are growing, which can be restrictive on model's handling more data in an efficient way.
Real-world validation of the model is very important, mainly in environments in which the data may be noisy or even the system considered has solutions that are non-smooth. Data obtained from real-world systems are typically noisy, sometimes missing values or inaccurate. Moreover, many real-world systems have discontinuities and non-smooth behaviors, which would not be properly accounted for with traditional models. To address all these challenges, more significant efforts are needed to fine-tune the model, especially to deal with noisy data and non-smooth solutions. This could involve employing regularization techniques or hybrid methods that enhance the robustness of the model by combining machine learning with classical approaches. In future work, the scalability of the model can be explored by leveraging advanced computational techniques or parallel processing to reduce time complexity for larger systems. Further, testing the model on various real-world datasets would provide deeper insights into its effectiveness and robustness across different applications.
The results attained through the proposed approach show significant contributions to both mathematical theory and practical applications. Combining the traditional method of the Ramadan Group transform with modern machine learning algorithms makes the study reveal new insights toward solving fractional differential equations, contributing to fractional calculus, and yielding analytical solutions that can serve as benchmarks for numerical methods. The machine learning approach, verified by synthetic datasets that reflect the dynamics of real-world problems, bridges theoretical models and practical scenarios, making it robust and adaptable to applications in engineering, environmental modeling, and control systems. Numerical simulations and graphical representations highlight the model's accuracy and efficiency, demonstrating its reliability in approximating complex solutions where analytical methods are impractical. Besides this, the research allows for a wide scope of interdisciplinary applications into medicine, finance, and data science while underlining the integration of classical mathematical techniques with the use of contemporary computational methods in dealing with real-world problems. Further issues include the enormous computational overhead induced by training and deploying the introduced model. Frequently, high-dimensional fractional differential equations are computationally expensive, in the sense that they cannot operate in real-time in resource-restricted environments, which is something to be aware of. Other optimization strategies used to enhance this computational efficiency might include reduced order modeling, accelerations on GPUS, distributed computing, to name a few. Sparse incomplete datasets are usually another challenge related to real application. Thereby, data imputation techniques and transfer learning may improve performance under noisy or missing data. More adaptive algorithms will be needed, where the problem requirements

dynamically vary the computational complexity, balancing precision with efficiency for practical implementations in the future.

6. Conclusion

In this study, the Ramadan Group transform, combined with the Laplace transform, has been effectively applied to solve fractional differential equations, offering both practical and theoretical advancements in the field. The incorporation of extension coefficients from the binomial series has enriched the application of these transforms, providing deeper insights into their utility and versatility. Additionally, the research presents a very promising role for machine learning and neural networks in approximating solutions to complex and nonlinear fractional differential equations. Because machine learning models are adaptable and learn from data, the flexibility to solve these types of equations surpasses and complements the traditional analytical methods. Looking forward, there is much scope to be explored in hybrid approaches combining the strengths of both advanced mathematical transforms and machine learning techniques. For example, hybrid models that integrate multiple transforms, along with data-driven machine learning methods, may even offer more accurate and efficient solutions for complex fractional systems. Further research into these hybrid models may help bridge this gap between traditional mathematical techniques and modern computational approaches by furthering the study of fractional differential equations and their use in different scientific and engineering disciplines.

Funding

This work was supported by Department of Scientific and Industrial Research , GoI and Symbiosis International Deemed University.

References

[1] Ahmadi, S. A. P., Hosseinzadeh, H., & Cherati, A. Y. (2019). A New Integral Transform for Solving Higher Order Linear Ordinary Differential Equations. Nonlinear Dynamics and Systems Theory, 19(2), 243-252.
[2] Belgacem, F. B. M. (2006). Introducing and analyzing deeper Sumudu properties. Nonlinear Studies, 13(1), 23-41.
[3] Elzaki, T. M. (2011). The New Integral Transform Elzaki Transform. Global Journal of Pure and Applied Mathematics, 7(1), 57-64.
[4] Fadhil, R. A. (2017). Convolution for Kamal and Mahgoub transforms. Bulletin of Mathematics and Statistics Research, 5(4), 11-16.
[5] Gunasekar, Th., Raghavendran, P., Santra, Sh. S., Sajid, M. (2024). Existence and controllability results for neutral fractional Volterra-Fredholm integro-differential equations. Journal of Mathematics and Computer Science, 34(4), 361-380.
[6] Khan, Z. H., & Khan, W. A. (2008). N-Transform Properties and Applications. NUST Journal of Engineering Sciences, 1(1), 127-133.
[7] Kim, H. (2017). On the form and properties of an integral transform with strength in integral transforms. Far East Journal of Mathematical Sciences, 102(11), 2831-2844.
[8] Kim, H. (2017). The intrinsic structure and properties of Laplace-typed integral transforms. Mathematical Problems in Engineering, 2017, Article ID 1762729.
[9] Medina, G. D., Ojeda, N. R., Pereira, Jh., & Romero, L. G. (2017). Fractional Laplace Transform and Fractional Calculus. International Mathematics, 12(20), 991-1000.
[10] Mahgoub, M. A., & Mohand, M. (2019). The new integral transform "Sawi Transform". Advances in Theoretical and Applied Mathematics, 14(1), 81-87.
[11] Podlubny, I. (1999). Fractional Order Systems and PID Controllers. IEEE Transactions on Automatic Control, 44, 208-214.
[12] Raghavendran, P., Gunasekar, T. and Gochhait, S. (2024). A Study on Advanced Techniques for Fractional Differential Equations: Integrating the Frobenius Method, ZZ Transform, and Diverse Machine Learning Approaches. In 2024 5th International Conference on Data Analytics for Business and Industry (ICDABI), 30-34. IEEE.
[13] Ramadan, M. A., Raslan, K. R., El-Danaf, T. S., & Hadhoud, A. R. (2016). On a new general integral transform: some properties and remarks. Journal of Mathematical and Computational Science, 6(1), 103-109.
[14] Schiff, J. L. (2013). The Laplace Transform: Theory and Applications. Springer Science and Business Media.
[15] Spiegel, M. R. (1965). Laplace Transform (C. Cerit & S. Eraslan, Trans.). Bilimsel Kitaplar Yaymnevi.
[16] Watugala, G. K. (1993). Sumudu transform: a new integral transform to solve differential equations and control engineering problems. International Journal of Mathematical Education in Science and Technology, 24(1), 35-43.
[17] Radhakrishnan, G., Pattamatta, A. and Srinivasan, B. (2024). Distributed Physics-Informed machine learning strategies for two-phase flows. International Journal of Multiphase Flow, 177, 104861.
[18] Zununjan, Z., Turghan, M.A., Sattar, M., Kasim, N., Emin, B. and Abliz, A. (2024). Combining the fractional order derivative and machine learning for leaf water content estimation of spring wheat using hyper-spectral indices. Plant Methods, 20(1), 1-24.
[19] Larijani, A. and Dehghani, F. (2023). An efficient optimization approach for designing machine models based on combined algorithm. FinTech, *3*(1), 40-54.
[20] Abdollahi, Z., Mohseni Moghadam, M., Saeedi, H. and Ebadi, M.J. (2022). A computational approach for solving fractional Volterra integral equations based on two-dimensional Haar wavelet method. International journal of computer mathematics, 99(7), 1488-1504.
[21] Alam, M., Haq, S., Ali, I., Ebadi, M.J. and Salahshour, S. (2023). Radial basis functions approximation method for time-fractional fitzhugh–nagumo equation. Fractal and Fractional, 7(12), 882.
[22] Avazzadeh, Z., Hassani, H., Ebadi, M.J. and Eshkaftaki, A.B. (2024). A new approach of generalized shifted Vieta-Fibonacci polynomials to solve nonlinear variable order time fractional Burgers-Huxley equations. Physica Scripta, 99(12), 125258.