fractal and fractional

MDPI

Article

Quantitative Analysis of the Fractional Fokker–Planck–Levy Equation via a Modified Physics-Informed Neural Network Architecture

Fazl Ullah Fazal [1], Muhammad Sulaiman [1], David Bassir [2,3,*], Fahad Sameer Alshammari [4] and Ghaylen Laouini [5]

[1] Department of Mathematics, Abdul Wali Khan University Mardan, Mardan 23200, Pakistan; fazufaz@awkum.edu.pk (F.U.F.); msulaiman@awkum.edu.pk (M.S.)
[2] Smart Structural Health Monitoring and Control Lab (SSHMC) Lab, CNAM-Dongguan University of Technology, D1, Daxue Rd., Songshan Lake, Dongguan 523000, China
[3] UTBM, IRAMAT UMR 7065-CNRS, Rue de Leupe, CEDEX, 90010 Belfort, France
[4] Department of Mathematics, College of Science and Humanities in Alkharj, Prince Sattam bin Abdulaziz University, Al-Kharj 11942, Saudi Arabia; f.alshammari@psau.edu.sa
[5] College of Engineering and Technology, American University of the Middle East, Egaila 54200, Kuwait; ghaylen.laouini@aum.edu.kw
* Correspondence: david.bassir@utbm.fr

Abstract: An innovative approach is utilized in this paper to solve the fractional Fokker–Planck–Levy (FFPL) equation. A hybrid technique is designed by combining the finite difference method (FDM), Adams numerical technique, and physics-informed neural network (PINN) architecture, namely, the FDM-APINN, to solve the fractional Fokker–Planck–Levy (FFPL) equation numerically. Two scenarios of the FFPL equation are considered by varying the value of (i.e., 1.75, 1.85). Moreover, three cases of each scenario are numerically studied for different discretized domains with 100, 200, and 500 points in $x \in [-1, 1]$ and $t \in [0, 1]$. For the FFPL equation, solutions are obtained via the FDM-APINN technique via 1000, 2000, and 5000 iterations. The errors, loss function graphs, and statistical tables are presented to validate our claim that the FDM-APINN is a better alternative intelligent technique for handling fractional-order partial differential equations with complex terms. The FDM-APINN can be extended by using nongradient-based bioinspired computing for higher-order fractional partial differential equations.

Keywords: fractional Fokker–Planck–Levy equation; physics-informed neural networks (PINNs); finite difference method (FDM); Adams numerical technique; hybrid numerical method; fractional partial differential equations; computational fluid dynamics; bioinspired computing

Citation: Fazal, F.U.; Sulaiman, M.; Bassir, D.; Alshammari, F.S.; Laouini, G. Quantitative Analysis of the Fractional Fokker–Planck–Levy Equation via a Modified Physics-Informed Neural Network Architecture. *Fractal Fract.* **2024**, *8*, 671. https://doi.org/10.3390/fractalfract8110671

Academic Editors: Xiao-Jun Yang, Wenxue Zhou, Sunil Dutt Purohit, Ali Turab and Ahmed Refaie Ali

Received: 24 September 2024
Revised: 8 November 2024
Accepted: 15 November 2024
Published: 18 November 2024

1. Introduction

The fractional Fokker–Planck (FFP) equation, also known as the Fokker–Planck–Levy (FPL) equation, is a generalization of the traditional Fokker–Planck (FP) equation, which includes Levy processes, such as heavy-tailed distributions or leaps. The fractional Fokker–Planck–Levy (FFPL) equation is applied in finance, ecology, and physics for anomalous diffusion in complex systems, abrupt, long-range movements in animal movement patterns, and jumps or heavy tails in the underlying asset. The traditional Fokker–Planck equation describes the time evolution of a particle's probability density function of velocity under the influence of forces and Gaussian white noise. Gaussian processes, however, cannot fully explain the leaps and heavy tails observed in a wide range of physical and economic phenomena. A more suitable mathematical framework for such situations is provided by Levy processes, which comprise a larger class of stochastic processes defined by stable distributions and leaps. The fractional derivative causes nonlocality, which requires specialized numerical techniques that are capable of handling integral terms efficiently. By

adding fractional derivatives, the fractional Fokker–Planck–Levy (FFPL) equation extends the traditional Fokker–Planck (FP) equation, which is capable of modelling these jump-like, nonlocal dynamics. Recently, fractional calculus has improved financial system modelling by providing a more complete understanding of asset dynamics. Fractional calculus extends integration and differentiation to fractional orders, empowering the introduction of memory and long-range dependencies in mathematical models [1–3]. While conventional numerical techniques are excellent for solving classic PDEs, they may not handle fractional derivatives effectively. Scholars such as Chen et al. [4] and Nikan et al. [5] have attempted to improve conventional numerical techniques by adopting mesh-free approaches such as the radial basis function (RBF) method to solve fractional PDEs. However, the RBF technique complicates the selection of the basis function. Moreover, ref. [6] discussed Hilfer's fractional impulsive neutral Levin–Nohel integrodifferential system with variable time delay, ref. [7] discussed the problem of a two-order Caputo–Hadamard fractional pantograph with Dirichlet boundary conditions, and ref. [8] discussed a mathematical study of a coupled fully hybrid (k, Φ)-fractional order system.

Solving the FFPL equation numerically is challenging because both the large discrete changes caused by the jump term and the small-scale behaviour driven by the diffusion term are needed. The FFPL equation of interest in this paper poses additional important challenges, where classical grid-based approaches fail because of the increase in computational requirements with the nonlocality of the partial differential equation (PDE), making them impractical. Physics-informed neural networks (PINNs) [9] are currently popular in solving partial differential equations (PDEs) because neural networks have high universal approximation characteristics [10], robust optimization [11], meshless plus grid-free training, and generalization capacities [12].

The PINN is a promising strategy for solving large PDEs because it incorporates physical rules and constraints directly into the learning process [13,14]. The PINN technique uses fractional derivatives in the governing partial differential equation (PDE) to obtain an acceptable approximation of the solution. The PINN design includes a loss function that enforces the differential equation, boundary, and initial conditions, resulting in precise solutions even with insufficient input. The PINN algorithm integrates the fractional derivative into the FFPL equation and uses neural networks to acquire the basic solution. The PINN ensures that the solution has physical stability and responds to real-world financial complications. Several studies, including those of Ibrahim et al. [15,16], Jamshaid et al. [17], and Lou et al. [18], have employed this approach to solve PDEs instead of other approaches. High-dimensional PINNs [19,20] use automatic differentiation to calculate integer-order derivatives. Compared with automatic differentiation, the FDM-APINN is faster and consistently results in fewer errors [21]. A PDE solver PINN technique is generated by [22] and inspired by the FDM-APINN, which shows that the method consistently results in minimal errors. Automatic differentiation and FDM-APINN libraries currently do not cover fractional-order derivatives. Approximating the fractional Laplacian still remains a significant challenge. For the approximation of the fractional Laplacian, we used the finite difference method (FDM) in this manuscript. Fractional FPL equations present a challenge for both classical and novel methodologies, such as PINNs.

We introduce a special type of FDM-APINN technique to solve the fractional FPL equation. The loss function used in this technique considers not only the mean squared error (MSE) between actual and predicted data (data loss) but also physics-based loss. This additional loss element requires the neural network to follow the physical principles specified by the FFPL equation. The fractional Laplacian is also under consideration in this technique. The fractional Laplacian is an essential component in fractional PDEs, distinguishing them from classical PDEs. The FDM is used for the approximation of the fractional Laplacian and then utilized in the PINN. This technique employs automatic differentiation to produce the gradients required for calculating the physics-based loss. This enables the FDM-APINN to handle the complex derivatives found in the FFPL equation. Furthermore, the physics loss is also computed. The neural network predictions are

guaranteed to satisfy the FFPL equation via physics loss, which makes the model physics informed. Therefore, the proposed technique is unique and efficient.

1.1. Related Work

FPL and FP equations are frequently encountered in statistical mechanics, and their large dimensionality presents notable difficulties for conventional analytical techniques such as finite difference [23,24]. However, physics-informed neural networks (PINNs), which are machine learning techniques, present a promising mesh-free approach that seamlessly integrates observational data. PINNs were used in Chen et al.'s research [25] to solve forward and inverse problems related to the Fokker–Planck–Levy equations. Zhang et al. [26] addressed FP equations involving sparse data via deep KD-tree techniques. Similarly, deep learning was used by Zhai et al. [27] and Wang et al. [28] to solve steady-state FP equations. Moreover, Lu et al. [29] concentrated on employing normalizing flows to learn high-dimensional probability densities modelled by FP equations. Furthermore, normalization flow approaches were used for FP equations by Feng et al. [30]. Guo et al. [31] and Tang et al. [32] presented an adaptive deep density approximating technique for steady-state FP equations based on normalizing flows. For FP equations, Hu et al. [33] presented a score-based SDE solution. This paper introduces an innovative technique that combines the FDM, ADAMS, and PINN to solve the fractional Fokker–Planck–Levy (FFPL) equation.

1.2. Paper Highlights

- An innovative approach is utilized in this paper to solve the fractional Fokker–Planck–Levy (FFPL) equation. The equation contains the Levy noise and fractional Laplacian, making the equation computationally complex.
- A hybrid technique is designed by combining the finite difference method (FDM), Adams numerical technique, and physics-informed neural network (PINN) architecture, namely, the FDM-APINN, to solve the fractional Fokker–Planck–Levy (FFPL) equation numerically.
- Related work on solving partial differential equations (PDEs) and fractional partial differential equations (FPDEs) is discussed.
- The fractional Fokker–Planck-Levy (FFPL) equation is solved numerically via the proposed technique.
- The manuscript is categorized into two main scenarios by varying the value of the fractional order parameter α. The equation is solved for the two values of α, i.e., ($\alpha = 1.75, 1.85$).
- The loss values given for each case can be visualized in the tables. The loss values are very minimal, ranging between 10^{-5} and 10^{-6}, indicating the precision of the proposed technique.
- All the solutions of the proposed technique are compared with those of the score-fPINN technique, which is a well-known technique for solving fractional differential equations.
- The residual error graph and tables show the errors between both techniques. The errors range between 10^{-2} and 10^{-4}. The small error indicates the validity of our proposed technique.
- Furthermore, loss and error graphs have been added to the manuscript to explore the proposed technique further. The histogram graph shows the consistency of the proposed technique.
- All the results presented in the tables and graphs indicate that the proposed technique is a state-of-the-art technique that is robust.

2. Defining the Problem

Stochastic differential equations are commonly used to represent the time development of dynamic systems in fields such as natural sciences, engineering, economics, and statistics [34–36]. The density is obtained by solving the Fokker–Planck equation (FPE), which may be written as [37]:

$$\partial_t p_t(x) = -\nabla \times (f p_t(x) - G \nabla p_t), \tag{1}$$

where $p_t(x) \in \mathbb{R} \geq 0$ represents the value of the density at time t, $G(x,\ t)$ is a tensor called a diffusion matrix, and $f(x,\ t)$ is a vector field known as the drift function. If the α-stable Levy noise term is introduced to Equation (1) and $p_t(x)$ is replaced as $u(x,t)$, then the fractional Fokker–Planck–Levy (FFPL) equation is given as:

$$\frac{\partial u(x,\ t)}{\partial t} = -\nabla \times (f\ u(x,\ t)) + \nabla \times (\mathrm{G}\ \nabla u(x,\ t)) + \sigma\ \mathcal{L}^{\alpha}\ u(x,t), \tag{2}$$

where

- f, is defined as

$$f = -x\tanh\left(\frac{||\ x\ ||}{\sqrt{d}}\right), \tag{3}$$

➢ *G(x, t)* represents the diffusion term; in this problem, it is taken as the identity matrix.

$$G = I \tag{4}$$

where σ is the Levy noise.

➢ α is the order of the fractional Laplacian, where $0 < \alpha < 2$.
➢ $\mathcal{L}^{\alpha}$ represents the fractional Laplacian operator of order α.

Given the constants and definitions mentioned above, the equation's full form can be expressed as follows.

$$\frac{\partial u(x,\ t)}{\partial t} = -\nabla \times \left(\left(-x\tanh\left(\frac{||x||}{\sqrt{2}}\right)\right)u(x,\ t)\right) + \nabla \times (I\ \nabla u(x,\ t)) + \sigma\ \mathcal{L}^{\alpha}\ u(x,t), \tag{5}$$

$$\frac{\partial u(x,\ t)}{\partial t} = \nabla \times \left(\left(x\tanh\left(\frac{||x||}{\sqrt{2}}\right)\right)u(x,\ t)\right) + \nabla \times (\ \nabla u(x,\ t)) + \sigma\ \mathcal{L}^{\alpha}\ u(x,t), \tag{6}$$

$$\frac{\partial u(x,\ t)}{\partial t} = \nabla \times \left(\left(x\tanh\left(\frac{||x||}{\sqrt{2}}\right)\right)u(x,\ t)\right) + \left(\nabla^2 u(x,\ t)\right) + \sigma\ \mathcal{L}^{\alpha}\ u(x,t), \tag{7}$$

here,

➢ $\nabla \times \left(\left(x\tanh\left(\frac{||x||}{\sqrt{2}}\right)\right)u(x,\ t)\right)$ is the drift term.
➢ $\nabla^2 u(x,\ t)$ is the diffusion term

There are several specific distributions regarding α values:

- If $\alpha = 2$, then it is a Gaussian distribution.
- If $\alpha = 1$, then it is a Cauchy distribution.
- If $\alpha = 1.5$, then it is a Holtsmark distribution, whose PDF is a hypergeometric function.

Fractional Laplacian

The term $\mathcal{L}^{\alpha}$ represents the fractional Laplacian term. This term is beneficial for solving fractional equations involving fractional derivatives, such as the FFPL solved in this paper. The fractional Laplacian used in this paper is calculated through the finite difference method (FDM).

The fractional Laplacian $\Delta^{\frac{\alpha}{2}}$ is a nonlocal operator that generalizes the classical Laplacian Δ to fractional powers, with α generally between 0 and 2. The fractional Laplacian can be defined in various ways, but one of the most frequent uses is the spectral method or an integral representation, which reflects its nonlocal character.

In $\mathbb{R}^d$, the fractional Laplacian through the FDM of u can be defined as follows:

$$(-\Delta)^{\frac{\alpha}{2}}\, u(x_i) \approx \sum_{j \neq i} \frac{\left(u(x_j) - u(x_i)\right)}{\left|x_j - x_i\right|^{d+\alpha}} \tag{8}$$

where

- α represents the fractional order of the Laplacian.
- d is the dimension.

The domain is divided into N grid points with regular spacing dx.

$$dx = \frac{L}{N-1}, \tag{9}$$

where

$L = 2$ is the length of the domain of x ($[-1,\ 1]$).

The fractional Laplacian is represented in matrix form. The matrix D is designed to approximate the fractional Laplacian operator. The primary goal is to simulate the fractional Laplacian with a discrete sum over the grid points. The discrete fractional Laplacian is represented in matrix form as:

$$(-\Delta)^{\frac{\alpha}{2}}\, u_i \approx \sum_{j=1}^{N} D_{ij}\, u_j, \tag{10}$$

where D_{ij}, defined as

$$D_{ij} = \begin{cases} -\sum\limits_{k \neq i} \dfrac{1}{\left|x_i - x_k\right|^{d+\alpha}}, & if\ i = j \\ \dfrac{1}{\left|x_i - x_j\right|^{d+\alpha}}, & if\ i \neq j \end{cases} \tag{11}$$

The matrix element D_{ij} reflects the contribution from point x_j to the fractional Laplacian at x_i. The contribution is inversely proportional to the distance. $|x_i - x_j|$ is raised to the power $d + \alpha$. The fraction is scaled by dx to accommodate the integral's discretization. The diagonal element D_{ii} ensures that the sum of all the elements in each matrix row is zero. The Laplacian's conservation property requires that the total flux into a point x_i be zero if u is constant. The proposed flexible technique can be used in different boundary conditions and domains.

3. Proposed Methodology

Since the beginning, the PINN has undergone extensive study to improve its ability to solve complicated problems in numerous fields. Several researchers have focused on implementing advanced algorithms in PINNs, such as adaptive sampling approaches [38–42], adaptive activation functions [43–45], dynamic weighting for loss terms [46–48], and sequential training [49,50]. New architectural frameworks have been developed to address specific problems more efficiently. Some notable examples include the conservative PINN (cPINN) [51], parareal physics-informed neural network (PPINN) [52], Bayesian PINN (BPINN) [53], extended PINN (XPINN) [54,55], physics-informed generative adversarial network (PI-GAN) [56], and the gradient-enhanced neural network (gPINN) [57–59]. Although PINNs have a promising future, they have yet to be applied to some PDEs. More importantly, fractional problems, such as the fractional Fokker–Planck–Levy (FFPL) equation discussed in this paper, are challenging to solve. The FDM-APINN technique is used to solve the FFPL equation.

The proposed technique has two input values, i.e., x and t. There are two fully connected hidden layers, each having 50 neurons. The ReLU activation function is used in

the proposed approach. The deep learning phenomenon is used. The optimizer used is the Adam optimizer. The proposed technique evaluates the trained network on the test data, and the predicted solutions are obtained from the results. The loss values are obtained as the sum of two types of loss. One is data loss, and the second is physics loss. To calculate the physics loss, the fractional Laplacian and all the gradients are calculated, ensuring that all the operations comply with the FFPL equation.

The general working phenomenon of the proposed technique is shown in Figure 1. In Figure 1, the inputs are associated with hidden layers consisting of hidden neurons. After performing some hidden calculations, the output layer is obtained. After that, the loss is calculated as the sum of the data loss and physics loss. The minimum loss obtained means that the technique converges to the optimal solutions.

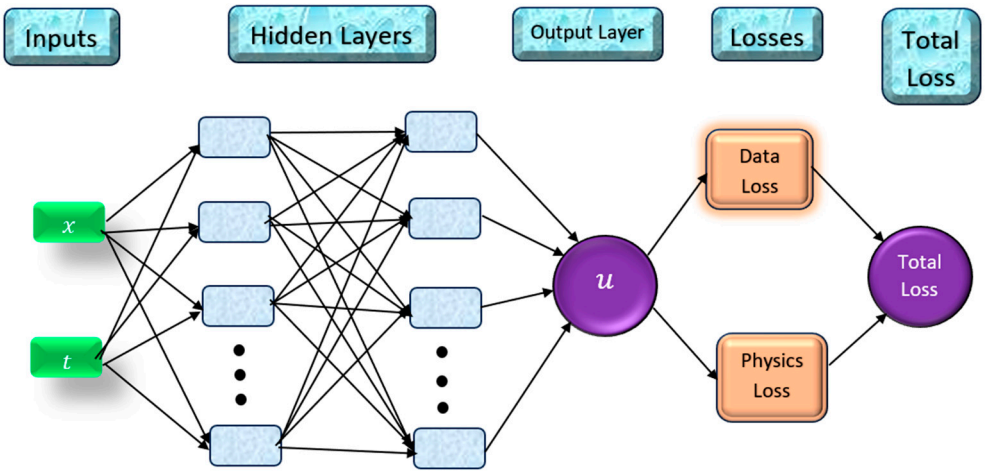


Figure 1. General working procedure of the PINN.

3.1. Loss Function

The loss function consists of two elements: data loss and physical loss. This ensures that the trained neural network matches the training data and follows the physical rules stated in the fractional Fokker–Planck–Levy (FFPL) equation.

3.1.1. Data Loss

Data loss determines how closely the neural network's predictions correspond to the training data. It is the mean squared error (MSE) between the actual predicted values.

$$Data\ loss = \frac{1}{N}\sum_{i=1}^{N}\left(u_{pred}(x_i,\ t_i\ ) - u_{actual}(x_i,\ t_i)\right)^2, \tag{12}$$

where N represents the sample of training data, $u(x_i,\ t_i)$ represents the predicted solution at (x_i, t_i), and $u_{actual}(x_i,\ t_i)$ is the actual solution at $(x_i,\ t_i)$.

3.1.2. Errors

The mathematical model under consideration in Equation (1) is given as

$$\frac{\partial u(x,\ t)}{\partial t} = -\ \nabla\times\ (f(x,\ t)u(x,\ t)) + \ \nabla\times\ (\mathrm{G(x,\ t)}\ \nabla u(x,\ t)) + \ \sigma\ \mathcal{L}^{\alpha}\ u(x,t). \tag{13}$$

The errors for the given FFPL equation are given as

$$E\left(u_{pred}\right) = \frac{\partial u(x,\ t)}{\partial t} + \ \nabla\times\ (f(x,\ t)u(x,\ t)) - \ \nabla\times\ (\mathrm{G(x,\ t)}\ \nabla u(x,\ t)) - \ \sigma\ \mathcal{L}^{\alpha}\ u(x,t). \tag{14}$$

3.1.3. Physical Loss

The physical loss for the proposed technique is defined as the mean square of the

$$Physics\ loss = \frac{1}{N}\sum_{i=1}^{N}\Big(E\Big(u_{pred}(x_i,\ t_i\ )\Big)^2. \tag{15}$$

3.1.4. Total Loss

The total loss for the proposed technique is calculated as

$$Total\ loss = Physics\ loss + Data\ loss. \tag{16}$$

In short, the total loss function combines data loss with physics loss. The proposed technique influences the rule of neural networks to predict data and ensures that the solution is physically meaningful.

The network contains an input layer for x and t, two hidden layers with ReLU activation and an output layer for $u(x,t)$. There are 50 neurons attached to each layer. Synthetic data have been generated to mimic $u(x,t)$ while training the network. Different physical parameters are used in the mathematical model. The values of the parameters of the FFPL equation are defined, including α (fractional order), σ, and the functions f and G. The network is trained via mini-batches across several epochs. The loss function combines data loss with physics loss. Gradients are calculated and utilized to adjust network settings via the Adam optimizer. The overall loss is calculated by adding data loss and physics loss. During training, the network is assessed on test data to predict $u(x,t)$, which is then visualized. This strategy integrates data-driven machine learning with the physical constraints of the FFPL equation, utilizing the strengths of each approach to discover a solution. One of the primary novelties here is the incorporation of physics loss into neural network training, which ensures that the acquired solution follows basic physical principles.

3.2. Optimizer

The optimizer used in the proposed technique is the Adam optimizer. Adam is the short form of adaptive moment estimation. It is an optimization algorithm that integrates the benefits of two widely used optimization algorithms, AdaGrad and RMSProp.

Operating of the Adam Optimizer

Set the time step (t) to 0, the 1st moment vector (m) to 0, and the 2nd moment vector (v) to 0. Define the hyperparameters. Then, the gradients are computed by

$$g_t = \nabla_\theta f(\theta_{t-1}), \tag{17}$$

update the biassed 1st moment value by

$$m_t = \beta_1 \times m_{t-1} + (1-\beta_1) \times g_t, \tag{18}$$

update the biassed 1st moment value by

$$v_t = \beta_2 \times v_{t-1} + (1-\beta_2) \times g_t^2, \tag{19}$$

compute the bias-corrected 1st moment estimates

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}, \tag{20}$$

compute the bias-corrected 2nd moment estimates

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t}, \tag{21}$$

update the parameters by

$$\theta_t = \theta_{t-1} - \alpha_a \times \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon}, \tag{22}$$

where

- t is the time step.
- θ_t represents the parameters at time step t.
- g_t is the gradient of the objective function with respect to θ_t at time step t.
- m_t is the 1st moment vector (mean of the gradients).
- v_t is the 2nd moment vector (uncentered variance of the gradients).
- α_a is the learning rate.
- β_1, β_2 are the exponential decay rates for the moment estimates.
- where ϵ is a small constant for numerical stability.

The FDM-APINN technique is used to solve the given mathematical model in the manuscript. The working phenomenon of the proposed architecture is explained in the pseudocode given in Algorithm 1.

The flow chart of the paper is given in Figure 2. All the working phenomena of the proposed technique in the manuscript are given in the flow chart of the paper in Figure 2. The flow chart shows the procedure followed in the manuscript.

Algorithm 1: Pseudocode representing all the working procedures of FDM-APINN

Starting FDM-APINN
Defining Neural Networks
Select number of input layers as ***2***
50 *is the number of hidden neurons selected*
    Parameters setting
    Select the number of iterations and learning rate
    Set values of physical parameters $\boldsymbol{\alpha}$ *and* $\boldsymbol{\sigma}$
        Generate training data
        Grid points creation of $\boldsymbol{x}$ *and* $\boldsymbol{t}$
        Set the initial condition
            Training Loop
            Select a mini-batch sample
            Prepare input data and convert it to a deep-learning array
            Compute gradients and loss functions
            Use the Adam optimizer to update the network
            Display loss
                Evaluation of the network
                Predict and plot $\boldsymbol{u}(\boldsymbol{x}, \boldsymbol{t})$ *using test data*
                    Calculate the targeted functions
                    Compute the derivatives u_x, u_t, u_{xx}.
                    Compute the function of fraction Laplacian using FDM
                    Compute the data loss function
                    Compute the physics loss functions
                    Compute gradients and total loss
End of the algorithm FDM-APNN

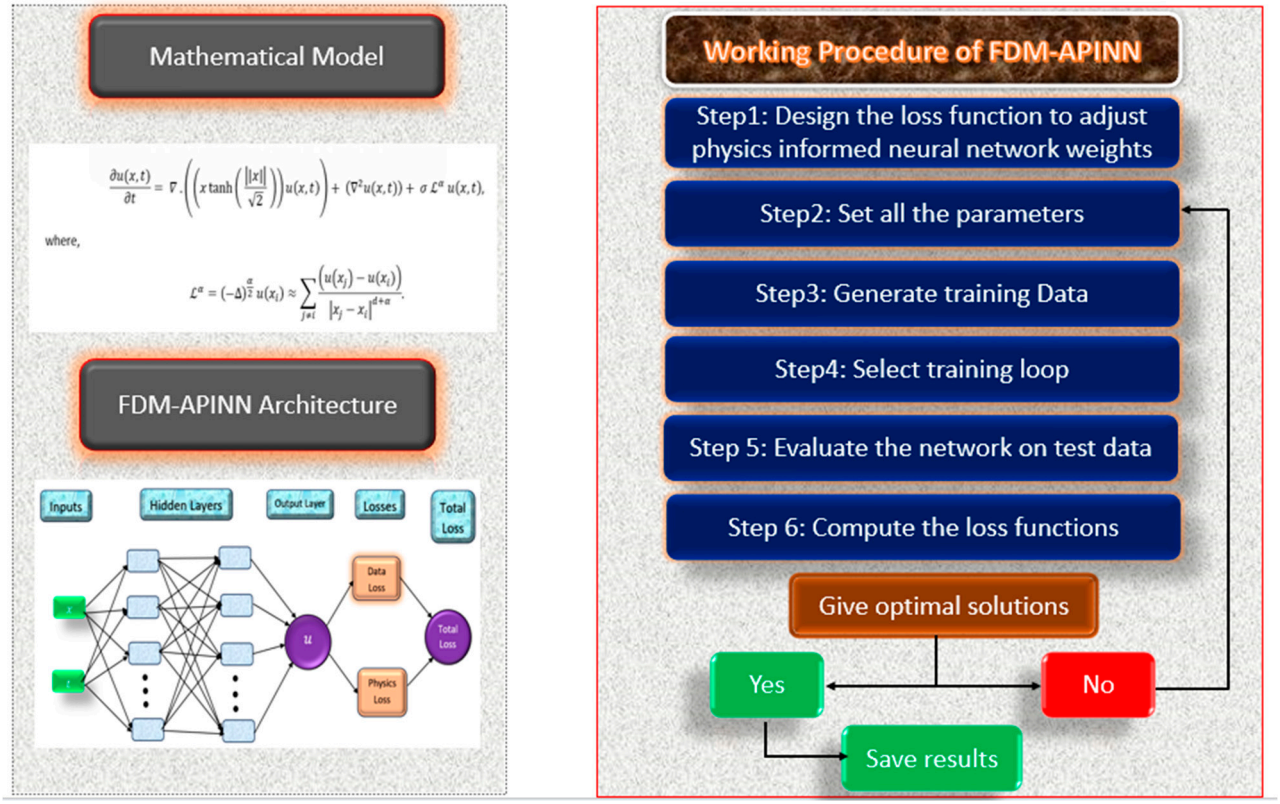


Figure 2. Flowchart for solving the fractional Fokker–Planck–Levy equation via the FDM-APINN.

The FDM-APINN methodology is used in this paper to solve the FFPL equation. The neural networks used in the PINNs are trained several times to explore the technique's efficiency thoroughly, while the fractional Laplacian term is approximated through the FDM. The paper is divided into two different scenarios by varying the value of the fractional order parameter α. For each value of α, the domain of the inputs x and t is distributed to 100, 200, and 500 different points. For each distribution, 1000, 2000, and 5000 iterations are performed. We used the outputs of the score-fPINN technique as a reference solution to validate our results. Different loss and error graphs are given for each number of iterations to evaluate the technique's validity and accuracy. From all the graphs, it is clear that the proposed technique is state-of-the-art with great accuracy. Moreover, different tables are also provided in the paper for further evaluation of the proposed method. The MATLAB code of the designed methodology is as follows: https://github.com/ffazal1/FDM_APINN.git (accessed on 9 September 2024) whenever our paper is accepted for publication.

4. Results and Discussion

This section presents all the results of the proposed technique, i.e., the FDM-APINN technique. Two scenarios are created by varying the value of the fractional order parameter α. For each value of the fractional parameter α, the domain of x and t is distributed to 100, 200, and 500 different equidistant points. Then, for each distribution, 1000, 2000, and 5000 iterations are performed. All the results obtained are given in this section. The residual error tables and graphs presented in the manuscript are the errors between the outputs of the proposed technique (FDM-APINN) and the reference technique (score-fPINN). Graphical and statistical illustrations are discussed in detail in this section.

Table 1 shows the pattern of the paper. In the first column of Table 1, the value of the fractional order α varies. The mathematical model is solved for each value, and the results are obtained via the FDM-APINN in the MATLAB window. In the first scenario, the mathematical model is solved for the value of the fractional order $\alpha = 1.75$. The second column of Table 1 shows the distribution of points taken between the domains

of x and t. We distributed the domain of x (i.e., $[-1, 1]$) into 100 different equidistant points and solved the FFPL equation. Similarly, the domain of t (i.e., $[0, 1]$) is distributed to 100 different equidistant points to obtain the solutions about 100 different points. The third row of the second column shows the distribution of the domain to 200 points. We distributed the domain of x (i.e., $[-1, 1]$) into 200 different equidistant points and solved the FFPL equation.

Table 1. Table showing all the patterns of the manuscript for both scenarios.

Scenarios	Cases	Iterations
$\alpha = 1.75$	100 points	1000 2000 5000
	200 points	1000 2000 5000
	500 points	1000 2000 5000
$\alpha = 1.85$	100 points	1000 2000 5000
	200 points	1000 2000 5000
	500 points	1000 2000 5000

Similarly, the domain of t (i.e., $[0, 1]$) is distributed to different 200 different equidistant points to obtain solutions about 200 different points. Similarly, the same procedure is repeated for the distribution of the domains into 500 points, as shown in the fourth row of the second column. After that, we trained the FDM-APINN for 1000 iterations for 100 points and obtained the results. Then, the FDM-APINN is run for 2000 iterations for the same distribution, i.e., 100 points distribution. In the final round, 5000 iterations are performed to obtain the solutions at 100 different points. In the same manner, 1000, 2000, and 5000 iterations are performed to obtain the results for 200 and 500 points for the same value of $\alpha = 1.75$. All the procedures discussed above are repeated for the values of the fractional order $\alpha = 1.85$ and $\alpha = 1.95$.

4.1. Scenario 1

This section discusses all the results obtained for the first scenario. In the first scenario, the value of the fractional order parameter α is set to 1.75. Then, three cases are formed for the first scenario by dividing the domains of x and t into 100, 200, and 500 different points. The mathematical model is solved for each case by performing 1000, 2000, and 5000 iterations. The graphical and tabular results of the first scenario are discussed in this section of the manuscript.

Table 2 shows the average loss for the value of the fractional order $\alpha = 1.75$. The first column of the table shows the 100-, 200-, and 500- point distributions between the domains of x and t. For 100 points, 1000, 2000, and 5000 iterations are performed for each distribution individually, as given in the second column of Table 2. The third column of Table 2 shows the average loss obtained after 1000, 2000, and 5000 iterations. Table 2 shows that the average loss obtained in this paper is minimal, which confirms the validity of our proposed technique. Each value of the average loss is in the range of 10^{-5} and 10^{-6}. This finding indicates that our proposed technique is highly consistent.

Table 2. Average loss values after performing 1000, 2000, and 5000 iterations of the first scenario for 100-, 200-, and 500-point distributions of the domains, respectively.

$\alpha = 1.75$		
Points	**Iterations**	**Average Loss**
100	1000	1.79×10^{-5}
	2000	4.69×10^{-6}
	5000	1.05×10^{-6}
200	1000	3.17×10^{-6}
	2000	1.27×10^{-6}
	5000	2.06×10^{-6}
500	1000	5.56×10^{-6}
	2000	2.21×10^{-6}
	5000	1.12×10^{-6}

Figure 3 shows a graphical illustration of the solution graphs of the fractional Fokker–Planck–Levy (FFPL) equation. From Figure 3, the density $u(x, t)$ is the maximum at the mean and decreases as we move towards the extreme points. The maximum value of the density is 1. Solution graphs obtained after solving the fractional Fokker–Planck–Levy (FFPL) equation for 100-, 200-, and 500-point distributions. Five thousand iterations are performed for each case.

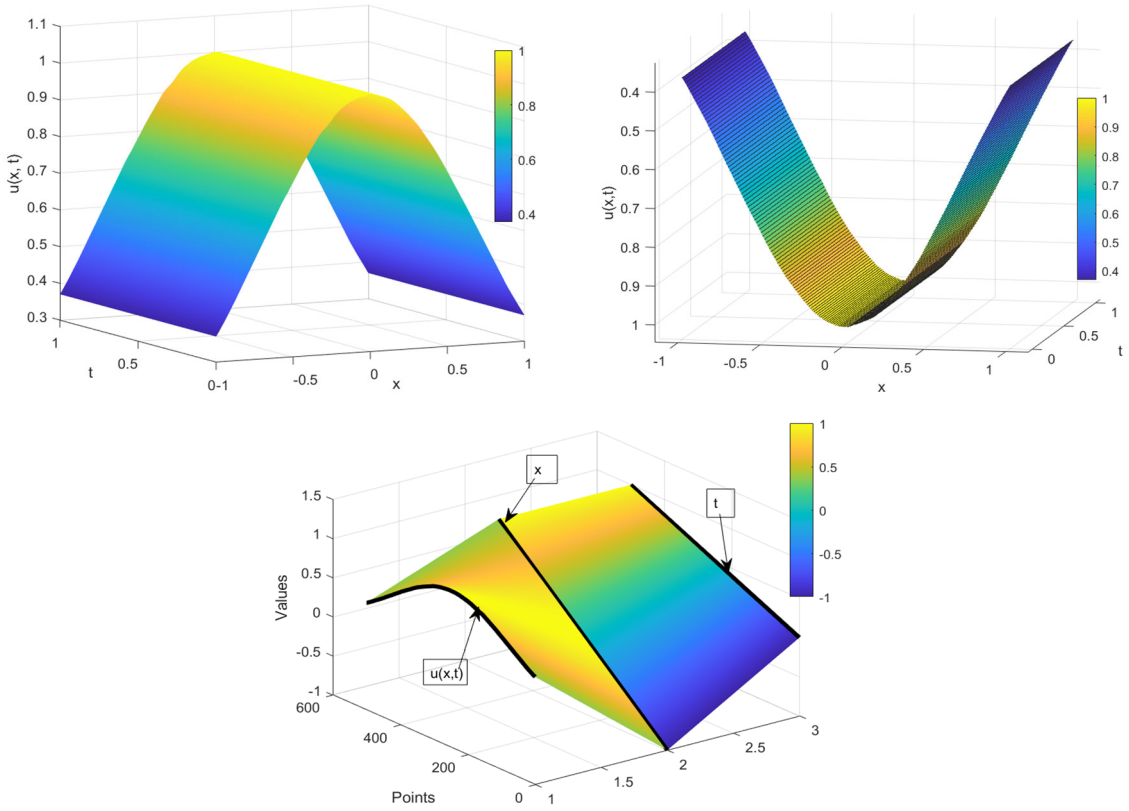


Figure 3. Solution graphs of the fractional Fokker–Planck–Levy equation.

After performing 1000, 2000, and 5000 iterations, the residual error values are obtained, as shown in Table 3. These values are obtained by taking 100 different points between the domains of x and t, while the value of the fractional parameter is $\alpha = 1.75$. Table 3 shows the residual error values between the solutions of the proposed technique and the score-fPINN. The first column of the table shows the input x values. These values are obtained for the distribution of the domains of x and t into 100 different points. The second

column of the table shows the input t values. The third column represents the residual error values after 1000 iterations are performed. Similarly, the fourth and five columns show the residual error values after performing 2000 and 5000 iterations, respectively. The table clearly shows that the residual errors are close to 0, ranging between 10^{-2} and 10^{-4}, which demonstrates that the proposed technique is a state-of-the-art, accurate, and valid technique.

Table 3. Residual error values of the 100-point distribution when the value of the fractional order parameter is $\alpha = 1.75$.

x	t	Residual Errors (1000 Iterations)	Residual Errors (2000 Iterations)	Residual Errors (5000 Iterations)
−1.00	0.00	-2.29×10^{-2}	2.49×10^{-2}	-1.60×10^{-3}
−0.80	0.10	1.90×10^{-3}	-5.20×10^{-3}	-7.00×10^{-4}
−0.60	0.20	5.90×10^{-3}	-4.00×10^{-3}	2.20×10^{-3}
−0.40	0.30	2.80×10^{-3}	-8.80×10^{-3}	-8.00×10^{-4}
−0.20	0.40	-9.50×10^{-3}	8.60×10^{-3}	1.40×10^{-3}
0.00	0.50	-1.00×10^{-3}	4.00×10^{-3}	2.10×10^{-3}
0.20	0.60	-1.00×10^{-4}	3.20×10^{-3}	-1.70×10^{-3}
0.40	0.70	-5.20×10^{-3}	6.00×10^{-3}	-2.20×10^{-3}
0.60	0.80	7.70×10^{-3}	1.90×10^{-3}	-1.20×10^{-3}
0.80	0.90	3.60×10^{-3}	-4.00×10^{-3}	6.00×10^{-4}
1.00	1.00	9.90×10^{-3}	5.00×10^{-4}	1.60×10^{-3}

Figure 4a shows the graph of the loss values. Figure 4a shows the 1000 iterations for the 100-point distribution of the domain of x and t, while the value of the fractional order is $\alpha = 1.75$. The x-axis of the graph represents the number of iterations, and the y-axis represents the loss values. From Figure 4a, the loss value decreases and approaches zero as the number of iterations increases, which shows that we are approaching the optimal solutions. At the beginning, the loss value is not close to zero, but at the 1000th iteration, the loss value is very close to zero. This means that as we approach the 1000th iteration, the solutions approach their optimal values. Figure 4b shows the graph of the residual error values between the proposed technique and the score-fPINN technique. Figure 4b shows the errors obtained during the 1000 iterations for the 100-point distributions of the domains of x and t. Figure 4b shows the values of x, t, and the corresponding errors. All 100 points lie between 10^{-2} and 10^{-3}, which shows that minimal residual errors are present. These minimal residual errors indicate the accuracy and consistency of the proposed technique.

The graph of the loss values of the proposed technique is shown in Figure 4c. Figure 4c shows the loss values for 2000 iterations of the 100-point distribution of the domains of x and t, where the value of the fractional order parameter is $\alpha = 1.75$. The x-axis of the graph represents the number of iterations, and the y-axis shows the loss values. Figure 4c shows that the loss value decreases and approaches zero as the number of iterations increases. This shows that we are approaching the optimal solutions. At the start of the iterations, the loss value is not close to zero, but after 500 iterations, the loss values oscillate around 10^{-5}. As we approach 2000 iterations, the loss value approaches 0. This means that as we approach the 2000th iteration, the solutions approach their best optimal values. Figure 4d shows the graph of the residual error values. Figure 4d shows the errors obtained during the 2000 iterations for the 100-point distributions of the domains of x and t. Figure 4d shows the values of x, t, and the corresponding errors. We take 11 points to observe the residual errors. Figure 4d shows that the errors are very close to zero, indicating the accuracy and consistency of our proposed technique.

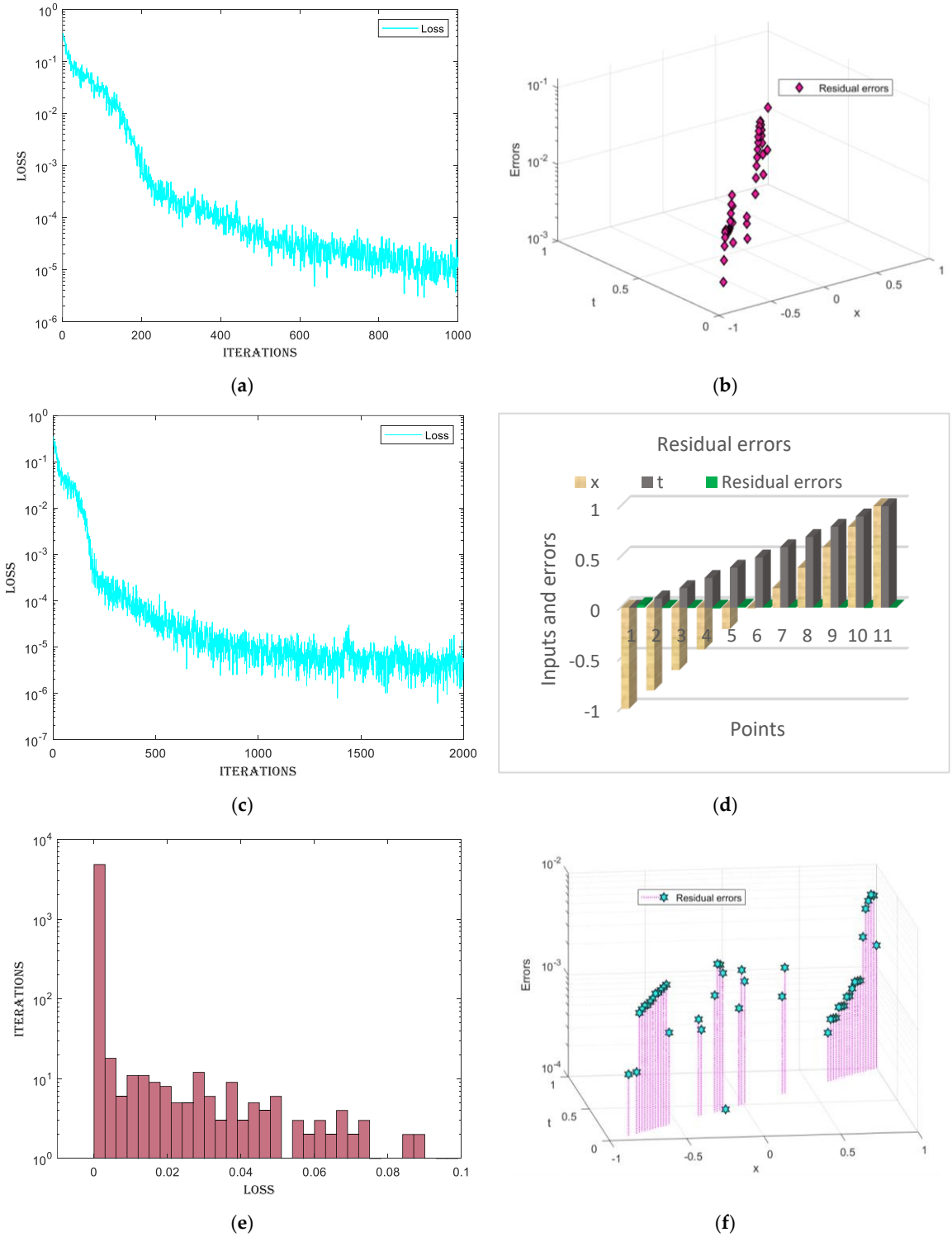


Figure 4. Graphs of the loss and residual error values for the 100-point distributions of the domains of x and t after 1000, 2000, and 5000 iterations. (**a**) Loss value graph for 1000 iterations for the 100-point distribution. (**b**) Residual error graph After 1000 iterations for 100 points. (**c**) Loss value graph for 2000 iterations for the 100-point distribution. (**d**) Residual error graph after 2000 iterations for 100 points. (**e**) Graphical illustration of 5000 loss values for 100 points distribution of the domain of x and t. (**f**) Graphical illustration of the error values after 5000 iterations performed for 100 points distribution of the domain of x and t..

Figure 4e shows the histogram graph for the loss values. Figure 4e shows the loss values for 5000 iterations of the 100-point distribution of the domains of x and t, where the value of the fractional order parameter is $\alpha = 1.75$. The x-axis of the graph represents the loss values, and the y-axis of the graph represents the number of iterations. The graph shows that at approximately 3700 iterations, the loss values are the same and very close to 0. This finding indicates that our proposed technique is highly consistent and reliable. Figure 4f shows the graph of the residual error values. Figure 4f shows the errors obtained during the 5000 iterations for the 100-point distributions of the domains of x and t. Figure 4f shows the x, t, and residual error values. All the error values lie between 10^{-2} and 10^{-4}. Figure 4f shows that the errors are very close to zero, indicating the accuracy and consistency of our proposed technique.

Table 4 shows the residual error values between the actual and predicted values. The domains of x and t are divided into 100 different equidistant points. The fractional order α is 1.75. In Table 4, the first column shows the input x values, whereas the second column represents the input t values. The third, fourth, and fifth columns show the residual errors. The third column shows the residual error values after performing 1000 iterations. Similarly, after 2000 iterations, the residual errors obtained are shown in the fourth column. The fifth column shows the residual error values after performing 5000 iterations. All the values of the residual errors are minimal, confirming the validity of the technique. Furthermore, the residual error values are also very close to each other, indicating the consistency of the proposed technique. The residual error values range from 10^{-2} to 10^{-4}, which is very good for fractional problems.

Table 4. Tabular illustration of the error values of the 200-point distribution when $\alpha = 1.75$.

x	t	Residual Error (1000 Iterations)	Residual Error (2000 Iterations)	Residual Error (5000 Iterations)
−1.00	0.00	1.51×10^{-2}	1.70×10^{-3}	-3.40×10^{-3}
−0.80	0.10	5.70×10^{-3}	3.50×10^{-3}	-1.10×10^{-3}
−0.60	0.20	1.00×10^{-3}	-8.00×10^{-4}	1.30×10^{-3}
−0.40	0.30	-4.50×10^{-3}	4.00×10^{-4}	-4.00×10^{-4}
−0.20	0.40	3.30×10^{-3}	2.40×10^{-3}	2.60×10^{-3}
0.00	0.50	3.10×10^{-3}	-9.00×10^{-4}	-1.90×10^{-3}
0.20	0.60	-1.60×10^{-3}	-1.10×10^{-3}	9.00×10^{-4}
0.40	0.70	-7.00×10^{-3}	-1.50×10^{-3}	9.00×10^{-4}
0.60	0.80	3.70×10^{-3}	-2.10×10^{-3}	-3.00×10^{-3}
0.80	0.90	-8.30×10^{-3}	0.00	-1.80×10^{-3}
1.00	1.00	1.00×10^{-3}	4.60×10^{-3}	-2.30×10^{-3}

Figure 5a shows the graph for the loss values. Figure 5a shows the loss values for 1000 iterations of the 200-point distribution of the domains of x and t, where the value of the fractional order parameter is $\alpha = 1.75$. The x-axis of the graph represents the number of iterations, and the y-axis of the graph represents the loss values. Figure 5a shows that as the number of iterations increases, the loss value decreases and approaches zero. This shows that we are approaching the optimal solutions. At the start of the iterations, the loss value is not close to zero, but after 1000th iterations, the loss value approaches 0. As we reach the 1000 iterations, the loss value approaches 0. Figure 5b shows the graph of the residual error values in the output of the proposed technique and the score-fPINN technique. Figure 5b shows the errors obtained during the 1000 iterations for the 200-point distributions of the domains of x and t. Figure 5b shows the values of x, t, and the corresponding errors. All 200 points have very minimal residual error values, as shown in Figure 5b, indicating the accuracy and consistency of our proposed technique.

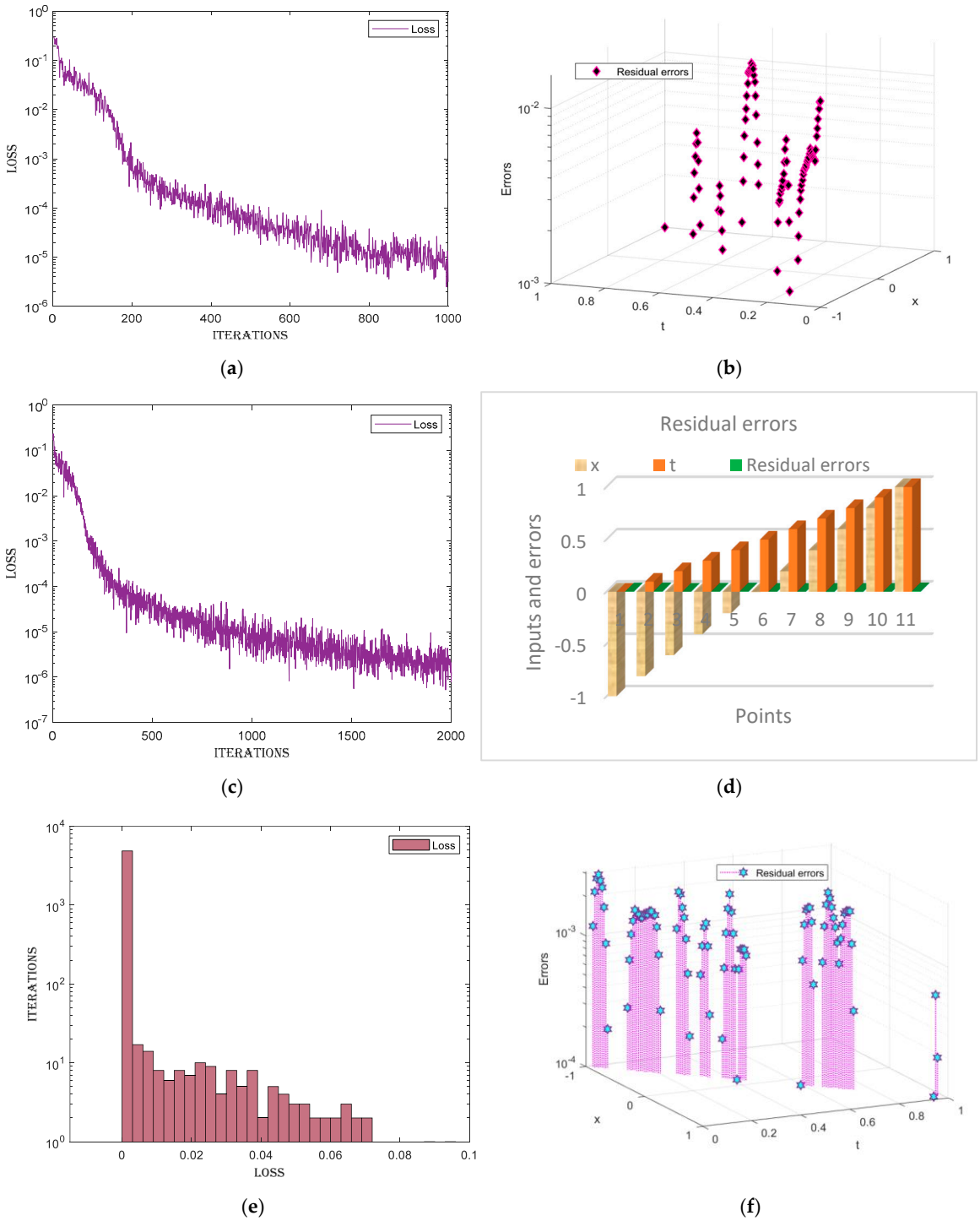


Figure 5. Graphical illustration of the loss and residual error values after performing 1000, 2000, and 5000 iterations, respectively, for the 200-point distributions of the domains of x and t. (**a**) Graphical representation of loss values after 1000 iterations for the 200-point distribution when value of $\alpha = 1.75$. (**b**) The residual error graph shows an 200 error values after performing 1000 iterations when $\alpha = 1.75$. (**c**) Graph of the loss values after performing 2000 iterations for 200 points distribution for the first scenario. (**d**) Residual error graph when the domain of inputs is distributed to 200 different points when the value of $\alpha = 1.75$. (**e**) Error histogram graph of the loss values each iteration. (**f**) Graph of the residual errors after performing 5000 iterations.

The loss value graph is shown in Figure 5c. The graph shown in Figure 5c is for the loss values obtained after approximately 2000 iterations by taking 200 different points

between the domains of x and t. The value of the fractional parameter α is 1.75. The iterations are represented through the x-axis, while the loss values are shown on the y-axis of Figure 5c. As the number of iterations increases, the loss value decreases and approaches zero, indicating that we are approaching the optimal solutions. At the beginning of the graph, the loss values are not close to zero, whereas at the end of the graph, the values are close to zero. Additionally, at the start of the graph, the loss values vary, but after 1000 iterations, the graph becomes smooth, showing that the solutions obtained through the proposed technique approach optimal solutions. The error values are shown in Figure 5d. These errors are obtained through 200 different points between the domains of x and t. 2000 iterations are performed to obtain these errors. We take 11 different points from the 200 points. Clearly, the errors are very close to 0, confirming the validity and consistency of the proposed technique.

The graph shown in Figure 5e is for the loss values obtained after approximately 5000 iterations by taking 200 different points between the domains of x and t. The value of the fractional parameter α is 1.75. The iterations are represented through the y-axis, while the loss values are shown on the x-axis of Figure 5e. In approximately 3800 iterations, the loss values overlap each other. The histogram in Figure 5e shows the consistency of our proposed technique. The residual error values are shown in Figure 5f. These are the errors obtained through 200 different points between the domains of x and t. These residual errors were obtained after 5000 iterations. Clearly, the errors are very close to 0, confirming the validity and consistency of the proposed technique.

Table 5 shows the residual error values between the actual and predicted values. The domains of x and t are divided into 500 different equidistant points. The fractional order α is 1.75. The table's first column shows the input x values, whereas the second column represents the input t values. The third, fourth and fifth columns show the residual errors. The third column shows the residual error values after performing 1000 iterations. Similarly, after 2000 iterations, the residual errors obtained are demonstrated by the fourth column. Moreover, the fifth column shows the residual error values after performing 5000 iterations. All the values of the residual errors are minimal, confirming the technique's validity. Furthermore, the residual error values are also very close to each other, confirming the consistency of the proposed technique. The residual error values range from 10^{-2} to 10^{-3}, which is very good for fractional problems.

Table 5. Residual error values for the proposed technique and the reference solutions of the 500-point distribution when $\alpha = 1.75$.

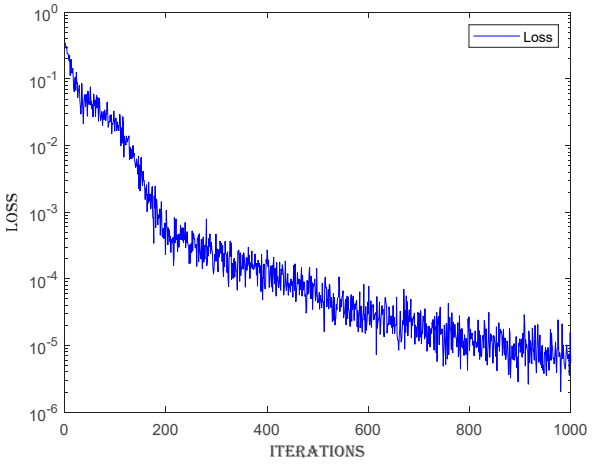
x	t	Residual Error (1000 Iterations)	Residual Error (2000 Iterations)	Residual Error (5000 Iterations)
−1.00	0.00	1.30×10^{-3}	2.80×10^{-3}	-7.00×10^{-4}
−0.80	0.10	7.10×10^{-3}	2.40×10^{-3}	-2.20×10^{-3}
−0.60	0.20	7.20×10^{-3}	3.70×10^{-3}	-3.70×10^{-3}
−0.40	0.30	-6.00×10^{-4}	1.20×10^{-3}	2.10×10^{-3}
−0.20	0.40	-6.40×10^{-3}	1.80×10^{-3}	-1.70×10^{-3}
0.00	0.50	2.30×10^{-3}	-5.10×10^{-3}	3.20×10^{-3}
0.20	0.60	-5.10×10^{-3}	4.10×10^{-3}	6.20×10^{-3}
0.40	0.70	-5.20×10^{-3}	1.46×10^{-2}	-2.00×10^{-4}
0.60	0.80	-1.33×10^{-2}	6.00×10^{-3}	5.80×10^{-3}
0.80	0.90	-9.40×10^{-3}	6.50×10^{-3}	2.30×10^{-3}
1.00	1.00	1.00×10^{-4}	6.00×10^{-3}	2.00×10^{-4}

Figure 6a shows the graph for the loss values. Figure 6a shows the loss values for 1000 iterations of the 500-point distribution of the domains of x and t, where the value of the fractional order parameter is $\alpha = 1.75$. The x-axis of the graph represents the number of iterations, and the y-axis of the graph represents the loss values. Figure 6a shows that the loss value decreases and approaches zero as the number of iterations increases. This

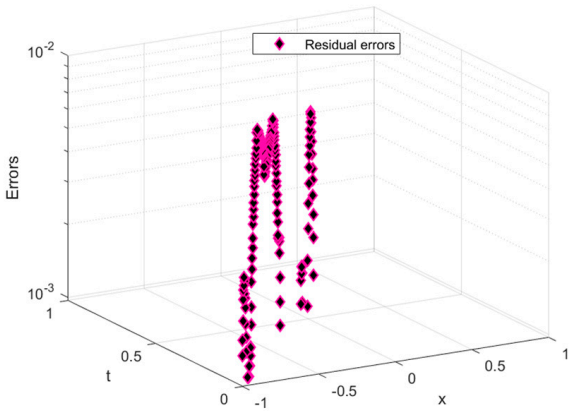
indicates that the proposed technique converges toward the optimal solutions. At the start of the iterations, the loss value is not close to zero, but at the 1000th iteration, the loss value approaches 0. As we reach the 1000 iterations, the loss value approaches 0. Figure 6b shows the graph of the residual error values. Figure 6b shows the errors obtained during the 1000 iterations for the 500-point distributions of the domains of x and t. Figure 6b shows the x, t, and error values. All the 500 error values range from 10^{-2} to 10^{-4}. In Figure 5b, the errors are very close to zero, indicating the accuracy and consistency of our proposed technique.

The loss value graph is shown in Figure 6c. The graph shown in Figure 6c is for the loss values obtained after approximately 2000 iterations by taking 500 different points between the domains of x and t. The value of the fractional parameter α is 1.75. The iterations are represented through the x-axis, while the loss values are shown on the y-axis of Figure 6c. As the number of iterations increases, the loss value decreases and approaches zero, indicating that the optimal solutions are obtained. At the beginning of the graph, the loss values are not close to zero, whereas at the end, the values are close to zero. Additionally, at the start of the graph, the loss values vary, but after 1000 iterations, the graph becomes smooth, showing that the solutions obtained through the proposed technique approach optimal solutions. The error values are shown in Figure 6d. These errors were obtained through 500 different points between the domains of x and t. 2000 iterations are performed to obtain these errors. We take 11 different points from the 500 points. The errors are very close to 0, confirming the validity and consistency of the proposed technique.

Figure 6e shows the graph for the loss values. Figure 6f shows the loss values for 5000 iterations of the 500-point distribution of the domains of x and t, where the value of the fractional order parameter is $\alpha = 1.75$. The x-axis of the graph represents the loss values, and the y-axis of the graph represents the number of iterations. The histogram graph shows the consistency of the proposed technique. The graph is given for 5000 iterations. Out of 5000 iterations, approximately 3800 iterations of the loss values overlap. Figure 6f shows the graph of the residual error values. Figure 6f shows the errors obtained during the 5000 iterations for the 500-point distributions of the domains of x and t. Figure 6f shows the values of x, t and the corresponding errors. Figure 6f shows that the errors are very close to zero for all 500 points, indicating the accuracy and consistency of our proposed technique.

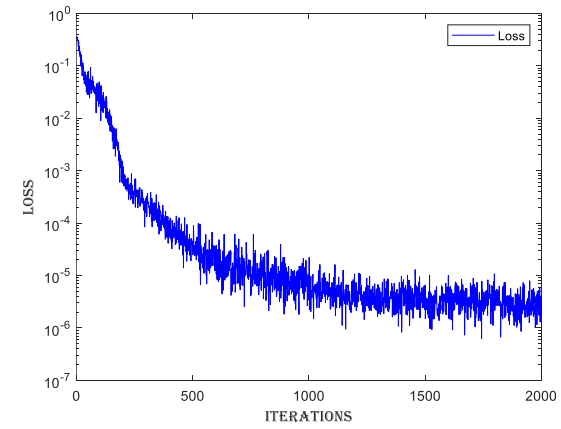


(**a**)

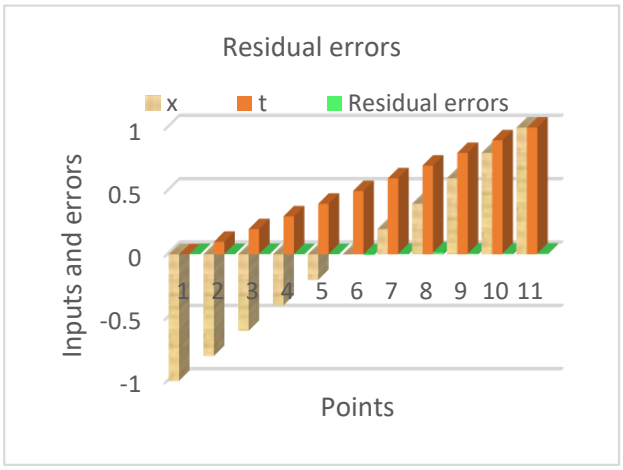


(**b**)

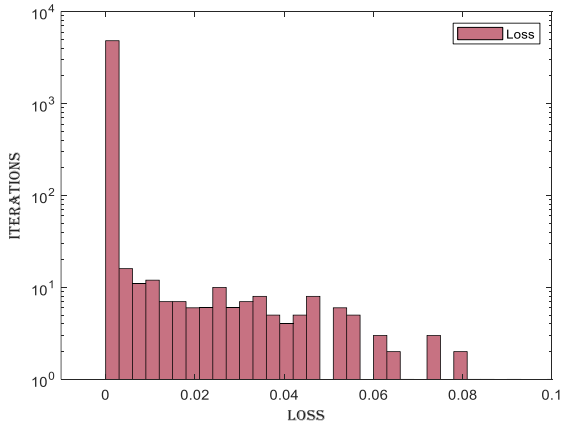
Figure 6. *Cont.*



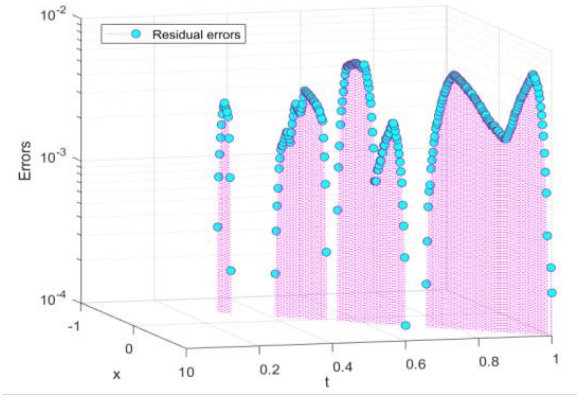
(**c**)



(**d**)



(**e**)



(**f**)

Figure 6. Graphical illustration of all the loss and residual error values for 500. Distribution of the domain when the value of the fractional order parameter $\alpha = 1.75$. (**a**) Graph of the loss values of 1000 iterations for 500 points when the value of the fractional order is $\alpha = 1.75$. (**b**) graphical illustration of all the 500 residual error values for the first scenario. (**c**) Total loss values of 2000 iterations for the first scenario of 500 distribution points. (**d**) Residual errors when the value of $\alpha = 1.75$, after 2000 Iterations for the 500 point distribution. (**e**) Error histogram graph of all the 5000 loss values when value of the fractional order $\alpha = 1.75$, while the domain is distributed to 500 different points. (**f**) Graphical representation of all the 500 residual error values of the first scenario.

4.2. Scenario 2

In this section, all the results obtained for the second scenario are discussed. In the second scenario, the fractional order parameter alpha value is 1.85. Three cases are formed for the second scenario by dividing the domains of x and t into 100, 200, and 500 different points. The mathematical model is solved for each case by performing 1000, 2000, and 5000 iterations. All the graphical and tabular results of the second scenario are discussed in this section of the manuscript.

Table 6 shows the average loss for the value of the fractional order. $\alpha = 1.75$. The first column of Table 6 shows the 100-, 200-, and 500-point distributions between the domains of x and t. For 100 points, 1000, 2000, and 5000 iterations are performed for each distribution individually, as given in the second column of Table 6. The third column of Table 6 shows the average loss obtained after 1000, 2000, and 5000 iterations. The third column of Table 6 shows that the average loss obtained in this paper is minimal, which shows the validity of our technique. Each value of the average loss ranges between 10^{-5} and 10^{-6}. This shows the consistency of our technique.

Table 6. Residual error values of the 100-point distribution when the value of the fractional order parameter is $\alpha = 1.85$.

α = 1.75		
Points	**Iterations**	**Average Loss**
100	1000	3.38×10^{-5}
	2000	2.63×10^{-6}
	5000	1.90×10^{-6}
200	1000	9.25×10^{-6}
	2000	5.61×10^{-6}
	5000	3.30×10^{-6}
500	1000	2.36×10^{-5}
	2000	4.27×10^{-6}
	5000	4.33×10^{-6}

Figure 7 shows a graphical illustration of the solution graphs of the fractional Fokker–Planck–Levy (FFPL) equation. From Figure 7, the density $u(x,\ t)$ is the maximum at the mean and decreases as we move towards the extreme points. The maximum value of the density is 1. These are the solutions obtained after performing 5000 iterations for the 100, 200, and 500 distributions, respectively.

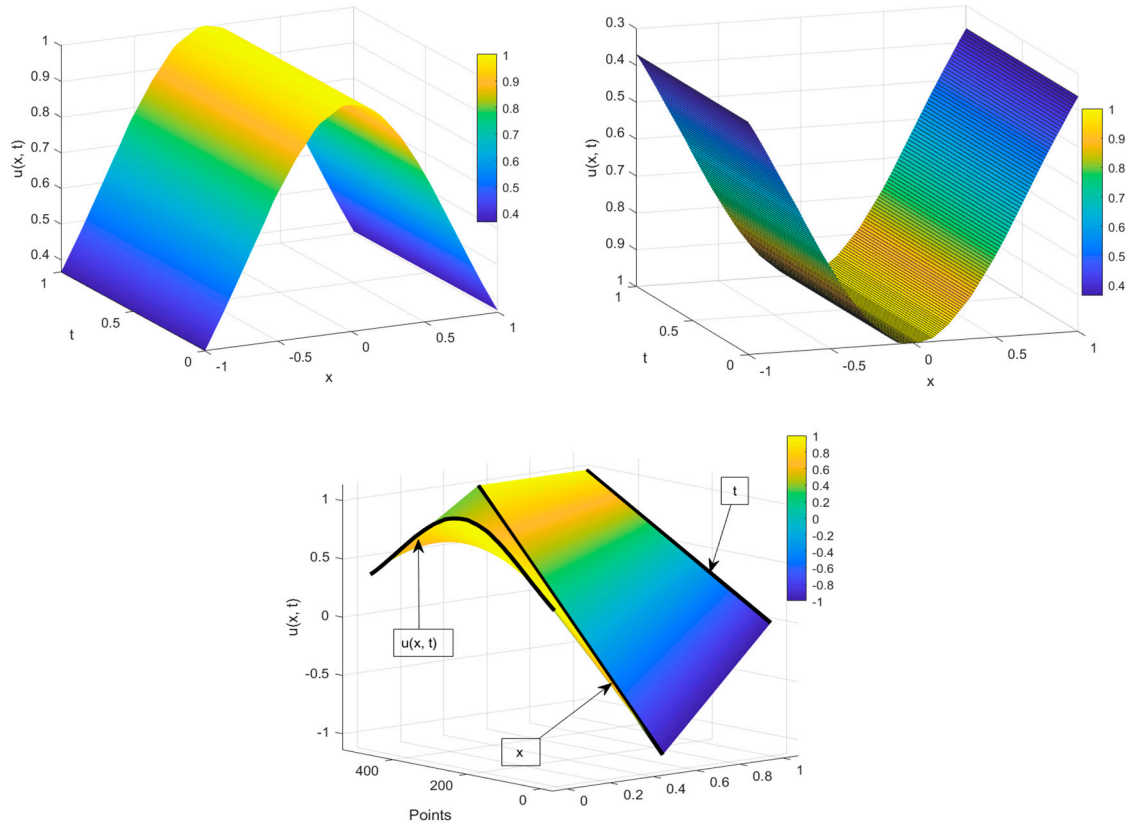


Figure 7. Solution graphs of the FFPL equation via the FDM-APINN.

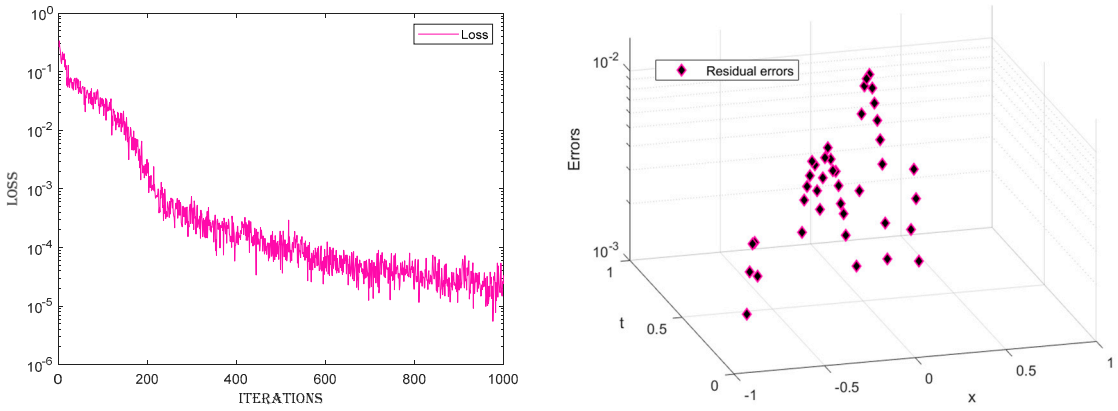
After performing 1000, 2000, and 5000 iterations, respectively. The residual error values are obtained, as shown in Table 7. These values are obtained by taking 100 different points between the domains of x and t, while the value of the fractional parameter is $\alpha = 1.85$. Table 7 shows the residual errors obtained from the proposed technique. The first column of the table shows the input x values. These values are obtained for the distribution of the domains of x and t into 100 different points. The second column of the table shows

the input t values. The third column of the table represents the residual error values after 1000 iterations are performed. Similarly, the fourth and fifth columns show the residual error values after performing 2000 and 5000 iterations, respectively. Table 7 clearly shows that the residual errors are close to 0, ranging between 10^{-2} and 10^{-4}, which demonstrates that the proposed technique is a state-of-the-art, accurate, and valid technique.

Table 7. Residual error values for the proposed technique and the reference solutions of the 100-point distribution when $\alpha = 1.85$.

x	t	Residual Error (1000 Iterations)	Residual Error (2000 Iterations)	Residual Error (5000 Iterations)
−1.00	0.00	-5.00×10^{-4}	3.10×10^{-3}	-1.36×10^{-2}
−0.80	0.10	2.80×10^{-3}	1.60×10^{-3}	-6.00×10^{-3}
−0.60	0.20	-1.80×10^{-3}	-2.70×10^{-3}	-2.40×10^{-3}
−0.40	0.30	6.60×10^{-3}	-1.18×10^{-2}	-3.20×10^{-3}
−0.20	0.40	5.80×10^{-3}	9.00×10^{-4}	3.20×10^{-3}
0.00	0.50	9.80×10^{-3}	-1.80×10^{-3}	6.00×10^{-4}
0.20	0.60	1.40×10^{-3}	-2.30×10^{-3}	-1.90×10^{-3}
0.40	0.70	3.50×10^{-3}	1.30×10^{-3}	3.10×10^{-3}
0.60	0.80	-6.00×10^{-3}	1.40×10^{-3}	7.00×10^{-3}
0.80	0.90	-7.60×10^{-3}	-6.00×10^{-4}	6.40×10^{-3}
1.00	1.00	5.00×10^{-4}	-3.60×10^{-3}	8.10×10^{-3}

Figure 8a shows the graph for the loss values. Figure 8a shows the loss values for 1000 iterations of the 100-point distribution of the domains of x and t, where the value of the fractional order parameter is $\alpha = 1.85$. The x-axis of the graph represents the number of iterations, and the y-axis of the graph represents the loss values. Figure 8a shows that the loss value decreases and approaches zero as the number of iterations increases. This indicates that the proposed technique converges toward the optimal solutions. At the start of the iterations, the loss value is not close to zero, but at the 1000th iteration, the loss value approaches zero. As we reach the 1000 iterations, the loss value approaches 0. Figure 8b shows the graph of the residual error values. Figure 8b shows the errors obtained during the 1000 iterations for the 100-point distributions of the domains of x and t. Figure 8b shows the values of x, t, and the corresponding errors. All the 100 error values range from 10^{-2} to 10^{-3}. Figure 8b shows that the errors are very close to zero, indicating the accuracy and consistency of our proposed technique.



(**a**) (**b**)

Figure 8. *Cont.*

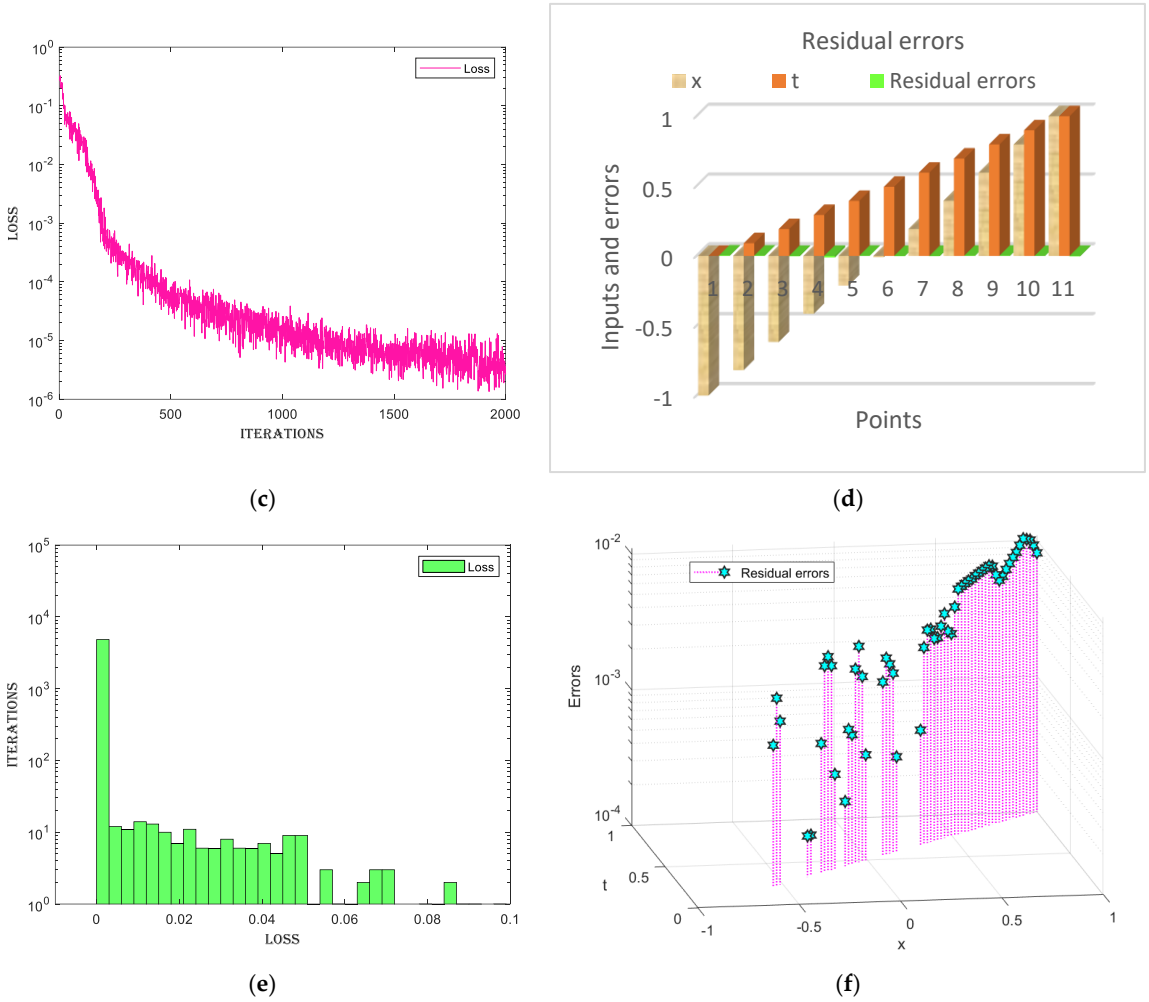


Figure 8. Graphs of the loss and residual error values for the 100-point distribution of the domains of x and t after 1000, 2000, and 5000 iterations when the value of the fractional order is $\alpha = 1.85$. (**a**) Graph of the loss values of 1000 iterations for 100 values when the value of the fractional order is $\alpha = 1.85$. (**b**) graphical illustration of all the 100 residual error values for the second scenario. (**c**) Graphical representation of loss values after 2000 iterations for the 100-point distribution when value of $\alpha = 1.85$. (**d**) The residual error graph shows an 100 error values after performing 2000 iterations when $\alpha = 1.85$. (**e**) Graphical illustration of the 5000 loss values for 100 points distribution of the domains of x and t. (**f**) Graphical illustration of the error values after 5000 iterations 100 points distribution of the domains of x and t.

The loss value graph is shown in Figure 8c. The graph shown in Figure 8c is for the loss values obtained after approximately 2000 iterations by taking 100 different points between the domains of x and t. The value of the fractional parameter α is 1.85. The iterations are represented through the x-axis, while the loss values are shown on the y-axis of Figure 8c. As the number of iterations increases, the loss value decreases and approaches zero, indicating that the optimal solutions are obtained. At the beginning of the graph, the loss values are not close to zero, whereas at the end, the values are close to zero. Additionally, at the start of the graph, the loss values vary, but after 1500 iterations, the graph becomes smooth, showing that the solutions obtained through the proposed technique approach optimal solutions. The error values are shown in Figure 8d. These are the errors obtained through 100 different points between the domains of x and t. 2000 iterations are performed to obtain these errors. We take 11 different points from the 100 points. Clearly, the errors are very close to 0, confirming the validity and consistency of the proposed technique.

The graph shown in Figure 8e is for the loss values obtained after approximately 5000 iterations by taking 100 different points between the domains of x and t. The value of the fractional parameter α is 1.85. The iterations are represented through the y-axis, while the loss values are shown on the x-axis of Figure 8e. In approximately 3800 iterations, the loss values overlap each other. The histogram in Figure 8e shows the consistency of our proposed technique. The residual error values are shown in Figure 8f. These are the errors obtained through 100 different points between the domains of x and t. 5000 iterations are used to obtain these residual errors. The errors are very close to those of 0, clearly showing the validity and consistency of the proposed technique.

Table 8 shows the residual error values between the actual and predicted values. The domains of x and t are divided into 200 different equidistant points. The fractional order α is 1.85. In Table 8, the first column shows the input x values, whereas the second column represents the input t values. The third, fourth and fifth columns show the residual errors. The third column shows the residual error values after performing 1000 iterations. Similarly, after 2000 iterations, the residual errors obtained are shown in the fourth column. Moreover, the fifth column shows the residual error values after performing 5000 iterations. All the values of the residual errors are minimal, confirming the technique's validity. Furthermore, the residual error values are also very close to each other, indicating the consistency of the proposed method. The residual error values range from 10^{-2} to 10^{-4}, which is very good for fractional problems.

Table 8. Residual error values between the proposed technique and the reference solutions of the 200-point distribution when $\alpha = 1.85$.

x	t	Residual Error (1000 Iterations)	Residual Error (2000 Iterations)	Residual Error (5000 Iterations)
−1.00	0.00	2.17×10^{-2}	3.10×10^{-3}	1.90×10^{-3}
−0.80	0.10	2.30×10^{-3}	-6.00×10^{-4}	4.10×10^{-3}
−0.60	0.20	-3.10×10^{-3}	3.00×10^{-4}	3.30×10^{-3}
−0.40	0.30	-4.00×10^{-3}	2.00×10^{-4}	2.40×10^{-3}
−0.20	0.40	3.30×10^{-3}	9.00×10^{-4}	-4.10×10^{-3}
0.00	0.50	4.60×10^{-3}	1.90×10^{-3}	-2.50×10^{-3}
0.20	0.60	6.00×10^{-3}	9.60×10^{-3}	-3.20×10^{-3}
0.40	0.70	7.30×10^{-3}	1.60×10^{-3}	-2.30×10^{-3}
0.60	0.80	1.06×10^{-2}	-7.00×10^{-4}	-4.20×10^{-3}
0.80	0.90	-4.00×10^{-4}	3.30×10^{-3}	-6.50×10^{-3}
1.00	1.00	8.80×10^{-3}	7.90×10^{-3}	-1.21×10^{-2}

The graph of the loss values is shown in Figure 9a. Figure 9a shows the loss values for 1000 iterations of the 200-point distribution of the domains of x and t, where the value of the fractional order parameter is $\alpha = 1.85$. The x-axis of the graph represents the number of iterations, and the y-axis represents the loss values. From Figure 9a, the loss value decreases and approaches zero as the number of iterations increases, which shows that we are approaching the optimal solutions. At the start of the iterations, the loss value is not close to zero, but at the 1000th iteration, the loss value is very close to zero. This means that as we approach the 1000th iteration, the solutions approach their best optimal values. Figure 9b shows the graph of the residual error values. Figure 4b shows the errors obtained during the 1000 iterations for the 200-point distributions of the domains of x and t. Figure 9b shows the x, t, and error values. All the 200 error values range from 10^{-2} to 10^{-3}. Figure 4b shows that the errors are very close to 0, indicating the accuracy and consistency of our proposed technique.

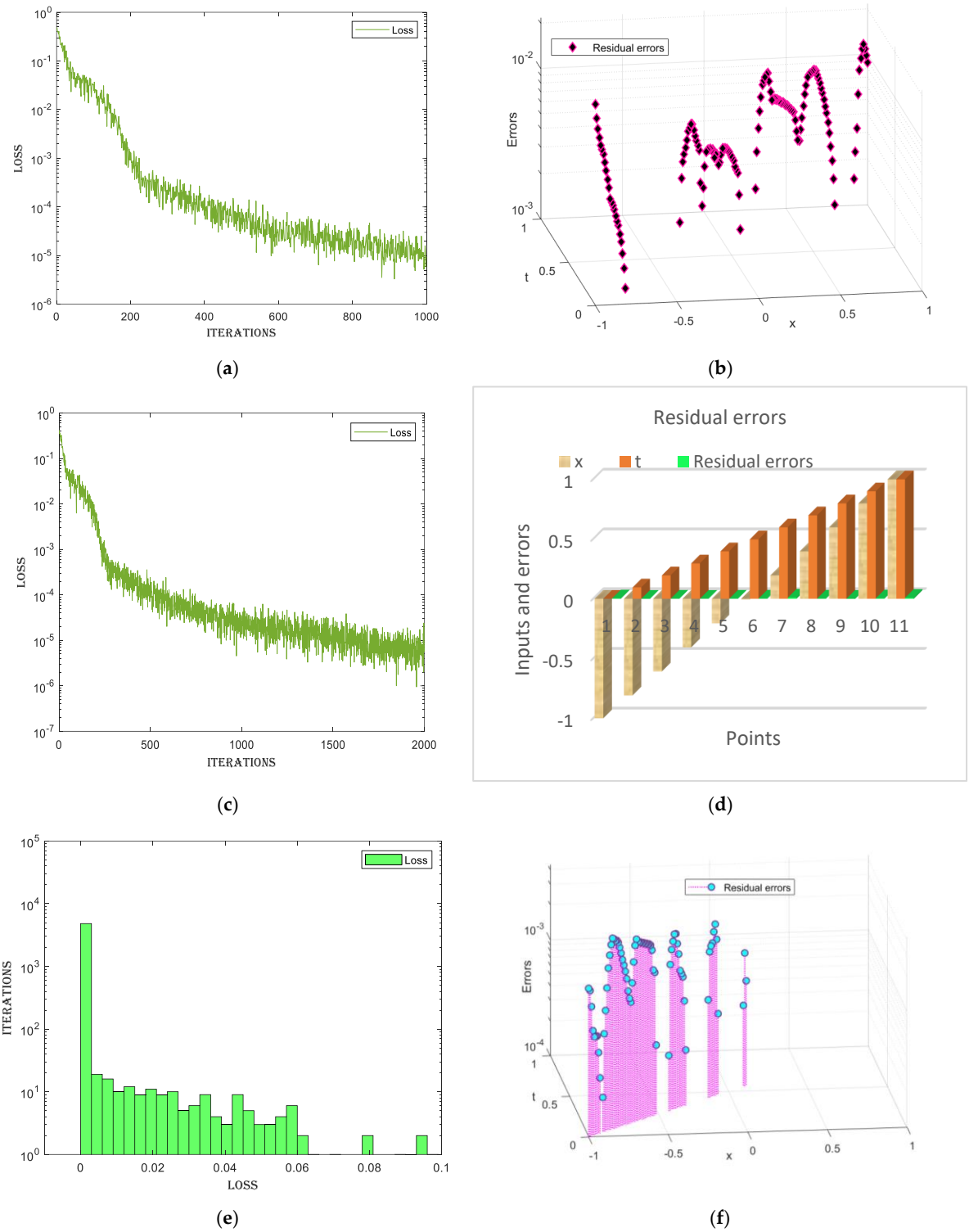


Figure 9. Graphical illustration of all the loss and residual error values for 200-point distribution of the domain when the value of the fractional order parameter $\alpha = 1.85$. (**a**) Graphical representation of loss values after 1000 iterations for the 200-point distribution when value of $\alpha = 1.85$. (**b**) The residual error graph shows an 200 error values after performing 1000 iterations when the value of $\alpha = 1.85$. (**c**) Graph of the loss values of 2000 iterations for 200 points when the value of $\alpha = 1.85$. (**d**) graphical illustration of all the 200 residual error values for the second scenario. (**e**) Error histogram graph of all the 5000 loss values when value of the fractional order $\alpha = 1.85$, where the domain is distributed to 200 different points. (**f**) Graphical representation of all the 200 residual error values of the second scenario.

The loss value graph is shown in Figure 9c. The graph shown in Figure 9c is for the loss values obtained after approximately 2000 iterations by taking 200 different points between the domains of x and t. The value of the fractional parameter α is 1.85. The iterations are represented on the x-axis, while the loss values are shown on the y-axis of Figure 9c. As the number of iterations increases, the loss value decreases and approaches zero, indicating that we are approaching the optimal solutions. At the beginning of the graph, the loss values are not close to zero, whereas at the end, the values are close to zero. Additionally, at the start of the graph, the loss values vary, but after 1000 iterations, the graph becomes smooth, showing that the solutions obtained through the proposed technique approach optimal solutions. The error values are shown in Figure 9d. These errors are obtained through 200 different points between the domains of x and t. 2000 iterations are performed to obtain these errors. We take 11 different points from the 200 points. Clearly, the errors are very close to 0, confirming the validity and consistency of the proposed technique.

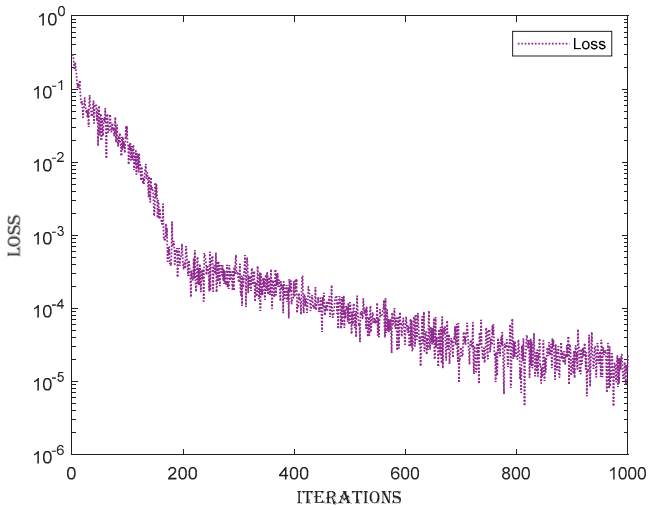
Figure 9e shows the histogram graph for the loss values. Figure 9e shows the loss values for 5000 iterations of the 200-point distribution of the domains of x and t, where the value of the fractional order parameter is $\alpha = 1.85$. The x-axis of the graph represents the loss values, and the y-axis of the graph represents the number of iterations. The graph shows that at approximately 3700 iterations, the loss values are the same and very close to 0. This finding indicates that our proposed technique is highly consistent and reliable. Figure 9f shows the graph of the residual error values. Figure 9f shows the errors obtained during the 5000 iterations for the 200-point distributions of the domains of x and t. Figure 9f shows the x, t and residual error values. All the error values lie between 10^{-3} and 10^{-4}. Figure 9f shows that the errors are very close to zero, indicating the accuracy and consistency of our proposed technique.

After performing 1000, 2000, and 5000 iterations, respectively. The residual error values are obtained and shown in Table 9. These values are obtained by taking 500 different points between the domains of x and t, while the value of the fractional parameter is $\alpha = 1.85$. Table 9 shows the residual errors obtained from the proposed technique. The first column of the table shows the input x values. These values are obtained for the distribution of the domains of x and t into 500 different points. The second column of the table shows the input t values. The third column of Table 9 represents the residual error values after 1000 iterations were performed. Similarly, the fourth and fifth columns show the residual error values after performing 2000 and 5000 iterations, respectively. Table 9 clearly shows that the residual errors are close to 0, which demonstrates that the proposed technique is state-of-the-art, accurate, and valid.

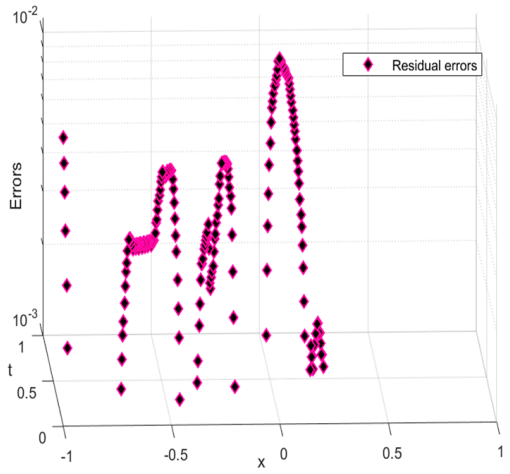
Table 9. Residual error values of the 500-point distribution when the value of the fractional order parameter is $\alpha = 1.85$.

x	t	Residual Error (1000 Iterations)	Residual Error (2000 Iterations)	Residual Error (5000 Iterations)
−1.00	0.00	9.00×10^{-3}	8.50×10^{-3}	7.00×10^{-3}
−0.80	0.10	-2.10×10^{-3}	1.29×10^{-2}	5.30×10^{-3}
−0.60	0.20	3.40×10^{-3}	9.50×10^{-3}	3.30×10^{-3}
−0.40	0.30	-2.60×10^{-3}	2.00×10^{-3}	4.80×10^{-3}
−0.20	0.40	5.30×10^{-3}	5.00×10^{-3}	5.00×10^{-4}
0.00	0.50	5.10×10^{-3}	-6.00×10^{-4}	1.80×10^{-3}
0.20	0.60	1.20×10^{-3}	4.20×10^{-3}	2.50×10^{-3}
0.40	0.70	-3.60×10^{-3}	1.10×10^{-3}	-4.20×10^{-3}
0.60	0.80	-5.70×10^{-3}	-7.70×10^{-3}	-2.50×10^{-3}
0.80	0.90	-2.80×10^{-3}	-4.50×10^{-3}	-5.80×10^{-3}
1.00	1.00	-1.95×10^{-2}	-7.90×10^{-3}	-5.30×10^{-3}

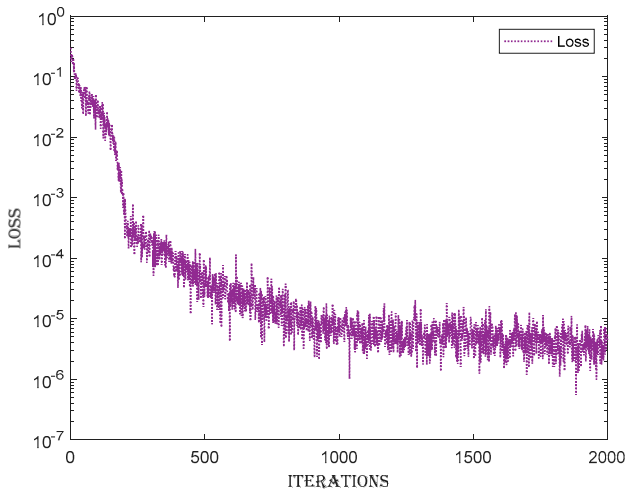
Figure 10a shows the graph for the loss values. Figure 10a shows the loss values for 1000 iterations of the 500-point distribution of the domains of x and t and the value of the fractional order parameter. $\alpha = 1.85$. The x-axis of the graph represents the number of iterations, and the y-axis represents the loss values. From Figure 10a, the loss value decreases and approaches zero as the number of iterations increases, which shows that we are approaching the optimal solutions. At the start of the iterations, the loss value is not close to zero, but after reaching the 1000th iteration, the loss value approaches 0. This means that as we approach the 1000th iteration, the solutions approach their best optimal values. Figure 10b shows the graph of the residual error values. Figure 10b shows the errors obtained during the 1000 iterations for the 500-point distributions of the domains of x and t. Figure 10b shows the x, t and error values. All 500 error values range from 10^{-2} to 10^{-3}. In Figure 6b, the errors are very close to zero, indicating the accuracy and consistency of our proposed technique.



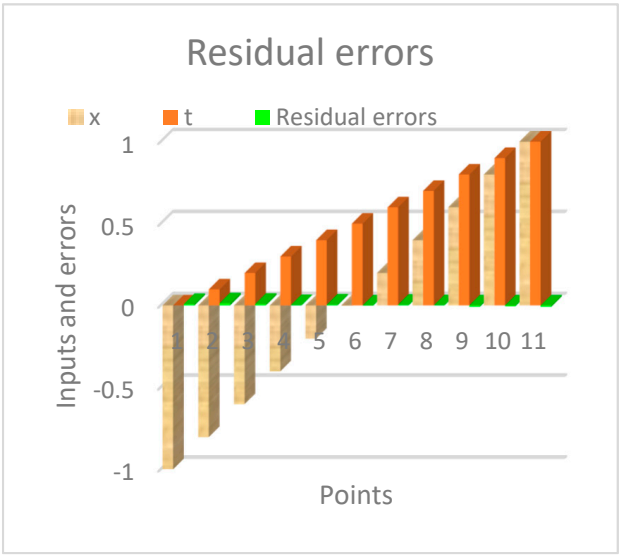
(**a**)



(**b**)

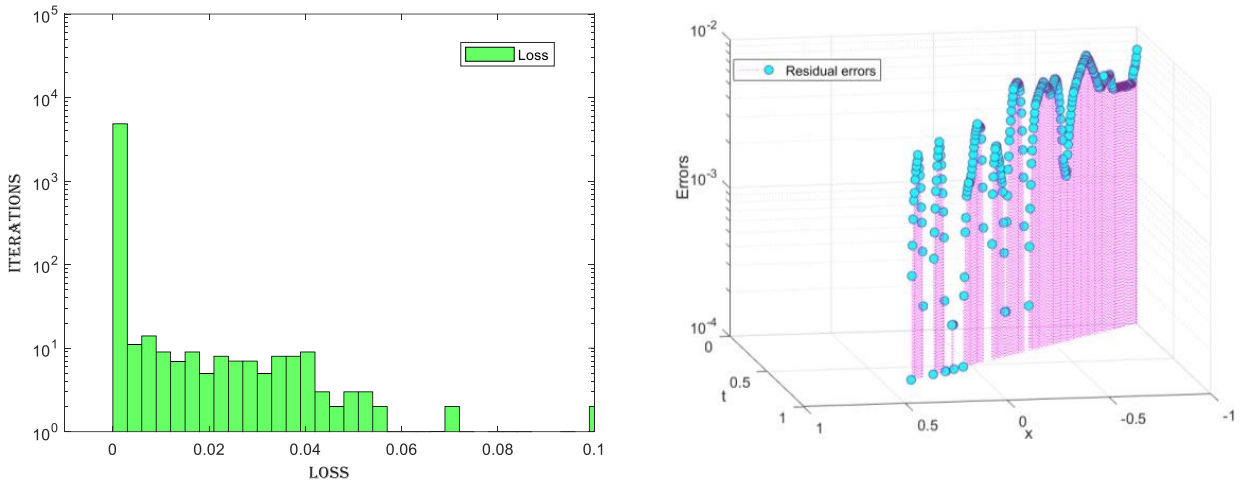


(**c**)



(**d**)

Figure 10. *Cont.*



(**e**) (**f**)

Figure 10. Graphs showing the loss and residual error values for the 500-point distribution of the domains of x and t after 1000, 2000, and 5000 iterations when the value of $\alpha = 1.85$. (**a**) Graphical representation of loss values after 1000 iterations for the 500-point distribution when value of $\alpha = 1.85$. (**b**) The residual error graph shows an 500 error values after performing 1000 iterations when $\alpha = 1.85$. (**c**) Graph of the loss values after performing 2000 iterations for 500 points distribution for the second scenario. (**d**) Residual error graph when the domain of inputs is distributed to 500 different points when the value of $\alpha = 1.85$. (**e**) Error histogram graph of all the 5000 loss values when value of the fractional order $\alpha = 1.85$, where the domain is distributed to 500 different points. (**f**) Graphical representation of all the 500 residual error values of the second scenario.

The graph of the loss values is shown in Figure 10c. Figure 10c shows the loss values for 2000 iterations of the 500-point distribution of the domains of x and t, where the value of the fractional order parameter is $\alpha = 1.85$. The x-axis of the graph represents the number of iterations, and the y-axis of the graph represents the loss values. Figure 10c shows that as the number of iterations increases, the loss value decreases and approaches zero. This indicates that the proposed technique converges to the optimal solutions. At the start of the iterations, the loss value is not close to zero, but after 1000 iterations, the loss value approaches zero. As we approach the 2000 iterations, the loss value approaches 0. This means that as we approach the 2000th iteration, the solutions approach their best optimal values. Figure 10d shows the graph of the error values. Figure 10d shows the errors obtained during the 5000 iterations for the 500-point distributions of the domains of x and t. Figure 10d shows the x, t and error values. We take 11 points to observe the errors. Figure 10d shows that the errors are very close to zero, indicating the accuracy and consistency of our proposed technique.

The graph shown in Figure 10e is for the loss values obtained after approximately 5000 iterations by taking 500 different points between the domains of x and t. The value of the fractional parameter α is 1.85. The iterations are represented through the y-axis, while the loss values are shown on the x-axis of Figure 6e. In approximately 3800 iterations, the loss values overlap each other. The histogram in Figure 10e shows the consistency of our proposed technique. Moreover, the loss values are very close to 0, which confirms the validity of the proposed technique. The residual error values are shown in Figure 10f. The graph shows the residual errors between the proposed and score-fPINN techniques. These errors are obtained through 500 different points between the domains of x and t. 5000 iterations are performed to obtain these residual errors. Furthermore, all 500 error values range between 10^{-2} and 10^{-4}. Clearly, the errors are very close to 0, confirming the validity and consistency of the proposed technique.

5. Conclusions

The conclusions of the analysis performed throughout the manuscript are given in this section. The following are some key concluding remarks.

- ➢ The fractional Fokker–Planck-Levy (FFPL) equation is solved in this manuscript. The equation contains the Levy noise and fractional Laplacian, making the equation computationally complex.
- ➢ The analytical solution is not possible. The proposed technique to solve the FFPL equation is a hybrid technique that involves the finite difference method (FDM), Adams numerical techniques, and physics-informed neural networks (PINNs).
- ➢ The FDM technique calculates the term fractional Laplacian, which is used in the equation.
- ➢ The PINN technique then approximates the solutions via the ADAMS optimizer and minimizes the overall loss function.
- ➢ The manuscript is categorized into two main scenarios by varying the value of the fractional order parameter α. The equation is solved for the two values of α, i.e., ($\alpha = 1.75, 1.85$).
- ➢ For each value of the fractional order parameter α, three cases are made by distributing the domain of x and t into 100, 200, and 500 points.
- ➢ The equation is solved via the proposed technique for each case. 1000, 2000, and 5000 iterations are performed for each case individually.
- ➢ The loss values given for each case can be visualized in the tables. The loss values are very minimal, ranging between 10^{-5} and 10^{-6}, indicating the precision of the proposed technique.
- ➢ All the solutions of the proposed technique are compared with those of the score-fPINN technique, which is a well-known technique for solving fractional differential equations.
- ➢ The residual error graph and tables show the errors between both techniques. The errors range between 10^{-2} and 10^{-4}. The small error indicates the validity of our proposed technique.
- ➢ Furthermore, loss and error graphs have been added to the manuscript to explore the proposed technique further. The histogram graph shows the consistency of the proposed technique.
- ➢ All the results presented in the tables and graphs indicate that the proposed technique is a state-of-the-art technique that is robust.

Author Contributions: F.U.F.: writing—original draft, software, methodology, investigation, formal analysis, data curation. D.B.: writing—original draft, investigation, formal analysis, conceptualization, supervision, writing—review and editing. M.S.: writing—original draft, investigation, formal analysis, conceptualization, supervision, writing—review and editing. F.S.A.: investigation, formal analysis, data curation, conceptualization, writing—review and editing. G.L.: investigation, formal analysis, conceptualization, visualization, writing—review and editing. All authors have read and agreed to the published version of the manuscript.

Funding: This study is supported via funding from Prince Sattam bin Abdulaziz University project number (PSAU/2024/R/1446).

Data Availability Statement: Data will be available upon publication of this manuscript at https://github.com/ffazal1/FDM_APINN.git (accessed on 9 September 2024).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Mainardi, F. *Fractional Calculus and Waves in Linear Viscoelasticity: An Introduction to Mathematical Models*; World Scientific: Singapore, 2022.
2. Cen, Z.; Huang, J.; Xu, A.; Le, A. Numerical approximation of a time-fractional Black–Scholes equation. *Comput. Math. Appl.* **2018**, *75*, 2874–2887. [CrossRef]
3. Nuugulu, S.M.; Gideon, F.; Patidar, K.C. A robust numerical scheme for a time-fractional Black-Scholes partial differential equation describing stock exchange dynamics. *Chaos Solitons Fractals* **2021**, *145*, 110753. [CrossRef]

4. Chen, Q.; Sabir, Z.; Raja, M.A.Z.; Gao, W.; Baskonus, H.M. A fractional study based on the economic and environmental mathematical model. *Alex. Eng. J.* **2023**, *65*, 761–770. [CrossRef]
5. Nikan, O.; Avazzadeh, Z.; Tenreiro Machado, J.A. Localized kernel-based meshless method for pricing financial options underlying fractal transmission system. *Math. Methods Appl. Sci.* **2024**, *47*, 3247–3260. [CrossRef]
6. Abdeljawad, T.; Thabet, S.T.; Kedim, I.; Vivas-Cortez, M. On a new structure of multiterm Hilfer fractional impulsive neutral Levin-Nohel integrodifferential system with variable time delay. *AIMS Math.* **2024**, *9*, 7372–7395. [CrossRef]
7. Rafeeq, A.S.; Thabet, S.T.; Mohammed, M.O.; Kedim, I.; Vivas-Cortez, M. On Caputo-Hadamard fractional pantograph problem of two different orders with Dirichlet boundary conditions. *Alex. Eng. J.* **2024**, *86*, 386–398. [CrossRef]
8. Boutiara, A.; Etemad, S.; Thabet, S.T.; Ntouyas, S.K.; Rezapour, S.; Tariboon, J. A mathematical theoretical study of a coupled fully hybrid (k, Φ)-fractional order system of BVPs in generalized Banach spaces. *Symmetry* **2023**, *15*, 1041. [CrossRef]
9. Raissi, M.; Perdikaris, P.; Karniadakis, G.E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **2019**, *378*, 686–707. [CrossRef]
10. Barron, A. Universal approximation bounds for superposition of a sigmoidal function. *IEEE Trans. Inf. Theory* **1991**, *19*, 930–944. [CrossRef]
11. Kawaguchi, K. On the theory of implicit deep learning: Global convergence with implicit layers. *arXiv* **2021**, arXiv:2102.07346.
12. Huang, W.; Jiang, T.; Zhang, X.; Khan, N.A.; Sulaiman, M. Analysis of Beam-Column Designs by Varying Axial Load with Internal Forces and Bending Rigidity Using a New Soft Computing Technique. *Complexity* **2021**, *19*, 6639032. [CrossRef]
13. Kharazmi, E.; Cai, M.; Zheng, X.; Zhang, Z.; Lin, G.; Karniadakis, G.E. Identifiability and predictability of integer-and fractional-order epidemiological models using physics-informed neural networks. *Nat. Comput. Sci.* **2021**, *1*, 744–753. [CrossRef] [PubMed]
14. Pang, G.; Lu, L.; Karniadakis, G.E. fPINNs: Fractional physics-informed neural networks. *SIAM J. Sci. Comput.* **2019**, *41*, A2603–A2626. [CrossRef]
15. Avcı, İ.; Lort, H.; Tatlıcıoğlu, B.E. Numerical investigation and deep learning approach for fractal–fractional order dynamics of Hopfield neural network model. *Chaos Solitons Fractals* **2023**, *177*, 114302. [CrossRef]
16. Avcı, İ.; Hussain, A.; Kanwal, T. Investigating the impact of memory effects on computer virus population dynamics: A fractal–fractional approach with numerical analysis. *Chaos Solitons Fractals* **2023**, *174*, 113845. [CrossRef]
17. Ul Rahman, J.; Makhdoom, F.; Ali, A.; Danish, S. Mathematical modelling and simulation of biophysics systems using neural network. *Int. J. Mod. Phys. B* **2024**, *38*, 2450066. [CrossRef]
18. Lou, Q.; Meng, X.; Karniadakis, G.E. Physics-informed neural networks for solving forward and inverse flow problems via the Boltzmann-BGK formulation. *J. Comput. Phys.* **2021**, *447*, 110676. [CrossRef]
19. Hu, Z.; Shi, Z.; Karniadakis, G.E.; Kawaguchi, K. Hutchinson trace estimation for high-dimensional and high-order physics-informed neural networks. *Comput. Methods Appl. Mech. Eng.* **2024**, *424*, 116883. [CrossRef]
20. Hu, Z.; Shukla, K.; Karniadakis, G.E.; Kawaguchi, K. Tackling the curse of dimensionality with physics-informed neural networks. *Neural Netw.* **2024**, *176*, 106369. [CrossRef]
21. Lim, K.L.; Dutta, R.; Rotaru, M. Physics informed neural network using finite difference method. In Proceedings of the 2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Prague, Czechia, 9–12 October 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 1828–1833.
22. Lim, K.L. Electrostatic Field Analysis Using Physics Informed Neural Net and Partial Differential Equation Solver Analysis. In Proceedings of the 2024 IEEE 21st Biennial Conference on Electromagnetic Field Computation (CEFC), Jeju, Korea, 2–5 June 2024; IEEE: Piscataway, NJ, USA, 2024; pp. 1–2.
23. Deng, W. Finite element method for the space and time fractional Fokker–Planck equation. *SIAM J. Numer. Anal.* **2009**, *47*, 204–226. [CrossRef]
24. Sepehrian, B.; Radpoor, M.K. Numerical solution of nonlinear Fokker–Planck equation using finite differences method and the cubic spline functions. *Appl. Math. Comput.* **2015**, *262*, 187–190.
25. Chen, X.; Yang, L.; Duan, J.; Karniadakis, G.E. Solving Inverse Stochastic Problems from Discrete Particle Observations Using the Fokker–Planck Equation and Physics-Informed Neural Networks. *SIAM J. Sci. Comput.* **2021**, *43*, B811–B830. [CrossRef]
26. Zhang, H.; Xu, Y.; Liu, Q.; Wang, X.; Li, Y. Solving Fokker–Planck equations using deep KD-tree with a small amount of data. *Nonlinear Dyn.* **2022**, *108*, 4029–4043. [CrossRef]
27. Zhai, J.; Dobson, M.; Li, Y. A deep learning method for solving Fokker-Planck equations. In *Mathematical and Scientific Machine Learning*; PMLR: London, UK, 2022; pp. 568–597.
28. Wang, T.; Hu, Z.; Kawaguchi, K.; Zhang, Z.; Karniadakis, G.E. Tensor neural networks for high-dimensional Fokker-Planck equations. *arXiv* **2024**, arXiv:2404.05615.
29. Lu, Y.; Maulik, R.; Gao, T.; Dietrich, F.; Kevrekidis, I.G.; Duan, J. Learning the temporal evolution of multivariate densities by normalizing flows. *Chaos Interdiscip. J. Nonlinear Sci.* **2022**, *32*, 033121. [CrossRef]
30. Feng, X.; Zeng, L.; Zhou, T. Solving time dependent Fokker-Planck equations via temporal normalizing flow. *arXiv* **2021**, arXiv:2112.14012. [CrossRef]
31. Guo, L.; Wu, H.; Zhou, T. Normalizing field flows: Solving forward and inverse stochastic differential equations using physics-informed flow models. *J. Comput. Phys.* **2022**, *461*, 111202. [CrossRef]
32. Tang, K.; Wan, X.; Liao, Q. Adaptive deep density approximation for Fokker-Planck equations. *J. Comput. Phys.* **2022**, *457*, 111080. [CrossRef]

33. Hu, Z.; Zhang, Z.; Karniadakis, G.E.; Kawaguchi, K. Score-based physics-informed neural networks for high-dimensional Fokker-Planck equations. *arXiv* **2024**, arXiv:2402.07465.
34. Evans, L.C. *An Introduction to Stochastic Differential Equations*; American Mathematical Soc.: Pawtucket, RI, USA, 2012; Volume 82.
35. Gardiner, C. *Stochastic Methods*, 4th ed.; Springer: Berlin/Heidelberg, Germany, 2009.
36. Oksendal, B. *Stochastic Differential Equations*, 6th ed.; Springer: Berlin/Heidelberg, Germany, 2003.
37. Boffi, N.M.; Vanden-Eijnden, E. Probability flow solution of the fokker–planck equation. *Mach. Learn. Sci. Technol.* **2023**, *4*, 035012. [CrossRef]
38. Zeng, S.; Zhang, Z.; Zou, Q. Adaptive deep neural networks methods for high-dimensional partial differential equations. *J. Comput. Phys.* **2022**, *463*, 111232. [CrossRef]
39. Hanna, J.M.; Aguado, J.V.; Comas-Cardona, S.; Askri, R.; Borzacchiello, D. Residual-based adaptivity for two-phase flow simulation in porous media using physics-informed neural networks. *Comput. Methods Appl. Mech. Eng.* **2022**, *396*, 115100. [CrossRef]
40. Wu, C.; Zhu, M.; Tan, Q.; Kartha, Y.; Lu, L. A comprehensive study of nonadaptive and residual-based adaptive sampling for physics-informed neural networks. *Comput. Methods Appl. Mech. Eng.* **2023**, *403*, 115671. [CrossRef]
41. Lu, L.; Meng, X.; Mao, Z.; Karniadakis, G.E. DeepXDE: A deep learning library for solving differential equations. *SIAM Rev.* **2021**, *63*, 208–228. [CrossRef]
42. Nabian, M.A.; Gladstone, R.J.; Meidani, H. Efficient training of physics-informed neural networks via importance sampling. *Comput.-Aided Civ. Infrastruct. Eng.* **2021**, *36*, 962–977. [CrossRef]
43. Jagtap, A.D.; Shin, Y.; Kawaguchi, K.; Karniadakis, G.E. Deep Kronecker neural networks: A general framework for neural networks with adaptive activation functions. *Neurocomputing* **2022**, *468*, 165–180. [CrossRef]
44. Jagtap, A.D.; Karniadakis, G.E. How important are activation functions in regression and classification? A survey, performance comparison, and future directions. *J. Mach. Learn. Model. Comput.* **2023**, *4*, 21–75. [CrossRef]
45. Jagtap, A.D.; Kawaguchi, K.; Em Karniadakis, G. Locally adaptive activation functions with slope recovery for deep and physics-informed neural networks. *Proc. R. Soc. A* **2020**, *476*, 20200334. [CrossRef]
46. Wang, S.; Teng, Y.; Perdikaris, P. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM J. Sci. Comput.* **2021**, *43*, A3055–A3081. [CrossRef]
47. Wang, S.; Yu, X.; Perdikaris, P. When and why PINNs fail to train: A neural tangent kernel perspective. *J. Comput. Phys.* **2022**, *449*, 110768. [CrossRef]
48. Xiang, Z.; Peng, W.; Liu, X.; Yao, W. Self-adaptive loss balanced physics-informed neural networks. *Neurocomputing* **2022**, *496*, 11–34. [CrossRef]
49. Haghighat, E.; Amini, D.; Juanes, R. Physics-informed neural network simulation of multiphase poroelasticity using stress-split sequential training. *Comput. Methods Appl. Mech. Eng.* **2022**, *397*, 115141. [CrossRef]
50. Amini, D.; Haghighat, E.; Juanes, R. Inverse modelling of nonisothermal multiphase poromechanics using physics-informed neural networks. *J. Comput. Phys.* **2023**, *490*, 112323. [CrossRef]
51. Jagtap, A.D.; Kharazmi, E.; Karniadakis, G.E. Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems. *Comput. Methods Appl. Mech. Eng.* **2020**, *365*, 113028. [CrossRef]
52. Meng, X.; Li, Z.; Zhang, D.; Karniadakis, G.E. PPINN: Parareal physics-informed neural network for time-dependent PDEs. *Comput. Methods Appl. Mech. Eng.* **2020**, *370*, 113250. [CrossRef]
53. Yang, L.; Meng, X.; Karniadakis, G.E. B-PINNs: Bayesian physics-informed neural networks for forward and inverse PDE problems with noisy data. *J. Comput. Phys.* **2021**, *425*, 109913. [CrossRef]
54. Hu, Z.; Jagtap, A.D.; Karniadakis, G.E.; Kawaguchi, K. When do extended physics-informed neural networks (XPINNs) improve generalization? *arXiv* **2021**, arXiv:2109.09444. [CrossRef]
55. Jagtap, A.D.; Karniadakis, G.E. Extended physics-informed neural networks (XPINNs): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. *Commun. Comput. Phys.* **2020**, *28*, 2002–2041. [CrossRef]
56. Yang, L.; Zhang, D.; Karniadakis, G.E. Physics-informed generative adversarial networks for stochastic differential equations. *SIAM J. Sci. Comput.* **2020**, *42*, A292–A317. [CrossRef]
57. Yu, J.; Lu, L.; Meng, X.; Karniadakis, G.E. Gradient-enhanced physics-informed neural networks for forward and inverse PDE problems. *Comput. Methods Appl. Mech. Eng.* **2022**, *393*, 114823. [CrossRef]
58. Eshkofti, K.; Hosseini, S.M. A gradient-enhanced physics-informed neural network (gPINN) scheme for the coupled nonfickian/non-Fourierian diffusion-thermoelasticity analysis: A novel gPINN structure. *Eng. Appl. Artif. Intell.* **2023**, *126*, 106908. [CrossRef]
59. Eshkofti, K.; Hosseini, S.M. The novel PINN/gPINN-based deep learning schemes for non-Fickian coupled diffusion-elastic wave propagation analysis. *Waves Random Complex Media* **2023**, *33*, 1–24. [CrossRef]